UE19CS301

DATABASE MANAGEMENT SYSTEM

Online Food Delivery System

TEAM 16

Sayonika Das - PES1UG19CS439

Sejal Maurya - PES1UG19CS440

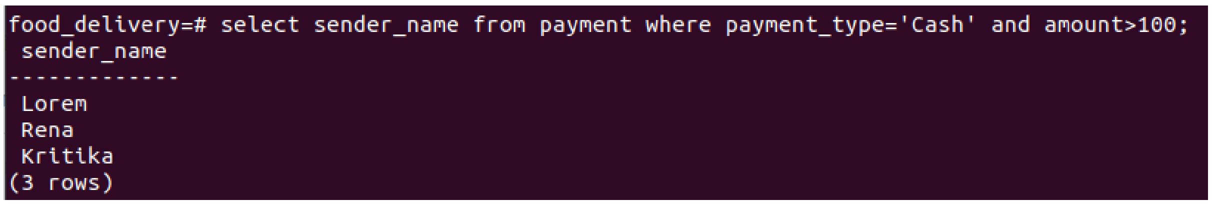
Sheetal S - PES1UG19CS455

Section G, Semester 5, CSE

QUERIES AND CORRESPONDING ANSWERS

- **Display all customers that paid above 100 and by cash.**

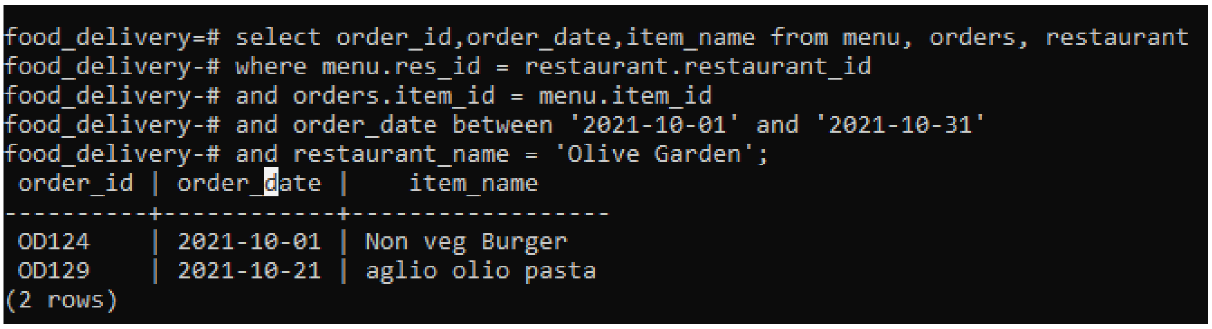
  select sender_name from payment where payment_type='Cash' and amount>100;



```
food_delivery=# select sender_name from payment where payment_type='Cash' and amount>100;
 sender_name
-------------
 Lorem
 Rena
 Kritika
(3 rows)
```

- **Display all items bought in the month of october from Olive Garden.**

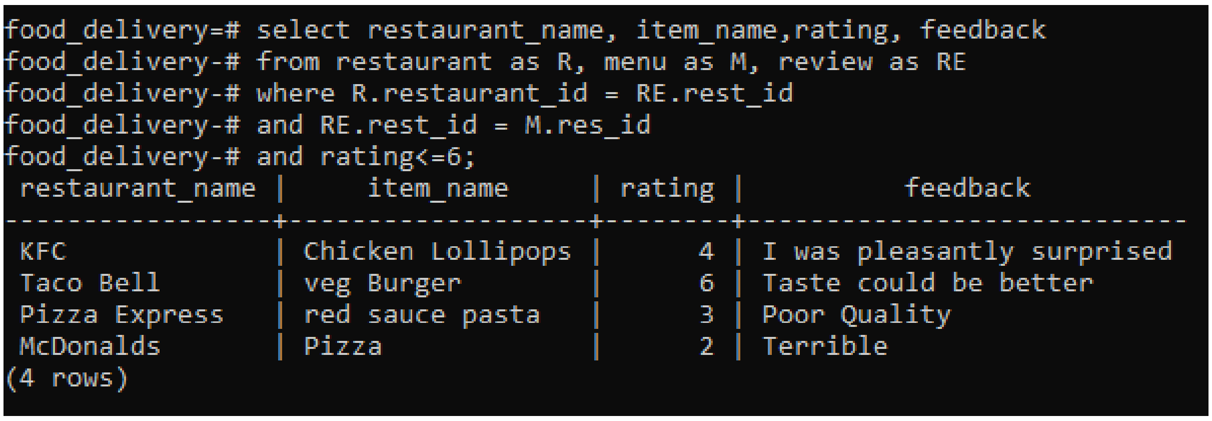
  select order_id,order_date,item_name from menu, orders, restaurant
  where menu.res_id = restaurant.restaurant_id
  and orders.item_id = menu.item_id
  and order_date between '2021-10-01' and '2021-10-31'
  and restaurant_name = 'Olive Garden';



```
food_delivery=# select order_id,order_date,item_name from menu, orders, restaurant
food_delivery-# where menu.res_id = restaurant.restaurant_id
food_delivery-# and orders.item_id = menu.item_id
food_delivery-# and order_date between '2021-10-01' and '2021-10-31'
food_delivery-# and restaurant_name = 'Olive Garden';
 order_id | order_date |     item_name
----------+------------+------------------
 OD124    | 2021-10-01 | Non veg Burger
 OD129    | 2021-10-21 | aglio olio pasta
(2 rows)
```

- **Display Restaurant name and Item name and review for all rating <= 6**

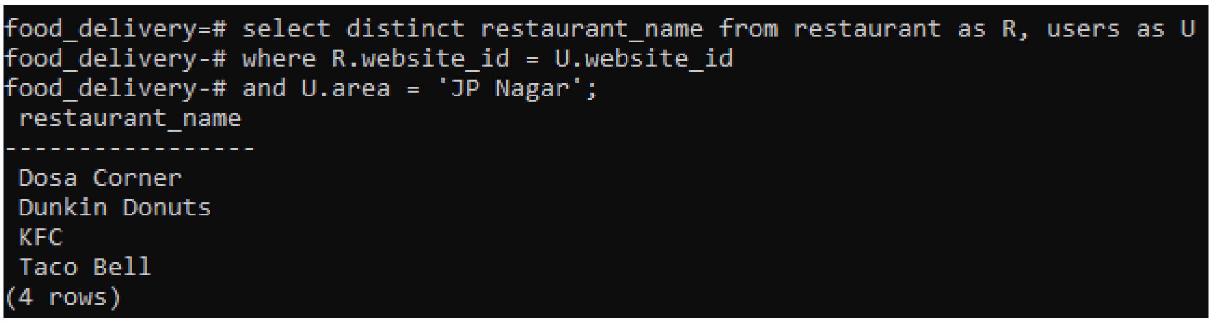
  select restaurant_name, item_name,rating, feedback
  from restaurant as R, menu as M, review as RE
  where R.restaurant_id = RE.rest_id
  and RE.rest_id = M.res_id
  and rating<=6;



```
food_delivery=# select restaurant_name, item_name,rating, feedback
food_delivery-# from restaurant as R, menu as M, review as RE
food_delivery-# where R.restaurant_id = RE.rest_id
food_delivery-# and RE.rest_id = M.res_id
food_delivery-# and rating<=6;
 restaurant_name |     item_name     | rating |          feedback
-----------------+-------------------+--------+-----------------------------
 KFC             | Chicken Lollipops |      4 | I was pleasantly surprised
 Taco Bell       | veg Burger        |      6 | Taste could be better
 Pizza Express   | red sauce pasta   |      3 | Poor Quality
 McDonalds       | Pizza             |      2 | Terrible
(4 rows)
```

- **Display names of all restaurants which users from JP Nagar are ordering from**

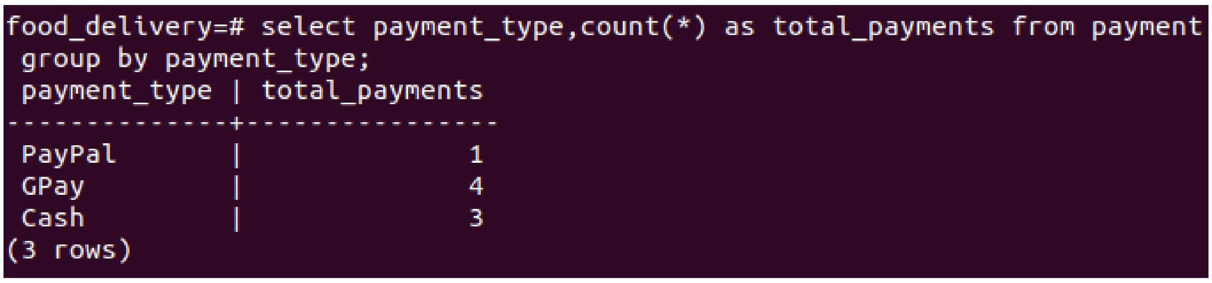
  select distinct restaurant_name from restaurant as R, users as U
  where R.website_id = U.website_id
  and U.area = 'JP Nagar';



```
food_delivery=# select distinct restaurant_name from restaurant as R, users as U
food_delivery-# where R.website_id = U.website_id
food_delivery-# and U.area = 'JP Nagar';
 restaurant_name
-----------------
 Dosa Corner
 Dunkin Donuts
 KFC
 Taco Bell
(4 rows)
```

