CSE 560 Data Models and Query Language

Milestone II

Project name: E-Commerce Management System

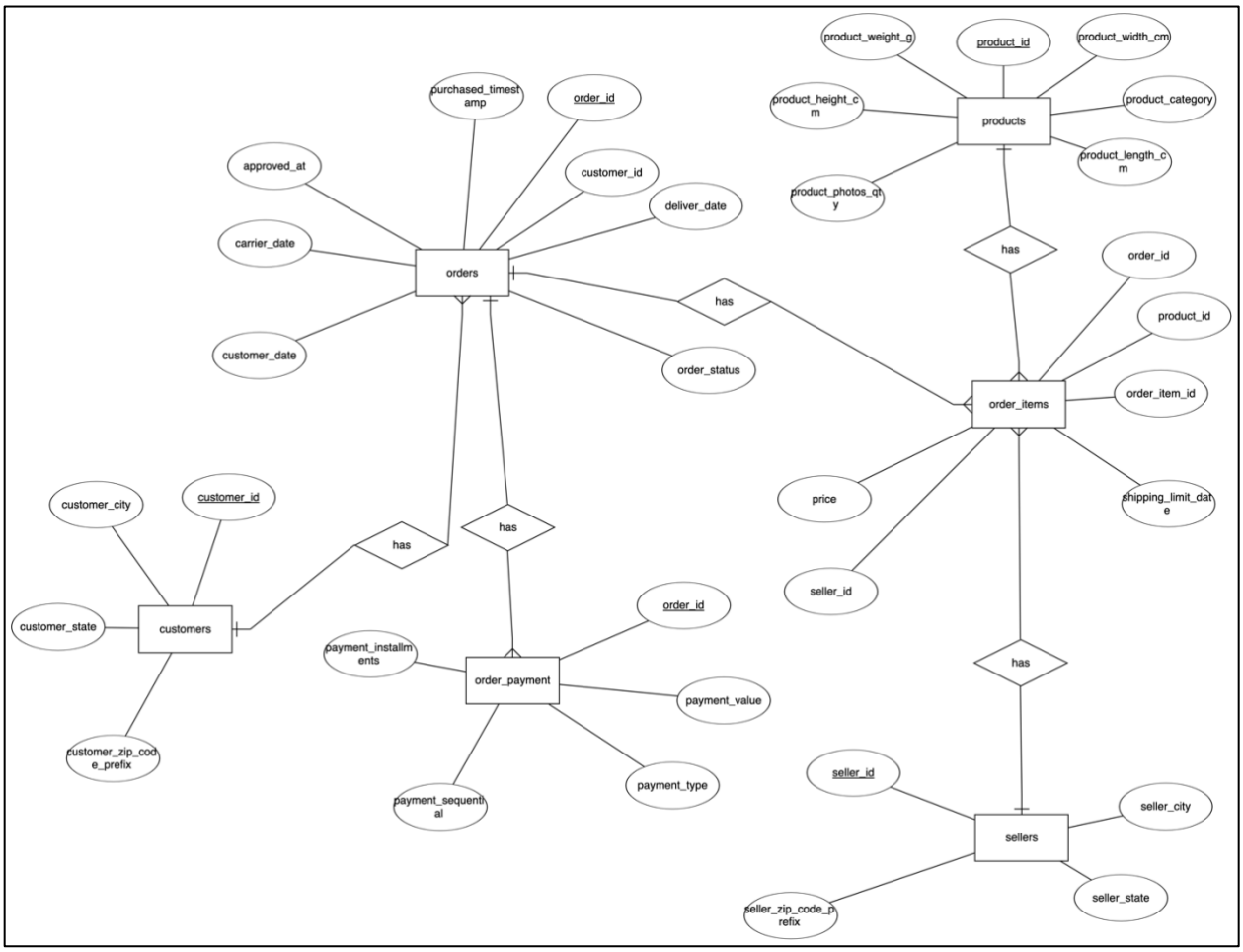
Group name	OnePiece

Member name	**UB id**
Bhargav Anudeep Koripalli	bkoripal
Suchitra Krishnakumar Nair	snair8
Vishnuvardhan Reddy Kaithapuram	vkaithap

Problem Statement:
E-commerce is all about the combination of, a plethora of products available in the market and the customers who purchase them. It would be difficult to manage the information about these colossal market data just using excel. Such a humongous amount of data needs a defined and structured storing approach, to keep a track of various market activities that would be happening on a day-to-day basis. This can be done using database systems, as it is efficient enough to store the data, manipulate them as per needs and can be used for analyzing the market trends as per needs.

Target User:
Any retail business involved in commercial activity of buying/selling products.

ER Diagram:



<u>Description of relations and their attributes</u>:

1. **orders**:

   → This relation comprises of all the information associated with the orders made by the customers and their status.

