This command use to backup database when NO insertion is allowed

RESTORE DATABASE triggers

FROM DISK = 'H:\triggers.bak'

WITH REPLACE;

-- Database Commands

-- 1. CREATE DATABASE: Create a new database.

CREATE DATABASE MyDatabase;

-- 2. ALTER DATABASE: Modify database properties.

ALTER DATABASE MyDatabase SET READ\_COMMITTED\_SNAPSHOT ON;

-- 3. DROP DATABASE: Delete a database.

DROP DATABASE MyDatabase;

-- 4. DATABASE RENAME

sp\_renamedb 'rise', 'rise1';

-- Table Commands

-- 5. CREATE TABLE: Create a new table.

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DepartmentID INT

);

-- 6. ALTER TABLE: Modify an existing table.

ALTER TABLE Employees ADD Email VARCHAR(100);

-- 7. DROP TABLE: Delete a table.

DROP TABLE Employees;

-- Data Manipulation Commands

-- 8. INSERT INTO: Insert data into a table.

INSERT INTO Employees (EmployeeID, FirstName, LastName, DepartmentID) VALUES (1, 'John', 'Doe', 101);

-- 9. UPDATE: Update existing records in a table.

UPDATE Employees SET DepartmentID = 102 WHERE LastName = 'Doe';

-- 10. Pivot Table

-- ---pivot

SELECT DISTINCT customer\_name, country, SUM(profit) AS profit FROM sales\_data\_final

GROUP BY customer\_name, country

ORDER BY customer\_name, country;

-- 11. DELETE: Delete row/records from a table.

DELETE FROM Employees WHERE EmployeeID = 1;

-- 12. TABLE RENAMING QUERY in MS SQL

sp\_rename 'OldTableName', 'NewTableName';

-- 13. Table Rename Column

sp\_rename 'Table\_Name.ColumnName', 'New\_Column\_Name';

-- 14. DATABASE RENAME

ALTER DATABASE (OLD NAME) MODIFY NAME = (NEW NAME);

-- Query Commands

-- 15. Retrieve distinct values:

SELECT DISTINCT DepartmentID FROM Employees;

-- 16. Specify the source of data for a query:

SELECT FirstName, LastName FROM Employees;

-- 17. Combine rows from multiple tables with INNER JOIN:

SELECT Employees.FirstName, Departments.DepartmentName FROM Employees

INNER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID;

-- 18. Filter rows using WHERE:

SELECT FirstName, LastName FROM Employees WHERE DepartmentID = 102;

-- 19. Group rows using GROUP BY:

SELECT DepartmentID, COUNT(\*) AS EmployeeCount FROM Employees GROUP BY DepartmentID;

-- 20. Filter grouped rows using HAVING:

SELECT DepartmentID, COUNT(\*) AS EmployeeCount FROM Employees GROUP BY DepartmentID HAVING COUNT(\*) > 5;

-- 21. Sort rows using ORDER BY:

SELECT FirstName, LastName FROM Employees ORDER BY LastName, FirstName;

-- 22. Limit the number of rows using OFFSET and FETCH (SQL Server 2012+):

SELECT FirstName, LastName FROM Employees ORDER BY LastName, FirstName OFFSET 10 ROWS FETCH NEXT 5 ROWS ONLY;

-- 23. Retrieve a specified number of rows using TOP (SQL Server):

SELECT TOP 5 FirstName, LastName FROM Employees ORDER BY LastName, FirstName;

-- Aggregate Functions

-- 24. Count the number of rows:

SELECT COUNT(\*) AS TotalEmployees FROM Employees;

-- 25. Calculate the sum of values in a column:

SELECT SUM(Salary) AS TotalSalary FROM EmployeeSalaries;

-- 26. Calculate the average of values in a column:

SELECT AVG(Age) AS AverageAge FROM Persons;

-- 27. Find the minimum value in a column:

SELECT MIN(Price) AS LowestPrice FROM Products;

-- 28. Find the maximum value in a column:

SELECT MAX(Salary) AS HighestSalary FROM EmployeeSalaries;

-- Subqueries

-- 29. Check if a value matches any value in a subquery result using IN:

SELECT ProductName FROM Products WHERE CategoryID IN (SELECT CategoryID FROM Categories WHERE CategoryName = 'Electronics');

-- 30. Check for existence using EXISTS:

SELECT OrderID FROM Orders WHERE EXISTS (SELECT \* FROM OrderDetails WHERE OrderDetails.OrderID = Orders.OrderID);

-- 31. Check for non-existence using NOT EXISTS:

