SQL - Temporary Tables

What are Temporary Tables?

Temporary tables are pretty much what their name describes: they are the tables which are created in a database to store temporary data. We can perform SQL operations similar to the operations on permanent tables like CREATE, UPDATE, DELETE, INSERT, JOIN, etc. But these tables will be automatically deleted once the current client session is terminated. In addition to that, they can also be explicitly deleted if the users decide to drop them manually.

Various RDBMS, like MySQL, support temporary tables starting from version 3.23 onwards. If you are using an older version of MySQL than 3.23, you can't use temporary tables, but you can use heap tables.

As stated earlier, temporary tables will only last as long as the client session is alive. If you run the code in a PHP script, the temporary table will be destroyed automatically when the script finishes executing. If you are connected to the MySQL database server through a MySQL client program, then the temporary table will exist until you close the client connection or manually destroy the table.

Creating Temporary Tables in MySQL

```
CREATE TEMPORARY TABLE table_name(
  column1 datatype,
  column2 datatype,
  column3 datatype,
  .....
  columnN datatype,
  PRIMARY KEY( one or more columns )
);
```

```
CREATE TEMPORARY TABLE CUSTOMERS(
  ID   INT            NOT NULL,
  NAME VARCHAR (20)     NOT NULL,
  AGE  INT            NOT NULL,
  ADDRESS  CHAR (25) ,
  SALARY   DECIMAL (18, 2),
  PRIMARY KEY (ID)
);

INSERT INTO CUSTOMERS VALUES
(1, 'Ramesh', 32, 'Ahmedabad', 2000.00 ),
(2, 'Khilan', 25, 'Delhi', 1500.00 ),
(3, 'Kaushik', 23, 'Kota', 2000.00 );
```

NOTE-:

When you issue a SHOW TABLES command, then your temporary table will not be displayed in the list of tables. To verify whether the temporary table is created you need to retrieve its data using the SELECT statement. Since all the temporary tables will be removed when the current session is closed, if you log out of the MySQL session and then issue a SELECT command, you will not find temporary table in the database.

Dropping Temporary Tables in MySQL

Though all the temporary tables are deleted by MySQL when your database connection gets terminated, still, if you want to delete them manually, then you can do so by issuing a DROP TEMPORARY TABLE command.

```
DROP TEMPORARY TABLE table_name;
```

```
DROP TEMPORARY TABLE CUSTOMERS;
```

Verification

```
SELECT * FROM CUSTOMERS;
```

Temporary Tables in SQL Server

The temporary table created in MySQL is visible only within the current session. But, in Microsoft SQL Server you can create two types of temporary tables.

- **Local Temporary Tables:** A Local Temporary Table is accessible only in the session that has created it. It is automatically deleted when the connection that has created it gets closed. If the Temporary Table is created inside the stored procedure, it get dropped automatically upon the completion of stored procedure execution.

- स्थानीय अस्थायी तालिकाएँ: एक स्थानीय अस्थायी तालिका केवल उस सत्र में ही पहुँच योग्य होती है जिसने इसे बनाया है। जब इसे बनाने वाला कनेक्शन बंद हो जाता है, तो यह स्वचालित रूप से हटा दी जाती है। यदि अस्थायी तालिका संग्रहीत प्रक्रिया के अंदर बनाई गई है, तो संग्रहीत प्रक्रिया निष्पादन के पूरा होने पर यह स्वचालित रूप से हटा दी जाती है।
- **Global Temporary Tables:** Global Temporary Tables are visible to all connections and Dropped when the last connection referencing the table is closed.
- वैश्विक अस्थायी तालिकाएँ: वैश्विक अस्थायी तालिकाएँ सभी कनेक्शनों को दिखाई देती हैं और तालिका को संदर्भित करने वाला अंतिम कनेक्शन बंद होने पर हटा दी जाती हैं।

Syntax of the Local Temporary Tables

```
CREATE TABLE #table_name(

   column1 datatype,

   column2 datatype,

   column3 datatype,

   .....

   columnN datatype,

   PRIMARY KEY( one or more columns )

);
```

```
CREATE TABLE #CUSTOMERS(
  ID   INT              NOT NULL,
  NAME VARCHAR (20)     NOT NULL,
  AGE  INT              NOT NULL,
  ADDRESS  CHAR (25) ,
  SALARY   DECIMAL (18, 2),
  PRIMARY KEY (ID)
);
```

