# SQL (Structured Query Language)

## Create Table, Alter Table, Drop Table

CREATE TABLE customer (

customer\_id INT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

email VARCHAR(100),

phone\_number VARCHAR(20),

address VARCHAR(255),

city VARCHAR(100),

state VARCHAR(50),

postal\_code VARCHAR(20),

country VARCHAR (100));

ALTER TABLE customer

ADD birthdate DATE;

DROP TABLE customer;

## INSERT, UPDATE, DELETE

INSERT INTO customer (customer\_id, first\_name, last\_name, email, phone\_number, address, city, state, postal\_code, country)

VALUES (5, 'Emily', 'Davis', 'emily@example.com', '+188877766655', '210 Maple St', 'Villageton', 'OH', '54321', 'United States');

UPDATE customer

SET last\_name = 'Johnson'

WHERE customer\_id = 2;

DELETE FROM customer

WHERE customer\_id = 3;

## Select

SELECT \* FROM Customers;

SELECT DISTINCT Country FROM Customers;

SELECT COUNT(DISTINCT Country) FROM Customers; --gibt Anzahl der eindeutigen Länder zurück

SELECT \* FROM Customers  
WHERE Country='Mexico';

--Between, Like, In

SELECT \* FROM customers

WHERE age BETWEEN 20 AND 30;

SELECT \* FROM customers

WHERE last\_name LIKE 'Smith%';

SELECT \* FROM customers

WHERE city IN ('New York', 'Los Angeles', 'Chicago');

SELECT \* FROM Products

ORDER BY Price;

SELECT MIN(Price)  
FROM Products;

SELECT MAX(Price)  
FROM Products;

SELECT COUNT(ProductID)  
FROM Products  
WHERE Price > 20;

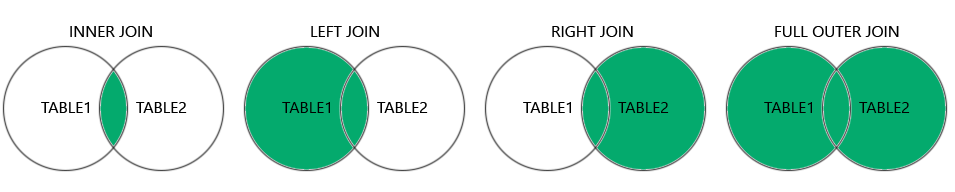
SELECT SUM(Quantity)  
FROM OrderDetails;

SELECT AVG(Price)  
FROM Products;

## Join

SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate  
FROM Orders  
INNER JOIN Customers

ON Orders.CustomerID=Customers.CustomerID;



SQL Self Join

SELECT e.employee\_name AS employee\_name,

m.employee\_name AS manager\_name

FROM Employees e

JOIN Employees m

ON e.manager\_id = m.employee\_id;

Dadurch erhalten wir die Namen der Mitarbeiter und ihrer jeweiligen Manager in der Ausgabe.

## Union

SELECT City FROM Customers  
UNION  
SELECT City FROM Suppliers  
ORDER BY City;

Gibt die Städte (nur unterschiedliche Werte) sowohl aus der Tabelle „Kunden“ als auch aus der Tabelle „Lieferanten“ zurück.

SELECT City FROM Customers  
UNION ALL  
SELECT City FROM Suppliers  
ORDER BY City;

Gibt die Städte (auch doppelte Werte) aus der Tabelle „Kunden“ und „Lieferanten“ zurück.

## Group By

SELECT column\_name(s)

FROM table\_name

WHERE condition

GROUP BY column\_name(s)

ORDER BY column\_name(s);

SELECT COUNT(CustomerID), Country

FROM Customers

GROUP BY Country

ORDER BY COUNT(CustomerID) DESC;

## Having

SELECT column\_name(s)  
FROM table\_name  
WHERE condition  
GROUP BY column\_name(s)  
HAVING condition  
ORDER BY column\_name(s);

Die HAVING-Klausel wurde zu SQL hinzugefügt, da das Schlüsselwort WHERE nicht mit Aggregatfunktionen verwendet werden kann.

SELECT COUNT(CustomerID), Country  
FROM Customers  
GROUP BY Country  
HAVING COUNT(CustomerID) > 5  
ORDER BY COUNT(CustomerID) DESC;

## Any, All

SQL ANY und ALL sind Vergleichsoperatoren, die häufig in Verbindung mit Unterabfragen verwendet werden, um Bedingungen zu überprüfen.

SELECT ProductID, Price

FROM Products

WHERE Price > ALL (SELECT AVG(Price) FROM Products);

SELECT ProductID, Price

FROM Products

WHERE Price >= ANY (SELECT Price FROM Products WHERE ProductID <> outer.Products.ProductID);

## CASE

SELECT OrderID, Quantity,  
CASE  
    WHEN Quantity > 30 THEN 'The quantity is greater than 30'  
    WHEN Quantity = 30 THEN 'The quantity is 30'  
    ELSE 'The quantity is under 30'  
END AS QuantityText  
FROM OrderDetails;

## With

In SQL wird die WITH-Klausel verwendet, um sogenannte "Common Table Expressions" (CTEs) zu erstellen. CTEs ermöglichen es, temporäre Resultsets zu definieren, die innerhalb einer einzelnen Abfrage verwendet werden können.

WITH EmployeeDepartments AS (

SELECT

e.FirstName || ' ' || e.LastName AS EmployeeName,

d.DepartmentName,

COUNT(\*) AS EmployeeCount

FROM

Employees e

INNER JOIN

Departments d ON e.DepartmentID = d.DepartmentID

GROUP BY

e.FirstName, e.LastName, d.DepartmentName

)

SELECT

ed.EmployeeName,

ed.DepartmentName,

ed.EmployeeCount

FROM

EmployeeDepartments ed;

## PL SQL

DECLARE

<declarations section>

BEGIN

<executable command(s)>

EXCEPTION

<exception handling>

END;

set serveroutput on

DECLARE

message varchar2(20):= 'Hello, World!';

BEGIN

dbms\_output.put\_line(message);

END;

DECLARE

a integer := 10;

b integer := 20;

c integer;

f real;

BEGIN

c := a + b;

dbms\_output.put\_line('Value of c: ' || c);

f := 70.0/3.0;

dbms\_output.put\_line('Value of f: ' || f);

END;

https://www.tutorialspoint.com/plsql/plsql\_variable\_types.htm