- **Display number of payments made by each payment type**

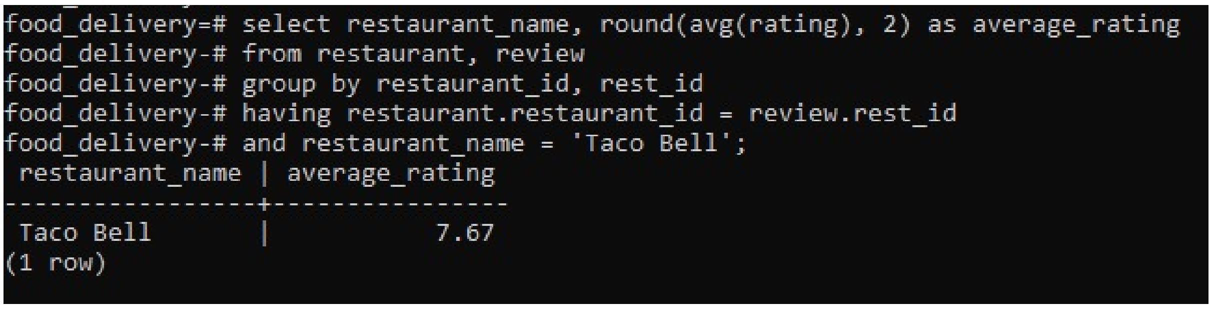
  select payment_type,count(*) as total_payments from payment group by payment_type;



```
food_delivery=# select payment_type,count(*) as total_payments from payment
 group by payment_type;
 payment_type | total_payments
--------------+----------------
 PayPal       |              1
 GPay         |              4
 Cash         |              3
(3 rows)
```

- **Display the average Rating given to Taco Bell**

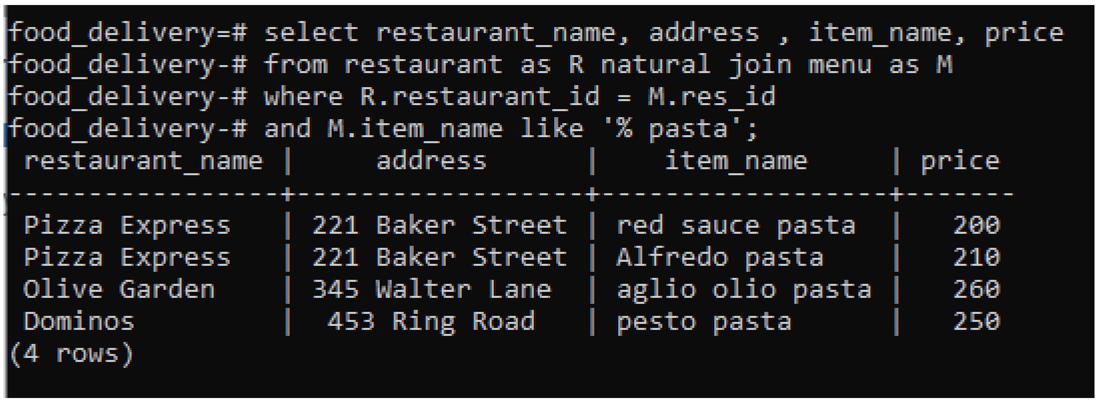
  select restaurant_name, round(avg(rating), 2) as average_rating from restaurant, review
  group by restaurant_id, rest_id
  having restaurant.restaurant_id = review.rest_id and restaurant_name = 'Taco Bell';



```
food_delivery=# select restaurant_name, round(avg(rating), 2) as average_rating
food_delivery-# from restaurant, review
food_delivery-# group by restaurant_id, rest_id
food_delivery-# having restaurant.restaurant_id = review.rest_id
food_delivery-# and restaurant_name = 'Taco Bell';
 restaurant_name | average_rating
-----------------+----------------
 Taco Bell       |           7.67
(1 row)
```

- **Display the names and addresses along with prices of all restaurants that serve pasta**

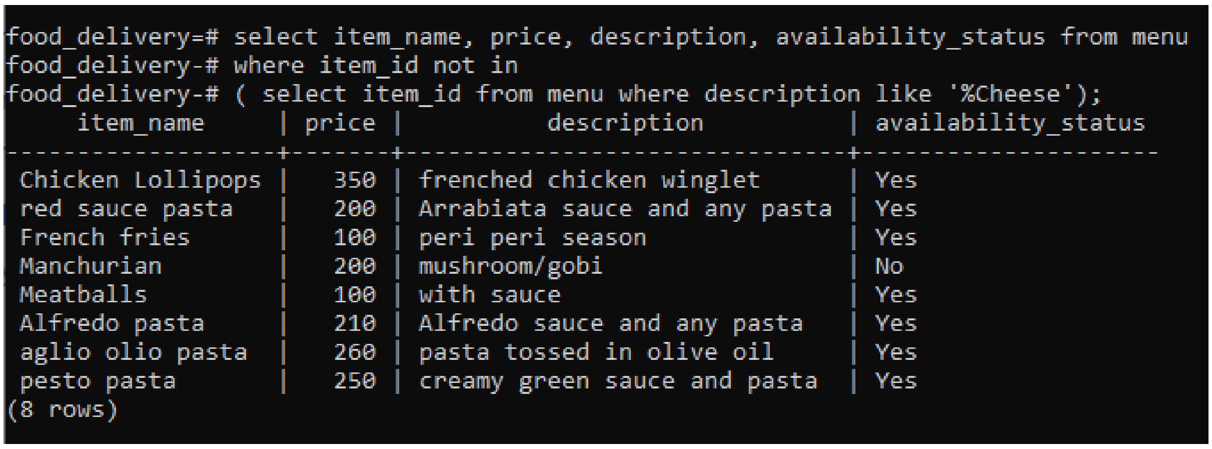
  select restaurant_name, address , item_name, price from restaurant as R natural join menu as M
  where R.restaurant_id = M.res_id and M.item_name like '% pasta';



```
food_delivery=# select restaurant_name, address , item_name, price
food_delivery-# from restaurant as R natural join menu as M
food_delivery-# where R.restaurant_id = M.res_id
food_delivery-# and M.item_name like '% pasta';
 restaurant_name |     address      |     item_name     | price
-----------------+------------------+-------------------+-------
 Pizza Express   | 221 Baker Street | red sauce pasta   |   200
 Pizza Express   | 221 Baker Street | Alfredo pasta     |   210
 Olive Garden    | 345 Walter Lane  | aglio olio pasta  |   260
 Dominos         |  453 Ring Road   | pesto pasta       |   250
(4 rows)
```

- **Display all the dishes from the menu that do not contain cheese in them for a lactose intolerant customer.**

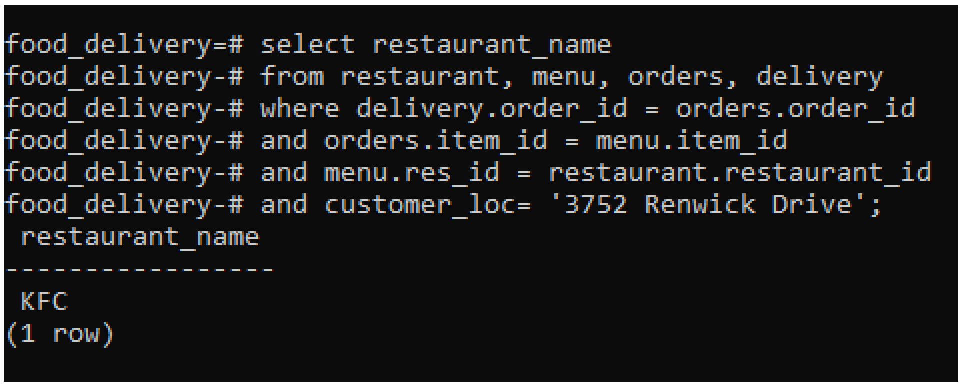
  select item_name, price, description, availability_status from menu
  where item_id not in (select item_id from menu where description like '%Cheese');



```
food_delivery=# select item_name, price, description, availability_status from menu
food_delivery-# where item_id not in
food_delivery-# ( select item_id from menu where description like '%Cheese');
     item_name      | price |          description           | availability_status
--------------------+-------+--------------------------------+---------------------
 Chicken Lollipops  |   350 | frenched chicken winglet       | Yes
 red sauce pasta    |   200 | Arrabiata sauce and any pasta  | Yes
 French fries       |   100 | peri peri season               | Yes
 Manchurian         |   200 | mushroom/gobi                  | No
 Meatballs          |   100 | with sauce                     | Yes
 Alfredo pasta      |   210 | Alfredo sauce and any pasta    | Yes
 aglio olio pasta   |   260 | pasta tossed in olive oil      | Yes
 pesto pasta        |   250 | creamy green sauce and pasta   | Yes
(8 rows)
```

