IE6700 19106 Data Management for Analytics SEC 04 Fall 2022

Project Group 28

(Siddhartha Setty, Shrunali Salian)

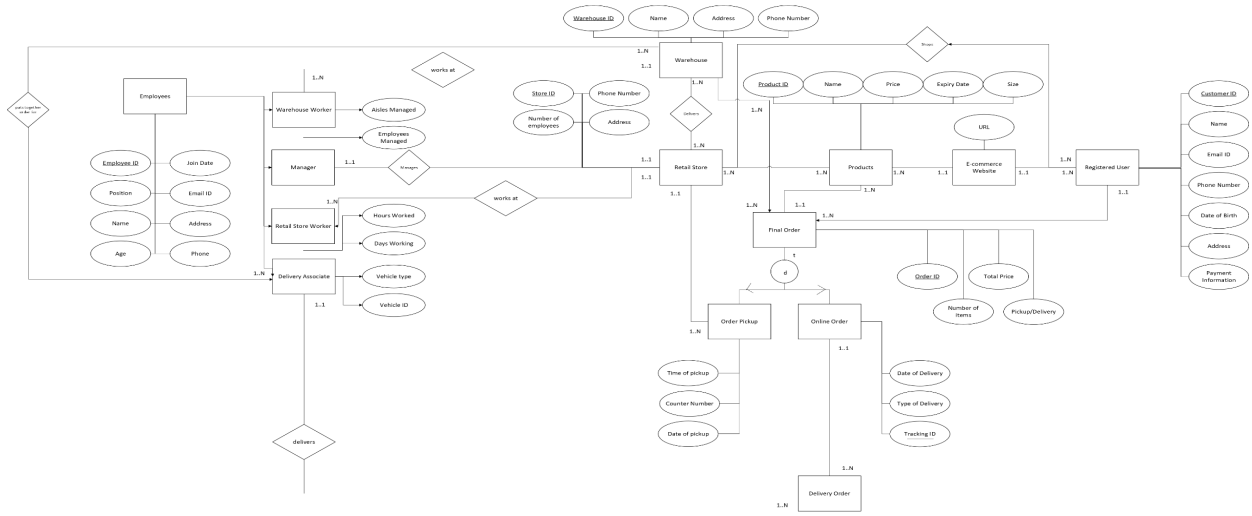
Problem Definition:

The business problem that is being implemented here is for a database system of a retail and e-commerce store. The database will include information about the Warehouse, Retail Store, and the employees that are part of the entire operation. This database will also contain information about the e-commerce website, the products that are available at the retail and the e-commerce website, and the final order and will contain information regarding the type of order the customer has, whether it is an online order or an in-person pick up of their order. The aim of this project is to have a functional database that can maintain information regarding the products being offered by the Retail store and the e-commerce website, it can also keep track of the order, have information regarding the different employees involved in the operation and contain the information about the registered users who use the services of the company.

Entities:

1. Warehouse: This contains information regarding the details about warehouse, its ID, name address, and phone number.
2. Retail store: This contains information regarding the number of employees working at the store, and the details of the store such as an address, name, and number.
3. Products: This entity will have all the information related to the products that are being offered by the retail store and the e-commerce website, such as the product ID, the name, price, size, and expiry date of the products.
4. Registered User: This entity will hold all the information related to the customer who orders products online through the eCommerce website, such as their customer ID, name, age, email ID, phone number, Date of Birth, address, and Payment Information.
5. Employees: This is another major entity in this database, this will contain all the details regarding the employees who are working on the entire operation. These details include their employee ID, the position they work at, their date along with all their personal information.
6. Final Order: This entity contains all the information regarding a registered user's final order. This includes order ID, total price, number of items, and whether the customer wants to opt for pickup or delivery.

EER Diagram:



