SQL

In today's lesson, we are going to look at SQL. We will get to the basics of SQL by understanding why it is important to have a Database Management System (DBMS). Today's lesson will cover the following objectives:

- To discuss the fundamentals of SQL.
- To describe the various SQL Queries.
- To outline the fundamental sorting processes in SQL.

Fundamentals of SQL

SQL is the short form of Structured Query Language. This is a subtype of one of the major forms of Database Management System known as Relational DBMS.
First of all, let us look at what Database Management System means.

What is a Database?

This simply means an orderly collection of a set of data. Data can be stored and controlled in a database. As such, it becomes easy to manage that particular information.

A simple form of a database comprises of the following: in your household, your electricity provider keeps all the billings, clients concerns, reports on repair works on a database.

Now that we know what a database is, we can build up on that and define a Database Management System (DBMS).

This term refers to a group of programs which allows the clients to obtain and modify data.

The management system is also used to regulate the accessibility of any form of data, taking into consideration that some types of data are confidential.

The origin of Database Management Systems dates back to the 1960s. A computer scientist by the name Charles Bachman, is attributed to having invented the first form of DBMS. It was referred to as Integrated Data Store (IDS).

That form of DBMS has seen some major reformations up to date. Currently, DBMS has immense and diverse functionalities.

Basic Components of a DBMS

Each DBMS contains some basic elements which enhances the functionality. These elements are described below:

1. **Table** - Any set of data in a DBMS is stored in form of a table. This table is made up of a group of

similar data. It also has several rows and columns.
2. **Field** - After storing data in a table, the tables are then sub divided into fields. The field is where specific data is kept. For example, a database which has a client's data will have different fields of the client's age, sex, name, etc.
3. **Row** - A row comprises of a cluster of each distinct data entry found in the table.
4. **Column** - This represents the upright data entry found in a table which is related to a particular field.

From the above definitions, let us look further at some of the components of a Table, as it forms an integral part of the DBMS.

SQL Table Constraints
This term refers to the applied rules on table columns. These rules control the type of data put on the table. This ensures that the database is accurate and valid.
Constraints can be in form of column or table. The following are a list of SQL Constraints that are normally used:

1. **Primary key** – it is used to identify a discrete record in the table.
2. **Foreign key** – it identifies a discrete row from another table.
3. **Unique constraint** – it differentiates the values in a given column.
4. **Default constraint** – it offers an automatic element for a column in case there is no specified value.
5. **Check constraint** – it verifies that all values have met the required conditions before being keyed in a column.
6. **Not null** – it supplies a notification that a value has to be put on a column. The column should not lack a value.

Data Integrity

This component is used to maintain data consistency in a database. Data integrity comprises of the following:

1. **Domain integrity** – this element is used to verify the relevant data entries made in a column by limiting the scope and format of values.
2. **Entity integrity** – it ensures no repeated rows are put on a table.
3. **User-defined integrity** – it implements particular operational regulations that are not found in the other types of data integrity.
4. **Referential integrity** – it safeguards the rows to ensure they cannot be erased.

Database Normalization
This refers to the process of ensuring that data is stored effectively in a database. The aim of this process is to eradicate unnecessary data as well as making sure that the data is logically structured.

Types of Database Management Systems (DBMS)

DBMS is categorized into four types which are discussed below:

Network DBMS

As the name suggests, this type of DBMS comprises of diverse relations which form complex database programs.

Hierarchical DBMS

It utilizes hierarchies of data storage. The main data is subdivided into smaller bits which are stored in compartments.

Object oriented relational DBMS

It enables the data to be stored as objects as opposed to values. The objects are characterized distinctly.

Relational DBMS

This is the most prominent DBMS available. The database integration is defined as tables. It also contains predefined data that is supported by the relations.

Relational DBMS is programmed by the use of a language known as Structured Query Language (SQL). This is the basic language for all Relational Database Management Systems (RDMS).

RDMS is comprised of several types which we will brush through briefly:

MySQL

MySQL is a decentralized SQL database attributed to a Swedish company. This language is applicable in operation systems such as MS Windows, Linux, Mac OS X as well as UNIX.

It can be used both for commercial and personal purposes. MySQL has the following features:

- It has ease of use.
- It is efficient.
- It has the capability to increase its output.
- It is operational on a 24/7 basis.
- It is easily manageable.