- **Display all restaurants that have delivered to 3752 Renwick Drive.**

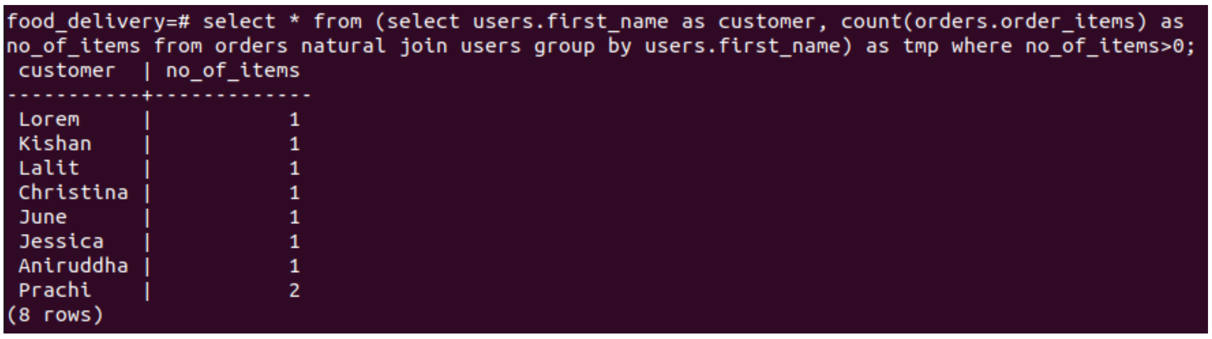
  select restaurant_name from restaurant, menu, orders, delivery
  where delivery.order_id = orders.order_id
  and orders.item_id = menu.item_id
  and menu.res_id = restaurant.restaurant_id
  and customer_loc= '3752 Renwick Drive';



```
food_delivery=# select restaurant_name
food_delivery-# from restaurant, menu, orders, delivery
food_delivery-# where delivery.order_id = orders.order_id
food_delivery-# and orders.item_id = menu.item_id
food_delivery-# and menu.res_id = restaurant.restaurant_id
food_delivery-# and customer_loc= '3752 Renwick Drive';
 restaurant_name
-----------------
 KFC
(1 row)
```

- **Display number of items ordered by a customer**

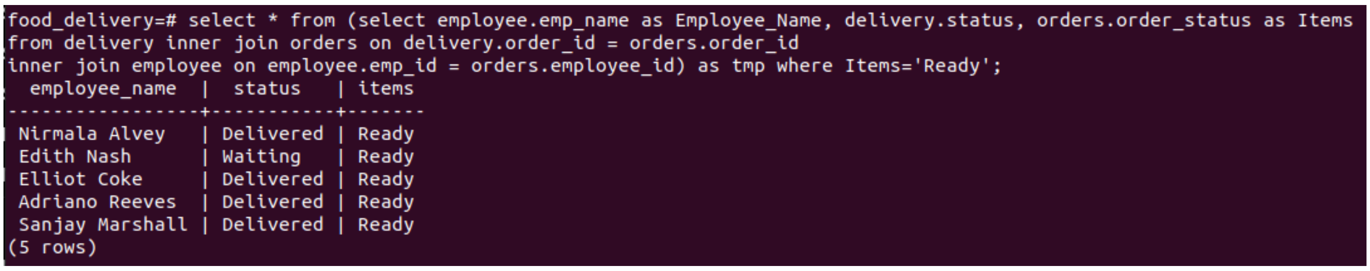
  select * from (select users.first_name as customer, count(orders.order_items)
  as no_of_items from orders natural join users group by users.first_name)
  as tmp where no_of_items>0;



```
food_delivery=# select * from (select users.first_name as customer, count(orders.order_items) as
no_of_items from orders natural join users group by users.first_name) as tmp where no_of_items>0;
 customer  | no_of_items
-----------+-------------
 Lorem     |           1
 Kishan    |           1
 Lalit     |           1
 Christina |           1
 June      |           1
 Jessica   |           1
 Aniruddha |           1
 Prachi    |           2
(8 rows)
```

- **Display delivery person's name along with delivery status for items that are ready.**

  select * from (select employee.emp_name as Employee_Name, delivery.status, orders.order_status as Items from delivery inner join orders on delivery.order_id = orders.order_id inner join employee on employee.emp_id = orders.employee_id) as tmp where Items='Ready';



```
food_delivery=# select * from (select employee.emp_name as Employee_Name, delivery.status, orders.order_status as Items
from delivery inner join orders on delivery.order_id = orders.order_id
inner join employee on employee.emp_id = orders.employee_id) as tmp where Items='Ready';
  employee_name  |  status   | items
-----------------+-----------+-------
 Nirmala Alvey   | Delivered | Ready
 Edith Nash      | Waiting   | Ready
 Elliot Coke     | Delivered | Ready
 Adriano Reeves  | Delivered | Ready
 Sanjay Marshall | Delivered | Ready
(5 rows)
```

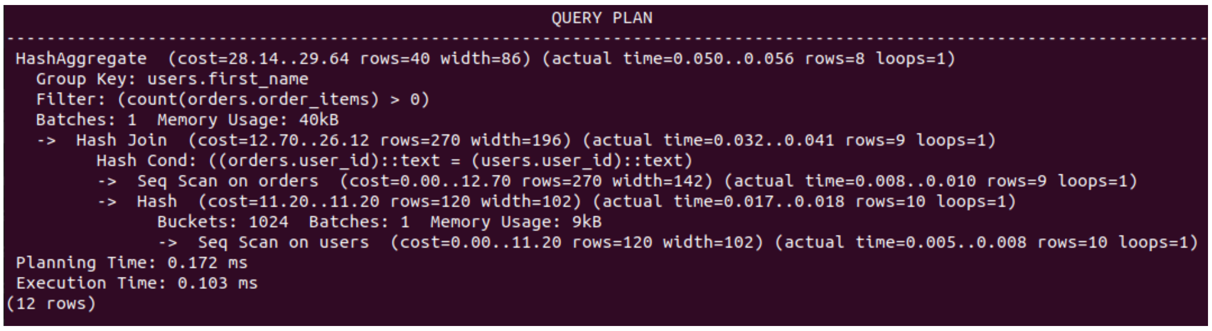
QUERY PLAN AND PERFORMANCE ANALYSIS

- PostgreSQL devises a query plan for each query it receives. Choosing the right plan to match the query structure and the properties of the data is essential for good performance.

- The tool used for understanding and optimizing SQL queries is EXPLAIN ANALYZE, which is a Postgres command that accepts a statement such as SELECT ..., UPDATE ..., or DELETE ..., executes the statement, and instead of returning the data provides a query plan detailing what approach the planner took to executing the statement provided.

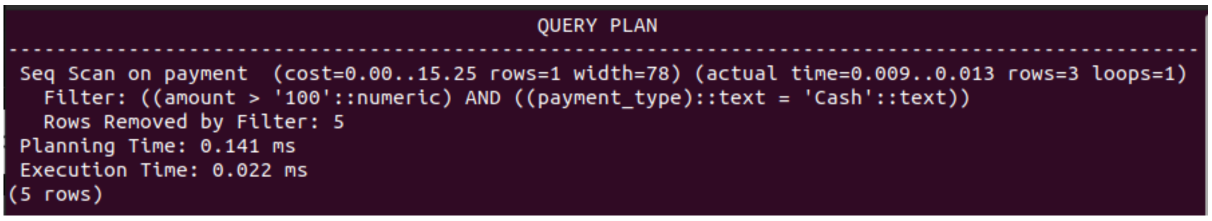
- It displays the true row counts and true run time accumulated within each plan node.

1. EXPLAIN ANALYZE select * from (select users.first_name as customer, count(orders.order_items) as no_of_items from orders natural join users group by users.first_name) as tmp where no_of_items>0;



```
                                                    QUERY PLAN
------------------------------------------------------------------------------------------------------------------------
 HashAggregate  (cost=28.14..29.64 rows=40 width=86) (actual time=0.050..0.056 rows=8 loops=1)
   Group Key: users.first_name
   Filter: (count(orders.order_items) > 0)
   Batches: 1  Memory Usage: 40kB
   ->  Hash Join  (cost=12.70..26.12 rows=270 width=196) (actual time=0.032..0.041 rows=9 loops=1)
         Hash Cond: ((orders.user_id)::text = (users.user_id)::text)
         ->  Seq Scan on orders  (cost=0.00..12.70 rows=270 width=142) (actual time=0.008..0.010 rows=9 loops=1)
         ->  Hash  (cost=11.20..11.20 rows=120 width=102) (actual time=0.017..0.018 rows=10 loops=1)
               Buckets: 1024  Batches: 1  Memory Usage: 9kB
               ->  Seq Scan on users  (cost=0.00..11.20 rows=120 width=102) (actual time=0.005..0.008 rows=10 loops=1)
 Planning Time: 0.172 ms
 Execution Time: 0.103 ms
(12 rows)
```

2. EXPLAIN ANALYZE select sender_name from payment where payment_type='Cash' and amount>100;



```
                                       QUERY PLAN
-----------------------------------------------------------------------------------------
 Seq Scan on payment  (cost=0.00..15.25 rows=1 width=78) (actual time=0.009..0.013 rows=3 loops=1)
   Filter: ((amount > '100'::numeric) AND ((payment_type)::text = 'Cash'::text))
   Rows Removed by Filter: 5
 Planning Time: 0.141 ms
 Execution Time: 0.022 ms
(5 rows)
```

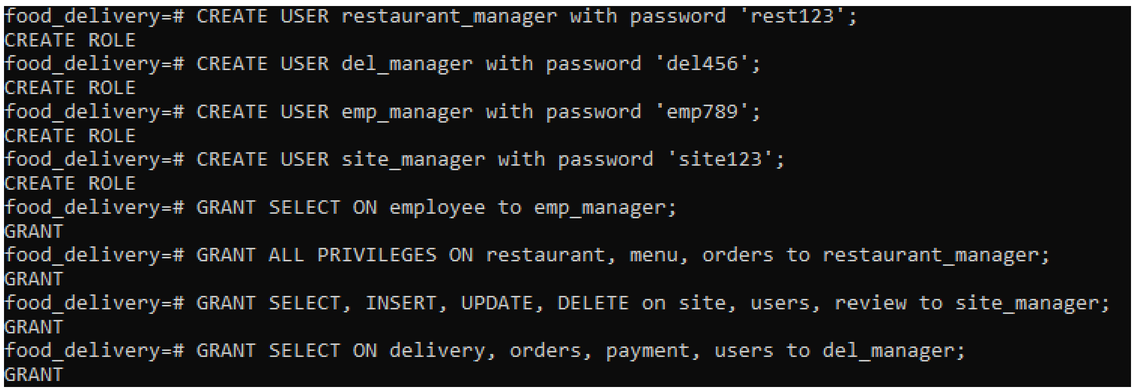
MULTIPLE USER ACCESS

- Four admins are created with different access privileges for different parts of the database.