   → <u>Attributes are as follows</u>:

   - **order_id:** A unique id assigned to an order made by the customer. It is referred as the **<u>primary key</u>** for the "orders" relation while it is the "**<u>foreign key</u>**" for "order_items" relation. Datatype for this attribute is "**varchar[100]**". It is set to "**NOT NULL**".
   - **customer_id:** A unique id assigned to the respective customer who made the order. It is referred as the **<u>foreign key</u>** for this relation, while it is the **<u>primary key</u>** for the relation "customers". Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL"**.
   - **order_estimated_deliver_date:** This attribute comprises of the date on which the item is to be delivered or has been delivered. Datatype for this attribute is **"datetime"**. It is set to **"NOT NULL"**.
   - **order_status:** This attribute indicates the current status of the order, whether it is delivered, or yet to be delivered. Datatype for this attribute is **"varchar[20]"**. It is set to **"NOT NULL"**.
   - **order_delivery_customer_date:** This attribute comprises of the date on which the customer placed the order. Datatype for this attribute is **"datetime"**. It is set to **"NOT NULL".**
   - **order_delivery_carrier_date:** It comprises of the timestamp, on which the order was handed over to the respective logistic partner. Datatype for this attribute is **"datetime"**. It is set to **"NOT NULL"**.
   - **order_approved_at:** This attribute comprises of the information of the date on which the order placed by the customer was approved on. Datatype for this attribute is **"datetime"**. It is set to **"NOT NULL"**.
   - **order_purchased_timestamp:** This attribute comprises of the exact date and time on which the monetary transaction of the ordered item was completed. Datatype for this attribute is **"datetime".** It is set to **"NOT NULL"**.

2. **customers**:
   → This relation stores all the information of the customers.
   → Attributes are as follows:
   - **customer_id:** A unique id assigned to the customer, in order to identify them. This attribute is the **"primary key"** for the **"customers"** relation, while a **"foreign key"** for **"orders"** relation. Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL".**
   - **customer_unique_id:** A unique id assigned to a customer at the time of purchase. These are subject to change every time a customer makes a new order. Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL".**
   - **customer_zip_code_prefix:** This attribute stores the zip code of the location from the customer placed and order. Datatype for this attribute is **"int"**. It is set to **"NOT NULL"**.

3. **customer_addr:**
   → This relation stores all the information of the customers.
   → Attributes are as follows:
   - **customer_zip_code_prefix:** This attribute stores the zip code of the location from the customer placed and order. This attribute is the **"primary key"** for the **"customer_addr"** relation, while a **"foreign key"** for **"customers"** relation. Datatype for this attribute is **"int"**. It is set to **"NOT NULL"**.
   - **customer_city:** It comprises of the information of the city where the customer resides/the city from where the customer placed an order. Datatype for this attribute is **"varchar[50]"**. It is set to **"NOT NULL"**.
   - **Customer_state:** It comprises of the information of the state where the customer resides/the state from where the customer placed an order. Datatype for this attribute is **"varchar[50]"**. It is set to **"NOT NULL"**.

4. **order_pays**:
   → This relation stores all information regarding the payment of the order.
   → Attributes associated to this relation are as follows:
   - **payment_value:** It describes the transaction value. Datatype for this attribute is **"float"**. It is set to **"NOT NULL"**.
   - **product_id**: Unique identification number assigned to every different product. This is the **"foreign key"** for this relation, but the **"primary key"** for the relation "products". Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL"**.
   - **payment_type:** It stores the information about the mode of payment chosen by the customer. Datatype for this attribute is **"varchar[50]"**. It is set to **"NOT NULL"**.
   - **payment_sequential:** A customer has the liberty to pay for an order in more than one payment mode. On such multiple payment modes, a sequence is assigned to all the payments done. Datatype for this attribute is **"int"**. It is set to **"NOT NULL"**.
   - **payment_installments:** The number of installments taken by the customer to complete the payment for the product purchased. Datatype for this attribute is **"int"**. It is set to **"NOT NULL"**.

5. **order_item_logistic**:
   → This relation stores all information about the ordered items' overall as well as all the logistic information.
   → Attributes associated to this relation are as follows:
   - **order_item_id:** An order may contain single or multiple items in it. On the occasion of multiple items, sequence numbers are assigned to multiple items within the same order having one order_id. That sequence number is stored in this attribute. Datatype for this attribute is **"int"**. It is set to **"NOT NULL"**.
   - **shipping_limit_date:** Shows the date by which the order will be handed over to the logistic partner from the seller. Datatype for this attribute is **"datetime"**. It is set to **"NOT NULL"**.
   - **order_id:** This is a unique identification number given to every distinct order made. This attribute is the **"foreign key"** for this

relation, but the **"<u>primary key</u>"** for the relation "orders". Datatype for this attribute is **"varchar[100]".** It is set to **"NOT NULL"**.

- **product_id:** Unique identification number assigned to every different product. This is the **"<u>foreign key</u>"** for this relation, but the **"<u>primary key</u>"** for the relation "products". Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL"**.
- **freight_value**: A freight_value (historically and in ship chartering simply freight) is a price at which a certain cargo is delivered from one point to another. Datatype for this attribute is **"float"**. It is set to **"NOT NULL".**

6. **order_items_info:**
   → This relation stores all information about the ordered items' seller, its price and the product id.
   → <u>Attributes associated to this relation are as follows:</u>
   - **seller_id:** This attribute stores the unique identification number of every seller. Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL"**.
   - **price:** Price of the product. Datatype for this attribute is **"float"**. It is set to **"NOT NULL"**.
   - **product_id:** Unique identification number assigned to every different product. Datatype for this attribute id **Varchar[100]** it is set to **"NOT NULL".** This is the **Primary key** for this relation.

