**SQL Server for beginners:**

1. **Logical query processing order: •** It defines the order in which query need to be interpreted.
2. **FROM<table>: •** In this phase, we will indicate the table we want to query and table operators.
3. **WHERE: •** This phase is used to filtering the row based on predicates. It will return only those rows for which predicate will true.

• WHERE clause is evaluated before rows are grouped.

• It is evaluated per row.

1. **GROUP BY: •** This phase defines a group for each distinct combination of values in the group elements.

• This phase is applicable only when a query has GROUP BY clause.

1. **HAVING: •** This phase is responsible for filtering data based on predicates specify in this clause.

• This phase is evaluated after the data has been grouped.

• Hence, it is evaluated per group and filters group as whole, because it acts on the result on the result of GROUP BY clause.

• Aggregation function can be used in HAVING clause predicate.

• It is applicable only a query has HAVING clause.

• It is evaluated per group.

• It is evaluated after the rows are grouped.

1. **SELECT: •** In logical query processing order, it is evaluated at last, but in query it appears first.

• The result returns in this phase are rational (It may be in order or may be not).

1. **ORDER BY: •** It order the result in either ascending or descending order according to expression that appear in order by list.
2. **TOP(n): •** This phase is responsible for limit the no. of records to return.

• This phase is not supported by all SQL databases.

1. **SQL Fundamentals:**

• SQL stands for Structured Query Language.

• It is the ANSI Standard Language which is used to manage and manipulate data and

Structure in Microsoft SQL Server.

• The SQL language is very easy to understand and write. It is like plain English.

• The SQL language is subdivided into several language elements.

1. **Queries: •** It is used to retrieve the data based on specific criteria.

• Queries are used to reach into the database and pull out the useful information.

• It is also called statements.

**EX:** SELECT PRODUCTID, PRODUCTNAME, QUANTITY, PRICE

FROM DBO.PRODUCTS

WHERE PRICE > 18

ORDER BY PRODUCTNAME;

1. **Clauses: •** It is used to managing data such as filtering, sorting etc.

**EX:**  WHERE PRICE > 18

ORDER BY PRODUCTNAME;

1. **Predicates: •** Predicate is an expression which is used in search condition of WHERE clause and HAVING clause and evaluates to TRUE, FALSE or UNKNOWN.

• Predicates are used to limit the result of the query.

**EX:** PRICE > 18

1. **Expressions: •** It is a combination of one or more values, operators and SQL functions that evaluates against the common values.

• It is like a formula written in query languages.

**SELECT Statement Fundamentals:**

**•** SQL SELECT statement is used for fetching some data from a database. The statement is just a query that claims what data to be retrieved, where to search for it, and how to modify it before returning.

**Syntax: -- TO RETRIEVE ALL COLUMN RECORDS**

**SELECT \***

**FROM <table>**

**WHERE <predicate on rows> --(Optional Clause)**

**GROUP BY <column1, column2 . . .> --(Optional Clause)**

**HAVING <predicate on groups> --(Optional Clause)**

**ORDER BY <column1, column2 . . .> --(Optional Clause)**

**Syntax: -- TO RETRIEVE RECORDS ACCORDING TO INDIVIDUAL COLUMN**

**SELECT <column1, column2 . . .>**

**FROM <table>**

**WHERE <predicate on rows> --(Optional Clause)**

**GROUP BY <column1, column2 . . .> --(Optional Clause)**

**HAVING <predicate on groups> --(Optional Clause)**

**ORDER BY <column1, column2 . . .> --(Optional Clause)**

**-- General structure or syntax of SELECT statement.**

**-- To retrieve all columns.**

**SELECT \***

**FROM <table>**

**WHERE <predicate on rows> --(Optional Clause)**

**GROUP BY <column1, column2 . . .> --(Optional Clause)**

**HAVING <predicate on groups> --(Optional Clause)**

**ORDER BY <column1, column2 . . .> --(Optional Clause)**

**-- Retrieve records according to individual column**

**SELECT <column1, column2 . . .>**

**FROM <table>**

**WHERE <predicate on rows> --(Optional Clause)**

**GROUP BY <column1, column2 . . .> --(Optional Clause)**

**HAVING <predicate on groups> --(Optional Clause)**

**ORDER BY <column1, column2 . . .> --(Optional Clause)**

**--Filtering data with WHERE clause.**

**CREATE TABLE EMPLOYEE1(ETYPE VARCHAR (20), TITLE VARCHAR (10), FIRST\_NAME VARCHAR (30), LAST\_NAME VARCHAR (20), DOJ DATETIME);**

