WHOLESALE MANAGEMENT SYSTEM DATABASE

Wameedh Mohammed Ali
920678405
GitHub ID: wameedh

Milestone/Version	Date
M2V1	April 20, 2021
M1V2	April 20, 2021
M1V1	March 9, 2021

Table of Contents

Section I: Project Description

The project is a wholesale management system database. The system would manage the process of buying and selling products in wholesale on a database level. The system would make sure that customers can find products easily on the website without any discrepancy within the inventory of business, for example if an item is out of stock then the database should reflect that in real time. The system would have the customers information stored and protected. The database would be based on the businesses requirements such as what to sell, when to sell and the prices of products. It manages the type of purchase if it is online or in store. For online orders the database would provide shipping information such as tracking number, estimate of delivery time and shipping cost. The system would mantin a purchase history record of customers. Customers would be able to be members so that they collect reward points for every purchase they make. They also can be regular users in the system with an account but no membership. The system also manages admin types of users who are able to modify and make changes in the database that are specific to them only. For example, an admin can activate and deactivate an account in the database. This project is version one so it only takes case of the features that are considered priority for implementation. It could be developed to be bigger and more accommodating to complex problems that a wholesale management system faces. There are two main things that have been considered in the design and the build of this project, the one is simplicity, which means that the product was designed to be as simple as possible in terms of its component. I tried to minimize using extra entities that could be compromised for simplicity. The other aspect that has been considered in this project is efficiency, I tried to make sure that the main features of the design provide the solution for the main problem that this project is trying to solve. Which is a system that can provide an adoptable and responsive database interface for a website that deals with wholesale inventory and sale transactions.

Section II: Use Cases

Use Case Title:	Create an account
Actors:	Bob
Description:	Bob wants to have an account. He started the process of creating an account. He provided the following info: full name, username, email, date of birth, address, payment info, and a password. Then Bob submitted the request. Now Bob is a user in our database. Bob can add his family to his account. Bob is now a primary user. Bob can now place orders. Bob can now become a member.

Use Case Title:	Adding a user to an account
Actors:	Primary user
Description:	Bob is a primary user. He can add one or many users to his account. He started the process of adding a user to his account, he enter the new username, Alice. Bob wants to add another user, however he can't do that until he finishes the process of adding Alice then he can start a new process to add another user.

Use Case Title:	Become a member
Actors:	Bob
Description:	Bob already has an account in the database. He submitted a request to become a member. In the request he provided/confirmed payment info and agreed terms and condition of becoming a member. Now Bob has a track record of his reward points in the database.

Use Case Title:	Purchase a product

Actors:	Bob
Description:	Bob is a non-member user who is placing an order online. Bob searches for the item. He finds it then he adds it to his cart. Bob proceeds to check out. Bob must create an account so he can place the order. When Bob provides all the info he now can place the order and a confirmation email sent to him with the receipt and order number.

Use Case Title:	Updating inventory
Actors:	Admin
Description:	Product X gets restocked in the inventory, the admin would use his/her (or it in case it is an automated system) privileges to update the system to reflect the goods that have been added to the inventory.

Use Case Title:	Adding new product
Actors:	Admin
Description:	The admin starts the process of adding a new item to the products list. He/she must provide the following info, item name, category, pictures, serial number (optional), price, and adding any other necessary attributes.

Section III: Database Requirements (Business Rules)

1. General User
    1.1. A general user shall have only one unique user ID
    1.2. A general user shall create only one account using a unique email address.
    1.3. A general user shall have a full name.
    1.4. A general User shall have an email.
    1.5. A general User is a registered User.

2. Account
    2.1. An account shall have only one unique account ID.
    2.2. An account shall have only one credential.
    2.3. An account shall belong to only and only one user.
    2.4. A registered user shall have one or many profiles.
    2.5. An account shall have one or many addresses.
    2.6. An account shall have zero or many payment methods.
    2.7. An account shall have zero or one membership.
    2.8. An account shall have a creation date.

3. Credential
    3.1. A credential shall have account ID.
    3.2. An account shall have only one encrypted password.
    3.3. A credential shall have only one unique email.

