Swami Sahajanand College of Computer Science

B.C.A. SEM-V [NEP]

Subject: DATABASE TECHNOLOGY IN INDIA
Major12 - 26516

UNIT-2

BASIC SQL*PLUS

- ◆ Introduction of SQL, Characteristics of SQL.
- ◆ Basic Data Types of ORACLE, Oracle Operators.
- ◆ Data Definition Language (DDL).
- ◆ Data Manipulation Language (DML).
- ◆ Data Control Language (DCL).
- ◆ Transaction Processing Language (TPL).
- ◆ Query Generation using Clause:
- ◆ Where, Between, Distinct, Like, Order by, IN, NOT IN.

Q-1 What is SQL? Explain history and characteristics of SQL.

❖ **Introduction of SQL:**

- SQL is a language that provides an interface to relational database systems.
- SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in relational database.
- SQL is the standard language for Relation Database System. Many RDBMS like MySQL, MS Access, Oracle, Sybase, and SQL Server use SQL as standard database language.
- SQL is standard tool to retrieve and store data from and to relational database.
- SQL is very popular due to appropriate structure for the client-server architecture.
- Most of the desktop and web application access relational databases using SQL language.
- SQL enables a programmer or database administrator to do the following:
    - Modify a database's structure.
    - Change system security settings.
    - Add user permissions on databases or tables.
    - Query a database for information.
    - Update the contents of a data.

❖ **History of SQL:**

- SQL was developed by IBM in the 1970.
- SQL language was first implemented by Oracle Corporation in the year 1979.
- There was no official standard till 1986.
- ANSI (American National Standard Institute) and ISO (international Standard Organization) jointly published SQL In 1986.
- It was revised again in 1989 to enforce referential integrity.
- As world becoming more connected and heterogeneous computing become more popular, developer and user want some Language to query database.
- ISO developed SQL 92 standard, in this version data is contained in table, tables are grouped into schema and schema are group into catalogs (database). The features are:
    - Schema definition, Temporary tables.
    - Built in join operators.
    - Read Only, Scrollable, and dynamic cursor.
    - Standard connection procedure, Standard error code.
    - Altering and dropping of objects.

❖ **Characteristics of SQL (Features of SQL):**

- Works around Relational Database Management System.
- SQL commands can operate on several tables as single object.
- It can process any amount of data retrieved from these tables.
- SQL use temporary tables store intermediate result. These tables are deleted when they no longer require.
- SQL is well suited for client server environment.

- It provides flexible transaction management.
- One can create constraints for restriction of value that can be placed in table.
- Privileges can be granted or denied using SQL Statements.
- It is a non-procedural language.
- It is an English-like language.

Q-2 Explain Basic Data types of Oracle and Oracle Operators in detail.

❖ **Data types of ORACLE:**

"Data type specifies what type of value we can store in field of table"

- Each and every field in table have a data type.

No.	Data type	Description
1	**number(p,s)**	"number data type is used to store numeric value with decimal point or without decimal point." ➢ Here, p means Precision and s means Scale. ➢ The range of Precision is 1 to 38. The range of Scale -84 to +127. ➢ E.g.: number(7,2) - Here, total 7 digit. 5 digits before the decimal and 2 digits after the decimal.
2	**char(size)**	"char data type is used to store alpha-numeric value." ➢ Where size means the number of characters to store. ➢ Maximum size is 2000 bytes. Fixed-length strings. ➢ Data store in this data type is consider as a text. ➢ E.g.: char(20) - Here, total 20 character we can store.
3	**varchar2(size)**	"varchar2 data type is used to store alpha-numeric value." ➢ Where size means the number of characters to store. ➢ Maximum size is 4000 bytes. Variable-length strings. ➢ Data store in this data types is consider as a text. ➢ E.g.: varchar2(20) - Here, total 20 character we can store.
4	**Date**	"date data type is used to store date and time only." ➢ Store the date in dd-mon-yyyy format. ➢ Store the time in HH:MM:SS ➢ A valid date from 1-jan-4712 B.C to 31-dec-9999 A.D. ➢ E.g.: date
5	**Long**	"long data type is used to store variable length character string." ➢ Store the data up to 2 GB. ➢ Store the arrays of binary data in ASCII format. ➢ It is also called memo data type. ➢ Long values cannot be used in suqueries, functions, expressions etc. ➢ E.g.: long
6	**long row**	➢ It contain binary data up to 2 GB.
7	**Rowid**	➢ Hexadecimal string that represent the unique address of row in table. It is returned by the pseudo-column ROWID.

