Unit - 2

Unit-2

Syllabus

- Relational model
- Introduction Relational model
- Concepts of domain, tuple, relation.
- Importance of NULL values.
- Constraints (Domain, key, integrity constraints) and their importance.
- Basic SQL
- Simple database schema.
- Data types.
- table definitions (create, alter)
- different DML operations (insert, delete, update)
- Basic SQL Querying (select & project) using where clause,
- Arithmetic and logical operations.
- SQL functions (Date & time, Numeric, String functions).

Important Questions:

** 1. Explain Relational model, domain, tuple, relation and null values.

*** 2. Explain Constraints in DBMS.

* 3. What is SQL? and Explain about simple database schema.

** 4 Explain different data types in SQL and create a table perform the different operations.

** 5. Explain SQL Querying using where clause

* 6. Explain Arithmetic and logical operations.

** 7. Explain SQL functions.

* Unit - 2 *

Introduction to Relational Model :-

→ Relational Model was proposed by E.F codd. to model data in the form of relations or tables. After designing the conceptual model of database using ER diagram, we need to convert the conceptual model in the relational model which can be implemented using any RDBMS (Relational database Management system) like SQL, My SQL etc

Relational model :-

→ Relational model represents how data is stored in relational databases.

→ A Relational database stores data in the form of relations (tables).

→ Consider a relation STUDENT with attributes ROLL-NO, NAME, ADDRESS, PHONE and AGE shown in table.

STUDENT

ROLL-NO	NAME	ADDRESS	PHONE	AGE
1.	Nishma	Hyderbad	9455123451	28
2.	Sai	Guntur	9652431843	27
3.	Swetha	Nellore	9156253131	26
4.	Raji	Ongole	9215635311	25

* **Attribute:** Attributes are the properties that define a relation

Eg: ROLL-NO, NAME

* **Tuple:** Each row in the relation is known as tuple.

→ The above relation contain 4 tuples.

Eg: 1 Nishma Hyderbad 9455123412 28

* **Degree:** The Number of attributes in the relation is known as degree of the relation.

→ The STUDENT Relation defined above has degree 5

* **Column:** column represents the set of values for a particular attribute.

→ The column ROLL-NO is extracted from relation STUDENT

Eg: ROLL-NO

1.

* **NULL values:** the value which is not known or unavailable is called NULL value. it is represented by blank space.

Eg: PHONE of STUDENT having ROLL-NO 4 is NULL

Concept of Domain:

→ The domain of a database is the set of all allowable values (or) attributes of the database

Eg: Gender (Male, Female, others).

Attribute: Attributes are the properties that define a relation.

Eg: Roll-No, name

Tuples: Each row in the relation is known as tuple

Eg: Consider a relation employee with attributes Name, Id, Salary

employee

Name	Id	Salary
Nishma	501	20000
Sai	502	20500
Swetha	503	30000

In the above table

→ Nishma 501 20000

→ Sai 502 20500

→ Swetha 503 30000 are the tuples.

Relation:

→ A Relation is defined as a set of tuples that have same attributes. A relation Consist of relation schema and relation instance.

Relation Schema: A Relation schema represents the name of the relation with it's attributes.

Eg: STUDENT (ROLL-NO, NAME, ADDRESS, PHONE and AGE) is relation schema for STUDENT

Relation instance: The set of tuples of a relation at a particular instance of time is called as relation instance.

→ STUDENT table shows the relation instance of STUDENT at a particular time. It can change whenever there is insertion, deletion or updation in the database.

→ An instance of 'employee' relation.

Fields (columns / attributes).

Emp-code	Emp-name	Dept-name → Field names.
01234	John	HR
12567	Smith	Sales
21678	Sai	Production
12456	Jay.	design.

Tuples.

Fig: Instance of an 'employee' relation.

Importance of NULL values:-

→ SQL supports a special value known as NULL which is used to represent the values of attributes that may be unknown or not apply to a tuple.

→ For example, the Apartment-number attribute of an address applies only to addess that are in apartment buildings and not to other types of residences.

→ It is important to understand that a NULL value is different from zero value.

→ A NULL value is used to represent a missing value, but that it usually has one of three different interpretations.

- Value unknown (value exist but is not known)
- Value not available (exists but it is purposely with held)
- Attribute not applicable (undefined for this tuple).

→ It is often not possible to determine which of the meanings is intended.

Constraints :-

→ on modeling the design of the Relational database we can put some rules (conditions) like what values are allowed to be inserted in the relation.

→ Constraints are the rules enforced on the data columns of a table. these are used to limit the type of data that can go in to a table.

→ This ensures the accuracy and reliability of the data in the database. Constraints could be either on a column level or a table level.

Domain constraints in DBMS :-

→ In DBMS table is viewed as a combination of rows and columns.

