FINAL EVALUATION REPORT
DBMS
GROUP - 17

Group members:

Ishaan Marwah - 2020068
Jai Singh - 2020070
Mohit Sharma - 2020086
Nakul Periwal - 2020316

Scope of the project

This project covers a database design with an E-R diagram of a **retail store system**. We have identified the majority of **entities**, their respective **attributes,** and their **relationships**. This database design has been designed keeping in mind the entities such as **employees, departments, product, category, supplier, customer, etc.,** while also specifying their attributes and the corresponding relationships that they share. The dependency between distinct entities has also been demonstrated in the **E-R diagram.**

What have we done so far?

We started our discussion by thinking about all the possible entities for the retail store database design. We thought of all the potential stakeholders of our retail store.
We figured out all the general and necessary entities. Then we started brainstorming about the potential attributes of these entities. After that, we recognised the relationships between the selected entities and laid a basic outlay of our ER diagram in this way.

After that, we started to draw our ER diagram in an online space, visualised all the entities(with their attributes), and demonstrated their corresponding relationships.
While finalising our ER diagram, we noted down our relational schema.
We identified all the attributes as well as the primary key in our schema.

We have also identified the weak entity and the ternary relationship in our database design.

The database for our project was generated in an online fashion using a random data generator.

We have also written a set of SQL queries for justifying the functioning of our database design, such as:

1. **Add product to the cart:-** Using this query, the user would be able to add a new product to the cart.

2. **Search in product list:**- Using this query, the user would be able to search for the desired product that they are looking for.

3. **Sort product list:**- This query will help in sorting the product list in both alphabetical and reverse alphabetical order as per the requirements.

4. **Search based on cost range:-** This query will help find a product as per the user's budget. This is a very beneficial feature as it will help identify only those products in the user's budget range.

5. **Search based on multiple constraints:-** This query demonstrates the enhanced searching effectiveness of our design. A user can search for a product by applying multiple constraints and can find the appropriate product.

6. **Details of the product:-** This query will help display all the details of a particular product we are willing to seek.

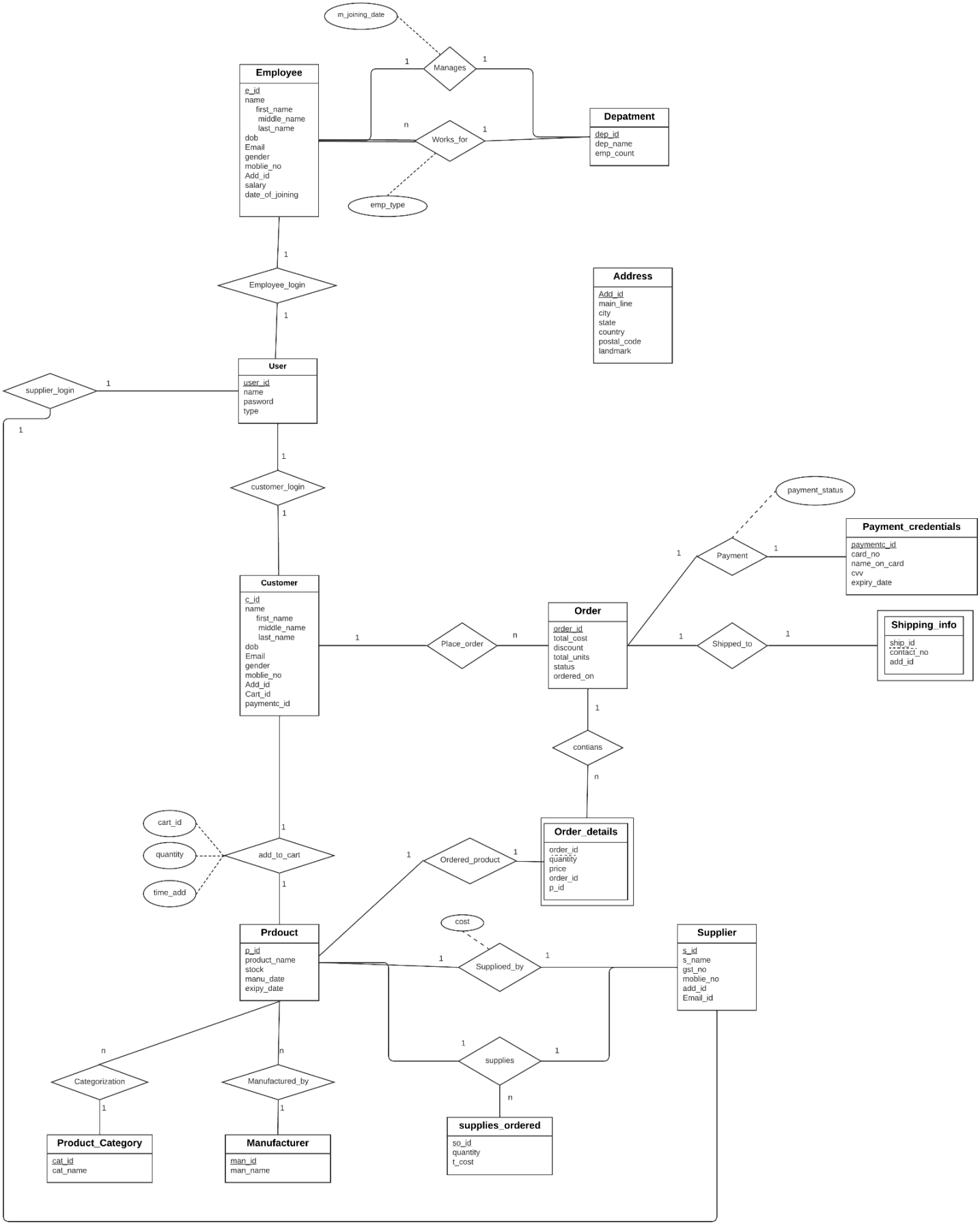
7. **All products of a category:-** This query will display all the products of a certain category. Hence we can find all the products of the concerned category.

