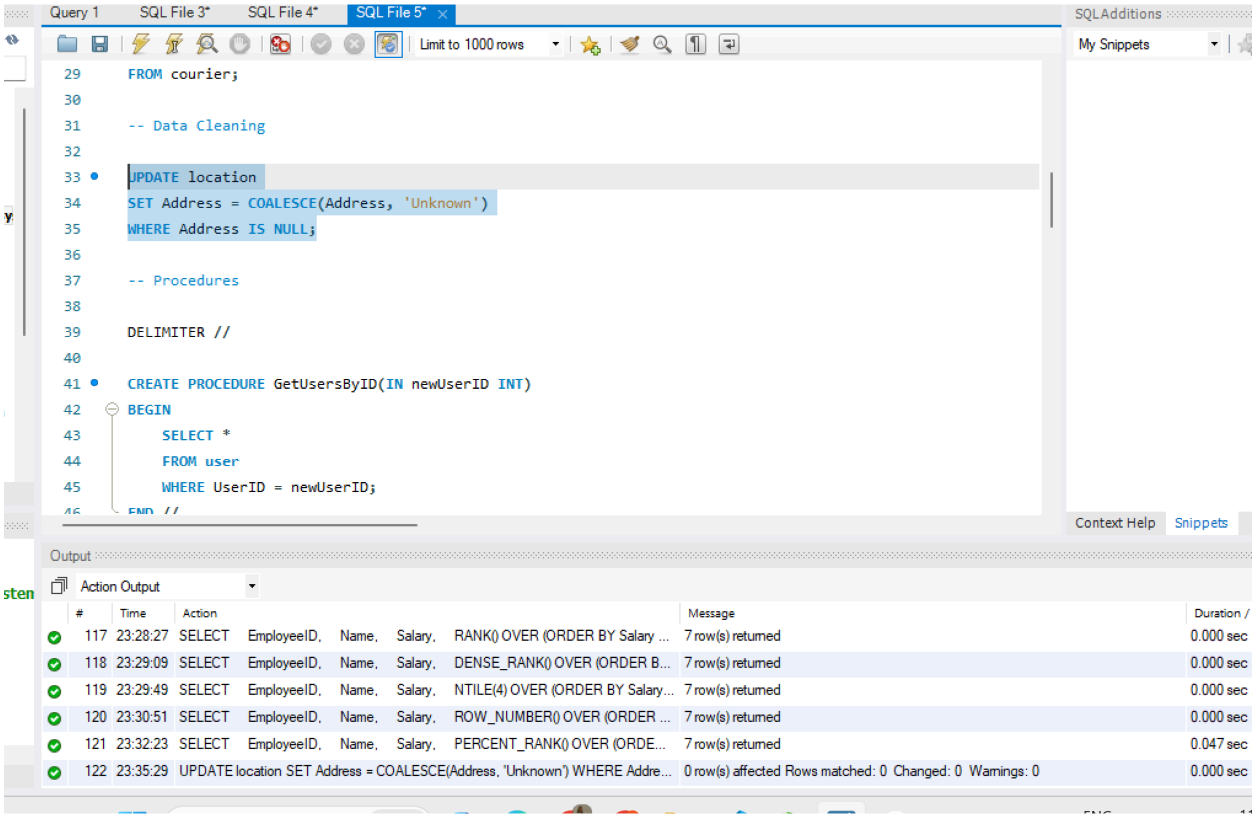
Asmita Porwal
Batch-1
Day-5
23/1/2023
Data engineering

Data Cleaning And Transformation :