SELECT CustomerID FROM Customers WHERE NOT EXISTS (SELECT \* FROM Orders WHERE Orders.CustomerID = Customers.CustomerID);

-- 32. Compare a value to all values in a subquery using ANY / ALL:

SELECT ProductName FROM Products WHERE Price > ALL (SELECT Price FROM Products WHERE CategoryID = 1);

-- 33. Use a subquery in SELECT to retrieve data for a column:

SELECT ProductName, (SELECT AVG(Price) FROM Products) AS AvgPrice FROM Products;

-- String Functions

-- 34. Concatenate strings using CONCAT:

SELECT CONCAT(FirstName, ' ', LastName) AS FullName FROM Employees;

-- 35. Extract a substring using SUBSTRING:

SELECT SUBSTRING(ProductName, 1, 10) AS ShortName FROM Products;

-- 36. Find the position of a substring within a string using CHARINDEX:

SELECT CHARINDEX('SQL', 'Learn SQL Server') AS Position;

-- 37. Calculate the length of a string using LEN:

SELECT LEN(ProductName) AS NameLength FROM Products;

-- 38. Convert a string to uppercase using UPPER:

SELECT UPPER(ProductName) AS UppercaseName FROM Products;

-- 39. Remove leading or trailing spaces using LTRIM / RTRIM:

SELECT LTRIM(RTRIM(CompanyName)) AS TrimmedName FROM Customers;

-- Date and Time Functions

-- 40. Get the current date and time using GETDATE:

SELECT GETDATE() AS CurrentDateTime;

-- 41. Add an interval to a date or time using DATEADD:

SELECT DATEADD(DAY, 7, OrderDate) AS NewOrderDate FROM Orders;

-- 42. Calculate the difference between two dates or times using DATEDIFF:

SELECT DATEDIFF(YEAR, Birthdate, GETDATE()) AS Age FROM Employees;

-- 43. Format a date or time value using FORMAT (SQL Server 2012+):

SELECT FORMAT(OrderDate, 'yyyy-MM-dd HH:mm:ss') AS FormattedDate FROM Orders;

-- NULL Handling

-- 44. Check for NULL values using IS NULL:

SELECT FirstName, LastName FROM Employees WHERE Email IS NULL;

-- 45. Check for non-NULL values using IS NOT NULL:

SELECT ProductName FROM Products WHERE Description IS NOT NULL;

-- 46. Return the first non-NULL value in a list using COALESCE:

SELECT COALESCE(ShippingAddress, BillingAddress, 'N/A') AS Address FROM Orders;

-- 47. Insert Bulk data

-- BULK INSERT TargetTable FROM 'C:\Path\To\Your\File.csv' WITH (FIELDTERMINATOR = ',', ROWTERMINATOR = '\n', FIRSTROW = 2);

-- 48. Update single/multiple cell value

-- UPDATE Employee SET emp\_name = 'vineeta singh' WHERE emp\_id = 28;

-- 49. Check Available database in MS SQL

-- SELECT name FROM sys.databases WHERE database\_id > 4;

-- 50. To check tables name in database

-- SELECT name FROM sys.tables;

-- Identity Key

-- 51. To turn off auto-increment

-- Set identity Tablename on;

-- 52. To turn on auto-increment

-- Set identity Tablename off;

-- Using UNION (removes duplicates)

-- 53. SELECT: Retrieve data from a table.

-- SELECT column1, column2, ... FROM table1 WHERE condition

-- UNION

-- SELECT column1, column2, ... FROM table2 WHERE condition;

-- Using UNION ALL (includes duplicates)

-- 54. SELECT: Retrieve data from a table.

-- SELECT column1, column2, ... FROM table1 WHERE condition

-- UNION ALL

-- SELECT column1, column2, ... FROM table2 WHERE condition;

-- OPERATOR COMMAND

-- 55. OR QUERY

-- SELECT \* FROM staff WHERE Dep\_ID = 'DA' AND stack\_ID = 'DS';

-- 56. NOT Query

-- SELECT \* FROM staff WHERE NOT Dep\_ID = 'DA';

-- DATE Functions IN MS SQL

-- 57. For today's date

-- SELECT GETDATE();

-- 58. To calculate the difference between two dates

-- DATEDIFF(Days/Month/Year, start\_date, end\_date/getdate());

-- 59. To add value in the mentioned date

-- SELECT DATEADD(MONTH, 10, '2023-06-20');

-- 60. To get a part of the date

-- SELECT DATEPART(MONTH, '2023-06-20');

-- ORDER BY ASC/DESC

-- 61. SELECT: Retrieve data from a table.

