Database Management System (MCA-112)
MCA 1st Sem (2020-21)

Dr. D. S. Singh
Associate Professor
Department of ITCA
MMMUT Gorakhpur
Email: dssitca@mmmut.ac.in

Relational Data Model and Languages

- Relational model can be represented as a table with columns and rows. Each row is known as a tuple and column as an attribute.

- Relational instance: In the relational database system, the relational instance is represented by a finite set of tuples. Relation instances do not have duplicate tuples.

- Relational schema: A relational schema contains the name of the relation and name of all columns or attributes.

- Relational key: In the relational key, each row has one or more attributes. It can identify the row in the relation uniquely.

Relational Data Model and Languages

❖ Example: STUDENT Relation

NAME	ROLL_NO	PHONE_NO	ADDRESS	AGE
Ram	14795	7305758992	Noida	24
Shyam	12839	9026288936	Delhi	35
Laxman	33289	8583287182	Gurugram	20
Mahesh	27857	7086819134	Ghaziabad	27
Ganesh	17282	9028 9i3988	Delhi	40

- ❖ In the given table, NAME, ROLL_NO, PHONE_NO, ADDRESS, and AGE are the attributes.
- ❖ The instance of schema STUDENT has 5 tuples.
- ❖ t3 = <Laxman, 33289, 8583287182, Gurugram, 20>

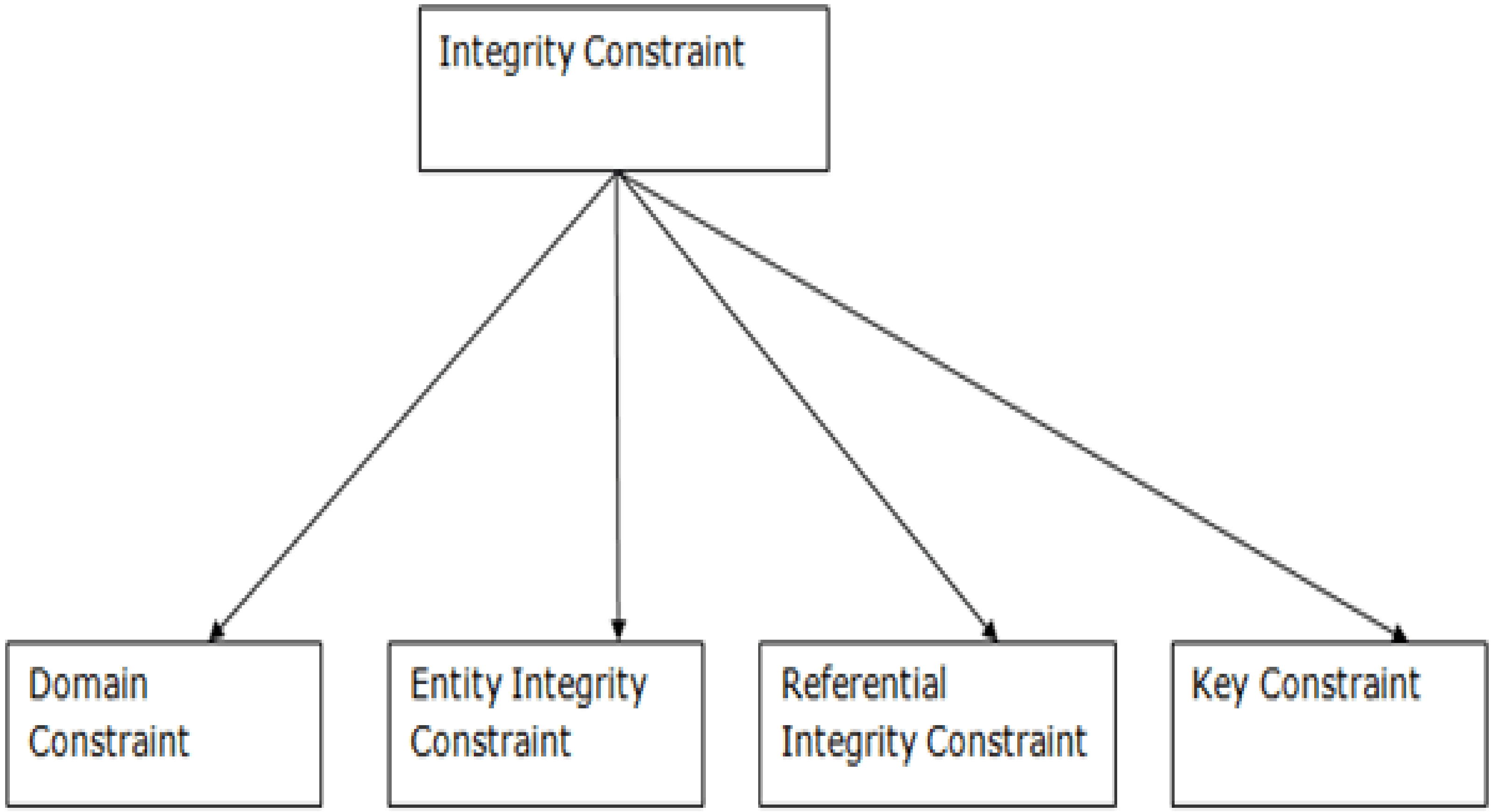
Properties of Relations

- Name of the relation is distinct from all other relations.
- Each relation cell contains exactly one atomic (single) value
- Each attribute contains a distinct name
- Attribute domain has no significance
- Tuple has no duplicate value
- Order of tuple can have a different sequence

Integrity Constraints

- Integrity constraints are a set of rules.
- It is used to maintain the quality of information.
- Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.
- Thus, integrity constraint is used to guard against accidental damage to the database.

