Power
Learn
Project

SQL Create Table: week 6

Power
Learn
Project

Database Design

You are live with Gerald Macherechedze

SQL Create Table

The SQL Create Table statement is a fundamental command in Structured Query Language (SQL) used to create a new table within a database. Tables are the foundation of a relational database, serving as containers for organized data. With the Create Table statement, you can define the table's structure, specifying the columns, their data types, and any constraints to ensure data integrity. Here's an introduction to the SQL Create Table statement and an example to illustrate its usage.

Introduction to SQL Create Table Statement

When designing a database, one of the initial steps is creating tables to store data in a structured manner. The SQL Create Table statement allows you to define the structure of these tables. Here are the key components of the Create Table statement:

Table Name: You specify the name of the table after the CREATE TABLE keywords. This name must be unique within the database.

Columns: You define the columns that make up the table. For each column, you specify its name, data type (e.g., text, integer, date), and any constraints (e.g., primary key, not null).

Constraints: Constraints are rules that enforce data integrity. Common constraints include primary key, foreign key, unique, and check constraints.

Example:

```
CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DepartmentID INT,

HireDate DATE

);
```

In this example, a table named "Employees" is created with columns for EmployeeID, FirstName, LastName, DepartmentID, and HireDate. The EmployeeID is designated as the primary key.

SQL Create Table Example

Here's a practical example of how the SQL Create Table statement can be used to create a "Products" table:

```
CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(255) NOT NULL,

CategoryID INT,

Price DECIMAL(10, 2),

StockQuantity INT CHECK (StockQuantity >= 0)

);
```

-In this example, the "Products" table is defined with columns for ProductID, ProductName,

CategoryID, Price, and StockQuantity. The ProductID is specified as the primary key, and the ProductName is set as a non-null field. Additionally, the StockQuantity column is constrained to have a value greater than or equal to 0.

-The SQL Create Table statement is a powerful tool for structuring and organizing data within a database. It plays a vital role in defining the schema of a database, allowing you to store and manage information effectively.

SQL ALTER TABLE: Add a Column

The SQL ALTER TABLE statement is a powerful command that allows you to make structural changes to an existing database table.

One common operation is adding a new column to a table. This can be necessary when you need to include additional information or adapt your database schema to changing requirements. Here's how you can use the SQL ALTER TABLE

statement to add a column to an existing table:

-- Syntax for adding a column

ALTER TABLE table_name

ADD column_name data_type;

table_name: Replace this with the name of the table to which you want to add a new column.

column_name: Specify the name of the new column you want to add.

data_type: Define the data type for the new column (e.g., INT, VARCHAR(50), DATE).

Example:

Suppose you have an existing "Employees" table, and you want to add a new column called "Email" to store employees' email addresses:

-- Adding an "Email" column to the "Employees" table

```
ALTER TABLE Employees
ADD Email VARCHAR(255);
```

This SQL statement will modify the "Employees" table by adding a new column named "Email"
with a data type of VARCHAR(255) to store email addresses.

SQL ALTER TABLE: Modify a Column

In addition to adding columns, the ALTER TABLE statement can also be used to modify existing

columns in a table. This is helpful when you need to change the data type, length, or other

attributes of a column. Here's the syntax for modifying a column:

Syntax for modifying a column

ALTER TABLE table_name

MODIFY column_name new_data_type;

table_name: The name of the table containing the column you want to modify.

column_name: The name of the column you wish to modify.

new_data_type: Specify the new data type for the column.

Example:

Let's say you want to change the data type of the "Salary" column in the "Employees" table from INT to DECIMAL(10, 2):

Modifying the "Salary" column in the "Employees" table

```
ALTER TABLE Employees
MODIFY Salary DECIMAL(10, 2);
```

This SQL statement will modify the "Salary" column's data type to DECIMAL(10, 2) in the
"Employees" table.

SQL ALTER TABLE: Drop a Column

Sometimes, you may need to remove a column from a table if it's no longer needed or if the
schema changes. The ALTER TABLE statement can be used to drop or delete a column. Here's
the syntax for dropping a column:

-- Syntax for dropping a column

ALTER TABLE table_name

DROP COLUMN column_name;

table_name: Specify the name of the table from which you want to remove the column.

column_name: Indicate the name of the column you wish to drop.

Example:

Suppose you want to remove the "Phone" column from the "Employees" table:

-- Dropping the "Phone" column from the "Employees" table

```
ALTER TABLE Employees
DROP COLUMN Phone;
```

This SQL statement will remove the "Phone" column from the "Employees" table, effectively
eliminating it from the table's structure.

Introduction to SQL DROP TABLE Statement

In SQL, the DROP TABLE statement is a powerful command used to remove an existing table from a database.

This operation is essential when you no longer need a particular table, or you want to start afresh by deleting all the data and the table's structure.

The DROP TABLE statement should be used with caution since it irreversibly deletes the table and its contents.

Here's an introduction to the SQL DROP TABLE statement and examples illustrating its usage.

SQL DROP TABLE Examples

Example 1: Basic Table Deletion

The simplest use case for the DROP TABLE statement involves removing a single table from a database.

Here's the basic syntax:

```
-- Syntax for dropping a single table
DROP TABLE table_name;
```

table_name: Replace this with the name of the table you want to delete.

Example: Let's say you have a table called "Customers," and you want to delete it from the database:

-- Deleting the "Customers" table

DROP TABLE Customers;

This SQL statement will remove the "Customers" table and all of its associated data from the database.

Example 2: Removing Multiple Tables

You can also use the DROP TABLE statement to remove multiple tables in a single command. This can be useful when you have several tables to delete simultaneously.

Here's how you can do it:

-- Syntax for dropping multiple tables

DROP TABLE table_name1, table_name2, ...;

table_name1, table_name2, etc.: List the names of the tables you want to delete, separated by commas.

Example:

Suppose you have three tables, "Products," "Orders," and "OrderItems," that you no longer need in your database. You can remove them all at once using the DROP TABLE statement:

-- Deleting multiple tables in one command

DROP TABLE Products, Orders, OrderItems;

This SQL statement will delete all three tables and their associated data from the database.