8. **Employee's salary:-** This query can help in finding the total and average salary of an employee. We can also update an employee's salary using this query.

9. **Employees branching based on department:**- This query will help in searching all the employees of a particular department. This query can be very useful while evaluating the employees of the concerned department.

10. **Address Insertion/deletion:**- This query will help us in adding the address credentials of a new user. This query can also be used in deleting the address of a particular user who isn't willing to avail our retail store services in the future.

ER diagram - LINK (miro), LINK (Tabular form)


m_joining_date
Manages
Employee
e_id
name
first_name
middle_name
last_name
dob
Email
gender
moblie_no
Add_id
salary
date_of_joining
Works_for
Depatment
dep_id
dep_name
emp_count
emp_type
Employee_login
Address
Add_id
main_line
city
state
country
postal_code
landmark
User
user_id
name
pasword
type
supplier_login
customer_login
payment_status
Payment_credentials
paymentc_id
card_no
name_on_card
cvv
expiry_date
Payment
Customer
c_id
name
first_name
middle_name
last_name
dob
Email
gender
moblie_no
Add_id
Cart_id
paymentc_id
Place_order
Order
order_id
total_cost
discount
total_units
status
ordered_on
Shipped_to
Shipping_info
ship_id
contact_no
add_id
contians
cart_id
quantity
time_add
add_to_cart
Ordered_product
Order_details
order_id
quantity
price
order_id
p_id
cost
Prdouct
p_id
product_name
stock
manu_date
exipy_date
Supplioed_by
Supplier
s_id
s_name
gst_no
moblie_no
add_id
Email_id
supplies
Categorization
Manufactured_by
supplies_ordered
so_id
quantity
t_cost
Product_Category
cat_id
cat_name
Manufacturer
man_id
man_name

Changes In ER diagram

1. Cost removed from product and added to the table supplied by. The problem was that multiple suppliers could have provided different prices for the same product. So it did not make sense to add a new product ID for the same product.

2. Manager joining date and employee joining date shifted to the manages and works_for table. The problem was that earlier if an employee was promoted to the manager then we were losing previous information about the employee joining date.

Stakeholders

1. **Customer** - They are the end-user of the software. A customer needs to log in to the portal after login, he/she can view/ buy the products Available.

2. **Employees** - These are the people hired by the organisation to work under different departments. Depending on the system/Departments, there can be many types of employees. There are four types of employees in our system - IT, storage, customer service, marketing.