Types of Integrity Constraints

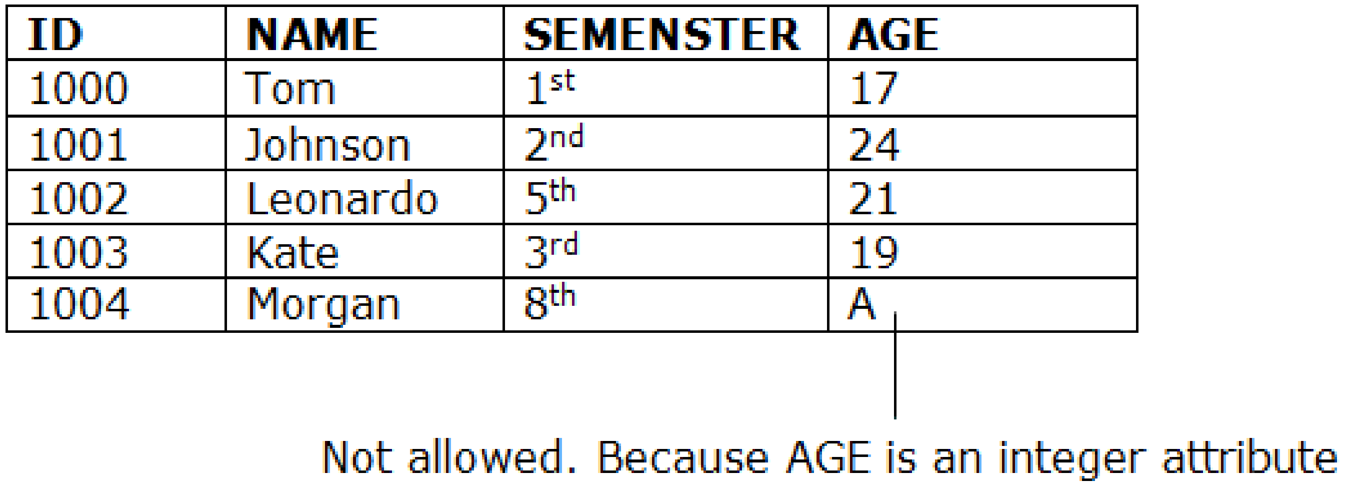


Domain Constraints

- ❖ Domain constraints can be defined as the definition of a valid set of values for an attribute.

- ❖ The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain.

Example:



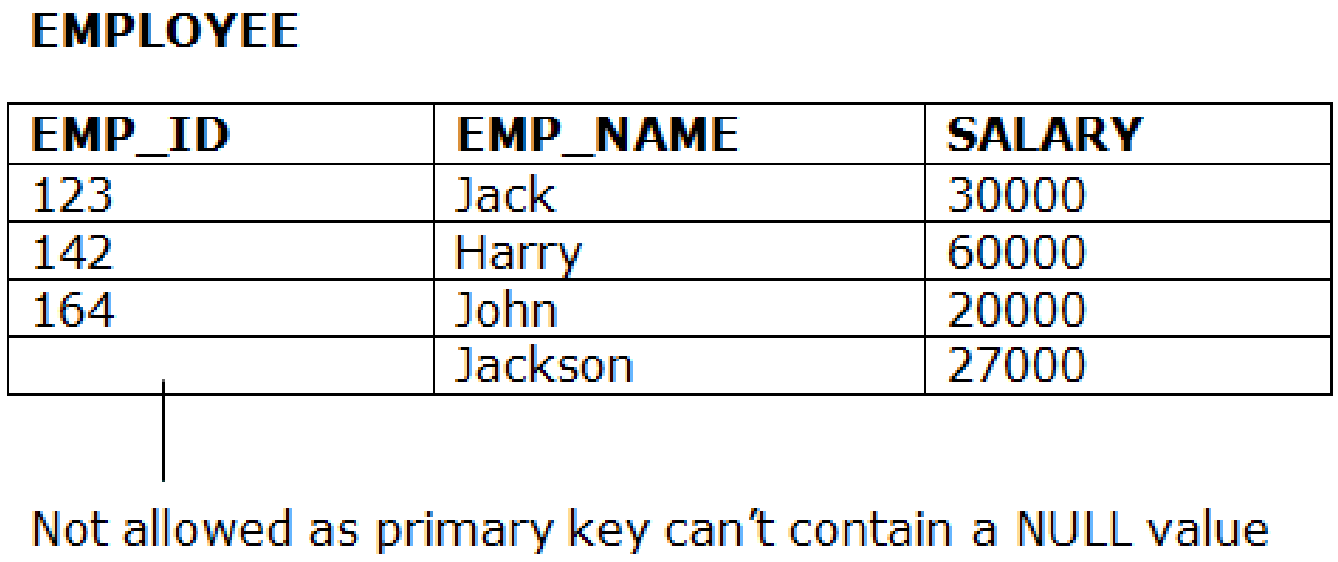
ID	NAME	SEMENSTER	AGE
1000	Tom	1st	17
1001	Johnson	2nd	24
1002	Leonardo	5th	21
1003	Kate	3rd	19
1004	Morgan	8th	A

Not allowed. Because AGE is an integer attribute

Entity integrity constraints

- ❖ The entity integrity constraint states that primary key value can't be null.
- ❖ This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.
- ❖ A table can contain a null value other than the primary key field.

Example



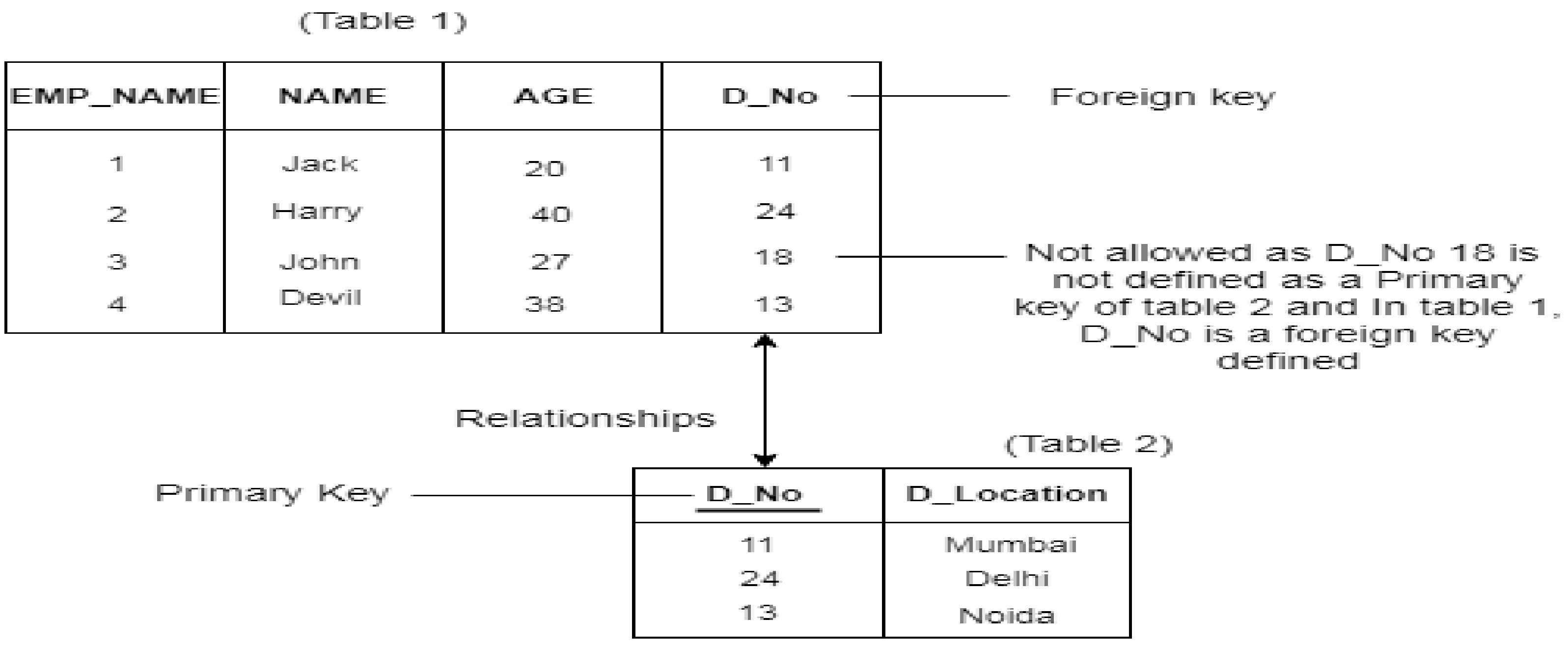
EMPLOYEE

EMP_ID	EMP_NAME	SALARY
123	Jack	30000
142	Harry	60000
164	John	20000
	Jackson	27000