For a detailed view of the EER diagram: https://tinyurl.com/yjf85eww

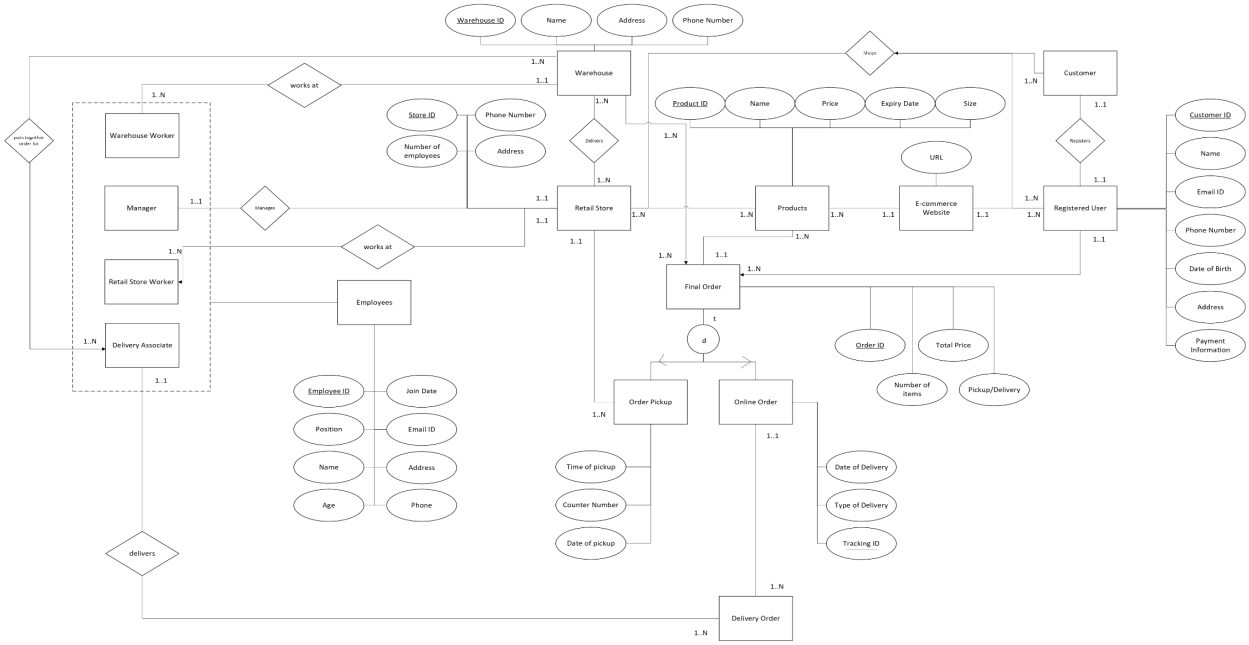
Reference data:

1. Order PickUp and Online Order both refer to Final Order for the finorder_ID
2. Delivery Order refers to Online Order for the finorder_ID
3. Order PickUp refers to the Retail Store for the RS_ID
4. Different types of employees i.e., warehouse workers, managers, retail store workers, delivery
   associate refers to the employee table for emp_ID
5. Manager refers to Retail Store for RS_ID
6. Warehouse worker refers to Warehouse for WH_ID
7. Final Order refers to the Product table for prod_ID

Transactional data:

1. The final order table contains transactional data in terms of the total price for the particular order_ID.
2. User credit card details are encrypted for privacy in the separate Registered_User_Payment table.
3. Retail Store contains the number of employees working in each store.
4. The product table contains the prod_price for each product, hence we can calculate the average
   price of a product at the supermarket.
5. The employee table contains the emp_DoB, thus Age of every employee can be found.
6. Warehouse_Worker contains the WH_WWaisles_managed thus the total number of aisles
   managed by each employee working at that warehouse can be found.

UML Diagram:



Relational Model:

Customer(user_ID, user_name, user_PhoneNo, user_email, user_DoB)

Customer_Address(*user_ID*, Apt_no, Street_name, City, State, Zipcode)

- user_ID: Foreign key refers to user_ID in Registered_User. Null values are not acceptable.

Customer_Payment(user_id,Credit_Card_No,CVV,Expiry_date)

- user_ID: Foreign key refers to user_ID in Registered_User. Null values are not acceptable. Website(URL)

Product(prod_ID, prod_name, prod_price, prod_expiry, prod_quantity)

Final_Order(finorder_ID, finorder_totalprice, finorder_numbofitems, findorder_type, *user_ID*)

- user_ID: Foreign key refers to user_ID in Registered_User. Null values are not acceptable.

Order-Product(*finorder_ID, prod_ID*)

- finorder_ID: Foreign key refers to finorder_ID in Final_Order. Null values not acceptable.
- prod_ID: Foreign key refers to user_ID in Product. Null values are not acceptable.

Online_Order(tracking_ID, delivery_date, delivery_type, *finorder_ID*)

- finorder_ID: Foreign key refers to finorder_ID in Final_Order. Null values are not acceptable.

Order_Pickup(counter_number, pickup_time, pickup_date, *finorder_ID, RS_ID*)

- finorder_ID: Foreign key refers to finorder_ID in Final_Order. Null values not acceptable.
- RS_ID: Foreign key refers to RS_ID in Retail_Store. Null values are not acceptable.

Retail_Store(RS_ID, num_emp, RS_phone, RS_address,)

Warehouse(WH_ID, WH_Name, WH_Address, WH_Phone,)

Employee(emp_ID, emp_Jobtype, emp_Name, emp_DoB, emp_joindate, emp_email, emp_address, emp_phone)

Retail_Store_Worker(*emp_ID, RS_ID*)

- RS_ID: Foreign key refers to RS_ID in Retail_Store. Null values are not acceptable.
- emp_ID: Foreign key refers to Employee. Null values are not acceptable.

