SQL (Structured Query Language)

Day- 2

Dimple Waghela and Goldie Gill

SQL Data Types

Data Types

- In SQL Server, each column, local variable, expression, and parameter has a related data type. A data type is an attribute that specifies the type of data that the object can hold: integer data, character data, monetary data, date and time data, binary strings, and so on.

Eg: Int, Varchar, Char, etc.

Categories of Data Type

Data types in SQL Server are organized into the following categories:

Exact numerics	Unicode character strings
Approximate numerics	Binary strings
Date and time	Other data types
Character strings	

Microsoft Word Document

Fractal® Analytics

WHERE Clause in SELECT

- The SQL **WHERE** clause is used to specify a condition while fetching the data from single table or joining with multiple table.

- The WHERE clause not only used in SELECT statement, but it is also used in UPDATE, DELETE statement etc. which we would examine in subsequent chapters.

WHERE Clause Example

Syntax:

SELECT column1, column2, columnN FROM table_name WHERE [condition]

Example:

SELECT ID, NAME, SALARY FROM CUSTOMERS WHERE SALARY > 2000;

SQL Operators

Comparison Conditions

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to
`BETWEEN ...AND...`	Between two values (inclusive)
`IN(set)`	Match any of a list of values
`LIKE`	Match a character pattern
`IS NULL`	Is a null value

LIKE Operator

- The SQL **LIKE** clause is used to compare a value to similar values using wildcard operators. There are two wildcards used in conjunction with the LIKE operator:
- 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.

LIKE Operator

Syntax:

SELECT FROM table_name WHERE column LIKE 'XXXX%' (or)

SELECT FROM table_name WHERE column LIKE '%XXXX%' (or)

SELECT FROM table_name WHERE column LIKE 'XXXX_' (or)

SELECT FROM table_name WHERE column LIKE '_XXXX' (or)

SELECT FROM table_name WHERE column LIKE '_XXXX_'

LIKE Operator-Example

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 start with 2 and are at least 3 characters in length
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 five-digit number that start with 2 and end with 3

BETWEEN ... AND Operator

- The operator BETWEEN and AND, are used to compare data for a range of values.

Example:

```
SELECT customer, product
FROM orders
WHERE quantity BETWEEN '1' AND '3';
```

IN Operator:

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

Example:

SELECT * FROM orders

where quantity

IN ('1' ,'3');

IS Null Operator

```
SELECT ProductID, Name, Color
FROM Production.Product
WHERE Color IS NULL
```

Logical Conditions

Operator	Meaning
`AND`	Returns `TRUE` if *both* component conditions are true
`OR`	Returns `TRUE` if *either* component condition is true
`NOT`	Returns `TRUE` if the following condition is false

AND Operator-Example:

- SELECT ID, NAME, SALARY FROM CUSTOMERS WHERE SALARY > 2000 AND age < 25;

OR Operator-Example:

- SELECT ID, NAME, SALARY FROM CUSTOMERS WHERE SALARY > 2000 OR age < 25;

NOT Operator-Example

```
SELECT ID, NAME, SALARY FROM CUSTOMERS
   WHERE  NOT SALARY = 2000;
```

TOP Clause

- The SQL **TOP** clause is used to fetch a TOP N number or X percent records from a table.

Syntax:

SELECT TOP number|percent column_name(s) FROM table_name WHERE [condition];

Example:

- SELECT TOP 3 * FROM CUSTOMERS;

ORDER BY Clause

- The SQL **ORDER BY** clause is used to sort the data in ascending or descending order, based on one or more columns. Some database sorts query results in ascending order by default.

Syntax:

```
SELECT column-list FROM table_name [WHERE
  condition] [ORDER BY column1, column2, .. columnN]
  [ASC | DESC];
```

ORDER BY Example

- SELECT * FROM CUSTOMERS ORDER BY NAME, SALARY;

- SELECT * FROM CUSTOMERS ORDER BY NAME DESC;

Fractal® Analytics

Distinct Keyword

- The SQL **DISTINCT** keyword is used in conjunction with SELECT statement to eliminate all the duplicate records and fetching only unique records.

- 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.

Distinct Keyword

Syntax:

```
SELECT DISTINCT column1, column2,.....columnN FROM
  table_name WHERE [condition];
```

Example:

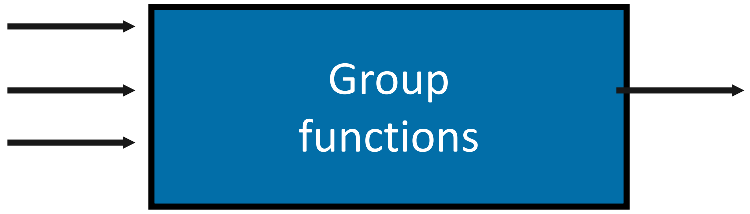
```
SELECT SALARY FROM CUSTOMERS ORDER BY
  SALARY;
```

```
SELECT DISTINCT SALARY FROM CUSTOMERS
  ORDER BY SALARY;
```

Aggregate Functions OR Group Functions

Types of Group Functions

- AVG
- COUNT
- MAX
- MIN
- SUM



Fractal® Analytics

COUNT ()-Example

- If you want the total number of employees in all the department, the query would take the form:

Example:

```
SELECT COUNT (*) FROM employee;


SELECT COUNT (*) FROM employee
  WHERE dept = 'Electronics';
```

COUNT, MAX, MIN, AVG, SUM, DISTINCT

- SELECT MAX (salary) FROM employee;
- SELECT MIN (salary) FROM employee;
- SELECT AVG (salary) FROM employee;
- SELECT SUM (salary) FROM employee;
- SELECT COUNT (DISTINCT name) FROM employee;
- SELECT DISTINCT dept FROM employee;

Group By Clause

- The SQL **GROUP BY** clause is used in collaboration with the SELECT statement to arrange identical data into groups.
- The GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.

Syntax:

SELECT column1, column2 FROM table_name WHERE [conditions] GROUP BY column1, column2 ORDER BY column1, column2

Group By-Example

- SELECT NAME, SUM(SALARY) FROM CUSTOMERS GROUP BY NAME;

- SELECT NAME, SUM(SALARY) FROM CUSTOMERS GROUP BY NAME;

HAVING Clause

- Having clause is used to filter data based on the group functions.
- This is similar to WHERE condition but is used with group functions.
- Group functions cannot be used in WHERE Clause but can be used in HAVING clause.

HAVING Example

```
SELECT dept, SUM (salary)
  FROM employee
  GROUP BY dept
  HAVING SUM (salary) > 25000
```

Thank You

Thank You

Fractal® Analytics

Fortune 500 companies recognize analytics as a competitive differentiator to **understand customers** and make **better decisions**.

We deliver insight, impact and innovation to them through **predictive analytics** and **visual story-telling**.

fractalanalytics.com

United States | Canada | United Kingdom | Italy | Switzerland | Singapore | UAE | India | China