→ for example, if you are having a column called month and you want only [Jan, Feb, March...] as values allowed to be entered for that particular column. which is referred to as domain for that particular column.

Definition: Domain constraints ensures two things.

It makes sure that the data value entered for that particular column matches with the data type defined by that column.

It shows that the constraints (NOT NULL / UNIQUE / PRIMARY KEY / FOREIGN KEY / CHECK / DEFAULT)

Domain Constraints

Roll_NO	Name	Age
1	Arya	21
2	Bravo	19
3	John	24
4	Max.	24

→ domain.

Age must be greater than 18 and age must be integer

check (age >= 18)

→ domain constraint = data type check for the column + constraints

Example:

→ For example, we want to create a table "student" with "stu_id" field having a value greater than 100, can create a domain and table like this:

> create domain id_value int constraint id_test check (value > 100);

> create table student (stud_id id_value primary key, stu_name Varchar(30), stu_age int);

Key constraints in DBMS:

→ Constraints or nothing but the rules that are to be followed while entering data in to columns of the database table.

→ Constraints ensure that data entered by the user into columns must be with in the criteria specified by condition.

→ We have 6 types of key constraints in DBMS.

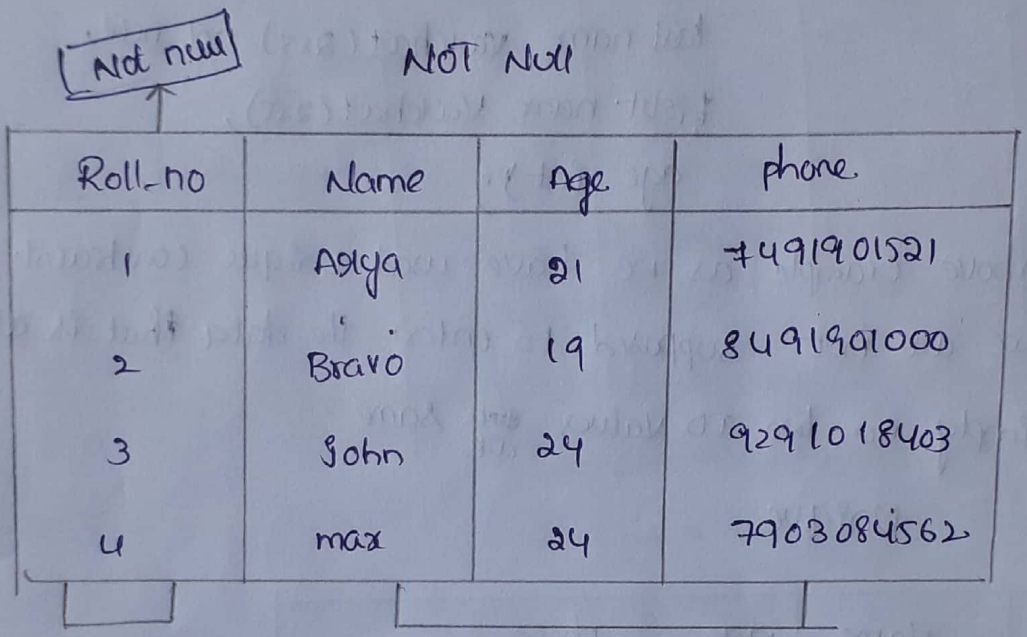
1. Not null
2. unique.
3. default
4. check.
5. primary key
6. Foreign key.

1 Not null:

→ Null represents a record where data may be missing data or data for that record may be optional.

→ Once not null is applied to a particular column, you can not enter null values to that column.

→ A not-null constraint can not be applied at table level.



Example:

```
> create table student ( id int NOT null ,
                         name varchar NOT null,
                         Age int not null,
                         Address char (25),
                         salary decimal (18, 2),
                         primary key (id));
```

→ In the above ex, we have applied not null on three columns id, name and age which means when ever a record is entered using insert statement all three columns should contain a value other than Null

→ We have two other columns address and salary, where not null is not applied which means that you can leave the row as empty.

2. **Unique:** Some times we need to maintain only unique data in the column of a database table, this is possible by using a unique constraint.

→ Unique constraint ensures that all values in a column are unique.

Example:

```
> Create table persons ( Id int unique,
                         last name varchar(255) not null,
                         First name varchar(255),
                         age int );
```

→ In the above example, as we have used unique constraint on ID column we are not supposed to enter the data that is already present, simply no two ID values are same

Unique Unique

Roll_no	Name	age	phone
1	Arya	21	7491901512
2	Bravo	19	8491901000
3	John	24	9291084403
4	Max	24	7903084562

All values are different

Allow duplicate values.

3. **Default:** default in SQL is used to add default data to the columns.

