**SQL (Structured Query Language)**

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.

* Allows users to access data in the relational database management systems.
* Allows users to describe the data.
* Allows users to define the data in a database and manipulate that data.
* Allows to embed within other languages using SQL modules, libraries & pre-compilers.
* Allows users to create and drop databases and tables.
* Allows users to create view, stored procedure, functions in a database.
* Allows users to set permissions on tables, procedures and views.

**RDBMS(Relational Database Management System)**

* It is a database management system (DBMS) that is based on the relational model.
* *TABLE*: data in an RDBMS is stored in database objects which are called as tables.
* *FIELDS*:Every table is broken up into smaller entities called fields. The fields in the CUSTOMERS table consist of ID, NAME, AGE, ADDRESS and SALARY.
* *ROW*: A record is also called as a row of data is each individual entry that exists in a table.
* *COLUMN*:A column is a vertical entity in a table that contains all information associated with a specific field in a table.
* *Schema*:Schema diagram / ER diagram (Entity relationship).Schema is a blueprint.

**SQL data types:**

* int
* smallint
* tinyint
* float
* real
* date
* time
* datetime
* smalldatetime
* char
* varchar
* text
* image
* timestamp

**Stored Procedures**

These are nothing but functions/Methods which will hold verified SQL statements.

**SQL Commands**

* **CREATE DATABASE** : creates a new database

Syntax:

CREATE DATABASE DatabaseName;

* **DROP DATABASE**: statement is used to drop an existing database in SQL schema.

Syntax:

DROP DATABASE DatabaseName;

* **USE**: statement is used to select any existing database in the SQL schema.

Syntax:

USE DatabaseName;

* **CREATE TABLE** - creates a new table

Syntax:

CREATE TABLE table\_name(

column1 datatype,

column2 datatype,

column3 datatype,

.....

columnN datatype,

PRIMARY KEY( one or more columns )

);

* NOT NULL  − Ensures that a column cannot have NULL value.
* DEFAULT  − Provides a default value for a column when none is specified.
* UNIQUE  − Ensures that all values in a column are different.
* PRIMARY KEY− Uniquely identifies each row/record in a database table.
* FOREIGN KEY − Uniquely identifies a row/record in any of the given database table.
* CHECK− The CHECK constraint ensures that all the values in a column satisfies certain conditions.
* INDEX − Used to create and retrieve data from the database very quickly.
* **DROP TABLE** - deletes a table

Syntax:

DROP TABLE table\_name;

* **INSERT INTO** - inserts new data into a database

Syntax:

INSERT INTO TABLE\_NAME (column1, column2, column3,...columnN)

VALUES (value1, value2, value3,...valueN);

or

INSERT INTO TABLE\_NAME VALUES (value1,value2,value3,...valueN);

* **SELECT**- extracts data from a database

Syntax:

SELECT column1, column2, columnN FROM table\_name;

Select all

SELECT \* FROM table\_name;

* **WHERE clause :**

To specify a condition while fetching the data from a single table or by joining with multiple tables.

Syntax:Select with where

SELECT column1, column2, columnN

FROM table\_name

WHERE [condition] ;

* **UPDATE**- updates data in a database

Syntax:

UPDATE table\_name

SET column1 = value1, column2 = value2...., columnN = valueN

WHERE [condition];

* **DELETE**- delete the existing records from a table.

Syntax:

DELETE FROM table\_name

WHERE [condition];

* **ALTER TABLE**

add, delete or modify columns in an existing table.

