BAOU
Education for All

DR.BABASAHEB AMBEDKAR OPEN UNIVERSITY

DCA

DIPLOMA IN COMPUTER APPLICATION

DCAR-202
Database Management System (DBMS)

DATABASE MANAGEMENT SYSTEM

DR. BABASAHEB AMBEDKAR OPEN UNIVERSITY
AHMEDABAD

Editorial Panel

Authors : Hiral R. Patel
Assistant Professor
Department of Computer Science
Ganpat University.

Editor : Dr. Dharmesh Bhavsar
Associate Professor & I/C Director,
Shri Chimanbhai Patel Institute of
Computer Applications, Ahmedabad.

Language Editor : Dr. Jagdish Vinayakrao Anerao
Associate Professor,
Smt A. P. Patel Arts And,
N. P. Patel Commerce College,
Ahmedabad.

ISBN 978-81-949223-0-8

Edition : 2020

Acknowledgment

Every attempt has been made to trace the copyright holders of material reproduced in this book. Should an infringement have occurred, we apologize for the same and will be pleased to make necessary correction/amendment in future edition of this book.

ROLE OF SELF–INSTRUCTIONAL MATERIAL IN DISTANCE LEARNING

The need to plan effective instruction is imperative for a successful distance teaching repertoire. This is due to the fact that the instructional designer, the tutor, the author (s) and the student are often separated by distance and may never meet in person. This is an increasingly common scenario in distance education instruction. As much as possible, teaching by distance should stimulate the student's intellectual involvement and contain all the necessary learning instructional activities that are capable of guiding the student through the course objectives. Therefore, the course / self–instructional material is completely equipped with everything that the syllabus prescribes.

To ensure effective instruction, a number of instructional design ideas are used and these help students to acquire knowledge, intellectual skills, motor skills and necessary attitudinal changes. In this respect, students' assessment and course evaluation are incorporated in the text.

The nature of instructional activities used in distance education self–instructional materials depends on the domain of learning that they reinforce in the text, that is, the cognitive, psychomotor and affective. These are further interpreted in the acquisition of knowledge, intellectual skills and motor skills. Students may be encouraged to gain, apply and communicate (orally or in writing) the knowledge acquired. Intellectual–skills objectives may be met by designing instructions that make use of students' prior knowledge and experiences in the discourse as the foundation on which newly acquired knowledge is built.

The provision of exercises in the form of assignments, projects and tutorial feedback is necessary. Instructional activities that teach motor skills need to be graphically demonstrated and the correct practices provided during tutorials. Instructional activities for inculcating change in attitude and behaviour should create interest and demonstrate need and benefits gained by adopting the required change. Information on the adoption and procedures for practice of new attitudes may then be introduced.

Teaching and learning at a distance eliminate interactive communication cues, such as pauses, intonation and gestures, associated with the face–to–face method of teaching. This is

particularly so with the exclusive use of print media. Instructional activities built into the instructional repertoire provide this missing interaction between the student and the teacher. Therefore, the use of instructional activities to affect better distance teaching is not optional, but mandatory.

Our team of successful writers and authors has tried to reduce this.

Divide and to bring this Self–Instructional Material as the best teaching and communication tool. Instructional activities are varied in order to assess the different facets of the domains of learning.

Distance education teaching repertoire involves extensive use of self–instructional materials, be they print or otherwise. These materials are designed to achieve certain pre–determined learning outcomes, namely goals and objectives that are contained in an instructional plan. Since the teaching process is affected over a distance, there is need to ensure that students actively participate in their learning by performing specific tasks that help them to understand the relevant concepts. Therefore, a set of exercises is built into the teaching repertoire in order to link what students and tutors do in the framework of the course outline. These could be in the form of students' assignments, a research project or a science practical exercise. Examples of instructional activities in distance education are too numerous to list. Instructional activities, when used in this context, help to motivate students, guide and measure students' performance (continuous assessment)

PREFACE

We have put in lots of hard work to make this book as user–friendly as possible, but we have not sacrificed quality. Experts were involved in preparing the materials. However, concepts are explained in easy language for you. We have included many tables and examples for easy understanding.

We sincerely hope this book will help you in every way you expect.

All the best for your studies from our team!

DATABASE MANAGEMENT SYSTEM

Contents

BLOCK 1 : INTRODUCTION, DATA MODELS AND ER MODEL

Unit 1 INTRODUCTION TO DATABASE MANAGEMENT SYSTEM

Introduction, Definition of DBMS, What is Database ?, What is DBMS ?, Functions of a DBMS, Data Abstraction, Comparison of File Processing System and DBMS, Advantages and Disadvantages of DBMS, Users of DBMS, Capabilities of DBMS

Unit 2 DATA MODELS

Introduction, Types of Data Models, Object Base Logical Model, Record Base Logical Model, Physical Data Models, Relational, Network, Hierarchical Model

Unit 3 ENTITY RELATIONSHIP MODEL AND DIAGRAMS

Introduction, Entity Set, What is Entity Set ?, What is weak Entity Set ?, Attribute, Relationship Set, ER Diagrams

BLOCK 2 : RELATIONAL DATABASE AND DATABASE DESIGN

Unit 4 INTRODUCTION TO RELATIONAL DATABASE

Introduction, Codd's 12 Rules, Terms, Keys, Anomalies of Un–normalized Database, Comparison Hierarchical, Network and Relational Databases

Unit 5 DATABASE DESIGN

Introduction, Database Development Life Cycle, Logical Design, Physical Model, Capacity Planning, Advantages and Disadvantages of Normalization

Unit 6 NORMALISATION

Introduction, What is Normalization ?, Database Normal Forms and Example, 1NF (First Normal Form), 2NF (Second Normal Form), 3NF (Third Normal Form), BCNF (Boyce–Codd Normal Form), 4NF (Fourth Normal Form), 5NF & 6NF (Fifth & Sixth Normal Form)

BLOCK 3 : SQL AND OODBMS

Unit 7 SQL (STRUCTURED QUERY LANGUAGE)

Introduction, History, Basic Structure, DDL Commands, DML Commands, Simple Queries, Nested Queries, Aggregate Functions

Unit 8 SQL CONSTRAINTS

Introduction, Not Null Constraint, Default Constraint, Unique Constraint, Primary Key, Foreign Key, Check Constrait

Unit 9 TRANSACTION PROCESSING

Introduction, Types of Transactions, Concurrent Transactions, Discreet Transactions, Distributed Transactions, In–Doubt Transactions, Normal Transactions, Read–Only Transactions, Remote Transactions, Read–Consistency, Steps to Processing a Transaction, Entering DML/DDL Statements, Assigning Rollback Segments, Long–Running Transactions and Rollback Segment Allocation, Using the Optimizer, Cost–Based Analysis, Rule–Based Analysis, Overriding the Optimizer_Mode Parameter, Parsing Statements, Handling Locks, Stepping Through the Transaction, Processing a Remote or Distributed Transaction, Entering DDL/DML Statements, Assigning Rollback Segments, Breaking down Statements, Optimizing Local Statements, Forwarding Remote Commands, Assigning Remote Rollback Segments and Writing Redo Logs, Optimizing Remote Statement, Returning Data to the Local Database, Summarizing Remote and Distributed Transactions

Unit 10 OBJECT ORIENTED DATABASE MANAGEMENT SYSTEM

Introduction, Introduction to Database Management Systems (DBMS), Example of Bank Transactions, Object Oriented Database (OODB), Related terms, Distributed Object Computing (DOC), Objects Methods Users, Interfaces, Associations, Persistent Objects, Persistence Data, Transient Data, Referential Integrity, MDBS, ODBC (Open Database Connectivity), Locks, ActiveX, OOSAD, CORBA, DCOM, OMG, CORBA Open DOC ActiveX, Virtual DBMS, Object Oriented Database Management Systems (OODBMS), Comparison between RDBMS and OODBMS, A Three Schema Architecture, Mapping of OODBMS to RDBMS, Example of Railway Reservation System

BLOCK 4 : DATA (WARE HOUSING AND MINING) AND SECURITY

Unit 11 TYPES OF DATABASE

Introduction, Centralized Database, Distributed Database, Personal Database, End–User Database,

Commercial Database, NoSQL Database, Operational Database, Relational Database, Cloud Database, Object–Oriented Database, Graph Database

Unit 12 DATA WAREHOUSING AND DATA MINING

Introduction, Concept, Architecture, Various Tools in Data Warehousing, Tools in Data Mining, Difference Between Data Mining and Normal Query

Unit 13 DATABASE SECURITY

Introduction, Password Authentication, Operating System Authentication, Why Protect Passwords ?, Control, Protection, Integrity, Privileged Accounts, SYS, SYSTEM, Other Issues, Operating System Group : DBA, Object Security, Access Rights, Resolving Object Synonyms, System Security, Defined System Privileges, Object Security Model, Database Auditing, Recovery from Various Problems of Volatile and Non–Volatile Storage Devices

Unit 14 RECOVERY MECHANISMS

Introduction, Concept–Properties–States of Transaction, Introduction to Mechanisms, Log, Deferred Update, Immediate Update, Caching/Buffering, Checkpoint, Shadow Paging

Dr. Babasaheb Ambedkar
Open University Ahmedabad

BCAR-202/
DCAR-202

Database Management System

BLOCK 1 : INTRODUCTION, DATA MODELS AND ET MODEL

UNIT 1 INTRODUCTION TO DATABASE MANAGEMENT SYSTEM

UNIT 2 DATA MODELS

UNIT 3 ENTITY RELATIONSHIP MODEL AND DIAGRAMS

INTRODUCTION, DATA MODELS AND ET MODEL

Block Introduction :

An important aspect of most every business is record keeping. In our information society, this has become an important aspect of business, and much of the world's computing power is dedicated to maintaining and using databases.

Databases of all kinds pervade almost every business. All kinds of data, from emails and contact information to financial data and records of sales, are stored in some form of a database. The quest is on for meaningful storage of less–structured information, such as subject knowledge.

Unit 1 provides general overview of a nature and purpose of database systems. You will learn how database has developed, problems with file processing systems, common features of database system and users of database system.

Unit 2 provides need of data modelling to organize data elements and standardize how the data elements relate to one another. You will get overview of three basic models conceptual, physical and logical models.

Unit 3 provides basic concepts of high level conceptual data model i.e. the ER model. ER model views the real world as entities and relationships. You will learn a basic component of the model – the Entity – Relationship diagram which is used to visually represent data objects. It helps in database design and provides overall logical structure of database.

Block Objectives :

After learning this block, you will be able :

- To understand concept of database, Database management system and its capabilities
- To understand the need of data modeling
- To study physical and logical database design, concept of data models
- To study basic concepts of Entity Relational model

Block Structure :

Unit 1 : Introduction to Database Management System

Unit 2 : Data Models

Unit 3 : Entity Relationship Model and Diagrams

Unit 01

INTRODUCTION TO DATABASE MANAGEMENT SYSTEM

UNIT STRUCTURE

1.0 Learning Objectives

1.1 Introduction

1.2 Definition of DBMS

- **1.2.1 What is Database ?**
- **1.2.2 What is DBMS ?**
- **1.2.3 Functions of a DBMS**
- **1.2.4 Data Abstraction**

1.3 Comparison of File Processing System and DBMS

1.4 Advantages and Disadvantages of DBMS

1.5 Users of DBMS

1.6 Capabilities of DBMS

1.7 Let Us Sum Up

1.8 Suggested Answer for Check Your Progress

1.9 Glossary

1.10 Assignment

1.11 Activities

1.12 Case Study

1.13 Further Readings

1.0 Learning Objectives :

After learning this unit, you will be able to understand :

- Meaning and importance of DBMS
- Definition of DBMS
- Advantages and Disadvantages of DBMS
- Comparison of File Management and DBMS
- Users of DBMS
- Capabilities of DBMS

1.1 Introduction :

In the age of Information Technology, the significance of Data is very crucial. Database Management System (DBMS) is a separate branch of IT that comprises of several aspects regarding management of data.

Data can be defined as the collection of facts and figures.

The data can be generated in any type of transaction taking place in an organization. For example, amount withdrawn from a particular account on a particular day is the data which is generated in any bank. The basic unit

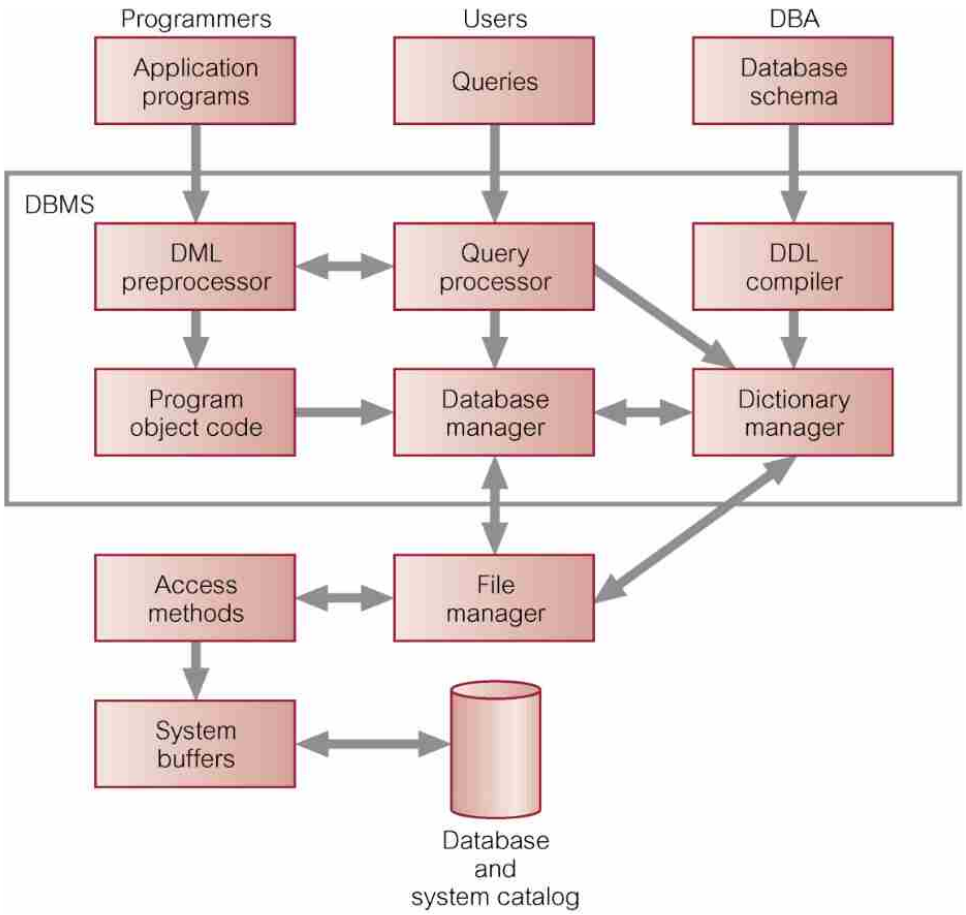
of data is byte. One byte comprises of eight bits. Bit refers to as binary digit i.e. zero and one. Thus the data comprises of several numbers of bits in form of zero's and ones. The other unit of data is Kilobyte. One kilobyte is equivalent to 1028 bytes (i.e. 2^{10} bytes). The other units are Megabyte, Gigabyte, and Terabyte and so on.

There is a difference between data and information. When we process data, it becomes information. For example, we have recorded daily financial transaction of a company. This forms a data. When we process this data for generating some report, say journal, profit and loss account or balance sheet, it becomes information.

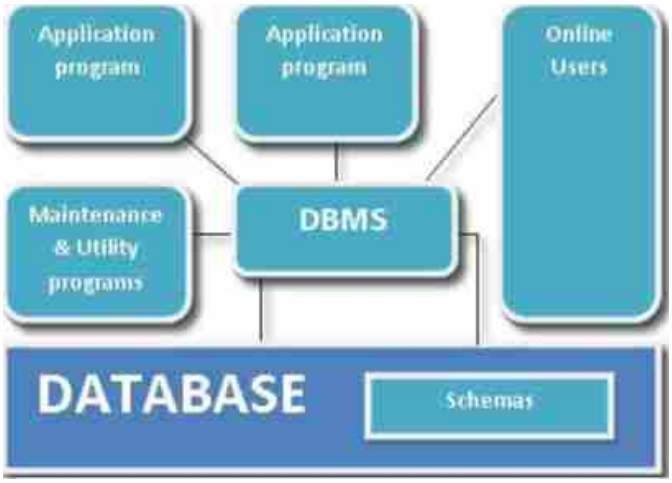
In short, information is the processed form of data which helps managers to make a decision. On these lines, let us compare data and information.

Data	Information
Data is a collection of facts and figures.	Information is the processed form of data.
It is difficult to draw any conclusion from data.	Information can be used to draw some conclusion.
It is difficult to use data for decision making.	Information can be used for the purpose of decision making.
Data is a raw material for processing. The data is required to be processed further.	Information can also be further utilized to produce highly densified information.
Data is not generally time bound.	Information is generally time bound

A Database management system (DBMS) is a collection of inter–related data and a set of programs/methods to access those data. The database refers to the data underlying the system. The goal of a DBMS is to provide a user–friendly environment, which is convenient. For example a telephone directory is an example of a database, where the underlying data consists of the name, address and phone number. However it is not exactly a DBMS since, there is no convenient way to get the address of the person who has a given phone number. Another example, which is a DBMS, is the one underlying http:// www.amazon.com. The database underlying the system, consists of its members and their information (name, login, password, credit card number, shipping address, billing address, when they made their last purchase...), information regarding the books (name, author, ISBN number, should an order be placed for it), information regarding transactions in progress. In this case the database is stored in amazon. Com's database server, the user's web browser initiates the transaction (causes amazon.com to run the program to modify the database) and the user interface is provided by the ser's web browser with the help of the web server and the database server.



1.2 Definition of DBMS :



Definition of DBMS

1.2.1 What is Database ?

A database is an orderly collection of information. Sales order, stock control or student records systems are all examples of databases

- A database categorizes information into logical groups, which are physically stored in files called tables
- A table is an orderly collection of records
- A record is a collection of fields

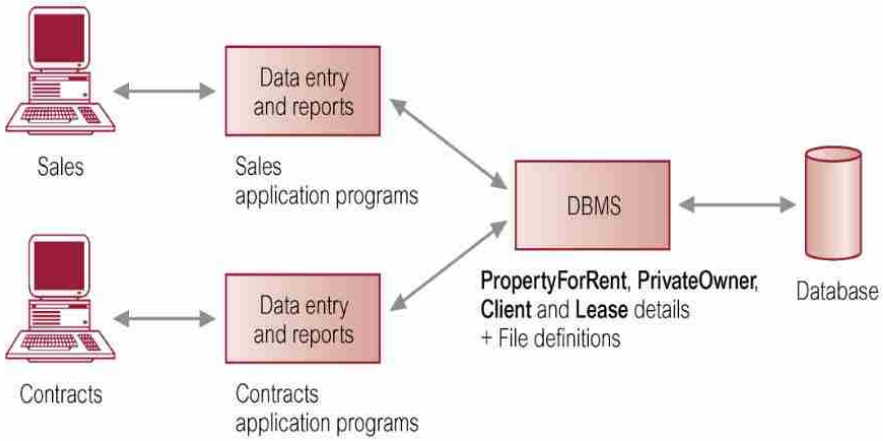
A database can be viewed as a storeroom of data. A collection of actual information regarding an organization is stored in a database. For example, there are 1000 students in a college and we want to store their personal details, marks details etc. This information can be recorded in a database.

A collection of programs that enables you to store, modify, and extract information from a database is known as DBMS. The major goal of a DBMS is to provide a way to store & retrieve database information in convenient and efficient manner.

Database systems are designed to handle large number of information. Management of data involves both defining structures for storage of information & providing way for manipulation of data. In addition, the database system must ensure safety and accuracy of data.

1.2.2 What is DBMS ?

As DBMS is set of programs that enable you to store, modify, and extract important information from a database. There are many different types of DBMS, ranging from small systems that run on personal computers to huge systems that run on mainframes.



Good data management is an essential prerequisite to corporate success.

Data → Information

Information → Knowledge

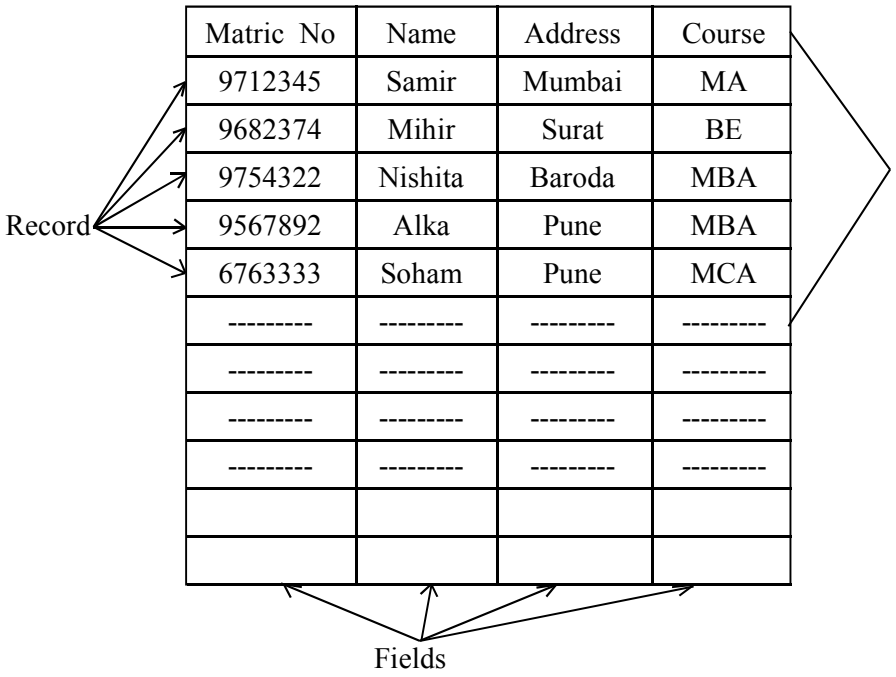
Knowledge → Judgment

Judgment → Decision

Decision → Success

A student record database holds details of students, courses and number

Table



The student record database contains tables for students, courses and staff. The student table has a record for each student, and each student record has fields Matriculation Number, Student Name, Address, Date–of–birth, Course etc.

One of the rules of relational databases is that only these "rectangular" tables are permitted.

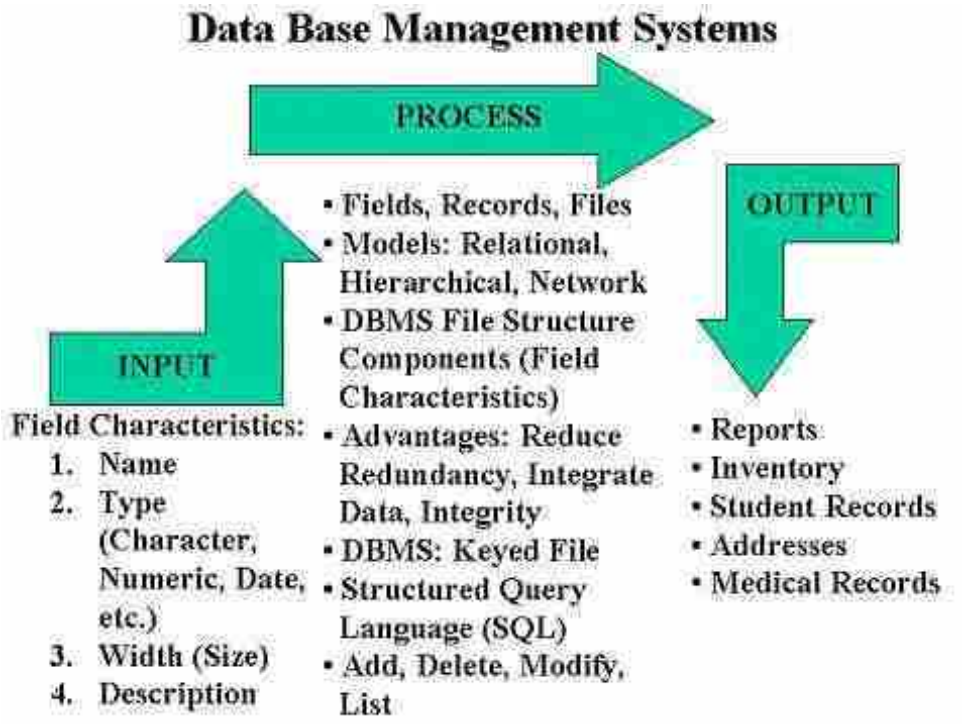
A database comprises 'tables', which contain 'records', which in turn contain 'fields'.

A Database Management System (DBMS) provides facilities for the storage and retrieval of information and, in addition, provides facilities for preservation of the 'integrity' of the data

There are a number of different types of database management systems, of which the 'relational' model, as used in Microsoft Access, is currently the most common.

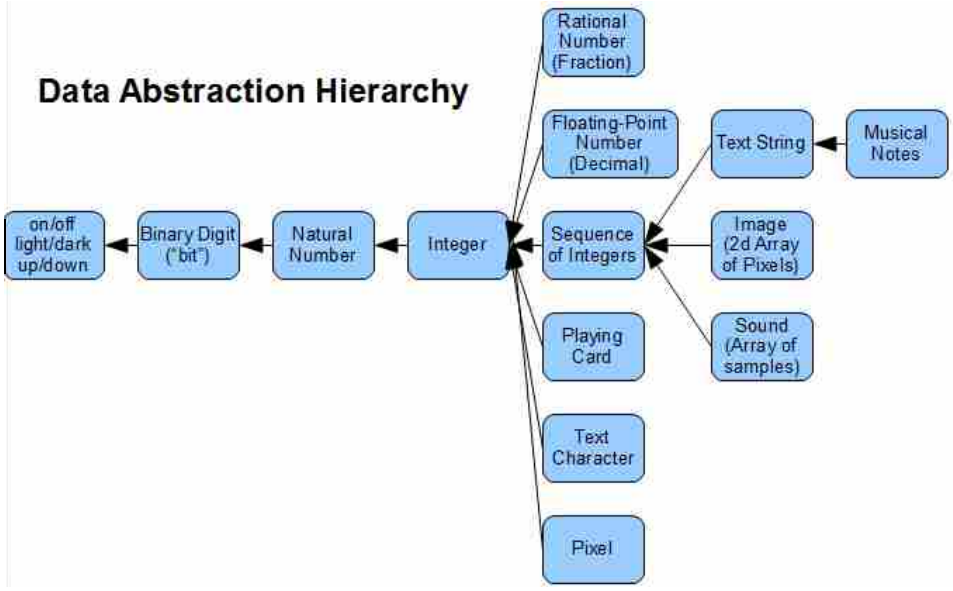
1.2.3 Functions of a DBMS :

1. Data Storage, Retrieval, and Update.
2. A User–Accessible Catalog.
3. Transaction Support.
4. Concurrency Control Services.
5. Recovery Services.
6. Authorization Services.
7. Support for Data Communication.
8. Integrity Services.
9. Services to Promote Data Independence.
10. Utility Services.



1.2.4 Data Abstraction :

The main objective of database management systems is to provide the users with an abstract view of data. The users of database may not be computer professionals. Hence the complexity of system is hidden from them through several levels of abstraction. Data abstraction is the process of hiding certain details of how data is stored and maintained in database.



Data Abstraction

❖ **Levels of Abstraction :**

Various levels of abstraction are as follows :

Physical Level : It is the lowest level of abstraction. It describes how the data is actually stored and describes the data structure and access methods to be used by the database.

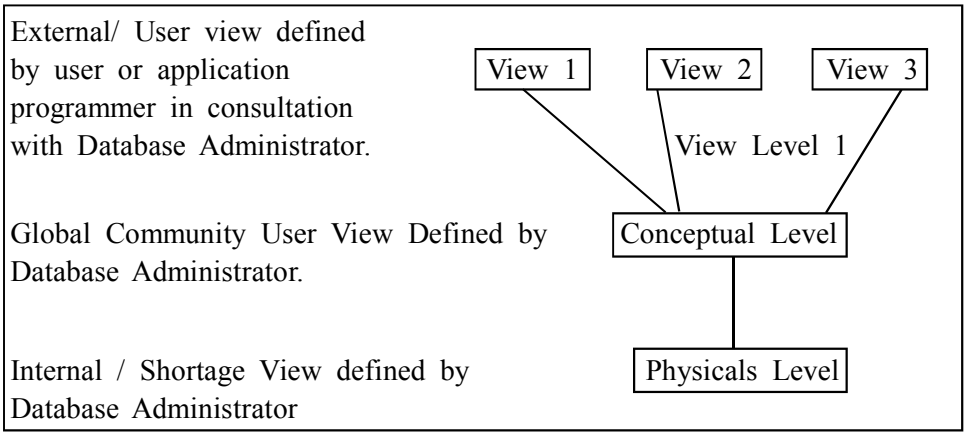
At the physical level, complex low level data is described in detail. The internal view is expressed by the internal schema which contains the definition

of stored record, the method of representing the data fields and access aids used.

Conceptual level or logical level : It is the next higher level of abstraction. It describes what data is stored actually in the database. It also explains what relationships exist among the data.

At the conceptual level, the entire database is described in terms of relatively simple data structures by conceptual schema.

View Level : It is the highest level of abstraction. At this level, only a part of the database is described because the users of database may not be concerned with the entire database. To simplify their interaction with database, the view level is defined. This system may provide many views with same database.



The various levels discussed above can be understood from this example-

```
Struct customer {
Char customer–name (50);
Char social–security (50);
Char customer–street (50);
Char customer–city (50);
} cust;
```

The structure of a customer can be defined using the above four fields. Each field has a name and data type associated with it.

Physical Level : This level is expressed as a block of memory location. The customer structure has 200 bytes block. This level is hidden from programmers.

Conceptual Level : At this level of abstraction, the structure is defined along with interrelationships with other structures. The database schema is defined.

View Level : At this level, the user can see the final result of the program for different inputs. Similarly, at the view level, several views of the database are defined and the database users can get the required output.

Instances and Schemes : Collection of information stored in database at a particular instance of time moment is called an instance of database. The overall design of the database is called database schema.

The database system supports three types of database schema –

Physical Schemas : These are the schema at the physical level. They are at the lowest level.

Logical Schemas : These are the schema at the conceptual level. These are provided at the next or intermediate level.

Sub–Schemas : These are the schema at view level. These are at the highest level.

❖ **Data Independence :**

The ability to modify schema definition at one level without affecting schema definition in the next higher level is known as data independence.

There are two levels of data independence – physical data independence and logical data independence. Physical data independence means the ability to modify physical schema without causing application programs to be rewritten.

On similar lines, logical data independence means the ability to modify the logical schema without causing application programs to be rewritten.

It is more difficult to achieve Logical data independence than achieving Physical data independence

Data Model : It is the collection of conceptual tools for describing Data, Data schema and Consistency constraints.

Data models are classified into following three categories –

(1) Object based logical model

(2) Record–based logical model

(3) Physical data model

❑ **Check Your Progress – 1 :**

1. What is database ? Explain with example ?

.....

.....

.....

.....

.....

2. What is DBMS ? Explain with example ?

.....

.....

.....

.....

.....

1.3 Comparison of File Processing System and DBMS :

Typical file–processing system is supported by conventional operating systems. The system stores permanent records in various files. So various application programs are needed to add, modify and extract information. There is a need to protect data from inconsistency due to multiple concurrent users. Crash recovery, security and access control should be taken into account. The

following drawbacks of typical file–processing systems are :

- Data redundancy and inconsistency
- Difficulty in accessing data
- Data isolation – multiple files and formats
- Integrity problems
- Atomicity of updates
- Concurrent access by multiple users
- Security problems

A database management system is software that provides services for accessing a database, while maintaining all the required features of data. The major components of a DBMS are the following :

- **Transaction management –** A transaction is a sequence of database operations that represents a logical unit of work and that accesses a database and transforms it from one state to another. Transaction management ensures that the database remains in a consistent (correct) state despite system failures (e.g., power failures and operating system crashes) and transaction failures.
- **Concurrency Control –** This is the database management activity of coordinating the actions of database manipulation processes that operate concurrently that access shared data and potentially interfere with one another. It ensures the consistency of the database in spite of interaction among the concurrent transactions.
- **Recovery Management –** This in a database ensures the aborted or failed transactions create no adverse effects on the database or the other transactions
- **Security Management –** Refers to the protection of data against un–authorized access.
- **Language Interface –** The DBMS provides support languages for definition and manipulation of data in the database.
- **Storage Management –** Storage manager is a program module that provides the interface between the low–level data stored in the database and the application programs and queries submitted to the system.

  The storage manager is responsible to the following tasks :

- Interaction with the file manager
- Efficient storing, retrieving and updating of

❑ **Check Your Progress – 2 :**

1. Write a note on Four significant differences between file system & DBMS

.....

.....

.....

.....

.....

1.4 Advantages and Disadvantages of DBMS :

- **Advantages of DBMS –** One of the main advantages of using a database system is that the organization can make use of, via the DBA, centralized management and control over the data. The database administrator is the focus of the centralized control. Any application requiring a change in the structure of a data record requires an arrangement with the DBA, who makes the necessary modifications.
- **Reduction of Redundancies –** Centralized control of data by the DBA avoids unnecessary duplication of data and effectively reduces the total amount of data storage required. It also eliminates the extra processing necessary to trace the required data in a large mass of data.
- **Elimination of Inconsistencies –** The main advantage of avoiding duplication is the elimination of inconsistencies that tend to be present in redundant data files. Any redundancies that exist in the DBMS are controlled and the system ensures that these multiple copies are consistent.
- **Shared Data –** A database allows the sharing of data under its control by any number of application programs or users. For example, the applications for the public relations and payroll departments can share the same data.
- **Integrity –** Centralized control can also ensure that adequate checks are incorporated in the DBMS to provide data integrity. Data integrity means that the data contained in the database is both accurate and consistent. Therefore, data values being entered for the storage could be checked to ensure that they fall within a specified range and are of the correct format.
- **Security –** Data is of vital importance to an organization and may be confidential. Such confidential data must not be accessed by unauthorized persons. The DBA who has the ultimate responsibility for the data in the DBMS can ensure that proper access procedures are followed, including proper authentication schemes for access to the DBMS and additional checks before permitting access to sensitive data. Different levels of security could be implemented for various types of data and operations.
- **Conflict Resolution –** Since the database is under the control of the DBA, he/she should resolve the conflicting requirements of various users and applications. In essence, the DBA chooses the best file structure and access method to get optimal performance for the response–critical applications, while permitting less critical applications to continue to use the database, albeit with a relatively slower response.
- **Data Independence –** It is advantageous in the database environment since it allows for changes at one level of the database without affecting other levels. These changes are absorbed by the mapping between the levels.

Disadvantages are as follows :

1. A complex conceptual design process
2. The need for multiple external databases
3. The need to hire database–related employees
4. High DBMS acquisition costs

5. A more complex programmer environment
6. Potentially catastrophic program failures
7. A longer running time for individual applications
8. Highly dependent DBMS operations

❑ **Check Your Progress – 3 :**

1. What are the Advantages of DBMS ?

.....

.....

.....

.....

.....

1.5 Users of DBMS :

Following people are interacting with the database Management System in the organization :

(1) Database Administrators – The Database Administrators have overall control on everything related to the database. They are responsible for administering the resources like database, the object in database, authentications to use data, coordinating and monitoring the activities, acquiring software and hardware resources.

(2) Database Designers – Database designers are responsible for designing database objects such as tables, columns, their data types, forms, reports, etc. these objects must be designed properly as per requirement so that those can be flexibly used by the users. The database designers have to frequently communicate with all respective database users, understand and study their requirements. They develop a view of database that supports the requirement. Once the database design is complete, they assist DBA i.e. Database Administrator.

(3) Application Programmers – These people write the program (Code) according to the suggested by the database designers, so as to meet the requirement of the end users. They also test, debus, document and maintain these programs.

(4) End Users – End Users are the ultimate users of the system. The end users access database for different purposes the end users can be categorizes as follows :

(a) Casual end users – they occasionally access database but may need different information every time the casual end users are typically high or middle level managers.

(b) Naïve or Parametric end users – these users keep querying constantly and updating database using common programmed queries. The examples of such end users are production supervisors, store keepers of the industry, reservation clerks of railway or airline.

(c) Sophisticated End Users – these people are business analysts, consultants, scientists. They generally need complex information.

(d) Standalone users – these users maintain personal database by using readymade program packages. These packages provide flexible environment with easy to use menu or graphical user interface.

❑ **Check Your Progress – 4 :**

1. Explain different users of database management system.

.....

.....

.....

.....

.....

1.6 Capabilities of DBMS :

- **Data dictionary –** stores data about the data (meta–data) In Access we can see a list of all the tables we have defined. We can view and edit the various fields.
- **Data storage, retrieval, updating –** In Access we can view and edit the data using the "datasheet" view. This looks rather like a spread sheet. Usually we will hide this facility in a released application – but it can be very useful during development.
- **Integrity provisions –** ensures that data remains consistent with itself we can restrict the types of data that can be entered. This may be as simple as ensuring that the date of birth is a real date. We can also prevent some more subtle errors creeping into our data.
- **Transaction integrity –** changes should be stored (committed) only when transaction is successful
- **We have to cope with the fact that computers go wrong –** we may suffer power failures or disk failures. Often it is important that operations do not fail part way through. Complete failure of a transaction can be recovered.
- **Concurrency control –** protects against simultaneous access to a single record by multiple users
- **Security mechanisms –** protect against accidental or deliberate misuse
- **Recovery mechanisms –** restoration of database where failure occurs
- **Data communications interface –** increasingly databases are 'distributed'

❑ **Check Your Progress – 5 :**

1. Explain functions of DBMS.

.....

.....

.....

.....

.....

2. Database is collections of _____.

(A) Modules (B) Data (C) Programs (D) None

3. _____ is collection of interrelated data and set of program to access them.

   (A) Data Structure (B) Programming language

   (C) Database Management System (D) Database

4. DBMS should provide following feature(s)_____.

   (A) Protect data from system crash

   (B) All of these

   (C) Safety of the information stored

   (D) Authorized access

5. Which of the following is considered as DBMS ?

   (A) Oracle (B) Foxpro (C) All of these (D) Access

6. _____ stores data about the data (meta–data).

   (A) Model (B) Data Dictionary

   (C) Template (D) None

1.7 Let Us Sum Up :

In this unit we have learned :

- Data can be defined as the collection of facts and figures. When we process data, it becomes information.
- Database is an orderly collection of information. It categorizes information into logical groups, which are physically stored in files called tables. a table is an orderly collection of records. a record is a collection of fields.
- A collection of programs that enables you to store, modify, and extract information from a database is known as DBMS. The major goal of a DBMS is to provide a way to store & retrieve database information in convenient and efficient manner.
- The main objective of database management systems is to provide the users with an abstract view of data. Data abstraction is the process of hiding certain details of how data is stored and maintained in database. Various levels of abstraction are physical, conceptual, view level.
- Data Independence is the ability to modify schema definition at one level without affecting schema definition in the next higher level is known as data independence. There are two levels of data independence – physical data independence and logical data independence.
- Typical file–processing system is supported by conventional operating system has drawbacks like Data redundancy and inconsistency, Difficulty in accessing data, Data isolation – multiple files and formats, Integrity problems, Atomicity of updates, Concurrent access by multiple users, Security problems
- Components of DBMS are Transaction management, Concurrency Control, Recovery Management, Security Management and Storage Management
- The DBMS users are Database Administrators, Database Designers, Application Programmers and End Users like Casual end users, Naïve or Parametric, Sophisticated and Standalone users.

1.8 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 1.2

- **Check Your Progress 2 :**

  See Section 1.3

- **Check Your Progress 3 :**

  See Section 1.4

- **Check Your Progress 4 :**

  See Section 1.5

- **Check Your Progress 5 :**

  1 : See Section 1.6

  2 : Data

  3 : Database Management System

  4 : All of These

  5 : All of These

  6 : Data Dictionary

1.9 Glossary :

1. **Data** – It can be defined as the collection of facts and figures.
2. **Information** – When we process data, it becomes information.
3. **Database** – A database is an orderly collection of information.
4. **Database Management System (DBMS)** – It is a collection of inter–related data and a set of programs/methods to access those data.
5. **Data Abstraction** – It is the process of hiding certain details of how data is stored and maintained.
6. **Instances and Schemas** – Collection of information stored in database at a particular instance of time moment is called an instance of database. The overall design of the database is called database schema.
7. **Data Independence** – The ability to modify schema definition at one level without affecting schema definition in the next higher level is known as data independence.
8. **Physical Data Independence** – It means the ability to modify physical schema without causing application programs to be rewritten.
9. **Logical Data Independence** – It means the ability to modify the logical schema without causing application programs to be rewritten.
10. **Integrity** – The property of the database that ensures that the data contained in the database is as accurate and consistent as possible.
11. **Data Dictionary** – The database stores metadata in an area called the data dictionary, which describes the tables, fields, indexes, constraints, and other related items that make up the database.
12. **Concurrent Access** – Two or more users operating on the same records in the same database table at the same time.

1.10 Assignment :

Design a file structure of student with consists of student name, student roll no., student address, student course and student phone no. and save it, so that further it will be used to convert it in to database

1.11 Activities :

List major steps that you would take in setting up a database for particular organization.

1.12 Case Study :

List five responsibilities of a database management system. For each responsibility, explains the problems that would arise if the responsibility were not dis–charged.

1.13 Further Reading :

1. Introduction to Database Systems by C. J. Date
2. Database Management Systems – Rajesh Narang –PHI Learning Pvt Ltd
3. Database System Concepts by Silberschatz,Korth –Tata McGraw–Hill Publication
4. An Introduction to Database Systems – Bipin Desai– Galgotia Publication
5. Database Management System by Raghu Ramkrishnan– Tata McGraw–Hill Publication
6. SQL, PL/SQL:The Programming Language Oracle – Ivan Bayross– BPB Publication

Some useful websites are as follows :

http://www.eiilmuniversity.ac.in/coursepack/Computing/Database_Management_ System.pdf

http://www.slideshare.net/singhanshu21/unit3rd

http://placementjump.blogspot.in/2013/10/dbms–important–questions–for–interview.html

http://www.slideshare.net/singhanshu21/unit3rd http://www.orafaq.com/wiki/Talk:DBMS http://www.authorstream.com/Presentation/nehasinghal–761246–dbms

http://placementjump.blogspot.in/2013/10/dbms–important–questions–for–interview.html

http://www.authorstream.com/Presentation/nehasinghal–761246–dbms

http://arts.nprcolleges.org/e%20content/swca/Database%20System%20Technique s%20–%20SST8C51.pdf

Unit
02

DATA MODELS

UNIT STRUCTURE

2.0 Learning Objectives

2.1 Introduction

2.2 Types of Data Models

- **2.2.1 Object Base Logical Model**
- **2.2.2 Record Base Logical Model**
- **2.2.3 Physical Data Models**
- **2.2.4 Relational**
- **2.2.5 Network**
- **2.2.6 Hierarchical Model**

2.3 Let Us Sum Up

2.4 Suggested Answer for Check Your Progress

2.5 Glossary

2.6 Assignment

2.7 Activities

2.8 Case Study

2.9 Further Readings

2.0 Learning Objectives :

After learning this unit, you will be able to understand :

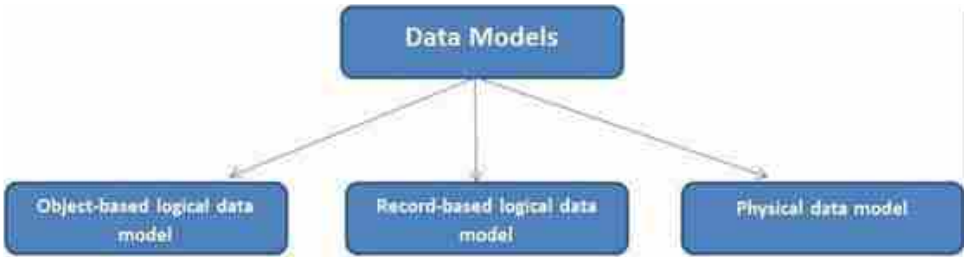
- Concept of Data Models
- Importance in System Analysis and design
- Types of Data Models

2.1 Introduction :

In the earlier unit, we had a discussion on the fundamental concepts of Database and Database Management Systems. One of the main purposes of Database Management System is to provide some level of data abstraction by hiding details of data storage that are not required by most of the database users. Now, let us consider how the structure of data, the relationship between the data, its semantics and consistency constraints can be conceptually described. These details are provided by the tool called Data Model.

Data model is integrated collection of concepts for describing data, relationships between data, and constraints on the data in an organization.

2.2 Types of Data Models :



Types of Data Models

A DBMS can take any one of the several approaches to manage data. Each approach constitutes a database model. A data model is a collection of descriptions of data structures and their contained fields, together with the operations or functions that manipulate them. A data model is a comprehensive scheme for describing how data is to be represented for manipulation by humans or computer programs. A thorough representation details the types of data, the topological arrangements of data, spatial and temporal maps onto which data can be projected, and the operations and structures that can be invoked to handle data and its maps. The various Data Models fall into three groups : (1) Object Based Logical Model, (2) Record Based Logical Model and (3) physical model.

2.2.1 Object Based Logical Model :

Object Based Logical Models describe data at the conceptual and view levels. These models provide fairly flexible structuring capabilities and allow one to specify data constraints explicitly. There are many different models under this:

- Entity–relationship model.
- Object–oriented model.
- Semantic data model.
- Functional data model

The Entity–Relationship model has emerged as one of the main techniques for modelling database design and forms the basis for the database design methodology,

2.2.2 Record Base Logical Model :

Record based logical models are used in describing data at the logical and view levels. In contrast to object based data models, they are used to specify the overall logical structure of the database and to provide a higher–level description of the implementation. Record based models are so named because the database is structured in fixed format records of several types. Each record type defines a fixed number of fields, or attributes, and each field is usually of a fixed length.

The three most widely accepted record based data models are:

- Hierarchical Model
- Network Model
- Relational Model

The relational model has gained popularity over the other two in recent years. The network and hierarchical models are still used in a large number of older databases.

2.2.3 Physical Data Models :

Physical data models describe how data is stored in the computer, representing information such as record structures, record ordering, and access paths. Thus these models describe data at the lowest level. There are not as many physical data models as logical data models, the most common one are:

- The Unifying model.
- Frame memory model

2.2.4 Relational :

Relational model stores data in the form of tables. This concept purposed by Dr. E. F. Codd, a researcher of IBM in the year 1960s. The relational model consists of three major components:

1. The set of relations and set of domains that defines the way data can be represented (data structure).
2. Integrity rules that define the procedure to protect the data (data integrity).
3. The operations that can be performed on data (data manipulation).

A rational model database is defined as a database that allows you to group its data items into one or more independent tables that can be related to one another by using fields common to each related table.

Properties of Relational Tables:

a. Values Are Atomic

b. Each Row is Unique

c. Column Values Are of the Same Kind

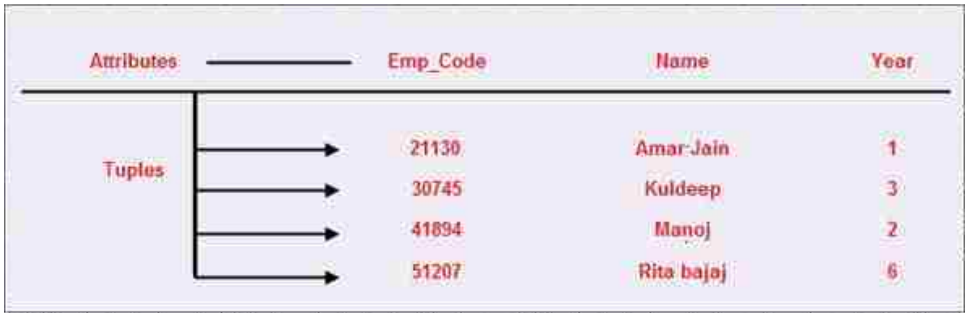
d. The Sequence of Columns is Insignificant

e. The Sequence of Rows is Insignificant

f. Each Column Has a Unique Name

Basic Terminology used in Relational Model

The figure shows a relation with the formal names of the basic components marked the entire structure is, as we have said, a relation.



- **Tuples of a Relation –** Each row of data is a tuple. Actually, each row is an n–tuple, but the "n–" is usually dropped.
- **Cardinality of a relation –** The number of tuples in a relation determines its cardinality. In this case, the relation has a cardinality of 4.
- **Degree of a relation –** Each column in the tuple is called an attribute. The number of attributes in a relation determines its degree. The relation in figure has a degree of 3.

- **Domains –** A domain definition specifies the kind of data represented by the attribute. Particularly, a domain is the set of all possible values that an attribute may validly contain. Domains are often confused with data types. Data type is a physical concept while domain is a logical one. "Number" is a data type and "Age" is a domain.

  To give another example "Street Name" and "Surname" might both be represented as text fields, but they are obviously different kinds of text fields; they belong to different domains.

- **Body of a Relation –** The body of the relation consists of an unordered set of zero or more tuples. There are some important facts :
  - First the relation is unordered. Record numbers do not apply to relations.
  - Second a relation with no tuples still qualifies as a relation.
  - Third, a relation is a set. The items in a set are, by definition, uniquely identifiable. Therefore, for a table to qualify as a relation each record must be uniquely identifiable and the table must contain no duplicate records.
  - In this model, two sets of data are linked by a relationship. The relationship could be 'one–to–one', 'one–to–many' or 'many–to–many'.

❖ **Relational View of Sample Database :**

- Let us take an example of a sample database consisting of supplier, parts and shipments tables. The table structure and some sample records for supplier, parts and shipments tables are given as Tables as shown below :

The Supplier Records

Sno	Name	Status	Status
S1	Suneet	20	Qadian
S2	Ankit	10	Amritsar
S3	Amit	10	Qadian

The Part Records

Pno	Name	Status	Weight	City
P1	Nut	Red	12	Qadian
P2	Bolt	Green	17	Amritsar
P3	Screw	Blue	17	Jalandhar
P4	Screw	Red	14	Qadian

The Shipment Records

Sno	Name	Qty
S1	P1	250
S1	P2	300
S1	P3	500
S2	P1	250
S2	P2	500
S3	P2	300

- Each row in Supplier table is identified by a unique SNo (Supplier Number), which uniquely identifies the entire row of the table. Likewise each part has a unique PNo (Part Number). Also, we assume that no

more than one shipment exists for a given supplier/part combination_in the shipments table.

