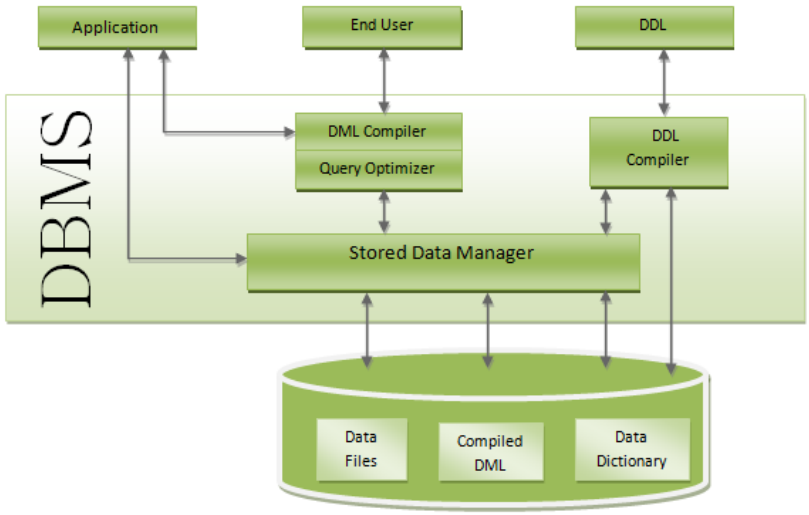
DBMS IMPORTANT QUESTIONS

Unit-1:

1. **Describe the Structure of Database Management system with a neat diagram.**

Ans:



1. Query Processor :

It interprets the requests (queries) received from end user via an application program into instructions. It also executes the user request which is received from the DML compiler.

Query Processor contains the following components –

DML Compiler –

It processes the DML statements into low level instruction (machine language), so that they can be executed.

DDL Interpreter –

It processes the DDL statements into a set of table containing meta data (data about data).

Embedded DML Pre-compiler –

It processes DML statements embedded in an application program into procedural calls.

Query Optimizer –

It executes the instruction generated by DML Compiler.

2. Storage Manager:

Storage Manager is a program that provides an interface between the data stored in the database and the queries received. It is also known as Database Control System. It maintains the consistency and integrity of the database by applying the constraints and executes the DCL statements. It is responsible for updating, storing, deleting, and retrieving data in the database.

It contains the following components –

Authorization Manager

Integrity Manager

Transaction Manager

File Manager

Buffer Manager

3. Disk Storage:

It contains the following components –

Data Files – It stores the data.

Data Dictionary – It contains the information about the structure of any database object. It is the repository of information that governs the metadata.

Indices – It provides faster retrieval of data item**.**

2. Define Database Management System (DBMS). List out the advantages of DBMS.

Ans: Database Management System (DBMS) Definition - What does Database Management System (DBMS) mean? A database management system (DBMS) is a software package designed to define, manipulate, retrieve and manage data in a database. A DBMS generally manipulates the data itself, the data format, field names, record structure and file structure.

Applications of DBMS:

- Banking.
- Airlines.
- Universities.
- Manufacturing and selling.
- Human resourses.

Advantages of DBMS:

1. Data Redundancy

Unlike traditional file-system storage, Data Redundancy in DBMS is very less or not present.

2. Data Inconsistency

In traditional file system storage, the changes made by one user in one application doesn't update the changes in other application, given both have the same set of details.

3. Data Sharing

Data Sharing is the primary advantage of Database management systems. DBMS system allows users and applications to share Data with multiple applications and users.

4. Data Searching

Searching and retrieving of data is very easy in DBMS systems.

5. Data Security

DBMS systems provide a strong framework to protect data privacy and security.

6. Data Concurrency

In DBMS, Data are stored in one or more servers in the network and that there is some software locking mechanism that prevents the same set of data from being changed by two people at the same time.

7. Data Integration

Data integration is a process of combining the data residing at different locations and present the user with a unified view of data.

8. Data Backup and Recovery

This is another advantage of DBMS as it provides a strong framework for Data backup, users are not required to back up their data periodically and manually, it is automatically taken care by DBMS. Moreover, in case of a server crash, DBMS restores the Database to its previous condition.

3.**What is Data Model? List out various Data Model and explain any three data models with suitable example.**

Ans:

Data Models define how the logical structure of a database is modeled. Data Models are fundamental entities to introduce abstraction in a DBMS. Data models define how data is connected to each other and how they are processed and stored inside the system.