- There are managers to maintain records related to:
  1. Restaurants
  2. Deliveries
  3. Employees
  4. Website.
- User who can view relevant tables.

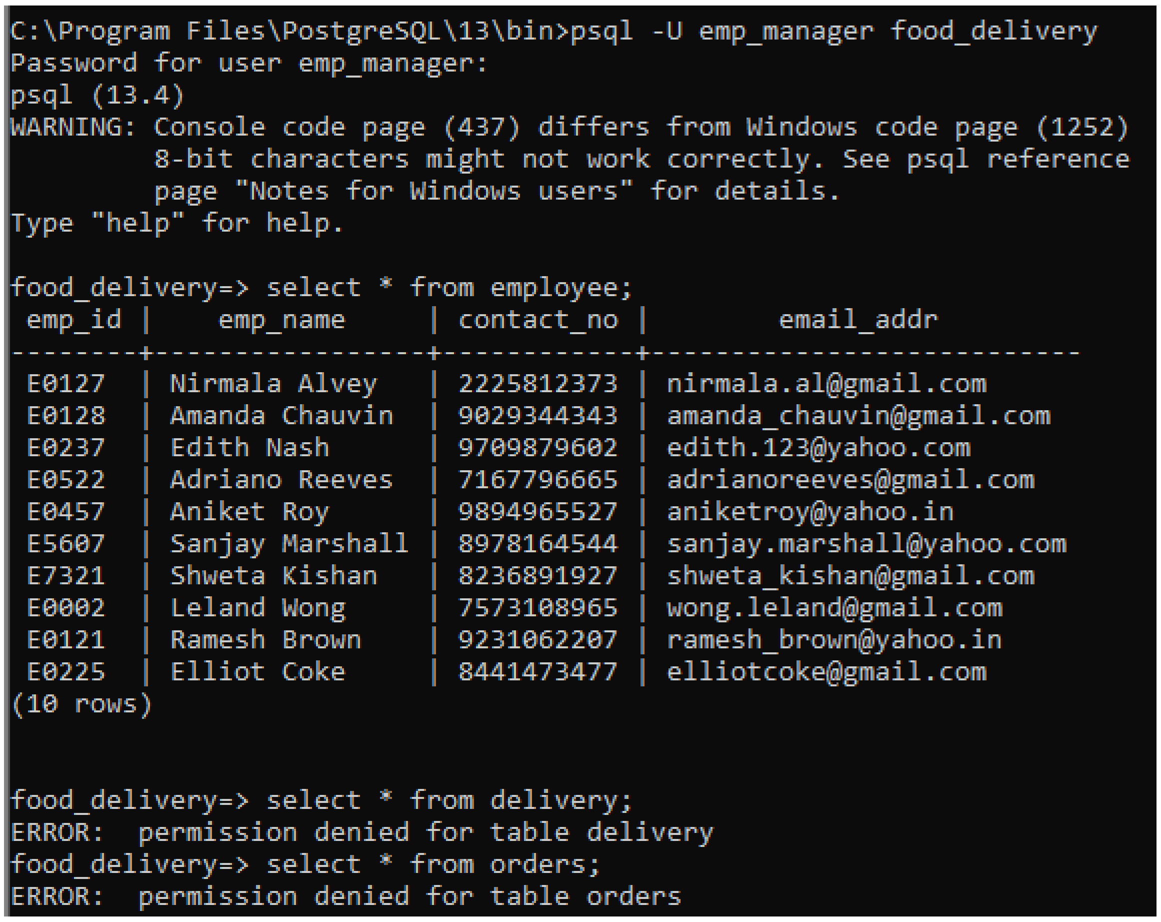
CREATE USER restaurant_manager with password 'rest123';
CREATE USER del_manager with password 'del456';
CREATE USER emp_manager with password 'emp789';
CREATE USER site_manager with password 'site123';
CREATE USER user27 WITH PASSWORD 'u123';

GRANT SELECT ON employee to emp_manager;
GRANT ALL PRIVILEGES ON restaurant, menu, orders to restaurant_manager;
GRANT SELECT, INSERT, UPDATE, DELETE on site, users, review to site_manager;
GRANT SELECT ON delivery, orders, payment, users to del_manager;
GRANT SELECT ON orders, menu, review, payment, delivery to user27;



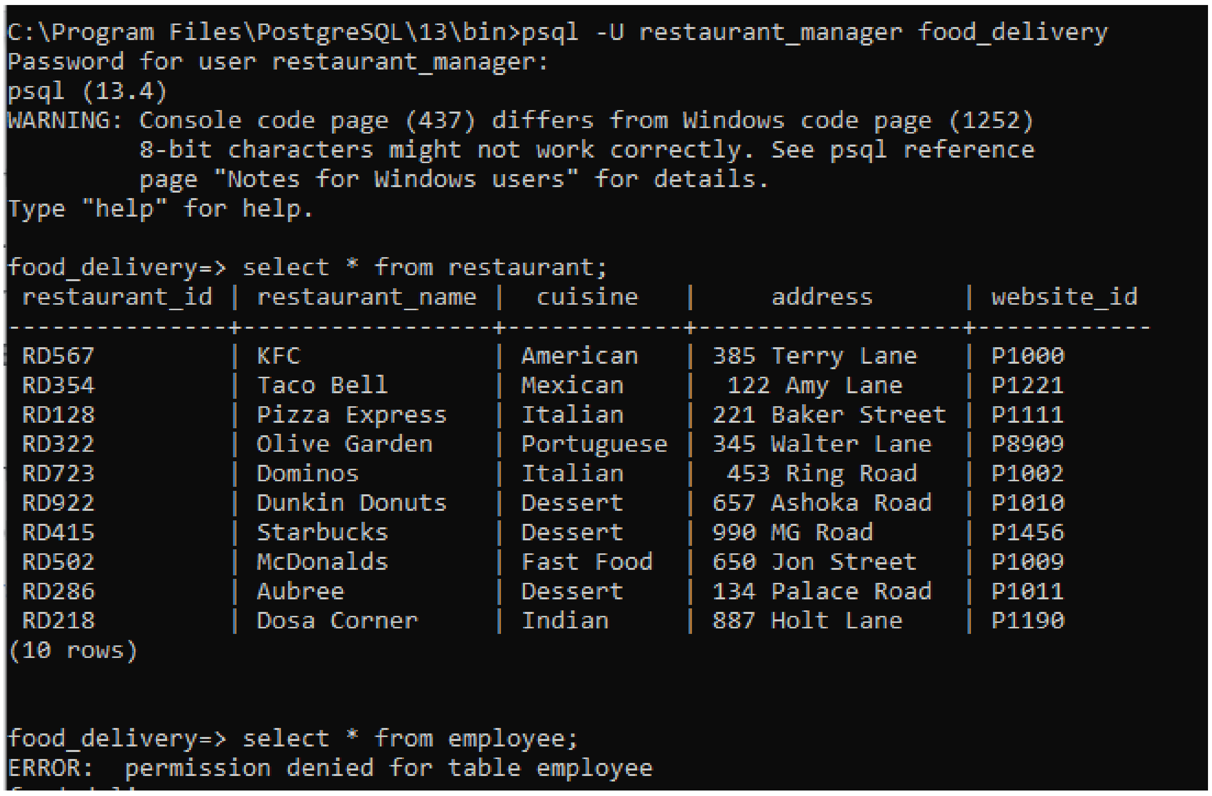
```
food_delivery=# CREATE USER restaurant_manager with password 'rest123';
CREATE ROLE
food_delivery=# CREATE USER del_manager with password 'del456';
CREATE ROLE
food_delivery=# CREATE USER emp_manager with password 'emp789';
CREATE ROLE
food_delivery=# CREATE USER site_manager with password 'site123';
CREATE ROLE
food_delivery=# GRANT SELECT ON employee to emp_manager;
GRANT
food_delivery=# GRANT ALL PRIVILEGES ON restaurant, menu, orders to restaurant_manager;
GRANT
food_delivery=# GRANT SELECT, INSERT, UPDATE, DELETE on site, users, review to site_manager;
GRANT
food_delivery=# GRANT SELECT ON delivery, orders, payment, users to del_manager;
GRANT
```

