Lux Tech Academy
Saturday 3rd August 2024

Teacher: Vick
Email: info.victoromondi@gmail.com

Data Science Bootcamp
SQL for Data Analytics

—

Notes

Introduction: Overview

Structured Query Language (SQL) is a standard language that allows for storing, manipulating, and retrieving data from relational database management systems.

These types of databases, relational database management systems, store data in tables which are a collection of related data entries containing columns and rows.

- A record is a horizontal entry in a table while a column is a vertical entity in a table.

There are many types of relational database management systems (RDMS) some of

which include:

- Oracle
- MySQL
- SQL Server
- PostgreSQL
- Microsoft Access

There are minor differences in how the above databases implement SQL, however, how they implement SQL largely remains the same. If you ever implement SQL code that doesn't work, you can always refer to the official documentation of the respective RDMS for clarity.

With SQL we can perform the following basic operations:

- We can execute queries against a database
- We can create new tables in a database
- We can retrieve data from a database
- We can insert records into a database
- We can update records in a database
- We can delete records from a database
- We can create new databases

Applications

SQL is used in almost every industry where significant amounts of data are involved.

- In the finance industry, banking applications store and operate data about financial transactions. This data is usually stored in an SQL Databases i.e. Oracle Database.
- Web applications such as social media platforms like Facebook, Twitter, etc., use SQL to store users' profile information in an SQL Database i.e. Postgres or MySQL Database.
- In the healthcare industry, SQL stores patient medical records data.
- Within businesses, SQL is used to store inventory and transactional data.

Who works with SQL

- Database Developers
- Database Administrators
- Data Analysts
- Business Analysts
- Data Scientists

SQL Fundamentals

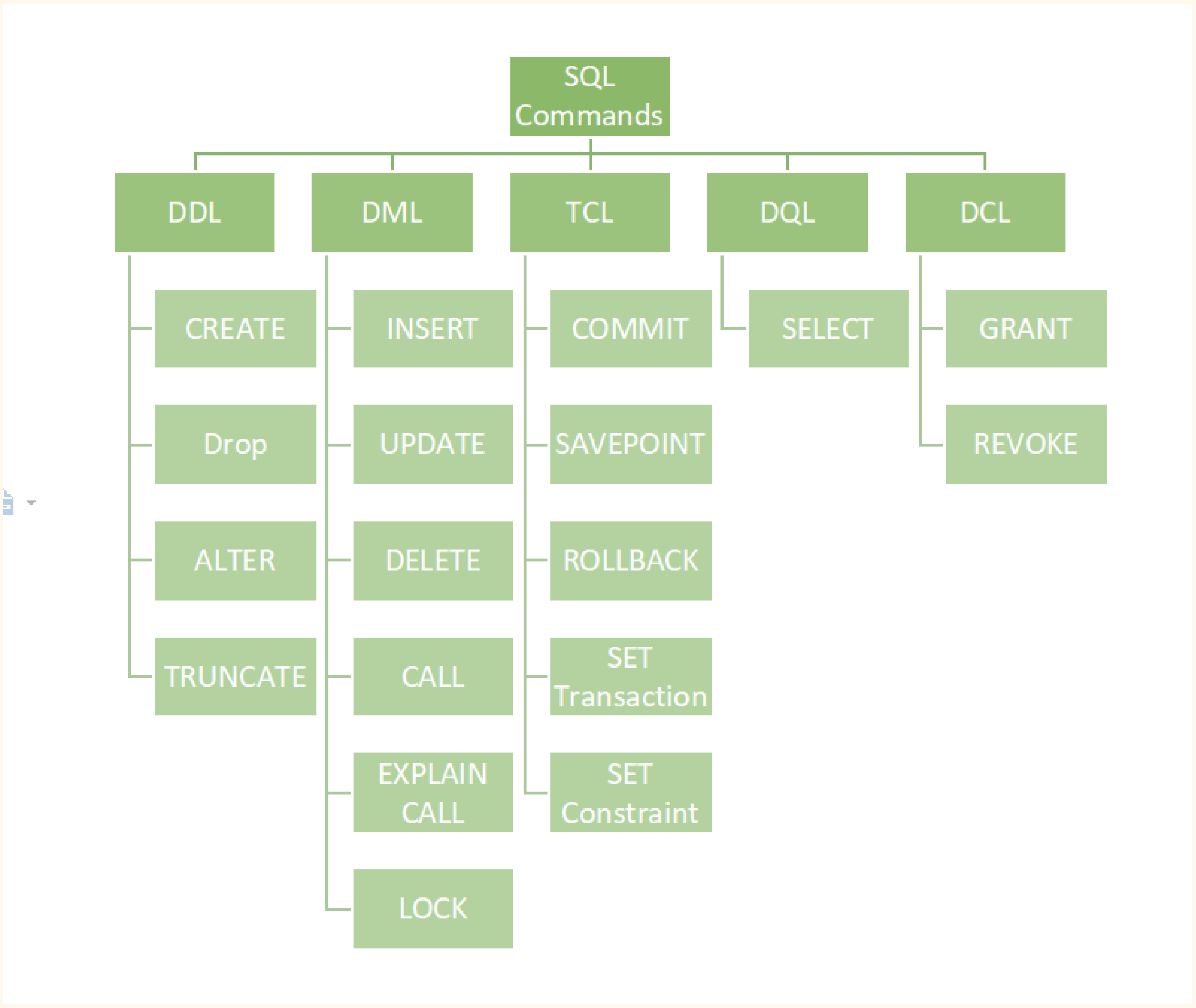
We will cover the following concepts as a start in becoming familiar with SQL:

- Creating a Tables
- Altering a Tables
- Dropping a Table
- Loading a Dataset
- Selecting Data from a Table
- Inserting a Table
- Updating Data in a Table
- Performing Calculations with SQL
- Subqueries

- Joining Tables

Prerequisites

1. For this session, we will use the Programiz SQL Online Compiler: https://www.programiz.com/sql/online-compiler/



Creating SQL Tables

As mentioned, while working with RDMS, we use tables to store data. The `CREATE TABLE` statement is used to create a new table in an RDMS database.

The following example creates a table called "Students" that contains four columns:

```
Unset
CREATE TABLE Students (
      StudentID int,
      FirstName varchar(255),
      LastName varchar(255),
      Class varchar(255),
);
```

The `StudentID` column is of type int and will hold an integer.

The `FirstName`, `LastName`, and `Class` columns are of type varchar and will hold characters, and the maximum length for these fields is 255 characters.

Altering SQL Tables

We use The `ALTER TABLE` statement to add, delete, or modify columns in an existing

table.

The following SQL adds an "Email" column to the "Students" table:

```
Unset

ALTER TABLE Students

ADD Email varchar(255);
```

Dropping a Table

We use the `DROP TABLE` statement to drop/delete an existing table in a database.

```
Unset

DROP TABLE Students;
```

Loading our Datasets

In the context where we would like to select data from a dataset and import it in our notebook, we can easily use pandas for importation and then load our dataset to a new table through the use of SQL as shown;

```
Unset

cities = pd.read_csv('http://bit.ly/CitiesDB')

%sql DROP TABLE if EXISTS cities;

%sql PERSIST cities;
```

We use the persist command to create a table in the database to which we are connected, with the table name being the same as the data frame variable.

Selecting Data from a Table

The `SELECT` statement is used to fetch records from a table.

The following SQL statement selects the all records from the "Students" table:

```
Unset

SELECT * FROM Students;
```

We can also select the first three records from the Students table as shown below:

```
Unset

SELECT * FROM Students

LIMIT 3;
```

Inserting to a Table

The `INSERT INTO` statement is used to insert new records in a table. The following SQL statement inserts a new record in the "Students" table:

```
Unset
INSERT INTO Customers (StudentID, FirstName, LastName,
Class)
VALUES ('ST001', Victor, Omondi, 'Data Science');
```

Updating Data in a Table

We can use the `UPDATE` statement is used to modify the existing records in a table.

The following SQL statement updates the student with the given ID (StudentID = ST001) with a new class.

```
Unset
UPDATE Students
SET Class= 'Frankfurt'
WHERE CustomerID = 1;
```

Performing Calculations with SQL

We can perform calculations in an SQL statement through the use of an arithmetic expression.

For example, the following SQL statement calculates the total price before the discount and after discount for each order item.

```
Unset
SELECT
    OrderID,
    ProductID,
    UnitPrice*Quantity AS "Regular Price",
    UnitPrice*Quantity-UnitPrice*Quantity*Discount AS "Price
After Discount"
FROM order_details;
```

Aggregates

An aggregate function is a function that performs a calculation on a set of values and returns a single value.

Aggregate functions are often used with the GROUP BY clause of the SELECT statement. The GROUP BY clause splits the result-set into groups of values and the aggregate function can be used to return a single value for each group.

The most commonly used SQL aggregate functions are:

- `MIN()` - returns the smallest value within the selected column
- `MAX()` - returns the largest value within the selected column
- `COUNT()` - returns the number of rows in a set
- SUM() - returns the total sum of a numerical column
- AVG() - returns the average value of a numerical column

Subqueries

Subqueries allow us to contain SQL statements within SQL statements. This allows us to perform queries that can join two tables as shown below. The following subquery will list products with order quantities greater than 100. Note the two tables Product and OrderItem.

```
Unset
SELECT
    ProductName
FROM Product
WHERE Id IN (
    SELECT ProductId
    FROM OrderItem
    WHERE Quantity > 100)
```

Joining Tables

An SQL `JOIN` statement combines records from two tables.

In the following example, the "StudentID" column in the "Grades" table refers to the "StudentID" in the "Books" table. The relationship between the two tables is the "StudentID" column.

```
Unset
SELECT Grades.BookID, Books.FirstName, Grades.BookDate
FROM Grades
INNER JOIN Books ON Grades.StudentID=Books.StudentID;
```

References

- SQL Introduction to Beginners. Link: https://learntocodewith.me/posts/sql-guide/
- SQL. W3Schools. Link: https://www.w3schools.com/sql/