3. **Suppliers/Distributors** - Suppliers are the ones that sell goods to the business (our portal). A supplier can supply new products or discontinue existing products and deliver the supplies to the business

4. **Manufacturers** - Manufacturers are the ones who supply the product to the suppliers. We are storing the details to the manufacturers to improve the Filter/Search quality for the end-users.

Identification of Weak entities:

1. **Shipping information -** The **Shipping_info** is a weak entity because it is dependent on the Order entity, and there will be no use of shipping_info if the customer has not placed the Order.

2. **Order details -** The **order_details** is also a weak entity because we can't store the details of an order if the customer has not placed the Order.

Ternary Relationship:

1. **Supplies -** There is a Ternary relation between the supplier, supplies_ordered and product

    - One supplier with one specific supplies_ordered supplies one product (product entry, not it's stock)
    - One supplier supplies one Product for n Supplies_ordered

Detailed Report

Customer - The customer is a basic entity with attributes customer id (c_id) Name (first_name, middle_name, last_name), gender, Date of birth (dob), Email, Mobile number (mobile_no), Address (add_id),, A customer can perform many tasks like the searching product, adding to cart etc.

- Each customer can be uniquely identified by the id assigned to it.
- A new customer can search the products using their name.
- He/she can filter the products based on the manufacturer or category type.
- Each customer is also assigned a unique cart Id (cart_id) where he/she can store the product he /she wants to buy
- He/she can add new products to the cart or remove the previously added products from the cart
- Then the customer can proceed to buy all the products added to the cart
- To place an order he/she needs to provide the shipping details and payment credentials
- In this step, he/she has an option to use his saved address and payment credentials (paymentc_id).
- There are two types of payment options available cash on delivery and by card.
- Only one user can log in using one mobile number

Employee - These are the people hired by the organization to work under different departments. There are four types of departments and each employee belongs to one of the departments (storage, IT, customer service, marketing). In addition to this, an employee can also be a manager of the respective department.

Each employee is assigned a unique employee id (e_id). Employees also have the following attributes - Name (first_name, middle_name, last_name), Date of Birth (dob), Gender (gender), Email ID (Email), Mobile Number (mobile_no), Address (add_id), Salary (salary), Date of Joining (date_of_joining) and employee Type (emp_type)

- According to the department types and employees perform different tasks
- An employee working in the Storage department can order/add new products
- Employees are there for IT, storage management, marketing for the company and in the customer service department and each of these departments is managed by the manager who is also an employee in the same department who looks after the management of his/her particular department.

Department - There are mainly four departments

Storage IT

customer service marketing

- Each of the departments has a manager and many employees
- Each department can be identified by its unique department id (dep_id)
- Department has Its department name (dep_name), the total employee working in that department (emp_count) and the joining date of the current manager (m_joining_date)

User - To verify all the peoples who are accessing the portal each verified customer employee and supplier is assigned a unique login id (user_id) which contains the name of the person/ institute (name), a password (password) and the type (type) of the user namely employee supplier or customer

- We can add a new entry to the table i.e, registering a new customer/ employee/ supplier
- Or we can delete an entry from the table i.e, removing an existing customer/ employee/ supplier

Products - These are the goods available at the online retail store. This entity table stores all the required information for the product (can depend on the platform)

Each product has its unique product id (p_id) assigned to it. Its name (product_name), cost (cost), the stock currently available (stock), manufacturing and expiry dates (manu_date, expiry_date)

- To uniquely identify the products each product are assigned a unique product id (p_id)
- We further categorize the products into different categories (like vegetables, fruits, dairy products, disposables etc.)
- Employees working in the Storage department handles the information of the products they can add new products, remove existing products etc.

Supplier - Supplies are the entities that supply the product to our portal from the manufacturers

- Every supplier has a unique supplier id (s_id) to identify the supplier.
- Each supplier has a supplier name (s_name), GST Number (gst_no), and its contact information mobile number (moblie_no), address (Add_id) and email id (Email)
- A supplier supplies the products to the business
- A supplier can supply new products and can also discontinue the supply of existing product

Manufacturer - To help the customers and improve the filter we are also storing the manufacturer name so that the customers can easily search all the products from a single manufacturer.