4. Registered User
    4.1. A registered user shall have only one unique registered user ID
    4.2. A registered user shall have one or many roles.
    4.3. A registered user shall have only one shopping cart.
    4.4. A registered user shall use one or many devices.

5. Profile
    5.1. A profile shall have only one unique profile ID
    5.2. A profile shall belong to only one account.
    5.3. A profile shall have one avatar or profile picture.
    5.4. A profile shall have only one date of birth.
    5.5. A registered user shall have only one username.

6. Devices
    6.1. A device shall have one unique device ID
    6.2. A device shall have a type.
    6.3. A device shall have a name.
    6.4. A device shall login to one or many accounts.

7. Shopping Cart
    7.1. A shopping cart shall have only one unique cart ID
    7.2. A shopping cart shall belong to only one registered user.
    7.3. A shopping cart shall have product id.
    7.4. A shopping cart shall have an added date.

8. Role
    8.1. A role shall have only one unique role ID.
    8.2. A role shall belong to one or many registered users.
    8.3. A role is admin or customer.

9. Admin
    9.1. An admin shall have only one unique admin ID.
    9.2. An admin shall add one or many products to the inventory
    9.3. An admin shall remove one or many products from the inventory.
    9.4. An admin can make one or many registered users members.
    9.5. An admin can update user info.
    9.6. An admin can add or remove a membership from an account.

10. Costumer
    10.1. A customer shall have one unique customer ID.
    10.2. A customer can buy one or many products.
    10.3. A customer shall be able to redeem reward points.

11. Membership
    11.1. A membership shall have one unique membership ID.
    11.2. A membership shall have reward points.
    11.3. A membership shall belong to only one account.
    11.4. A membership shall have one renewal date.

12. Payment Method
    12.1. A payment method shall have only one unique payment ID.
    12.2. A payment method shall belong to only one account.
    12.3. A payment method is a bank account or credit card.
    12.4. A payment method shall have one or many billing addresses.

13. Bank account
    13.1. A bank account shall have one unique bank account ID.
    13.2. A bank account shall have a routing number.
    13.3. A bank account shall have a bank account number.

14. Credit Card
    14.1. A credit card shall have one unique credit card ID

14.2. A credit card shall have a full name.
14.3. A credit card shall have a card number.
14.4. A credit card shall have CVV.
14.5. A credit card shall have only one expiration date.

15. Address
15.1. An Address shall have only one unique address ID.
15.2. An Address shall belong to zero or many accounts
15.3. An Address shall have only one zip code
15.4. An Address shall have only one number.
15.5. An Address shall have only one street.
15.6. An Address shall have only one state.
15.7. An Address shall have only one country

16. Inventory
16.1. An inventory shall have only one unique inventory ID.
16.2. An inventory shall have one or many products.
16.3. An inventory shall be managed by a registered user with an admin role.

17. Product
17.1. A Product shall have only one unique product ID.
17.2. A Product shall have only one price.
17.3. A Product shall have a name.
17.4. A Product shall have only one description.
17.5. A Product shall have images .
17.6. A Product can be bought by a registered User.
17.7. A Product shall have weight.
17.8. A Product shall have quantity.
17.9. A Product is an Electronics, Clothes, Food, Beauty Products, and Health Products.

18. Electronics
18.1. Electronics Shall have only one unique Electronics ID.
18.2. Electronics is a product
18.3. Electronics shall have a serial number.
18.4. Electronics shall have a model date.
18.5. Electronics shall have brand name.

19. Clothes
19.1. Clothes Shall have only one unique Clothes ID.
19.2. Clothes is a product
19.3. Clothes shall have one or many sizes
19.4. Clothes shall have one or many size type (male, femail, kids, etc).

20. Food
    20.1. Food Shall have only one unique food ID.
    20.2. Food is a product
    20.3. Food shall have only one expiration date.
    20.4. Food shall have food type.

21. Beauty Products
    21.1. Beauty Products Shall have only one unique Beauty Products ID.
    21.2. Beauty Products is a product
    21.3. Beauty Products shall have one or many sizes
    21.4. Beauty Products shall have type.