- Note that the relations Parts and Shipments have PNo (Part Number) in common and Supplier and Shipments relations have SNo (Supplier Number) in common. The Supplier and Parts relations have City in common. For example, the fact that supplier S3 and part P2 are located in the same city is represented by the appearance of the same value, Amritsar, in the city column of the two tuples in relations.

❖ **Keys of a Relation :**

It is a set of one or more columns whose combined values are unique among all occurrences in a given table. A key is the relational means of specifying uniqueness. Some different types of keys are :

- **Primary key –** is an attribute or a set of attributes of a relation which possess the properties of uniqueness and irreducibility (No subset should be unique). For example: Supplier number in S table is primary key, Part number in P table is primary key and the combination of Supplier number and Part Number in SP table is a primary key
- **Foreign key –** is the attributes of a table, which refers to the primary key of some another table. Foreign key permit only those values, which appears in the primary key of the table to which it refers or may be null (Unknown value). For example: SNO in SP table refers the SNO of S table, which is the primary key of S table, so we can say that SNO in SP table is the foreign key. PNO in SP table refers the PNO of P table, which is the primary key of P table, so we can say that PNO in SP table is the foreign key.

❖ **Operations in Relational Model :**

The four basic operations Insert, Update, Delete and Retrieve operations are shown below on the sample database in relational model:

- **Insert Operation –** Suppose we wish to insert the information of supplier who does not supply any part, can be inserted in S table without any anomaly e.g. S4 can be inserted in Stable. Similarly, if we wish to insert information of a new part that is not supplied by any supplier can be inserted into a P table. If a supplier starts supplying any new part, then this information can be stored in shipment table SP with the supplier number, part number and supplied quantity. So, we can say that insert operations can be performed in all the cases without any anomaly.
- **Update Operation –** Suppose supplier S1 has moved from Qadian to Jalandhar. In that case we need to make changes in the record, so that the supplier table is up–to–date. Since supplier number is the primary key in the S (supplier) table, so there is only a single entry of S 1, which needs a single update and problem of data inconsistencies would not arise. Similarly, part and shipment information can be updated by a single modification in the tables P and SP respectively without the problem of inconsistency. Update operation in relational model is very simple and without any anomaly in case of relational model.
- **Delete Operation –** Suppose if supplier S3 stops the supply of part P2, then we have to delete the shipment connecting part P2 and supplier S3 from shipment table SP. This information can be deleted from SP

table without affecting the details of supplier of S3 in supplier table and part P2 information in part table. Similarly, we can delete the information of parts in P table and their shipments in SP table and we can delete the information suppliers in S table and their shipments in SP table.

The Relational database model is based on the Relational Algebra.

Advantages :

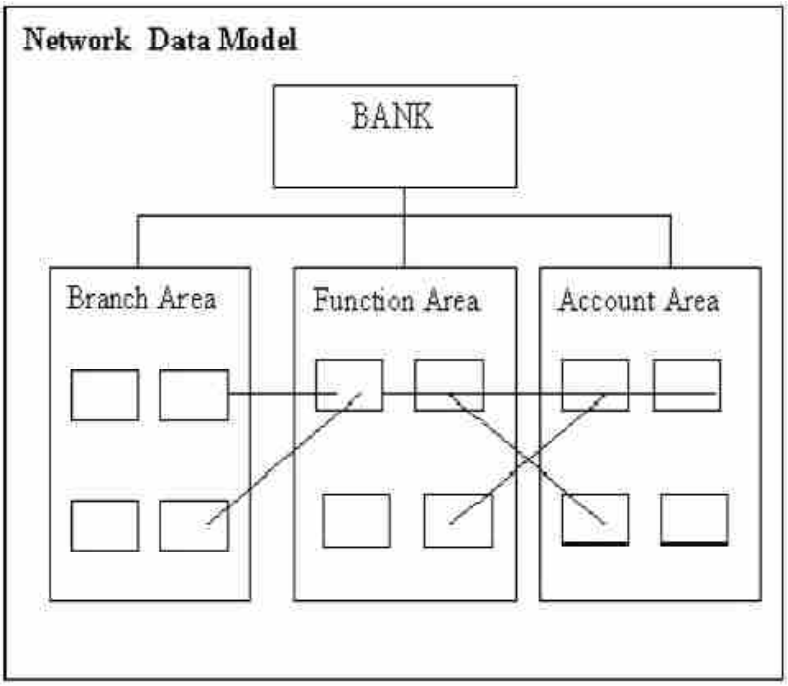
1. Structural Independence
2. Conceptual Simplicity
3. Ease of design, implementation, maintenance and usage.
4. Ad hoc query capability

Disadvantages :

1. Hardware Overheads
2. Ease of design can lead to bad design

2.2.5 Network :

The popularity of the network data model coincided with the popularity of the hierarchical data model. Some data were more naturally modeled with more than one parent per child. So, the network model permitted the modeling of many–to–many relationships in data. In 1971, the Conference on Data Systems Languages (CODASYL) formally defined the network model. The basic data modeling construct in the network model is the set construct. A set consists of an owner record type, a set name, and a member record type.



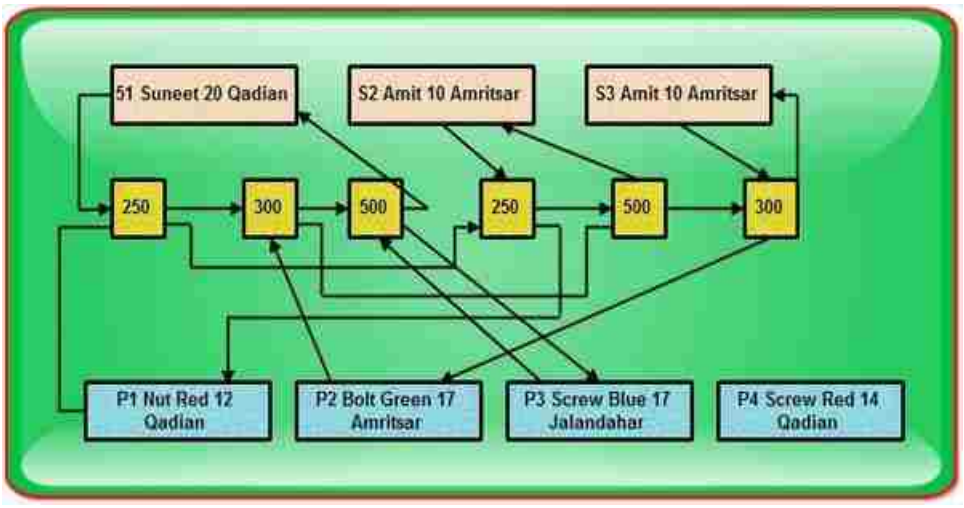
A member record type in the Network Model can have that role in more than one set; hence the multi parent concept is supported. An owner record type can also be a member or owner in another set. The data model is a simple network, and link and intersection record types (called junction records by IDMS) may exist, as well as sets between them.

Thus, the complete network of relationships is represented by several pair wise sets; in each set some (one) record type is owner (at the tail of the network arrow) and one or more record types are members (at the head

of the relationship arrow). Usually, a set defines a 1 : M relationship, although 1 : 1 is permitted. The CODASYL network model is based on mathematical set theory.

❖ **Network view of Sample Database :**

Considering again the sample supplier–part database, its network view is shown. In addition to the part and supplier record types, a third record type is introduced which we will call as the connector. A connector occurrence specifies the association (shipment) between one supplier and one part. It contains data (quantity of the parts supplied) describing the association between supplier and part records.



All connector occurrences for a given supplier is placed on a chain.The chain starts from a supplier and finally returns to the supplier. Similarly, all connector occurrences for a given part are placed on a chain starting from the part and finally returning to the same part.

❖ **Operations on Network Model :**

Detailed description of all basic operations in Network Model is as under :

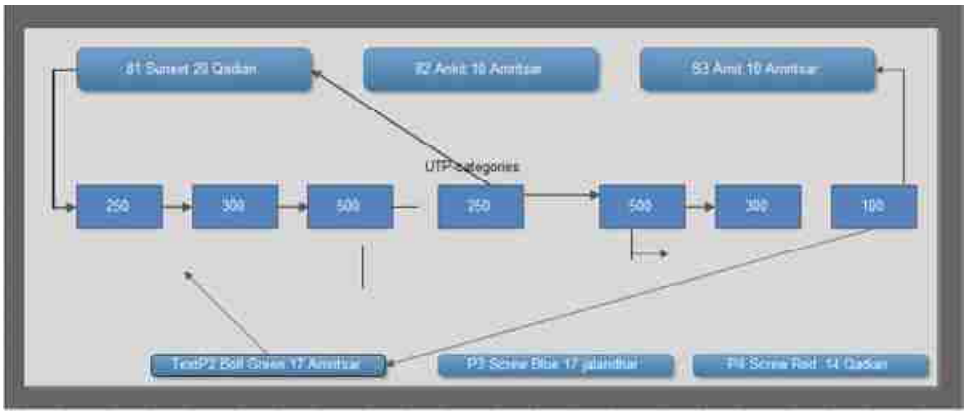
- **Insert Operation –** To insert a new record containing the details of a new supplier, we simply create a new record occurrence. Initially, there will be no connector. The new supplier's chain will simply consist of a single pointer starting from the supplier to it.

  For example, supplier S4 can be inserted in network model that does not supply any part as a new record occurrence with a single pointer from S4 to itself. This is not possible in case of hierarchical model. Similarly a new part can be inserted who does not supplied by any supplier.

  Consider another case if supplier S 1 now starts supplying P3 part with quantity 100, then a new connector containing the 100 as supplied quantity is added in to the model and the pointer of S1 and P3 are modified as shown in the below. We can summarize that there is no insert anomalies in network model as in hierarchical model.

- **Update Operation –** Unlike hierarchical model, where updating was carried out by search and had many inconsistency problems, in a network model updating a record is a much easier process. We can change the city of S I from Qadian to Jalandhar without search or inconsistency problems because the city for S1 appears at just one place in the network

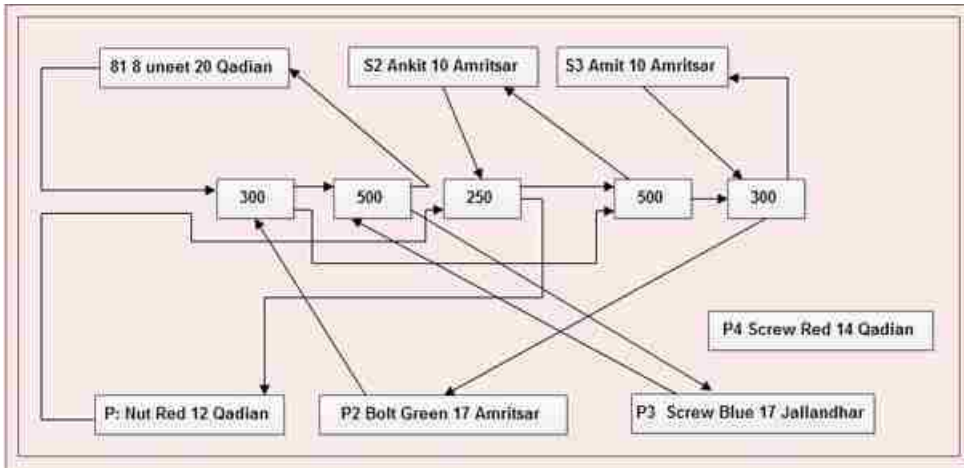
model. Similarly, same operation is performed to change the any attribute of part.



- **Delete operation –** If we wish to delete the information of any part say PI, then that record occurrence can be deleted by removing the corresponding pointers and connectors, without affecting the supplier who supplies that part i.e. P1, the model is modified as shown. Similarly, same operation is performed to delete the information of supplier.

  In order to delete the shipment information, the connector for that shipment and its corresponding pointers are removed without affecting supplier and part information.

  For example, if supplier SI stops the supply of part PI with 250 quantity the model is modified as shown below without affecting P1 and S1 information.

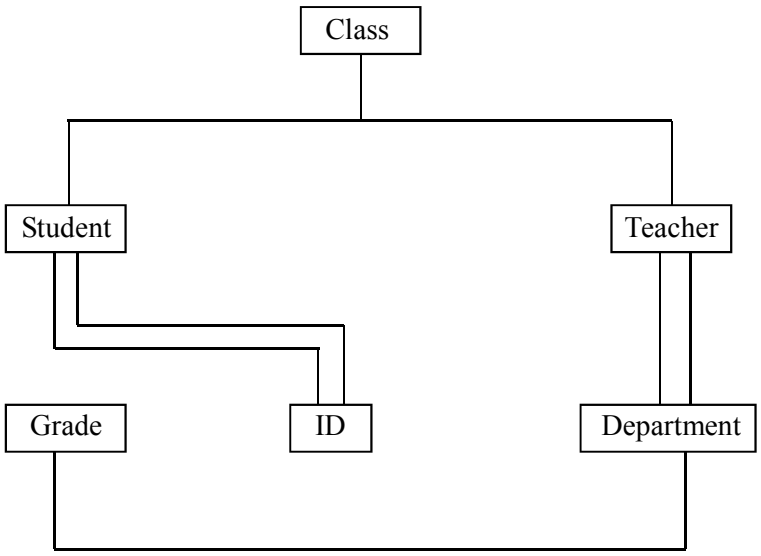


Advantages :

1. Conceptual Simplicity
2. Ease of data access
3. Data Integrity and capability to handle more relationship types
4. Data independence
5. Database standards

Disadvantages :

1. System complexity
2. Absence of structural independence



Network Data Model

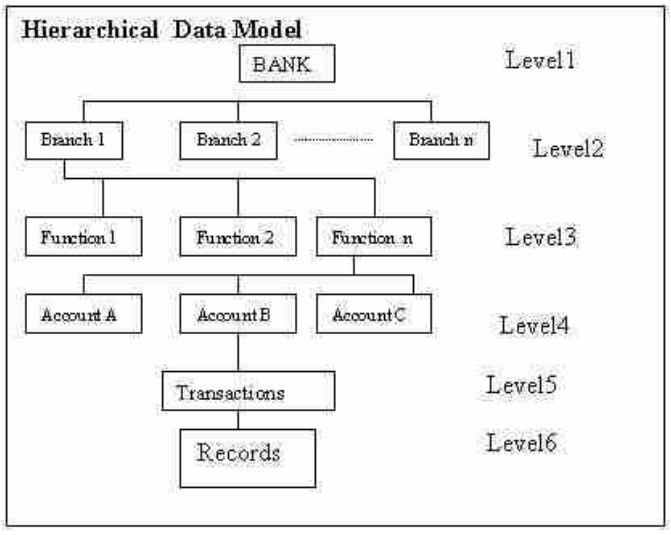
2.2.6 Hierarchical Data Model :

Hierarchical Database model is one of the oldest database models, dating from late 1950s. One of the first hierarchical databases Information Management System (IMS) was developed jointly by North American Rockwell Company and IBM. This model is like a structure of a tree with the records forming the nodes and fields forming the branches of the tree.

Thus hierarchical data model organizes data in a tree structure. There is a hierarchy of parent and child data segments. This structure implies that a record can have repeating information, generally in the child data segments. Data is a series of records, which have a set of field values attached to it. It collects all the instances of a specific record together as a record type. These record types are the equivalent of tables in the relational model, and with the individual records being the equivalent of rows.

This model is applicable when data in the organization can be put down in hierarchical or in terms of levels, one after the other. This model is equivalent to a tree. The tree has roots, branches and leaves, their equivalents in this model beings records, nodes and fields.

In the hierarchical model, data elements are organized in a tree structure in parent – child format. For example, the hierarchical model of a bank database is as shown in the figure.



In the hierarchical model, any information about the child can be accessed only through the parent. For example, if you want any transaction information about an account in the branch, then it will be accessed through the branch – function account route. Direct access to the transaction, even if you know its identity key, is not possible.

In this model, there are issues about loss of data and information, if you delete the entity. When you delete the entity, you lose the information of the child as well as of the tree structure e.g. if you delete account B, its transaction and records are lost.

The typical characteristics of a hierarchical model are as follows:

1. Hierarchical model starts with a root and it has several roots.
2. A root will have several branches.
3. Each branch is connected to one and only one root.
4. A branch has several leaves and a set of leaves is connected to one branch.
5. Thus, the hierarchical tree structure is made up of branches (nodes) and leaves (fields).

In order to understand the hierarchical data model better, let us take the example of the sample database consisting of supplier, parts and shipments. The record structure and some sample records for supplier, parts and shipments elements are as given in following tables.

The Supplier records

Sno	Name	Status	City
S1	Suneet	20	Qadian
S2	Ankit	10	Amritsar
S3	Amit	10	Amritsar

The Part records

Pno	Name	Color	Weight	City
PI	Nut	Red	12	Qadian
P2	Bolt	Green	17	Amritsar
P3	Screw	Bule	17	Jalandhar
P4	Screw	Red	14	Qadian

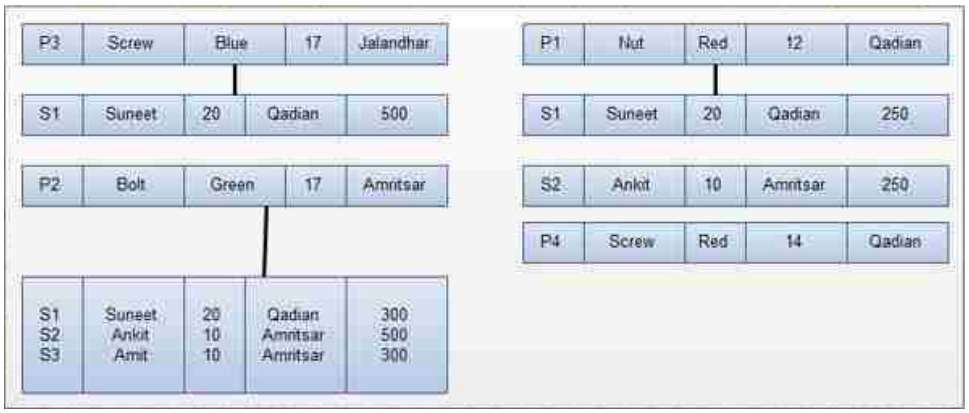
The Shipment records

Sno	Pno	Qty
S1	P1	250
S1	P2	300
S1	P3	500
S2	P1	250
S2	P2	500
S3	P2	300

We assume that each row in Supplier table is identified by a unique SNo (Supplier Number) that uniquely identifies the entire row of the table. Likewise each part has a unique Pno (Part Number). Also we assume that no more than one shipment exists for a given supplier/part combination in the shipments table.

❖ **Hierarchical View for the Suppliers–Parts Database :**

The tree structure has parts record superior to supplier record. That is parts form the parent and supplier forms the children. Each of the four trees figure, consists of one part record occurrence, together with a set of subordinate supplier record occurrences. There is one supplier record for each supplier of a particular part. Each supplier occurrence includes the corresponding shipment quantity.



For example, supplier S3 supplies 300 quantities of part P2. Note that the set of supplier occurrences for a given part occurrence may contain any number of members, including zero (for the case of part P4). Part PI is supplied by two suppliers, S1 and S2. Part P2 is supplied by three suppliers, S1, S2 and S3 and part P3 supplied by only supplier SI as shown in figure.

❖ **Operations on Hierarchical Model :**

There are four basic operations Insert, Update, Delete and Retrieve that can be performed on each model. Now, we consider in detail that how these basic operations are performed in hierarchical database model.

- **Insert Operation –** It is not possible to insert the information of the supplier e.g. S4 who does not supply any part. This is because a node cannot exist without a root. Since, a part P5 that is not supplied by any supplier can be inserted without any problem, because a parent can exist without any child. So, we can say that insert anomaly exists only for those children, which has no corresponding parents.
- **Update Operation –** Suppose we wish to change the city of supplier S1 from Qadian to Jalandhar, then we will have to carry out two operations such as searching S1 for each part and then multiple updates for different occurrences of S1. But, if we wish to change the city of part P1 from Qadian to Jalandhar, then these problems will not occur because there is only a single entry for part P I and the problem of inconsistency will not arise. So, we can say that update anomalies only exist for children not for parent because children may have multiple entries in the database.

- **Delete Operation –** In hierarchical model, quantity information is incorporated into supplier record. Hence, the only way to delete a shipment (or supplied quantity) is to delete the corresponding supplier record. But such an action will lead to loss of information of the supplier, which is not desired. For example: Supplier S2 stops supplying 250 quantity of part PI, then the whole record of S2 has to be deleted under part PI which may lead to loss the information of supplier. Another problem will arise if we wish to delete a part information and that part happens to be only part supplied by some supplier. In hierarchical model, deletion of parent causes the deletion of child records also and if the child occurrence is the only occurrence in the whole database, then the information of child records will also lost with the deletion of parent. For example: if we wish to delete the information of part P2 then we also lost the information of S3, S2 and S1 supplier. The information of S2 and Sl can be obtained from PI, but the information about supplier S3 is lost with the deletion of record for P2.

❖ **Program Work Area / User Work Area :**

In an application environment, as a number of records are accessed from a database, they are retrieved into a set of program variables. The program then accesses the field values of the database records through them. The database system maintains a program work area or user work area. The PWA/ UWA is a buffer storage area for these variables.

The following variables are maintained in a PWA/UWA for each application program:

a. **Record templates –** The variables with data types and width of each field of the record types accessed by the application programs.

b. **Currency Pointers –** A set of pointers, one for each database tree containing the address of the record in that particular tree accessed most recently by the application program.

c. **Status flag –** A variable set (DB – status) by the system to indicate to the application program, the outcome of the last database operation. If DB–status = 0, then it indicates that the last operation succeeded.

❑ **Check Your Progress – 1 :**

1. Explain the concept of Data models.
2. State the difference between Network data model and Hierarchical Model.
3. Explain advantages and disadvantages of relational model.

.....

.....

.....

.....

.....

4. Data Model is collection of conceptual tools for describing –

(A) Data (B) Schema (C) Constraints (D) All of these

5. Data Models in DBMS are classified into _____ categories.
   (A) 3 (B) 2 (C) 5 (D) 4

6. Object based logical model(s) are used to describe data at – [Select Appropriate Option(s)
   (A) View Level (B) Logical Level
   (C) Physical Level (D) None

7. Which of the following is example of Object based logical model ?
   (A) Relational Model (B) Hierarchical Model
   (C) Network Layer (D) ERM

8. _____ is a variable set (DB – status) by the system to indicate to the application program, the outcome of the last database operation
   (A) Status flag (B) Currency Pointer
   (C) None (D) Both

2.3 Let Us Sum Up :

Data Model can be defined as an integrated collection of concepts for describing and manipulating data, relationships between data, and constraints on the data in an organization. The purpose of a data model is to represent data and to make the data understandable. There have been many data models proposed in the literature. They fall into three broad categories: Object Based Data Models, Physical Data Models and Record Based Data Models. The object based and record based data models are used to describe data at the conceptual and external levels, the physical data model is used to describe data at the internal level. Some of the more common types of object based data model are: Entity–Relationship, Object Oriented, Semantic and Functional. The three most widely accepted record based data models are: Hierarchical Model, Network Model and Relational Model. The most common Physical model is the Unifying Model.

2.4 Suggested Answer for Check Your Progress :

❑ **Check Your Progress 1 :**

1 : See Section 2.2

2 : See Section 2.2.5 and 2.2.6

3 : See Section 2.2.4

4 : All of these **5 :** 3

6 : Logical Level **7 :** ERM

8 : Status flag

2.5 Glossary :

1. **Data Model –** It can be defined as an integrated collection of concepts for describing and manipulating data, relationships between data, and constraints on the data in an organization.
2. **Object based data model –** the model which uses concepts such as entities, attributes, and relationships
3. **Record based logical model –** the model is used in describing data at the logical and view levels.

4. **Physical Model –** Physical data models describe how data is stored in the computer, representing information such as record structures, record ordering, and access paths.
5. **Relational model –** the model stores data in the form of tables.
6. **Network model –** The model which organizes entity as graph and can be accessed by several paths.
7. **Hierarchical model –** This model is like a structure of a tree with the records forming the nodes and fields forming the branches of the tree.
8. **CODASYL –** Conference on Data Systems Languages (CODASYL) formally defined the network model.
9. **HDML –** Hierarchical Data Manipulation Language for Hierarchical model
10. **IMS –** Information Management System is one of the oldest and widely used database system introduced by IBM.
11. **DDL –** Data definition Language.

2.6 Assignment :

Pick the various Databases which you are very much familiar with. And identify the data model that suits those databases

2.7 Activities :

Study different object based logical models in detail.

2.8 Case Study :

Study different record based logical models in detail.

2.9 Further Reading :

1. Introduction to Computer Graphics, N Krishnamurthy, Tata McGraw Hill
2. Computer Graphics – Theory into Practice, Jeffrey J. McConnell
3. Procedural element for Computer Graphics, Rogers, 2nd ed, Tata McGraw Hill.
4. Computer Graphics: Principles and Practice, James D. Foley, Andries van Dam, Steven K. Feiner, John F. Hughes.

Some useful websites are as follows :

http://ecomputernotes.com

http://www.w3.org/TR/DOM–Level–2–Core/glossary.html

http://www.w3.org/TR/PR–DOM–Level–1/glossary.html

http://arts.nprcolleges.org/e%20content/swca/Database%20System%20Technique s%20–%20SST8C51.pdf

http://www.cs.iit.edu/~cs561/cs425/PANDURENGAN_VIGNESH RelationalData baseIntro/test/DBModels.html

http://unixspace.com/context/databases.html

http://www.unixspace.com/context/databases.html

http://rakeshpurohit.wordpress.com

http://www.gdcbemina.com/Study–Material/BCOM–FINAL–YEAR–STUDY–MATERIAL(COMPUTER)/B.Com.3rd–Study–mat...

http://sql–plsql–notes.blogspot.in/2011/07/relational–database–management–system.html

http://www.cs.iit.edu/~cs561/cs425/PANDURENGAN_VIGNESH RelationalData baseIntro/test/DBModels.html

http://www.unixspace.com/context/databases.html

Unit 03

ENTITY RELATIONSHIP MODEL & DIAGRAMS

UNIT STRUCTURE

3.0 Learning Objectives

3.1 Introduction

3.2 Entity Set

3.2.1 What is Entity Set ?

3.2.2 What is weak Entity Set ?

3.3 Attribute

3.4 Relationship Set

3.5 ER Diagrams

3.6 Let Us Sum Up

3.7 Suggested Answer for Check Your Progress

3.8 Glossary

3.9 Assignment

3.10 Activities

3.11 Case Study

3.12 Further Readings

3.0 Learning Objectives :

After learning this unit, you will be able to understand :

- The need for entity–relationship modeling
- The terms and notations related to entity–relationship model, entity–relationship diagram
- To understand various terms like entity type, entity, attribute, attribute value, primary key, relationship, relationship type etc.

3.1 Introduction :

The E–R data model is based on a perception of a real world that consists of a set of basic objects called entities and of relationships among these objects. It was developed to facilitate database design by allowing the specification of an enterprise schema, which represents the overall logical structure of a database. The E–R model is extremely useful in mapping the meanings and interactions of real–world enterprises onto a conceptual schema. Because of this utility, many database–design tools draw on concepts from the E–R model.

There are three basic notations that the E–R Model employs:

1. Entity Sets.
2. Relationship sets.
3. Attributes.

Entities and Entity sets :

3.2 Entity Set :

3.2.1 What is Entity Set ?

An Entity is any object of interest to and organization or for the representation in the database. They represent objects in the real world which is distinguishable from all other objects.

For Eg : Every person in a college is an entity.

Every room in a college is an entity.

Associated with an entity is a set of properties. These properties are used to distinguish to from one entity to another entity.

For Eg : 1. The Attributes of the entity of student are

USN, Name, Address.

2. The Attributes of the Entity of Vehicle are

Vehicle no, Make, Capacity.

For the purpose of accessing and storing information only certain attributes are used.

Those attributes which uniquely identify every instance of the entity is termed as primary key.

An Entity which has a set of attributes which can uniquely identify all the entities is termed as Strong entity.

An entity whose primary key does not determine all the instance of the entity uniquely termed as weak entity.

A collection of similar entities, which has certain properties which are common forms an entity set for organization such as a college the object of concern include Student, Teacher, Rooms, Subjects. The collection of similar entities forms entity set.

3.2.2 What is Weak Entity Set ?

An entity set may not have sufficient attributes to form a primary key, and its primary key compromises of its partial key and primary key of its parent entity, then it is said to be Weak Entity set.

❑ **Check Your Progress – 1 :**

1. What is Entity Set ?

.....

.....

.....

.....

.....

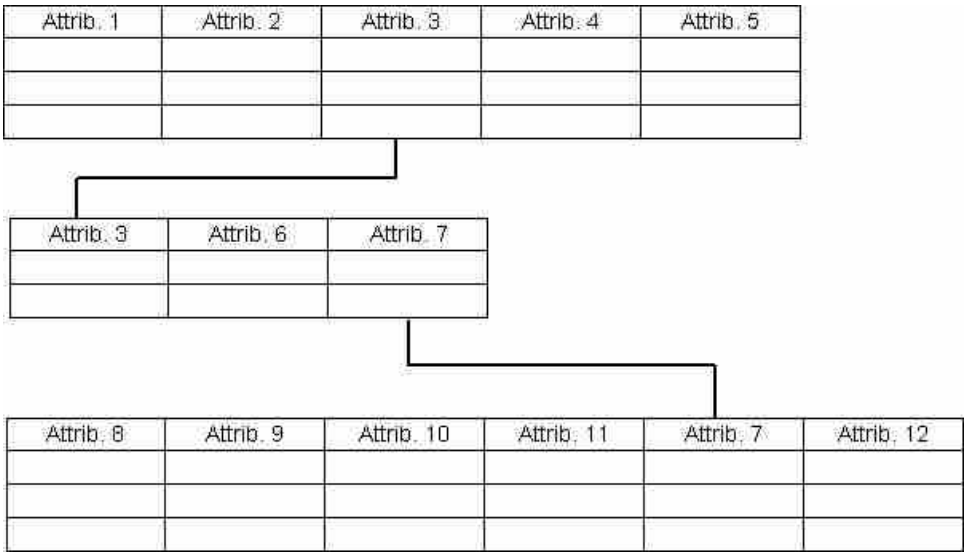
3.3 Attribute :

❖ **Attribute / Column :**

A column stores an attribute of the entity. For example, if details of students are stored then student name is an attribute; course is another attribute and so on.

In general, an attribute is a property or characteristic. Colour, for example, is an attribute of your hair. In using or programming computers, an attribute is a changeable property or characteristic of some component of a program that can be set to different values.

In a database management system (DBMS), an attribute may describe a component of the database, such as a table or a field, or may be used itself as another term for a field.



❖ **Types of attributes :**

1. **Simple Attributes –** The attributes which cannot be further divided into subparts.

   Eg : University Seat Number of a student is unique which cannot be further divided.

2. **Composite Attributes –** The attributes can be further divided into portions.

   Eg : The attribute name in the Student Database can be further divided into First

   name, Middle name, Last name.

   Name

   First name Middle name Last name

3. **Single valued Attributes –** The attribute at any instant contains only a specific value at any instant.

   for eg The USN is unique

4. **Multivalued Attributes –** Certain attributes for example the dependent name in the policy database may have set of values assigned to it. There may be more than one dependent for a single policy holder.

5. **Stored Attributes –** For a person entity, the value of age can be determined from the current date and the value of that person's birthdate. The Age attribute is hence derived attribute and is said to be derivable from the birthdate attributes, which is called a stored attributes.

6. **NULL Attributes –** A NULL value attribute is used when an attributes does not have any values.

❑ **Check Your Progress – 2 :**

1. Explain concept of Attribute with proper example

.....

.....

.....

.....

.....

3.4 Relationship Set :

A relationship is an association of entities where the association includes one entity from each participating entity type. The relationship between entities may be

One–to–one, one–to–many or many–to–many.

If you take entities student, batch and subject, the following are the possible relationships.

There is one–to–one relationship between batch and subject. One batch is associated with only one subject.

Three is one–to–many relationship between batch and student entities. One batch may contain many students.

There is many–to–many relationship between student and subject entities. A single student may take many subjects and a single subject may be taken by multiple students.

❑ **Check Your Progress – 3 :**

1. State with example – What is Relationship Set ?

.....

.....

.....

.....

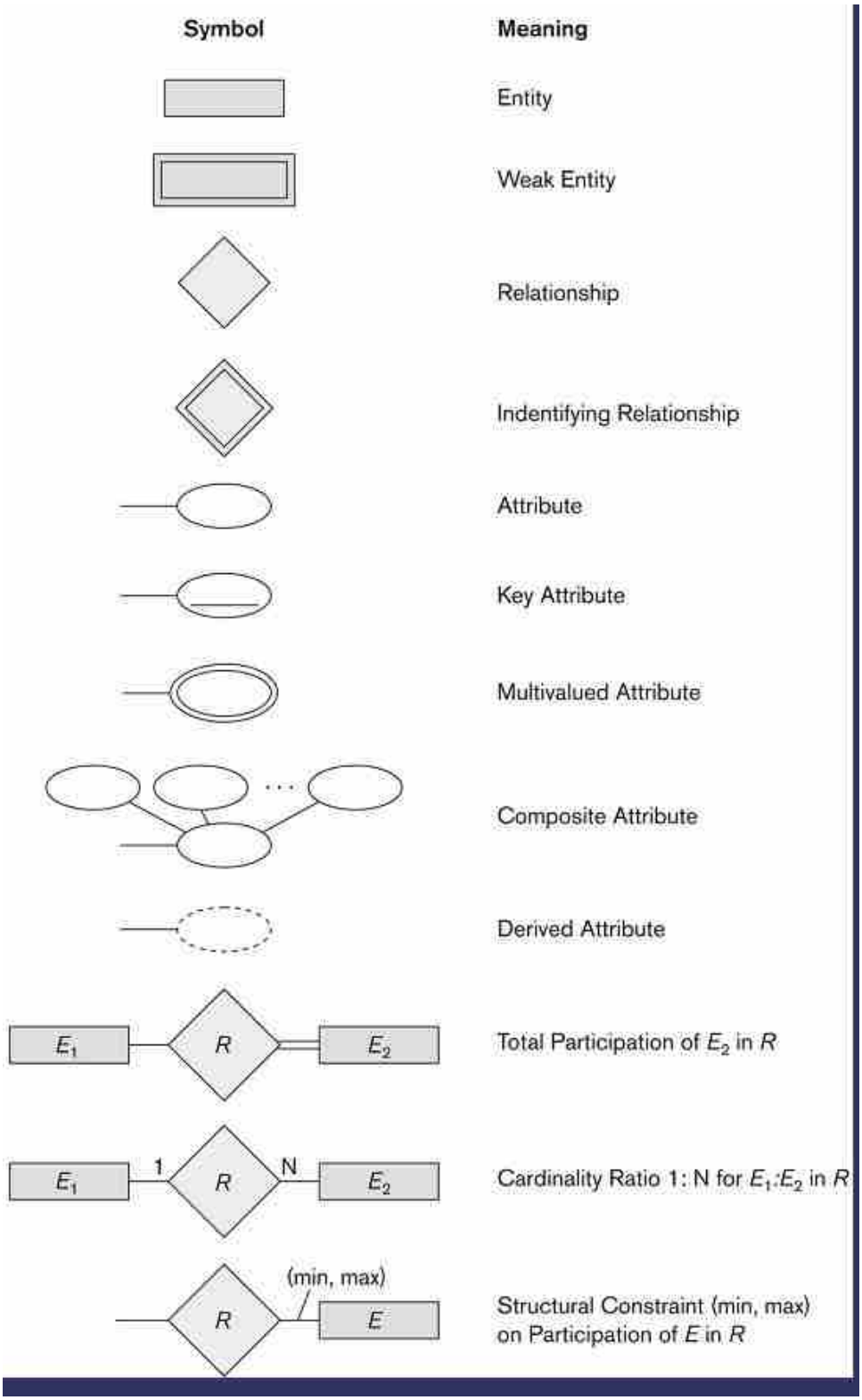
.....

3.5 ER Diagrams :

Figure depicts the entity relationship diagram for a store system. In this diagram, the line connecting entities are shown. The arrows also indicate the nature of relationships. The one to one (1:1) relationship occurs when one entity is associated exactly with the other. For example, one receipt causes only one payable and there is only one payable for every receipt so the lines connecting these entities have only one arrow.

There can also be a one–to–many (1: n) relationship. One order contains many different order items so the line connecting order–to–order item has a dark arrow pointing to the order item and a single arrow pointing to the order. Finally there are many to many (n: n) relationships. There can be many items on a sale and many sales of item double arrow point to each entity.

The symbols used in ER diagram are as follows :



Here we are going to design an Entity Relationship (ER) model for a college database. Say we have the following statements.

1. A college contains many departments
2. Each department can offer any number of courses
3. Many instructors can work in a department
4. An instructor can work only in one department
5. For each department there is a Head

6. An instructor can be head of only one department
7. Each instructor can take any number of courses
8. A course can be taken by only one instructor
9. A student can enroll for any number of courses
10. Each course can have any number of students

Step 1 : Identify the Entities

What are the entities here ?

From the statements given, the entities are

1. Department
2. Course
3. Instructor
4. Student

Step 2 : Identify the relationships

1. One department offers many courses. But one particular course can be offered by only one department. hence the cardinality between department and course is One to Many (1:N)
2. One department has multiple instructors. But instructor belongs to only one department. Hence the cardinality between department and instructor is One to Many (1:N)
3. One department has only one head and one head can be the head of only one department. Hence the cardinality is one to one. (1:1)
4. One course can be enrolled by many students and one student can enroll for many courses. Hence the cardinality between course and student is Many to Many (M:N)
5. One course is taught by only one instructor. But one instructor teaches many courses. Hence the cardinality between course and instructor is Many to One (N :1)

Step 3 : Identify the key attributes

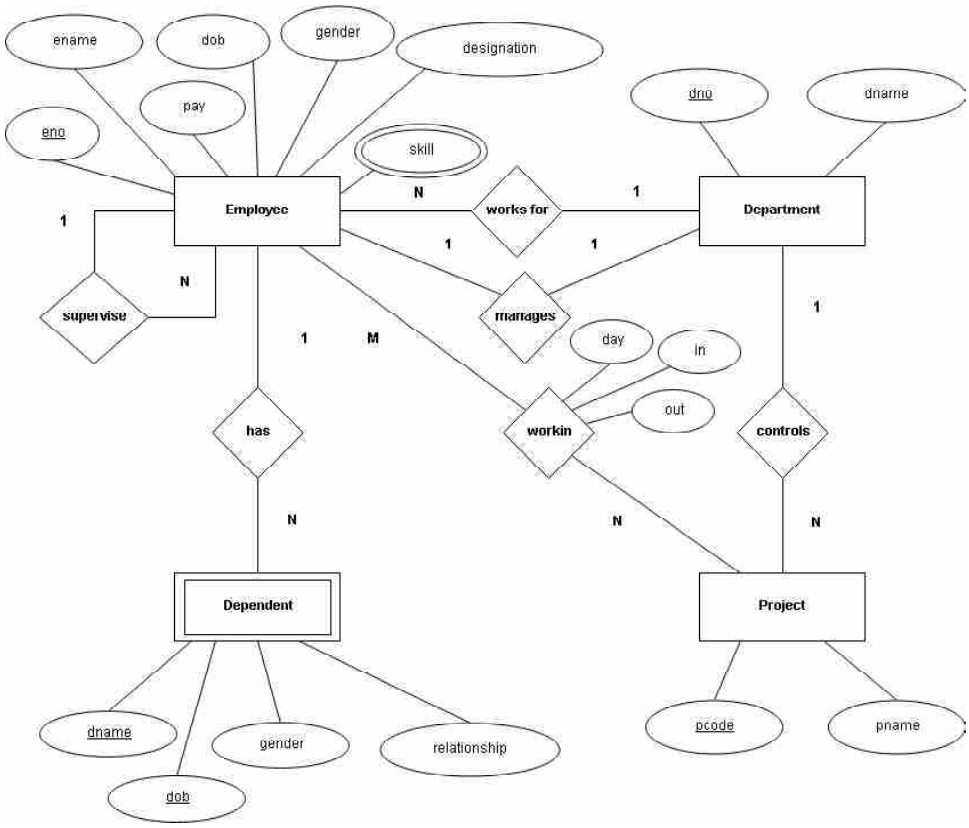
- "Departmen_Name" can identify a department uniquely. Hence Department_Name is the key attribute for the Entity "Department".
- Course_ID is the key attribute for "Course" Entity.
- Student_ID is the key attribute for "Student" Entity.
- Instructor_ID is the key attribute for "Instructor" Entity.

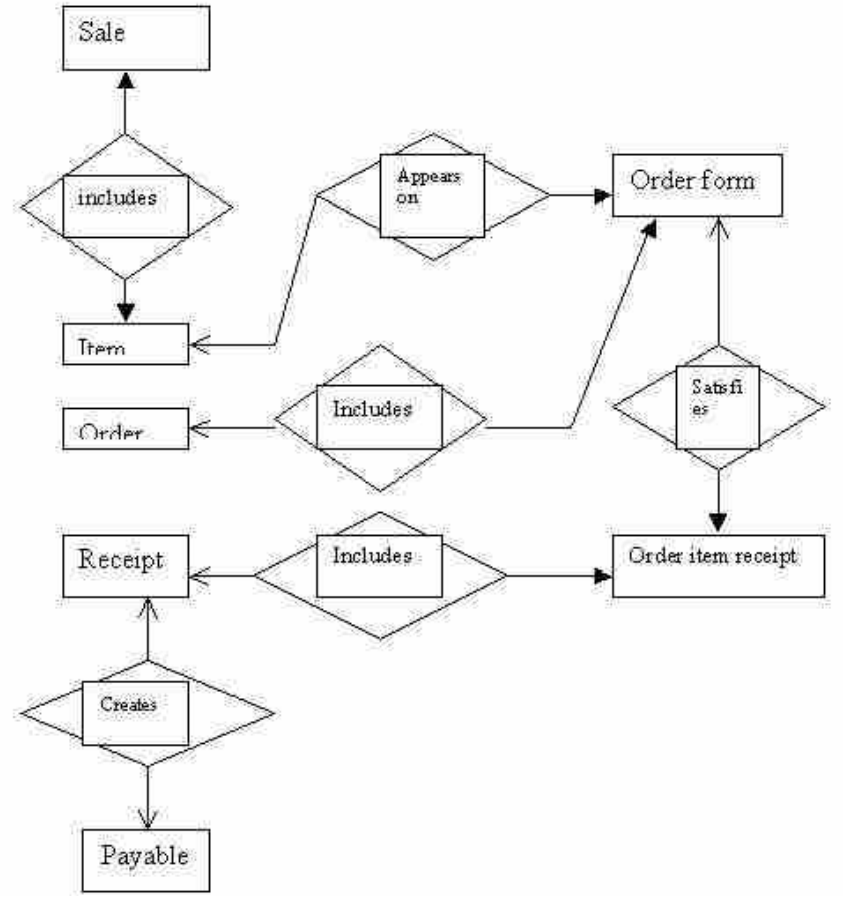
Step 4 : Identify other relevant attributes

- For the department entity, other attributes are location
- For course entity, other attributes are course_name,duration
- For instructor entity, other attributes are first_name, last_name, phone
- For student entity, first_name, last_name, phone

Step 5 : Draw complete ER diagram

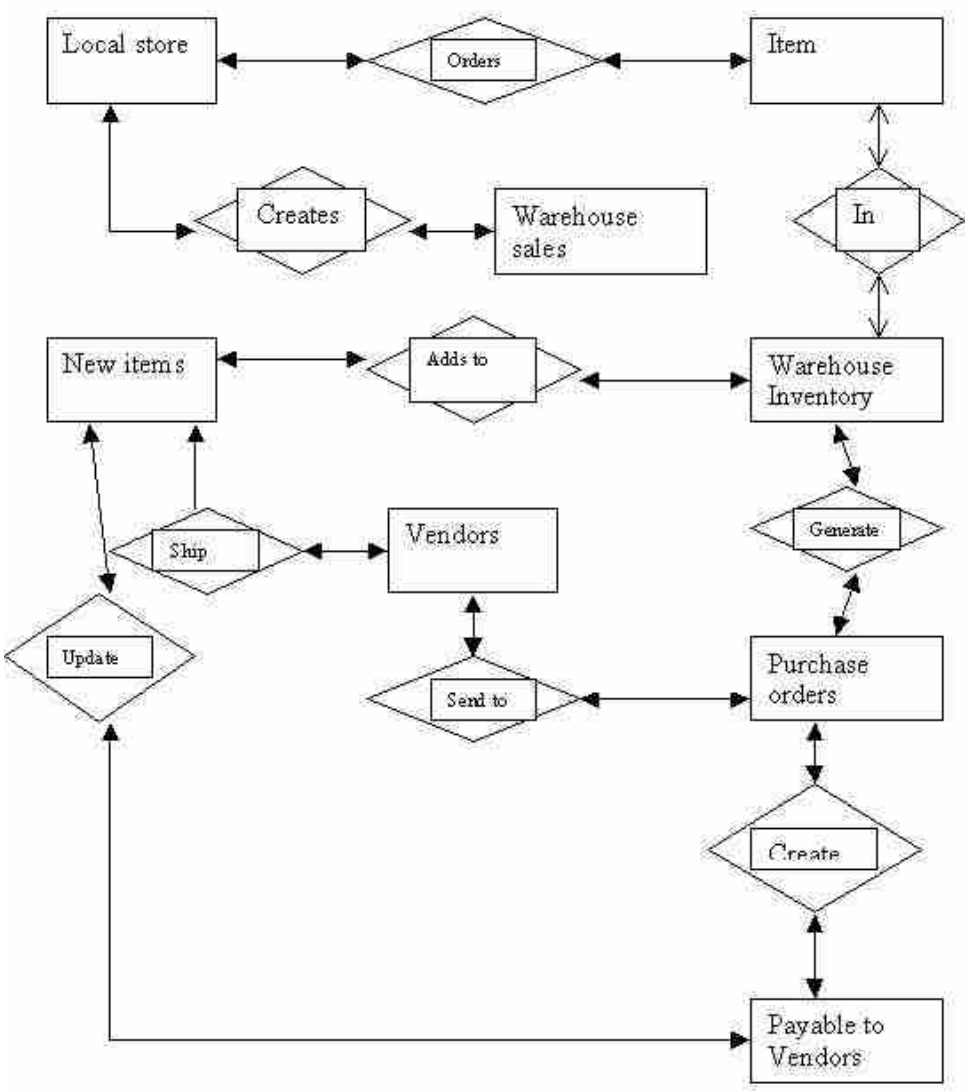
By connecting all these details, we can now draw ER diagram as given below.





Entity relationship diagram of store system

Figure is a rough draft ER diagram of the warehouse system. Here we use the convention of single and dark arrows to represent 1:1, 1: n etc relationships.



Entry–relationship diagram of warehouse system

❑ **Check Your Progress – 4 :**

1. What are Entity Relationship Diagrams ? Explain with example

.....

.....

.....

.....

.....

2. Entity Relationship model consists of collection of basic objects called _____ and relationship among these objects.

(A) Functions (B) models (C) entities (D) None

3. Object which is distinguishable from other objects by specific set of attributes is called as _____.

(A) Entity (B) Classes (C) Attributes (D) None

4. Association among several entities is called as _____.

   (A) Relationship (B) Association

   (C) Combination (D) Extraction

5. _____ express the number of entities to which another entity can be associated via a relationship set.

   (A) Mapping Cardinality (B) Messaging Cardinality

   (C) Logical Cardinality (D) None

6. is an association of entities where the association includes one entity from each participating entity type.

   (A) Entity (B) Relationships

   (C) Diagram (D) Node

3.6 Let Us Sum Up :

An entity is any object that is stored in the database. Each entity is associated with a collection of attributes. For example, if you take a data of a training institute, student is an entity as we store information about each student in the database. Each student is associated with certain values such as roll number, name, course etc., which are called as attributes of the entity.

In general, an attribute is a property or characteristic. Colour, for example, is an attribute of your hair. In using or programming computers, an attribute is a changeable property or characteristic of some component of a program that can be set to different values.

There will be relationship among entities. The relationship between entities may be

One–to–one, one–to–many or many–to–many.

If you take entities student, batch and subject, the following are the possible relationships.

There is one–to–one relationship between batch and subject. One batch is associated with only one subject.

Three is one–to–many relationship between batch and student entities. One batch may contain many students.

There is many–to–many relationship between student and subject entities. A single student may take many subjects and a single subject may be taken by multiple students.

3.7 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 3.2

- **Check Your Progress 2 :**

  See Section 3.3

- **Check Your Progress 3 :**

  See Section 3.4

❑ **Check Your Progress 4 :**

1 : See Section 3.5

2 : Entities

3 : Entity

4 : Relationships

5 : Mapping Cardinality

6 : Relationships

3.8 Glossary :

1. **ER model –** It is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity–Relationship diagram, which is used to visually represent data objects.
2. **Entity –** An entity is something which is described in the database by storing its data, it may be a concrete entity a conceptual entity.
3. **Entity set –** An entity set is a collection of similar entities.
4. **Attribute –** An attribute describes a property associated with entities. Attribute will have a name and a value for each entity.
5. **Domain –** A domain defines a set of permitted values for a attribute.
6. **Relationship –** A relationship is a link or association between entities. Relationships are usually denoted by verb phrases.
7. **Weak entity –** An entity may not have sufficient attributes to form a primary key, and its primary key compromises of its partial key and primary key of its parent entity, then it is said to be Weak Entity.
8. **Candidate key –** If an attribute can be thought of as a unique identifier for an entity, it is called a candidate key.
9. **Primary key –** When a candidate key is chosen to be the unique identifier, it becomes the primary key for the entity.

3.9 Assignment :

Explain in your own words how to define Entity, attribute and relationship with example.

3.10 Activities :

Explain ER–design methodology.What are problems with ER Model ?

3.11 Case Study :

A small accounting firm wants a simple HR application that will help it to keep track of its employees, their positions, allowances, salary scales, and which company vehicles their employees drive.

The application must keep track of all the positions at the firm, the employees filling these positions, the allowances for these positions, the salary scales for these positions, and the company vehicles assigned to these positions.

Identify entities, relationship within entities and draw ER diagram for the above case study.

