KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed to be University)

(Established Under Section 3 of UGC Act, 1956)

Coimbatore - 641 021, India

FACULTY OF ARTS, SCIENCE AND HUMANITIES (FASH)

Department of CS,CA & IT

II B.Sc CS **IV SEMESTER** **BATCH : 2016 - 2019**

16CSU403 **DATA BASE MANAGEMENT SYSTEMS** **4H – 4C**

Instruction Hours / week: L: 4 T: 0 P: 0 **Marks:** Int : **40** Ext : **60** Total: **100**

SCOPE

The Objective of Database Management System includes learning of relational data model, database design and Transaction model.

OBJECTIVES

- Understand the role and nature of relational database management systems in today's IT environment.
- Translate written business requirements into conceptual entity-relationship data models.
- Manipulate a database using SQL.
- Assess the quality and ease of use of data modeling and diagramming tools.

LEARNING OUTCOMES

At the end of the course, students will be able to,

- To learn the basic concepts of DBMS
- To Know the concepts of SQL
- To understand PL/SQL, Triggers and cursors
- To know the concept of Normalization

UNIT-I

Introduction: Characteristics of database approach, data models, database system architecture and data independence. **Entity Relationship(ER) Modeling:** Entity types, relationships, constraints.

UNIT-II

Relation data model: Relational model concepts, relational constraints, relational algebra.

UNIT-III

Relation data model: SQL queries **Database design:** Mapping ER/EER model to relational database, functional dependencies, Lossless decomposition.

UNIT-IV

Database design: Normal forms (upto BCNF).**Transaction Processing :** ACID properties, concurrency control

UNIT-V

File Structure and Indexing (8 Lectures) Operations on files, File of Unordered and ordered records, overview of File organizations, Indexing structures for files(Primary index, secondary index, clustering index), Multilevel indexing using B and B+ trees.

References:

1. Elmasri, R., & Navathe, S.B. (2011). Fundamentals of Database Systems (6th ed.). New Delhi: Pearson Education,.
2. Ramakrishanan, R., & Gehrke, J. (2002). Database Management Systems (3rd ed.). New Delhi: McGraw-Hill.
3. Silberschatz, A., Korth, H.F., & Sudarshan, S. (2010).Database System Concepts (6th ed.). New Delhi: McGraw-Hill
4. Elmasri, R., & Navathe, S.B. (2013). Database Systems Models, Languages, Design and application Programming (6th ed.). New Delhi: Pearson Education.

WEB SITES

1. http://en.wikipedia.org/wiki/RDBMS
2. http://aspalliance.com/1211_Relational_Database_Management_Systems__Concepts_and_Terminologies
3. www.compinfo-center.com/apps/rdbms.html

KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed to be University)

(Established Under Section 3 of UGC Act 1956)

Coimbatore – 641 021.

LECTURE PLAN
DEPARTMENT OF COMPUTER SCIENCE

STAFF NAME : A.JEEVARATHINAM

SUBJECT NAME : DATABASE MANAGEMENT SYSTEM SUB.CODE : 16CSU403

SEMESTER : IV CLASS : II B.Sc (CS)

UNIT I

S.NO	Lecture Duration (Hours)	Topics To Be Covered	Support Materials/ Pg.No
		Introduction	
1	1	Characteristics of database approach	T1 : 9
2	1	Data models	T1 : 30
3	1	Database system architecture	T1 : 33
4	1	Data independence	T1 : 35
		Entity Relationship(ER) Modeling	
5	1	Entity types	T2 : 262
6	1	Relationships	T2 : 274
7	1	ER Diagram	
8	1	Constraints	T2 : 269
9	1	Examples	W1
10	1	Recapitulation and Discussuin of Important Questions	
		Total No of Hours Planned for Unit I	**10**

UNIT II

S.NO	Lecture Duration (Hours)	Topics To Be Covered	Support Materials/ Pg.No
		RELATIONAL DATA MODEL	
1	1	Relational model concepts	T1:60
		Relational constraints	T3:63-68
2	1	Key Constraints	
3	1	Foreign Key Constraints	
4	1	General Constraints	
		Relational Algebra	T3:102-110 ,W2
5	1	Selection and Projection	
6	1	Set Operations	
7	1	Renaming	
8	1	Joins	
9	1	Division	
10	1	Recapitulation and Discussuin of Important Questions	
		Total No of Hours Planned for Unit II	**10**

UNIT III

S.NO	Lecture Duration (Hours)	Topics To Be Covered	Support Materials/ Pg.No
		RELATIONAL DATA MODEL	
		SQL Queries	
		Basic SQL	T1:89
1	1	INSERT, DELETE	T1:107
2	1	UPDATE Statements in SQL	
		More SQL	
3	1	Complex Queries	T1:115
4	1	TriggerS	T1:131,W3
5	1	Views	T1:133
6	1	Schema Modification	T1:137
		Database design	
7	1	ER model	T1:221
		EER Model	
8	1	Subclasses, Superclasses and Inheritance	T1:246
9	1	Constraints and Characteristics of Specialization and Generalization Hierarchies	T1:251
10	1	Modeling of UNION Types Using Categories	T1:258
11	1	Mapping ER/EER model to relational database	T1:286
12	1	functional dependencies	T1:513
13	1	Lossless decomposition	T3:619
14	1	Recapitulation and Discussion of Important Questions	
		Total No of Hours Planned for Unit III	**14**

UNIT IV

S.NO	Lecture Duration (Hours)	Topics To Be Covered	Support Materials/ Pg.No
		DATA BASE DESING	
		Normal Forms	
1	1	Normal Forms Based on Primary Keys	T1 : 516 - 534,W3
2	1	General Definitions of Second and Third Normal Forms	
3	1	Boyce-Codd Normal Form	
4	1	Multivalued Dependency and Fourth Normal Form	
5	1	Join Dependencies and Fifth Normal Form	
		Transaction Processing	
6	1	Transaction Concept	T2 : 627 - 629
7	1	A Simple Transaction Model	
8	1	ACID properties	T3:520
		Concurrency control	T2 : 661-686
9	1	Lock-Based Protocols	
10	1	Deadlock Handling	
11	1	Multiple Granularity	
12	1	Timestamp-Based Protocols	
13	1	Validation-Based Protocols	
14	1	Recapitulation and Discussuin of Important Questions	
		Total No of Hours Planned for Unit IV	**14**

UNIT V

S.NO	Lecture Duration (Hours)	Topics To Be Covered	Support Materials/ Pg.No
		FILE STRUCTURE AND INDEXING	
		File Structure	
1	1	Operations on files	T1 : 599
2	1	File of Unordered records	T1 : 601
3	1	File of ordered records	T1 : 603
4	1	overview of File organizations	T2 : 451
		Indexing structures for files	
5	1	Primary index	T1 : 632 - 643
6	1	secondary index	
7	1	clustering index	
8	1	Multilevel indexing using B and B+ trees	T1 : 646
9	1	Recapitulation and Discussuin of Important Questions	
10	1	Discussion of previous ESE question papers	
11	1	Discussion of previous ESE question papers	
12	1	Discussion of previous ESE question papers	
		Total No of Hours Planned for Unit V	**12**

S.NO	TEXT BOOKS
T1	Elmasri, R., & Navathe, S.B. (2011). Fundamentals of Database Systems (6th ed.). New Delhi: Pearson Education
T2	Silberschatz, A., Korth, H.F., & Sudarshan, S. (2011).Database System Concepts (6th ed.). New Delhi: McGraw-Hill
T3	Ramakrishanan, R., & Gehrke, J. (2002). Database Management Systems (3rd ed.). New Delhi: McGraw-Hill

S.NO	WEB SITES
W1	https://www.tutorialspoint.com
W2	www.studytonight.com
W3	https://www.w3schools.com

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

SYLLABUS

Introduction: Characteristics of database approach, data models, database system architecture and data independence. **Entity Relationship(ER) Modeling:** Entity types, relationships, constraints.

INTRODUCTION

UNDERSTANDING DATABASE FUNDAMENTALS

DATA

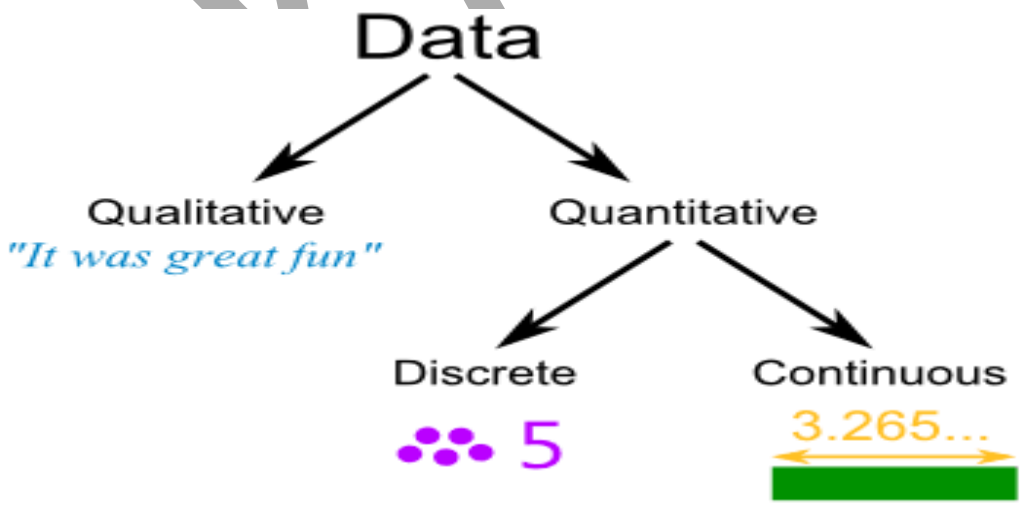
Data is a collection of facts, such as values or measurements.

It can be numbers, words, measurements, observations or even just descriptions of things.

Qualitative vs Quantitative

Data can be qualitative or quantitative.

- **Qualitative data** is descriptive information (it *describes* something)
- **Quantitative data**, is numerical information (numbers).



And **Quantitative data** can also be Discrete or Continuous:

- **Discrete data** can only take certain values (like whole numbers)
- **Continuous data** can take any value (within a range)

Put simply: **Discrete data** is counted, **Continuous data** is measured

INFORMATION

Information is valuable because it can affect behavior, a decision, or an outcome. For example, if a manager is told his/her company's net profit decreased in the past month, he/she may use this information as a reason to cut financial spending for the next month. A piece of information is considered valueless if, after receiving it, things remain unchanged. For a technical definition of information see information theory.

Information is defined as the knowledge of something; particularly, an event, situation, or knowledge derived based on research or experience.

Data is any information related to an organization that should be stored for any purpose according to the requirements of an organization.

DBMS

A database management system (DBMS) is the software that allows a computer to perform database functions of storing, retrieving, adding, deleting and modifying data. Relational database management systems (RDBMS) implement the relational model of tables and relationships.

Examples:

Microsoft Access, MySQL, Microsoft SQL Server, Oracle and FileMaker Pro are all examples of database management systems.

The following are examples of database applications:

- computerized library systems
- automated teller machines
- flight reservation systems
- computerized parts inventory systems

RDBMS

Short for **r***elational* **d***atabase* **m***anagement* **s***ystem* and pronounced as separate letters, a type of database management system (DBMS) that stores data in the form of related tables. Relational databases are powerful because they require few assumptions about how data is related or how it will be extracted from the database. As a result, the same database can be viewed in many different ways.

An important feature of relational systems is that a single database can be spread across several tables. This differs from flat-file databases, in which each database is self-contained in a single table.

SQL

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in relational database. SQL is the standard language for Relation Database System. All relational database management systems like MySQL, MS Access, Oracle, Sybase, Informix, postgres and SQL Server use SQL as standard database language.

Also, they are using different dialects, such as:

- MS SQL Server using T-SQL,
- Oracle using PL/SQL,

History:

- **1970 --** Dr. E. F. "Ted" of IBM is known as the father of relational databases. He described a relational model for databases.
- **1974 --** Structured Query Language appeared.
- **1978 --** IBM worked to develop Codd's ideas and released a product named System/R.
- **1986 -- IBM** developed the first prototype of relational database and standardized by ANSI. The first relational database was released by Relational Software and its later becoming Oracle.

SQL is pronounced as “S-Q-L” or “see-quill”.

SQL uses -- character sequence as a single line comment identifier.

SQL commands are not case sensitive and the following SQL queries are equivalent:

SELECT * FROM Users

select * from Users

Characteristics of Database Approach

1. Represent Some Aspects of real world applications

A database represents some features of real world applications. Any change in the real world is reflected in the database. If we have some changes in our real applications like railway reservation system then it will be reflected in database too.

For example, let us take railway reservation system; we have in our mind some certain applications of maintaining records of attendance, waiting list, train arrival and departure time, certain day etc. related to each train.

2. Manages Information

A database always takes care of its information because information is always helpful for whatever work we do. It manages all the information that is required to us. By managing information using a database, we become more deliberated user of our data.

Also See: What is Database?

3. Easy Operation implementation

All the operations like insert, delete, update, search etc. are carried out in a flexible and easy way. Database makes it very simple to implement these operations. A user with little knowledge can perform these operations. This characteristic of database makes it more powerful.

4. Multiple views of database

Basically, a view is a **subset of the database**. A view is defined and devoted for a particular user of the system. Different users of the system may have different views of the same system.

Every view contains only the data of interest to a user or a group of users. It is the responsibility of users to be aware of how and where the data of their interest is stored.

5. Data for specific purpose

A database is designed for data of specific purpose. **For example**, a database of student management system is designed to maintain the record of student's marks, fees and attendance etc. This data has a specific purpose of maintaining student record.

Also See: Advantages Of Database Management System

6. It has Users of Specific interest

A database always has some indented group of users and applications in which these user groups are interested.

For example, in a library system, there are three users, official administration of the college, the librarian, and the students.

7. Self Describing nature

A database is of self describing nature; it always describes and narrates itself. It contains the description of the whole data structure, the constraints and the variables.

It makes it different from traditional file management system in which definition was not the part of application program. These definitions are used by the users and DBMS software when needed.

8. Logical relationship between records and data

A database gives a logical relationship between its records and data. So a user can access various records depending upon the logical conditions by a single query from the database.

Data Model

A database model shows the logical structure of a database, including the relationships and constraints that determine how data can be stored and accessed. Individual database models are designed based on the rules and concepts of whichever broader data model the designers adopt. Most data models can be represented by an accompanying database diagram.

Types of database models

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

There are many kinds of data models. Some of the most common ones include:

- Hierarchical database model
- Relational model
- Network model
- Object-oriented database model
- Entity-relationship model
- Document model
- Entity-attribute-value model
- Star schema
- The object-relational model, which combines the two that make up its name

Flat Data Model

Flat data model is the first and foremost introduced model and in this all the data used is kept in the same plane. Since it was used earlier this model was not so scientific.

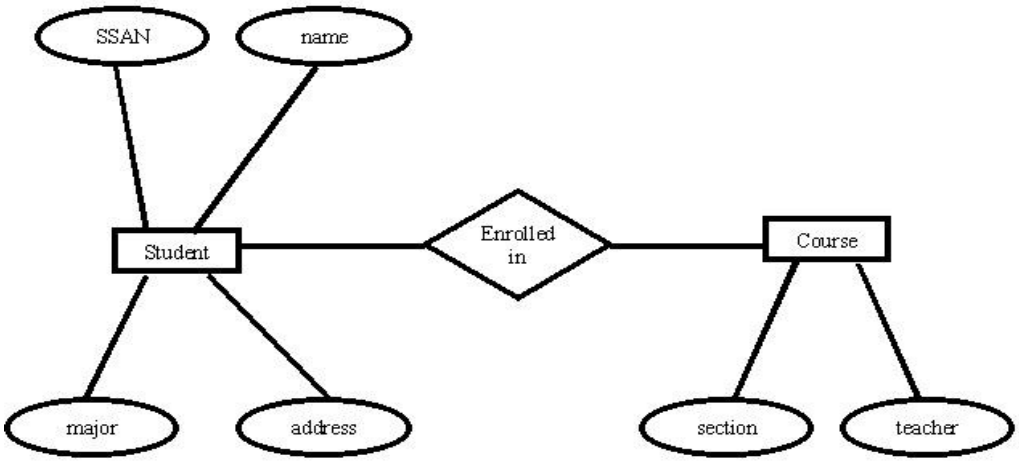
Roll No	Name	Course
5482	Mark	Web Designing
5486	Steve	Java
5496	Smith	Oracle

Entity Relationship Data Model

Entity relationship model is based on the notion of the real world entities and their relationships. While formulating the real world scenario in to the database model an entity set is created and this model is dependent on two vital things and they are :

- Entity and their attributes
- Relationships among entities

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

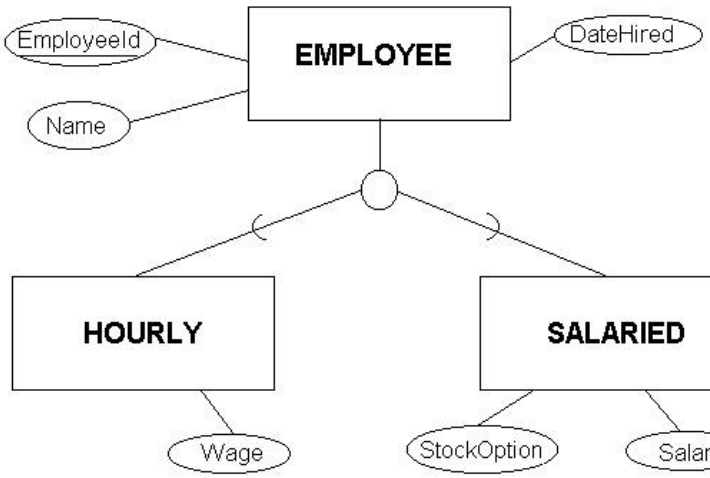


An entity has a real world property called attribute and attribute define by a set of values called domain. For example, in a university a student is an entity, university is the database, name and age and sex are the attributes. The relationships among entities define the logical association between entities.

Relational Data Model

Relational model is the most popular model and the most extensively used model. In this model the data can be stored in the tables and this storing is called as relation, the relations can be normalized and the normalized relation values are called atomic values. Each row in a relation contains unique value and it is called as tuple, each column contains value from same domain and it is called as attribute.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I



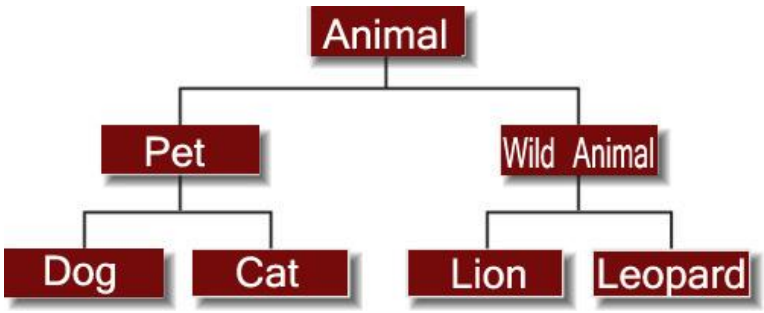
Network Data Model

Network model has the entities which are organized in a graphical representation and some entities in the graph can be accessed through several paths.

Hierarchical Data Model

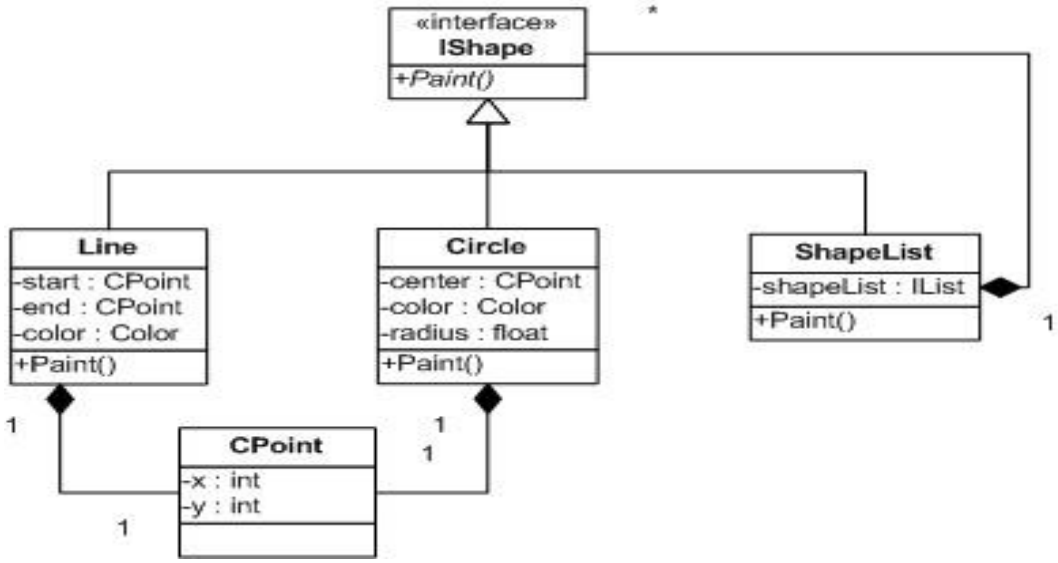
Hierarchical model has one parent entity with several children entity but at the top we should have only one entity called root. For example, department is the parent entity called root and it has several children entities like students, professors and many more.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I



Object oriented Data Model

Object oriented data model is one of the developed data model and this can hold the audio, video and graphic files. These consist of data piece and the methods which are the DBMS instructions.



Record base Data Model

Record base model is used to specify the overall structure of the database and in this there are many record types. Each record type has fixed no. of fields having the fixed length.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

Object relation Data Model

Object relation model is a very powerful model but coming to it's design it is quiet complex. This complexity is not problem because it gives efficient results and widespread with huge applications. It has a feature which allows working with other models like working with the very known relation model.

Semi structured Data Model

Semi structured data model is a self describing data model, in this the information that is normally associated with a scheme is contained within the data and this property is called as the self describing property.

Associative Data Model

Associative model has a division property, this divides the real world things about which data is to be recorded in two sorts i.e. between entities and associations. Thus, this model does the division for dividing the real world data to the entities and associations.

Database Architecture

Three Level Architecture of DBMS

Following are the three levels of database architecture,

1. Physical Level
2. Conceptual Level
3. External Level

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

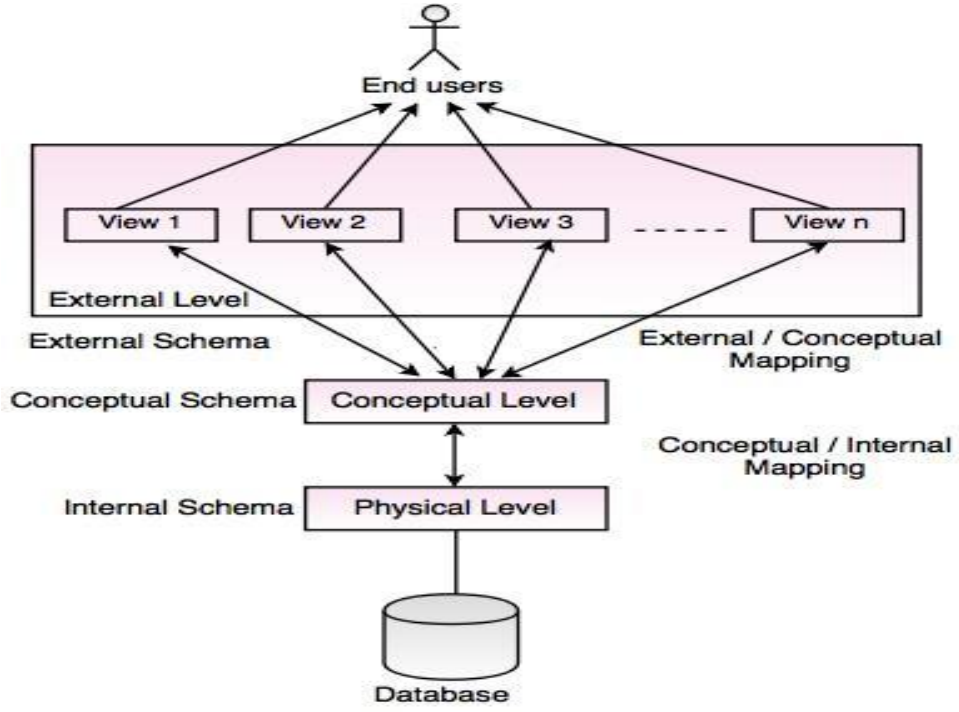


Fig. Three Level Architechture of DBMS

In the above diagram,

- It shows the architecture of DBMS.
- Mapping is the process of transforming request response between various database levels of architecture.
- Mapping is not good for small database, because it takes more time.
- In External / Conceptual mapping, DBMS transforms a request on an external schema against the conceptual schema.
- In Conceptual / Internal mapping, it is necessary to transform the request from the conceptual to internal levels.

1. Physical Level

- Physical level describes the physical storage structure of data in database.
- It is also known as Internal Level.
- This level is very close to physical storage of data.

- At lowest level, it is stored in the form of bits with the physical addresses on the secondary storage device.
- At highest level, it can be viewed in the form of files.
- The internal schema defines the various stored data types. It uses a physical data model.

2. Conceptual Level

- Conceptual level describes the structure of the whole database for a group of users.
- It is also called as the data model.
- Conceptual schema is a representation of the entire content of the database.
- These schema contains all the information to build relevant external records.
- It hides the internal details of physical storage.

3. External Level

- External level is related to the data which is viewed by individual end users.
- This level includes a no. of user views or external schemas.
- This level is closest to the user.
- External view describes the segment of the database that is required for a particular user group and hides the rest of the database from that user group.

Data Abstraction and Data Independence

Database systems comprise of complex data-structures. In order to make the system efficient in terms of retrieval of data, and reduce complexity in terms of usability of users, developers use abstraction i.e. hide irrelevant details from the users. This approach simplifies database design.

There are mainly **3** levels of data abstraction:

Physical: This is the lowest level of data abstraction. It tells us how the data is actually stored in memory. The access methods like sequential or random access and file organisation methods like B+ trees, hashing used for the same. Usability, size of

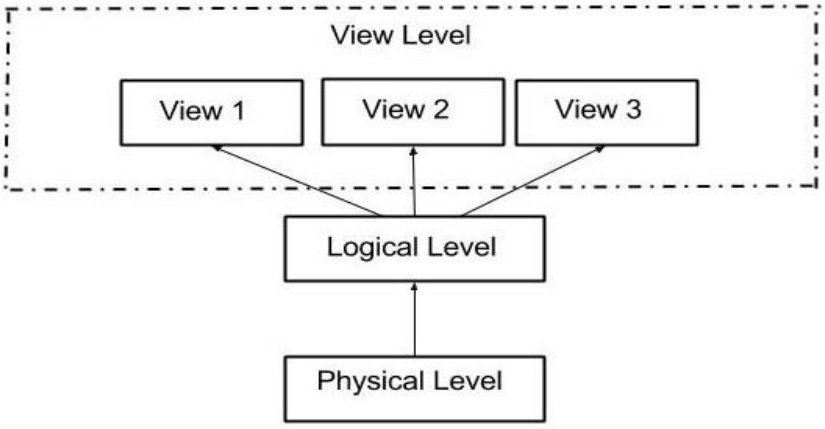
memory, and the number of times the records are factors which we need to know while designing the database.

Suppose we need to store the details of an employee. Blocks of storage and the amount of memory used for these purposes is kept hidden from the user.

Logical: This level comprises of the information that is actually stored in the database in the form of tables. It also stores the relationship among the data entities in relatively simple structures. At this level, the information available to the user at the view level is unknown.

We can store the various attributes of an employee and relationships, e.g. with the manager can also be stored.

View: This is the highest level of abstraction. Only a part of the actual database is viewed by the users. This level exists to ease the accessibility of the database by an individual user. Users view data in the form of rows and columns. Tables and relations are used to store data. Multiple views of the same database may exist. Users can just view the data and interact with the database, storage and implementation details are hidden from them.



The main purpose of data abstraction is achieving data independence in order to save time and cost required when the database is modified or altered.
We have namely two levels of data independence arising from these levels of abstraction :

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

Physical level data independence :

It refers to the characteristic of being able to modify the physical schema without any alterations to the conceptual or logical schema, done for optimisation purposes, e.g., Conceptual structure of the database would not be affected by any change in storage size of the database system server. Changing from sequential to random access files is one such example.These alterations or modifications to the physical structure may include:

- Utilising new storage devices.
- Modifying data structures used for storage.
- Altering indexes or using alternative file organisation techniques etc.

Logical level data independence:

It refers characteristic of being able to modify the logical schema without affecting the external schema or application program. The user view of the data would not be affected by any changes to the conceptual view of the data. These changes may include insertion or deletion of attributes, altering table structures entities or relationships to the logical schema etc.

ENTITY RELATIONSHIP (ER) MODELING

Introduction to ER Model

ER Model is a high-level data model, developed by Chen in 1976. This model defines the data elements and relationships for a specified system. It is useful in developing a conceptual design for the database & is very simple and easy to design logical view of data.

Importance of ER Model

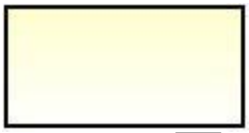
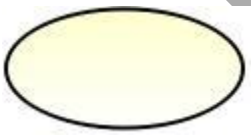
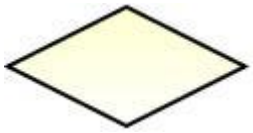
- ER Model is plain and simple for designing the structure.
- It saves time.
- Without ER diagrams you cannot make a database structure & write production code.

- It displays the clear picture of the database structure.

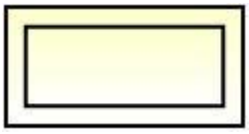
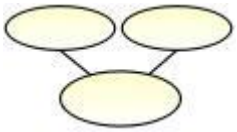
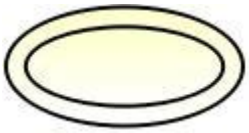
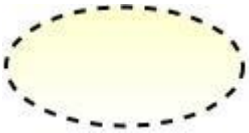
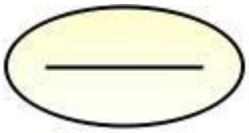
ER Diagrams

- ERD stands for Entity Relationship diagram.
- It is a graphical representation of an information system.
- ER diagram shows the relationship between objects, places, people, events etc. within that system.
- It is a data modeling technique which helps in defining the business process.
- It used for solving the design problems.

Following are the components of ER Diagram,

Notations	Representation	Description
	Rectangle	It represents the Entity.
	Ellipse	It represents the Attribute.
	Diamond	It represents the Relationship.
	Line	It represents the link between attribute and entity set to relationship set.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

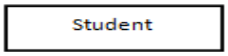
	Double Rectangle	It represents the weak entity.
	Composite Attribute	It represents composite attribute which can be divided into subparts. For eg. Name can be divided into First Name and Last Name
	Multi valued Attribute	It represents multi valued attribute which can have many values for a particular entity. For eg. Mobile Number.
	Derived Attribute	It represents the derived attribute which can be derived from the value of related attribute.
	Key Attribute	It represents key attribute of an entity which have a unique value in a table. For eg. Employee → EmpId (Employee Id is Unique).

Entity, Entity Type, Entity Set

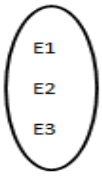
An Entity may be an object with a physical existence – a particular person, car, house, or employee – or it may be an object with a conceptual existence – a company, a job, or a university course.

An Entity is an object of Entity Type and set of all entities is called as entity set. e.g.; E1 is an entity having Entity Type Student and set of all students is called Entity Set. In ER diagram, Entity Type is represented as:

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I



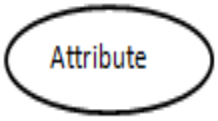
Entity Type



Entity Set

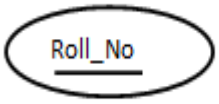
Attribute

Attributes are the **properties which define the entity type**. For example, Roll_No, Name, DOB, Age, Address, Mobile_No are the attributes which defines entity type Student. In ER diagram, attribute is represented by an oval.



Key Attribute

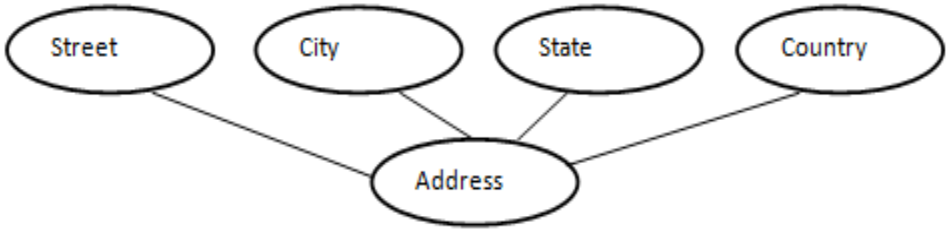
The attribute which **uniquely identifies each entity** in the entity set is called key attribute.For example, Roll_No will be unique for each student. In ER diagram, key attribute is represented by an oval with underlying lines.



Composite Attribute

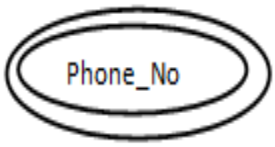
An attribute **composed of many other attribute** is called as composite attribute. For example, Address attribute of student Entity type consists of Street, City, State, and Country. In ER diagram, composite attribute is represented by an oval comprising of ovals.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I



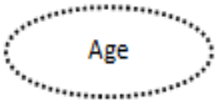
Multivalued Attribute

An attribute consisting **more than one value** for a given entity. For example, Phone_No (can be more than one for a given student). In ER diagram, multivalued attribute is represented by double oval.