22. Health Products
    22.1. Health Products Shall have only one unique Health Products ID.
    22.2. Health Products is a product
    22.3. Health Products shall have only one expiration date.
    22.4. Health Products shall have type.

Section IV: Detailed List of Main Entities, Attributes and Keys

1. General User (Strong)
    - user_id: key, numeric
    - full name: alphanumeric, composite
        1. First name
        2. Last name
    - email: key, alphanumeric

2. Account (Weak)
    - account_id: key, numeric
    - User_id: weak key, numeric
    - creation date: compuset, date

3. Credential (Weak)
    - account_id: key, numeric
    - password: alphanumeric
    - email: key, alphanumeric

4. Registered User (Strong)
    - Registered_user_id: key, numeric
    - User_id: key, numeric
    - role_id: weak key, numeric

5. Profile (Weak)
    - Profile_id: key, numeric
    - Avatar_link: alphanumeric
    - DoB: date
    - Username: alphanumeric
    - account_id: weak key, numeric

6. Devices (Strong)
    - Device_id: key, numeric
    - Type: alphanumeric
    - Name: alphanumeric

7. Shopping Cart (Weak)
    - Shopping_cart_id: key, numeric
    - product_id: weak key, numeric
    - date_added: date‹ TIMESTAMP
    - Reg_user_id: weak key, numeric

8. Role (Strong)
    - role_id: key, numeric

9. Admin (weak)
    - admin_id: key, numeric
    - creation_date: date
    - description: alphanumeric
    - role_id: weak key, numeric

10. Costumer (weak)
    - costumer_id: key, numeric
    - role_id: weak key, numeric
    - creation_date: date

11. Membership (weak)
    - membership_id: key, numeric
    - reward_points: numeric
    - creation_date: date
    - renewal_date: date
    - account_id: weak key, numeric
12. Rule (Strong)
    - Rule_id: key, numeric
13. Action (Weak)
    - Action_id key, numeric
    - Description: alphanumeric

14. Payment Method (Strong)
    - payment_id: key, numeric
    - address_id: weak key, numeric

15. Bank Account (weak)
    - Bank_acct_id: key, numeric
    - routing_number: numeric
    - acct_number: numeric
    - payment_id: weak key, numeric

16. Credit Card (weak)
    - credit_card_id: key, numeric
    - full name: compuset, alphanumeric
    - card_number: numeric
    - cvv: numeric
    - expiration_date: compuset, date
    - payment_id: weak key, numeric

17. Address (Strong)
    - address_id: key, numeric

- zip_code: numeric
- number: numeric
- street: alphanumeric
- city: alphanumeric
- state: alphanumeric
- country: alphanumeric

18. Inventory (Strong)
    - inventory_id: key, numeric
    - description: alphanumeric

19. Product (Strong)
    - product_id: key, numeric
    - price: numeric
    - name: alphanumeric
    - description: alphanumeric
    - wight: numeric
    - quantity: numeric
    - Image_id: weak key, numeric
    - Color_id: weak key, numeric
    - Size_id: weak key, numeric
    - Inventory_id: weak key, numeric

20. Images (Strong)
    - Image_id: key, numeric
    - Name: alphanumeric
    - Width: numeric
    - Hight: numeric
    - Product_id: weak key, numeric
21. CDN (Strong)
    - CDN_id: key, numeric
    - Name: alphanumeric

22. Color (Strong)
    - Color_id: key, numeric
    - Name: alphanumeric
    - Hex_value: alphanumeric

23. Size (Strong)
    - Size_id: key, numeric
    - Size_name: alphanumeric
    - Size_desc: alphanumeric
    - Size_value: numeric
24. Category (Strong)

- category_id: key, numeric
- Name: alphanumeric

25. Electronics (weak)
    - electronics_id: key, numeric
    - serial_num: numeric
    - model_year: year
    - brand: alphanumeric
    - category_id: weak key, numeric