→ When a column is specified as default with some value then all the rows will use the same value. i.e each and every time while entering the data we need not enter that value.

→ But default column value can be customized i.e it can be overridden when inserting a data for that row based on the requirement

Default in DBMS

ID_NO	name	company
1	Arya	abc
2	Bravo	abc
3	John	abc
4	Max	abc

Row with default value "abc"

Example:

```
> Create table emp ( id int notnull,
                     lastname Varchar(255) Not null,
                     first name Varchar(255),
                     age int,
                     city Varchar(255) Default 'hydesbad');
```

→ As a result, when ever you insert a new row each time you need not enter a value for this default column that is entering a column value for a default column is optional.

4. check:

→ check constraint ensures that the data entered by the user for that column is with in the range of values or possible values specified.

Example:

```
> create table student( id int,
                        name Varchar(255),
                        age int,
                        check (age>=18));
```

check

Roll-no	name	age
1	Asiya	21
2	Bravo	19
3	John	24
4	max	24.

→ check.

Allow data only if age >= 18.

check (Age>=18)

→ As we have used a check constraint as (Age>=18) which means value entered by user for this age column while inserting the data must be less than or equal to 18

5. primary key:

→ A primary key is a constraint in a table which uniquely identifies each row record in a database table. by enabling one or more the column in the table as primary key.

primary key

primary key

Roll_no	Name	age	GPA
1	Asiya	21	4
2	Bravo	19	3
3	john	24	4.3
4	max	24	1

Creating a primary key

→ A particular column is made as a primary key column by using the primary key keyword followed with the column name.

> Create table emp (id int,
name varchar (20),
age int,
course varchar (10),
primary key (ID));

→ Here we have used the primary key on ID column then ID column must contain unique values i.e one ID can not be used for anthor student

6. Foreign key:

→ The foreign key constraint is a column or list of columns which points to the primary key column of anthor table.

→ the main purpose of the foreign key is only those values are allowed in the present table that will match to the primary key column of anthor table

Foreign key.

student_details

Roll_no	name	course_id
1	Arya	IT-101
2	bravo	IT-102
3	John	IT-103

→ key

student_marks

course_ID	Marks
IT-101	100
IT-102	99
IT-103	93

Primary Foreign key. ←

primary key

Example:

Reference table:

```
> create table customers1 ( id int ,
                            name varchar (20),
                            city varchar (10),
                            primary key (id));
```

child table:

```
> create table customers2 ( id int ,
                            marks int ,
                            references customers1 (id));
```

Integrity Constraints in DBMS:

→ there are two types of integrity Constraints.

1. Entity Integrity constraint
2. Referential Integrity Constraint

1. Entity integrity constraint:-

→ Entity integrity constraint is used to ensure the uniqueness of each record or row in the data table

→ Entity integrity constraints says that no primary key can take NULL value, since using primary key we identify each tuple uniquely in a relation.

Eg:

E id	Name	phone
01	Sony	7002494274
02	Pinky	60265 26747
NULL	lalli	92345 67892

Explanation:

→ In the above relation, EID is made primary key, and the primary key cant take NULL values. but in the third tuple, the primary key is NULL, so it is Violating Entity integrity constraint.

2. Referential Integrity constraint :-

→ The Referential integrity constraint is specified between two relations or tables. and used to maintain the consistency among the tuples in two relations.

→ This constraint is enforced through foreign key, when an attribute in the foreign key of relation R_1 have the same domain as the primary key of relation R_2, then the foreign key of R_1 is said to reference or refer to the primary key of relation R_2

→ The values of the foreign key in a tuple of relation R_1 can either take the values of the primary key for some tuple in relation R_2, or can take NULL values, but can't be empty.

Example:

EID	Name	DNO
01	Divin	12
02	Dino	22
04	Vivi	14

DNO → Foreign key.

primary key ←

DNO	place
12	Jaipur
13	mumbai
14	Delhi

Explanation:

→ In the above, DNO of the first relation is the foreign key. and DNO in the second relation is the primary key.

→ DNO = 22 in the foreign key of the first relation is not allowed since DNO = 22

→ is not defined in the primary key of the second relation. therefore Referential integrity constraints is violated here.

Basic SQL :-

→ SQL stands for structured Query language. it is used for storing and managing data in relational database management system

→ It is standard language for relational database system. it enables a user to create, read, update and delete relational databases and tables.

→ All the RDBMS like MySQL, Oracle, MS Access and SQL server use SQL as their standard database language

→ SQL allows users to query the database in a number of ways using English like statements

Rules : SQL follows following rules :

→ structured Query language is not case sensitive. Generally keywords of SQL are written in uppercase

→ Using the SQL statements, you can perform most of the actions in a database.

→ statements of SQL are dependent on text lines. We can use a single SQL statement on one or multiple text line.