Check for missing value

Isnull and coalesce

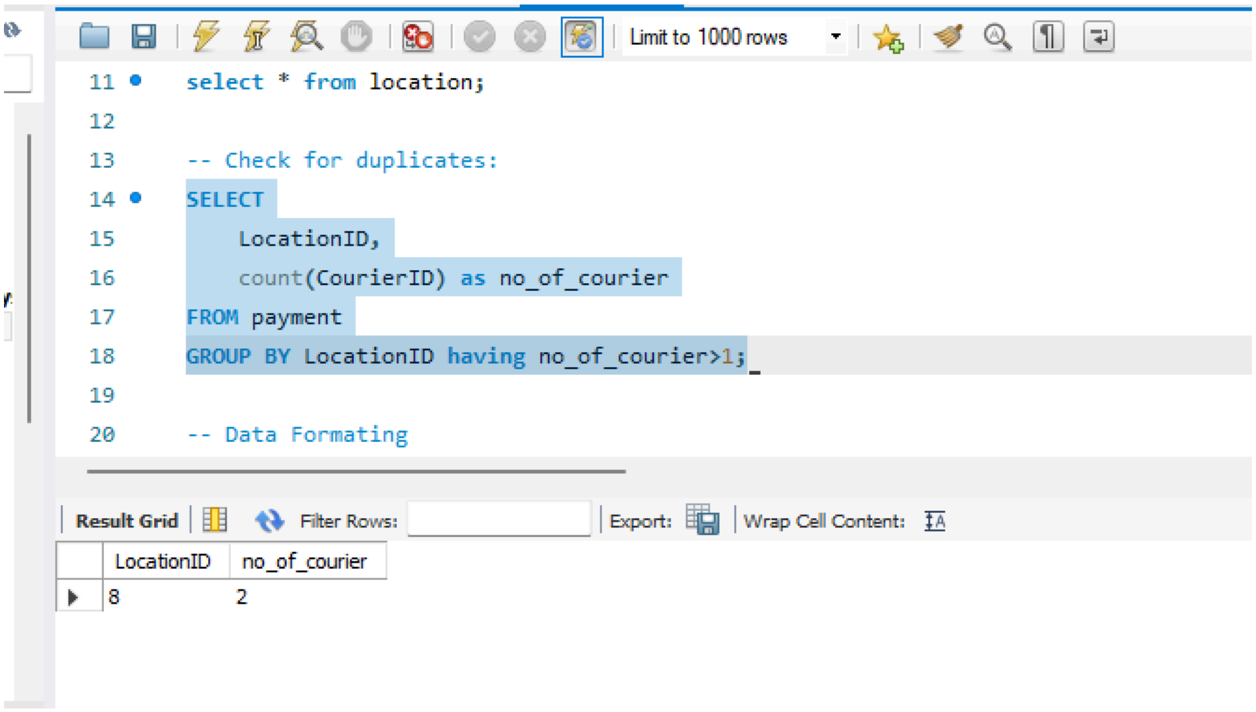


```
FROM courier;

-- Data Cleaning

UPDATE location
SET Address = COALESCE(Address, 'Unknown')
WHERE Address IS NULL;

-- Procedures

DELIMITER //

CREATE PROCEDURE GetUsersByID(IN newUserID INT)
BEGIN
    SELECT *
    FROM user
    WHERE UserID = newUserID;
END //
```

```
    #    Time      Action                                                        Message                                                          Duration /
    117  23:28:27  SELECT  EmployeeID,  Name,  Salary,  RANK() OVER (ORDER BY Salary ...     7 row(s) returned                                                0.000 sec
    118  23:29:09  SELECT  EmployeeID,  Name,  Salary,  DENSE_RANK() OVER (ORDER B...      7 row(s) returned                                                0.000 sec
    119  23:29:49  SELECT  EmployeeID,  Name,  Salary,  NTILE(4) OVER (ORDER BY Salary...  7 row(s) returned                                                0.000 sec
    120  23:30:51  SELECT  EmployeeID,  Name,  Salary,  ROW_NUMBER() OVER (ORDER ...       7 row(s) returned                                                0.000 sec
    121  23:32:23  SELECT  EmployeeID,  Name,  Salary,  PERCENT_RANK() OVER (ORDE...       7 row(s) returned                                                0.047 sec
    122  23:35:29  UPDATE location SET Address = COALESCE(Address, 'Unknown') WHERE Addre...  0 row(s) affected Rows matched: 0  Changed: 0  Warnings: 0   0.000 sec
```

Check for duplicates:

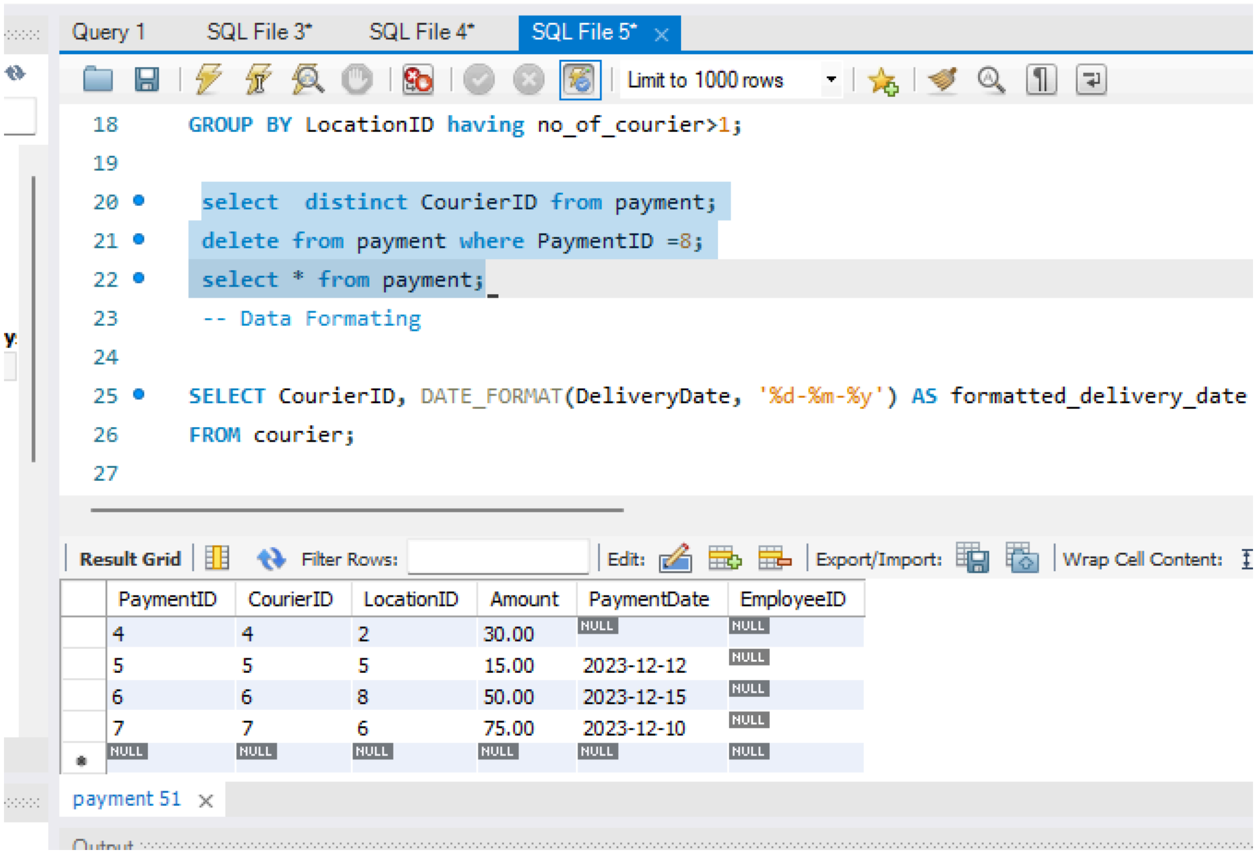
Group by and having: find duplicate records



```sql
select * from location;

-- Check for duplicates:
SELECT
    LocationID,
    count(CourierID) as no_of_courier
FROM payment
GROUP BY LocationID having no_of_courier>1;

-- Data Formating
```