7. **products**:
   →This relation stores all the information of products.
   →<u>Attributes associated to this relation are as follows:</u>
   - **product_id:** Unique identification number assigned to every different product. This is the **"<u>primary key</u>"** for this relation, but the **"<u>foreign key</u>"** for the relation "order_item_logistic", "order_pays". Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL"**.
   - **product_width_cm:** Stores information about the width of the product. Datatype for this attribute is **"float"**. It is set to **"NOT NULL"**.

- **product_height_cm:** Stores information about the height of the product. Datatype for this attribute is **"float"**. It is set to **"NOT NULL"**.
- **product_weight_cm:** Stores information about the weight of the product. Datatype for this attribute is **"float"**. It is set to **"NOT NULL".**
- **product_length_cm:** Stores information about the length of the product. Datatype for this attribute is **"float"**. It is set to **"NOT NULL"**.
- **product_category:** Information about the category in which the product belongs to. Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL"**.
- **product_phots_qty:** Number of times the product photo was published. Datatype for this attribute is **"int"**. It is set to **"NOT NULL"**.

8. **sellers**:
   →This relation stores all the information about the sellers.
   →<u>Attributes associated to this relation are as follows:</u>
   - **seller_id:** Unique identification number given to individual sellers. This is the **"<u>primary key</u>"** for this relation, while the **"foreign key"** for "order_items". Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL"**.
   - **seller_zip_code_prefix:** Information about the seller's zip code is stored in this attribute. Datatype for this attribute is **"int"**. It is set to **"NOT NULL".**

9. **sellers_addr:**
   → This relation stores all the information about the sellers.
   → Attributes associated to this relation are as follows:
   - **seller_city:** Information about the seller's city is stored in this attribute. Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL".**
   - **seller_state:** Information about the seller's state is stored in this attribute. Datatype for this attribute is **"varchar[100]"**. It is set to **"NOT NULL".**
   - **seller_zip_code_prefix:** Information about the seller's zip code is stored in this attribute. This is the **"primary key"** for this relation. Datatype for this attribute is **"int"**. It is set to **"NOT NULL".**

Description of relations between tables:

→ The relations between the tables can be explained using the ER-diagram as shown above:

- The tables "**orders"**, "**product"** and "**sellers"** has **one to many** relations with table "**order_items**".
- The table "**orders**" has one to many relation with table "**order_payment**".
- The table "**customers**" has one to many relations with the table "**orders**".

Explanation of the actions taken on a foreign key when a primary key (which is the foreign key in this context) is deleted.

→When a data from table A having the primary key (which is the foreign key to table B) is deleted, then there are 4 actions that can be performed on the same data in table B. This can be handled using the 4 options available with "ON DELETE CASCADE".

- NO ACTION:
  → In this case, no action is performed on the data tuple of table B when, the corresponding data tuple present Table A having primary key is deleted.

- DELETE CASCADE:
  → In this, the data in table B is either deleted when the corresponding data from Table A having primary key is deleted.

- SET NULL:
  →The data present in Table B with foreign key is set to NULL, when the data present in Table A having primary key is deleted.

- SET DEFAULT:
  → The data present in Table B with foreign key is set to their default values, when the data present in Table A having primary key is deleted

Schemas in the database and BCNF improvements:

1. **Customer Table Relation and its associated Functional dependencies:**
   - Creation of Customer Table:
     CREATE TABLE Customer(Customer_ID VARCHAR(100), Customer_unique_ID VARCHAR(100), Customer_Zipcode_prefix VARCHAR(100), Customer_City VARCHAR(100), Customer_State VARCHAR(100));
   - Functional dependencies:
     i) Customer_ID → Customer_unique_ID, Customer_Zipcode_prefix, Customer_City, Customer_State
     ii) Customer_City , Customer_State → Customer_Zipcode_prefix
     → In the above Functional Dependency, the left side attributes Customer_City, Customer_state is not a primary key for the relation and hence the relation is not in BCNF.
     → Hence we split the table into **Customer** and **Customer_addr**
     - Creation of new Customer Table:
       CREATE TABLE Customer(Customer_ID VARCHAR(100), Customer_unique_ID VARCHAR(100), Customer_Zipcode_prefix VARCHAR(100));

     - Functional dependencies based on the new Customer table:
       Customer_ID→Customer_unique_ID,Customer_Zipcode_prefix

→ In above FD, the left side attribute is a primary key for the relation and hence the relation is in BCNF.

- <u>Creation of new Customer_addr Table:</u>
  CREATE TABLE Customer_addr(Customer_City VARCHAR(100), Customer_Zipcode_prefix VARCHAR(100), Customer_State VARCHAR(100)
  Primary key(Customer_Zipcode_prefix));

- <u>Functional dependencies based on the new Customer_addr table:</u>
  Customer_Zipcode_prefix → Customer_State, Customer_City

  → In above FD, the left side attribute is a primary key for the relation and hence the relation is in BCNF.

