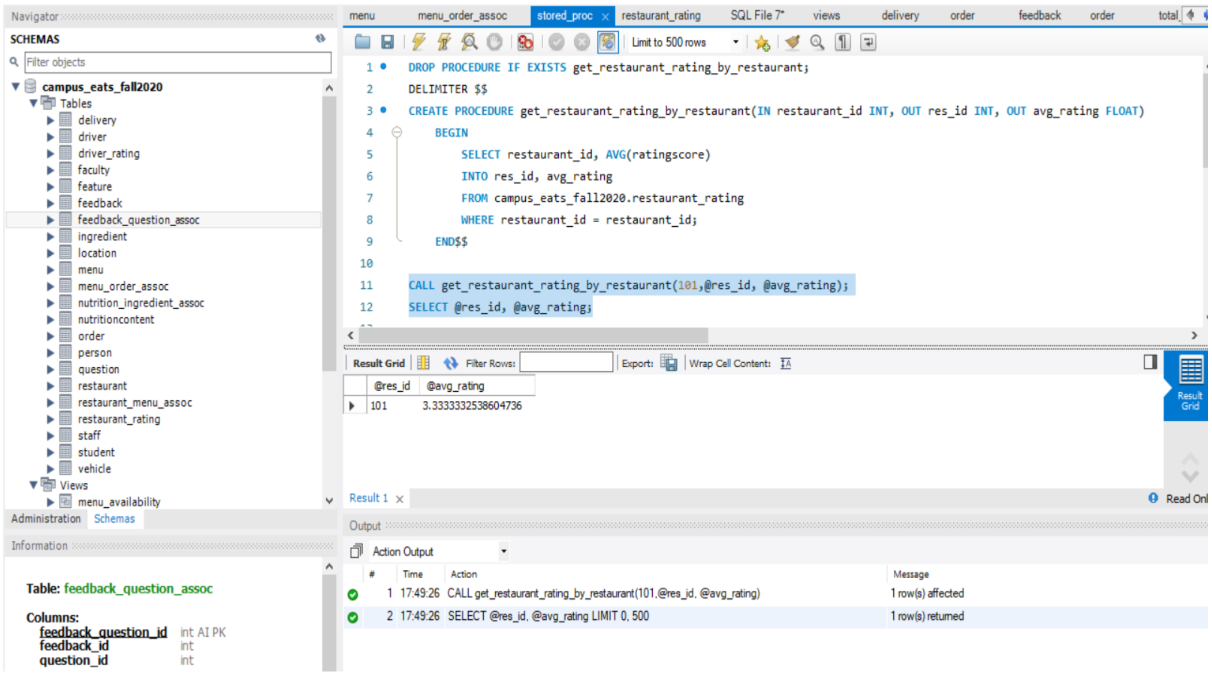
Deliverable 3
Stored Procedures

- Stored Procedure 1:



```
DROP PROCEDURE IF EXISTS get_restaurant_rating_by_restaurant;
DELIMITER $$
CREATE PROCEDURE get_restaurant_rating_by_restaurant(IN restaurant_id INT, OUT res_id INT, OUT avg_rating FLOAT)
    BEGIN
        SELECT restaurant_id, AVG(ratingscore)
        INTO res_id, avg_rating
        FROM campus_eats_fall2020.restaurant_rating
        WHERE restaurant_id = restaurant_id;
    END$$

CALL get_restaurant_rating_by_restaurant(101,@res_id, @avg_rating);
SELECT @res_id, @avg_rating;
```

```
@res_id   @avg_rating
101       3.3333332538604736
```

```
#  Time      Action                                                            Message
1  17:49:26  CALL get_restaurant_rating_by_restaurant(101,@res_id, @avg_rating)  1 row(s) affected
2  17:49:26  SELECT @res_id, @avg_rating LIMIT 0, 500                            1 row(s) returned
```