emp_manager -



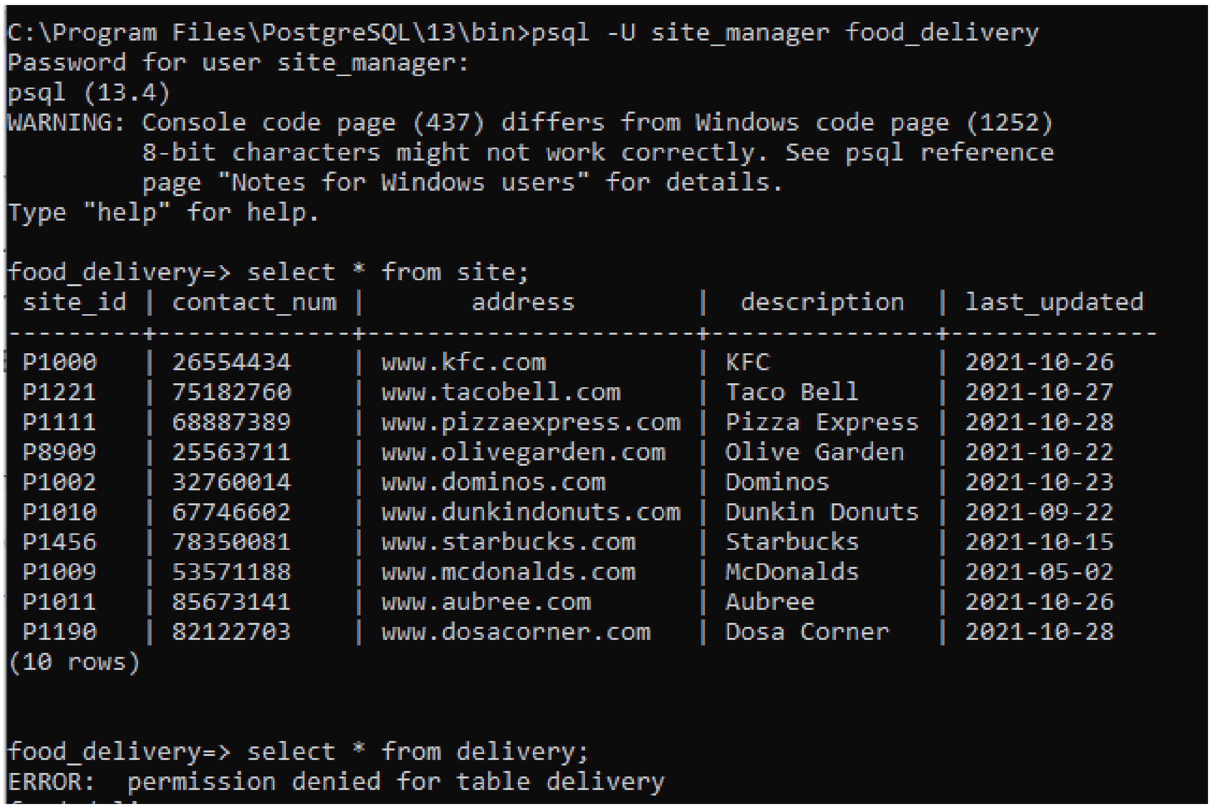
```
C:\Program Files\PostgreSQL\13\bin>psql -U emp_manager food_delivery
Password for user emp_manager:
psql (13.4)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

food_delivery=> select * from employee;
 emp_id |    emp_name     | contact_no |        email_addr
--------+-----------------+------------+---------------------------
 E0127  | Nirmala Alvey   | 2225812373 | nirmala.al@gmail.com
 E0128  | Amanda Chauvin  | 9029344343 | amanda_chauvin@gmail.com
 E0237  | Edith Nash      | 9709879602 | edith.123@yahoo.com
 E0522  | Adriano Reeves  | 7167796665 | adrianoreeves@gmail.com
 E0457  | Aniket Roy      | 9894965527 | aniketroy@yahoo.in
 E5607  | Sanjay Marshall | 8978164544 | sanjay.marshall@yahoo.com
 E7321  | Shweta Kishan   | 8236891927 | shweta_kishan@gmail.com
 E0002  | Leland Wong     | 7573108965 | wong.leland@gmail.com
 E0121  | Ramesh Brown    | 9231062207 | ramesh_brown@yahoo.in
 E0225  | Elliot Coke     | 8441473477 | elliotcoke@gmail.com
(10 rows)


food_delivery=> select * from delivery;
ERROR:  permission denied for table delivery
food_delivery=> select * from orders;
ERROR:  permission denied for table orders
```

restaurant_manager -



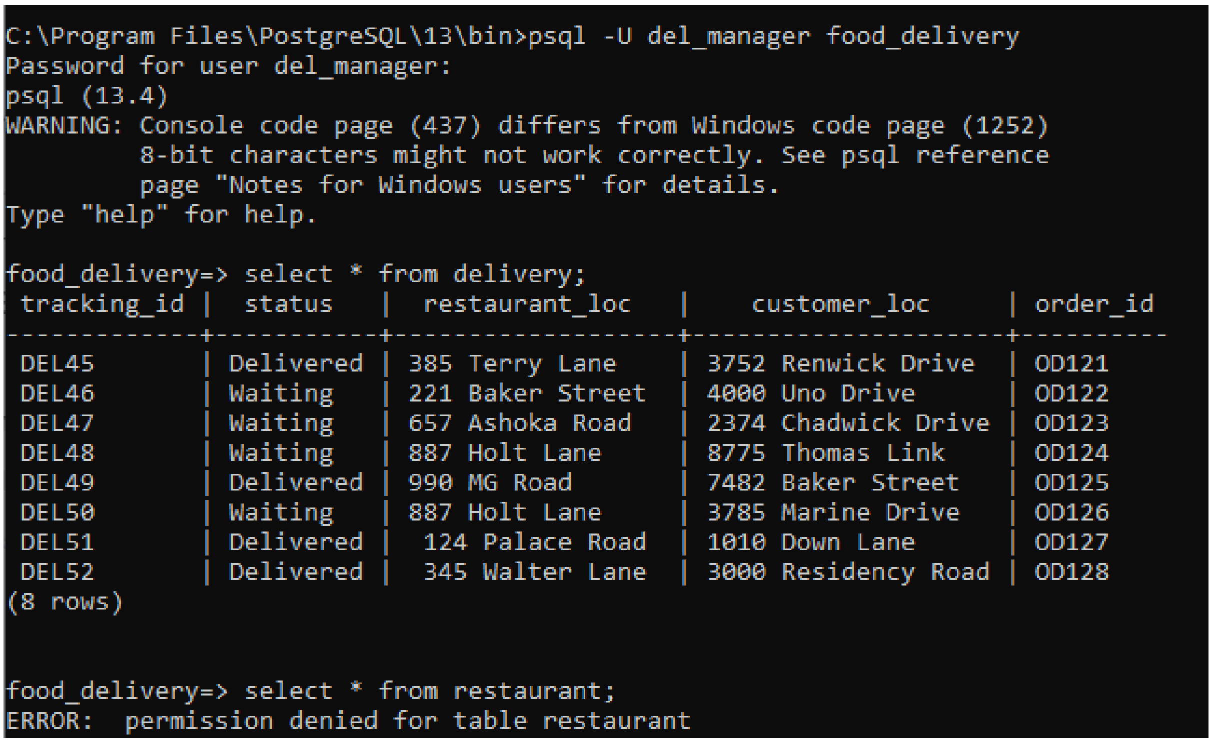
```
C:\Program Files\PostgreSQL\13\bin>psql -U restaurant_manager food_delivery
Password for user restaurant_manager:
psql (13.4)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

food_delivery=> select * from restaurant;
 restaurant_id | restaurant_name |  cuisine   |     address      | website_id
---------------+-----------------+------------+------------------+------------
 RD567         | KFC             | American   | 385 Terry Lane   | P1000
 RD354         | Taco Bell       | Mexican    |  122 Amy Lane    | P1221
 RD128         | Pizza Express   | Italian    | 221 Baker Street | P1111
 RD322         | Olive Garden    | Portuguese | 345 Walter Lane  | P8909
 RD723         | Dominos         | Italian    |  453 Ring Road   | P1002
 RD922         | Dunkin Donuts   | Dessert    | 657 Ashoka Road  | P1010
 RD415         | Starbucks       | Dessert    | 990 MG Road      | P1456
 RD502         | McDonalds       | Fast Food  | 650 Jon Street   | P1009
 RD286         | Aubree          | Dessert    | 134 Palace Road  | P1011
 RD218         | Dosa Corner     | Indian     | 887 Holt Lane    | P1190
(10 rows)


food_delivery=> select * from employee;
ERROR:  permission denied for table employee
```