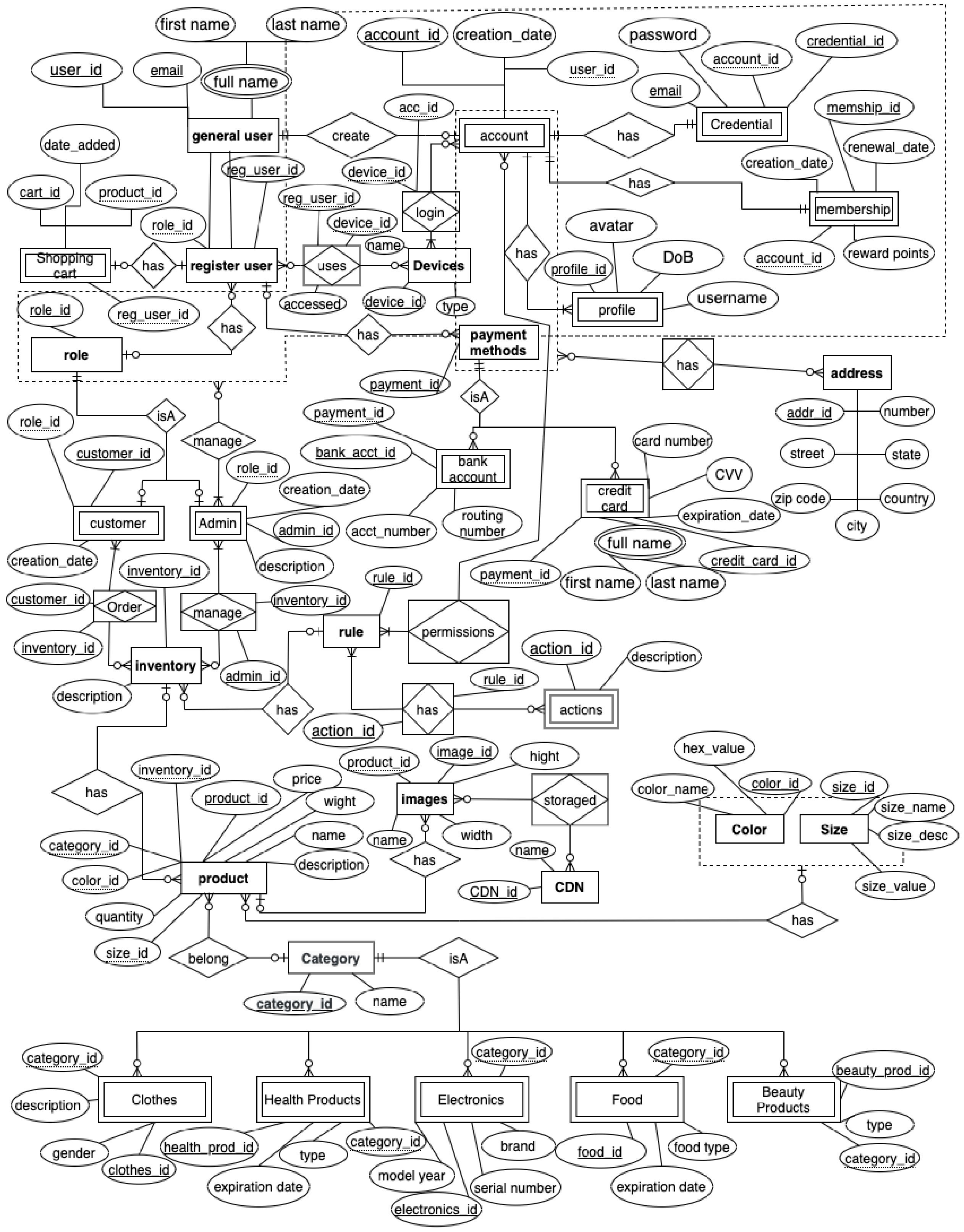
26. Clothes (weak)
    - clothes_id: key, numeric
    - gender: alphanumeric
    - description: alphanumeric
    - category_id: weak key, numeric

27. Food (weak)
    - food_id: key, numeric
    - expiration_date: compuset, date
    - Food_type: alphanumeric
    - category_id: weak key, numeric