FOUR TYPES:

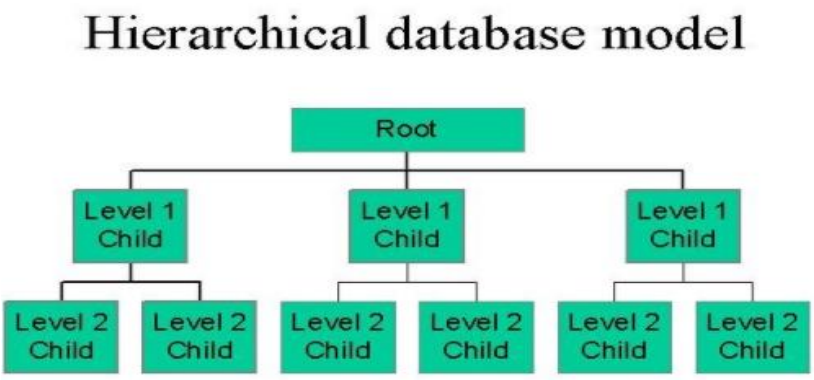
1. Hierarchical database
2. Network database
3. Relational database
4. Object-Oriented database

1. Hierarchical database: In a Hierarchical database model, data is organized in a tree-like structure.

- Data is Stored Hierarchically (top down or bottom up) format.
- Data is represented using a parent-child relationship.
- In Hierarchical DBMS, parent may have many children, but children have only one parent.

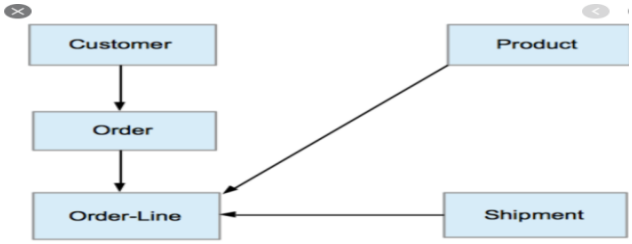
Note: This is old model, presently not using.

Because of Single DB concept**.**

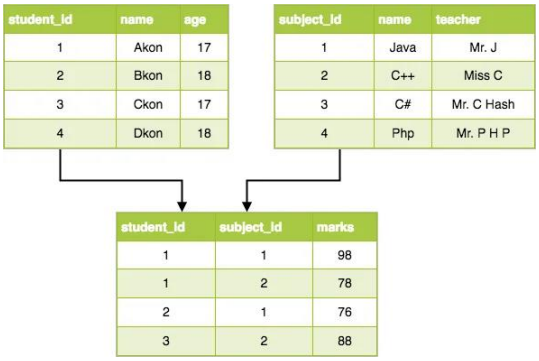


2. Network database: The network database model allows each child to have multiple parents.

- It helps you to address the need to model more complex relationships like as the orders/parts many-to-many relationship.
- In this model, entities are organized in a graph which can be accessed through several paths.
- Entity in DBMS can be a real-world object with an existence, For example, in a College database, the entities can be Professor, Students, Courses, etc. ... The attribute value gets stored in the database.



3. Relational database: In this model, data is organized in two-dimensional tables and the relationship is maintained by storing a common field.



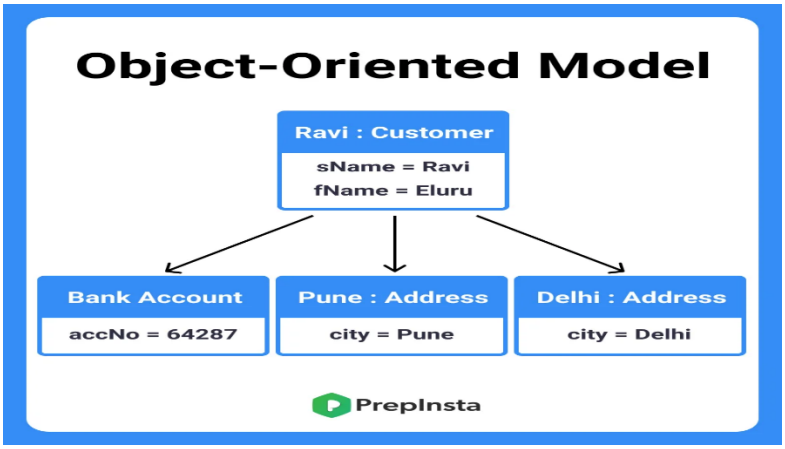
student_id	name	age
1	Akon	17
2	Bkon	18
3	Ckon	17
4	Dkon	18

subject_id	name	teacher
1	Java	Mr. J
2	C++	Miss C
3	C#	Mr. C Hash
4	Php	Mr. P H P

student_id	subject_id	marks
1	1	98
1	2	78
2	1	76
3	2	88

4. Object-Oriented database:

- **In Object-oriented Model data stored in the form of objects.**
- **The structure which is called classes which display data within it.**
- **It defines a database as a collection of objects which stores both data members values and operations.**