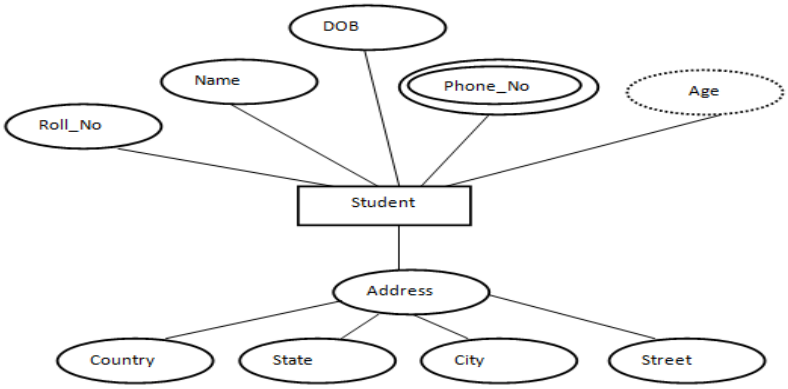
Derived Attribute

An attribute which can be **derived from other attributes** of the entity type is known as derived attribute. e.g.; Age (can be derived from DOB). In ER diagram, derived attribute is represented by dashed oval.



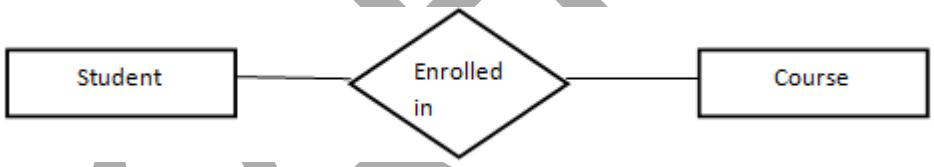
The complete entity type **Student** with its attributes can be represented as:

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

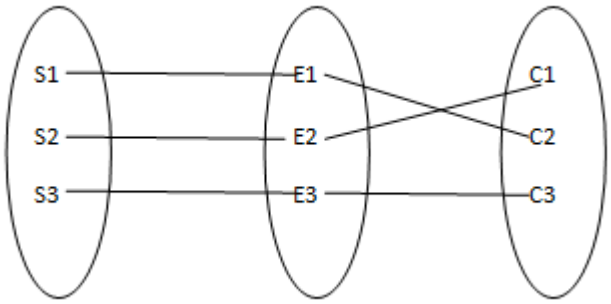


Relationship Type and Relationship Set

A relationship type represents the **association between entity types**. For example,'Enrolled in' is a relationship type that exists between entity type Student and Course. In ER diagram, relationship type is represented by a diamond and connecting the entities with lines.



A set of relationships of same type is known as relationship set. The following relationship set depicts S1 is enrolled in C2, S2 is enrolled in C1 and S3 is enrolled in C3.



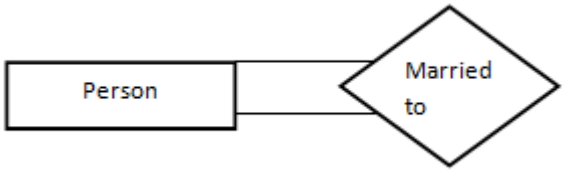
Degree of a relationship set

The number of different entity sets **participating in a relationship** set is called as degree of a relationship set.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

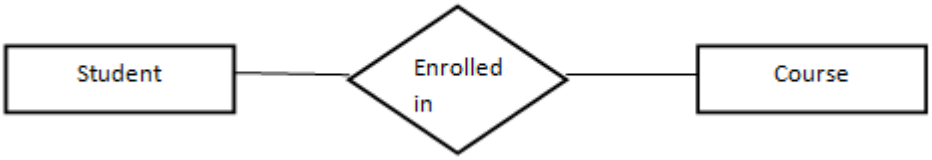
Unary Relationship

When there is **only ONE entity set participating in a relation**, the relationship is called as unary relationship. For example, one person is married to only one person.



Binary Relationship

When there are **TWO entities set participating in a relation**, the relationship is called as binary relationship.For example, Student is enrolled in Course.



n-ary Relationship

When there are n entities set participating in a relation, the relationship is called as n-ary relationship.

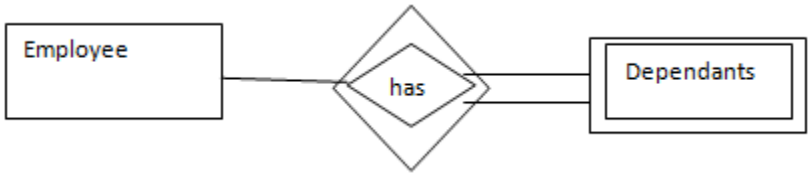
Weak Entity Type and Identifying Relationship

As discussed before, an entity type has a key attribute which uniquely identifies each entity in the entity set. But there exists **some entity type for which key attribute can't be defined**. These are called Weak Entity type.

For example, A company may store the information of dependants (Parents, Children, Spouse) of an Employee. But the dependents don't have existence without the employee. So Dependent will be weak entity type and Employee will be Identifying Entity type for Dependant.

A weak entity type is represented by a double rectangle. The participation of weak entity type is always total. The relationship between weak entity type and its identifying strong entity type is called identifying relationship and it is represented by double diamond.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I



Relationship Mapping in ER Diagram of Databases

Cardinality

The **number of times an entity of an entity set participates in a relationship** set is known as cardinality.

Cardinality can be of different types:

Following are the types of Relationship Mapping,

1. One - to - One Relationship
2. One - to - Many Relationship
3. Many - to - One Relationship
4. Many - to - Many Relationship

1. One - to - One Relationship

- In One - to - One Relationship, one entity is related with only one other entity.
- One row in a table is linked with only one row in another table and vice versa.

For example: A Country can have only one Capital City.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

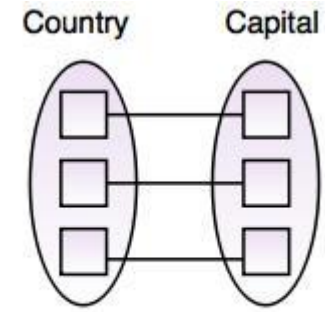


Fig. One to One Mapping

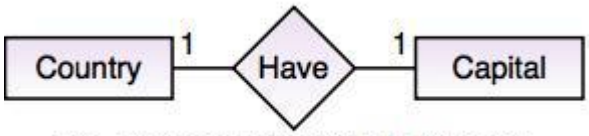
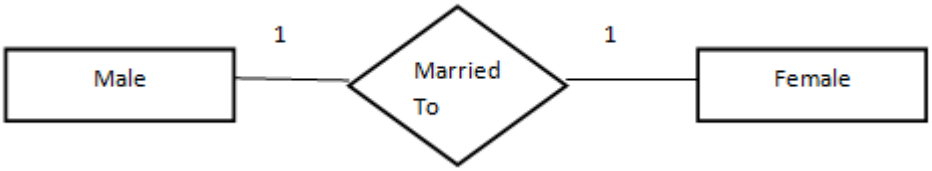


Fig. Representation in ER Diagram



2. One - to - Many Relationship

- In One - to - Many Relationship, one entity is related to many other entities.
- One row in a table A is linked to many rows in a table B, but one row in a table B is linked to only one row in table A.

For example: One Department has many Employees.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

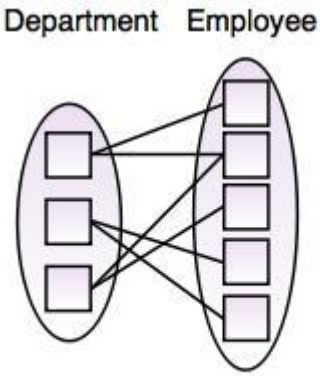


Fig. One to Many Mapping

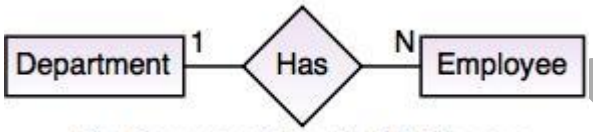


Fig. Representation in ER Diagram

3. Many - to - One Relationship

- In Many - to - One Relationship, many entities can be related with only one other entity.

For example: No. of Employee works for Department.

- Multiple rows in Employee table is related with only one row in Department table.

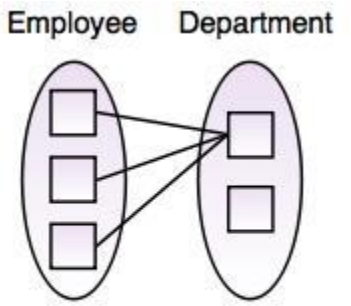


Fig. Many to One Mapping

Fig. Representation in ER Diagram

4. Many - to - Many Relationship

- In Many - to - Many Relationship, many entities are related with the multiple other entities.
- This relationship is a type of cardinality which refers the relation between two entities.

For example: Various Books in a Library are issued by many Students.

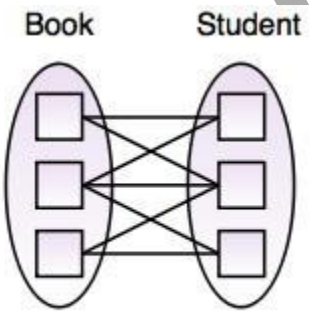


Fig. Many to Many Mapping

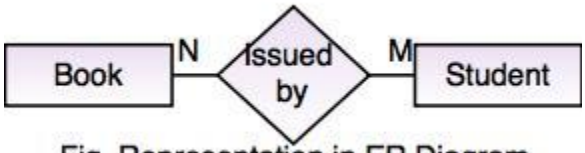


Fig. Representation in ER Diagram

Participation Constraint

Participation Constraint is applied on the entity participating in the relationship set.

Total Participation:

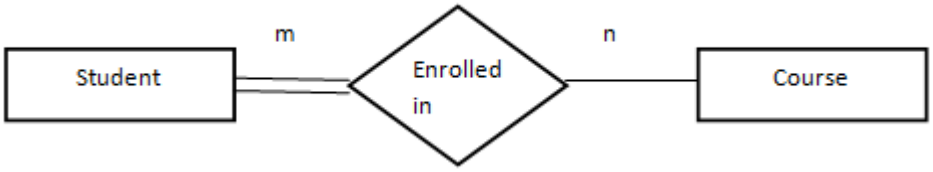
- Each entity in the entity set **must participate** in the relationship. If each student must enroll in a course.
- In Total Participation, every entity in the set is involved in some association of the relationship.
- It is indicated by a double line (═══) between entity and relationship.

For example: Every Department must have a Manager.

Fig. Total Participation

Partial Participation:

- The entity in the entity set **may or may NOT participat**e in the relationship.
- If some courses are not enrolled by any of the student, the participation of course will be partial.
- The diagram depicts the 'Enrolled in' relationship set with Student Entity set having total participation and Course Entity set having partial participation.



- In Partial Participation, not all entities in the set are involved in association of the relationship.
- It is indicated by a single line (———) between entity and relationship.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

Fig. Partial Participation

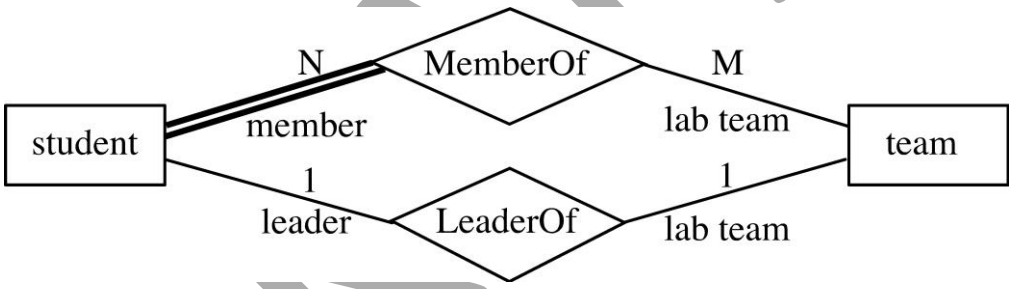
Example : Participation Constraints

Every student must be a member of a team, or, in other words, a student entity is of interest only if it participates in a *MemberOf* relationship. Thus, we can include in an ER diagram a **participation constraint** in which participation

of *student* in *MemberOf* is **total**. A double line indicates the total participation constraint in an ER model

of *student* in *LeaderOf*in is **partial**, because a student might be a team leader.

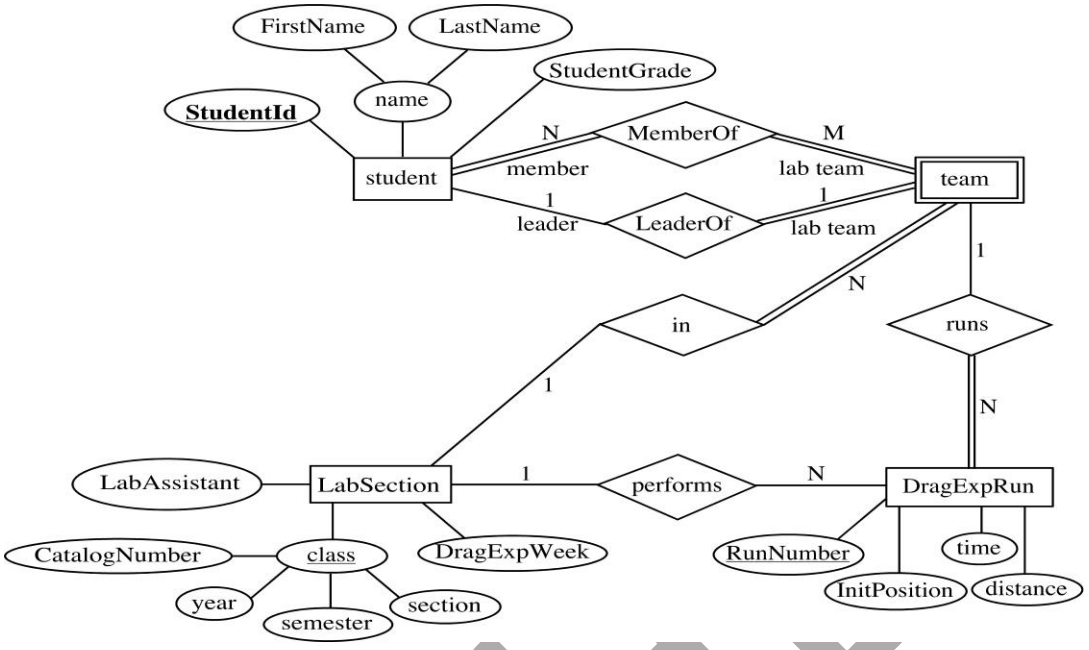
Figure 17. ER diagram notation for total participation constraint



Using the above components,

Figure 18. ER diagram notation for total participation constraint

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

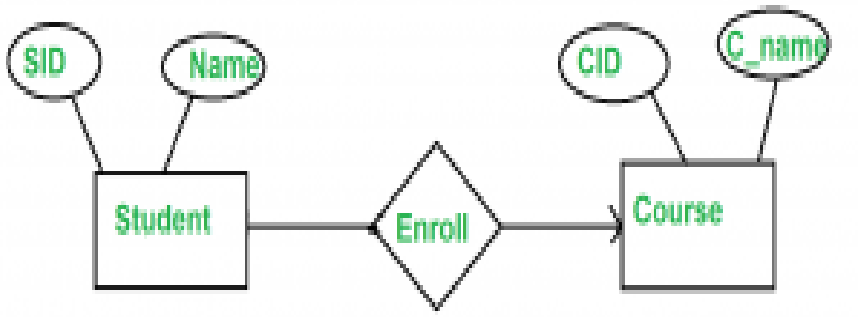


Entity Relationship (ER) Diagram – Relationship

Example

1) When there is One to Many cardinality in ER diagram.
For example, a student can be enrolled only in one course, but a course can be enrolled by many students

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I



Many To One

For Student(SID, Name), SID is the primary key. For Course (CID, C_name), CID is the primary key

Student
(SID Name)
---
1 A
2 B
3 C
4 D

Course
(CID C_name)
---
c1 Z
c2 Y
c3 X

Enroll
(SID CID)
---
1 C1
2 C1
3 c3
4 C2

Now the question is, what should be the primary key for Enroll SID or CID or combined. We can't have CID as primary key as you can see in enroll for the same CID we have multiples SID. (SID , CID) can distinguish table uniquely, but it is not minimum. So SID is the primary key for the relation enroll.

For above ER diagram, we considered three tables in database

Student

Enroll

Course

But we can combine Student and Enroll table renamed as Student_enroll.

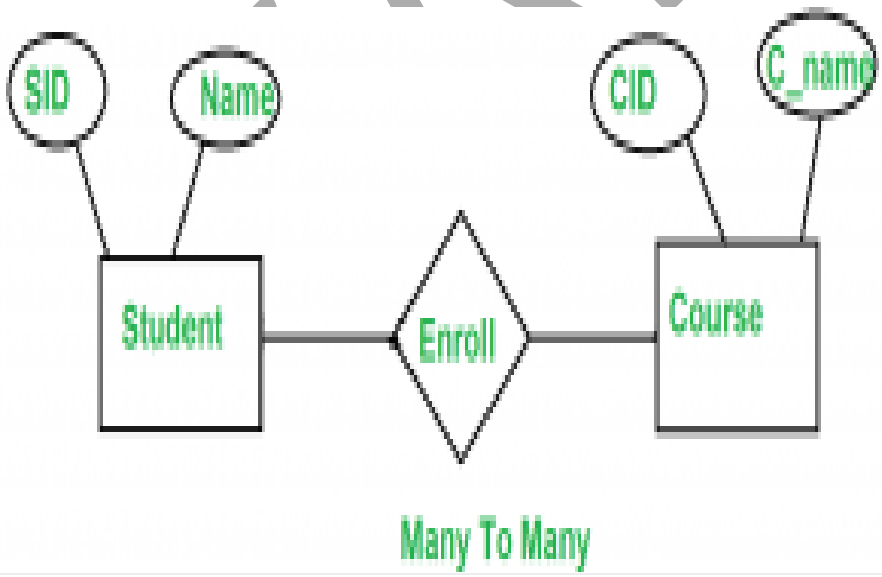
Student_Enroll
(SID Name CID)
---
1 A c1
2 B c1
3 C c3
4 D c2

Student and enroll tables are merged now .

So require minimum two DBMS tables for Student_enroll and Course.

Note: In One to Many relationship we can have minimum two tables.

2. When there is Many to Many cardinality in ER Diagram.
Let us consider above example with the change that now student can also enroll more than 1 course.



Many To Many

Student
(SID Name)

Course
(CID C_name)

1	A
2	B
3	C
4	D

c1	Z
c2	Y
c3	X

Enroll

SID	CID
1	C1
1	C2
2	C1
2	C2
3	c3
4	C2

Now, same question what is the primary key of Enroll relation, if we carefully analyse the Enroll primary key for Enroll
table is (SID , CID).

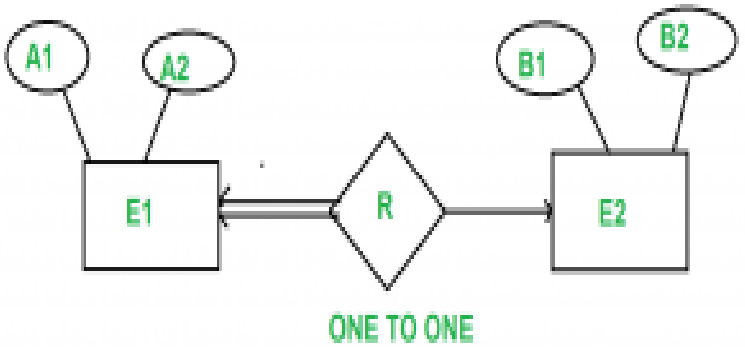
But in this case we can't merge Enroll table with any one of Student and Course. If we try to merge Enroll with any one of the Student and Course it will create redundant data.

Note: Minimum three tables are required in Many to Many relationship.

3. One to One Relationship
There are two possibilities

A) If we have One to One relationship and we have total participation at at-least one end.
For example, consider the below ER diagram.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I



ONE TO ONE

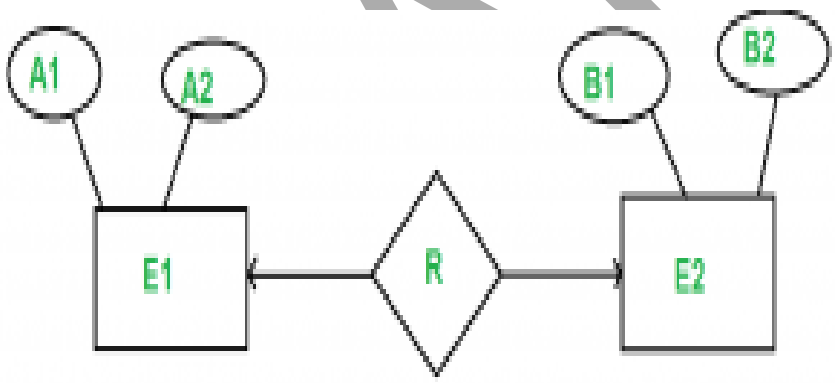
A1 and B1 are primary keys of E1 and E2 respectively.

In the above Diagram we have total participation at E1 end.

Only the primary key of E1, which is in total participation should be allowed as the primary key of the reduced table, since if the primary key of E2 is used, it might have null values for many of its entries, since its participation is only partial and may not have corresponding entries for all its values.

Note – Only one table required.

B) One to One relationship with no total participation.



ONE TO ONE

A1 and B1 are primary keys of E1 and E2 respectively.
Primary key of R can be A1 or B1, but we can't still combine all the three table into one. if we do, so some entries in combined table may have NULL entries. So idea of merging all three table into one is not good.

But we can merge R into E1 or E2. So minimum 2 tables are required.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT I

POSSIBLE QUESTIONS

2 MARK QUESTIONS

1. List the types of database models.
2. List the data types used in DBMS.
3. What is the difference between char and varchar2?
4. Mention the three levels of database schema.
5. Define primary key with example.

8 MARK QUESTIONS

1. Define database and explain various database models with neat diagram.
2. Draw and explain data base architecture.
3. What are the typical ERD symbols and draw a neat ERD for an employee table.
   b) Explain i) Entity types ii) Relationships
4. Discuss i) Data Independence ii) Constraints
5. What are the different data models present and explain briefly?

KARPAGAM ACADEMY OF HIGHER EDUCATION

Department of Computer Science

II B.Sc(CS) (BATCH 2016-2019) IV SEMESTER

DATABASE MANAGEMENT SYSTEMS (16CSU403)

PART-A OBJECTIVE TYPE/ MULTIPLE CHOICE QUESTIONS

UNIT I

S.NO	QUESTIONS	OPT 1	OPT 2	OPT 3	OPT 4	ANSWER
1	_____was adopted by the ANSI and ISO.	PSQL	SQL	R-SQL	Sequel	SQL
2	_____is a collection of high-level data description constructs that hide many low-level storage details	Data Model	ER Model	Network Model	none	Data Model
3	Database management System based on _____	Network model	Hierarchical model	Relational model	Object-based model	Relational model
4	A widely used Semantic model called _____	Network model	ER Model	Object-based model	Hierarchical model	ER Model
5	_____is a more abstract	ER model	Semantic data model	Conceptual data Model	Physical data model	Semantic data model
6	_____model is used to pictorially denote entities & relationships	Physical data model	ER model	network model	structure chart	ER model
7	A description Of data in terms of a data model is called_____	Schema	relation	record	entities	Schema
8	Field is otherwise known as _____	Column	Entity	Relationship	Relation	Column
9	Column is otherwise known as _____	Entity	Relationship	Relation	attribute	attribute

10	_____is a software designed to assist in maintaining and utilizing large collections of data.	Database	DBMS	Entities	attributes.	DBMS
11	_____model used Object store & versant.	Network Model	Hierarchical Model	Object Oriented Model	Record based Model	Object Oriented Model
12	____is used to define the external and conceptual model	DDL	DML	DCL	TCL	DDL
13	Conceptual model otherwise called as _____	Physical Schema	Internal Schema	Logical Schema	relations	Logical Schema
14	Physical model specifies _____details	Information	data	Storage	relationships	Storage
15	The _____ enviroment involves dumb terminals	mainframe	client/server	internet computing	LAN	mainframe
16	A host computer in internet computing eniroment is called _	server	data server	PC	web server	web server
17	____ is the primary unit of storage in a database	table	column	row	number	table
18	database design involves conversion of ____ to stuctured database model.	business process	business model	entity	relationships	business model
19	The architecture of a hierarchical database is based on ____the concept of relationships.	set structure	tree/node	parent/child	server/client	parent/child
20	The relationship between tables in the network model is called a	parent/child	set structure	client/server	tree/node	set structure
21	Set structures can represent a __ relationship between tables	one-to-one	one-to-many	many-to-many	many-to-one	one-to-many
22	SQL has been developed and used for ___ model	relational	Hierarchical	network	flat file	relational

23	A class is the equivalent of a _____ in a relational database	row	column	table	primary key	table
24	SQL3 is also referred to as	SQL97	SQL98	SQL99	SQL100	SQL99
25	_____ is the process of creating an interface for the end user through which the database can be accessed	dabase design	business model	interface design	Application design	Application design
26	The process of reducing data redundancy in a relational database is called	data security	data accuracy	data protection	normalization	normalization
27	Static, or _____ data is seldom or never modified once stored in the database.	dynamic	historic	information	transactional	historic
28	_____ or transactional data, is data that is frequently modified once stored in the database.	dynamic	historic	information	transactional	dynamic
29	BPR is	Business product re-engineering	Business product repair	Business process re-engineering	Business procedure re-engineering	Business process re-engineering
30	___ is the process of ensuring that data is consistent between related tables	primary key	database security	performance	Referential integrity	Referential integrity
31	Foreign keys are defined in ___ tables	parent	child	one	database	child
32	_____is an object in the real world	Entity	Attribute	Relationship	Property	Entity
33	in database model the data is stored in objects	hierarchical	network	relational	object_oriented	object_oriented
34	In relational model the data is stored in ___	table	files	objects	sets	table

35	Infromation about the conceptual,external and physical schemas is stored in _____	Directory	System Catalogs	IMS	Information System	System Catalogs
36	Conceptual schema otherwise called as _____	Physical Schema	Internal Schema	Logical Schema	relations	Logical Schema
37	Physical Schema specifies _____details	Information	data	Storage	relationships	Storage
38	_____is used to speed up data retrieval operations.	DML Operations	Select Operation	Indexes	select operation with where condition	Indexes
39	A Structure of database using the given data model is called a _____	Database	Relation	Schema	design	Schema
40	SQL was developed as an integral part of	A hierarchical database	A relational database	A OO database	A network database	A relational database
41	Which of the following is CORRECT about database management system's languages?	Data definition languages are used to specify the conceptual schema only.	Data manipulation languages are used to create the databases.	Data manipulation languages are used for retrieval, insertion, deletion and modification of data.	Data definition languages are only used to update data in the DBMS	Data manipulation languages are used for retrieval, insertion, deletion and modification of data.
42	An E-R modelling for given application leads to	conceptual data model	logical data model	external data model	internal data model	conceptual data model
43	A conceptual data model is converted using a Relational Data Base Management System to a	logical data model	external data model	internal data model	an entity-relation data model	logical data model
44	A subset of logical data model accessed by programmers is called a	conceptual data model	external data model	internal data model	an entity-relation data model	external data model

45	When a logical model is mapped into a physical storage such as a disk store the resultant data model is known as	conceptual data model	external data model	internal data model	disk data model	internal data model
46	By data integrity we mean	maintaining consistent data values	integrated data values	banning improper access to data	not leaking data values	maintaining consistent data values
47	Data integrity is ensured by	good data editing	propagating data changes to all data items	preventing unauthorized access	preventing data duplication	propagating data changes to all data items
48	By data security in DBMS we mean	preventing access to data	allowing access to data only to authorized users	preventing changing data	introducing integrity constraints	allowing access to data only to authorized users
49	By redundancy in a file based system we mean that	unnecessary data is stored	same data is duplicated in many files	data is unavailable	files have redundant data	same data is duplicated in many files
50	Data integrity in a file based system may be lost because	the same variable may have different values in different files	files are duplicated	unnecessary data is stored in files	redundant data is stored in files	the same variable may have different values in different files
51	Data availability is often difficult in file based system	as files are duplicated	as unnecessary data are stored in files	as one has to search different files and these files may be in different	redundant data are stored in files	as one has to search different files and these files may be in
52	An entity is	an inanimate object in an application	a collection of items in an application	a data structure	a distinct real world item in an application	a distinct real world item in an application

53	A relationship is	an item in an application	a meaningful dependency between entities	a collection of related entities	related data	a meaningful dependency between entities
54	Pick the relationship from the following:	a classroom	teacher	attends	cost per dozen	attends
55	Pick the meaningful relationship between entities	vendor supplies goods	vendor talks with customers	vendor complains to vendor	vendor asks prices	vendor supplies goods
56	The entity set is a	set of entities	collection of different entities	collection of related entities	collection of similar entities	collection of similar entities
57	Pick entity set from the following	all vendors supplying to an organization	vendors and organizations they supply	vendors and transporters	a vendor supplying to many organizations	all vendors supplying to an organization
58	The expansion of E-R diagram is	Entity-Relationship diagram	Entity-Relative diagram	Entity-Relation diagram	Entity-Rationalized diagram	Entity-Relationship diagram
59	In an E-R diagram entities are represented by	circles	rectangles	diamond shaped box	ellipse	rectangles
60	In an E-R diagram relationship is represented by	circles	rectangles	diamond shaped box	ellipse	diamond shaped box

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II

SYLLABUS

Relation data model: Relational model concepts, relational constraints, relational algebra

Relation Data Model

Relational data model is the primary data model, which is used widely around the world for data storage and processing. This model is simple and it has all the properties and capabilities required to process data with storage efficiency.

Concepts

Tables − In relational data model, relations are saved in the format of Tables. This format stores the relation among entities. A table has rows and columns, where rows represents records and columns represent the attributes.

Tuple − A single row of a table, which contains a single record for that relation is called a tuple.

Relation instance − A finite set of tuples in the relational database system represents relation instance. Relation instances do not have duplicate tuples.

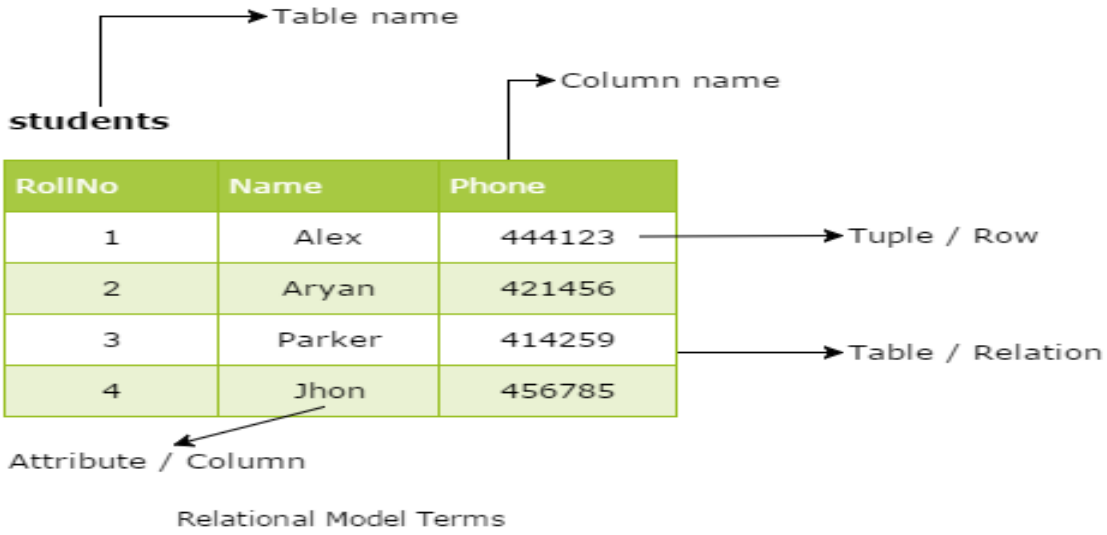
Relation schema − A relation schema describes the relation name (table name), attributes, and their names.

Relation key − Each row has one or more attributes, known as relation key, which can identify the row in the relation (table) uniquely.

Attribute domain − Every attribute has some pre-defined value scope, known as attribute domain.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II

Some Common Relational Model Terms



Relational Model Terms

- Relation: A relation is a table with columns and rows.
- Attribute: An attribute is a named column of a relation.
- Domain: A domain is the set of allowable values for one or more attributes.
- Tuple: A tuple is a row of a relation.

Relational Constraints

What are Database Constraints in DBMS ??

Database constraints are restrictions on the contents of the database or on database operations. It is a condition specified on a database schema that restricts the data to be inserted in an instance of the database.

Every relation has some conditions that must hold for it to be a valid relation. These conditions are called **Relational Integrity Constraints**.

Need of Constraints :

Constraints in the database provide a way to guarantee that :

- the values of individual columns are valid.
- in a table, rows have a valid primary key or unique key values.
- in a dependent table, rows have valid foreign key values that reference rows in a parent table.

Types of constraints in DBMS:

- **Domain Constraints**
- **Tuple Uniqueness Constraints**
- **Key Constraints**
- **Single Value Constraints**
- **Integrity Rule 1 (Entity Integrity Rule or Constraint)**
- **Integrity Rule 2 (Referential Integrity Rule or Constraint)**
- **General Constraints**

Domain Constraints –

Domain Constraints specifies that what set of values an attribute can take. Value of each attribute X must be an atomic value from the domain of X. The data type associated with domains include integer, character, string, date, time, currency etc. An attribute value must be available in the corresponding domain. Consider the example below –

SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	1st	18
8003	Somvir	4th	22
8004	Sourabh	6th	A

Not Allowed. Because Age is an Integer Attribute.