2. **<u>Order_items Table Relation and its associated Functional dependencies:</u>**
   - <u>Creation of order_items Table:</u>
     Create table order_items (order_ID VARCHAR(100), order_item_ID VARCHAR(100), product_ID VARCHAR(100), seller_ID VARCHAR(100), Shipping_Limit_data VARCHAR(100), price VARCHAR(100), freight_value VARCHAR(100));
   - <u>Functional dependencies:</u>
     i) order__ID → order_item_ID, product_ID, Seller_ID, Shipping_Limit_data, Price, Freight_value
     ii) product_ID → seller_id
     iii) product_ID → price

     → In above FD, the left side attribute product_ID is a not primary key for the relation and hence the relation is not in BCNF so we divided the tables in two tables **order_item_logistic** and **order_items_info**
     - <u>Creation of new order_item_logistic Table:</u>
       Create table order_item_logistic (order_ID VARCHAR(100), order_item_ID VARCHAR(100), product_ID VARCHAR(100), Shipping_Limit_data VARCHAR(100), freight_value VARCHAR(100));

- <u>Functional dependencies:</u>
  order__ID → order_item_ID, product_ID, Shipping_Limit_data, Freight_value
  → In above FD, the left side attribute is a primary key for the relation and hence the relation is in BCNF.

- <u>Creation of new order_items_info Table:</u>
  Create table order_items_info (product_ID VARCHAR(100), Seller_ID VARCHAR(100), price VARCHAR(100)
  primary key(product_ID)

- <u>Functional dependencies:</u>
  product_ID->Seller_ID,price
  → In above FD, the left side attribute is a primary key for the relation and hence the relation is in BCNF

3. **<u>Products Table Relation and its associated Functional dependencies:</u>**
   - <u>Creation of Products Table:</u>
     CREATE TABLE products(product_ID VARCHAR(100), product_category_name VARCHAR(100), product_name_length VARCHAR(100), product_description_length VARCHAR(100), product_photos_qty VARCHAR(100), product_weight_g VARCHAR(100), product_length_cm VARCHAR(100), product_height_cm VARCHAR(100), product_width_cm VARCHAR(100));

   - <u>Functional dependencies:</u>
     product_ID → product_category_name, product_name_length, Product_description_length, product_photos_qty, Product_weight_g, product_length_cm, Product_height_cm, product_width_cm
     → In the above FD, the left side attribute is a primary key for the relation and hence the relation is in BCNF

4. **<u>order_pays Relation and its associated Functional dependencies:</u>**
   - <u>Creation of order_pays table:</u>

Create table order_pays(product_ID VARCHAR(1000), payment_sequential VARCHAR(1000), payment_type VARCHAR(1000), payment_installments VARCHAR(1000), payment_value VARCHAR(1000));

- <u>Functional dependencies:</u>

  product_ID → payment_sequential, payment_type, payment_installments, payment_value

  → In above FD, the left side attribute is a primary key for the relation And hence the relation is in BCNF.

5. **<u>Orders relation and its associated Functional Dependency:</u>**

- <u>Creation of orders table:</u>

  Create table orders(order_id VARCHAR(100), customer_id VARCHAR(100), order_status VARCHAR(100), order_purchase_timestamp VARCHAR(100), order_approved_at VARCHAR(100), order_delivered_carrier_date VARCHAR(100), order_delivered_customer_date VARCHAR(100), order_estimated_delivery_date VARCHAR(100));

- <u>Functional dependencies:</u>

  Order_id → customer_id, order_status, order_purchase_timestamp, order_approved_at, order_delivered_carrier_date, order_delivered_customer_date, Order_estimated_delivery_date

  → In above FD, the left side attribute is a primary key for the relation and hence the relation is in BCNF.

6. **<u>Sellers relation and its associated Functional Dependency:</u>**

- <u>Creation of sellers table:</u>

  Create table sellers(seller_id VARCHAR(100), seller_zip_code_prefix VARCHAR(100), seller_city VARCHAR(100), seller_state VARCHAR(100));

- <u>Functional dependencies:</u>

  i) seller_id → seller_zip_code_prefix, seller_city, seller_state

  ii) seller_city, seller_state->seller_zip_code_prefix

→ In above FD, the left side attribute is not a primary key for the relation and hence the relation not in BCNF hence this table was decomposed into two tables: "**sellers**" and "**sellers_addr**"

- <u>Creation of new sellers Table:</u>

  Create table sellers(seller_id VARCHAR(100),
  seller_zip_code_prefix VARCHAR(100)
  primary key(seller_id));

- <u>Functional dependency:</u>

  Seller_id → seller_zip_code_prefix
  → In above FD, the left side attribute is a primary key for the relation and hence the relation is in BCNF

- <u>Creation of new sellers_addr Table:</u>

  Create table sellers_addr(seller_zip_code_prefix
  VARCHAR(100), seller_city VARCHAR(100),
  seller_state VARCHAR(100),
  primary key(seller_zip_code_prefix));

- <u>Functional dependency:</u>

  Seller_zip_code_prefix → seller_state, seller_city
  → In the above FD, the left side attribute is a primary key for the relation and hence the relation is in BCNF.