LocationID	no_of_courier
8	2

Distinct and delete from:



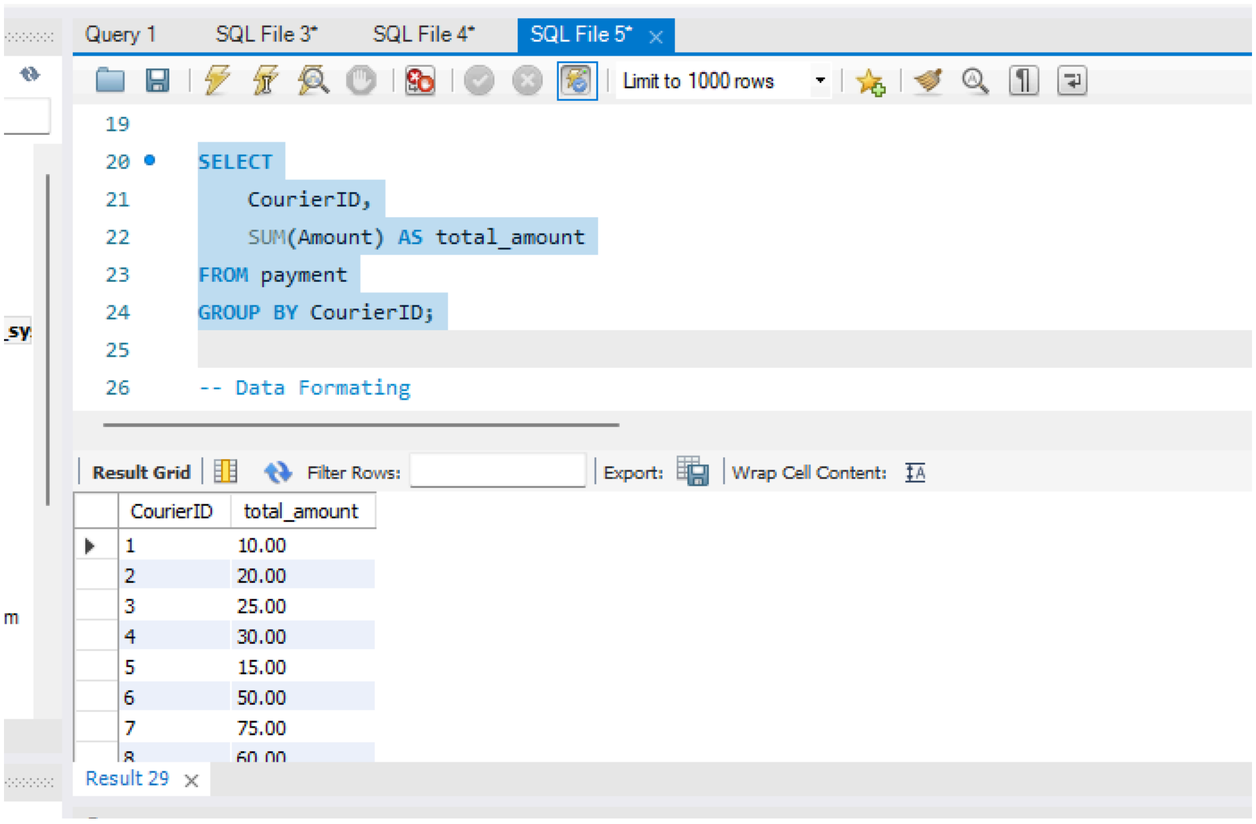
```sql
GROUP BY LocationID having no_of_courier>1;

 select  distinct CourierID from payment;
 delete from payment where PaymentID =8;
 select * from payment;
 -- Data Formating

SELECT CourierID, DATE_FORMAT(DeliveryDate, '%d-%m-%y') AS formatted_delivery_date
FROM courier;

```

PaymentID	CourierID	LocationID	Amount	PaymentDate	EmployeeID
4	4	2	30.00	NULL	NULL
5	5	5	15.00	2023-12-12	NULL
6	6	8	50.00	2023-12-15	NULL
7	7	6	75.00	2023-12-10	NULL
NULL	NULL	NULL	NULL	NULL	NULL

