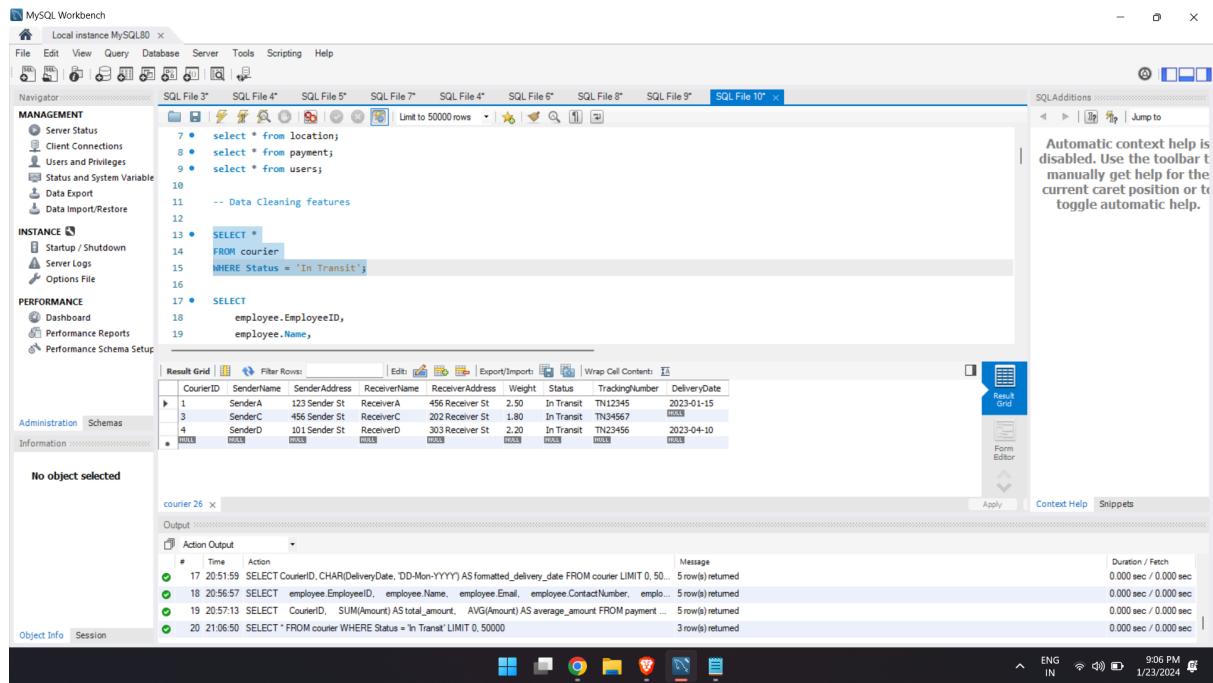


Name : Parth Nandedkar
Date : 23 Jan 2024
Topics : Advanced SQL Concepts
Batch : Data Engineering Batch-1

Data cleaning & Transformation queries ,Ranking function,stored procedure

Data Cleaning And Transformation :

Filtering Function :



The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```
7 •    select * from location;
8 •    select * from payments;
9 •    select * from users;
10
11 -- Data Cleaning features
12
13 •    SELECT *
14      FROM courier
15     WHERE Status = 'In Transit';
16
17 •    SELECT
18       employee.EmployeeID,
19       employee.Name,
```

The results grid displays the following data for the query:

| CourierID | SenderName | SenderAddress | ReceiverName | ReceiverAddress | Weight | Status | TrackingNumber | DeliveryDate |
|-----------|------------|---------------|--------------|-----------------|--------|------------|----------------|--------------|
| 1 | SenderA | 123 Sender St | ReceiverA | 456 Receiver St | 2.50 | In Transit | TN12345 | 2023-01-15 |
| 3 | SenderC | 456 Sender St | ReceiverC | 202 Receiver St | 1.80 | In Transit | TN04567 | NULL |
| 4 | SenderD | 101 Sender St | ReceiverD | 303 Receiver St | 2.20 | In Transit | TN23456 | 2023-04-10 |

The output pane shows the execution history:

| # | Time | Action | Message | Duration / Fetch |
|----|----------|--|-------------------|-----------------------|
| 17 | 20:51:59 | SELECT CourierID, CHAR(DeliveryDate, DD-Mon-YYYY) AS formatted_delivery_date FROM courier LIMIT 0, 50... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 18 | 20:56:57 | SELECT employee.EmployeeID, employee.Name, employee.Email, employee.ContactNumber, emplo... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 19 | 20:57:13 | SELECT CourierID, SUM(Amount) AS total_amount, AVG(Amount) AS average_amount FROM payment ... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 20 | 21:06:50 | SELECT * FROM courier WHERE Status = 'In Transit' LIMIT 0, 50000 | 3 row(s) returned | 0.000 sec / 0.000 sec |

Retrieving Data from different Table :

The screenshot shows the MySQL Workbench interface with a query editor window. The query retrieves data from the 'employee' and 'location' tables:

```

16
17 • SELECT
18     employee.EmployeeID,
19     employee.Name,
20     employee.Email,
21     employee.ContactNumber,
22     employee.Role,
23     employee.Salary,
24     location.LocationName,
25     location.Address AS LocationAddress
26 FROM employee
27 JOIN location ON employee.EmployeeID = location.LocationID;
28

```

The results grid displays the following data:

| EmployeeID | Name | Email | ContactNumber | Role | Salary | LocationName | LocationAddress |
|------------|-----------|---------------------|---------------|------------------|----------|--------------|-----------------|
| 1 | EmployeeA | employeeA@email.com | 111223333 | Courier Handler | 30000.00 | LocationA | 789 Main Street |
| 2 | EmployeeB | employeeB@email.com | 222334444 | Delivery Driver | 35000.00 | LocationB | 456 Oak Avenue |
| 3 | EmployeeC | employeeC@email.com | 333445555 | Manager | 50000.00 | LocationC | 101 Pine Lane |
| 4 | EmployeeD | employeeD@email.com | 444556666 | Customer Service | 28000.00 | LocationD | 202 Elm Street |
| 5 | EmployeeE | employeeE@email.com | 555667777 | Courier Handler | 32000.00 | LocationE | 303 Cedar Road |

The output pane shows the execution log:

| # | Time | Action | Message | Duration / Fetch |
|----|----------|---|-------------------|-----------------------|
| 18 | 20:56:57 | SELECT employee.EmployeeID, employee.Name, employee.Email, employee.ContactNumber, emplo... 5 row(s) returned | | 0.000 sec / 0.000 sec |
| 19 | 20:57:13 | SELECT CounterID, SUM(Amount) AS total_amount, AVG(Amount) AS average_amount FROM payment ... 3 row(s) returned | | 0.000 sec / 0.000 sec |
| 20 | 21:06:50 | SELECT * FROM courier WHERE Status = 'In Transit' LIMIT 0, 50000 | 3 row(s) returned | 0.000 sec / 0.000 sec |
| 21 | 21:07:22 | SELECT employee.EmployeeID, employee.Name, employee.Email, employee.ContactNumber, emplo... 5 row(s) returned | | 0.000 sec / 0.000 sec |

Aggregate Function :

The screenshot shows the MySQL Workbench interface with a query editor window. The query uses aggregate functions to calculate totals and averages:

```

25
26     FROM employee
27 JOIN location ON employee.EmployeeID = location.LocationID;
28
29 • SELECT
30     CourierID,
31     SUM(Amount) AS total_amount,
32     AVG(Amount) AS average_amount
33     FROM payment
34     GROUP BY CourierID;
35
36 -- Data Formating
37

```

The results grid displays the following data:

| CourierID | total_amount | average_amount |
|-----------|--------------|----------------|
| 1 | 35.00 | 35.000000 |
| 2 | 55.00 | 55.000000 |
| 3 | 60.00 | 60.000000 |
| 4 | 35.00 | 35.000000 |
| 5 | 40.00 | 40.000000 |

The output pane shows the execution log:

| # | Time | Action | Message | Duration / Fetch |
|----|----------|---|-------------------|-----------------------|
| 19 | 20:57:13 | SELECT CourierID, SUM(Amount) AS total_amount, AVG(Amount) AS average_amount FROM payment ... 3 row(s) returned | | 0.000 sec / 0.000 sec |
| 20 | 21:06:50 | SELECT * FROM courier WHERE Status = 'In Transit' LIMIT 0, 50000 | 3 row(s) returned | 0.000 sec / 0.000 sec |
| 21 | 21:07:22 | SELECT employee.EmployeeID, employee.Name, employee.Email, employee.ContactNumber, emplo... 5 row(s) returned | | 0.000 sec / 0.000 sec |
| 22 | 21:08:51 | SELECT CourierID, SUM(Amount) AS total_amount, AVG(Amount) AS average_amount FROM payment ... 5 row(s) returned | | 0.000 sec / 0.000 sec |

Data formatting :

MySQL Workbench

Local instance MySQL80 X

File Edit View Query Database Server Tools Scripting Help

Navigator

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variable
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

PERFORMANCE

- Dashboard
- Performance Reports
- Performance Schema Setup

No object selected

SQL File 3* SQL File 4* SQL File 5* SQL File 7* SQL File 4* SQL File 6* SQL File 8* SQL File 9* SQL File 10*