Queries:

1. **Total Count of Customers from each state:**

   Query:

   Select customer_state,count(customer__id) from customer
   group by customer_state order by count(customer__id) DESC LIMIT 20;

   Output:

vishnu/postgres@PostgreSQL 14

Query Editor | Query History

```
Select customer_state,count(customer__id) from customer
group by customer_state order by count(customer__id) DESC LIMIT 20;
```

Data Output | Explain | Messages | Notifications

	customer_state character varying (100)	count bigint
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020
11	PE	1652
12	CE	1336

2. **TOP 10 shippings on particular date**

Query:

```
SELECT  date(order_purchase_timestamp),
count(order_id) AS NO_OF_Orders  FROM orders
where order_status='shipped'
GROUP BY date(order_purchase_timestamp)
order by count(order_id) DESC LIMIT 10;
```

Output:

vishnu/postgres@PostgreSQL 14

Query Editor Query History

```
SELECT  date(order_purchase_timestamp),count(order_id) AS NO_OF_Orders  FROM orders
where order_status='shipped'
GROUP BY date(order_purchase_timestamp)
order by count(order_id) DESC LIMIT 10;
```

Data Output Explain Messages Notifications

	date date	no_of_orders bigint
1	2017-11-24	12
2	2018-04-11	9
3	2018-02-28	8
4	2018-03-15	8
5	2018-04-06	7
6	2018-05-14	7
7	2018-03-23	7
8	2018-03-19	7
9	2018-03-21	7
10	2018-03-04	7

3. **Seller having the max price of the product**

Query:

Select seller_id, price from order_items where price = (Select max(price) from order_items);

Output:

vishnu/postgres@PostgreSQL 14

Query Editor | Query History

```
Select seller_id,price from order_items where price = (Select max(price) from order_items);
```

Data Output | Explain | Messages | Notifications

	seller_id character varying (100)	price character varying (100)
1	610f72e407cdd7caaa2f8167b0163fd8	999.99
2	7e93a43ef30c4f03f38b393420bc753a	999.99
3	7e93a43ef30c4f03f38b393420bc753a	999.99
4	87b740daf17b5d1be335a64164ec6842	999.99
5	87b740daf17b5d1be335a64164ec6842	999.99
6	f9244d45189d3a3605499abddeade7d5	999.99
7	f9244d45189d3a3605499abddeade7d5	999.99
8	87b740daf17b5d1be335a64164ec6842	999.99
9	87b740daf17b5d1be335a64164ec6842	999.99
10	966cb4760537b1404caedd472cc610a5	999.99
11	f9244d45189d3a3605499abddeade7d5	999.99
12	87b740daf17b5d1be335a64164ec6842	999.99

4. **Number of quantities per product:**

Query:

```
SELECT a.product_category_name AS product_CAT_Name,
COUNT(b.product_category_name) AS quantity
FROM products AS a INNER JOIN products AS b
ON a.product_category_name = b.product_category_name
GROUP BY product_CAT_Name ORDER BY quantity  limit 5;
```

Output:

vishnu/postgres@PostgreSQL 14

Query Editor | Query History

```
SELECT a.product_category_name AS product_CAT_Name,
COUNT(b.product_category_name) AS quantity
FROM products AS a
INNER JOIN products AS b
ON a.product_category_name = b.product_category_name
GROUP BY product_CAT_Name ORDER BY quantity  limit 5
```

Data Output | Explain | Messages | Notifications

	product_cat_name character varying (100)	quantity bigint
1	cds_dvds_musicais	1
2	seguros_e_servicos	4
3	pc_gamer	9
4	casa_conforto_2	25
5	fashion_roupa_infanto_juvenil	25

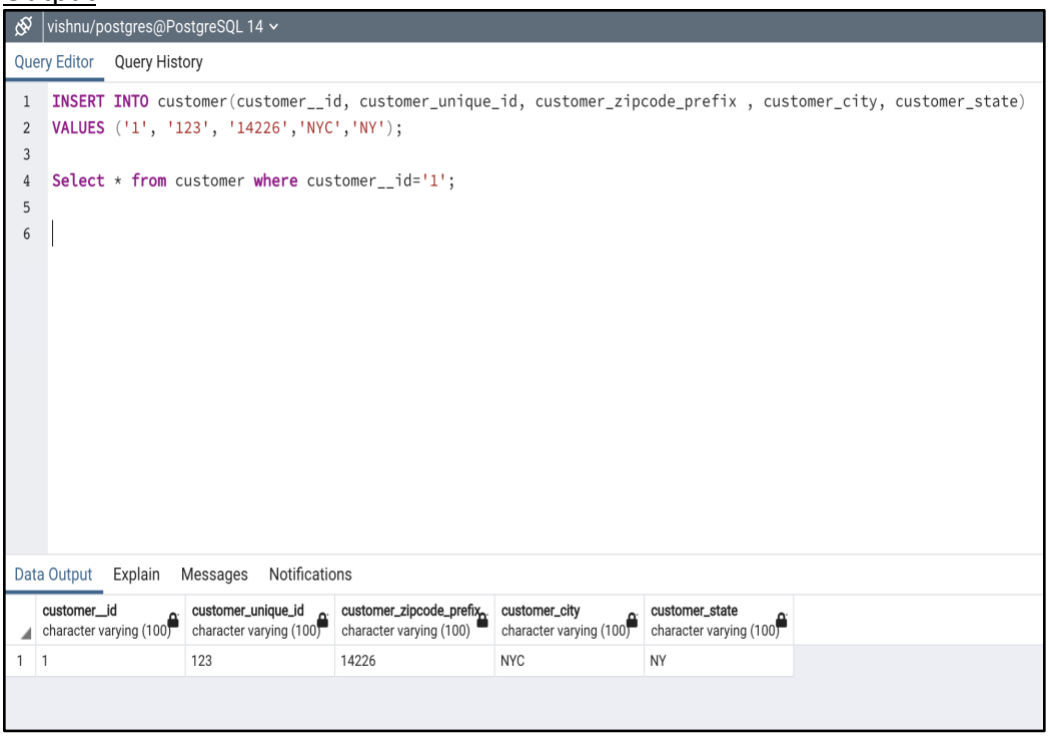
5. Insertion into customer table :

