PORA ACADEMY

Data Science

Introduction to SQL

Introduction to SQL: A Beginner’s Guide

This document will guide you through the basics of Structured Query Language (SQL). We’ll start with the fundamentals and slowly work our way up to more advanced concepts

What is a Database?

Imagine a filing cabinet, that’s exactly what a database is- an organised collection of information.

A **database** is an organized collection of data that can be accessed, managed, and updated.

Real-World Examples:

- **Banking Systems:** SQL is used to manage customer transactions, account balances, and fraud detection.
- **E-Commerce Websites:** SQL helps store and retrieve product information, customer orders, and payment details.
- **Social Media Platforms:** SQL is used for storing user profiles, posts, comments, and interactions between users.

What is SQL (Structured Query Language)

SQL (Structured Query Language) is used to communicate with databases.

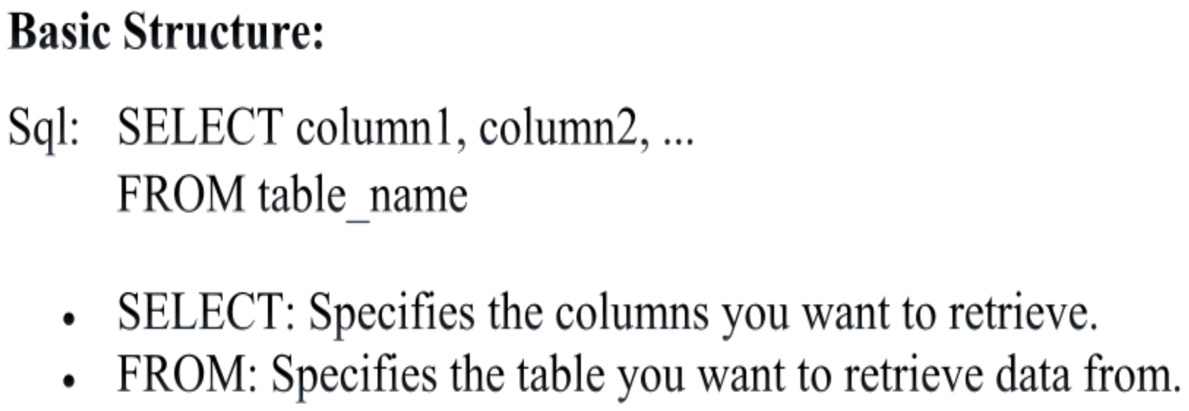
It allows us to:

- Retrieve Data
- Insert new Data
- Update existing Data
- Delete Data

- It is the standard language for relational database management systems (RDBMS).
- **SQL vs. DBMS:**
  - SQL: A language for querying data.
  - DBMS: Software that manages databases (MySQL, PostgreSQL, SQL Server).

Basic SQL Syntax: The SELECT Statement

The most fundamental SQL command is the ‘Select’ statement. It is used to retrieve data from a table.



Basic Structure:

```
Sql:   SELECT column1, column2, ...
       FROM table_name
```

- SELECT: Specifies the columns you want to retrieve.
- FROM: Specifies the table you want to retrieve data from.

Example:

Let's say we have a Customers table:

CustomerID	FirstName	LastName	City
1	John	Doe	New York
2	Jane	Smith	London
3	David	Lee	Paris

To retrieve the FirstName and LastName of all customers, we would use:

```
SELECT FirstName, LastName
FROM Customers;
```

Result:

FirstName	LastName
John	Doe
Jane	Smith
David	Lee

Selecting All Columns:

To select all columns, use the * wildcard:

```
SELECT *
FROM Customers;
```

Filtering Data: The Where Clause

The WHERE clause allows us to filter data based on specific conditions.

Example: To retrieve the customers who live in 'London':

Sql:
```
SELECT * FROM Customers
WHERE City = 'London';
```

Comparison Operators:

=	Equal to
!= or <>	Not equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Logical Operators:

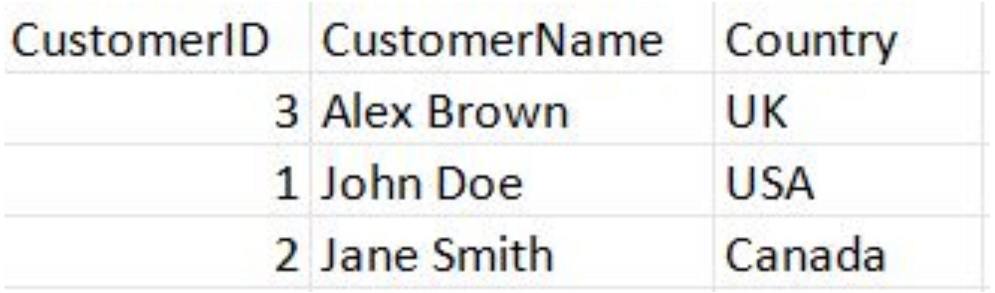
- AND: Combines multiple conditions (both must be true).
- OR: Combines multiple conditions (at least one must be true).
- NOT: Negates a condition.