Not allowed as primary key can't contain a NULL value

Referential Integrity Constraints

- ❖ A referential integrity constraint is specified between two tables.
- ❖ In the Referential integrity constraints, 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. **Example:**



(Table 1)

EMP_NAME	NAME	AGE	D_No
1	Jack	20	11
2	Harry	40	24
3	John	27	18
4	Devil	38	13

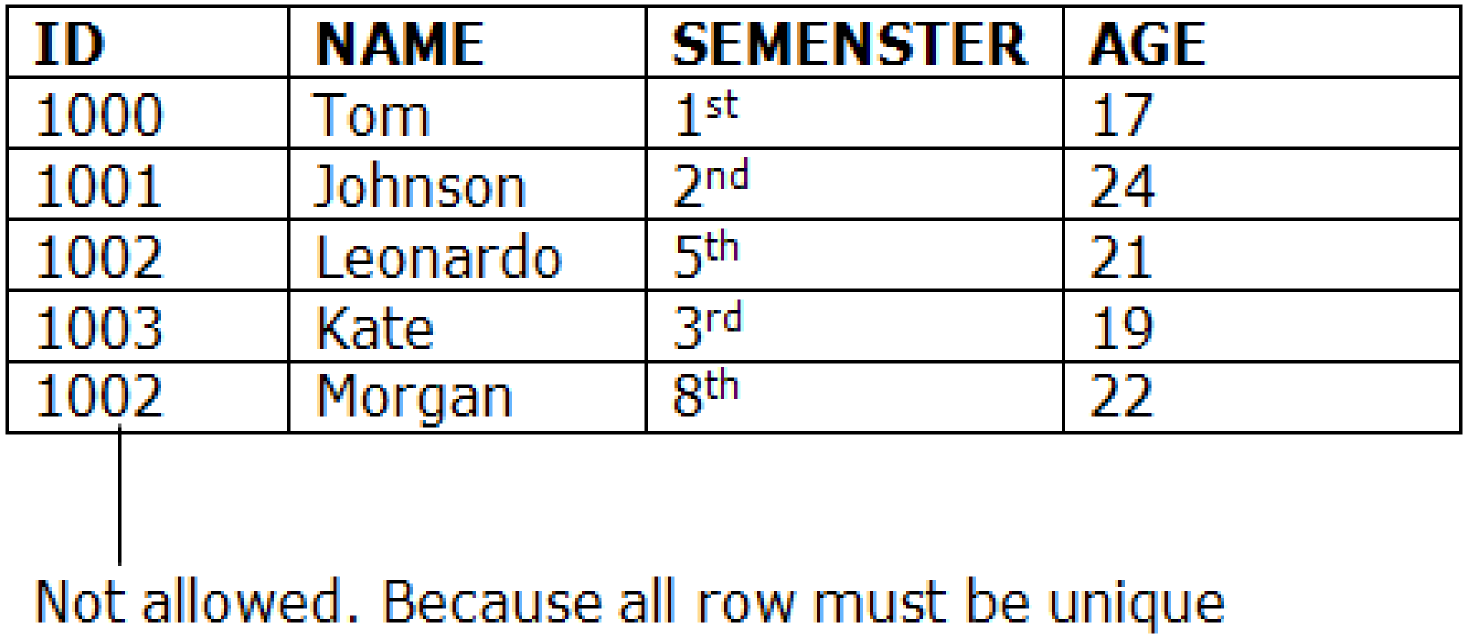
(Table 2)

D_No	D_Location
11	Mumbai
24	Delhi
13	Noida

Foreign key
Not allowed as D_No 18 is not defined as a Primary key of table 2 and In table 1, D_No is a foreign key defined
Relationships
Primary Key

Key Constraints

- Keys are the entity set that is used to identify an entity within its entity set uniquely.
- An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique value in the relational table. **Example:**

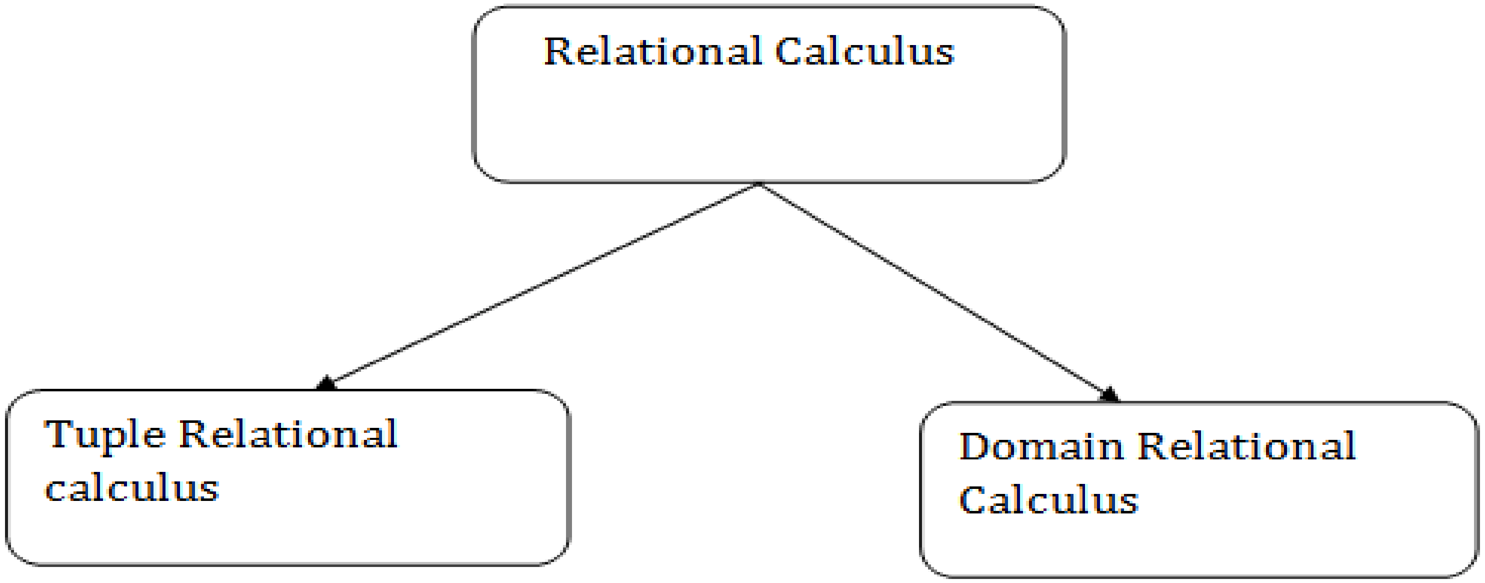


ID	NAME	SEMENSTER	AGE
1000	Tom	1st	17
1001	Johnson	2nd	24
1002	Leonardo	5th	21
1003	Kate	3rd	19
1002	Morgan	8th	22