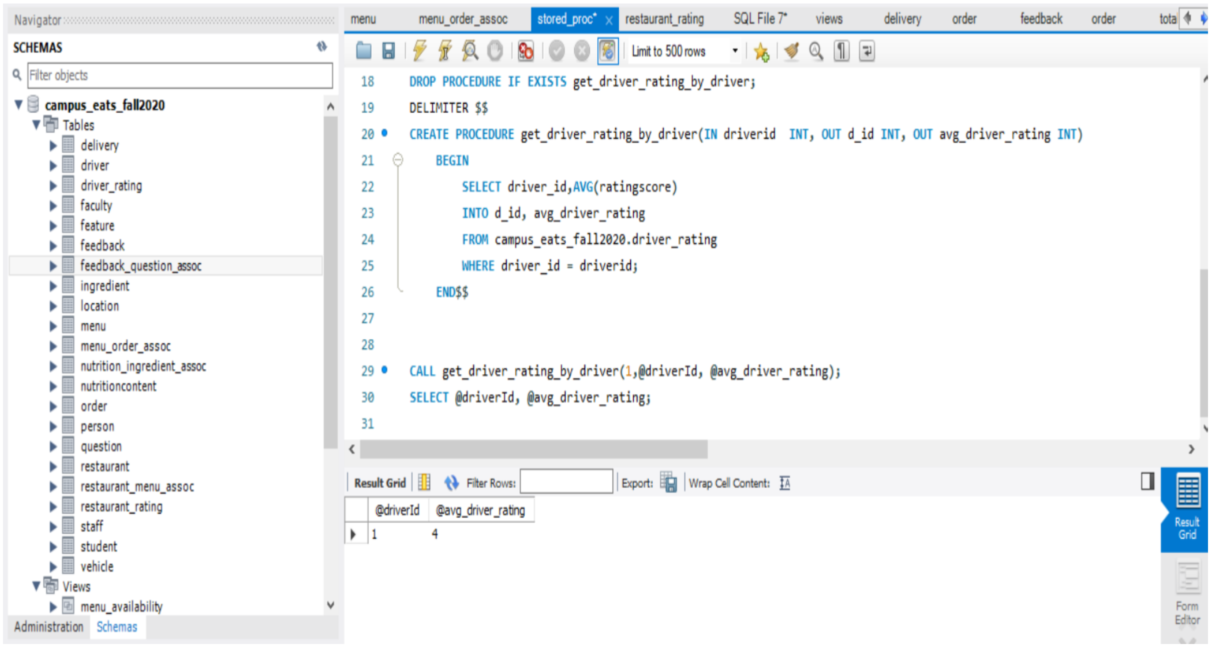
Table: feedback_question_assoc

Columns:

```
feedback_question_id   int AI PK
feedback_id            int
question_id            int
```

Stored procedure 1 returns the Average Rating of a particular restaurant.

- Stored Procedure 2:



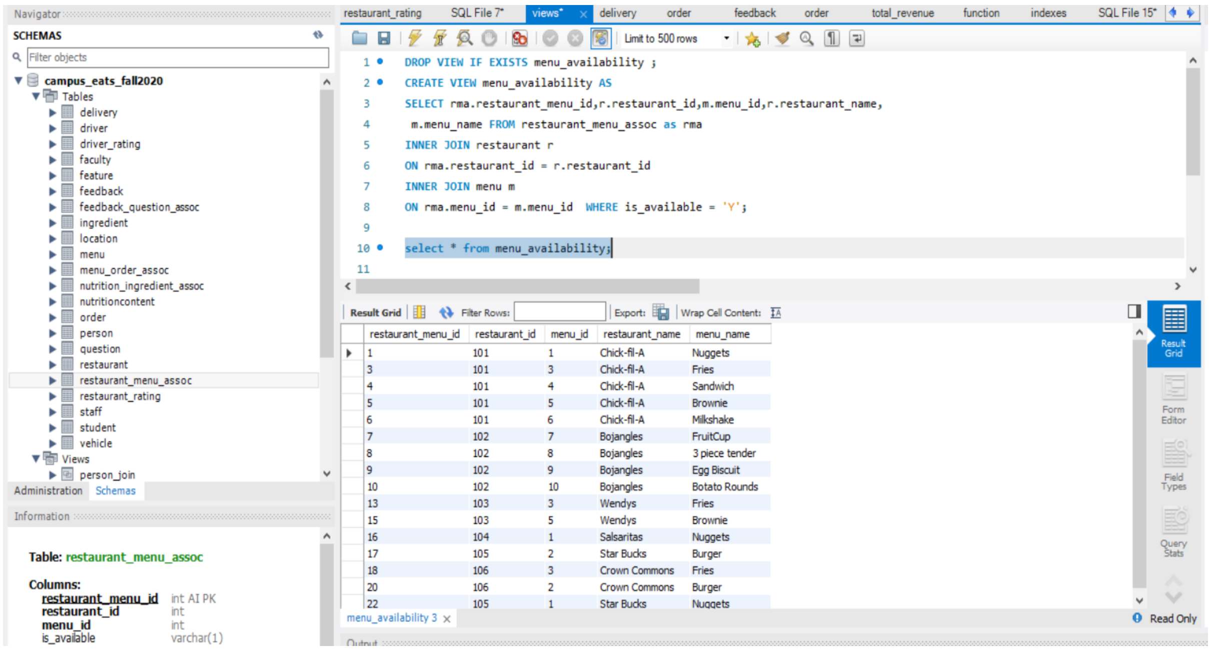
```
DROP PROCEDURE IF EXISTS get_driver_rating_by_driver;
DELIMITER $$
CREATE PROCEDURE get_driver_rating_by_driver(IN driverid  INT, OUT d_id INT, OUT avg_driver_rating INT)
    BEGIN
        SELECT driver_id,AVG(ratingscore)
        INTO d_id, avg_driver_rating
        FROM campus_eats_fall2020.driver_rating
        WHERE driver_id = driverid;
    END$$


CALL get_driver_rating_by_driver(1,@driverId, @avg_driver_rating);
SELECT @driverId, @avg_driver_rating;
```

```
@driverId   @avg_driver_rating
1           4
```

Stored procedure 2 returns the Average Rating of a driver.

Views

- View 1:

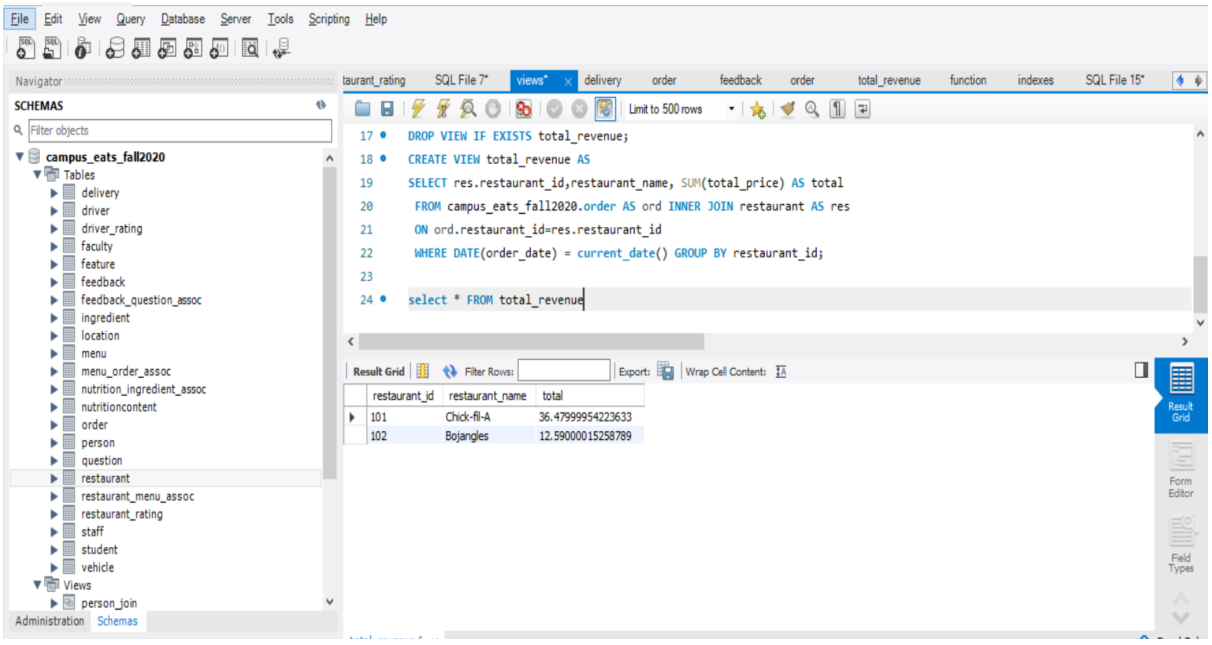


```sql
DROP VIEW IF EXISTS menu_availability ;
CREATE VIEW menu_availability AS
SELECT rma.restaurant_menu_id,r.restaurant_id,m.menu_id,r.restaurant_name,
 m.menu_name FROM restaurant_menu_assoc as rma
INNER JOIN restaurant r
ON rma.restaurant_id = r.restaurant_id
INNER JOIN menu m
ON rma.menu_id = m.menu_id  WHERE is_available = 'Y';

select * from menu_availability;
```