- To uniquely identify the manufacturer we are storing manufacturer id (man_id) and manufacturer name.
- An employee working in the storage department can add details of a new manufacturer

Product category - To help the customers and improve the filter we are also storing the product categories so that the customers can easily search all the products under the same category (cat_name)

- To uniquely identify the product category we are storing category id (cat_id) and category name.
- An employee working in the storage department can add details of a new category available in the online shop

Address - This entity stores the Addresses used in different tables (customer, supplier, employee, shipping details).

- Each address Line has a unique address is assigned to it (Add_id)

- Each Address entry in the table has an address line 1 (main_line), city, state, country, PIN code (postal_code), landmark

Order - This entity stores the overall detail of the order.

Each entry in the table has a unique order id (order_id), It also stores the total cost (total_cost), discount, the status of the order (shipped/ delivered), Total units in the particular order (total_units) and the date on which order was placed

Payment Credentials - This entity stores the payment information of the orders.

Each entry in the Table has a unique Id (paymentc_id), Card number (card_no), name on the card(name_on_card), CVV (cvv) and Its expiry date (expiry_date) This table also save the information (if the user wants) to skip this step for future orders.

Shopping Info - A weak entity. This entity stores the shopping information of the orders placed by the customer.

Each entry in the Table has a unique Id (ship_id), Contact details(contact_no) of the customer and the address (add_id) where the order has to be shipped

Order Details - A weak entity. This entity stores the details of the products ordered and their quantity in each order.

Each entry in the table has a unique Id (detail_id), an order (order_id) and the product details - product Id (p_id), Price (price), Quantity (quantity)

Relation Schemas

Customer(c_id# , first_name, middle_name, last_name, gender, dob, Email, mobile_no, add_id#, cart_id#, paymentc_id#)

Employee(e_id#, first_name, middle_name, last_name, dob, gender, Email, mobile_no, add_id#, salary, date_of_joining, emp_type)

Department(dep_id#, dep_name, emp_count, m_joining_date)

User(user_id#, name, password, type)

Product(p_id#, product_name, cost, stock, manu_date, expiry_date)

Supplier(s_id#, s_name, gst_no#, moblie_no, Add_id#, Email)

supplies_ordered(so_id#, t_cost, quantity)

Product_category(cat_id#, cat_name)

Manufacturer(man_id#, man_name)

Address(Add_id#, main_line, city, state, country, postal_code, landmark)

Order(order_id#, total_cost, discount, status, total_units, ordered_on)

payment_credentials(paymentc_id#, card_no, name_on_card, cvv, expiry_date)

payment(order_id#, paymentc_id, payment_mode, payment_status)

Shipping_info(dotted_underline(ship_id#), contact_no, add_id)

order_details(dotted_underline(detail_id#), order_id, p_id, price, quantity)

Manages(e_id, dep_id)

Works_for(e_id, dep_id)

Customer_login(user_id#, c_id#)

Employee_login(user_id#, e_id#)

Supplier_login(user_id#, s_id#)

add_to_cart(c_id, p_id, cart_id, quantity, purchased, time_add)

supplied_by(p_id , s_id)

supplies(so_id#, s_id, p_id)

Categorization(p_id, cat_id)

Manufacured_by(p_id, man_id)

place_order(c_id, order_id #)

shipped_to(order_id#, ship_id)

Relation

Employee “works_for” Department
Employee “manages” Department

Customer “places_order” Order
Order “shipped_to: shipping_info
Order “contains” order_details
Order_details “ordered_products” products
Order “payment” payment_credentials

Customer “add_to_cart” product
Product “categorization” product category
Product “maufactured_by” manufacturer
Product “supplied_by” supplier

Supplier “suppliers” supplies_ordered for products
Customer “customer_login” user
Supplier “supplier_login” user
Employee “Employee_login” user