Not allowed. Because all row must be unique

Relational Calculus

- ❖ Relational calculus is a non-procedural query language. In the non-procedural query language, the user is notconcerned with the details of how to obtain the end results.
- ❖ The relational calculus tells what to do but never explains how to do. There are two types of relational calculus.



Tuple Relational Calculus

- ❖ 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)}

Where, T is the resulting tuples and

P(T) is the condition used to fetch T.

Example of Tuple Relational Calculus

1. { T.name | Author(T) AND T.article = 'database' }

OUTPUT: This query selects the tuples from the AUTHOR relation. It returns a tuple with 'name' from Author who has written an article on 'database'.

TRC (tuple relation calculus) can be quantified. In TRC, we can use Existential (∃) and Universal Quantifiers (∀).

2.{ R| ∃T ∈ Authors(T.article='database' AND R.name=T.name)}

Output: This query will yield the same result as the previous one.

Domain Relational Calculus (DRC)

- The second form of relation is known as Domain relational calculus. In domain relational calculus, filtering variable uses the domain of attributes.
- Domain relational calculus uses the same operators as tuple calculus. It uses logical connectives ∧ (and), ∨ (or) and ┐ (not).
- It uses Existential (∃) and Universal Quantifiers (∀) to bind the variable.

Notation:

{ a1, a2, a3, ..., an | P (a1, a2, a3, ... ,an)}

Where

a1, a2 are attributes

P stands for formula built by inner attributes

Example of Domain Relational Calculus (DRC)

1.{< article, page, subject > | ∈ Javatpoint ∧ subject = 'database'}

Output: This query will yield the article, page, and subject from the relation javatpoint, where the subject is a database.

Structured Query Language (SQL)

- SQL is used to perform operations on the records stored in the database such as updating records, deleting records, creating and modifying tables, views, etc.
- SQL is just a query language not a database. To perform SQL queries, we need to install any database, for example, Oracle, MySQL, MongoDB, PostGre SQL, SQL Server, DB2, etc.
- 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 query language, it is used for database creation, deletion, fetching rows, and modifying rows, etc.
- SQL is based on relational algebra and tuple relational calculus.

Need of SQL

- 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

Functions of SQL

- ❖ With SQL, we can query our database in several ways, using English-like statements.
- ❖ With SQL, a user can access data from a relational database management system.
- ❖ It allows the user to describe the data.
- ❖ It allows the user to define the data in the database and manipulate it when needed.
- ❖ It allows the user to create and drop database and table.
- ❖ It allows the user to create a view, stored procedure, function in a database.
- ❖ It allows the user to set permission on tables/procedures/ views.

Characteristics of SQL

- SQL is easy to learn.
- SQL is used to access data from relational DBMS.
- SQL can execute queries against the database.
- SQL is used to describe the data.
- SQL is used to define the data in the database and manipulate it when needed.
- SQL is used to create and drop the database and table.
- SQL is used to create a view, stored procedure, function in a database.
- SQL allows users to set permissions on tables/procedures/views.

Advantages of SQL

High speed

- ✓ Using the SQL queries, the user can quickly and efficiently retrieve a large amount of records from a database.

No coding needed

- ✓ In the standard SQL, it is very easy to manage the database system. It doesn't require a substantial amount of code to manage the database system.

Well defined standards

- ✓ Long established are used by the SQL databases that are being used by ISO and ANSI.

Advantages of SQL

Portability

- ✓ SQL can be used in laptop, PCs, server and even some mobile phones.

Interactive language

- ✓ SQL is a domain language used to communicate with the database. It is also used to receive answers to the complex questions in seconds.

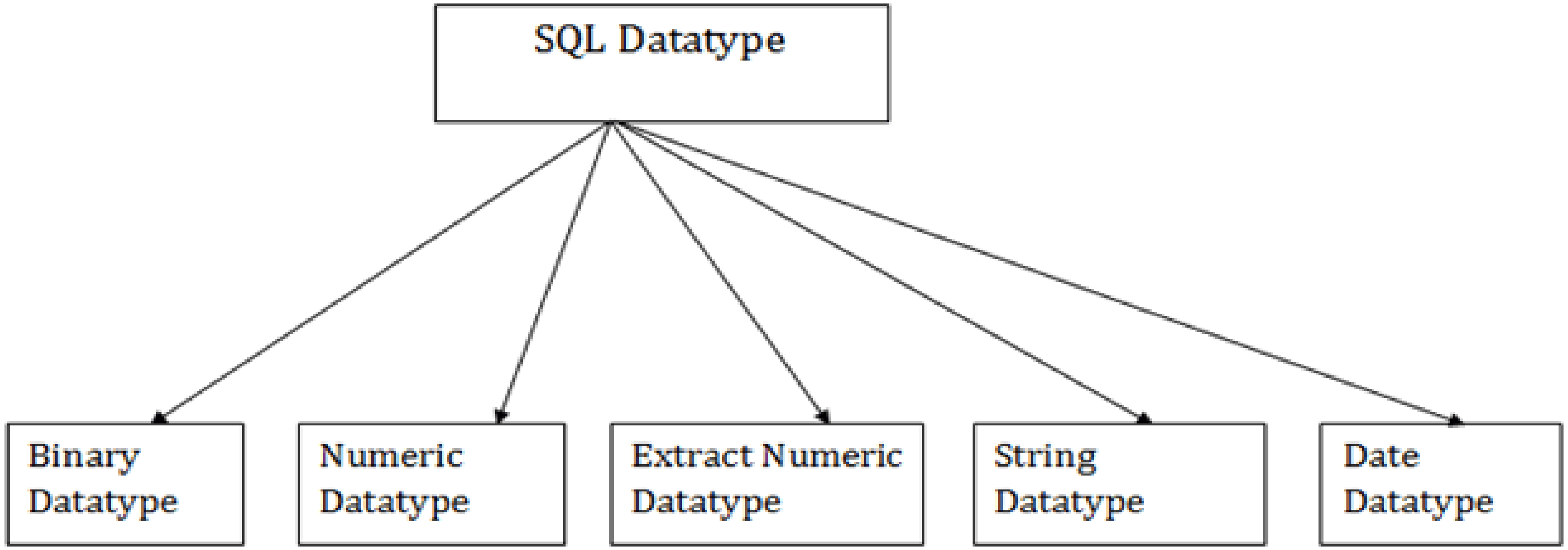
Multiple data view

- ✓ Using the SQL language, the users can make different views of the database structure.

