1.8 SQL Queries - Comparing Strings

In order to compare strings to search for a specified pattern we must use the "LIKE" operator.

The use of this operator needs the use of wildcards characters that can be used to substitute

any other character(s) in a string. The most common wildcards are the following

- "%" - a substitute for zero or more characters;
- "_" - a substitute for a single character.

In the first example we define a wildcard (missing letters) before the pattern.

```
SELECT * FROM Customers

WHERE name LIKE 'E%';
```

This instruction selects all customers with a name starting with the letter "E".

In the second example we define a wildcard after the pattern. The statement is very similar to previous.

```
SELECT * FROM Customers WHERE name LIKE '%a';
```

This instruction selects all customers with a name ending with the letter "a".

```
SELECT * FROM Customers WHERE name LIKE '%e%';
```

This instruction selects all customers with a name containing the character "e".

Finally we use two examples to demonstrate the use of "_" wildcard.

The first scenario consists only in the substitution of a missing character.

```
SELECT * FROM Customers WHERE name LIKE '_lena';
```

This instruction selects all customers with a name containing the string "lena" and starting by any character.

The second scenario consists in the substitution of multiple characters.

```
SELECT * FROM Customers WHERE name LIKE 'E_e_a';
```

This instruction selects all customers with a name containing the string initiated by "E" with a third character equal to "e" and last character equal to "a".

1.9 SQL Queries - Aggregation Operators

SQL has many built-in functions for performing calculations on data. SQL aggregate functions return a single value, calculated from values in a column. The most common aggregation operators are the following:

SQL Queries - Aggregation Operators

- AVG() - returns the average value;
- COUNT( ) - returns the number of rows;
- MAX( ) - returns the largest value;
- MIN( ) - returns the smallest value;
- SUM( ) - returns the sum.

Example: If you have customer table that show in the below executes the following queries?

cust_id	cust_name	cust_age	cust_mobile	cust_address
1	ali	34	345	baghdad
2	sad	23	3356	diyala
3	samer	56	7890	theqar
4	rad	21	3342	basra
5	suha	43	3366	babel
6	rasha	47	3466	babel
7	masha	34	2266	NULL

1- AVG( ) Operator

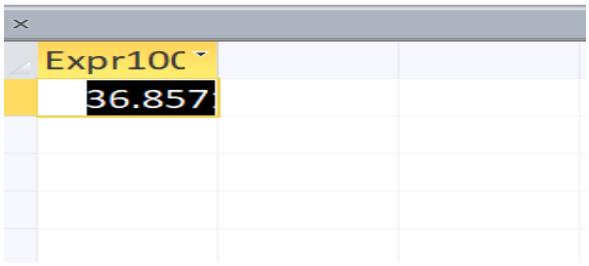
The AVG( ) function returns the average value of a numeric column. Here we show three examples using AVG( ) operator.

Find the Average of cust_age field in customer table and display its results?

Sol:-

```
SELECT AVG(cust_age) FROM Customer;
```

Results display as:-

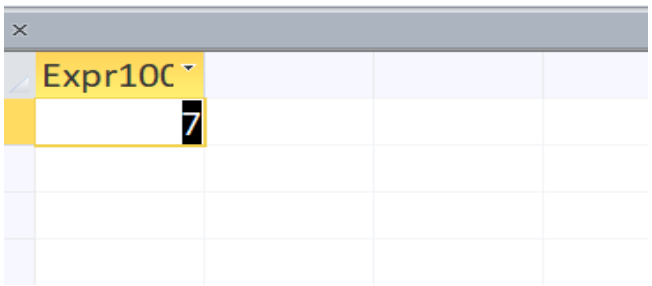


2- COUNT( ) Operator

The COUNT( ) function returns the number of rows that matches a specified criteria. Here we show two examples of using COUNT( ) operator.

EX: Find the number of Rows in customer table using SQL query?

Sol:- SELECT count(*) from Customer;

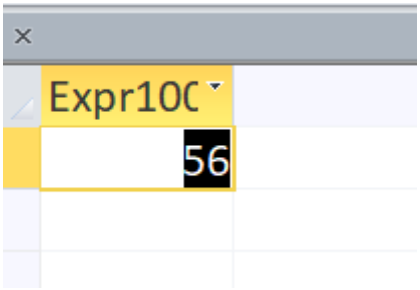


3. MAX( ) Operator

The MAX( ) function returns the largest value of the selected column. Here we show two examples using MAX( ) operator.

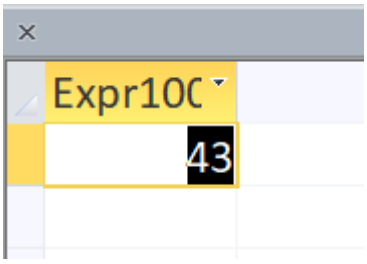
Ex: Find the highest customer age in the customer table?

Sol:- SELECT MAX(cust_age) FROM Customer;



Ex 2: Find the highest customer age in the customer table where the customer mobile number less than 7000?

Sol:- SELECT MAX(cust_age) FROM Customer where cust_mobile <7000;

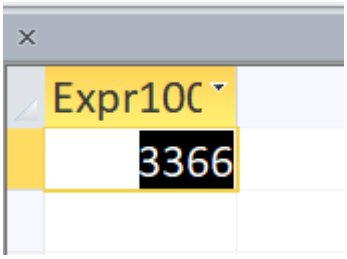


4-MIN( ) Operator

The MIN( ) function returns the smallest value of the selected column. Here we show two examples using MIN () operator.

Ex: Find the lowest customer mobile number in the customer table where the customer id more than 4 and customer address is babel?

Sol:- SELECT MIN(cust_mobile) FROM Customer WHERE cust_id >4 and cust_address='babel';

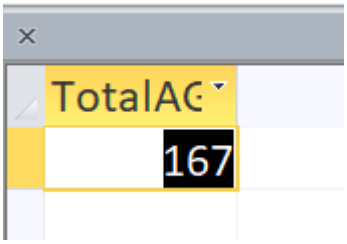


5- SUM( ) Operator

The SUM( ) function returns the total sum of a numeric column. Here we show two examples using SUM( ) operator.

Ex 1: Find and display the summation of the customer age from the customer table for the customer that their ages more 30 years?

Sol:- SELECT SUM(cust_age) AS TotalAGE FROM Customer where cust_age>30;



Ex 2: Find and display the customer name and customer age where the customer mobile<2500 and the summation of the customer ages from the customer table for the customer ages less than 30 years more than 50?

Sol: SELECT cust_name, cust_age from Customer where cust_mobile<2500 and(select SUM(cust_age) as total from Customer where cust_age<30) > 50;