3.12 Further Reading :

1. Introduction to Database Systems by C.J.Date
2. Database Management Systems – Rajesh Narang –PHI Learning Pvt Ltd
3. Database System Concepts by Silberschatz,Korth –Tata McGraw–Hill Publication
4. An Introduction to Database Systems – Bipin Desai– Galgotia Publication
5. Database Management System by Raghu Ramkrishnan– Tata McGraw–Hill Publication
6. SQL, PL/SQL:The Programming Language Oracle – Ivan Bayross– BPB Publication

Some useful websites are as follows :

http://ecomputernotes.com/fundamental/what–is–a–database/network–model

http://rakeshpurohit.wordpress.com

http://www.authorstream.com/Presentation/nehasinghal–761246–dbms

http://www.slidsseshare.net/the10dar/document–33319174

http://rakeshpurohit.wordpress.com

http://www.gdcbemina.com/Study–Material/BCOM–FINAL–YEAR–STUDY–MATERIAL(COMPUTER)/B.Com.3rd–Study–mat...

http://www.orafaq.com/wiki/Talk:DBMS

http://www.answers.com/Q/What_are_the_attributes_of_an_Object

http://www.programmingking.in/2013_05_26_archive.html

BLOCK SUMMARY :

Unit 1 provides concept of database and database management system. Database is well organized data about particular enterprise. DBMS is a collection of interrelated data and set of programs to access those data. The primary goal of DBMS is to provide a way to store and retrieve database information that is both convenient and efficient. Keeping organizational information in file processing system has major disadvantages like data redundancy and inconsistency, difficulty in accessing data, data isolation, integrity and security and concurrent access anomalies. In order to remove drawbacks, database management system comes into existence. A major purpose of database management system is to provide users with an abstract view of the data. Data abstraction is provided at physical, logical and view level. Database system has schemas according to levels of abstraction: physical, logical and sub–schema. There are different types of database users: naïve users, application programmers and sophisticated users and specialized users. Database Administrator is person who has central control over database. Components of DBMS are Transaction manager, storage manager, query processor, file manager and buffer manager.

Unit 2 provides the concept of data models. Data Model can be defined as an integrated collection of concepts for describing and manipulating data, relationships between data, and constraints on the data in an organization. The purpose of a data model is to represent data and to make the data understandable. They fall into three broad categories: Object Based Data Models, Physical Data Models and Record Based Data Models.

Unit 3 explains the concept of ER models. In Entity–Relationship model a database is modeled as a collection of entities and relationship among entities. The ER model views the real world as a construct of entities and association between entities. A basic component of the model is the Entity–Relationship diagram, which is used to visually represent data objects. For the database designer, the utility of the ER model is wide. It maps well to the relational model. The constructs used in the ER model can easily be transformed into relational tables. It is simple and easy to understand with a minimum of training. Therefore, the model can be used by the database designer to communicate the design to the end user. In addition, the model can be used as a design plan by the database developer to implement a data model in specific database management software.

BLOCK ASSIGNMENT :

❖ **Short Questions :**

1. List different users of database management system.
2. What are different types of record based logical model ?
3. What are different types of object based logical model ?
4. Define: Entity, attribute, relationship
5. List different types of relationships among entities.

❖ **Long Questions :**

1. What is DBMS ? State its advantages and disadvantages.
2. Explain Relational model in detail.
3. Explain different attribute types with respect to ER model.
4. Draw the ER Diagram for Library Management System.

BIBLIOGRAPHY

http://www.webopedia.com/TERM/D/database_management_system_DBMS.html

http://searchsoa.techtarget.com/definition/attribute

http://www.fidelcaptain.com/casestudy1/entitiesandattributescs1.html

❖ **Enrolment No. :** ☐

1. How many hours did you need for studying the units ?

Unit No.	1	2	3
No. of Hrs.			

2. Please give your reactions to the following items based on your reading of the block :

Items	Excellent	Very Good	Good	Poor	Give specific example if any
Presentation Quality	☐	☐	☐	☐	_____
Language and Style	☐	☐	☐	☐	_____
Illustration used (Diagram, tables etc)	☐	☐	☐	☐	_____
Conceptual Clarity	☐	☐	☐	☐	_____
Check your progress Quest	☐	☐	☐	☐	_____
Feed back to CYP Question	☐	☐	☐	☐	_____

3. Any other Comments

.....

.....

.....

.....

.....

.....

.....

.....

Dr. Babasaheb Ambedkar Open University Ahmedabad

BCAR-202/ DCAR-202

Database Management System

BLOCK 2 : RELATIONAL DATABASE AND DATABASE DESIGN

UNIT 4 INTRODUCTION TO RELATIONAL DATABASE

UNIT 5 DATABASE DESIGN

UNIT 6 NORMALISATION

RELATIONAL DATABASE AND DATABASE DESIGN

Block Introduction :

Unit 4 provides overview of relational database. "Relational database was proposed by Edgar Codd (of IBM Research) around 1969. It has since become the dominant database model for commercial applications (in comparison with other database models such as hierarchical, network and object models). Today, there are many commercial Relational Database Management System (RDBMS), such as Oracle, IBM DB2 and Microsoft SQL Server. There are also many free and open–source RDBMS, such as MySQL, mSQL (mini–SQL) and the embedded JavaDB" (Apache Derby).

A relational database organizes data in tables (or relations). A table is made up of rows and columns. A row is also called a record (or tuple). A column is also called a field (or attribute). A database table is similar to a spreadsheet. However, the relationships that can be created among the tables enable a relational database to efficiently store huge amount of data, and effectively retrieve selected data.

Unit 5 discusses design issues in relational databases. The goal of relational database design is to generate a set of relational schemas that allows us to store information without redundancy and allows us to retrieve it easily. One approach is to design schemas that are in appropriate normal forms.

In Unit 6 the need of normalization process is explained. Also basic three normal forms are discussed with examples. It also explained all the 6 NF with examples and give detail understanding.

Block Objectives :

After learning this block, you will be able to :

- Understand concept of Relational database
- Understand Codd's 12 Rules and key concepts
- Understand Integrity Constraints
- Understand drawbacks of un–normalized database
- Understand concept of normalization and different normal forms

Block Structure :

Unit 4 : Introduction To Relational Database

Unit 5 : Database Design

Unit 6 : Normalisation

Unit 04

INTRODUCTION TO RELATIONAL DATABASE MANAGEMENT SYSTEM

UNIT STRUCTURE

4.0 Learning Objectives

4.1 Introduction

4.2 Codd's 12 Rules

4.3 Terms

4.4 Keys

4.5 Anomalies of Un–normalized Database

4.6 Comparison Hierarchical, Network and Relational Databases

4.7 Let Us Sum Up

4.8 Suggested Answer for Check Your Progress

4.9 Glossary

4.10 Assignment

4.11 Activities

4.12 Case Study

4.13 Further Readings

4.0 Learning Objectives :

After learning this unit, you will be able to :

- Understand Codd's 12 Rules
- Understand primary key, foreign key concept
- Understand Integrity Constraints
- Understand drawbacks of un–normalized database.
- Understand comparison of different data models.

4.1 Introduction :

Database of business computing from the beginning of the digital age is a fastener. EF Codd, a researcher at IBM, wrote a letter outlining the process, when in fact, was born in 1970 in a relational database. Since then, relational databases have become popular and standard.

Originally, databases were flat. This means that the information was stored in one long text file, called a tab delimited file. Each entry in the tab delimited fil e is separated by a special character, such as a vertical bar (|). Each entry contains multiple pieces of information about a particular object or person grouped togethe r as a record. It is very difficult to search for specific information or to create reports in text files.

A relational database allows you to easily obtain specific information. It also lets you sort the data by any field and each record contains only a few areas that allows you to generate reports. Relational databases use tables to store information. Standard fields and records in a table columns (fields) and rows (records) are represented.

With a relational database, you quickly learn to arrange the data in the columns can compare. To increase the speed and versatility of the database similar uses relationship data. The name "relational" part is derived from the mathematical relationships. On the table to gather information from other tables, each table can key in a column or columns.

Relational databases are created using a special computer language, structured query language (SQL) that is the standard for database interoperability. SQL is the foundation for all of the popular database applications available today.

4.2 Codd's 12 Rules :

"Dr. Edgar F. Codd's relational model database system has done extensive research and came up with twelve rules, According to him, the database must comply in order to be a true relational database. These rules by using their relational data warehousing capabilities and retrieving a database system that is able to be applied

Rule 1 : Information rule – This rule is stored in the database, all data should be a part of a table cell. Everything in the database should be stored in tables or relations. This information can be user data or Meta data.

Rule 2 : Guaranteed Access rule – The rule table every data element names, attribute names in conjunction with the logical primary key and is assured to be accessible explains.

Rule 3 : Systematic Treatment of NULL values – This rule states in the database should be a systematic treatment of NULL values. After the tap may be interpreted as a data iss missing, the data is not known, the data is not applicable etc.

Rule 4 : Active online data dictionary – This rule states in the database should be a systematic treatment of NULL values. After the tap may be interpreted as a data is missing, the data is not known, the data is not applicable etc

Rule 5 : Comprehensive data sub–language rule – This rule is a database data definition, data manipulation and transaction management that is capable of conducting linear syntax which must be a support for a language which tells. Database itself, either directly or through the language of the application can be accessed through. Database access or the language without any help that can be manipulated in some way, it is a violation

Rule 6 : View updating rule – This rule could theoretically be updated, the views of the database, it should also be updatable by the system.

Rule 7 : High–level insert, update and delete rule – The rule database support high–level entry, updating and deleting states should employ. It should not be confined to a single line, set it to record data Union, intersection and should support the operation of the loan

Rule 8 : Physical data independence – Application data is physically stored are rules about how to be ignorant. In addition, any change in its physical structure should not affect the application

Rule 9 : Logical data independence – The rules of logical data view user (application) should be independent of the states. Logical data using a change should not affect any change in the application

Rule 10 : Integrity independence – This rule should be independent of the database application that states using it. The lack of honesty freely without requiring any changes to the application can be modified. This rule makes the front–end application and its interface is independent of the database

Rule 11 : Distribution independence – This rule is distributed at various locations of data that must be unaware that states how the end user. User data is located on the same site would see only. The rules in distributed database systems has been proven as a base.

Rule 12 : Non–subversion rule –

This rule is a system that provides low –level access to records is an interface, the interface of the system and bypass the security and integrity of the barrier should not be able to destroy the state."

❑ **Check Your Progress – 1 :**

1. Explain Codd's 12 rules.

.....

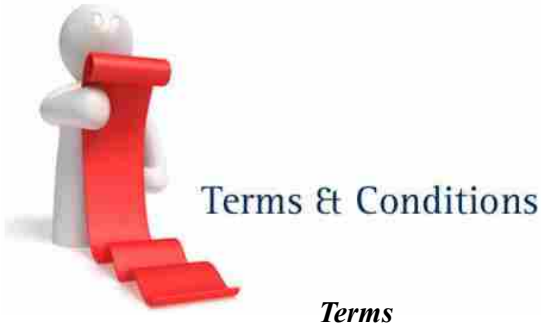
.....

.....

.....

.....

4.3 Term :



Terms

- In the relational model, a database is a collection of relational tables.
- Relations can be represented as two–dimensional data tables with rows and columns
- The rows of a relation are called tuples.
- The columns of a relation are called attributes.
- The attributes draw values from a domain (a legal pool of values).
- The number of tuples in a relation is called its cardinality while the number of attributes in a relation is called its degree.
- A relation also consists of a schema and an instance.
- Schema defines the structure of a relation which consists of a fixed set of attribute domain pairs.
- An instance of a relation is a time–varying set of tuples where each tuple consists of attribute–value pairs.

Relational tables can be expressed concisely by eliminating the sample data and showing just the table name and the column names.

For example,

AUTHOR (au_id, au_lname, au_fname, address, city, state, zip)

TITLE (title_id, title, type, price, pub_id)

PUBLISHER (pub_id, pub_name, city)

AUTHOR TITLE (au_id, title_id)

❖ **Properties of Relational Tables :**

Relational tables have six properties :

1. Values are atomic.
2. Column values are of the same kind.
3. Each row is unique.
4. The sequence of columns is insignificant.
5. The sequence of rows is insignificant.
6. Each column must have a unique name.

❑ **Check Your Progress – 2 :**

1. Explain properties of relational table.

.....

.....

.....

.....

.....

4.4 Keys :

❖ **Key :**

A set of attributes whose values, k, a tuple to identify any specific examples and none of Kashmir proper subsets property

Example : {rollNumber} is a key for studentrelation. {rollNumber, name} – values can uniquely identify a tuple but the set is not minimal. So it is not a Key. A particular example of a key cannot be determined from the data. It's an integral scheme. It property can be determined only by means of attributes. A relationship may be more than one key.

Each of the keys is called a candidate key.

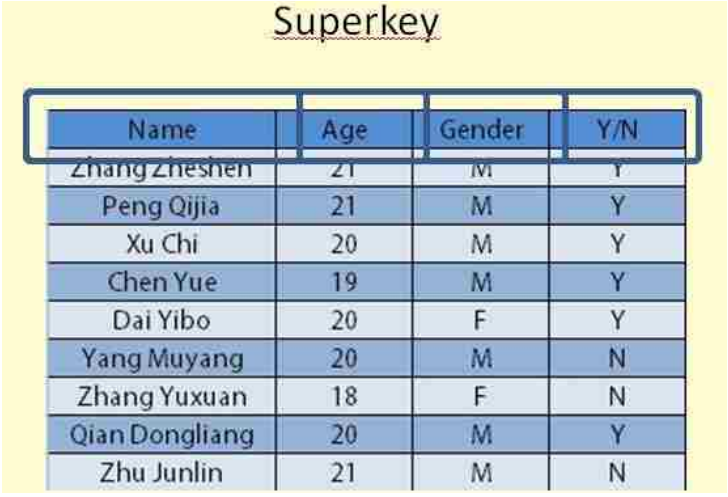
Example : book (isbnNo, authorName, title, publisher, year)

(Assumption : books have only one author)

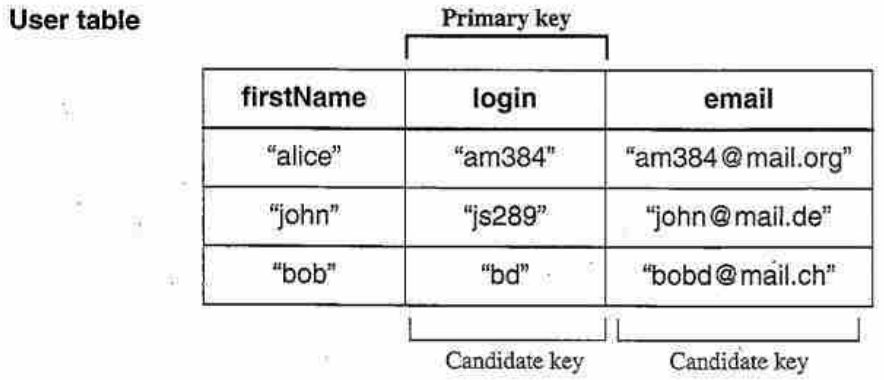
Keys : {isbnNo}, {authorName, title}

f A relation has at least one key i.e. The set of all attributes, in case no proper subset is a key.

- **Superkey –** A set of attributes that contains any key as a subset. A key can also be defined as a minimal superkey.



- **Primary Key –** One of the candidate keys chosen for indexing purposes. This is key whose values uniquely identify each row in a table.

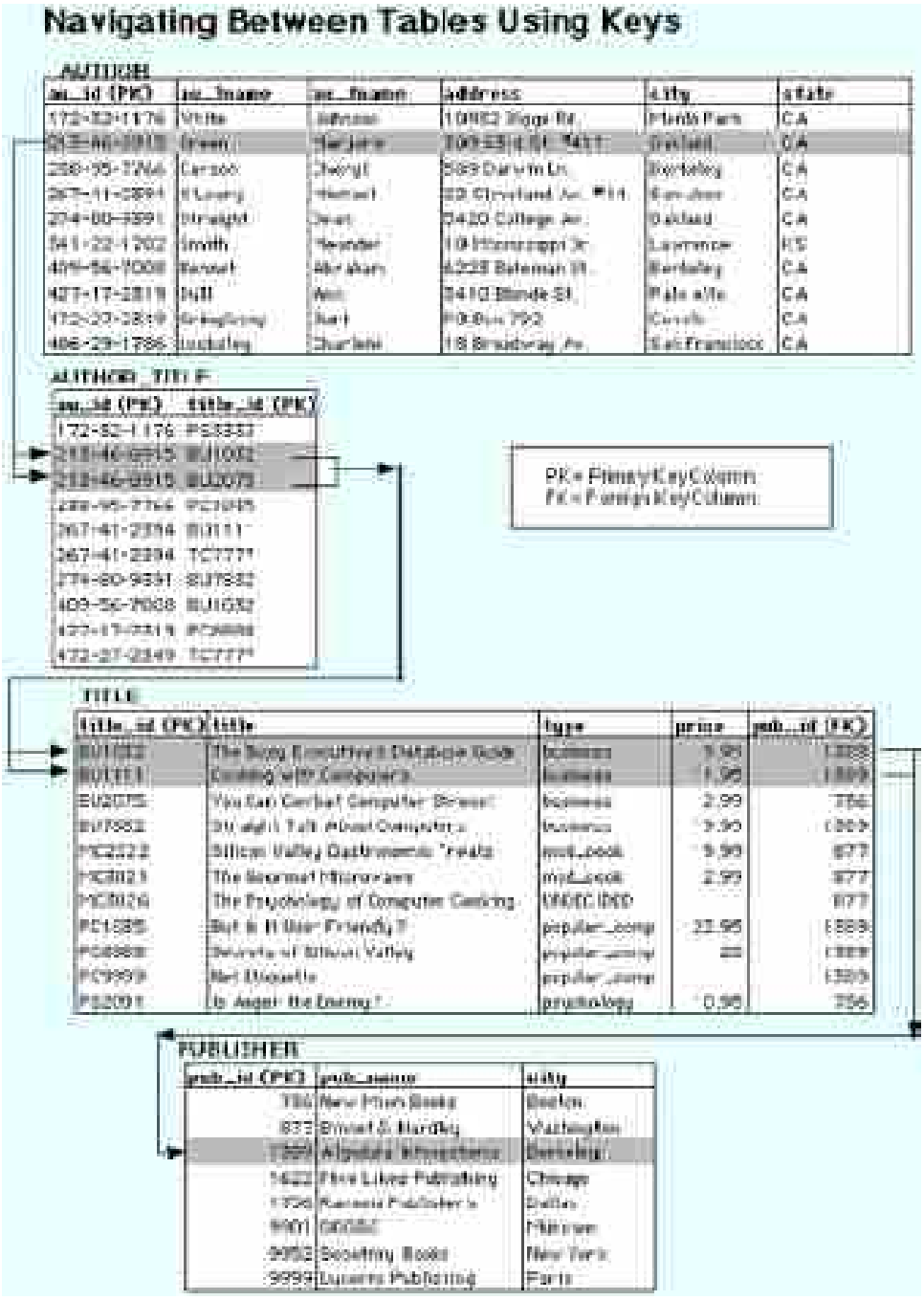


- **A foreign key –** is key whose values are the same as the primary key of another table. Foreign key is a copy of primary key from another relational table. The relationship is made between two relational tables by matching the values of the foreign key in one table with the values of the primary key in another.

Keys are fundamental to the concept of relational databases because they enable tables in the database to be related with each other. Navigation around a relational database depends on the ability of the primary key to unambiguously identify specific rows of a table. Navigating between tables requires that the

foreign key is able to correctly and consistently reference the values of the primary keys of a related table.

For example, the figure below shows how the keys in the relational tables are used to navigate from AUTHOR to TITLE to PUBLISHER. AUTHOR_TITLE is an all key table used to link AUTHOR and TITLE. This relational table is required because AUTHOR and TITLE have a many–to–many relationship.



How the keys are used to navigate Data Integrity

Relational Database Scheme and Instance –

Relational database scheme : it consists of a finite one. I have a set of plans and the lack of integrity of the relationship. Lack of integrity of the data values, so that the necessary conditions set constitute a meaningful relational database data values in the cases are to be satisfied. The following are the constraints –

- Domain constraints
- Key constraints
- Referential integrity constraints

Domain constraints – Properties have been associated with the domain. Domain is set for a specific type of atomic data values. Domain constraint declared the actual values of an attribute in a tuple to be related to the domain stipulates that. For example, the employee ID must be unique, the employee birthday is in the range [Jan 1, 1950, Jan 1, 2000]. Such information is provided in logical statements called integrity constraints.

Key constraints or Entity constraint – Each table has a primary key is required. Primary key, nor any part of the primary key, can contain null values. We identify a few lines for the primary key cannot be null value because it means. Thus entity integrity constraint states that the primary key cannot be null. For example, in the EMPLOYEE table, mobile cannot be a key since some people may not have a mobile.

Referential integrity constraints – This integrity constraints works on the concept of Foreign Key. A key attribute of a relation can be referred in other relation, where it is called ***foreign key***. The referential integrity rule states that if a relational table has a foreign key, then every value of the foreign key must either be null or match the values in the relational table in which that foreign key is a primary key.

❑ **Check Your Progress – 3 :**

1. Define primary, superkey and foreign key.

.....

.....

.....

.....

.....

4.5 Anomalies of Un–normalized Database :

In a relational database, a logical and efficient design is very important. A poorly designed database can be difficult to use, may provide incorrect information, or it may fail to work properly.

Most of these problems are the result of design features, two bad redundant data and inconsistencies. Redundant data set of data is twofold. Discrepancies, delete, insert and update to undermine the integrity of your data due to irregularity any event that causes inconsistent data.

A company obtains parts from a number of suppliers. Each supplier is located in one city. A city can have more than one supplier located there and each city has a status code associated with it. Each supplier may provide many parts. The company creates a simple relational table to store this information that can be expressed in relational notation as :

FIRST (s#, status, city, p#, qty)

FIRST

s#	status	city	p#	qty
s1	20	London	p1	300
s1	20	London	p2	200
s1	20	London	p3	400
s1	20	London	p4	200
s1	20	London	p5	100
s1	20	London	p6	100
s2	10	Paris	p1	300
s2	10	Paris	p2	400
s3	10	Paris	p2	200
s4	20	London	p2	200
s4	20	London	p4	300
s4	20	London	p5	400

- Insert Anomaly When certain attributes cannot be inserted into database without presence of other attributes insert anomaly occurs. For example, in above relation INSERT anomaly would occur if a certain supplier (s5) is located in a particular city (Athens) is added, and he has not supplied a part.
- DELETE Anomaly occurs when certain attributes are lost because of deletion of other attributes. In above example, if a row is deleted, then the information about quantity, part and the supplier would be lost.
- UPDATE Anomaly exists when one or more instances of duplicated data is updated, but not at all. If supplier s1 moved from London to New York, then six rows would have to be updated with this new information.

❑ **Check Your Progress – 4 :**

1. Explain anomalies of un–normalized database.

.....

.....

.....

.....

.....

4.6 Comparison Hierarchical, Network and Relational Databases :

Data models such as hierarchical model, network model and relational model can be compared in terms of data structures, Data manipulation and Data integrity.

Characteristic	Hierarchical model	Network model	Relational model
Data structure	One to many or one to one relationships	Allowed the network model to support many to many relationships	One to One, One to many, Many to many relationships
	Based on parent child relationship	A record can have many parents as well as many children.	Based on relational data structures

	Record types are organized in the form of a rooted tree	Model multiple branches emanating from one or more nodes.	A relation which is a two-dimensional table.
Data manipulation	Does not provide an independent stand-alone query interface	CODASYL (Conference on Data Systems Languages)	Relational databases are what brings many sources into a common query (such as SQL)
	Retrieve algorithms are complex and asymmetric	Retrieve algorithms are complex and symmetric	Retrieve algorithms are simple and symmetric
Data integrity	Cannot insert the information of a child who does not have any parent.	Does not suffer from any insertion anomaly.	Does not suffer from any insert anomaly.
	Multiple occurrences of child records which lead to problems of inconsistency during the update operation	Free from update anomalies.	Free form update anomalies
	Deletion of parent results in deletion of child records	Free from delete anomalies	Free from delete anomalies

❑ **Check Your Progress – 5 :**

1. Compare Hierarchical, network and relational model

   .....

   .....

2. A _____ in a table represents a relationship among a set of values.

   (A) Column (B) Key (C) Row (D) Entry

3. The term _____ is used to refer to a row.

   (A) Attribute (B) Tuple (C) Field (D) Instance

4. For each attribute of a relation, there is a set of permitted values, called the _____ of that attribute.

   (A) Domain (B) Relation (C) Set (D) Schema

5. Database _____ which is the logical design of the database, and the database _____ which is a snapshot of the data in the database at a given instant in time.

   (A) Instance, Schema (B) Relation, Schema

   (C) Relation, Domain (D) Schema, Instance

6. A relational database allows you to easily obtain specific information.

   (A) Relational (B) Hierarchical (C) Temporal (D) None

4.7 Let Us Sum Up :

Dr. Edgar F. Codd's relational model database systems did some extensive research and according to him, a true relational database is a database that must be followed in order to come up with your own twelve rules

- Relations can be represented as two–dimensional data tables with rows and columns
- The rows of a relation are called tuples.
- The columns of a relation are called attributes.
- The attributes draw values from a domain (a legal pool of values).
- The number of tuples in a relation is called its cardinality while the number of attributes in a relation is called its degree
- A relation also consists of a schema and an instance
- Schema defines the structure of a relation which consists of a fixed set of attribute domain pairs.
- An instance of a relation is a time–varying set of tuples where each tuple consists of attribute–value pairs.
- Keys are fundamental to the concept of relational databases because they enable tables in the database to be related with each other.
- Candidate key is an attribute or set of attributes that uniquely identifies a row.
- The Candidate key that you choose to identify each row uniquely is called the Primary Key.
- When a primary key of one table appears as an attribute in another table, it is called the foreign key in the second table. A foreign key is used to relate two table.
- Integrity constraints are necessary conditions to be satisfied by the data values in the relational instances so that the set of data values constitute a meaningful database.
- Data Integrity falls into the following categories : Domain integrity,Entity integrity and Referential integrity
- Domain integrity refers to the range of valid entries for a given column. It ensures that, there are only valid entries in the column
- Entity integrity ensures that each row can be uniquely identified by an attribute called the Primary key. The Primary key cannot have a NULL value.
- Referential integrity ensures that for every value of a foreign key, there is a matching value of the Primary key.
- Most of these problems, two redundant data and discrepancies are the result of bad design. Unnecessary data is unnecessary recurring data. Discrepancies due to irregular or inconsistent storage to undermine the integrity of your data to any event.
- When we move with the data models such as hierarchical model, network model, relational model we can identify number of difference in terms of data structures, Data manipulation and Data integrity.

4.8 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 4.2

- **Check Your Progress 2 :**

  See Section 4.3

- **Check Your Progress 3 :**

  See Section 4.4

- **Check Your Progress 4 :**

  See Section 4.5

- **Check Your Progress 5 :**

  1 : See Section 4.6

  2 : Row **3 :** Tuple

  4 : Domain

  5 : Schema, Instance, Answer 6: Relational

4.9 Glossary :

1. **Relations** – It can be represented as two–dimensional data tables with rows and columns
2. **Tuples** – The rows of a relation.
3. **Attributes** – The columns of a relation.
4. **Schema** – It defines the structure of a relation which consists of a fixed set of attribute domain pairs.
5. **Instance of a relation** – It is a time–varying set of tuples where each tuple consists of attribute–value pairs.
6. **Candidate key** – It is an attribute or set of attributes that uniquely identifies a row.
7. **Primary Key** – The Candidate key that you choose to identify each row uniquely.
8. **Foreign key** – When a primary key of one table appears as an attribute in another table, it is called Foreign key the in the second table. A foreign key is used to relate two tables.
9. **Integrity constraints** – these are necessary conditions to be satisfied by the data values in the relational instances so that the set of data values constitute a meaningful database.
10. **Domain integrity** – It refers to the range of valid entries for a given column. It ensures that there are only valid entries in the column
11. **Entity integrity** – It ensures that each row can be uniquely identified by an attribute called the Primary key. The Primary key cannot have a NULL value.
12. **Referential integrity** – It ensures that for every value of a foreign key, there is a matching value of the Primary key.

4.10 Assignment :

Compare all three (Hierarchical, relational and Network) models of Database with Suitable example.

4.11 Activities :

Study and write Codd's 12 rules in detail.

4.12 Case Study :

Discuss the problems with un–normalized data with proper example and explain need of normalization.

4.13 Further Reading :

1. Database Management Systems – Rajesh Narang – PHI Learning Pvt. Ltd.
2. Database System Concepts by Silberschatz, Korth – Tata McGraw–Hill Publication
3. An Introduction to Database Systems – Bipin Desai – Galgotia Publication
4. Database Management System by Raghu Ramkrishnan – Tata McGraw–Hill Publication

Unit
05

DATABASE DESIGN

UNIT STRUCTURE

5.0 Learning Objectives

5.1 Introduction

5.2 Database Development Life Cycle

5.3 Logical Design

5.4 Physical Model

5.5 Capacity Planning

5.6 Advantages and Disadvantages of Normalization

5.7 Let Us Sum Up

5.8 Suggested Answer for Check Your Progress

5.9 Glossary

5.10 Assignment

5.11 Activities

5.12 Case Study

5.13 Further Readings

5.0 Learning Objectives :

After learning this unit, you will be able to :

- Understand the requirements for designing database
- Understand how the designing of database is done.
- Method of representing the logical model through an Entity–Relationship (E–R) Diagram
- Understand the use of various keys in Database Design
- Understand how Normalization technique is used to group attributes
- Understand the use of DDL script
- Understand the concept De–normalization and Striping

5.1 Introduction :

Database design is made up of two main phases : logical and physical database design.

Logical database design is the process of constructing a model of the data used in a company based on a specific data model, but independent of a particular DBMS and other physical considerations.

In the **logical database design** phase we build the logical representation of the database, which includes identification of the important entities and relationships, and then translate this representation to a set of tables. The logical data model is a source of information for the physical design phase, providing the physical database designer with a vehicle for making tradeoffs that are very important to the design of an efficient database.

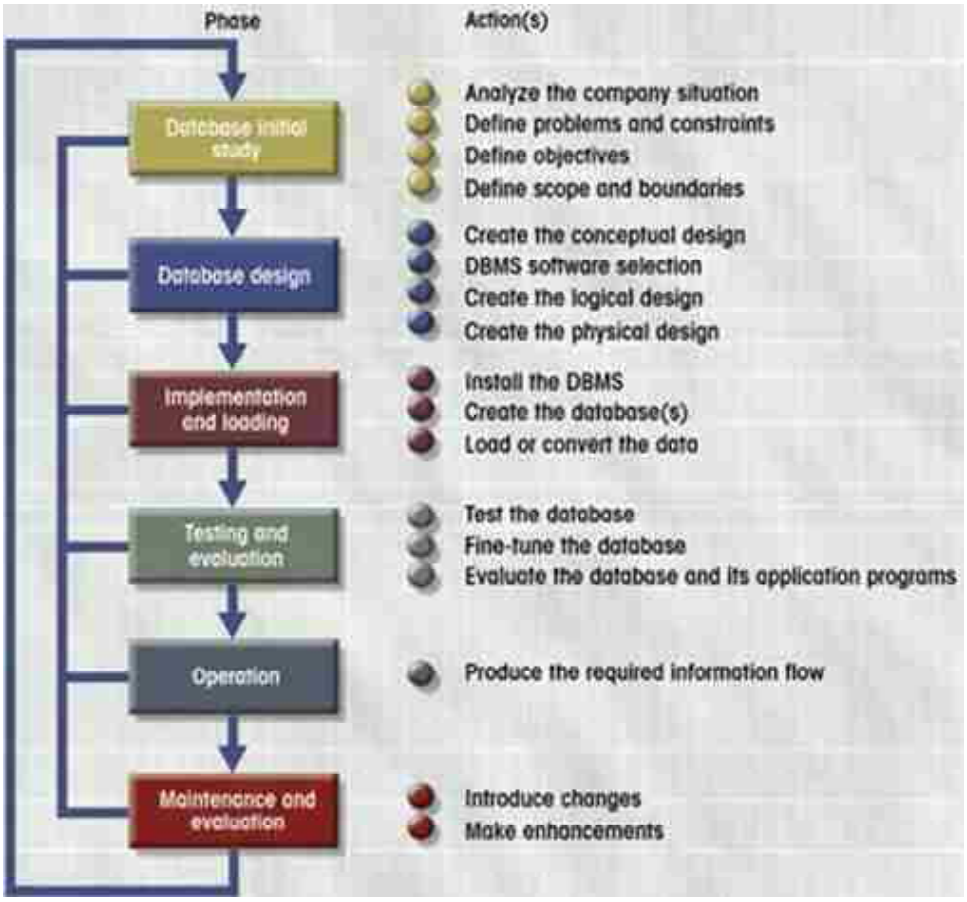
Physical database design is the process of producing a description of the implementation of the database on secondary storage; it describes the base tables, file organizations, and indexes used to achieve efficient access to the data, and any associated integrity constraints and security restrictions. In the physical database design phase we decide how the logical design is to be physically implemented in the target relational DBMS. This phase allows the designer to make decisions on how the database is to be implemented. Therefore, physical design is tailored to a specific DBMS.

5.2 Database Development Life Cycle :

A database system is a fundamental component of the larger enterprise information system. The database development life cycle (DDLC) is a process of designing, implementing and maintaining a database system to meet strategic or operational information needs of an organisation or enterprise such as :

- Improved customer support and customer satisfaction
- Better production management.
- Better inventory management.
- More accurate sales forecasting.

The database development life cycle (DDLC) is inherently associated with the software development life cycle (SDLC) of the information system. DDLC goes hand–in–hand with the SDLC and database development activities starts right at the requirement phase. The Database Life Cycle (DBLC) contains six phases, as shown in the following Figure: database initial study, database design, implementation and loading, testing and evaluation, operation, and maintenance and evolution.



5.3 Logical Design :

The logical database design phase of the methodology is divided into two main steps.

- In **Step 1** we create a data model and check that the data model has minimal redundancy and is capable of supporting user transactions. The output of this step is the creation of a logical data model, which is a complete and accurate representation of the company (or part of the company) that is to be supported by the database.

  The main tasks associated with Step 1 of logical database design are as follows :

  Step 1.0 Create and check ER model

  Step 1.1 Identify entities

  Step 1.2 Identify relationships

  Step 1.3 Identify and associate attributes with entities or relationships

  Step 1.4 Determine attribute domains

  Step 1.5 Determine candidate, primary, and alternate key attributes

  Step 1.6 Specialize/Generalize entities (optional step)

  Step 1.7 Check model for redundancy

  Step 1.8 Check model supports user transactions

  Step 1.9 Review model with users

- In **Step 2** we map the ER model to a set of tables. The structure of each table is checked using normalization. Normalization is an effective means of y consistent, logical, with minimal redundancy. The tables are also checked to ensure that they are capable of supporting the required transactions. The required integrity constraints on the database are also defined.

  The main purpose and tasks of Step 2 of the logical database design are as follows :

  To create tables for the logical data model and to check the structure of the tables.

  The tasks involved in Step 2 are :

  Step 2.1 Create tables

  Step 2.2 Check table structures using normalization

  Step 2.3 Check tables support user transactions

  Step 2.4 Check business rules

  Step 2.5 Review logical database design with users

❑ **Check Your Progress – 1 :**

1. What do you mean by logical design of database ?

   .....

   .....

   .....

   .....

5.4 The Physical design :

Physical database design is divided into six main steps :

- **Step 1** involves the desigsn of the base tables and integrity constraints using the available functionality of the target DBMS.

  You will need to know :

  - How to create base tables;
  - Whether the system supports the definition of primary keys, foreign keys, and alternate keys;
  - Whether the system supports the definition of required data (that is, whether the system allows columns to be defined as NOT NULL);
  - Whether the system supports the definition of domains;
  - Whether the system supports relational integrity rules;
  - Whether the system supports the definition of business rules.

- **Step 2** involves choosing the file organizations and indexes for the base tables. Typically, DBMSs provide a number of alternative file organizations for data, with the exception of PC DBMSs, which tend to have a fixed storage structure.

  You can't make meaningful physical design decisions until you understand in detail the transactions that have to be supported. In analyzing the transactions, you're attempting to identify performance criteria, such as:

  - the transactions that run frequently and will have a significant impact on performance;
  - the transactions that are critical to the operation of the business;
  - The times of the day/week when there will be a high demand made on the database (called the peak load).

You'll use this information to identify the parts of the database that may cause performance problems. At the same time, you need to identify the high–level functionality of the transactions, such as the columns that are updated in an update transaction or the columns that are retrieved in a query. You'll use this information to select appropriate file organizations and indexes.

When would you not add any indexes to a table ?

(1) Do not index small tables. It may be more efficient to search the table in memory than to store an additional index structure.

(2) Avoid indexing a column or table that is frequently updated.

(3) Avoid indexing a column if the query will retrieve a significant proportion (for example, 25%) of the records in the table, even if the table is large. In this case, it may be more efficient to search the entire table than to search using an index.

(4) Avoid indexing columns that consist of long character strings.

Some of the main reasons for selecting a column as a potential candidate for indexing as follows :

(1) In general, index the primary key of a table if it's not a key of the file organization. Although the SQL standard provides a clause for the

specification of primary, but note that this does not guarantee that the primary key will be indexed in some RDBMSs.

(2) Add a secondary index to any column that is heavily used for data retrieval.

(3) Add a secondary index to a foreign key if there is frequent access based on it.

(4) Add a secondary index on columns that are frequently involved in:

(a) selection or join criteria;

(b) ORDER BY;

(c) GROUP BY;

(d) Other operations involving sorting (such as UNION or DISTINCT).

(5) Add a secondary index on columns involved in built–in functions, along with any columns used to aggregate the built–in functions.

- **Step 3** involves the design of the user views originally identified in the requirements analysis and collection stage of the database system development lifecycle.
- **Step 4** involves designing the security measures to protect the data from unauthorized access.

System security covers access and use of the database at the system level, such as a username and password.

Data security covers access and use of database objects (such as tables and views) and the actions that users can have on the objects.

Access control facilities of SQL.

- Each database user is assigned an **authorization identifier** by the Database Administrator (DBA); usually, the identifier has an associated password, for obvious security reasons. Every SQL statement that is executed by the DBMS is performed on behalf of a specific user. The authorization identifier is used to determine which database objects that user may reference, and what operations may be performed on those objects. Each object that is created in SQL has an owner, who is identified by the authorization identifier. By default, the owner is the only person who may know of the existence of the object and perform any operations on the object.
- **Privileges** are the actions that a user is permitted to carry out on a given base table or view. For example, SELECT is the privilege to retrieve data from a table and UPDATE is the privilege to modify records of a table. When a user creates a table using the SQL CREATE TABLE statement, he or she automatically becomes the owner of the table and receives full privileges for the table. Other users initially have no privileges on the newly created table. To give them access to the table, the owner must explicitly grant them the necessary privileges using the SQL GRANT statement. A WITH GRANT OPTION clause can be specified with the GRANT statement to allow the receiving user(s) to pass the privilege(s) on to other users. Privileges can be revoked using the SQL REVOKE statement.

➢ When a user creates a view with the CREATE VIEW statement, he or she automatically becomes the owner of the view, but does not necessarily receive full privileges on the view. To create the view, a user must have SELECT privilege to all the tables that make up the view. However, the owner will only get other privileges if he or she holds those privileges for every table in the view.

- **Step 5** considers relaxing the normalization constraints imposed on the tables to improve the overall performance of the system. This is a step that you should undertake only if necessary, because of the inherent problems involved in introducing redundancy while still maintaining consistency.

  Formally, the term **denormalisation** refers to a change to the structure of a base table, such that the new table is in a lower normal form than the original table. However, we also use the term more loosely to refer to situations where we combine two tables into one new table, where the new table is in the same normal form but contains more nulls than the original tables.

- **Step 6** is an ongoing process of monitoring and tuning the operational system to identify and resolve any performance problems resulting from the design and to implement new or changing requirements.

❑ **Check Your Progress – 2 :**

1. What is physical design of database ?

.....

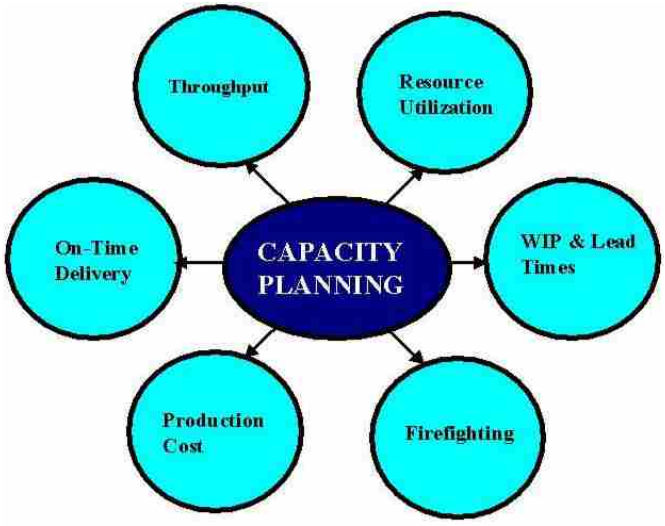
.....

.....

.....

.....

5.5 Capacity Planning :



Possible improvements from CP

Capacity planning is important in ensuring that adequate storage is available for future growth. The DDL scripts for each database object are invaluable in determining the overall storage required by the database. In fact, the process of capacity planning actually begins with the DDL scripts.

It starts with defining the column attributes. These column attributes, in turn, determine the size of each row in the table. The column attributes also determine the size of each row in indexes created on the columns. These attribute, combined with the estimated total number of rows (including provisions for future growth) are used in defining the storage clause for tables and indexes.

Most of these problems, two redundant data and discrepancies are the result of bad design. Unnecessary data is unnecessary recurring data. Discrepancies due to irregular or inconsistent storage to undermine the integrity of your data to any event.

The total size of the database by simply adding the size of the data files can be determined. Sometimes it is accurate and complete estimates, however, assume that. In addition, a number of additional considerations are to be made.

Ability to plan, design must accommodate the anticipated growth. A general rule of thumb, at least 25 percent (preferably 50 percent) after each disk should be free as early establishment. Tables are expected to be larger than the additional data will allow files to be made where necessary.It is common to experience unprecedented growth, and to reduce the requirements for temporary and rollback segments, or after completion of the initial design to identify the need for additional indexes and is more common.

The initial hardware configuration and data file layout should accommodate these possibilities. Capacity limitations can be crippling in the event that additional rollback segments or temporary segments cannot be created when needed.

For this reason, the importance of capacity planning should not be underestimated

❑ **Check Your Progress – 3 :**

1. Why capacity planning is important ?

.....

.....

.....

.....

.....

5.6 Advantages and Disadvantages of Normalization :

Normalisation improves storage capacity, in order to improve data integrity and scalability of the table is unnecessary data cleaning process. The correction tables joining query – time generalized increase in complexity and performance loss is balanced.

- There are two goals of the normalization process:
- eliminating redundant data (for example, storing the same data in more than one table)
- Ensuring data dependencies that exist i.e. only storing related data in a table.
- Both of these objectives will help in reducing the amount of space a database consumes and ensuring that data is logically stored.

❖ **Need of Normalization :**

Normalization is a well–designed relational database management system (RDBMS) is aimed. The data is put in its simplest forms, step by step set of rules by which we need normalization in the relational database data for the following reasons:

- In order to minimize data redundancy.
- It is possible to add new data values and rows without reorganizing the database structure.
- The Data should be consistent throughout the database it means it should not suffer from following anomalies.
- Insert Anomaly – Due to the lack of data i.e., all the data available for insertion such that null values in keys should be avoided. This kind of anomaly can seriously damage a database
- Update Anomaly – It is due to data redundancy i.e. multiple occurrences of same values in a column. This can lead to inefficiency.
- Deletion Anomaly – It leads to loss of data for rows that are not stored elsewhere. It could result in loss of vital data.
- Complex queries required by the user should be easy to handle.
- On decomposition of a relation into smaller relations with fewer attributes on normalization the resulting relations whenever joined must result in the same relation without any extra rows. The join operations can be performed in any order. This is known as Lossless Join decomposition.

The resulting relations (tables) should possess qualities such as the normalization of each line received, resulting relations (tables), a unique key, no repeating groups must be identified by identical columns, each column is assigned a unique name, etc.

They are six normal forms as follows :

- First Normal Form
- Second Normal Form
- Third Normal Form
- Boyce–Codd Normal Form
- Fourth Normal Form
- Fifth Normal Form
- Sixth or Domain–key Normal form

How to identify whether table is in normalized form or not ?

Let's take an example to understand this,

Suppose we want to create a database which stores friends name and their three favorite artists. The database would be quite a simple so initially there is only one table called friends table. Here FID is the primary key.

FID	FNAME	FavoriteArtist
1	Srihari	Akon, The Corrs, Robbie Williams.
2	Arvind	Engima, Chicane, Shania Twain

This table is not in normal form as Favorite Artist column is not atomic or having more than one value.

So we can modify this table

FID	FNAME	FavoriteArtist1	FavoriteArtist2	FavoriteArtist3
1	Srihari	Akon	The Corrs	Robbie Williams.
2	Arvind	Engima	Chicane	Shania Twain

Still this table is also not in normal form as we are having multiple columns with same kind of value i.e. repeating group of data or repeating columns.

To bring it in First Normal Form, we will perform the following steps:

- We'll first break single table into two.
- Each table should have information about only one entity. So we will store friend's information in one table and his favorite artists' information in another

FID	FNAME
1	Srihari
2	Arvind

FID	FavoriteArtist
1	Akon
1	The Corrs
1	Robbie Williams.
2	Engima
2	Chicane
2	Shania Twain

FID foreign key in Favorite Artist table which refers to FID in our Friends Table.

Now we can say that our table is in first normal form.

So First Normal Form :

- Column values should be atomic, scalar or should be holding single value
- No repetition of information or values in multiple columns.

So what does Second Normal Form means ?

For second normal form our database should already be in first normal form and every non–key column must depend on entire primary key.

Here we can say that our Friend database was already in second normal form because we don't have composite primary key in our friends and favorite artists table.

Composite primary keys are– primary keys made up of more than one column.

But there is no such thing in our database.

But still let's try to understand second normal form with another example This is our new table

Gadgets	Supplier	Cost	Supplier Address
Headphone	Abaci	123$	New York
Mp3 Player	Sagas	250$	California
Headphone	Mayas	100$	London

In above table ITEM+SUPPLIER together form a composite primary key. Let's check for dependency. If we know gadget, still we cannot find the cost as same gadget is provided my different supplier at different rate.

If we know supplier, then still we unable to find cost because same supplier can provide me with different gadgets.

If we know both gadget and supplier, then we can find cost. So cost is fully dependent (functionally dependent) on our composite primary key (Gadgets+Supplier)

Let's start with another non–key column Supplier Address.

If we know gadget, then from that we cannot find supplier address.

If we know who the supplier, then we can find address.

So here supplier is not completely dependent on (partial dependent) on our composite primary key (Gadgets+Supplier).

This table is surely not in Second Normal Form.

To bring it in second normal form, we'll break the table in two.

Gadgets	Supplier	Cost
Headphone	Abaci	123$
Mp3 Player	Sagas	250$
Headphone	Mayas	100$

Supplier	Supplier Address
Abaci	New York
Sagas	California
Mayas	London

We know how to normalize till second normal form.

But let's take a break over here and learn some definitions and terms.

Composite Key :– Composite key is a primary key composed of multiple columns.

Functional Dependency – When value of one column is dependent on another column.

So that if value of one column changes the value of other column changes as well.

E.g. Supplier Address is functionally dependent on supplier name. If supplier's name is changed in a record we need to change the supplier address as well.

Supplier–>SupplierAddress

"In our s table supplier address column is functionally dependent on the supplier column"

Partial Functional Dependency – A non – key columns in a composite primary key is dependent on some of the columns.

In our above example Supplier Address was partially dependent on our composite key columns (Gadgets+Supplier).

Transitive Dependency – In a non – key column is a transitive dependency and non – key column value is determined by the value, which is a type of functional dependency

With these definitions let's move to Third Normal Form.

For a table in third normal form

- It should already be in Second Normal Form.
- There should be no transitive dependency, i.e. we shouldn't have any non–key column depending on any other non–key column.

Album	Artist	No. of tracks	Country
Come on over	Shania Twain	11	Canada
History	Michael Jackson	15	USA
Up	Shania Twain	11	Canada
MCMXC A.D.	Enigma	8	Spain
The cross of changes	Enigma	8	Spain

Again we need to make sure that the non–key columns depend upon the primary key and not on any other non–key column.

Although the above table looks fine but still there is something in it because of which we will normalize it further.

Album is the primary key of the above table.

Artist and No. of tracks are functionally dependent on the Album (primary key).

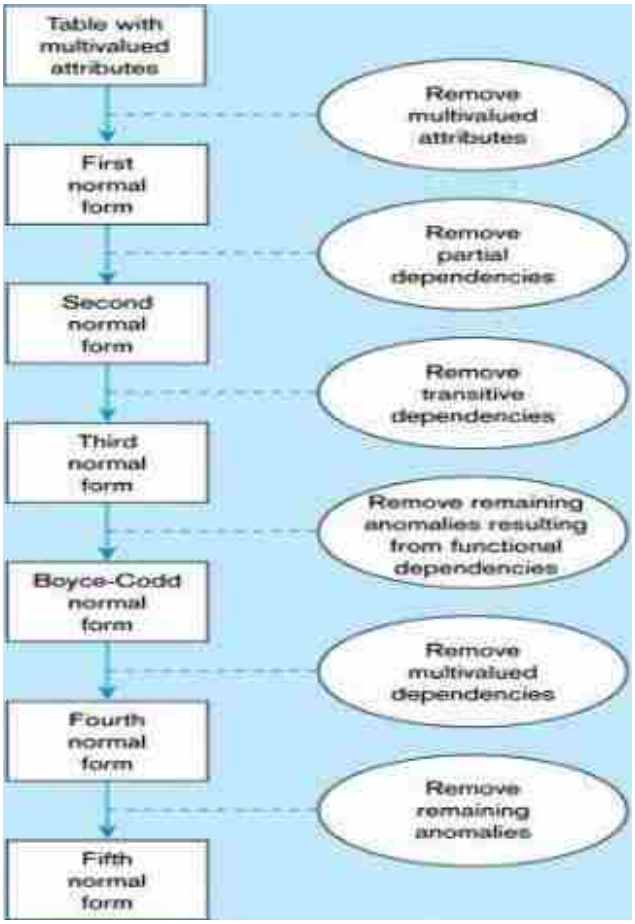
In the above table Country value is getting repeated because of artist.

