PROJECT REPORT

Finished by : Eng / Mahmoud Mohamed Ahmed Hussain
E-mail : mahmoud.mohamed002@outlook.com

In the project I have created an amazon e-commerce database system which fulfils basic functional requirements for an e-commerce website. The database system fulfils the following functional requirements -

1. **A user can register**– A user can be a buyer or a seller
2. **A user can place order**- each order contains multiple products.
3. **A seller can add products**- this contains the details of a product.
4. **A buyer can give review**- a user can write reviews about a product.
5. **A user can add products to a Wishlist**- user can store the products which they like but not yet ready to buy
6. **A user can add products to a shopping cart**- users add to the shopping cart the products they want to buy.
7. **Product can be of multiple category**- it defines the category of a product like clothing, electronics etc
8. **Product can be carried by different carrier**- the shipping service through which a product can be shipped.
9. **A user can have multiple address and contact details**- users address and phone number saved in the account
10. **A user can have multiple Card info**- users saved card in the account
11. **A buyer can add images to its review**- each review can have images associated with it
12. **A seller can add images to its product** - each product can have multiple images associated with it, so taken in separate table.

RELATIONSHIPS

1. **User-contact_details:** each user can have multiple contact details saved, while each contact detail will have only 1 user linked. Thus, cardinality is 1: N
2. **Buyer-card_info:** each buyer can have many cards, while each card is associated with 1 buyer. Thus, cardinality is 1: N
3. **Buyer-order:** buyer places order. A buyer can place many orders, while each order is linked with only 1 buyer. Thus, cardinality is 1: N
4. **Order-product:** each order can contain many products, and each product can come in many orders. Thus, cardinality is M: N
5. **Seller-product:** seller sells products. Each seller can sell many products, while each product has only 1 seller. Thus, cardinality is 1: N
6. **Buyer-reviews:** buyer can write a review. Each buyer can write multiple reviews, while each review has only 1 buyer. Thus, cardinality is 1: N

7. **Review-products:** each review is for 1 product while each product can have many reviews. Thus, cardinality is 1: N
8. **Buyer-wishlist:** each buyer can have only 1 Wishlist, and each Wishlist has 1 buyer. Thus, cardinality is 1:1.
9. **Buyer- shopping cart:** each buyer has only 1 shopping cart while each shopping cart is associated with 1 buyer. Thus, cardinality is 1:1.
10. **Wishlist-product:** each Wishlist contains many products, and each product can be in many wish lists. Thus, cardinality is N:M
11. **Shopping_cart-product:** each cart contains many products, and each product can be in many carts. Thus, cardinality is N:M
12. **Product-category:** each product has only 1 category while each category has many products. Thus, cardinality is 1: N
13. **Product-carrier:** each product is shipped by 1 carrier, while each carrier ships many products.
14. **Review-review_images:** each review has many review images while each review image is linked with 1 review. Thus, cardinality is 1: N
15. **Product-product_images:** each product has many product images while each product image is linked with 1 product. Thus, cardinality is 1: N
16. **Order-card_info:** each order has a card linked with it, while each card can be used for many orders. Thus, cardinality is 1: N
17. **Order-contact details:** each order has 1 contact detail (address, phone), while each contact detail is linked with multiple orders. Thus, cardinality is 1: N

- Number 1:1 relationship = 2
- Number of N: M relationships = 3
- Number of 1: N relationships = 12
- Total Relationships = 17

ENTITY RELATION DIAGRAM

IMAGE URL: https://drive.google.com/drive/folders/1Z5GZC-_1xq8R5TdcaSBOElE96McfDIa7?usp=sharing

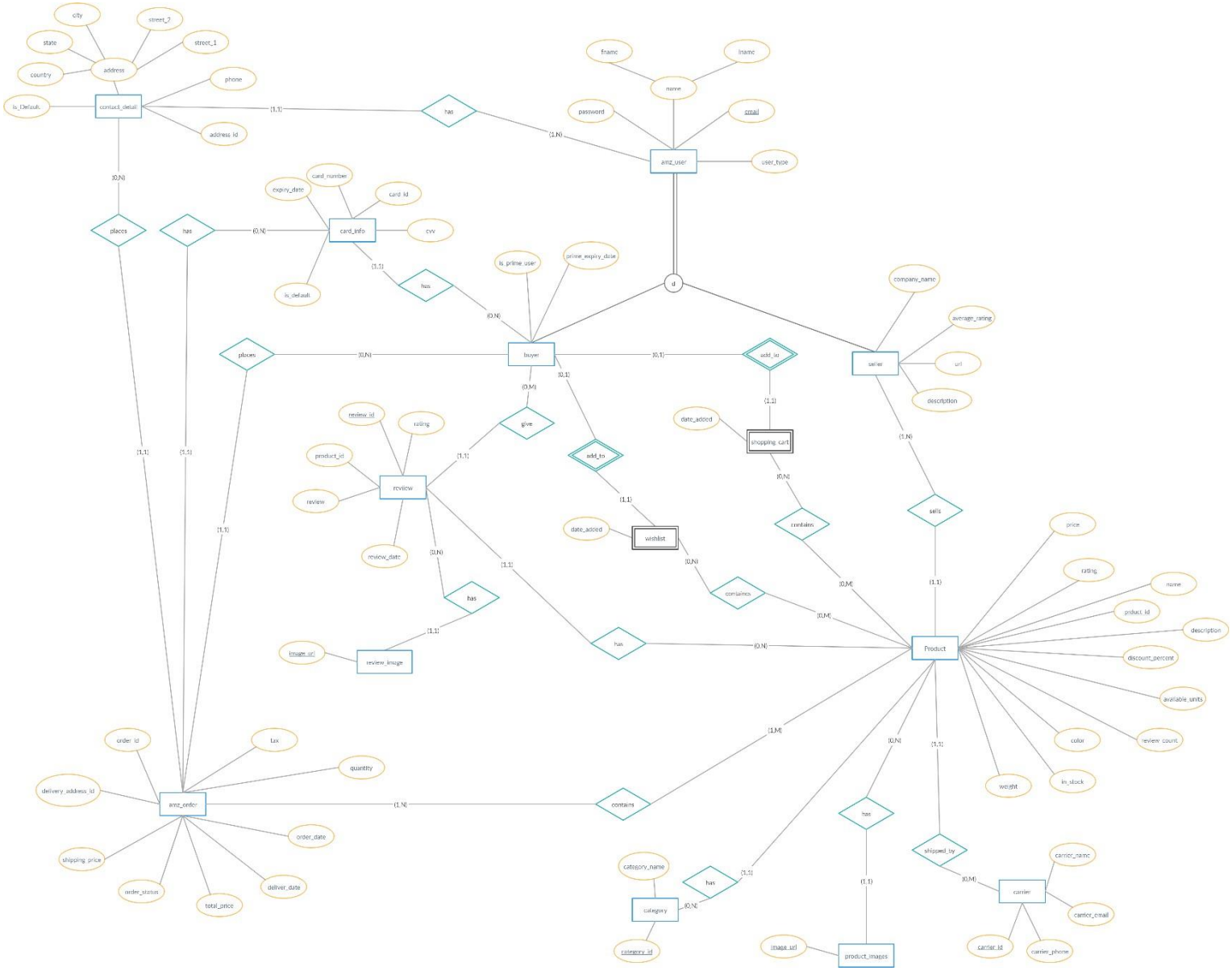

Fig 1.0 Amazon's Entity Relation diagram (big picture)