MS SQL Server

This has been developed by Microsoft Incorporation. Its main features comprise of the following:

- It is efficient.
- It is easily accessible.
- It can create and maintain redundant database copies.
- It can create Data Definition Language (DDL).
- It can combine the Common Language Runtime (CLR).
- It can also utilize the Extensible Markup Language (XML).

Oracle

This was developed by Oracle Corporation. This RDMS can serve multiple users. It works well for both the clients and server database input. It can accommodate diverse operating systems such as MS DOS, UnixWare, Netware, etc. It has the following features:

- It can display commands concurrently.
- It is consistent.
- The database can be inactivated.
- It can accommodate bitmap indexes.
- It has a component for managing the database resources.
- It has the ability to evaluate large databases to establish patterns.
- It has data warehousing function. This system allows data to be reported and analyzed.

MS Access

This is a simple database management software. It is freely accessible and suitable for simple tasks. The interface is also user-friendly. Features of MS Access include:

- It can be integrated with macros to design tables, queries and reports.
- It facilitates data import and export in various formats like Excel and FoxPro.
- Some of its versions can accommodate multiple values and attachment fields in a single file.
- MS Access has prepared statements which can be cited from programs such as VB6 or ADO.
- This database is based on the server file. This means that it does not execute database triggers or keep a history of actions executed by the DBMS.

Difference between SQL and NoSQL

NoSQL is defined as Not only Structured Query Language. This type of database management system is non-relational.

It is used to provide storage and retrieval of data which is not computed in tabular form.

Like the SQL, this type also became prominent in the 1960s and has seen a tremendous advancement in its functionality. The main features of a NoSQL system are;

- It has simple interface and design
- It has flexible data structures compared to those of relational DBMS like the SQL, thus some of its operations are executed faster.

On the other hand, NoSQL has faced several entry barriers which has made SQL to be considered more popular.

Some of these barriers include; shortage of standardized interfaces and some NoSQL networks exhibit lost writes where the system displays data as being blocked, thus the user is unable to view.

When its comes to Structured Query Language (SQL), it is considered the standard language for relational database management systems.

An SQL is able to retrieve data from a database. It is also used to execute queries on a database. It therefore acts as the medium for communication between the user and the database.

This communication is executed through the use of queries. A query is simply defined as a request for data through the DBMS.

This lesson is sorely focused on SQL but it is also important to highlight other tenets of the database management system. Now that we know some of the differences, let us now turn our attention to SQL.

Importance of SQL

Structured Query Language (SQL) is efficient because of the following reasons:

- It facilitates data description.
- It enables data definition and data manipulation.
- SQL provides data accessibility options.
- SQL can be embedded in other languages through modules or pre-compilers.
- It has multiple functions which make it effective. A user is able to develop functions and views in a database.
- It facilitates permission settings to be done on views, tables and procedures.

Advantages and Disadvantages of SQL

Advantages	Disadvantages
High Performance. SQL queries are used to obtain large data loads efficiently and quickly.	SQL Interfacing is a complex process.
No coding. Database can be easily managed without codes.	It is too costly and advanced even for some programmers.
Portability. SQL is easily used in PCs, laptops and servers.	It cannot be manipulated by programmers as they please. It has some hidden operational regulations which hinder full accessibility.
Facilitates multiple data views. Users can have diverse views on the database structure.	