Manager(*emp_ID, RS_ID*)

- emp_ID: Foreign key refers to Employee. Null values are not acceptable.
- RS_ID: Foreign key refers to RS_ID in Retail_Store. Null values are not acceptable.

Warehouse_Worker(*emp_ID*, *WH_ID*, WWaisles_managed)

- emp_ID: Foreign key refers to Employee. Null values are not acceptable.
- WH_ID: Foreign key refers to Warehouse. Null values are not acceptable.

Delivery_Associate(*emp_ID*, *WH_ID*, DA_vehicle, DA_vehicle_ID)

- emp_ID: Foreign key refers to Employee. Null values are not acceptable.
- WH_ID: Foreign key refers to Warehouse. Null values are not acceptable.

Delivery_Order(*finorder_ID,* picked_up_by)

- Picked_up_by refers to the person who picked up the order.
- finorder_ID: Foreign key refers to finorder_ID in Final_Order. Null values not acceptable.

User-RetailStore(*RS_ID, user_ID*)

- RS_ID: Foreign key refers to RS_ID in Retail_Store. Null values are not acceptable.
- user_ID: Foreign key refers to Registered_User. Null values are not acceptable.

Warehouse-Order(*finorder_ID, WH_ID*)

- finorder_ID: Foreign key refers to finorder_ID in Final_Order. Null values not acceptable.
- WH_ID: Foreign key refers to Warehouse. Null values are not acceptable.

Warehouse-RetailStore(*RS_ID, WH_ID*)

- WH_ID: Foreign key refers to Warehouse. Null values are not acceptable.
- RS_ID: Foreign key refers to RS_ID in Retail_Store. Null values are not acceptable.

Warehouse-Delivery-Associate(*WH_ID*, *emp_ID*)

- emp_ID: Foreign key refers to Employee. Null values are not acceptable.
- WH_ID: Foreign key refers to Warehouse. Null values are not acceptable.

SQL Queries:

1. In recent news, it was said that there's swine flu in Colorado hence all meat products especially chicken have been contaminated, hence the supermarket has decided to recall all the chicken products sold to customers that came from the Warehouse in Colorado. Retrieve the list of all the customers the supermarket should contact.

```
5
6   select user_ID,user_name,user_PhoneNo,user_email
7   from DMA_Project_1.Registered_User2
8   where user_ID in(
9                   select user_ID from DMA_Project_1.Warehouse_Order as WO inner join DMA_Project_1.Final_Order as FO
10                  where WO.finorder_ID = FO.finorder_ID and
11                  WO.WH_ID in (select WH_ID from DMA_Project_1.Warehouse where WH_Name like "%colorado%") and
12                  FO.finorder_ID in ( select finorder_ID from DMA_Project_1.Order_Product as OP inner join DMA_Project_1.Product as P
13                  where OP.prod_ID = P.prod_ID and P.prod_name like "%chicken%"));
14
```

100% 1:113

Result Grid | Filter Rows: Search | Export:

user_ID	user_name	user_PhoneNo	user_email
28-2297938	Ashika Kalmadi	617-206-8766	ashika@ezinearticles.com
18-2217938	Ishita Kalmadi	617-226-8766	ishita@ezinearticles.com
28-2297123	Luna Potter	617-216-8766	luna@ezinearticles.com
28-2297121	Harry Potter	617-006-8766	harry@ezinearticles.com

Result Grid | Form Editor

2. Registered user Samantha received her order however, there was one item missing in the order. She informed the customer care about the same. Retrieve the contact details of the warehouse the customer care should contact in order to check about Samantha's order.

```
select * from DMA_Project_1.Warehouse
where WH_ID in(
                select WH_ID
                from DMA_Project_1.Delivery_Associate
                where finorder_ID in (
                                        select finorder_ID
                                        from DMA_Project_1.Final_Order
                                        where user_ID in (
                                                            select user_ID
                                                            from DMA_Project_1.Registered_User2
                                                            where user_name like '%samantha%')));
```

Result Grid | Filter Rows: | Export:

WH_ID	WH_Name	Wh_Phone	WH_address
33-7488679	Rio Grande	461-656-7003	Nevada

3. The management is curious to compare the generation-wise spending habits of its customers

```
Select
case
     when year(user_DoB) between 1945 and 1964 then "Baby Boom Generation"
     when year(user_DoB) between 1965 and 1980 then "Gen X"
     when year(user_DoB) between 1981 and 1996 then "Millennial"
     when year(user_DoB) between 1997 and 2010 then "Gen Z"
     else "Silent Generation"

end as user_DoB , Avg(FO.finorder_totalprice) as Average_Money_Spent_Generation_wise
From DMA_Project_1.Registered_User2  as RU inner join DMA_Project_1.Final_Order_Data as FO
on FO.user_ID = RU.user_ID
Group by
          case
               when year(user_DoB) between 1945 and 1964 then "Baby Boom Generation"
               when year(user_DoB) between 1965 and 1980 then "Gen X"
               when year(user_DoB) between 1981 and 1996 then "Millennial"
               when year(user_DoB) between 1997 and 2010 then "Gen Z"
               else "Silent Generation"
          end
;
```