Tuple Uniqueness Constraints –

A relation is defined as a set of tuples. All tuples or all rows in a relation must be unique or distinct. Suppose if in a relation, tuple uniqueness constraint is applied, then all the rows of that table must be unique i.e. it does not contain the duplicate values. For example,

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II

SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	2nd	18
8003	Somvir	4th	22
8004	Sourabh	6th	19

Not Allowed. Because all rows must be unique.

Key Constraints –

Keys are attributes or sets of attributes that uniquely identify an entity within its entity set. An Entity set E can have multiple keys out of which one key will be designated as the primary key. Primary Key must have unique and not null values in the relational table. In an subclass hierarchy, only the root entity set has a key or primary key and that primary key must serve as the key for all entities in the hierarchy.

Types of keys in DBMS

1. **Primary Key** – A primary is a column or set of columns in a table that uniquely identifies tuples (rows) in that table.

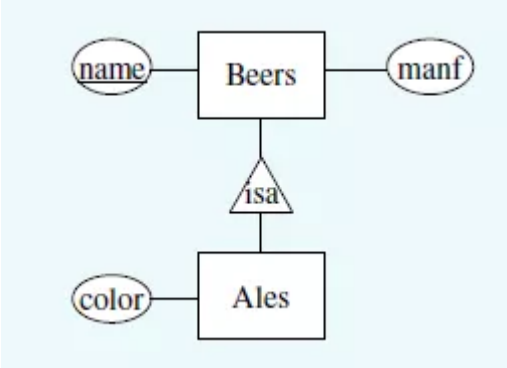
Example of Key Constraints in a simple relational table –

SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	1st	18
8003	Somvir	4th	22
8004	Sourabh	6th	45
8002	Tony	5th	23

Not allowed as Primary Key Values must be unique

Example of Key Constraints in an subclass hierarchy –

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II



2. **Super Key** – A super key is a set of one of more columns (attributes) to uniquely identify rows in a table. Often people get confused between super key and candidate key, so we will also discuss a little about candidate key here.

How candidate key is different from super key?

Answer is simple – Candidate keys are selected from the set of super keys, the only thing we take care while selecting candidate key is: It should not have any redundant attribute. That's the reason they are also termed as minimal super key.

Let's take an example to understand this: **Employee table**

Emp_SSN	Emp_Number	Emp_Name
123456789	226	Steve
999999321	227	Ajeet
888997212	228	Chaitanya
777778888	229	Robert

Super keys:

- {Emp_SSN}

- {Emp_Number}
- {Emp_SSN, Emp_Number}
- {Emp_SSN, Emp_Name}
- {Emp_SSN, Emp_Number, Emp_Name}
- {Emp_Number, Emp_Name}

All of the above sets are able to uniquely identify rows of the employee table.

3. **Candidate Key** – A super key with no redundant attribute is known as candidate key

A candidate key is a column, or set of columns, in a table that can uniquely identify any database record without referring to any other data. Each table may have one or more candidate keys, but one candidate key is unique, and it is called the primary key. This is usually the best among the candidate keys to use for identification.

When a key is composed of more than one column, it is known as a composite key.

A super key with no redundant attribute is known as candidate key. Candidate keys are selected from the set of super keys, the only thing we take care while selecting candidate key is: It should not have any redundant attributes. That's the reason they are also termed as minimal super key.

For example:

Emp_Id	Emp_Number	Emp_Name
E01	2264	Steve
E22	2278	Ajeet
E23	2288	Chaitanya
E45	2290	Robert

There are two candidate keys in above table:

{Emp_Id}

{Emp_Number}

Note: A primary key is being selected from the group of candidate keys. That means we can either have Emp_Id or Emp_Number as primary key.

4.**Alternate Key** – Out of all candidate keys, only one gets selected as primary key, remaining keys are known as alternate or secondary keys.

For example: Consider the below table

Emp_Id	Emp_Number	Emp_Name
E01	2264	Steve
E22	2278	Ajeet
E23	2288	Chaitanya
E45	2290	Robert

There are two candidate keys in above table:

{Emp_Id}

{Emp_Number}

Since we have selected Emp_Id as primary key, the remaining key Emp_Number would be called alternative or secondary key.

5**. Composite Key** – A key that consists of more than one attribute to uniquely identify rows (also known as records & tuples) in a table is called composite key.

Example: Table – Sales

cust_Id	order_Id	product_code	product_count
C01	O001	P007	23
C02	O123	P007	19
C02	O123	P230	82
C01	O001	P890	42

Key in above table: {cust_id, order_id}

This is a composite key as it consists of more than one attribute.

6. **Foreign Key** – Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables.

Single Value Constraints –

Single value constraints refers that each attribute of an entity set has a single value. If the value of an attribute is missing in a tuple, then we cal fill it with a "null" value. The null value for a attribute will specify that either the value is not known or the value is not applicable. Consider the below example-

SID	Name	Class (semester)	Age	Driving License Number
8001	Ankit	1st	19	DL-45698
8002	Srishti	2nd	18	DL-45871, DL-89740
8003	Somvir	4th	22	DL-95687
8004	Sourabh	6th	19	

Not allowed as a person does not have two driving licenses.

Allowed as a person may or may not have a driving license.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II

Integrity Rule 1 (Entity Integrity Rule or Constraint) –

The Integrity Rule 1 is also called Entity Integrity Rule or Constraint. This rule states that no attribute of primary key will contain a null value. If a relation have a null value in the primary key attribute, then uniqueness property of the primary key cannot be maintained. Consider the example below-

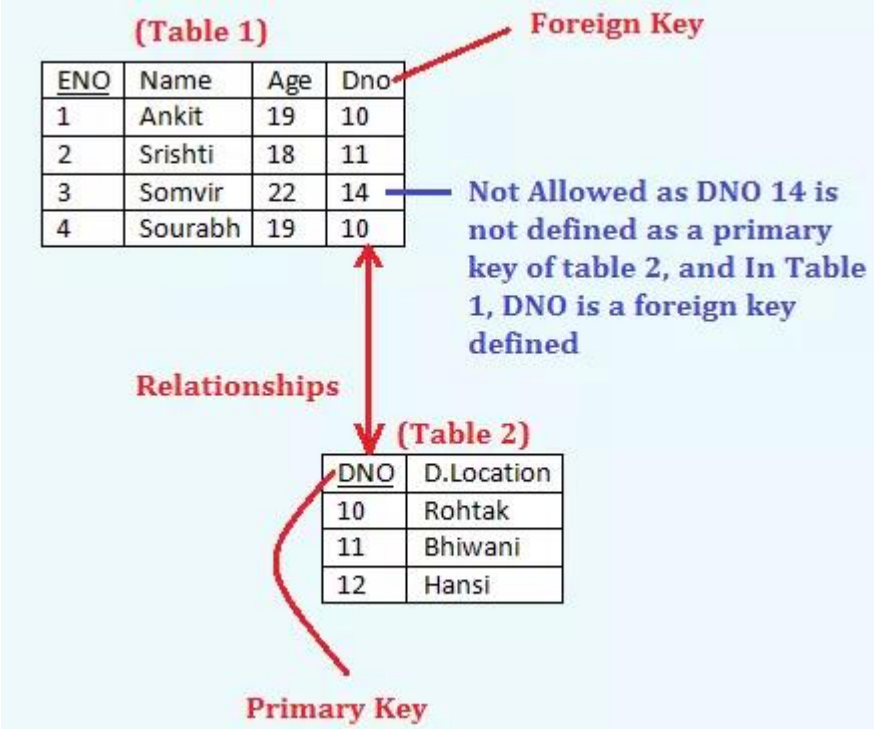
SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	2nd	18
8003	Somvir	4th	22
	Sourabh	6th	19

Not allowed as primary key cannot contain a NULL value

Integrity Rule 2 (Referential Integrity Rule or Constraint) –

The integrity Rule 2 is also called the Referential Integrity Constraints. This rule states that if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2. For example,

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II



Some more Features of Foreign Key –

Let the table in which the foreign key is defined is Foreign Table or details table i.e. Table 1 in above example and the table that defines the primary key and is referenced by the foreign key is master table or primary table i.e. Table 2 in above example. Then the following properties must be hold :

- Records cannot be **inserted** into a F**oreign table** if corresponding records in the master table do not exist.
- Records of the **master table or Primary Table** cannot be **deleted** or **updated** if corresponding records in the detail table actually exist.

General Constraints –

General constraints are the arbitrary constraints that should hold in the database. Domain Constraints, Key Constraints, Tuple Uniqueness Constraints, Single Value Constraints, Integrity Rule 1 (Entity Integrity) and 2 (Referential Integrity Constraints) are considered to be a fundamental part of the relational data model.

However, sometimes it is necessary to specify more general constraints like the CHECK Constraints or the Range Constraints etc.

Check constraints can ensure that only specific values are allowed in certain column. For example , if there is a need to allow only three values for the color like 'Bakers Chocolate', 'Glistening Grey' and 'Superior White', then we can apply the check constraint. All other values like 'GREEN' etc would yield an error. Range Constraints is implemented by BETWEEN and NOT BETWEEN. For example, if it is a requirement that student ages be within 16 to 35, then we can apply the range constraints for it.

The below example will explain Check Constraint and Range Constraint –

CarID	Name	Model	Color
C-12378	Wagon-R	2008	Bakers Chocolate
C-23478	Wagon-R	2008	Glistening Grey
C-45823	Wagon-R	2004	Superior White
C-45874	Wagon-R	2009	Green

Not Allowed, as CHECK Constraint is applied.

SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	2nd	18
8003	Somvir	4th	22
NULL	Sourabh	6th	65

Not allowed, as the range defined is in between 16 and 35

RELATIONAL ALGEBRA

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II

Relational Algebra is procedural query language, which takes Relation as input and generate relation as output. Relational algebra mainly provides theoretical foundation for relational databases and SQL.

- *Domain:* set of relations
- Based on set theory
- Contains extensions to manipulate tables
- Functional language
- Procedural, i.e., order to operations, algorithm implicit in the functional evaluation

Relational Algebra Operations

Below are fundamental operations that are "complete". That is, this set of operations alone can define any retrieval.

- Select
- Project
- Rename
- Union
- Set Difference
- Cartesian Product

Convenient, natural additions to the set of operations makes

- Set Intersection
- Natural Join
- Division
- Assignment

Projection

- Produce a subset of attributes from a relation
- Unselected columns are eliminated
- Duplicate rows are eliminated
- Result is a relation

Syntax: $\pi_{\text{attribute- list}}(\textit{relation})$

Example: The table **E** (for **EMPLOYEE**)

nr	name	salary
1	John	100
5	Sarah	300
7	Tom	100

SQL	Result	Relational algebra
select salary from E	salary 100 300	**PROJECT**$_{salary}$(E)
select nr, salary from E	nr salary 1 100 5 300 7 100	**PROJECT**$_{nr,\ salary}$(E)

Note that there are no duplicate rows in the result.

Selection

Choose a subset of tuples from a relation based on some criteria, results in another relation called a ***"result set"***

Notation uses lower case sigma:

Synatx: $\sigma_{condition}$(relation)

The same table **E** (for **EMPLOYEE**) as above.

SQL	Result	Relational algebra
select * from E where salary < 200	nr name salary 1 John 100 7 Tom 100	**SELECT**$_{salary < 200}$(E)

select * from E where salary < 200 and nr >= 7	nr | name | salary 7 | Tom | 100	**SELECT**$_{\text{salary < 200 and nr >= 7}}$(E)

Set Operators

One of the characteristics of RDBMS is that it should support all the transaction on the records in the table by means relational operations. That means it should have strong query language which supports relational algebra. There are three main relational algebras on sets – UNION, SET DIFFERENCE and SET INTERSECT. The same is implemented in database query language using set operators.

There are 4 main set operators used in the query language.

1. UNION

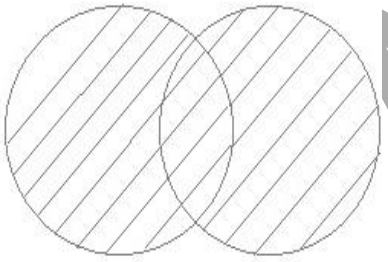
It combines the similar columns from two tables into one resultant table. All columns that are participating in the UNION operation should be Union Compatible. This operator combines the records from both the tables into one. If there are duplicate values as a result, then it eliminates the duplicate. The resulting records will be from both table and distinct.

EMP_TEST			
EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
104	Kathy	Holland	324343

Union

EMP_DESIGN			
EMP_ID	ENAME	EMP_ADDRESS	SSN
103	Rose	Freser Town	6744545
102	Marry	Novi	343613
105	Laurry	Rochester Hills	97676
104	Kathy	Holland	324343

UNION			
EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
102	Marry	Novi	343613
103	Rose	Freser Town	6744545
104	Kathy	Holland	324343
105	Laurry	Rochester Hills	97676



We can notice that Result will have same column names as first query. Duplicate record – 104 from EMP_TEST and EMP_DESIGN are showed only once in the result set. Records are sorted in the result.

UNION ALL

This operation is also similar to UNION, but it does not eliminate the duplicate records. It shows all the records from both the tables. All other features are same as UNION. We can have conditions in the SELECT query. It need not be a simple SELECT query.

Look at the same example below with UNION ALL operation.

EMP_TEST

EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
104	Kathy	Holland	324343

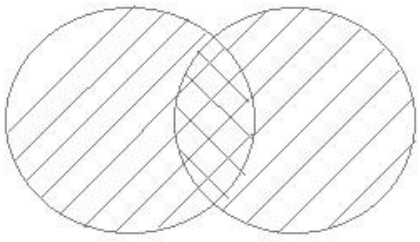
Union ALL

EMP_DESIGN

EMP_ID	ENAME	EMP_ADDRESS	SSN
103	Rose	Freser Town	6744545
102	Marry	Novi	343613
105	Laurry	Rochester Hills	97676
104	Kathy	Holland	324343

UNION

EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
102	Marry	Novi	343613
103	Rose	Freser Town	6744545
104	Kathy	Holland	324343
104	Kathy	Holland	324343
105	Laurry	Rochester Hills	97676

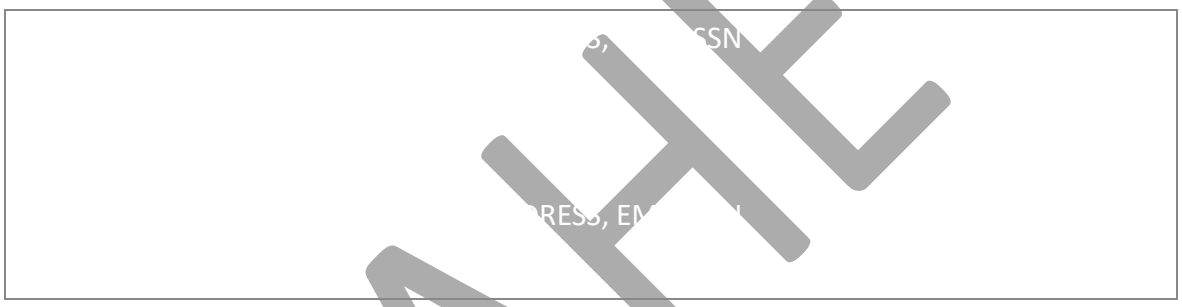


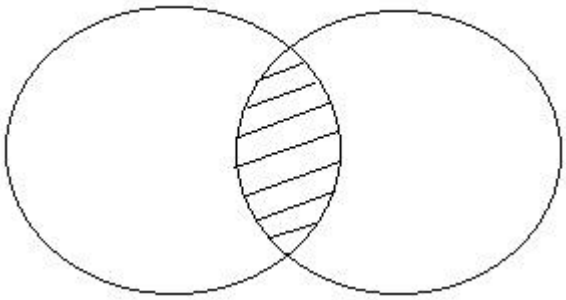
CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II

2. INTERSECT

This operator is used to pick the records from both the tables which are common to them. In other words it picks only the duplicate records from the tables. Even though it selects duplicate records from the table, each duplicate record will be displayed only once in the result set. It should have UNION Compatible columns to run the query with this operator.

Same example above when used with INTERSECT operator, gives below result.





CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II

EMP_TEST			
EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
104	Kathy	Holland	324343

INTERSECT

EMP_DESIGN			
EMP_ID	ENAME	EMP_ADDRESS	SSN
103	Rose	Freser Town	6744545
102	Marry	Novi	343613
105	Laurry	Rochester Hills	97676
104	Kathy	Holland	324343

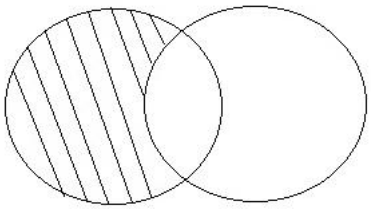
UNION			
EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
104	Kathy	Holland	324343

3. MINUS

This operator is used to display the records that are present only in the first table or query, and doesn't present in second table / query. It basically subtracts the first query results from the second.

Let us see the same example with MINUS operator.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II



EMP_TEST			
EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
104	Kathy	Holland	324343

MINUS

EMP_DESIGN			
EMP_ID	ENAME	EMP_ADDRESS	SSN
103	Rose	Freser Town	6744545
102	Marry	Novi	343613
105	Laurry	Rochester Hills	97676
104	Kathy	Holland	324343

UNION			
EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434

We can notice in the above result that only the records that do not exists in EMP_DESIGN are displayed in the result. The record which appears in both the tables is eliminated. Similarly, the records that appear in second query but not in the first query are also eliminated.

4. Division Operator (÷):

Table 1

STUDENT_SPORTS

ROLL_NO	SPORTS
1	Badminton
2	Cricket

2	Badminton
4	Badminton

Table 2

ALL_SPORTS

SPORTS
Badminton
Cricket

Division Operator (÷): Division operator A÷B can be applied if and only if:

- Attributes of B is proper subset of Attributes of A.
- The relation returned by division operator will have attributes = (All attributes of A – All Attributes of B)
- The relation returned by division operator will return those tuples from relation A which are associated to every B's tuple.

STUDENT_SPORTS÷ ALL_SPORTS

- The operation is valid as attributes in ALL_SPORTS is a proper subset of attributes in STUDENT_SPORTS.
- The attributes in resulting relation will have attributes {ROLL_NO,SPORTS}-{SPORTS}=ROLL_NO
- The tuples in resulting relation will have those ROLL_NO which are associated with all B's tuple {Badminton, Cricket}. ROLL_NO 1 and 4 are associated to Badminton only. ROLL_NO 2 is associated to all tuples of B. So the resulting relation will be:

ROLL_NO
2

Join in SQL

SQL Join is used to fetch data from two or more tables, which is joined to appear as single set of data. SQL Join is used for combining column from two or more tables by using values common to both tables.**Join** Keyword is used in SQL queries for joining two or more tables. Minimum required condition for joining table, is **(n-1)** where **n**, is number of tables. A table can also join to itself known as, **Self Join**.

Types of Join

The following are the types of JOIN that we can use in SQL.

- Inner
- Outer
- Left
- Right

Cross JOIN or Cartesian Product

This type of JOIN returns the cartesian product of rows from the tables in Join. It will return a table which consists of records which combines each row from the first table with each row of the second table.

Cross JOIN Syntax is,

```
SELECT column-name-list
from table-name1
CROSS JOIN
table-name2;
```

Example of Cross JOIN

The **class** table,

ID	NAME
1	abhi
2	adam
4	alex

The **class_info** table,

ID	Address
1	DELHI
2	MUMBAI
3	CHENNAI

Cross JOIN query will be,

```
SELECT *
 from class,
 cross JOIN class_info;
```

The result table will look like,

ID	NAME	ID	Address
1	abhi	1	DELHI
2	adam	1	DELHI
4	alex	1	DELHI
1	abhi	2	MUMBAI
2	adam	2	MUMBAI

4	alex	2	MUMBAI
1	abhi	3	CHENNAI
2	adam	3	CHENNAI
4	alex	3	CHENNAI

INNER Join or EQUI Join

This is a simple JOIN in which the result is based on matched data as per the equality condition specified in the query.

Inner Join Syntax is,

```
SELECT column-name-list
from table-name1
INNER JOIN
table-name2
WHERE table-name1.column-name = table-name2.column-name;
```

Example of Inner JOIN

The **class** table,

ID	NAME
1	abhi
2	adam
3	alex
4	anu

The **class_info** table,

ID	Address
1	DELHI
2	MUMBAI
3	CHENNAI

Inner JOIN query will be,

```
SELECT * from class, class_info where class.id = class_info.id;
```

The result table will look like,

ID	NAME	ID	Address
1	abhi	1	DELHI
2	adam	2	MUMBAI
3	alex	3	CHENNAI

Natural JOIN

Natural Join is a type of Inner join which is based on column having same name and same datatype present in both the tables to be joined.

Natural Join Syntax is,

```
SELECT *
from table-name1
NATURAL JOIN
table-name2;
```

Example of Natural JOIN

The **class** table,

ID	NAME
1	abhi
2	adam
3	alex
4	anu

The **class_info** table,

ID	Address
1	DELHI
2	MUMBAI
3	CHENNAI

Natural join query will be,

SELECT * from class NATURAL JOIN class_info;

The result table will look like,

ID	NAME	Address
1	abhi	DELHI
2	adam	MUMBAI
3	alex	CHENNAI

In the above example, both the tables being joined have ID column(same name and same datatype), hence the records for which value of ID matches in both the tables will be the result of Natural Join of these two tables.

Outer JOIN

Outer Join is based on both matched and unmatched data. Outer Joins subdivide further into,

- Left Outer Join
- Right Outer Join
- Full Outer Join

Left Outer Join

The left outer join returns a result table with the **matched data** of two tables then remaining rows of the**left** table and null for the **right** table's column.

Left Outer Join syntax is,

```
SELECT column-name-list
from table-name1
LEFT OUTER JOIN
table-name2
on table-name1.column-name = table-name2.column-name;
```

Left outer Join Syntax for **Oracle** is,

```
select column-name-list
from table-name1,
table-name2
on table-name1.column-name = table-name2.column-name(+);
```

Example of Left Outer Join

The **class** table,

ID	NAME

1	abhi
2	adam
3	alex
4	anu
5	ashish

The **class_info** table,

ID	Address
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

Left Outer Join query will be,

```
SELECT * FROM class LEFT OUTER JOIN class_info ON (class.id=class_info.id);
```

The result table will look like,

ID	NAME	ID	Address
1	abhi	1	DELHI
2	adam	2	MUMBAI
3	alex	3	CHENNAI

4	anu	null	null
5	ashish	null	null

Right Outer Join

The right outer join returns a result table with the **matched data** of two tables then remaining rows of the**right table** and null for the **left** table's columns.

Right Outer Join Syntax is,

```
select column-name-list
from table-name1
RIGHT OUTER JOIN
table-name2
on table-name1.column-name = table-name2.column-name;
```

Right outer Join Syntax for **Oracle** is,

```
select column-name-list
from table-name1,
table-name2
on table-name1.column-name(+) = table-name2.column-name;
```

Example of Right Outer Join

The **class** table,

ID	NAME
1	abhi
2	adam
3	alex

4	anu
5	ashish

The **class_info** table,

ID	Address
1	DELHI
2	MUMBAI
3	CHENNAI
7	NOIDA
8	PANIPAT

Right Outer Join query will be,

```
SELECT * FROM class RIGHT OUTER JOIN class_info on (class.id=class_info.id);
```

The result table will look like,

ID	NAME	ID	Address
1	abhi	1	DELHI
2	adam	2	MUMBAI
3	alex	3	CHENNAI
null	null	7	NOIDA
null	null	8	PANIPAT

Full Outer Join

The full outer join returns a result table with the **matched data** of two table then remaining rows of both**left** table and then the **right** table.

Full Outer Join Syntax is,

```
select column-name-list
from table-name1
FULL OUTER JOIN
table-name2
on table-name1.column-name = table-name2.column-name;
```

Example of Full outer join is,

The **class** table,

ID	NAME
1	abhi
2	adam
3	alex
4	anu
5	ashish

The **class_info** table,

ID	Address
1	DELHI
2	MUMBAI

3	CHENNAI
7	NOIDA
8	PANIPAT

Full Outer Join query will be like,

```
SELECT * FROM class FULL OUTER JOIN class_info on (class.id=class_info.id);
```

The result table will look like,

ID	NAME	ID	Address
1	abhi	1	DELHI
2	adam	2	MUMBAI
3	alex	3	CHENNAI
4	anu	null	null
5	ashish	null	null
null	null	7	NOIDA
null	null	8	PANIPAT

Aggregate Functions in SQL

In database management an aggregate function is a function where the values of multiple rows are grouped together as input on certain criteria to form a single value of more significant meaning.

Various Aggregate Functions

1) Count()

2) Sum()

3) Avg()

4) Min()

5) Max()

Now let us understand each Aggregate function with a example: Table : E

Id	Name	Salary
1	A	80
2	B	40
3	C	60
4	D	70
5	E	60
6	F	Null

1. COUNT()

Syntax:

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

Example

```
SELECT COUNT(ID) FROM E;
```

Count(*): Returns total number of records .i.e 6.

Count(salary): Return number of Non Null values over the column salary. i.e 5.

Count(Distinct Salary): Return number of distinct Non Null values over the column salary .i.e 4

2. SUM()

Syntax

SELECT SUM(*column_name*)
FROM *table_name*
WHERE *condition*;

Example

SELECT sum(ID) FROM E

sum(salary): Sum all Non Null values of Column salary i.e., 310

sum(Distinct salary): Sum of all distinct Non-Null values i.e., 250.

3. AVG()

Syntax

SELECT AVG(*column_name*)
FROM *table_name*
WHERE *condition*;

Example

SELECT AVG(salary) FROM E;

Avg(salary) = Sum(salary) / count(salary) = 310/5

Avg(Distinct salary) = sum(Distinct salary) / Count(Distinct Salary) = 250/4

4. MIN()

Syntax

SELECT MIN(*column_name*)
FROM *table_name*
WHERE *condition*;

Example

SELECT MIN(salary) FROM E;

Min(salary): Minimum value in the salary column except NULL i.e., 40.

5**. MAX()**

Syntax

SELECT MAX(*column_name*)
FROM *table_name*
WHERE *condition*;

Example

SELECT MAX(salary) FROM E;

Max(salary): Maximum value in the salary i.e., 80.

POSSIBLE QUESTIONS

2 MARK QUESTIONS

1. Draw an ER diagram for 1:M relationship.
2. Draw an ER diagram for M:M relationship
3. Write the syntax for DELETE FROM WHERE command in SQL.
4. What is an Entity type?
5. Define cardinalities.
6. Draw an ER model for M:M relationship.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT II

6 MARK QUESTIONS

1. IIustrate with syntax and example of relational constraints.
2. Discuss relational algebra with proper example.
3. Explain the following: i) Key constraints ii) Integrity constraints.
4. Discuss in detail the operators SELECT, PROJECT and UNION with suitable examples.
4. Describe SET operators and Joins in DBMS with example
5. Explain in detail about various key constraints used in database system.

KARPAGAM ACADEMY OF HIGHER EDUCATION

DEPARTMENT OF COMPUTER SCIENCE
II BSc CS BATCH(2016 - 2019)
PART-A OBJECTIVE TYPE/ MULTIPLE CHOICE QUESTIONS
UNIT-2

SNO	Questions	opt1	opt2	opt3	opt4	Answer
1	The rows of a relation	must be in specified order	may be in any order	in ascending order of key	in descending order of key	may be in any order
2	The columns of a relation	must be in specified order	may be in any order	with key field in first column	with largest width column last	may be in any order
3	A_____is a dependency of one non primary key attribute on another non primary key attribute.	Functional dependency	Transitive dependency	Partial dependency	Non primary key dependency	Partial dependency
4	functional depency of the form X →Y where Y is a subset of X are called _____	Functional dependency	Transitive dependency	Partial dependency	Trivial dependency	Trivial dependency
5	Group function is otherwise known as_____	Collection	Aggregate	function	Count	Aggregate
6	Pattern matching has done through _____operator.	Comparison	arithmetic	Logical	Aggregate	Aggregate
7	Which keyword is used to check if an element is in a given set?	not	in	not exist	except	in
8	Any two tables that are Union-Compatible that is, have the same number of _____	Columns	rows	Columns and rows	null values	Columns
9	_____keyword is used to eliminates the duplicates	Union	Union all	Intersect all	Except all	Union
10	_____keyword is used to retain the duplicates	Union	Union all	Intersect	Except	Union all
11	_____technique is used to reduce the redundancy	Closure set	Decomposition	Normalization	Null Values	Normalization
12	if and only if the right-hand side is not a subset of the left-hand side, then functional dependency is said to be as _____	Non-trivial	Trivial	Transitive	Augmentation	Non-trivial
13	The Closure of F denoted as_____	Fc	F—	FXX	F+	F+
14	All decomposition are used to eliminate_____	duplicates	null values	empty values	not null values	duplicates
15	_____is a kind of IC, that generalizes the concept of a key.	Decomposition	Functional dependency	cursors	Triggers	Functional dependency
16	_____Consists of replacing the relation schema by two relation schemas that each contain a subsets of the attributes.	Decomposition	Functional dependency	cursors	Triggers	Decomposition
17	X is a proper subset of some key K. Such a dependency is sometimes called_____	dependency	Partial dependency	transitive dependency	Decompostion	Partial dependency
18	X is not a proper subset of some key K. Such a dependency is sometimes called_____	dependency	Partial dependency	transitive dependency	Decompostion	transitive dependency
19	Indicated by using arrow from entities to relationships in the ER diagram.	Arrow	Thick line	Dotted line	Shaded line	Arrow
20	Aggragation is indicated by _____in ER diagram	Solid line	Thick line	Thin line	Dotted line	Dotted line
21	ISA is indicated by _____symbol	Rectangle	Ellipse	Triangle	Diamond	Triangle
22	_____is a set of associated values	Entity	Attribute	Relationships	Domain	Domain
23	_____consists of a relation schema and a relation instance.	relation	table	domain	entity	relation
24	An instance of a relation is a set of _____	tuple	domain	attribute	relationships	tuple
25	Each tuple is a _____	Column	row	table	instance	row
26	_____is an object in the real world	Entity	Attribute	Relationship	Property	Entity
27	Collection Of Similar entities are_____	Attributes	Entity	Entity Set	Relationship	Entity Set
28	An Entity is described using a set Of _____	Entity	Entity Set	Attributes	Relationship	Attributes
29	_____is used to uniquely identify an entity in the set.	Key	Lock	Attributes	Entity	Key
30	DBMS is a collection of …….. that enables user to create and maintain a database.	Keys	Translators	Program	Language Activity	Program
31	In a relational schema, each tuple is divided into fields called	Relations	Domains	Queries	All of the above	Domains
32	In an ER model, ……. is described in the database by storing its data.	Entity	Attribute	Relationship	Notation	Entity
33	DFD stands for	Data Flow Document	Data File Diagram	Data Flow Diagram	Non of the above	Data Flow Diagram
34	A top-to-bottom relationship among the items in a database is established by a	Hierarchical schema	Network schema	Relational Schema	All of the above	Hierarchical schema
35	…… table store information about database or about the system.	SQL	Nested	System	None of these	System
36	……..defines the structure of a relation which consists of a fixed set of attribute-domain pairs.	Instance	Schema	Program	Super Key	Schema
37	…… clause is an additional filter that is applied to the result.	Select	Group-by	Having	Order by	Having

38	A logical schema	is the entire database	is a standard way of organizing information into accessible parts.	Describes how data is actually stored on disk.	All of the above	is a standard way of organizing information into accessible parts.
39	…… is a full form of SQL.	Standard query language	Sequential query language	Structured query language	Server side query language	Structured query language
40	A relational database developer refers to a record as	a criteria	a relation	a tuple	an attribute	a tuple
41	……. keyword is used to find the number of values in a column.	TOTAL	COUNT	ADD	SUM	COUNT
42	An advantage of the database management approach is	data is dependent on programs	data redundancy increases	data is integrated and can be accessed by multiple programs	none of the above	data is integrated and can be accessed by multiple programs
43	The collection of information stored in a database at a particular moment is called as ……	schema	instance of the database	data domain	independence	instance of the database
44	A …… is used to define overall design of the database	schema	application program	data definition language	code	schema
45	Key to represent relationship between tables is called	primary key	secondary key	foreign key	none of the above	foreign key
46	Grant and revoke are ……. statements	DDL	TCL	DCL	DML	DCL
47	DBMS helps achieve	Data independence	Centralized control of data	Neither A nor B	Both A and B	Both A and B
48	……. command can be used to modify a column in a table	alter	update	set	create	alter
49						
50	The candidate key is that you choose to identify each row uniquely is called ……..	Alternate Key	Primary Key	Foreign Key	None of the above	Primary Key
51	…….. is used to determine whether of a table contains duplicate rows.	Unique predicate	Like Predicate	Null predicate	In predicate	Unique predicate
52	To eliminate duplicate rows …… is used	NODUPLICATE	ELIMINATE	DISTINCT	None of these	DISTINCT
53	DCL stands for	Data Control Language	Data Console Language	Data Console Level	Data Control Level	Data Control Language
54	…… is the process of organizing data into related tables.	Normalization	Generalization	Specialization	None of the above	Normalization
55	A ……. Does not have a distinguishing attribute if its own and mostly are dependent entities, which are part of some another entity.	Weak entity	Strong entity	Non attributes entity	Dependent entity	Weak entity
56	…….. is the complex search criteria in the where clause.	Sub string	Drop Table	Predict	Predicate	Predicate
57	…… is preferred method for enforcing data integrity	Constraints	Stored Procedure	Triggers	Cursors	Constraints
58	The number of tuples in a relation is called its ……. While the number of attributes in a relation is called it's ……..	Degree, Cardinality	Cardinality, Degree	Rows, Columns	Columns, Rows	Cardinality, Degree
59	The language that requires a user to specify the data to be retrieved without specifying exactly how to get it is	Procedural DML	Non-Procedural DML	Procedural DDL	Non-Procedural DDL	Procedural DML
60	Which two files are used during operation of the DBMS?	Query languages and utilities	DML and query language	Data dictionary and transaction log	Data dictionary and query language	Data dictionary and transaction log

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

SYLLABUS

Relation data model: SQL queries **Database design:** Mapping ER/EER model to relational database, functional dependencies, Lossless decomposition.