Standarizing and transforming data

For consistency and accurate analysis

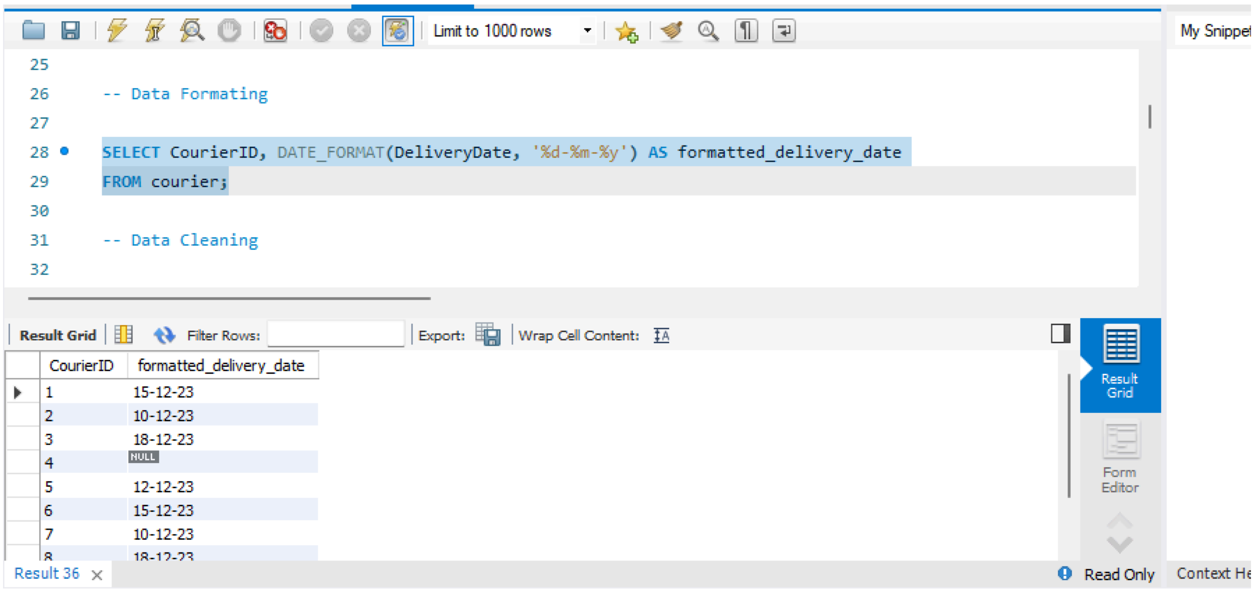
Aggregate Function :



```
SELECT
    CourierID,
    SUM(Amount) AS total_amount
FROM payment
GROUP BY CourierID;

-- Data Formating
```

CourierID	total_amount
1	10.00
2	20.00
3	25.00
4	30.00
5	15.00
6	50.00
7	75.00
8	60.00

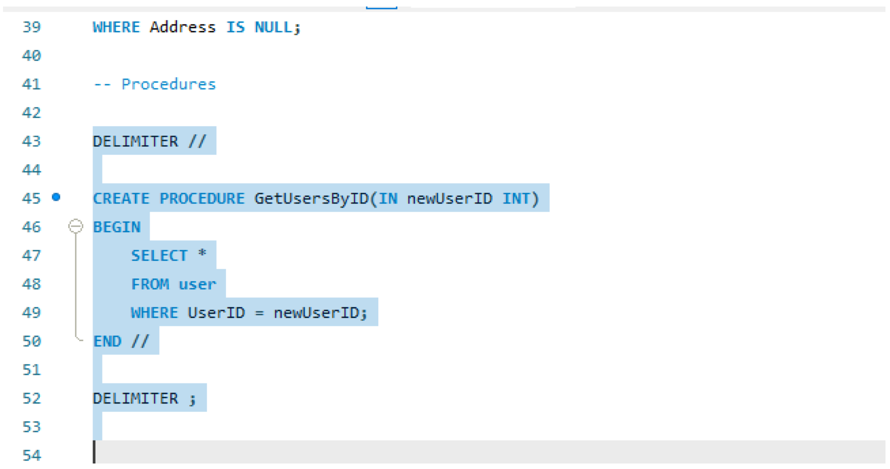
Data formatting :