Sorting Data - Order BY Clause

The ‘ORDER BY’ clause sorts the result set.

Example:
SELECT * FROM Customers ORDER BY CustomerName ASC;

Sorts customers by name in ascending order.



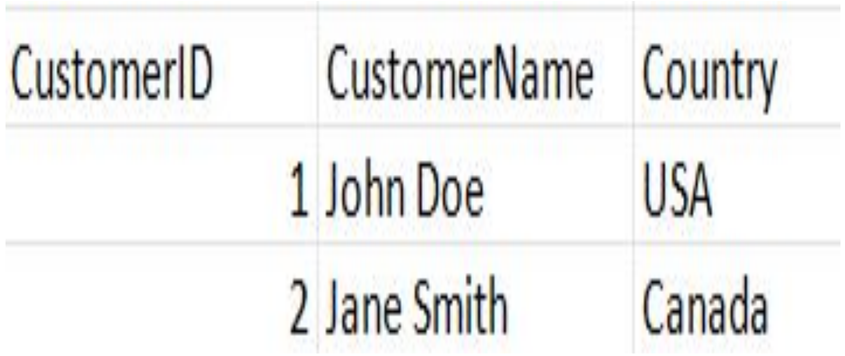
CustomerID	CustomerName	Country
3	Alex Brown	UK
1	John Doe	USA
2	Jane Smith	Canada

Limiting Results: LIMIT/TOP

- LIMIT (MySQL, PostgreSQL, SQLite): Limits the number of rows returned.
- TOP (SQL Server): Limits the number of rows returned.

Example (MySQL/PostgreSQL/SQLite):

Sql:
```
SELECT * FROM Customers
LIMIT 2;
```



CustomerID	CustomerName	Country
1	John Doe	USA
2	Jane Smith	Canada

Example (SQL Server):

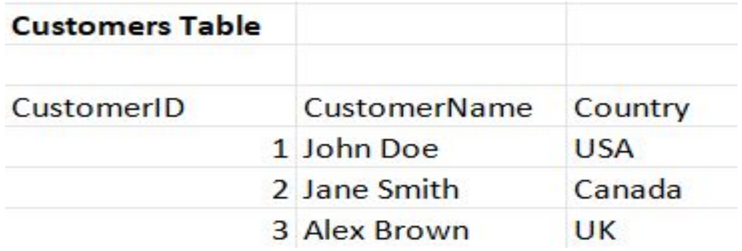
Sql:
```
SELECT TOP 2 * FROM Customers;
```

Joining Tables

‘JOIN’ combines records from two or more tables.

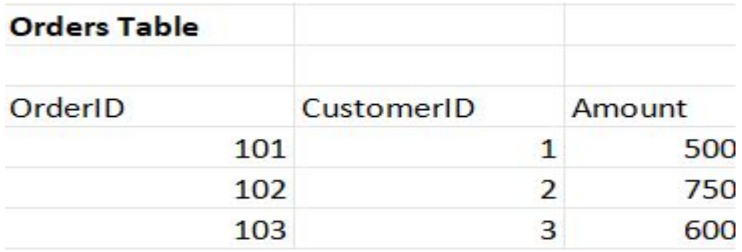
Example:
SELECT Orders.OrderID, Customers.CustomerName FROM Orders INNER JOIN Customers ON
Orders.CustomerID = Customers.CustomerID;

- Fetches orders along with customer names.



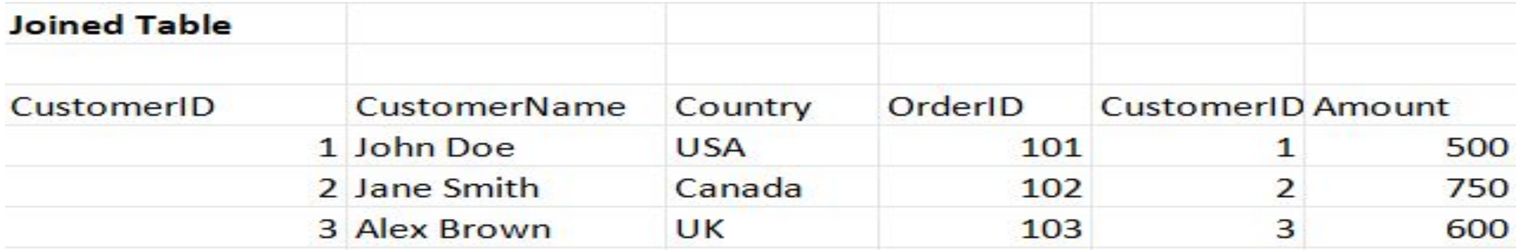
Customers Table

CustomerID	CustomerName	Country
1	John Doe	USA
2	Jane Smith	Canada
3	Alex Brown	UK