* add a New Column in an existing

ALTER TABLE table\_name ADD column\_name datatype;

* DROP COLUMN in an existing table

ALTER TABLE table\_name DROP COLUMN column\_name;

* change the DATA TYPE of a column in a table

ALTER TABLE table\_name MODIFY COLUMN column\_name datatype;

* add a NOT NULL constraint to a column in a table

ALTER TABLE table\_name MODIFY column\_name datatype NOT NULL;

* ADD UNIQUE CONSTRAINT to a table

ALTER TABLE table\_name

ADD CONSTRAINT MyUniqueConstraint UNIQUE(column1,column2...);

* ADD CHECK CONSTRAINT to a table

ALTER TABLE table\_name

ADD CONSTRAINT MyUniqueConstraint CHECK (CONDITION);

* ADD PRIMARY KEY constraint to a table

ALTER TABLE table\_name

ADD CONSTRAINT MyPrimaryKey PRIMARY KEY (column1,column2...);

* DROP CONSTRAINT from a table

ALTER TABLE table\_name

DROP CONSTRAINT MyUniqueConstraint;

* **JOIN:**

Joins clause is used to combine records from two or more tables in a database. A JOIN is a means for combining fields from two tables by using values common to each.

* INNER JOIN − returns rows when there is a match in both tables.
* LEFT JOIN − returns all rows from the left table, even if there are no matches in the right table.
* RIGHT JOIN − returns all rows from the right table, even if there are no matches in the left table.
* FULL JOIN − returns rows when there is a match in one of the tables.
* SELF JOIN − is used to join a table to itself as if the table were two tables, temporarily renaming at least one table in the SQL statement.
* CARTESIAN JOIN − returns the Cartesian product of the sets of records from the two or more joined tables.
* **Order by**

Used to sort the data in ascending or descending order, based on one or more columns. Some databases sort the query results in an ascending order by default.

Syntax:

SELECT column-list

FROM table\_name

[WHERE condition]

[ORDER BY column1, column2, .. columnN] [ASC | DESC];

* **Group by**

Used in collaboration with the SELECT statement to arrange identical data into groups.

Syntax:

SELECT column1, column2

FROM table\_name

WHERE [ conditions ]

GROUP BY column1, column2

ORDER BY column1, column2

* **SQL View**

View is a virtual table based on the result-set of an SQL statement.

Contains rows and columns.

Syntax:

Create view :

CREATE VIEW view\_name AS

SELECT column1, column2.....

FROM table\_name

WHERE [condition];

Update view:

UPDATE view\_name

column= value

WHERE [condition];

Delete view:

DELETE FROM view\_name

WHERE [condition];

* **SQL Index**

Indexes are used to retrieve data from the database more quickly than otherwise. The users cannot see the indexes, they are just used to speed up searches/queries.

Syntax:

Create index:

CREATE INDEX index\_name ON table\_name;

Single column:

CREATE INDEX index\_name

ON table\_name (column\_name);

Unique :

CREATE UNIQUE INDEX index\_name

on table\_name (column\_name);

compositive:

CREATE INDEX index\_name

on table\_name (column1, column2);

Drop Index:

DROP INDEX index\_name;

* **Subquery**

A Subquery or Inner query or a Nested query is a query within another SQL query and embedded within the WHERE clause.

A subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved.

Subqueries can be used with the SELECT, INSERT, UPDATE, and DELETE statements along with the operators like =, <, >, >=, <=, IN, BETWEEN, etc.

Syntax:

**Select**:

SELECT column\_name [, column\_name ]

FROM table1 [, table2 ]

WHERE column\_name OPERATOR

(SELECT column\_name [, column\_name ]

FROM table1 [, table2 ]

[WHERE])

**Insert**:

INSERT INTO table\_name [ (column1 [, column2 ]) ]

SELECT [ \*|column1 [, column2 ]

FROM table1 [, table2 ]

[ WHERE VALUE OPERATOR ]

**Update**:

UPDATE table

SET column\_name = new\_value

[ WHERE OPERATOR [ VALUE ]

(SELECT COLUMN\_NAME

FROM TABLE\_NAME)

[ WHERE) ]

**Delete**:

DELETE FROM TABLE\_NAME

[ WHERE OPERATOR [ VALUE ]

(SELECT COLUMN\_NAME

FROM TABLE\_NAME)

[ WHERE) ]