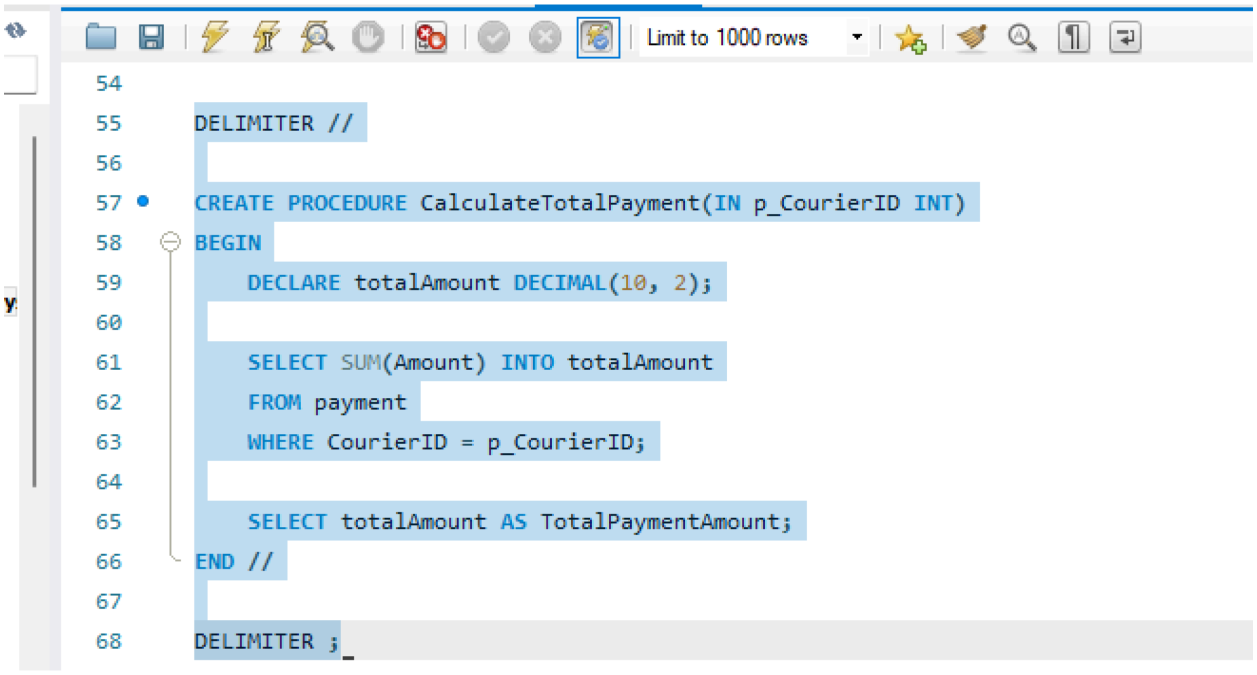
```
-- Data Formating

SELECT CourierID, DATE_FORMAT(DeliveryDate, '%d-%m-%y') AS formatted_delivery_date
FROM courier;

-- Data Cleaning

```

```
CourierID  formatted_delivery_date
1          15-12-23
2          10-12-23
3          18-12-23
4          NULL
5          12-12-23
6          15-12-23
7          10-12-23
8          18-12-23
```

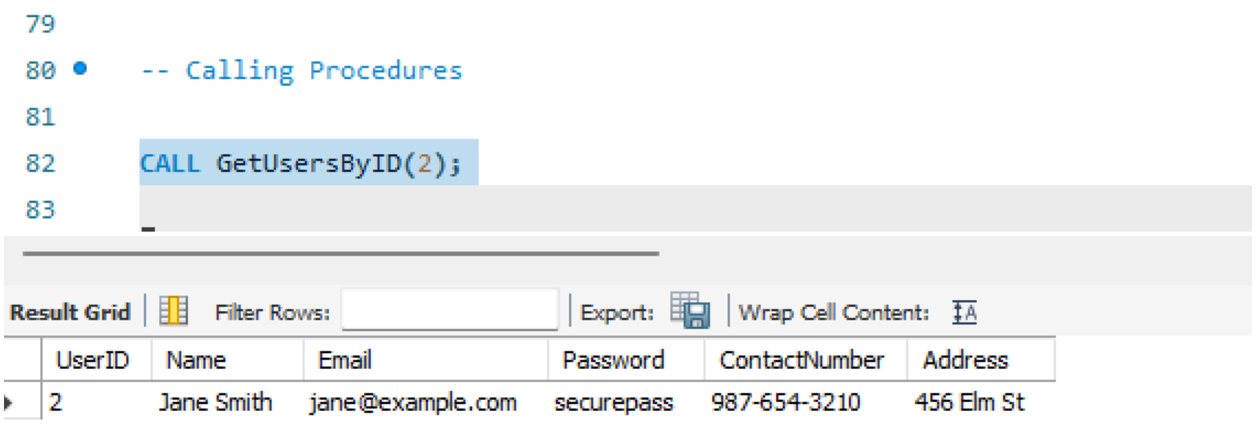
Procedures :



```
WHERE Address IS NULL;

-- Procedures

DELIMITER //

CREATE PROCEDURE GetUsersByID(IN newUserID INT)
BEGIN
    SELECT *
    FROM user
    WHERE UserID = newUserID;
END //

DELIMITER ;