Result Grid | Filter Rows: | Export:

user_DoB	Average_Money_Spent_Generation_w...
Gen X	333.8181818181818
Gen Z	323.2857142857143
Millennial	304

4. The year-wise trend of vegan products vs meat products

```
select prod_type as PRODUCT ,YEAR(prod_expiry_date)as YEAR,sum(prod_quantity) as SUM_OF_PRODTYPE
from Project_Data.Product
where prod_type like "%vegan%" or prod_type like "%meat%" group by YEAR(prod_Expiry_date),prod_type
Order by YEAR(prod_Expiry_date);
```

Result Grid | Filter Rows: | Export:

PRODUCT	YEAR	SUM_OF_PRODTYPE
Deli(meat)	2013	6900
Vegan(meat)	2013	80
Deli(meat)	2014	5100
Vegan(meat)	2014	100
Deli(meat)	2015	5000
Vegan(meat)	2015	500
Deli(meat)	2016	4500
Vegan(meat)	2016	3800
Deli(meat)	2017	4000
Vegan(meat)	2017	200
Deli(meat)	2018	3800
Vegan(meat)	2018	4000
Deli(meat)	2019	3500
Vegan(meat)	2019	4800
Deli(meat)	2020	1000
Vegan(meat)	2020	5100
Deli(meat)	2021	500
Vegan(meat)	2021	5500
Deli(meat)	2022	3100
Vegan(meat)	2022	6000

5. The management wants to look at the inventory to evaluate the value of each product and product category to allocate proper inventory space. They ask us to calculate the worth of the inventory for each product as well as each product category

```
36
37  select
38      prod_type as "Product Type",
39      prod_name as "Product Name",
40      prod_price*prod_quantity as "Inventory Value"
41  from DMA_Project_1.Product
42  order by prod_price*prod_quantity desc;
43
44  select
```

100% 40:42

Result Grid | Filter Rows: Search | Export:

Product Type	Product Name	Inventory Value
Womens Fashion	Lilac Pants	9000000
Womens Fashion	Neon Pants	9000000
Womens Fashion	Yellow Pants	2700000
Mens Fashion	Yellow Shirt	2700000
Womens Fashion	Pink Dress	2700000
Kids	Hamleys toy	1200000
Womens Fashion	Yellow Pants	900000
Kids	Story Book	809000
Kids	Story Book	809000
Kids	Boys shirt	328000
Kids	Barbie Toy	320000
Mens Fashion	Blue shoe	306000
Womens Fashion	Blue Dress	270000

Result 7 | Read Only

```
40      prod_price*prod_quantity as "Inventory Value"
41  from DMA_Project_1.Product
42  order by prod_price*prod_quantity desc;
43
44  select
45      prod_type as "Product Category",
46      sum(prod_quantity) as "Total Quantity",
47      sum(prod_quantity) * sum(prod_price) as "Total Value"
48  from DMA_Project_1.Product
49  group by prod_type
50  order by sum(prod_quantity) * sum(prod_price) desc;
51
52
53
```

100% 52:50

Result Grid | Filter Rows: Search | Export:

Product Categ...	Total Quanti...	Total Value
Womens Fashion	12900	216720000
Kids	37180	57257200
Deli(seafood)	28700	10619000
Mens Fashion	5400	9558000
Grocery	24400	4806800
Deli(meat)	7100	781000
Electronics	88	160184.63999999998

6. Retrieve the average age of employees based on their job type

```
57  select emp_jobtype, avg(age) from
58      (SELECT *, DATE_FORMAT(FROM_DAYS(DATEDIFF(NOW(), emp_DoB)), '%Y') + 0 AS age
59      FROM DMA_Project_1.Employee e1) as t1
60  group by emp_Jobtype;
61
62
63
```

100% 22:60

Result Grid | Filter Rows: Search | Export:

emp_Jobtype	avg(age)
Warehouse Worker	36.76923076923077
Manager	47.63636363636363
Delivery Associate	37.3
Retail Store Worker	37.375

7. Number of people who are choosing for 'pick up' or 'delivery' depending on the number of items they purchased

```
SELECT
    finorder_numbofitems,
    COUNT(IF(findorder_type = 'Standard', 1, NULL)) Delivery,
    COUNT(IF(findorder_type = 'PickUp', 1, NULL)) Pickup,
    COUNT(IF(findorder_type = 'Standard', 1, NULL))/COUNT(IF(findorder_type = 'PickUp', 1, NULL)) as ratio
FROM
    DMA_Project_1.Final_Order
group by finorder_numbofitems
order by finorder_numbofitems;
```