site manager -



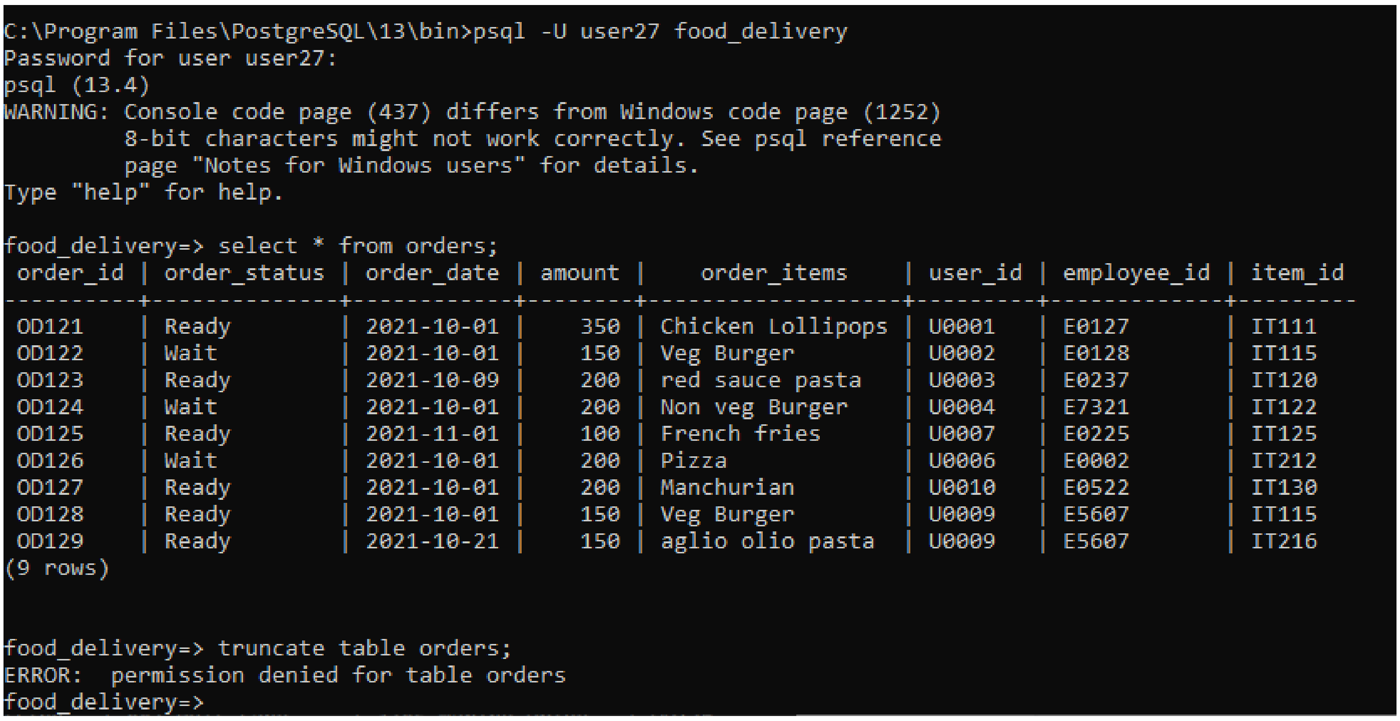
```
C:\Program Files\PostgreSQL\13\bin>psql -U site_manager food_delivery
Password for user site_manager:
psql (13.4)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

food_delivery=> select * from site;
 site_id | contact_num |        address         |  description   | last_updated
---------+-------------+------------------------+----------------+--------------
 P1000   | 26554434    | www.kfc.com            | KFC            | 2021-10-26
 P1221   | 75182760    | www.tacobell.com       | Taco Bell      | 2021-10-27
 P1111   | 68887389    | www.pizzaexpress.com   | Pizza Express  | 2021-10-28
 P8909   | 25563711    | www.olivegarden.com    | Olive Garden   | 2021-10-22
 P1002   | 32760014    | www.dominos.com        | Dominos        | 2021-10-23
 P1010   | 67746602    | www.dunkindonuts.com   | Dunkin Donuts  | 2021-09-22
 P1456   | 78350081    | www.starbucks.com      | Starbucks      | 2021-10-15
 P1009   | 53571188    | www.mcdonalds.com      | McDonalds      | 2021-05-02
 P1011   | 85673141    | www.aubree.com         | Aubree         | 2021-10-26
 P1190   | 82122703    | www.dosacorner.com     | Dosa Corner    | 2021-10-28
(10 rows)


food_delivery=> select * from delivery;
ERROR:  permission denied for table delivery
```

del_manager -



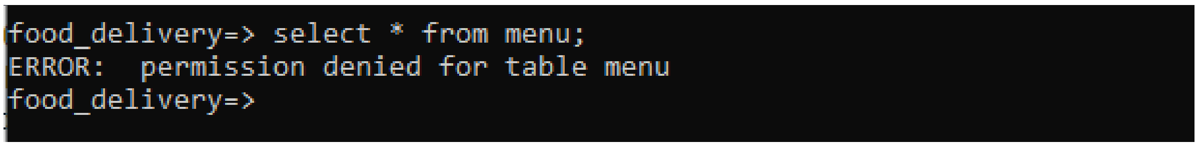
```
C:\Program Files\PostgreSQL\13\bin>psql -U del_manager food_delivery
Password for user del_manager:
psql (13.4)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

food_delivery=> select * from delivery;
 tracking_id |  status   |  restaurant_loc    |    customer_loc      | order_id
-------------+-----------+--------------------+----------------------+----------
 DEL45       | Delivered | 385 Terry Lane     | 3752 Renwick Drive   | OD121
 DEL46       | Waiting   | 221 Baker Street   | 4000 Uno Drive       | OD122
 DEL47       | Waiting   | 657 Ashoka Road    | 2374 Chadwick Drive  | OD123
 DEL48       | Waiting   | 887 Holt Lane      | 8775 Thomas Link     | OD124
 DEL49       | Delivered | 990 MG Road        | 7482 Baker Street    | OD125
 DEL50       | Waiting   | 887 Holt Lane      | 3785 Marine Drive    | OD126
 DEL51       | Delivered |  124 Palace Road   | 1010 Down Lane       | OD127
 DEL52       | Delivered |  345 Walter Lane   | 3000 Residency Road  | OD128
(8 rows)


food_delivery=> select * from restaurant;
ERROR:  permission denied for table restaurant
```

user -



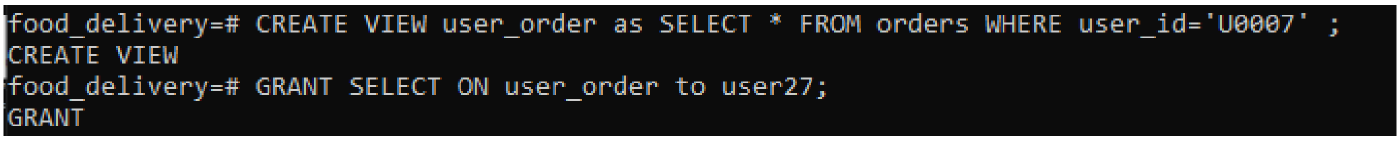
```
C:\Program Files\PostgreSQL\13\bin>psql -U user27 food_delivery
Password for user user27:
psql (13.4)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

food_delivery=> select * from orders;
 order_id | order_status | order_date | amount |    order_items     | user_id | employee_id | item_id
----------+--------------+------------+--------+--------------------+---------+-------------+---------
 OD121    | Ready        | 2021-10-01 |    350 | Chicken Lollipops  | U0001   | E0127       | IT111
 OD122    | Wait         | 2021-10-01 |    150 | Veg Burger         | U0002   | E0128       | IT115
 OD123    | Ready        | 2021-10-09 |    200 | red sauce pasta    | U0003   | E0237       | IT120
 OD124    | Wait         | 2021-10-01 |    200 | Non veg Burger     | U0004   | E7321       | IT122
 OD125    | Ready        | 2021-11-01 |    100 | French fries       | U0007   | E0225       | IT125
 OD126    | Wait         | 2021-10-01 |    200 | Pizza              | U0006   | E0002       | IT212
 OD127    | Ready        | 2021-10-01 |    200 | Manchurian         | U0010   | E0522       | IT130
 OD128    | Ready        | 2021-10-01 |    150 | Veg Burger         | U0009   | E5607       | IT115
 OD129    | Ready        | 2021-10-21 |    150 | aglio olio pasta   | U0009   | E5607       | IT216
(9 rows)


food_delivery=> truncate table orders;
ERROR:  permission denied for table orders
food_delivery=>
```

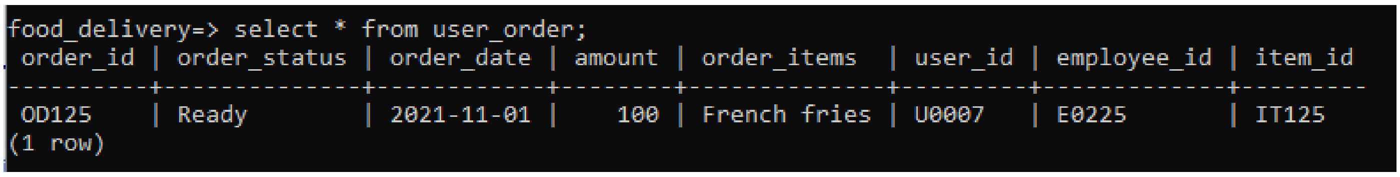
Revoking permissions -
REVOKE select on orders, payment, review, menu, delivery from user27;



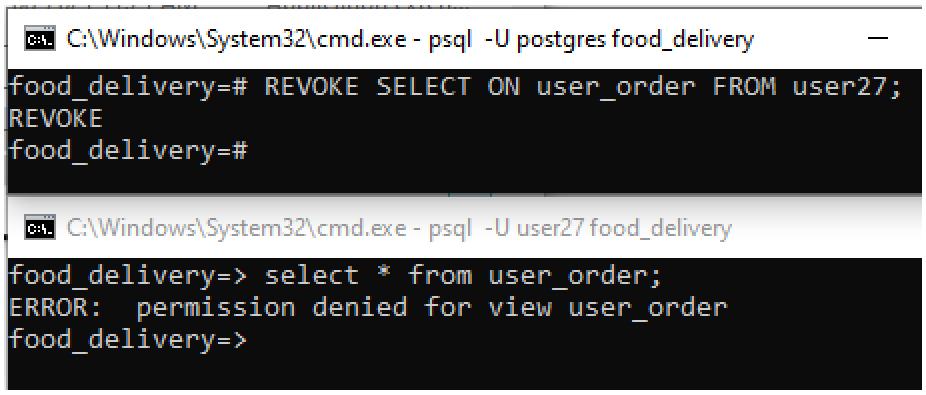
```
food_delivery=> select * from menu;
ERROR:  permission denied for table menu
food_delivery=>
```