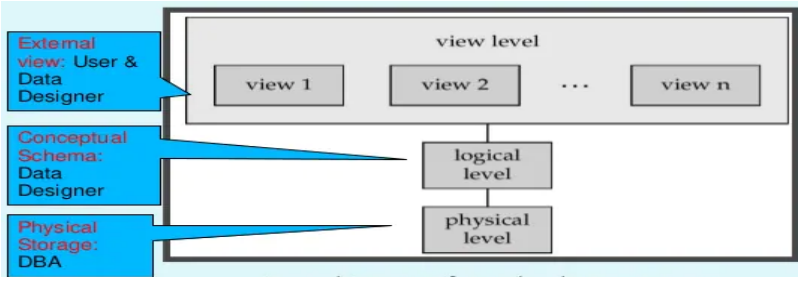


4.**What is meant by Data Abstraction? Describe various levels of Data Abstraction.**

Ans: Data Abstraction: We use Data Structures (DS) to store(represent) data in the Database (DB). Managing the DB efficiently in order to access data and perform calculations and manipulations to get required report/output is known as DBMS. Or Data Abstraction is a process of hiding unwanted or irrelevant details from the end user.

It provides a different view and helps in achieving data independence which is used to enhance the security of data.

Levels of Data Abstractions in a DBMS – There are 3 View Levels of Abstractions –



Physical level/Internal level : The physical representation of the database on the computer. This level describes *how* the data is stored in the database.

- It includes :
  - Where the data is located
  - File structures
  - Access methods
  - Indexes.

The physical schema is managed by the DBA.

Logical level/Conceptual level: The community view of the database. This level describes *what* data is stored in the database and the relationships among the data.

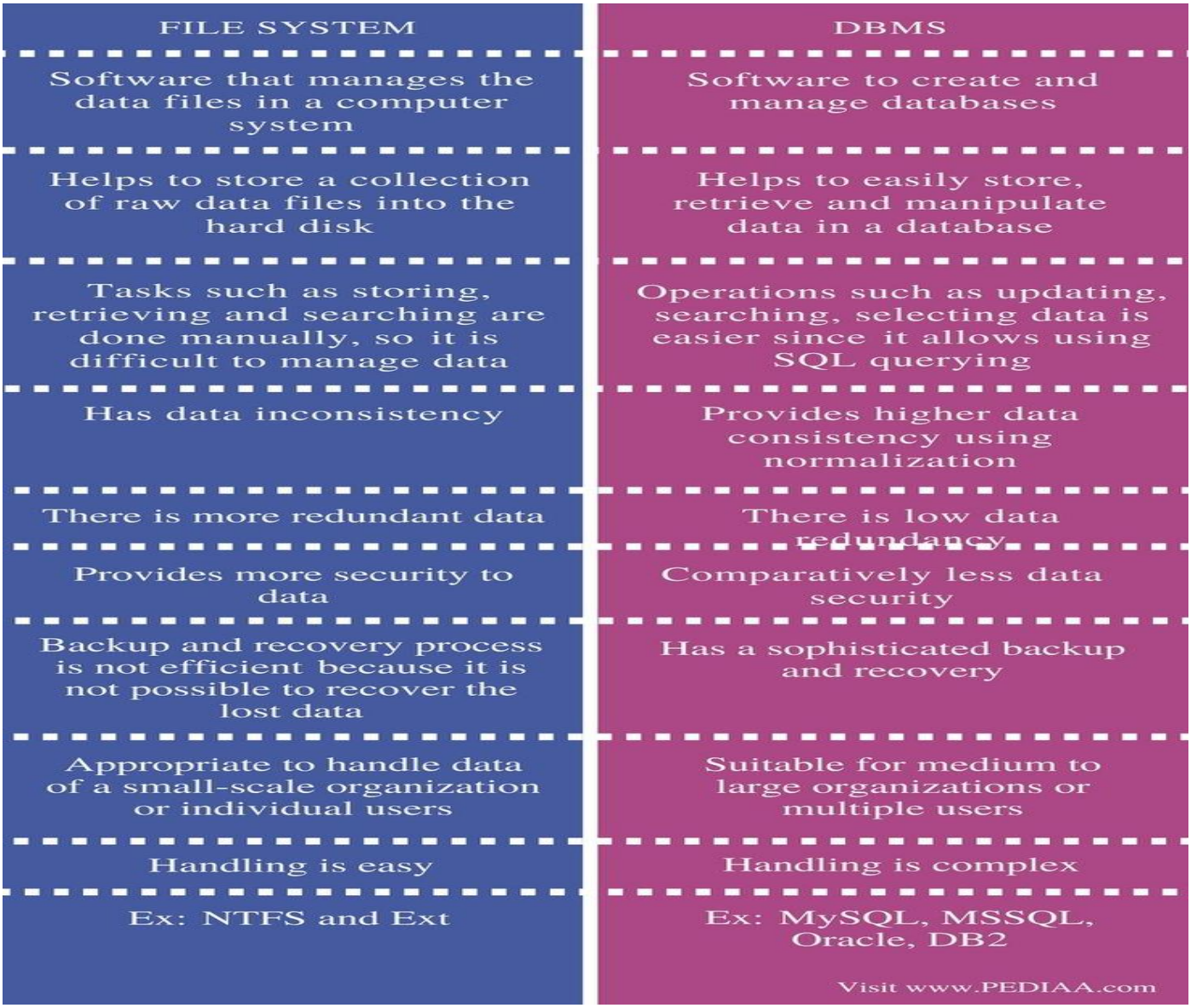
- What are the entities and Relationships in organization.
- What information these entities and relationships should store in database.
- What integrity constraints/business rules it should have?
- It consists of the schemas we have described with CREATE TABLE statements.

