MySQL Derived Tables

The Derived tables are pretty much what their name describes: they are the tables that are derived from another MySQL database table (main table). In other words, the derived table is a virtual result-set obtained from a SELECT statement given as a subquery to another SELECT statement of the main table.

This table is similar to a temporary table. But unlike temporary tables, you need not create a derived table separately; the records in it are retrieved from the main table using a subquery. Therefore, similar to the actual database table, a derived table can also be displayed as a result-set of computations, aggregate functions, etc.

व्युत्पन्न तालिकाएँ लगभग वैसी ही होती हैं जैसा कि उनके नाम से पता चलता है: वे ऐसी तालिकाएँ होती हैं जो किसी अन्य MySQL डेटाबेस तालिका (मुख्य तालिका) से व्युत्पन्न होती हैं। दूसरे शब्दों में, व्युत्पन्न तालिका एक वर्चुअल परिणाम-सेट होती है जो मुख्य तालिका के किसी अन्य SELECT कथन को सबक्वेरी के रूप में दिए गए SELECT कथन से प्राप्त होती है।

यह तालिका एक अस्थायी तालिका के समान होती है। लेकिन अस्थायी तालिकाओं के विपरीत, आपको अलग से एक व्युत्पन्न तालिका बनाने की आवश्यकता नहीं होती है; इसमें मौजूद रिकॉर्ड एक सबक्वेरी का उपयोग करके मुख्य तालिका से प्राप्त किए जाते हैं। इसलिए, वास्तविक डेटाबेस तालिका के समान, एक व्युत्पन्न तालिका को भी संगणनाओं, समग्र कार्यों आदि के परिणाम-सेट के रूप में प्रदर्शित किया जा सकता है।

Syntax

Following is the basic syntax to display a derived table in MySQL −

```
SELECT column_name(s) FROM (subquery) AS derived_table_name;
```

```
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)
);
```

```
INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY) VALUES
(1, 'Ramesh', 32, 'Ahmedabad', 2000.00 ),
(2, 'Khilan', 25, 'Delhi', 1500.00 ),
(3, 'Kaushik', 23, 'Kota', 2000.00 ),
(4, 'Chaitali', 25, 'Mumbai', 6500.00 ),
(5, 'Hardik', 27, 'Bhopal', 8500.00 ),
(6, 'Komal', 22, 'Hyderabad', 4500.00 ),
(7, 'Muffy', 24, 'Indore', 10000.00 );
```

```
SELECT * FROM CUSTOMERS;
```

Now, we are retrieving a **derived table** from this CUSTOMERS table using the following query –

```
SELECT ID, NAME, SALARY FROM (SELECT * FROM CUSTOMERS) AS DERIVED_CUSTOMERS;
```

Using WHERE Clause

We can also use the WHERE clause to filter records (or rows) from the derived table. Following is the syntax for it −

```
SELECT column_name(s) FROM (subquery) AS derived_table_name WHERE [condition];
```

Example

In the following query, we are retrieving a derived table from the CUSTOMERS table created initially. We are doing this by filtering rows from it using the **WHERE** clause –

```
SELECT ID, NAME, SALARY FROM (SELECT * FROM CUSTOMERS) AS DERIVED_CUSTOMERS
WHERE DERIVED_CUSTOMERS.SALARY > 5000.00;
```

Aliasing a Column in Derived Table

In derived tables, not only the table name, but we can also alias a column name while displaying the contents. Following is the syntax −

```
SELECT column_name(s) AS alias_name(s) FROM (subquery) AS derived_table_name;
```

Example

In the example below, we are displaying the derived table from the CUSTOMERS table with the aliased columns using the following query –

```
SELECT ID AS DERIVED_ID, NAME AS DERIVED_NAME, SALARY AS DERIVED_SALARY
FROM (SELECT * FROM CUSTOMERS) AS DERIVED_CUSTOMERS;
```

Displaying Aggregate Functions as Derived Tables

We can also show the result of an aggregate function or calculations performed on the main table's records as a derived table.

Following is the syntax to display aggregate functions as a derived table –

```
SELECT function_name() FROM (subquery) AS derived_table_name;
```

Example

In the following query, we are using the aggregate SUM() function to calculate the total salary from the CUSTOMERS table –

```
SELECT SUM(SALARY) FROM (SELECT SALARY FROM CUSTOMERS) AS DERIVED_CUSTOMERS;
```

Example

In the following query, we use the aggregate AVG() function to calculate the average salary of customers from the CUSTOMERS table.

```
SELECT AVG(DERIVED_SUM) AS AVERAGE_SALARY
FROM (SELECT SUM(SALARY) AS DERIVED_SUM FROM CUSTOMERS) AS DERIVED_CUSTOMERS;
```

Deriving Table Using a Client Program

Syntax

Following are the syntaxes of this operation in various programming languages

```
sql="SELECT col_1, col_2 FROM table_name WHERE col_name IN (SELECT col_name FROM table_name)";

cursorObj.execute(sql);
```

```
import mysql.connector

#establishing the connection

connection = mysql.connector.connect(

   host='localhost',

   user='root',

   password='password',

   database='tut'

)

table_name = 'tutorials_tbl'

# Main query with a derived table (subquery)

main_query = """

SELECT d.tutorial_id, d.tutorial_title, d.tutorial_author, d.submission_date

FROM (

   SELECT tutorial_id, tutorial_title, tutorial_author, submission_date

   FROM tutorials_tbl

   WHERE submission_date >= '2023-01-01'

) AS d

WHERE d.tutorial_author LIKE '%Paul%'

"""

cursorObj = connection.cursor()

cursorObj.execute(main_query)

result = cursorObj.fetchall()

print("Derived Table Result:")

for row in result:

   print(f"| {row[0]:<11} | {row[1]:<15} | {row[2]:<15} | {row[3]:<15} |")

cursorObj.close()

connection.close()
```