```

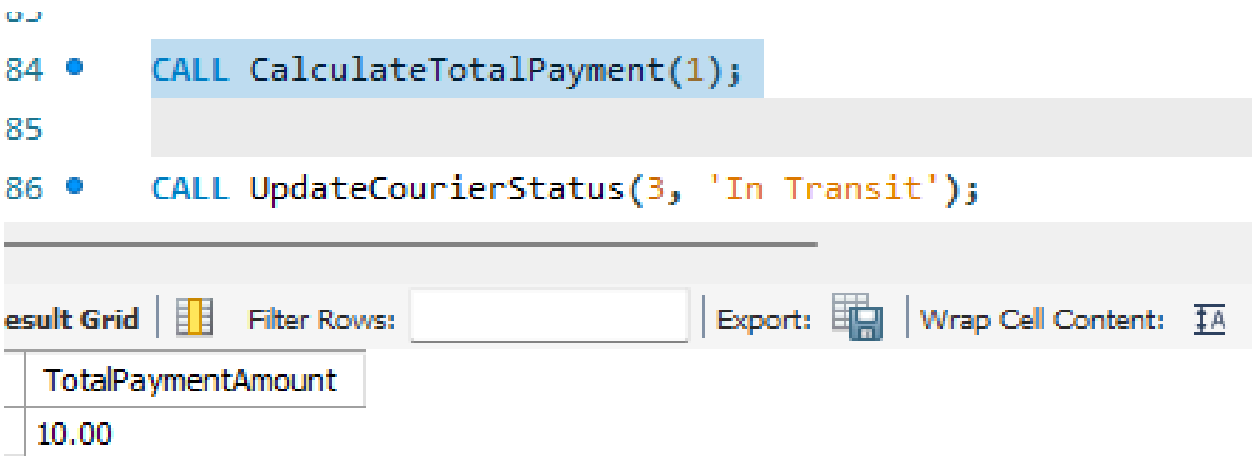


```
DELIMITER //

CREATE PROCEDURE CalculateTotalPayment(IN p_CourierID INT)
BEGIN
    DECLARE totalAmount DECIMAL(10, 2);

    SELECT SUM(Amount) INTO totalAmount
    FROM payment
    WHERE CourierID = p_CourierID;

    SELECT totalAmount AS TotalPaymentAmount;
END //

DELIMITER ;
```



```
-- Calling Procedures

CALL GetUsersByID(2);
```

UserID	Name	Email	Password	ContactNumber	Address
2	Jane Smith	jane@example.com	securepass	987-654-3210	456 Elm St

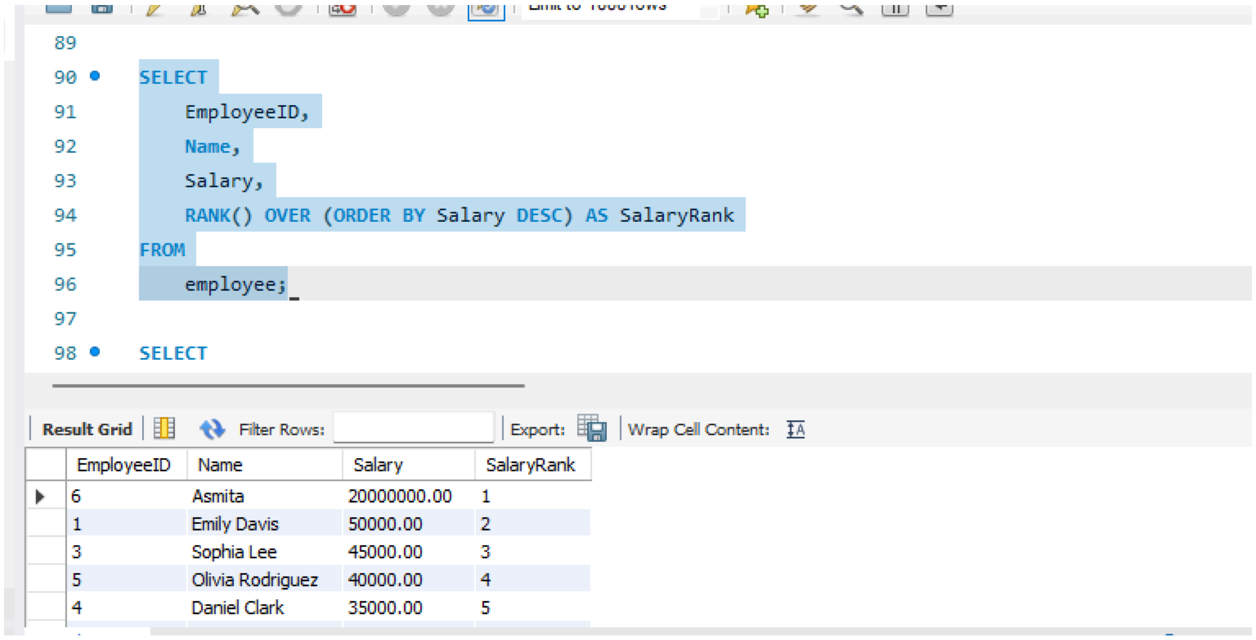


```
CALL CalculateTotalPayment(1);

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