So in our above table Country column is depended on Artist column which is a non–key column.

So we will move that information in another table and could save table from redundancy i.e. repeating values of Country column.

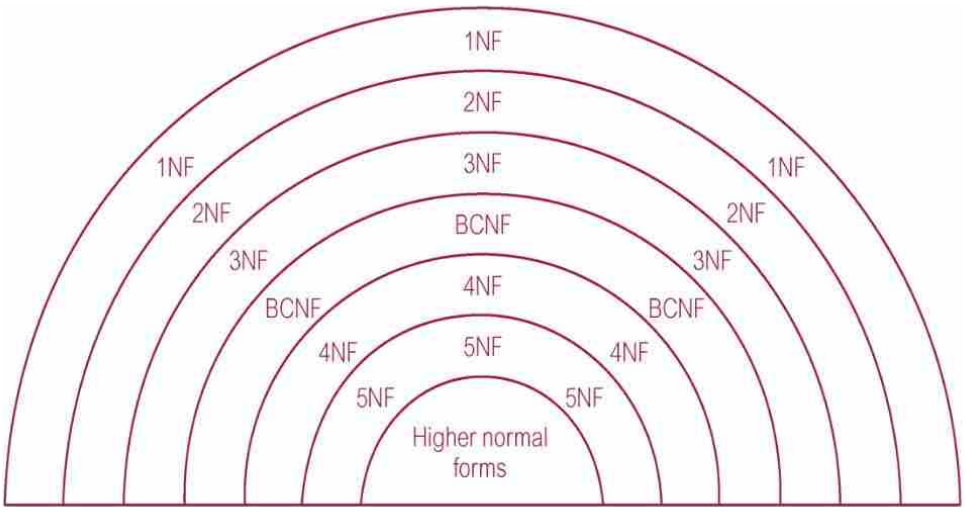
Album	Artist	No. of tracks
Come on over	Shania Twain	11
History	Michael Jackson	15
Up	Shania Twain	11
MCMXC A.D.	Enigma	8
The cross of changes	Enigma	8

Artist	Country
Shania Twain	Canada
Michael Jackson	USA
Shania Twain	Canada
Enigma	Spain
Enigma	Spain



Steps in Normalization

Relationship between Normal Forms



Let us summarize basic three forms :

First Normal Form :

This is defined in the definition of relations (tables) itself. This rule defines that all the attributes in a relation must have atomic domains. Values in atomic domain are indivisible units.

A relation is in first normal form if it meets the definition of a relation:

1. Each attribute (column) value must be a single value only.
2. All values for a given attribute (column) must be of the same type.
3. Each attribute (column) name must be unique.
4. The order of attributes (columns) is insignificant
5. No two tuples (rows) in a relation can be identical.
6. The order of the tuples (rows) is insignificant.

Converting from UNF to 1NF :

- Select attribute(s) to act as the key.
- Identify the repeating group(s) in the unnormalised table which repeats for the key attribute(s).
- Remove the repeating group by Entering data into empty columns of rows which contain the repeating data or by placing the repeating data along with a copy of the original key attribute(s) into a separate relation.

Second Normal Form (2NF) :

It is based on the concept of full functional dependency

- A relation is in second normal form (2NF) if all of its non–key attributes are dependent on all of the key.
- Another way to say this: A relation is in second normal form if it is free from partial–key dependencies.
- Relations that have a single attribute for a key are automatically in 2NF.

A 2NF relation is in 1NF and every non–primary–key attribute is fully functionally dependent on the primary key.

Converting from 1NF to 2NF :

- Identify the primary key for the 1NF relation.
- Identify the functional dependencies in the relation.
- If partial dependencies exist on the primary key remove them by placing then in a new relation along with a copy of their determinant.

Third Normal Form :

It is based on the concept of transitive dependency. A relation that is in 1NF and 2NF and in which no non–primary–key attribute is transitively dependent on the primary key.

For a relation to be in Third Normal Form, it must be in Second Normal form and the following must satisfy:

- No non–prime attribute is transitively dependent on prime key attribute
- For any non–trivial functional dependency, X ? A, then either
- X is a superkey or,
- A is prime attribute.

Converting from 2NF to 3NF :

- Identify the primary key in the 2NF relation.
- Identify functional dependencies in the relation.
- If transitive dependencies exist on the primary key remove them by placing them in a new relation along with a copy of their dominant.

Advantages of Normalization

The advantages of the normalization have been discussed below.

- Here the data structure is more efficient.
- It also avoids redundant fields or columns.
- We are able to add new rows and data values easily
- It helps in Better understanding of data.
- It ensures that distinct tables exist when necessary.
- It is very easy to perform operations and complex queries can be easily handled.
- It even minimizes duplication of data.
- Close modeling of real world entities, processes and their relationships.

Disadvantages of Normalization

The disadvantages of the normalization have been discussed below.

1. One cannot start building the database before he know what the user needs.
2. In the process of normalizing the relations to higher normally forms that simply means. 4NF, 5NF the performance degrades.
3. This is a very time taking and difficult process in normalizing relations of higher degree.
4. Careless decomposition may leads to bad design of database which may leads to serious problems.

❑ **Check Your Progress – 4 :**

1. What is normalization ? What is need of normalization ?

   .....

   .....

2. _____ can help us detect poor E–R design.

   (A) Database Design Process (B) E–R Design Process

   (C) Relational scheme (D) Functional dependencies

3. If a multivalued dependency holds and is not implied by the corresponding functional dependency, it usually arises from one of the following sources.

   (A) A many–to–many relationship set

   (B) A multivalued attribute of an entity set

   (C) A one–to–many relationship set

   (D) Both A many–to–many relationship set and A multivalued attribute of an entity set

4. Which of the following has each related entity set has its own schema and there is an additional schema for the relationship set.

   (A) A many–to–many relationship set

   (B) A multivalued attribute of an entity set

   (C) A one–to–many relationship set

   (D) All of the mentioned

5. In which of the following, a separate schema is created consisting of that attribute and the primary key of the entity set.

   (A) A many–to–many relationship set

   (B) A multivalued attribute of an entity set

   (C) A one–to–many relationship set

   (D) All of the mentioned

6. A function that has no partial functional dependencies is in _____ form :

   (A) 3NF (B) 2NF (C) 4NF (D) BCNF

5.7 Let Us Sum Up :

Logical database design stage but how information is shared with the logical, data will be stored physically does not deal with how. Requirements in terms of business entities and relationships to represent the data, provides a level of abstraction from the physical database that are translated into a model, rather than in terms of tables and columns.

Physical design characteristics and individual data elements are defined as columns in the table, which is HASE. At this stage the performance index, rollback segments, the idea of the creation of temporary segments, and is concerned with the physical layout of data files on disk. DDL (Data Definition Language) scripts to create database objects to be used for capacity planning and is written.

Sufficient storage capacity planning is available for future development is important in ensuring that. DDL script for each database object databases are invaluable in determining the required total storage. In fact, the capacity planning process begins with the DDL scripts

5.8 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 5.2

- **Check Your Progress 2 :**

  See Section 5.3

- **Check Your Progress 3 :**

  See Section 5.4

- **Check Your Progress 4 :**

  See Section 5.5

- **Check Your Progress 5 :**

  See Section 5.6

❑ **Check Your Progress 6 :**

1 : See Section 5.7

2 : Functional dependencies

3 : Both A many–to–many relationship set and A multivalued attribute of an entity set

4 : A many–to–many relationship set

5 : A multivalued attribute of an entity set

6 : 2NF

5.9 Glossary :

1. **Prime attribute –** an attribute, which is part of prime–key, is prime attribute.
2. **Non–prime attribute –** an attribute, which is not a part of prime–key, is said to be a non–prime attribute.
3. **Normalization –** It is a method to remove all these anomalies and bring database to consistent state and free from any kinds of anomalies.
4. **First Normal Form (1NF) –** This rule defines that all the attributes in a relation must have atomic domains.
5. **Second Normal Form (2NF) –** Second normal form says, that every non–prime attribute should be fully functionally dependent on prime key attribute.

   That is, if X ? A holds, then there should not be any proper subset Y of X, for that Y ? A also holds.
6. **Third Normal Form (3NF) –** For a relation to be in Third Normal Form, it must be in Second Normal form and the following must satisfy: No non–prime attribute is transitively dependent on prime key attribute, For any non–trivial functional dependency, X ? A, then either X is a superkey or, A is prime attribute.

5.10 Assignment :

State the conceptual difference between simple bifurcation and Normalization of DBMS view.

5.11 Activities :

Study the BCNF, 4NF and 5NF normal forms in detail.

5.12 Case Study :

A businessman has to start his own Internet business, called Mastermind pieces Ltd, hiring paintings to private individuals and commercial companies.

Because of its reputation as a database designer to support new business for the design and implementation of a database is called upon their services.

Initial planning meeting, the design, and the user has requested the requirements discussed.

System, customers should be able to manage the details of the images on the pictures to existing customers rent. Client B (Bronze), S (Silver), G (gold) or P (platinum) is classified as. Respectively in these categories : 0%, 5%, 10% or 15% discount entitle a customer.

Customers often have a particular artist or theme (such as animal, landscape, seascape, navy, still life, etc.) to request images.

Over time a customer may hire the same painting more than once.

Each image is defined by the owner of a customer is allocated monthly rental price. The owner of the painting so that the customer has paid 10% of the rental price. Hired within six months are not any pictures that are returned to the owner. However, three months later, the owner returned the painting can resubmit.

Each painting can only have one artist associated with it.

Several reports are required from the system. Three main ones are:

1. For each client, they are hired or a report showing an overview of all the pictures are currently recruiting
2. All pictures for each artist to hire a report submitted
3. For each artist a return to the past six months rent pictures Reports

What are you supposed to take each in turn report and the third is to make a set of relationships in general. At each step in the normalization process you must show the relationship

5.13 Further Reading :

1. Database Management Systems – Rajesh Narang – PHI Learning Pvt. Ltd.
2. Database System Concepts by Silberschatz, Korth – Tata McGraw–Hill Publication
3. An Introduction to Database Systems – Bipin Desai – Galgotia Publication
4. Database Management System by Raghu Ramkrishnan – Tata McGraw–Hill Publication
5. Database Solutions (2nd Edition) By Thomas M Connolly & Carolyn E. Begg

Unit
06

NORMALIZATION

UNIT STRUCTURE

6.0 Learning Objectives

6.1 Introduction

6.2 What is Normalization ?

6.3 Database Normal Forms and Example

6.4 1NF (First Normal Form)

6.5 2NF (Second Normal Form)

6.6 3NF (Third Normal Form)

6.7 BCNF (Boyce–Codd Normal Form)

6.8 4NF (Fourth Normal Form)

6.9 5NF & 6NF (Fifth & Sixth Normal Form)

6.10 Let Us Sum Up

6.11 Suggested Answer for Check Your Progress

6.12 Glossary

6.13 Assignment

6.14 Activities

6.15 Case Study

6.16 Further Readings

6.0 Learning Objectives :

After learning this unit, you will be able to :

- Understand the requirements for designing database
- Understand how the designing of database is done.
- Normal forms with example

6.1 Introduction :

Database normalization is the way toward getting sorted out information into tables in such a way that the results of using the database are always unambiguous and as intended. Such normalization is intrinsic to relational database theory. It may have the effect of duplicating data within the database and often results in the creation of additional tables. In this unit learner will get insight about Normalization and its various forms.

6.2 What is Normalization ?

Normalization is a database design technique that reduces data redundancy and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies. Normalization rules divides larger tables into smaller tables and links them using relationships. The purpose of Normalization in SQL is to eliminate redundant (repetitive) data and ensure data is stored logically.

The inventor of the relational model Edgar Codd proposed the theory of normalization of data with the introduction of the First Normal Form, and he continued to extend theory with Second and Third Normal Form. Later he joined Raymond F. Boyce to develop the theory of Boyce–Codd Normal Form.

6.3 Database Normal Forms and Example :

Here is a list of Normal Forms

- 1NF (First Normal Form)
- 2NF (Second Normal Form)
- 3NF (Third Normal Form)
- BCNF (Boyce–Codd Normal Form)
- 4NF (Fourth Normal Form)
- 5NF (Fifth Normal Form)
- 6NF (Sixth Normal Form)

The Theory of Data Normalization in SQL server is still being developed further. For example, there are discussions even on 6th Normal Form. However, in most practical applications, normalization achieves its best in 3rd Normal Form. The evolution of SQL Normalization theories is illustrated below –

Database Normal Forms

Database Normalization With Examples

Database Normalization Example can be easily understood with the help of a case study. Assume, a video library maintains a database of movies rented out. Without any normalization in database, all information is stored in one table as shown below. Let's understand Normalization in database with tables example :

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

Now for all normal form we are using this table.

Problems Without Normalization :

If a table is not properly normalized and have data redundancy then it will not only eat up extra memory space but will also make it difficult to handle and update the database, without facing data loss. Insertion, Updation and Deletion Anomalies are very frequent if database is not normalized.

6.4 1NF :

For a table to be in the First Normal Form, it should follow the following 4 rules :

- It should only have single(atomic) valued attributes/columns.
- Values stored in a column should be of the same domain
- All the columns in a table should have unique names.
- And the order in which data is stored, does not matter.

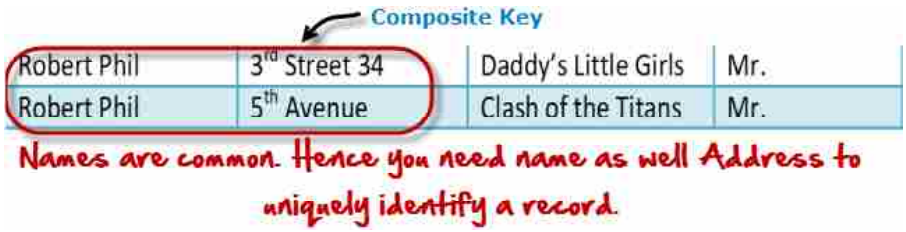
FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

Example of Table in 1NF as per rules

Before we proceed let's understand a few things:

What is Composite Key ?

A composite key is a primary key composed of multiple columns used to identify a record uniquely. In our database, we have two people with the same name Robert Phil, but they live in different places.



Hence, we require both Full Name and Address to identify a record uniquely. That is a composite key. Let's move into second normal form 2NF.

6.5 2NF :

For a table to be in the Second Normal Form,

- It should be in the First Normal form.
- And, it should not have Partial Dependency.

It is clear that we can't move forward to make our simple database in 2nd Normalization form unless we partition the table above.

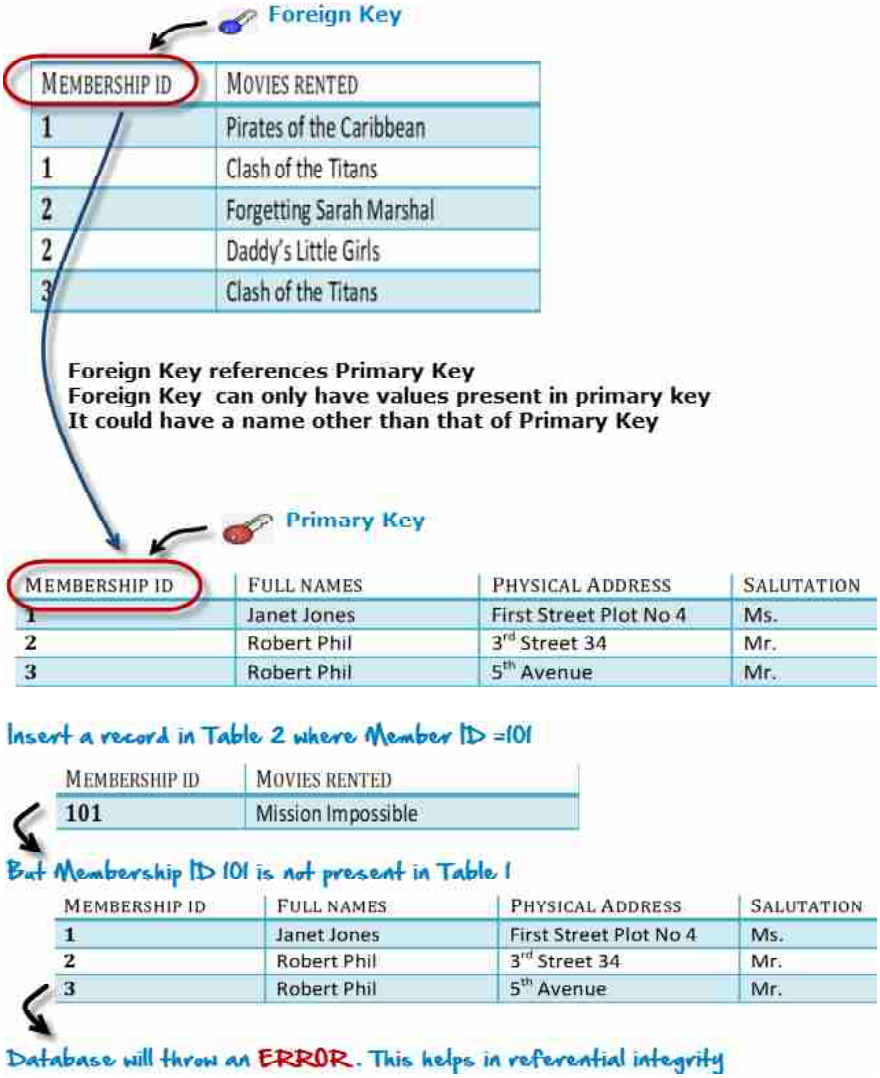
MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

Table 1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Table 2

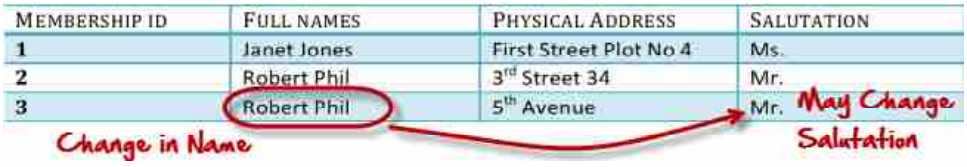
We have divided our 1NF table into two tables viz. Table 1 and Table2. Table 1 contains member information. Table 2 contains information on movies rented. We have introduced a new column called Membership_id which is the primary key for table 1. Records can be uniquely identified in Table 1 using membership id. Foreign Key references the primary key of another Table! It helps connect your Tables.



You will only be able to insert values into your foreign key that exist in the unique key in the parent table. This helps in referential integrity.

The above problem can be overcome by declaring membership id from Table2 as foreign key of membership id from Table1. Now, if somebody tries to insert a value in the membership id field that does not exist in the parent table, an error will be shown

A transitive functional dependency is when changing a non–key column, might cause any of the other non–key columns to change Consider the table 1. Changing the non–key column Full Name may change Salutation.



Let's move into 3NF.

6.6 3NF :

A table is said to be in the Third Normal Form when,

- It is in the Second Normal form.
- And, it doesn't have Transitive Dependency.

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3rd Street 34	1
3	Robert Phil	5th Avenue	1

TABLE 1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Table 2

SALUTATION ID	SALUTATION
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

Table 3

We have again divided our tables and created a new table which stores Salutations. There are no transitive functional dependencies, and hence our table is in 3NF

In Table 3 Salutation ID is primary key, and in Table 1 Salutation ID is foreign to primary key in Table 3

Now our little example is at a level that cannot further be decomposed to attain higher normal forms of normalization. In fact, it is already in higher normalization forms. Separate efforts for moving into next levels of normalizing data are normally needed in complex databases. However, we will be discussing next levels of normalizations in brief in the following. Advantage of removing Transitive Dependency.

The advantage of removing transitive dependency is, Amount of data duplication is reduced.

Data integrity achieved.

6.7 BCNF :

Boyce and Codd Normal Form is a higher version of the Third Normal form. This form deals with certain type of anomaly that is not handled by 3NF. A 3NF table which does not have multiple overlapping candidate keys is said to be in BCNF. For a table to be in BCNF, following conditions must be satisfied:

- R must be in 3rd Normal Form
- and, for each functional dependency ($X \rightarrow Y$), X should be a super Key.

The second point sounds a bit tricky, right ? In simple words, it means, that for a dependency A → B, A cannot be a non–prime attribute, if B is a prime attribute. Boyce–Codd Normal Form or BCNF is an extension to the third normal form, and is also known as 3.5 Normal Form.

6.8 4NF :

A table is said to be in the Fourth Normal Form when,

- It is in the Boyce–Codd Normal Form.
- And, it doesn't have Multi–Valued Dependency.

Fourth Normal Form comes into picture when Multi–valued Dependency occur in any relation. In this tutorial we will learn about Multi–valued Dependency, how to remove it and how to make any table satisfy the fourth normal form.

What is Multi–valued Dependency ?

A table is said to have multi–valued dependency, if the following conditions are true,

For a dependency A → B, if for a single value of A, multiple value of B exists, then the table may have multi–valued dependency. Also, a table should have at–least 3 columns for it to have a multi–valued dependency. And, for a relation R(A, B, C), if there is a multi–valued dependency between, A and B, then B and C should be independent of each other. If all these conditions are true for any relation(table), it is said to have multi–valued dependency.

6.9 5NF & 6NF :

A table is in 5th Normal Form only if it is in 4NF and it cannot be decomposed into any number of smaller tables without loss of data.

6th Normal Form is not standardized, yet however, it is being discussed by database experts for some time. Hopefully, we would have a clear & standardized definition for 6th Normal Form in the near future.

❑ Check Your Progress :

1. Demoralization is used for _____.
   (A) DB backup (B) Delete tables
   (C) Reduce redundancy (D) Performance tuning.
2. Atomicity of domains of attributes related to _____.
   (A) 1NF (B) 2NF (C) 3NF (D) BCNF
3. Redundancy in data may lead to its inconsistency.
   (A) True (B) False (C) Partially True (D) None
4. 5NF is designed to cope with _____.
   (A) Transitive dependency (B) Join dependency
   (C) Multi valued dependency (D) All of the above
5. Every Boyee–Codd normal form is in
   (A) First normal form (B) Second normal form
   (C) Third normal form (D) All of the above

6.10 Let Us Sum Up :

Lets us summaries the unit, This unit focus on the detailed understanding of Normalization process with first normal form to 6th normal form with example.

6.11 Suggested Answer for Check Your Progress :

❑ **Check Your Progress :**

1 : Performance Tuning **2 :** 1NF

3 : True **4 :** Join dependency

5 : All of above

6.12 Glossary :

1. **First Normal Form (1NF) –** This rule defines that all the attributes in a relation must have atomic domains.
2. **Second Normal Form (2NF) –** Second normal form says, that every non–prime attribute should be fully functionally dependent on prime key attribute.

   That is, if X $\rightarrow$ A holds, then there should not be any proper subset Y of X, for that Y $\rightarrow$ A also holds.
3. **Third Normal Form (3NF) –** For a relation to be in Third Normal Form, it must be in Second Normal form and the following must satisfy : No non–prime attribute is transitively dependent on prime key attribute, For any non–trivial functional dependency, X $\rightarrow$ A, then either X is a superkey or, A is prime attribute.

6.13 Assignment :

State the conceptual difference between simple bifurcation and Normalization of DBMS view.

6.14 Activities :

Study the BCNF, 4NF and 5NF normal forms in detail.

6.15 Case Study :

A businessman has to start his own Internet business, called Mastermind pieces Ltd, hiring paintings to private individuals and commercial companies.

Because of its reputation as a database designer to support new business for the design and implementation of a database is called upon their services.

Initial planning meeting, the design, and the user has requested the requirements discussed.

System, customers should be able to manage the details of the images on the pictures to existing customers rent. Client B (Bronze), S (Silver), G (gold) or P (platinum) is classified as. Respectively in these categories : 0%, 5%, 10% or 15% discount entitle a customer.

Customers often have a particular artist or theme (such as animal, landscape, seascape, navy, still life, etc.) to request images.

Over time a customer may hire the same painting more than once.

Each image is defined by the owner of a customer is allocated monthly rental price. The owner of the painting so that the customer has paid 10% of the rental price. Hired within six months are not any pictures that are returned to the owner. However, three months later, the owner returned the painting can resubmit.

Each painting can only have one artist associated with it.

Several reports are required from the system. Three main ones are :

1. For each client, they are hired or a report showing an overview of all the pictures are currently recruiting
2. All pictures for each artist to hire a report submitted
3. For each artist a return to the past six months rent pictures Reports

What are you supposed to take each in turn report and the third is to make a set of relationships in general. At each step in the normalization process you must show the relationship

6.16 Further Reading :

1. Database Management Systems – Rajesh Narang – PHI Learning Pvt. Ltd.
2. Database System Concepts by Silberschatz, Korth – Tata McGraw–Hill Publication
3. An Introduction to Database Systems – Bipin Desai – Galgotia Publication
4. Database Management System by Raghu Ramkrishnan – Tata McGraw–Hill Publication
5. Database Solutions (2nd Edition) By Thomas M Connolly & Carolyn E. Begg

BLOCK SUMMARY :

In unit 4, We discussed in a relational model real world objects are represented in tables. Each table is made out of rows and columns. It is also known as tuple or record each line, it is also known as attributes, is out of the fields. Each attribute stands for a certain feature of the real world object. An attribute is defined by a name and its value.

Tuples represent the relationship between the existing relationships between tables. In addition to the key features to be defined (usually appear underlined in a relationship). They allocation tables (relations) are required to allow accesses to the unique tables.

Integrity or consistency of data in a database system that stands for quality and reliability. The data referenced objects are displayed correctly if a database is consistent. Database vague or contradictory tuples, relations or tables there if it is inconsistent.

A relation model (scheme, entity) should reflect relationships that also logically (in the real world) belong together. Normalizations anomalies correspond to different types of databases to help avoid.

Unit 5 discussed issues of bad database design and explained the need of normalization process. It also explained logical, physical database design and need of capacity planning. The basics of normalization process is discussed.

In Unit 6, Normalization is process of efficiently organizing the database. It is accurate representation of data, relationships and Constraints. The goal of normalization is to eliminate redundant data in a database and ensure data dependencies make sense. There are guidelines for ensuring that databases are normalized called normal forms: 1NF, 2NF, 3NF, BCNF, 4NF, 5NF and 6NF.

BLOCK ASSIGNMENT :

❖ **Short Questions :**

1. Define Candidate key, primary key.
2. Define terms schema, tuple, instance, relation.
3. What is goal of normalization ?
4. Write any two advantages of normalization.
5. What is referential integrity ?

❖ **Long Questions :**

1. Explain the anomalies in un–normalized database.
2. What is normalization ? Explain basic three normal forms.
3. Compare Hierarchical, network and relational model.

❖ **Enrolment No. :** ☐

1. How many hours did you need for studying the units ?

Unit No.	4	5	6
No. of Hrs.			

2. Please give your reactions to the following items based on your reading of the block :

Items	Excellent	Very Good	Good	Poor	Give specific example if any
Presentation Quality	☐	☐	☐	☐	_____
Language and Style	☐	☐	☐	☐	_____
Illustration used (Diagram, tables etc)	☐	☐	☐	☐	_____
Conceptual Clarity	☐	☐	☐	☐	_____
Check your progress Quest	☐	☐	☐	☐	_____
Feed back to CYP Question	☐	☐	☐	☐	_____

3. Any other Comments

.....

.....

.....

.....

.....

.....

.....

.....

Dr. Babasaheb Ambedkar
Open University Ahmedabad

BCAR-202/
DCAR-202

Database Management System

BLOCK 3 : SQL AND OODBMS

UNIT 7 SQL (STRUCTURED QUERY LANGUAGE)

UNIT 8 SQL CONSTRAINTS

UNIT 9 TRANSACTION PROCESSING

UNIT 10 OBJECT ORIENTED DATABASE MANAGEMENT SYSTEMS

SQL AND OODBMS

Block Introduction :

Unit 7 will introduce query language for database. SQL (Structured Query Language) is a database computer language designed for managing data in relational database management systems RDBMS). SQL, is a standardized computer language that was originally developed by IBM for querying, altering and defining relational databases, using declarative statements. SQL has several parts like DDL, DML, and TCL etc. You will discover how to use SQL to sort and are trieve data from tables and how to use SQL to filter retrieved data. You will also learn how to gather significant statistics from data using aggregate functions, and how to extract data from multiple tables simultaneously using joins and subqueries. In addition, you'll learn how to manipulate data using the INSERT, UPDATE, and DELETE statements. So this unit will provide you SQL's basic constructs and concepts.

Unit 8 provides the understanding of types of constraints available in SQL. It explain each constraint with practical implementation aspects.

Unit 9 will provide you the concept of transaction processing. A transaction, also in simplified terms, is a specific task, or set of tasks, to be executed against the database. Transactions start with an executable DML statement and end when the statement or multiple statements are all either rolled back or committed to the database, or when a DDL (Data Definition Language) statement is issued during the transaction. You will learn different types of transactions in this unit. Also you will understand the steps involved in processing local and remote transaction. In addition, you will learn how to optimize the execution of transaction.

Unit 10 will give you overview of object oriented databases.Object oriented database technology is a blend of object oriented programming and database technology. The result of this blend is Object Oriented Database (OODB). The blending achieves integration of applications running on different platforms. OODB has more or less the same features of DBMS, but, in addition, is in a

position to handle OO features. In the real world, both the applications exist, namely, OO applications and RDBMS applications, with first generation client–server computing. So in an organisation, both OODBs and RDBs exist. The user must have access to both of them to manipulate data. The developer must therefore develop applications that could source data from all databases (OODB, relational database, etc.). Now there is a mismatch between the application objects and relational data that needs to be mapped for use in the application. The process of mapping and integrating begins with defining the relationships between the table structures in RDB and class structure in the object model in OODB.

Block Objectives :

After learning this block, you will be able to :

- Understand basic constructs and concept of SQL.
- Understand various DDL DML commands in SQL.
- Understand the concept of Transaction processing.
- Understand the need and concept of object oriented databases.
- Understand the features and advantages of OODBMS.

Block Structure :

Unit 7 : SQL (Structured Query Language)

Unit 8 : SQL Constraints

Unit 9 : Transaction Processing

Unit 10 : Object Oriented Database Management Systems (OODBMS)

Unit 07

SQL (STRUCTURED QUERY LANGUAGE)

UNIT STRUCTURE

7.0 Learning Objectives

7.1 Introduction

7.2 History

7.3 Basic Structure

7.4 DDL Commands

7.5 DML Commands

7.6 Simple Queries

7.7 Nested Queries

7.8 Aggregate Functions

7.9 Let Us Sum Up

7.10 Suggested Answer for Check Your Progress

7.11 Glossary

7.12 Assignment

7.13 Activities

7.14 Case Study

7.15 Further Readings

7.0 Learning Objectives :

After learning this unit, you will be able to understand :

- The purpose of using SQL in DBMS
- Distinguish between SQL and other programming languages
- Learn various data types in SQL
- Learn various Built in Functions in SQL.
- Lean how to write queries.
- Lean how to use CREATE statement.
- ANSI standardization for SQL

7.1 Introduction :

Dr. E. F. Codd published the paper, "A Relational Model of Data for Large Shared Data Banks", in June 1970 in the journal, Communications of the ACM. Codd's model is now accepted as the definitive model for relational database management systems (RDBMS). The language, Structured English Query Language (SEQUEL) was developed by IBM Corporation, Inc., to use Codd's model. SEQUEL later became SQL (still pronounced "sequel"). In 1979, Relational Software, Inc. (now Oracle) introduced the first commercially available implementation of SQL. Today, SQL is accepted as the standard RDBMS language.

SQL is a non–procedural language. Non–Procedural means that SQL lacks the traditional control constructs such as IF–THEN–ELSE, WHILE, FOR, and GO TO statements found in procedural languages like C, Perl, Python, and other 3GL's.

SQL is called a declarative language, meaning that you only need to specify what needs to be accomplished (such as a query or insert) and not how to do it; the DBMS determines and performs internally the step by step operation needed to obtain the result.

The sets of records can be manipulated instead of one record at a time. The syntax is free–flowing, enabling you to concentrate on the data presentation. Oracle has two optimizers (cost– and rule–based) that will parse the syntax and format it into an efficient statement before the database engine receives it for processing. The database administrator (DBA) determines which optimizer is in effect for each database instance.

7.2 History :

SQL History

The history of SQL begins in an IBM laboratory in San Jose, California, where SQL was developed in the late 1970s. The initials stand for Structured Query Language, and the language itself is often referred to as "sequel." It was originally developed for IBM's DB2 product (a relational database management system, or RDBMS, that can still be bought today for various platforms and environments). In fact, SQL makes an RDBMS possible. SQL is a nonprocedural language, in contrast to the procedural or third–generation languages (3GLs) such as COBOL and C that had been created up to that time.

Two standards organizations, the American National Standards Institute (ANSI) and the International Standards Organization (ISO), currently promote SQL standards to industry. Although these standard–making bodies prepare standards for database system designers to follow, all database products differ from the ANSI standard to some degree. In addition, most systems provide some proprietary extensions to SQL that extend the language into a true procedural language.

❑ **Check Your Progress – 1 :**

1. What is SQL ? Explain in brief History of SQL.

.....

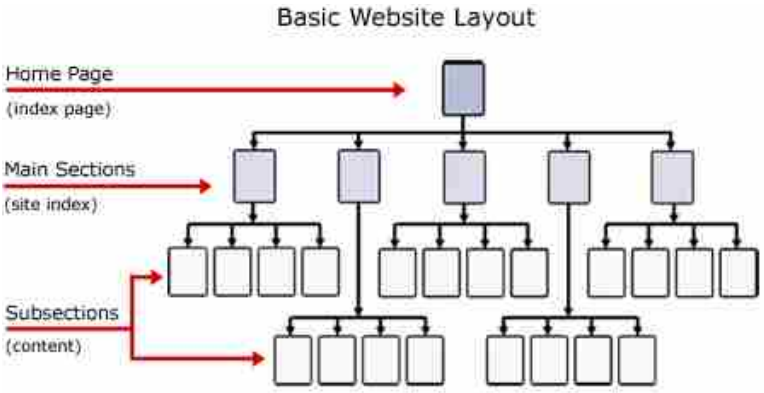
.....

.....

.....

.....

7.3 Basic Structure :



Basic Structure

SQL is based on set and relational operations with certain modifications and enhancements.A typical SQL query has the form:

Select A1, A2,..., An

From r1, r2,..., rm

Where P

–A is represent attribute

–r is represent relations

–P is a predicate or condition.

❖ **Select Clause :**

- The select clause corresponds to the projection operation of the relational algebra. It is used to list the attributes desired in the result of a query.
- The general form for a SELECT statement, retrieving all of the rows in the table is:

  Select Column Name, Column Name,...

  From Table Name;
- To get all columns of a table without typing all column names, use: Select * From Table Name;

  Note that SQL is not case sensitive. SELECT is the same as select.

  Sql> Select * From Emp;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
– – –	– – –	– – –	– – –	– – –	– – –	– – –	– – –
7369	Smith	Clerk	7902	17-Dec-80		800	20
7499	Allen	Salesman	7698	20-Feb-81	1600	300	30
7521	Ward	Salesman	7698	22-Feb-81	1250	500	30
7566	Jones	Manager	7839	02-Apr-81	2975		20
7654	Martin	Salesman	7698	28-Sep-81	1250	1400	30
7698	Blake	Manager	7839	01-May-81	2850		30
7782	Clark	Manager	7839	09-Jun-81	2450		10
7788	Scott	Analyst	7566	19-Apr-87	3000		20
7839	King	President		17-Nov-81	5000		10
7844	Turner	Salesman	7698	08-Sep-81	15000		30
7876	Adams	Clerk	7788	23-May-87	1100		20
7900	James	Clerk	7698	03-Dec-81	950		30
7902	Ford	Analyst	7566	03-Dec-81	3000		20
7934	Miller	Clerk	7782	23-Jan-82	1300		10

14 Rows Selected.

Select All The Records From Dept Table

Sql> Select * From Dept;

DEPTNO	DNAME	LOC
– – –	– – –	– – –
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

- SQL allows duplicates in relations as well as in query results. To force the elimination of duplicates, insert the keyword distinct after select.

  Example : Find the names of all employees in the emp relation, and remove duplicates.

  select distinct ename

  from emp;

- The keyword all specifies that duplicates should not be removed.

  select all ename

  from emp;

- The select clause can also contain arithmetic expressions involving the operators, +, ?, ?, and /, and operating on constants or attributes of tuples.

  The query:

  Select ename, sal*100 From emp;

Would return a relation which is the same as the emp relation, except that the attribute sal is multiplied by 100.

WHERE clause:

- The where clause corresponds to the selection predicate of the relational algebra. It consists of a predicate involving attributes of the relations that appear in the from clause.

  SYNTAX :

  Where <search condition>

  Find The Employee Name Who Is Working In Dept No 30

  Sql> Select Ename From Emp Where Deptno=30; Ename

  Allen

  Ward

  Martin

  Blake

  Turner

  James

  6 Rows Selected.

- SQL uses the logical connectives and, or, and not. It allows the use of arithmetic expressions as operands to the comparison operators. SQL includes a between comparison operator in order to simplify where clauses that specify that a value be less than or equal to some value and greater than or equal to some other value.
- Find ename Who Are Manager And Getting Salary More Than 2000

  Sql> Select ename From emp Where Job='Manager' And Sal>2000;

  Ename

  – – –

  Jones

  Blake

  Clark

- Find Those Employees Who Were Hired Between 1 Mar 1981 And 1 Jun 1983

  Sql> Select Ename From Emp Where Hiredate Between '1–Mar–1981' And '1–Jun–1983';

  Ename

  – – –

  Jones

  Martin

  Blake

  Clark

  King

  Turner

James

Ford

Miller

9 Rows Selected.

FROM Clause :

- The from clause corresponds to the Cartesian product operation of the relational algebra. It lists the relations to be scanned in the evaluation of the expression.
- Find the Cartesian product emp×dept

  Select ename, dname

  From emp, dept
- Find the name of employees working at location 'DALLAS'

  select distinct ename

  from emp, dept

  where emp.deptno = dept.deptno and loc= 'DALLAS';

Tuple variables :

Tuple variables are defined in the from clause via the use of the as clause.

- Find the emp names, deptno and their dept name

  Select T.ename, S.Deptno, S.Dname

  from emp as T, dept asS

  where T.deptno = S.deptno

  String Operations :
- SQL includes a string–matching operator for comparisons on character strings. Patterns are described using two special characters :

  – percent (%) : The % character matches any substring.

  – underscore : The character matches any character.
- Find the names of all employees whose name includes the substring 'john'.
  Select ename

  From emp

  Where ename like"%john%"

Ordering the Display of Tuples :

- List in alphabetic order the names of all employees

  Select ename

  from emp

  order by ename ;
- We may specify desc for descending order or asc for ascending order, for each attribute; ascending order is the default.
- Find Ename Who Are Working In Deptno 30 Order By Salary In Des. Order Sql> Select Ename From Emp Where Deptno=30 Order By Sal Des;

Ename

_ _ _

Blake

Allen

Turner

Ward

Martin

James

6 Rows Selected.

- SQL must perform a sort to fulfil an order by request. Since sorting a large number of tuples may be costly, it is desirable to sort only when necessary.

❑ **Check Your Progress – 2 :**

1. Explain the basic structure of SQL.

.....

.....

.....

.....

.....

7.4 DDL Commands :

Data Definition Language (DDL) statements are used to define the database structure or schema. Some basic commands are as follows:

Create	–	To Create Objects In The Database
Alter	–	Alters The Structure Of The Database
Drop	–	Delete Objects From The Database

Create Table

The PURPOSE of this command is to create a table, the basic structure to hold user data, specifying this information:

- column definitions
- integrity constraints
- the table's table space
- storage characteristics Syntax:

Create Table [Schema.]Table

({ Column Datatype [Default Expr] [Column_Constraint]...

| Table_Constraint}

[, { Column Datatype [Default Expr] [Column_Constraint]...

| Table_Constraint}]...)

Where:

- **Schema** – is the schema containing the table. If you omit schema, it is assumed that the table is in your own schema.
- **Table** – is the name of the table to be created.
- **Column** – specifies the name of a column of the table. The number of columns in a table can range from 1 to 254.
- **Datatype** – is the datatype of a column.

The data types permitted are

Data type	Syntax	Description	Example
Character data types	char(n)	It is fixed length character string of size n, default value 1 byte if n is not specified.	account_type char(6)
	varchar(n)	It is variable length character string with maximum size n.	employee_name varchar(50)
	Text	It is used to store large text data, no need to define a maximum.	work_experiene text
Numeric data types	Integer, int, serial	Serial is same as int, only that values are incremented automatically.	Eno int Eno serial
	Numeric	A real number with P digits, S of them after decimal point.	Sal numeric(5.2) Sal numeric(n)
	Float	Real number	Weight Float
Date and time type	Date	Stores date information	Birthdate date
	Time	Stores time information	Birthtime time
	Timestamp	Stores a date & time	Birth timestamp
Boolean and Binary type	Boolean, bool	Stores only 2 value : true or false, 1 or 0, yes or no, y or n, t or f	Flag Boolean

- **Default** – specifies a value to be assigned to the column if a subsequent INSERT statement omits a value for the column. The datatype of the expression must match the datatype of the column.
- **Column Constraint** – defines an integrity constraint as part of the column definition.

- **Table Constraint** – defines an integrity constraint as part of the table definition.

Constraints can be defined as either of the following :

Name	Name	Example
Column Level Table Level	When data constraint is defined only with respect to one column and hence defined after the column definition, it is called as column level constraint When data constraint spans across multiple column and hence defined after defining all the table columns when creating or altering a table structure. It is called as table level constraint	Create table name (attribute 1 datatype primary key, attribute 2 datatype constraint constraint–name,……..) Create table name (attribute 1 datatype, attribute 2 datatype2 ,……..,constraint pkey primary) key (attriburw 1 atrributr)

```
Create table emp (eno integer primary key,
                  ename varchar[50],
                  salary float);
                        OR
create table sales_order1(order_no char[10],
                  product_no char[10],,
                  qty_ordered integer,
                  product_rate numeric(8,2),
                  PRIMARY KEY(order_no,product_no));
```

❖ **Alter Table :**

The PURPOSE of this command is to alter the definition of a table in one of these ways:

- to add a column
- to add an integrity constraint
- to redefine a column (datatype, size, default value)
- to modify storage characteristics or other parameters
- to enable, disable, or drop an integrity constraint or trigger
- to explicitly allocate an extent
- to allow or disallow writing to a table
- to modify the degree of parallelism for a table

❖ **Syntax :**

```
ALTER TABLE [schema.]table
[ADD { { column datatype [DEFAULT expr] [column_constraint]...
| table_constraint}
| ( { column datatype [DEFAULT expr] [column_constraint]...
```

| table_constraint}

[, { column datatype [DEFAULT expr] [column_constraint]...

| table_constraint}]...) }]

[MODIFY { column [datatype] [DEFAULT expr] [column_constraint]...

| (column [datatype] [DEFAULT expr] [column_constraint]...

[, column datatype [DEFAULT expr] [column_constraint]...]...)}]

[DROP drop_clause]...

❖ **Where :**

- **Schema –** is the schema containing the table. If you omit schema, Oracle assumes the table is in your own schema.
- **Table** – is the name of the table to be altered.
- **Add** – adds a column or integrity constraint.
- **Modify** – modifies a the definition of an existing column. If you omit any of the optional parts of the column definition (datatype, default value, or column constraint), these parts remain unchanged.
- **Column** – is the name of the column to be added or modified.
- **Datatype** – specifies a datatype for a new column or a new datatype for an existing column.
- **Default** – specifies a default value for a new column or a new default for an existing column.
- **Column_Constraint** – adds or removes a NOT NULL constraint to or from and existing column.
- **Table constraint –** adds an integrity constraint to the table. For Example, To add an advisor column into the Student table

  SQL> ALTER TABLE Student AsDD (advisor VARCHAR2(30));

  Table altered.

❖ **Drop Table :**

The Purpose Of This Command Is To Remove A Table And All Its Data From The Database.

❖ **Syntax :**

DROP TABLE [schema.]table

[CASCADE CONSTRAINTS]

❖ **Where :**

- **Schema** – is the schema containing the table. If you omit schema, Oracle assumes the table is in your own schema.
- **Table** – is the name of the table to be dropped.
- **Cascade Constraints** – drops all referential integrity constraints that refer to primary and unique keys in the dropped table. If you omit this option, and such referential integrity constraints exist, SQL returns an error and does not drop the table.

For Example :

To drop the Student table, enter:

SQL> DROP TABLE Student;

Table dropped.

❑ **Check Your Progress – 3 :**

1. Explain basic DDL statements.

.....

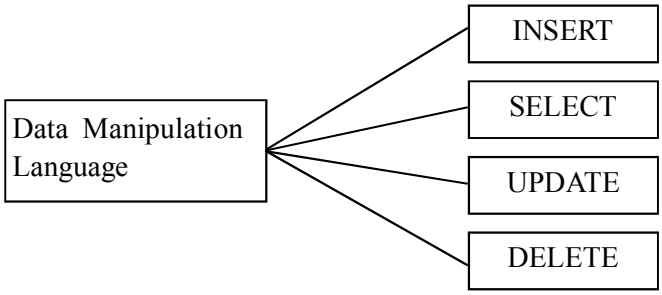
.....

.....

.....

.....

7.5 DML Commands :



Data Manipulation Language is used to insert, modify and delete the records in the database. It never modifies the schema of the database (table features, relationships etc.).

DML include three statements: INSERT, UPDATE and DELETE.

❖ **Insert :**

The Purpose of This Command Is To Add Rows To A Table Or To A View's Base Table.

❖ **Syntax :**

INSERT INTO [schema.]{table | view | subquery }

[(column [, column]...)]

{VALUES (expr [, expr]...) | subquery}

❖ **Where :**

- **schema** – is the schema containing the table or view. If you omit schema, SQL assumes the table or view is in your own schema.
- **table | view** – is name of the table into which rows are to be inserted. If you specify view, SQL inserts rows into the view's base table.
- **subquery** – is a SELECT statement that is treated as if it is a view.
- **column** – is a column of the table or view. In the inserted row, each column in this list is assigned a value from the VALUES clause or the subquery. If you omit one of the table's columns from this list, the column's value for the inserted row is the column's default value as specified when the table was created. If you omit the column list altogether, the VALUES clause or query must specify values for all columns in the table.

- **Values** – specifies a row of values to be inserted into the table or view. You must specify a value in the VALUES clause for each column in the column list.
- **Subquery** – is a SELECT statement that returns rows that are inserted into the table. The select list of this subquery must have the same number of columns as the column list of table.

For Example :

- To insert a student named Smith into the Student Table :

SQL> INSERT INTO Student

VALUES ('Smith', 17, 1, 'CS');

1 row created.

UPDATE:

The PURPOSE of this command is to change existing values in a table or in a view's base table.

❖ **Syntax :**

UPDATE [schema.]{table | view | subquery}[@dblink]

SET { (column [, column]...) = (subquery)

| column = { expr | (subquery) } }

[, { (column [, column]...) = (subquery)

| column = { expr | (subquery) } }]...

[WHERE condition]

Where :

- **schema** – is the schema containing the table or view. If you omit schema, SQL assumes the table or view is in your own schema.
- **table | view | subquery** – is the name of the table to be updated. If you specify view, SQL updates the view's base table. A subquery is treated the same as a view.
- **column** – is the name of a column of the table or view that is to be updated. If you omit a column of the table from the SET clause, that column's value remains unchanged.
- **expr** – is the new value assigned to the corresponding column. This expression can contain host variables and optional indicator variables.
- **Subquery** – is a SELECT statement that returns new values that are assigned to the corresponding columns.
- **Where** – restricts the rows updated to those for which the specified condition is TRUE. If you omit this clause, Oracle updates all rows in the table or view.

Example :

To change a major name from 'COCS' to 'CS' for all students, enter :

SQL> UPDATE Student

SET Major = 'CS'

WHERE Major = 'COSC';

6 rows updated.

❖ **Delete :**

The PURPOSE of this command is to remove rows from a table or from a view's base table.

❖ **Syntax :**

DELETE [FROM] [schema.]{table | view}[@dblink]

[WHERE condition]

❖ **Where :**

- **schema** – is the schema containing the table or view. If you omit schema, SQL assumes the table or view is in your own schema.
- **table | view** – is the name of a table from which the rows are to be deleted. If you specify view SQL deletes rows from the view's base table.
- **Where** – deletes only rows that satisfy the condition. The condition can reference the table and can contain a subquery. If you omit this clause, Oracle deletes all rows from the table.

  For Example : To delete all students who have major in 'VOODOO', enter:

  SQL> DELETE FROM Student

  WHERE MAJOR = 'VOODOO';

  0 rows deleted.

❑ **Check Your Progress – 4 :**

1. Explain DML statements with example.

.....

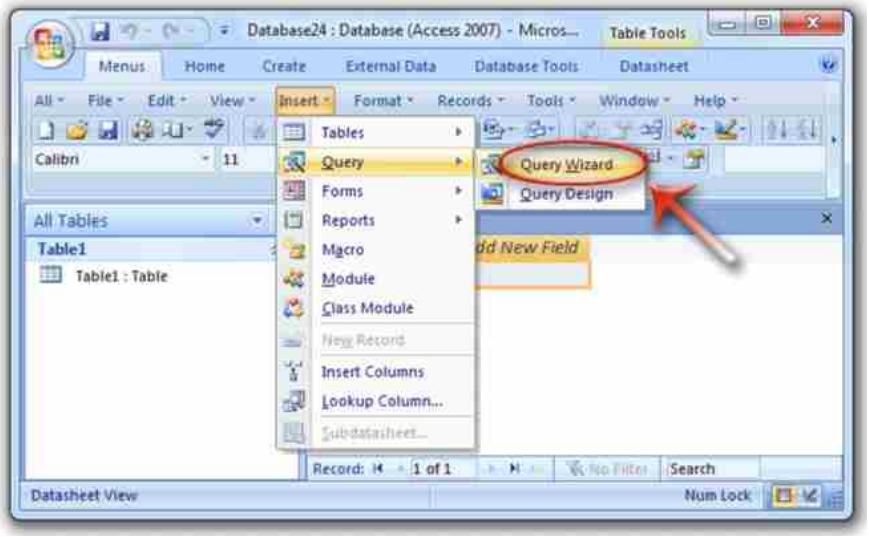
.....

.....

.....

.....