8	**Bfile**	➢ Represents pointer to a binary system file. It contain data up to 4 GB.
9	**nchar(size)**	➢ Data with fixed size, of type character, and with a maximum size of 2000 byte.
10	**nvarchar2(size)**	➢ Represent variable length character string up to 4000 byte.
11	**Nclob**	➢ Object of type character containing character of type multi-byte. It has fixed size up to 4 GB.
12	**Blob**	➢ A binary object with maximum size of 4 GB
13	**Clob**	➢ A binary object containing characters of type single byte. It allows a maximum size of 4 GB
14	**raw (size)**	➢ Data of type raw binary with maximum size of 2000 byte

❖ **Oracle operators:**

"An operator is a symbol or character that tells the computer to perform certain mathematical or task or function"

⌑ Operators are used to manipulate data.

⌑ Following are the operators available in oracle:

No.	**Type**	**Description**	**Name**	**Value**
1	**Arithmetic**	Operate on numeric operands.	- Division - Multiplication - Addition - Subtraction	/ * + -
2	**Character**	Manipulate strings.	- Concatenation	||
3	**Comparison (Relational)**	Check for similarities and differences.	- Equal to - Not equal to - Greater than - Less than - Greater than or equal to - Less than or equal to - NULL	= != > < >= <= IS NULL
4	**Logical**	Manipulate comparison expression.	- AND - OR - NOT	AND OR NOT
5	**Set**	Combine the results of two queries in one query.	- UNION - INTERSECT - MINUS	UNION INTERSECT MINUS

Q-3 Explain Data Definition Language (DDL) in detail.

- DDL commands are used to create, alter, drop and truncate the database structure/objects.
- Following are the DDL commands:

No.	Commands	Description
1	CREATE	Creates a table.
2	ALTER	Changes/insert column in existing table.
3	TRUNCATE	Remove all the rows from the database.
4	DROP	Remove table from database.

(1) CREATE:

- Create command is used to create table, indexes, view, etc.
- Generally it is used to create database tables.
- Create table command defines each column of the table uniquely.
- Each column has a minimum of three attributes: name, datatype and size.
- Each column definition is separated from the other by a comma.
- The SQL statement is terminated with a semi colon.
- There is no difference between names in lower case letter and name in upper case letters.

:: Syntax ::

```
create table <Table_Name>
(
<Column_Name 1> <data type> (size),
<Column_Name 2> <data type> (size),
 ………………………………………………….
……………………………………………….,
<Column_Name N> <data type> (size)
);
```

:: Example ::

```
create table student
(
    ID number(5),
    NAME varchar2(30),
    DOB date,
    MOBILE_NO number(10)
);
Output: Table created.
```

❖**Rules for Table and Column name:**

- Tables and column names can be up to 30 characters.
- Names must begin with an alphabet.
- Names cannot contain quotes.
- Names are NOT case sensitive.
- Names can contain character, a-z, 0-9, _ (underscore), $ (doller sign), and # (Hash sign).
- Names cannot be reserved words like select, create, insert etc.
- Space is not allowed.

(2) ALTER:

- The structure of a table can be modified by using the ALTER command.
- The ALTER TABLE allows changing the structure of an existing table.
- The ALTER command is used to add, delete or change a column in a table.
- This command is also used change the data type of existing columns or rename column.

