Online Retail Store

Group: 66 (Hemant Latiyan, Sahil Sahu, Parmesh Yadav, Mehul Goel)

Problem Statement:

Create a relational schema for an online retail store covering all aspects from suppliers to customers.

Stakeholders:

1. Customers
2. Suppliers
3. Employees

Entities:

- Customers
- Supplier
- Warehouse
- Employees
- Product
- Account
- Reviews
- Query

Attributes of entities along with primary keys:

- Customers (Cid, Full Name, Mobile no., email,City,State,PINCODE,DOB,age)
- Supplier(Sid, Company Name, Mobile No., City, State, Pincode)
- Warehouse(Wid, City, State, Pincode)

- Employees(Eid,,Full Name, Mobile No, email, DOJ, City, State, Pincode, salary, designation)
- Product(Pid, Name, Sid,Category,Price, stock, brand,Rating, offer)
- Account (Email-ID, password, premium subscription,Aid)
- Reviews (Pid, Star) {WEAK}
- Query (Cid, message, status, forum, time stamp) {WEAK}

Relationship between entities (with entity participation types):

1. Supplier -**SUPPLIES**- Product **(M-M)**
2. Product - **STORED IN** - Warehouse **(M-M)**
3. Employees - **WORK IN** - Warehouse **(M-1)**
4. Customer - **ORDERS -** Product **(M-M)**
5. Employee - **UPDATES** - Product (**M-M)**
6. Employee - **DEALS WITH - QUERIES - BY** - Customers **(1-M-M)**
7. Customer - **HAS AN** - Account **(1-1)**
8. Employee(Admin) - **MANAGES** - Warehouse **(1-1)**
9. Customer - **HAS A - REVIEW - OF** - Product **(1-1-M)**
10. Customer - **RETURNS** - Product **(M-M)**

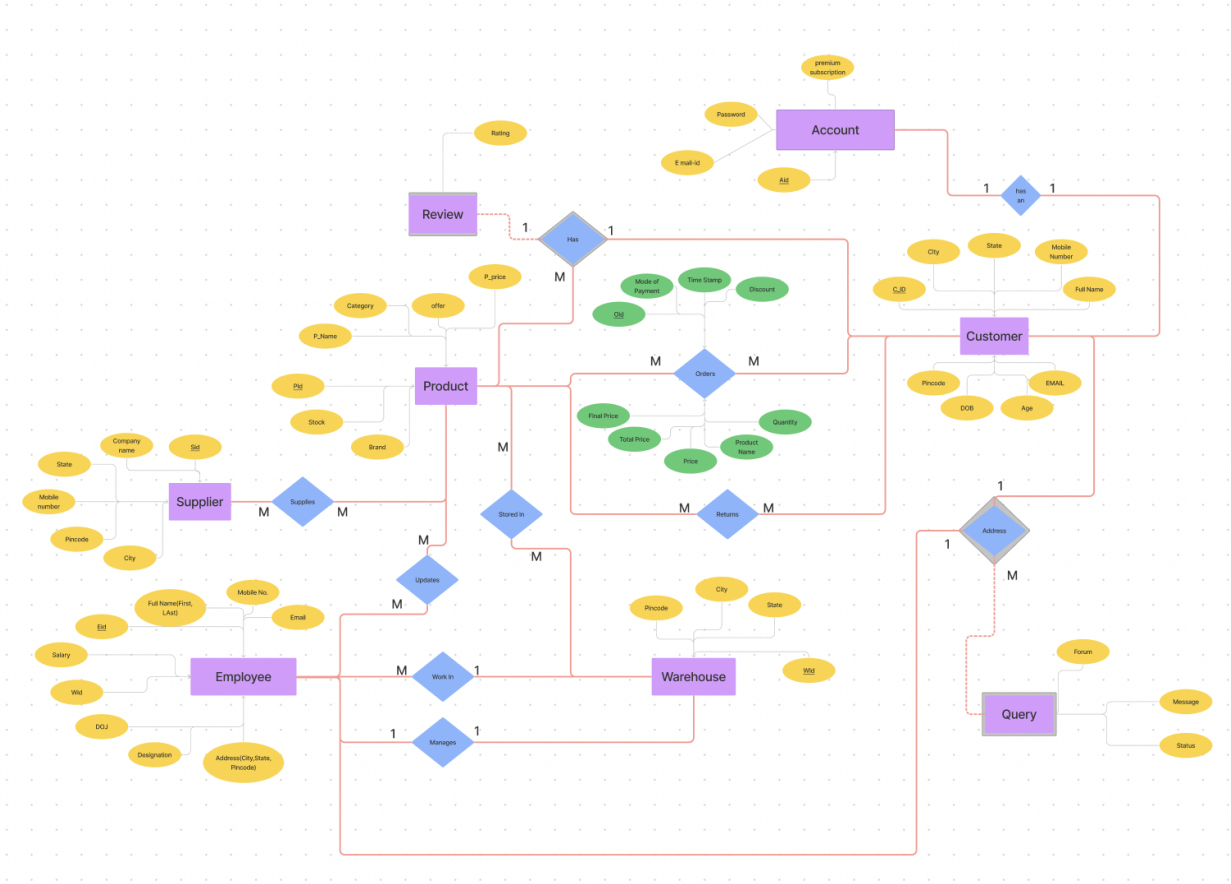
Relations:

Supplies(Sid, Pid)
Stored in(Wid, Pid)
Returns(Cid, Pid)
Orders(Oid, Cid, Pid, Quantity, Product Name.Price, Total Price, Discount, Final Price,Mode of Payment,Time Stamp)
Updates(Pid, Eid)
Has(Cid, Pid, rating)

Address(Cid, Eid, message, status, forum, time stamp(date_posted))

ER Diagram:

https://www.figma.com/file/XRgdcpYDpcuoY5IgGF3kRG/PROJECT_ER?node-id=0%3A1



Relational Schema:

- Customers (Cid, Full Name, Mobile no., email,City,State,PINCODE,DOB,age)
- Supplier(Sid, Company Name, Mobile No., City, State, Pincode)
- Warehouse(Wid, City, State, Pincode,Eid)
- Employees(Eid,,Full Name, Mobile No, email, DOJ, City, State, Pincode, salary, designation,Wid)
- Product(Pid, Name, Sid,Category,Price, stock, brand,Rating, offer)

- Account (Email-ID, password, premium subscription,Aid,Cid)
- Reviews (Pid, Star) {WEAK}
- Query (Cid, message, status, forum, time stamp) {WEAK}
- supplies(Sid, Pid)
- Stored in(Wid, Pid)
- Returns (Cid, Pid)
- Orders (Oid, Cid, Pid, Quantity, Product Name.Price, Total Price, Discount, Final Price,Mode of Payment,Time Stamp)
- Updates(Pid, Eid)
- Has (Cid, Pid, rating)
- Address (Cid, Eid, message, status, forum, time stamp(date_posted))

Weak Entity and why:

1. Query
2. Reviews

These are the weak entities because they cannot be uniquely identified based on their own attributes. Thus we have used cid as the foreign key in Query and Pid as the foreign key in reviews.

Ternary relationships and why:

1. Has
2. Address

Has is a ternary relationship because it connects customer, product and reviews.
Address is a ternary relationship because it connects customer to employee and queries

Constraints

- Customers: Primary - Cid
- Supplier: Primary - Sid
- Warehouse: Primary - Wid, Foreign - Eid
- Employee: Primary - Eid

- Product: Primary - Pid
- Account: Primary - Email-ID, Foreign Key - Aid
- Reviews: Primary - Pid
- Query: Primary - Cid
- Supplies: Primary - Sid, Foreign - Pid
- Stored in: Primary - Wid, Foreign - Pid
- Returns: Primary - Cid, Foreign - Pid
- Orders: Primary - Oid, Foreign - Cid, Pid
- Updates: Primary - Pid, Foreign - Eid
- Has: Primary - Cid, Foreign - Pid
- Address: Primary - Cid, Foreign - Eid