SQL Data Types and Literals

SQL Datatype

- ❖ SQL Datatype is used to define the values that a column can contain.
- ❖ Every column is required to have a name and data type in the database table. There are various types of SQL Datatypes.



Binary Datatypes

There are Three types of binary Datatypes which are given below:

Data Type	Description
binary	It has a maximum length of 8000 bytes. It contains fixed-length binary data.
varbinary	It has a maximum length of 8000 bytes. It contains variable-length binary data.
image	It has a maximum length of 2,147,483,647 bytes. It contains variable-length binary data.

Approximate Numeric Datatype

The subtypes are given below:

Data type	From	To	Description
float	-1.79E + 308	1.79E + 308	It is used to specify a floating-point value e.g. 6.2, 2.9 etc.
real	-3.40e + 38	3.40E + 38	It specifies a single precision floating point number

Exact Numeric Datatype

The subtypes are given below:

Data type	Description
int	It is used to specify an integer value.
smallint	It is used to specify small integer value.
bit	It has the number of bits to store.
decimal	It specifies a numeric value that can have a decimal number.
numeric	It is used to specify a numeric value.

Character String Datatype

The subtypes are given below:

Data type	Description
char	It has a maximum length of 8000 characters. It contains Fixed-length non-unicode characters.
varchar	It has a maximum length of 8000 characters. It contains variable-length non-unicode characters.
text	It has a maximum length of 2,147,483,647 characters. It contains variable-length non-unicode characters.

Date and time Datatype

The subtypes are given below:

Datatype	Description
date	It is used to store the year, month, and days value.
time	It is used to store the hour, minute, and second values.
timestamp	It stores the year, month, day, hour, minute, and the second value.

SQL Commands

❖ SQL commands are instructions.

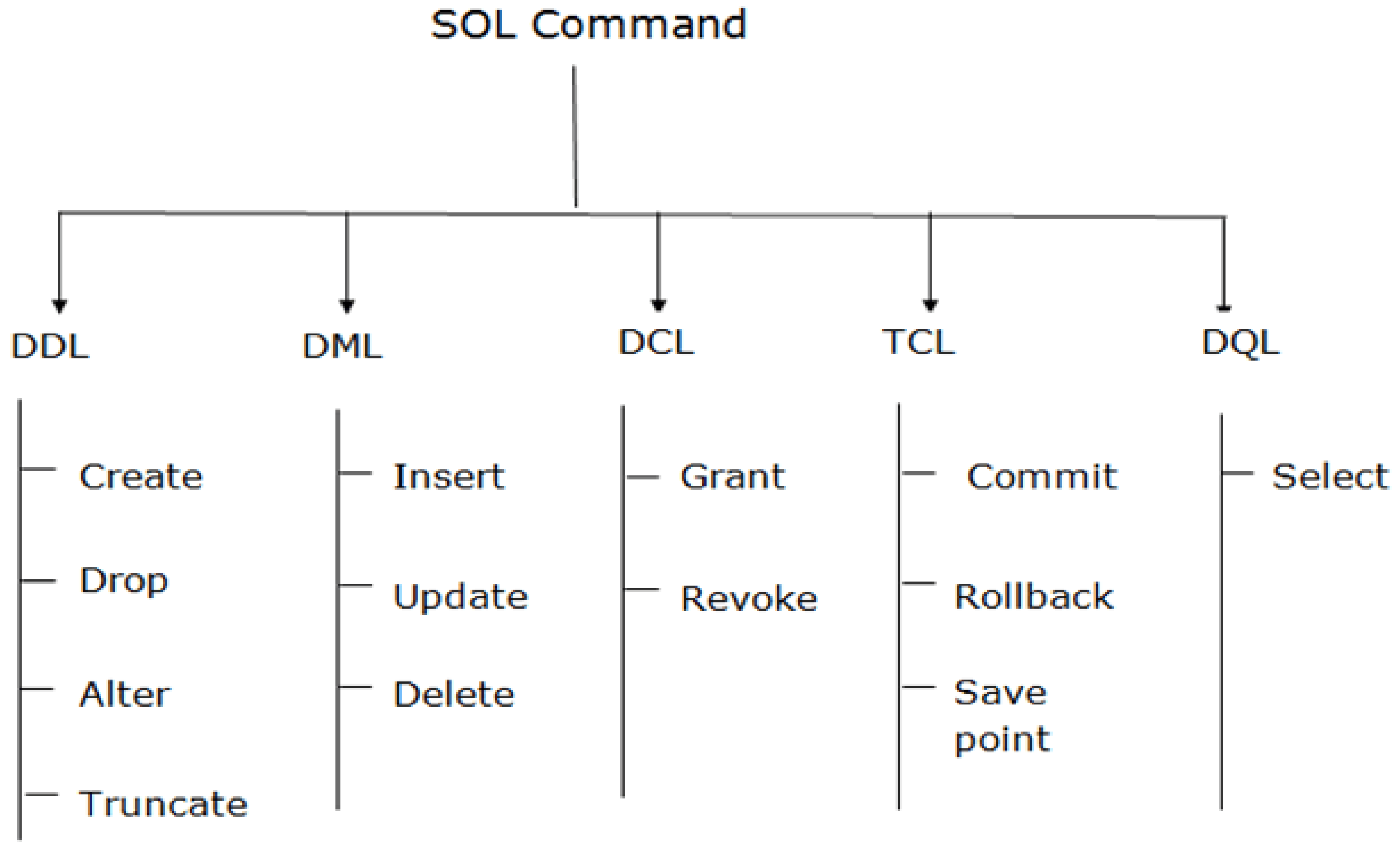
❖ It is used to communicate with the database.

❖ It is also used to perform specific tasks, functions, and queries of data.

❖ SQL can perform various tasks like create a table, add data to tables, drop the table, modify the table, set permission for users etc.

Types of SQL Commands

There are five types of SQL commands: DDL, DML, DCL, TCL, and DQL.



Data Definition Language (DDL)

- DDL changes the structure of the table like creating a table, deleting a table, altering a table, etc.