1. **Adding new columns[ADD]:**
   Syntax: alter table <TABLE_NAME> ADD (NEW_COLUMN_NAME DATATYPE (SIZE), …);
   Example: alter table student ADD (Gender char(1));

Output: Table altered.
Here, new column Gender is added into the existing table student.

2. **Modify columns[MODIFY]:**
   Syntax: alter table <TABLE_NAME> MODIFY (COLUMN_NAME DATATYPE (NEW_SIZE), …);
   Example: alter table student MODIFY (ID number(7));
   Output: Table altered.
   Note: - If the data is already being entered in the table we cannot modify the data type and we cannot decrease the size of the column. We can only increase the size of the column.

3. **Drop Columns (delete column) [DROP COLUMN]:**
   Syntax: alter table <TABLE_NAME> DROP COLUMN <COLUMN_NAME>;
   Example: alter table student DROP COLUMN Gender;
   Output: Table altered.
   Here, column Gender is drop or deleted from the table student.

(3) TRUNCATE:
- The TRUNCATE command is used to remove all rows from the table.
- The TRUNCATE statement is similar to the DELETE statement. Both of the statements will delete rows from a table.
- So we cannot give any condition to delete specific row from table with TRUNCATE command.
- Truncate operations is faster than deleting one by one rows.
- The number of deleted rows are not returned.

Syntax: truncate table <TABLE_NAME>;
Example: truncate table student;
Output: Table truncated.
Here, all the rows are remove from table student. But table structure is not remove.

(4) DROP:
- When a table is no longer needed, it can be dropped.
- Both the table definition and the rows in the table are dropped.
- The space allocated for the table is free so it is used for other database objects.

Syntax: drop table <TABLE_NAME>;
Example: drop table student;
Output: Table dropped.
Here, all the rows are remove from table student and structure is also remove.

❖**RENAME***:*
- The RENAME command is used to change the table name.
- We give new table name from old one.

Syntax: RENAME <old_table_name> TO <new_table_name>;
Example: RENAME student TO stud;
- Here, we change ‘student’ table to ‘stud’ table.

❖**DESC or DESCRIBE***:*
- DESC command is used to display the structure of the table.

Syntax: DESC <Table_Name>;
Example: DESC stud;

❖**CREATING A TABLE FROM ANOTHER TABLE:**

:: Syntax ::

```
CREATE TABLE <Table_Name>
[(Column1, Column2,…, ColumnN)]
AS SELECT column1, Column2,…, ColumnN
FROM <Table_Name>;
```

:: Example ::

```
CREATE TABLE stud (S_ID, S_NAME, S_DOB)
AS SELECT ID, NAME, DOB
FROM student;
```

Q-4 Explain Data Manipulation Language (DML) in detail.

- DML commands are used for manipulate the data.
- Following are the DDL commands:

No.	Commands	Description
1	SELECT	Select data from a table or view, this is the main commands in SQL.
2	INSERT	Insert row or record in table.
3	DELETE	Delete rows from the table.
4	UPDATE	Changes the contents of the columns of the table.

(1) SELECT:

- The select command is used to retrieve rows from one or more table.
- Only view the selected data.
- Select command is not change the content of the table.

:: Syntax ::

```
SELECT [DISTINCT] <COLUMNS>
FROM <Table_Name>
[WHERE <condition>]
[ORDER BY <columns> {asc|desc}]
[GROUP BY <columns>]
[HAVING<condition>];
```

:: Example ::

SELECT * FROM student;
Above query retrieve and show all the column and all the record from table.

SELECT ID, NAME FROM student;
Above query retrieve and show all the record but only student id (ID) and student name (NAME) column from table.

SELECT * FROM student WHERE ID<10;
Above query show only those record whose student id (ID) is less than 10.

SELECT ID, NAME FROM student WHERE ID<10;
Above query show only those record whose student id (ID) is less than 10 but only ID and NAME column.