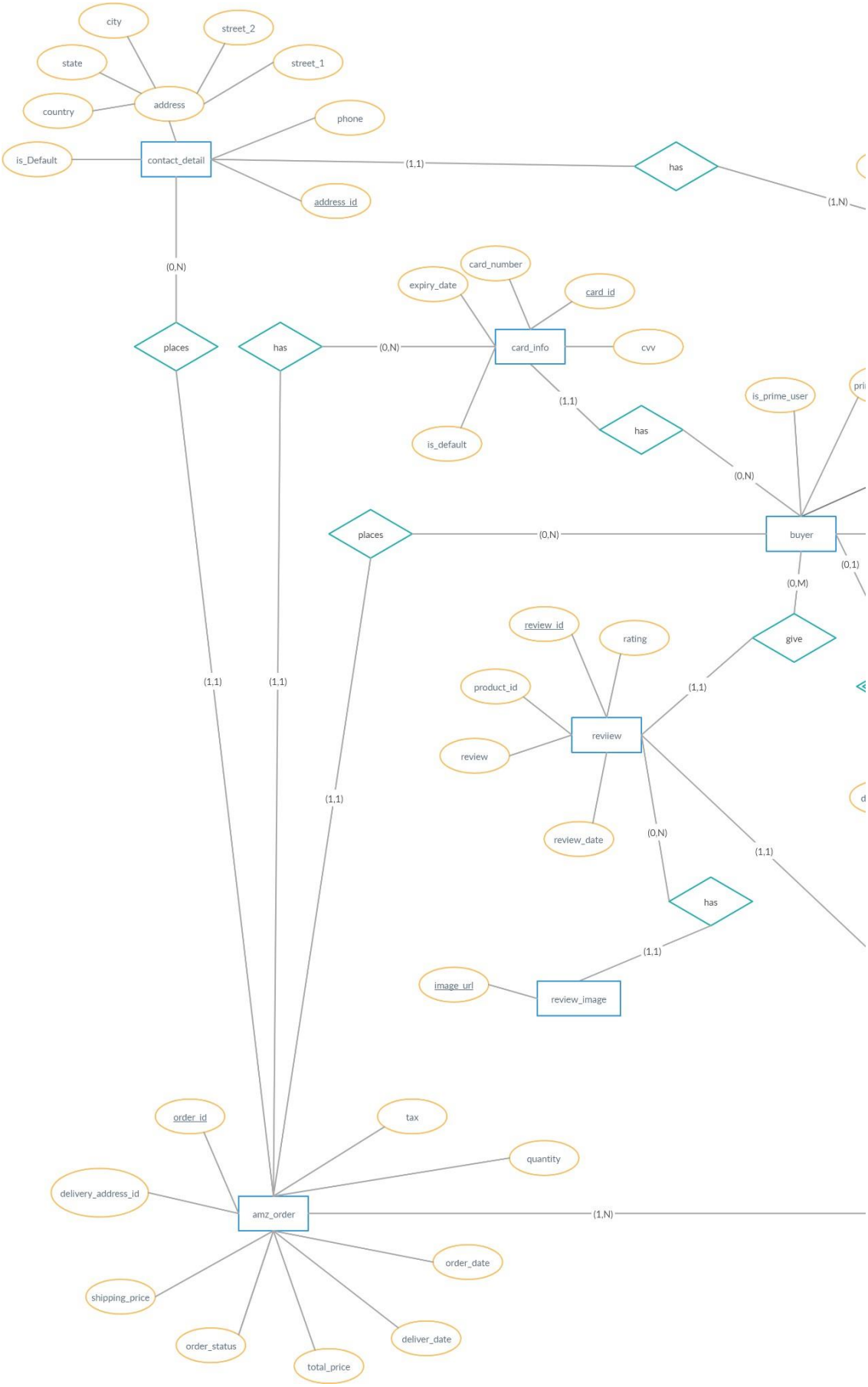

Fig 1.1 Amazon's Entity Relation diagram (left portion)

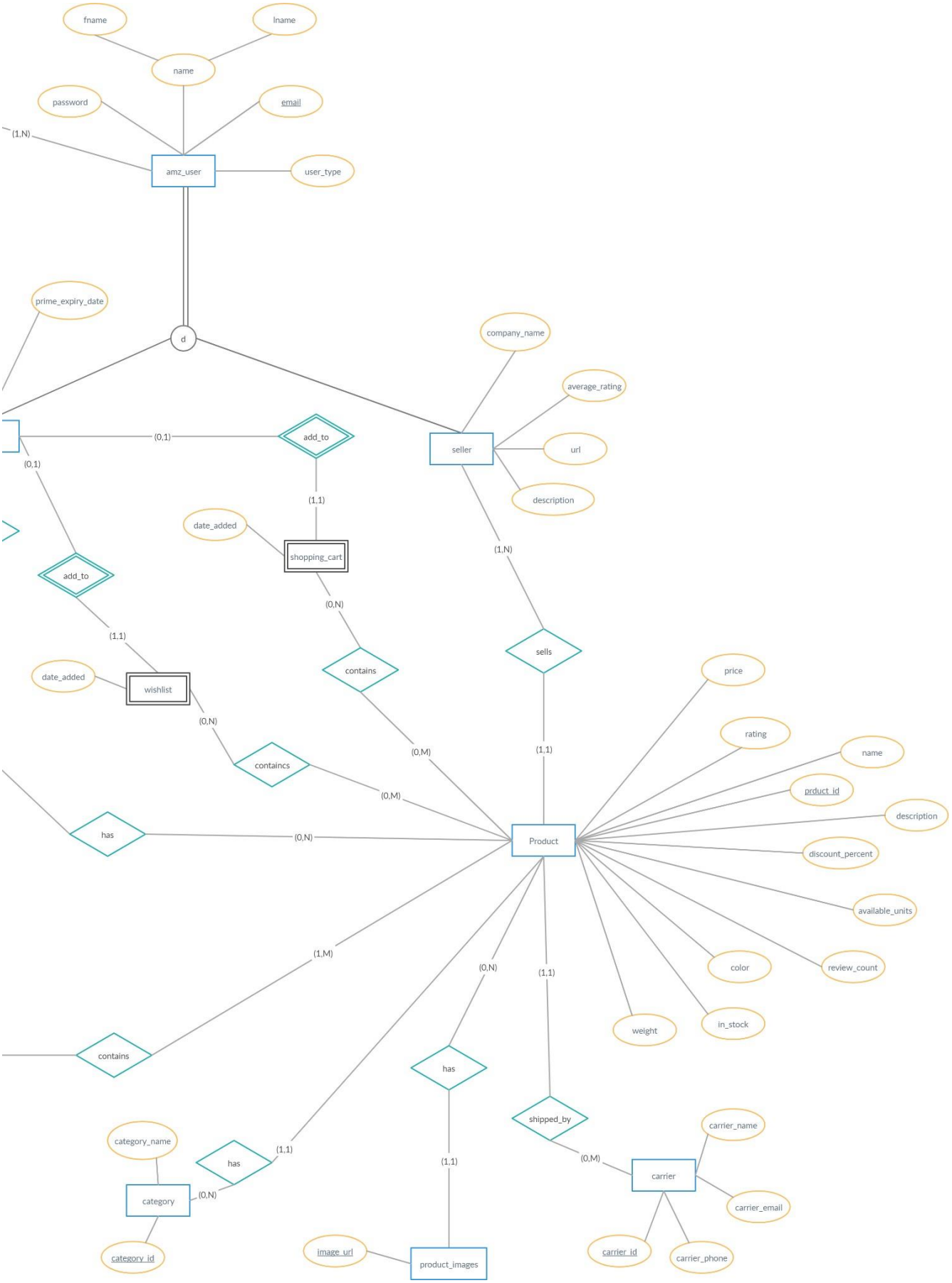

Fig 1.2 Amazon's Entity Relation diagram (right portion)

RELATIONAL SCHEMA

To map ER diagram into a relational schema, we considered the following mapping rules.