7.6 Simple Queries :



Queries

To retrieve data from the database, use the SELECT statement. Once again, proper privileges are required and are maintained by the DBA. The SELECT statement has the following format:

```
SELECT column(s)
FROM tables(s)
WHERE conditions are met
GROUP BY selected columns
ORDER BY column(s);
```

Every SQL statement ends with a semicolon (;). When you are writing scripts (disk files) that will be executed, you can also use a slash (\) to terminate the SQL statement.

When SELECT column(s) is used, it is assumed that all of the columns fitting the WHERE clause will be retrieved. It is sometimes necessary to only retrieve columns that are distinct from one another. To do this, use the reserved word DISTINCT before the column descriptions. In the following example, a SELECT statement is used to retrieve all of the cities and states from the addresses table (defined previously).

```
SELECT city, state
FROM addresses;
```

When you run this code, every city and state will be retrieved from the table. If 30 people lived in Rochester, NY, the data would be displayed 30 times. To see only one occurrence for each city and state use the DISTINCT qualifier, as shown in the following example:

```
SELECT DISTINCT city, state
FROM addresses;
```

This will cause only one row to be retrieved for entries with Rochester, NY.

The FROM clause is a listing of all tables needed for the query. You can use table aliases to help simplify queries, as shown in the following example:

```
SELECT adrs.city, adrs.state
FROM addresses adrs;
```

In this example, the alias adrs has been given to the table addresses. The alias will be used to differentiate columns with the same name from different tables.

The 'WHERE' clause is used to list the criteria necessary to restrict the output from the query or to join tables in the FROM clause. See the following example.

```
SELECT DISTINCT city, state
FROM addresses
WHERE state in ('CA','NY','CT')
AND city is NOT NULL;
```

The above example will retrieve cities and states that are in the states of California, New York, and Connecticut. The check for NOT NULL cities will not bring data back if the city field was not filled in.

The GROUP BY clause tells Oracle how to group the records together when certain functions are used.

```
SELECT dept_no, SUM(emp_salary)
FROM emp
GROUP BY dept_no;
```

The GROUP BY example will list all department numbers once with the summation of the employee salaries for that particular department.

❑ **Check Your Progress – 5 :**

1. Explain group by and order by clause with example.

.....

.....

.....

.....

.....

7.7 Nested Queries :

SQL provides a mechanism for the nesting of subqueries.

- A subquery is a select–from–where expression that is nested within another query.
- You can build powerful statements out of simple ones by using sub–queries. They can be very useful when you need to select rows from a table with a condition that depends on the data in the table itself.

General form of sub query

```
is SELECT select_list
FROM table
WHERE expr operator
( SELECT select_List
FROM table);
```

- The subquery (inner query) executes once before the main query.
- The result of the sub–query is used by the main query (outer query).
- You can place the sub–query in a number of SQL clauses
- WHERE clause
- HAVING clause
- FROM clause in the syntax;

operator includes a comparison operatör such as >, =, or IN

Note : Comparison operutors fall i11to two classes: single–row operators (>,=,>=,<,<>,<=)

and multiple–row operators (IN, ANY, ALL).

For example : Find employee name whose salary is greater than empno = 7566

```
SELECT ename
FROM EMP
WHERE sal >
( SELECT sal
FROM emp
WHERE empno=7566);
```

Output :

```
ENAME
FORD
SCOTT
KING
FORD
```

- A common use of sub queries is to perform tests for set membership, set comparisons, and set cardinality.

SET MEMBERSHIP:

The 'in' & 'not in' connectivity tests are used for set membership & absence of set membership respectively.

For example, display the employee number and name for all employees who work in a department with any employee whose name contains a T.

```
SELECT empno, ename
FROM emp
WHERE deptno IN
( SELECT deptno
FROM emp
WHERE ename LIKE '%T%' );
```

Set Comparisons:

The < some, > some, <= some, >= some, = some, <> some are the constructs allowed for comparison.

= some is same as the 'in' connectivity.

<> some is not the same as the 'not in' connectivity.

Similarly sql also provides < all, >all, <=all, >= all, <> all comparisons.

<>all is same as the 'not in' construct.

```
SELECT empno, ename, job
FROM emp
WHERE sal < ANY
( SELECT sal
FROM emp
WHERE job = 'CLERK' );
```

Output :

EMPNO ENAME JOB

7369 SMITH CLERK

7900 JAMES CLERK

7876 ADAMS CLERK

7521 WARD SALESMAN

7654 MARTIN SALESMAN

❖ **Set Cardinality :**

The 'exists' construct returns the value true if the argument subquery is nonempty. We can test for the non–existence of tuples in a subquery by using the 'not exists' construct. The 'not exists 'construct can also be used to simulate the set containment operation (the super set). We can write "relation A contains relation

B" as "not exists (B except A)".

For example, display employee and their dept. names whose salary > 3500

SELECT ename, deptno

FROM emp

WHERE EXISTS

(SELECT *

FROM emp

WHERE sal >3500)

ENAME DEPTNO

SMITH 20

ALLEN 30

WARD 30

JONES 20

MARTIN 30

BLAKE 30

CLARK 10

SCOTT 20

KING 10

TURNER 30

ADAMS 20

JAMES 30

FORD 20

MILLER 10

❑ **Check Your Progress – 6 :**

1. Explain SQL statements involving set membership, set comparisons and set cardinality operations.

.....

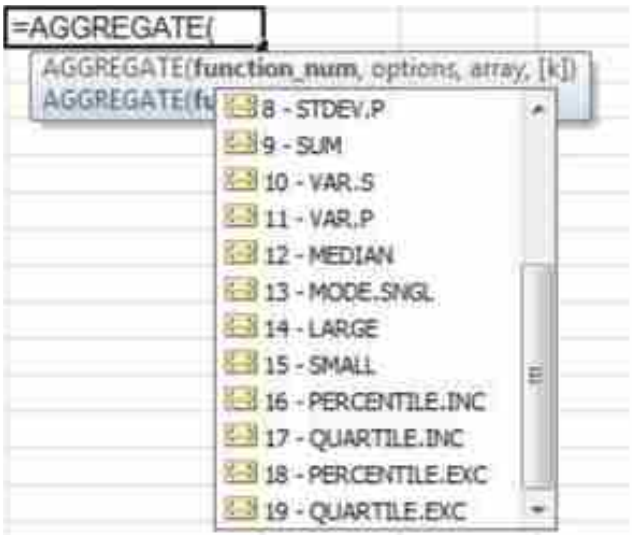
.....

.....

.....

.....

7.8 Aggregate Functions :



Aggregate Functions

The five important aggregate functions are SUM, AVG, MAX, MIN, and COUNT. They are called aggregate functions because they summarize the results of a query, rather than listing all of the rows.

(1) SUM () gives the total of all the rows, satisfying any conditions, of the given column, where the given column is numeric.

(2) AVG () gives the average of the given column.

(3) MAX () gives the largest figure in the given column.

(4) MIN () gives the smallest figure in the given column.

(5) COUNT (*) gives the number of rows satisfying the conditions.

- FIND ENAME WHO WAS HIRED FIRST :

  SQL> SELECT ENAME FROM EMP WHERE HIREDATE IN(SELECT MIN(HIREDATE) FROM EMP);

  ENAME

  – – – – –

  SMITH

- FIND TOTAL SALARY FOR THOSE WHO ARE NOT MANAGER
  SQL> SELECT SUM(SAL) FROM EMP WHERE JOB<>'MANAGER';
  SUM(SAL)

  8275

- FIND TOTAL SALARY OF EACH DEPT EXCLUDING THE EMPLOYEE WHO ARE NOT SALESMAN AND DISPLAY ONLY THOSE DEPT WHOSE TOTAL>7000

  SQL> SELECT DEPTNO,SUM(SAL) FROM EMP WHERE

  JOB!='SALESMAN' GROUP BY DEPTNO HAVING SUM(SAL)>7000;

DEPTNO	SUM (SAL)
10	8750
20	10875

- FIND AVG SALARY FOR ALL THE JOB TYPES WITH MORE THAN 2 EMPLOYEES

  SQL> SELECT JOB,AVG(SAL) FROM EMP GROUP BY JOB HAVING COUNT(JOB)>2;

JOB	AVG(SAL)
CLERK	1037.5
MANAGER	2758.3333
SALESMAN	1400

- DISPLAY EMP. COUNT FOR EACH JOB CATAGORY

  SQL> SELECT JOB,DEPTNO,COUNT(ENAME) FROM EMP GROUP BY JOB,DEPTNO;

JOB	DEPTNO	COUNT(ENAME)
_ _ _ _ _	_ _ _ _ _	_ _ _ _ _
ANALYST	20	2
CLERK	10	1
CLERK	20	2
CLERK	30	1
MANAGER	10	1
MANAGER	20	1
MANAGER	30	1
PRESIDENT	10	1
SALESMAN	30	4

  9 ROWS SELECTED.

❑ **Check Your Progress – 7 :**

1. Explain any five aggregate functions.

.....

.....

.....

.....

.....

2. Designers use which of the following to tune the performance of systems to support time–critical operations ?

   (A) Denormalization (B) Redundant optimization

   (C) Optimization (D) Realization

3. If one attribute is determinant of second, which in turn is determinant of third, then the relation cannot be:

   (A) Well–structured (B) 1NF

   (C) 2NF (D) 3NF

4. 5NF is designed to cope with :

   (A) Transitive dependency (B) Join dependency

   (C) Multi valued dependency (D) None of these

5. Third normal form is inadequate in situations where the relation :

   (A) has multiple candidate keys

   (B) has candidate keys that are composite

   (C) has overlapped candidate keys

   (D) none of the above

6. _____ function gives the total of all the rows.

   (A) sum() (B) Avg() (C) Min() (D) Max()

7.9 Let Us Sum Up :

Almost all relational database management systems use SQL (Structured Query Language) for data manipulation and retrieval. SQL is the standard language for relational database systems. SQL is a non–procedural language, where you need to concentrate on what you want, not on how you get it. Put it in other way, you need not be concerned with procedural details.

SQL Commands are divided into four categories, depending upon what they do.

_ DDL (Data Definition Language)

_ DML (Data Manipulation Language)

_ DCL (Data Control Language)

_ Query (Retrieving data)

DDL commands are used to define the data. For example, CREATE TABLE.

DML commands such as, INSERT and DELETE are used to manipulate data.

DCL commands are used to control access to data. For example, GRANT.

Query is used to retrieve data using SELECT.

DML and Query are also collectively called as DML. And DDL and DCL are called as DDL.

7.10 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 1.2

- **Check Your Progress 2 :**

  See Section 1.3

- **Check Your Progress 3 :**

  See Section 1.4

- **Check Your Progress 4 :**

  See Section 1.5

- **Check Your Progress 5 :**

  See Section 1.6

- **Check Your Progress 6 :**

  See Section 1.7

- **Check Your Progress 7 :**

  1 : See Section 1.8 **2 :** Denormalization

  3 : 3NF **4 :** Join dependency

  5 : none of the above, **6 :** sum()

7.11 Glossary :

1. **Aggregate** – A single [summary] field's value, based on results calculated from values found in like fields across an entire set or subset of records; typically counts, sums, averages, first, last, minimum and maximum.
2. **Column** – Synonymous with field.
3. **Constraints** – Data restrictions specified in a database; rules that determine what values the field to the table can assume.
4. **Data Retrieval** – Data extraction from disparate sources, most operational, some legacy — typically in different formats.
5. **Data Type** – Every field in every table in a database must be declared as a specific type of data with defined parameters and limitations (e.g. numeric, character or text, date, logical, etc.), known as a data type.
6. **Key** – A key is a field, or combination of fields, that uniquely identifies a record in a table.
7. **Candidate Key** – (1) One or more fields that will uniquely identify one record in a table; (2) A potential primary key.
8. **Composite Key** – A key made up of two or more table columns that, together, guarantee uniqueness, when there is no single column available that can guarantee uniqueness by itself.
9. **Foreign key** – A column or group of columns in a table that corresponds to or references a primary key in another table in the database. A foreign Key need not itself be unique, but must uniquely identify the field or fields in the table that the key references.

10. **Primary key** – A field or combination of fields that uniquely identifies each record in a table, so that each record can be uniquely distinguished from every other occurring in the table. A table cannot have more than one primary key, and a primary key, by definition may not contain a null value.
11. **Nonprocedural Language –** For example, SQL; a computer language in which the programmer cannot dictate in what order a procedure will process its contained commands.
12. **Procedural Language –** A computer language that solves a problem by sequentially executing steps in either a linear or loop back procedure (or both) until the operation is completed.
13. **Query Language –** A computer language that enables an end–user to create and run queries that interact directly with a DBMS to retrieve, and possibly modify, the data it contains.
14. **Nested query –** A statement that contains one or more sub queries.
15. **Referential Integrity –** (1) A series of rules that defines and manages the link between parent and child records; (2) A state in which all the tables in the database are consistent with each other; (3) The facility of any DBMS that ensures the validity of predefined relationships.
16. **Schema –** (1) The database's metadata — the structure of an entire database, which specifies, among other things, the tables, their fields, and their domains. In some database systems, the linking or join fields are also specified as part of the schema; (2) The description of a single table.
17. **Table –** Synonymous with relation. A collection of data organized into records and fields (a ka rows and columns), with fields being descriptions of the kinds of information contained in each record (attributes); and records being specific instances usually referring to specific objects or persons (entities).
18. **SQL –** Pronounced "Sequel", it stands for Structured Query Language, the standard format for commands that most database software understands.

7.12 Assignment :

Explain in our own words importance of the SQL in today's IT world.

7.13 Activities :

Explain in brief various relational databases available in IT world.

7.14 Case Study :

Consider the following case study :

A music company wants to go in for automation of their requirements. They want to develop a database for maintaining the information of their music albums, singers, musicians, instruments.

The following facts are relevant :

a. each album is produced by many musicians, a musician can produce many albums

b. A singer can sing for many albums, but an album consists of songs of only one singer.

c. A musician can play many instruments, an instrument can be played by many musicians.

The following constraints are to be placed on the relations :

a. each musician is paid a minimum of 50000 Rs. foreach album

b. all singers are from either pune, Mumbai or Chennai

c. each instrument cost is maximum 10000

Design the relational database for the above, so that the following queries can be answered :

1. List the names of musicians who have played guitar for the album 'zindagi'
2. list the names of musicians who palsy at least one instrument same as the one "joshi" plays.
3. List the names of albums, in which "sonu nigam" has sung.
4. List the names of albums released in 1998
5. List the names of albums that have more than two instruments being played in it
6. Delete all information of singers who have not sung in any album
7. Delete all information of musicians, who have worked in the album "zindagi"

7.15 Further Reading :

1. Database Management Systems – Rajesh Narang – PHI Learning Pvt. Ltd.
2. Database System Concepts by Silberschatz, Korth – Tata McGraw – Hill Publication
3. An Introduction to Database Systems – Bipin Desai – Galgotia Publication
4. Database Management System by Raghu Ramkrishnan – Tata McGraw – Hill Publication
5. SQL, PL/SQL : The Programming Language Oracle – Ivan Bayross – BPB Publication

Unit 08

SQL CONSTRAINTS

UNIT STRUCTURE

8.0 Learning Objectives

8.1 Introduction

8.2 Not Null Constraint

8.3 Default Constraint

8.4 Unique Constraint

8.5 Primary Key

8.6 Foreign Key

8.7 Check Constrait

8.8 Let Us Sum Up

8.9 Suggested Answer for Check Your Progress

8.10 Glossary

8.11 Assignment

8.12 Activities

8.13 Case Study

8.14 Further Readings

8.0 Learning Objectives :

After learning this unit, you will be able to understand :

- Not Null Constraint
- Default Constraints
- Unique Constraints
- Primary and Foreign Key
- Check constraint

8.1 Introduction :

Constraints are the rules enforced on the data columns of a table. These are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the database. Constraints could be either on a column level or a table level. The column level constraints are applied only to one column, whereas the table level constraints are applied to the whole table.

Following are some of the most commonly used constraints available in SQL. These constraints have already been discussed in SQL – RDBMS Concepts chapter, but it's worth to revise them at this point.

- NOT NULL Constraint – Ensures that a column cannot have NULL value.
- DEFAULT Constraint – Provides a default value for a column when none is specified.

- UNIQUE Constraint – Ensures that all values in a column are different.
- PRIMARY Key – Uniquely identifies each row/record in a database table.
- FOREIGN Key – Uniquely identifies a row/record in any of the given database table.
- CHECK Constraint – The CHECK constraint ensures that all the values in a column satisfies certain conditions.

Constraints can be specified when a table is created with the CREATE TABLE statement or you can use the ALTER TABLE statement to create constraints even after the table is created.

❖ **Dropping Constraints :**

Any constraint that you have defined can be dropped using the ALTER TABLE command with the DROP CONSTRAINT option.

For example, to drop the primary key constraint in the EMPLOYEES table, you can use the following command.

ALTER TABLE EMPLOYEES DROP CONSTRAINT CONSTRAINT_NAME;

Some implementations may provide shortcuts for dropping certain constraints. For example, to drop the primary key constraint for a table in Oracle, you can use the following command.

ALTER TABLE EMPLOYEES DROP PRIMARY KEY;

8.2 Not Null Constraint :

By default, a column can hold NULL values. If you do not want a column to have a NULL value, then you need to define such a constraint on this column specifying that NULL is now not allowed for that column.

A NULL is not the same as no data, rather, it represents unknown data.

For example, the following SQL query creates a new table called CUSTOMERS and adds five columns, three of which, are ID NAME and AGE, In this we specify not to accept NULLs ?

```
CREATE TABLE CUSTOMERS(
   ID INT          NOT NULL,
   NAME VARCHAR (20) NOT NULL,
   AGE INT         NOT NULL,
   ADDRESS CHAR (25),
   SALARY DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

If CUSTOMERS table has already been created, then to add a NOT NULL constraint to the SALARY column in Oracle and MySQL, you would write a query like the one that is shown in the following code block.

ALTER TABLE CUSTOMERS MODIFY SALARY DECIMAL (18, 2) NOT NULL;

8.3 Default Constraint :

The DEFAULT constraint provides a default value to a column when the INSERT INTO statement does not provide a specific value.

Example :

For example, the following SQL creates a new table called CUSTOMERS and adds five columns. Here, the SALARY column is set to 5000.00 by default, so in case the INSERT INTO statement does not provide a value for this column, then by default this column would be set to 5000.00.

```
CREATE TABLE CUSTOMERS(
   ID INT          NOT NULL,
   NAME VARCHAR (20) NOT NULL,
   AGE INT         NOT NULL,
   ADDRESS CHAR (25),
   SALARY DECIMAL (18, 2) DEFAULT 5000.00,
   PRIMARY KEY (ID)
);
```

If the CUSTOMERS table has already been created, then to add a DEFAULT constraint to the SALARY column, you would write a query like the one which is shown in the code block below.

```
ALTER TABLE CUSTOMERS
MODIFY SALARY DECIMAL (18, 2) DEFAULT 5000.00;
```

Drop Default Constraint

To drop a DEFAULT constraint, use the following SQL query.

```
ALTER TABLE CUSTOMERS ALTER COLUMN SALARY DROP DEFAULT;
```

8.4 Unique Constraint :

The UNIQUE Constraint prevents two records from having identical values in a column. In the CUSTOMERS table, for example, you might want to prevent two or more people from having an identical age.

Example :

For example, the following SQL query creates a new table called CUSTOMERS and adds five columns. Here, the AGE column is set to UNIQUE, so that you cannot have two records with the same age.

```
CREATE TABLE CUSTOMERS(
   ID INT          NOT NULL,
   NAME VARCHAR (20) NOT NULL,
   AGE INT         NOT NULL UNIQUE,
   ADDRESS CHAR (25),
   SALARY DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

If the CUSTOMERS table has already been created, then to add a UNIQUE constraint to the AGE column. You would write a statement like the query that is given in the code block below.

```
ALTER TABLE CUSTOMERS MODIFY AGE INT NOT NULL UNIQUE;
```

You can also use the following syntax, which supports naming the constraint in multiple columns as well.

```
ALTER TABLE CUSTOMERS ADD CONSTRAINT myUniqueConstraint UNIQUE(AGE, SALARY);
```

DROP a UNIQUE Constraint

To drop a UNIQUE constraint, use the following SQL query.

```
ALTER TABLE CUSTOMERS DROP CONSTRAINT myUniqueConstraint;
```

If you are using MySQL, then you can use the following syntax ?

```
ALTER TABLE CUSTOMERS DROP INDEX myUniqueConstraint;
```

8.5 Primary Key Constraint :

A primary key is a field in a table which uniquely identifies each row/ record in a database table. Primary keys must contain unique values. A primary key column cannot have NULL values.

A table can have only one primary key, which may consist of single or multiple fields. When multiple fields are used as a primary key, they are called a composite key.

If a table has a primary key defined on any field(s), then you cannot have two records having the same value of that field(s).

Note – You would use these concepts while creating database tables.

Create Primary Key

Here is the syntax to define the ID attribute as a primary key in a CUSTOMERS table.

```
CREATE TABLE CUSTOMERS(
   ID INT          NOT NULL,
   NAME VARCHAR (20) NOT NULL,
   AGE INT         NOT NULL,
   ADDRESS CHAR (25),
   SALARY DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

To create a PRIMARY KEY constraint on the "ID" column when the CUSTOMERS table already exists, use the following SQL syntax ?

```
ALTER TABLE CUSTOMER ADD PRIMARY KEY (ID);
```

NOTE ? If you use the ALTER TABLE statement to add a primary key, the primary key column(s) should have already been declared to not contain NULL values (when the table was first created).

For defining a PRIMARY KEY constraint on multiple columns, use the SQL syntax given below.

```
CREATE TABLE CUSTOMERS(
   ID INT          NOT NULL,
   NAME VARCHAR (20) NOT NULL,
   AGE INT         NOT NULL,
   ADDRESS CHAR (25),
   SALARY DECIMAL (18, 2),
   PRIMARY KEY (ID, NAME)
);
```

To create a PRIMARY KEY constraint on the "ID" and "NAMES" columns when CUSTOMERS table already exists, use the following SQL syntax.

```
ALTER TABLE CUSTOMERS
   ADD CONSTRAINT PK_CUSTID PRIMARY KEY (ID, NAME);
```

Delete Primary Key

You can clear the primary key constraints from the table with the syntax given below.

```
ALTER TABLE CUSTOMERS DROP PRIMARY KEY ;
```

8.6 Foreign Key Constraint :

A foreign key is a key used to link two tables together. This is sometimes also called as a referencing key. A Foreign Key is a column or a combination of columns whose values match a Primary Key in a different table.

The relationship between 2 tables matches the Primary Key in one of the tables with a Foreign Key in the second table. If a table has a primary key defined on any field(s), then you cannot have two records having the same value of that field(s).

Example

Consider the structure of the following two tables.

CUSTOMERS table

```
CREATE TABLE CUSTOMERS(
   ID INT          NOT NULL,
   NAME VARCHAR (20) NOT NULL,
   AGE INT         NOT NULL,
   ADDRESS CHAR (25),
   SALARY DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

ORDERS table

```
CREATE TABLE ORDERS (
   ID INT          NOT NULL,
   DATE            DATETIME,
   CUSTOMER_ID INT references CUSTOMERS(ID),
   AMOUNT double,
   PRIMARY KEY (ID)
);
```

If the ORDERS table has already been created and the foreign key has not yet been set, the use the syntax for specifying a foreign key by altering a table.

ALTER TABLE ORDERS ADD FOREIGN KEY (Customer_ID) REFERENCES CUSTOMERS (ID);

DROP a FOREIGN KEY Constraint

To drop a FOREIGN KEY constraint, use the following SQL syntax.

ALTER TABLE ORDERS DROP FOREIGN KEY;

8.7 Check Constraint :

The CHECK Constraint enables a condition to check the value being entered into a record. If the condition evaluates to false, the record violates the constraint and isn't entered the table.

Example

For example, the following program creates a new table called CUSTOMERS and adds five columns. Here, we add a CHECK with AGE column, so that you cannot have any CUSTOMER who is below 18 years.

```
CREATE TABLE CUSTOMERS(
   ID INT          NOT NULL,
   NAME VARCHAR (20) NOT NULL,
   AGE INT         NOT NULL CHECK (AGE >= 18),
   ADDRESS CHAR (25),
   SALARY DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

If the CUSTOMERS table has already been created, then to add a CHECK constraint to AGE column, you would write a statement like the one given below.

ALTER TABLE CUSTOMERS MODIFY AGE INT NOT NULL CHECK (AGE >= 18);

You can also use the following syntax, which supports naming the constraint in multiple columns as well ?

ALTER TABLE CUSTOMERS ADD CONSTRAINT myCheckConstraint CHECK(AGE >= 18);

DROP a CHECK Constraint

To drop a CHECK constraint, use the following SQL syntax. This syntax does not work with MySQL.