SELECT * FROM student ORDER BY ID;
Show all the record in ascending order of student id (ID).

SELECT * FROM student ORDER BY ID DESC;
Show all the record in descending order of student id (ID).

SELECT DISTINCT NAME FROM student;
Distinct clause is used to remove duplicate rows while displaying data. If two name is same then display only first name record.

(2) INSERT:

- The insert command is used to insert values into table.
- Once a table is created then after we insert the data into the table for manipulation.
- If field is string or date than value should be enclosed in single quote (‘).

:: Syntax ::

```
INSERT INTO <Table_Name> (<Column_Name1>, <Column_Name2>,…., <Column_NameN>)
VALUES (<Value1>, <Value2>,…., <ValueN>);
```

:: Example ::

```
INSERT INTO student (ID, NAME, DOB, MOBILE_NO)
VALUES (101, ’Mahesh’, ’10-OCT-1992’, 9898989898);
                    OR
INSERT INTO student VALUES (102, ’Ramesh’, ’25-Feb-1990’, 7777777777);
```

- For each of the listed columns, a corresponding (matching) value must be specified.
- Thus an insertion does not necessarily have to follow the order of the attributes as specified in the create table statement.
- If a column is omitted, the value NULL is inserted.

(3) DELETE:

- Delete command is used to delete single or multiple record from table.
- One should always apply condition on delete command using where clause otherwise delete command deletes all record from able.

:: Syntax::

```
DELETE FROM <Table_Name> WHERE <condition>;
```

:: Example::

```
DELETE FROM student WHERE ID=105;
```

- Above query delete the record of student ID=105.
- If we not specify any condition means WHERE clause then delete all the record of table.

(4) UPDATE:

- Update command is used to update existing record in table.
- We can update any number of existing columns in any sequence using update command.
- Update command should also be used with where clause to give condition to update record otherwise it updates all record in table.

:: Syntax ::

```
UPDATE <Table_Name> SET <Column_Name1>=New_Value, <Column_Name2>=New_Value
```

WHERE <Condition>;

:: Example ::
UPDATE student SET NAME='Ram' WHERE ID=105;

- Above query change the name like 'Ram' of student ID=105.

Q-5 Explain Data Control Language (DCL) in detail.

- DCL command is used to control access to the data and database in oracle.
- Following are the DCL commands:

No.	**Commands**	**Description**
1	GRANT	Used to grant privileges to table, views, and other object in oracle.
2	REVOKE	Used to revoke assigned privileges of table, views and other object in oracle.

❖ Create User account in oracle:

Syntax: - **CREATE USER <Username> IDENTIFIED BY <Password>;**

Example: - create user Ram identified by ram123;
Here, Ram is user name and ram123 is password.

(1) GRANT:

- The GRANT command is used to assign a permission to user.
- It is used to assign **system privileges** as well as **object privileges**.
- Appropriate rights can be assigned to the user by the DBA.
- The rights that allow the use of some or all of Oracle's resources on the server are called **Privileges**.
- Objects that are created by a user are owned and controlled by that user.
- If a user wishes to access any of the objects belonging to another user, the owner have to give permissions for such access. This is called **Granting of Privileges.**

:: Syntax ::

```
GRANT <system/object privileges>
ON <object name/ table name>
TO <user name> [WITH ADMIN OPTION];
```

- There are two types of privileges:
  1. System privileges.
  2. Object privileges.

- **System privileges:**
  - A system privileges is the right or permission to execute certain database action.
  - Most of create command is used in system privileges.

Example: **GRANT create table TO Ram;**
Here, will assign create table permission to user Ram.

CREATE	To create a table/ Objects.
ALTER	To modify a table/ Objects.
DROP	To delete a table/ Objects.

- **Object privileges:**
  - An object privileges is the right or permission to perform certain action on specific object.
  - When a user creates any object only user can view it and can perform operation on that object.