View level/ External Level: The users view of the database. This level describes that part of the database that is relevant to each user.

- Each external schema is a combination of base tables and views, tailored to the needs of a single user.
- It is managed by the data designer and the user.

5.Differentiate between File System and Database Management System.

Ans:



Unit-2:

1. **Construct an Entity Relationship diagram for an University database. Assume your own entities (Minimum of 5 entities), attributes and relations.**

Ans:

Entity: A thing in the real world with independent existence. Any particular row (a record) in a relation(table) is known as an entity.

An entity is written in

Relation: A relationship is an important part of any Entity relationship diagram as it shows the relation between two different entities.

There are four types of relations:

One to one – 1:1

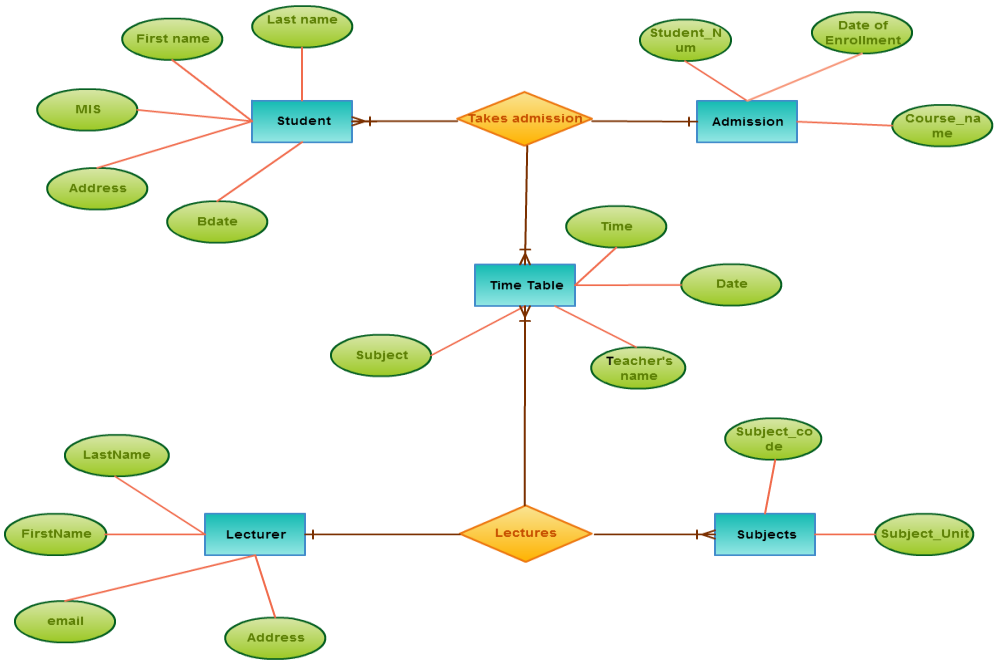
One to many – 1:M

Many to one – M:1

Many to many – M:M

Attribute: A relationship is an important part of any Entity relationship diagram as it shows the relation between two different entities.