What is SQL

- SQL stands for **Structured Query Language**.
- It is designed for managing data in a relational database management system (RDBMS).
- It is pronounced as S-Q-L or sometime **See-Qwell**.
- SQL is a database language, it is used for database creation, deletion, fetching rows and modifying rows etc.
- SQL is based on relational algebra and tuple relational calculus.

All DBMS like MySQL, Oracle, MS Access, Sybase, Informix, Postgres and SQL Server use SQL as standard database language.

Why SQL is required

SQL is required:

- To create new databases, tables and views
- To insert records in a database
- To update records in a database
- To delete records from a database
- To retrieve data from a database

What SQL does

- With SQL, we can query our database in a numbers of ways, using English-like statements.
- With SQL, user can access data from relational database management system.
- It allows user to describe the data.

- It allows user to define the data in database and manipulate it when needed.
- It allows user to create and drop database and table.
- It allows user to create view, stored procedure, function in a database.
- It allows user to set permission on tables, procedure and view.

SQL Table

Table is a collection of data, organized in terms of rows and columns. In DBMS term, table is known as relation and row as tuple.

Note: A table has a specified number of columns, but can have any number of rows.

Table is the simple form of data storage. A table is also considered as a convenient representation of relations.

Let's see an example of an employee table:

Employee		
EMP_NAME	**ADDRESS**	**SALARY**
Ankit	Lucknow	15000
Raman	Allahabad	18000
Mike	New York	20000

In the above table, "Employee" is the table name, "EMP_NAME", "ADDRESS" and "SALARY" are the column names. The combination of data of multiple columns forms a row e.g. "Ankit", "Lucknow" and 15000 are the data of one row.

SQL TABLE Variable

The **SQL Table variable** is used to create, modify, rename, copy and delete tables. Table variable was introduced by Microsoft.

It was introduced with SQL server 2000 to be an alternative of temporary tables.

It is a variable where we temporary store records and results. This is same like temp table but in the case of temp table we need to explicitly drop it.

Table variables are used to store a set of records. So declaration syntax generally looks like CREATE TABLE syntax.

```
create table "tablename"
("column1" "data type",
"column2" "data type",
...
"columnN" "data type");
```

When a transaction rolled back the data associated with table variable is not rolled back.

A table variable generally uses lesser resources than a temporary variable.

Table variable cannot be used as an input or an output parameter.

SQL CREATE TABLE

SQL CREATE TABLE statement is used to create table in a database.

If you want to create a table, you should name the table and define its column and each column's data type.

Let's see the simple syntax to create the table.

```
create table "tablename"
("column1" "data type",
"column2" "data type",
"column3" "data type",
...
"columnN" "data type");
```

The data type of the columns may vary from one database to another. For example, NUMBER is supported in Oracle database for integer value whereas INT is supported in MySQL.

Let us take an example to create a STUDENTS table with ID as primary key and NOT NULL are the constraint showing that these fields cannot be NULL while creating records in the table.

```
SQL> CREATE TABLE STUDENTS (
ID INT NOT NULL,
NAME VARCHAR (20) NOT NULL,
AGE INT NOT NULL,
ADDRESS CHAR (25),
PRIMARY KEY (ID)
);
```

You can verify it, if you have created the table successfully by looking at the message displayed by the SQL Server, else you can use DESC command as follows:

```
SQL> DESC STUDENTS;
```

FIELD	TYPE	NULL	KEY	DEFAULT	EXTRA
ID	Int(11)	NO	PRI		
NAME	Varchar(20)	NO			
AGE	Int(11)	NO			
ADDRESS	Varchar(25)	YES		NULL	

4 rows in set (0.00 sec)

Now you have the STUDENTS table available in your database and you can use to store required information related to students.

SQL INSERT STATEMENT

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

SQL INSERT statement is a SQL query. It is used to insert a single or a multiple records in a table.

There are two ways to insert data in a table:

1. By SQL insert into statement
    1. By specifying column names
    2. Without specifying column names
2. By SQL insert into select statement

1) Inserting data directly into a table

You can insert a row in the table by using SQL INSERT INTO command. But there are 2 ways to do this.

You can specify or ignore the column names while using INSERT INTO statement.

To insert partial column values, you must have to specify the column names. But if you want to insert all the column values, you can specify or ignore the column names.

If you specify the column names, syntax of the insert into statement will be as follows:

```
INSERT INTO TABLE_NAME
[(col1, col2, col3,.... col N)]
VALUES (value1, value2, value 3, .... Value N);
```

Here col1, col2, col3, ... colN are the columns of the table in which you want to insert data.

But, If you ignore the column names, syntax of the insert into statement will be as follows:

```
INSERT INTO TABLE_NAME
VALUES (value1, value2, value 3, .... Value N);
```

2) Inserting data through SELECT Statement

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

SQL INSERT INTO SELECT Syntax

INSERT INTO table_name
[(column1, column2, ... **column**)]
SELECT column1, column2, ... **Column** N
FROM table_name [**WHERE** condition];

Note: when you add a new row, you should make sure that data type of the value and the column should be matched.

If any integrity constraints are defined for the table, you must follow them.

SQL INSERT INTO VALUE

There are two ways to insert values in a table.

In the first method there is no need to specify the column name where the data will be inserted, you need only their values.

INSERT INTO table_name
VALUES (value1, value2, value3....);

The second method specifies both the column name and values which you want to insert.

INSERT INTO table_name (column1, column2, column3....)
VALUES (value1, value2, value3.....);

Let's take an example of table which has five records within it.

INSERT INTO STUDENTS (ROLL_NO, **NAME**, AGE, CITY)
VALUES (1, ABHIRAM, 22, ALLAHABAD);
INSERT INTO STUDENTS (ROLL_NO, **NAME**, AGE, CITY)
VALUES (2, ALKA, 20, GHAZIABAD);
INSERT INTO STUDENTS (ROLL_NO, **NAME**, AGE, CITY)
VALUES (3, DISHA, 21, VARANASI);
INSERT INTO STUDENTS (ROLL_NO, **NAME**, AGE, CITY)

```
VALUES (4, ESHA, 21, DELHI);
INSERT INTO STUDENTS (ROLL_NO, NAME, AGE, CITY)
VALUES (5, MANMEET, 23, JALANDHAR);
```

It will show the following table as the final result.

ROLL_NO	NAME	AGE	CITY
1	ABHIRAM	22	ALLAHABAD
2	ALKA	20	GHAZIABAD
3	DISHA	21	VARANASI
4	ESHA	21	DELHI
5	MANMEET	23	JALANDHAR

You can create a record in CUSTOMERS table by using this syntax also.

```
INSERT INTO CUSTOMERS
VALUES (6, PRATIK, 24, KANPUR);
```

The following table will be as follow:

ROLL_NO	NAME	AGE	CITY
1	ABHIRAM	22	ALLAHABAD
2	ALKA	20	GHAZIABAD
3	DISHA	21	VARANASI
4	ESHA	21	DELHI
5	MANMEET	23	JALANDHAR
6	PRATIK	24	KANPUR

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

SQL SELECT

The most commonly used SQL command is **SELECT statement**. It is used to query the database and retrieve selected data that follow the conditions we want.

In simple words, we can say that the select statement used to query or retrieve data from a table in the database.

Let's see the syntax of select statement.

SELECT expressions
FROM tables
WHERE conditions;

Here expression is the column that we want to retrieve.

Tables indicate the tables, we want to retrieve records from.

Optional clauses in SELECT statement

There are some optional clauses in SELECT statement:

[WHERE Clause] : It specifies which rows to retrieve.

[GROUP BY Clause] : Groups rows that share a property so that the aggregate function can be applied to each group.

[HAVING Clause] : It selects among the groups defined by the GROUP BY clause.

[ORDER BY Clause] : It specifies an order in which to return the rows.

For example, let a database table: student_details;

ID	First_name	Last_name	Age	Subject	Hobby
1	Amar	Sharma	20	Maths	Cricket

2	Akbar	Khan	22	Biology	Football
3	Anthony	Milton	25	Commerce	Gambling

From the above example, select the first name of all the students. To do so, query should be like this:

SELECT first_name **FROM** student_details;

Note: the SQL commands are not case sensitive. We can also write the above SELECT statement as:

select first_name **from** student_details;

Now, you will get following data:

Amar
Akbar
Anthony

We can also retrieve data from more than one column. For example, to select first name and last name of all the students, you need to write

SELECT first_name, last_name **FROM** student_details;

Now, you will get following data:

Amar	Sharma
Akbar	Khan
Anthony	Milton

We can also use clauses like WHERE, GROUP BY, HAVING, ORDER BY with SELECT statement.

Here a point is notable that only SELECT and FROM statements are necessary in SQL SELECT statements. Other clauses like WHERE, GROUP BY, ORDER BY, HAVING may be optional.

SQL SELECT DISTINCT

The **SQL DISTINCT command** is used with SELECT key word to retrieve only distinct or unique data.

In a table, there may be a chance to exist a duplicate value and sometimes we want to retrieve only unique values. In such scenarios, SQL SELECT DISTINCT statement is used.

Note: SQL SELECT UNIQUE and SQL SELECT DISTINCT statements are same.

Let's see the syntax of select distinct statement.

```
SELECT DISTINCT column_name ,column_name
FROM  table_name;
```

Let's try to understand it by the table given below:

Student_Name	Gender	Mobile_Number	HOME_TOWN
Rahul Ojha	Male	7503896532	Lucknow
Disha Rai	Female	9270568893	Varanasi
Sonoo Jaiswal	Male	9990449935	Lucknow

Here is a table of students from where we want to retrieve distinct information For example: distinct home-town.

```
SELECT DISTINCT home_town
FROM students
```

Now, it will return two rows.

HOME_TOWN
Lucknow
Varanasi

SQL SELECT COUNT

The **SQL COUNT()** function is used to return the number of rows in a query.

The COUNT() function is used with SQL SELECT statement and it is very useful to count the number of rows in a table having enormous data.

For example: If you have a record of the voters in selected area and want to count the number of voters then it is very difficult to do it manually but you can do it easily by using the SQL SELECT COUNT query.

Let's see the syntax of SQL COUNT statement.

```
SELECT COUNT (expression)
FROM tables
WHERE conditions;
```

Let's see the examples of sql select count function.

SQL SELECT COUNT(column_name)

```
SELECT COUNT(name) FROM employee_table;
```

It will return the total number of names of employee_table. But null fields will not be counted.

SQL SELECT COUNT(*)

```
SELECT COUNT(*) FROM employee_table;
```

The "select count(*) from table" is used to return the number of records in table.

SQL SELECT COUNT(DISTINCT column_name)

```
SELECT COUNT(DISTINCT name) FROM employee_table;
```

It will return the total distinct names of employee_table.

SQL SELECT AS

SQL AS is used to assign temporarily a new name to a table column.

It makes easy presentation of query results and allows the developer to label results more accurately without permanently renaming table columns.

Let's see the example of select as:

```
SELECT day_of_order AS "Date"
Customer As "Client",
Product,
Quantity,
FROM orders;
```

Let us take a table named orders, it contains:

Day_of_order	Customer	Product	Quantity
11-09-2001	Ajeet	Mobile	2
13-12-2001	Mayank	Laptop	20
26-12-2004	Balaswamy	Water cannon	35

After applying this SQL AS example syntax

```
SELECT day_of_order AS "Date"
Customer As "Client",
Product,
Quantity,
```

FROM orders;

Result will be shown as this table:

Date	Client	Product	Quantity
11-09-2001	Ajeet	Mobile	2
13-12-2001	Mayank	Laptop	20
26-12-2004	Balaswamy	Water cannon	35

Note: SQL AS is same as SQL ALIAS.

SQL SELECT IN

SQL IN is an operator used in a SQL query to help reduce the need to use multiple SQL "OR" conditions.

It is used in SELECT, INSERT, UPDATE or DELETE statement.

Advantage of SQL SELECT IN

It minimizes the use of SQL OR operator.

Let's see the syntax for SQL IN:

Expression IN (value 1, value 2 ... value n);

Take an example with character values.

```
SELECT *
FROM students
WHERE students_name IN ( Amit , Raghav, Rajeev)
```

Let?s take another example with numeric values.

```
SELECT *
```

```
FROM marks
WHERE roll_no IN (001, 023, 024);
```

SQL UPDATE

The SQL commands (*UPDATE* and *DELETE*) are used to modify the data that is already in the database. The SQL DELETE command uses a WHERE clause.

SQL UPDATE statement is used to change the data of the records held by tables. Which rows is to be update, it is decided by a condition. To specify condition, we use WHERE clause.

The UPDATE statement can be written in following form:

```
UPDATE table_name SET [column_name1= value1,... column_nameN = valueN] [WHERE condition]
```

Let's see the Syntax:

```
UPDATE table_name
SET column_name = expression
WHERE conditions
```

Let's take an example: here we are going to update an entry in the source table.

SQL statement:

```
UPDATE students
SET User_Name = 'beinghuman'
WHERE Student_Id = '3'
```

Source Table:

Student_Id	FirstName	LastName	User_Name

1	Ada	Sharma	sharmili
2	Rahul	Maurya	sofamous
3	James	Walker	jonny

See the result after updating value:

Student_Id	FirstName	LastName	User_Name
1	Ada	Sharma	sharmili
2	Rahul	Maurya	sofamous
3	James	Walker	**beinghuman**

Updating Multiple Fields:

If you are going to update multiple fields, you should separate each field assignment with a comma.

SQL UPDATE statement for multiple fields:

```
UPDATE students
SET User_Name = 'beserious', First_Name = 'Johnny'
WHERE Student_Id = '3'
```

Result of the table is given below:

Student_Id	FirstName	LastName	User_Name
1	Ada	Sharma	sharmili
2	Rahul	Maurya	sofamous
3	**Johnny**	Walker	**beserious**

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

SQL DROP TABLE

A SQL DROP TABLE statement is used to delete a table definition and all data from a table.

This is very important to know that once a table is deleted all the information available in the table is lost forever, so we have to be very careful when using this command.

Let's see the syntax to drop the table from the database.

```
DROP TABLE "table_name";
```

Let us take an example:

First we verify STUDENTS table and then we would delete it from the database.

SQL> **DESC** STUDENTS;

FIELD	TYPE	NULL	KEY	DEFAULT	EXTRA
ID	Int(11)	NO	PRI		
NAME	Varchar(20)	NO			
AGE	Int(11)	NO			
ADDRESS	Varchar(25)	YES		NULL	

4 rows in set (0.00 sec)

This shows that STUDENTS table is available in the database, so we can drop it as follows:

SQL>**DROP TABLE** STUDENTS;

Now, use the following command to check whether table exists or not.

SQL> **DESC** STUDENTS;
Query OK, 0 rows affected (0.01 sec)

As you can see, table is dropped so it doesn't display it.

SQL DELETE

The **SQL DELETE statement** is used to delete rows from a table. Generally DELETE statement removes one or more records from a table.

SQL DELETE Syntax

Let's see the Syntax for the SQL DELETE statement:

DELETE FROM table_name [**WHERE** condition];

Here table_name is the table which has to be deleted. The *WHERE clause* in SQL DELETE statement is optional here.

SQL DELETE Example

Let us take a table, named ?EMPLOYEE? table.

ID	EMP_NAME	CITY	SALARY
101	Adarsh Singh	Obra	**20000**
102	Sanjay Singh	Meerut	**21000**
103	Priyanka Sharma	Raipur	**25000**
104	Esha Singhal	Delhi	**26000**

Example of delete with WHERE clause is given below:

DELETE FROM EMPLOYEE **WHERE** ID=101;

Resulting table after the query:

ID	EMP_NAME	CITY	SALARY
102	Sanjay Singh	Meerut	**21000**
103	Priyanka Sharma	Raipur	**25000**
104	Esha Singhal	Delhi	**26000**

Another example of delete statement is given below

DELETE FROM EMPLOYEE;

Resulting table after the query:

ID	EMP_NAME	CITY	SALARY

It will delete all the records of EMPLOYEE table.

It will delete the all the records of EMPLOYEE table where ID is 101.

The WHERE clause in the SQL DELETE statement is optional and it identifies the rows in the column that gets deleted.

WHERE clause is used to prevent the deletion of all the rows in the table, If you don't use the WHERE clause you might loss all the rows.

SQL DELETE TABLE

The DELETE statement is used to delete rows from a table. If you want to remove a specific row from a table you should use WHERE condition.

DELETE FROM table_name [**WHERE** condition];

But if you do not specify the WHERE condition it will remove all the rows from the table.

DELETE FROM table_name;

There are some more terms similar to DELETE statement like as DROP statement and TRUNCATE statement but they are not exactly same there are some differences between them.

Difference between DELETE and TRUNCATE statements

There is a slight difference b/w delete and truncate statement. The **DELETE statement** only deletes the rows from the table based on the condition defined by WHERE clause or delete all the rows from the table when condition is not specified.

But it does not free the space containing by the table.

The **TRUNCATE statement:** it is used to delete all the rows from the table **and free the containing space.**

Let's see an "employee" table.

Emp_id	Name	Address	Salary
1	Aryan	Allahabad	22000
2	Shurabhi	Varanasi	13000
3	Pappu	Delhi	24000

Execute the following query to truncate the table:

TRUNCATE TABLE employee;

Difference b/w DROP and TRUNCATE statements

When you use the drop statement it deletes the table's row together with the table's definition so all the relationships of that table with other tables will no longer be valid.

When you drop a table:

- Table structure will be dropped
- Relationship will be dropped
- Integrity constraints will be dropped
- Access privileges will also be dropped

On the other hand when we **TRUNCATE** a table, the table structure remains the same, so you will not face any of the above problems.

SQL DELETE ROW

Let us take an example of student.

Original table:

ID	STUDENT _NAME	ADDRESS
001	AJEET MAURYA	GHAZIABAD
002	RAJA KHAN	LUCKNOW
003	RAVI MALIK	DELHI

If you want to delete a student with id 003 from the student_name table, then the SQL DELETE query should be like this:

```
DELETE FROM student_name
WHERE id = 003;
```

Resulting table after SQL DELETE query:

ID	STUDENT_NAME	ADDRESS
001	AJEET MAURYA	GHAZIABAD
002	RAJA KHAN	LUCKNOW

SQL DELETE ALL ROWS

The statement SQL DELETE ALL ROWS is used to delete all rows from the table. If you want to delete all the rows from student table the query would be like,

```
DELETE FROM STUDENT_NAME;
```

Resulting table after using this query:

ID	STUDENT_NAME	ADDRE

SQL ALTER TABLE

The ALTER TABLE statement is used to add, modify or delete columns in an existing table. It is also used to rename a table.

You can also use SQL ALTER TABLE command to add and drop various constraints on an existing table.

SQL ALTER TABLE Add Column

If you want to add columns in SQL table, the SQL alter table syntax is given below:

```
ALTER TABLE table_name ADD column_name column-definition;
```

If you want to add multiple columns in table, the SQL table will be

```
ALTER TABLE table_name
ADD (column_1 column-definition,
      column_2 column-definition,
      .....
      column_n column-definition);
```

SQL ALTER TABLE Modify Column

If you want to modify an existing column in SQL table, syntax is given below:

ALTER TABLE table_name **MODIFY** column_name column_type;

If you want to modify multiple columns in table, the SQL table will be

```
ALTER TABLE table_name
MODIFY (column_1 column_type,
        column_2 column_type,
       .....
        column_n column_type);
```

More SQL

Complex Queries

The SQL AND & OR operators are used to combine multiple conditions to narrow data in an SQL statement. These two operators are called as the conjunctive operators.

These operators provide a means to make multiple comparisons with different operators in the same SQL statement.

The AND Operator

The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause.

Syntax

The basic syntax of the AND operator with a WHERE clause is as follows −

```
SELECT column1, column2, columnN

FROM table_name

WHERE [condition1] AND [condition2]...AND [conditionN];
```

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

You can combine N number of conditions using the AND operator. For an action to be taken by the SQL statement, whether it be a transaction or a query, all conditions separated by the AND must be TRUE.

Example

Consider the CUSTOMERS table having the following records −

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

Following is an example, which would fetch the ID, Name and Salary fields from the CUSTOMERS table, where the salary is greater than 2000 and the age is less than 25 years −

```
SQL> SELECT ID, NAME, SALARY FROM CUSTOMERS WHERE SALARY > 2000 AND age < 25;
```

This would produce the following result −

```
+----+-------+----------+
| ID | NAME  | SALARY   |
+----+-------+----------+
|  6 | Komal |  4500.00 |
|  7 | Muffy | 10000.00 |
+----+-------+----------+
```

The OR Operator

The OR operator is used to combine multiple conditions in an SQL statement's WHERE clause.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

Syntax

The basic syntax of the OR operator with a WHERE clause is as follows −

SELECT column1, column2, columnN

FROM table_name

WHERE [condition1] OR [condition2]...OR [conditionN]

You can combine N number of conditions using the OR operator. For an action to be taken by the SQL statement, whether it be a transaction or query, the only any ONE of the conditions separated by the OR must be TRUE.

Example

Consider the CUSTOMERS table having the following records −

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

The following code block hasa query, which would fetch the ID, Name and Salary fields from the CUSTOMERS table, where the salary is greater than 2000 and the age is less than 25 years.

SQL> SELECT ID, NAME, SALARY FROM CUSTOMERS WHERE SALARY > 2000 OR age < 25;

This would produce the following result −

```
+----+----------+----------+
| ID | NAME     | SALARY   |
```

```
+----+----------+----------+
| 3 | kaushik  |  2000.00 |
| 4 | Chaitali |  6500.00 |
| 5 | Hardik   |  8500.00 |
| 6 | Komal    |  4500.00 |
| 7 | Muffy    | 10000.00 |
+----+----------+----------+
```

The NOT Operator

EX: SELECT Id, Name, salary FROM Customers WHERE NOT address = 'MP'

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
| 1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
| 2 | Khilan   |  25 | Delhi     |  1500.00 |
| 3 | kaushik  |  23 | Kota      |  2000.00 |
| 4 | Chaitali |  25 | Mumbai    |  6500.00 |
| 5 | Hardik   |  27 | Bhopal    |  8500.00 |
| 7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

LIKE Clause

The SQL LIKE clause is used to compare a value to similar values using wildcard operators. There are two wildcards used in conjunction with the LIKE operator.

The percent sign (%)

The underscore (_)

The percent sign represents zero, one or multiple characters. The underscore represents a single number or character. These symbols can be used in combinations.

Syntax

The basic syntax of % and _ is as follows −

SELECT FROM table_name WHERE column LIKE 'XXXX%'

Or

SELECT FROM table_name WHERE column LIKE '_XXXX'

Example

Following is an example, which would display all the records from the CUSTOMERS table, where the SALARY starts with 200.

SQL> SELECT * FROM CUSTOMERS WHERE SALARY LIKE '200%';

This would produce the following result −

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
+----+----------+-----+-----------+----------+
```

TOP, LIMIT

Syntax

The basic syntax of the TOP clause with a SELECT statement would be as follows.

SELECT TOP number|percent column_name(s)

FROM table_name

WHERE [condition]

The following query is an example on the SQL server, which would fetch the top 3 records from the CUSTOMERS table.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

SQL> SELECT TOP 3 * FROM CUSTOMERS;

This would produce the following result −

```
+----+---------+-----+-----------+---------+
| ID | NAME    | AGE | ADDRESS   | SALARY  |
+----+---------+-----+-----------+---------+
|  1 | Ramesh  |  32 | Ahmedabad | 2000.00 |
|  2 | Khilan  |  25 | Delhi     | 1500.00 |
|  3 | kaushik |  23 | Kota      | 2000.00 |
+----+---------+-----+-----------+---------+
```

If you are using MySQL server, then here is an equivalent example −

SQL> SELECT * FROM CUSTOMERS LIMIT 3;

This would produce the following result −

```
+----+---------+-----+-----------+---------+
| ID | NAME    | AGE | ADDRESS   | SALARY  |
+----+---------+-----+-----------+---------+
|  1 | Ramesh  |  32 | Ahmedabad | 2000.00 |
|  2 | Khilan  |  25 | Delhi     | 1500.00 |
|  3 | kaushik |  23 | Kota      | 2000.00 |
+----+---------+-----+-----------+---------+
```

If you are using an Oracle server, then the following code block has an equivalent example.

SQL> SELECT * FROM CUSTOMERS WHERE ROWNUM <= 3;

This would produce the following result −

```
+----+---------+-----+-----------+---------+
| ID | NAME    | AGE | ADDRESS   | SALARY  |
+----+---------+-----+-----------+---------+
|  1 | Ramesh  |  32 | Ahmedabad | 2000.00 |
|  2 | Khilan  |  25 | Delhi     | 1500.00 |
|  3 | kaushik |  23 | Kota      | 2000.00 |
+----+---------+-----+-----------+---------+
```

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

ORDER BY Clause

The SQL ORDER BY clause is used to sort the data in ascending or descending order, based on one or more columns. Some databases sort the query results in an ascending order by default.

Syntax

The basic syntax of the ORDER BY clause is as follows −

SELECT column-list

FROM table_name

[WHERE condition]

[ORDER BY column1, column2, .. columnN] [ASC | DESC];

You can use more than one column in the ORDER BY clause. Make sure whatever column you are using to sort that column should be in the column-list.

Example

The following code block has an example, which would sort the result in an ascending order by the NAME and the SALARY −

SQL> SELECT * FROM CUSTOMERS ORDER BY NAME, SALARY;

This would produce the following result −

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
```

```
| 7 | Muffy    | 24 | Indore    | 10000.00 |
| 1 | Ramesh   | 32 | Ahmedabad |  2000.00 |
+----+----------+-----+-----------+----------+
```

The following code block has an example, which would sort the result in the descending order by NAME.

```
SQL> SELECT * FROM CUSTOMERS  ORDER BY NAME DESC;
```

This would produce the following result −

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
| 1 | Ramesh   | 32 | Ahmedabad |  2000.00 |
| 7 | Muffy    | 24 | Indore    | 10000.00 |
| 6 | Komal    | 22 | MP        |  4500.00 |
| 2 | Khilan   | 25 | Delhi     |  1500.00 |
| 3 | kaushik  | 23 | Kota      |  2000.00 |
| 5 | Hardik   | 27 | Bhopal    |  8500.00 |
| 4 | Chaitali |  25 | Mumbai    |  6500.00 |
+----+----------+-----+-----------+----------+
```

Group By

The SQL GROUP BY clause is used in collaboration with the SELECT statement to arrange identical data into groups. This GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.

Syntax

The basic syntax of a GROUP BY clause is shown in the following code block. The GROUP BY clause must follow the conditions in the WHERE clause and must precede the ORDER BY clause if one is used.

```
SELECT column1, column2

FROM table_name
```

WHERE [conditions]

GROUP BY column1, column2

ORDER BY column1, column2

Example

Now, let us look at a table where the CUSTOMERS table has the following records with duplicate names −

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Ramesh   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | kaushik  |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

Now again, if you want to know the total amount of salary on each customer, then the GROUP BY query would be as follows −

```
SQL> SELECT NAME, SUM(SALARY) FROM CUSTOMERS    GROUP BY NAME;
```

This would produce the following result −

```
+---------+-------------+
| NAME    | SUM(SALARY) |
+---------+-------------+
| Hardik  |     8500.00 |
| kaushik |     8500.00 |
| Komal   |     4500.00 |
| Muffy   |    10000.00 |
| Ramesh  |     3500.00 |
+---------+-------------+
```

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

SQL | Views

Views in SQL are kind of virtual tables. A view also has rows and columns as they are in a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain condition.

Creating Views

Database views are created using the **CREATE VIEW** statement. Views can be created from a single table, multiple tables or another view.

To create a view, a user must have the appropriate system privilege according to the specific implementation.

The basic **CREATE VIEW** syntax is as follows −

```
CREATE VIEW view_name AS
SELECT column1, column2.....
FROM table_name
WHERE [condition];
```

You can include multiple tables in your SELECT statement in a similar way as you use them in a normal SQL SELECT query.

Example

Consider the CUSTOMERS table having the following records −

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

Following is an example to create a view from the CUSTOMERS table. This view would be used to have customer name and age from the CUSTOMERS table.

```
SQL > CREATE VIEW CUSTOMERS_VIEW AS
SELECT name, age
FROM  CUSTOMERS;
```

Now, you can query CUSTOMERS_VIEW in a similar way as you query an actual table. Following is an example for the same.

```
SQL > SELECT * FROM CUSTOMERS_VIEW;
```

This would produce the following result.

```
+----------+-----+
| name     | age |
+----------+-----+
| Ramesh   |  32 |
| Khilan   |  25 |
| kaushik  |  23 |
| Chaitali |  25 |
| Hardik   |  27 |
| Komal    |  22 |
| Muffy    |  24 |
+----------+-----+
```

Updating a View

Syntax:

UPDATE **view-name**

set value

WHERE condition;

```
SQL > UPDATE CUSTOMERS_VIEW
```

```
SET AGE = 35

WHERE name = 'Ramesh';
```

This would ultimately update the base table CUSTOMERS and the same would reflect in the view itself. Now, try to query the base table and the SELECT statement would produce the following result.

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  35 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

Inserting Rows into a View

Rows of data can be inserted into a view. The same rules that apply to the UPDATE command also apply to the INSERT command.

Deleting Rows into a View

Rows of data can be deleted from a view. The same rules that apply to the UPDATE and INSERT commands apply to the DELETE command.

Following is an example to delete a record having AGE = 22.

```
SQL > DELETE FROM CUSTOMERS_VIEW

  WHERE age = 22;
```

This would ultimately delete a row from the base table CUSTOMERS and the same would reflect in the view itself. Now, try to query the base table and the SELECT statement would produce the following result.

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  35 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

Dropping Views

Obviously, where you have a view, you need a way to drop the view if it is no longer needed. The syntax is very simple and is given below −

```
DROP VIEW view_name;
```

Following is an example to drop the CUSTOMERS_VIEW from the CUSTOMERS table.