- User can grant a permission to access this object to other user with the help of grant command
  Example: **GRANT select ON student TO Ram;**
  Here, will assign select command permission on student table to user Ram.
- Following are the object privileges:

ALTER	To modify a table.
SELECT	To select data in a table or view.
INSERT	To insert rows in a table or view.
DELETE	To delete rows in a table or view.
UPDATE	To update the specified column data.
INDEX	To create or eliminate the indexes of a table.

- **WITH ADMIN OPTION:** Allows the user or role that receives the privileges to grant it to other users, modify it or even delete it.

(2) REVOKE:

- The REVOKE command is used to revoke assigned privileges to users.
- Once privileges is given can be denied to a user using the REVOKE command.
- The object owner can revoke privileges granted to another user.
- A user of an object who is not the owner, but has been granted than also revoke.

:: Syntax ::

```
REVOKE <system/object privileges>
ON <object name/ table name>
FROM <user name>;
```

:: Example ::

```
REVOKE select ON student FROM Ram;
```

Here, will reverted back select permission On student table from user Ram.

Q-6 Explain Transaction Processing Language (TPL) in detail.

- Transaction Processing Language is also known as Transaction Control Language (TCL).
- A series of operation performed on oracle database is called **transactions**.
- If transaction is completed successfully than it is required to store it permanently.
- A transaction can be closed by using either a commit or rollback statement.
- TPL is a part of SQL, used to control transactional Processing in a database.
- Transactional control is the ability to manage various transactions that may occur within a relational database management system.
- Following are the TPL commands:

No.	Commands	Description
1	COMMIT	Used to save the effect of transaction on permanent basis.
2	ROLLBACK	Used to undo all changes of a transaction made on data (same as undo).
3	SAVEPOINT	Used to divide the transaction into smaller sections to commit or rollback.

(1) COMMIT:

- The COMMIT command is used to permanently store all changes and closes the transaction.
- A sequence of database changes, for example a sequence of insert, update, and delete statements, is called a transaction.
- Changes made in records are temporarily stored in the database system.
- They become permanent only after the statement commit; has been issued.

- ➢ A commit ends up the current transaction and makes permanent changes to database made earlier by users during transaction.
  Syntax: COMMIT;
  Example: COMMIT;

(2) ROLLBACK:

- ➢ A rollback does exactly the opposite of commit.
- ➢ It ends the transaction but undo any changes made by transaction.
  Syntax: ROLLBACK TO SAVEPOINT<SAVEPOINT_NAME>;
  Example: ROLLBACK TO SAVEPOINT SP1;
- Rollback without savepoint performs following operations:
  1. Undo all the unsaved changes in current transaction.
  2. Erase all save point in current transaction.
  3. Release the transaction lock.
- Rollback with savepoint performs following operations:
  1. It is optional, used to specify up to which level you want to rollback the changes.
  2. Loses those save point created after the named savepoint.
  3. Release the transaction lock up to rollback is performed.

(3) SAVEPOINT:

- ➢ SAVEPOINT marks logical breakpoint in transaction.
- ➢ It does not save or undo the transaction but it only mark the transaction point.
- ➢ It is used with rollback command to specify up to what level you want to undo the information.
- ➢ An active savepoint one that is specified since last commit or rollback.
  Syntax: SAVEPOINT <savepoint name>;
  Example: SAVEPOINT SP1;

Q-7 Explain the Query Generation using Clause: (Where, Between, Distinct, Like, Order by, In, NOT IN.)