For each 1: 1 binary relationship, in the total participation entity add the primary key of the other entity as the foreign key.

For 1: N binary relationship, add to the entity on the N side the primary key of the other entity as the foreign key.

For M: N binary relationship, make a new entity with foreign key as the primary key of the two participating entities. Their combination forms the new primary key.

- In buyer table we have user_id as foreign key.
- In seller table we have user_id as foreign key.
- In product table we have seller_id, carrier_id and category_id as foreign keys.
- In order table we have buyer_id as foreign key.
- In card_info we have buyer_id as foreign key.
- In reviews we have buyer_id, product_id as foreign keys.
- In wishlist we have buyer_id as foreign key.
- In shopping cart we have buyer_id as foreign key.
- In contact_details we have user_id as foreign key.
- In review_images we have review_id as foreign key.
- In product_images we have product_id as foreign key.
- We make a new table name product_wishlist which as product_id and wishlist_id as foreign key.
- We make a new table name product_shopping_cart which as product_id and buyer_id as foreign key.
- We make a new table name product_order which as product_id and order_id as foreign key.

After converting the ER Model of our system into relational tables by following the strict guidelines of mapping, we analysed and confirmed that **the resultant tables do not violate any conditions of 3NF normal form**. Thus, the resultant relational tables formed are already in a 3NF normalised form.

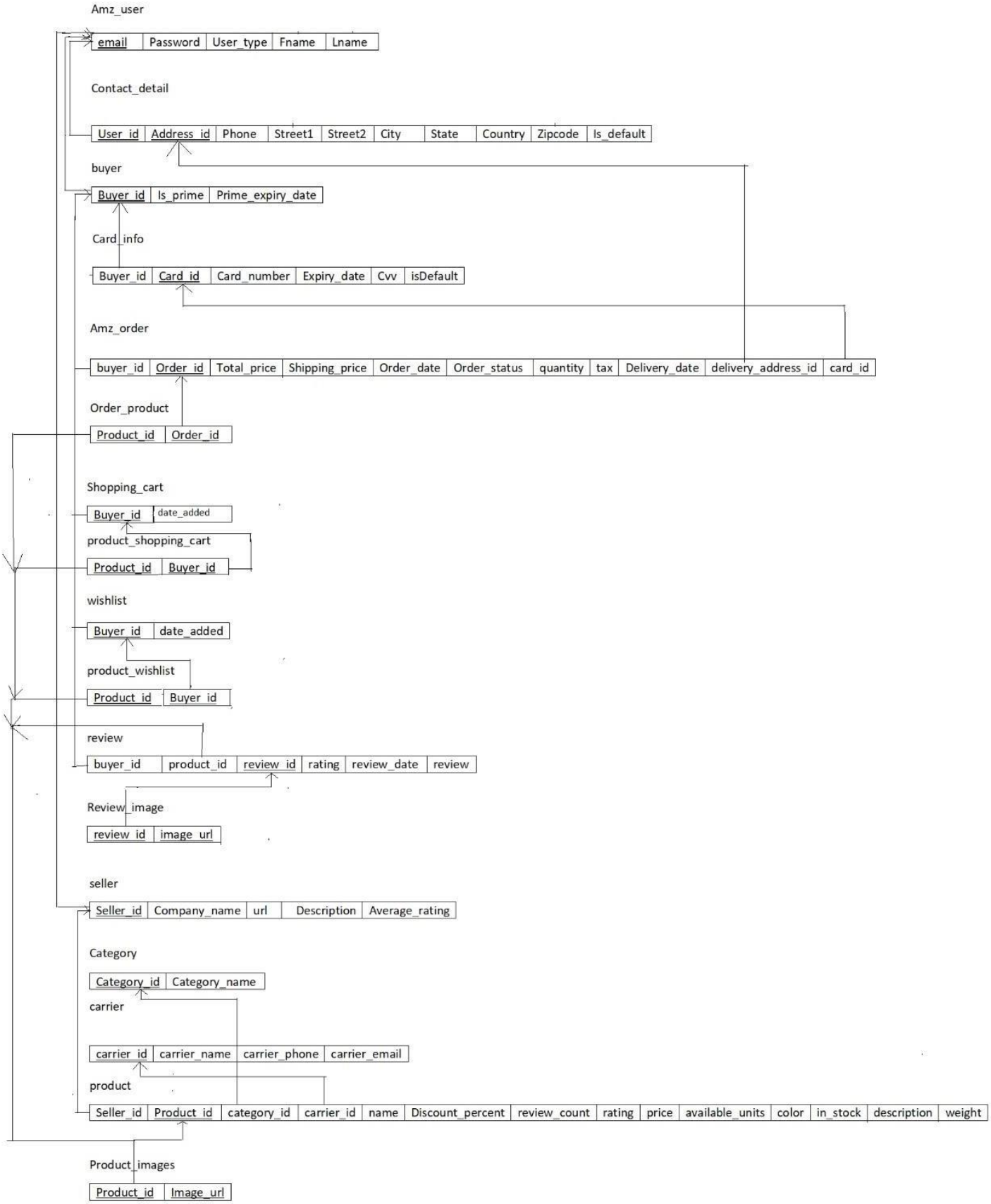

Fig 2 Relational Tables

SIGNIFICANT PROCEDURES

1. **Register buyer:** invoked by buyer responsible for
   a) registering user given email, fname, lname and password.
   b) registering the buyer itself setting its prime membership as false by default and prime member expiry date as Null.

2. **Register seller** has 2 responsibilities
   a) registering user given email, fname, lname and password.
   b) registering the seller itself given company name, url, description, setting its average rating as 2.5 default.

3. **Place order:** Given a buyer id, place order is responsible for the following –
   a) To iterate over the shopping cart of a particular buyer and remove each item from buyer's cart.
   b) While removing each product, sum up the price of each product to the total price for the order.
   c) If the user is **"prime user"** do not include shipping charges for that order.
   d) Fetch the default address set by the buyer from the list of addresses for that buyer from the contact_details tables.
   e) Fetch the default card details set by the buyer from the list of card details for that buyer from the card_info table.
   f) To make sure not to include certain products from the shopping car in the order table who's available unity is zero (in_stock bit is set to 0).
   g) To add an entry in the order table, containing details of the invoice (total price, total quantity of products in order, tax, shipping charge, card_details used for the order, delivery contact details used for the order)
   h) To invokes a trigger responsible for updating the available_units of each product being bought in that order.

4. **Give review:** A buyer can add a review by providing product_id, buyer_id, review, rating, and image_url (if any). It adds a review in the review table and image of it in the image table. After execution It invokes two triggers.
   a) update_product_rating
   b) updte_seller_rating.

5. **Add_contact_details:** adds details about address and the user's number. A user can add multiple contact details and can set a single to contact_details to be used by default.

6. **Add_card_info:** add cards information like card number, expiry_date etc. A user can add mupltiple card details and can set a single to card_info to be used by default.

7. **Add_prodcut:** adds a product sold a by a seller. It also asks for the image URL if any, A seller can upload multiple images for a product.

8. **Add_to_shopping_cart:** adds a product in the shopping cart for a buyer, given the buyer Id
9. **Add_to_wishlist:** adds a product to the wishlist of a buyer, given the buyer id.

10. **Update_membership:** update users prime membership information.
11. **Cancel_membership:** cancel user's prime membership

12. **Populate_product_categories:** adds all the available categories of product in it
13. **Populate_carrier_categories:** adds all the available carrier serviced responsible for delivering products.