```
DROP VIEW CUSTOMERS_VIEW;
```

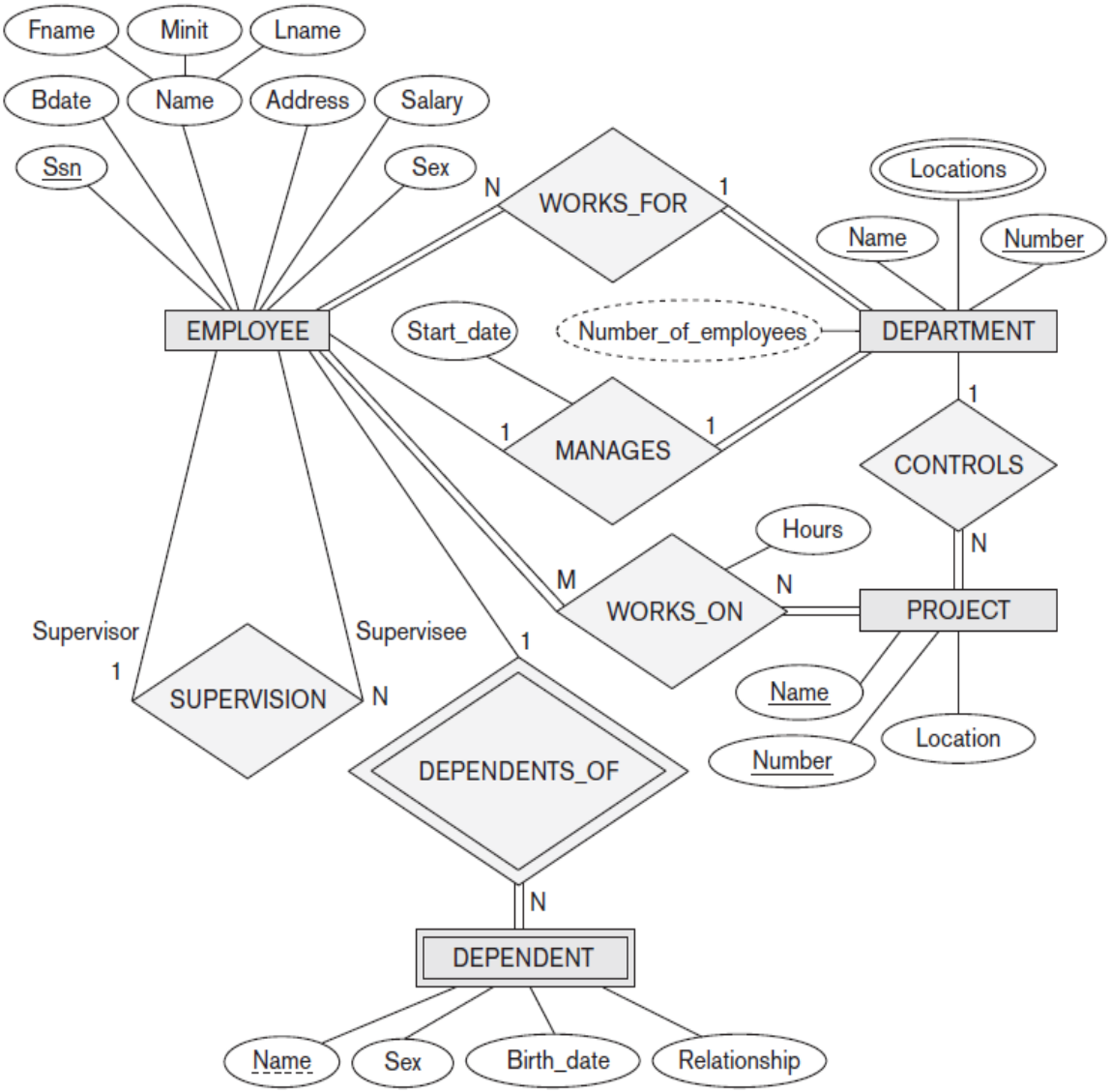
DATABASE DESIGN

Relational Database Design by ER- and EER-to Relational Mapping

Design a relational database schema Based on a conceptual schema design
Seven-step algorithm to convert the basic
ER model constructs into relations Additional steps for EER model

Mapping ER/EER model to relational database

Figure 9.1
The ER conceptual schema diagram for the COMPANY database.



CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

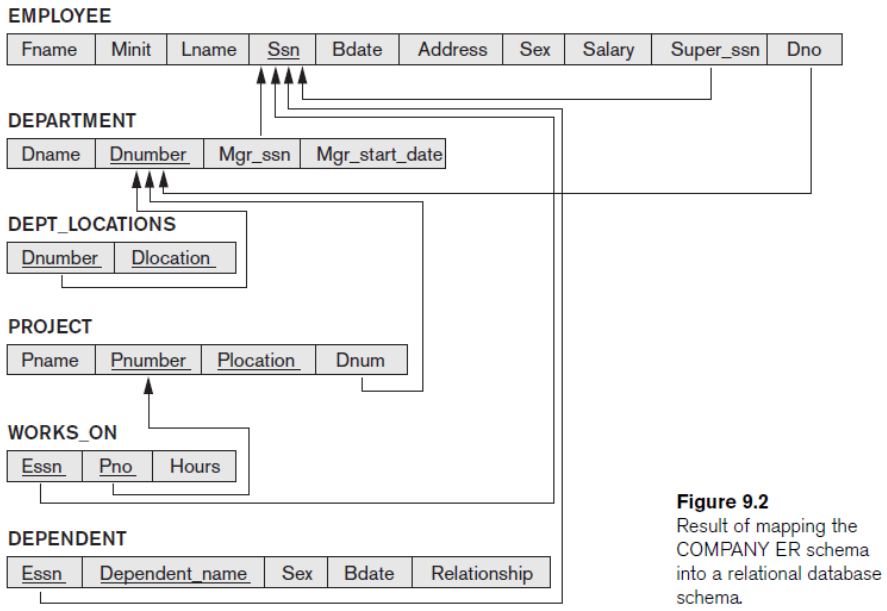


Figure 9.2
Result of mapping the COMPANY ER schema into a relational database schema.

ER-to-Relational Mapping Algorithm

COMPANY database example

Assume that the mapping will create tables with simple single-valued attributes

Step 1: Mapping of Regular Entity Types

For each regular entity type, create a relation *R* that includes all the simple attributes of *E* Called **entity relations**

• Each tuple represents an entity instance

Step 2: Mapping of Weak Entity Types

For each weak entity type, create a relation *R*and include all simple attributes of the entity type as attributes of *R*

Include primary key attribute of owner as foreign key attributes of *R*

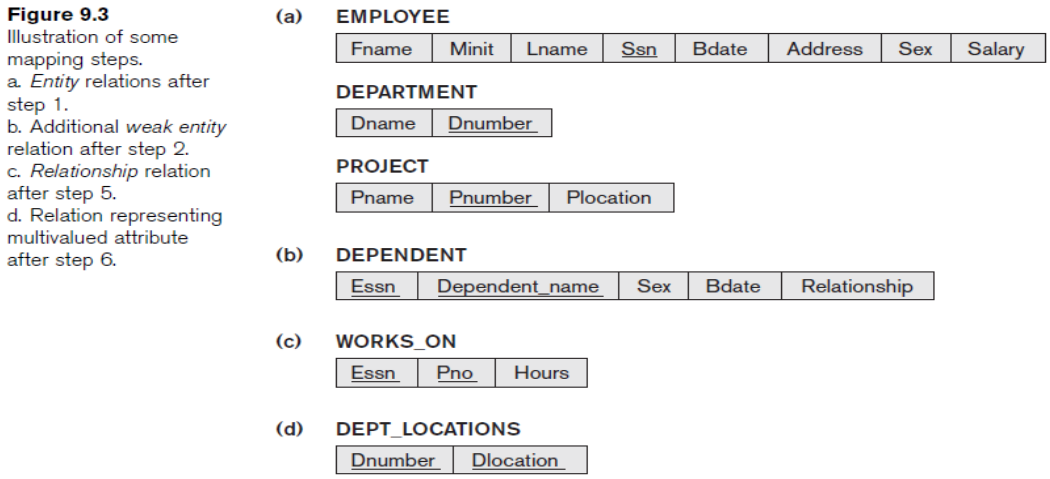


Figure 9.3
Illustration of some mapping steps.
a. *Entity* relations after step 1.
b. Additional *weak entity* relation after step 2.
c. *Relationship* relation after step 5.
d. Relation representing multivalued attribute after step 6.

Step 3: Mapping of Binary 1:1 Relationship

Types

For each binary 1:1 relationship type
Identify relations that correspond to entity types participating in *R*

Possible approaches:

- Foreign key approach
- Merged relationship approach
- Crossreference or relationship relation approach

Step 4: Mapping of Binary 1:*N* Relationship

Types

For each regular binary 1:*N* relationship type

- Identify relation that represents participating entity type at *N*-side of relationship type
- Include primary key of other entity type as foreign key in *S*
- Include simple attributes of 1:*N* relationship type as attributes of *S*

Alternative approach

- Use the **relationship relation** (cross-reference) option as in the third option for binary 1:1 relationships

Step 5: Mapping of Binary *M*:*N* Relationship

Types

For each binary *M*:*N* relationship type

- Create a new relation *S*

- Include primary key of participating entity types as foreignkey attributes inS
- Include any simple attributes of *M*:*N* relationship type

Step 6: Mapping of Multivalued Attributes

For each multivalued attribute

- Create a new relation
- Primary key of *R* is the combination of *A* and *K*
- If the multivalued attribute is composite, include its simple components

Step 7: Mapping of *N*-ary Relationship Types

For each *n*-ary relationship type *R*

- Create a new relation *S* to represent *R*
- Include primary keys of participating entity types as foreign keys
- Include any simple attributes as attributes

Table 9.1 Correspondence between ER and Relational Models

ER MODEL	RELATIONAL MODEL
Entity type	*Entity* relation
1:1 or 1:N relationship type	Foreign key (or *relationship* relation)
M:N relationship type	*Relationship* relation and *two* foreign keys
n-ary relationship type	*Relationship* relation and *n* foreign keys
Simple attribute	Attribute
Composite attribute	Set of simple component attributes
Multivalued attribute	Relation and foreign key
Value set	Domain
Key attribute	Primary (or secondary) key

In a relational schema relationship, types are not represented explicitly
Represented by having two attributes *A* and *B*: one a primary key and the other a foreign key

Mapping EER Model Constructs to Relations

Extending ER-to-relational mapping Algorithm

Mapping of Specialization or Generalization

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

Step 8: Options for Mapping Specialization or Generalization

Option 8A: Multiple relations—superclass and subclasses
• For any specialization (total or partial, disjoint or overlapping)

Option 8B: Multiple relations—subclass relations only
• Subclasses are total
• Specialization has disjointedness constraint

Option 8C: Single relation with one type attribute
• Type or discriminating attribute indicates subclass of tuple
• Subclasses are disjoint
• Potential for generating many NULL values if many specific attributes exist in the subclasses

Option 8D: Single relation with multiple type attributes
• Subclasses are overlapping
• Will also work for a disjoint specialization

Apply any of the options discussed in step 8 to a shared subclass

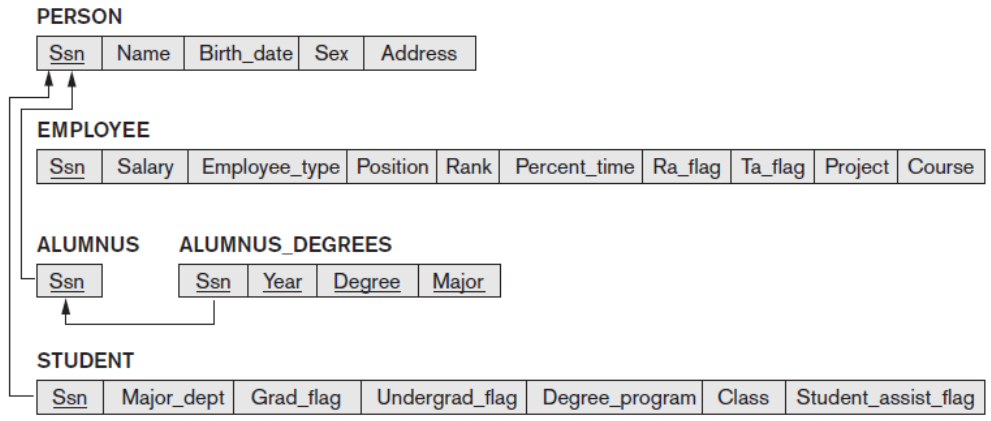


Figure 9.6
Mapping the EER specialization lattice in Figure 8.8 using multiple options.

Step 9: Mapping of Union Types (Categories)

Defining superclasses have different keys
Specify a new key attribute
• **Surrogate key**

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

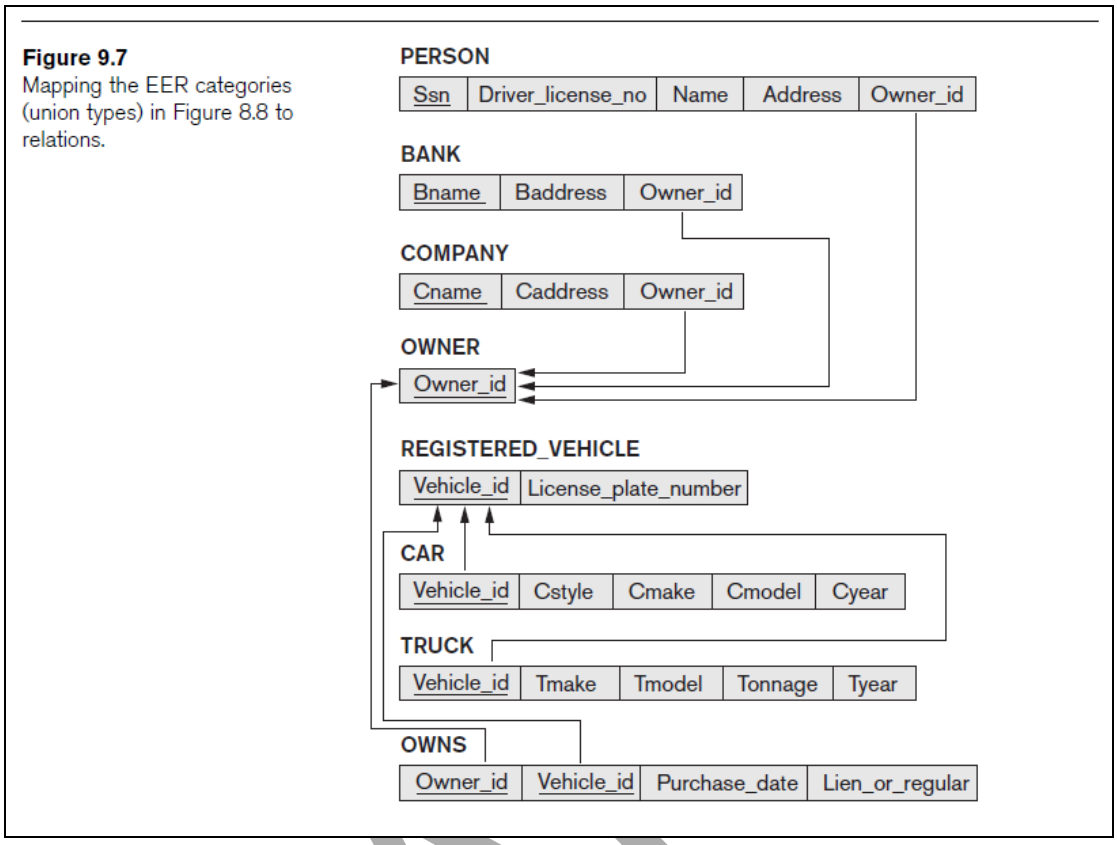


Figure 9.7
Mapping the EER categories (union types) in Figure 8.8 to relations.

Summary

Map conceptual schema design in the ER model to a relational database schema
Algorithm for ER-to-relational mapping
Illustrated by examples from the COMPANY database
Include additional steps in the algorithm for mapping constructs from EER model into relational model

What is functional dependency?

- **Functional Dependency** is a relationship that exists between multiple attributes of a relation.
- This concept is given by **E. F. Codd.**
- Functional dependency represents a formalism on the infrastructure of relation.
- It is a type of constraint existing between various attributes of a relation.
- It is used to define various normal forms.
- These dependencies are restrictions imposed on the data in database.

- If P is a relation with A and B attributes, a functional dependency between these two attributes is represented as {A → B}. It specifies that,

A	It is a determinant set.
B	It is a dependent attribute.
{A → B}	A functionally determines B. B is a functionally dependent on A.

- Each value of A is associated precisely with one B value. A functional dependency is trivial if B is a subset of A.
- 'A' Functionality determines 'B' {A → B} (Left hand side attributes determine the values of Right hand side attributes).

For example: <Employee> Table

EmpId	EmpName

- In the above <Employee> table, EmpName (employee name) is functionally dependent on EmpId (employee id) because the EmpId is unique for individual names.
- The EmpId identifies the employee specifically, but EmpName cannot distinguish the EmpId because more than one employee could have the same name.
- The functional dependency between attributes eliminates the repetition of information.
- It is related to a candidate key, which uniquely identifies a tuple and determines the value of all other attributes in the relation.

Advantages of Functional Dependency

- Functional Dependency avoids data redundancy where same data should not be repeated at multiple locations in same database.
- It maintains the quality of data in database.

- It allows clearly defined meanings and constraints of databases.
- It helps in identifying bad designs.
- It expresses the facts about the database design.

Introduction to Axioms Rules

- Armstrong's Axioms is a set of rules.
- It provides a simple technique for reasoning about functional dependencies.
- It was developed by William W. Armstrong in 1974.
- It is used to infer all the functional dependencies on a relational database.

Various Axioms Rules

A. Primary Rules

Rule 1	**Reflexivity** If A is a set of attributes and B is a subset of A, then A holds B. { A → B }
Rule 2	**Augmentation** If A hold B and C is a set of attributes, then AC holds BC. {AC → BC} It means that attribute in dependencies does not change the basic dependencies.
Rule 3	**Transitivity** If A holds B and B holds C, then A holds C. If {A → B} and {B → C}, then {A → C} A holds B {A → B} means that A functionally determines B.

B. Secondary Rules

Rule 1	**Union** If A holds B and A holds C, then A holds BC. If{A → B} and {A → C}, then {A → BC}
Rule 2	**Decomposition** If A holds BC and A holds B, then A holds C. If{A → BC} and {A → B}, then {A → C}

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

Rule 3	**Pseudo Transitivity** If A holds B and BC holds D, then AC holds D. If{A → B} and {BC → D}, then {AC → D}

Sometimes Functional Dependency Sets are not able to reduce if the set has following properties,

1. The Right-hand side set of functional dependency holds only one attribute.
2. The Left-hand side set of functional dependency cannot be reduced, it changes the entire content of the set.
3. Reducing any functional dependency may change the content of the set.

A set of functional dependencies with the above three properties are also called as **Canonical or Minimal.**

Trivial Functional Dependency

Trivial	If A holds B {A → B}, where A is a subset of B, then it is called a **Trivial Functional Dependency**. Trivial always holds Functional Dependency.
Non-Trivial	If A holds B {A → B}, where B is not a subset A, then it is called as a **Non-Trivial Functional Dependency.**
Completely Non-Trivial	If A holds B {A → B}, where A intersect Y = Φ, it is called as a **Completely Non-Trivial Functional Dependency.**

Example:
Consider relation E = (P, Q, R, S, T, U) having set of Functional Dependencies (FD).

P → Q P → R
QR → S Q → T
QR → U PR → U

Calculate some members of Axioms are as follows,

1. P → T
2. PR → S
3. QR → SU
4. PR → SU

Solution:

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

1. P → T
In the above FD set, P → Q and Q → T
So, Using Transitive Rule: If {A → B} and {B → C}, then {A → C}
∴ If P → Q and Q → T, then **P → T.**
P → T

2. PR → S
In the above FD set, P → Q
As, QR → S
So, Using Pseudo Transitivity Rule: If{A → B} and {BC → D}, then {AC → D}
∴ If P → Q and QR → S, **then PR → S.**
PR → S

3. QR → SU
In above FD set, QR → S and QR → U
So, Using Union Rule: If{A → B} and {A → C}, then {A → BC}
∴ If QR → S and QR → U, **then QR → SU.**
QR → SU

4. PR → SU
So, Using Pseudo Transitivity Rule: If{A → B} and {BC → D}, then {AC → D}
∴ If PR → S and PR → U, **then PR → SU.**
PR → SU

What is decomposition?

- Decomposition is the process of breaking down in parts or elements.
- It replaces a relation with a collection of smaller relations.
- It breaks the table into multiple tables in a database.
- It should always be lossless, because it confirms that the information in the original relation can be accurately reconstructed based on the decomposed relations.
- If there is no proper decomposition of the relation, then it may lead to problems like loss of information.

Properties of Decomposition

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

Following are the properties of Decomposition,

1. Lossless Decomposition
2. Dependency Preservation
3. Lack of Data Redundancy

1. Lossless Decomposition

- Decomposition must be lossless. It means that the information should not get lost from the relation that is decomposed.
- It gives a guarantee that the join will result in the same relation as it was decomposed.

Example:
Let's take 'E' is the Relational Schema, With instance 'e'; is decomposed into: E1, E2, E3, ... En; With instance: e1, e2, e3, ... en, If e1 ⋈ e2 ⋈ e3 . . . . ⋈ en, then it is called as **'Lossless Join Decomposition'.**

- In the above example, it means that, if natural joins of all the decomposition give the original relation, then it is said to be lossless join decomposition.

Example: <Employee_Department> Table

Eid	Ename	Age	City	Salary	Deptid	DeptName
E001	ABC	29	Pune	20000	D001	Finance
E002	PQR	30	Pune	30000	D002	Production
E003	LMN	25	Mumbai	5000	D003	Sales
E004	XYZ	24	Mumbai	4000	D004	Marketing
E005	STU	32	Bangalore	25000	D005	Human Resource

- Decompose the above relation into two relations to check whether a decomposition is lossless or lossy.
- Now, we have decomposed the relation that is Employee and Department.

Eid	Ename	Age	City	Salary
E001	ABC	29	Pune	20000

E002	PQR	30	Pune	30000
E003	LMN	25	Mumbai	5000
E004	XYZ	24	Mumbai	4000
E005	STU	32	Bangalore	25000

Relation 1 : <Employee> Table

- Employee Schema contains (Eid, Ename, Age, City, Salary).

Relation 2 : <Department> Table

Deptid	Eid	DeptName
D001	E001	Finance
D002	E002	Production
D003	E003	Sales
D004	E004	Marketing
D005	E005	Human Resource

- Department Schema contains (Deptid, Eid, DeptName).
- So, the above decomposition is a Lossless Join Decomposition, because the two relations contains one common field that is 'Eid' and therefore join is possible.
- Now apply natural join on the decomposed relations.

Employee ⋈ Department

Eid	Ename	Age	City	Salary	Deptid	DeptName
E001	ABC	29	Pune	20000	D001	Finance
E002	PQR	30	Pune	30000	D002	Production
E003	LMN	25	Mumbai	5000	D003	Sales
E004	XYZ	24	Mumbai	4000	D004	Marketing

E005	STU	32	Bangalore	25000	D005	Human Resource

Hence, the decomposition is Lossless Join Decomposition.

- If the <Employee> table contains (Eid, Ename, Age, City, Salary) and <Department> table contains (Deptid and DeptName), then it is not possible to join the two tables or relations, because there is no common column between them. And it becomes **Lossy Join Decomposition.**

2. Dependency Preservation

- Dependency is an important constraint on the database.
- Every dependency must be satisfied by at least one decomposed table.
- If {A → B} holds, then two sets are functional dependent. And, it becomes more useful for checking the dependency easily if both sets in a same relation.
- This decomposition property can only be done by maintaining the functional dependency.
- In this property, it allows to check the updates without computing the natural join of the database structure.

3. Lack of Data Redundancy

- Lack of Data Redundancy is also known as a **Repetition of Information.**
- The proper decomposition should not suffer from any data redundancy.
- The careless decomposition may cause a problem with the data.
- The lack of data redundancy property may be achieved by Normalization process.

POSSIBLE QUESTIONS

2 MARK QUESTIONS

1. What is a view?
2. Define referential integrity.
3. Mention the difference between drop and truncate command in SQL.
4. Define Triggers.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

5. Define constraints
6. Write any seven attributes of student entity.
7. Define Concatenation operator.
8. Define functional dependencies.
9. Define candidate key with example.
10. Write syntax for any 2 set operators.
11. Define triggers with example.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT III

6 MARK QUESTIONS

1. What is Functional Dependency? Explain types and properties of FD's.
2. Explain the following with example
   i. i) INSERT II) UPDATE III) DROP
3. Briefly explain about views of data with proper example.
4. Discuss Lossless decomposition with neat sketch
5. Explain with example. i) Complex Queries ii) Views
6. How to Map ER/EER model to relational database?
7. Explain the advantages of decomposition? Discuss the problems faced in decomposition.
8. Explain with example i) Aggregation functions. Ii) UPDATE Statements in SQL

KARPAGAM ACADEMY OF HIGHER EDUCATION
DEPARTMENT OF COMPUTER SCIENCE
II BSc CS BATCH(2016 - 2019)
PART-A OBJECTIVE TYPE/ MULTIPLE CHOICE QUESTIONS
UNIT-3

sno	Questions	opt1	opt2	opt3	opt4	Answer
1	What does SQL stand for?	Structured Query Language	Strong Question Language	Structured Question Language	Structure Query Language	Structured Query Language
2	Create.Alter,Drop commands are_____ language commands	DML	TCL	DCL	DDL	DDL
3	Insert,Select,Update,delete commands are _____language commands	DML	TCL	DCL	DDL	DML
4	Selection Operation is used to ____from a relation	Select the columns	Select the rows	Select the table	none.	Select the rows
5	_____command is used to remove the table definition information	Delete	Remove	Destroy	Drop	Drop
6	_____is used to modify the structure of an existing table.	Modify	Alter	Change	Recreate	Alter
7	View can be dropped using _____command	Delete View	Remove View	Replace View	Drop View	Drop View
8	_____columns are not allowed to contain null values	Primary Key	Candidate key	foreign Key	Unique Key	Primary Key
9	_____are valuable to give the security to our original table	Original table	Duplicate table	Views	Tables	Views
10	_____clause is used to modify a particular row.	Update	Modify	Alter	Where	Where
11	_____is a condition specified on a database schema & restricts the data that can be stored in an instance of the database.	integrity Constraint	restriction	key	check	integrity Constraint
12	A set of fields that uniquely identifies a tuple according to a key constraint is called_____ for the relation	primary key	Candidate key	foreign key	Super key	Candidate key
13	_____is used to refer the primary key in another entity	Candidate key	referential key	foreign key	Primary key	foreign key
14	_____value is unknown or not applicable	not null	zero	unknown	null	null
15	_____statement is used to define a new table.	Create	Produce	Insert	Add	Create
16	Rows are inserted using the _____command	Create	Insert	Add	Make	Insert
17	rows are deleted using the _____command	Delete	drop	remove	alter	Delete
18	Modify the column values in an existing row using_____command	Modify	Alter	Update	Change	Update
19	_____command is used to remove the table definition information	Delete	Remove	Destroy	Drop	Drop
20	___is used to modify the structure of an existing table.	Modify	Alter	Change	Recreate	Alter
21	View can be dropped using _____command	Delete View	Remove View	Replace View	Drop View	Drop View
22	Duplicates are eliminated by using _____ keyword.	Remove	Distinct	RM	Redundant	Distinct
23	_____is a query that has another query embedded within it	Query	Subquery	QBE	QUEL	Subquery
24	_____ to calculate the number of values in the Column.	Count	aggregate	Cal	Calculate	Count
25	__ is used to calculate the sum of all values in the column	Total	Sum	Count	Collection	Sum
26	____is used to calculate the average of all values in the column	Total	Sum	Average	avg	avg
27	Which function is used to extract the maximum values in the relations?	max	maximum	excess	large	max
28	Which function is used to extract the maximum values in the relations?	Small	minimum	lower	min	min
29	Which clause is used, when we are using Group by clause, instead of where clause?	where	Having	Distinct	Group	Having
30	_____is used when the column value is either unknown or inapplicable.	zero	all	Empty	null	null
31	_____keyword specifies that the join condition is equality on all common attributes and the where clause is not required.	full outerjoin	left outer join	Natural	Right outerjoin	Natural
32	_____constraints for a single table.	Assertion	table constraints	default	union	table constraints
33	_____keyword is used to assign a default value with a domain.	Static	Default	Permanent	distinct	Default
34	Which constraint is used to check the ranges in the column values?	range	verify	check	condition	check
35	_____operator is another set comparison operator such as IN.	Exists	IN	avail	present	Exists
36	Which keyword is similar to NOT IN?	Exist	IN	Not EXIST	Except	Except
37	Which keyword is similar to IN?	Exist	IN	Not EXIST	Except	Exist
38	With SQL, how can you return the number of records in the "Persons" table?	SELECT COLUMNS(*) FROM Persons	SELECT COUNT() FROM Persons	SELECT COUNT(*) FROM Persons	SELECT COLUMNS() FROM Persons	SELECT COUNT(*) FROM Persons

39	How can you change "Hansen" into "Nilsen" in the "LastName" column in the Persons table?	UPDATE Persons SET LastName='Nilsen' WHERE LastName='Hansen'	MODIFY Persons SET LastName='Hansen' INTO LastName='Nilsen	UPDATE Persons SET LastName='Hansen' INTO LastName='Nilsen'	MODIFY Persons SET LastName='Nilsen' WHERE LastName='Hansen'	UPDATE Persons SET LastName='Nilsen' WHERE LastName='Hansen'
40	With SQL, how can you return all the records from a table named "Persons" sorted descending by "FirstName"?	SELECT * FROM Persons SORT 'FirstName' DESC	SELECT * FROM Persons ORDER BY FirstName DESC	SELECT * FROM Persons ORDER FirstName DESC	SELECT * FROM Persons SORT BY 'FirstName' DESC	SELECT * FROM Persons ORDER BY FirstName DESC
41	Which SQL keyword is used to sort the result-set?	ORDER	SORT	SORT BY	ORDER BY	ORDER BY
42	With SQL, how do you select all the records from a table named "Persons" where the value of the column "FirstName" starts with an "a"?	SELECT * FROM Persons WHERE FirstName LIKE '%a'	SELECT * FROM Persons WHERE FirstName='a'	SELECT * FROM Persons WHERE FirstName LIKE 'a%'	SELECT * FROM Persons WHERE FirstName='%a%'	SELECT * FROM Persons WHERE FirstName LIKE 'a%'
43	You need to calculate the total of all salaries in the accounting department. Which group function should you use?	SUM	COUNT	TOTAL	LARGEST	SUM
44	SELECT ROUND (45.953, -1), TRUNC (45.936, 2) FROM dual; which values are displayed?	46 and 45	46 and 45.93	50 and 45.93	50 and 45.9	46 and 45.93
45	Select operator is not a unary operator	TRUE	FALSE	not opeator	operator	FALSE
46	Project operator chooses subset of attributes or columns of a relation	TRUE	FALSE	not opeator	operator	TRUE
47	…… database is used as template for all databases created.	Master	Model	Tempdb	None of the above	Model
48	One aspect that has to be dealt with by the integrity subsystem is to ensure that only valid values can be assigned to each data items. This is referred to as	Data Security	Domain access	Data Control	Domain Integrity	Domain Integrity
49	…….. operator is basically a join followed by a project on the attributes of first relation.	Join	Semi-Join	Full Join	Inner Join	Semi-Join
50	Which of the following is not a binary operator in relational algebra?	Join	Semi-Join	Assignment	Project	Project
51	Centralizing the integrity checking directly under the DBMS …….. duplication and ensures the consistency and validity of the database.	Increases	Skips	Does not reduce	Reduces	Reduces
52	Which of the following is/are the DDL statements?	Create	Drop	Alter	All of the above	All of the above
53	In snapshot, …….. clause tells oracle how long to wait between refreshes.	Complete	Force	Next	Refresh	Refresh
54	…… defines rules regarding the values allowed in columns and is the standard mechanism for enforcing database integrity.	Column	Constraint	Index	Trigger	Constraint
55	If B is a subset of A ,then A→B indicates_____	Reflexivity	Augmentation	Transitivity	Decomposition	Reflexivity
56	If A→B, then AC→BC indicates_____	Reflexivity	Augmentation	Transitivity	Decomposition	Augmentation
57	If A→B and B→C, then A→C indicates_____	Reflexivity	Augmentation	Transitivity	Decomposition	Transitivity
58	If A→BC, then A→B and A→C indicates_____	Reflexivity	Augmentation	Transitivity	Decomposition	Decomposition
59	If A→B and A→C, then A→BC indicates_____	Reflexivity	Augmentation	Transitivity	Union	Union
60	If A→B and C→D, then AC→BD indicates_____	Composition	Augmentation	Transitivity	Union	Composition

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV

SYLLABUS

Database design: Normal forms (upto BCNF).**Transaction Processing :** ACID properties, concurrency control

Normal forms (upto BCNF).

Introduction to Normalization

- **Normalization** is a process of organizing the data in the database.
- It is a systematic approach of decomposing tables to eliminate data redundancy.
- It was developed by **E. F. Codd.**
- Normalization is a multi-step process that puts the data into a tabular form by removing the duplicate data from the relation tables.
- It is a step by step decomposition of complex records into simple records.
- It is also called as Canonical Synthesis.
- It is the technique of building database structures to store data.

Definition of Normalization

"Normalization is a process of designing a consistent database by minimizing redundancy and ensuring data integrity through decomposition which is lossless."

Features of Normalization

- Normalization avoids the data redundancy.
- It is a formal process of developing data structures.
- It promotes the data integrity.
- It ensures data dependencies make sense that means data is logically stored.
- It eliminates the undesirable characteristics like Insertion, Updation and Deletion Anomalies.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV

Types of Normalization

Following are the types of Normalization:

1. First Normal Form
2. Second Normal Form
3. Third Normal Form
4. Fourth Normal Form
5. Fifth Normal Form
6. BCNF (Boyce – Codd Normal Form)
7. DKNF (Domain Key Normal Form)

1. First Normal Form (1NF)

- First Normal Form (1NF) is a simple form of Normalization.
- It simplifies each attribute in a relation.
- In 1NF, there should not be any repeating group of data.
- Each set of column must have a unique value.
- It contains atomic values because the table cannot hold multiple values.

Example: Employee Table

ECode	Employee_Name	Department_Name
1	ABC	Sales, Production
2	PQR	Human Resource
3	XYZ	Quality Assurance, Marketing

Employee Table using 1NF

ECode	Employee_Name	Department_Name
1	ABC	Sales

1	ABC	Production
2	PQR	Human Resource
3	XYZ	Quality Assurance
3	XYZ	Marketing

2. Second Normal Form (2NF)

- In 2NF, the table is required in 1NF.
- The main rule of 2NF is, *'No non-prime attribute is dependent on the proper subset of any candidate key of the table.'*
- An attribute which is not part of candidate key is known as non-prime attribute.

Example : Employee Table using 1NF

ECode	Employee_Name	Employee_Age
1	ABC	38
1	ABC	38
2	PQR	38
3	XYZ	40
3	XYZ	40

Candidate Key: ECode, Employee_Name
Non prime attribute: Employee_Age

- The above table is in 1NF. Each attribute has atomic values. However, it is not in 2NF because non prime attribute Employee_Age is dependent on ECode alone, which is a proper subset of candidate key. This violates the rule for 2NF as the rule says 'No non-prime attribute is dependent on the proper subset of any candidate key of the table'.

2NF (Second Normal Form) : Employee1 Table

ECode	Employee_Age

1	38
2	38
3	40

Employee2 Table

ECode	Employee_Name
1	ABC
1	ABC
2	PQR
3	XYZ
3	XYZ

- Now, the above tables comply with the Second Normal Form (2NF).

3. Third Normal Form (3NF)

- Third Normal Form (3NF) is used to minimize the transitive redundancy.
- In 3NF, the table is required in 2NF.
- While using the 2NF table, there should not be any transitive partial dependency.
- 3NF reduces the duplication of data and also achieves the data integrity.

Example : <Employee> Table

EId	Ename	DOB	City	State	Zip
001	ABC	10/05/1990	Pune	Maharashtra	411038
002	XYZ	11/05/1988	Mumbai	Maharashtra	400007

- In the above <Employee> table, EId is a primary key but City, State depends upon Zip code.

- The dependency between Zip and other fields is called Transitive Dependency.
- Therefore we apply 3NF. So, we need to move the city and state to the new <Employee_Table2> table, with Zip as a Primary key.

<Employee_Table1> Table

EId	Ename	DOB	Zip
001	ABC	10/05/1990	411038
002	XYZ	11/05/1988	400007

<Employee_Table2> Table

City	State	Zip
Pune	Maharashtra	411038
Mumbai	Maharashtra	400007

- The advantage of removing transitive dependency is, it reduces the amount of data dependencies and achieves the data integrity.
- In the above example, using with the 3NF, there is no redundancy of data while inserting the new records.
- The City, State and Zip code will be stored in the separate table. And therefore the updation becomes more easier because of no data redundancy.

4. BCNF (Boyce – Code Normal Form)

- BCNF which stands for Boyce – Code Normal From is developed by Raymond F. Boyce and E. F. Codd in 1974.
- BCNF is a higher version of 3NF.
- It deals with the certain type of anomaly which is not handled by 3NF.
- A table complies with BCNF if it is in 3NF and any attribute is fully functionally dependent that is A → B. (Attribute 'A' is determinant).

- If every determinant is a candidate key, then it is said to be BCNF.
- Candidate key has the ability to become a primary key. It is a column in a table.

Example : <EmployeeMain> Table

Empid	Ename	DeptName	DepType
E001	ABC	Production	D001
E002	XYZ	Sales	D002

The functional dependencies are:
Empid → EmpName
DeptName → DeptType

Candidate Key:
Empid
DeptName

- The above table is not in BCNF as neither Empid nor DeptName alone are keys.
- We can break the table in three tables to make it comply with BCNF.

<Employee> Table

Empid	EmpName
E001	ABC
E002	XYZ

<Department> Table

DeptName	DeptType
Production	D001
Sales	D002

<Emp_Dept> Table

Empid	DeptName

E001	Production
E002	Sales

Now, the functional dependencies are:
Empid → EmpName
DeptName → DeptType

Candidate Key:
<Employee> Table : Empid
<Department> Table : DeptType
<Emp_Dept> Table : Empid, DeptType

- So, now both the functional dependencies left side part is a key, so it is in the BCNF.

5. Fourth Normal Form (4NF)

- Fourth Normal Form (4NF) does not have non-trivial multivalued dependencies other than a candidate key.
- 4NF builds on the first three normal forms (1NF, 2NF and 3NF) and the BCNF.
- It does not contain more than one multivalued dependency.
- This normal form is rarely used outside of academic circles.

For example : A table contains a list of three things that is 'Student', 'Teacher', 'Book'. Teacher is in charge of Student and recommended book for each student. These three elements (Student, Teacher and Book) are independent of one another. Changing the student's recommended book, for instance, has no effect on the student itself. This is an example of multivalued dependency, where an item depends on more than one value. In this example, the student depends on both teacher and book.

- Therefore, 4NF states that a table should not have more than one dependencies.

6. Fifth Normal Form (5NF)

- 5NF is also knows as Project-Join Normal Form (PJ/NF).

- It is designed for reducing the redundancy in relational databases.
- 5NF requires semantically related multiple relationships, which are rare.
- In 5NF, if an attribute is multivalued attribute, then it must be taken out as a separate entity.
- While performing 5NF, the table must be in 4NF.

7. DKNF (Domain Key Normal Form)

- DKNF stands for Domain Key Normal Form requires the database that contains no constraints other than domain constraints and key constraints.
- In DKNF, it is easy to build a database.
- It avoids general constraints in the database which are not clear domain or key constraints.
- The 3NF, 4NF, 5NF and BCNF are special cases of the DKNF.
- It is achieved when every constraint on the relation is a logical consequence of the definition.

Transaction Processing

What is Transaction?

A set of logically related operations is known as transaction. The main operations of a transaction are:

Read(A): Read operations Read(A) or R(A) reads the value of A from the database and stores it in a buffer in main memory.

Write (A): Write operation Write(A) or W(A) writes the value back to the database from buffer.

Let us take a debit transaction from an account which consists of following operations:

R(A);
A=A-1000;
W(A);

Assume A's value before starting of transaction is 5000.

The first operation reads the value of A from database and stores it in a buffer.
Second operation will decrease its value by 1000. So buffer will contain 4000.
Third operation will write the value from buffer to database. So A's final value will be 4000.

But it may also be possible that transaction may fail after executing some of its operations. The failure can be because of **hardware, software or power** etc. For example, if debit transaction discussed above fails after executing operation 2, the value of A will remain 5000 in the database which is not acceptable by the bank. To avoid this, Database has two important operations:

Commit: After all instructions of a transaction are successfully executed, the changes made by transaction are made permanent in the database.

Rollback: If a transaction is not able to execute all operations successfully, all the changes made by transaction are undone.

Properties of a transaction

Atomicity: As a transaction is set of logically related operations, **either all of them should be executed or none**. A debit transaction discussed above should either execute all three operations or none.If debit transaction fails after executing operation 1 and 2 then its new value 4000 will not be updated in the database which leads to inconsistency.

Consistency: If operations of debit and credit transactions on same account are executed concurrently, it may leave database in an inconsistent state.
For Example, T1 (debit of Rs. 1000 from A) and T2 (credit of 500 to A) executing concurrently, the database reaches inconsistent state.

Let us assume Account balance of A is Rs. 5000. T1 reads A(5000) and stores the value in its local buffer space. Then T2 reads A(5000) and also stores the value in its local buffer space.

T1 performs A=A-1000 (5000-1000=4000) and 4000 is stored in T1 buffer space. Then T2 performs A=A+500 (5000+500=5500) and 5500 is stored in T2 buffer space. T1 writes the value from its buffer back to database.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV

A's value is updated to 4000 in database and then T2 writes the value from its buffer back to database. A's value is updated to 5500 which shows that the effect of debit transaction is lost and database has become inconsistent.

To maintain consistency of database, we need **concurrency control protocols** which will be discussed in next article. The operations of T1 and T2 with their buffers and database have been shown in Table 1.

T1	T1's buffer space	T2	T2's Buffer Space	Database
				A=5000
R(A);	A=5000			A=5000
	A=5000	R(A);	A=5000	A=5000
A=A-1000;	A=4000		A=5000	A=5000
	A=4000	A=A+500;	A=5500	
W(A);			A=5500	A=4000
		W(A);		A=5500

Table 1

Isolation: Result of a transaction should not be visible to others before transaction is committed. For example, Let us assume that A's balance is Rs. 5000 and T1 debits Rs. 1000 from A. A's new balance will be 4000. If T2 credits Rs. 500 to A's new balance, A will become 4500 and after this T1 fails. Then we have to rollback T2 as well because it is using value produced by T1. So a transaction results are not made visible to other transactions before it commits.

Durable: Once database has committed a transaction, the changes made by the transaction should be permanent. e.g.; If a person has credited $500000 to his account, bank can't say that the update has been lost. To avoid this problem, multiple copies of database are stored at different locations.

What is a Schedule?

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV

A schedule is series of operations from one or more transactions. A schedule can be of two types:

Serial Schedule: When one transaction completely executes before starting another transaction, the schedule is called serial schedule. A serial schedule is always consistent. e.g.; If a schedule S has debit transaction T1 and credit transaction T2, possible serial schedules are T1 followed by T2 (T1->T2) or T2 followed by T1 ((T1->T2). A serial schedule has low throughput and less resource utilization.

Concurrent Schedule: When operations of a transaction are interleaved with operations of other transactions of a schedule, the schedule is called Concurrent schedule. e.g.; Schedule of debit and credit transaction shown in Table 1 is concurrent in nature. But concurrency can lead to inconsistency in database. The above example of concurrent schedule is also inconsistent.

Question: Consider the following transaction involving two bank accounts x and y:

read(x);
x := x – 50;
write(x);
read(y);
y := y + 50;
write(y);

The constraint that the sum of the accounts x and y should remain constant is that of?
Atomicity
Consistency
Isolation
Durability

Solution: As discussed in properties of transactions, consistency properties says that sum of accounts x and y should remain constant before starting and after completion of transaction. So, the correct answer is B.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV

ACID Properties in DBMS

A transaction is a single logical unit of work which accesses and possibly modifies the contents of a database. Transactions access data using read and write operations. In order to maintain consistency in a database, before and after transaction, certain properties are followed. These are called **ACID** properties.

Atomicity

By this, we mean that either the entire transaction takes place at once or doesn't happen at all. There is no midway i.e. transactions do not occur partially. Each transaction is considered as one unit and either runs to completion or is not executed at all. It involves following two operations.

—**Abort**: If a transaction aborts, changes made to database are not visible.
—**Commit**: If a transaction commits, changes made are visible.

Atomicity is also known as the 'All or nothing rule'.

Consider the following transaction **T** consisting of **T1** and **T2**: Transfer of 100 from account **X** to account **Y**.

Before: X: 500		Y: 200
	Transaction T	
	T_1	T_2
	Read(X)	Read(Y)
	X:=X-100	Y:=Y+100
	Write(X)	Write(Y)
After : X: 400		Y:300

If the transaction fails after completion of **T1** but before completion of **T2**.(say, after **write(X)** but before **write(Y)**), then amount has been deducted from **X** but not added to **Y**. This results in an inconsistent database state. Therefore, the transaction must be executed in entirety in order to ensure correctness of database state.

Consistency

This means that integrity constraints must be maintained so that the database is consistent before and after the transaction. It refers to correctness of a database. Referring to the example above,

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV

The total amount before and after the transaction must be maintained.

Total **before T** occurs = **500 + 200 = 700**.
Total **after T occurs** = **400 + 300 = 700**.

Therefore, database is **consistent**. Inconsistency occurs in case **T1**completes but **T2** fails. As a result T is incomplete.

Isolation

This property ensures that multiple transactions can occur concurrently without leading to inconsistency of database state. Transactions occur independently without interference. Changes occurring in a particular transaction will not be visible to any other transaction until that particular change in that transaction is written to memory or has been committed. This property ensures that the execution of transactions concurrently will result in a state that is equivalent to a state achieved these were executed serially in some order.

Let **X**= 500, **Y** = 500.
Consider two transactions **T** and **T".**

T	T''
Read(X)	Read(X)
X:=X*100	Read(Y)
Write(X)	Z:=X + Y
Read(Y)	Write(Z)
Y:=Y-50	
Write(Y)	

Suppose **T** has been executed till **Read (Y)** and then **T''** starts. As a result , interleaving of operations takes place due to which **T''** reads correct value of **X** but incorrect value of **Y** and sum computed by

T'': (X+Y = 50, 000+500=50, 500)
is thus not consistent with the sum at end of transaction:

T: (X+Y = 50, 000 + 450 = 50, 450).
This results in database inconsistency, due to a loss of 50 units. Hence, transactions must take place in isolation and changes should be visible only after a they have been made to the main memory.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV

Durability:

This property ensures that once the transaction has completed execution, the updates and modifications to the database are stored in and written to disk and they persist even is system failure occurs. These updates now become permanent and are stored in a non-volatile memory. The effects of the transaction, thus, are never lost.

The **ACID** properties, in totality, provide a mechanism to ensure correctness and consistency of a database in a way such that each transaction is a group of operations that acts a single unit, produces consistent results, acts in isolation from other operations and updates that it makes are durably stored.

Concurrency Control

As we have seen above, when there are multiple transactions executing at the same time on same data, it may affect the result of the transaction. Hence it is necessary to maintain the order of execution of those transactions. In addition, it should not alter the ACID property of a transaction.

In order to maintain the concurrent access of transactions, two protocols are introduced.

- **Lock Based Protocol: -** Lock is in other words called as access. In this type of protocol any transaction will not be processed until the transaction gets the lock on the record. That means any transaction will not retrieve or insert or update or delete the data unless it gets the access to that particular data.

These locks are broadly classified as Binary locks and shared / exclusive locks.

In **binary lock** data can either be locked or unlocked. It will have only these two states. It can be locked for retrieve or insert or update or delete the data or unlocked for not using the data.

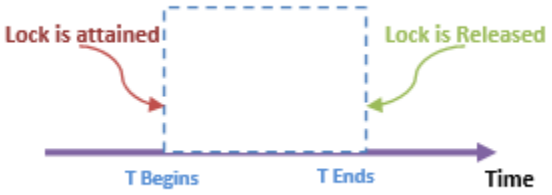
CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV

In **shared / exclusive lock** technique the data is said to be exclusively locked if for insert / update /delete. When it is exclusively locked no other transaction can read or write the data. When a data is read from the database, then its lock is shared i.e.; the data can be read by other transaction too but it cannot be changed while retrieving the data.

Lock based protocols are of 4 types

Simplistic Lock Protocol: -As the name suggests it is the simplest way of locking the data during the transaction. This protocol allows all the transaction to get the lock on the data before insert / update /delete on it. After completing the transaction, it will unlock the data.

Pre-claiming Protocol: - In this protocol, it evaluates the transaction to list all the data items on which transaction needs lock. It then requests DBMS for the lock on all those data items before the transaction begins. If DBMS gives the lock on all the data, then this protocol allows the transaction to begin. Once the transaction is complete, it releases all the locks. If all locks are given by DBMS, then it reverts the transactions and waits for the lock.

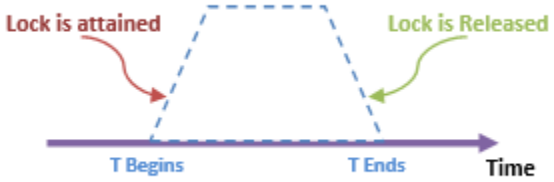


For example, if we have to calculate total marks of 3 subjects, then this protocol will evaluate the transaction and list the locks on subject1 marks, subject2 marks and then subject3 marks. Once it gets all the locks, it will start the transaction.

Two Phase Locking Protocol (2PL): -

In this type of protocol, as the transaction begins to execute, it starts requesting for the locks that it needs. It goes on requesting for the locks as and when it is needed. Hence it has a growing phase of locks. At one stage it will have all the locks. Once the transaction is complete it goes on releasing the locks. Hence it will have descending phase of locks. Thus this protocol has two phases – growing phase of locks and shrinking phase of locks.

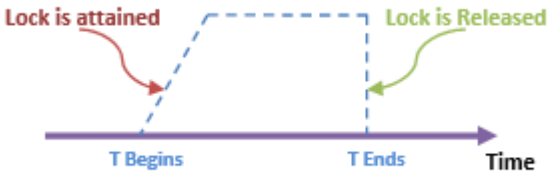
CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV



For example, if we have to calculate total marks of 3 subjects, then this protocol will go on asking for the locks on subject1 marks, subject2 marks and then subject3 marks. As and when it gets the locks on the subject marks it reads the marks. It does not wait till all the locks are received. Then it will have total calculation. Once it is complete it release the lock on subject3 marks, subject2 marks and subject1 marks.

In this protocol, if we need to have exclusive lock on any data for writing, then we have to first get the shared lock for reading. Then we have to request / modify the lock to exclusive lock.

Strict Two Phase Locking (Strict 2PL): - This protocol is similar to 2PL in the first phase. Once it receives the lock on the data, it completes the transaction. Here it does not release the locks as it is used and no more required. It waits till whole transaction to complete and commit, then it releases all the locks at a time. This protocol hence does not have shrinking phase of lock release.



In the example of calculating total marks of 3 subjects, locks are achieved at growing phase of the transaction and once it receives all the locks, it executes the transaction. Once the transaction is fully complete, it releases all the locks together.

- **Time Stamp Based Protocol: -** As we have seen above in lock based protocol, it acquires locks at the time of execution. But in this method, as soon as a transaction is created it assigns the order of the transaction. The order of the transaction is nothing but the ascending order of the transaction creation. The priority for older transaction is given to execute first. This protocol uses system time or logical counter to determine the time stamp of the transaction.

Suppose there are two transactions T1 and T2. Suppose T1 has entered the system at time 0005 and T2 has entered the system at 0008 clock time. Priority will be given to T1 to execute first as it is entered the system first.

In addition to the timestamp of a transaction, this protocol also maintains the timestamp of last 'read' and 'write' operation on a data. Based on the timestamp of transaction and the data which it is accessing a timestamp ordering protocol is defined.

According to this protocol:

```
IF TS (T) < W_TS(X) THEN
        Reject T
ELSE If TS (T)>= W_TS(X) Then
        Execute T
        Update W_TS(X) and R_TS(X)
END
```

Here TS (T) → Timestamp of transaction T
W_TS (X) → Timestamp of write operation on data X
R_TS (X) → Timestamp of read operation on data X

- If a transaction T is a read transaction on data X then

This algorithm states that if there is an active write operation on data X when a transaction T is requesting for X, then reject the transaction T. If the transaction T is started as soon as write is complete or no going write operation on X, then execute T.

For example, if there is an update on marks1 on MARKS table and meanwhile there is a request to read marks1, then do not perform read marks1. This is because there is an update being happening on marks1. If there was an update on marks1 which is executed long back or it is complete just now and there is a request to read marks1, then system will allow reading marks1.

- If a transaction T is a write transaction on data X then

```
IF TS (T) < R_TS (X) THEN
        Reject T
ELSE IF TS (T) < W_TS (X) THEN
        Reject T
        Rollback T
ELSE
        Execute T
END
```

Thomas' Write Rule

This algorithm describes about write operation. If there is an active read or write on data X, and at the same time if the transaction T is requesting for X, then the transaction is rejected. If there is no active read / write operation on X, then execute the transaction.

Suppose T1 is reading marks1 from MARKS table. Meanwhile transaction T2 begins and tries to update marks1 in MARKS. Then the transaction T2 is rejected and rolled back.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT IV

POSSIBLE QUESTIONS

2 MARK QUESTIONS

1. List the Properties of a transaction in DBMS.
2. Define concurrency control.
3. List the ACID properties in DBMS.
4. What is Transaction?
5. Write the disadvantages of Normalization.
6. Define Boyce code.
7. Define First Normal Forms
8. State 4NF
9. How to convert a table from 1NF to 2NF?
10. How to convert a table from 3NF to 4NF?

6 MARK QUESTIONS

1. Illustrate with syntax and example of first THREE Normal forms
2. List and explain ACID properties with example
3. State BCNF. How does it differ from 3NF?
4. Explain the time stamp based protocol for concurrency control in a DBMS..
5. What is transaction? Mention the desirable properties of a transaction.
6. Discuss any three Normal Forms with proper example.

KARPAGAM ACADEMY OF HIGHER EDUCATION
DEPARTMENT OF COMPUTER SCIENCE
II BSc CS BATCH(2016 - 2019)
PART-A OBJECTIVE TYPE/ MULTIPLE CHOICE QUESTIONS
UNIT-4

sno	Questions	opt1	opt2	opt3	opt4	Answer
1	____property enables us to recover any instance of the decomposed relation from corresponding instance of the smaller relations.	dependency preservation	lossless-join	normal forms	decomposition	lossless-join
2	BCNF stands for_____	Boyce Codd Normal Form	Boy Codd Normal Form	Basic Codd Normal Form	Basic Codd Normalization Form	Boyce-Codd Normal Form
3	Boye Codd Normal Form (BCNF) is needed when	two non-key attributes are dependent	there is more then one possible composite key	there are two or more possible composite overlapping keys and one attribute of a composite key is dependent on an attribute of another composite key	there are two possible keys and they are dependent on one another	there are two or more possible composite overlapping keys and one attribute of a composite key is dependent on an attribute of another composite key
4	A relation is said to be in BCNF when	it has overlapping composite keys	it has no composite keys	it has no multivalued dependencies	it has no overlapping composite keys which have related attributes	it has no overlapping composite keys which have related attributes
5	A 3 NF relation is converted to BCNF by	removing composite keys	removing multivalued dependencies	dependent attributes of overlapping composite keys are put in a separate relation	dependent non-key attributes are put in a separate table	dependent attributes of overlapping composite keys are put in a separate relation
6	BCNF is needed because	otherwise tuples may be duplicated	when a data is deleted tuples may be lost	updating is otherwise difficult	when there is dependent attributes in two possible composite keys one of the attributes is unnecessarily duplicated in the tuples	when there is dependent attributes in two possible composite keys one of the attributes is unnecessarily duplicated in the tuples
7	Fourth normal form (4 NF) relations are needed when	there are multivalued dependencies between attributes in composite key	there are more than one composite key	there are two or more overlapping composite keys	there are multivalued dependency between non-key attributes	there are multivalued dependencies between attributes in composite key
8	A 3 NF relation is split into 4 NF	by removing overlapping composite keys	by splitting into relations which do not have more than one independent multivalued dependency	removing multivalued dependency	by putting dependent non-key attribute in a separate table	by removing overlapping composite keys

9	A third Normal Form (3 NF) relation should	be in 2 NF	not have complete key	not be 1 NF	should not have non-key attributes depend on key attribute	be in 2 NF
10	The process of normalization	is automatic using a computer program	requires one to understand dependency between attributes	is manual and requires semantic information	is finding the key of a relation	requires one to understand dependency between attributes
11	A relation is said to be in 1NF if	there is no duplication of data	there are no composite attributes in the relation	there are only a few composite attributes	all attributes are of uniform type	there are no composite attributes in the relation
12	The number of normal forms which has been proposed and discussed in the book are	3	4	5	6	6
13	A relation which is in a higher normal form	implies that it also qualifies to be in lower normal form	does not necessarily satisfy the conditions of lower normal form	is included in the lower normal form	is independent of lower normal forms	implies that it also qualifies to be in lower normal form
14	Given an attribute x, another attribute y is dependent on it, if for a given x	there are many y values	there is only one value of y	there is one or more y values	there is none or one y value	there is only one value of y
15	Given the following relation vendor order (vendor no, order no, vendor name, qty supplied , price/unit) the second normal form relations are	vendor (vendor no, vendor name) qty (qty supplied, price/unit) order (order no, qty supplied)	vendor (vendor no, vendor name) order (order no, qty supplied, price/unit)	vendor (vendor no, vendor name) order (order no, qty supplied, price/unit) vendor order (vendor no, order no)	vendor (vendor no, vendor name, qty supplied, price/unit) vendor order (order no, vendor no)	vendor (vendor no, vendor name) order (order no, qty supplied, price/unit) vendor order (vendor no, order no)
16	__technique is used to reduce the redundancy	Closure set	Decompositio n	Normalization	Null Values	Normalization
17	3NF is also referred to as_____	LCNF	BCNF	information-preserving	desirable form	BCNF
18	Projection-join normal form also known as _____	1NF	2NF	3NF	5NF	5NF
19	if and only if the right-hand side is not a subset of the left-hand side, then functional dependency is said to be as _____	Non-trivial	Trivial	Transitive	Augmentation	Non-trivial
20	A relation is in _____ if and only if the nonkey attributes are mutually independent	3NF	1NF	2NF	5NF	3NF
21	_____attribute does not participate in the primary key of the relation concerned	key attribute	Non-key attribute	Variable	none	Non-key attribute
22	A relation is in _____if and only if , in every legal value of that relation, every tuple contains exactly one value for each attribute.	2NF	3NF	1NF	4NF	1NF
23	_technique is used to eliminate the redundancy	Null Values	Decompositio n	Normalizations	Concatenation	Normalizations
24	A relation is in _____ if and only if it is in 1NF and every nonkey attribute is irreducibly dependent on the primary key.	1NF	2NF	3NF	4NF	2NF

25	A relation is in _____ if and only if it is in 2NF and every nonkey attribute is non transitively dependent on the primary key.	1NF	2NF	3NF	4NF	3NF
26	A relation is _____ if and only if the only determinants are candidate keys.	1NF	2NF	3NF	4NF	2NF
27	In the functional dependency, left-hand side indicates_____	determinants	Dependencies	trivial	non-trivial	determinants
28	In the functional dependency, right-hand side indicates_____	determinants	Dependencies	trivial	non-trivial	Dependencies
29	_____property anables us to enforce any constraint on the original relation by simply enforcing some constraints on each of the smaller relations.	dependency preservation	lossless-join	normal forms	decomposition	dependency preservation
30	_____ package constructs are declared in the package specifiaction and defined in package body.	public	private	internal	external	public
31	_____ package constructs are not declared in the package specifiaction but defined in package body.	public	private	internal	external	private
32	Global package item are declared in package ____ for external users to use it.	defintion	specification	body	declaration	specification
33	Variables declared in package specification are intialised to ___ by default	zero	space	null	one	null
34	which command is used to invoke a procedure in a package	call	execute	invoke	/	execute
35	____is the process of taking a normalized database and modifying table structures to allow controlled redundancy for increased database performance.	Denormalizatio n	normalization	functional dependency	Decomposition	Denormalization
36	in select statement Duplicates are eliminated by using _____ keyword.	Remove	Distinct	RM	Redundant	Distinct
37	Group function is otherwise known as_____	Collection	Aggregate	function	Count	Aggregate
38	string Pattern matching is done through_____operator.	Comparison	arithmetic	Logical	like	like
39	Which keyword is used to check if an element is in a given set?	not	in	not exist	except	in
40	Any two tables that are Union-Compatible if they have the same number of _____	Columns	rows	Columns and rows	null values	Columns
41	Which clause is used, when we are using Group by clause, instead of where clause?	where	Having	Distinct	Group	Having
42	_____clause is used to group the values under particular characteristics	Order By	Group by	Count	Sum	Order By
43	which of the following is not a set operator	union	minus	intersect	plus	plus
44	check constraint is ___ level constraint	table	column	row	database	column
45	a view is updatable when it is based upon ____ tables	two	one	three	any numberof	one
46	____ clause is used in to make a sequence repeat the series.	cycle	nocycle	cache	start with	cycle

47	what is the maximum value for a descending sequence	-1	0	null	10^27	-1
48	___ is used and maintained automatically by oracle server	view	table	index	sequence	index
49	The procedure has only ____ priveledge	alter	delete	execute	select	execute
50	……. provides option for entering SQL queries as execution time, rather than at the development stage.	PL/SQL	SQL*Plus	SQL	Dynamic SQL	Dynamic SQL
51	The RDBMS terminology for a row is	tuple	relation	attribute	degree	tuple
52	To change column value in a table the …… command can be used	create	insert	alter	update	update
53	A set of possible data values is called	attribute	degree	tuple	domain	domain
54	…… is critical in formulating database design.	row column order	number of tables	functional dependency	normalizing	functional dependency
55	A ……. represents the number of entities to which another entity can be associated	mapping cardinality	table	schema	information	mapping cardinality
56	Which two files are used during operation of the DBMS	Query languages and utilities	DML and query language	Data dictionary and transaction log	Data dictionary and query language	Data dictionary and transaction log
57	A …….. is a set of column that identifies every row in a table	composite key	candidate key	foreign key	super key	super key
58	The relational model is based on the concept that data is organized and stored in two-dimensional tables called ……	Fields	Records	Relations	Keys	Relations
59	…….. contains information that defines valid values that are stored in a column or data type.	View	Rule	Index	Default	Index
60	insert into <table_name> values <list of values>	TRUE	FALSE	Table	None	TRUE

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V

SYLLABUS

File Structure and Indexing (8 Lectures) Operations on files, File of Unordered and ordered records, overview of File organizations, Indexing structures for files(Primary index, secondary index, clustering index), Multilevel indexing using B and B+ trees.

File Structure and Indexing

overview of File organizations

A database consist of a huge amount of data. The data is grouped within a table in RDBMS, and each table have related records. A user can see that the data is stored in form of tables, but in acutal this huge amount of data is stored in physical memory in form of files.

File – A file is named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tables and optical disks.

What is File Organization?

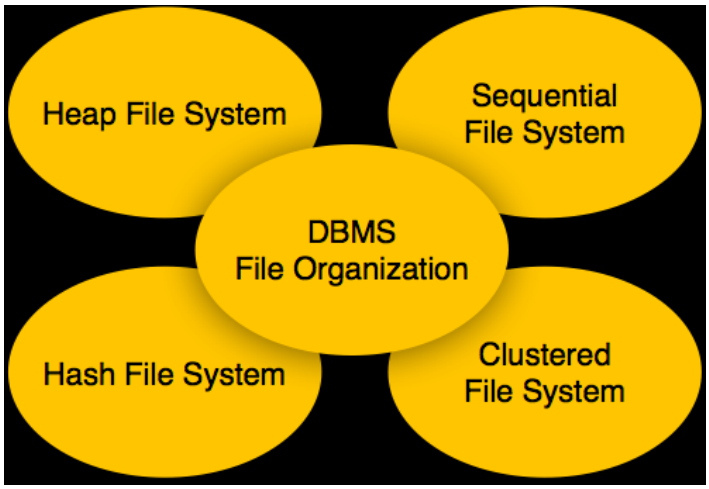
File Organization refers to the logical relationships among various records that constitute the file, particularly with respect to the means of identification and access to any specific record. In simple terms, Storing the files in certain order is called file Organization. **File Structure** refers to the format of the label and data blocks and of any logical control record.

File organization: is a method of arranging the records in a file when the file is stored on disk. A relation is typically stored as a file of records

Types of File Organizations :

Various methods have been introduced to Organize files. These particular methods have advantages and disadvantages on the basis of access or selection . Thus it is all upon the programmer to decide the best suited file Organization method according to his requirements.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V



Some types of File Organizations are :

- Sequential File Organization
- Heap File Organization
- Hash File Organization
- B+ Tree File Organization
- Clustered File Organization

Heap File Organization

When a file is created using Heap File Organization, the Operating System allocates memory area to that file without any further accounting details. File records can be placed anywhere in that memory area. It is the responsibility of the software to manage the records. Heap File does not support any ordering, sequencing, or indexing on its own.

Sequential File Organization

Every file record contains a data field (attribute) to uniquely identify that record. In sequential file organization, records are placed in the file in some sequential order based on the unique key field or search key. Practically, it is not possible to store all the records sequentially in physical form.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V

Hash File Organization

Hash File Organization uses Hash function computation on some fields of the records. The output of the hash function determines the location of disk block where the records are to be placed.

Clustered File Organization

Clustered file organization is not considered good for large databases. In this mechanism, related records from one or more relations are kept in the same disk block, that is, the ordering of records is not based on primary key or search key.

File Operations

Operations on database files can be broadly classified into two categories −

- **Update Operations**
- **Retrieval Operations**

Update operations change the data values by insertion, deletion, or update. Retrieval operations, on the other hand, do not alter the data but retrieve them after optional conditional filtering. In both types of operations, selection plays a significant role. Other than creation and deletion of a file, there could be several operations, which can be done on files.

- **Open** − A file can be opened in one of the two modes,**read mode** or **write mode**. In read mode, the operating system does not allow anyone to alter data. In other words, data is read only. Files opened in read mode can be shared among several entities. Write mode allows data modification. Files opened in write mode can be read but cannot be shared.
- **Locate** − Every file has a file pointer, which tells the current position where the data is to be read or written. This pointer can be adjusted accordingly. Using find (seek) operation, it can be moved forward or backward.
- **Read** − By default, when files are opened in read mode, the file pointer points to the beginning of the file. There are options where the user can tell the

operating system where to locate the file pointer at the time of opening a file. The very next data to the file pointer is read.

- **Write** − User can select to open a file in write mode, which enables them to edit its contents. It can be deletion, insertion, or modification. The file pointer can be located at the time of opening or can be dynamically changed if the operating system allows to do so.
- **Close** − This is the most important operation from the operating system's point of view. When a request to close a file is generated, the operating system
  - removes all the locks (if in shared mode),
  - saves the data (if altered) to the secondary storage media, and
  - releases all the buffers and file handlers associated with the file.

The organization of data inside a file plays a major role here. The process to locate the file pointer to a desired record inside a file various based on whether the records are arranged sequentially or clustered.

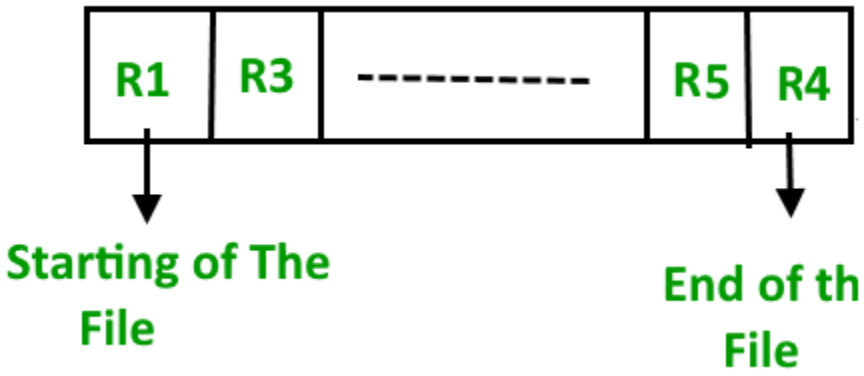
File of Unordered and ordered records

Sequential File Organization (Ordered File)

The easiest method for file Organization is Sequential method. In this method the the file are stored one after another in a sequential manner. There are two ways to implement this method:

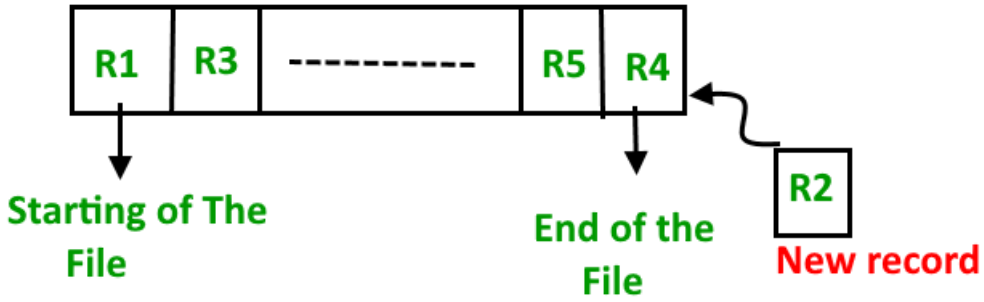
1. **Pile File Method –** This method is quite simple, in which we store the records in a sequence i.e one after other in the order in which they are inserted into the tables.

CLASS : II.B.Sc CS BATCH : 2016 - 2019
COURSE NAME : DTATBASE MANAGEMENT SYSTEM COURSE CODE : 16CSU403
UNIT V

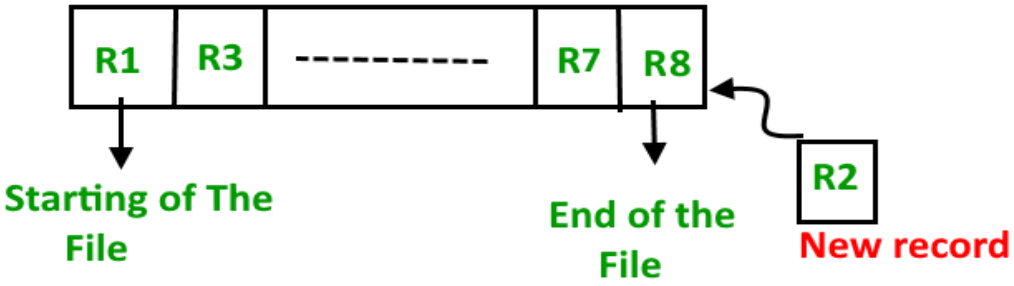


Insertion of new record

Let the R1, R3 and so on upto R5 and R4 be four records in the sequence. Here, records are nothing but a row in any table. Suppose a new record R2 has to be inserted in the sequence, then it is simply placed at the end of the file.



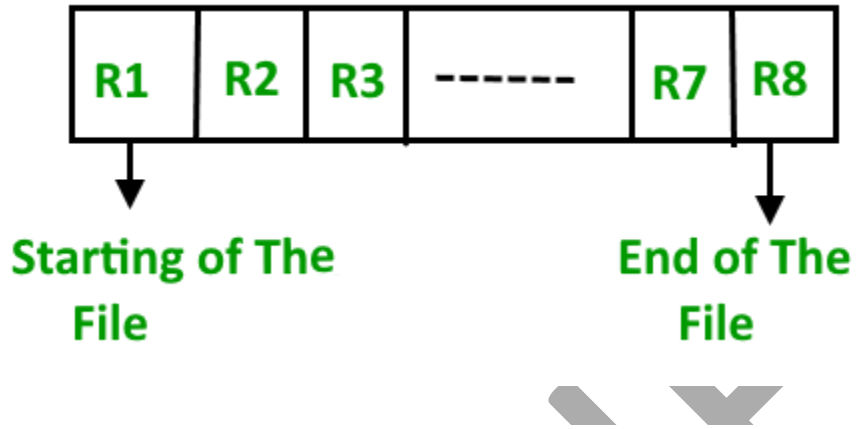
2. **Sorted File Method –**In this method, As the name itself suggest whenever a new record has to be inserted, it is always inserted in a sorted (ascending or descending) manner. Sorting of records may be based on any primary key or any other key.



CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V

Insertion of new record –

Let us assume that there is a preexisting sorted sequence of four records R1, R3, and so on upto R7 and R8. Suppose a new record R2 has to be inserted in the sequence, then it will be inserted at the end of the file and then it will sort the sequence .



Pros and Cons of Sequential File Organization –

Pros –

- Fast and efficient method for huge amount of data.
- Simple design.
- Files can be easily stored in magnetic tapes i.e cheaper storage mechanism.

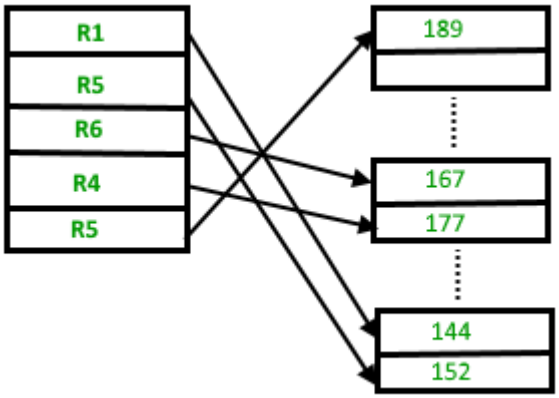
Cons –

- Time wastage as we cannot jump on a particular record that is required, but we have to move in a sequential manner which takes our time.
- Sorted file method is inefficient as it takes time and space for sorting records.

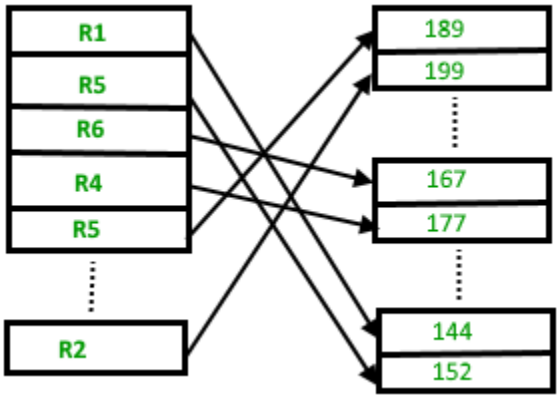
Heap File Organization (Unordered File)

Heap File Organization works with data blocks. In this method records are inserted at the end of the file, into the data blocks. No Sorting or Ordering is required in this method. If a data block is full, the new record is stored in some other block, Here the other data block need not be the very next data block, but it can be any block in the memory. It is the responsibility of DBMS to store and manage the new records.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V



Insertion of new record –
Suppose we have four records in the heap R1, R5, R6, R4 and R3 and suppose a new record R2 has to be inserted in the heap then, since the last data block i.e data block 3 is full it will be inserted in any of the database selected by the DBMS, lets say data block 1.



If we want to search, delete or update data in heap file Organization the we will traverse the data from the beginning of the file till we get the requested record. Thus if the database is very huge, searching, deleting or updating the record will take a lot of time.

Pros and Cons of Heap File Organization –
Pros –

- Fetching and retrieving records is faster than sequential record but only in case of small databases.
- When there is a huge number of data needs to be loaded into the database at a time, then this method of file Organization is best suited.

Cons –

- Problem of unused memory blocks.
- Inefficient for larger databases.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V

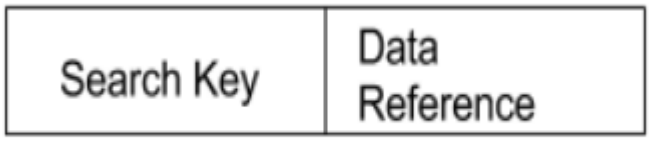
Indexing in Databases

Indexing is a way to optimize performance of a database by minimizing the number of disk accesses required when a query is processed.

An index or database index is a data structure which is used to quickly locate and access the data in a database table.

Indexes are created using some database columns.

- The first column is the Search key that contains a copy of the primary key or candidate key of the table. These values are stored in sorted order so that the corresponding data can be accessed quickly (Note that the data may or may not be stored in sorted order).
- The second column is the Data Reference which contains a set of pointers holding the address of the disk block where that particular key value can be found.
-



Structure of an index

There are two kinds of indices:

1. **Ordered indices:** Indices are based on a sorted ordering of the values.
2. **Hash indices:** Indices are based on the values being distributed uniformly across a range of buckets. The buckets to which a value is assigned is determined by function called a hash function.

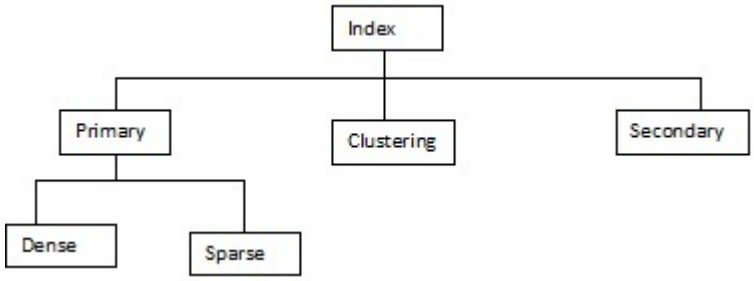
There is no comparison between both the techniques, it depends on the database application on which it is being applied.

- **Access Types**: e.g. value based search, range access, etc.
- **Access Time**: Time to find particular data element or set of elements.
- **Insertion Time**: Time taken to find the appropriate space and insert a new data time.
- **Deletion Time**: Time taken to find an item and delete it as well as update the index structure.
- **Space Overhead**: Additional space required by the index.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V

Indexing Methods

Types of Index



Ordered Indices

The indices are usually sorted so that the searching is faster. The indices which are sorted are known as ordered indices.

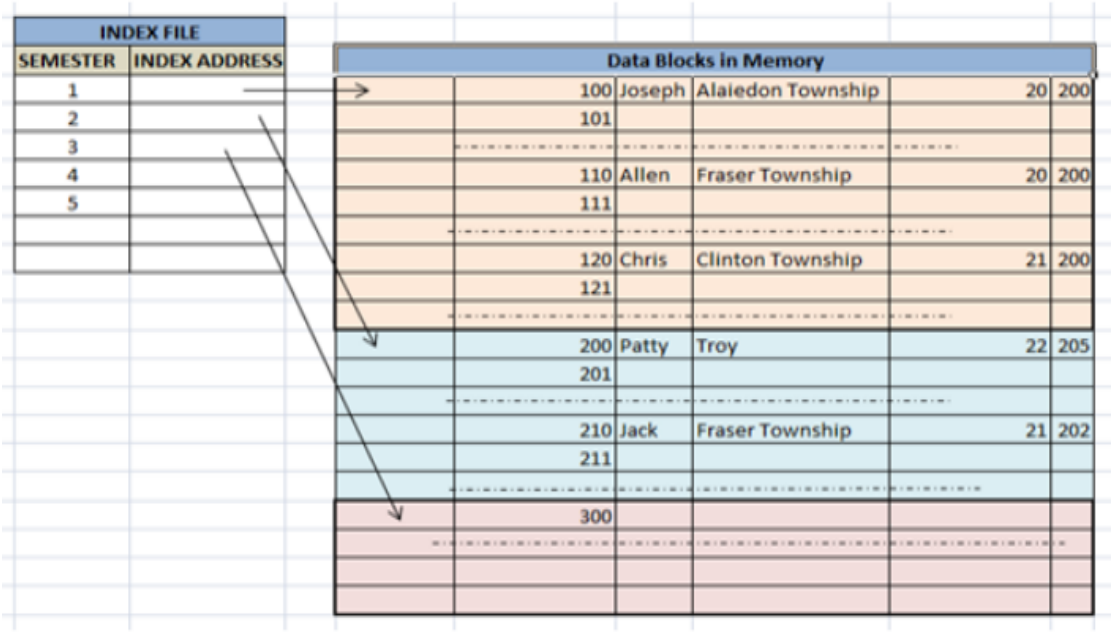
- If the search key of any index specifies same order as the sequential order of the file, it is known as primary index or clustering index.
  Note: The search key of a primary index is usually the primary key, but it is not necessarily so.
- If the search key of any index specifies an order different from the sequential order of the file, it is called the secondary index or non-clustering index.

Clustered Indexing

Clustering index is defined on an ordered data file. The data file is ordered on a non-key field. In some cases, the index is created on non-primary key columns which may not be unique for each record. In such cases, in order to identify the records faster, we will group two or more columns together to get the unique values and create index out of them. This method is known as clustering index. Basically, records with similar characteristics are grouped together and indexes are created for these groups.

For example, students studying in each semester are grouped together. i.e. 1st Semester students, 2nd semester students, 3rd semester students etc are grouped.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V



Clustered index sorted according to first name (Search key)

Primary Index

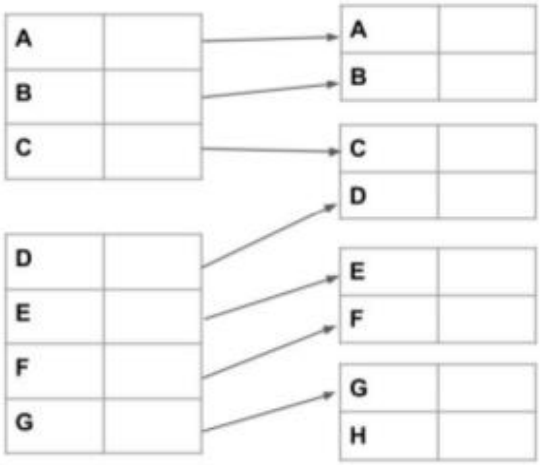
In this case, the data is sorted according to the search key. It induces sequential file organisation.

In this case, the primary key of the database table is used to create the index. As primary keys are unique and are stored in sorted manner, the performance of searching operation is quite efficient. The primary index is classified into two types
Dense Index and **Sparse Index**.

(I) Dense Index :

- For every search key value in the data file, there is an index record.
- This record contains the search key and also a reference to the first data record with that search key value.

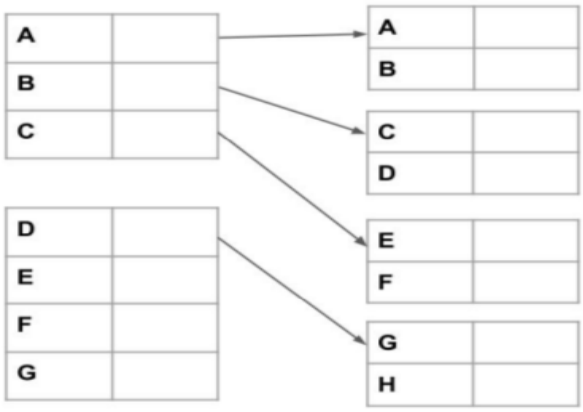
CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V



Dense Index

(II) Sparse Index :

- The index record appears only for a few items in the data file. Each item points to a block as shown.
- To locate a record, we find the index record with the largest search key value less than or equal to the search key value we are looking for.
- We start at that record pointed to by the index record, and proceed along the pointers in the file (that is, sequentially) until we find the desired record.

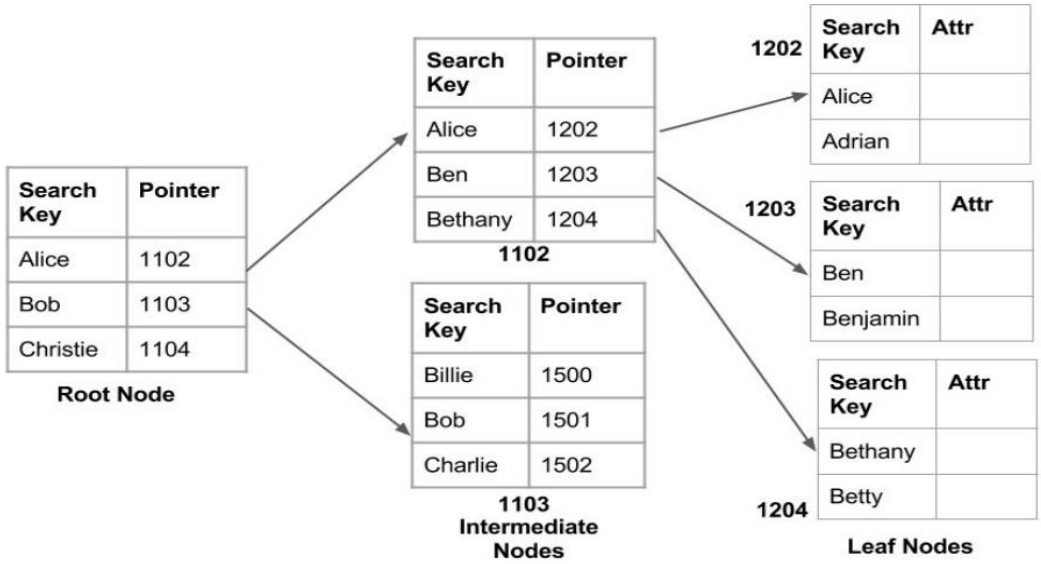


Sparse Index

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V

Non-Clustered Indexing

A non clustered index just tells us where the data lies, i.e. it gives us a list of virtual pointers or references to the location where the data is actually stored. Data is not physically stored in the order of the index. Instead , data is present in leaf nodes. For eg. the contents page of a book. Each entry gives us the page number or location of the information stored. The actual data here(information on each page of book) is not organised but we have an ordered reference(contents page) to where the data points actually lie.



Non clustered index

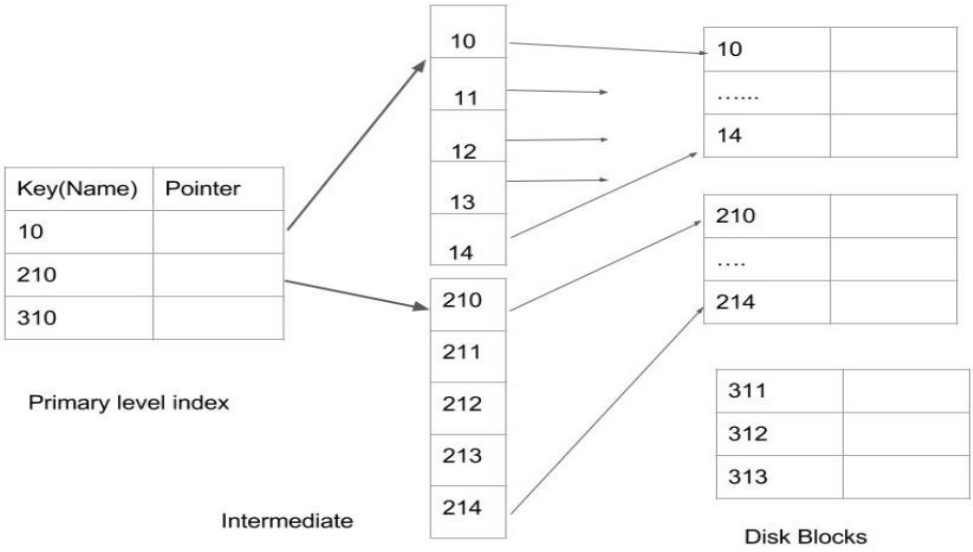
It requires more time as compared to clustered index because some amount of extra work is done in order to extract the data by further following the pointer. In case of clustered index, data is directly present in front of the index.

Secondary Index

It is used to optimize query processing and access records in a database with some information other than the usual search key (primary key). In this two levels of indexing are used in order to reduce the mapping size of the first level and in general. Initially, for the first level, a large range of numbers is selected so that the mapping size is small. Further, each range is divided into further sub ranges.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V

In order for quick memory access, first level is stored in the primary memory. Actual physical location of the data is determined by the second mapping level.



Secondary level Indexing

Index in a Tree like Structure

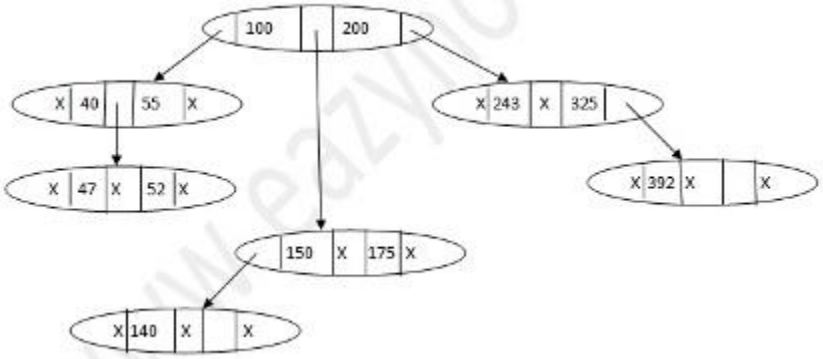
We can use tree-like structures as index as well. For example, a binary search tree can also be used as an index. If we want to find out a particular record from a binary search tree, we have the added advantage of binary search procedure, that makes searching be performed even faster. A binary tree can be considered as a **2-way Search Tree**, because it has two pointers in each of its nodes, thereby it can guide you to two distinct ways. Remember that for every node storing 2 pointers, the number of value to be stored in each node is one less than the number of pointers, i.e. each node would contain 1 value each.

M-Way Search Tree

The abovementioned concept can be further expanded with the notion of the m-Way Search Tree, where m represents the number of pointers in a particular node. If m = 3, then each node of the search tree contains 3 pointers, and each node would then contain 2 values.

A sample m-Way Search Tree with m = 3 is given in the following.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V



B+ Tree

A B^+ tree is a balanced binary search tree that follows a multi-level index format. The leaf nodes of a B^+ tree denote actual data pointers. B^+ tree ensures that all leaf nodes remain at the same height, thus balanced. Additionally, the leaf nodes are linked using a link list; therefore, a B^+ tree can support random access as well as sequential access.

Structure of B^+ Tree

Every leaf node is at equal distance from the root node. A B^+ tree is of the order **n** where **n** is fixed for every B^+ tree.

Internal nodes −

- Internal (non-leaf) nodes contain at least ⌈n/2⌉ pointers, except the root node.
- At most, an internal node can contain **n** pointers.

Leaf nodes −

- Leaf nodes contain at least ⌈n/2⌉ record pointers and ⌈n/2⌉ key values.
- At most, a leaf node can contain **n** record pointers and **n** key values.
- Every leaf node contains one block pointer **P** to point to next leaf node and forms a linked list.

B^+ Tree Insertion

- B^+ trees are filled from bottom and each entry is done at the leaf node.
- If a leaf node overflows −

- Split node into two parts.
- Partition at **i = ⌊(m+1)/2⌋.**
- First **i** entries are stored in one node.
- Rest of the entries (i+1 onwards) are moved to a new node.
- **i^{th}** key is duplicated at the parent of the leaf.

- If a non-leaf node overflows −
  - Split node into two parts.
  - Partition the node at **i = ⌈(m+1)/2⌉.**
  - Entries up to **i** are kept in one node.
  - Rest of the entries are moved to a new node.

B^+ Tree Deletion

- B^+ tree entries are deleted at the leaf nodes.
- The target entry is searched and deleted.
  - If it is an internal node, delete and replace with the entry from the left position.
- After deletion, underflow is tested,
  - If underflow occurs, distribute the entries from the nodes left to it.
- If distribution is not possible from left, then
  - Distribute from the nodes right to it.
- If distribution is not possible from left or from right, then
  - Merge the node with left and right to it.

CLASS : II.B.Sc CS **BATCH : 2016 - 2019**
COURSE NAME : DTATBASE MANAGEMENT SYSTEM **COURSE CODE : 16CSU403**
UNIT V

POSSIBLE QUESTIONS

2 MARK QUESTIONS

1. Define Clustered Indexing with example
2. Define Clustered Indexing with example.
3. What is Primary Index.Give Example?
4. What is Secondary Index?
5. List the Properties of B-Tree.
6. List the Properties of B+-Tree.

6 MARK QUESTIONS

1. Discuss Indexing structures with neat sketch.
2. Mention various types of records. Describe how they are organized inside a file?
3. Explain with example i) Primary index ii) clustering index
4. Discuss File Structure in DBMS with example.
5. Discuss in detail about cluster and Multilevel indexes.