Entity Relationship diagram for an University database:



2. **Draw an Entity Relationship diagram for Bus Reservation System. Assume your own entities, attributes and relations.**

Ans:Entity: A thing in the real world with independent existence. Any particular row (a record) in a relation(table) is known as an entity.

An entity is written in

Relation: A relationship is an important part of any Entity relationship diagram as it shows the relation between two different entities.

There are four types of relations:

One to one – 1:1

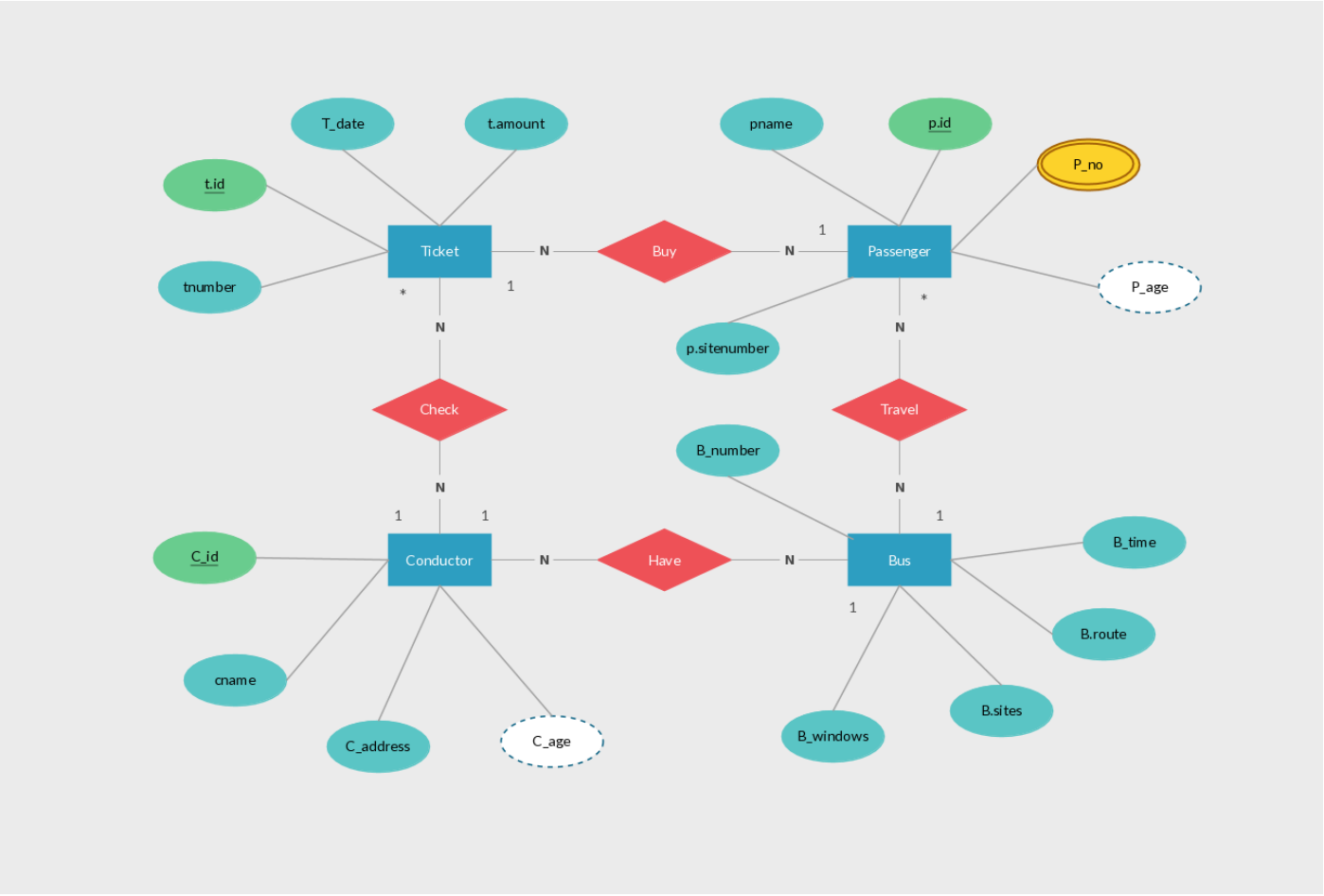
One to many – 1:M

Many to one – M:1

Many to many – M:M

Attribute: A relationship is an important part of any Entity relationship diagram as it shows the relation between two different entities.