IMPORTANT TRIGGERS

1. **Update available units:**
   - Trigger is invoked whenever a user places an order. "**Update available units"** is responsible for updating the value of available units for each product in the product table that is being ordered by a buyer.
   - The procedures iterate over all the entries of order_product table for a specific order and iteratively updates each products quantity in the product table.
   - If the quantity reaches 0, the product is marked as out of stock, setting its in_stock value as 0.

2. **Update_product_rating:**
   - It is responsible for updating the rating of a product every time a buyer gives a review by averaging the earlier rating with this buyers rating.
   - It also updates the count of rating given for that particular product.

3. **Update_seller_rating**
   - It is responsible for updating the rating of a seller every time a buyer gives a review to a product by averaging the earlier rating with this buyers rating.

4. **Remove_products_from_cart**
   - After placing an order by a buyer, the trigger is responsible for removing all the ordered products from the shopping cart of that particular buyer.

PROJECT IMPLEMTATION AND RESULTS

1. seller table registering the seller,

	SELLER_ID	COMPANY_NAME	URL	DESCRIPTION	AVERAGE_RATING	RATING_COUNT
1	kushagradar@gmail.com	kushagra Co and Co	www.kusharga.com	company of shoes	3.9	3
2	ruchisingh@gmail.com	ruchi Co and Co	www.ruchi.com	company of metals	2.5	0
3	anantprakash@gmail.com	anant Co and Co	www.anant.com	company of iphones	3	1

2. buyer table registering the buyer

	BUYER_ID	IS_PRIME	PRIME_EXPIRY_DATE
1	anshulpardhi@gmail.com	0	(null)
2	ashwanikashyap@gmail.com	0	(null)
3	gunjanagicha@gmail.com	0	(null)

3. updating the buyer's membership as prime user

	BUYER_ID	IS_PRIME	PRIME_EXPIRY_DATE
1	anshulpardhi@gmail.com	0	(null)
2	ashwanikashyap@gmail.com	0	(null)
3	gunjanagicha@gmail.com	1	28-NOV-20

4. contact_info table after adding contact details by a buyer

	USER_ID	ADDRESS_ID	STREET1	STREET2	CITY	STATE	COUNTRY	ZIPCODE	PHONE	IS_DEFAULT
1	anshulpardhi@gmail.com	1	7825 McCallum Blvd	Apt 007	Dallas	Texas	USA	752528	8888888888	1
2	gunjanagicha@gmail.com	2	7825 McCallum Blvd	Apt 1702	Dallas	Texas	USA	752528	8888888888	0
3	gunjanagicha@gmail.com	3	7825 McCallum Blvd	Apt 1702	Dallas	Texas	USA	752524	692309274	1

5. card_info table after adding card details for the payment by a buyer

	CARD_ID	CARD_NUMBER	EXPIRY_DATE	CVV	BUYER_ID	IS_DEFAULT
1	1	1234123412341234	09-DEC-23	666	gunjanagicha@gmail.com	1
2	2	2341234123412341	09-DEC-23	777	gunjanagicha@gmail.com	0
3	3	2341234123412341	09-DEC-23	777	anshulpardhi@gmail.com	1

6. product, product image table after uploading products by a seller on the amazon db

	PRODUCT_ID	NAME	SELLER_ID	PRICE	RATING	REVIEW_COUNT	CATEGORY_ID	DESCRIPTION	DISCOUNT_PERCENT	AVAILABLE_UNITS	COLOR	IN_STOCK	WEIGHT	CARRIER_ID
1	1	OnePlus 7	kushagradar@gmail.com	400	0	0	1	Best Phone	0	2	Blue	1	2	2
2	2	Harry Potter	kushagradar@gmail.com	15	0	0	2	Best Book	0	5	Black	1	8	2
3	3	Nike Shoes	anantprakash@gmail.com	50	0	0	3	Best shoes	0	2	yellow	1	5	1
4	4	I phone	anantprakash@gmail.com	500	0	0	1	Better than android	0	3	Black	1	2	3
5	5	Metal Detector	ruchisingh@gmail.com	20	0	0	1	Best metal detector	0	4	Grey	1	12	2

	REVIEW_ID	IMAGE_URL
1	1	www.my image.com
2	2	www.my image.com
3	3	www.my image.com
4	4	www.my image.com
5	5	www.my image.com
6	6	www.my image.com

7. wishlist, product_wishlist table after adding items to wishlist by a user

	BUYER_ID	DATE_ADDED
1	anshulpardhi@gmail.com	04-DEC-19
2	anshulpardhi@gmail.com	04-DEC-19
3	anshulpardhi@gmail.com	04-DEC-19
4	gunjanagicha@gmail.com	04-DEC-19

	PRODUCT_ID	BUYER_ID
1	1	anshulpardhi@gmail.com
2	4	anshulpardhi@gmail.com
3	3	anshulpardhi@gmail.com
4	2	gunjanagicha@gmail.com

8. shopping cart, product_shopping table after adding products to shopping cart by a buyer

	BUYER_ID	DATE_ADDED
1	anshulpardhi@gmail.com	04-DEC-19
2	anshulpardhi@gmail.com	04-DEC-19
3	gunjanagicha@gmail.com	04-DEC-19
4	gunjanagicha@gmail.com	04-DEC-19

	PRODUCT_ID	BUYER_ID
1	1	anshulpardhi@gmail.com
2	3	anshulpardhi@gmail.com
3	2	gunjanagicha@gmail.com
4	1	gunjanagicha@gmail.com

9. Multiple resultant tables after placing the order,
   - Order table, order product tables (shipping charge is set to zero if user is prime)

	ORDER_ID	BUYER_ID	CARD_ID	TOTAL_PRICE	ORDER_DATE	TAX	SHIPPING_PRICE	DELIVERY_ADDRESS_ID	DELIVERY_DATE	ORDER_STATUS	QUANTITY
1	1	qunjanaqicha@qmail.com	1	425	04-DEC-19	10	0	3	28-DEC-19	c	2
2	2	anshulpardhi@qmail.com	3	470	04-DEC-19	10	10	1	28-DEC-19	c	2

	ORDER_ID	PRODUCT_ID
1	1	1
2	1	2
3	2	1
4	2	3

   - Reduce the product's quantity. (if reached zero, set as out of stock)

	PRODUCT_ID	NAME	SELLER_ID	PRICE	RATING	REVIEW_COUNT	CATEGORY_ID	DESCRIPTION	DISCOUNT_PERCENT	AVAILABLE_UNITS	COLOR	IN_STOCK	WEIGHT	CARRIER_ID
1	1	OnePlus 7	kushaqradar@qmail.com	400	0	0	1	Best Phone	0	0	Blue	0	2	2
2	2	Harry Potter	kushaqradar@qmail.com	15	0	0	2	Best Book	0	4	Black	1	8	2
3	3	Nike Shoes	anantprakash@qmail.com	50	0	0	3	Best shoes	0	1	yellow	1	5	1
4	4	I phone	anantprakash@qmail.com	500	0	0	1	Better than android	0	3	Black	1	2	3
5	5	Metal Detector	ruchisingh@qmail.com	20	0	0	1	Best metal detector	0	4	Grey	1	12	2

   - Results of trigger after removing bought products from the shopping cart (empty cart)

	BUYER_ID	DATE_AD...