```
ALTER TABLE CUSTOMERS DROP CONSTRAINT myCheckConstraint;
```

❑ **Check Your Progress :**

1. Which of the following is not a integrity constraint ?
   (A) Not null (B) Positive
   (C) Unique (D) Check 'predicate'
2. Foreign key is the one in which the _____ of one relation is referenced in another relation.
   (A) Foreign key (B) Primary key
   (C) References (D) Check constraint
3. Point out the correct statement.
   (A) CHECK constraints enforce domain integrity
   (B) UNIQUE constraints enforce the uniqueness of the values in a set of columns
   (C) In a UNIQUE constraint, no two rows in the table can have the same value for the columns
   (D) All of the mentioned
4. Which of the following constraint does not enforce uniqueness ?
   (A) UNIQUE (B) Primary key
   (C) Foreign key (D) None of the mentioned
5. Constraints can be applied on _____
   (A) Column (B) Table
   (C) Field (D) All of the mentioned

8.8 Let Us Sum Up :

The constraints are the key points in data base management system. In this unit all the constraints and practical tasks are explained in detail. It also discuss the usage and characteristics of the constraints. All available constraints in DBMS is explained in detail.

8.9 Suggested Answer for Check Your Progress :

❑ **Check Your Progress :**

1 : Positive **2 :** Primary Key
3 : All of the mentioned **4 :** Foreign Key
5 : All of the mentioned

8.10 Glossary :

1. **NOT NULL –** Ensures that a column cannot have a NULL value
2. **UNIQUE –** Ensures that all values in a column are different
3. **PRIMARY KEY –** A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
4. **FOREIGN KEY –** Uniquely identifies a row/record in another table

5. **CHECK –** Ensures that all values in a column satisfies a specific condition
6. **DEFAULT –** Sets a default value for a column when no value is specified
7. **INDEX –** Used to create and retrieve data from the database very quickly

8.11 Assignment :

Create user registration table with all constraints utilization.

8.12 Activities :

Study the constraints in detail.

8.13 Case Study :

Create the tables for User registration and login process.

8.14 Further Reading :

i. Database Management Systems – Rajesh Narang – PHI Learning Pvt. Ltd.

ii. Database System Concepts by Silberschatz, Korth – Tata McGraw–Hill Publication

iii. An Introduction to Database Systems – Bipin Desai – Galgotia Publication

iv. Database Management System by Raghu Ramkrishnan – Tata McGraw–Hill Publication

v. Database Solutions (2nd Edition) By Thomas M Connolly & Carolyn E Begg

Unit 09

TRANSACTION PROCESSING

UNIT STRUCTURE

9.0 Learning Objectives

9.1 Introduction

9.2 Types of Transactions

- 9.2.1 Concurrent Transactions
- 9.2.2 Discreet Transactions
- 9.2.3 Distributed Transactions
- 9.2.4 In–Doubt Transactions
- 9.2.5 Normal Transactions
- 9.2.6 Read–Only Transactions
- 9.2.7 Remote Transactions

9.3 Read–Consistency

9.4 Steps to Processing a Transaction

- 9.4.1 Entering DML/DDL Statements
- 9.4.2 Assigning Rollback Segments
- 9.4.3 Long–Running Transactions and Rollback Segment Allocation
- 9.4.4 Using the Optimizer
- 9.4.5 Cost–Based Analysis
- 9.4.6 Rule–Based Analysis
- 9.4.7 Overriding the Optimizer_Mode Parameter
- 9.4.8 Parsing Statements
- 9.4.9 Handling Locks
- 9.4.10 Stepping Through the Transaction

9.5 Processing a Remote or Distributed Transaction

- 9.5.1 Entering DDL/DML Statements
- 9.5.2 Assigning Rollback Segments
- 9.5.3 Breaking down Statements
- 9.5.4 Optimizing Local Statements
- 9.5.5 Forwarding Remote Commands
- 9.5.6 Assigning Remote Rollback Segments and Writing Redo Logs
- 9.5.7 Optimizing Remote Statement
- 9.5.8 Returning Data to the Local Database
- 9.5.9 Summarizing Remote and Distributed Transactions

9.6 Let Us Sum Up

9.7 Suggested Answer for Check Your Progress

9.8 Glossary

9.9 Assignment

9.10 Activities

9.11 Case Study

9.12 Further Readings

9.0 Learning Objectives :

After learning this unit, you will be able to understand :

- The difference between a session and a transaction
- Know what happens when a transaction is started
- How rollback segments are assigned
- How the database handles multiple and concurrent transactions
- Locking techniques used to ensure data integrity
- Execution of SQL and PL/SQL statements
- In–doubt and distributed transactions
- Commits, set points, and rollbacks
- Behind–the–scenes processes that comprise a session and transaction
- Database parameters that affect the execution of a transaction

9.1 Introduction :

Understanding how a transaction begins, executes, and ends, and knowing what happens along each step of the way are vital parts of making Oracle work for you. This knowledge is helpful not only to system and database administrators, but to Oracle developers as well. Knowing when a transaction is issued, a rollback segment, or how locking occurs in the database can drastically change the strategy of creating applications or nightly processes.

A transaction is directly related to a session, but it is still considered a separate entity. A session, simply stated, is a single connection to a database instance based upon a username and, optionally, a password. All sessions in a database instance are unique, which means that they have a unique identifier setting them apart from the other users and processes accessing the database. This unique identifier, called a SID, is assigned by the instance and can be reused after a session is ended. The combination of the SID and a session serial number guarantees that each no session, even if the number is reused, is identical.

A transaction, also in simplified terms, is a specific task, or set of tasks, to be executed against the database. Transactions start with an executable DML statement and end when the statement or multiple statements are all either rolled back or committed to the database, or when a DDL (Data Definition Language) statement is issued during the transaction.

If COMMIT or ROLLBACK statements are issued from the command line, the transaction is said to have been explicitly ended. However, if you issue a DDL command (DROP TABLE, ALTER TABLE, etc.) the previous statements in your transaction will be committed (or rolled back if unable to commit), the transaction will be implicitly ended, and a new transaction will begin and then end.

To illustrate these rules, assume that you log into your database to update and modify your customer tables. What you would like to do is enter 100 new customers to your database. You do this by creating a temporary table to hold that customer information, search your customer table for duplicates, and update those records if they do not exist. Though this is unlikely, assume that you must update customer table before checking for duplicate entries. The sequence would look like the steps listed in the following sequence of session and transaction begins and ends without save points.

1. Connect to SQL*Plus (begin session #1).
2. Create temporary customer table (begin and end transaction #1).
3. Insert new customer information into temporary table (begin transaction #2).
4. Step through entries in temporary table (continue transaction #2).
5. Update customer table (continue transaction #2).
6. Check for duplicate entries (continue transaction #2). If duplicates exist, roll back entire transaction (end transaction #2).
7. Repeat steps 4–7 until complete or duplicates are found.
8. Drop temporary table (end transaction #2, begin and end transaction #3).
9. Exit from SQL*Plus (end session #1).

Notice how the create–table and drop–table steps (2 and 8) begin and end a transaction. If you found duplicate entries in your tables, step 8 would actually end transaction #3 and begin and end transaction #4. Also note that the DDL command in step 5 implicitly ended transaction #2 by committing any changes made before beginning transaction #3. Finally, it is important to realize that if you had done another update between steps 5 and 6, the exit from SQL*Plus would have implicitly ended transaction #4 (started by the update) by issuing a commit before exiting.

One other form of implicitly ending a transaction includes terminating a session either normally or abnormally. When these situations arise, the instance automatically attempts to commit the current transaction. If that is not possible, the transaction will be rolled back.

Commits, Rollbacks, and Save points

Although these topics are discussed elsewhere in this book, it is important to note how they affect a given transaction. As mentioned earlier, commits and rollbacks both end a transaction. Commit makes all changes made to the data permanent. Rollback reverses all changes made during the transaction by restoring the previous state of all modified data. With the use of save points, the ROLLBACK command can also be used to roll back only a portion of a transaction.

Save points were designed to be used as logical stopping points from within a single transaction. They are helpful in splitting up extremely long transactions into smaller portions, and they provide points of recovery along the way. Using save points within a transaction enables you to roll back the transaction to any given save point as long as a commit has not been issued (which immediately commits all data, erases all save points, and ends the transaction). Refer to Chapter 6, "SQL*Plus," to learn more about the

SAVEPOINT command as well as how to use ROLLBACK to roll the current transaction back to a specified save point.

The following list is an update to the previously shown sequence with the addition of save points. Refer to this example to show how save points affect the transaction.

1. Connect to SQL*Plus (begin session #1).
2. Create temporary customer table (begin and end transaction #1).
3. Insert new customer information into temporary table (begin transaction #2).
4. Step through each entry in the temporary table (continue transaction #2).
5. Create unique save point (continue transaction #2).
6. Update customer table with information from temporary table (continue transaction #2).
7. If duplicate customer is found, roll back to save point (continue transaction #2).
8. Repeat steps 4–7 until finished.
9. Issue commit (end transaction #2).
10. Drop temporary table (begin and end transaction #3).
11. Exit from SQL*Plus (end session #1).

Notice how the save point enables you to roll back to a point within your current transaction without affecting the previous updates before the save point. Anywhere within your procedure, you can roll back to any save point or you can roll back the entire transaction. By using the save points, you are providing a collection of recovery points that are available to you until you end that transaction. Once the transaction is ended, all save points are erased.

Multiple save points give you even greater flexibility in rolling back portions of long or complex transactions if an error occurs before the completion of the process.

There is no limit to the number of save points you can create during any given transaction, but be careful that the ones you do create are logically named in case you must roll back to them.

Transaction Control Statements

Transaction control statements are statements that affect the execution or properties of a transaction, whether it is the management of data or characteristics of how the transaction executes. The family of transaction control statements includes:

- COMMIT
- SAVEPOINT
- ROLLBACK
- SET TRANSACTION

9.2 Types of Transactions :

Several names are used to identify transactions and their states. Knowing these terms is helpful in understanding the terms mentioned by Oracle and

interpreting Oracle errors returned during a transaction. These terms cover types of transactions as well as other terms used in identifying them.

9.2.1 Concurrent Transactions :

Concurrent transactions are transactions that are executed in the same general time. These transactions, because they have started so close to each other, generally do not see the changes made by the other transactions. Any data that has been updated by a concurrent transaction and requested by another concurrently running transaction must be read from rollback segments until the transaction requesting the data has completed.

9.2.2 Discreet Transactions :

A discreet transaction is used to improve the performance of short transactions. For developers creating custom applications, the procedure BEGIN_DISCREET_TRANSACTION() changes the steps followed during the duration of a session in order to speed its processing. The main differences are as follows :

- All changes are held until the transaction ends
- Other transactions cannot see uncommitted changes
- Redo information is stored in a separate location in the SGA
- No rollback information is written because all changes are held until a commit and then applied directly to the data block(s)

9.2.3 Distributed Transactions :

Distributed transactions are transactions in which one or more statements manipulate data on two or more nodes, or remote systems of a distributed database. If a transaction manipulates data on only one node, it is considered a remote transaction. As in a remote transaction, none of the redo information is stored locally.

9.2.4 In–Doubt Transactions :

An in–doubt transaction is actually a state of a transaction instead of a type and refers to transactions within a distributed database environment. One situation that causes this state is if an instance involved in a currently running transaction fails, that transaction must be either rolled back or committed. It is difficult, however, to do either without knowing the state of the transaction in the affected database. In this case, all other instances in the distributed environment mark this transaction as in–doubt. Once the instance is restarted, the transaction can be analyzed and all instances can either commit or rollback.

9.2.5 Normal Transactions :

Normal transaction is a term used to refer to a local (non–remote) transaction. All redo information is stored in the local database, and all data manipulation is done to the same database. This type of transaction is the focus for the discussion on transaction processing.

9.2.6 Read–Only Transactions :

Read–only refers to the type of read consistency that is set or defaulted to for a given transaction. By default, the level of read consistency is statement level, which is also known as read–write. This means that each consecutive statement in your transaction will see the changes made to the database by

any previous statements regardless of whose transaction has committed the changes.

9.2.7 Remote Transactions :

Remote transactions are transactions containing single or multiple statement(s) to be executed against a non–local database. All these statements reference the same node. If they do not, they are considered separate remote transactions and the instance will split them up. One of the major differences between remote and normal transactions is that redo and rollback information against a remote transaction is stored on the remote database. None of this information is transferred to your local database to be used for recovery.

❑ Check Your Progress – 1 :

1. Explain all the types of transactions.

.....

.....

.....

.....

.....

9.3 Read–Consistency :

Read–consistency is not a difficult concept to grasp. In short, read–consistency guarantees that the data you are viewing while executing a transaction does not change during that transaction. With read–consistency, if two users are updating the same table at the same time, user 1 will not see the changes made by the other user during their transaction. User 2, likewise, cannot see any changes committed by user 1 until both transactions are complete. If they happen to be working on the same row in the table, this becomes a locking issue instead of read–consistency. A later section discusses locking.

Read–consistency is the major building block that enables multiple users to update, select, and delete from the same tables without having to keep a private copy for each user. When combined with locking techniques, read–consistency provides the foundation for a multi–user database in which users can do similar or identical operations without difficulty.

Take a look at an example of the way read–consistency works in a theoretical telemarketing department; user1 is entering an order for a customer, while user 2 is changing customer information. Both users have concurrent transactions (they are executing at the same time), but user1 began the transaction first. Suppose that user 2 makes a mistake and changes the phone number for the same customer whose order is being entered. Because user1 began the transaction first, they will always be looking at the 'before picture' of the customer data and will see the customer's previous phone number when querying the user's data. This is true even if user 2 commits their changes. Why ? Because it is possible that user1's transaction is solely dependent on the data that existed when their transaction began. Imagine the confusion that would result if data could be changed while an already executing query was making changes based on that data! It would be nearly impossible to guarantee the coordination and functioning of all processing within the application.

Read–consistency is also a secondary function of rollback segments. Aside from being able to undo changes from a transaction, they also provide other users with a 'before picture' of the data being modified by any process. If a transaction must review data that has been modified by a concurrent uncommitted transaction, it must look in the rollback segments to find that data.

❑ **Check Your Progress – 2 :**

1. Explain the concept read consistency.

.....

.....

.....

.....

.....

9.4 Steps to Processing a Transaction :

Understanding the steps followed during the execution of a transaction can be quite helpful in planning and implementing custom applications. It is also important for the database administrator to know these steps because they can help in understanding and tuning the database parameters and processes. This discussion covers normal transactions.

Other transactions, such as distributed, remote, and discreet, are treated a bit differently, because these transactions are short in nature, and those differences are documented throughout this chapter. The processing steps follow.

1. DML/DDL statement is entered.
2. Rollback segment is assigned or requested.
3. Statement is optimized.
4. Optimizer generates an execution plan.
5. The execution plan is followed to manipulate/return data.
6. Transaction loops through steps 1–5 until commit, rollback or session termination.

The following sections examine each step individually.

9.4.1 Entering DML/DDL Statements :

The issuing of DML or DDL statements can take place through a number of ways, including SQL*Forms, SQL*Plus, and custom C programs. The rules governing the start and end of transactions are the same no matter which way a SQL statement is issued.

9.4.2 Assigning Rollback Segments :

Rollback segments are assigned randomly by Oracle at the beginning of each transaction (not session) when the first DML statement is issued, and they are used to roll back the transaction to a specified save point or to the beginning of a transaction. The selection of a rollback segment is based on which rollback segments are currently available and how busy each segment is. By default, DDL statements are not issued rollback segments due to the nature of DDL commands. They are, however, issued redo entries so that the modifications can be reapplied if the database must be recovered.

Two types of transactions do not acquire rollback segments. These are read–only transactions and remote transactions. Read–only transactions, by their nature, do not modify data, so they do not require a rollback segment. Remote transactions actually do acquire rollback segments, but these rollback segments are allocated on the remote database that the transaction is executed on. Distributed transactions are really a form of remote transactions and follow the same rule.

There is no limit to the number of rollback segments a user can access throughout a given session, but only one rollback segment will be used at any given time for any transaction. In other words, a transaction will acquire one and only one rollback segment to be used for the duration of the transaction. Once the transaction is complete, the rollback segment is released. Space used in that rollback segment is dictated by the amount of data that is modified.

9.4.3 Long–Running Transactions and Rollback Segment Allocation :

Transactions that modify large amounts of data require larger rollback segments. By using the SET TRANSACTION command, you can specify a specific rollback segment to be used by a given transaction. Reference the section on SET TRANSACTION for a further explanation of how to do this. It is important to note, however, that a SET TRANSACTION command must be the very first command issued in a transaction. If it is not, an error message will be returned.

Once a transaction is completed, the rollback segment is released. This does not mean that the segment's data is overwritten immediately, though. Sometimes other transactions that started before this transaction finished need access to the unmodified data for read–consistency. In this case, the rollback segment containing the 'before picture' will be used. Unfortunately, Oracle does not lock this data into the rollback segment to prevent the data blocks from being reused if needed.

If your rollback segments are too small, you may encounter the error rollback segment too old. If this error occurs, the transaction that received the error is forced to rollback. This error implies two things:

- A rollback segment's extent was reused for a new transaction
- A transaction that started before the other transaction ended required access to the previously unmodified data for read–consistency

In either situation, a transaction was accessing the before picture of some data that was still in a rollback segment when the system was forced to reclaim that extent to use for a currently executing transaction. Because this before picture is no longer available, the executing transaction cannot continue.

You can use three steps (separately or in conjunction with each other) to alleviate this problem:

1. Increase the size of your rollback segments.
2. Increase the OPTIMAL size of your rollback segments.
3. Reschedule your processing so that no two processes are updating and/ or reading from updated tables while the other is running.

9.4.4 Using the Optimizer :

Oracle's optimizer is a critical part in the execution of a transaction. The optimizer is responsible for taking a SQL statement, identifying the most efficient way of executing the statement, and then returning the data requested. There is a high likelihood that a SQL statement can be executed in more than one way. The optimizer is responsible for identifying the most efficient means of executing that statement.

Optimization can take many steps to complete, depending on the SQL statement. The steps used to execute the SQL statement are called an execution plan. Once the execution plan is completed, it is then followed to provide the desired results (updated or returned data).

9.4.5 Cost–Based Analysis :

Cost–based analysis is a mode of analyzing SQL statements to provide the most efficient way of execution. When the optimizer is running in cost–based mode, it follows these steps to decide which plan is the best way to execute the statement unless the developer has provided a hint to use in the execution.

1. Generate a set of execution plans based on available access paths.
2. Rank each plan based on estimated elapsed time to complete.
3. Choose the plan with the lowest ranking (shortest elapsed time).

Cost–based analysis uses statistics generated by the ANALYZE command for tables, indexes, and clusters to estimate the total I/O, CPU, and memory requirements required to run each execution plan. Because the goal of the cost–based approach is to provide maximum throughput, the execution plan with the lowest ranking or lowest estimated I/O, CPU, and memory requirements will be used.

The analysis used to provide the final cost of an execution plan is based on the following data dictionary views:

•	USER_TABLES, USER_CLUSTERS	USER_TAB_COLUMNS,	USER_INDEXES,
•	ALL_TABLES, ALL_CLUSTERS	ALL_TAB_COLUMNS,	ALL_INDEXES,
•	DBA_TABLES, DBA_CLUSTERS	DBA_TAB_COLUMNS,	DBA_INDEXES,

9.4.6 Rule–Based Analysis :

Rule–based analysis rates the execution plans according to the access paths available and the information in Table 9.1. The rule–based approach uses those rankings to provide an overall rating on the execution plan and uses the plan with the lowest ranking. Generally speaking the lower the rating, the shorter the execution time though this is not always the case.

Table 9.1 : Access type ratings Ranking Type of Access

1	Single row by ROWID
2	Single row by cluster join
3	Single row by hash cluster key with unique or primary key
4	Single row by unique or primary key
5	Cluster join
6	Hash cluster key
7	Indexed cluster key
8	Composite index
9	Single–column index
10	Bounded range search on indexed columns
11	Unbounded range search on indexed columns
12	Sort–merge join
13	MAX() or MIN() of indexed column
14	ORDER BY on indexed columns
15	Full table scan

9.4.7 Overriding the Optimizer_Mode Parameter :

Because the developer can sometimes optimize code more efficiently than the optimizer can, various directives, called hints, can be issued from within the SQL statement to force the optimizer to choose a different method of optimization. This method works at the statement level from within the transaction and affects only the current statement.

To affect all statements at the transaction level, the SQL command ALTER SESSION SET OPTIMIZER_GOAL can be used. This command overrides the OPTIMIZER_MODE initialization parameter and forces all statements within the current transaction to be optimized according to this value. This parameter has four possible values :

- **CHOOSE.** Tells the optimizer to search the data dictionary views for data on at least one related table (referenced in the SQL statement). If the data exists, the optimizer will optimize the statement according to the cost–based approach. If no data exists for any tables being referenced, the optimizer will use rule–based analysis.
- **ALL_ROWS.** Chooses cost–based analysis with the goal of best throughput.
- **FIRST_ROWS.** Chooses cost–based analysis with the goal of best response time.
- **RULE.** Chooses rule–based analysis regardless of the presence of data in the data dictionary views related to the tables being referenced.

This parameter affects all SQL statements issued from within the transaction, including functions and stored procedures that are called. OPTIMIZER_MODE is still used for any recursive SQL calls issued by Oracle on behalf of the transaction, though.

9.4.8 Parsing Statements :

A parsed statement is not to be confused with an execution plan of a statement. Whereas an execution plan examines the most efficient way to execute a statement, parsing the statement creates the actual executable statements to be used in retrieving the data. Parsing a statement is a one–step process by the optimizer to do the following:

- Check semantics and syntax
- Verify that the user has the appropriate privileges to execute this statement
- Allocate private SQL space to store the statement
- Check for duplicate statements in the shared SQL area
- Generate an executable version of parsed SQL if necessary
- Allocate and stores SQL in shared library cache if it does not already exist

When checking the syntax and semantics, the instance is verifying that no key words or necessary parameters are missing. If the statement is in correct form, the instance then verifies that the user has the correct privileges required to carry out the execution of the statement. Once these have been verified, space is allocated in the private SQL area for the user's statement. This statement is saved until either it is needed again or the memory space is required to store another parsed statement.

After allocating space in the private SQL area, the instance searches through the shared SQL area for any duplicate statements. If a duplicate statement is found, the executable version of the statement is retrieved from memory and executed by the process, and the private SQL area is pointed to the statement in the shared area. If it is not found, an executable version is created and stored in the private SQL area only.

9.4.9 Handling Locks :

The locking of data rows and/or tables is completely automated and transparent to the user. Once the executable version of the SQL statement is run, Oracle automatically attempts to lock data at the lowest level required. This means that if possible, a row will be locked instead of the entire table. This is dependent solely on how the SQL statement was written and what types of access are required (full table scan versus single rows).

A form of manual, or explicit, locking can take place by using the LOCK TABLE command. By default, these commands are not necessary in day–to–day processing. Oracle recommends that you allow the database to handle all locking of data whenever possible.

9.4.10 Stepping Through the Transaction :

From this point, there are several paths that a transaction can take to completion. Most commonly, the transaction is committed. Still, handling must be taken into account for transactions that are rolled back. Following are the steps taken during a commit:

1. Instance's transaction table marks transaction as complete.
2. A unique SCN (system change number) is generated.
3. Redo log entries are written to disk.

4. Any acquired locks are released.
5. Transaction is marked as having completed.

Should any of these steps fail, the transaction cannot be committed. Depending on the nature of the error, the transaction will either wait for the problem to be fixed so it can complete the transaction, or it will be rolled back.

The following steps illustrate what must take place if a transaction is rolled back:

1. All changes are rolled back to previous savepoint and savepoint is preserved (or beginning of transaction if no savepoints have been specified).
2. If savepoint is not the last specified savepoint, all savepoints after this one are erased.
3. Acquired locks are released.
4. Transaction continues (if no savepoints were specified then the transaction is ended).

If transaction is ended, rollback segments are released as well, though no SCN is recorded

❑ **Check Your Progress – 3 :**

1. Explain the steps in processing transaction in brief.

.....

.....

.....

.....

.....

9.5 Processing a Remote or Distributed Transaction :

The steps required to process remote and distributed transactions are nearly identical to the way normal transactions are processed. The biggest difference is where the statement is parsed, and the instance whose resources are used for processing.

9.5.1 Entering DDL/DML Statements :

All statements for remote and distributed transactions are entered on a local database or a database where local data resides. It is not necessary to log in to a database where data will be manipulated in order to issue queries against that database, because that is essentially what a remote or distributed transaction is.

9.5.2 Assigning Rollback Segments :

Just as in a normal transaction, if any part of the transaction's statements modifies data on the local database, a rollback segment is assigned to track any changes made to the data.

9.5.3 Breaking down Statements :

Oracle must break down all statements that query or modify remote data in order to send them as a group to the remote database(s). Once they are grouped according to remote database, the statements are sent, via SQL*Net, to their intended destination.

9.5.4 Optimizing Local Statements :

Just as in a normal transaction, the local statements are optimized, based on either the database parameter OPTIMIZER_MODE or the transaction–level parameter OPTIMIZER_GOAL. Once the desired explain plan is created and executed, data is returned and held until all data from remote transactions has been received.

9.5.5 Forwarding Remote Commands :

All remote commands are forwarded to the intended database before they are optimized or parsed. Once the remote database has received the information, it acts identically as it would on a local database: the statement is parsed; the shared SQL area is searched in order to find an identical parsed representation of the statement, the statement is optimized if necessary, and then the statement is executed.

At this point, all data is returned to the local database user or application. If data is to be compared with other data from the local or another remote database, that action takes place on the local database. The local database is responsible for correlating all returned data to provide the final desired output.

9.5.6 Assigning Remote Rollback Segments and Writing :

Redo Logs

All statements that are sent to remote databases to update/manipulate data are assigned a rollback segment on the remote database as they would if manipulating data. The remote database is then responsible for all recovery operations should the database fail or should the transaction require a rollback. Remote transactions function like normal transactions when a commit or rollback statement is issued.

9.5.7 Optimizing Remote Statements :

Statements that are sent to remote databases are not parsed by the local database. This is so that the remote database's shared SQL area can be searched for identical statements. Once the statement is parsed, it is either optimized or the optimized execution plan for the identical statement is used. Data is then returned to the local database.

9.5.8 Returning Data to the Local Database :

As stated earlier, it is the responsibility of the local database, or the database from where the transaction was initiated, to receive data from all remote databases, correlate it, and return only the data that the original statement requested. This can include joins, WHERE and IN clauses, and GROUP BY statements.

9.5.9 Summarizing Remote and Distributed Transactions :

Despite the differences in where the bulk of the transaction processing resides, the steps are much the same. When working in a distributed environment, though, you need to take into account quite a few other steps when dealing with complex updates and error handling. Should a transaction be abnormally terminated or an instance in the distributed environment go down, there are quite a few extra steps needed to help decide whether a distributed transaction should be committed or rolled back. It is better to refer to more in–depth documentation to learn more about two–phase commits and exactly how Oracle

deals with the problems resulting from a downed instance or terminated session from within a distributed environment.

❑ **Check Your Progress – 4 :**

1. Explain the steps in processing a Remote or Distributed transaction

   .....

   .....

2. Collections of operations that form a single logical unit of work are called _____.

   (a) Views (b) Networks (c) Units (d) Transactions

3. Which of the following is a property of transactions ?

   (a) Atomicity (b) Durability

   (c) Isolation (d) All of the mentioned

4. Execution of translation in isolation preserves the _____ of a database

   (a) Atomicity (b) Consistency

   (c) Durability (d) All of the mentioned

5. Which of the following is not a property of a transaction ?

   (a) Atomicity (b) Simplicity (c) Isolation (d) Durability

6. _____ analysis rates the execution plans according to the access paths available.

   (a) Rule based (b) Statement Based

   (c) Temp based (d) Constraint based

9.6 Let Us Sum Up :

Knowing the steps that must be taken to process a transaction can greatly affect how a custom application is developed. If a transaction is querying data only, using a read–only transaction can greatly reduce the amount of processing overhead required to run an application. If many users are running the same read–only query, this savings on overhead can be a considerable amount.

Likewise, knowing more about how statements are optimized can save a great deal of time in reaching the goals set for a new application. Because optimization methods take a critical role in meeting those goals, it is imperative that you take this into account.

Overall, knowing transaction processing steps is just plain helpful for administrators and developers alike.

Understanding how a transaction begins, executes, and ends, and knowing what happens along each step of the way are vital parts of making Oracle work for you. This knowledge is helpful not only to system and database administrators, but to Oracle developers as well. Knowing when a transaction is issued a rollback segment, or how locking occurs in the database can drastically change the strategy of creating applications or nightly processes.

A transaction is directly related to a session, but it is still considered a separate entity. A session, simply stated, is a single connection to a database instance based upon a username and, optionally, a password. All sessions in a database instance are unique, which means that they have a unique identifier setting them apart from the other users and processes accessing the database.

This unique identifier, called a SID, is assigned by the instance and can be reused after a session is ended. The combination of the SID and a session serial number guarantees that each no session, even if the number is reused, is identical.

A transaction, also in simplified terms, is a specific task, or set of tasks, to be executed against the database. Transactions start with an executable DML statement and end when the statement or multiple statements are all either rolled back or committed to the database, or when a DDL (Data Definition Language) statement is issued during the transaction.

It would look like the steps listed in the following sequence of session and transaction begins and ends without save points.

1. Connect to SQL*Plus (begin session #1).
2. Create temporary customer table (begin and end transaction #1).
3. Insert new customer information into temporary table (begin transaction #2).
4. Step through entries in temporary table (continue transaction #2).
5. Update customer table (continue transaction #2).
6. Check for duplicate entries (continue transaction #2).If duplicates exist, roll back entire transaction (end transaction #2).
7. Repeat steps 4–7 until complete or duplicates are found.
8. Drop temporary table (end transaction #2, begin and end transaction #3).
9. Exit from SQL*Plus (end session #1).

Commits, Rollbacks, and Save points

Commits and rollbacks both end a transaction. Commit makes all changes made to the data permanent. Rollback reverses all changes made during the transaction by restoring the previous state of all modified data. With the use of save points, the ROLLBACK command can also be used to roll back only a portion of a transaction.

The following list is an update to the previously shown sequence with the addition of save points. Refer to this example to show how save points affect the transaction.

- Connect to SQL*Plus to (begin session #1).
- Create temporary cusmer table (begin and end transaction #1).
- Insert new customer information into temporary table (begin transaction #2).
- Step through each entry in the temporary table (continue transaction #2).
- Create unique save point (continue transaction #2).
- Update customer table with information from temporary table (continue transaction #2).
- If duplicate customer is found, roll back to save point (continue transaction #2).
- Repeat steps 4–7 until finished.
- Issue commit (end transaction #2).
- Drop temporary table (begin and end transaction #3).

- Exit from SQL*Plus (end session #1).

Transaction Control Statements

The family of transaction control statements include:

- COMMIT
- SAVEPOINT
- ROLLBACK
- SET TRANSACTION

Types of Transactions

Concurrent transactions are transactions that are executed in the same general time. These transactions, because they have started so close to each other, generally do not see the changes made by the other transactions.

A discreet transaction is used to improve the performance of short transactions. For developers creating custom applications, the procedure BEGIN_DISCREET_TRANSACTION() changes the steps followed during the duration of a session in order to speed its processing.

Distributed transactions are transactions in which one or more statements manipulate data on two or more nodes, or remote systems, of a distributed database. If a transaction manipulates data on only one node, it is considered a remote transaction. As in a remote transaction, none of the redo information is stored locally.

An in–doubt transaction is actually a state of a transaction instead of a type and refers to transactions within a distributed database environment. One situation that causes this state is if an instance involved in a currently running transaction fails, that transaction must be either rolled back or committed.

Normal transaction is a term used to refer to a local (non–remote) transaction. All redo information is stored in the local database, and all data manipulation is done to the same database. This type of transaction is the focus for the discussion on transaction processing.

Read–only refers to the type of read consistency that is set or defaulted to for a given transaction. By default, the level of read consistency is statement level, which is also known as read–write. This means that each consecutive statement in your transaction will see the changes made to the database by any previous statements regardless of whose transaction has committed the changes.

Remote transactions are transactions containing single or multiple statement(s) to be executed against a non–local database. These statements all reference the same node. If they do not, they are considered separate remote transactions and the instance will split them up.

Read–consistency is not a difficult concept to grasp. In short, read–consistency guarantees that the data you are viewing while executing a transaction does not change during that transaction. With read–consistency, if two users are updating the same table at the same time, user1 will not see the changes made by the other user during their transaction. User 2, likewise, cannot see any changes committed by user1 until both transactions are complete.

Entering DML/DDL Statements

The issuing of DML or DDL statements can take place through a number of ways, including SQL*Forms, SQL*Plus, and custom C programs. The rules governing the start and end of transactions are the same no matter which way a SQL statement is issued.

Rollback segments are assigned randomly by Oracle at the beginning of each transaction (not session) when the first DML statement is issued, and they are used to roll back the transaction to a specified save point or to the beginning of a transaction. The selection of a rollback segment is based on which rollback segments are currently available and how busy each segment is. By default, DDL statements are not issued rollback segments due to the nature of DDL commands. They are, however, issued redo entries so that the modifications can be reapplied if the database must be recovered.

Using the Optimizer

Oracle's optimizer is a critical part in the execution of a transaction. The optimizer is responsible for taking a SQL statement, identifying the most efficient way of executing the statement, and then returning the data requested. There is a high likelihood that a SQL statement can be executed in more than one way. The optimizer is responsible for identifying the most efficient means of executing that statement.

Cost–based analysis is a mode of analyzing SQL statements to provide the most efficient way of execution. When the optimizer is running in cost–based mode, it follows these steps to decide which plan is the best way to execute the statement unless the developer has provided a hint to use in the execution.

Rule–Based Analysis

Rule–based analysis rates the execution plans according to the access paths available. The rule–based approach uses those rankings to provide an overall rating on the execution plan and uses the plan with the lowest ranking.

Handling Locks

The locking of data rows and/or tables is completely automated and transparent to the user. Once the executable version of the SQL statement is run, Oracle automatically attempts to lock data at the lowest level required. This means that if possible, a row will be locked instead of the entire table. This is dependent solely on how the SQL statement was written and what types of access are required (full table scan versus single rows).

Stepping Through the Transaction –

Following are the steps taken during a commit.

- Instance's transaction table marks transaction as complete.
- A unique SCN (system change number) is generated.
- Redo log entries are written to disk.
- Any acquired locks are released.

Transaction is marked as having completed.

9.7 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 9.2

- **Check Your Progress 2 :**

  See Section 9.3

- **Check Your Progress 3 :**

  See Section 9.4

- **Check Your Progress 4 :**

  1 : See Section 9.5 **2 :** Transactions

  3 : All of the mentioned **4 :** Consistency

  5 : Simplicity **6 :** Rule based

9.8 Glossary :

1. **Transaction –** In simplified terms, it is a specific task, or set of tasks, to be executed against the database
2. **Commit –** The COMMIT command is the transactional command used to save changes invoked by a transaction to the database.
3. **Rollback –** The ROLLBACK command is the transactional command used to undo transactions that have not already been saved to the database.
4. **Save point –** A SAVEPOINT is a point in a transaction when you can roll the transaction back to a certain point without rolling back the entire transaction.
5. **Set Transaction –** The Set Transaction command can be used to initiate a database transaction.
6. **Concurrent transactions –** These are transactions that are executed in the same general time.
7. **Discreet Transaction –** It is used to improve the performance of short transactions.
8. **Distributed transactions –** These are transactions in which one or more statements manipulate data on two or more nodes, or remote systems, of a distributed database
9. **Remote transactions –** These are transactions containing single or multiple statement(s) to be executed against a non–local database
10. **Execution plan –** The steps used to execute the SQL statement are called an execution plan
11. **Cost–based analysis –** It is a mode of analyzing SQL statements to provide the most efficient way of execution.
12. **Rule–based analysis –** It rates the execution plans according to the access paths available and the information.
13. **Locking –** Reserving the use of a database, table, record, or other collection of data for access by one user at any one time to prevent the resulting anomalous database behavior when, for example, one user reads or modifies data in a record which is in the process of being modified by another user.

14. **Read Lock –** An imposed access barrier that allows a user to query and view all or part of a database, but not to modify it. Some database management systems will allow many users to have simultaneous read locks.

15. **Write Lock –** A lock that allows the user to read and to modify the database. Write locks almost always imply exclusive control of the database so that other users will not be allowed to have either active read or write locks as long as one write lock is operational.

9.9 Assignment :

State the difference between Data processing and Transaction Processing.

9.10 Activities :

Explain the use of optimizer in execution of transaction.

9.11 Case Study :

Discuss the transaction control statements with example.

9.12 Further Reading :

1. Database Management Systems – Rajesh Narang – PHI Learning Pvt. Ltd.
2. Database System Concepts by Silberschatz, Korth – Tata McGraw –Hill Publication
3. An Introduction to Database Systems – Bipin Desai – Galgotia Publication
4. Database Management System by Raghu Ramkrishnan – Tata McGraw–Hill Publication
5. SQL, PL/SQL: The Programming Language Oracle – Ivan Bayross – BPB Publication.

Unit 10

OBJECT ORIENTED DATABASE MANAGEMENT SYSTEMS (OODBMS)

UNIT STRUCTURE

10.0 Learning Objectives

10.1 Introduction

10.2 Introduction to Database Management Systems (DBMS)

10.3 Example of Bank Transactions

10.4 Object Oriented Database (OODB)

10.5 Related terms

- 10.5.1 Distributed Object Computing (DOC)
- 10.5.2 Objects Methods Users
- 10.5.3 Interfaces
- 10.5.4 Associations
- 10.5.5 Persistent Objects
- 10.5.6 Persistence Data
- 10.5.7 Transient Data
- 10.5.8 Referential Integrity
- 10.5.9 MDBS
- 10.5.10 ODBC (Open Database Connectivity)
- 10.5.11 Locks
- 10.5.12 ActiveX
- 10.5.13 OOSAD
- 10.5.14 CORBA
- 10.5.15 DCOM
- 10.5.16 OMG
- 10.5.17 CORBA Open DOC ActiveX
- 10.5.18 Virtual DBMS

10.6 Object Oriented Database Management Systems (OODBMS)

10.7 Comparison between RDBMS and OODBMS

10.8 A Three Schema Architecture

10.9 Mapping of OODBMS to RDBMS

10.10 Example of Railway Reservation System

10.11 Let Us Sum Up

10.12 Suggested Answer for Check Your Progress

10.13 Glossary

10.14 Assignment

10.15 Activities

10.16 Case Study

10.17 Further Readings

10.0 Learning Objectives :

After learning this unit, you will be able to understand :

- Basic concepts of Database, DBMS, RDBMS and OODBMS
- Use of OO concepts in database
- Practical applications such as bank transactions, railway reservations
- Principle of mapping of OODBMS to RDBMS
- Difference between OODBMS and RDBMS
- A three schema architecture
- Basic Concept of Persistent Object
- Basic Concept of Transient Object

10.1 Introduction :

Relational database systems have proved their worth in the domain of business

Applications, particularly those dealing with accounting. The relational data model, however, is not suitable for all application domains. New applications involving complex data modeling (i.e. that do not map well to tables) now require the services normally associated with database systems: persistence, transactions, authorization, distribution, versioning, data stability, buffering, and associative access.

To illustrate, let's examine a CAD/CAM application for a company that manufactures airplanes. The application supports both the specification and design of all parts required to build an airplane. Modeling physical objects does not reduce easily to tabular, or relational, form. In particular, an airplane requires many duplicate parts, each of which would require a unique tag to be stored as a distinct entity in a relational database. Furthermore, the relations representing sets of different parts that are mostly similar would require separate, independent schemas. Finally, the application programmer almost definitely would prefer to manipulate part designs as complex abstractions at a level higher than that provided in the relational model.

Our example application, however, requires database services. An airplane design team typically consists of several people, all of whom will desire access to the current state of the design. In today's workplace, it is likely that these designers will be using workstations distributed over a network. In addition, some people should not be allowed total access to certain aspects of the design (e.g. documenters do not need update access). Finally, a completed design can involve hundreds of thousands of parts and direct access to each part becomes impractical: thus, associative access is essential. For instance, a designer may wish to know how many times a given part has been used before deciding to change its specification.

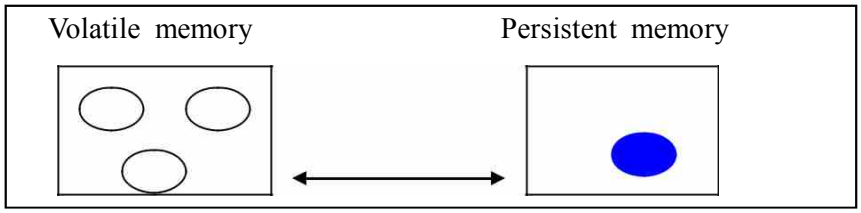
Object–oriented databases, then, are an attempt to solve the problems mentioned and still maintain the advantages of database systems. Object–oriented databases treat each entity as a distinct object. An assembly composed of several parts, therefore, can refer directly to its components instead of explicitly associating some unique identifier with each component in some relation. In addition, application programmers can manipulate database entities at any desired level of abstraction by extending the set of types recognized

by the database system. This is an important point – it means that the programmer need not be concerned with transforming an application's persistent data into a form manipulable by the underlying storage subsystem. In many systems, a programmer can also incorporate totally new, variable–sized data types (e.g. multimedia objects). Finally, object–oriented databases allow embedded semantics by associating procedural information with objects.

10.2 Introduction to Database Management Systems (DBMS) :

The primary memory of computer has a limited capacity. Therefore, for storage a large amount of data (database), a secondary memory is used. A secondary storage provides efficient ways to access objects.

The second requirement is of persistence. The object store stores objects that have ability to survive even though the program or system that created them is no longer live. Such objects are called Persistent Objects. Persistence often means that objects are copied from fast and volatile memory to a slow and persistent secondary memory.



volatile memory persistent memory

The object is copied from/to the volatile memory to/from the persistent memory.

The data that has no value once the process that creates it is no more in existence is called Transient data. Variables allocated dynamically belong to Transient data. The data once created by a process survives for as long as it is created externally, is called persistence data. The data that exists due to its need for ever is categorized under persistence data.

The end User of database is not worried about how the data is stored in the memory. Therefore, we make use of database management system (DBMS). The application programmer is only interested in a logical view of the database. He may not be interested to take part in decisions about how the physical storage is done.

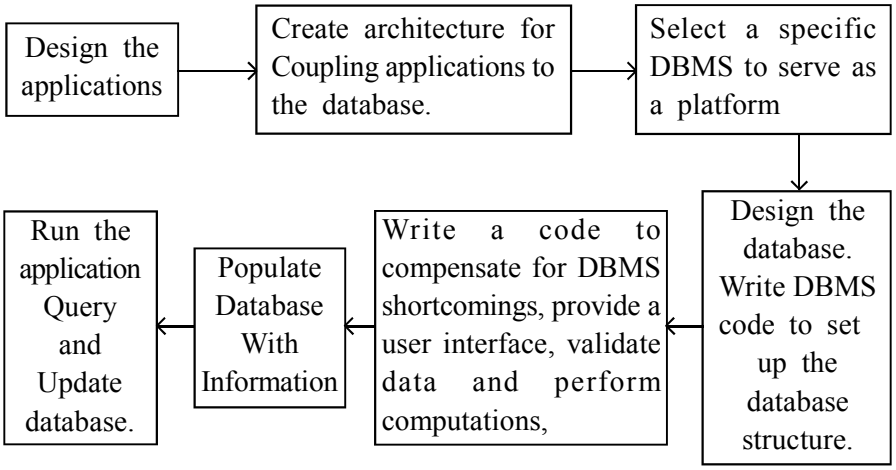
Instead of using DBMS we can also create data files for our applications.

This is, however, laborious and time consuming.

Database specialization is an integration of DBMS product in the development of the system. The specialization of OOSE depends on type of DBMS used. According to Jacobson and others, the properties that indicate a need for a DBMS are as stated below –

- Information that needs to be persistent.
- More than one application sharing of the data.
- Information structure with large number of instances.
- Complex searching in the information structure.
- Advanced generation of reports from stored information.
- Handling of user transactions.
- A log for system restart.

The steps in life cycle of database applications are shown in Figure below



life cycle of database applications

The main and perhaps difficult part is a database design. The database design is often called as a data model or schema.

Broadly, there are two approaches of database design –

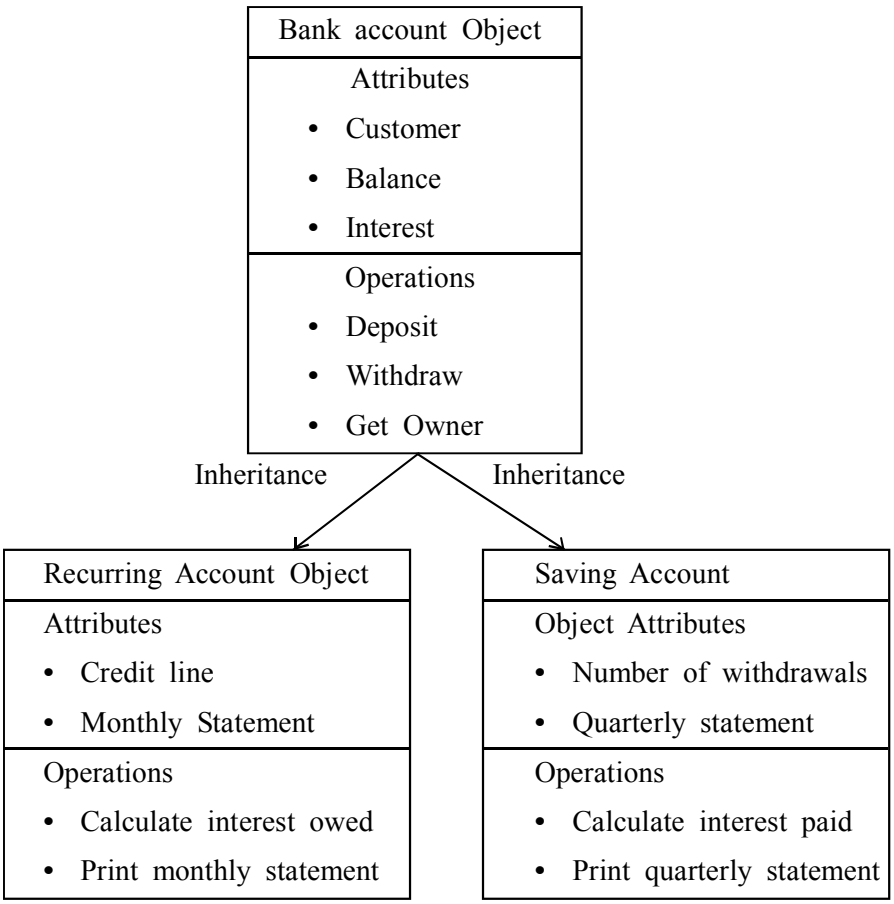
(1) Attribute driven approach – In this approach, a list of attributes relevant to the application is compiled. Synthesize the groups of attributes that preserve functional entities.

(2) Entity driven approach – In this approach the entities meaningful to the application are discovered. The object modeling is in form of entity design. Entity design is more tractable.

10.3 Example of Bank Transactions :

You have studied that the object consists of data values describing attributes of an entity, plus operations that can be performed upon data. This encapsulation capacity allows OO model to better handle more complex types of data (graphs, pictures, voice, text etc.) than the other database structure. The OO model also supports inheritance i.e. new objects can be automatically created by replicating some or all of characteristics of one or more parent objects.

As shown in Figure both recurring and saving accounts inherit common attributes and operations of parent bank account object. Such capability has made OODBMS popular. For example, OOT allows designer to develop product design, store them as objects in OODB and replicate and modify them to create a new product design.



Use of OODBMS in bank application

You have already understood the basic concept of RDBMS. The relational data model was invented by E. F. Codd. As you have studied, the concept of RDBMS is based on a table. RDBMS is a computer program for managing these tables.

According to Codd, the RDBMS can be defined in three major parts:

- Data that is presented as tables.
- Operations for manipulating tables.
- Integrity rules on tables.

The mismatch between application objects and relational data needs to be mapped for use in the application. Mapping of multiple inheritance into a single table is possible in RDBMS.

The schemas are moved in Table with columns indicating data type and row containing data. Relational RDBMS is built on schema representing complex relationships between entities. Mapping of multiple inheritance into a single table is possible in RDBMS.

Since there is heterogeneity in computing environment platform, programming languages and different application architecture, we are living in a non–standard environment. In real world, we have to source data from multiple data sources that have different data models. When the data have a complex structure, the operation to deliver customer requirements becomes complex to design and deliver. The engineering objects data set is very complex. It requires complex modeling to represent deep and multidimensional structures. Every new requirement calls for the ability to define, create, modify and delete the

existing data structure. The following parameters are empowered with authorization capabilities to perform only certain operations.

Objects Systems Users

The company may have functional databases such as BOM (Bill Of Material), drawings, equipment. These databases are heterogeneous in nature and are distributed physically, logically and by responsibility of control

❑ **Check Your Progress – 1 :**

1. Why there is need of OODBMS ?

.....

.....

.....

.....

.....

10.4 Object Oriented Database (OODB) :

Object oriented databases defined as databases that integrate object orientation with database capabilities. Object orientation allows a more direct representation and modeling of real world problems, and database functionality is needed to ensure persistence and concurrent sharing of information in applications.

Object databases store objects rather than data such as integers, strings or real numbers. Objects basically consist of the following :

- **Attributes –** Attributes are data which defines the characteristics of an object. This data may be simple such as integers, strings, and real numbers or it may be a reference to a complex object.
- **Methods** – Methods define the behavior of an object and are what was formally called procedures or functions.

Therefore objects contain both executable code and data. Classes are used in object oriented programming to define the data and methods the object will contain. The class is like a template to the object. The class does not itself contain data or methods but defines the data and methods contained in the object. The class is used to create (instantiate) the object. Classes may be used in object databases to recreate parts of the object that may not actually be stored in the database. Methods may not be stored in the database and may be recreated by using a class.

The complex data structures are handled effectively by OODB. In order to tackle the problems related with complex data structures, OODB makes use of what is called as PDM (Product Data Management). Change in data item should make relevant changes in overall database. This facility is provided by OODB. Large complex systems requiring complex data support are handled through distributed data sources or Databases. The process of mapping and integrating begins with defining the relationships between the table structure in RDB and class structure in object model in OODB.

In the real world, both the applications exist, namely OO applications and RDBMS applications with first generation client server computing.

Client applications accessing OODB do not have to specify relationships because objects are stored with established relationships.

OODBs store and deliver objects and provide facilities to manipulate objects.

When we need to create multiple versions of entity, OODB becomes very handy.

OODB can be a solution to handle complex data structures.

The strength of OODB is best seen when:

- Data and relationships are irregular and complex
- Objects are tightly coupled
- Non–DBMS data sources and applications are integrated

OODB should meet following requirements to design a system in OO technology.

1. User interface support
2. Change management
3. Object translation into various activities.
4. OODB security
5. OODB recovery
6. Accessibility to legacy systems data
7. Maintenance of class libraries
8. Maintenance of data dictionary
9. OODB integrity
10. Providing concurrent access by multiple users
11. Distribute database with location transparency
12. Handling query with efficiency
13. C++ based OO model
14. Maintaining persistent objects
15. Object translation into various activities.

The user must have access to both OODB and RDB to manipulate data. The developer therefore must develop applications that could source data from all databases (OODB, RDB etc.).

❑ Check Your Progress – 2 :

1. What are requirements OODB should meet to design a system in OO technology ?

.....

.....

.....

.....

.....

10.5 Related Terms :

10.5.1 Distributed Object Computing (DOC) :

In first generation client–server computing, methods/applications were available to serve on demand at server side. Having independent locations, the objects need to communicate with each other for processing and computing. This is called second generation client–server computing.

Having independent locations, the objects need to communicate with each other for processing and computing. This is called second generation client–server computing.

In first generation client–server computing, methods/applications were available to serve on demand at server side.

Since there is so much heterogeneity distributed in the network in terms of servers, clients, databases, platforms, applications and architecture, which cannot be easily dispended with, what is needed is therefore standard DOC system.

The second generation client–server computing is based on distributed object computing (DOC). DOC enables the development of extremely flexible client–server systems as it is possible to locate reusable objects/components stored anywhere in the network and manipulate them as per the application requirements. DOC introduces a higher level of abstraction of the application.

10.5.2 Objects Methods Users :

With the advent of DOC, it is now possible to use following components anywhere in the network and running on different platforms.

a. Reusable Components

b. Distributed Components

c. Resident Components

10.5.3 Interfaces :

In the normal course, the developer would need to write interfaces for each method or server application whatever resident.

10.5.4 Associations :

Associations show logical link between objects.

The cardinality of association is specified when the class is defined. The links are expressed in cardinality of associations.

10.5.5 Persistent Objects :

The object store stores objects that have ability to survive even though the program or system that created them is no longer live. Such objects are called Persistent Objects.

Persistent Object Storage System under DBMS can provide the following:

- Unique ID to objects so as to reach correctly
- Communication between objects
- Large, stable and reliable storage capacity

The stores for persistent objects, such as disk files do not support queries or interactive user interface operations. In such stores following problems are seen :

- Controlling concurrent access
- Handling ad–hoc queries
- Controlling physical locals of data entry

The stores for persistent objects, such as disk files do not support queries or interactive user interface operations. These problems are solved by DBMS.

10.5.6 Persistence Data :

The data once created by a process survives for long till it is created externally, is called persistence data.

The data that exists due to its need for ever is categorized under persistence data.

10.5.7 Transient Data :

The data that has no value once the process that creates it is no more in existence is called Transient data. Variables allocated dynamically belong to Transient data. The following are some examples of transient data:

- Discounts calculated to arrive at net price
- Intermediate values computed in discount algorithms
- Data extracted from external sources for reference or use.

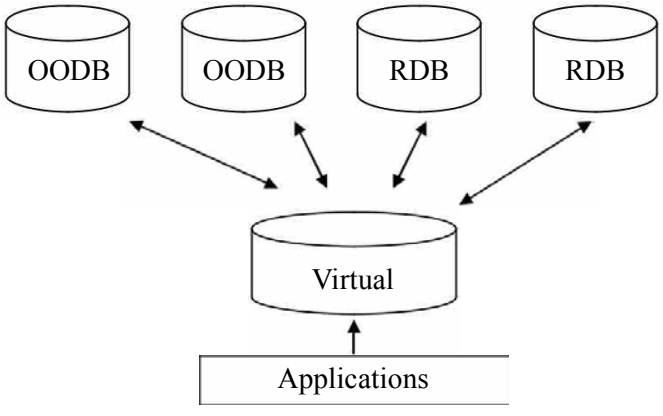
10.5.8 Referential Integrity :

The integrity that guarantees existence of all objects that need to be referred is known as Referential Integrity

10.5.9 MDBS :

The Multi–database (MDBS) is a database system that sits on the top of existing relational and OODBMS but gives impression of being a single database to the users.

MDBS provides diverse set of interfaces to reach data stored at any database and then manipulates it to produce the result. MDBS helps to integrate homogeneous database system with multiple database and data types other than relational data types. MDBS handles cross database functionality through virtual database, by constructing unified schemes. MDBS coordinates the processing of global transactions split into various local databases to achieve consistency and integrity of data and results. MDBS translates global queries and updates data sending it to the appropriate local database for actual processing. Automatic generation of unified global database using a local database (OODB, RDB, file system) is done by MDBS. Fig. shows virtual database model.



MDBS virtual database model

10.5.10 ODBC (Open Database Connectivity) :

An API (Application Programming Interface) that provides solution to reach the intended database is called as ODBC.ODBC is developed by Microsoft Corporation

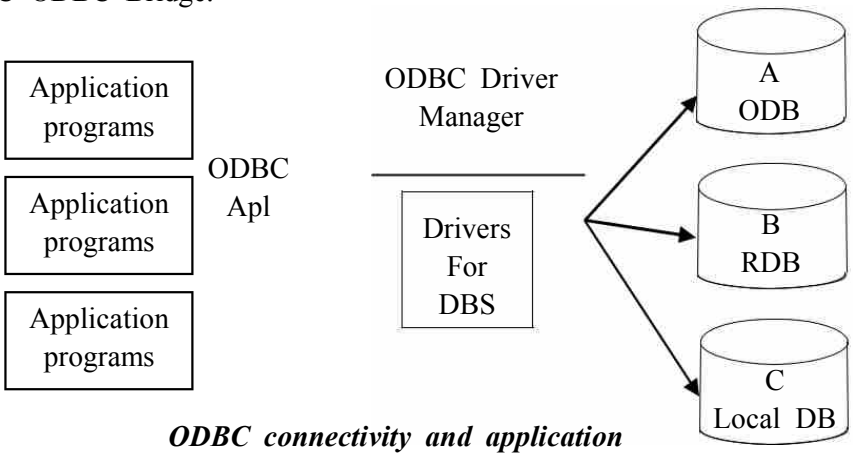
The ODBC driver manager manages an application's data needs by mapping data requirements to data source in the database.

OODBMS provides all required advanced features in addition to all features of RDBMS. ODBC with API provides standard database access through a common interface, independent of application. With its strength, ODBC has become an industry standard for inter–operatability on different databases.

The ODBC–API driver drives to reach to intended database, runs the application, manipulates the data and puts it back in position as the new data image.

Developers write applications using different databases. In order to interconnect these databases, developer has to write a driver program called ODBC driver.

JDBC (Java database Connectivity) and ODBC can be combined to form a JDBC–ODBC Bridge.



ODBC connectivity and application

10.5.11 Locks :

Concurrency control and synchronization of different processes is achieved by using locks. Following types of locks are available:

- **Read Lock –** No other transaction is committed till the earlier transaction completes the task
- **Writes Lock –** The lock that ensures that no other transaction is committed till the transaction is in the mode of updating and creation of new version of object is called Write Lock.
- **Null lock –** The lock in which objects are managed in a cache is called null lock.
- **Notify lock –** Cause interruption of the program when a new value is committed in OODB.

Null and Notify locks are generally used outside transaction boundary in a cache. On other hand, Read and Write locks are generally used inside transaction boundary.

10.5.12 ActiveX :

ActiveX controls are third party controls. For example, if you are using Visual C++, you can import. doc files from MS Word. ActiveX is also known as OLE (Object Linking and embedding). The ActiveX technology is developed by Microsoft Corporation.

10.5.13 OOSAD :

When we move to OOSAD, server–resident applications will give way to objects and distributed objects in distributed server environment.

10.5.14 CORBA :

The advantage of CORBA is that it allows developer through Interface Definition language (IDL) language to specify language neutral object–oriented interfaces for applications and their components.

CORBA provides a communication channel through which applications can access object interfaces and request data and services.

Microsoft's Component Object Model (COM) and Distributed Component Object Model (DCOM) are alternatives to CORBA.

10.5.15 DCOM :

DCOM is the internet and component strategy where ActiveX plays a role of DCOM object. An easy passage to distributed object computing technology such as CORBA and DCOM is provided by ALC (Access Layer Classes).

DCOM is supported by Microsoft's web browser Internet Explorer.

Since there is heterogeneity in computing environment platform, programming languages and different application architecture, we are living in a non–standard environment.

10.5.16 OMG :

The Standard DOC system was developed by the organization of about 500 companies named Object Management Group (OMG).

The Object Management Group (OMG) suggested an abstract object model with following features :

- Object can be distinguished clearly.
- The object methods and operations are governed by parameters and their values.
- The object's nature is persistent or transient.
- An object at any time has state
- An object has set of operations
- Object provides services to other clients.

Many organizations are now adopting OMG's Object Request Broker Architecture Standard for integrating.

- Distributed business applications
- Heterogeneous business applications
- Business data types

Object Management Group (OMG) specifies an architecture for Open Software Bus.

At present, following are the computing standards of OSB (Open Software Bus).

10.5.17 CORBA Open DOC ActiveX :

Open Software Bus provides a platform for objects components to operate across the networks and on different operating systems.

10.5.18 Virtual DBMS :

In multi–database environment when we are dealing with multiple heterogeneous databases for application processing we need an interface to access the Virtual DBMS.

Access to the Virtual DBMS can be provided by API. API is used by developers to interface the application with the database.

❑ **Check Your Progress – 3 :**

1. Explain the terms like Persistent objects, Persistent data, Transient Data
2. Explain terms ODBC, MDBS,DCOM, CORBA

   .....

   .....
3. Full form of CORBA
4. Full Form of ODBC
5. Full form of OMG

10.6 Object Oriented Database Management Systems (OODBMS) :

OODBMS can handle a wide range of data types supporting complex data and structure. OODBMS is a result of blending OOP and database technology to meet the application needs of systems defined in OOT. OODBMS has more or less the same features of DBMS, but in addition, is in position to handle OO features.

The row of table in RDBMS contains data which maps instance of class in OODBMS. The corresponding equivalent of table in OODBMS is class with attributes and methods. OODBMS scores over pure relational database as it gives benefit of object orientation. Since OODBMS supports distribution of objects, it must have a feature of location transparency so that OO application need not take efforts to add mechanisms to local objects.

OO systems while in operation generate large amount of data. Each item of such data remains 'alive' during certain time and provides some service during that period.

Besides finding where the object is in distributed environment, OODBMS is able to move an object from one location to the other for optimum distribution of load.

OODBMS provides such security features that protect operations and objects from illegal actions from unauthorized quarters.

OODBMS ensures that in the event of failure the recovery process enforces consistency in database status after a transaction is terminated by force externally.

The key feature that enables users/application to cooperate and collaborate in application processing is named as concurrency.

OODBMS must be accessible by multiple users. That is if users/application access a selection of database, other user/application should also be in a position to access other sections of database.

The OODBMS recovery process ensures that the system failure does not result in an inconsistent database state.

The significant features of OODBMS are as follows:

- OODBMS has a feature of managing traffic of persistent objects.
- OODBMS ensures that when a pointer refers to an object, the object exists
- OODBMS manages logical link association and its cardinality
- OODBMS can work on following Operating systems on server :– MS Windows, OS/2, Sun OS In context of transactions, OODBMS
  a. Handles nested transactions
  b. Supports MROW
  c. Supports long transactions.
- Access to other OODBMS comprises of
  a. Read data resident in other OODB
  b. Modify data resident in other OODB
  c. Read data resident on RDBMS
- ODBMS Queries comprise of
  a. Ad–hoc queries with C++
  b. Ad–hoc queries with 4GL
  c. Ad–hoc queries with LISP
- O2, Objectivity, Object Store and Versant are some of the Object Oriented Database produces
- All OODBMS products namely O2, Objectivity, Object Store and Versant support following DBMS criteria
  a. SUN OS
  b. AIX
  c. MS Windows
- All OODBMS products namely O2, Objectivity, Object Store and Versant support following DBMS criteria
  a. single–user, single–tasking environment
  b. single–user, multi–tasking environment
  c. multi–user environment
- All OODBMS products namely O2, Objectivity, Object Store and Versant support following criteria for access to other DBMS whenever an application is running on the OODBMS
  a. can read data that resides in other OODBMS
  b. can modify data that resides in other OODBMS
  c. can read data that resides in RDBMS ORACLE

- All OODBMS products namely O2, Objectivity, Object Store and Versant support following DBMS criteria:
  a. embedded queries with a 4GL
  b. ad–hoc updates of DB–schema with GUI
  c. ad–hoc updates of DB–schema with OOL.
- All OODBMS products namely O2, Objectivity, Object Store and Versant support ad–hoc queries with the following
  a. GUI
  b. 4GL
  c. C++
- All OODBMS products namely O2, Objectivity, Object Store and Versant support following DBMS criteria
  a. OQL
  b. ODMG C++ binding
  c. Smalltalk binding
- All OODBMS products namely O2, Objectivity, Object Store and
- Versant support application programming in following languages.
  a. C++
  b. JAVA
  c. SMALLTALK
- All OODBMS products namely O2, Objectivity, Object Store and Versant support following DBMS criteria
  a. replication of data
  b. data encryption
  c. database language based on SQL
- In OODBMS product O2, the attributes of objects have to be defined in following languages.
  a. C++
  b. JAVA
  c. SMALLTALK
- All OODBMS products namely O2, Objectivity, Object Store and Versant support following DBMS criteria
  a. multiple inheritance
  b. concept of version
  c. checking of cardinality between objects
- All OODBMS products namely O2, Objectivity, Object Store and
- Versant support following DBMS criteria
  a. user defined data type
  b. is a relationship
  c. part of relationship

- OODBMS supports comprise of
  a. user defined data type
  b. is a relationship
  c. part of relationship
- OODBMS standards comprise of
  a. ODL
  b. OQL
  c. ODMG C++ binding
- Following are the OO concepts managed by OODBMS
  a. Complex objects storage structures
  b. Query processing & displaying results
  c. Concurrency control on multiple accesses
- Some parameters of OODBMS

Parameter	Example
OODBMS standards	ODL, OQL, ODMG C++ binding, standard SQL in interactive and embedded mode
OODBMS supports	user derived datatype, is_a relationship, part_of relationship
Queries	Ad–hoc queries with 4GL, Lisp, C++
Accesstoother OODBMS	Read/Modify data in other OODB

❑ **Check Your Progress – 4 :**

1. Explain the features of OODBMS.

.....

.....

.....

.....

.....

10.7 Comparison between RDBMS and OODBMS :

There are certain significant benefits of OODBMS over RDBMS. This can be understood from following comparison :

RDBMS	OODBMS
Not suitable for engineering, manufacturing, office automation and complex MIS systems	Highly suitable for engineering, manufacturing, office automation and complex MIS systems
Data access performance is normal	Significant improvement in data access performance
There is no explicit management of inheritance of attributes and methods	Inheritance property represents relations explicitly

Records in RDBMS are passive	Records in OODBMS are active
Cannot handle large number of data types, relationships and behaviors	Handles large number of data types, relationships and behaviors.
Poor in navigational and associative access to information	Supports navigational and associative access to information
Data representation has no character and no identity in itself	Data has character
RDBMS does not cover advanced features of OODBMS.	OODBMS provides all required advanced features in addition to all features of RDBMS

❑ **Check Your Progress – 5 :**

1. Differentiate RDBMS and OODBMS

.....

.....

.....

.....

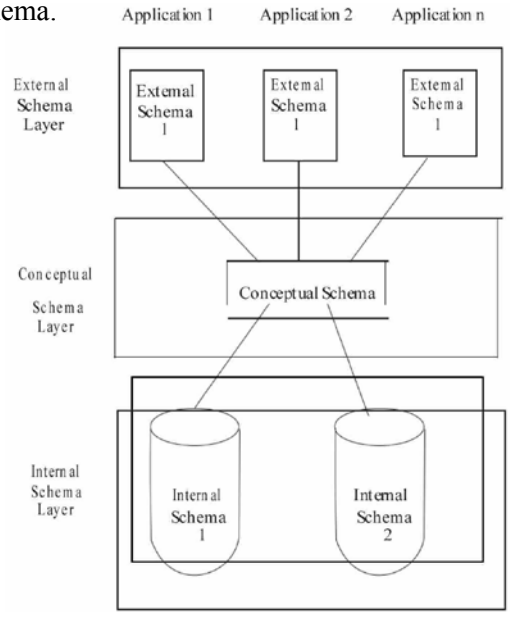
.....

10.8 A Three Schema Architecture :

Fig. depicts a three schema architecture. It is a standard architecture for related database applications. Earlier, this architecture was proposed by the ANSI/SPARC committee on DBMS.

As shown in the figure, there are three layers in this design – external, conceptual and internal schemas.