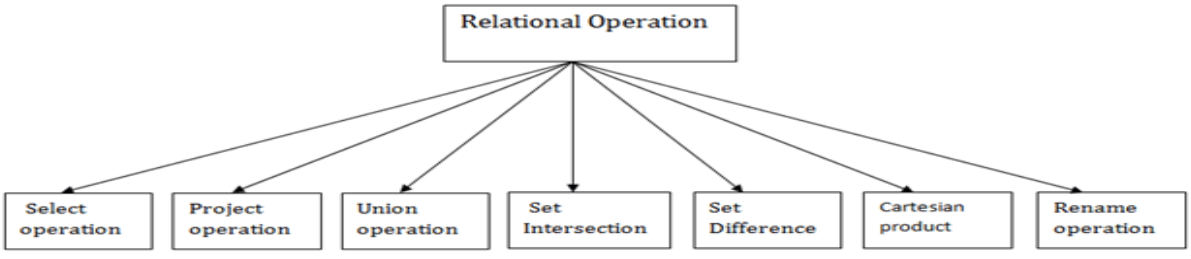
Entity Relationship diagram for Bus Reservation System



3. What is Relational Algebra? List out various types of Relational operations and explain operations with examples.

Ans: Relational algebra is a procedural query language. It gives a step by step process to obtain the result of the query (SQL). It uses operators to perform queries.

Types of Relational operation – 7



1. Select Operation: The select operation selects tuples that satisfy a given predicate.(codition)

* It is denoted by sigma (σ).

Notation: σ p(r)

Ex: **σ BRANCH_NAME=“Hyderabad" (LOAN)**

BRANCH_NAME	LOAN_NO	AMOUNT
Hyderabad	L-17	1000
Bangalore	L-23	2000
Chennai	L-15	1500
Vijayawada	L-14	1500
Hyderabad	L-13	500
Delhi	L-11	900
Bombay	L-16	1300

Output:

BRANCH_NAME	LOAN_NO	AMOUNT
Hyderabad	L-17	1000
Hyderabad	L-13	500

2. Project Operation: This operation shows the list of those attributes that we wish to appear in the result. Rest of the attributes are eliminated from the table.

* It is denoted by ∏

Notation: ∏ A1, A2, An (r)

Ex: ∏ NAME, CITY (CUSTOMER)

NAME	STREET	CITY
Jones	Main	Harrison
Smith	North	Rye
Hays	Main	Harrison
Curry	North	Rye
Johnson	Alma	Brooklyn
Brooks	Senator	Brooklyn

Output:

NAME	CITY
Jones	Harrison
Smith	Rye
Hays	Harrison
Curry	Rye
Johnson	Brooklyn
Brooks	Brooklyn

3. Union Operation: Suppose there are two tuples in R and S table. The union operation contains all the tuples that are either in R or S or both .

* It eliminates the duplicate tuples (records/rows). It is denoted by ∪.

Notation: R ∪ S

Ex: DEPOSITOR RELATION(R table)

CUSTOMER_NAME	ACCOUNT_NO
Johnson	A-101
Smith	A-121
Mayes	A-321
Turner	A-176
Johnson	A-273
Jones	A-472
Lindsay	A-284

BARROW RELATION (S table)

CUSTOMER_NAME	LOAN_NO
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17

∏ CUSTOMER_NAME (BORROW) ∪ ∏ CUSTOMER_NAME (DEPOSITOR)

CUSTOMER_NAME
Johnson
Smith
Hayes
Turner
Jones
Lindsay
Jackson
Curry
Williams
Mayes

4. Set Intersection Operation: Suppose there are two tuples (records/rows) in table R and S. The set intersection operation contains all tuples that are in both R & S.

* It is denoted by intersection ∩.

Notation: R ∩ S

Ex: ∏ **CUSTOMER_NAME (BORROW)** ∩ ∏ **CUSTOMER_NAME (DEPOSITOR)**

CUSTOMER_NAME
Smith
Jones

5. Set Difference Operation: Suppose there are two tuples in R and S table. The set difference operation contains all tuples that are in R but not in S.