10. Multiple tables are updated after giving a review
    - Review table, product_review, review image

	REVIEW_ID	PRODUCT_ID	BUYER_ID	REVIEW	RATING	REVIEW_DATE
1	1	1	anshulpardhi@qmail.com	cool phone with great camera	5	04-DEC-19
2	2	3	anshulpardhi@qmail.com	good running shoes	3	04-DEC-19
3	3	2	qunjanaqicha@qmail.com	nice book	3.5	04-DEC-19
4	4	1	qunjanaqicha@qmail.com	okay phone	3	04-DEC-19
5	5	5	qunjanaqicha@qmail.com	doesnt work	1	04-DEC-19
6	6	5	qunjanaqicha@qmail.com	doesnt work at all	0	04-DEC-19

	REVIEW_ID	IMAGE_URL
1	1	www.my image.com
2	2	www.my image.com
3	3	www.my image.com
4	4	www.my image.com
5	5	www.my image.com
6	6	www.my image.com

    - Updated product rating based on the given review

	PRODUCT_ID	NAME	SELLER_ID	PRICE	RATING	REVIEW_COUNT	CATEGORY_ID	DESCRIPTION	DISCOUNT_PERCENT	AVAILABLE_UNITS	COLOR	IN_STOCK	WEIGHT	CARRIER_ID
1	1	OnePlus 7	kushaqradar@qmail.com	400	4	2	1	Best Phone	0	0	Blue	0	2	2
2	2	Harry Potter	kushaqradar@qmail.com	15	3.5	1	2	Best Book	0	4	Black	1	8	2
3	3	Nike Shoes	anantprakash@qmail.com	50	3	1	3	Best shoes	0	1	yellow	1	5	1
4	4	I phone	anantprakash@qmail.com	500	0	0	1	Better than android	0	3	Black	1	2	3
5	5	Metal Detector	ruchisingh@qmail.com	20	0	0	1	Best metal detector	0	4	Grey	1	12	2

	PRODUCT_ID	NAME	SELLER_ID	PRICE	RATING	REVIEW_COUNT	CATEGORY_ID	DESCRIPTION	DISCOUNT_PERCENT	AVAILABLE_UNITS	COLOR	IN_STOCK	WEIGHT	CARRIER_ID
1	1	OnePlus 7	kushaqradar@qmail.com	400	5	1	1	Best Phone	0	0	Blue	0	2	2
2	2	Harry Potter	kushaqradar@qmail.com	15	0	0	2	Best Book	0	4	Black	1	8	2
3	3	Nike Shoes	anantprakash@qmail.com	50	3	1	3	Best shoes	0	1	yellow	1	5	1
4	4	I phone	anantprakash@qmail.com	500	0	0	1	Better than android	0	3	Black	1	2	3
5	5	Metal Detector	ruchisingh@qmail.com	20	0	0	1	Best metal detector	0	4	Grey	1	12	2

	PRODUCT_ID	NAME	SELLER_ID	PRICE	RATING	REVIEW_COUNT	CATEGORY_ID	DESCRIPTION	DISCOUNT_PERCENT	AVAILABLE_UNITS	COLOR	IN_STOCK	WEIGHT	CARRIER_ID
1	1	OnePlus 7	kushaqradar@qmail.com	400	0	0	1	Best Phone	0	0	Blue	0	2	2
2	2	Harry Potter	kushaqradar@qmail.com	15	0	0	2	Best Book	0	4	Black	1	8	2
3	3	Nike Shoes	anantprakash@qmail.com	50	0	0	3	Best shoes	0	1	yellow	1	5	1
4	4	I phone	anantprakash@qmail.com	500	0	0	1	Better than android	0	3	Black	1	2	3
5	5	Metal Detector	ruchisingh@qmail.com	20	0	0	1	Best metal detector	0	4	Grey	1	12	2

	PRODUCT_ID	NAME	SELLER_ID	PRICE	RATING	REVIEW_COUNT	CATEGORY_ID	DESCRIPTION	DISCOUNT_PERCENT	AVAILABLE_UNITS	COLOR	IN_STOCK	WEIGHT	CARRIER_ID
1	1	OnePlus 7	kushaqradar@gmail.com	400	4	2	1	Best Phone	0	0	Blue	0	2	2
2	2	Harry Potter	kushaqradar@gmail.com	15	3.5	1	2	Best Book	0	4	Black	1	8	2
3	3	Nike Shoes	anantprakash@gmail.com	50	3	1	3	Best shoes	0	1	yellow	1	5	1
4	4	I phone	anantprakash@gmail.com	500	0	0	1	Better than android	0	3	Black	1	2	3
5	5	Metal Detector	ruchisingh@gmail.com	20	0.5	2	1	Best metal detector	0	4	Grey	1	12	2

- Updated seller rating based on review

	SELLER_ID	COMPANY_NAME	URL	DESCRIPTION	AVERAGE_RATING	RATING_COUNT
1	kushaqradar@gmail.com	kushaqra Co and Co	www.kusharqa.com	company of shoes	3.9	3
2	ruchisingh@gmail.com	ruchi Co and Co	www.ruchi.com	company of metals	2.5	0
3	anantprakash@gmail.com	anant Co and Co	www.anant.com	company of iphones	3	1

	SELLER_ID	COMPANY_NAME	URL	DESCRIPTION	AVERAGE_RATING	RATING_COUNT
1	kushaqradar@gmail.com	kushaqra Co and Co	www.kusharqa.com	company of shoes	3.9	3
2	ruchisingh@gmail.com	ruchi Co and Co	www.ruchi.com	company of metals	0.5	2
3	anantprakash@gmail.com	anant Co and Co	www.anant.com	company of iphones	3	1

SQL CODE FOR THE AMAZON DB SYSTEM

SOURCE CODE: https://drive.google.com/drive/folders/1Z5GZC-_1xq8R5TdcaSBOElE96McfDIa7?usp=sharing

