Chapter-6 Data Definition Language (DDL) with MS SQL Server

Data Definition Language (DDL) refers to the subset of SQL commands used to define and manage the structure of database objects such as tables, schemas, views, indexes, and constraints. DDL operations allow you to create, alter, and delete database objects.

In MS SQL Server, DDL statements are used to define and modify the database schema, and they generally do not affect the data stored in the tables directly. However, they can affect how data is organized and accessed. The main DDL commands are CREATE, ALTER, DROP, and TRUNCATE.

---

1. CREATE

The CREATE statement is used to define new database objects, such as databases, tables, views, indexes, and stored procedures.

1.1 CREATE DATABASE

Used to create a new database.

Syntax:

```
CREATE DATABASE database_name;
```

Example:

```
CREATE DATABASE EmployeeDB;
```

This creates a new database called EmployeeDB.

1.2 CREATE TABLE

Used to create a new table in the database, including defining columns and their data types.

Syntax:

```
CREATE TABLE table_name (
   column1 datatype [constraints],
   column2 datatype [constraints],
   ...
```

```
);
```

Example:

```
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   EmployeeName NVARCHAR(100),
   DepartmentID INT,
   Salary DECIMAL(10, 2)
);
```

This creates a table called Employees with columns EmployeeID, EmployeeName, DepartmentID, and Salary.

1.3 CREATE INDEX

Used to create an index on a table to improve query performance.

Syntax:

```
CREATE INDEX index_name
ON table_name (column1, column2, ...);
```

Example:

```
CREATE INDEX idx_salary
ON Employees (Salary);
```

This creates an index on the Salary column of the Employees table to improve search performance on salary values.

1.4 CREATE VIEW

Used to create a view (virtual table) based on the result of a query.

Syntax:

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Example:

```
CREATE VIEW EmployeeDetails AS
SELECT EmployeeID, EmployeeName, DepartmentID
```

```
FROM Employees
WHERE Salary > 50000;
```

This creates a view called EmployeeDetails, which shows details of employees whose salary is greater than 50,000.

---

2. ALTER

The ALTER statement is used to modify the structure of an existing database object. You can use ALTER to add, modify, or drop columns in a table, rename a table or column, and modify constraints.

2.1 ALTER TABLE

Used to modify an existing table structure.

- **Add a Column:**

```
ALTER TABLE table_name
ADD column_name datatype;
```

Example:

```
ALTER TABLE Employees
ADD Email NVARCHAR(100);
```

This adds a new column Email to the Employees table.

- **Modify a Column Data Type:**

```
ALTER TABLE table_name
ALTER COLUMN column_name datatype;
```

Example:

```
ALTER TABLE Employees
ALTER COLUMN Salary DECIMAL(12, 2);
```

This modifies the Salary column to have a larger precision (12 digits with 2 decimal places).

- **Drop a Column:**

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

Example:

```
ALTER TABLE Employees
DROP COLUMN DepartmentID;
```

This drops the DepartmentID column from the Employees table.

- **Rename a Table:**

```
EXEC sp_rename 'old_table_name', 'new_table_name';
```

Example:

```
EXEC sp_rename 'Employees', 'Staff';
```

This renames the Employees table to Staff.

---

3. DROP

The DROP statement is used to delete an existing database object, such as a table, database, view, or index, from the database.

3.1 DROP DATABASE

Used to remove an entire database.

Syntax:

```
DROP DATABASE database_name;
```

Example:

```
DROP DATABASE EmployeeDB;
```

This deletes the EmployeeDB database along with all its tables and data.

3.2 DROP TABLE

Used to delete a table and all of its data.

Syntax:

```
DROP TABLE table_name;
```

Example:

```
DROP TABLE Employees;
```

This deletes the Employees table and all of its data permanently.

3.3 DROP INDEX

Used to delete an index from a table.

Syntax:

```
DROP INDEX index_name ON table_name;
```

Example:

```
DROP INDEX idx_salary ON Employees;
```

This deletes the index idx_salary on the Employees table.

3.4 DROP VIEW

Used to delete a view.

Syntax:

```
DROP VIEW view_name;
```

Example:

```
DROP VIEW EmployeeDetails;
```

This deletes the EmployeeDetails view.

---

4. TRUNCATE

The TRUNCATE statement is used to delete all rows from a table while keeping the table structure intact. Unlike DELETE, it does not log individual row deletions and cannot be rolled back once committed.

Syntax:

```
TRUNCATE TABLE table_name;
```

Example:

```
TRUNCATE TABLE Employees;
```

This removes all rows from the Employees table but keeps the table structure, indexes, and constraints.

---

5. RENAME

In MS SQL Server, you can rename a database object like a table or column using the sp_rename stored procedure.

Syntax:

```
EXEC sp_rename 'old_name', 'new_name';
```

Example:

```
EXEC sp_rename 'Employees.EmployeeName', 'FullName';
```

This renames the EmployeeName column to FullName in the Employees table.

---

6. Constraints in DDL

Constraints are used to define rules for the data in a table. They are defined at the time of table creation or altered later.

Types of Constraints:

- **PRIMARY KEY**: Uniquely identifies each record in a table. Ensures no duplicate values in the specified column.

```
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   EmployeeName NVARCHAR(100)
);
```

- **FOREIGN KEY**: Enforces a link between two tables, ensuring data integrity by restricting the values in the foreign key column to the ones in the referenced column.

```
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   DepartmentID INT,
   FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);
```

- **UNIQUE: Ensures all values in a column are unique.**

```
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   Email NVARCHAR(100) UNIQUE
);
```

- **CHECK**: Ensures that all values in a column satisfy a specified condition.

```
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   Salary DECIMAL(10, 2) CHECK (Salary > 0)
);
```

- **DEFAULT**: Provides a default value for a column if no value is specified.

```
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   JoinDate DATETIME DEFAULT GETDATE()
);
```