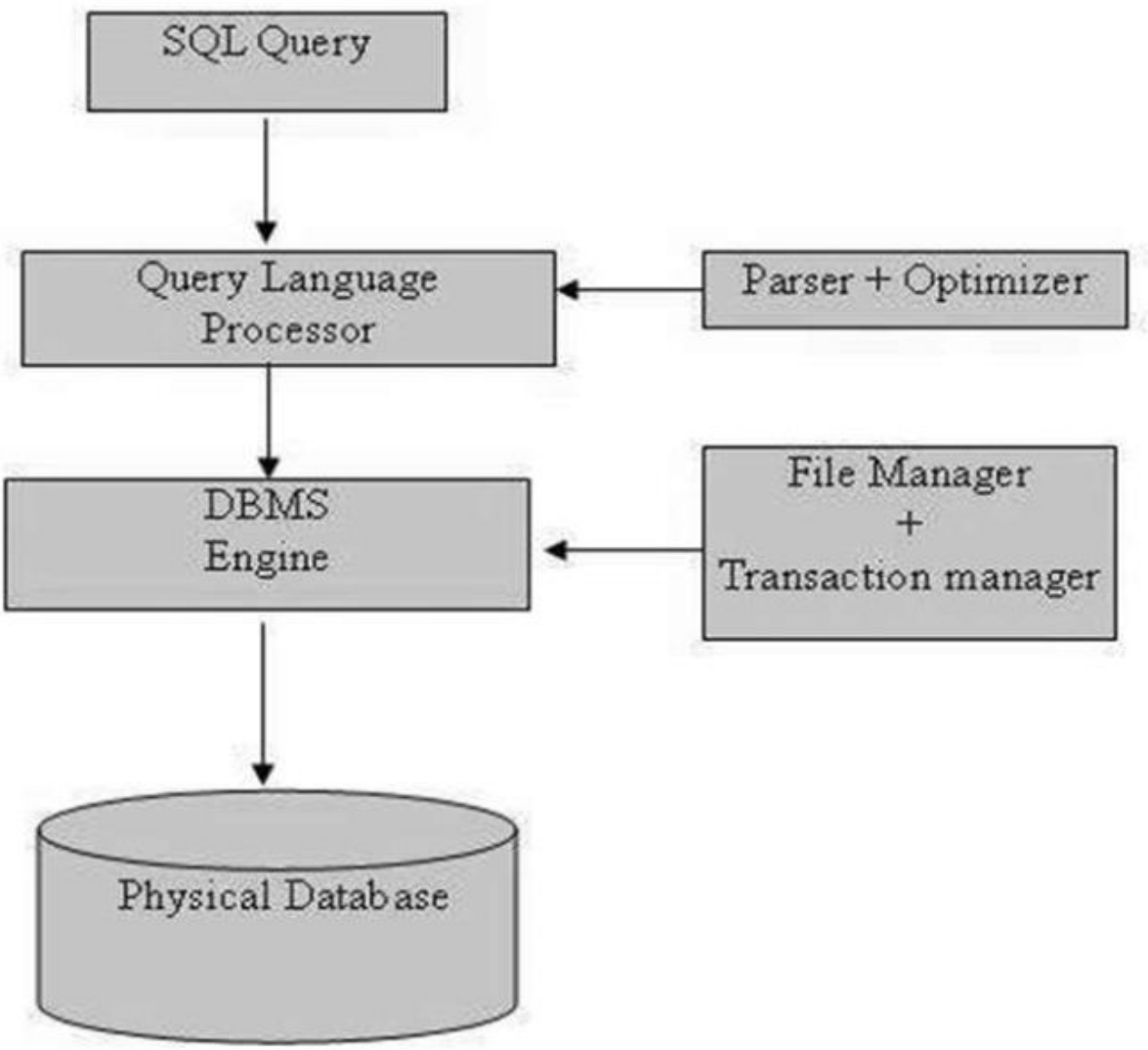
Components of the SQL Process

When implementing any SQL command for a given RDBMS, the system will identify the most suitable way of executing that command. SQL mechanism will interpret that command. As such, this process entails the following components:

- Query Dispatcher
- Optimization engine
- Classic query engine
- SQL query engine
- SQL Syntax and Statements. SQL syntax refers to the unique set of guidelines used.
  SQL statements begin with any of the SQL command and ends with a semicolon. For example,
  SELECT column FROM table name;

SQL Architecture

Executing an SQL command goes through the following cycle.



SQL Commands

In order to execute any SQL command, we will be required to first understand what these commands do. SQL has several commands which are outlined below:
Create – as its name suggests, this command is used to develop a new table, table view or any other database object.
Alter – this command is used to modify any database object.
Drop – if you want to delete any database component like the table, column value or any other object, this is the command to use.
Select – it is used to highlight any row from the table.
Insert – it is used to create a row.
Update – this command is used to modify or alter the rows (records).
Delete – rows can be deleted by this command.
Grant – this command provides a user with certain privileges.
Revoke – this command is used to take back the user privileges.

SQL Operators

These operators are simply characters which are used to carry out operations like making comparisons. There are three basic types of SQL operators. They include:

- SQL Logical Operators
- SQL Arithmetic Operators
- SQL Comparison Operators

The following table describes some of the Arithmetic operators

Arithmetic Operators

Operator	Description	Example
+	It adds the values	a + b
-	It takes away the right hand value from the left	a - b
*	It multiples the two values	a*b
/	It divides left hand value from right hand value	b/a
%	It divides left hand from right hand value and give back any remaining value	b % a

SQL Comparison Operators

The following table will highlight some of the comparison operators. They are easy to understand.