- **Code to create tables and apply contraints**

```
CREATE TABLE amz_user (
   email       VARCHAR(255) PRIMARY KEY,
   fname       VARCHAR(255) NOT NULL,
   lname       VARCHAR(255),
   password    VARCHAR(30) NOT NULL,
   user_type   NUMBER(1) NOT NULL
);

CREATE TABLE contact_detail (
   user_id      VARCHAR(255) NOT NULL,
   address_id   INTEGER PRIMARY KEY,
   street1      VARCHAR(255) NOT NULL,
   street2      VARCHAR(255),
   city         VARCHAR(50) NOT NULL,
   state        VARCHAR(50) NOT NULL,
   country      VARCHAR(50) NOT NULL,
   zipcode      NUMBER(5) NOT NULL,
   phone        VARCHAR(20) NOT NULL,
   is_default   NUMBER(1) DEFAULT 0
);

CREATE TABLE card_info (
   card_id       INTEGER PRIMARY KEY,
   card_number   NUMBER(16) NOT NULL,
   expiry_date   DATE NOT NULL,
   cvv           NUMBER(3) NOT NULL,
```

```
  buyer_id      VARCHAR(255) NOT NULL,
  is_default    NUMBER(1)
);

CREATE TABLE buyer (
  buyer_id          VARCHAR(255) PRIMARY KEY,
  is_prime          NUMBER(1) DEFAULT 0,
  prime_expiry_date  DATE
);

CREATE TABLE seller (
  seller_id       VARCHAR(255) PRIMARY KEY,
  company_name    VARCHAR(255) NOT NULL,
  url             VARCHAR(255),
  description     VARCHAR(255),
  average_rating  NUMBER(2, 1) DEFAULT 2.5,
  rating_count    NUMBER DEFAULT 0
);

CREATE TABLE category (
  category_id    INTEGER PRIMARY KEY,
  category_name  VARCHAR(255) NOT NULL
);

CREATE TABLE product (
  product_id        INTEGER PRIMARY KEY,
  name              VARCHAR(255) NOT NULL,
  seller_id         VARCHAR(255) NOT NULL,
  price             NUMBER(10, 2) NOT NULL,
  rating            NUMBER(2, 1),
  review_count      INTEGER,
  category_id       INTEGER,
  description       VARCHAR(255),
  discount_percent  NUMBER(4, 2),
  available_units   INTEGER,
  color             VARCHAR(30),
  in_stock          NUMBER(1),
  weight            NUMBER(10, 2),
  carrier_id        INTEGER
);

CREATE TABLE product_image (
  product_id  INTEGER,
  image_url   VARCHAR(255),
```

```
   PRIMARY KEY ( product_id,
                 image_url )
);

CREATE TABLE shopping_cart (
   buyer_id     VARCHAR(255),
   date_added   DATE
);

CREATE TABLE product_shoppingcart (
   product_id   INTEGER,
   buyer_id     VARCHAR(255),
   PRIMARY KEY ( product_id,
                 buyer_id )
);

CREATE TABLE wish_list (
   buyer_id     VARCHAR(255),
   date_added   DATE
);

CREATE TABLE product_wishlist (
   product_id   INTEGER,
   buyer_id     VARCHAR(255),
   PRIMARY KEY ( product_id,
                 buyer_id )
);

CREATE TABLE amz_order (
   order_id             INTEGER PRIMARY KEY,
   buyer_id             VARCHAR(255) NOT NULL,
   card_id              INTEGER NOT NULL,
   total_price          NUMBER(10, 2),
   order_date           DATE,
   tax                  NUMBER(4, 2) DEFAULT 10,
   shipping_price       NUMBER(4, 2) DEFAULT 10,
   delivery_address_id  INTEGER,
   delivery_date        DATE,
   order_status         CHAR(1) NOT NULL,
   quantity             INTEGER NOT NULL
);

CREATE TABLE order_product (
   order_id     INTEGER,
```

```
   product_id   INTEGER,
   PRIMARY KEY ( order_id,
                 product_id )
);

CREATE TABLE review (
   review_id     INTEGER PRIMARY KEY,
   product_id    INTEGER NOT NULL,
   buyer_id      VARCHAR(255) NOT NULL,
   review        VARCHAR(1000),
   rating        NUMBER(2, 1),
   review_date   DATE
);

CREATE TABLE review_image (
   review_id   INTEGER,
   image_url   VARCHAR(255),
   PRIMARY KEY ( review_id,
                 image_url )
);

CREATE TABLE carrier (
   carrier_id      INTEGER PRIMARY KEY,
   carrier_name    VARCHAR(255) NOT NULL,
   carrier_phone   NUMBER(10) NOT NULL,
   carrier_email   VARCHAR(255) NOT NULL
);

ALTER TABLE contact_detail
   ADD CONSTRAINT contact_detail_user_id_fk FOREIGN KEY ( user_id )
      REFERENCES amz_user ( email )
         ON DELETE CASCADE;

ALTER TABLE card_info
   ADD CONSTRAINT card_info_buyer_id_fk FOREIGN KEY ( buyer_id )
      REFERENCES buyer ( buyer_id )
         ON DELETE CASCADE;

ALTER TABLE product
   ADD CONSTRAINT product_seller_id_fk FOREIGN KEY ( seller_id )
      REFERENCES seller ( seller_id )
         ON DELETE CASCADE;

ALTER TABLE product
```

```
  ADD CONSTRAINT product_category_id_fk FOREIGN KEY ( category_id )
    REFERENCES category ( category_id )
      ON DELETE CASCADE;

ALTER TABLE product
  ADD CONSTRAINT product_carrier_id_fk FOREIGN KEY ( carrier_id )
    REFERENCES carrier ( carrier_id )
      ON DELETE CASCADE;

ALTER TABLE product_image
  ADD CONSTRAINT product_image_product_id_fk FOREIGN KEY ( product_id )
    REFERENCES product ( product_id )
      ON DELETE CASCADE;

ALTER TABLE shopping_cart
  ADD CONSTRAINT shopping_cart_buyer_id_fk FOREIGN KEY ( buyer_id )
    REFERENCES buyer ( buyer_id )
      ON DELETE CASCADE;

ALTER TABLE product_shoppingcart
  ADD CONSTRAINT product_sc_buyer_id_fk FOREIGN KEY ( buyer_id )
    REFERENCES buyer ( buyer_id )
      ON DELETE CASCADE;

ALTER TABLE product_shoppingcart
  ADD CONSTRAINT product_sc_product_id_fk FOREIGN KEY ( product_id )
    REFERENCES product ( product_id )
      ON DELETE CASCADE;

ALTER TABLE wish_list
  ADD CONSTRAINT wishlist_buyer_id_fk FOREIGN KEY ( buyer_id )
    REFERENCES buyer ( buyer_id )
      ON DELETE CASCADE;

ALTER TABLE product_wishlist
  ADD CONSTRAINT product_wishlist_product_id_fk FOREIGN KEY ( product_id )
    REFERENCES product ( product_id )
      ON DELETE CASCADE;

ALTER TABLE product_wishlist
  ADD CONSTRAINT product_wishlist_buyer_id_fk FOREIGN KEY ( buyer_id )
    REFERENCES buyer ( buyer_id )
      ON DELETE CASCADE;
```

```
ALTER TABLE amz_order
   ADD CONSTRAINT order_buyer_id_fk FOREIGN KEY ( buyer_id )
      REFERENCES buyer ( buyer_id )
         ON DELETE CASCADE;

ALTER TABLE amz_order
   ADD CONSTRAINT order_card_id_fk FOREIGN KEY ( card_id )
      REFERENCES card_info ( card_id )
         ON DELETE CASCADE;

ALTER TABLE amz_order
   ADD CONSTRAINT order_delivery_address_id_fk FOREIGN KEY ( delivery_address_id )
      REFERENCES contact_detail ( address_id )
         ON DELETE CASCADE;

ALTER TABLE order_product
   ADD CONSTRAINT order_product_order_id_fk FOREIGN KEY ( order_id )
      REFERENCES amz_order ( order_id )
         ON DELETE CASCADE;

ALTER TABLE order_product
   ADD CONSTRAINT order_product_product_id_fk FOREIGN KEY ( product_id )
      REFERENCES product ( product_id )
         ON DELETE CASCADE;

ALTER TABLE review
   ADD CONSTRAINT review_product_id_fk FOREIGN KEY ( product_id )
      REFERENCES product ( product_id )
         ON DELETE CASCADE;

ALTER TABLE review
   ADD CONSTRAINT review_buyer_id_fk FOREIGN KEY ( buyer_id )
      REFERENCES buyer ( buyer_id )
         ON DELETE CASCADE;

ALTER TABLE review_image
   ADD CONSTRAINT review_image_review_id_fk FOREIGN KEY ( review_id )
      REFERENCES review ( review_id )
         ON DELETE CASCADE;
```