28. Beauty Products (weak)
    - beauty_id: key, numeric
    - type: alphanumeric
    - category_id: weak key, numeric

29. Health Products (weak)
    - health_prod_id: key, numeric
    - expiration_date: date, TIMESTAMP
    - type: alphanumeric
    - category_id: weak key, numeric

Section V: Entity Relationship Diagram (ERD)



Section VI: Testing Table

Rule	Entity A	Relation	Entity B	Cardinalit y	Pass/Fail	Error Description
1	General User	Creates	Account	1-to-1	Pass	None
2	Account	Has	credential	1-to-M	Fail	Users can have an account with only one unique email. There can't be many credentials.
3	General User	IsA	Registered User	1-to-1	Pass	None
4	Registered User	Uses	Devices	1-to-M	Pass	None
5	Devices	Login	Account	M-to-M	Pass	None
6	Account	Has	Profile	1-to-M	Pass	None
7	Account	Has	Membership	1-to-1	Pass	None
8	Account	Has	Payment method	1-to-M	Pass	None
9	Account	Has	Address	M-to-M	Pass	None
10	Registered User	Has	Shopping cart	1-to-1	Pass	None
11	Registered Use	Has	Role	M-to-M	Pass	None
12	Role	IsA	Customer	1-to-1	Pass	None
13	Role	IsA	Admin	1-to-1	Pass	None
14	Customer	Order	Inventory	M-to-M	Pass	None
15	Admin	Manage	Inventory	M-to-M	Pass	None
16	Payment method	IsA	Bank account	1-to-M	Pass	None

17	Payment method	IsA	Credit card	1-to-M	Pass	None
18	Payment method	Has	Address (billing address)	M-to-1	Fail	Payment method could have many credit cards or bank accounts which means there could be many different billing addresses.
19	Inventory	Has	product	1-to-M	Pass	None
20	product	IsA	Clothes	1-to-M	Pass	None
21	product	IsA	Health Products	1-to-M	Pass	None
22	product	IsA	Electronics	1-to-M	Pass	None
23	product	IsA	Food	1-to-M	Pass	None
24	product	IsA	Beauty Products	1-to-M	Pass	None
25	Admin	Manage	Account and related entities	M-to-M	Pass	None
26	Admin	Manage	Role	M-to-M	Pass	None