-- SELECT \* FROM intern ORDER BY identity\_no ASC;

-- SELECT \* FROM intern ORDER BY identity\_no DESC;

-- Left Join

-- 62. SELECT: Retrieve data from intern left join staff on intern.Dep\_ID = staff.HOD\_Name ORDER BY identity\_no ASC;

-- LEFT Outer Join

-- 63. SELECT: Retrieve data from intern left join staff on intern.Dep\_ID = staff.HOD\_Name WHERE Emp\_ID IS NULL ORDER BY identity\_no ASC;

-- Right Join

-- 64. SELECT: Retrieve data from intern right join staff on intern.Dep\_ID = staff.HOD\_Name ORDER BY identity\_no ASC;

-- Right Outer Join

-- 65. SELECT: Retrieve data from intern right join staff on intern.Dep\_ID = staff.HOD\_Name WHERE Emp\_ID IS NULL ORDER BY identity\_no ASC;

-- Self Join

-- 66. SELECT: Retrieve data from a.enrollment\_id, b.student\_name FROM table1 a, table1 b WHERE a.enrollment\_id < b.enrollment\_id;

-- EQUI Join

-- 67. SELECT: Retrieve data from table1.student\_name, table1.enrollment\_id, table2.student\_dob, table2.contact\_no FROM table1 JOIN table2 ON table1.enrollment\_id = table2.enrollment\_id;

-- NON EQUI Join

-- 68. SELECT: Retrieve data from table1 join table2 on table1.enrollment\_id < table2.enrollment\_id;

-- Create database joins

-- 69. CREATE DATABASE: Create a new database.

CREATE DATABASE joins;

-- Create table order\_status

-- 70. CREATE TABLE: Create a new table.

CREATE TABLE order\_status (

order\_ID BIGINT,

customer\_id BIGINT,

order\_date DATE

);

-- Create table customer\_detail

-- 71. CREATE TABLE: Create a new table.

CREATE TABLE customer\_detail (

order\_ID BIGINT,

Customer\_Name NVARCHAR(40),

order\_date DATE

);

-- Bulk insert data into customer\_detail

-- 72. INSERT INTO: Insert data into a table.

BULK INSERT customer\_detail FROM 'D:\DATABASE\Customer Details.csv' WITH (FORMAT = 'CSV', FIELDTERMINATOR = ',', FIRSTROW = 2, ROWTERMINATOR = '\n');

-- Bulk insert data into order\_status

-- 73. INSERT INTO: Insert data into a table.

BULK INSERT order\_status FROM 'D:\DATABASE\Order Details.csv' WITH (FORMAT = 'CSV', FIELDTERMINATOR = ',', FIRSTROW = 2, ROWTERMINATOR = '\n');

-- SELECT data from order\_status

-- 74. SELECT: Retrieve data from a table.

SELECT \* FROM order\_status;

-- SELECT data from customer\_detail

-- 75. SELECT: Retrieve data from a table.

SELECT \* FROM customer\_detail;

-- Alter table customer\_detail

-- 76. ALTER TABLE: Modify an existing table.

ALTER TABLE customer\_detail ALTER COLUMN Order\_ID BIGINT NOT NULL;

-- Add primary key constraint to customer\_detail

-- 77. ADD CONSTRAINT: Add a constraint to a table.

ALTER TABLE customer\_detail ADD CONSTRAINT PK\_Order\_ID PRIMARY KEY (order\_ID);

-- Add foreign key constraint to order\_status

-- 78. ADD CONSTRAINT: Add a constraint to a table.

ALTER TABLE order\_status ADD CONSTRAINT FK\_Order\_ID FOREIGN KEY (order\_ID) REFERENCES customer\_detail(order\_ID) ON UPDATE CASCADE ON DELETE CASCADE;

-- sp\_help command on order\_status

-- 79. SYSTEM PROCEDURE: Execute a system procedure.

sp\_help order\_status;

-- Rename column in customer\_detail

-- 80. SYSTEM PROCEDURE: Execute a system procedure.

sp\_rename 'customer\_detail.order\_ID', 'Customer\_ID';

-- SELECT data from joined tables

-- 81. SELECT: Retrieve data from order\_status INNER JOIN customer\_detail ON order\_status.customer\_id = customer\_detail.customer\_ID.

SELECT order\_status.order\_ID, customer\_detail.Customer\_Name, customer\_detail.customer\_ID FROM order\_status

INNER JOIN customer\_detail ON order\_status.customer\_id = customer\_detail.customer\_ID;