External schema layer comprises of number of external schemas viz. external schema1, external schema2 …….. external scheman etc. Each external schema is a database design form, a perspective of single application. The external schema is a view or abstraction of the global, overall conceptual schema. The external schema isolates applications from most changes in the conceptual schema.



Three Schema Structure for ANSI / SPAPC

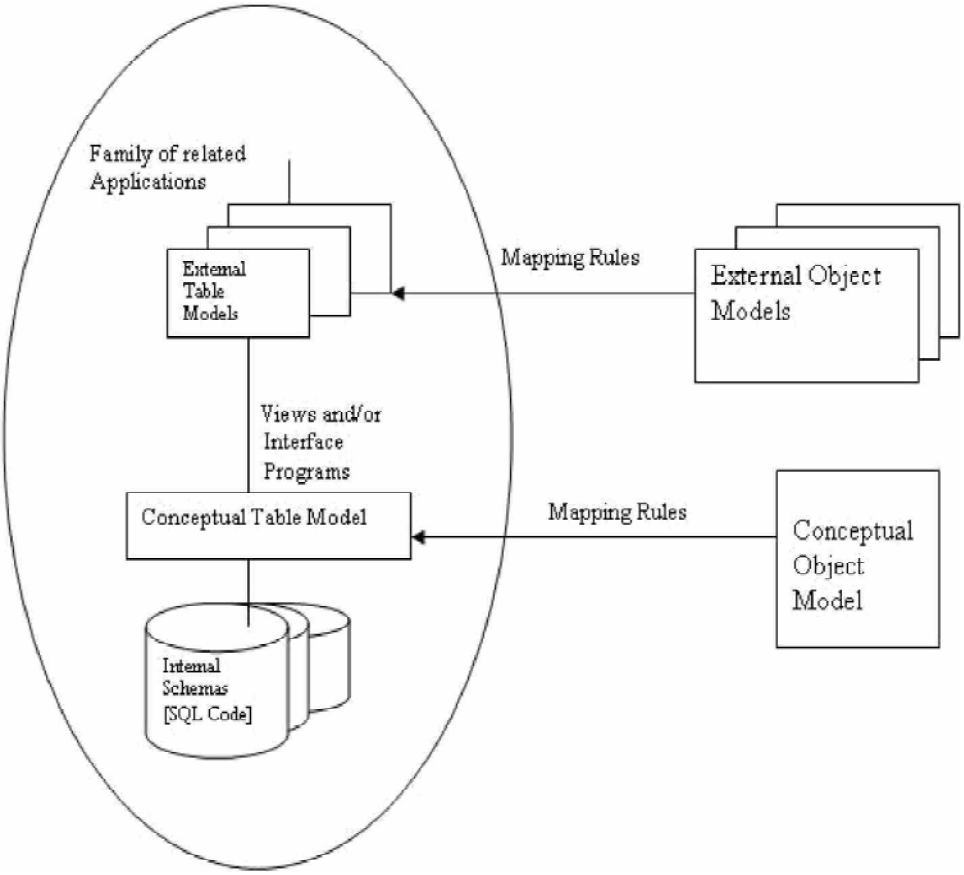
The conceptual schema layer integrates internal and external schemas.

The internal schema layer consists of actual DBMS code required to implement the conceptual schema.

The object modeling can be applied for designing both, internal and external schemas.

Figure shows how object modeling is related to the three schema architecture.

First of all, you have to formulate object models for external and internal schemas. Next, translate each object to a table model. As shown in the figure, views and/or interface programs connect external table to conceptual tables. Conceptual tables convert to internal schema.



Three Schema Architecture and object modeling

The object model comprises of classes, associations, generalizations and attributes. The object models emphasize on logical data structures. Each table model comprises of number of ideal tables. In order to translate from object model to ideal table, a number of alternatives are available.

The internal schema comprises of SQL commands that create attributes, tables, indexes etc.

❑ Check Your Progress – 6 :

1. Explain in brief how object modelling is related to the three schema architecture.

.....

.....

.....

10.9 Mapping of OODBMS to RDBMS :

The RDBMS is built on schema representing complex relationships between entities. The schemas are then translated into a table. Under the context of OO Systems, the class is mapped into tables in the following manner as per approach given by Jacobson and others.

1. Assign one table for each class.

   Assign attribute as a column in the table. The attributes are primitive. The columns of table usually indicate data type and its name.

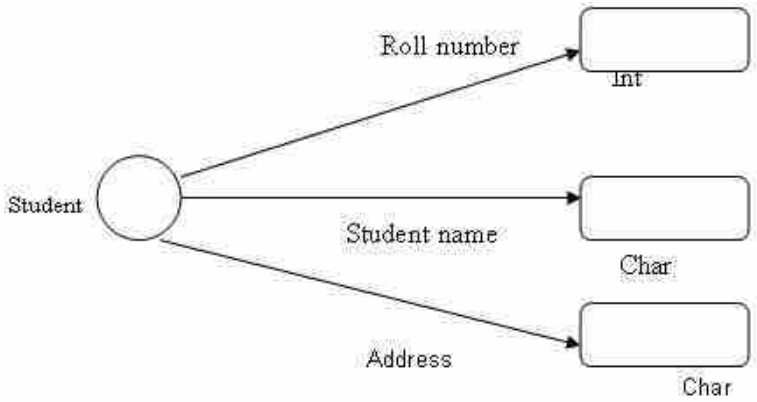
   In case of complex attributes additional table is required to be created.

2. The column representing the primary key will be a unique instance identifier. Using primary key column, the instance is uniquely recognized.

3. Each instance of the class will now be represented by a row in this table. The row of a table contains data for a single entity. This maps with instance of class in OODBMS.

4. Each acquaintance association with cardinality greater than 1 will become a new table. This new table will connect the tables representing the objects that are to be associated.

This can be described with the example of 'student '. The class or object 'student' has the attributes such as roll_number, student_name and address. The object 'student' and its attributes are shown in



Object 'Student' and its attitudes

The object 'student' and its attributes can be transformed into a table as follows :

Roll Number	Student Name	Address
10	S.A. Patil	1102, Sadashiv Peth, Pune
50	N.M. Joshi	102, M.G. Road, Pune
456	P.G. Deshpande	119, Narayan Peth, Pune
7890	M.R. Shinde	234, Parvati, Pune

Primary and secondary keys –

Each table has one primary key. A primary key is the combination of one or more attributes whose value unambiguously locates each row in a table. In the following table primary key of student table is roll_number. College_ID is the primary key of college table.

Student Table

Roll Number	Student Name	Address	College_Id
10	S.A. Patil	1102, Sadashiv Peth, Pune	1001
50	N.M. Joshi	102, M.G. Road, Pune	1002
456	P.G. Deshpande	119, Narayan Peth, Pune	1001
7890	M.R. Shinde	234, Parvati, Pune	1005

College Table

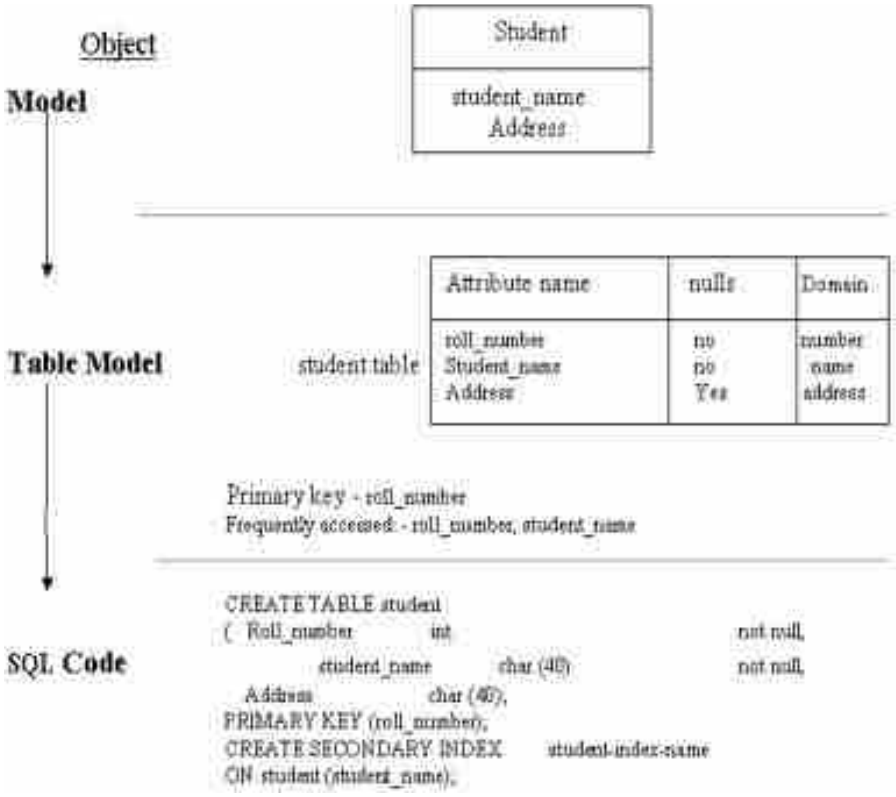
College_ID	College Name	Address
1001	National College	123, M.G. Road
1002	Model College	456, ShanwarPeth
1005	City College	678, Karvenagar

Roll number is a primary key of Student Table

College_ID is a primary key of College Table

College_ID is a foreign key of Student Table

The conversion of object class to table is shown in below figure. Class student has attributes student name and address as discussed above. These attributes are listed by table model and added to implicit object ID. It is required to specify that the field roll_number cannot be null since it is a primary key. Similarly, student_name also cannot be null. The name must be entered for every student. However, the student_name is not a primary key because two or more students may have the same name. The attribute 'address' can be null. The domain is assigned to each attribute. The primary key is specified for each table. The SQL code is used to create Student table. The index on student_name enables quick retrieval for this attribute since it is frequently accessed. The SQL also maps domain to data types.



Example of mapping from class to table

Advantages of OODBMS :

- **Enriched modeling capabilities –** The object–oriented data model allows the 'real world' to be modeled more closely. The object, which encapsulates both state and behavior, is a more natural and realistic representation of real–world objects. An object can store all the relationships it has with other objects, including many–to–many relationships, and objects can be formed into complex objects that the traditional data models cannot cope with easily.
- **Extensibility –** OODBMSs allow new data types to be built from existing types. The ability to factor out common properties of several classes and form them into a superclass that can be shared with subclasses can greatly reduce redundancy within system is regarded as one of the main advantages of object orientation. Further, the reusability of classes promotes faster development and easier maintenance of the database and its applications.

  Capable of handling a large variety of data types Unlike traditional databases (such as hierarchical, network or relational), the object oriented database are capable of storing different types of data, for example, pictures, voice video, including text, numbers and so on.
- **Removal of impedance mismatch –** A single language interface between the Data Manipulation Language (DML) and the programming language overcomes the impedance mismatch. This eliminates many of the efficiencies that occur in mapping a declarative language such as SQL to an imperative 'language such as 'C'. Most OODBMSs provide a DML that is computationally complete compared with SQL, the 'standard language of RDBMS.
- **More expressive query language –** Navigational access from the object is the most common form of data access in an OODBMS. This is in contrast to the associative access of SQL (that is, declarative statements with selection based on one or more predicates). Navigational access is more suitable for handling parts explosion, recursive queries, and so on.
- **Support for schema evolution –** The tight coupling between data and applications in an OODBMS makes schema evolution more feasible.
- **Support for long –** duration, transactions: Current relational DBMSs enforce serializability on concurrent transactions to maintain database consistency. OODBMSs use a different protocol to handle the types of long–duration transaction that are common in many advanced database application.
- **Applicability to advanced database applications –** There are many areas where traditional DBMSs have not been particularly successful, such as, Computer–Aided Design (CAD), Computer–Aided Software Engineering (CASE), Office Information System (OIS), and Multimedia Systems. The enriched modeling capabilities of OODBMSs have made them suitable for these applications.
- **Improved performance –** There have been a number of benchmarks that have suggested OODBMSs provide significant performance improvements over relational DBMSs. The results showed an average 30–fold performance improvement for the OODBMS over the RDBMS.

Disadvantages of OODBMS :

- **Lack of universal data model –** There is no universally agreed data model for an OODBMS, and most models lack a theoretical foundation. This disadvantage is seen as a significant drawback, and is comparable to pre–relational systems.
- **Lack of experience –** In comparison to RDBMSs the use of OODBMS is still relatively limited. This means that we do not yet have the level of experience that we have with traditional systems. OODBMSs are still very much geared towards the programmer, rather than the naïve end–user. Also there is a resistance to the acceptance of the technology. While the OODBMS is limited to a small niche market, this problem will continue to exist.
- **Lack of standards –** There is a general lack of standards of OODBMSs. We have already mentioned that there is not universally agreed data model. Similarly, there is no standard object–oriented query language.
- **Competition –** Perhaps one of the most significant issues that face OODBMS vendors is the competition posed by the RDBMS and the emerging ORDBMS products. These products have an established user base with significant experience available. SQL is an approved standard and the relational data model has a solid theoretical formation and relational products have many supporting tools to help.both end–users and developers.
- **Query optimization compromises encapsulations –** Query optimization requires. An understanding of the underlying implementation to access the database efficiently. However, this compromises the concept of incrassation.

  Locking at object level may impact performance Many OODBMSs use locking as the basis for concurrency control protocol. However, if locking is applied at the object level, locking of an inheritance hierarchy may be problematic, as well as impacting performance.
- **Complexity –** The increased functionality provided by the OODBMS (such as the illusion of a single–level storage model, pointer sizzling, long–duration transactions, version management, and schema evolution––makes the system more complex than that of traditional DBMSs. In complexity leads to products that are more expensive and more difficult to use.
- **Lack of support for views –** Currently, most OODBMSs do not provide a view mechanism, which, as we have seen previously, provides many advantages such as data independence, security, reduced complexity, and customization.
- **Lack of support for security –** Currently, OODBMSs do not provide adequate security mechanisms. The user cannot grant access rights on individual objects or classes.

  If OODBMSs are to expand fully into the business field, these deficiencies must be rectified.

❑ **Check Your Progress – 7 :**

1. How OODBMS can be mapped to RDBMS ?

.....

.....

.....

.....

.....

10.10 Example of Railway Reservation System :

The mapping can be also explained with the example of Railway Reservation system. Figure shows the table class mapping. Each row of the table represents an object instance and each column in the table represents to an attribute of the object.

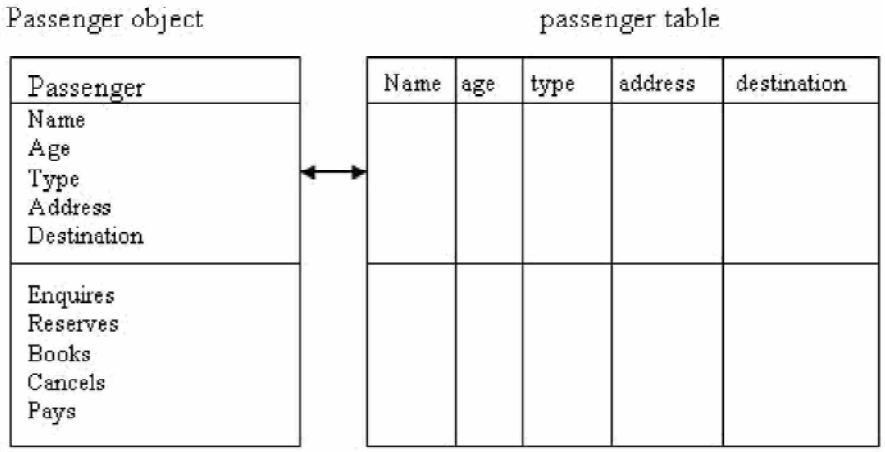


Table Class mapping: Single Object and single Table

For simplicity, the mapping of single object to single table is shown. However, in practice we are required to map multiple classes to a table in RDBMS. The single table mapping to multiple classes is shown in Figure.

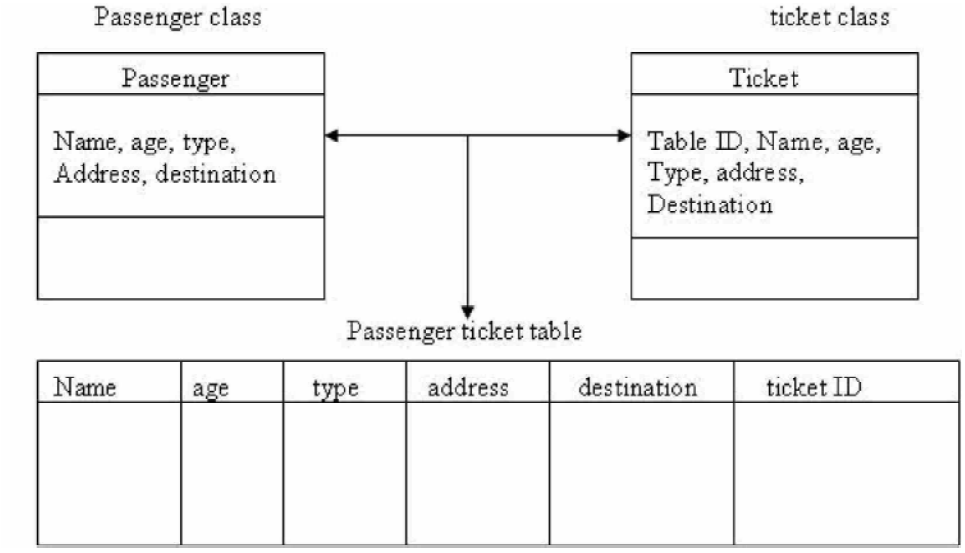


Table Class mapping : Single table Vs Multiple Classes

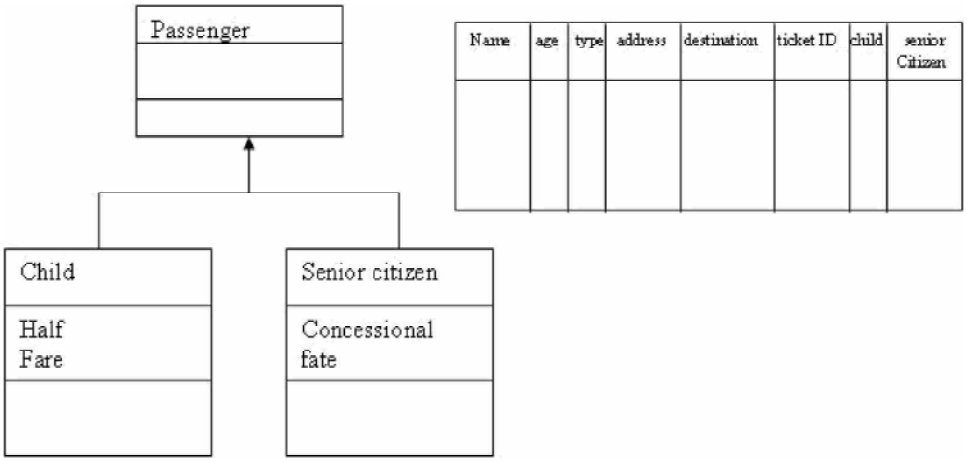


Table Verses multiple inheritance

❑ **Check Your Progress – 7 :**

1. Explain mapping of ODBMS to RDBMS for Railway reservation system.

   .....

   .....

2. _____ lock in which objects are managed in a cache.

   (A) Null (B) Notify (C) Writes (D) Read

3. The data once created by a process survives for long till it is created externally, is called _____

   (A) Meta (B) Transactional (C) persistence (D) None

4. _____ lock – Cause interruption of the program when a new value is committed in OODB

   (A) Null (B) Notify (C) Writes (D) Read

5. _____ locks allow processes to bulk copy data concurrently into the same table.

   (A) Bulk update (B) Bulk import (C) Bulk export (D) Bulk copy

6. _____ are those locks which are acquired during read operations such as SELECT.

   (A) Read locks (B) Shared locks

   (C) Exclusive locks (D) None of above

10.11 Let Us Sum Up :

The database management system (DBMS) is a set of computer programs that controls the creation, maintenance and use of databases of users and computer–using organizations.

Some objects continue to exist even though the program that created them is not alive.

The object consists of data values describing attributes of an entity, plus operations that can be performed upon data. This encapsulation capacity allows OO model to better handle more complex types of data (graphs, pictures, voice, text etc.) than the other database structure. The OO model also supports inheritance i.e. new objects can be automatically created by replicating some or all of the characteristics of one or more parent objects. Mapping of multiple

inheritance into a single table is possible in RDBMS. Relational RDBMS is built on schema representing complex relationships between entities.

The relational data model was invented by E.F. Codd. The concept of RDBMS is based on a table.

Complex data structures are handled effectively by OODB. OODB makes use of Product data management for this purpose.

The user must have access to both OODB and RDB to manipulate data. The developer therefore must develop applications that could source data from all databases (OODB, RDB etc). The Multi–database (MDBS) is a database system that sits on the top of existing relational and OODBMS but gives impression of being a single database to the users.

OODBMS can handle wide range of data types supporting complex data and structure. OODBMS is a result of blending OOP and database technology to meet the application needs of systems defined in OOT. OODBMS has more or less the same features of DBMS, but in addition, is in position to handle OO features.

A three schema architecture is a standard architecture for related database applications. Earlier, this architecture was proposed by the ANSI/SPARC committee on DBMS.

Objects at one location should be able to refer to objects at another location.

The object being independent and not linked with an application, once method is defined, is usable universally in all applications.

Large complex systems requiring complex data support are handled through distributed data sources or databases

10.12 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 10.2

- **Check Your Progress 2 :**

  See Section 10.3

- **Check Your Progress 3 :**

  See Section 10.4

- **Check Your Progress 4 :**

  See Section 10.5

- **Check Your Progress 5 :**

  See Section 10.6

- **Check Your Progress 6 :**

  See Section 10.7

- **Check Your Progress 7 :**

  1 : See Section 10.8

  2 : null

  3 : persistence

  4 : notify

  5 : bulk update

  6 : Shared locks

10.13 Glossary :

1. **Object oriented database management system (OODBMS) –** applies concepts of object–oriented programming, and applies them to the management of persistent objects on behalf of multiple users, with capabilities for security, integrity, recovery and contention management. An OODBMS is based on the principles of "objects," namely abstract data types, classes, inheritance mechanisms, polymorphism, dynamic binding and message passing.
2. **CAD –** Computer–aided design (CAD) is the use of computer systems to assist in the creation, modification, analysis, or optimization of a design.
3. **CAM –** Computer–aided manufacturing (CAM) is the use of computer software to control machine tools and related machinery in the manufacturing of work pieces
4. **OOSE –** Object–oriented software engineering (commonly known by acronym OOSE) is an object modeling language and methodology.
5. **Object –** It is an instance of class.
6. **Class –** In object–oriented programming, a class is an extensible program–code–template for creating objects, providing initial values for state (member variables) and implementations of behavior (member functions, methods).
7. **Methods –** Methods define the behavior of an object and are what was formally called procedures or functions.
8. **PDM –** Product data management (PDM) is the business function often within product lifecycle management(PLM) that is responsible for the management and publication of product data. In software engineering, this is known as version control.
9. **Persistent Objects –** The object store stores objects that have ability to survive even though the program or system that created them is no longer live. Such objects are called Persistent Objects.
10. **Persistence Data –** The data once created by a process survives for long till it is created externally, is called persistence data.
11. **Transient Data –** The data that has no value once the process that creates it is no more in existence is called Transient data
12. **A Multimedia database (MMDB) –** is a collection of related multimedia data. The multimedia data include one or more lprimary media data types such as text images, graphic objects (including drawings, sketches and illustrations) anim ation sequences, audio and video.
13. **A Multimedia Database Management System (MMDBMS) –** is a framework that manages different types of data potentially represented in a wide diversity of formats on a wide array of media sources. It provides support for multimedia data types, and facilitate for creation, storage, access, query and control of a multimedia database
14. **ODBC (Open Database Connectivity) –** is a standard programming language middleware API for accessing database management systems (DBMS). The designers of ODBC aimed to make it independent of database systems and operating systems.

15. **API –** API, an abbreviation of application program interface, is a set of routines, protocols, and tools for building software applications. The API specifies how software components should interact and are used when programming graphical user interface (GUI) components.

16. **JDBC –** Java Database Connectivity, a Java API that enables Java programs to execute SQL statements. This allows Java programs to interact with any SQL–compliant database.

17. **Active X –** is a software framework created by Microsoft that adapts its earlier Component Object Model (COM) and Object Linking and Embedding (OLE) technologies for content downloaded from a network, particularly in the context of the World Wide Web.

18. **CORBA –** Common Object Request Broker Architecture (CORBA) is an architecture and specification for creating, distributing, and managing distributed program objects in a network.

19. **DCOM –** DCOM (Distributed Component Object Model) is a set of Microsoft concepts and program interfaces in which client program object s can request services from server program objects on other computers in a network. DCOM is based on the Component Object Model (COM), which provides a set of interfaces allowing clients and servers to communicate within the same computer

10.14 Assignment :

Explain the concept of Object Oriented Database Management Systems (OODBMS) in our own words with suitable example.

10.15 Activities :

Explain the main concepts of object oriented data model like object structure, object classes, Inheritance, object Identity, object containment.

10.16 Case Study :

For each of the following application areas, explain why RDBMS is inadequate. List all specific system components that would need to be modified.

(a) Computer aided design

(b) Multimedia databases.

10.17 Further Reading :

1. Fundamentals of Database Management Systems – M.L. Gillenson, Wiely Publishing

2. Understanding Database management Systems – J.A. Vasta, Wadsworth Publishing Company.

BLOCK SUMMARY :

Unit 7 explains query language for databases that is SQL. SQL commands are instructions, coded into SQL statements, which are used to communicate with the database to perform specific tasks, work, functions and queries with data. SQL commands can be used not only for searching the database but also to perform various other functions like, for example, you can create tables, add data to tables, or modify data, drop the table, set permissions for users. SQL commands are grouped into four major categories depending on their functionality :

- **Data Definition Language (DDL) –** These SQL commands are used for creating, modifying, and dropping the structure of database objects. The commands are CREATE, ALTER, DROP.
- **Data Manipulation Language (DML) –** These SQL commands are used for storing, retrieving, modifying, and deleting data. These Data Manipulation Language commands are : SELECT, INSERT, UPDATE, and DELETE.

Unit 8 demonstrate the different constraints with practical aspects. The Primary key, Foreign Key, Not Null and Check Constraints all explained in detail.

Unit 9 explains the concept of transaction processing.Knowing the steps that must be taken to process a transaction can greatly affect how a custom application is developed. If a transaction is querying data only, using a read–only transaction can greatly reduce the amount of processing overhead required to run an application. If many users are running the same read–only query, this savings on overhead can be a considerable amount.

Likewise, knowing more about how statements are optimized can save a great deal of time in reaching the goals set for a new application. Because optimization methods take a critical role in meeting those goals, it is imperative that you take this into account.

Overall, knowing transaction processing steps is just plain helpful for administrators and developers alike.

Unit 10 explains the need and concept of object oriented databases. Object oriented databases are also called Object Database Management Systems (ODBMS).Object databases store objects rather than data such as integers, strings or real numbers. Objects are used in object oriented languages such as Smalltalk, C++, Java, and others. Objects basically consist of the following: Attributes and methods. Attributes are data which defines the characteristics of an object. This data may be simple such as integers, strings, and real numbers or it may be a reference to a complex object.

Methods – Methods define the behavior of an object and are what was formally called procedures or function With traditional databases, data manipulated

by the application is transient and data in the database is persisted (Stored on a permanent storage device).

In object databases, the application can manipulate both transient and persisted data. Object databases should be used when there is complex data and/or complex data relationships. This includes a many to many object relationship.

Object databases should not be used when there would be few join tables and there are large volumes of simple transactional data.

Object databases work well with: CAS Applications (CASE–computer aided software engineering, CAD–computer aided design, and CAM–computer aided manufacture), Multi–media Applications, Object projects that change over time and Commerce.

BLOCK ASSIGNMENT :

❖ **Short Questions :**

1. What is the use of order by clause ?
2. List all DML statements.
3. What is the use of group by clause ?
4. List the family of Transaction control statements.
5. What is object oriented databases ?

❖ **Long Questions :**

1. Explain Basic structure of SQL.
2. Explain the steps in transaction processing in detail.
3. Explain need and features of object oriented databases.
4. Compare RDBMS and OODBMS

BIBLIOGRAPHY

http://www.webbasedprogramming.com/Oracle–Unleashed/oun19fi.htm

http://repository.mdp.ac.id/ebook/programming–books/Best%20DataBase%20Reference/Oracle%20Unleashed/oun19fi.htm

http://ecomputernotes.com/database–system/adv–database/object–oriented–database–oodb

http://whatis.techtarget.com/definition/DCOM–Distributed–Component–Object–Model

❖ **Enrolment No. :** ☐

1. How many hours did you need for studying the units ?

Unit No.	7	8	9	10
No. of Hrs.				

2. Please give your reactions to the following items based on your reading of the block :

Items	Excellent	Very Good	Good	Poor	Give specific example if any
Presentation Quality	☐	☐	☐	☐	_____
Language and Style	☐	☐	☐	☐	_____
Illustration used (Diagram, tables etc)	☐	☐	☐	☐	_____
Conceptual Clarity	☐	☐	☐	☐	_____
Check your progress Quest	☐	☐	☐	☐	_____
Feed back to CYP Question	☐	☐	☐	☐	_____

3. Any other Comments

.....

.....

.....

.....

.....

.....

.....

.....

Dr. Babasaheb Ambedkar
Open University Ahmedabad

BCAR-202/
DCAR-202

Database Management System

BLOCK 4 : DATA (WARE HOUSING AND MINING) AND SECURITY

UNIT 11 TYPES OF DATABASE

UNIT 12 DATA WAREHOUSING AND DATA MINING

UNIT 13 DATABASE SECURITY

UNIT 14 RECOVERY MECHANISMS

DATA (WARE HOUSING AND MINING) AND SECURITY

Block Introduction :

Unit 11 gives overview of concepts of Data bases and its types and unit 12 gives more information of Data Mining and Data ware Housing and Unit 13 & 14 gives detailed discussion of Data Security and Recovery mechanisms. In simple terms data warehouse is a single, complete and consistent store of data obtained from a variety of different sources made available to end users in what they can understand and use in a business context. A process of transforming data into information and making it available to users in a timely enough manner to make a difference is Data warehousing. Data Mining works with Warehouse Data. Data mining is a process of discovering interesting patterns or knowledge from a (typically) large amount of data stored either in databases, data warehouses, or other information repositories. Data Warehousing provides the Enterprise with a memory. Data Mining provides the Enterprise with intelligence.

Database security is a broad area that addresses many issues like various legal and ethical issues regarding the right to access certain information, Policy issues at the governmental, institutional, or corporate level as to what kinds of information should not be made publicly available and System–related issues such as the system levels at which various security functions should be enforced. The need in some organizations to identify multiple security levels and to categorize the data and users based on these classifications. For example, top secret, secret, confidential, and unclassified. The security policy of the organization with respect to permitting access to various classifications of data must be enforced. Threats to databases can result in the loss or degradation of some or all of the following commonly accepted security goals : integrity, availability, and confidentiality.

A database can become unusable because of hardware or software failure, or both. You may, at one time or another, encounter storage problems, power interruptions, or application failures, and each failure scenario requires a different recovery action. Protect your data against the possibility of loss by having a well–rehearsed recovery strategy in place. A database recovery strategy should ensure that all information is available when it is required for database recovery. It should include a regular schedule for taking database backups and, in the case of partitioned database systems, include backups when the system is scaled (when database partition servers or nodes are added or dropped). Your overall strategy should also include procedures for recovering command scripts, applications, user–defined functions and stored procedure codes.

Block Objectives :

After learning this block, you will be able to :

- Understand need and basic concept of Data ware housing.
- Understand various architecture and tools in Data ware housing.
- Understand the concept of Data Mining.
- Understand various tools in data mining.
- Understand the need and concept of Database Security and Recovery.
- Understand the techniques of Database Recovery.

Block Structure :

Unit 11 : Types of Database

Unit 12 : Data Warehousing and Data Mining

Unit 13 : Database Security

Unit 14 : Recovery Mechanisms

Unit
11

TYPES OF DATABASE

UNIT STRUCTURE

11.1 Introduction

11.2 Centralized Database

11.3 Distributed Database

11.4 Personal Database

11.5 End–User Database

11.6 Commercial Database

11.7 NoSQL Database

11.8 Operational Database

11.9 Relational Database

11.10 Cloud Database

11.11 Object–Oriented Database

11.12 Graph Database

11.13 Let Us Sum Up

11.14 Suggested Answers to Check Your Progress

11.15 Glossary

11.16 Assignment

11.17 Activities

11.18 Case Study

11.19 Further Readings

11.0 Learning Objectives :

After learning this unit, you will be able to :

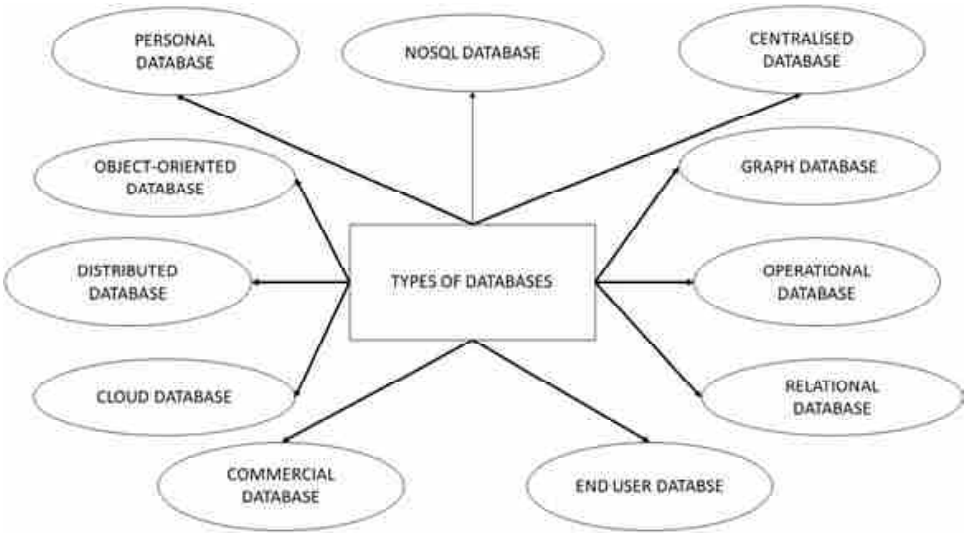
- Understand modern trends in DBMS
- Understand the basic concept of Data Warehousing
- Understand practical applications of Data Warehousing.
- Understand the basic concept of Data Mining
- Understand practical applications of Data Mining

11.1 Introduction :

Depending upon the usage requirements, there are following types of databases available in the market ?

- Centralised database.
- Distributed database.
- Personal database.
- End–user database.

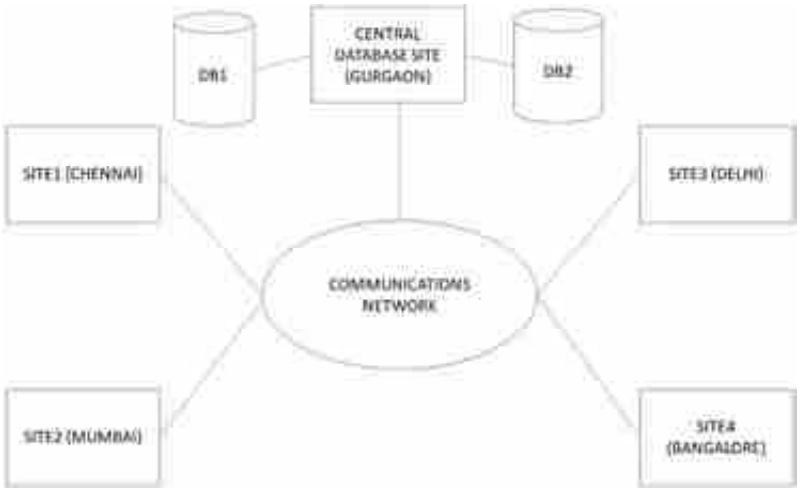
- Commercial database.
- NoSQL database.
- Operational database.
- Relational database.
- Cloud database.
- Object–oriented database.
- Graph database.



11.2 Centralized Database :

The information(data) is stored at a centralized location and the users from different locations can access this data. This type of database contains application procedures that help the users to access the data even from a remote location.

Various kinds of authentication procedures are applied for the verification and validation of end users, likewise, a registration number is provided by the application procedures which keeps a track and record of data usage. The local area office handles this thing.



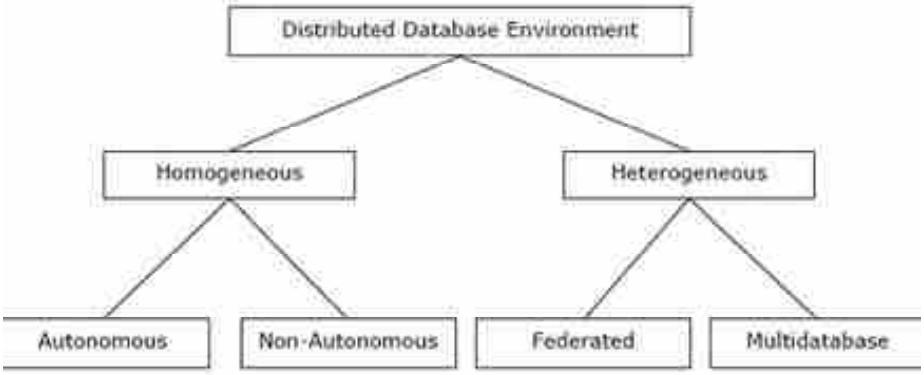
11.3 Distributed Database :

Just opposite of the centralized database concept, the distributed database has contributions from the common database as well as the information captured

by local computers also. The data is not at one place and is distributed at various sites of an organization. These sites are connected to each other with the help of communication links which helps them to access the distributed data easily.

You can imagine a distributed database as a one in which various portions of a database are stored in multiple different locations(physical) along with the application procedures which are replicated and distributed among various points in a network.

There are two kinds of distributed database, viz. homogenous and heterogeneous. The databases which have same underlying hardware and run over same operating systems and application procedures are known as homogeneous DDB, for eg. All physical locations in a DDB. Whereas, the operating systems, underlying hardware as well as application procedures can be different at various sites of a DDB which is known as heterogeneous DDB.



11.4 Personal Database :

Data is collected and stored on personal computers which is small and easily manageable. The data is generally used by the same department of an organization and is accessed by a small group of people.

11.5 End User Database :

The end user is usually not concerned about the transaction or operations done at various levels and is only aware of the product which may be a software or an application. Therefore, this is a shared database which is specifically designed for the end user, just like different levels' managers. Summary of whole information is collected in this database.

11.6 Commercial Database :

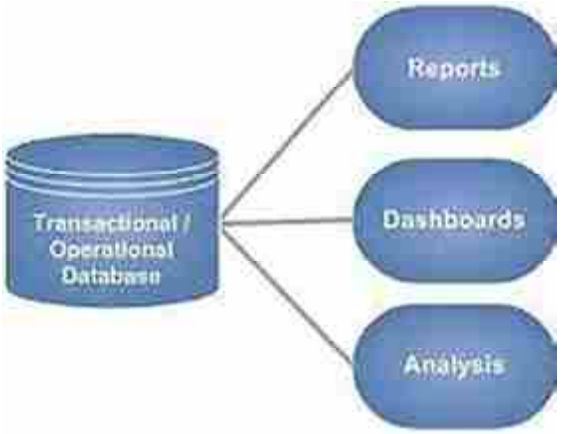
These are the paid versions of the huge databases designed uniquely for the users who want to access the information for help. These databases are subject specific, and one cannot afford to maintain such a huge information. Access to such databases is provided through commercial links.

11.7 NoSQL Database :

These are used for large sets of distributed data. There are some big data performance issues which are effectively handled by relational databases, such kind of issues are easily managed by NoSQL databases. There are very efficient in analyzing large size unstructured data that may be stored at multiple virtual servers of the cloud.

11.8 Operational Database :

Information related to operations of an enterprise is stored inside this database. Functional lines like marketing, employee relations, customer service etc. require such kind of databases.



11.9 Relational Database :

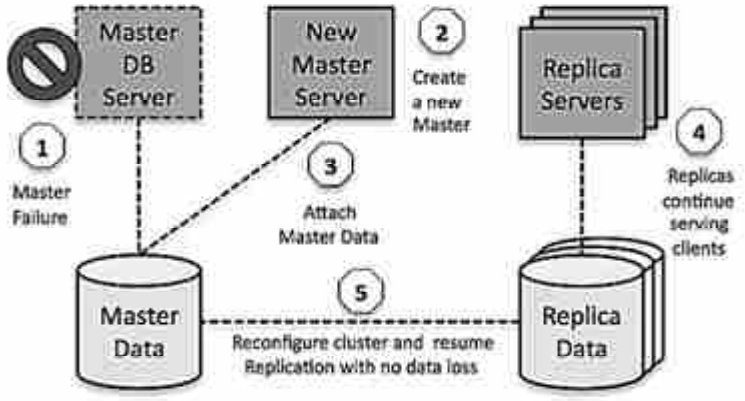
These databases are categorized by a set of tables where data gets fit into a pre–defined category. The table consists of rows and columns where the column has an entry for data for a specific category and rows contains instance for that data defined according to the category. The Structured Query Language (SQL) is the standard user and application program interface for a relational database.

There are various simple operations that can be applied over the table which makes these databases easier to extend, join two databases with a common relation and modify all existing applications.

11.10 Cloud Databases :

Now a day, data has been specifically getting stored over clouds also known as a virtual environment, either in a hybrid cloud, public or private cloud. A cloud database is a database that has been optimized or built for such a virtualized environment. There are various benefits of a cloud database, some of which are the ability to pay for storage capacity and bandwidth on a per–user basis, and they provide scalability on demand, along with high availability.

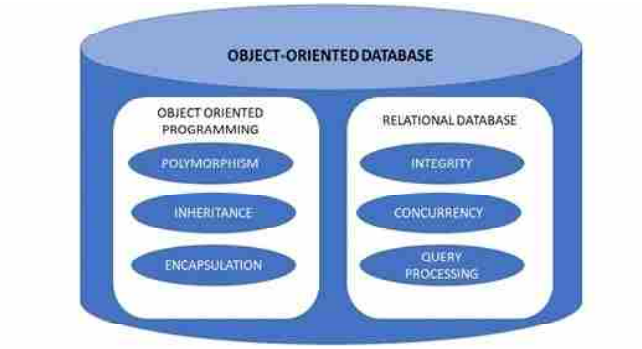
A cloud database also gives enterprises the opportunity to support business applications in a software–as–a–service deployment.



11.11 Object–Oriented Databases :

An object–oriented database is a collection of object–oriented programming and relational database. There are various items which are created using object–oriented programming languages like C++, Java which can be stored in relational databases, but object–oriented databases are well–suited for those items.

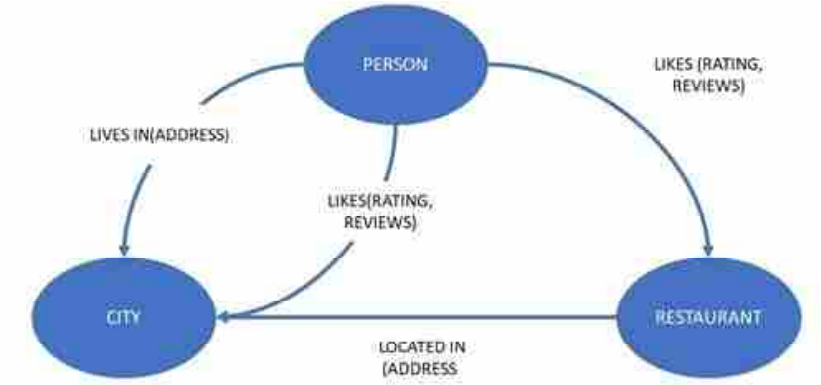
An object–oriented database is organized around objects rather than actions, and data rather than logic. For example, a multimedia record in a relational database can be a definable data object, as opposed to an alphanumeric value.



11.2 Graph Databases :

The graph is a collection of nodes and edges where each node is used to represent an entity and each edge describes the relationship between entities. A graph–oriented database, or graph database, is a type of NoSQL database that uses graph theory to store, map and query relationships.

Graph databases are basically used for analyzing interconnections. For example, companies might use a graph database to mine data about customers from social media.



❑ Check Your Progress – 4 :

1. Is A graph–oriented database, or graph database, is a type of NoSQL.

   (A) True (B) False

2. Which is true about Object oriented database ?

   (A) An object–oriented database is a collection of object–oriented programming and relational database

   (B) An object–oriented database is organized around objects rather than actions, and data rather than logic.

   (C) Both (D) None

3. The Cloud database have the ability to pay for storage capacity and bandwidth on a per–user basis, and they provide scalability on demand, along with high availability.

   (A) True (B) False

4. Which type of database is efficient in analyzing large size unstructured data that may be stored at multiple virtual servers of the cloud.

   (A) Centralized (B) Cloud (C) Relational (D) NOSQL DB

5. The operating systems, underlying hardware as well as application procedures can be different at various sites of a DDB which is known as _____ DDB

   (A) Heterogeneous (B) Homogenous

   (C) Centralised (D) End User

11.13 Let Us Sum Up :

This unit is focus on different types of database. It explain the types and usage of types of database.

11.14 Suggested Answers to Check Your Progress :

❑ **Check Your Progress :**

1 : True **2 :** Both

3 : True **4 :** NoSQL DB

5 : heterogeneous

11.15 Glossary :

Distributed databases : A distributed database is a type of database that has contributions from the common database and information captured by local computers. In this type of database system, the data is not in one place and is distributed at various organizations.

Relational databases : This type of database defines database relationships in the form of tables. It is also called Relational DBMS, which is the most popular DBMS type in the market. Database example of the RDBMS system include MySQL, Oracle, and Microsoft SQL Server database.

Object–oriented databases : This type of computers database supports the storage of all data types. The data is stored in the form of objects. The objects to be held in the database have attributes and methods that define what to do with the data. PostgreSQL is an example of an object–oriented relational DBMS.

Centralized database : It is a centralized location, and users from different backgrounds can access this data. This type of computers databases store application procedures that help users access the data even from a remote location.

11.16 Assignment :

Understand the different dbms.

11.17 Activities :

Understand when to use which database

11.18 Case Study :

List five database management system for five different domain context.

11.19 Further Reading :

1. Introduction to Database Systems by C. J. Date
2. Database Management Systems – Rajesh Narang – PHI Learning Pvt. Ltd.
3. Database System Concepts by Silberschatz, Korth – Tata McGraw–Hill Publication
4. An Introduction to Database Systems – Bipin Desai – Galgotia Publication
5. Database Management System by Raghu Ramkrishnan – Tata McGraw–Hill Publication
6. SQL, PL/SQL : The Programming Language Oracle – Ivan Bayross – BPB Publication

Unit
12

DATA WAREHOUSING AND DATA MINING

UNIT STRUCTURE

12.0 Learning Objectives

12.1 Introduction

12.2 Concept

12.3 Architecture

12.4 Various Tools in Data Warehousing

12.5 Tools in Data Mining

12.6 Difference Between Data Mining and Normal Query

12.7 Let Us Sum Up

12.8 Suggested Answer for Check Your Progress

12.9 Glossary

12.10 Assignment

12.11 Activities

12.12 Case Study

12.13 Further Readings

12.0 Learning Objectives :

After learning this unit, you will be able to :

- Understand modern trends in DBMS
- Understand the basic concept of Data Warehousing
- Understand practical applications of Data Warehousing.
- Understand the basic concept of Data Mining
- Understand practical applications of Data Mining

12.1 Introduction :

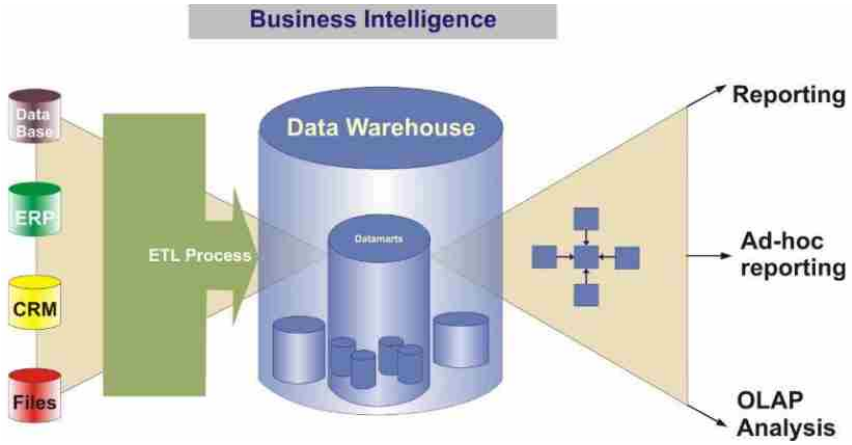
A data warehouse is formed through the integration of data from multiple heterogeneous sources. The concept of "Data Warehouse" was first introduced by Bill Inmon in 1990. He stated that a data warehouse is an integrated, subject oriented, time–variant, and non–volatile collection of data. The Data helps analysts to take decisions in an organization. It even supports analytical reporting, structured and/or ad hoc queries and decision making.

An operational database goes through nonstop changes on a daily basis because of the regular transactions that take place. In order to understand it in a much better way let us assume an employee of a business who wants to examine earlier response on any data such as a product, a supplier, or any customer data, then the executive will have no data offered to analyze because the previous data has been efficient due to transactions.

A data warehouses offer us a general and consolidated data in multidimensional view. Besides this it even provides us Online Analytical Processing tools. These tools are of great help to us in interactive and effective analysis of data in a multi–dimensional space. This analysis is of great help in data generalization and data mining.

The functions of Data mining such as association clustering, classification, prediction can be included with Online Analytical Processing operations in order to enhance the interactive mining of knowledge at multiple level of abstraction. That's the reason why data warehouse has become an essential platform for data analysis as well as for online analytical processing.

12.2 Concept :



Data Warehouse

_Data Warehouse' and _Data Warehousing' have been defined differently by different people. However few of the important definitions are given below :

According to Barry Devlin, –Data Warehouse is a single, complete and consistent store of data obtained from a variety of different sources made available to end users in a way they can understand and use in a business context ||

The table design, dimensions and organization should always be consistent enough throughout a data warehouse so that reports or queries across the data warehouse are always consistent. It can also be viewed as a database for historical data from different functions within the same company.

A Data Warehouse is nothing but a collection of data which is basically used for important decision making of the organisation. A Data warehouse is always :

- Subject–oriented
- Integrated
- Time–varying
- Non–volatile

The meaning of each term used above in the definition is as follows –

Subject–oriented – The data warehouse is mostly organized around the major key subjects of the enterprise. Major subjects includes customers, patients, students and products.

Integrated – The data housed in the Data Warehouse are defined using consistent naming conventions, formats, encoding structures and related characteristics.

Time–variant – Data in data warehouse contain a time dimension so that they may be used as historical record of the business.

Non–volatile – Data in data warehouse are loaded and refreshed from operational systems, but cannot be updated by end users.

❖ **Understanding a Data Warehouse :**

- It is a database, which is reserved split from the organization's operational database.
- Even there is no recurring updating in a data warehouse.
- It has consolidate historical data that helps the organization to examine its business properly.
- It provides executives to organize, understand, and use their data to take strategic decisions.
- It helps in the integration of variety of application systems.
- This system helps in consolidated chronological analysis of data.
- A data warehouses is always kept split from operational databases because of the following reasons :
- The Operational database is always constructed keeping in mind for well–known errands and workloads. In contract, data warehouse queries are often very complex and they present a general form of data.
- The Operational databases also support simultaneous processing of multiple transactions. In case of operational database concurrency control and recovery mechanisms are a must they have to be very robust and there should be consistency of the database.
- The operational database inquiry allows reading as well as modifying operations, while an OLAP query needs read only access of stored data.
- This should be kept in mind that operational database maintains current data. On the other hand, a data warehouse maintains historical data.

Data warehousing is the method of constructing and using a data warehouse. A data warehouse is constructed by integrating data from multiple heterogeneous sources that support analytical reporting, structured and/or ad hoc queries, and decision making. Data warehousing involves data cleaning, data integration, and data consolidations.

❖ **Data Warehouse Applications :**

As discussed earlier, a data warehouse helps a lot to the business executives in organizing, examining, and use their data for important decision making. A data warehouse is of great helps as a sole part of a plan–execute–assess "closed–loop" feedback system for the enterprise management. Data warehouses have been broadly used in the following fields :

- The Financial Services
- The Banking Services
- The Consumer Goods

- The Retail Sectors
- The Controlled Manufacturing

❑ **Check Your Progress – 1 :**

1. Explain the term Data Warehouse.

.....

.....

.....

.....

.....

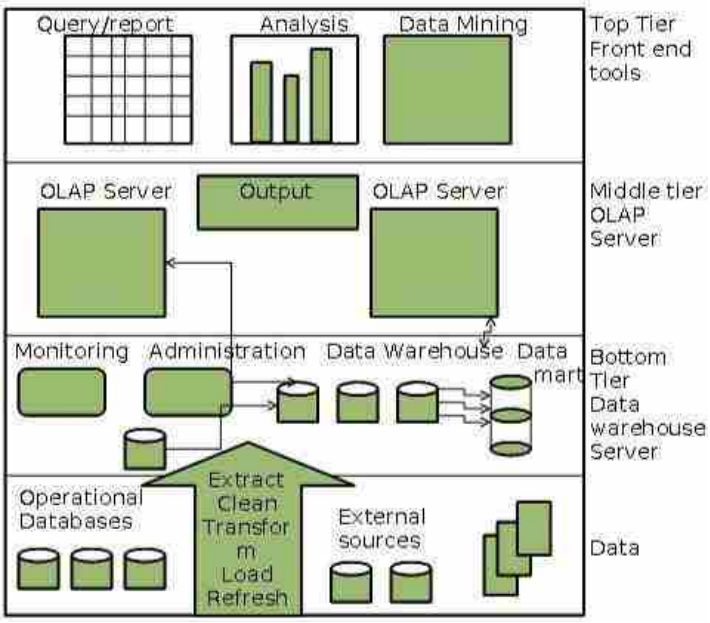
12.3 Architecture :

Three–Tier Data Warehouse Architecture :

Generally a data warehouses has a three–tier architecture. The three tiers of the data warehouse architecture are as follows.

- **Top–Tier –** This tier is the front–end client layer. This layer keeps the query tools and reporting tools, analysis tools and data mining tools.
- Middle Tier – In the middle tier, we have the OLAP Server that is used in either of the following ways.
  a. Through Multidimensional OLAP also known as (MOLAP) model in short, this starts the multidimensional data and operations.
  b. Through Relational OLAP also known ROLAP in short, it is an extended type of relational database management system. It maps the operations on multidimensional data to the standard relational operations.
- **Bottom Tier –** This architecture is treated as the data warehouse database server. It is also a type of the relational database system. It is the back end tools and utilities to feed data into the bottom tier and these back end tools and utilities help in the process of the Extract, Clean, Load, and refresh functions.

❖ **The Three–Tier architecture of a data warehouse :**



❖ **Data Warehouse Models :**

If we make a study from the point of view of data warehouse architecture then we have the following models :

(a) First is Virtual Warehouse

(b) Second is the Data mart

(c) Thirdly is the Enterprise Warehouse

(a) Virtual Warehouse :

An operational data warehouse is identified as a virtual warehouse. It is easy to create a virtual warehouse. Creating a virtual warehouse operations require additional capacity on the database server.

(b) Data Mart :

Organization–wide data mart is a subset of the data. This subset of data is important for an organization's specific groups in other words, data marts include precise figures for a particular group. For example, marketing data mart items, customers, and may include data related to sales. Data Mart are restricted to subjects. The following points should be considered in applying data mart

- Mostly windows or Unix/Linux–based servers are used to implement data marts. They are implemented on low–cost servers.
- The implementation data mart cycles are deliberate in short periods of time, i.e., in weeks rather than months or years.
- The life cycle of a data mart might be very complex in long run but if its planning and design are not organization–wide.
- The Data marts are very small in size.
- The Data marts are personalized by department.
- The source of a data mart is departmentally prepared data warehouse.
- Data marts are flexible.

(c) Enterprise Warehouse :

- The enterprise warehouse obtains all the information and the subjects spanning from whole organization
- They gives us enterprise wide data integration.
- The data is included from operational systems and external information providers.
- This data or information can differ from a few gigabytes to hundreds of gigabytes, terabytes or even beyond that.

In the data warehouse environment usually specific architecture transforms relational data model. There are several schema models designed for data storage, but are most commonly used :

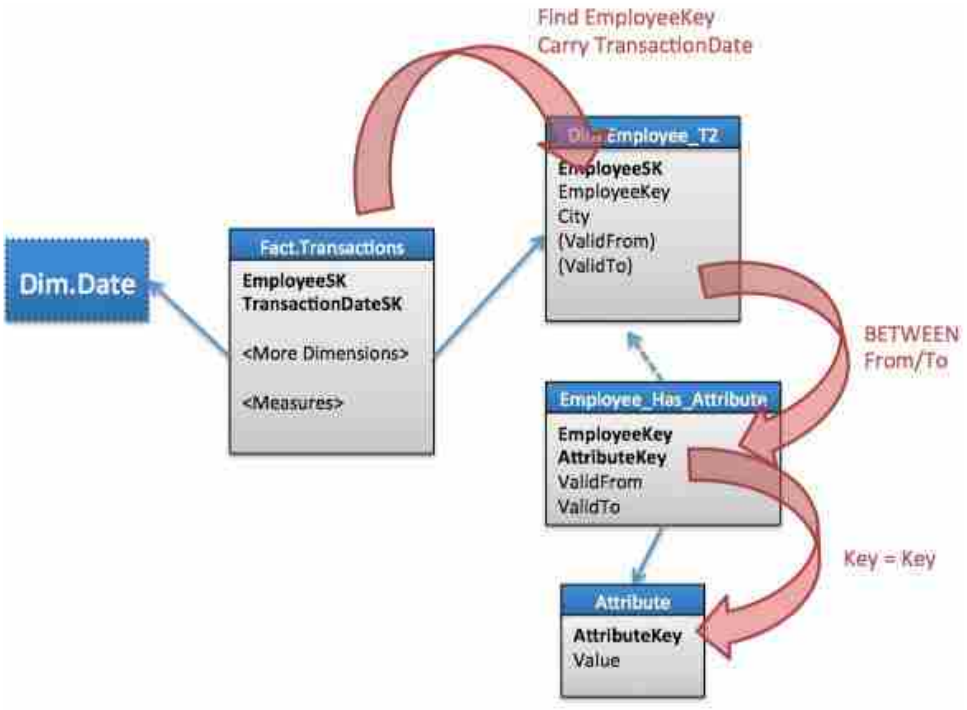
– Star Schema

The determination of which schema model should be used for a data warehouse should be based upon the analysis of project requirements, accessible tools and project team preferences.

❖ **Star Schema :**

Its architectural Diagram resembles very much to a star with a center point of the scorching it is called a star schema. This architecture is the simplest

data warehouse schema. Star of the center points of the fact table and dimension tables are the star. Fact tables in a star schema typically de– dimensional tables are normalized, the third normal form (3NF) are in. Despite the fact that the star schema is the simplest architecture, it is most often used today and is recommended by Oracle.



Fact and Dimension table

❖ **Fact Tables :**

Dimension tables are numerical facts and behavior of those foreign key : A fact table typically has two types of columns. Aggregated level of detail or the fact that a fact table can contain data.

❖ **Dimension Tables :**

Levels of the hierarchy and is not a dimension, then it is called flat dimension or list. One dimension is usually the primary key of each of dimension tables are part of a composite primary key of the fact table categorizes the data that is a structure composed of one or more hierarchy. Dimensional characteristics that help to describe the dimensional value. They usually meaningful, literal values. Dimension tables are usually small in size, the fact table.

General facts concerning the sale of tables to store data, the geographical area (markets, city), customers, products, times, channels, data about the dimension tables.

The main characteristic of the star schema :

1. Simple structure, easy to understand schema
2. Immense query and swagger : the small number of tables to join
3. The relatively long time to load the data in the dimension tables : D – General, redundancy may cause the data table that is large.
4. The most often used in data warehouse implementation is widely supported by a large number of business intelligence tools.

❑ **Check Your Progress – 2 :**

1. Explain three tier architecture of Data warehouse in detail.

.....

.....

.....

.....

.....

12.4 Various tools in Data Warehousing :

Typically, a data warehouse, data extraction, transformation and loading of data from one or more sources, known as ETL is loaded through the process. To pull the data out of a database and place it in another database that automates the process are combined into one device that serves three databases.

- **Extract –** to read data from a given source database and a desired subset of information processing of data.
- **Transform –** it can be inserted into the database, so it is necessary to form removed or is the process of converting data from your past. Changes in regulations or lookup tables or by combining with other data.
- **Load –** Summary sorting process, strengthen the integrity check, and construction of the index and data division. It also includes warehouse refresh the data sources including updating.

❑ **Check Your Progress – 3 :**

1. Explain the ETL process.

.....

.....

.....

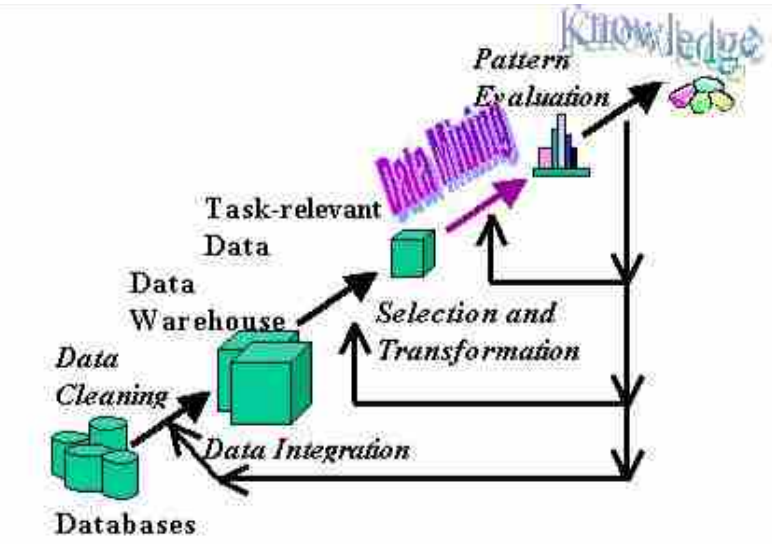
.....

.....

12.5 Tools in Data Mining :

Files, databases, and other large–scale data stored in repositories, with analysis and interpretation of such data can help in decision making and for the return of the interesting knowledge to rapidly develop a powerful tool is important.

Data mining also KDD (Knowledge Discovery in Databases), known as the data in the database, previously unknown and potentially useful information nontrivial underlying refers to remove. Data Mining and KDD are synonyms of each other, data mining, knowledge discovery process is considered a real part. The figures below a step in an iterative process of knowledge discovery shows as data mining.



Data Mining is the core of Knowledge Discovery process

The process of knowledge discovery in databases starting from raw data collection and steps leading to new knowledge involves some form. Iterative process includes the following steps :

- **Data cleaning –** It is also known as data cleaning, data unrelated to the noisy data and are removed from the data collection, which is a platform.
- **Data integration –** Here at this stage, multiple data sources, often asymmetrical, can be collected or stored in a common source.
- **Data Selection –** This step is determined on the basis of data analysis and data collection have been included.
- **Data transformation –** it is also called as data consolidation, the demand for data mining process is transformed into the appropriate forms, which is a stage.
- **Data Mining –** The intelligent technique to extract potentially useful patterns are working, which is taken to be one of the important steps.
- **Pattern evaluation –** Here at this stage, knowledge representation tightly interesting patterns are based on the measures, which are recognized.
- **Knowledge Representation –** It turns out that the knowledge / information for the user has appeared in the final stage. The move users to understand and interpret the data mining results, use visualization techniques to help.

This joint is common to combine some of these steps. For example, data cleaning and data integration to generate a data warehouse as a pre–processing stage can be together. Consolidation of data as in the case of data warehouses, data selection is changed, the result of selection, or where the data selection and data transformation can also be shared.

Revealed knowledge is available to the user once, iterative process knowledge discovery in databases, which can be extended to the evaluation measures, mining may be more sophisticated, new data selected or turned on, or a new data source can be can be different, and in order to get more relevant results, is included.

Data mining a vein of valuable ore mining a large database and search for information in the rocks really derives its name from the resemblance between. Sifting through a large amount of the material or are inventively values actually mean the point is to examine the contents. Mining for gold in rocks usually "gold mining" and not "rock mining is called" since it follows, rather than data mining "knowledge mining" should have been asked by analogy, however, is a misnomer.

However, data mining has become traditional customary period, and even that is such a more complete description of the process database (KDD) as the pursuit of knowledge that greatly overshadowed more general terms a trend rapidly. Other similar words referring to data mining : the data dredging, knowledge extraction and pattern discovery.

In standard, data mining is a kind of media or data is not accurate. Data mining information should be applied to any type of repository. Although applicable to different types of data, the algorithms and approaches may be different. In reality, the challenges presented by different types of data vary greatly. Data mining relational databases, object–relational databases and object–oriented databases, data warehouses, transactional databases, such as the World Wide Web as unstructured and semi – structured repositories, such as advanced spatial database, including the database to be used being studied databases, multimedia databases, time–series database and textual databases, and even flat files.

May be exposed to that type of pattern employed depend largely on data mining tasks. By and large, there are two types of data mining functions :

Existing data that describe the general properties of descriptive data mining.

Predictions based on data available at the conclusion that it is attempting to predictive data mining.

Data mining functionalities and they are briefly presented in the following list shows the variety of knowledge :

Used tools for data mining are :

Artificial neural networks – learn through training and resemble biological neural network structure prognostic model non–linear.

Decision trees – tree–shaped structures that correspond to the set of decisions. These decisions generate rules for the classification of a dataset.

Rule Induction – useful if the statistical significance of the return on the rules of data.

Genetic algorithms – genetic combination optimization techniques, mutation, and natural selection will depend on the concepts.

Nearest neighbour – a classification technique that classifies each record in a historical database to the most depend on similar records.

❖ **Data Mining Applications :**

Data mining "knowledge from huge set of data extraction process" can be defined as. To put it in other words, we say that data mining data mining and knowledge that can help. This information can be used for any of the following applications :

- The Analysis of Market
- The Detection of Fraud
- Retention of the valuable Customer
- The Control of Production
- The Exploration of Science
- The Risk management and Corporate Analysis
- The Analysis of Financial Data
- The Retail Industry
- The Industry of Telecommunication
- Analysis of the Biological Data
- Some Other Scientific Applications
- Detection of Intrusion

❑ **Check Your Progress – 4 :**

1. What is Data Mining ? Explain the steps in KDD process.

.....

.....

.....

.....

.....

12.6 Difference between Data Mining and Normal Query :

Virtually all modern, commercial database systems are based on the relational model formalized by Codd in the 60s and 70s and the SQL language which allows the user to efficiently and effectively manipulate a database. In this model a database table is a representation of a mathematical relation, that is, a set of attribute of items that share certain characteristics or attributes. Here, each table column represents the relation and each record in the table represents a member of this relation. Relational databases do not only allow for the creation of tables but also for the manipulation of the tables and the data within them. The most fundamental operation on a database is the query. This operation enables the user to retrieve data from database tables by asserting that the retrieved data needs to fulfill certain criteria. Thus Query Tools are tools that help analyze the data in a database. They provide query building, query editing, searching, and finding, reporting and summarizing functionalities.

In disparity to a query which basically returns the data that fulfills assured constraints, data mining constructs models of the data in question. The models can be viewed as high level summaries of the essential data and are in most cases more useful than the raw data, since in a business sense they typically represent reasonable and actionable items .Depending on the questions of interest, data mining models can take on very different forms. They include decision trees and decision rules for classification tasks, association rules for market basket analysis, as well as clustering for market segmentation among several other possible models.

Thus we previously unknown and interesting information from the data mining of raw unprocessed data extraction is the function of which is to deal

with an area of computer science, you can say that. Used as input data for the process usually is stored in the database. Data users who are inclined toward the use of data mining. Look for hidden patterns in the data, various statistical model / tool. Data miners is always profitable for businesses which ultimately fruitful relationship between different data elements are interested in finding.

❑ **Check Your Progress – 5 :**

1. Differentiate data mining and normal query.
2. Full form of KDD
   (A) Knowledge Discovery Database
   (B) Knowledge DD
   (C) KDD
   (D) Kmean Data
3. Algorithm is
   (A) It uses machine–learning techniques. Here program can learn from past experience and adapt themselves to new situations.
   (B) Computational procedure that takes some value as input and produces some value as output.
   (C) Science of making machines performs tasks that would require intelligence when performed by humans.
   (D) None of these
4. In following blanks find right sequence of option from given five option.
   - At _____ step multiple data sources, often asymmetrical, can be collected or stored in a common source.
   - At _____ step is determined on the basis of data analysis and data collection have been included
   - At _____ step intelligent technique to extract potentially useful patterns are working
   - At _____ step the user has appeared in the final stage.

   (A) Data Integration (B) Data Selection
   (C) Data Transformation (D) Data Mining
   (E) Knowledge Representation
5. ETL full form.
   (A) Extract Transfrom Load (B) extra load
   (C) Extract Load Transform (D) None

12.7 Let Us Sum Up :

The terms _Data Warehouse' and _Data Warehousing' are defined in different ways by different people.

Devlin, Data Warehouse is a single, complete and consistent store of data obtained from a variety of different sources made available to end users in a way they can understand and used in a business context.

Table design, dimensions and organization should be consistent throughout a data warehouse so that reports or queries across the data warehouse are consistent. A data warehouse can also be viewed as a database for historical data from different functions within a company.

Data Warehouse is a collection of data that is used primarily in organizational decision making.

Data warehouse is

- Subject–oriented
- Integrated
- Time–varying
- Non–volatile
- Bill Inmon, Building the Data Warehouse 1996

Data Warehousing is the process whereby organizations extract meaning from their informational assets through the use of Data Warehouses. Structured analytical reporting a data warehouse to support various different sources, and / or ad hoc queries, and decisions are made by integrating the data. The data cleaning, data integration and consolidation of data included. Data available with the help of decision support technologies used in a data warehouse. Using these technologies quickly and effectively to the warehouse is of great help to the authorities. It becomes very easier for them to gather the data, analyze it and take decisions accordingly on the basis of information present in the warehouse.

Generally a data warehouses adopts a three–tier architecture. From the perspective of data warehouse architecture, data warehouse models are Virtual Warehouse, Data mart and Enterprise Warehouse. Typically, the data in a data warehouse is loaded through the process of ETL, i.e. extraction, transformation and loading, from one or more sources of data.

Data Mining, also popularly known as Knowledge Discovery in Databases (KDD), refers to the nontrivial extraction of implicit, previously unknown and potentially useful information from data in databases. There is enormous amount of data available in Information Industry and this data is of no use unless and until it is converted into some useful information. Analysing this huge amount of data and extracting useful information from it is necessary. Data Mining is process of extracting the information from the huge set of data. In other words we can say that data mining is mining the knowledge from data.

The Knowledge Discovery in Databases process comprises of a few steps leading from raw data collections to some form of new knowledge. The iterative process consists of the steps : Data cleaning, Data integration, Data selection, Data transformation, Data mining, Pattern evaluation and knowledge representation.

The data mining functionalities are Characterization, Discrimination, Association analysis, Classification, Prediction, Clustering, Outlier analysis and Evolution and deviation analysis. Applications of data mining are like Market Analysis, Fraud Detection, Customer Retention, Production Control and Science Exploration etc.

12.8 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 12.2

- **Check Your Progress 2 :**

  See Section 12.3

- **Check Your Progress 3 :**

  See Section 12.4

- **Check Your Progress 4 :**

  See Section 12.5

- **Check Your Progress 5 :**

  1 : See Section 12.6,

  2 : Computational procedure that takes some value as input and produces some value as output,

  3 : Knowledge Discovery Database,

  4 : Data Integration, Data Selection, Data Mining,Knowledge representation,

  5 : Extract Transform Load

12.9 Glossary :

1. **Data Warehouse –** It is a collection of data that is used primarily in organizational decision making.
2. **Data warehouse is :** Subject–oriented, Integrated, Time–varying and Non–volatile.
3. **Data warehousing –** it can be explained as the process of building and using a data warehouse. The process of data warehousing involves data cleaning, integration, and consolidations.
4. **OLAP –** Its full form is Online analysis processing.
5. **Data mart contains a subset of organization –** wide data. This type of data is very important for specific groups of an organization.
6. **Enterprise Warehouse –** It collects all the information and the subjects spanning an entire organization.
7. **Classification –** analysis means organization of the data in given classes.
8. **Clustering –** Similar to classification, clustering is the organization of data in classes. However, unlike classification, clustering, the class labels are unknown to the discovery of admissible classes depending on the clustering algorithm.
9. **Outliers –** These cannot be shared in a given class or cluster the data elements. Also known as exceptions or surprises, they are often very important to recognize.

12.10 Assignment :

Write difference between Data Ware housing and Data mining with suitable example. Search information on internet and relate it with real world

12.11 Activities :

How data mining is useful in market analysis? Explain with example.

12.12 Case Study :

Study different types of data mining tools which perform tasks like classification and association analysis.

12.13 Further Reading :

1. Fundamentals of Database Management Systems – M. L. Gillenson, Wiely Publishing
2. Understanding Database management Systems – J. A. Vasta, Wadsworth Publishing Company

Unit

13

DATABASE SECURITY

UNIT STRUCTURE

13.0 Learning Objectives

13.1 Introduction

13.2 Password Authentication

13.3 Operating System Authentication

13.4 Why Protect Passwords ?

- **13.4.1 Control**
- **13.4.2 Protection**
- **13.4.3 Integrity**
- **13.4.4 Privileged Accounts**
- **13.4.5 SYS**
- **13.4.6 SYSTEM**

13.5 Other Issues

- **13.5.1 Operating System Group : DBA**
- **13.5.2 Object Security**
- **13.5.3 Access Rights**
- **13.5.4 Resolving Object Synonyms**
- **13.5.5 System Security**
- **13.5.6 Defined System Privileges**
- **13.5.7 Object Security Model**
- **13.5.8 Database Auditing**

13.6 Recovery from Various Problems of Volatile and Non–Volatile Storage Devices

13.7 Let Us Sum Up

13.8 Suggested Answers to Check Your Progress

13.9 Glossary

13.10 Assignment

13.11 Activities

13.12 Case Study

13.13 Further Readings

13.0 Learning Objectives :

After learning this unit, you will be able to :

- Understand the concept of security of database
- Understand Password Authentication
- Understand Operating System Authentication

- Understand the need for password protection
- Understand the concept of Object Security
- Understand the concept of Database Auditing
- Understand Access Rights
- Understand Object synonym
- Understand Transaction properties and recovery mechanism.

13.1 Introduction :

Although we have been living in extremely technological advanced era where we are surrounded by so many satellite to monitor the earth and every second billions of people are getting connected through information technology, failure is expected but not at every time.

DBMS is considered to be one of the most complex system where hundreds of transactions are being executed every second and the availability of DBMS depends to a very great extent on its complex architecture and fundamental hardware as well as system software. In the event of failure or crashes amid transactions being executed, it is predictable that the system will follow some sort of algorithm or techniques through which data could be recovered from such crashes or failures.

13.2 Password Authentication :

Password authentication is one of the additional traditional method generally used for user authentication. Most of the time passwords are comparatively secure but then too they have some weakness that can make them more vulnerable to breach or attack.

In most cases, the user creates a unique password. But remember that people generally susceptible to attack password authentication, which makes simple password like to keep. Use multiple databases, users often use the same password over and over again and if a user's password is compromised, it makes a massive security breach that the offending party is part of the password for the database gives access.

To reduce the risk of any breach password below should follow some of the rules :

- First, the length should be at least eight characters.
- Secondly, it should contain a combination of characters and numbers or special characters that represent the actual spelling of words (such as Block for bicycle).
- Thirdly, it should not contain easily understandable passwords.
- Lastly, it should be frequently changed.

❑ Check Your Progress – 1 :

1. What are the general standards for giving passwords?

.....

.....

.....

.....

.....

13.3 Operating System Authentication :

Operating system password authentication methods require an underlying operating system level, it authenticated at the operating system level in mind that the use of accounts is very important to protect the bearing. The Oracle database account to an account that matches the operating system that exists within the database to verify whether that means. For the operating system to the database to verify the user. Successful match is found, the user is allowed to use.

If used properly, the operating system is secure authentication intelligently. Under this method to change a password is needed. Connect to the database user name and password to continue just like a faucet is very simple.

❑ **Check Your Progress – 2 :**

1. How operating system is authenticated ?

.....

.....

.....

.....

.....

13.4 Why Protect Passwords ?

Authentication method and a long conversation about the accounts of defense I After, why it is necessary to think it is very common. This database is going to be a debatable issue, depending on the environment. At many sites, the concept of safety is taken very seriously. Other sites, organizations, especially those that are not traditional, the question needs. Control, security, and integrity of the database to limit the use of three basic reasons.

13.4.1 Control :

The word several times every hour to control the users need to file paperwork that evokes images of mainframe shops is a heavy–handed words. DBA in day–to–day operations barrier and regret, no malicious intent or by accident or due to excessive should not impose restrictions, however, a database is simply too much power in the database can be corrupted

13.4.2 Protection :

In addition to issues of control, to limit the use of the database and the database within a database for data protection is one of the most effective ways. Most users would not maliciously damage the database, as a subversion of an accident can be just as overwhelming.

13.4.3 Integrity :

Efforts to protect password protect the integrity of the data within the database objects to. Because of the many privileges within the database while users to arbitrarily manipulate database data allows. Generally these privileges to update any table in any table and deletions include the system–level privileges. With these privileges, a user object–level security can supersede and modify the data in any database table.

To control objects within the database for data changes is one of the most difficult task. After all, users should be able to use the data from the database. Access control and defensive with physical database objects within the same database as the data should be protected and any one design and implement a security plan should be the paramount concern.

13.4.4 Privileged Accounts :

The Privileged accounts for the Oracle DBMS are of number of forms. Some of them are conventional password authenticated accounts, some specific database privileges are the operating system group while. Each site has its own custom–defined or privileged DBA can access account.

13.4.5 SYS :

The database user _SYS' is the main proprietor of all base tables, user views, and data dictionary views. At the time of creation of database these tables and views are created during internal mechanisms.

13.4.6 SYSTEM :

Just like SYS, the SYSTEM account is shaped when the database is created. The SYS user owns tables and views that orientation internal database information, the SYSTEM account owns tables that are owned by Oracle tools, such as SQL*Menu. Generally this schema is used to install software products for most third–party tools.

❑ Check Your Progress – 3 :

1. What are the basic reasons for limiting database access ?

.....

.....

.....

.....

.....

13.5 Other Issues :

13.5.1 Operating System Group – DBA :

The Oracle provides us a third method of authentication and is very specific in nature. Oracle provides convinced privileges and must be carefully monitored.

UNIX / group file, specify the DB Group is a member of the user, the database can connect to the interior. With special privileges to connect to the database as SYS enables him an opportunity to use by the UK level. Oracle DBA group normally steal user group. Startup and shut down operations on the database to obtain necessary to connect the internal.

13.5.2 Object Security :

If a user on a specific database object security object that is responsible for defining the specific rights. In Oracle Database Appliance, is a way of default protection. A user that owns the database object database is full privileges. The user, in turn, these objects to another database user can provide any and all privileges. He has been denied access, which tries to access a database object, the user receives an error message if any.

13.5.3 Access Rights :

One of the key aspects of the object privileges to accept fully understand what privileges are available to the user's database. There are nine Oracle object privileges

ALTER one such operation, lack of storage add adjusting column, which includes performance and enables the user to change a database table.

DELETE SQL database objects using the Delete command to remove rows from the user's rights.

EXECUTE Within the database to execute a stored procedure or package enables the user.

INDEX enables the user to create a new index or to change or modify a database table to an attainable index.

INSERT to create new rows in a database table enables.

REFERENCES Through a foreign key in another table, the attribute information to the reference table gives the user the right.

SELECT Information within a database object that enables the user to view the rows.

UPDATE – Users within a database object that enables to change the existing lines.

ALL Is a database object privileges on the user's previous.

Multiple databases within a single grant option privilege is possible to join. Example,

% sqlplus system

Password :.....

SQL>grant select, insert, update on emp table to

efcko; Grant succeeded.

13.5.4 Resolving Object Synonyms :

Synonymous is nothing but just to be referred to some other enables a database object is a designation. Private and public : There are two types of synonyms. He uses a private synonym is created by the user, which is a synonym; you can use it is made synonymous none other than the user, on the other hand, is available to all users in a public synonym database.

13.5.5 System Security :

Where object opportunity deals with what a user can do to database objects and system privilege deals with what actions a user can perform against the database itself. The actions include connecting to the database, creating the tables of database, and dropping an entire table space with all the database objects in it.

Under Oracle6, the Oracle RDBMS resembled UNIX in its overall security scheme. UNIX maintains that an account is either the root user or a regular user. Admittedly, UNIX has evolved to enable a superior deal of scalability by using things such as access control lists (ACLs) and the root set user id programs. Oracle6 is set up so that all users are either the DBA or not the DBA. With the release of Oracle7, it moves away from this methodology. It is possible now to grant specific privileges to non–DBA users,

thereby enabling them to perform certain applications without giving them full DBA access.

13.5.6 Defined System Privileges :

In Oracle6, three system privileges are available. Over 80 system privileges are available in Oracle7.

ANY Privileges

The ANY privileges are a type of special class of privileges within the database system privileges. They are improved system privileges that gives the user the ability to perform particular actions without restrictions. If a user has these system privileges, he can override normal default security. So, he has access to other database objects, regardless of whether an object–level grant is made.

13.5.7 Object Security Model :

The DBA must also consider the other factors while setting up a security plan. Not only the setup of the database users be considered, but also the ownership of the database objects should be considered. Although there is no right or wrong way or any other fundamental rule to go about this but the following sections outline some of the concerns faced by the DBA when setting up object ownership models.

❖ **Protected Object Ownership Schema :**

One of the security model which is implemented by many sites is the protected schema sometimes know by the name ?pure schema'. Under this, the DBA sets up an account that is not linked with any specific database user. This account is used as an ownership account for all the database objects : tables, views, and so on. Public synonyms are set up for each of the database object, and grants are made to each user for each of the database object. Therefore, a single user owns the objects, but the account can be constrained by not issuing passwords to any users apart from those who perform database object maintenance.

❖ **Creating Roles :**

Apart from defining the privileges essential for each role and selecting suitably descriptive names, the process of creating a role is very simple. The syntax is very similar to creating a database user. For example,