❖ **WHERE:**

- ⌘ The SQL WHERE clause is used to specify a condition while fetching the data from single table or joining with multiple tables.
- ⌘ If the given condition is satisfied then only it returns specific value from the table.
- ⌘ You would use WHERE clause to filter the records and fetching only necessary records.
- ⌘ The WHERE clause is not only used in SELECT statement, but it is also used in UPDATE, DELETE statement, etc.
- ⌘ You can specify a condition using comparison of logical operators like >, <, =, LIKE, NOT, etc.
- ⌘ Following are the data of student table:

ID	NAME	DOB	MOBILE_NO
101	Mahesh	10-Oct-1992	9898989898
102	Ramesh	25-Feb-1990	7777777777
103	Ram	15-Dec-1988	7878787878
104	Laxman	07-Apr-1994	9999999999
105	Sita	17-Aug-1991	9090909090

:: Syntax ::

```
SELECT Column1, Column2,..., ColumnN
FROM <Table_Name>
WHERE [CONDITION];
```

:: Example ::

```
SELECT ID, NAME, DOB
FROM student
WHERE ID>102;
```

- Here, in above example display the ID, NAME and DOB of student table whose ID is >101.
- Following are the output of above query:

ID	NAME	DOB
103	Ram	15-Dec-1988
104	Laxman	07-Apr-1994
105	Sita	17-Aug-1991

❖ **BETWEEN:**

- In order to select data is within range of values the BETWEEN operator is used.
- The BETWEEN operator allows the selection of rows that contain values within a specified lower and upper limit.
- The range is written after the word BETWEEN.
- The lower value must be written first.
- The two values in between the range must be linked with the keyword **AND**.
- The BETWEEN operator can be used with both character and numeric data type.

:: Syntax ::

```
SELECT Column1, Column2,..., ColumnN
FROM <Table_Name>
WHERE <Column_Name>
BETWEEN <lower_limit> AND <upper_limit>;
```

:: Example ::

```
SELECT ID, NAME, DOB
FROM student
WHERE ID BETWEEN 101 AND 103;
```

- Following are the output of above query:

ID	NAME	DOB
101	Mahesh	10-Oct-1992
102	Ramesh	25-Feb-1990
103	Ram	15-Dec-1988

❖ **DISTINCT:**

- The DISTINCT command is used for removing duplicate rows while selecting data.
- Fetch only unique record.
- There may be a situation when you have multiple duplicate records in a table. While fetching such records, it makes more sense to fetch only unique records instead of fetching duplicate records.

:: Syntax ::

```
SELECT DISTINCT Column1, Column2,..., ColumnN
FROM <Table_Name>
WHERE [condition];
```

:: Example ::

```
SELECT DISTINCT ID, NAME, DOB
FROM student;
```

- Following are the output of above query:

ID	NAME	DOB
101	Mahesh	10-Oct-1992
102	Ramesh	25-Feb-1990
103	Ram	15-Dec-1988
104	Laxman	07-Apr-1994
105	Sita	17-Aug-1991

❖ **LIKE:**

- The LIKE allows comparison of one string value with another string value, which is not identical.
- This is achieved by wildcard characters.
- Two wildcard characters that are available are:
  - **The percent sign (%)**
  - **The underscore (_)**
- The percent sign represents zero, one, or multiple characters.
- The underscore represents a single number or character.
- The symbols can be used in combinations.

:: Syntax ::

```
• SELECT * FROM <Table Name> WHER COLUMN LIKE '####%'
• SELECT * FROM <Table Name> WHER COLUMN LIKE '%####%'
• SELECT * FROM <Table Name> WHER COLUMN LIKE '####_'
• SELECT * FROM <Table Name> WHER COLUMN LIKE '_####'
• SELECT * FROM <Table Name> WHER COLUMN LIKE '_####_'
```

- You can combine N number of conditions using AND or OR operators. Here, #### could be any numeric or string value.
- Here are number of examples showing WHERE part having different LIKE clause with '%' and '_' operators:

Statement	Description
WHERE SALARY LIKE '200%'	Finds any values that start with 200.
WHERE SALARY LIKE '%200%'	Finds any values that have 200 in any position.
WHERE SALARY LIKE '_00%'	Finds any values that have 00 in the second and third positions.
WHERE SALARY LIKE '%2'	Finds any values that end with 2.
WHERE SALARY LIKE '_2%3'	Finds any values that have a 2 in the second position and end with a 3.
WHERE SALARY LIKE '2___3'	Finds any values in a 5-digit number that start with 2 and end with 3.