**INSERT INTO EMPLOYEE1 VALUES('EM','MS.','GAURANG','TIWARI','2022/11/11');**

**INSERT INTO EMPLOYEE1(ETYPE, FIRST\_NAME, LAST\_NAME, DOJ) VALUES('TR','ABHISHEK','GOYAL','1991/07/11');**

**SELECT \* FROM EMPLOYEE1;**

**--Write a query to retrieve data whose last name is 'Singh'**

**SELECT \* FROM EMPLOYEE1 WHERE LAST\_NAME='SINGH';**

**--Write a query to retrieve data whose last name is 'SHARMA'**

**SELECT \* FROM EMPLOYEE1 WHERE LAST\_NAME='SHARMA';**

**--Write a query to retrieve data whose title is not 'Mr.'**

**SELECT \* FROM EMPLOYEE1 WHERE Title <> 'Mr.'; --It will also discard NULL value rows.**

**--Write a query to retrieve all employee data whose title is 'Mr.'**

**SELECT \* FROM EMPLOYEE1 WHERE Title = 'Mr.'; --It will also discard NULL value rows.**

**--Write a query to retrieve all employee data whose title is 'Ms.'**

**SELECT \* FROM EMPLOYEE1 WHERE Title = 'Ms.'; --It will also discard NULL value rows.**

**SELECT \* FROM EMPLOYEE1 WHERE Title <> 'Ms.' OR TITLE IS NULL; -- It will show NULL value rows and also Title values.**

**-- ADD a column by using ALTER command.**

**ALTER TABLE EMPLOYEE1 ADD AGE INT;**

**SELECT \* FROM EMPLOYEE1;**

**--UPDATE a row after adding column by using ALTER command.**

**UPDATE EMPLOYEE1 SET AGE=40 WHERE FIRST\_NAME='GAURANG';**

**--BETWEEN**

**SELECT \* FROM EMPLOYEE1 WHERE AGE BETWEEN 30 AND 40;**

**SELECT \* FROM EMPLOYEE1 WHERE AGE BETWEEN 22 AND 26;**

**SELECT \* FROM EMPLOYEE1 WHERE AGE BETWEEN 20 AND 30 ORDER BY FIRST\_NAME;**

**--COMPARISON OPERATOR**

**SELECT \* FROM EMPLOYEE1 WHERE AGE>25;**

**SELECT \* FROM EMPLOYEE1 WHERE AGE>35 ORDER BY TITLE;**

**SELECT \* FROM EMPLOYEE1 WHERE AGE<30;**

**SELECT \* FROM EMPLOYEE1 WHERE AGE<28 ORDER BY LAST\_NAME;**

**SELECT \* FROM EMPLOYEE1 WHERE AGE>=22;**

**SELECT \* FROM EMPLOYEE1 WHERE AGE>=40;**

**SELECT \* FROM EMPLOYEE1 WHERE AGE<=30;**

**SELECT \* FROM EMPLOYEE1 WHERE AGE<=24;**

**--LIKE OPERATOR**

**SELECT ETYPE, FIRST\_NAME, LAST\_NAME FROM EMPLOYEE1 WHERE LAST\_NAME LIKE 'SIN%';**

**SELECT ETYPE, FIRST\_NAME, LAST\_NAME FROM EMPLOYEE1 WHERE LAST\_NAME LIKE '%MA';**

**SELECT \* FROM EMPLOYEE1 WHERE FIRST\_NAME LIKE 'A%';**

**SELECT ETYPE, FIRST\_NAME, LAST\_NAME FROM EMPLOYEE1 WHERE LAST\_NAME LIKE '%ARM%';**

**SELECT ETYPE, FIRST\_NAME, LAST\_NAME FROM EMPLOYEE1 WHERE FIRST\_NAME LIKE '%URA%';**

**SELECT \* FROM EMPLOYEE1 WHERE FIRST\_NAME LIKE '\_\_\_\_\_V';**

**SELECT \* FROM EMPLOYEE1 WHERE FIRST\_NAME LIKE '%N';**

**--IN OPERATOR**

**SELECT \* FROM EMPLOYEE1 WHERE AGE IN (30,40);**

**SELECT \* FROM EMPLOYEE1 WHERE AGE IN (20,27);**

**SELECT \* FROM EMPLOYEE1 WHERE TITLE IN ('MR.','MS.');**

**SELECT \* FROM EMPLOYEE1 WHERE AGE=20 OR AGE=27;**

**--NOT IN OPERATOR**

**SELECT \* FROM EMPLOYEE1 WHERE AGE NOT IN (20,29);**

**SELECT \* FROM EMPLOYEE1 WHERE AGE NOT IN (24,27);**

**--LOGICAL OPERATOR**

**SELECT \* FROM EMPLOYEE1 WHERE FIRST\_NAME='GAURAV' AND AGE=27;**

**SELECT \* FROM EMPLOYEE1 WHERE FIRST\_NAME='SONU' AND LAST\_NAME='SINGH';**

**SELECT \* FROM EMPLOYEE1 WHERE FIRST\_NAME='GAURANG' OR LAST\_NAME='SHARMA'; --It will show record in different ways according to whose first\_name is 'GAURANG' or last\_name is 'SHARMA'.**