SQL process :

→ When an SQL command is executing for any RDBMS, then the system figure out the best way to carry out the request and the SQL engine determines that how to interpret the task.

→ in the process, various components are included. these components can be optimization engine, query engine, query dispatcher etc.

→ All the non-SQL queries are handled by the classic query engine, but SQL query engine won't handle logical files.

characteristics of SQL:

→ SQL is easy to learn

→ SQL is used to access data from relational database management system.

→ SQL is used to describe the data

→ SQL is used to create and drop the database and table

→ SQL allows users to set permissions on tables, procedures, and views.

Simple database schema:-

→ A database schema is a structure that represents the logical storage of the data in a database.

→ it represents the organization of data and provides information about the relationships between the tables in a given database

→ A database schema is the logical representation of a database, which shows how the data is stored logically in the entire database.

→ It contains list of attributes and instruction that informs the database engine that how the data is organized and how the elements are related to each other.

→ A database schema contains schema objects that may include tables, fields, packages, views, relationships, primary key, Foreign key.

→ In actual, the data is physically stored in files that may be in unstructured form, but to retrieve it and use it, we need to put it in a structured form. To do this a database schema is used. it provides knowledge about how the data is Organized in a database and how it is associated with other data.

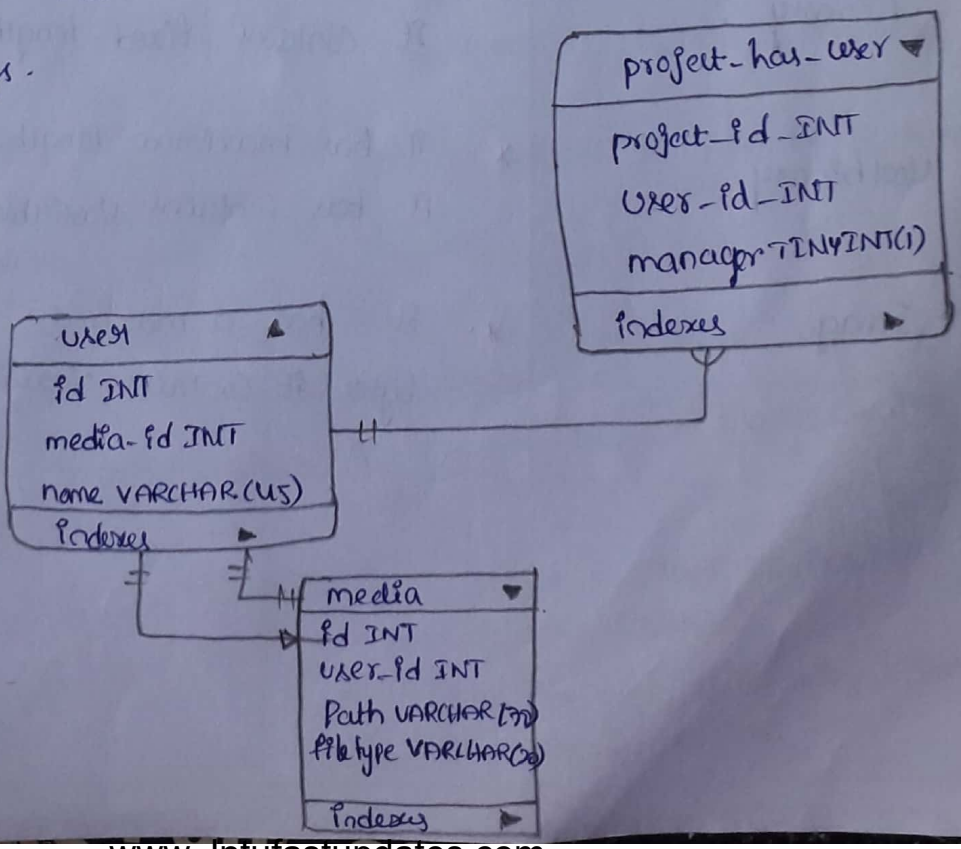
→ A database schema object includes the following:

→ Consistent formatting for all data entries.

→ Database objects and unique keys for all data entries

→ tables with multiple columns, and each column contains it's name and datatype.

→ The given diagram is an example of a database schema it Contains three tables, their data types. this also represents the relationships between the tables and primary keys as well as foreign keys.