6. What is an index? Explain its role in improving database access.
7. Explain in detail about B+ trees.

KARPAGAM ACADEMY OF HIGHER EDUCATION
DEPARTMENT OF COMPUTER SCIENCE
II BSc CS BATCH(2016 - 2019)
PART-A OBJECTIVE TYPE/ MULTIPLE CHOICE QUESTIONS
UNIT-5

sno	Questions	opt1	opt2	opt3	opt4	Answer
1	Index which has an entry for some of key value is classified as	linear index	dense index	non dense index	cluster index	non dense index
2	Primary indexes, secondary indexes and cluster indexes are all types of	ordered indexes	unordered indexes	linear indexes	relative search indexes	ordered indexes
3	In multilevel indexes, primary index created for its first level is classified as	zero level of multilevel index	third level of multilevel index	second level of multilevel index	first level of multilevel index	second level of multilevel index
4	Indexes which specifies address of records on disk with a physical pointer are classified as	structural index	hashing index	physical index	logical index	physical index
5	Example of non dense index is	ternary index	secondary index	primary index	clustering index	clustering index
6	The method of access which uses key transformation is known as	direct	hash	random	sequential	hash
7	The physical location of a record is determined by a mathematical formula that transforms	B-Tree File	Hashed File	Indexed File	Sequential file	Hashed File
8	What is the purpose of index in sql server	To enhance the query performance	To provide an index to a record	To perform fast searches	All of the mentioned	All of the mentioned
9	How many types of indexes are there in sql server?	1	2	3	4	2
10	How non clustered index point to the data?	It never points to anything	It points to a data row	It is used for pointing data rows containing key values	None of the mentioned	It is used for pointing data rows containing key values
11	Which one is true about clustered index?	Clustered index is not associated with table	Clustered index is built by default on unique key columns	Clustered index is not built on unique key columns	None of the mentioned	is built by default on unique key columns
12	What is true about indexes?	Indexes enhance the performance even if the table is updated frequently	It makes harder for sql server engines to work to work on index which have large keys	It doesn't make harder for sql server engines to work to work on index which have large keys	None of the mentioned	It makes harder for sql server engines to work to work on index which have large keys
13	Does index take space in the disk ?	It stores memory as and when required	Yes, Indexes are stored on disk	Indexes are never stored on disk	Indexes take no space	Yes, Indexes are stored on disk
14	What are composite indexes ?	Are those which are composed by database for its internal use	A composite index is a combination of index on 2 or more columns	Composite index can never be created	None of the mentioned	A composite index is a combination of index on 2 or more columns
15	In _____ index instead of storing all the columns for a record together, each column is stored separately with all other rows in an index.	Clustered	Non clustered	Column store	Row store	Column store
16	A _____ index is the one which satisfies all the columns requested in the query without performing further lookup into the clustered index.	Clustered	Non Clustered	Covering	B-Tree	Covering
17	A(n) _____ can be used to preserve the integrity of a document or a message.	Message digest	Message summary	Encrypted message	None of the mentioned	Encrypted message
18	A hash function must meet _____ criteria.	Two	Three	Four	Five	Three
19	Index which has an entry for some of key value is classified as	linear index	dense index	non dense index	cluster index	non dense index
20	In data file, first record of any of block is called	anchor record	dense record	non dense record	none of above	anchor record
21	File which has secondary index for its every field is classified as	fully inverted file	fully indexed file	secondary indexed file	primary indexed file	fully inverted file
22	First field in primary index having same data type as in ordering field is considered as	indexed key	ternary key	secondary key	primary key	primary key
23	In multilevel indexes, primary index created for its second level is classified as	second level of multilevel index	first level of multilevel index	zero level of multilevel index	third level of multilevel index	third level of multilevel index
24	The SQL database language includes statements for:	Database definition.	Database manipulation.	Database control.	All of the above.	All of the above.
25	A command to remove a relation from an SQL database	Delete table table name	Drop table table name	Erase table table name	Alter table table name	Drop table table name
26	Which SQL Query is use to remove a table and all its data from the database?	Create Table	Alter Table	Drop Table	None of these	Drop Table

27	A type of query that is placed within a WHERE or HAVING clause of another query is called	Super query	Sub query	Master query	Multi-query	Sub query
28	Aggregate functions are functions that take a _____ as input and return a single value.	Collection of values	Single value	Aggregate value	Both a & b	Collection of values
29	Which of the following should be used to find the mean of the salary ?	Mean(salary)	Avg(salary)	Sum(salary)	Count(salary)	Avg(salary)
30	All aggregate functions except _____ ignore null values in their input collection.	Count(attribute)	Count(*)	Avg	Sum	Count(*)
31	A Boolean data type that can take values true, false, and_____ .	1	0	Null	Unknown	Unknown
32	If we do want to eliminate duplicates, we use the keyword _____in the aggregate expression.	Distinct	Count	Avg	Primary key	Distinct
33	The ____ connective tests for set membership, where the set is a collection of values produced by a select clause.	Or	Not in	In	and	In
34	The ____ connective tests for the absence of set membership.	Or	Not in	In	and	Not in
35	We can test for the nonexistence of tuples in a subquery by using the _____ construct.	Not exist	Not exists	Exists	Exist	Not exists
36	Dates must be specified in the format	mm/dd/yy	yyyy/mm/dd	dd/mm/yy	yy/dd/mm	yyyy/mm/dd
37	Which of the following is used to store movie and image files ?	Clob	Blob	Dlob	None of the above	Blob
38	Entities are identified from the word statement of a problem by	picking words which are adjectives	picking words which are nouns	picking words which are verbs	picking words which are pronouns	picking words which are nouns
39	Relationships are identified from the word statement of a problem by	picking words which are adjectives	picking words which are nouns	picking words which are verbs	picking words which are pronouns	picking words which are pronouns
40	One entity may be	related to only one other entity	related to itself	related to only two other entities	related to many other entities	related to many other entities
41	By relation cardinality we mean	number of items in a relationship	number of relationships in which an entity can appear	number of items in an entity	number of related occurrences for each of the two entities	number of related occurrences for each of the two entities
42	If an entity appears in only one relationship then it is	a 1:1 relationship	a 1:N relationship	a N:1 relationship	a N:M relationship	a 1:1 relationship
43	If an entity appears in N relationships then it is	a 1:1 relationship	a 1:N relationship	a N:1 relationship	a N:M relationship	a N:M relationship
44	If an entity appears in not more than 5 relationships then it is a	1:1 relationship	1:5 relationship	5:1 relationship	5:5 relationship	1:5 relationship
45	A relation is	an entity	a relationship	members of a relationship set	members of an entity set or a relationship set	an entity
46	Rows of a relation are called	tuples	a relation row	a data structure	an entity	tuples
47	The database schema is written in	HLL	DML	DDL	DCL	DDL
48	The way a particular application views the data from the database that the application uses is a	module	relational model	schema	sub schema	sub schema
49	The relational model feature is that there	is no need for primary key data	is much more data independence than some other database models	are explicit relationships among records.	are tables with many dimensions	is much more data independence than some other database models
50	Which of the following are the properties of entities?	Groups	Table	Attributes	Switchboards	Attributes
51	Which database level is closest to the users?	External	Internal	Physical	Conceptual	External
52	Which are the two ways in which entities can participate in a relationship?	Passive and active	Total and partial	Simple and Complex	All of the above	Total and partial
53	…….. data type can store unstructured data	RAW	CHAR	NUMERIC	VARCHAR	RAW
54	insert into <table_name> (column list) values <list of values>	TRUE	FALSE	Table	None	TRUE
55	……. first proposed the process of normalization in DBMS.	Edgar. W	Edgar F. Codd	Edward Stephen	Edward Codd	Edgar F. Codd
56	Which of the following is not comparison operator?	<>	<	=<	>=	=<
57	An outstanding functionality of SQL is its support for automatic …… to the target data.	programming	functioning	navigation	notification	notification
58	……..specifies a search condition for a group or an aggregate	GROUP BY Clause	HAVING Clause	FROM Clause	WHERE Clause	HAVING Clause
59	Drop Table cannot be used to drop a table referenced by a …… constraint.	Local Key	Primary Key	Composite Key	Foreign Key	Foreign Key
60	The user defined data type can be created using	Create datatype	Create data	Create definetype	Create type	Create type

Reg .No…….

[16CSU403]

KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed to be University)

(Established Under Section 3 of UGC Act, 1956)

COIMBATORE – 641 021

(For the candidates admitted from 2016 onwards)

B.Sc DEGREE EXAMINATION

Fourth Semester

First Internal Examination – January 2018

COMPUTER SCIENCE

DATA BASE MANAGEMENT SYSTEMS

Date :19 – 01 – 18 **Time** : 2:00 Hrs

Class : II.B.Sc CS (A & B) **Maximum** : 50

SECTION A – (20*1 = 20 Marks)
Answer All Questions

1. _____is a collection of high-level data description constructs that hide many low-level storage details

a. Data Model b. ER Model c. Network Model d. Relational Model

2. Database management System based on _____

a. Data Model b. ER Model c. Network Model **d. Relational Model**

3. A widely used Semantic model called _____

a. Data Model **b. ER Model** c. Hierarchical model d. Object-based model

4. _____is a more abstract

a. ER model b. **Semantic data model**

c. Conceptual data Mode networks d. Physical data model

5. _____model is used to pictorially denote entities & relationships

a. **ER model** b. Semantic data model

c. Conceptual data Mode networks d. Physical data model

6. A description Of data in terms of a data model is called_____

a. Schema b. Relation c. record d. entities

7. Field is otherwise known as _____

a. Column b. Entity c. Relation d. Relationship

8. Column is otherwise known as _____

a. Relation b. Entity **c. attribute** d. Relationship

9. ____is used to define the external and conceptual model.

a. DDL b. DML c. DCL d. TCL

10. Physical model specifies _____details.

a. Information b. data **c. Storage** d. relationships

11. The _____layer is responsible for process to process delivery.

a. physical b. presentation c. networks d. transport

12. ____ is the primary unit of storage in a database

a. table b. column c. row d. number

13. Set structures can represent a __ relationship between tables.

a. one-to-one **b. one-to-many** c. many-to-many d. many-to-one

14. An E-R modeling for given application leads to

a. conceptual data mode b. logical data model

c external data model d. internal data model

15. An entity is _____.

a. collection of different entities b. collection of related entities

c. related data d. **collection of similar entities**

16. What does SQL stand for?

a. Structured Query Language b. Strong Question Language

c. Structured Question Language d. Structure Query Language

17 Create, Alter, Drop commands are_____ language command..

a. DDL **b. DML** c. DCL d. TCL

18. Create.Alter,Drop commands are_____ language command

a. DDL **b. DML** c. DCL d. TCL

19. Selection Operation is used to ____from a relation

a. Select the Column **b. Select the row** c. Select the table d. Select DMBS

20. _____command is used to remove the table definition information

a. Delete b. Remove c. Destroy **d. Drop**

SECTION B – (3*2 = 6 Marks)

Answer All Questions

21. List the Characteristics of Database Approach

22. Define entity and attributes

23. Draw an ER diagram for M:M relationship

SECTION C – (3*8 = 24 Marks)

Answer the Questions

24. a) Define database and explain various database models with neat diagram.

[OR]

b) Discuss data base architecture with neat sketch.

25. a) What are the typical ERD symbols and draw a neat ERD for an employee table.

[OR]

b) Explain i) Entity types ii) Relationships

26. a) IIustrate with syntax and example of relational constraints.

[OR]

b) Discuss relational algebra with proper example.

Reg .No…….

[16CSU403]

KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed to be University

(Established Under Section 3 of UGC Act, 1956)

COIMBATORE – 641 021

(For the candidates admitted from 2016 onwards)

B.Sc DEGREE EXAMINATION

Fourth Semester

First Internal Examination – January 2018

COMPUTER SCIENCE

DATA BASE MANAGEMENT SYSTEMS

Date : 19 – 01 – 18 **Time** : 2:00 Hrs

Class : II.B.Sc CS (A & B) **Maximum** : 50

SECTION A – (20*1 = 20 Marks)
Answer All Questions

1. _____is a collection of high-level data description constructs that hide many low-level storage details

a. Data Model b. ER Model c. Network Model d. Relational Model

2. Database management System based on _____

a. Data Model b. ER Model c. Network Model **d. Relational Model**

3. A widely used Semantic model called _____

a. Data Model **b. ER Model** c. Hierarchical model d. Object-based model

4. _____is a more abstract

a. ER model b. **Semantic data model**

c. Conceptual data Mode networks d. Physical data model

5. _____model is used to pictorially denote entities & relationships

a. **ER model** b. Semantic data model

c. Conceptual data Mode networks d. Physical data model

6. A description Of data in terms of a data model is called_____

a. Schema b. Relation c. record d. entities

7. Field is otherwise known as _____

a. Column b. Entity c. Relation d. Relationship

8. Column is otherwise known as _____

a. Relation b. Entity **c. attribute** d. Relationship

9. ____is used to define the external and conceptual model.

a. DDL b. DML c. DCL d. TCL

10. Physical model specifies _____details.

a. Information b. data **c. Storage** d. relationships

11. The _____layer is responsible for process to process delivery.

a. physical b. presentation c. networks d. transport

12. ____ is the primary unit of storage in a database

a. table b. column c. row d. number

13. Set structures can represent a __ relationship between tables.

a. one-to-one **b. one-to-many** c. many-to-many d. many-to-one

14. An E-R modeling for given application leads to

a. conceptual data mode b. logical data model

c external data model d. internal data model

15. An entity is _____.

a. collection of different entities b. collection of related entities

c. related data d. **collection of similar entities**

16. What does SQL stand for?

a. Structured Query Language b. Strong Question Language

c. Structured Question Language d. Structure Query Language

17 Create, Alter, Drop commands are_____ language command..

a. DDL **b. DML** c. DCL d. TCL

18. Create.Alter,Drop commands are_____ language command

a. DDL **b. DML** c. DCL d. TCL

19. Selection Operation is used to ____from a relation

a. Select the Column **b. Select the row** c. Select the table d. Select DMBS

20. _____command is used to remove the table definition information

a. Delete b. Remove c. Destroy **d. Drop**

SECTION B – (3*2 = 6 Marks)

Answer All Questions

21. List the Characteristics of Database Approach

- Represent Some Aspects of real world applications
- Manages Information
- Easy Operation implementation
- Multiple views of database
- Data for specific purpose
- It has Users of Specific interest
- Self Describing nature
- Logical relationship between records and data

22. Define entity and attributes

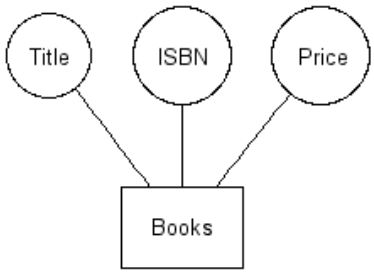
Entity

An entity is a database object that represents a thing in the real world. Entities are expressed as nouns.

Attributes

An attribute is a characteristic of an entity. Every entity has at least one attribute

EX: The books entity set



23. Draw an ER diagram for M:M relationship

- In Many - to - Many Relationship, many entities are related with the multiple other entities.
- This relationship is a type of cardinality which refers the relation between two entities.

For example: Various Books in a Library are issued by many Students.

Fig. Representation in ER Diagram

SECTION C – (3*8 = 24 Marks)

Answer the Questions

24. a) Define database and explain various database models with neat diagram.

Database

A database is a collection of information that is organized so that it can be easily accessed, managed and updated.

Data Model

A database model shows the logical structure of a database, including the relationships and constraints that determine how data can be stored and accessed. Individual database models are designed based on the rules and concepts of whichever broader data model the designers adopt. Most data models can be represented by an accompanying database diagram.

Flat Data Model

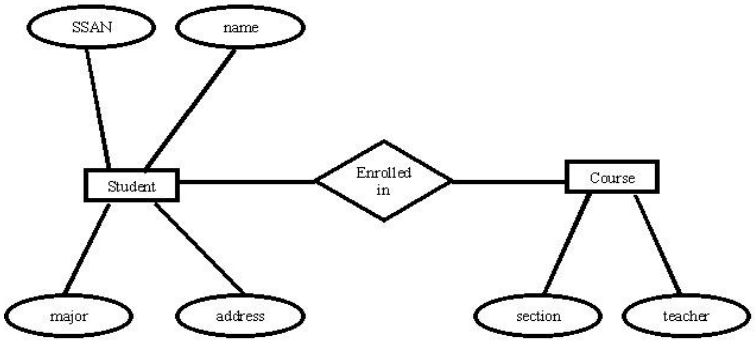
Flat data model is the first and foremost introduced model and in this all the data used is kept in the same plane. Since it was used earlier this model was not so scientific.

Roll No	Name	Course
5482	Mark	Web Designing
5486	Steve	Java
5496	Smith	Oracle

Entity Relationship Data Model

Entity relationship model is based on the notion of the real world entities and their relationships. While formulating the real world scenario in to the database model an entity set is created and this model is dependent on two vital things and they are :

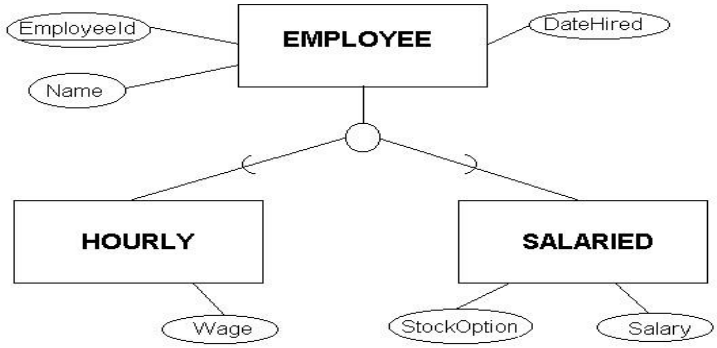
- Entity and their attributes
- Relationships among entities



An entity has a real world property called attribute and attribute define by a set of values called domain. For example, in a university a student is an entity, university is the database, name and age and sex are the attributes. The relationships among entities define the logical association between entities.

Relational Data Model

Relational model is the most popular model and the most extensively used model. In this model the data can be stored in the tables and this storing is called as relation, the relations can be normalized and the normalized relation values are called atomic values. Each row in a relation contains unique value and it is called as tuple, each column contains value from same domain and it is called as attribute.

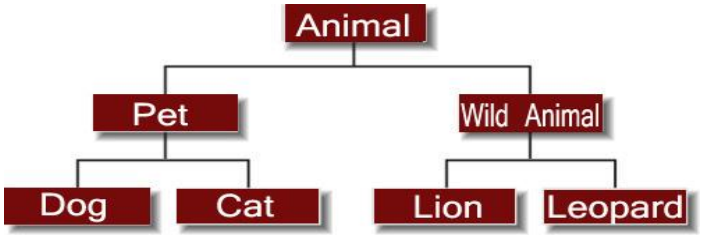


Network Data Model

Network model has the entities which are organized in a graphical representation and some entities in the graph can be accessed through several paths.

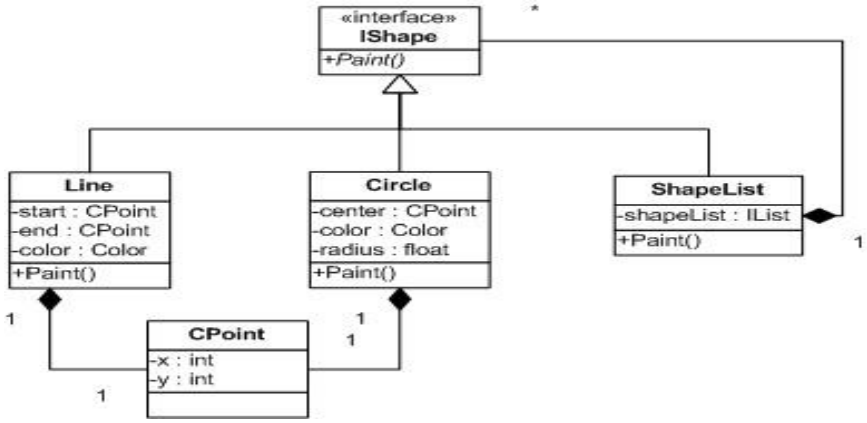
Hierarchical Data Model

Hierarchical model has one parent entity with several children entity but at the top we should have only one entity called root. For example, department is the parent entity called root and it has several children entities like students, professors and many more.



Object oriented Data Model

Object oriented data model is one of the developed data model and this can hold the audio, video and graphic files. These consist of data piece and the methods which are the DBMS instructions.



[OR]

b) Discuss data base architecture with neat sketch.

Following are the three levels of database architecture,

1. Physical Level
2. Conceptual Level
3. External Level

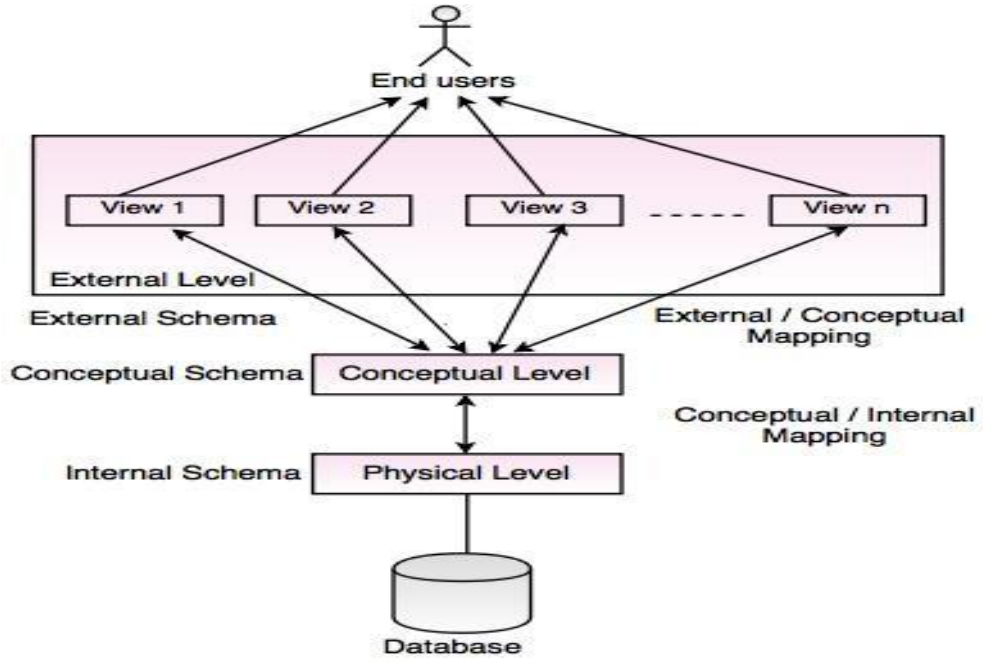


Fig. Three Level Architechture of DBMS

In the above diagram,

- It shows the architecture of DBMS.
- Mapping is the process of transforming request response between various database levels of architecture.
- Mapping is not good for small database, because it takes more time.
- In External / Conceptual mapping, DBMS transforms a request on an external schema against the conceptual schema.
- In Conceptual / Internal mapping, it is necessary to transform the request from the conceptual to internal levels.

1. Physical Level

- Physical level describes the physical storage structure of data in database.
- It is also known as Internal Level.
- This level is very close to physical storage of data.

- At lowest level, it is stored in the form of bits with the physical addresses on the secondary storage device.
- At highest level, it can be viewed in the form of files.
- The internal schema defines the various stored data types. It uses a physical data model.

2. Conceptual Level

- Conceptual level describes the structure of the whole database for a group of users.
- It is also called as the data model.
- Conceptual schema is a representation of the entire content of the database.
- These schema contains all the information to build relevant external records.
- It hides the internal details of physical storage.

3. External Level

- External level is related to the data which is viewed by individual end users.
- This level includes a no. of user views or external schemas.
- This level is closest to the user.

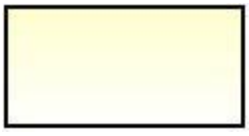
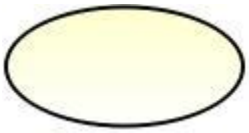
External view describes the segment of the database that is required for a particular user group and hides the rest of the database from that user group.

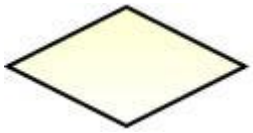
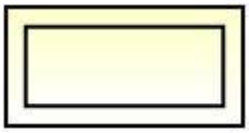
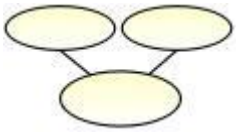
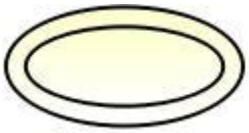
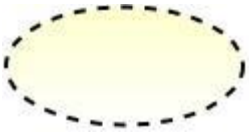
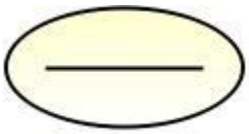
25. a) What are the typical ERD symbols and draw a neat ERD for an employee table.

ER Diagrams

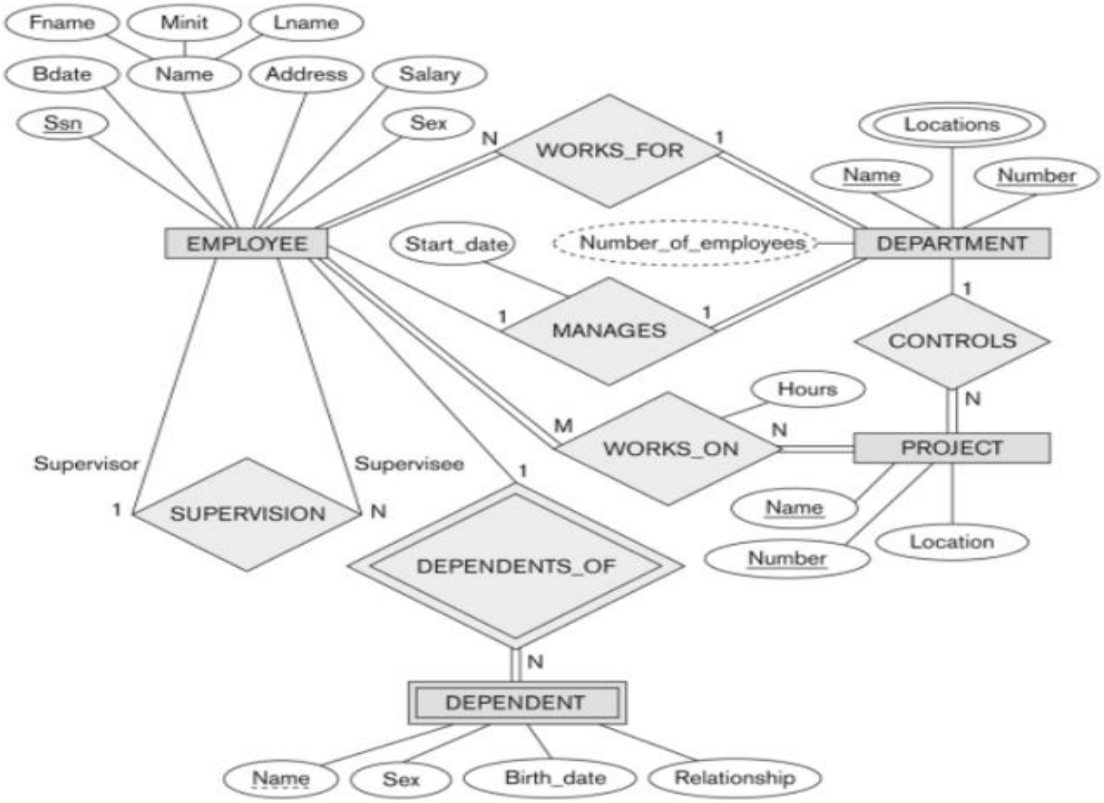
- ERD stands for Entity Relationship diagram.
- It is a graphical representation of an information system.
- ER diagram shows the relationship between objects, places, people, events etc.

Following are the components of ER Diagram

Notations	Representation	Description
	Rectangle	It represents the Entity.
	Ellipse	It represents the Attribute.

	Diamond	It represents the Relationship.
	Line	It represents the link between attribute and entity set to relationship set.
	Double Rectangle	It represents the weak entity.
	Composite Attribute	It represents composite attribute which can be divided into subparts. For eg. Name can be divided into First Name and Last Name
	Multi valued Attribute	It represents multi valued attribute which can have many values for a particular entity. For eg. Mobile Number.
	Derived Attribute	It represents the derived attribute which can be derived from the value of related attribute.
	Key Attribute	It represents key attribute of an entity which have a unique value in a table. For eg. Employee → EmpId (Employee Id is Unique).

ERD for an employee table



[OR]

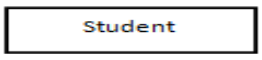
b) Explain i) Entity types ii) Relationships

i) Entity types

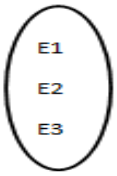
Entity, Entity Type, Entity Set

An Entity may be an object with a physical existence – a particular person, car, house, or employee – or it may be an object with a conceptual existence – a company, a job, or a university course.

An Entity is an object of Entity Type and set of all entities is called as entity set. e.g.; E1 is an entity having Entity Type Student and set of all students is called Entity Set. In ER diagram, Entity Type is represented as:



Entity Type



Entity Set

ii) Relationships

1. One - to - One Relationship

- In One - to - One Relationship, one entity is related with only one other entity.
- One row in a table is linked with only one row in another table and vice versa.

For example: A Country can have only one Capital City.

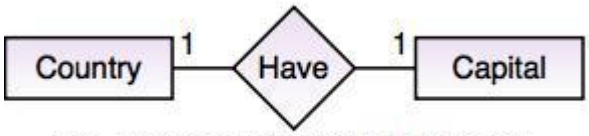


Fig. Representation in ER Diagram

2. One - to - Many Relationship

- In One - to - Many Relationship, one entity is related to many other entities.
- One row in a table A is linked to many rows in a table B, but one row in a table B is linked to only one row in table A.

For example: One Department has many Employees.

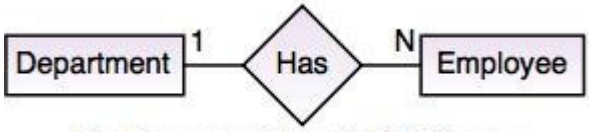


Fig. Representation in ER Diagram

3. Many - to - One Relationship

- In Many - to - One Relationship, many entities can be related with only one other entity.

For example: No. of Employee works for Department.

- Multiple rows in Employee table is related with only one row in Department table.

Fig. Representation in ER Diagram

4. Many - to - Many Relationship

- In Many - to - Many Relationship, many entities are related with the multiple other entities.
- This relationship is a type of cardinality which refers the relation between two entities.

For example: Various Books in a Library are issued by many Students.

Fig. Representation in ER Diagram

26. a) Ilustrate with syntax and example of relational constraints.

What are Database Constraints in DBMS ??

Database constraints are restrictions on the contents of the database or on database operations. It is a condition specified on a database schema that restricts the data to be inserted in an instance of the database.

Every relation has some conditions that must hold for it to be a valid relation. These conditions are called **Relational Integrity Constraints**.

Domain Constraints –

Domain Constraints specifies that what set of values an attribute can take. Value of each attribute X must be an atomic value from the domain of X.
The data type associated with domains include integer, character, string, date, time, currency etc. An attribute value must be available in the corresponding domain. Consider the example below –

SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	1st	18
8003	Somvir	4th	22
8004	Sourabh	6th	A

Not Allowed. Because Age is an Integer Attribute.

Tuple Uniqueness Constraints –

A relation is defined as a set of tuples. All tuples or all rows in a relation must be unique or distinct. Suppose if in a relation, tuple uniqueness constraint is applied, then all the rows of that table must be unique i.e. it does not contain the duplicate values. For example,

SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	2nd	18
8003	Somvir	4th	22
8004	Sourabh	6th	19

Not Allowed. Because all rows must be unique.

Key Constraints –

Keys are attributes or sets of attributes that uniquely identify an entity within its entity set. An Entity set E can have multiple keys out of which one key will be designated as the primary key. Primary Key must have unique and not null values in the relational table. In an subclass hierarchy, only the root entity set has a key or primary key and that primary key must serve as the key for all entities in the hierarchy.

Types of keys in DBMS

1. **Primary Key** – A primary is a column or set of columns in a table that uniquely identifies tuples (rows) in that table.

Example of Key Constraints in a simple relational table –

SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	1st	18
8003	Somvir	4th	22
8004	Sourabh	6th	45
8002	Tony	5th	23

Not allowed as Primary Key Values must be unique

Single Value Constraints –

Single value constraints refers that each attribute of an entity set has a single value. If the value of an attribute is missing in a tuple, then we cal fill it with a "null" value. The null value for a attribute will specify that either the value is not known or the value is not applicable. Consider the below example-

SID	Name	Class (semester)	Age	Driving License Number
8001	Ankit	1st	19	DL-45698
8002	Srishti	2nd	18	DL-45871, DL-89740
8003	Somvir	4th	22	DL-95687
8004	Sourabh	6th	19	

Not allowed as a person does not have two driving licenses.

Allowed as a person may or may not have a driving license.

Integrity Rule 1 (Entity Integrity Rule or Constraint) –

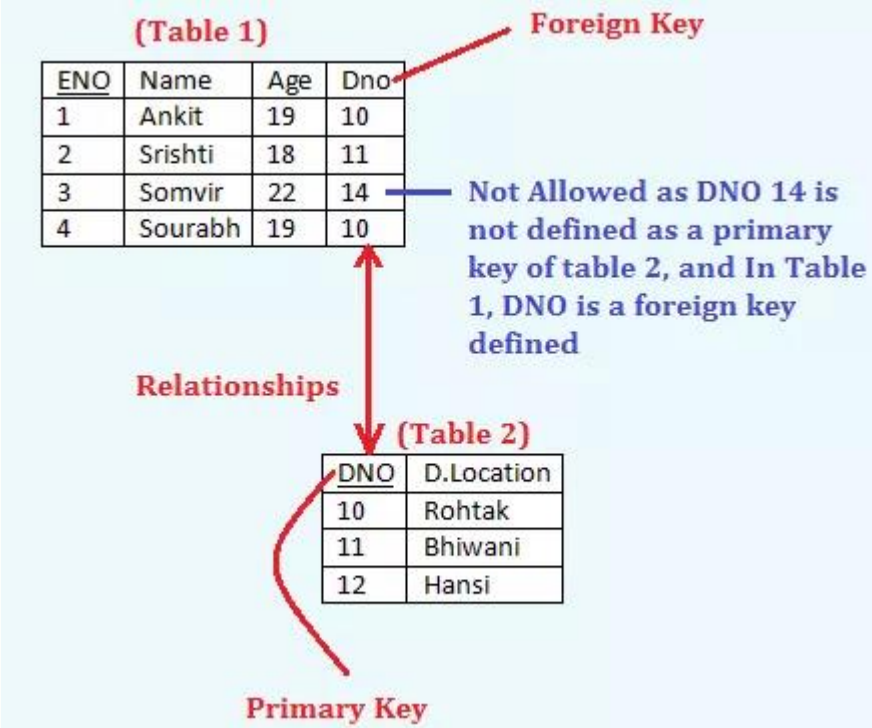
The Integrity Rule 1 is also called Entity Integrity Rule or Constraint. This rule states that no attribute of primary key will contain a null value. If a relation have a null value in the primary key attribute, then uniqueness property of the primary key cannot be maintained. Consider the example below-

SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	2nd	18
8003	Somvir	4th	22
	Sourabh	6th	19

Not allowed as primary key cannot contain a NULL value

Integrity Rule 2 (Referential Integrity Rule or Constraint) –

The integrity Rule 2 is also called the Referential Integrity Constraints. This rule states that if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2. For example,



[OR]

b) Discuss relational algebra with proper example.

RELATIONAL ALGEBRA

Relational Algebra is procedural query language, which takes Relation as input and generate relation as output. Relational algebra mainly provides theoretical foundation for relational databases and SQL.

- *Domain:* set of relations

- Based on set theory
- Contains extensions to manipulate tables
- Functional language
- Procedural, i.e., order to operations, algorithm implicit in the functional evaluation

Relational Algebra Operations

Below are fundamental operations that are "complete". That is, this set of operations alone can define any retrieval.

- Select
- Project
- Rename
- Union
- Set Difference
- Cartesian Product

Convenient, natural additions to the set of operations makes

- Set Intersection
- Natural Join
- Division
- Assignment

Projection

- Produce a subset of attributes from a relation
- Unselected columns are eliminated
- Duplicate rows are eliminated
- Result is a relation

Syntax: $\pi_{\text{attribute- list}}(\textit{relation})$

Example: The table **E** (for **EMPLOYEE**)

nr	name	salary
1	John	100
5	Sarah	300
7	Tom	100

SQL	Result	Relational algebra
select salary from E	salary 100	**PROJECT**$_{\text{salary}}$(E)

	300	
select nr, salary from E	nr | salary 1 | 100 5 | 30 7 | 100	**PROJECT**$_{nr,\ salary}$(E)

Note that there are no duplicate rows in the result.

Selection

Choose a subset of tuples from a relation based on some criteria, results in another relation called a ***"result set"***

Notation uses lower case sigma:

Synatx: $\sigma_{condition}$(relation)

The same table **E** (for **EMPLOYEE**) as above.

SQL	Result	Relational algebra
select * from E where salary < 200	nr | name | salary 1 | John | 100 7 | Tom | 100	**SELECT**$_{salary\ <\ 200}$(E)
select * from E where salary < 200 and nr >= 7	nr | name | salary 7 | Tom | 100	**SELECT**$_{salary\ <\ 200\ and\ nr\ >=\ 7}$(E)

Set Operators

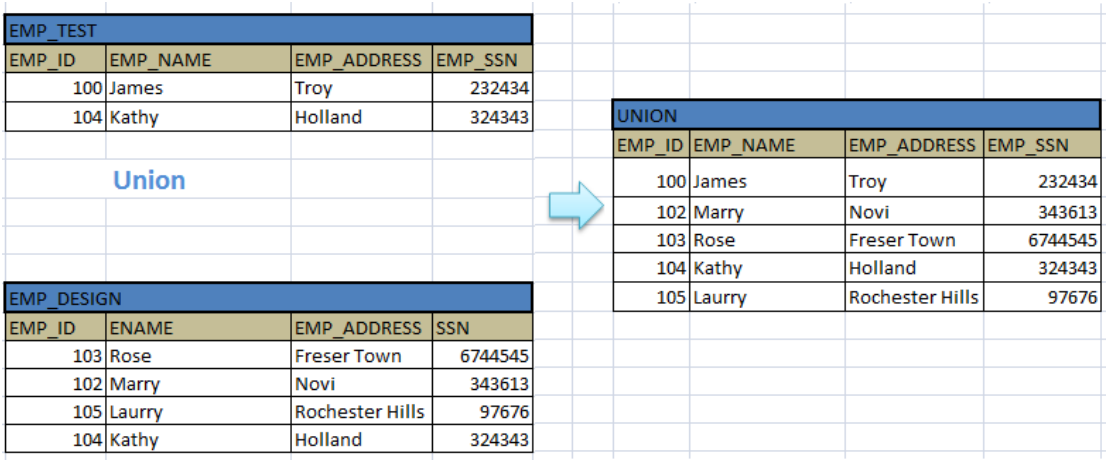
There are 4 main set operators used in the query language.

1. UNION

It combines the similar columns from two tables into one resultant table. All columns that are participating in the UNION operation should be Union Compatible.