restaurant_menu_id	restaurant_id	menu_id	restaurant_name	menu_name
1	101	1	Chick-fil-A	Nuggets
3	101	3	Chick-fil-A	Fries
4	101	4	Chick-fil-A	Sandwich
5	101	5	Chick-fil-A	Brownie
6	101	6	Chick-fil-A	Milkshake
7	102	7	Bojangles	FruitCup
8	102	8	Bojangles	3 piece tender
9	102	9	Bojangles	Egg Biscuit
10	102	10	Bojangles	Botato Rounds
13	103	3	Wendys	Fries
15	103	5	Wendys	Brownie
16	104	1	Salsaritas	Nuggets
17	105	2	Star Bucks	Burger
18	106	3	Crown Commons	Fries
20	106	2	Crown Commons	Burger
22	105	1	Star Bucks	Nuggets

View 1 shows menu availability in a restaurant and returns restaurant name and menu name.

- View 2:

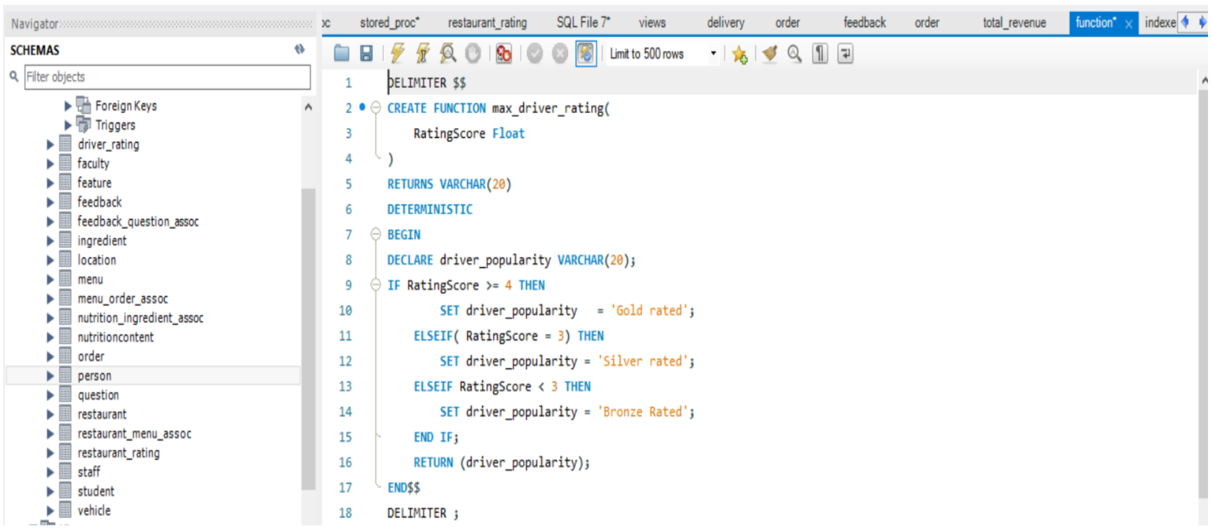


```sql
DROP VIEW IF EXISTS total_revenue;
CREATE VIEW total_revenue AS
SELECT res.restaurant_id,restaurant_name, SUM(total_price) AS total
 FROM campus_eats_fall2020.order AS ord INNER JOIN restaurant AS res
 ON ord.restaurant_id=res.restaurant_id
 WHERE DATE(order_date) = current_date() GROUP BY restaurant_id;

select * FROM total_revenue
```