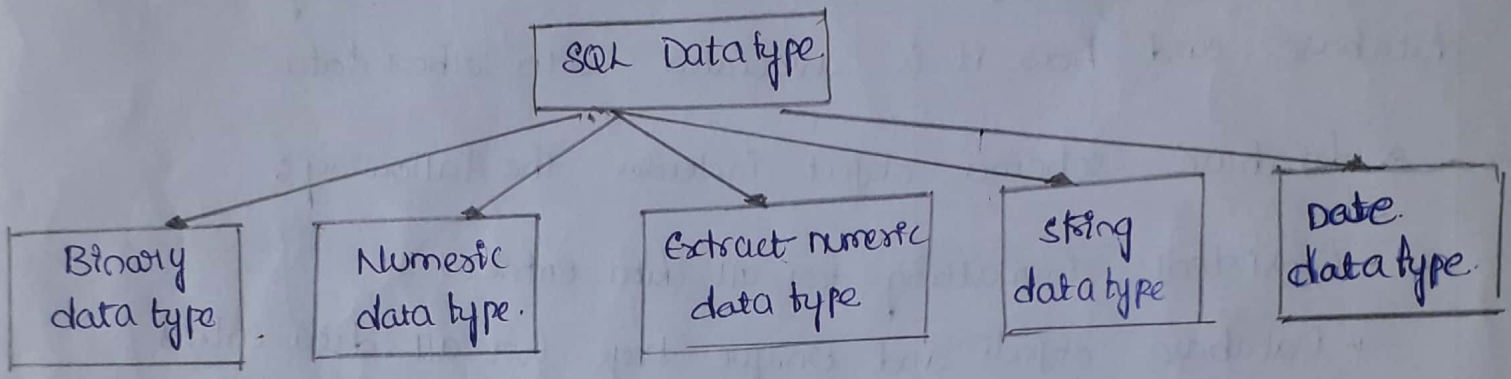


SQL Data type :-

SQL Data type 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

Data types of SQL :



1. Binary data types :-

There are three types of binary data types 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 maximum length of 8000 bytes It has Contains Variable-length binary data
* Image	* it has a maximum length of 2,147,483,647 bytes. it Contains Variable length binary data

2. Numeric Data type:

Data type	from	to	Description
* float	-1.79E +308	1.79E +308	* It is used to specify a floating-point value eg: 6.2, 2.9 etc
* real	-3.40E +38	3.40E +38	* It specifies a single precision floating point number.

3. Exact Numeric Data type:-

Data type	Description
* int	* It is used to specify an integer value.
* small int	* 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.

4. Date and time data types:

Data type	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
* time stamp	* It stores the year, month, day, hour, minute, and the second value

5. String datatype :

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.

table definition : (create, alter) :-

SQL table: SQL table is a collection of data which is organized in terms of rows and columns.

→ In DBMS, the table is known as relation and row as a tuple

→ Let's see an example of the "EMPLOYEE" table.

EMP-ID	EMP-NAME	CITY	PHONE-NO
1	Kristen	Washington	7289201223
2.	Anna	Franklin	9378282882
3	Jackson	California	9264783838
4	Daniel.	Hawai	9638482678.

→ In the above table, "EMPLOYEE" is the table name, "EMP_ID", "EMP_NAME", "CITY", "PHONE_NO" are the column names.

→ The combination of data of multiple columns forms a row.

eg: 1, "Kolisten", "Washington" and "7289201223" are the data of one row.

Operations on table:

1. Create table
2. Alter table
3. Drop table

1. Create table: SQL create table is used to create a table in the database. To define the table you should define the name of the table and also define it's column and column's data type.

Syntax:

```
create table table_name ("column1" "datatype",
                         "column2" "datatype",
                         "column3" "datatype",
                         :
                         "columnN" "datatype");
```

Example:

```
SQL> create table employee ( emp_id int,
                             emp_name varchar(25),
                             phone_no int,
                             Address. char(30));
```

→ If you create the table successfully, you can verify the table by looking at the message by the SQL server. else you can use DESC command as follows:

SQL > DESC employee;

Field	type	NULL	Default	Extra
emp_id	Int(11)	NO	NULL	
emp_name	Varchar(25)	NO	NULL	
phone_no	NO	Int(11)	NULL	
Address	YES		NULL.	char(30)

Alter table:

→ The Alter table command adds, delete, or modifies columns in a table.

→ The alter table command also adds and deletes various constraints in a table.

→ The following SQL adds an "Email" column to the "employee" table:

Syntax: Alter table table name add column1 datatype;

Example: Alter table employee add email varchar(255);

SQL > DESC employee;

Field	type	NULL	default	extra.
emp_id	Int(11)	NO	NULL	
emp_name	Varchar(25)	NO	NULL	
phone_no	NO	Int(11)	NULL	
Address	YES		NULL	char(30)
email	Varchar(255)		NULL	

3. Drop table:

→ The drop table command deletes a table in the database.

→ The following example SQL deletes the table "employee"

Syntax: drop table tablename;

Example: drop table employee;

→ Dropping a table results in loss of all information stored in the table.

Different DML operations: (insert, delete, update)

→ DML - Data Manipulation Language.

→ Data Manipulation commands are used to manipulate data in the database.