Query:

INSERT INTO customer(customer__id, customer_unique_id, customer_zipcode_prefix , customer_city, customer_state)
VALUES ('1', '123', '14226','NYC','NY');

Select * from customer where customer__id='1';

Output:



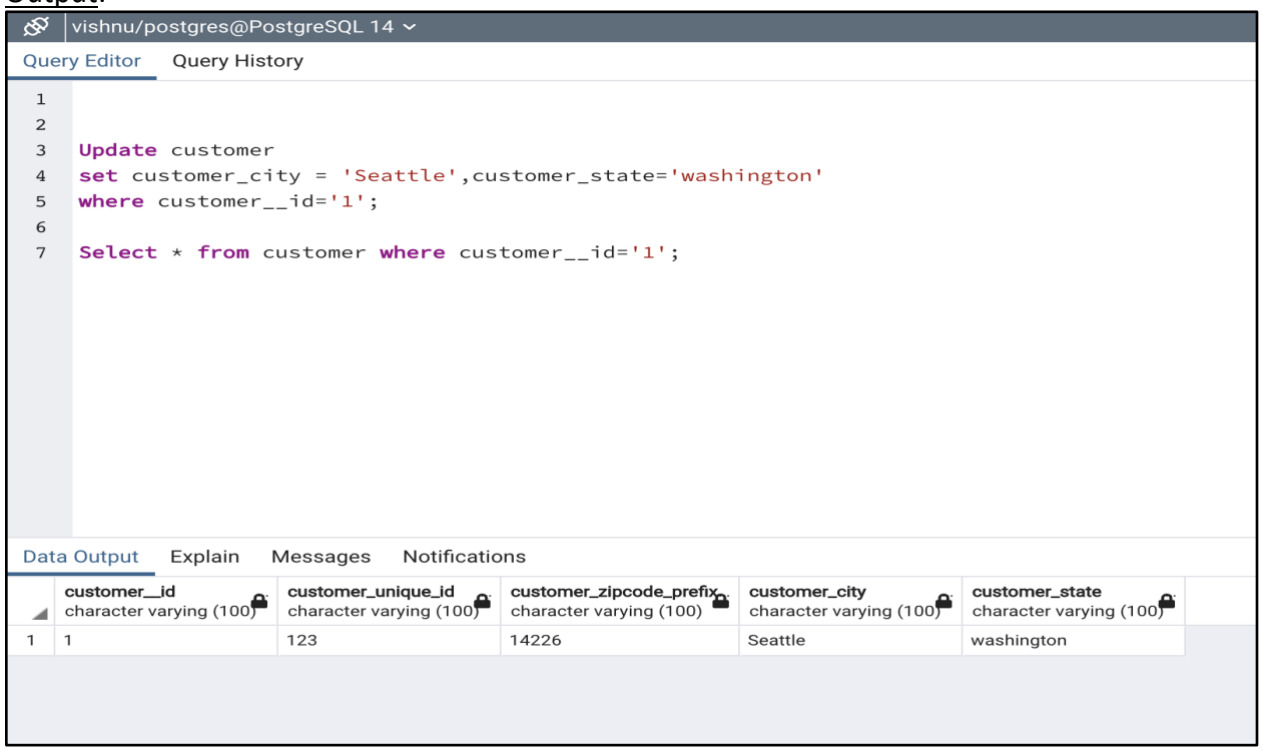
6. **Updated the customer table attributes:**

Query:

```
Update customer
set customer_city = 'Seattle',customer_state='washington'
where customer__id='1';

Select * from customer where customer__id='1';
```

Output:



7. Indexing:

Query:

Date	Rows Affected	Duration
5/5/2022 5:17:48 PM	20	137 msec

Copy | Copy to Query Editor

```
Select count(customer__id),customer_state from customer grou
```

Messages

```
Successfully run. Total query runtime: 137 msec.
20 rows affected.
```

Date	Rows Affected	Duration
5/5/2022 6:22:39 PM	20	727 msec

Copy | Copy to Query Editor

```
CREATE UNIQUE INDEX PK
ON customer (customer__id);

Select customer_state,count(customer__id) from customer
group by customer_state order by count(customer__id) DESC LI
```

Messages

```
Successfully run. Total query runtime: 727 msec.
20 rows affected.
```