Section VII: Database Model/EER

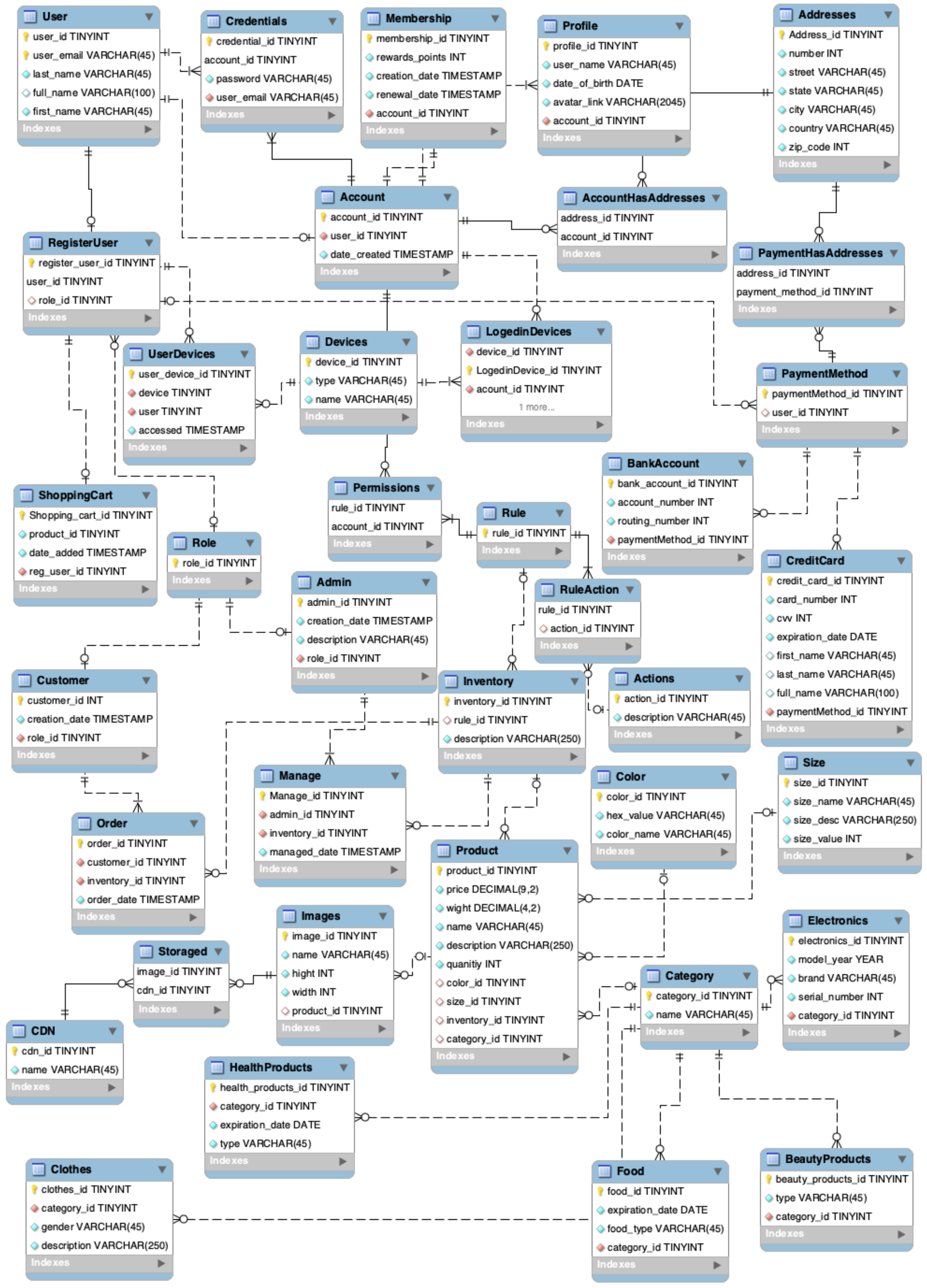


Table	FK	ON DELETE	ON UPDATE	Comment
Credentials	account	CASCADE	CASCADE	If the account gets deleted then the credentials associated with it should be deleted as well.
Credentials	user	CASCADE	CASCADE	If user gets deleted then the credentials associated with it should be deleted as well.
Account	user	CASCADE	CASCADE	User should be deleted and updated if the account gets deleted or updated.
AccountHasAddresses	account	CASCADE	CASCADE	If account get updated or deleted then the reference to it in AccountHasAddresses table should have the same effect.
AccountHasAddresses	Addresses	CASCADE	CASCADE	A reference to addresses in AccountHasAddresses table would update or deleted in according with what happen to its FK in Addresses.
Admin	Role	CASCADE	CASCADE	If role got deleted admin gets deleted as well and on update the relationship is the same.
BankAccount	paymentMethod	CASCADE	CASCADE	If paymentMethod gets deleted then no need for its reference to be in BankAccount table so it gets deleted as well and same for update.
CreditCard	paymentMethod	CASCADE	CASCADE	CreditCard is a paymentMethod If paymentMethod gets deleted then no need for its reference to be in BankAccount table so it gets deleted as well and same for update.
BeautyProducts	category	CASCADE	CASCADE	BeautyProducts is a category so if we delete category the we delete BeautyProducts and the same for update.
Clothes	category	CASCADE	CASCADE	Clothes is a category so if we delete category the we delete Clothes and the same for update.
Electronics	category	CASCADE	CASCADE	Electronics is a category so if we delete category the we delete Electronics and the same for update.