- **Code for stored procedure and triggers**

```
CREATE OR REPLACE PROCEDURE register_buyer (
```

```
    email      IN   VARCHAR,
    fname      IN   VARCHAR,
    lname      IN   VARCHAR,
    password   IN   VARCHAR
) AS
BEGIN
    INSERT INTO amz_user VALUES (
        email,
        fname,
        lname,
        password,
        0
    );

    INSERT INTO buyer VALUES (
        email,
        0,
        NULL
    );

END register_buyer;

-------------------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE register_seller (
    email            IN   VARCHAR,
    fname            IN   VARCHAR,
    lname            IN   VARCHAR,
    password         IN   VARCHAR,
    company_name     IN   VARCHAR,
    url              IN   VARCHAR,
    description_var  IN   VARCHAR
) AS
BEGIN
    INSERT INTO amz_user VALUES (
        email,
        fname,
        lname,
        password,
        1
    );

    INSERT INTO seller VALUES (
        email,
```

```
        company_name,
        url,
        description_var,
        2.5,
        0
    );

END register_seller;

-----------------------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE add_contact_details (
    user_id      IN  VARCHAR,
    address_id   IN  INTEGER,
    street1      IN  VARCHAR,
    street2      IN  VARCHAR,
    city         IN  VARCHAR,
    state        IN  VARCHAR,
    country      IN  VARCHAR,
    zipcode      IN  NUMBER,
    phone        IN  VARCHAR
) AS
BEGIN
    INSERT INTO contact_detail VALUES (
        user_id,
        address_id,
        street1,
        street2,
        city,
        state,
        country,
        zipcode,
        phone,
        0
    );

END add_contact_details;

-----------------------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE set_default_contact_details (
    contact_id  IN  INTEGER,
    buyer_id    IN  VARCHAR
) AS
```

```
BEGIN
   UPDATE contact_detail
   SET
      is_default = 1
   WHERE
      user_id = buyer_id
      AND address_id = contact_id;

END set_default_contact_details;
```

```
CREATE OR REPLACE PROCEDURE add_card_info (
   buyer_id      IN  VARCHAR,
   card_id       IN  INTEGER,
   card_number   IN  NUMBER,
   expiry_date   IN  DATE,
   cvv           IN  NUMBER
) AS
BEGIN
   INSERT INTO card_info VALUES (
      card_id,
      card_number,
      expiry_date,
      cvv,
      buyer_id,
      0
   );

END add_card_info;
```

```
CREATE OR REPLACE PROCEDURE set_default_card_info (
   card_id_var    IN  INTEGER,
   buyer_id_var   IN  VARCHAR
) AS
BEGIN
   UPDATE card_info
   SET
      is_default = 1
   WHERE
      buyer_id = buyer_id_var
      AND card_id = card_id_var;
```

```
END set_default_card_info;

-------------------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE add_product (
   product_id       IN  INTEGER,
   name             IN  VARCHAR,
   seller_id        IN  VARCHAR,
   price            IN  NUMBER,
   category_id      IN  INTEGER,
   description      IN  VARCHAR,
   available_units  IN  INTEGER,
   color            IN  VARCHAR,
   weight           IN  NUMBER,
   carrier_id       IN  INTEGER,
   image_url        IN  VARCHAR
)   AS
BEGIN
   INSERT INTO product VALUES (
      product_id,
      name,
      seller_id,
      price,
      0,
      0,
      category_id,
      description,
      0,
      available_units,
      color,
      1,
      weight,
      carrier_id
   );

   INSERT INTO product_image VALUES (
      product_id,
      image_url
   );

END add_product;

-------------------------------------------------------------------------------------------------------
```

```
CREATE OR REPLACE PROCEDURE add_to_shopping_cart (
   buyer_id     IN  VARCHAR,
   product_id   IN   INTEGER
) AS
BEGIN
   INSERT INTO shopping_cart VALUES (
      buyer_id,
      sysdate
   );

   INSERT INTO product_shoppingcart VALUES (
      product_id,
      buyer_id
   );

END add_to_shopping_cart;
```

```
CREATE OR REPLACE PROCEDURE add_to_wish_list (
   buyer_id     IN  VARCHAR,
   product_id   IN   INTEGER
) AS
BEGIN
   INSERT INTO wish_list VALUES (
      buyer_id,
      sysdate
   );

   INSERT INTO product_wishlist VALUES (
      product_id,
      buyer_id
   );

END add_to_wish_list;
```

```
CREATE OR REPLACE PROCEDURE give_review (
   review_id   IN  NUMBER,
   product_id   IN   INTEGER,
   buyer_id    IN    VARCHAR,
   review     IN  VARCHAR,
```

```
    rating      IN  NUMBER,
    image_url   IN  VARCHAR
) AS
BEGIN
    INSERT INTO review VALUES (
        review_id,
        product_id,
        buyer_id,
        review,
        rating,
        sysdate
    );

    INSERT INTO review_image VALUES (
        review_id,
        image_url
    );

END give_review;

-----------------------------------------------------------------------------------------------------

CREATE OR REPLACE TRIGGER update_product_rating AFTER
    INSERT ON review
    FOR EACH ROW
DECLARE
    new_rating        NUMBER(2, 1);
    review_count_old  INTEGER;
BEGIN
    SELECT
        review_count
    INTO review_count_old
    FROM
        product
    WHERE
        product_id = :new.product_id;

    new_rating := :new.rating;
    UPDATE product
    SET
        rating = ( ( rating * review_count_old ) + new_rating ) / ( review_count_old + 1 ),
        review_count = review_count_old + 1
    WHERE
        product_id = :new.product_id;
```

```
END;

-------------------------------------------------------------------------------------------------------

CREATE OR REPLACE TRIGGER update_seller_rating AFTER
   INSERT OR UPDATE OF rating ON review
   FOR EACH ROW
DECLARE
   new_rating           NUMBER(2, 1);
   seller_id_to_update  VARCHAR(255);
BEGIN
   new_rating := :new.rating;
   SELECT
      seller_id
   INTO seller_id_to_update
   FROM
      product
   WHERE
      product_id = :new.product_id;

   UPDATE seller
   SET
      average_rating = ( ( average_rating * rating_count ) + new_rating ) / ( rating_count + 1 ),
      rating_count = rating_count + 1
   WHERE
      seller_id = seller_id_to_update;

END;

-------------------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE update_membership (
   buyer_id_input IN VARCHAR
) AS
BEGIN
   UPDATE buyer
   SET
      is_prime = 1,
      prime_expiry_date = add_months(DATE '2019-11-28', 12)
   WHERE
      buyer_id = buyer_id_input;

END update_membership;
```

```
-----------------------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE cancel_membership (
   buyer_id_input IN VARCHAR
) AS
BEGIN
   UPDATE buyer
   SET
      is_prime = 0,
      prime_expiry_date = NULL
   WHERE
      buyer_id = buyer_id_input;

END cancel_membership;

-----------------------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE place_order (
   order_id       IN  INTEGER,
   buyer_id_var   IN   VARCHAR
) AS

   card_id_var          INTEGER;
   address_id_var       INTEGER;
   total_price_var      NUMBER := 0;
   curr_price_var       NUMBER;
   total_qty_var        NUMBER := 0;
   available_units_var  NUMBER(1);
   shipping_price_var    NUMBER := 10;
   is_prime_var         NUMBER := 0;
   CURSOR products_cur IS
   SELECT
      product_id
   FROM
      product_shoppingcart
   WHERE
      buyer_id = buyer_id_var;

   product_id_var       INTEGER;
BEGIN
   OPEN products_cur;
   LOOP
      FETCH products_cur INTO product_id_var;
```

```
        EXIT WHEN products_cur%notfound;
        SELECT
           price,
           available_units
        INTO