Result Grid

finorder_numbofitems	Delivery	Pickup	ratio
1	5	3	1.6667
2	4	7	0.5714
3	4	7	0.5714
4	2	5	0.4000
5	3	17	0.1765
6	4	2	2.0000
7	6	8	0.7500
8	5	7	0.7143
9	6	4	1.5000
10	5	6	0.8333

8. Retrieve the list of products that will expire in the next 7 days

```
select * from DMA_Project_1.Product where prod_expiry_date < date_add(now(), interval 7 day);
```

Result Grid

prod_ID	prod_type	prod_name	prod_price	prod_Expiry_date	prod_quant...
193054321	Deli(seafood)	Salmon	20	2022-11-27	3000
523054321	Grocery	Yogurt	5	2022-11-27	1000
623054321	Grocery	Watermelon	5	2022-11-24	1000
666666321	Grocery	Spinach	5	2022-11-23	1000
812304321	Deli(meat)	Chicken Breasts	10	2022-11-25	1000
913334321	Grocery	Spinach	5	2022-11-21	1000
917054006	Grocery	Spinach	15	2022-11-21	100

Product 12

9. Retrieve names of customers and their credit card information encrypted with only the last 4 digits un-encrypted

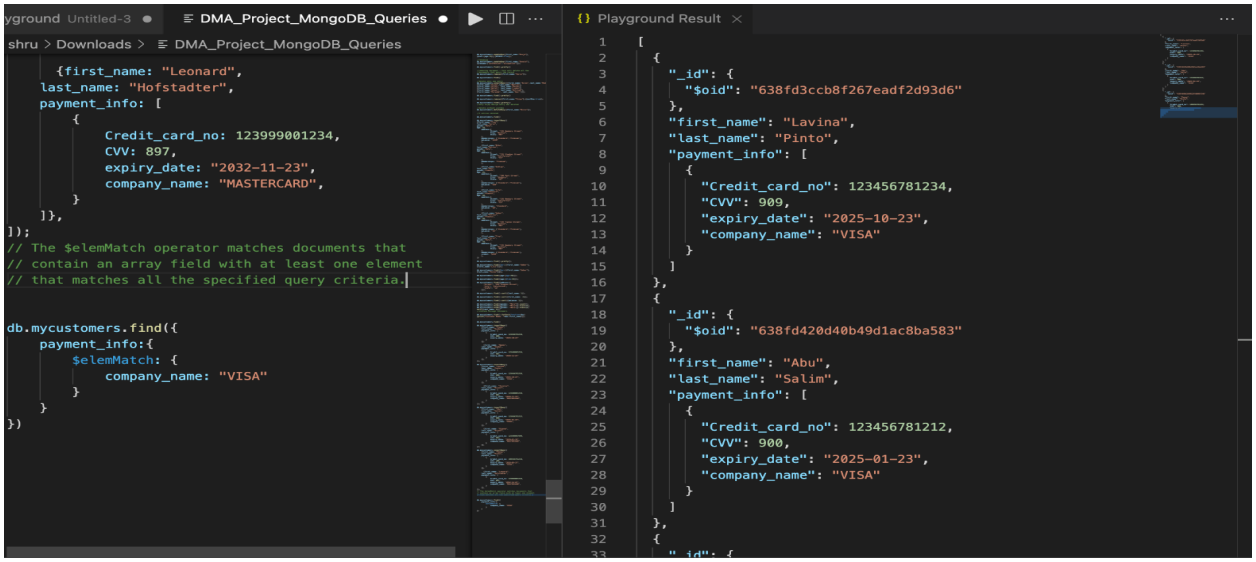
```
Select ru.user_name, concat(repeat('*',length(rup.card_no)-4),
Substring(rup.card_no,-4)) as CreditCardNumber
From DMA_Project_1.Registered_User2 as ru
Inner join DMA_Project_1.Registered_User_Payment as rup using (user_ID);
```