Granting user permission to view their order details,


```
food_delivery=# CREATE VIEW user_order as SELECT * FROM orders WHERE user_id='U0007' ;
CREATE VIEW
food_delivery=# GRANT SELECT ON user_order to user27;
GRANT
```


```
food_delivery=> select * from user_order;
 order_id | order_status | order_date | amount | order_items  | user_id | employee_id | item_id
----------+--------------+------------+--------+--------------+---------+-------------+---------
 OD125    | Ready        | 2021-11-01 |    100 | French fries | U0007   | E0225       | IT125
(1 row)
```

Permission is revoked. This also demonstrates concurrency.


```
C:\Windows\System32\cmd.exe - psql  -U postgres food_delivery
food_delivery=# REVOKE SELECT ON user_order FROM user27;
REVOKE
food_delivery=#
```
```
C:\Windows\System32\cmd.exe - psql  -U user27 food_delivery
food_delivery=> select * from user_order;
ERROR:  permission denied for view user_order
food_delivery=>
```

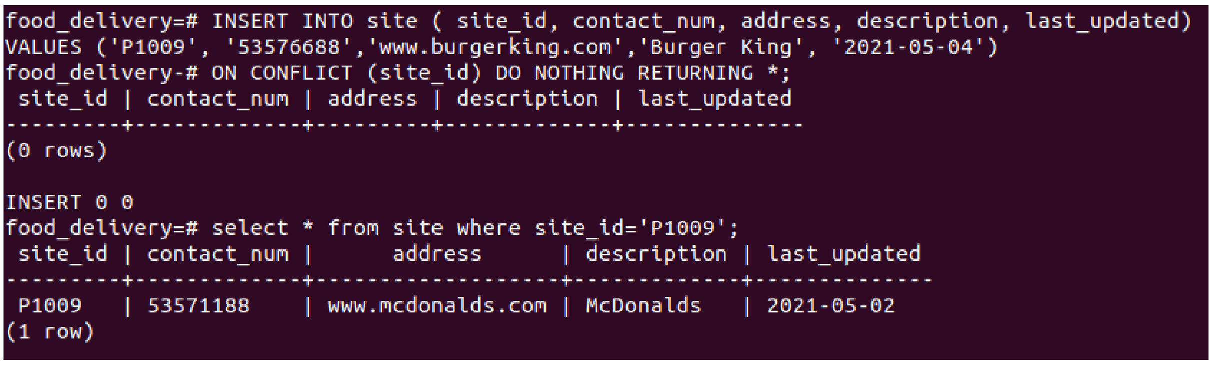
TRANSACTIONS

- In PSQL, a transaction symbolizes a unit of work within a DBMS against a database, and treated in a coherent and reliable way independent of other transactions. A transaction generally represents any change in the DB.
- Concurrency is a situation that arises in a database due to the transaction process when two or more users are trying to access the same data or information, it can lead to inconsistent results or invalid behaviour.

Conflict Resolution

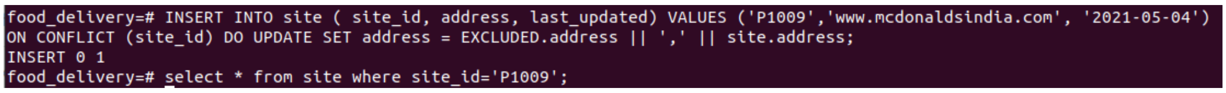
1. On trying to insert a record with a duplicate site ID (pk), no record is inserted.

   INSERT INTO site (site_id, contact_num, address, description, last_updated) VALUES ('P1009', '53576688','www.burgerking.com','Burger King', '2021-05-04')
   ON CONFLICT (site_id) DO NOTHING RETURNING *;

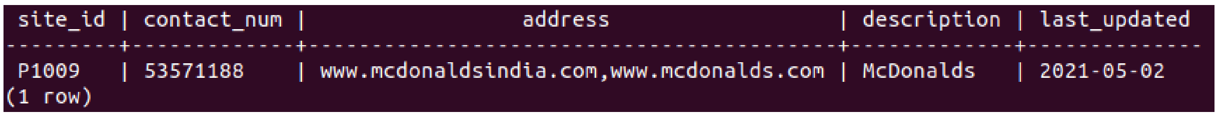

```
food_delivery=# INSERT INTO site ( site_id, contact_num, address, description, last_updated)
VALUES ('P1009', '53576688','www.burgerking.com','Burger King', '2021-05-04')
food_delivery-# ON CONFLICT (site_id) DO NOTHING RETURNING *;
 site_id | contact_num | address | description | last_updated
---------+-------------+---------+-------------+--------------
(0 rows)

INSERT 0 0
food_delivery=# select * from site where site_id='P1009';
 site_id | contact_num |      address      | description | last_updated
---------+-------------+-------------------+-------------+--------------
 P1009   | 53571188    | www.mcdonalds.com | McDonalds   | 2021-05-02
(1 row)
```

2. On inserting a record with a new email for the site, the existing record is updated.

   INSERT INTO site (site_id, address, last_updated)
   VALUES ('P1009','www.mcdonalds índia.com', '2021-05-04')
   ON CONFLICT (site_id) DO UPDATE SET address = EXCLUDED.address || ',' || site.address;



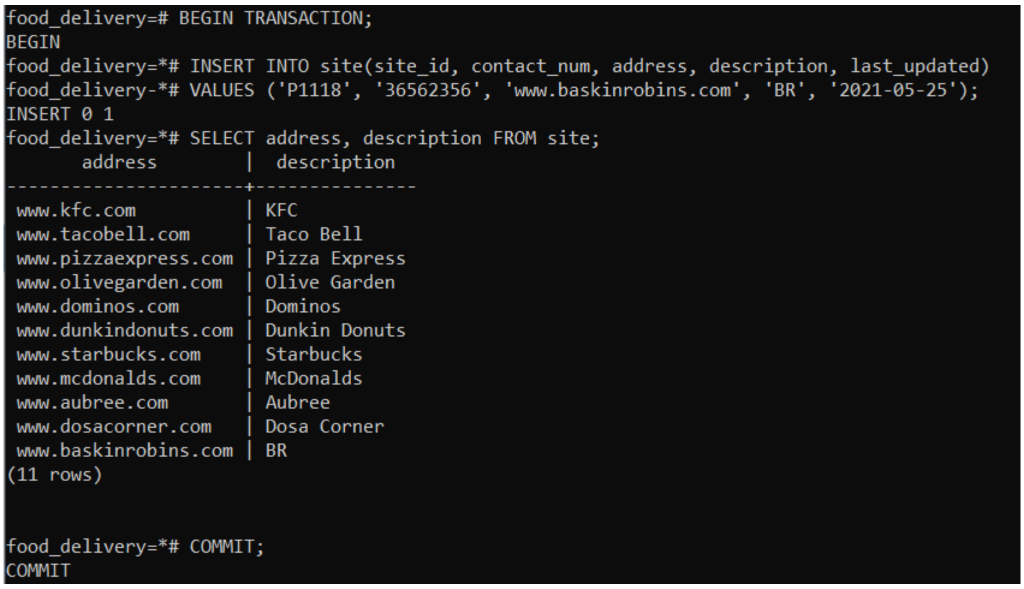
```
food_delivery=# INSERT INTO site ( site_id, address, last_updated) VALUES ('P1009','www.mcdonaldsindia.com', '2021-05-04')
ON CONFLICT (site_id) DO UPDATE SET address = EXCLUDED.address || ',' || site.address;
INSERT 0 1
food_delivery=# select * from site where site_id='P1009';
```



```
 site_id | contact_num |                   address                   | description | last_updated
---------+-------------+---------------------------------------------+-------------+--------------
 P1009   | 53571188    | www.mcdonaldsindia.com,www.mcdonalds.com | McDonalds   | 2021-05-02
(1 row)
```

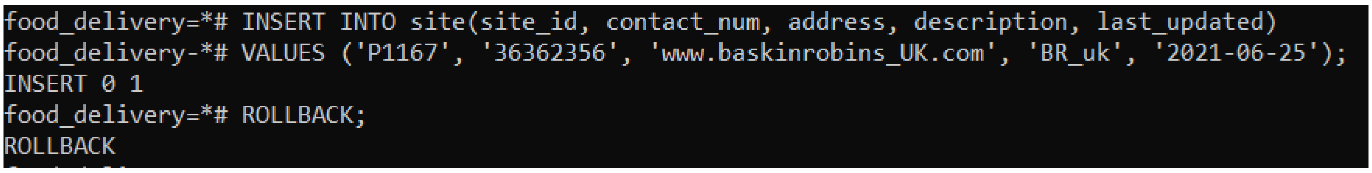
Transaction Program

3. We start a transaction, insert a record into the relation 'site' and to make the change visible to other sessions (or users) we commit the transaction by using COMMIT