8. **Trigger**:

```
CREATE OR REPLACE FUNCTION rec_insert()
        RETURN trigger AS
$$
BEGIN
        INSERT INTO sellers(seller_id,seller_zipcode_prefix,seller_city,seller_state)
        VALUES('vishnuk','14226','NYC','NY');
    RETURN NEW;
END;
$$

LANGUAGE 'plpgsql'

CREATE TRIGGER ins_same_rec
  AFTER INSERT
ON sellers
FOR EACH ROW
EXECUTE PROCEDURE rec_insert();
```

Data Output | Explain | Messages | Notifications

	seller_id character varying (100)	seller_zip_code_prefix character varying (100)	seller_city character varying (100)	seller_state character varying (100)
1	14	14226	NYC	NY
2	vishnuk	14226	NYC	NY

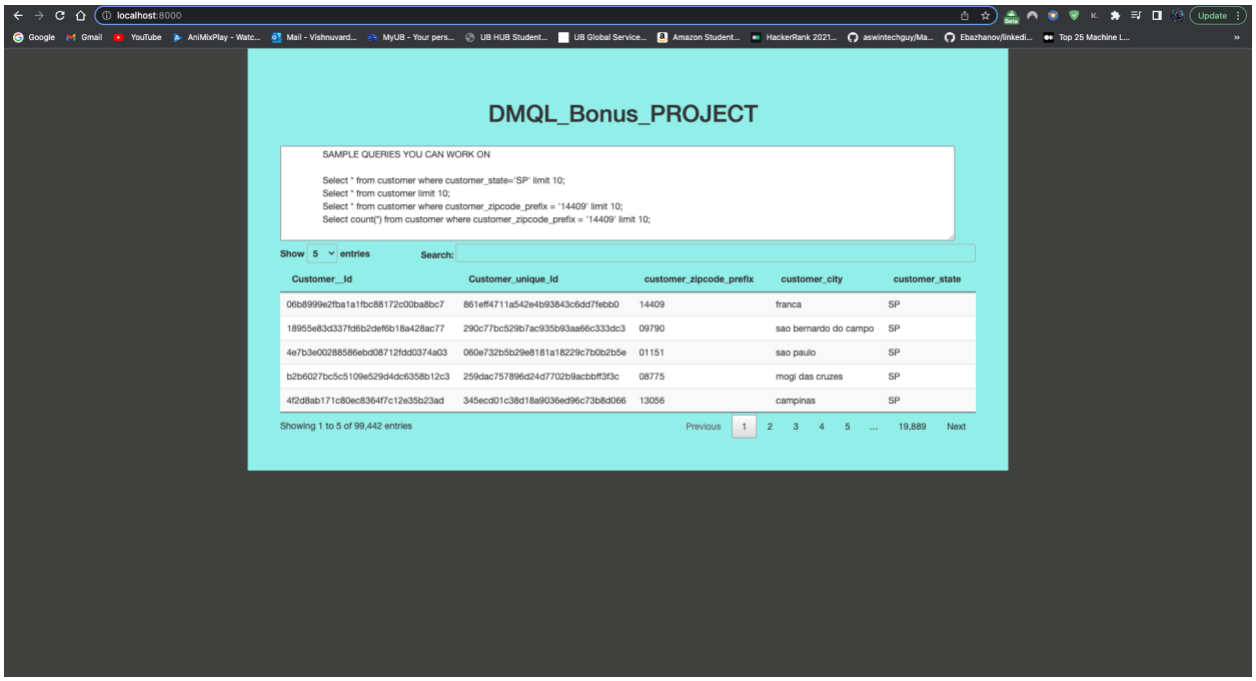
<u>Web Application</u>:

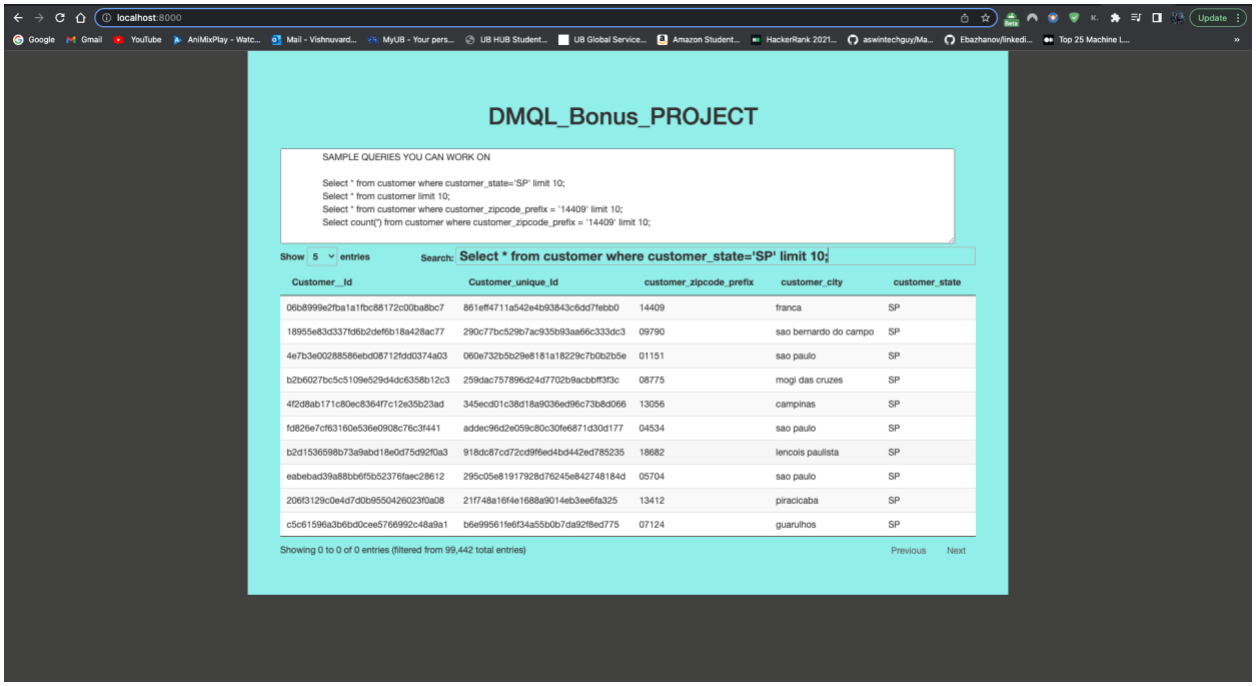
i) We created our Web Application with the help of technologies such as: Flask, HTML, CSS, PostgreSQL, AJAX.