```
TotalPaymentAmount
10.00
```

Rank Function:

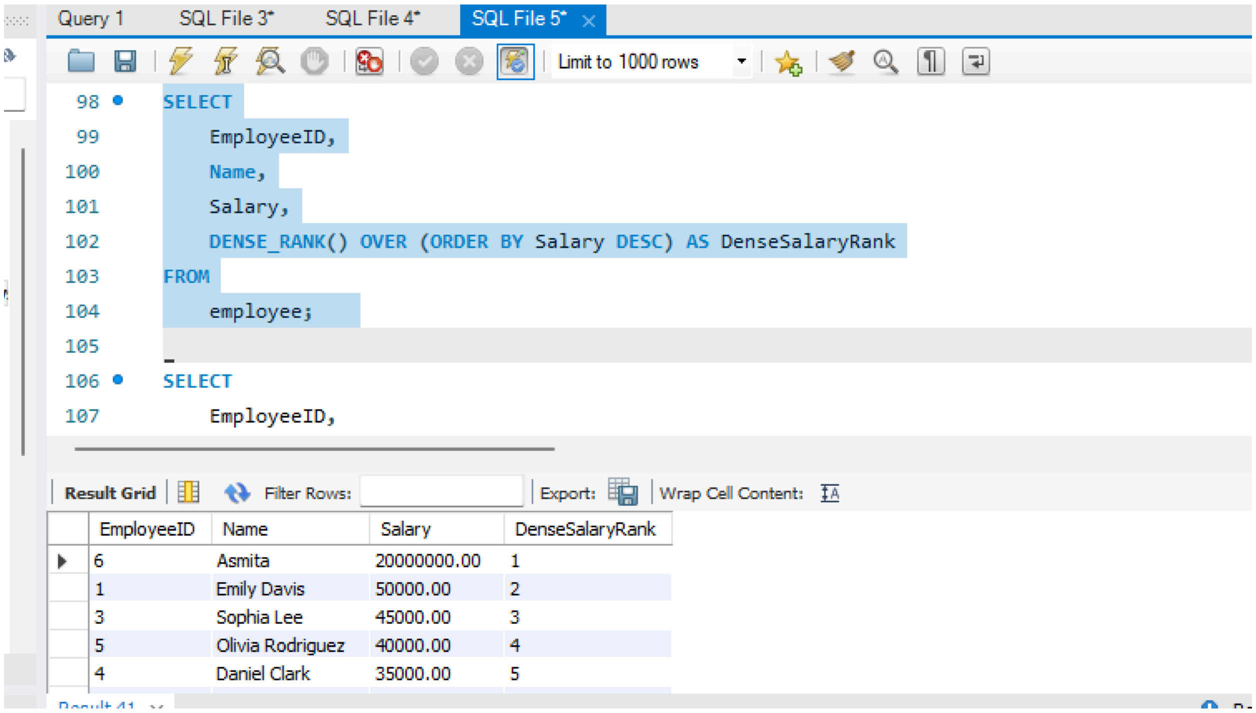
Rank()



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

SELECT
```

EmployeeID	Name	Salary	SalaryRank
6	Asmita	20000000.00	1
1	Emily Davis	50000.00	2
3	Sophia Lee	45000.00	3
5	Olivia Rodriguez	40000.00	4
4	Daniel Clark	35000.00	5

Dense_rank()