Orders Table

OrderID	CustomerID	Amount
101	1	500
102	2	750
103	3	600



Joined Table

CustomerID	CustomerName	Country	OrderID	CustomerID	Amount
1	John Doe	USA	101	1	500
2	Jane Smith	Canada	102	2	750
3	Alex Brown	UK	103	3	600

INNER JOIN: Returns rows where there is a match in both tables.

```
SELECT Customers.FirstName, Orders.OrderDate

FROM Customers

INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

Explanation:

- We select FirstName from Customers and OrderDate from Orders.
- We use INNER JOIN to combine the tables.
- The ON clause specifies the join condition (matching CustomerIDs).

Left Join: Returns all rows from the left table, and the matched rows from the right table.

Right Join: Returns all rows from the right table, and the matched rows from the left table.

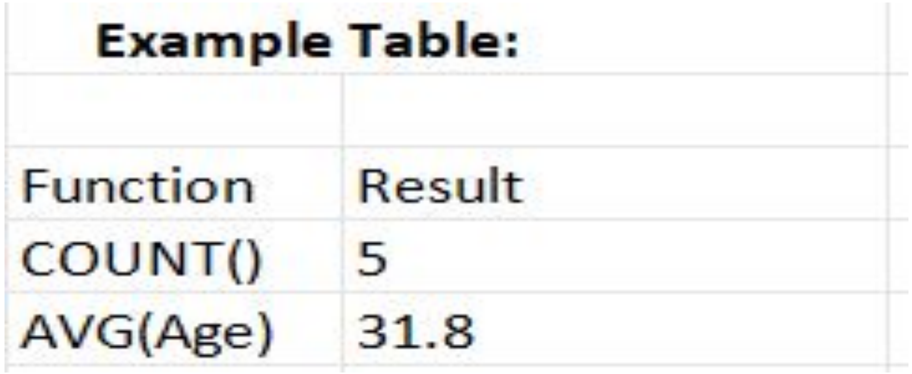
Aggregate Functions

Aggregate functions perform calculations on multiple rows.

- **Common Functions:** `COUNT()`, `SUM()`, `AVG()`, `MAX()`, `MIN()`.

  Example:
  SELECT COUNT(CustomerID) FROM Customers;

  Counts the number of customers.



Example Table:

Function	Result
COUNT()	5
AVG(Age)	31.8

Grouping Data: The GROUP BY Clause

- Used to group records with similar values.

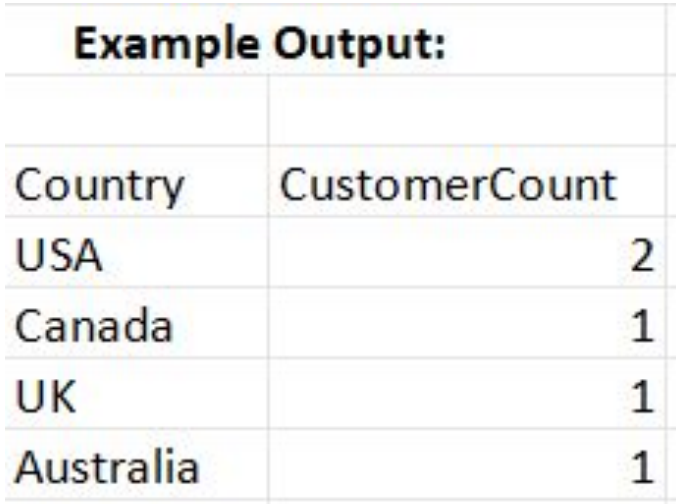
  Example:

  ```
  SELECT Country, COUNT(CustomerID)

   FROM Customers

  GROUP BY Country;
  ```

- Counts customers by country.



Example Output:

Country	CustomerCount
USA	2
Canada	1
UK	1
Australia	1

Data Manipulation: INSERT, UPDATE, DELETE

- INSERT: Adds new rows to a table.
- UPDATE: Modifies existing rows.
- DELETE: Removes rows from a table.

Example INSERT:

```
Sql:  INSERT INTO Customers (FirstName, LastName, City)

      VALUES ('Alice', 'Johnson', 'Berlin');
```

Example UPDATE:

```
Sql:  UPDATE Customers

      SET City = 'Rome'

      WHERE CustomerID = 1;
```

Example DELETE:

```
Sql:  DELETE FROM Customers

      WHERE CustomerID = 3;
```

Important: Always use a WHERE clause with UPDATE and DELETE to avoid accidentally modifying or deleting all rows.

Subqueries

A subquery is a query inside another query.

Example:

```
SELECT CustomerName FROM Customers

WHERE CustomerID IN (SELECT CustomerID FROM Orders WHERE Amount > $500);
```

- Finds customers who placed orders above $500.

Conclusions and Next Steps

- SQL is essential for managing relational databases.
- Learn **Advanced SQL:** Stored procedures, indexing, performance tuning.
- Explore SQL with **practice platforms** like SQLZoo, LeetCode, W3Schools.
- **Q&A Session**.

PORA
ACADEMY

THANK YOU!