Syntax of the Global Temporary Tables

```
CREATE TABLE ##table_name(
  column1 datatype,
  column2 datatype,
  column3 datatype,
  .....
  columnN datatype,
  PRIMARY KEY( one or more columns )
);
```

```
CREATE TABLE ##Buyers(
  ID   INT              NOT NULL,
  NAME VARCHAR (20)     NOT NULL,
  AGE  INT              NOT NULL,
  ADDRESS  CHAR (25) ,
  SALARY   DECIMAL (18, 2),
  PRIMARY KEY (ID)
);
```

Dropping Temporary Tables in SQL Server

If you want to drop a temporary table in SQL Server manually, you need to execute the DROP TABLE statement by placing **#** before the local temporary table name and **##** before the global temporary table name.

Example

Following query removes the Local temporary table **Customers** created in the previous example.

```
DROP TABLE #Customers;
```

Whereas, following query removes the global temporary table **Buyers**.

```
DROP TABLE ##Buyers;
```

Basic Example: Creating a Simple Temporary Table

```
CREATE TEMPORARY TABLE TempEmployees (
   EmployeeID INT,
   EmployeeName VARCHAR(100),
   Department VARCHAR(50)
);
```

-- Inserting data into the temporary table

```
INSERT INTO TempEmployees (EmployeeID, EmployeeName, Department)
VALUES (1, 'John Doe', 'HR'),
       (2, 'Jane Smith', 'IT'),
       (3, 'Robert Brown', 'Finance');
```

-- Selecting data from the temporary table

```
SELECT * FROM TempEmployees;
```

-- Dropping the temporary table

```
DROP TEMPORARY TABLE IF EXISTS TempEmployees;
```

Intermediate Example: Using Temporary Table with a SELECT Statement

```
-- Creating a temporary table using the result of a SELECT statement
CREATE TEMPORARY TABLE TempSales AS
SELECT
    CustomerID,
    SUM(Amount) AS TotalAmount
FROM Sales
GROUP BY CustomerID;

-- Selecting data from the temporary table
SELECT * FROM TempSales;

-- Dropping the temporary table
DROP TEMPORARY TABLE IF EXISTS TempSales;
```

Advanced Example: Using Temporary Tables with Indexes and Joins

```
-- Creating a temporary table with indexes
CREATE TEMPORARY TABLE TempProductSales (
    ProductID INT,
    TotalSales DECIMAL(10, 2),
    INDEX idx_product (ProductID)
);
```

```
-- Inserting data into the temporary table
INSERT INTO TempProductSales (ProductID, TotalSales)
SELECT
   ProductID,
   SUM(SalesAmount)
FROM Sales
GROUP BY ProductID;

-- Creating another temporary table for joining
CREATE TEMPORARY TABLE TempProductDetails (
   ProductID INT,
   ProductName VARCHAR(100),
   Price DECIMAL(10, 2)
);

-- Inserting data into the second temporary table
INSERT INTO TempProductDetails (ProductID, ProductName, Price)
VALUES (101, 'Product A', 50.00),
       (102, 'Product B', 30.00),
       (103, 'Product C', 70.00);
```

```
-- Joining the temporary tables to get detailed sales report
SELECT
   pd.ProductName,
   ps.TotalSales,
   pd.Price
FROM
   TempProductSales ps
JOIN
   TempProductDetails pd
ON
   ps.ProductID = pd.ProductID;

-- Dropping the temporary tables
DROP TEMPORARY TABLE IF EXISTS TempProductSales;
DROP TEMPORARY TABLE IF EXISTS TempProductDetails;
```

Key Points to Note:

1. **Temporary Table Scope**: The temporary tables exist only within the session or transaction in which they are created.
2. **Indexes**: You can create indexes on temporary tables to improve query performance, especially for large datasets.
3. **Joining Temporary Tables**: Temporary tables can be used in complex queries, including joins, to help break down and simplify large queries.