Bangladesh Open University

Diploma in Computer Science and Application Program (DCSA)

**Database Management System**

TMA – 2

50 SQL commands

Submitted by

MD JASHEDUL ISLAM SHAUN

ID : 21150890015

Here are 50 SQL commands with examples:

1. SELECT: Retrieves data from a database table. Example: SELECT \* FROM Customers;
2. INSERT: Inserts data into a database table. Example: INSERT INTO Employees (Name, Age) VALUES ('John Doe', 30);
3. UPDATE: Modifies data in a database table. Example: UPDATE Products SET Price = 15.99 WHERE ProductID = 1;
4. DELETE: Removes data from a database table. Example: DELETE FROM Orders WHERE OrderID = 100;
5. CREATE TABLE: Creates a new table in the database. Example: CREATE TABLE Employees (EmployeeID INT, Name VARCHAR(50));
6. ALTER TABLE: Modifies an existing table structure. Example: ALTER TABLE Customers ADD COLUMN Email VARCHAR(100);
7. DROP TABLE: Deletes a table from the database. Example: DROP TABLE Customers;
8. CREATE INDEX: Creates an index on a table for faster data retrieval. Example: CREATE INDEX idx\_lastname ON Employees (Lastname);
9. DROP INDEX: Removes an index from a table. Example: DROP INDEX idx\_lastname ON Employees;
10. CREATE VIEW: Creates a virtual table based on the result of a query. Example: CREATE VIEW HighValueOrders AS SELECT \* FROM Orders WHERE TotalAmount > 1000;
11. DROP VIEW: Deletes a view from the database. Example: DROP VIEW HighValueOrders;
12. CREATE PROCEDURE: Creates a stored procedure. Example: CREATE PROCEDURE GetCustomers AS SELECT \* FROM Customers;
13. DROP PROCEDURE: Deletes a stored procedure. Example: DROP PROCEDURE GetCustomers;
14. CREATE TRIGGER: Creates a trigger that automatically executes actions when a specified event occurs. Example: CREATE TRIGGER UpdateStock AFTER INSERT ON Orders FOR EACH ROW UPDATE Products SET Stock = Stock - NEW.Quantity;
15. DROP TRIGGER: Removes a trigger from a table. Example: DROP TRIGGER UpdateStock;
16. GRANT: Provides user privileges to access database objects. Example: GRANT SELECT, INSERT ON Customers TO John;
17. REVOKE: Removes user privileges from accessing database objects. Example: REVOKE SELECT, INSERT ON Customers FROM John;
18. UNION: Combines the result sets of two or more SELECT statements. Example: SELECT \* FROM Customers UNION SELECT \* FROM Suppliers;
19. INNER JOIN: Retrieves records that have matching values in both tables. Example: SELECT Orders.OrderID, Customers.CustomerName FROM Orders INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
20. LEFT JOIN: Retrieves all records from the left table and the matching records from the right table. Example: SELECT Customers.CustomerName, Orders.OrderID FROM Customers LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
21. RIGHT JOIN: Retrieves all records from the right table and the matching records from the left table. Example: SELECT Customers.CustomerName, Orders.OrderID FROM Customers RIGHT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
22. FULL OUTER JOIN: Retrieves all records when there is a match in either left or right table. Example: SELECT Customers.CustomerName, Orders.OrderID FROM Customers FULL OUTER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
23. WHERE: Filters records based on specified conditions. Example: SELECT \* FROM Customers WHERE Country = 'USA';
24. ORDER BY: Sorts the result set in ascending or descending order. Example: SELECT \* FROM Customers ORDER BY CustomerName ASC;
25. GROUP BY: Groups the result set based on one or more columns. Example: SELECT Country, COUNT(\*) FROM Customers GROUP BY Country;
26. HAVING: Filters records based on aggregated values. Example: SELECT Country, COUNT(*) FROM Customers GROUP BY Country HAVING COUNT(*) > 10;
27. DISTINCT: Retrieves unique values from a column. Example: SELECT DISTINCT Country FROM Customers;
28. COUNT: Calculates the number of rows in a result set. Example: SELECT COUNT(\*) FROM Customers;
29. AVG: Calculates the average value of a column. Example: SELECT AVG(Price) FROM Products;
30. SUM: Calculates the sum of values in a column. Example: SELECT SUM(Quantity) FROM Orders;
31. MIN: Retrieves the minimum value from a column. Example: SELECT MIN(Price) FROM Products;
32. MAX: Retrieves the maximum value from a column. Example: SELECT MAX(Price) FROM Products;
33. LIKE: Filters records based on a specified pattern. Example: SELECT \* FROM Customers WHERE CustomerName LIKE 'A%';
34. IN: Checks if a value exists in a list. Example: SELECT \* FROM Customers WHERE Country IN ('USA', 'Canada');
35. BETWEEN: Retrieves records within a specified range. Example: SELECT \* FROM Orders WHERE OrderDate BETWEEN '2022-01-01' AND '2022-12-31';
36. NULL: Checks for NULL values in a column. Example: SELECT \* FROM Customers WHERE ContactName IS NULL;
37. EXISTS: Checks if a subquery returns any rows. Example: SELECT \* FROM Customers WHERE EXISTS (SELECT \* FROM Orders WHERE Orders.CustomerID = Customers.CustomerID);
38. NOT: Negates a condition. Example: SELECT \* FROM Customers WHERE NOT Country = 'USA';
39. CASE: Performs conditional logic in a query. Example: SELECT OrderID, Quantity, CASE WHEN Quantity > 10 THEN 'High' ELSE 'Low' END AS 'QuantityStatus' FROM Orders;
40. SUBSTRING: Extracts a substring from a string. Example: SELECT SUBSTRING(CustomerName, 1, 3) FROM Customers;
41. CONCAT: Concatenates two or more strings. Example: SELECT CONCAT(FirstName, ' ', LastName) AS 'FullName' FROM Employees;
42. DATEFORMAT: Formats a date value. Example: SELECT OrderID, FORMAT(OrderDate, 'yyyy-MM-dd') AS 'FormattedDate' FROM Orders;
43. UPPER: Converts a string to uppercase. Example: SELECT UPPER(ProductName) FROM Products;
44. LOWER: Converts a string to lowercase. Example: SELECT LOWER(ProductName) FROM Products;
45. ROUND: Rounds a numeric value to a specified number of decimal places. Example: SELECT ROUND(Price, 2) FROM Products;
46. TRIM: Removes leading and trailing spaces from a string. Example: SELECT TRIM(ProductName) FROM Products;
47. TOP: Limits the number of rows returned in a query. Example: SELECT TOP 10 \* FROM Customers;
48. ROW\_NUMBER: Assigns a unique sequential number to each row in a result set. Example: SELECT ROW\_NUMBER() OVER(ORDER BY ProductID) AS 'RowNumber', ProductName FROM Products;
49. COALESCE: Returns the first non-null value in a list. Example: SELECT COALESCE(Discount, 0) FROM Products;
50. UNION ALL: Combines the result sets of two or more SELECT statements, including duplicate rows. Example: SELECT \* FROM Customers UNION ALL SELECT \* FROM Suppliers;

These examples provide a glimpse of the versatility and power of SQL commands in manipulating and retrieving data from a database.