Database Schema

PK - Primary key
NN - not null
UQ - Unique

Tables	fields	Datatype	constraints
add_to _cart	cart_id c_id p_id quantity purchased time_added	int int int int varchar(5) datetime	PK, NN, UQ NN PK, NN NN NN PK, NN
address	add_id main_line city state country postal_code landmark	int varchar(60) varchar(45) varchar(45) varchar(45) int varchar(100)	PK, NN, UQ NN NN NN NN NN
categorization	cat_id p_id	int int	PK, NN PK, NN foreign key (p_id) references (product) foreign key (cat_id) references (product_category)
customer	c_id first_name middle_name last_name gender date_of_birth email phone add_id cart_id paymentc_id	int varchar(45) varchar(45) varchar(45) varchar(1) date varchar(70) varchr(20) int int int	PK, NN, UQ NN NN NN NN NN, UQ UN NN, UQ foreign key (add_id) references (address) foreign key (paymentc_id) references (payment_credential)
customer_login	user_id	int	PK, NN, UQ

	c_id	int	NN, UQ foreign key (user_id) references (user) foreign key (c_id) references (customer)
department	dep_id dep_name emp_count	int varchar(45) int	PK, NN, UQ NN NN
employee	e_id first_name middle_name last_name gender date_of_birth email phone salary date_of_joining add_id	int varchar(45) varchar(45) varchar(45) varchar(1) date varchar(70) varchr(20) decimal(10,2) date int	PK, NN, UQ NN NN NN NN NN NN NN, UQ
employee_login	user_id e_id	int int	PK, NN, UQ NN, UQ foreign key (user_id) references (user) foreign key (e_id) references (employee)
manages	e_id dep_id m_joining_date	int int date	PK, NN PK, NN NN foreign key (dep_id) references (department) foreign key (e_id) references (employee)
manufactured_by	p_id man_id	int int	PK, NN NN foreign key (p_id) references (product) foreign key (man_id) references (manufacurer)
manufacturer	man_id man_name	int varchar(70)	PK, NN, UQ NN
order	order_id total_cost discount status total_units ordered_on	int decimal(10,2) decimal(10,2) varchar(45) int date	PK, NN, UQ NN NN NN NN NN
order_details	detial_id order_id p_id	int int int	PK, NN, UQ NN NN

	price quantity	decimal(10,2) int	NN NN foreign key (p_id) references (product) foreign key (order_id) references (order)
payment	order_id paymentc_id payment_status	int int varchar(45)	PK, NN NN foreign key (paymentc_id) references (payment_credentials) foreign key (order_id) references (order)
payment_credentials	paymentc_id card_no name_on_card cvv expiry_date	int bigint varchar(100) int date	PK, NN, UQ NN NN NN NN
place_order	c_id order_id	int int	PK, NN PK, NN, UQ foreign key (c_id) references (customer) foreign key (order_id) references (order)
product	p_id product_name stock manu_date expiry_date	int varchar(45) int date date	PK, NN, UQ NN NN NN NN
product_category	cat_id cat_name	int varchar(45)	PK, NN, UQ NN, UQ
shipped_to	order_id ship_id	int int	PK, NN, UQ NN, UQ foreign key (ship_id) references (shipping_info) foreign key (order_id) references (order)
shipping_info	ship_id contact_no add_id	int varchar(15) int	PK, NN, UQ NN NN foreign key (add_id) references (address)
supplied_by	p_id s_id cost	int int decimal(10,2)	PK, NN PK, NN NN foreign key (p_id) references (product) foreign key (s_id) references (supplier)
supplier	s_id s_name gst_no	int varchar(70) bigint	PK, NN, UQ NN NN, UQ

	phone add_id email	varchar(45) int varchar(70)	NN NN, UQ NN foreign key (add_id) references (address)
suppler_login	user_id s_id	int int	PK, NN, UQ NN, UQ foreign key (user_id) references (user) foreign key (s_id) references (supplier)
supplies_ordered	so_id s_id p_id t_cost quantity received	int int int decimal(10,2) int varchar(5)	PK, NN, UQ NN NN NN NN NN foreign key (p_id) references (product) foreign key (s_id) references (supplier)
user	user_id name password type	int varchar(45) varchar(45) varchar(45)	PK, NN, UQ NN NN NN
works_for	e_id dep_id emp_type	int int varchar(45)	PK, NN, UQ PK, NN NN foreign key (dep_id) references (department) foreign key (e_id) references (employee)

FEW THINGS BEFORE STARTING ON YOUR SYSTEM

grant_revoke.py and new_user.py- Change the password of the root user

```
passwd = 'mohit20086'
```

