**Chapter 3: QUERYING AND REPORTING**

**PART I**

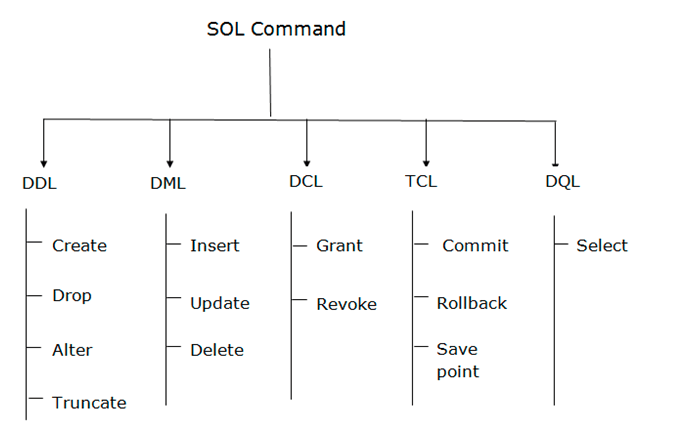
**RETRIEVING DATA**

* We retrieve data/information stored in a table/database.
* To access data from SQL server we use a Structured Query Language (SQL).

**STRUCTURED QUERY LANGUAGE**

* Structured Query Language (SQL) is a standard computer language for relational database management and data manipulation.
* SQL is used to query, insert, update, and modify data**.**
* Most relational databases support SQL, which is an added benefit for database administrators (DBAs), as they are often required to support databases across several different platforms.
* It translates a logical name into a linked set of physical locations.
* It manages all the locks, and as the result, eliminates need for explicit locking statement.

**SQL Commands**

****

**DDL**

* DDL stands for **data definition language**. DDL Commands deal with the schema, i.e., the table in which our data is stored.
* All the structural changes such as creation, deletion and alteration on the table can be carried with the DDL commands in SQL.

**DML**

* DML stands for *Data Manipulation Language*. Using DML commands in SQL, we can make changes in the data present in tables.
* Whenever we wish to manipulate the data or fetch the data present in SQL tables, we can use DML commands in SQL.
* DML commands in SQL will change the data, such as **inserting new records, deleting or updating existing records from the SQL tables**. We can also retrieve all the data from SQL tables according to our requirements.

**DCL**

* DCL stands for **Data Control Language.**
* Whenever we want to control the access to the data present in SQL tables, we will use DCL commands in SQL. Only the authorized users can access the data stored in the tables.
* Every user will have some pre-defined privileges; accordingly, the data can be accessed by that user. Using the DCL commands in SQL, we can give privileges to the user on the SQL database and tables, or we can also revoke the given privileges from the user.

**TCL**

* TCL stands for **Transaction Control Language**. TCL commands are generally used in transactions.
* Using TCL commands in SQL, we can save our transactions to the database and roll them back to a specific point in our transaction. We can also save a particular portion of our transaction using the SAVEPOINT command.

**SELECT Statement:**

* The SELECT statement is used to select data from a database.

**Selecting columns:**

* The SELECT statement can be used to retrieve specific columns from the table by specifying the column’s name.

**Syntax:**

SELECT column\_name1, column\_name2  
FROM table\_name;

**Example:**

SELECT [empid]

,[lastname]

,[firstname]

,[title]

,[titleofcourtesy]

,[birthdate]

,[hiredate]

FROM HR].[Employees];

Table

Description automatically generated

**Selecting All Columns:**

The SELECT statement can be used to retrieve all columns from the tables.

**Syntax:**

SELECT \* FROM table\_name;

**Examples:**

SELECT\*FROM [HR].[Employees];

Table

Description automatically generated

**Changing Column Sequence:**

* The order of the column can be changed in the result set of the SELECT statement.

**Examples:**

SELECT [lastname]

,[firstname]

,[title]

,[empid] [HR].[Employees];

Table

Description automatically generated

**Manipulating Column Names:**

* A user defined column heading can replace the default column heading, called **Alias**.
* SQL aliases are used to give a database table, or a column in a table, a temporary name.

**Table

Description automatically generatedSyntax:**

SELECT column\_name AS alias\_name  
FROM table\_name;

**Example 1:**

SELECT [firstname]

,[lastname] as [sur\_name]

,[title] as [Designation]

,[empid] as [EmployeeId]

FROM [HR].[Employees]

**Example 2:**

SELECT [empid],CONCAT([firstname],' ',[lastname] ) as Employee\_Detail FROM [HR].[Employees];**Table

Description automatically generated**

In the above SQL statement we combined two columns (firstname and lastname) and created an alias named“ Employee\_Detail”.

Table

Description automatically generated**Syntax:**

SELECT 'Column\_alias'=column\_name,

'column\_alias'=column\_name

FROM table\_name

**Example3:**

SELECT 'FIRST\_NAME'= firstname,

'SUR\_NAME'= lastname

FROM [HR].[Employees]

**Using Literals:**

* The result set of the data query can be made more reliable by using string, called literal in SELECT statement.
* Literals are enclosed in single quotes and gets printed exactly as they are written in SELECT statement.
* A string literal generally we use before the column name for which a string is to be displayed.