→ Some of the data Manipulation commands are.

1. insert
2. update
3. delete

1. insert: SQL insert statement is a SQL query. it is used to insert a single or multiple records in a table.

Syntax: insert into table name values (value 1, value 2, value 3);

let's take an example of table which has 3 records within it.

> insert into student values ('Alekhya', 501, 'Hyderbad');

> insert into student values ('Deepthi', 502, 'Guntur');

> insert into student values ('Ramya', 503, 'Nellore');

The fallowing table will be as fallow:

Name	Id	city.
Alekhya	501	Hyderbad
Deepthi	502	Guntur
Ramya	503	Nellore

2. Update:

→ The SQL commands update are used to modify the data that is already in the database. ~~the SQL~~

→ SQL update statement is used to change the data of the records held by tables. which row is to be update, it is decided by condition. to specify condition, we use WHERE clause

→ The update statement can be written in fallowing form:

Syntax: update table_name set column_name = expression where condition;

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

> update students set name = 'Rasi' where id = 503;

→ After update the table.

Name	id	city
Alekhya	501	Hyderbad
Deepthi	502	Guntur
Rasi	503	Nellore

3. delete :-

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

Syntax:

→ Delete from table_name [where condition];

Example:

→ Let us take a table, named "student" table

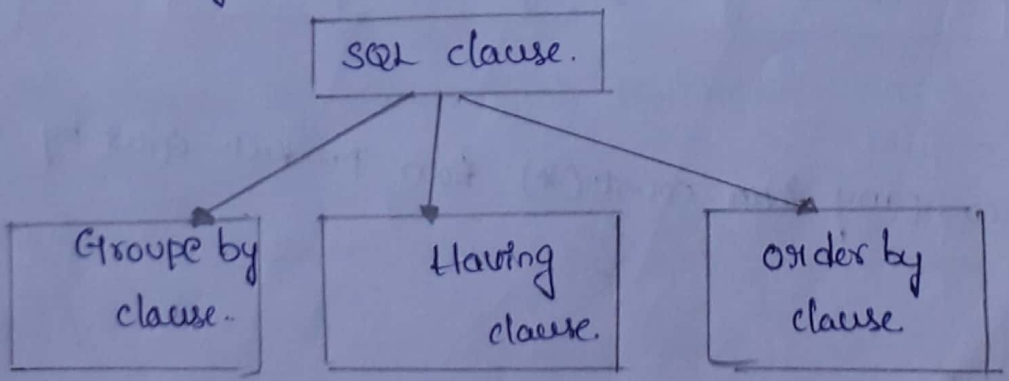
> Delete from students where id = 501;

→ Resulting table after the query:

Name	id	city.
Deepthi	502	Guntur
Rani	503	Nellore

Basic SQL querying (select and project) using where clause :-

→ The following are the various SQL clauses:



1. Group by:

→ SQL Group by statement is used to arrange identical data in to groups.

→ the Group by statement is used with the SQL select statement.

→ the Group by statement follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.

Syntax:

```
Select column from table_name where condition
                                Group by column
                                order by column;
```

Sample table: product.

product	company	QTY	state	cost
Item 1	com 1	2	10	20
Item 2	com 2	3	25	75
Item 3	com 1	2	30	60
Item 4	com 3	5	10	50
Item 5	com 2	2	20	40

example:

> Select company from count(*) from product Group by company;

out put:

com 1 2

com 2 3

com 3 5.

2. Having clause:

→ Having clause is used to specify a search condition for a group or an aggregate.

→ Having is used in a GroupBy clause. if you are not using Group by clause then you can use Having function like a where clause

Syntax:

```
Select column1, column2 from table_name
where conditions
Group by column1, column2
having conditions
order by column1, column2;
```

Example:

```
> Select company, count(*) from product
  group by company
  having count(*) > 2;
```

output:

```
com3    5
com2    3.
```

3. order by clause:

→ The order by clause sorts the result-set in ascending or descending order.

Syntax:

```
Select column1, column2 from table_name
where condition
order by column1, column2... ASC;
```

Sample table: take a student table.

Example: Select * from student order by name;

output:

Name	Id	City
Akshya	501	Hyderbad
Deepthi	502	Guntur
Rasi	503	Nellore

SQL where :-

→ A where clause in SQL is a data manipulation language statement.

→ Where clauses are not mandatory clauses of SQL DML statements. But it can be used to limit the number of rows affected by a SQL DML statement or returned by query.

→ Actually it filters the records. It returns only those queries which fulfill the specific conditions.

Syntax:

```
Select column1, column2, ... columnn from table_name
    where [condition];
```

→ Where clause uses some conditional selection.