ii) This application features, dropdowns, which retrieves information of rows to be dislayed on the front end.

iii) Initially our search bar searches for “customer_id”, but if given any query, it executes the same and give desired results using API calls.

iv) Search bar works for both DDL and DML queries.


localhost:8000
Update
Google Gmail YouTube AniMixPlay - Watc... Mail - Vishnuvard... MyUB - Your pers... UB HUB Student... UB Global Service... Amazon Student... HackerRank 2021... aswintechguy/Ma... Ebazhanov/linkedi... Top 25 Machine L...
DMQL_Bonus_PROJECT
SAMPLE QUERIES YOU CAN WORK ON
Select * from customer where customer_state='SP' limit 10;
Select * from customer limit 10;
Select * from customer where customer_zipcode_prefix = '14409' limit 10;
Select count(*) from customer where customer_zipcode_prefix = '14409' limit 10;
Show 5 entries
Search:
Customer__Id
Customer_unique_Id
customer_zipcode_prefix
customer_city
customer_state
06b8999e2fba1a1fbc88172c00ba8bc7 861eff4711a542e4b93843c6dd7febb0 14409 franca SP
18955e83d337fd6b2def6b18a428ac77 290c77bc529b7ac935b93aa66c333dc3 09790 sao bernardo do campo SP
4e7b3e00288586ebd08712fdd0374a03 060e732b5b29e8181a18229c7b0b2b5e 01151 sao paulo SP
b2b6027bc5c5109e529d4dc6358b12c3 259dac757896d24d7702b9acbbff3f3c 08775 mogi das cruzes SP
4f2d8ab171c80ec8364f7c12e35b23ad 345ecd01c38d18a9036ed96c73b8d066 13056 campinas SP
Showing 1 to 5 of 99,442 entries
Previous 1 2 3 4 5 ... 19,889 Next


localhost:8000
Update
Google Gmail YouTube AniMixPlay - Watc... Mail - Vishnuvard... MyUB - Your pers... UB HUB Student... UB Global Service... Amazon Student... HackerRank 2021... aswintechguy/Ma... Ebazhanov/linkedi... Top 25 Machine L...
DMQL_Bonus_PROJECT
SAMPLE QUERIES YOU CAN WORK ON
Select * from customer where customer_state='SP' limit 10;
Select * from customer limit 10;
Select * from customer where customer_zipcode_prefix = '14409' limit 10;
Select count(*) from customer where customer_zipcode_prefix = '14409' limit 10;
Show 5 entries
Search: Select * from customer where customer_state='SP' limit 10;
Customer__Id
Customer_unique_Id
customer_zipcode_prefix
customer_city
customer_state
06b8999e2fba1a1fbc88172c00ba8bc7 861eff4711a542e4b93843c6dd7febb0 14409 franca SP
18955e83d337fd6b2def6b18a428ac77 290c77bc529b7ac935b93aa66c333dc3 09790 sao bernardo do campo SP
4e7b3e00288586ebd08712fdd0374a03 060e732b5b29e8181a18229c7b0b2b5e 01151 sao paulo SP
b2b6027bc5c5109e529d4dc6358b12c3 259dac757896d24d7702b9acbbff3f3c 08775 mogi das cruzes SP
4f2d8ab171c80ec8364f7c12e35b23ad 345ecd01c38d18a9036ed96c73b8d066 13056 campinas SP
fd826e7cf63160e536e0908c76c3f441 addec96d2e059c80c30fe6871d30d177 04534 sao paulo SP
b2d1536598b73a9abd18e0d75d92f0a3 918dc87cd72cd9f6ed4bd442ed785235 18682 lencois paulista SP
eabebad39a88bb6f5b52376faec28612 295c05e81917928d76245e842748184d 05704 sao paulo SP
206f3129c0e4d7d0b9550426023f0a08 21f748a16f4e1688a9014eb3ee6fa325 13412 piracicaba SP
c5c61596a3b6bd0cee5766992c48a9a1 b6e99561fe6f34a55b0b7da92f8ed775 07124 guarulhos SP
Showing 0 to 0 of 0 entries (filtered from 99,442 total entries)
Previous Next