```
SELECT EMP_ID, EMP_NAME, EMP_ADDRESS, EMP_SSN
```

```
FROM EMP_TEST
UNION
SELECT EMP_ID, EMP_NAME, EMP_ADDRESS, EMP_SSN
FROM EMP_DESIGN;
```



EMP_TEST

EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
104	Kathy	Holland	324343

EMP_DESIGN

EMP_ID	ENAME	EMP_ADDRESS	SSN
103	Rose	Freser Town	6744545
102	Marry	Novi	343613
105	Laurry	Rochester Hills	97676
104	Kathy	Holland	324343

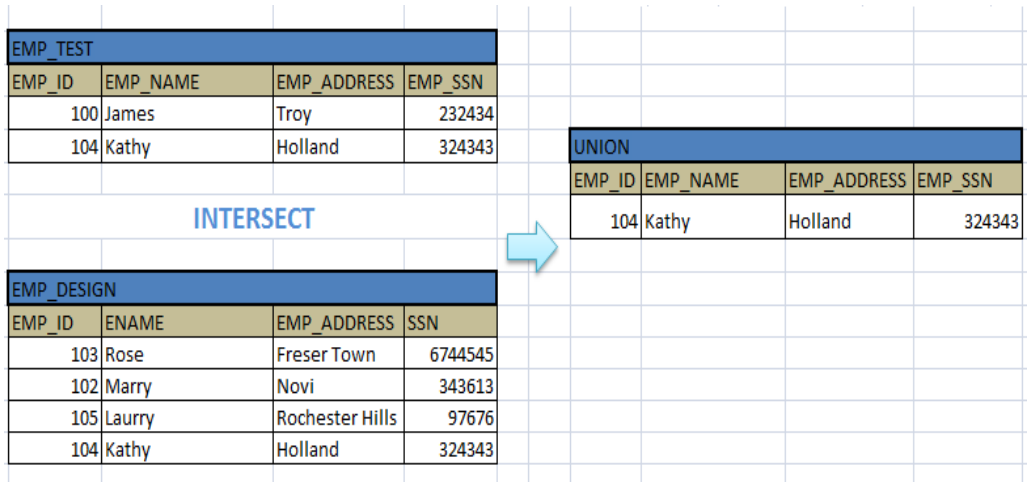
UNION

EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
102	Marry	Novi	343613
103	Rose	Freser Town	6744545
104	Kathy	Holland	324343
105	Laurry	Rochester Hills	97676

2. INTERSECT

This operator is used to pick the records from both the tables which are common to them

```
SELECT EMP_ID, EMP_NAME, EMP_ADDRESS, EMP_SSN
FROM EMP_TEST
INTERSECT
SELECT EMP_ID, EMP_NAME, EMP_ADDRESS, EMP_SSN
```



EMP_TEST

EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
104	Kathy	Holland	324343

EMP_DESIGN

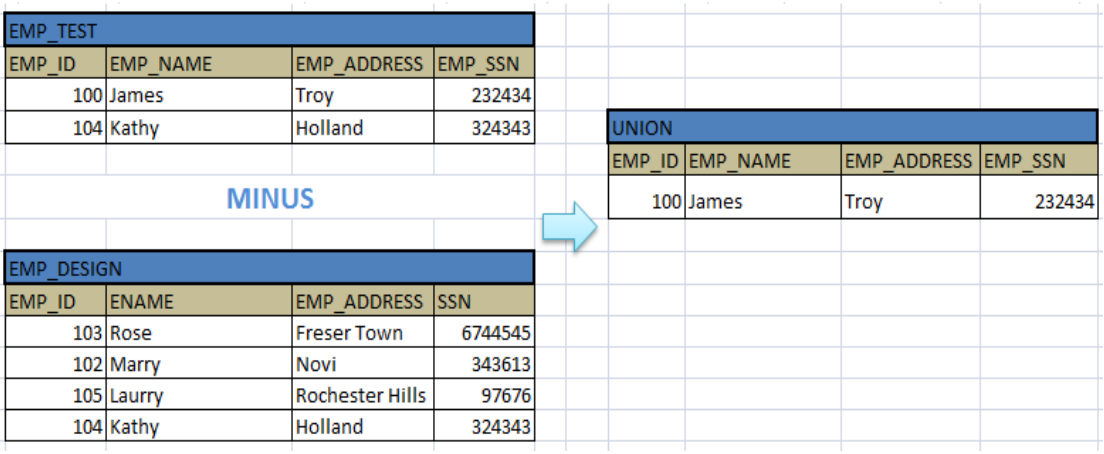
EMP_ID	ENAME	EMP_ADDRESS	SSN
103	Rose	Freser Town	6744545
102	Marry	Novi	343613
105	Laurry	Rochester Hills	97676
104	Kathy	Holland	324343

UNION

EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
104	Kathy	Holland	324343

```
SELECT EMP_ID, EMP_NAME, EMP_ADDRESS, EMP_SSN
```

```
FROM EMP_TEST
MINUS
SELECT EMP_ID, EMP_NAME, EMP_ADDRESS, EMP_SSN
FROM EMP_DESIGN;
```



EMP_TEST

EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434
104	Kathy	Holland	324343

MINUS

EMP_DESIGN

EMP_ID	ENAME	EMP_ADDRESS	SSN
103	Rose	Freser Town	6744545
102	Marry	Novi	343613
105	Laurry	Rochester Hills	97676
104	Kathy	Holland	324343

UNION

EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_SSN
100	James	Troy	232434