=	equal
>	greaterthan
<	less than
>=	greater than or equal
<=	less than or equal
<>	not equal to

Arithmetic and logical operations:

SQL operators:

→ SQL statements generally contain some reserved words or characters that are used to perform operations such as Arithmetic and logical operations e.t.c. these reserved words are known as operators.

SQL Arithmetic operators:

→ We can use various Arithmetic operators on the data stored in the tables.

→ Arithmetic operators are:

+	Addition
-	Substraction
/	Division
*	Multiplication
%	Modulus.

1. **Addition (+):** It is used to perform addition operation on the data items.

Sample table:

emp-id	emp-name	salary
1	alex	25000
2	John	55000
3	daniel	52000
4	Sam	12312.

> select emp-id, emp-name, salary, salary+100 as "salary+100" from addition;

output:

emp_id	emp_name	salary	salary+100
1	alex	25000	25100
2	John	55000	55100
3	daniel	52000	52100
4	sam	12312	12412.

→ Here we have done addition of 100 to each emp's salary.

2. subtraction (-):

→ It is use to perform subtraction operation on the data items.

ex: > select emp_id, emp_name, salary, salary-100 as "salary-100" from subtraction;

output:

emp_id	emp_name	salary	salary-100
1	alex	25000	24,900
2	John	55000	54,900
3	daniel	52000	51,900
4	sam	90000	89,900.

Here we have done substraction of 100 to each emp salary.

3. division (/):

→ the div function is used to integer division (x is divided by y) An integer value is returned.

ex: > select emp_id, emp_name, salary, salary/100 as "salary/100" from division;

emp_id	emp_name	salary	salary/100
1	alex	25000	250
2	John	55000	550
3	daniel	52000	520

4. Multiplication (*) :-

→ It is use to perform multiplication of data items.

> select emp_id, emp_name, salary, salary * 100 as "salary *100" from ~~addition~~ multiplication;

output:

emp-id	emp-name	salary	salary * 100
1	alex	25000	2,500,000
2	John	05000	5,500,000
3	daniel	52000	5,200,000
4	sam	90000	9,000,000

→ Here we have done multiplication of 100 to each emp salary.

5. Modulus :

→ It is use to get remainder when one data is divided by anthor.

> select emp_id, emp_name, salary, salary %o 25000 as "salary %o 25000" from ~~addition~~ modulus;

output:

emp_id	emp_name	salary	salary %o 25000
1	alex	25000	0
2	John	55000	5000
3	daniel	52000	2000
4	sam	90,000	15000

→ Here we have done Modulus operation to each emp salary.

Logical operations:

→ A logical operators allows you to test for the truth of a condition.

→ The following table illustrates the SQL Logical operators.

Operator	meaning.
ALL	* Return true if all comparisons are true
AND	* Return true if both expressions are true.
ANY	* Return true if any one of the comparision is true
BETWEEN.	* Return true if the operand is within a range.
IN.	* Return true if the operand is equal to one of the value in a list
EXISTS	* Return true if the sub query contains any rows

1. AND: The AND operator allows you to construct multiple conditions in the WHERE clause of an SQL statement such as select.

→ The following example finds all employees whose salaries are greater than 5000 and less than 7,000

> Select first_name, last_name, salary from employees where salary >5000 AND salary <7000 order by salary;

output:

first_name	last_name	salary
John	Wesly.	6000
eden	daniel	6000
Loies	popp	6900
Shanta	Suji	6500

2. ALL: The ALL operator compares a value to all values in anthor value set.

→ The fallowing example finds all employees whose salaries are greater than all salaries of employees

> Select first_name, last_name, salary from employees where salary >= ALL (select salary from employees where department_id = 8) order by salary DESC;

output:

first_name	last_name	salary
Steven	king	24000
John	Russel.	17000
Neena	kochhar.	14000

3. ANY: The ANY operator compares a value to any value in a set according to condition.

The fallowing example statement finds all employees whose salaries are greaterthan the average salary of every department.

> Select first_name, last_name, salary from employees where salary > ANY (Select Avg(salary) from employees group by department_id) order by first_name, last_name;

Output:

first_name	last_name	salary
Alexander	Hunold	9000.00
charles	johnson	6200.00
david	Austin	4800.00
eden	Flop	9000.00

4. Between: the between operator searches for values that are with in a set of values.

→ For example, the following statement finds all employees whoose salaries are between 9000 and 12000.

> Select first_name, last_name, salary from employees where salary between 9000 AND 12000 order by salary;

output:

First_name	last_name	salary
Alexander	Hunold	9000.00
Den	Rechards	10000.00
Nancy	prince	12000.00

5 IN: The IN operator compares a value to a list of specified values. The IN operator returns true if the compared value matches at least one value in the list.

→ the following statement finds all employees who work in the department id 8 or 9.

