**EXPERIMENT NO: 13**

**Title**: Implementation of Aggregate functions like min, max, avg, sum & count. Also use group by, having clauses

**Aim:**Implementation of Aggregate functions like min, max, avg, sum & count. Also use group by, having clauses

**Theory:**

***Aggregative operators:*** In addition to simply retrieving data, we often want to perform some computation or summarization. SQL allows the use of arithmetic expressions. We now consider a powerful class of constructs for computing aggregate values such as MIN and SUM.

1. **Count:** COUNT following by a column name returns the count of tuple in that column. If DISTINCT keyword is used then it will return only the count of unique tuple in the column. Otherwise, it will return count of all the tuples (including duplicates) count (\*) indicates all the tuples of thecolumn.

***Syntax:*** COUNT (Column name)

***Example:*** SELECT COUNT (Sal) FROM emp;

1. **SUM:** SUM followed by a column name returns the sum of all the values in thatcolumn.

***Syntax:*** SUM (Column name)

***Example:*** SELECT SUM (Sal) From emp;

1. **AVG:** AVG followed by a column name returns the average value of that columnvalues.

***Syntax:*** AVG (n1, n2...)

***Example:*** Select AVG (10, 15, 30) FROM DUAL;

1. **MAX:** MAX followed by a column name returns the maximum value of thatcolumn.

***Syntax:*** MAX (Column name)

***Example:*** SELECT MAX (Sal) FROM emp;

SQL> select deptno, max(sal) from emp group bydeptno; DEPTNO MAX(SAL)

|  |  |  |
| --- | --- | --- |
| 10 |  | 5000 |
| 20 |  | 3000 |
| 30 |  | 2850 |

SQL> select deptno, max (sal) from emp group by deptno havingmax(sal)<3000; DEPTNO MAX(SAL)

30 2850

1. **MIN:** MIN followed by column name returns the minimum value of thatcolumn.

***Syntax:*** MIN (Column name)

***Example:*** SELECT MIN (Sal) FROM emp;

###### SQL>select deptno,min(sal) from emp group by deptno having min(sal)>1000; DEPTNO MIN (SAL)

10 1300

**Conclusion :-**

Thus we have studied the Networking Components.

**Sample Questions**

1. What is Count aggregate function?
2. Is Max an aggregate function in SQL?
3. How AVG function works in SQL?
4. How do you use SUM and count together in SQL?

**EXPERIMENT NO: 14**

**Title**: Implementation of String operations.

**Aim:**Implementation of String operations.

**Theory:**

**String functions**  
are used to perform an operation on input string and return an output string.  
Following are the string functions defined in SQL:

1. **ASCII():** This function is used to find the ASCII value of a character.
2. **Syntax:** SELECT ascii('t');

**Output:** 116

1. **CHAR\_LENGTH():** Doesn’t work for SQL Server. Use LEN() for SQL Server. This function is used to find the length of a word.
2. **Syntax:** SELECT char\_length('Hello!');

**Output:** 6

1. **CHARACTER\_LENGTH():** Doesn’t work for SQL Server. Use LEN() for SQL Server. This function is used to find the length of a line.
2. **Syntax:** SELECT CHARACTER\_LENGTH('geeks for geeks');

**Output:** 15

1. **CONCAT():** This function is used to add two words or strings.
2. **Syntax:** SELECT 'Geeks' || ' ' || 'forGeeks' FROM dual;

**Output:** ‘GeeksforGeeks’

1. **CONCAT\_WS():** This function is used to add two words or strings with a symbol as concatenating symbol.
2. **Syntax:** SELECT CONCAT\_WS('\_', 'geeks', 'for', 'geeks');

**Output:** geeks\_for\_geeks

1. **FIND\_IN\_SET():** This function is used to find a symbol from a set of symbols.
2. **Syntax:** SELECT FIND\_IN\_SET('b', 'a, b, c, d, e, f');

**Output:** 2

1. **FORMAT():** This function is used to display a number in the given format.
2. **Syntax:** Format("0.981", "Percent");

**Output:** ‘98.10%’

1. **INSERT():** This function is used to insert the data into a database.
2. **Syntax:** INSERT INTO database (geek\_id, geek\_name) VALUES (5000, 'abc');

**Output:** successfully updated

1. **INSTR():** This function is used to find the occurrence of an alphabet.
2. **Syntax:** INSTR('geeks for geeks', 'e');

**Output:** 2 (the first occurrence of ‘e’)

**Syntax:** INSTR('geeks for geeks', 'e', 1, 2 );

**Output:** 3 (the second occurrence of ‘e’)

1. **LCASE():** This function is used to convert the given string into lower case.
2. **Syntax:** LCASE ("GeeksFor Geeks To Learn");

**Output:** geeksforgeeks to learn

1. **LEFT():** This function is used to SELECT a sub string from the left of given size or characters.
2. **Syntax:** SELECT LEFT('geeksforgeeks.org', 5);

**Output:** geeks

1. **LENGTH():** This function is used to find the length of a word.
2. **Syntax:** LENGTH('GeeksForGeeks');

**Output:** 13

1. **LOCATE():** This function is used to find the nth position of the given word in a string.
2. **Syntax:** SELECT LOCATE('for', 'geeksforgeeks', 1);

**Output:** 6

1. **LOWER():** This function is used to convert the upper case string into lower case.
2. **Syntax:** SELECT LOWER('GEEKSFORGEEKS.ORG');

**Output:** geeksforgeeks.org

**Conclusion :-**

Thus we have studied the String operations.

