RDBMS LANGUAGE

Eng. Lina Hammad

August 10th , 2022

DML language

Data Manipulation Language – SQL Comments

- Comments are used to explain sections of SQL statements, or to prevent execution of SQL statements.
- There are two types of comments:

1. **Single Line Comments**

- Single line comments start with --.
- Any text between -- and the end of the line will be ignored (will not be executed).
- The following example uses a single-line comment as an explanation:

```
--Select all:
SELECT * FROM Customers;
```

2. **Multi-line Comments:**

- Multi-line comments start with /* and end with */.
- Any text between /* and */ will be ignored.
- The following example uses a multi-line comment as an explanation:

```
/*Select all the columns
of all the records
in the Customers table:*/
SELECT * FROM Customers;
```

Data Manipulation Language – INSERT INTO statement

- The **INSERT INTO** statement is used to insert new records in a table.
- It is possible to write the INSERT INTO statement in two ways.
- **The first way** specifies both the column names and the values to be inserted:

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

- **The second way; i**f you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query. However, make sure the order of the values is in the same order as the columns in the table.
- The INSERT INTO syntax would be as follows:

```
INSERT INTO table_name
VALUES (value1, value2, value3, ...);
```

- To add multiple rows to a table at once, you use the following form of the **INSERT** statement:

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...),
VALUES (value1, value2, value3, ...),
VALUES (value1, value2, value3, ...)
;
```

Data Manipulation Language – UPDATE Statement

- The UPDATE statement is used to modify the existing records in a table.

- **Syntax:**

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

- **Note**: Be careful when updating records in a table! Notice the WHERE clause in the UPDATE statement. The WHERE clause specifies which record(s) that should be updated. If you omit the WHERE clause, all records in the table will be updated!

Data Manipulation Language – DELETE Statement

- The DELETE statement is used to delete existing record in a table.

- **Syntax:**

```
DELETE FROM table_name WHERE condition;
```

- **Note:** Be careful when deleting records in a table! Notice the WHERE clause in the DELETE statement. The WHERE clause specifies which record(s) should be deleted. If you omit the WHERE clause, all records in the table will be deleted!

Data Manipulation Language – Select Statement

- The SELECT statement is used to select data from a database.
- The data returned is stored in a result table, called the result-set.

- **Syntax:**

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

- Here, column1, column2, ... are the field names of the table you want to select data from.
- If you want to select all the fields available in the table, use the following **syntax**:

```
SELECT * FROM table_name;
```

Data Manipulation Language – Select DISTINCT Statement

- The SELECT DISTINCT statement is used to return only distinct (different) values.
- Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values.

- **Syntax:**

```
SELECT DISTINCT column1, column2, ...
FROM table_name;
```

Data Manipulation Language – WHERE Clause

- The WHERE clause is used to filter records.
- The WHERE clause is used to extract only those records that fulfill a specified condition.

- **Syntax:**

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

- **Note**: The WHERE clause is not only used in SELECT statement, but also in UPDATE, DELETE statement, etc.!

Data Manipulation Language – SQL NULL Values

- **What is a NULL Value?**
- A field with a NULL value is a field with no value.
- If a field in a table is optional, it is possible to insert a new record or update a record without adding a value to this field. Then, the field will be saved with a **NULL** value.
- **Note**: A NULL value is different from a zero value or a field that contains spaces. A field with a NULL value is one that has been left blank during record creation!

- **How to Test for NULL Values?**
- It is not possible to test for NULL values with comparison operators, such as =, <, or <>.
- We will have to use the IS NULL and IS NOT NULL operators instead.

- IS NULL **Syntax:**

```
SELECT column_names
FROM table_name
WHERE column_name IS NULL;
```

- IS NOT NULL **Syntax:**

```
SELECT column_names
FROM table_name
WHERE column_name IS NOT NULL;
```

- The IS NULL operator is used to test for empty values (NULL values).
- The IS NOT NULL operator is used to test for non-empty values (NOT NULL values).

Data Manipulation Language – And, Or, Not

- The AND operator displays a record if all the conditions separated by AND are TRUE.
- The OR operator displays a record if any of the conditions separated by OR is TRUE.
- The NOT operator displays a record if the condition(s) is NOT TRUE.
- Note: you can also combine the AND, OR and NOT operators.
- **AND Syntax:**

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 AND condition2 AND condition3 ...;
```

- **OR Syntax:**

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 OR condition2 OR condition3 ...;
```

- **NOT Syntax:**

```
SELECT column1, column2, ...
FROM table_name
WHERE NOT condition;
```

Data Manipulation Language – ORDER BY Keyword

- The ORDER BY keyword is used to sort the result-set in ascending or descending order.
- The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.
- **Note**: you also can ORDER BY Several Columns, and this means that the result will be ordered by coulmn1, but if some rows have the same value, we will be ordered them by coulmn2.

- **Syntax:**

```
SELECT column1, column2, ...
FROM table_name
ORDER BY column1, column2, ... ASC|DESC;
```

Data Manipulation Language – Limit clause

- The SELECT LIMIT clause is used to specify the number of records to return.
- The SELECT LIMIT clause is useful on large tables with thousands of records. Returning a large number of records can impact performance.

- **Syntax:**