```

34     GROUP BY CourierID;
35
36     -- Data Formatting
37
38 •   SELECT CourierID, CHAR(DeliveryDate, 'DD-Mon-YYYY') AS formatted_delivery_date
39   FROM couriers
40
41
42     -- Data Cleaning
43
44 •   SELECT *
45   FROM employees
46   WHERE salary > 50000;

```

Result Grid | Filter Rows: Export: Wrap Cell Contents: □

| CourierID | formatted_delivery_date |
|-----------|-------------------------|
| 1 | 2023-08-22 |
| 2 | 2023-08-22 |
| 3 | 2023-08-22 |
| 4 | 2023-08-22 |
| 5 | 2023-08-22 |

Result Grid Form Editor

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result 29 X

Action Output

| # | Time | Action | Message | Duration / Fetch |
|----|----------|--|-------------------|-----------------------|
| 20 | 21:06:50 | SELECT * FROM courier WHERE Status = 'In Transit' LIMIT 0, 50000 | 3 row(s) returned | 0.000 sec / 0.000 sec |
| 21 | 21:07:22 | SELECT employee.EmployeeID, employee.Name, employee.Email, employee.ContactNumber, emplo... 5 row(s) returned | | 0.000 sec / 0.000 sec |
| 22 | 21:08:51 | SELECT CourierID, SUM(Amount) AS total_amount, AVG(Amount) AS average_amount FROM payment ... 5 row(s) returned | | 0.000 sec / 0.000 sec |
| 23 | 21:11:51 | SELECT CourierID, CHAR(DeliveryDate, 'DD-Mon-YYYY') AS formatted_delivery_date FROM courier LIMIT 0, 50... 5 row(s) returned | | 0.015 sec / 0.000 sec |

Object Info Session

ENG IN 9:11 PM 1/23/2024

Data Cleaning :

MySQL Workbench

Local instance MySQL80 X

File Edit View Query Database Server Tools Scripting Help

Navigator

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variable
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

PERFORMANCE

- Dashboard
- Performance Reports
- Performance Schema Setup

No object selected

SQL File 3* SQL File 4* SQL File 5* SQL File 7* SQL File 4* SQL File 6* SQL File 8* SQL File 9* SQL File 10*

```

40
41
42     -- Data Cleaning
43
44 •   SELECT *
45   FROM employees
46   WHERE salary > 50000;
47
48 •   UPDATE location
49   SET Address = COALESCE(Address, 'Unknown')
50   WHERE Address IS NULL;
51
52
53
54
55
56     -- Procedures
57
58   DELIMITER //
59
60 •   CREATE PROCEDURE GetUserByID(IN p UserID INT)
61   BEGIN
62       SELECT * ...

```

Output

Action Output

| # | Time | Action | Message | Duration / Fetch |
|----|----------|--|--|-----------------------|
| 21 | 21:07:22 | SELECT employee.EmployeeID, employee.Name, employee.Email, employee.ContactNumber, emplo... 5 row(s) returned | | 0.000 sec / 0.000 sec |
| 22 | 21:08:51 | SELECT CourierID, SUM(Amount) AS total_amount, AVG(Amount) AS average_amount FROM payment ... 5 row(s) returned | | 0.000 sec / 0.000 sec |
| 23 | 21:11:51 | SELECT CourierID, CHAR(DeliveryDate, 'DD-Mon-YYYY') AS formatted_delivery_date FROM courier LIMIT 0, 50... 5 row(s) returned | | 0.015 sec / 0.000 sec |
| 24 | 21:12:29 | UPDATE location SET Address = COALESCE(Address, 'Unknown') WHERE Address IS NULL. | 0 row(s) affected Rows matched: 0 Changed: 0 Warnings: 0 | 0.000 sec |

Object Info Session

ENG IN 9:12 PM 1/23/2024

Procedures :

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the database structure with nodes like MANAGEMENT, INSTANCE, and PERFORMANCE.
- SQL Editor:** Displays the SQL code for creating two procedures:

```
DELIMITER //  
CREATE PROCEDURE GetUserByID(IN p.UserID INT)  
BEGIN  
    SELECT *  
    FROM users  
    WHERE UserID = p.UserID;  
END //  
  
DELIMITER ;  
  
DELIMITER //  
  
CREATE PROCEDURE CalculateTotalPayment(IN p.CourierID INT)  
BEGIN  
    DECLARE totalAmount DECIMAL(10, 2);  
  
    SELECT SUM(Amount) INTO totalAmount  
    FROM payment  
    WHERE CourierID = p.CourierID;
```
- Output:** Shows the execution log with four entries:

| # | Time | Action | Message | Duration / Fetch |
|---|----------|--|--|-----------------------|
| 1 | 21:07:22 | SELECT employee.EmployeeID, employee.Name, employee.Email, employee.ContactNumber, emplo... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 2 | 21:08:51 | SELECT CourierID, SUM(Amount) AS total_amount, AVG(Amount) AS average_amount FROM payment ... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 3 | 23:11:51 | SELECT CourierID, CHAR(DeliveryDate, 'DD-Mon-YYYY') AS formatted_delivery_date FROM courier LIMIT 0, 50... | 5 row(s) returned | 0.015 sec / 0.000 sec |
| 4 | 24:12:29 | UPDATE location SET Address = COALESCE(Address, 'Unknown') WHERE Address IS NULL | 0 row(s) affected Rows matched: 0 Changed: 0 Warnings: 0 | 0.000 sec |

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the database structure with nodes like MANAGEMENT, INSTANCE, and PERFORMANCE.
- SQL Editor:** Displays the SQL code for calling three procedures:

```
-- Calling Procedures  
CALL GetUserByID(2);  
CALL CalculateTotalPayment(1);  
  
CALL UpdateCourierStatus(3, 'In Transit');
```
- Result Grid:** Shows the results of a SELECT query on the Employee table:

| User ID | Name | Email | Password | Contact Number | Address |
|---------|------------|----------------------|------------|----------------|----------------|
| 2 | Jane Smith | jane.smith@email.com | securepass | 9876543210 | 456 Oak Avenue |
- Output:** Shows the execution log with five entries:

| # | Time | Action | Message | Duration / Fetch |
|---|----------|--|--|-----------------------|
| 1 | 23:11:51 | SELECT CourierID, CHAR(DeliveryDate, 'DD-Mon-YYYY') AS formatted_delivery_date FROM courier LIMIT 0, 50... | 5 row(s) returned | 0.015 sec / 0.000 sec |
| 2 | 24:12:29 | UPDATE location SET Address = COALESCE(Address, 'Unknown') WHERE Address IS NULL | 0 row(s) affected Rows matched: 0 Changed: 0 Warnings: 0 | 0.000 sec |
| 3 | 25:11:54 | CALL GetUserByID(2) | 1 row(s) returned | 0.000 sec / 0.000 sec |
| 4 | 26:13:54 | CALL CalculateTotalPayment(1) | 1 row(s) returned | 0.000 sec / 0.000 sec |

Rank :

RANK() :

The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

```
CALL UpdateCourierStatus(3, 'In Transit');

