

## Stored Procedure 1:

**Name of the stored procedure:** get\_count\_order\_by\_restaurant

**Purpose:** To get the count of orders for a restaurant based on restaurant Id

**Code:**

This code consist of code which drops the stored procedure if it exists. The stored procedure accept 1 input parameter which is restaurant Id and one output parameter which returns the count of orders for that restaurant. Then call to the stored procedure and displaying the results.

```
DROP PROCEDURE IF EXISTS get_count_order_by_restaurant;
DELIMITER $$
CREATE PROCEDURE get_count_order_by_restaurant(IN res_id INT, OUT count_orders INT)
BEGIN
    SELECT COUNT(*)
    INTO count_orders
    FROM campus_eats_fall2020.order
    WHERE restaurant_id = res_id;
END$$

CALL get_count_order_by_restaurant(5,@count_orders);
SELECT @count_orders as orders_count;
```

**Screenshot:**

The screenshot displays a database management interface. On the left, a tree view shows the database structure, with the 'order' table selected under the 'campus\_eats\_fall2020' schema. The main window shows the SQL script being executed, which includes dropping the procedure, creating it with parameters, and calling it with the restaurant ID 5. The results pane at the bottom shows a single row with the value 2 for the 'orders\_count' column. The table structure for 'order' is also visible at the bottom left.

**Table: order**

**Columns:**

order_id	int AI PK
person_id	int
delivery_id	int
location_id	int

**Result 5**

orders_count
2

## Stored Procedure 2:

**Name of the stored procedure:** get\_driver\_delivery\_count

**Purpose:** To get the number of deliveries done by the driver

**Code:**

This code consist of code which drops the stored procedure if it exists. The stored procedure accept 1 input parameter which is driver Id and one output parameter which returns the count of deliveries done by that driver. Then call to the stored procedure and displaying the results.

```
DROP PROCEDURE IF EXISTS get_driver_delivery_count;
DELIMITER $$
CREATE PROCEDURE get_driver_delivery_count(IN driverId INT, OUT count_delivery INT)
BEGIN
    SELECT COUNT(*)
    INTO count_delivery
    FROM campus_eats_fall2020.delivery
    WHERE driver_id = driverId;
END$$
```

```
CALL get_driver_delivery_count(4,@count_delivery);
SELECT @count_delivery as 'Number of deliveries';
```

**Screenshot:**

The screenshot displays a SQL IDE interface. On the left, a 'Filter objects' pane shows a tree view of the 'campus\_eats\_fall2020' database, including tables (delivery, driver, faculty, location, order, person, restaurant, staff, student, vehicle) and stored procedures (add\_person, get\_count\_order\_by\_restaurant, get\_driver\_delivery\_count). The 'get\_driver\_delivery\_count' procedure is selected. The main editor shows the SQL code for dropping the procedure, creating it with parameters, and calling it with driver ID 4. Below the code, a 'Result Grid' shows the output of the call, with a single row containing the value '16' under the column 'Number of deliveries'. The bottom status bar indicates the schema is 'campus\_eats\_fall2020' and the result is 'Result 2'.

```
1 • DROP PROCEDURE IF EXISTS get_driver_delivery_count;
2   DELIMITER $$
3 • CREATE PROCEDURE get_driver_delivery_count(IN driverId INT, OUT count_delivery INT)
4     BEGIN
5       SELECT COUNT(*)
6       INTO count_delivery
7       FROM campus_eats_fall2020.delivery
8       WHERE driver_id = driverId;
9     END$$
10
11 • CALL get_driver_delivery_count(4,@count_delivery);
12 SELECT @count_delivery as 'Number of deliveries';
```

Number of deliveries
16

Schema: campus\_eats\_fall2020

Result 2 x