Result Grid

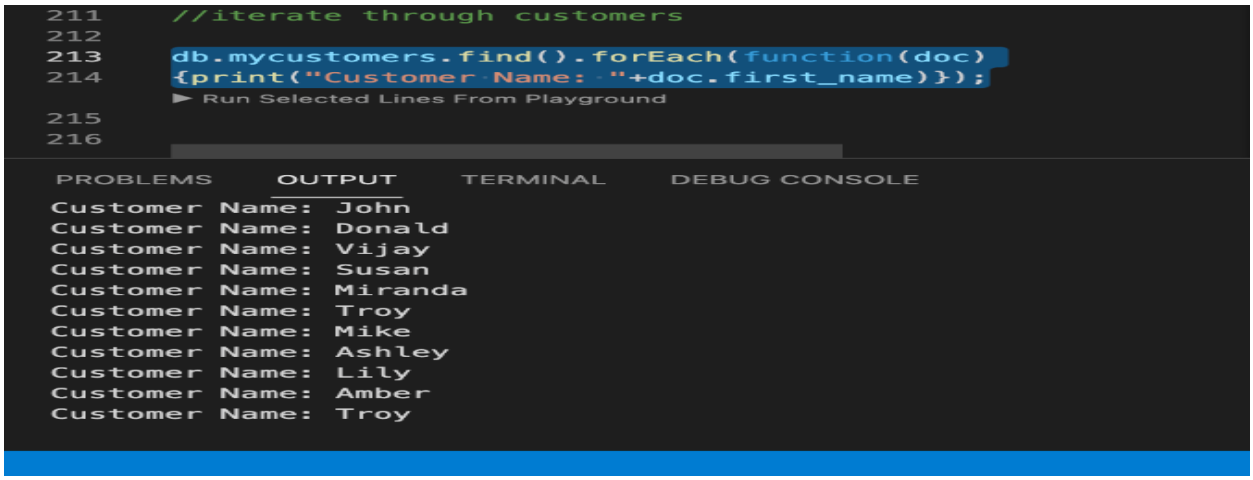
user_name	CreditCardNumb...
Ynes Polland	**********3065
Abran Chatenet	**********3890
Reggi Sandever	**********3870
Markus Ambresin	**********3001
Hugo Gerard	**********3002
Ammamaria Bagot	**********3003
Cleon Rawcliffe	**********3005

NoSQL Queries:

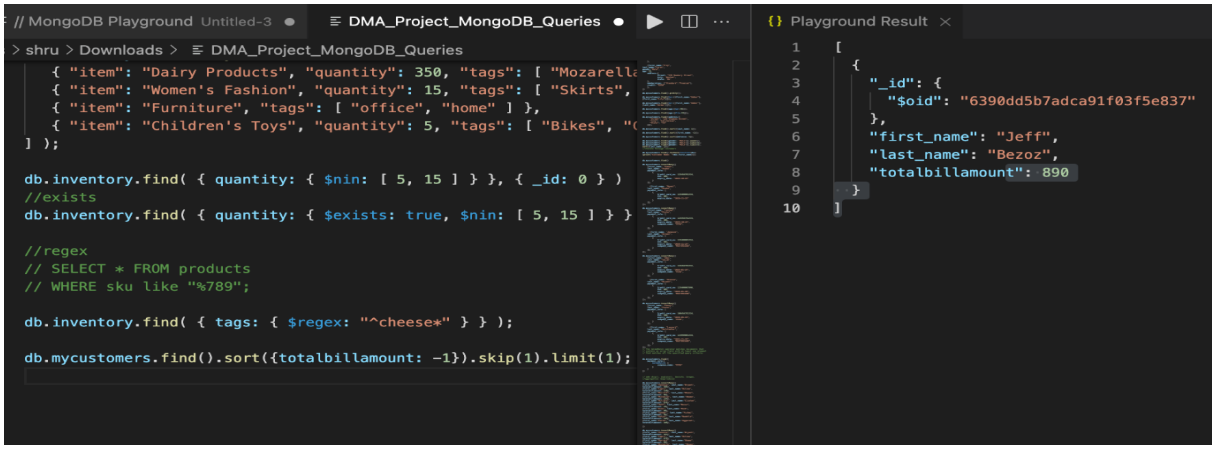
1. Retrieve the list of all customer names who use a VISA credit card.