Food	category	CASCADE	CASCADE	Food is a category so if we delete category the we delete Food and the same for update.
HealthProducts	category	CASCADE	CASCADE	HealthProducts is a category so if we delete category the we delete HealthProducts and the same for
Customer	Role	CASCADE	CASCADE	Role is a customer. If role gets deleted customer gets deleted as well and same with update.
Images	Product	SET NULL	CASCADE	If Product gets deleted we need to keep the images so we set to null until it would be reused. But we update images if product gets updated.
Inventory	Rule	SET NULL	CASCADE	If a rule gets deleted from Inventory, the Inventory that had that rule will have no role until a new one is assigned to it.
LogedinDevices	account	CASCADE	CASCADE	If account gets deleted the LogedinDevices that had that account gets deleted as well and updated in the same manner.
LogedinDevices	devices	CASCADE	CASCADE	If devices gets deleted the LogedinDevices that had that account gets deleted as well and updated in the same manner.
Manage	admin	CASCADE	CASCADE	If admin is deleted then its reference in Manage is deleted as well.
Manage	inventory	CASCADE	CASCADE	If inventory is deleted then its reference in Manage is deleted as well.
Membership	account	CASCADE	CASCADE	If account gets deleted then the membership associated with it in membership table gets deleted as well. Same for update.
Order	customer	CASCADE	CASCADE	If customer is deleted its reference in order table is deleted as well. Same with update.
Order	inventory	CASCADE	CASCADE	If inventory is deleted its reference in order table is deleted as well. Same with update.
PaymentMethod	registerUser	CASCADE	CASCADE	If registerUser gets deleted the PaymentMethod that belongs to that registerUser gets deleted as well and updated in the same manner.

PaymentHasAddresses	Addresses	CASCADE	CASCADE	When address in Addresses table is deleted then the its reference in PaymentHasAddresses table is deleted as well. And the same for update.
PaymentHasAddresses	paymentMethod	CASCADE	CASCADE	When paymentMethod in PaymentMethod table is deleted then the its reference in PaymentHasAddresses table is deleted as well. And the same for update.
Permissions	Rule	CASCADE	CASCADE	If Rule is deleted its reference in Permissions table is deleted as well. Same with update.
Permissions	Account	CASCADE	CASCADE	If account is deleted then the permissions that belong to it in Permissions table gets deleted as well. Same for update.
Product	color	SET NULL	CASCADE	If color is deleted then the reference of it in the product table would be set to Null until it get assigned a new value or color. For update it gets updated on both table whenever it’s updated in Color.
Product	size	SET NULL	CASCADE	If size is deleted then the reference of it in the product table would be set to Null until it get assigned a new value or size. For update it gets updated on both table whenever it’s updated in Size.
Product	inventory	SET NULL	CASCADE	If inventory gets deleted then the the product that was holding that inventory will belong to no inventory until a new one is assigned.
Product	category	CASCADE	CASCADE	If category is deleted then product gets deleted as well and same for update.
Profile	account	CASCADE	CASCADE	If an account is deleted, then the profile for that account should be deleted as well.
RegisterUser	user	CASCADE	CASCADE	RegisterUser is a user, if user is deleted then the RegisterUser of user should be deleted as well.
RegisterUser	role	SET NULL	CASCADE	If role gets deleted then RegisterUser that was holding that role will have no role until a new one is assigned to it.

RuleAction	rule	CASCADE	CASCADE	If a rule is deleted, then the reference to it in the RuleAction table should be deleted as well.
RuleAction	action	SET NULL	CASCADE	If action gets deleted then the rule that was associated with that action will have no action until a new one is assigned to it.
ShoppingCart	RegisterUser	CASCADE	CASCADE	If a RegisterUser is deleted, then the ShoppingCart that belongs to it should be deleted as well. And updated if RegisterUser is updated.
Storaged	images	CASCADE	CASCADE	If an image is deleted, then the reference to it in Storaged should be deleted as well.
Storaged	cdn	CASCADE	CASCADE	If a cdn is deleted, then the reference to it in Storaged should be deleted as well.
UserDevices	user	CASCADE	CASCADE	If a user is deleted, then the reference to it in UserDevices should be deleted as well.
UserDevices	devices	CASCADE	CASCADE	If a device is deleted, then the reference to it in UserDevices should be deleted as well.

Section VIII: Forward Engineering

No need to add anything to the milestone 2 document for this section.

Section IX: Inserting Data

No need to add anything to the milestone 2 document for this section.

Section X: Testing

No need to add anything to the milestone 2 document for this section.