Queries:

Update the discount to 20% on all products with discount = 5%

Update Product set offer = offer + 15 where offer = 5

List all the customers who has Premium subscription

Select name from Customer where cid in (select cid from account where premium subscription =1)

Products from highest to lowest rating in a category c1

Select * from product where category=c1 order by rating DESC

Products from highest to lowest price in a category c1

Select * from Product where category = c1 order by price DESC

Update premium subscription of a customer with id a1

Update Account set premiumsubscription = 1where aid=a1 and premiumsubscription=0;

Update stock of a product

Update Product set stock = stock + 1 where pid = P1;

Update rating of a product

Update Product set rating = rating + 1 where pid = P1 and rating < 5;

List all employees from city C with designation D in increasing order of salary

Select * from Employee where city = C and designation = D order by salary;

List all products in a closest warehouse to a customer c1

Select pid from storedIn where wid in (select W.wid from customers C, Warehouse W where C.cid = 100 and W.pincode-C.pincode = (select min(W1.pincode-C'.pincode) from customers C' , warehouse W1 where C'.cid = 100));

Which mode of payment was used most

Select Mode_of_Payment,count(Mode_of_Payment) from orders group by Mode_of_Payment having count(Mode_of_Payment) in (select max(mycount) from (Select Mode_of_Payment, count(Mode_of_Payment) mycount from orders group by Mode_of_Payment) as T1)**;**

Advanced aggregation:

Calculate the sparsity of offers in products

Select stddev(product.offer) from product;

Variance of prices of all products

Select VARIANCE(price) from products

Rank product base on rating

Select pid, rank() over (order by rating)ratingRank from product;

Percent rank of employee in warehouse

Select eid, percent_rank()
Over (order by salary desc)
As 'percent_rank' from employee;

Triggers:

When an order is inserted the stock of that product with pid p1 in order is reduced by 1

Create trigger check_stock after insert on order
For each row
Update Product set stock = stock-1 where pid=p1;

If a customer removes their account then they are removed from the customer table (try to remove from everywhere (orders, queries)

Create trigger removeAcc
After delete
on Account
For each Row
Delete from CUSTOMERS where cid = old.cid;

When a customer returns product its stock increases by one

create trigger updateStock
after INSERT
on Returns
For each Row
Update Product set stock = stock + 1 where pid = new.pid;

When an employee get's a promotion their salary is updated

Create trigger updateSalary
After update
On employee
For each row
Update employee set salary = salary + 10000 where eid = old.eid and old.designation = 'SERVICE' and new.designation = 'MANAGER'

Indexing:

Product

```
CREATE INDEX Product_index
ON Product (pid, rating, price, offer);
```

Account

```
CREATE INDEX Account_index
ON Account (aid);
```

Orders

```
CREATE INDEX Order_index
ON Orders (Mode_of_Payment);
```

Supplies

```
CREATE INDEX Supplies_index
ON Supplies (sid);
```

Warehouse

```
CREATE INDEX Warehouse_index
ON Warehouse (pincode);
```

Customer

```
CREATE INDEX customer_index
ON Customers (cid, pincode);
```

Storedin

```
CREATE INDEX storedIn_index
ON Storedin (pid);
```

Employee

```
CREATE INDEX Employee_index
ON Employee (salary);
```

Grant Queries:

Allow only manager to do anything on the employee table

Grant select, insert, delete, update on Employee to

Allow customers to only read the product table

Grant select on product to ‘customer’;

View Queries

View for all pending queries

Create view v1 as Select eid, cid from address where status = 0;

View for average rating of all products

Create view v2 as Select avg(reviews.star), pid from reviews group by pid;

Sufficient and Valid Data:

https://www.mockaroo.com

Responsibility of each member: All members contributed equally in the project