Make sure to add the database with the name retail_store
Add the user :

```
-- sql querry to create user 'admin_user with all privilages
CREATE USER 'admin_user'@'localhost' IDENTIFIED BY 'admin_user';
GRANT ALL privileges ON *.* TO 'admin_user'@'localhost';
GRANT CREATE USER ON *.* TO 'admin_user'@'localhost';
```

**Then run the grant_revoke file to create necessary view and user of the database

View And Grants

Created a python script to assign grants to all the users of the database and create appropriate views so that a user cannot access the database of the other user

QUERIES

1. Add in cart-

```
select P.p_id, P.product_name, (select cost from supplied_by where P.p_id = p_id) as cost, P.stock, P.manu_date, P.expiry_date from product P order by P.product_name
```

2. Search In product based on cost range

```
select * from (select P.p_id, P.product_name, (select min(cost) from supplied_by where P.p_id = p_id) as cost, P.stock, P.manu_date, P.expiry_date from product P) as P where P.cost between 10 and 110;
```

3. Search In product based on Manufacturer

```
select P.p_id, P.product_name, (select min(cost) from supplied_by where P.p_id = p_id) as cost, P.stock, P.manu_date, P.expiry_date from product P where P.p_id IN (select M.p_id from manufactured_by M where M.man_id in (select A.man_id from manufacturer A where man_name like '%"boa"%'))
```

4. Search In product based on category

```
select P.p_id, P.product_name, (select min(cost) from supplied_by where P.p_id = p_id) as cost, P.stock, P.manu_date, P.expiry_date from product P where P.p_id IN (select C.p_id from categorization C where C.cat_id in (select A.cat_id from product_category A where cat_name like '%t%'));"
```

5. Get all the order of the customer

```
select * from retail_store.order_1 where order_id in (select order_id from place_order_1 where c_id = 1);"
```

6. List of Employees working in a particular department

```
select * from employee where e_id in (select e_id from works_for where emp_type = '"+str(typ)+"');"
```

7.Insert a new Employee

```
insert into employee (first_name, middle_name,last_name,gender,
date_of_birth, email_id, phone, salary, date_of_joining,add_id) values
('"+str(f)+"'"+",'str(m)+"'"+",'"+str(la)+"','M'"+",'"+str(date)+"'"+",'"+str
(email)+"'"+",'"+str(phone)+"',"+str(salary)+",'"+str(jdate)+"'," +
str(max1)+");
```

8. Update Stock of the product

```
update product set stock = (stock-"+str(quantity[i])+") where p_id =
"+str(pid[i])
```

9. Get cost of the product with p_id

```
select (select min(cost) from supplied_by where P.p_id = p_id) as cost from
product P where P.p_id = 1
```

10. Mark Product as purchased

```

UPDATE add_to_cart_1 SET purchased='TRUE' where cart_id=1 and c_id = 1 and
purchased LIKE 'FALSE'; "
```

11. List of all the element form the cart of customer 1

```
"select * from add_to_cart_1 where c_id = 1 and purchased Like 'FALSE'"
```

12. Get the cost and supplier of the product

```
"select * from supplied_by where p_id=1 and cost in(select min(cost) from
supplied_by where p_id=1)"
```

13. Search in Customer table based on first name

```
select c_id, first_name, middle_name, last_name, gender, date_of_birth, email
from customer where first_name LIKE '%"+f+"%' "
```

Embedded Queries

1. ### Get the details of the product from cart

```
select P.p_id, P.product_name, (select min(cost) from supplied_by where
P.p_id = p_id) as cost, P.stock, A.quantity, A.time_added from product P,
```

```
add_to_cart_"+str(c_id)+" A where A.c_id = "+str(c_id)+" and A.p_id = P.P_id
and A.purchased = 'FALSE';"
```

2. List of order of a particular customer

```
"select * from retail_store.order_"+str(c_id)+" where order_id in (select
order_id from place_order_"+str(c_id)+" where c_id = "+str(c_id)+");"
```

3. Filter Product based on name

```
select P.p_id, P.product_name, (select min(cost) from supplied_by where
P.p_id = p_id) as cost, P.stock, P.manu_date, P.expiry_date from product P
where P.product_name like '%"+text+"%' "
```

4. Sort product based on descending order of name

```
select P.p_id, P.product_name, (select min(cost) from supplied_by where
P.p_id = p_id) as cost, P.stock, P.manu_date, P.expiry_date from product P
order by P.product_name desc; "
```