- All the command of DDL are auto-committed that means it permanently save all the changes in the database. There are some commands that come under DDL:

  - ✓ CREATE
  - ✓ ALTER
  - ✓ DROP
  - ✓ TRUNCATE

Data Definition Language (DDL)

1. **CREATE:** It is used to create a new table in the database.

Syntax:

CREATE TABLE TABLE_NAME (COLUMN_NAME DATATYPES[,....]);

Example:

CREATE TABLE EMPLOYEE(Name VARCHAR2(20), Email VARCHAR2(100), DOB DATE);

Data Definition Language (DDL)

2. DROP: It is used to delete both the structure and record stored in the table.

Syntax:

DROP TABLE Table Name;

Example:

DROP TABLE EMPLOYEE;

Data Definition Language (DDL)

3. ALTER: It is used to alter the structure of the database. This change could be either to modify the characteristics of an existing attribute or probably to add a new attribute.

Syntax:

1. To add a new column in the table

   ALTER TABLE table_name ADD column_name Datatype(Size);

2. To modify existing column in the table:

   ALTER TABLE MODIFY(COLUMN DEFINITION....);

EXAMPLE:

1.ALTER TABLE STU_DETAILS ADD(ADDRESS VARCHAR2(20));

2.ALTER TABLE STU_DETAILS MODIFY (NAME VARCHAR2(20));

Data Definition Language (DDL)

3. TRUNCATE: It is used to delete all the rows from the table and free the space containing the table.

Syntax:

```
1.TRUNCATE TABLE table_name;
```

Example:

```
1.TRUNCATE TABLE EMPLOYEE;
```

SQL Operators

- ❖ SQL statements generally contain some reserved words or characters that are used to perform operations such as comparison and arithmetical operations etc. These reserved words or characters are known as operators.

- ❖ Generally, there are three types of operators in SQL.

  - ✓ SQL Arithmetic Operators
  - ✓ SQL Comparison Operators
  - ✓ SQL Logical Operators

SQL Arithmetic Operators

Let's assume two variables "a" and "b". Here "a" is valued 50 and "b" valued 100. Example:

perators	Descriptions	Examples
+	It is used to add containing values of both operands	a+b will give 150
-	It subtracts right hand operand from left hand operand	a-b will give -50
*	It multiply both operand's values	a*b will give 5000
/	It divides left hand operand by right hand operand	b/a will give 2
%	It divides left hand operand by right hand operand and returns reminder	b%a will give 0

SQL Comparison Operators

Operator	Description	Example
=	Examine both operands value that are equal or not, if yes condition become true.	(a=b) is true
!=	This is used to check the value of both operands equal or not,if not condition become true.	(a!=b) is true
< >	Examines the operand's value equal or not, if values are not equal condition is true	(a<>b) is true
>	Examine the left operand value is greater than right Operand, if yes condition becomes true	(a>b) is not true
<	Examines the left operand value is less than right Operand, if yes condition becomes true	(a<="" td="">
>=	Examines that the value of left operand is greater than or equal to the value of right operand or not,if yes condition become true	(a>=b) is not true
<=	Examines that the value of left operand is less than or equal to the value of right operand or not, if yes condition becomes true	(a<=b) is true
!<	Examines that the left operand value is not less than the right operand value	(a!<="" td="">
!>	Examines that the value of left operand is not greater than the value of right operand	(a!>b) is true

SQL Logical Operators

Operator	Description
ALL	this is used to compare a value to all values in another value set.
AND	this operator allows the existence of multiple conditions in an SQL statement.
ANY	this operator is used to compare the value in list according to the condition.
BETWEEN	this operator is used to search for values, that are within a set of values
IN	this operator is used to compare a value to that specified list value
NOT	the NOT operator reverse the meaning of any logical operator
OR	this operator is used to combine multiple conditions in SQL statements
EXISTS	the EXISTS operator is used to search for the presence of a row in a specified table
LIKE	this operator is used to compare a value to similar values using wildcard operator

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.
- 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:

SQL 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.

SQL Table Variable

```
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");
```

SQL CREATE TABLE

- 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 (2)  NOT NULL,
NAME VARCHAR (20) NOT NULL,
AGE INT  (2) NOT NULL,
ADDRESS CHAR (25),
PRIMARY KEY (ID)
);
```

SQL CREATE TABLE

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	

SQL CREATE TABLE

```
CREATE TABLE Employee (
EmployeeID number(10),
FirstName varchar2(255),
LastName varchar2(255),
Email varchar2(255),
AddressLine varchar2(255),
City varchar2(255)
);
```

Create a Table using another table

```
CREATE TABLE table_name  AS
SELECT column1, column2,...
FROM old_table_name WHERE ..... ;
```

The following SQL creates a copy of the employee table.

```
CREATE TABLE Employee_New AS
SELECT Employee_ID, First_Name, Email
FROM Employee;
```

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.

- Syntax: DROP TABLE "table_name";

- SQL>DROP TABLE STUDENTS;

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 DROP statement and TRUNCATE statement, but they are not exactly same, there are some differences between them.

DIFFERENCE B/W DELETE and TRUNCATE

- 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.

DIFFERENCE B/W DROP and TRUNCATE

❖ 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 RENAME TABLE

- SQL RENAME TABLE syntax is used to change the name of a table. Sometimes, we choose non-meaningful name for the table. So, it is required to be changed.
- Let's see the syntax to rename a table from the database.
- Optionally, you can write following command to rename the table.
- RENAME old_table _name To new_table_name;
- Let us take an example of a table named "STUDENTS", now due to some reason we want to change it into table name "ARTISTS".
- RENAME STUDENTS To ARTISTS;

SQL TRUNCATE TABLE

- ❖ Truncate SQL statement is used to remove all rows (complete data) from a table. It is similar to the DELETE statement with no WHERE clause.

TRUNCATE TABLE Vs DELETE TABLE

- ❖ Truncate table is faster and uses lesser resources than DELETE TABLE command.

TRUNCATE TABLE Vs DROP TABLE

- ❖ Drop table command can also be used to delete complete table but it deletes table structure too. TRUNCATE TABLE doesn't delete the structure of the table.Let's see the syntax to truncate the table from the database.

SQL TRUNCATE TABLE

- TRUNCATE TABLE table_name;

- For example, you can write following command to truncate the data of employee table

- TRUNCATE TABLE Employee;

- The rollback process is not possible after truncate table statement. Once you truncate a table you cannot use a flashback table statement to retrieve the content of the table.

SQL COPY TABLE

- If you want to copy a SQL table into another table in the same SQL server database, it is possible by using the select statement.

- The syntax of copying table from one to another is given below:

- SELECT * INTO <destination_table> FROM <source_table>
- For example, you can write following command to copy the records of hr_employee table into employee table.

- SELECT * INTO admin_employee FROM hr_employee;
- SELECT INTO is totally different from INSERT INTO statement.

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;

SQL ALTER TABLE

❖ 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

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);
```

SQL ALTER TABLE

SQL ALTER TABLE DROP Column

- The syntax of alter table drop column is given below:
- ALTER TABLE table_name DROP COLUMN column_name;

SQL ALTER TABLE RENAME Column

- The syntax of alter table rename column is given below:
- ALTER TABLE table_name
- RENAME COLOUMN old_name to new_name;

SQL SELECT STATEMENT

- 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 is 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, and Tables indicate the name of tables, we want to retrieve records from.