restaurant_id	restaurant_name	total
101	Chick-fil-A	36.47999954223633
102	Bojangles	12.59000015258789

View 2 shows total revenue of a restaurant and returns the restaurant name and it’s revenue.

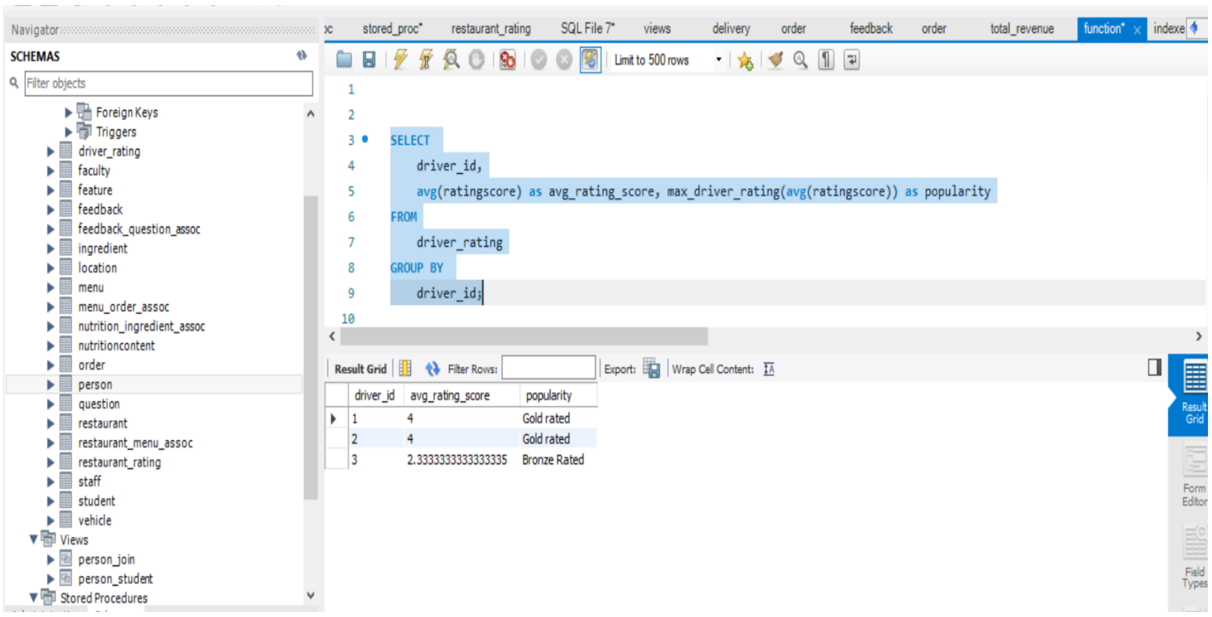
Function



```sql
DELIMITER $$
CREATE FUNCTION max_driver_rating(
    RatingScore Float
)
RETURNS VARCHAR(20)
DETERMINISTIC
BEGIN
DECLARE driver_popularity VARCHAR(20);
IF RatingScore >= 4 THEN
        SET driver_popularity   = 'Gold rated';
    ELSEIF( RatingScore = 3) THEN
        SET driver_popularity = 'Silver rated';
    ELSEIF RatingScore < 3 THEN
        SET driver_popularity = 'Bronze Rated';
    END IF;
    RETURN (driver_popularity);
END$$
DELIMITER ;
```

Created a function named max_driver_rating which returns a driver's rating.