* It is denoted by intersection minus (-).

Notation: R - S

Ex: ∏ **CUSTOMER_NAME (BORROW)** - ∏ **CUSTOMER_NAME (DEPOSITOR)**

CUSTOMER_NAME
Jackson
Hayes
Willians
Curry

6. Cartesian Product Operation: The Cartesian product is used to combine each row in one table with each row in the other table. It is also known as a cross product.

* It is denoted by X.

Notation: E X D

EMPLOYEE (table)

EMP_ID	EMP_NAME	EMP_DEPT
1	Smith	A
2	Harry	C
3	John	B

DEPARTMENT: (table)

DEPT_NO	DEPT_NAME
A	Marketing
B	Sales
C	Legal

EMPLOYEE X DEPARTMENT

EMP_ID	EMP_NAME	EMP_DEPT	DEPT_NO	DEPT_NAME
1	Smith	A	A	Marketing
1	Smith	A	B	Sales
1	Smith	A	C	Legal
2	Harry	C	A	Marketing
2	Harry	C	B	Sales
2	Harry	C	C	Legal
3	John	B	A	Marketing
3	John	B	B	Sales
3	John	B	C	Legal

7. Rename Operation:

*** The rename operation is used to rename the output relation.**

***It is denoted by rho (ρ).**

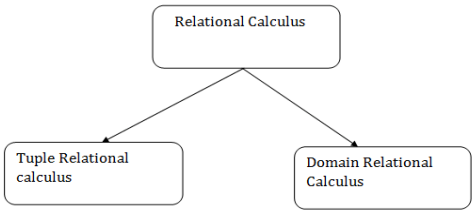
Input:

ρ(STUDENT1, STUDENT)

4.**What is Relational Calculus? Describe types of Relational Calculus with suitable example.**

Ans: Relational Calculus: Relational calculus is a non-procedural query language. In the non-procedural query language, the user is concerned with the details of how to obtain the end results.

Types of Relational calculus:



Tuple Relational Calculus (TRC):

- The tuple relational calculus is specified to select the tuples in a relation. In TRC, filtering variable uses the tuples of a relation.

The result of the relation can have one or more tuples.

- Notation:

  {T | P (T)} or {T | Condition (T)}
- P(T) – logically uses OR, AND, NOT connecttors.
- It also uses quantifiers– FORALL, THERE EXIST

Example:

FN	LN	Age
A	AA	35
B	BB	60

{t.LN/ student(t) AND t.age>30}

output:

LN
AA
BB

It returns a tuple with “name” from author who has written on article as “database”

Display the lastname of those students where age is greater than 30.

Domain Relational Calculus (DRC):

In domain relational calculus, filtering is done based on the domain of the attributes and not based on the tuple values.

Syntax: { c1, c2, c3, ..., cn | F(c1, c2, c3, ... ,cn)}

where, c1, c2... etc represents domain of attributes(columns)

and F defines the formula including the condition for fetching the data.

For example,

name	age
AA	19
BB	20

{< name, age > | ∈ Student ∧ age > 17}

OUTPUT:

name	age
AA	19
BB	20

Again, the above query will return the names and ages of the students in the table Student who are older than 17.

5. Translate Entity Relationship diagram into a collection of tables with associated constraints to a relational database schema.

Ans:

Unit-3:

1. **Identify and list various Data definition Language (DDL) commands. Create a table by specifying key and referential constraints in Structured Query Language.**

Ans:

DDL stands for Data Definition Language. As the name suggests, the DDL commands help to define the structure of the databases or schema. When we execute DDL statements, it takes effect immediately. The

changes made in the database using this command are saved permanently because its commands are auto-committed. The following commands come under DDL language:

- **CREATE:** It is used to create a new database and its objects such as table, views, function, stored procedure, triggers, etc.

Syntax –

CREATE TABLE table_name (column_1 datatype, column_2 datatype, column_3 datatype, ...);

Example –

CREATE TABLE Student_info (College_Id number(2), College_name varchar(30), Branch varchar(10));

- **DROP:** It is used to delete the database and its objects, including structures, from the server permanently.

Syntax –

DROP TABLE table_name;

Example –

If the College Authority wants to change their Database by deleting the Student_info Table.

DROP TABLE Student_info;

- **ALTER:** It's used to update the database structure by modifying the characteristics of an existing attribute or adding new attributes.

Syntax –

Syntax to add a column to an existing table.

ALTER TABLE table_name ADD column_name datatype;

Example –

In our Student_info table, we want to add a new column for CGPA. The syntax would be as below as follows.