Table

Description automatically generated**Syntax:**

SELECT column\_name1\'string\_literal’,

column\_name2\'string\_literal'

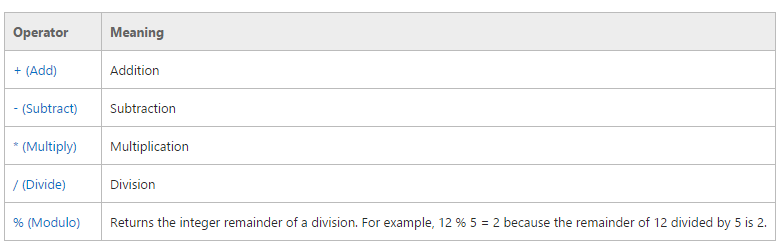
FROM table\_name

Examples:

SELECT firstname,'SUR\_NAME IS:', lastname

FROM [HR].[Employees]

**Arithmetic operators**



**Syntax:**

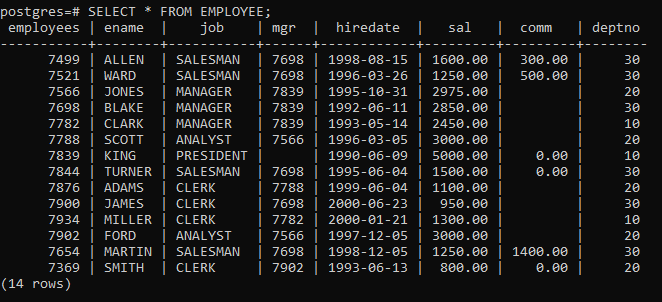
SELECT <Expression>[arithmetic operator]<expression>...

FROM [table\_name]

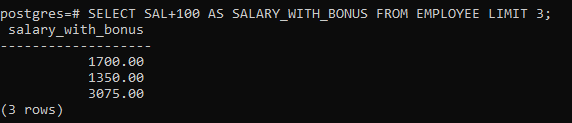
WHERE [expression];

**Example:**

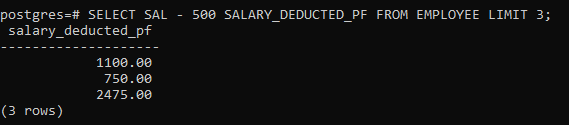
**EMPLOYEE TABLE**

****

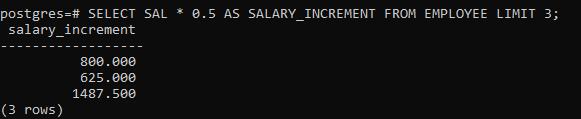
**ADDITION**

****

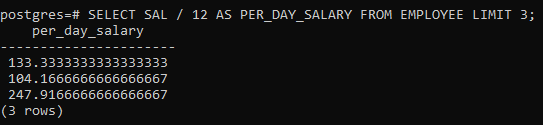
**SUBRACTION**

****

**MULTIPLICATION**

****

**DIVISION**

****

**Some rules regarding the use of arithmetic operators are:**

* Arithmetic operators can be performed on numeric columns or numeric constants.

**Operator Precedence:**

* Arithmetic operators take place according to below order:

1. \* Multiplication
2. / Division
3. % Modulo
4. + Addition
5. – Subtraction

* The precedence of the operators can be changed by using the primary grouping object called parenthesis ().
* When an arithmetic expression uses the same level of precedence, the order of execution is from the left to right.

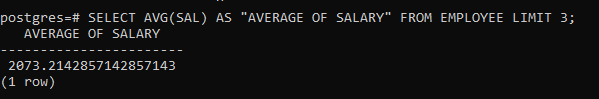
# 2. functions

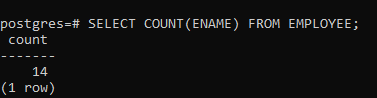
**SQL Aggregate Functions**

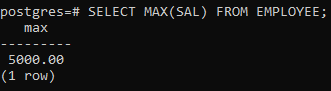
* SQL aggregate functions return a single value, calculated from values in a column.

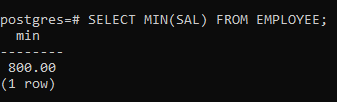
**Useful aggregate functions:**

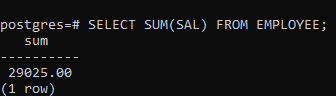
* AVG() - Returns the average value
* COUNT() - Returns the number of rows
* MAX() - Returns the largest value
* MIN() - Returns the smallest value
* SUM() - Returns the sum











**SQL Scalar functions**

* SQL scalar functions return a single value, based on the input value.

**Useful scalar functions:**

* UPPER() - Converts a field to upper case
* LOWER() - Converts a field to lower case
* LENGTH() - Returns the length of a text field
* ROUND() - Rounds a numeric field to the number of decimals specified
* CURRENT\_DATE() - Returns the current system date and time
* CURRENT\_TIMESTAMP – Returns the current time in the format of 'HH:MI:SS.MS'.

