**SQL Basic for QA**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name: customers | | | |
| CustomerID | custName | age | phone |
| 001 | Cust1 | 20 | 018471 |
| 002 | Mahir | 15 | 0214157 |
| 003 | Abdul | 35 | 5247185 |
| 004 | Mahir | 21 | 8457145 |

1. Find all data of the table

SELECT \* FROM customers;

1. Find Specific Column/(S)

SELECT custName, age FROM customers;

1. Find Distinc value of a table

SELECT DISTINCT custName FROM Customers;

1. Count the total number of rows

SELECT COUNT (DISTINCT custName) FROM Customers;

1. Finding Data with a condition (<,>,=, <=, >= | IS NULL )

SELECT \* FROM Customers  
WHERE CustomerID=001;

OR

SELECT \* FROM Customers  
WHERE CustomerID>003;

OR

SELECT CustomerName, ContactName, Address  
FROM Customers  
WHERE Address IS NULL;

1. Finding data with Multiple condition (AND | OR | NOT)

SELECT \* FROM Customers  
WHERE custName=Mahir AND age=21;

1. Finding data of Multiple Value (IN | NOT IN)

SELECT \* FROM Customers  
WHERE age IN (10, 21, 35);

OR

SELECT \* FROM Customers  
WHERE Country IN (SELECT Country FROM Suppliers);

1. Finding Data in Between a range

SELECT \* FROM customers  
WHERE age BETWEEN 10 AND 30;

OR

SELECT \* FROM Orders  
WHERE OrderDate BETWEEN '1996-07-01' AND '1996-07-31';

OR

SELECT \* FROM customers  
WHERE age BETWEEN 10 AND 30;  
AND customerID NOT IN (1,2,3);

1. Show Data With temporary Column Name

SELECT CustomerID AS ID, CustomerName AS Customer  
FROM Customers;

OR

SELECT CustomerID AS ID, CustomerName AS [Customer Name]  
FROM Customers;

OR

SELECT CustomerName, age + ', ' + phone+ AS Info  
FROM Customers;

1. Finding Data with Order By

SELECT \* FROM Customers  
ORDER BY custName;

OR

SELECT \* FROM Customers  
ORDER BY age DESC;

OR

SELECT \* FROM Customers  
ORDER BY custName ASC, CustomerName DESC;

1. Finding MAX, MIN value of a coumn

SELECT MAX(age) AS Elder  
FROM customers;

OR

SELECT MIN(age) AS younger  
FROM Products WHERE age >10;

1. Insert data into a table

INSERT INTO Customers (custName, age, phone)  
VALUES (Ashraf, 28, 5874571);

1. Update Data of a row

UPDATE Customers  
SET cuatName = 'Alfred Schmidt', age= 11  
WHERE customerID = 001;

1. Update All row of a Table

UPDATE Customers  
SET customerID=0003;

1. Delete row from a table (all operator of Where)

DELETE FROM customers WHERE CustName='Mahir’;

1. Delete All Data from a table

DELETE FROM Customers;

1. Finding data of similar text of a column

(starting with "a": a% | End With “a”:’%a’ | have "or" in any position: '%or%'| starts with "a" and ends with "o": 'a%o’)

SELECT \* FROM Customers  
WHERE CustomerName LIKE 'a%';

OR

SELECT \* FROM Customers  
WHERE CustomerName NOT LIKE 'a%';

1. Perform Operation on a Colum (COUNT, AVG, SUM)

SELECT COUNT(customerID)  
FROM customers;

OR

SELECT AVG(age)  
FROM customers;

OR

SELECT SUM(age)  
FROM custmers;

**SQL Join**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table: Customers** | | | |  | **Table: Orders** | | |
| **CustID** | **CustName** | **Country** | **Phone** | **OrderID** | **CustID** | **Date** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

## **Different Types of SQL JOINs**

Here are the different types of the JOINs in SQL:

* **(INNER) JOIN**: Returns records that have matching values in both tables
* **LEFT (OUTER) JOIN**: Returns all records from the left table, and the matched records from the right table
* **RIGHT (OUTER) JOIN**: Returns all records from the right table, and the matched records from the left table
* **FULL (OUTER) JOIN**: Returns all records when there is a match in either left or right table

      

1. Inner Join

The INNER JOIN keyword selects records that have matching values in both tables.

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

**OR**

SELECT Orders.OrderID, Customers.CustomerName, Shippers.ShipperName  
FROM ((Orders  
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID)  
INNER JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID);

1. Left Join

The LEFT JOIN keyword returns all records from the left table (table1), and the matched records from the right table (table2). The result is NULL from the right side, if there is no match.

SELECT Customers.CustomerName, Orders.OrderID  
FROM Customers  
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID  
ORDER BY Customers.CustomerName;

1. The RIGHT JOIN keyword returns all records from the right table (table2), and the matched records from the left table (table1). The result is NULL from the left side, when there is no match.

SELECT Orders.OrderID, Employees.LastName, Employees.FirstName  
FROM Orders  
RIGHT JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID  
ORDER BY Orders.OrderID;

1. The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.

**Note:** FULL OUTER JOIN can potentially return very large result-sets!

**Tip:** FULL OUTER JOIN and FULL JOIN are the same.

SELECT Customers.CustomerName, Orders.OrderID  
FROM Customers  
FULL OUTER JOIN Orders ON Customers.CustomerID=Orders.CustomerID  
ORDER BY Customers.CustomerName

1. A self-JOIN is a regular join, but the table is joined with itself.

SELECT A.CustomerName AS CustomerName1, B.CustomerName AS CustomerName2, A.City  
FROM Customers A, Customers B  
WHERE A.CustomerID <> B.CustomerID  
AND A.City = B.City  
ORDER BY A.City;

1. Group BY

The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country"

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

1. Create Database

CREATE DATABASE databasename;

1. Drop Database

DROP DATABASE databasename;

1. Create Table

CREATE TABLE Persons (  
    PersonID int,  
    LastName varchar(255),  
    FirstName varchar(255),  
    Address varchar(255),  
    City varchar(255)  
);

1. Drop Table

DROP TABLE table\_name;

1. Alter Table

The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.

The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

ALTER TABLE Customers  
ADD Email varchar(255);

OR

ALTER TABLE table\_name  
DROP COLUMN column\_name;

OR (change the data type of a column in a table)

ALTER TABLE table\_name  
ALTER COLUMN column\_name datatype;

OR