SQL 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;

SQL SELECT STATEMENT

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:

```
SQL> SELECT first_name FROM student_details;
```

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

```
SQL>select first_name from student_details;
```

SQL SELECT STATEMENT

```
SQL> SELECT first_name, last_name FROM student_details;
```

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

```
SQL> SELECT DISTINCT home_town FROM students;
```

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.Syntax of SQL COUNT statement.

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

SQL SELECT COUNT

SQL> 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(DISTINCT name) FROM employee_table;

❖ It will return the total distinct names of employee_table.

SQL> SELECT COUNT(*) FROM employee_table;

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

SQL> SELECT TOP 2 * FROM employee;

SQL SELECT CLAUSE

Table CUSTOMERS

CUSTOMER_ NAME	AGE	ADDRESS	EXPENDITURE
KAMAL SHARMA	26	GHAZIABAD	6000
ROBERT PETT	23	NEWYORK	26000
SHIKHA SRIVASTAV	22	DELHI	9000

SQL> SELECT column_name FROM table_name ORDER BY coloumn_name DESC;

SQL> SELECT LAST (CUSTOMER_NAME) AS LAST_CUSTOMER FROM CUSTOMERS;

SQL> SELECT ADDRESS **AS** “City" CUSTOMER_NAME **As** "Client", Age, Expenditure F**ROM CUSTOMERS**;

SQL SELECT IN

- ❖ SQL IN is an operator used in a SQL query to 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:

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

Take an example with character values.

SQL>SELECT * FROM students **WHERE** students_name IN (Amit , Raghav, Raje

ev);

SQL>SELECT * FROM marks **WHERE** roll_no IN (001, 023, 024);

SQL SELECT from Multiple Tables

❖ This statement is used to retrieve fields from multiple tables. To do so, we need to use join query to get data from multiple tables.

```
SQL> SELECT orders.order_id, suppliers.name  FROM suppliers
     INNER JOIN orders ON suppliers.supplier_id = orders.supplier_id
     ORDER BY order_id;
```

Let us take three tables, two tables of customers named customer1 and customer2 and the third table is product table.

Customer1 table

Cus_id	Name1
1	Jack
2	Jill

SQL SELECT from Multiple Tables

Customer2 table

Cus_id	Name2
1	Sandy
2	Venus

Product table

P_id	Cus_id	P_name
1	1	Laptop
2	2	Phone
3	P1	Pen
4	P2	Notebook

SQL SELECT from Multiple Tables

SQL>SELECT p. p_id, p.cus_id, p.p_name, c1.name1, c2.name2

FROM product **AS** p LEFT JOIN customer1 **AS** c1 **ON** p.cus_id=c1.cus_id

LEFT JOIN customer2 **AS** c2 **ON** p.cus_id = c2.cus_id ;

SQL WHERE

SQL> UPDATE suppliers

SET supplier_name = 'HP' **WHERE** supplier_name = 'IBM' AND offices = 8;

SQL> DELETE FROM suppliers **WHERE** supplier_name = 'IBM’

AND product = 'PC computers';

SQL> SELECT * FROM suppliers **WHERE** city = 'New York' OR

 available_products >= 250;

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

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

SQL>SELECT supplier_city **FROM** suppliers **WHERE** supplier_name =

'IBM' **ORDER BY** supplier_city **DESC**;

SQL GROUP BY

S.no	Name	AGE	Salary
1	**John**	**24**	**25000**
2	**Nick**	22	22000
3	**Amara**	25	15000
4	**Nick**	22	22000
5	**John**	24	25000

SQL> SELECT NAME, SUM (SALARY) **FROM** Employee
GROUP BY NAME;

NAME	SALARY
John	50000
Nick	44000
Amara	15000

SQL HAVING CLAUSE

SQL> SELECT NAME, SUM(SALARY) **FROM** Employee
GROUP BY NAME HAVING SUM(SALARY)>45000;

Name	SUM(SALARY)
John	50000

- The **GROUP BY** Clause is used to group the rows, which have the same values.
- The **SELECT** statement in SQL is used with the GROUP BY clause.
- In the **Group BY** clause, the SELECT statement can use **constants, aggregate functions, expressions,** and **column names**.
- The **GROUP BY** Clause is called when the HAVING clause is used to reduce the results.

Relational Algebra

- Relational algebra is a procedural query language, which takes instances of relations as input and yields instances of relations as output.
- It uses operators to perform queries. An operator can be either **unary** or **binary**. They accept relations as their input and yield relations as their output.
- Relational algebra is performed recursively on a relation and intermediate results are also considered relations.
- The fundamental operations of relational algebra are as follows:

Relational Algebra

- SELECT(σ)
- Projection(π)
- Rename (ρ)
- Union operation (υ)
- Set Difference (-)
- Intersection
- Cartesian product(X)
- Join Operations
- Inner Join
- Theta Join
- EQUI join
- NATURAL JOIN (⋈)
- OUTER JOIN
- Left Outer Join(A B)
- Right Outer Join (A B)
- Full Outer Join: (A B)

Relational Algebra

SELECTION (σ)

- The SELECT operation is used for selecting a subset of the tuples according to a given selection condition. Sigma(σ)Symbol denotes it. It is used as an expression to choose tuples which meet the selection condition. Select

$\sigma_p(r)$

σ is the predicate

r stands for relation which is the name of the table

p is prepositional logic

Relational Algebra

Example 1

$\sigma_{\text{topic = "Database"}}$ (Tutorials)

Output - Selects tuples from Tutorials where topic = 'Database'.

Example 2

$\sigma_{\text{topic = "Database" AND author = "Korth"}}$(Tutorials)

Output - Selects tuples from Tutorials where the topic is 'Database' and 'author' is Korth.

Example 3

$\sigma_{\text{sales > 50000}}$ (Customers)

Output - Selects tuples from Customers where sales is greater than 50000

Relational Algebra

Projection(π)

- The projection eliminates all attributes of the input relation but those mentioned in the projection list.
- The projection method defines a relation that contains a vertical subset of Relation.
- This helps to extract the values of specified attributes to eliminates duplicate values. (pi) symbol is used to choose attributes from a relation.
- This operator helps you to keep specific columns from a relation and discards the other columns.