```
SELECT column_name(s)
FROM table_name
WHERE condition
LIMIT number;
```

Data Manipulation Language – LIKE Operator

- The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.
- There are two wildcards often used in conjunction with the LIKE operator:
  1. % : The percent sign represents zero, one, or multiple characters.
  2. _ :The underscore represents a single character.
- Here are some examples showing different LIKE operators with '%' and '_' wildcards:

LIKE Operator	Description
WHERE CustomerName LIKE 'a%'	Finds any values that start with "a"
WHERE CustomerName LIKE ‘%a'	Finds any values that end with "a"
WHERE CustomerName LIKE '%or%'	Finds any values that have "or" in any position
WHERE CustomerName LIKE '_%r'	Finds any values that have "r" in the second position
WHERE CustomerName LIKE 'a_%'	Finds any values that start with "a" and are at least 2 characters in length
WHERE CustomerName LIKE 'a__%'	Finds any values that start with "a" and are at least 3 characters in length
WHERE ContactName LIKE 'a%o'	Finds any values that start with "a" and ends with "o"

Data Manipulation Language – LIKE Operator

- Here is some wildcard Characters in SQL Server.
- All the wildcards can also be used in combinations.

Symbol	Description	Example
%	Represents zero or more characters	bl% finds bl, black, blue, and blob
_	Represents a single character	h_t finds hot, hat, and hit
[]	Represents any single character within the brackets	h[oa]t finds hot and hat, but not hit
^	Represents any character not in the brackets	h[^oa]t finds hit, but not hot and hat
-	Represents a range of characters	c[a-b]t finds cat and cbt

Data Manipulation Language – LIKE Operator

- **Note**: The percent sign and the underscore can also be used in combinations!
- **Tip**: You can also combine any number of conditions using AND or OR operators.

- **Syntax**:

```
SELECT column1, column2, ...
FROM table_name
WHERE columnN LIKE pattern;
```

Data Manipulation Language – IN Operator

- The IN operator allows you to specify multiple values in a WHERE clause.
- The IN operator is a shorthand for multiple OR conditions.

- **Syntax**:

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2, ...);
```

- **Or:**

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (SELECT STATEMENT);
```

Data Manipulation Language – Between Operator

- The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates.
- The BETWEEN operator is inclusive: begin and end values are included.

- **Syntax**:

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1 AND value2;
```

Data Manipulation Language – Aggregate functions

- An aggregate function performs a calculation one or more values and returns a single value. The aggregate function is often used with the **GROUP BY** clause and **HAVING** clause of the **SELECT** statement.
- The following table shows the SQL Server aggregate functions:

Aggregate function	Description
Count()	The COUNT() aggregate function returns the number of rows that matches a specified criteria.
Avg()	The AVG() aggregate function returns the average value of a numeric column.
Sum()	The SUM() aggregate function returns the total sum of a numeric column.
Max()	The MAX() aggregate function returns the largest value of the selected column.
Min()	The MIN() aggregate function returns the smallest value of the selected column.

Data Manipulation Language – Aggregate functions

- **COUNT() Syntax:**

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

- **MAX() Syntax:**

```
SELECT MAX(column_name)
FROM table_name
WHERE condition;
```

- **AVG() Syntax:**

```
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

- **MIN() Syntax:**

```
SELECT MIN(column_name)
FROM table_name
WHERE condition;
```

- **SUM() Syntax:**

```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

Data Manipulation Language – GROUP BY Statement

- The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country".
- The GROUP BY statement is often used with aggregate functions (COUNT, MAX, MIN, SUM, AVG) to group the result-set by one or more columns.

- **Syntax**:

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
Having
ORDER BY column_name(s)
Limit;
```

Data Manipulation Language – Having clause

- The HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions.

- **Syntax**:

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition
ORDER BY column_name(s);
```

Data Manipulation Language – EXISTS

- The EXISTS operator is used to test for the existence of any record in a subquery.
- The EXISTS operator returns TRUE if the subquery returns one or more records.
- **Syntax**:

```
SELECT column_name(s)
FROM table_name
WHERE EXISTS
(SELECT column_name FROM table_name WHERE condition);
```

Data Manipulation Language – Any

- The ANY operator:
  - returns a boolean value as a result
  - returns TRUE if ANY of the subquery values meet the condition
- **ANY** means that the condition will be true if the operation is true for any of the values in the range.

- **Syntax**:

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator ANY
  (SELECT column_name
  FROM table_name
  WHERE condition);
```

Data Manipulation Language – All

- The ALL operator:
  - returns a boolean value as a result
  - returns TRUE if ALL of the subquery values meet the condition
  - is used with SELECT, WHERE and HAVING statements
- ALL means that the condition will be true only if the operation is true for all values in the range.
- **Syntax**:

```
SELECT ALL column_name(s)
FROM table_name
WHERE condition;
```

- **ALL Syntax With WHERE or HAVING Syntax**:

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator ALL
  (SELECT column_name
  FROM table_name
  WHERE condition);
```

Data Manipulation Language – All

- The ALL operator:
  - returns a boolean value as a result
  - returns TRUE if ALL of the subquery values meet the condition
  - is used with SELECT, WHERE and HAVING statements
- ALL means that the condition will be true only if the operation is true for all values in the range.
- **Syntax**:

```
SELECT ALL column_name(s)
FROM table_name
WHERE condition;
```

- **ALL Syntax With WHERE or HAVING Syntax**:

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator ALL
  (SELECT column_name
  FROM table_name
  WHERE condition);
```