2. Iterate the list of all customer first names using forEach



3. Retrieve the second-highest customer spender



4. Retrieve the product details of products that are not below the threshold quantity.

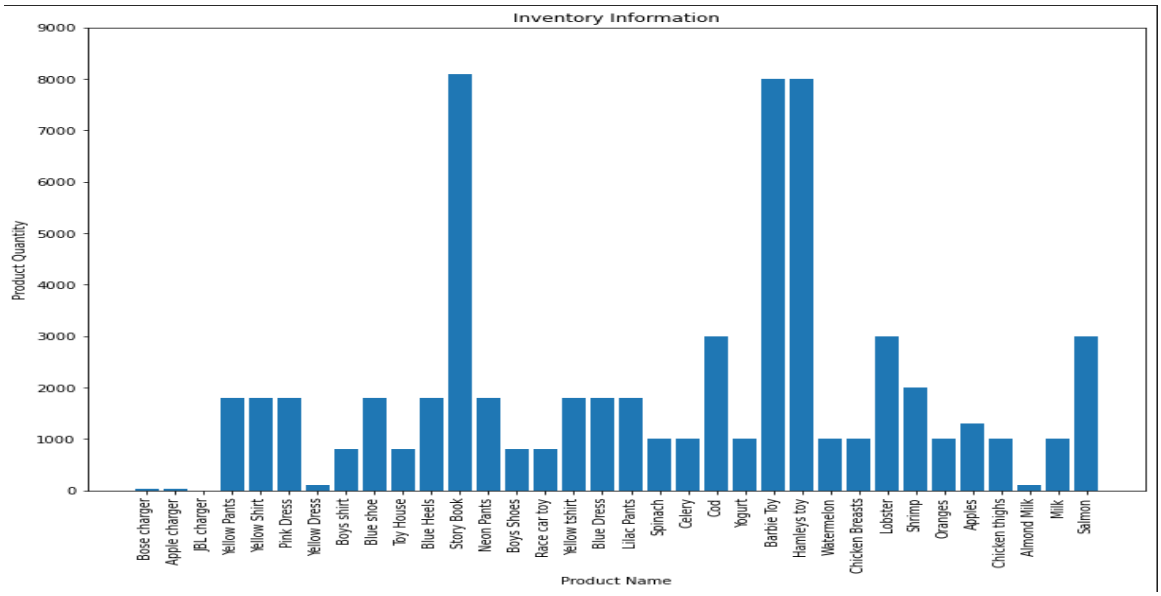
```
// MongoDB Playground Untitled-3 ●   DMA_Project_MongoDB_Queries ●
Users > shru > Downloads > DMA_Project_MongoDB_Queries

393  //             avg_totalbillamount: {$avg :"$totalbillamount"}
394  //         }
395  //     },
396  //     { $skip: 2}
397  // ]);
398
399  db.mycustomers.find().sort({totalbillamount: - 1}).skip(1);
400
401
402  use inventory;
403
404  db.inventory.insertMany( [
405     { "item": "Dairy Products", "quantity": 350, "tags": [ "Mozarella c
406     { "item": "Women's Fashion", "quantity": 15, "tags": [ "Skirts", "S
407     { "item": "Furniture", "tags": [ "office", "home" ] },
408     { "item": "Children's Toys", "quantity": 5, "tags": [ "Bikes", "Cri
409  ] );
410
411  db.inventory.find( { quantity: { $nin: [ 5, 15 ] } }, { _id: 0 } )
     ▶ Run Selected Lines From Playground
```

```
{} Playground Result
1   [
2     {
3       "item": "Dairy Products",
4       "quantity": 350,
5       "tags": [
6         "Mozarella cheese",
7         "cheddar cheese"
8       ]
9     },
10    {
11      "item": "Furniture",
12      "tags": [
13        "office",
14        "home"
15      ]
16    }
17  ]
```

Visualizations:

1. The quantity of each product stored in the warehouse inventory

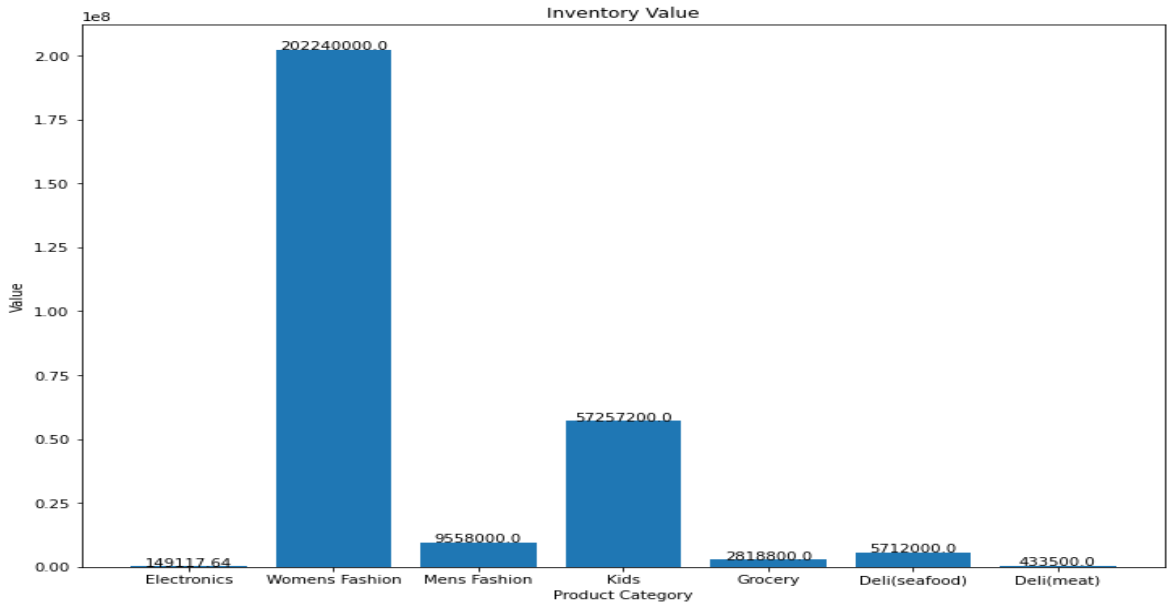


The visualization helps us to understand the quantity of each product stored in a particular warehouse. This can help us the dynamics of stocks of different products in the warehouse. Given the quantity of a particular product falls below the threshold, the management can restock the products. It can also be useful to find the worth of inventory products, and by virtue the value of each individual warehouse.