-- Rank Function

SELECT EmployeeID, Name, Salary, RANK() OVER (ORDER BY Salary DESC) AS SalaryRank
FROM employee;
```

The result grid displays the following data:

| EmployeeID | Name | Salary | SalaryRank |
|------------|-----------|----------|------------|
| 3 | EmployeeC | 50000.00 | 1 |
| 2 | EmployeeB | 35000.00 | 2 |
| 5 | EmployeeE | 32000.00 | 3 |
| 1 | EmployeeA | 30000.00 | 4 |
| 4 | EmployeeD | 28000.00 | 5 |

The session output shows the following log entries:

| Action | Time | Message | Duration / Fetch |
|---|-------------|--|-----------------------|
| UPDATE location SET Address = COALESCE(Address, 'Unknown') WHERE Address IS NULL. | 24 21:12:29 | 0 row(s) affected Rows matched: 0 Changed: 0 Warnings: 0 | 0.000 sec |
| CALL GetUserBy(D2) | 25 21:13:54 | 1 row(s) returned | 0.000 sec / 0.000 sec |
| CALL CalculateTotalPayment() | 26 21:13:54 | 1 row(s) returned | 0.000 sec / 0.000 sec |
| SELECT EmployeeID, Name, Salary, RANK() OVER (ORDER BY Salary DESC) AS SalaryRank FROM employee | 27 21:14:58 | 5 row(s) returned | 0.000 sec / 0.000 sec |

DENSERANK() :

The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

```
RANK() OVER (ORDER BY Salary DESC) AS SalaryRank
FROM employee;
SELECT EmployeeID, Name, Salary, DENSE_RANK() OVER (ORDER BY Salary DESC) AS DenseSalaryRank
FROM employee;
```

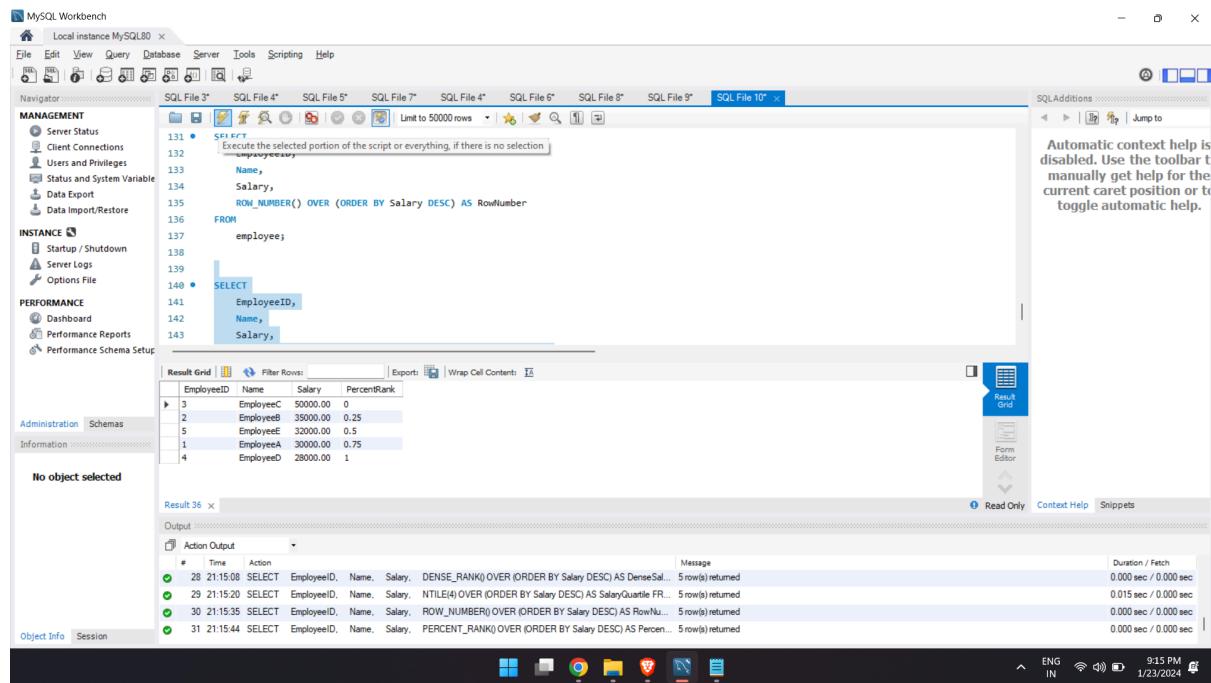
The result grid displays the following data:

| EmployeeID | Name | Salary | DenseSalaryRank |
|------------|-----------|----------|-----------------|
| 3 | EmployeeC | 50000.00 | 1 |
| 2 | EmployeeB | 35000.00 | 2 |
| 5 | EmployeeE | 32000.00 | 3 |
| 1 | EmployeeA | 30000.00 | 4 |
| 4 | EmployeeD | 28000.00 | 5 |

The session output shows the following log entries:

| Action | Time | Message | Duration / Fetch |
|---|-------------|-------------------|-----------------------|
| CALL GetUserBy(D2) | 25 21:13:54 | 1 row(s) returned | 0.000 sec / 0.000 sec |
| CALL CalculateTotalPayment() | 26 21:13:54 | 1 row(s) returned | 0.000 sec / 0.000 sec |
| SELECT EmployeeID, Name, Salary, RANK() OVER (ORDER BY Salary DESC) AS SalaryRank FROM employee | 27 21:14:58 | 5 row(s) returned | 0.000 sec / 0.000 sec |
| SELECT EmployeeID, Name, Salary, DENSE_RANK() OVER (ORDER BY Salary DESC) AS DenseSal... | 28 21:15:08 | 5 row(s) returned | 0.000 sec / 0.000 sec |

NTILE() :



The screenshot shows the MySQL Workbench interface with a query editor window. The code is:

```
131 • SELECT
132   Execute the selected portion of the script or everything, if there is no selection.
133
134   Name,
135   Salary,
136   ROW_NUMBER() OVER (ORDER BY Salary DESC) AS RowNumber
137
138   FROM
139
140 • SELECT
141   EmployeeID,
142   Name,
143   Salary,
```

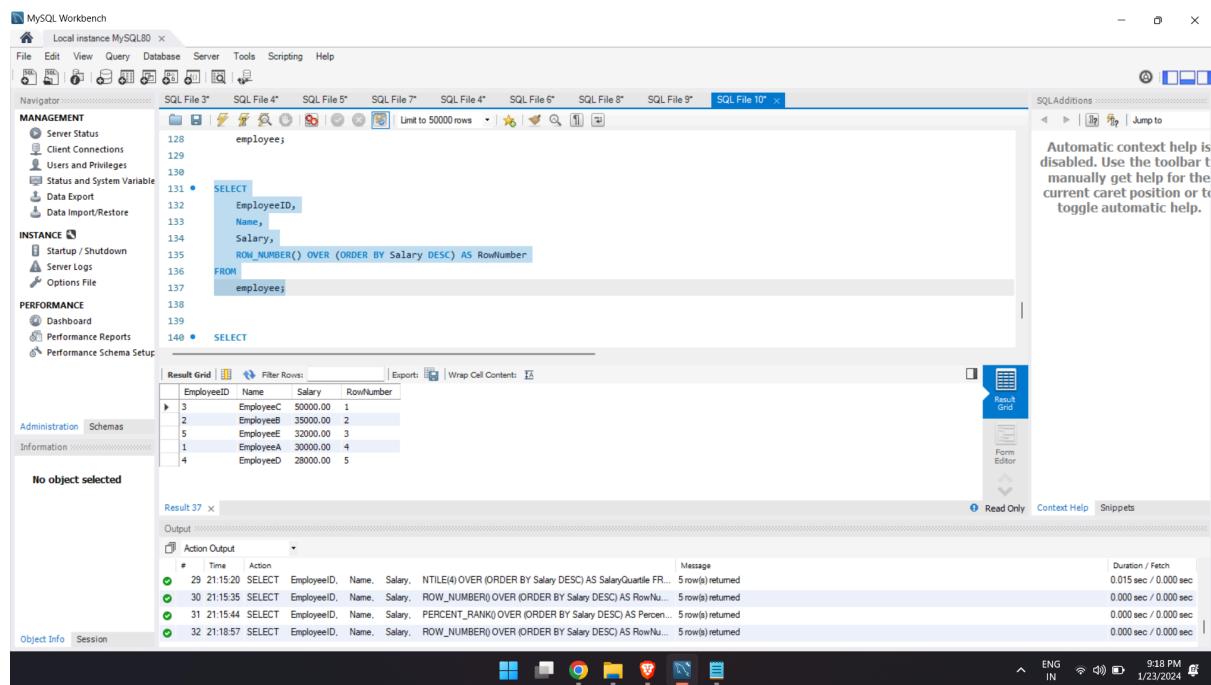
The result grid shows the following data:

| EmployeeID | Name | Salary | PercentRank |
|------------|-----------|----------|-------------|
| 3 | EmployeeC | 50000.00 | 0 |
| 2 | EmployeeB | 35000.00 | 0.25 |
| 5 | EmployeeE | 32000.00 | 0.5 |
| 1 | EmployeeA | 30000.00 | 0.75 |
| 4 | EmployeeD | 28000.00 | 1 |

The output pane shows the execution log:

| # | Time | Action | Message | Duration / Fetch |
|----|----------|---|-------------------|-----------------------|
| 28 | 21:15:08 | SELECT EmployeeID, Name, Salary, DENSE_RANK() OVER (ORDER BY Salary DESC) AS DenseSal... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 29 | 21:15:20 | SELECT EmployeeID, Name, Salary, NTILE(4) OVER (ORDER BY Salary DESC) AS SalaryQuartile FR... | 5 row(s) returned | 0.015 sec / 0.000 sec |
| 30 | 21:15:35 | SELECT EmployeeID, Name, Salary, ROW_NUMBER() OVER (ORDER BY Salary DESC) AS RowNu... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 31 | 21:15:44 | SELECT EmployeeID, Name, Salary, PERCENT_RANK() OVER (ORDER BY Salary DESC) AS Percen... | 5 row(s) returned | 0.000 sec / 0.000 sec |

ROW_NUMBER() :



The screenshot shows the MySQL Workbench interface with a query editor window. The code is:

```
128
129
130
131 • SELECT
132   EmployeeID,
133   Name,
134   Salary,
135   ROW_NUMBER() OVER (ORDER BY Salary DESC) AS RowNumber
136
137   FROM
138
139
140 • SELECT
```

The result grid shows the following data:

| EmployeeID | Name | Salary | RowNumber |
|------------|-----------|----------|-----------|
| 3 | EmployeeC | 50000.00 | 1 |
| 2 | EmployeeB | 35000.00 | 2 |
| 5 | EmployeeE | 32000.00 | 3 |
| 1 | EmployeeA | 30000.00 | 4 |
| 4 | EmployeeD | 28000.00 | 5 |

The output pane shows the execution log:

| # | Time | Action | Message | Duration / Fetch |
|----|----------|---|-------------------|-----------------------|
| 29 | 21:15:20 | SELECT EmployeeID, Name, Salary, NTILE(4) OVER (ORDER BY Salary DESC) AS SalaryQuartile FR... | 5 row(s) returned | 0.015 sec / 0.000 sec |
| 30 | 21:15:35 | SELECT EmployeeID, Name, Salary, ROW_NUMBER() OVER (ORDER BY Salary DESC) AS RowNu... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 31 | 21:15:44 | SELECT EmployeeID, Name, Salary, PERCENT_RANK() OVER (ORDER BY Salary DESC) AS Percen... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 32 | 21:18:57 | SELECT EmployeeID, Name, Salary, ROW_NUMBER() OVER (ORDER BY Salary DESC) AS RowNu... | 5 row(s) returned | 0.000 sec / 0.000 sec |

PERCENT_RANK() :

The screenshot shows the MySQL Workbench interface with a query editor and results grid.

Query Editor:

```
148 • SELECT
149   EmployeeID,
150     Name,
151     Salary,
152       PERCENT_RANK() OVER (ORDER BY Salary DESC) AS PercentRank
153
154 FROM
155   employees
156
157
158
159
160
161
162
```

Result Grid:

| EmployeeID | Name | Salary | PercentRank |
|------------|-----------|----------|-------------|
| 3 | EmployeeC | 50000.00 | 0 |
| 2 | EmployeeB | 35000.00 | 0.25 |
| 5 | EmployeeE | 32000.00 | 0.5 |
| 1 | EmployeeA | 30000.00 | 0.75 |
| 4 | EmployeeD | 28000.00 | 1 |

Action Output:

| # | Time | Action | Message | Duration / Fetch |
|----|----------|--|-------------------|-----------------------|
| 30 | 21:15:35 | SELECT EmployeeID, Name, Salary, ROW_NUMBER() OVER (ORDER BY Salary DESC) AS RowNu... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 31 | 21:15:44 | SELECT EmployeeID, Name, Salary, PERCENT_RANK() OVER (ORDER BY Salary DESC) AS Percen... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 32 | 21:18:57 | SELECT EmployeeID, Name, Salary, ROW_NUMBER() OVER (ORDER BY Salary DESC) AS RowNu... | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 33 | 21:20:09 | SELECT EmployeeID, Name, Salary, PERCENT_RANK() OVER (ORDER BY Salary DESC) AS Percen... | 5 row(s) returned | 0.000 sec / 0.000 sec |