ALTER TABLE Student_info ADD CGPA number;

- **TRUNCATE:** It is used to completely remove all data from a table, including their structure and space allocates on the server.

Syntax –

Syntax to remove an existing table.

TRUNCATE TABLE table_name;

Example –

TRUNCATE TABLE Student_info;

- **RENAME:** This command renames the content in the database.

Syntax:

alter table oldtable-name rename to newtablename;

Example query:

Alter table abc rename to def;

2. **Identify and list various Data Manipulation Language (DML) commands and Explain with suitable example.**

Ans:

- It stands for Data Manipulation Language. The DML commands deal with the manipulation of existing records of a database. It is responsible for all changes that occur in the database. The changes made in the database using this command can't save permanently because its commands are not auto-committed. Therefore, changes can be rollback. The following commands come under DML language:

SELECT: This command is used to extract information from a table.

Syntax:

SELECT *from table-name;

Example:

SELECT *from student;

INSERT: It is a SQL query that allows us to add data into a table's row.

Syntax:

INSERT INTO table-name values(colvalues);

Example:

INSERT INTO student values (name, lastname) VALUES (‘cde', ‘abc');

UPDATE: This command is used to alter or modify the contents of a table.

Syntax:

update tablename set col=‘new value’ where col=‘oldvalue’;

Example:

UPDATE student SET name = 'Dima' WHERE lastname = 'Shiva';

DELETE: This command is used to delete records from a database table, either individually or in groups.

Syntax:

DELETE FROM tablename WHERE colmn =‘value’;

Example:

DELETE FROM student WHERE name = 'Dima';

3.Define view. Write the syntax for creating ,updating and destroying a view with suitable example for Simple and Composite Views.

Ans: **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.

Simple views:

CREATE VIEW : We can create View using **CREATE VIEW** statement. A View can be created from a single table or multiple tables.

Syntax:

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

Example: CREATE VIEW DetailsView AS SELECT NAME, ADDRESS FROM StudentDetails WHERE S_ID < 5;

SELECT * FROM DetailsView;

- Output:

NAME	ADDRESS
Harsh	Kolkata
Ashish	Durgapur
Pratik	Delhi
Dhanraj	Bihar

CREATE VIEW StudentNames AS SELECT S_ID, NAME FROM StudentDetails ORDER BY NAME;

SELECT * FROM StudentNames;

Output:

S_ID	NAMES
2	Ashish
4	Dhanraj
1	Harsh
3	Pratik
5	Ram

DELETING VIEWS: We have learned about creating a View, but what if a created View is not needed any more? Obviously we will want to delete it. SQL allows us to delete an existing View. We can delete or drop a View using the DROP statement.

Syntax:

- **DROP VIEW view_name;**

Ex:DROP VIEW MarksView;

UPDATING VIEWS: There are certain conditions needed to be satisfied to update a view. If any one of these conditions is **not** met, then we will not be allowed to update the view.

Syntax:CREATE OR REPLACE VIEW view_name AS SELECT column1,coulmn2,.. FROM table_name ;

COMPOSITE VIEWS

CREATE VIEW:

In this example we will create a View named MarksView from two tables StudentDetails and StudentMarks. To create a View from multiple tables we can simply include multiple tables in the SELECT statement.

->CREATE VIEW MarksView AS SELECT StudentDetails.NAME, StudentDetails.ADDRESS, StudentMarks.MARKS FROM StudentDetails, StudentMarks WHERE StudentDetails.NAME = StudentMarks.NAME;

- SELECT * FROM MarksView;

Output:

NAME	ADDRESS	MARKS
Harsh	Kolkata	90
Pratik	Delhi	80
Dhanraj	Bihar	95
Ram	Rajasthan	85

UPDATING VIEWS:

For example, if we want to update the view **MarksView** and add the field AGE to this View from **StudentMarks** Table, we can do this as:

->CREATE OR REPLACE VIEW MarksView AS SELECT StudentDetails.NAME, StudentDetails.ADDRESS, StudentMarks.MARKS, StudentMarks.AGE FROM StudentDetails, StudentMarks WHERE StudentDetails.NAME = StudentMarks.NAME;

->SELECT * FROM MarksView;

Output:

NAME	ADDRESS	MARKS	AGE
Harsh	Kolkata	90	19
Pratik	Delhi	80	19
Dhanraj	Bihar	95	21
Ram	Rajasthan	85	18