The DROP TABLE statement is used to remove an existing table including all its triggers, constraints, permissions.

Syntax

Following is the Syntax of the DROP TABLE statement.

ij> DROP TABLE table\_name;

Example

Suppose you have a table named Student in the database. The following SQL statement deletes a table named Student.

ij> DROP TABLE Student;

0 rows inserted/updated/deleted

Since we have removed the table if we try to describe it, we will get an error as follows

ij> DESCRIBE Student;

IJ ERROR: No table exists with the name STUDENT

Drop Table using JDBC program

This section teaches you how to drop a table in Apache Derby database using JDBC application.

If you want to request the Derby network server using network client, make sure that the server is up and running. The class name for the Network client driver is org.apache.derby.jdbc.ClientDriver and the URL is jdbc:derby://localhost:1527**/DATABASE\_NAME;**create=true;user=**USER\_NAME**;passw ord=**PASSWORD**"

Follow the steps given below to drop a table in Apache Derby

Step 1: Register the driver

To communicate with the database, first of all, you need to register the driver. The **forName()** method of the class **Class** accepts a String value representing a class name loads it in to the memory, which automatically registers it. Register the driver using this method.

Step 2: Get the connection

In general, the first step we do to communicate to the database is to connect with it. The **Connection** class represents the physical connection with a database server. You can create a connection object by invoking the **getConnection()** method of the **DriverManager** class. Create a connection using this method.

Step 3: Create a statement object

You need to create a **Statement** or **PreparedStatement** or, **CallableStatement** objects to send SQL statements to the database. You can create these using the methods **createStatement(), prepareStatement() and, prepareCall()** respectively. Create either of these objects using the appropriate method.

Step 4: Execute the query

After creating a statement, you need to execute it. The **Statement** class provides various methods to execute a query like the **execute()** method to execute a statement that returns more than one result set. The **executeUpdate()** method execute queries like INSERT, UPDATE, DELETE. The **executeQuery()** method to results that returns data etc. Use either of these methods and execute the statement created previously.

Example

Following JDBC example demonstrates how to drop a table in Apache Derby using JDBC program. Here, we are connecting to a database named sampleDB (will create if it does not exist) using the embedded driver.

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.Statement;

public class DropTable {

public static void main(String args[]) throws Exception {

//Registering the driver

Class.forName("org.apache.derby.jdbc.EmbeddedDriver");

//Getting the Connection object

String URL = "jdbc:derby:sampleDB;create=true";

Connection conn = DriverManager.getConnection(URL);

//Creating the Statement object

Statement stmt = conn.createStatement();

//Executing the query

String query = "DROP TABLE Employees";

stmt.execute(query);

System.out.println("Table dropped");

}

}

Output

On executing the above program, you will get the following output −

Table dropped