5. Filter Product in a range of cost

```
"select * from (select P.p_id, P.product_name, (select min(cost) from
supplied_by where P.p_id = p_id) as cost, P.stock, P.manu_date, P.expiry_date
from product P) as P where P.cost between "+str(lower_bound)+ " and
"+str(upper_bound)+"; "
```

6. Filter Product of a particular manufacturer

```
select P.p_id, P.product_name, (select min(cost) from supplied_by where
P.p_id = p_id) as cost, P.stock, P.manu_date, P.expiry_date from product P
where P.p_id IN (select M.p_id from manufactured_by M where M.man_id in
(select A.man_id from manufacturer A where man_name like '%"+text+"%'))"
```

7. Filter Product of a particular category

```
select P.p_id, P.product_name, (select min(cost) from supplied_by where
P.p_id = p_id) as cost, P.stock, P.manu_date, P.expiry_date from product P
where P.p_id IN (select C.p_id from categorization C where C.cat_id in
(select A.cat_id from product_category A where cat_name like '%t%'));"
```

TRIGGERS

1. **In add_to_cart**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `add_to_cart_BEFORE_INSERT` BEFORE
INSERT ON `add_to_cart` FOR EACH ROW BEGIN
set new.time_added = cast((select CURRENT_TIMESTAMP) as time);
END
```

2. **order**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `order_BEFORE_INSERT` BEFORE INSERT ON
`order` FOR EACH ROW BEGIN
set new.status = 'confirmed';
set new.ordered_on = cast((select CURRENT_TIMESTAMP) as time);
END
```

3. **address**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `address_BEFORE_INSERT` BEFORE INSERT
ON `address` FOR EACH ROW BEGIN
set new.add_id = (select max(add_id) from address)+1;
END
```

4. **customer**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `customer_BEFORE_INSERT` BEFORE INSERT
ON `customer` FOR EACH ROW BEGIN
set new.c_id = (select max(c_id) from customer)+1;
END
```

5. **employee**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `employee_BEFORE_INSERT` BEFORE INSERT
ON `employee` FOR EACH ROW BEGIN
set new.e_id = (select max(e_id) from employee)+1;
END
```

6. **manufacturer**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `manufacturer_BEFORE_INSERT` BEFORE
INSERT ON `manufacturer` FOR EACH ROW BEGIN
set new.man_id = (select max(man_id) from manufacturer)+1;
END
```

7. payment_credentials

```
CREATE DEFINER=`root`@`localhost` TRIGGER `payment_credentials_BEFORE_INSERT`
BEFORE INSERT ON `payment_credentials` FOR EACH ROW BEGIN
set new.paymentc_id = (select max(paymentc_id) from payment_credentials)+1;
END
```

8. Product

```
CREATE DEFINER=`root`@`localhost` TRIGGER `product_BEFORE_INSERT` BEFORE INSERT ON
`product` FOR EACH ROW BEGIN
set new.p_id = (select max(p_id) from product)+1;
END
```

9. product_Catergory

```
CREATE DEFINER=`root`@`localhost` TRIGGER `product_category_BEFORE_INSERT` BEFORE
INSERT ON `product_category` FOR EACH ROW BEGIN
set new.cat_id = (select max(cat_id) from product_category)+1;
END
```

10. supplier

```
CREATE DEFINER=`root`@`localhost` TRIGGER `supplier_BEFORE_INSERT` BEFORE INSERT
ON `supplier` FOR EACH ROW BEGIN
set new.s_id = (select max(s_id) from supplier)+1;
END
```

11. user

```
CREATE DEFINER=`root`@`localhost` TRIGGER `user_BEFORE_INSERT` BEFORE INSERT ON
`user` FOR EACH ROW BEGIN
set new.user_id = (select max(user_id) from user)+1;
END
```

INDEXING

Indexing the customer Table based on the first_name, middle_name, last_name to optimise the performance of a database.

1. **Index on first name of customer table:**

   create index first_name on customer (first_name);
2. **Index on Middle name of customer table:**

   create index middle_name on customer (middle_name);
3. **Index on Last name of customer table:**

   create index last_name on customer (last_name);