**SELECT \* FROM EMPLOYEE1 WHERE FIRST\_NAME='RAHUL' OR LAST\_NAME='SINGH';**

**SELECT \* FROM EMPLOYEE1 WHERE FIRST\_NAME='SONU' AND NOT LAST\_NAME='SHARMA';**

**SELECT \* FROM EMPLOYEE1 WHERE NOT FIRST\_NAME='AMAN' AND NOT LAST\_NAME='SHARMA';**

**--ORDER BY CLAUSE**

**SELECT \* FROM EMPLOYEE1 ORDER BY DOJ;**

**SELECT \* FROM EMPLOYEE1 ORDER BY LAST\_NAME;**

**SELECT \* FROM EMPLOYEE1 ORDER BY FIRST\_NAME;**

**SELECT \* FROM EMPLOYEE1 ORDER BY FIRST\_NAME DESC;**

**SELECT \* FROM EMPLOYEE1 ORDER BY LAST\_NAME DESC;**

**SELECT \* FROM EMPLOYEE1 ORDER BY DOJ DESC;**

**SELECT \* FROM EMPLOYEE1 ORDER BY FIRST\_NAME, LAST\_NAME DESC;**

**SELECT \* FROM EMPLOYEE1 ORDER BY 3,4 DESC;**

**--GROUP BY CLAUSE**

**SELECT LAST\_NAME, COUNT (\*) AS TOTALEMPLOYEEWITHSAMELASTNAME FROM EMPLOYEE1 GROUP BY LAST\_NAME;**

**SELECT LAST\_NAME, COUNT (\*) AS TOTALEMPLOYEEWITHSAMELASTNAME FROM EMPLOYEE1 GROUP BY LAST\_NAME ORDER BY LAST\_NAME;**

**SELECT AGE, COUNT (\*) AS TOTALEMPLOYEEWITHSAMEAGE FROM EMPLOYEE1 GROUP BY AGE ORDER BY AGE;**

**SELECT COUNT (\*) FROM EMPLOYEE1;**

**SELECT LAST\_NAME FROM EMPLOYEE1 GROUP BY LAST\_NAME; --It will show unique last name records.**

**SELECT DISTINCT LAST\_NAME FROM EMPLOYEE1; --It will also show unique last name records.**

**SELECT YEAR (GETDATE ());**

**--AGGREGATE FUNCTION**

**SELECT COUNT (\*) FROM EMPLOYEE1; --It will return total no. of rows including NULL value.**

**SELECT COUNT(TITLE) FROM EMPLOYEE1; --It will ignore rows including NULL value.**

**SELECT COUNT (DISTINCT LAST\_NAME) FROM EMPLOYEE1;**

**SELECT \* FROM PRODUCT;**

**SELECT SUM(PRICE) AS TOTALSUM FROM PRODUCT;**

**SELECT SUM(PRICE) AS TOTALSUM FROM PRODUCT WHERE QUANTITY=2;**

**SELECT AVG(PRICE) AS TOTALSUM FROM PRODUCT;**

**SELECT SUM(PRICE) AS TOTALSUM FROM PRODUCT WHERE PRODUCT\_NAME='ITEM 1';**

**SELECT MAX(PRICE) AS TOTALSUM FROM PRODUCT;**

**SELECT MIN(PRICE) AS TOTALSUM FROM PRODUCT;**

**SELECT LAST\_NAME, COUNT (\*) FROM EMPLOYEE1 GROUP BY LAST\_NAME HAVING COUNT (\*)>=2;**

**SELECT PRODUCT\_ID, SUM(PRICE) FROM PRODUCT GROUP BY PRODUCT\_ID HAVING SUM(PRICE)>=1000;**

**SELECT PRODUCT\_ID, SUM(PRICE) FROM PRODUCT GROUP BY PRODUCT\_ID HAVING SUM(PRICE)<=200 ORDER BY PRODUCT\_ID;**

**--TOP KEYWORD**

**SELECT TOP 3 \* FROM EMPLOYEE1;**

**SELECT TOP 3 ETYPE, FIRST\_NAME, DOJ FROM EMPLOYEE1;**

**SELECT TOP 50 PERCENT ETYPE, FIRST\_NAME, DOJ FROM EMPLOYEE1;**

**SELECT TOP 30 PERCENT \* FROM EMPLOYEE1;**

**SELECT TOP 1 PERCENT \* FROM EMPLOYEE1;**

**SELECT \* FROM EMPLOYEE1;**

**SELECT TOP 4 WITH TIES LAST\_NAME, DOJ, AGE FROM EMPLOYEE1 ORDER BY LAST\_NAME;**