Operator	Description	Example
=	It evaluates whether values are equal or not.	a = b
!=	It verifies whether one value is equal to another.	a != b
<>	It equates the two values, thus proving the condition is true.	a <> b
>	It verifies whether the left value is greater than the right one.	a > b
<	It checks whether the left value is less than the right value.	a < b
>=	It checks whether the left value is greater than or equal to the right value.	a >= b
<=	It verifies whether the left value is less than or equal to the right value.	a <= b
!<	It ensures that the left value is not less than the right value.	(a !< b)
!>	It ensures that the left value is not greater than the right value.	(a !> b)

SQL Logical Operators

These operators are best described from the table below:

Operator	Description
All	It compares a value to other values in a set result.
And	It facilitates multiple conditions in an SQL statement.
Any	This operator compares the value as per the set condition.
Between	It identifies values within a given set of values.
In	This operator compares a value to its similar specified value.
Not	This operator reverses other logical operators.
Or	This operator is used to combine multiple values in SQL statements.
Exists	This operator looks for rows from a specific table.
Like	This operator compares values to other similar values.

SQL Queries

Now that we have looked at the different SQL commands and operators, let us delve into one of the most crucial aspects of the SQL, which is the SQL Queries. The core function of the SQL is to execute queries. SQL queries simply mean the formal expressions of a question. Database queries can be in the form of select queries or action queries.

Select query – this is used to retrieve data from the database.
Action query – this facilitates more modification to be done on the data, for example, deleting or inserting.

SQL has numerous queries which will be discussed below with simple examples. This will enhance our understanding.

SQL – INSERT Query

This statement is used to create additional records or rows to a table.
It has two syntaxes which are as follows;

```
INSERT INTO TABLE\_NAME (column 1, column 2…)
VALUES (value 1, value 2…)
```

The columns will be named as per the type of data you want to insert in the table.

SQL Select Query

This SQL statement is used to obtain data from the database table and displays the data as a result table, known as result-sets.

SQL statement syntax is as follows;

```
SELECT column 1, column 2 FROM table\_name
```

The columns represent the fields of the table whose value is to be selected.

SQL – WHERE

This clause is used to verify which particular condition will be set while selecting data from one table or integrating several tables.

It filters out the rows and selects the relevant ones.

This clause is also used with other SQL statements like the delete and update.

The SQL – WHERE syntax is as follows;

```
SELECT column 1, column 2…
FROM table\_name
WHERE (condition)
```

SQL – Update Query

This query is used to modify an existing data in a row. It is used in conjunction with the WHERE clause.

This syntax is as follows;

```
UPDATE table\_name
SET column 1 = value 1
WHERE (condition)
```

SQL - DELETE QUERY

This is used to erase any existing rows from a database table. In order to delete only a specific row or record, this query is used together with the WHERE clause. The basic syntax is as follows;

```
DELETE FROM table\_name
WHERE (condition)
```

SQL – AND / OR Conjunctive Operators

These are used to integrate several conditions to narrow down the data from the SQL statement. They facilitate the comparison of multiple operations within the SQL statement.

The AND Operator

This operator facilitates the existence of several sets of conditions in the WHERE clause.
An example of this syntax is as follows;

```
SELECT column 1, column 2
FROM table\_name
WHERE (condition1) AND (condition 2)…
```

SQL JOIN

This is used to combine several tables. It is also used in the following scenarios: in case a user wants to access several tables from a statement; it is used to combine rows in order to obtain data using the SELECT statement. Combination of the tables depends on the similar fields the tables have.

SQL join is comprised of five types;

- Inner join
- Left outer join
- Right outer join
- Full outer join
- Cross join.

From the above types, the inner join is the most common.

