Select commands

SELECT Statements – Syntax

- The SELECT statement is used to retrieve data from a table.
- The data returned is stored in a result table, is known as the result-set.

Syntax:

```
SELECT column1, column2, … FROM table_name;
```

SELECT Statement - Syntax

- column1, column2, ... are the column names of the table we want to select data from

- To select all the fields available in the table , follow the given syntax

```
SELECT * FROM table_name;
```

SELECT Statement - Data

Here are selections from "Customers" table in the “company” database:

CustomerID	FirstName	LastName	Country
1	Mike	Christensen	USA
2	Andy	Hollands	Australia
3	Rahul	Vedantam	India
4	Jeevan	Sharma	India

SELECT Column - Example

- Below statement selects the "first_name“ and “country” columns from the "Customers" table:

```
SELECT first_name, Country FROM Customers;
```

Output:

FirstName	Country
Mike	USA
Andy	Australia
Rahul	India
Jeevan	India

Selecting all Columns - Example

- Below statement selects the "first_name"and “country” columns from the "Customers" table:

```
SELECT * FROM Customers;
```

Output:

CustomerID	FirstName	LastName	Country
1	Mike	Christensen	USA
2	Andy	Hollands	Australia
3	Rahul	Vedantam	India
4	Jeevan	Sharma	India

WHERE Clause - Syntax

- WHERE clause is used to filter records
- WHERE clause is also used to return only those records that fulfill a specified condition

Syntax:

```
SELECT column1, column2, … FROM table_name WHERE condition;
```

Apart from SELECT statement, WHERE Clause is also used in UPDATE, DELETE statement, etc.!

WHERE Clause - Syntax

- Below statement selects all the customers from the country "India", in the "Customers" table:

```
SELECT * FROM Customers WHERE Country='India';
```

Output:

CustomerID	FirstName	LastName	Country
3	Rahul	Vedantam	India
4	Jeevan	Sharma	India

LIKE/NOT LIKE Commands

LIKE/NOT LIKE Clause - Syntax

- The LIKE clause is used to do a pattern match
- It extracts only those records that fulfill a given condition

Output:

```
SELECT column1, column2, … FROM table_name WHERE condition LiKE value;
```

Missing Data

Checking Missing Data

- The SQL NULL represents missing values.
- A NULL value in a table means a blank value in a field.
- IS NULL or IS NOT NULL operators are used to check for a NULL value

Checking Missing Data

Consider the following Employee table having the records as shown below

ID	NAME	AGE	ADDRESS	SALARY
1	Kellie	32	California	2000
2	Pete	25	Texas	1500
3	Popy	23	Boston	2000
4	Sam	25	Florida	
5	John	27	Hawaii	

IS NOT NULL Operator

Syntax:

```
SELECT  ID, NAME, AGE, ADDRESS, SALARY FROM Employee
WHERE SALARY IS NOT NULL;
```

Output:

ID	NAME	AGE	ADDRESS	SALARY
1	Kellie	32	California	2000
2	Pete	25	Texas	1500
3	Popy	23	Boston	2000

IS NULL Operator

Syntax:

```
SELECT  ID, NAME, AGE, ADDRESS, SALARY FROM Employee
WHERE SALARY IS NULL;
```

Output:

ID	NAME	AGE	ADDRESS	SALARY
4	Sam	25	Florida	
5	John	27	Hawaii	

Logical Operators

- The following logical operators can be used in WHERE clause

where Condition1 and condition2
or

Operator	Description
✓ AND	Both Conditions must be satisfied
✓ OR	Any one condition must be satisfied
✓ NOT	Negation of the given condition

Relational Operators

- The following relational operators can be used in WHERE clause

Operator	Description
=	Equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal

!=

ORDER BY Clause

- The Order by clause helps us to sort the data
- The data can be sorted in any order (ascending or descending).

default desc.

Syntax :

```
SELECT column1, column2, … FROM table_name
Order by column1 ( asc/desc);
```

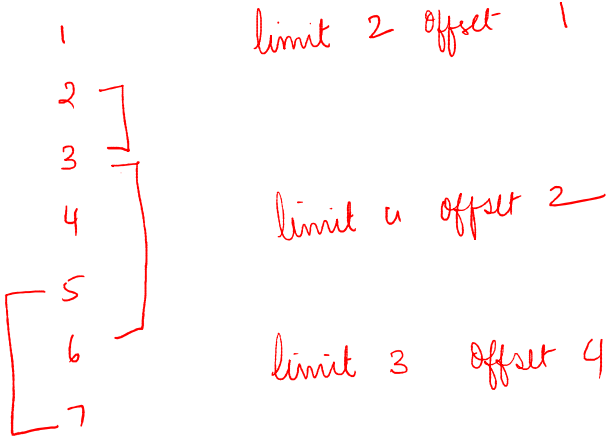
LIMIT/OFFSET Clause

- The Limit by clause is used to limit and print the rows
- Offset mentioned if rows need to be fetched in between

Syntax:

SELECT *column1, column2, …* **FROM** *table_name*
Limit 5 – *fetches first 5 rows*

LIMIT/OFFSET Clause



Arithmetic Operation Clause

- We can perform arithmetic operations on any numerical columns

Syntax:

```
SELECT column1, column2*0.1, … FROM table_name
```