Relational Algebra

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

$\Pi_{\text{CustomerName, Status}}$ (Customers)

CustomerName	Status
Google	Active
Amazon	Active
Apple	Inactive
Alibaba	Active

Relational Algebra

Rename (ρ)

Rename is a unary operation used for renaming attributes of a relation.

ρ (a/b)R will rename the attribute 'b' of relation by 'a'.

Relational Algebra

Union operation (U)

- ❖ UNION is symbolized by ∪ symbol. It includes all tuples that are in tables A or in B.
- ❖ It also eliminates duplicate tuples. So, set A UNION set B would be expressed as A ∪ B.
- ❖ For a union operation to be valid, the following conditions must hold:
    - ✓ R and S must be the same number of attributes.
    - ✓ Attribute domains need to be compatible.
    - ✓ Duplicate tuples should be automatically removed.

Relational Algebra

Table A			Table B	
column 1	**column 2**		**column 1**	**column 2**
1	1		1	1
1	2		1	3

A ∪ B gives

Table A ∪ B	
column 1	**column 2**
1	1
1	2
1	3

Relational Algebra

Set Difference (-)

- The result of A - B, is a relation which includes all tuples that are in A but not in B.
- The attribute name of A has to match with the attribute name in B.
- The two-operand relations A and B should be either compatible or Union compatible.
- It should be defined relation consisting of the tuples that are in relation A, but not in B.

Relational Algebra

Table A - B	
column 1	**column 2**
1	2

Intersection

- ❖ An intersection is defined by the symbol ∩
- ❖ A ∩ B
- ❖ Defines a relation consisting of a set of all tuple that are in both A and B. However, A and B must be union-compatible.

Table A ∩ B	
column 1	**column 2**
1	1

Relational Algebra

Cartesian Product(X) in DBMS

- **Cartesian Product in DBMS** is an operation used to merge columns from two relations.
- Generally, a cartesian product is never a meaningful operation when it performs alone.
- However, it becomes meaningful when it is followed by other operations.
- It is also called Cross Product or Cross Join.

Relational Algebra

Table A			**Table B**	
column 1	**column 2**		**column 1**	**column 2**
A	B		C	D
B	C		E	F

Table A X B			
column 1	**column 2**	**Column 3**	**Column 4**
A	B	C	D
A	B	E	F
B	C	C	D
B	C	E	F

Relational Algebra

Join Operations

- ❖ Join operation is essentially a cartesian product followed by a selection criterion.
- ❖ Join operation denoted by ⋈.
- ❖ JOIN operation also allows joining variously related tuples from different relations.

Types of JOIN

Various forms of join operation are:

Inner Joins

- ✓ Theta join
- ✓ EQUI join
- ✓ Natural join

Outer join

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

Relational Algebra

INNER JOIN

In an inner join, only those tuples that satisfy the matching criteria are included, while the rest are excluded. Let's study various types of Inner Joins.

1. Theta Join

❖ **THETA JOIN** allows you to merge two tables based on the condition represented by **theta**.

❖ **Theta joins** work for all comparison operators.

❖ It is denoted by symbol θ.

❖ The general case of **JOIN** operation is called a **Theta join**

Relational Algebra

Notation

R1 $\bowtie_{\theta}$ R2

Consider R1 and R2 are relations having attributes (A1, A2, .., An) and (B1, B2,.. ,Bn) such that the attributes don't have anything in common, that is R1 ∩ R2 = Φ.Theta join can use all kinds of comparison operators.

STUDENT		
SID	Name	Std
101	Alex	10
102	Maria	11

Relational Algebra

SUBJECT	
Class	Subject
10	Math
10	English
11	Music
11	Sports

Student_Detail –
STUDENT $\bowtie_{\text{Student.Std = Subject.Class}}$ SUBJECT

Relational Algebra

Student_detail				
SID	Name	Std	Class	Subject
101	Alex	10	10	Math
101	Alex	10	10	English
102	Maria	11	11	Music
102	Maria	11	11	Sports

2. Equijoin

When Theta join uses only **equality** comparison operator, it is said to be equijoin. The above example corresponds to equijoin.

Relational Algebra

3. Natural Join (⋈)

- ❖ Natural join does not use any comparison operator.
- ❖ It does not concatenate the way a Cartesian product does.
- ❖ We can perform a Natural Join only if there is at least one common attribute that exists between two relations.
- ❖ In addition, the attributes must have the same name and domain.
- ❖ Natural join acts on those matching attributes where the values of attributes in both the relations are same.

Relational Algebra

Courses		
CID	Course	Dept
CS01	Database	CS
ME01	Mechanics	ME
EE01	Electronics	EE

HoD	
Dept	Head
CS	Alex
ME	Maya
EE	Mira

Relational Algebra

Courses ⋈ HoD			
Dept	CID	Course	Head
CS	CS01	Database	Alex
ME	ME01	Mechanics	Maya
EE	EE01	Electronics	Mira

OUTER JOIN

Theta Join, Equijoin, and Natural Joins are called inner joins. An inner join includes only those tuples with matching attributes and the rest are discarded in the resulting relation. Therefore, we need to use outer joins to include all the tuples from the participating relations in the resulting relation. There are three kinds of outer joins − left outer join, right outer join, and full outer join.

Relational Algebra

1. Left Outer Join(R Left Outer Join S)

All the tuples from the Left relation, R, are included in the resulting relation. If there are tuples in R without any matching tuple in the Right relation S, then the S-attributes of the resulting relation are made NULL.

Left	
A	B
100	Database
101	Mechanics
102	Electronics

Relational Algebra

Right	
A	B
100	Alex
102	Maya
104	Mira

⟖ Courses HoD			
A	B	C	D
100	Database	100	Alex
101	Mechanics	---	---
102	Electronics	102	Maya

Relational Algebra

Right Outer Join: (R Right Outer Join S)

All the tuples from the Right relation, S, are included in the resulting relation. If there are tuples in S without any matching tuple in R, then the R-attributes of resulting relation are made NULL.

Courses ⟖ HoD			
A	B	C	D
100	Database	100	Alex
102	Electronics	102	Maya
---	---	104	Mira

Relational Algebra

Full Outer Join: (R Full Outer Join S)

- ❖ All the tuples from both participating relations are included in the resulting relation.
- ❖ If there are no matching tuples for both relations, their respective unmatched attributes are made NULL.

⟗ Courses HoD			
A	B	C	D
100	Database	100	Alex
101	Mechanics	---	---
102	Electronics	102	Maya
---	---	104	Mira

References

1. Date C J, “An Introduction To Database System”, Addision Wesley.
2. Korth, Silberchatz, Sudarshan, “Database Concepts”, McGraw Hill.
3. https://www.javatpoint.com/dbms-tutorial
4. https://www.tutorialspoint.com/dbms/index.htm