.

**Sample Questions**

1. What is string manipulation in SQL?
2. How do I concatenate strings in SQL?
3. How do you declare a string in SQL?
4. What is a string in database?

**EXPERIMENT NO: 15**

**Title**: Implementation of views for any created table.

**Aim:**Implementation of views for any created table.

**Theory:**

**VIEW:**

In SQL, a view is a virtual table based on the result-set of an SQL statement.

A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in thedatabase.

You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.

A view is a virtual table, which consists of a set of columns from one or more tables. It is similar to a table but it does not store in the database. View is a query stored as an object.

***Syntax:*** CREATE VIEW <view\_name> AS SELECT <set of fields>FROM relation\_name WHERE(Condition)

Example:

SQL> CREATE VIEW employee AS SELECT empno,ename,job FROM EMP WHERE job = ‘clerk’;

SQL> Viewcreated.

Example:

CREATE VIEW [Current Product List] AS SELECT ProductID, ProductName

FROM Products

WHERE Discontinued=No;

**UPDATING A VIEW :** A view can updated by using the following syntax :

**Syntax** : CREATE OR REPLACE VIEW view\_name AS SELECT column\_name(s)

FROM table\_name WHERE condition

**DROPPING A VIEW:** A view can deleted with the DROP VIEW command.

**Syntax**: DROP VIEW <view\_name>

**Conclusion**: Thus we Created & Update views for any created table.

**Sample Questions**

1. What are the views in database?
2. What is views and its types?
3. What is the use of views in SQL?
4. What are different types of views in DBMS?
5. Why views are used in database?
6. What is the use of view?

**EXPERIMENT NO: 16**

**Title**: Study of NoSql.

**Aim:**Study of NoSql.

## Theory:

## What is a NoSQL database?

When people use the term “NoSQL database,” they typically use it to refer to any non-relational database. Some say the term “NoSQL” stands for “non SQL” while others say it stands for “not only SQL.” Either way, most agree that NoSQL databases are databases that store data in a format other than relational tables.

### Brief history of NoSQL databases

NoSQL databases emerged in the late 2000s as the cost of storage dramatically decreased. Gone were the days of needing to create a complex, difficult-to-manage data model in order to avoid data duplication. Developers (rather than storage) were becoming the primary cost of software development, so NoSQL databases optimized for developer productivity.

As storage costs rapidly decreased, the amount of data that applications needed to store and query increased. This data came in all shapes and sizes — [structured, semi-structured,](https://www.mongodb.com/unstructured-data) and [polymorphic](https://www.mongodb.com/developer/how-to/polymorphic-pattern/) — and defining the schema in advance became nearly impossible. NoSQL databases allow developers to store huge amounts of unstructured data, giving them a lot of flexibility.

Additionally, the [Agile Manifesto](https://agilemanifesto.org/) was rising in popularity, and software engineers were rethinking the way they developed software. They were recognizing the need to rapidly adapt to changing requirements. They needed the ability to iterate quickly and make changes throughout their software stack — all the way down to the database. NoSQL databases gave them this flexibility.

Cloud computing also rose in popularity, and developers began using public clouds to host their applications and data. They wanted the ability to distribute data across multiple servers and regions to make their applications resilient, to scale out instead of scale up, and to intelligently geo-place their data. Some NoSQL databases like MongoDB provide these capabilities.

### NoSQL database features

Each NoSQL database has its own unique features. At a high level, many NoSQL databases have the following features:

* [Flexible schemas](https://docs.mongodb.com/manual/core/data-modeling-introduction/#flexible-schema)
* [Horizontal scaling](https://www.mongodb.com/basics/scaling)
* [Fast queries due to the data model](https://docs.mongodb.com/manual/core/data-modeling-introduction/#document-structure)
* [Ease of use for developers](https://www.mongodb.com/why-use-mongodb)

Check out [What are the Benefits of NoSQL Databases?](https://www.mongodb.com/nosql-explained/nosql-vs-sql#what-are-the-benefits-of-nosql-databases) to learn more about each of the features listed above.

### Types of NoSQL databases

Over time, four major [types of NoSQL databases](https://www.mongodb.com/scale/types-of-nosql-databases) emerged: document databases, key-value databases, wide-column stores, and graph databases.

* **Document databases** store data in documents similar to JSON (JavaScript Object Notation) objects. Each document contains pairs of fields and values. The values can typically be a variety of types including things like strings, numbers, booleans, arrays, or objects.
* **Key-value databases** are a simpler type of database where each item contains keys and values.
* **Wide-column stores** store data in tables, rows, and dynamic columns.
* **Graph databases** store data in nodes and edges. Nodes typically store information about people, places, and things, while edges store information about the relationships between the nodes.

### Difference between RDBMS and NoSQL databases

While a variety of differences exist between relational database management systems (RDBMS) and NoSQL databases, one of the key differences is the way the data is modeled in the database. In this section, we'll work through an example of modeling the same data in a relational database and a NoSQL database. Then, we'll highlight some of the other key differences between relational databases and NoSQL databases.

**Conclusion :-**

Thus we have studied NOSQL.

**Sample Questions**

1. What is difference between SQL and NoSQL?
2. What is NoSQL example?
3. Is MongoDB and NoSQL same?
4. What are the 4 types of NoSQL databases?