Query 1 | SQL File 3* | SQL File 4* | SQL File 5*

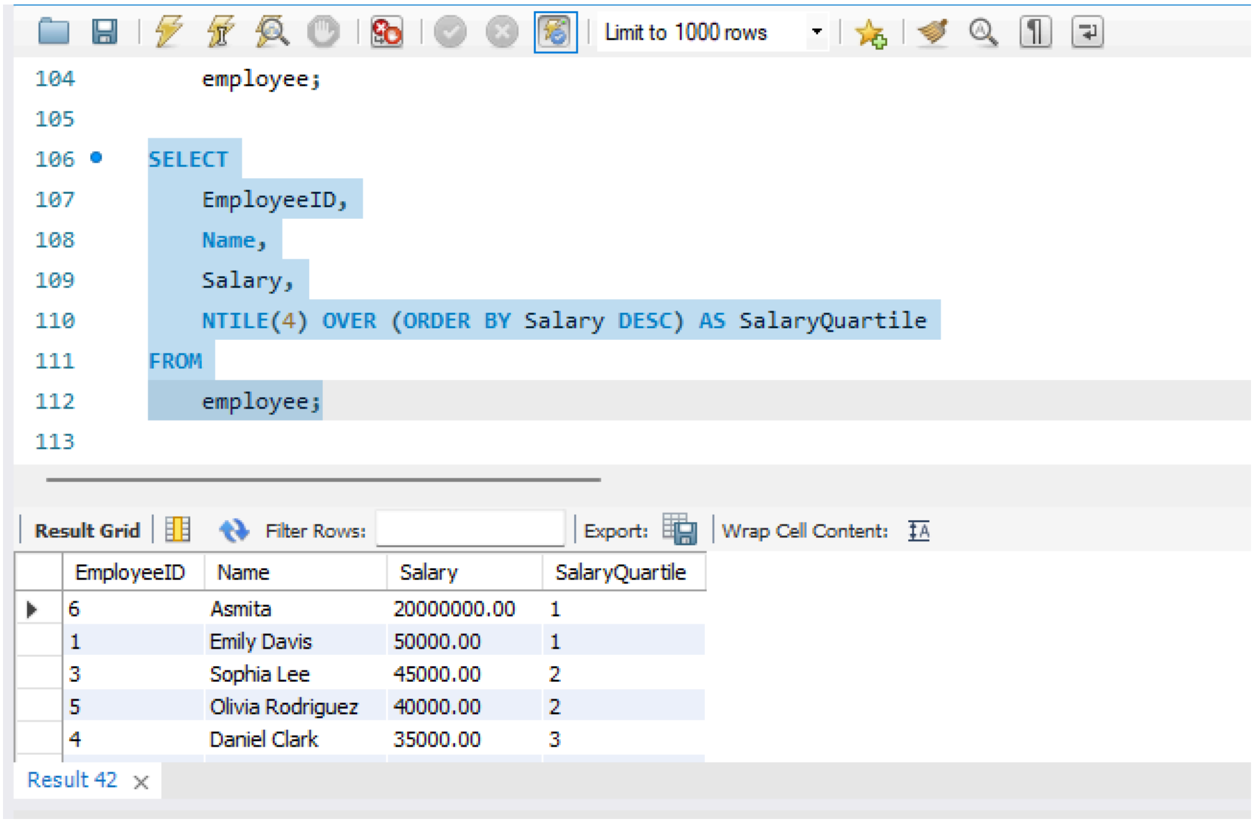
Limit to 1000 rows

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

SELECT
    EmployeeID,
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EmployeeID	Name	Salary	DenseSalaryRank
6	Asmita	20000000.00	1
1	Emily Davis	50000.00	2
3	Sophia Lee	45000.00	3
5	Olivia Rodriguez	40000.00	4
4	Daniel Clark	35000.00	5

NTILE()



Limit to 1000 rows

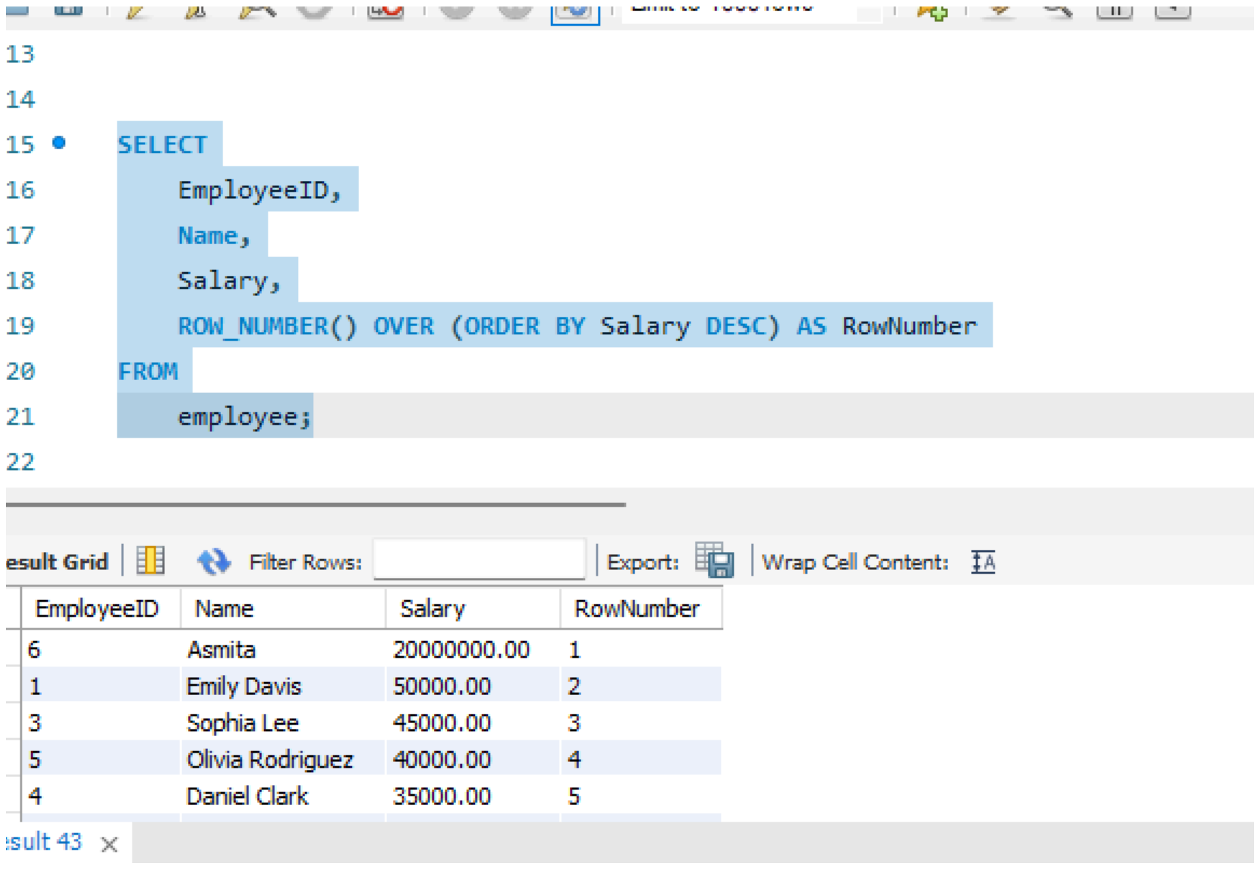
```sql
    employee;

SELECT
    EmployeeID,
    Name,
    Salary,
    NTILE(4) OVER (ORDER BY Salary DESC) AS SalaryQuartile
FROM
    employee;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EmployeeID	Name	Salary	SalaryQuartile
6	Asmita	20000000.00	1
1	Emily Davis	50000.00	1
3	Sophia Lee	45000.00	2
5	Olivia Rodriguez	40000.00	2
4	Daniel Clark	35000.00	3

Result 42

ROW_NUMBER()



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

EmployeeID	Name	Salary	RowNumber
6	Asmita	20000000.00	1
1	Emily Davis	50000.00	2
3	Sophia Lee	45000.00	3
5	Olivia Rodriguez	40000.00	4
4	Daniel Clark	35000.00	5