# SQL CREATE TABLE Statement

The CREATE TABLE statement is used to create a new table in a database. A table is organized into rows and columns, where each column has a specific **data type** and can have various **constraints**.

## Basic Syntax

The fundamental syntax for creating a table is as follows:

CREATE TABLE table\_name (

column1\_name DATATYPE (size) CONSTRAINT,

column2\_name DATATYPE (size) CONSTRAINT,

column3\_name DATATYPE (size) CONSTRAINT,

...

);

* CREATE TABLE table\_name: Specifies that you are creating a new table and gives it a name.
* column\_name: The name of the column.
* DATATYPE: The type of data that the column can hold (e.g., INT, VARCHAR, DATE).
* (size): (Optional) For some data types like VARCHAR, this specifies the maximum length of the data.
* CONSTRAINT: (Optional) Rules that enforce data integrity on the columns (e.g., PRIMARY KEY, NOT NULL, UNIQUE, DEFAULT).

## Common SQL Data Types

Here are some frequently used SQL data types:

* INT: Stores whole numbers (integers).
* VARCHAR(size): Stores variable-length string data. You must specify the maximum length.
* TEXT: Stores large text strings.
* DATE: Stores dates in 'YYYY-MM-DD' format.
* DATETIME: Stores date and time in 'YYYY-MM-DD HH:MI:SS' format.
* BOOLEAN or TINYINT(1) (depending on the database system): Stores true/false values.
* DECIMAL(P, S) or NUMERIC(P, S): Stores exact decimal numbers. P is the total number of digits (precision), and S is the number of digits after the decimal point (scale).

## Common SQL Constraints

Constraints are rules that you can apply to columns to limit the type of data that can be placed in them, ensuring data accuracy and reliability.

* NOT NULL: Ensures that a column cannot have a NULL value.
* UNIQUE: Ensures that all values in a column are different.
* PRIMARY KEY: A combination of NOT NULL and UNIQUE. Each table can have only one primary key, which uniquely identifies each row.
* FOREIGN KEY: A key used to link two tables together. It refers to the primary key in another table.
* DEFAULT value: Sets a default value for a column when no value is specified.
* CHECK condition: Ensures that all values in a column satisfy a specific condition.

## Example 1: Creating a Simple **Customers** Table

This example creates a basic Customers table with an auto-incrementing CustomerID as the primary key, and columns for FirstName, LastName, and Email.

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY AUTO\_INCREMENT, -- Unique identifier, automatically increments

FirstName VARCHAR(50) NOT NULL, -- Customer's first name, cannot be empty

LastName VARCHAR(50) NOT NULL, -- Customer's last name, cannot be empty

Email VARCHAR(100) UNIQUE -- Customer's email, must be unique

);

**Explanation:**

* CustomerID: An INT that serves as the PRIMARY KEY. AUTO\_INCREMENT (or IDENTITY in some SQL dialects like SQL Server) means the database automatically assigns a unique, sequential number when a new row is inserted.
* FirstName, LastName: VARCHAR(50) to store text up to 50 characters, and NOT NULL means these fields must always have a value.
* Email: VARCHAR(100) for email addresses, and UNIQUE ensures no two customers can have the same email.

## Example 2: Creating a **Products** Table with More Constraints

This example demonstrates creating a Products table with various data types and constraints, including a CHECK constraint for price and a DEFAULT value for StockQuantity.

CREATE TABLE Products (

ProductID INT PRIMARY KEY AUTO\_INCREMENT, -- Unique product identifier

ProductName VARCHAR(255) NOT NULL UNIQUE, -- Name of the product, must be unique

Description TEXT, -- Detailed description of the product

Price DECIMAL(10, 2) NOT NULL CHECK (Price > 0), -- Product price, must be positive

StockQuantity INT DEFAULT 0, -- Number of items in stock, defaults to 0

DateAdded DATE DEFAULT CURRENT\_DATE -- Date product was added, defaults to today

);

**Explanation:**

* ProductID: An INT PRIMARY KEY with AUTO\_INCREMENT.
* ProductName: VARCHAR(255) that is NOT NULL and UNIQUE.
* Description: TEXT for longer product descriptions.
* Price: DECIMAL(10, 2) for precise monetary values (10 total digits, 2 after the decimal). It's NOT NULL and has a CHECK constraint ensuring the price is always greater than 0.
* StockQuantity: INT with a DEFAULT value of 0 if not specified during insertion.
* DateAdded: DATE with a DEFAULT value of CURRENT\_DATE (or GETDATE() in SQL Server, NOW() in PostgreSQL/MySQL) which automatically records the date the product was added.