```
% sqlplus system
Password :.....
Connected.
SQL>create role global
dept; Role created.
```

13.5.8 Database Auditing :

Auditing is reactive function. It gives DBA information about an activity only after it has already occurred and this reactive information provides us a snapshot of what has occurred, depending on the level of detail being audited. It gives the DBA a basis for tracking the changes within the database.

Auditing extra rows to be inserted for each operation causes the database, it auditing requirements such as performance overhead and physical storage

to balance the odds being against becomes more important. Site–specific considerations require otherwise, unless the DBA is always limit the amount of information must be audited. DBAs users are added to the database, which tracks the higher level, it is not unusual to run audit trails. Database the SQL statements being issued by all users to track the time it is more unusual. As a rule of thumb, he suspected improper activity, the lower level of the DBA should be applied to the audit, and the audit must be specific about which is directed against.

To activate auditing for a database instance, the DBA must make certain that the AUDIT_TRAIL parameter of the INIT.ORA parameter file is set to DB or OS to indicate where the audit trail should be written. The default value for this parameter is NONE

❑ **Check Your Progress – 4 :**

1. Explain object security model.

.....

.....

.....

.....

.....

13.6 Recovery from Various Problems of Volatile and Non–Volatile Storage Devices :

A major responsibility of the database administrator is to prepare for the possibility of hardware, software, network, process, or system failure. If such a failure affects the operation of a database system, you must usually recover the database and return to normal operation as quickly as possible. Recovery should protect the database and associated users from unnecessary problems and avoid or reduce the possibility of having to duplicate work manually.

Recovery processes vary depending on the type of failure that occurred, the structures affected, and the type of recovery that you perform. If no files are lost or damaged, recovery may amount to no more than restarting an instance. If data has been lost, recovery requires additional steps.

❑ **Check Your Progress – 5 :**

1. Explain the difference between the three storage types–volatile, nonvolatile and stable.

.....

.....

2. The log is a sequence of _____ recording all the update activities in the database.

   (a) Log records (b) Records (c) Entries (d) Redo

3. In the _____ scheme, a transaction that wants to update the database first creates a complete copy of the database.

   (a) Shadow copy (b) Shadow Paging

   (c) Update log records (d) All of the mentioned

4. The _____ scheme uses a page table containing pointers to all pages; the page table itself and all updated pages are copied to a new location.

   (a) Shadow copy (b) Shadow Paging

   (c) Update log records (d) All of the mentioned

5. The current copy of the database is identified by a pointer, called _____ which is stored on disk.

   (a) Db–pointer (b) Update log

   (c) Update log records (d) All of the mentioned

13.7 Let Us Sum Up :

The structure of storage can be divided into various categories :

- **Volatile storage –** As the name suggests, this type of storage system does not survive from system crashes and mostly placed very near to CPU by embedding them onto the chipset itself for examples : main memory, cache memory. They are considered to be very fast but can only store a small amount of information.
- **Nonvolatile storage –** These memories have been made in order to survive from system crashes. They are huge enough in data storage capacity but slower in accessibility. Examples may include, hard disks, magnetic tapes, flash memory, non–volatile (battery backed up) RAM.
- **Stable storage –** Survives everything. Stable storage cannot really be implemented because all storage devices are made of hardware, and all hardware is vulnerable to mechanical or electronic device failures. Database systems approximate stable storage by writing data to multiple storage devices simultaneously. Even if one of the devices crashes, the data will still be available on a different device. Thus data loss becomes extremely unlikely.

13.8 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 13.2

- **Check Your Progress 2 :**

  See Section 13.3

- **Check Your Progress 3 :**

  See Section 13.4

- **Check Your Progress 4 :**

  See Section 13.5

- **Check Your Progress 5 :**

  1 : See Section 13.3 **2 :** Log records

  3 : Shadow copy **4 :** Shadow pagging

  5 : Db–Pointer

13.9 Glossary :

ATA (Advanced Technology Attachment) : A disk drive interface standard for IDE (Integrated Drive Electronics). A standard for storage devices that lets them be treated as if they were hard drives on the system. Any ATA compatible media can be read by any ATA device.

Access : Retrieval of data from or transfer of data into a storage device or area such as RAM or a register.

Access Time : The amount of time, including seek time, latency and controller time, needed for a storage device to retrieve information.

Adaptive Caching : A feature of hard disk drives that enables them to improve performance and throughput by adapting to the application being run.

13.10 Assignment :

Identify and understand the recovery mechanisms which apply when?

13.11 Activity :

Create the tables and recover the tables and users.

13.12 Case Study :

Read and understand the real time database crash and how it was recovered.

13.13 Further Reading :

1. Fundamentals of Database Management Systems – M. L. Gillenson, Wiely Publishing
2. Understanding Database management Systems – J. A. Vasta, Wadsworth Publishing Company

Unit
14

RECOVERY MECHANISMS

UNIT STRUCTURE

14.0 Learning Objectives

14.1 Introduction

14.2 Concept–Properties–States of Transaction

14.3 Introduction to Mechanisms

- **14.3.1 Log**
- **14.3.2 Deferred Update**
- **14.3.3 Immediate Update**
- **14.3.4 Caching/Buffering**
- **14.3.5 Checkpoint**
- **14.3.6 Shadow Paging**

14.4 Let Us Sum Up

14.5 Suggested Answer for Check Your Progress

14.6 Glossary

14.9 Assignment

14.10 Activities

14.11 Case Study

14.12 Further Readings

14.0 Learning Objectives :

After learning this unit, you will be able to understand :

- Transaction management in DBMS
- ACID property in detail
- Transaction management mechanisms

14.1 Introduction :

Database systems, like any other computer system, are subject to failures but the data stored in it must be available as and when required. When a database fails it must possess the facilities for fast recovery. It must also have atomicity i.e. either transactions are completed successfully and committed (the effect is recorded permanently in the database) or the transaction should have no effect on the database.

There are both automatic and non–automatic ways for both, backing up of data and recovery from any failure situations. The techniques used to recover the lost data due to system crash, transaction errors, viruses, catastrophic failure, incorrect commands execution etc. are database recovery techniques. So to prevent data loss recovery techniques based on deferred update and immediate update or backing up data can be used.

Recovery techniques are heavily dependent upon the existence of a special file known as a system log. It contains information about the start and end of each transaction and any updates which occur in the transaction. The log keeps track of all transaction operations that affect the values of database items. This information is needed to recover from transaction failure.

- The log is kept on disk start_transaction(T) : This log entry records that transaction T starts the execution.
- read_item(T, X) : This log entry records that transaction T reads the value of database item X.
- write_item(T, X, old_value, new_value) : This log entry records that transaction T changes the value of the database item X from old_value to new_value. The old value is sometimes known as a before an image of X, and the new value is known as an afterimage of X.
- commit(T) : This log entry records that transaction T has completed all accesses to the database successfully and its effect can be committed (recorded permanently) to the database.
- abort(T) : This records that transaction T has been aborted.
- checkpoint : Checkpoint is a mechanism where all the previous logs are removed from the system and stored permanently in a storage disk. Checkpoint declares a point before which the DBMS was in consistent state, and all the transactions were committed.

14.2 Concept–Properties–States of Transaction :

A Transaction should be understood as a Logical unit of database processing that includes one or more access operations (read –retrieval, write – insert or update, delete).

A transaction (set of operations) may be stand–alone specified in a high level language like SQL submitted interactively, or may be embedded within a program. The Transaction boundaries are Begin and End transaction. An application program may contain several transactions separated by the Begin and End transaction boundaries.

❖ **ACID Properties of Transaction :**

A transaction may hold several low level tasks and additional a transaction is a very little unit of any program. Any transaction in a database system must always uphold some properties in order to ensure the correctness of its completeness and data integrity. These properties are referred as ACID properties and have been mentioned below :

- **Consistency –** It states that after finishing of the transaction, its database must remain in a consistent state and there should not be any chance of some data is incorrectly affected by the execution of transaction. If the database is in a consistent state before the execution of the transaction, it must stay in consistent state after the execution of the transaction.
- **Durability –** This states that in any case all the updates that have been made on the database should be preserved even if the system fails or restarts. If a transaction writes or updates some data in database and commits any type of error then too that data will always be present in the database.

- **Atomicity –** Including many low–level operations of a transaction, but the property of a transaction executed or not their actions are always either means which should be treated as an atomic unit states that although. Left partially completed transactions where the state should not be in the database. States very clearly and properly or before the execution of the transaction or the transaction execution / abortion / failure should be defined after.
- **Isolation –** In a system of database where there is more than one transactions being executed simultaneously, the property of isolation states that all the transactions will be approved and executed as if it is the only transaction in the system. None of the transaction will affect the existence of any other transaction.

❖ **Transaction Failure :**

Whenever a transaction fails to perform or it reaches a point after which it is unable to be performed successfully it has to abort. This is called transaction failure where only few transaction or process are affected.

There are number of reasons because of which a transaction fails, they are

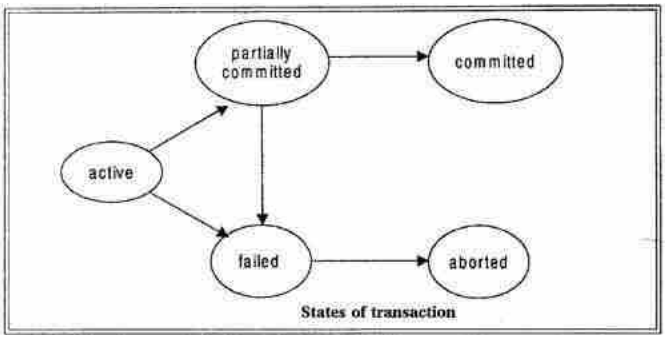
- **Logical errors –** Whenever a transaction cannot completed because of any type of code error or any internal error condition
- **System errors –** When the dbs itself ends an active transaction because DBMS is not able to execute it or it has to stop because of some system condition. For example, in case of deadlock or resource unavailability systems aborts an active transaction.
- **System Crash –** There are several problems, which are external to the system, and that may cause the system to stop abruptly and may cause the system to crash. For example due to failure in power supply, failure of underlying hardware or software failure.
- **Disk Failure –** During the very early days of technology evolution, it was a very common problem being faced by people where hard disk drives failure or storage drives failure was a very common issue. It include formation of bad sectors, unreachability to the disk, disk head crash or any other failure, which destroys all or part of disk storage.

❖ **Transaction states :**

A transaction must necessarily be in one of the following states :

- **Active –** This is the initial state, the transaction stays in this state while it is executing or performing.
- **Partially committed –** This transaction comes after the final statement has been executed.
- **Failed –** This occurs when the normal execution can no longer proceed.
- **Aborted –** This occurs after the transaction has been rolled back and the database has been restored to its state prior to the start of the transaction.
- **Committed –** after successful completion of transaction.

The state diagram corresponding to a transaction is shown in Figure.



States of transaction

We say that a transaction has committed only if it has entered the committed state. Similarly, we say that a transaction has aborted only if it has entered the aborted state. A transaction is said to have terminated if has either committed or aborted.

A transaction gets started in the active state. When it finishes, it enters the partially committed state and at this point, the transaction has completed its execution, but it is still possible that it may have to be aborted, since the actual output may still be temporarily hiding in main memory and thus a hardware failure may preclude its successful completion.

The database system then writes out enough information to disk that, even in the event of a failure, the updates performed by the transaction can be recreated when the system restarts after the failure. When the last of this information is written out, the transaction enters the committed state.

❑ **Check Your Progress – 1 :**

1. Explain ACID properties of transaction.

.....

.....

.....

.....

.....

14.3 Introduction to Mechanisms :

❖ **Database Recovery :**

Purpose of Database Recovery is to bring the database into the last consistent state, which existed prior to the failure and to preserve transaction properties (Atomicity, Consistency, Isolation and Durability).

In the event of crash of a system, it many have number of transactions being executed and different files opened for modification of data items. As we are aware that every transaction is made up of so many operations, which are very atomic in nature. But according to the ACID properties of DBMS, atomicity of transactions should always be maintained that is, either all operations are executed or none.

After the recovery of a DBMS from a crash it should be in a position to preserve the following :

- The system must make sure the states of all transactions, which were being executed.

- It may have happen that a transaction may have been in the middle of any operation; so DBMS should make sure the atomicity of transaction in this case is not lost
- DBMS should be able to make sure whether the transaction can be completed or needs to be rolled back.
- None of the transactions would be allowed to leave DBMS in an inconsistent state.

One should note that the two types of techniques, which can help DBMS in recovering as well as maintaining the atomicity of transaction to a very great extent, are :

- Proper maintenance of records or logs for each and every transaction, and writing them onto some steady and safe storage before actually modifying the database.
- Maintainence of the shadow paging properly, when the changes are done on a volatile memory and later the actual database is updated.

Thus a database recovery is the process of eliminating the effects of a failure from the database. A failure is a state where data inconsistency is visible to transactions if they are scheduled for execution.

14.3.1 Log :

Most widely used structure for recording database modification is the Log. The Log is a sequence of records and maintains a record of all the updates activities in the database.

Following data recorded in Log :

- **Transaction identifier –** Transaction that performed write operation is called Transaction identifier.
- **Data item identifier –** Unique identifier of data– item writer.
- **Old value –** Value of data – item prior to the write.
- **New value –** Value of the data–item after the write.
- Commit transaction marker or, abort or Rollback transaction marker.
- When the transaction T_i starts, it registers itself by writing a $<T_i$ start> log record
- And before Ti executes write(X), a log record $<T_i, X, V_1, V_2>$ is written, where V_1 is the value of X before the write, and V_2 is the value to be written to X.
- Log record notes that T_i has performed a write on data item X_j, X_j had value, V_1 before the write, and will have value V_2 after the write.
- When T_i finishes it last statement, the log record $<T_i$ commit> is written.
- We assume for now that log records are written directly to stable storage (that is, they are not buffered)
- Two approaches using logs
- Deferred database modification
- Immediate database modification

1. **Deferred database modification –** All logs are written on to a very steady storage and the database is updated as soon as transaction is committed.
2. **Immediate database modification –** Each and every entry is a real database modification. Is a database that has been adapted immediately after each operation.

14.3.2 Deferred update :

This technique does not physically update the database on disk until a transaction has reached its commit point. Before reaching commit, all transaction updates are recorded in the local transaction workspace. If a transaction fails before reaching its commit point, it will not have changed the database in any way so UNDO is not needed. It may be necessary to REDO the effect of the operations that are recorded in the local transaction workspace, because their effect may not yet have been written in the database. Hence, a deferred update is also known as the No–undo/redo algorithm

14.3.3 Immediate update :

In the immediate update, the database may be updated by some operations of a transaction before the transaction reaches its commit point. However, these operations are recorded in a log on disk before they are applied to the database, making recovery still possible. If a transaction fails to reach its commit point, the effect of its operation must be undone i.e. the transaction must be rolled back hence we require both undo and redo. This technique is known as **undo/ redo algorithm.**

14.3.4 Caching/Buffering :

In this one or more disk pages that include data items to be updated are cached into main memory buffers and then updated in memory before being written back to disk. A collection of in–memory buffers called the DBMS cache is kept under control of DBMS for holding these buffers. A directory is used to keep track of which database items are in the buffer. A dirty bit is associated with each buffer, which is 0 if the buffer is not modified else 1 if modified.

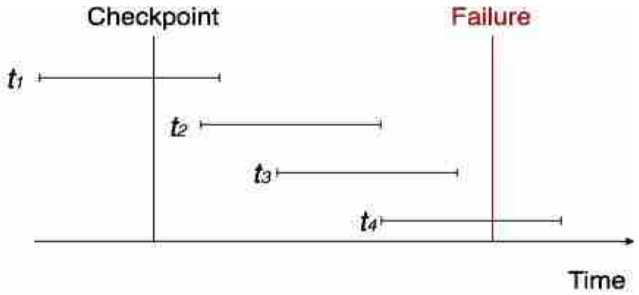
14.3.5 Check Point :

More than a transaction executed in parallel, when logs are interleaved and recovery time after all the logs have been difficult to recover the system, and then will start to recover. To avoid such a situation, most of the modern DBMS 'checkpoints' use of the concept.

The proper maintenance of logs in real time and environment may sometime fill out all the memory space on the system. And as the time passes log file may be too big to be handled at all. Thus, checkpoint is one of the mechanism where all the previous logs are removed from the system and stored enduringly in storage disk. Checkpoint declares a point before which the DBMS was in consistent state and all the transactions were committed.

❖ **Recovery :**

When a system with concurrent transaction crashes and recovers, it does act in the following manner :



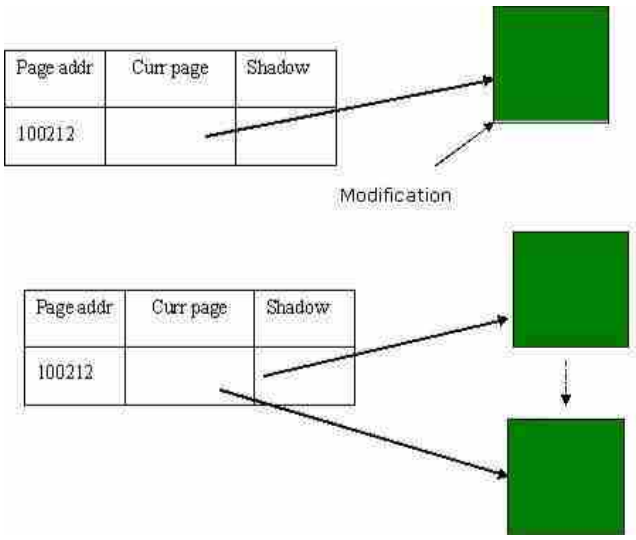
- The recovery system makes reading the logs backwards i.e. from the end to the preceding Checkpoint.
- It builds two lists, undo–list and redo–list.
- If a recovery system shows a log or record with <T_n, Start> and <T_n, Commit> or just <T_n, Commit>, it puts the transaction ina redo–list.
- If the recovery system shows a log or record with <T_n, Start> but no commit or abort log found, it puts the transaction ina undo–list.

Thereafter all the transactions in undo–list are undone and their logs are removed. All transaction in redo–list, their previous logs are detached and then redone another time and log saved.

14.3.6 Shadow Paging :

Shadow paging is a substitute to log based recovery techniques, which has both advantages as well as disadvantages. It may require fewer disk accesses, but it is very difficult to extend paging to allow multiple contemporaneous transactions. The paging is very much alike to paging schemes used by the operating system for memory management.

The idea is to maintain a two page tables during the life of a transaction : the current page table and the shadow page table. When the transaction starts, both tables are similar. The shadow page is never changed throughout the life of the transaction but the current page is updated with each write operation. Each table entry points to a page on the disk. When the transaction is performed, the shadow page entry becomes a copy of the recent page table entry and the disk block with the old data is released. If the shadow is stored in non–volatile memory and a system crash occurs, then the shadow page table is copied to the current page table. This ensures that the shadow page table will point to the database pages analogous to the state of the database prior to any transaction that was active at the time of the crash, making aborts automatic.



❑ **Check Your Progress – 2 :**

1. Explain Log based recovery.
2. If a transaction does not modify the database until it has committed, it is said to use the _____ technique.
   (a) Deferred–modification (b) Late–modification
   (c) Immediate–modification (d) Undo
3. If database modifications occur while the transaction is still active, the transaction is said to use the _____technique.
   (a) Deferred–modification (b) Late–modification
   (c) Immediate–modification (d) Undo
4. _____ using a log record sets the data item specified in the log record to the old value.
   (a) Deferred–modification (b) Late–modification
   (c) Immediate–modification (d) Undo
5. In the _____ phase, the system replays updates of all transactions by scanning the log forward from the last checkpoint.
   (a) Repeating (b) Redo (c) Replay (d) Undo
6. _____ module of a DBMS that restores the database to a correct condition
   (a) Transaction Manager (b) Recovery Manager
   (c) Save Point (D) None

14.4 Let Us Sum Up :

It is very important to appreciate that there are two types of privileges :

First is system privileges and the second is object privileges. They are considered to be the foundation of anything that a user can do while he is connected to an Oracle database.

The concept of a security policy should not be taken an easy thing to put into practice, especially in those environments in which one has never existed. Even so, it is a crucial piece of the database setup that should not be unnoticed.

Database security is a mission that the DBA must accept.

Password authentication is the more traditional method of user authentication. Although passwords are relatively secure, they have shortcomings that can make them more vulnerable to breach or attack.

DBA must also consider other factors when setting up a security plan. Not only the setup of the database users should be measured, but also the ownership of the database objects.

Aside from defining the privileges necessary for each role and selecting appropriately descriptive names, the process of creating a role is simple.

The Auditing gives the DBA the power to track information within the database. It gives information on who performed a certain operation and function and when was it performed. This is an influential security feature of the Oracle RDBMS, but this comes for a price.

A computer system like any other device is subject to failure. There are varieties of causes of such failure like disk crash, power failure, software errors etc. In addition to system failures, transaction may also fail due to violation of integrity or deadlock.

The storage structure can be divided in various categories : Volatile storage, Non–volatile storage and stable storage.

A Transaction is Logical unit of database processing that includes one or more access operations (read –retrieval, write – insert or update, delete).A transaction (set of operations) may be stand–alone specified in a high level language like SQL submitted interactively, or may be embedded within a program.

A transaction in a database system must maintain some properties in order to ensure the accuracy of its completeness and data integrity. These properties are refer to as ACID properties (Atomicity, Consistency, Isolation and Durability).

14.5 Suggested Answer for Check Your Progress :

- **Check Your Progress 1 :**

  See Section 14.2

- **Check Your Progress 2 :**

  1 : See Section 14.3.1 **2 :** Deferred-modification

  3 : Immediate-modification **4 :** Undo

  5 : Redo **6 :** Recovery Manager

14.6 Glossary :

1. **Aborted transaction –** A transaction in progress that terminates abnormally.
2. **Checkpoint –** A facility by which a DBMS periodically refuses to accept any new transactions. The system is in a quiet state, and the database and transaction logs are synchronized.
3. **Database log –** A log that contains before and after images of records that have been modified by transactions.
4. **Database recovery –** Mechanisms for restoring a database quickly and accurately after loss or damage.
5. **Database security –** Protection of database data against accidental or intentional loss, destruction, or misuse.
6. **Recovery manager –** A module of a DBMS that restores the database to a correct condition when a failure occurs and then resumes processing user questions.
7. **Transaction –** A discrete unit of work that must be completely processed or not processed at all within a computer system. Entering a customer order is an example of a transaction.
8. **Transaction boundaries –** The logical beginning and end of a transaction.
9. **Transaction log –** A record of the essential data for each transaction that is processed against the database.
10. **Transaction manager –** In a distributed database, a software module that maintains a log of all transactions and an appropriate concurrency control scheme.

11. **Atomicity –** Either all operations of the transaction are properly reflected in the database or none are.
12. **Consistency –** Execution of a transaction in isolation preserves the consistency of the database.
13. **Isolation –** Although multiple transactions may execute concurrently, each transaction must be unaware of other concurrently executing transactions. Intermediate transaction results must be hidden from other alongside executed transactions.
14. **Durability –** After a transaction completes successfully, the changes it has made to the database persevere, even if there are system failures.

14.7 Assignment :

Why Database security is important in DBMs? – Comment. How can we recovery any database in case of failure

14.8 Activities :

Study different recovery techniques in detail.

14.9 Case Study :

Suggest a scheme for the secure storage of passwords.

14.10 Further Reading :

1. Database Management Systems – Rajesh Narang –PHI Learning Pvt Ltd
2. Database System Concepts by Silberschatz, Korth –Tata McGraw–Hill Publication
3. An Introduction to Database Systems – Bipin Desai– Galgotia Publication
4. Database Management System by Raghu Ramkrishnan– Tata McGraw–Hill Publication
5. SQL, PL/SQL : The Programming Language Oracle – Ivan Bayross– BPB Publication

BLOCK SUMMARY :

Uni 11 provides the introduction of database types and detail information about different types of databases.

Unit 12 provides overview of Data Ware housing and Data mining. Generally a data warehouses adopts three–tier architecture. From the perspective of data warehouse architecture, data warehouse models are Virtual Warehouse, Data mart and Enterprise Warehouse. Typically, the data in a data warehouse is loaded through the process of ETL, i.e. extraction, transformation and loading, from one or more sources of data. Data Mining, also popularly known as Knowledge Discovery in Databases (KDD), refers to the nontrivial extraction of implicit, previously unknown and potentially useful information from data in databases. The iterative process of KDD consists of the steps : Data cleaning, Data integration, Data selection, Data transformation, Data mining, Pattern evaluation and knowledge representation. The data mining functionalities are Characterization, Discrimination, Association analysis, Classification, Prediction, Clustering, Outlier analysis and Evolution and deviation analysis. Applications of data mining are like Market Analysis, Fraud Detection, Customer Retention, Production Control and Science Exploration etc.

Unit 13 & 14 provides you the concept of Database security and recovery. The database stored in the database need protection from unauthorized access and malicious destruction or alteration. Database security refers to protection from malicious access. To protect database, security measures can be taken at several levels like database system, operating system, network, physical and human .This unit also explained the concept of database recovery. The different recovery mechanisms are also explained in unit 14.

A computer system like any other device is subject to failure. There are varieties of causes of such failure like disk crash, power failure, software errors etc. In addition to system failures, transaction may also fail due to violation of integrity or deadlock. An integral part of database system is recovery scheme that is responsible for detection of failures and for restoration of database to state that existed before occurrence of failure. So it is responsibility of recovery scheme to ensure atomicity and durability property. There are basically two approaches for ensuring these properties : log–based schemes and shadow paging.

BLOCK ASSIGNMENT :

❖ **Short Questions :**

1. Define the terms : Data Warehouse, Data Warehousing
2. What is data mining ?
3. What is ETL process ?
4. Why to protect passwords ?
5. What is check point ?

❖ **Long Questions :**

1. Explain the need of data warehouse and architecture of data warehouse.
2. Explain the steps in data mining or KDD process and give its applications.
3. Explain properties and states of transaction ?

❖ **Enrolment No. :** ☐

1. How many hours did you need for studying the units ?

Unit No.	11	12	13	14
No. of Hrs.				

2. Please give your reactions to the following items based on your reading of the block :

Items	Excellent	Very Good	Good	Poor	Give specific example if any
Presentation Quality	☐	☐	☐	☐	_____
Language and Style	☐	☐	☐	☐	_____
Illustration used (Diagram, tables etc)	☐	☐	☐	☐	_____
Conceptual Clarity	☐	☐	☐	☐	_____
Check your progress Quest	☐	☐	☐	☐	_____
Feed back to CYP Question	☐	☐	☐	☐	_____

3. Any other Comments

.....

.....

.....

.....

.....

.....

.....

.....

DR.BABASAHEB AMBEDKAR
OPEN UNIVERSITY

'Jyotirmay' Parisar,
Sarkhej-Gandhinagar Highway, Chharodi, Ahmedabad-382 481.
Website : www.baou.edu.in