SQL – LIKE Clause

This is used to compare specific values to other related values by the use of wildcard operators. Wildcard characters are used to substitute one or more values. The wildcard operators include;

- The percent sign (%) and the underscore (_).
- A simple syntax of these two is as follows;

  SELECT FROM table_name
  WHERE column LIKE AAA%

SQL – TOP, LIMIT OR ROWNUM Clause

The SQL TOP is used to obtain a certain percentage of records from a table. A simple syntax is as follows;

```
SELECT TOP number / percent / column name
FROM table\_name
WHERE (condition)
```

SQL – ORDER BY Clause

This command is used to arrange the data as per the user's preference, either in ascending or descending order. In other databases, the results are sorted automatically in ascending order. The syntax of this clause is as follows;

```
SELECT column-list
FROM table name
WHERE (condition)
ORDER BY column 1 (ASC | DESC)
```

SQL – GROUP BY Clause

This command is integrated with the SELECT statement to organize similar data into clusters. It is used before making the ORDER BY query and after using the WHERE Clause.

A simple syntax of this clause is as follows;

```
SELECT column 1, column 2
FROM table name
WHERE (condition)
GROUP BY column 1, column 2
ORDER BY column 1, column 2
```

SQL – DISTINCT keyword

This is integrated with the SELECT statement to eradicate all repeated rows by identifying distinct rows.

The basic syntax of DISTINCT keyword is as follows;

```
SELECT DISTINCT column 1, column 2
FROM table name
WHERE (condition)
```

Fundamental Sorting

The records or rows in a table are usually not ordered, therefore you end up viewing the rows randomly. SQL sorting will enable you to arrange the rows as per your preference.

This is enabled by the ORDER BY clause which sorts out either in ascending or descending. Different types of sorting include the following:

Sorting by one column – the syntax is as follows;

```
SELECT column
FROM table name
ORDER BY sort\_column (ASC | DESC)
```

Sorting by multiple columns - The following syntax is used;

```
SELECT column
FROM table
ORDER BY column 1, column 2, column N (ASC | DESC)
```

Sorting by relative column positions - This syntax is as follows;

```
SELECT column
FROM table
ORDER BY sort\_num1 (ASC | DESC); sort\_num2 (ASC |DESC); sort\_num N (ASC |
DESC)
```

Lastly, let us look at SQL replication.

SQL Replication

This is a fundamental aspect of SQL and can be described as a cluster of technologies which facilitate database mirroring and distribution.

SQL replication enables the distribution of data to remote network locations. Replication comprises of two components; the Publisher and Subscriber. In a SQL replication the Publisher displays the data to be distributed. The subscriber updates the data.

In the RDBMS that we saw, the MS SQL type has three categories of SQL Replication. We will look at them briefly;

The Merge replication

This utilizes both the Publisher and the Subscriber when they are required to alter their databases. This simply means that the two sets of databases are altered and then the replication combines the alterations in the two locations. This form of replication can create duplicate primary keys. In such a conflict, there is a set conflict resolution plan that the merge replication uses.

The Transactional replication

This is utilized where there is frequent data changes. These changes require data to be updated to ensure it is relevant. This replication process identifies any changes from the database found in the publisher. In case of any changes, it allocates the data to the subscriber.

The Snapshot replication. This replication is hardly used. It works by taking a photo of the publisher database at designated timeframes. The database is then allocated to the subscriber.

Important points to remember when dealing with any database

- Do not use the number-to-character conversions because these two are compared differently - such a comparison could lead to operation downgrade.
- Anytime you use the SELECT statement, only obtain the necessary information. Never use the * in the SELECT queries as it would lead to unnecessary system load.
- Ensure there are indexes for the tables which have frequent search operations. Try to minimize the number of indexes for tables which have few search operations.
- In case of columns which are not indexed while executing the WHERE clause, a full table scan will occur. In order to prevent this, create indexes on columns which are used as conditions during the execution of WHERE clause in the SQL statement.
- Avoid real values of dates or time when using the equality operators. Such values tend to prevent the results from matching, thus your queries will fail to return records.
- Ensure you have evaluated your SQL query structure and the syntax to identify whether the created tables can support swift data manipulation. Ensure also that the query is written in the best way possible to enable the DBMS to generate effective results.
- Use procedures for the frequent queries made. A procedure is a collection of SQL statements. The database compiles the procedures and executes them.
- The logical operator OR, should be avoided at all costs in any query as it tends to inhibit the query in a table.
- In case you want to maximize on the data load, you can execute the DROP INDEX command. This will enable you to access the tables on the database quicker. The DROP INDEX will

invalidate all the objects which depend on other tables that are not relevant at that moment.

- Create a regular habit of defragmenting the database. This process creates more room for faster accessibility of files, as the files are moved to contiguous clusters.