2. Scatter plot of the employee age distribution based on their profession/role

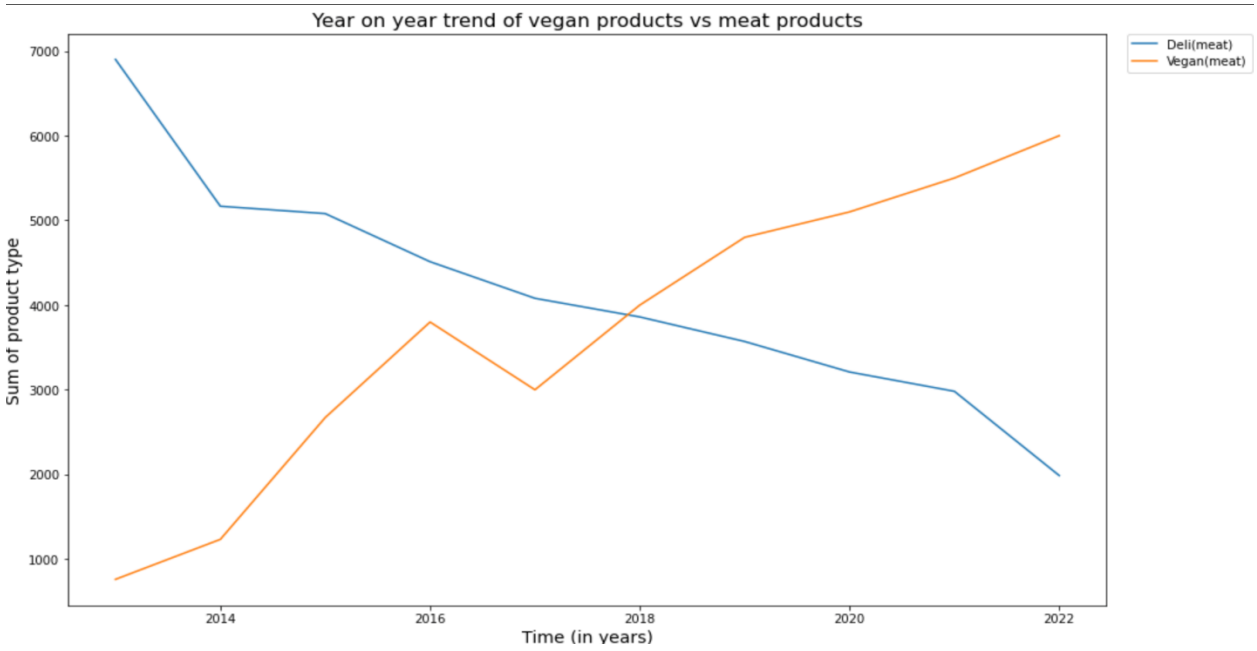
The visualization can help us understand the average age of an employee in the company. We can see that the company does not have employees in a particular age group rather the employees range from 18-year-olds to above 60 years of age. The scatter plot indicates that almost 50% of the warehouse workers are below the age of 30 and also that the youngest employee in the company is a manager.

3. The Inventory value for all categories of products in USD



The bar graph shows the inventory value of each product type. From this, we can extrapolate that since the supermarket might have a high sale of women's fashion products it has the highest inventory for these products.

4. The year-on-year trend of vegan vs meat products



Since veganism is a growing trend worldwide the management wants to find out the trajectory of the sale of vegan products. From the given graph the management anticipates steady growth of vegan products in the future.

Summary and Recommendation:

- The database helps us connect various entities of a supermarket. Thus, the management can keep a track of what products are being supplied to each retail store, and which retail store tends to sell which product type the most and accordingly divert the supply chain of the resources.
- Past consumer data also helps the management understand the seasonal nature of products, for instance, during Christmas consumers tend to buy candies, decorations, turkey, etc however on a regular day consumers will buy more milk and eggs. The database helps us in understanding what products are bought together, i.e, clusters can be formed, for instance, milk, eggs, and bread go together.
- After a successful implementation of the supermarket database, we can summarize as follows, for database management of structured data it is better for the management to continue using a relational database model. However, the management can use unstructured data such as customer bills in MongoDB to analyze consumer behavior.

THANK YOU