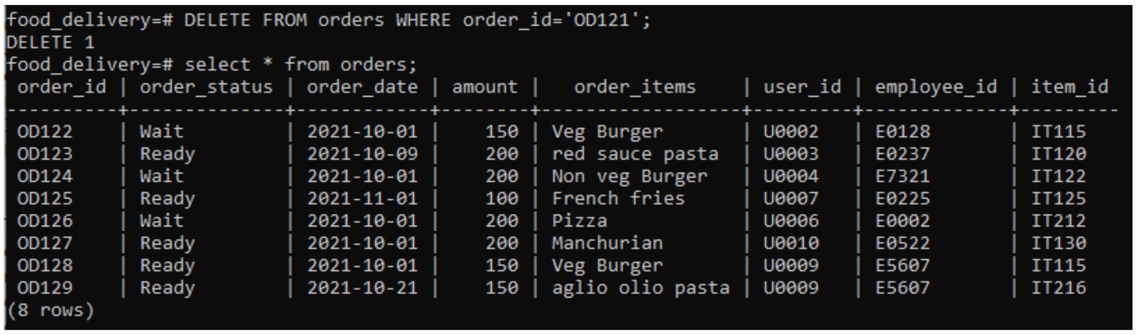
```
food_delivery=# BEGIN TRANSACTION;
BEGIN
food_delivery=*# INSERT INTO site(site_id, contact_num, address, description, last_updated)
food_delivery-*# VALUES ('P1118', '36562356', 'www.baskinrobins.com', 'BR', '2021-05-25');
INSERT 0 1
food_delivery=*# SELECT address, description FROM site;
       address        |  description
----------------------+--------------
 www.kfc.com          | KFC
 www.tacobell.com     | Taco Bell
 www.pizzaexpress.com | Pizza Express
 www.olivegarden.com  | Olive Garden
 www.dominos.com      | Dominos
 www.dunkindonuts.com | Dunkin Donuts
 www.starbucks.com    | Starbucks
 www.mcdonalds.com    | McDonalds
 www.aubree.com       | Aubree
 www.dosacorner.com   | Dosa Corner
 www.baskinrobins.com | BR
(11 rows)


food_delivery=*# COMMIT;
COMMIT
```

4. To roll back or undo the change of the current transaction

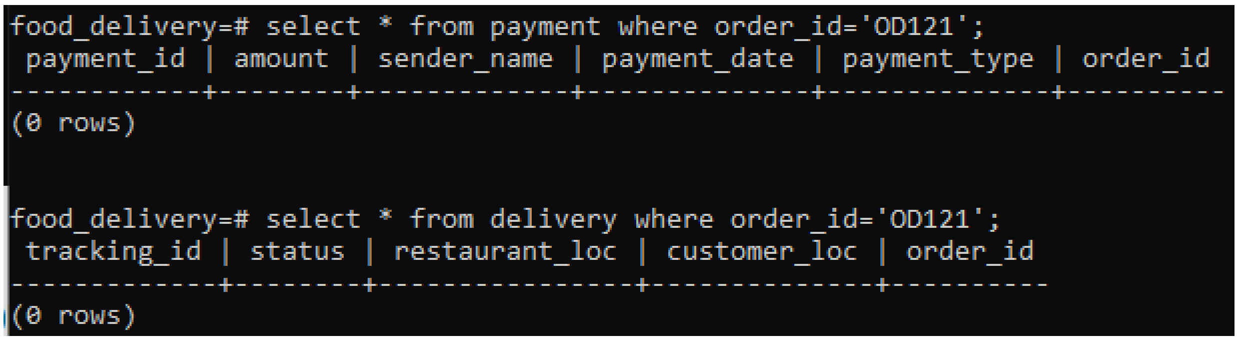


```
food_delivery=*# INSERT INTO site(site_id, contact_num, address, description, last_updated)
food_delivery-*# VALUES ('P1167', '36362356', 'www.baskinrobins_UK.com', 'BR_uk', '2021-06-25');
INSERT 0 1
food_delivery=*# ROLLBACK;
ROLLBACK
```

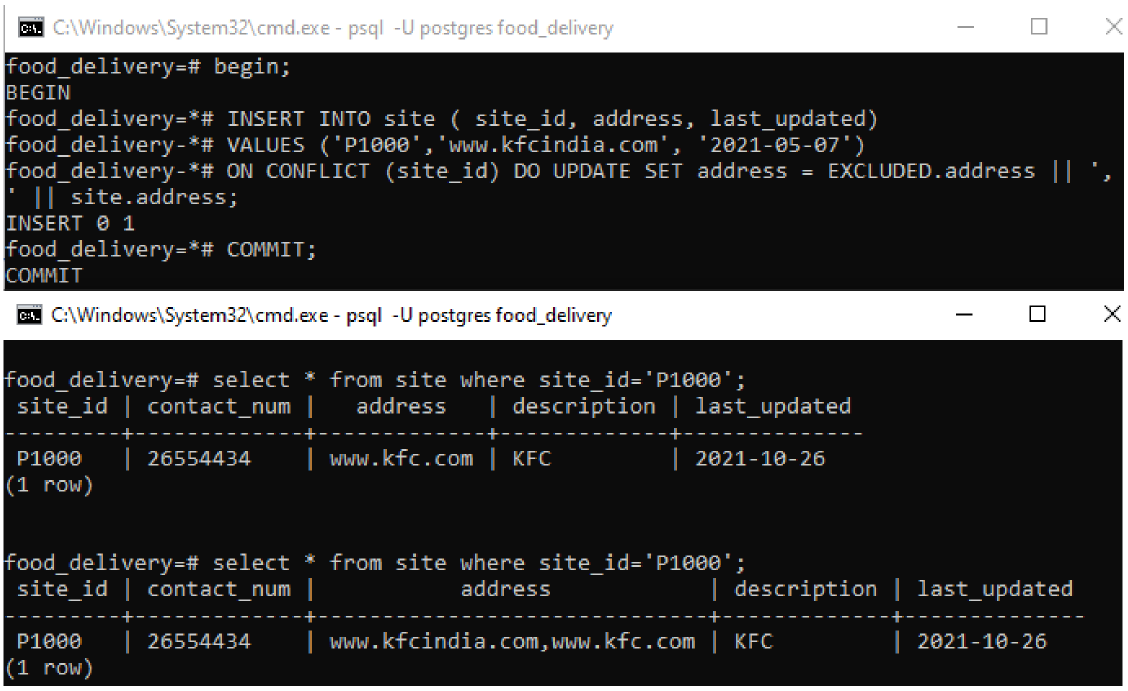
5. On deleting an order item, the relevant records are also deleted in payment and delivery tables.



```
food_delivery=# DELETE FROM orders WHERE order_id='OD121';
DELETE 1
food_delivery=# select * from orders;
 order_id | order_status | order_date | amount |    order_items     | user_id | employee_id | item_id
----------+--------------+------------+--------+--------------------+---------+-------------+---------
 OD122    | Wait         | 2021-10-01 |    150 | Veg Burger         | U0002   | E0128       | IT115
 OD123    | Ready        | 2021-10-09 |    200 | red sauce pasta    | U0003   | E0237       | IT120
 OD124    | Wait         | 2021-10-01 |    200 | Non veg Burger     | U0004   | E7321       | IT122
 OD125    | Ready        | 2021-11-01 |    100 | French fries       | U0007   | E0225       | IT125
 OD126    | Wait         | 2021-10-01 |    200 | Pizza              | U0006   | E0002       | IT212
 OD127    | Ready        | 2021-10-01 |    200 | Manchurian         | U0010   | E0522       | IT130
 OD128    | Ready        | 2021-10-01 |    150 | Veg Burger         | U0009   | E5607       | IT115
 OD129    | Ready        | 2021-10-21 |    150 | aglio olio pasta   | U0009   | E5607       | IT216
(8 rows)
```


```
food_delivery=# select * from payment where order_id='OD121';
 payment_id | amount | sender_name | payment_date | payment_type | order_id
------------+--------+-------------+--------------+--------------+----------
(0 rows)


food_delivery=# select * from delivery where order_id='OD121';
 tracking_id | status | restaurant_loc | customer_loc | order_id
-------------+--------+----------------+--------------+----------
(0 rows)
```

6. Concurrency control: Changes are not reflected until the transaction is committed.


```
C:\Windows\System32\cmd.exe - psql  -U postgres food_delivery
food_delivery=# begin;
BEGIN
food_delivery=*# INSERT INTO site ( site_id, address, last_updated)
food_delivery-*# VALUES ('P1000','www.kfcindia.com', '2021-05-07')
food_delivery-*# ON CONFLICT (site_id) DO UPDATE SET address = EXCLUDED.address || ',
' || site.address;
INSERT 0 1
food_delivery=*# COMMIT;
COMMIT
```

```
C:\Windows\System32\cmd.exe - psql  -U postgres food_delivery

food_delivery=# select * from site where site_id='P1000';
 site_id | contact_num |   address   | description | last_updated
---------+-------------+-------------+-------------+-------------
 P1000   | 26554434    | www.kfc.com | KFC         | 2021-10-26
(1 row)


food_delivery=# select * from site where site_id='P1000';
 site_id | contact_num |           address            | description | last_updated
---------+-------------+------------------------------+-------------+--------------
 P1000   | 26554434    | www.kfcindia.com,www.kfc.com | KFC         | 2021-10-26
(1 row)
```

CONTRIBUTIONS

1. Sayonika Das - PES1UG19CS439
   Queries, Transactions 2 hours

2. Sejal Maurya - PES1UG19CS440
   Queries, Multiple Users 2 hours

3. Sheetal S - PES1UG19CS455
   Queries, Performance Analysis 2 hours