> Select first_name, last_name, department_id from employees where department_id IN (8,9) order by department_id;

output:

First_name	last_name	department_id.
John	Russel.	8
Jack	livingstone	8
steven	king	9
Neena	kochhar	9

6 EXISTS:- The EXISTS operator tests if a subquery contains any rows

→ for example, the following statement finds all employees who have dependents

> select first_name, last_name from employees e where EXISTS (select 1 From dependent d where d.employee_id = e.employee_id);

First_name	last_name
steven	king
Neena	kochhar
Alexander	hunold.

SQL Functions: (Date and time, Numeric, string conversion):-

→ Function is a database object in SQL server. Basically, it is a set of SQL statements that accepts only input parameters, perform actions and return the result.

SQL Date and time functions:-

→ The date and time functions in DBMS are quite useful to manipulate and store values related to date and time.

→ The different date and time functions are.

1. ADD DATE (DATE, DAYS):-

→ The number of days in integer form (DAYS) is added to the specified date. this is the value returned by the function.

Ex:

> Select adddate ('2018-08-01', 31);

output:

2018-09-01

→ this functions adds 31 days to the given date.

2. ADD TIME:-

→ this function adds the two expressions exp1 and exp2 and displays the result. in this case exp1 can be a datetime or time expression and exp2 is a time expression

Ex: > Select addtime ('2018-08-01 23:59:59.999', '1 1:1:1.0002');

output: 2018-08-02 01:01:01.00001

→ the time '1 1:1:1.0002' is added to the datetime function '2018-08-01 23:59:59:999' to give the resultant value '2018-08-02 01:01:01.00001'.

3. curdate:

→ This returns the current date of the system in the YYYY-MM-DD format.

Ex:

> select curdate();

output: 2021-05-07.

→ This function returns the current date. '2021-05-07'.

4. curtime:

→ This returns the current time of the system from the current time zone in the format HH:MM:SS.

Ex:

> select curtime();

output: 10:56:35

→ This function returns the current time.

5. dayName (Date);

→ For the given date, this function returns the corresponding day of the week.

Ex: > select dayname ('2021-05-07');

output: Friday.

→ This function returns the day of the week.

6. dayofmonth (date);

→ For the given date, it returns the day of the month the date is on.

Ex: > select dayofmonth ('2021-05-07');

output: 05

→ This returns the day of the month.

7. dayofweek :-

→ For the given date, it returns the day of the week the date is on.

Ex:- >select dayofweek ('2021-05-07');

output: 5

8. date Time :-

→ this function displays the time part of a time or date time expression in the form of a string

Ex: > select time ('2021-05-07 10:33:25');

output: 10:33:25

SQL string Conversions:

→ String functions are used to perform an operation on input string and return the output string. following are the string functions.

1. concat(): This function is used to add two words (or) strings

Ex: > select 'database' || ' ' || 'Management system' from dual;

output: 'database Management system'

2. Instr(): This function is used to find the occurance of an alphabet.

Ex: > Instr ('database system', 'a');

o/p: 2 (the first occurance of 'a')

3. lower(): This function is used to convert the given string into lowercase

Ex: > select lower ('DATABASE');

o/p: database

4. upper(): this function is used to convert the Lower case string into uppercase.

Ex: > select upper ('database');

o/p: DATABASE

5. lpad(): this function is used to make the given string of the given size by adding the given symbol

Ex: > lpad ('system', 8, '0');

o/p: 00system.

6. rpad(): this function is used to make the given string as long as the given size by adding the given symbol on the right.

Ex: > rpad ('system', 8, '0');

o/p: system00

7. ltrim(): this function is used to cut the given substring from the original string.

Ex: > ltrim ('database', 'data');

o/p: base.

8. rtrim(): this function is used to cut the given substring from the Original string

Ex: > rtrim ('database', 'base');

o/p: data.

SQL Numeric functions:

→ Numeric functions are used to perform operations on numbers and return numbers.

→ Following are the numeric functions

1. **ABS()**: It returns the absolute value of a number.

   Ex: > select ABS(-243.5);

   o/p: 243.5

2. **ACOS()**: It returns the cosine of a number.

   Ex: > select ACOS(0.25);

   o/p 1.31811607165 2818

3. **ASIN()**: It returns the Arc sine of a number.

   Ex: > select ASIN(0.25);

   o/p: 0.25268025514 2

4. **CEIL()**: It returns the smallest integer value that is a greater than or equal to a number

   Ex: > select CEIL(25.75);

   o/p: 26.

5. **FLOOR()**: It returns the largest integer value that is a less than or equal to a number.

   Ex: > select floor(25.75);

   o/p: 25

6. **TURNCATE()**: this does not work for SQL server. It returns the truncated to 2 places right of the decimal point

   Ex: select TRUNCATE(7.53635, 2);

   o/p: 7.53.

//.