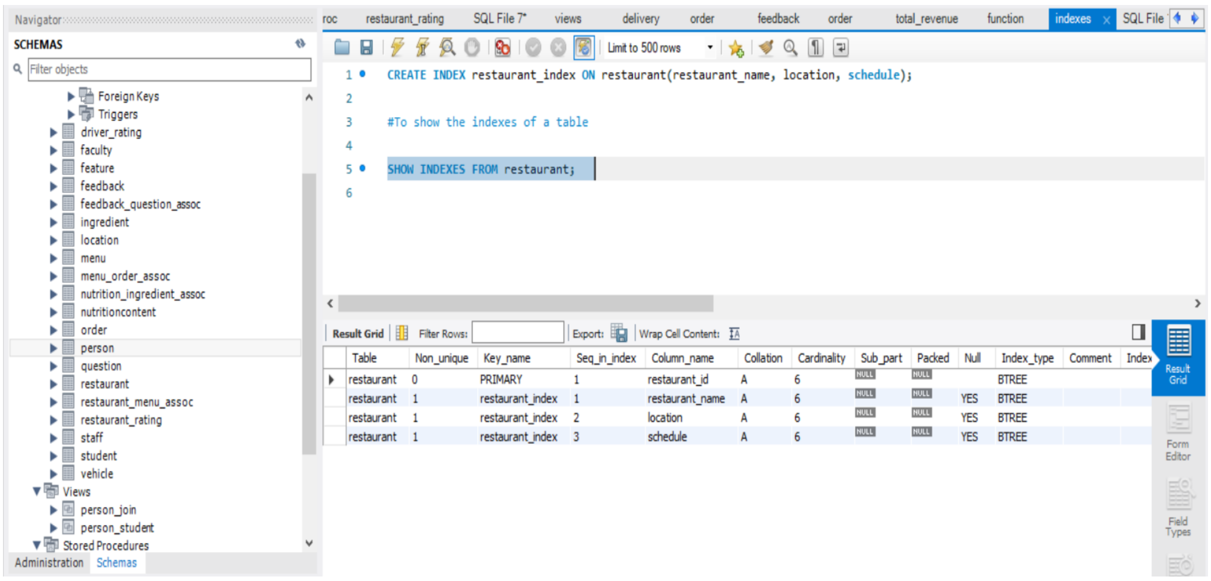
Function Call



```sql
SELECT
    driver_id,
    avg(ratingscore) as avg_rating_score, max_driver_rating(avg(ratingscore)) as popularity
FROM
    driver_rating
GROUP BY
    driver_id;
```

driver_id	avg_rating_score	popularity
1	4	Gold rated
2	4	Gold rated
3	2.3333333333333335	Bronze Rated

Index



```sql
CREATE INDEX restaurant_index ON restaurant(restaurant_name, location, schedule);

#To show the indexes of a table

SHOW INDEXES FROM restaurant;

```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index
restaurant	0	PRIMARY	1	restaurant_id	A	6	NULL	NULL		BTREE		
restaurant	1	restaurant_index	1	restaurant_name	A	6	NULL	NULL	YES	BTREE		
restaurant	1	restaurant_index	2	location	A	6	NULL	NULL	YES	BTREE		
restaurant	1	restaurant_index	3	schedule	A	6	NULL	NULL	YES	BTREE		

The index we have created returns three indexed columns which are restaurant_name, location and schedule.