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

  Example: SELECT * FROM CUSTOMER WHERE SALARY LIKE '200%';

❖ ORDER BY:

- The ORDER BY clause is used to sort the data in ascending or descending order.
- We can sort data on one or more columns.
- Some database sorts query results in ascending order by default.
- We used ASC or DESC keyword for arranging the data.

:: Syntax ::

```
SELECT Column1, Column2,..., ColumnN
FROM <Table_Name> [WHERE condition]
[ORDER BY Column1, Column2,...] [ASC/DESC];
```

:: Example ::

```
SELECT ID, NAME, DOB
FROM student ORDER BY NAME;
```

- Following are the output of above query:

ID	NAME	DOB
104	Laxman	07-Apr-1994
101	Mahesh	10-Oct-1992
103	Ram	15-Dec-1988
102	Ramesh	25-Feb-1990
105	Sita	17-Aug-1991

❖ IN:

- The IN operator is used when you want to compare a column with more than one value.
- It is similar to an OR condition.

:: Syntax ::

```
SELECT Column1, Column2,..., ColumnN
FROM <Table_Name>
WHERE <Column_Name> IN (value1, value2,...);
```

:: Example ::

```
SELECT * FROM student
WHERE NAME IN ('Ram', 'Sita');
```

- Following are the output of above query:

ID	NAME	DOB	MOBILE_NO
103	Ram	15-Dec-1988	7878787878
105	Sita	17-Aug-1991	9090909090

❖ NOT IN:

- The NOT IN operator is used when you want to retrieve a column that has no entries in the table or referencing table.

:: Syntax ::

```
SELECT Column1, Column2,..., ColumnN
FROM <Table_Name>
WHERE <Column_Name> NOT IN (value1, value2,...);
```

:: Example ::

```
SELECT * FROM student
WHERE NAME NOT IN ('Ram', 'Sita');
```

- Following are the output of above query:

ID	NAME	DOB	MOBILE_NO
101	Mahesh	10-Oct-1992	9898989898
102	Ramesh	25-Feb-1990	7777777777
104	Laxman	07-Apr-1994	9999999999

❖ AND:

- The AND operator are used to combine multiple conditions to narrow data in an SQL statement.
- AND operator are called conjunctive operator.
- Provide multiple comparisons with different operators in the same SQL statement.
- The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause.
- If all the conditions are true then only give output.

:: Syntax ::

```
SELECT Column1, Column2,…, ColumnN
FROM <Table_Name>
WHERE [condition] AND [condition]…. AND [condition];
```

:: Example ::

```
SELECT * FROM student
WHERE ID>101 AND NAME='Ram';
```

- Following are the output of above query:

ID	NAME	DOB	MOBILE_NO
103	Ram	15-Dec-1988	7878787878

❖ OR:

- The OR operator are used to combine multiple conditions to narrow data in an SQL statement.
- OR operator are called conjunctive operator.
- Provide multiple comparisons with different operators in the same SQL statement.
- The OR operator allows the existence of multiple conditions in an SQL statement's WHERE clause.
- If only one condition is true then give output.

:: Syntax ::

```
SELECT Column1, Column2,…, ColumnN
FROM <Table_Name>
WHERE [condition] OR [condition]…. OR [condition];
```

:: Example ::

```
SELECT * FROM student
WHERE ID>101 OR NAME='Ram';
```

- Following are the output of above query:

ID	NAME	DOB	MOBILE_NO
101	Mahesh	10-Oct-1992	9898989898
102	Ramesh	25-Feb-1990	7777777777
103	Ram	15-Dec-1988	7878787878
104	Laxman	07-Apr-1994	9999999999
105	Sita	17-Aug-1991	9090909090