Section XI: Testing Table

Entity	SQLQuery	Pass/Fail	Error Description	Possible Solution
User	DELETE	Fail	update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
User	UPDATE	Fail	update a table without a WHERE that uses a KEY column	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;.
Account	DELETE	Pass	None	None
Account	UPDATE	Fail	Cannot add or update a child row.	This makes sense as we should not be able to update Account without updating User.
Addresses	DELETE	Fail	update a table without a WHERE that uses a KEY column	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;.
Addresses	UPDATE	Pass	None	None
AccountHasAddresses	DELETE	Pass	None	None
AccountHasAddresses	UPDATE	Pass	None	None
Actions	DELETE	Pass	None	None
Actions	UPDATE	Pass	None	None
Admin	DELETE	Pass	None	None
Admin	UPDATE	Pass	None	None
BankAccount	DELETE	Pass	None	None

BankAccount	UPDATE	Pass	None	None
BeautyProducts	DELETE	Pass	None	None
BeautyProducts	UPDATE	Pass	None	None
Category	DELETE	Pass	None	None
Category	UPDATE	Pass	None	None
CDN	DELETE	Pass	None	None
CDN	UPDATE	Pass	None	None
Clothes	DELETE	Pass	None	None
Clothes	UPDATE	Pass	None	None
Color	DELETE	Pass	None	None
Color	UPDATE	Fail	update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
Credentials	DELETE	Pass	None	None
Credentials	UPDATE	Pass	None	None
CreditCard	DELETE	Fail	update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
CreditCard	UPDATE	Fail	update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
Customer	DELETE	Pass	None	None
Customer	UPDATE	Pass	None	None
Devices	DELETE	Fail	update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql

				SET SQL_SAFE_UPDATES =0;
Devices	UPDATE	Fail	update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
Electronics	DELETE	Fail	Update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
Electronics	UPDATE	Pass	None	None
Food	DELETE	Pass	None	None
Food	UPDATE	Pass	None	None
HealthProducts	DELETE	Pass	None	None
HealthProducts	UPDATE	Pass	None	None
Images	DELETE	Pass	None	None
Images	UPDATE	Fail	Update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
Inventory	DELETE	Pass	None	None
Inventory	UPDATE	Pass	None	None
LogedinDevices	DELETE	Pass	None	None
LogedinDevices	UPDATE	Pass	None	None
Manage	DELETE	Pass	None	None
Manage	UPDATE	Pass	None	None
Membership	DELETE	Pass	None	None

Membership	UPDATE	Pass	None	None
Order	DELETE	Pass	None	None
Order	UPDATE	Pass	None	None
PaymentHasAddresses	DELETE	Pass	None	None
PaymentHasAddresses	UPDATE	Pass	None	None
Permissions	DELETE	Pass	None	None
Permissions	UPDATE	Pass	None	None
Product	DELETE	Fail	Update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
Product	UPDATE	Pass	None	None
Profile	DELETE	Fail	Update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
Profile	UPDATE	Pass	None	None
RegisterUser	DELETE	Pass	None	None
RegisterUser	UPDATE	Pass	None	None
Role	DELETE	Pass	None	None
Role	UPDATE	Pass	None	None
Rulle	DELETE	Pass	None	None
Rulle	UPDATE	Pass	None	None
RuleAction	DELETE	Pass	None	None

RuleAction	UPDATE	Fail	Unknown column 'rul_id' in 'where clause'	Change spelling of rul_id to rule_id
ShoppingCart	DELETE	Pass	None	None
ShoppingCart	UPDATE	Pass	None	None
Size	DELETE	Fail	Update a table without a WHERE that uses a KEY column.	Disable safe mode by adding the following line in WholesaleDB.sql SET SQL_SAFE_UPDATES =0;
Size	UPDATE	Fail	Incorrect data type.	Size_value should be an int
Storaged	DELETE	Pass	None	None
Storaged	UPDATE	Pass	None	None
UserDevices	DELETE	Pass	None	None
UserDevices	UPDATE	Pass	None	None
Product	Procedures: update_product_price	Pass	None	None
ShoppingCart	Procedures: getTotalOfCart	Pass	None	None
Inventory	Function: getInventoryTotal	Pass	None	None