           curr_price_var,
           available_units_var
        FROM
           product
        WHERE
           product_id = product_id_var;

        IF available_units_var > 0 THEN
           total_price_var := ( total_price_var + curr_price_var );
           total_qty_var := total_qty_var + 1;
           INSERT INTO order_product VALUES (
              order_id,
              product_id_var
           );

        END IF;

--        DELETE FROM product_shoppingcart
--        WHERE product_id = product_id_var AND buyer_id = buyer_id_var;

    END LOOP;

    CLOSE products_cur;
    SELECT
       is_prime
    INTO is_prime_var
    FROM
       buyer
     WHERE
       buyer_id = buyer_id_var;

    IF is_prime_var = 1 THEN
       shipping_price_var := 0;
    END IF;
    SELECT
       card_id
    INTO card_id_var
    FROM
       card_info
```

```
    WHERE
        buyer_id = buyer_id_var
        AND is_default = 1;

    SELECT
        address_id
    INTO address_id_var
    FROM
        contact_detail
    WHERE
        user_id = buyer_id_var
        AND is_default = 1;

    total_price_var := total_price_var + shipping_price_var + 10;
    INSERT INTO amz_order VALUES (
        order_id,
        buyer_id_var,
        card_id_var,
        total_price_var,
        sysdate,
        10,
        shipping_price_var,
        address_id_var,
        add_months(DATE '2019-11-28', 1),
        'c',
        total_qty_var
    );

END place_order;

-----------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE populate_product_categories AS
BEGIN
    INSERT INTO category VALUES (
        1,
        'Electronics'
    );

    INSERT INTO category VALUES (
        2,
        'Books'
    );
```

```
   INSERT INTO category VALUES (
      3,
      'Clothing'
   );

END populate_product_categories;

-----------------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE populate_carriers AS
BEGIN
   INSERT INTO carrier VALUES (
      1,
      'DHL',
      1234567890,
      'DHL@gmail.com'
   );

   INSERT INTO carrier VALUES (
      2,
      'Fedex',
      1234567890,
      'Fedex@gmail.com'
   );

   INSERT INTO carrier VALUES (
      3,
      'UPS',
      1234567890,
      'UPS@gmail.com'
   );

END populate_carriers;

-----------------------------------------------------------------------------------------------------

CREATE OR REPLACE TRIGGER update_available_units AFTER
   INSERT ON amz_order
   FOR EACH ROW
DECLARE
   product_id_var       INTEGER;
   available_units_var  INTEGER;
   CURSOR products_cur IS
   SELECT
```

```
    product_id
  FROM
    order_product
  WHERE
    order_id = :new.order_id;

BEGIN
  OPEN products_cur;
  LOOP
    FETCH products_cur INTO product_id_var;
    EXIT WHEN products_cur%notfound;
    SELECT
      available_units
    INTO available_units_var
    FROM
      product
    WHERE
      product_id = product_id_var;

    IF available_units_var >= 2 THEN
      UPDATE product
      SET
        available_units = available_units - 1
      WHERE
        product_id = product_id_var;

    ELSIF available_units_var = 1 THEN
      UPDATE product
      SET
        available_units = available_units - 1,
        in_stock = 0
      WHERE
        product_id = product_id_var;

    END IF;

  END LOOP;

  CLOSE products_cur;
END;

CREATE OR REPLACE TRIGGER remove_items_from_cart AFTER
  INSERT ON amz_order
  FOR EACH ROW
```

```
DECLARE BEGIN
   DELETE FROM shopping_cart
   WHERE
      buyer_id = :new.buyer_id;

   DELETE FROM product_shoppingcart
   WHERE
      buyer_id = :new.buyer_id;

END;

BEGIN
   register_buyer('anshulpardhi@gmail.com', 'anshul', 'pardhi', 'abcd123');
   register_buyer('ashwanikashyap@gmail.com', 'ashwani', 'kashyap', 'abcd123');
   register_buyer('gunjanagicha@gmail.com', 'gunjan', 'agicha', 'abcd123');
END;

BEGIN
   register_seller('kushagradar@gmail.com', 'kushagra', 'dar', 'abcd123', 'kushagra Co and
Co',
            'www.kusharga.com', 'company of shoes');
   register_seller('ruchisingh@gmail.com', 'ruchi', 'singh', 'abcd123', 'ruchi Co and Co',
            'www.ruchi.com', 'company of metals');
   register_seller('anantprakash@gmail.com', 'anant', 'prakash', 'abcd123', 'anant Co and Co',
            'www.anant.com', 'company of iphones');
END;

BEGIN
   update_contact_details('anshulpardhi@gmail.com', 1, '7825 McCallum Blvd', 'Apt 007',
'Dallas',
                  'Texas', 'USA', 75252, 8888888888);

   update_contact_details('gunjanagicha@gmail.com', 2, '7825 McCallum Blvd', 'Apt 1702',
'Dallas',
                  'Texas', 'USA', 75252, 8888888888);

   update_contact_details('gunjanagicha@gmail.com', 3, '7825 McCallum Blvd', 'Apt 1702',
'Dallas',
                  'Texas', 'USA', 75252, 4692309274);

   set_default_contact_details(3, 'gunjanagicha@gmail.com');
   set_default_contact_details(1, 'anshulpardhi@gmail.com');
END;
```

```
BEGIN
   add_card_info('gunjanagicha@gmail.com', 1, 1234123412341234, TO_DATE('2023-12-09',
'YYYY-MM-DD'),
              666);
   add_card_info('gunjanagicha@gmail.com', 2, 0234123412341234, TO_DATE('2023-12-09',
'YYYY-MM-DD'),
              777);
   add_card_info('anshulpardhi@gmail.com', 3, 0234123412341234, TO_DATE('2023-12-09',
'YYYY-MM-DD'),
              777);
   set_default_card_info(1, 'gunjanagicha@gmail.com');
   set_default_card_info(3, 'anshulpardhi@gmail.com');
END;

BEGIN
   populate_product_categories();
   populate_carriers();
END;

BEGIN
   add_product(1, 'OnePlus 7', 'kushagradar@gmail.com', 400, 1,
            'Best Phone', 2, 'Blue', 2, 2,
            'bit.ly/sfdf4fg');

   add_product(2, 'Harry Potter', 'kushagradar@gmail.com', 15, 2,
            'Best Book', 5, 'Black', 8, 2,
            'bit.ly/sfdf4fg');

   add_product(3, 'Nike Shoes', 'anantprakash@gmail.com', 50, 3,
            'Best shoes', 2, 'yellow', 5, 1,
            'bit.ly/sfdf4fg');

   add_product(4, 'I phone', 'anantprakash@gmail.com', 500, 1,
            'Better than android', 3, 'Black', 2, 3,
            'bit.ly/sfdf4fg');

   add_product(5, 'Metal Detector', 'ruchisingh@gmail.com', 20, 1,
            'Best metal detector', 4, 'Grey', 12, 2,
            'bit.ly/sfdf4fg');

END;

BEGIN
   add_to_wish_list('anshulpardhi@gmail.com',  1);
```

```
  add_to_wish_list('anshulpardhi@gmail.com',  4);
  add_to_wish_list('anshulpardhi@gmail.com',  3);
  add_to_wish_list('gunjanagicha@gmail.com', 2);
END;

BEGIN
  add_to_shopping_cart('anshulpardhi@gmail.com',  1);
  add_to_shopping_cart('anshulpardhi@gmail.com',  3);
  add_to_shopping_cart('gunjanagicha@gmail.com',  2);
  add_to_shopping_cart('gunjanagicha@gmail.com', 1);
END;

BEGIN
  update_membership('gunjanagicha@gmail.com');
END;

BEGIN
  place_order(1, 'gunjanagicha@gmail.com');
END;

BEGIN
  place_order(2, 'anshulpardhi@gmail.com');
END;

BEGIN
  give_review(1, 1, 'anshulpardhi@gmail.com', 'cool phone with great camera', 5,
        'www.my_image.com');
  give_review(2, 3, 'anshulpardhi@gmail.com', 'good running shoes', 3,
        'www.my_image.com');
END;

BEGIN
  give_review(3, 2, 'gunjanagicha@gmail.com', 'nice book', 3.5,
        'www.my_image.com');
END;

BEGIN
  give_review(4, 1, 'gunjanagicha@gmail.com', 'okay phone', 3,
        'www.my_image.com');
END;

BEGIN
  give_review(5, 5, 'gunjanagicha@gmail.com', 'doesnt work', 1,
        'www.my_image.com');
```

```
END;

BEGIN
   give_review(6, 5, 'gunjanagicha@gmail.com', 'doesnt work at all', 0,
            'www.my_image.com');
END;
```