

A project report on

“eBay-1”

submitted in fulfilment of requirements for

CS6360 Database Design, Fall 2019

By-

Name	Net ID
Randeep Ahlawat	RSA190000
Jayant Bhopale	JHB180002
Ankur Gokhale	AAG180015



Erik Jonsson School of Engineering and Computer Science

The University of Texas at Dallas

800 W Campbell Rd, Richardson, TX 75080.

Table of contents

1. Introduction

1.1 The Stats on eBay.....	Page 2
1.2 Requirements Gathering.....	Page 3

2. System Design

2.1 Structure of Ebay.....	Page 7
2.2 EER Diagram.....	Page 10
2.3 Analysis of EER diagram.....	Page 11

3. Relational Schema

3.1 Relational Schema in 3NF.....	Page 14
3.2 Converting Relational Schema To BCNF.....	Page 17

4. SQL queries

4.1 SQL Statements to create Relations in DB and Add Constraints.....	Page 20
---	---------

5. PL-SQL Queries

5.1 Stored Procedures

5.1.1 Average rating of seller.....	Page 26
5.1.2 Purchase History of Buyer.....	Page 27

5.2 Triggers

5.2.1 Validate bid.....	Page 29
5.2.2 Update bid.....	Page 31

1. Introduction

1.1 The Stats on eBay

On an average day, eBay system runs through 26 billion SQL queries and keeps tabs on 100 million items available for purchase. 182 million registered users with over 1 billion photos 1 billion page views a day, 105 million listings, 2 petabytes of data, 3 billion API calls a month.

This humongous amount of data is handled exceptionally by the database system of ebay and it ensures 99.94% availability. Architecture is strictly divided into layers: data tier, application tier, search, operations. Ebay is built on Oracle databases and split databases by primary access path, modulo on a key. Every database has at least 3 on-line databases. Distributed over 8 data centers. Databases are segmented by function: user, item account, feedback, transaction, over 70 in all. No stored procedures are used. There are some very simple triggers.

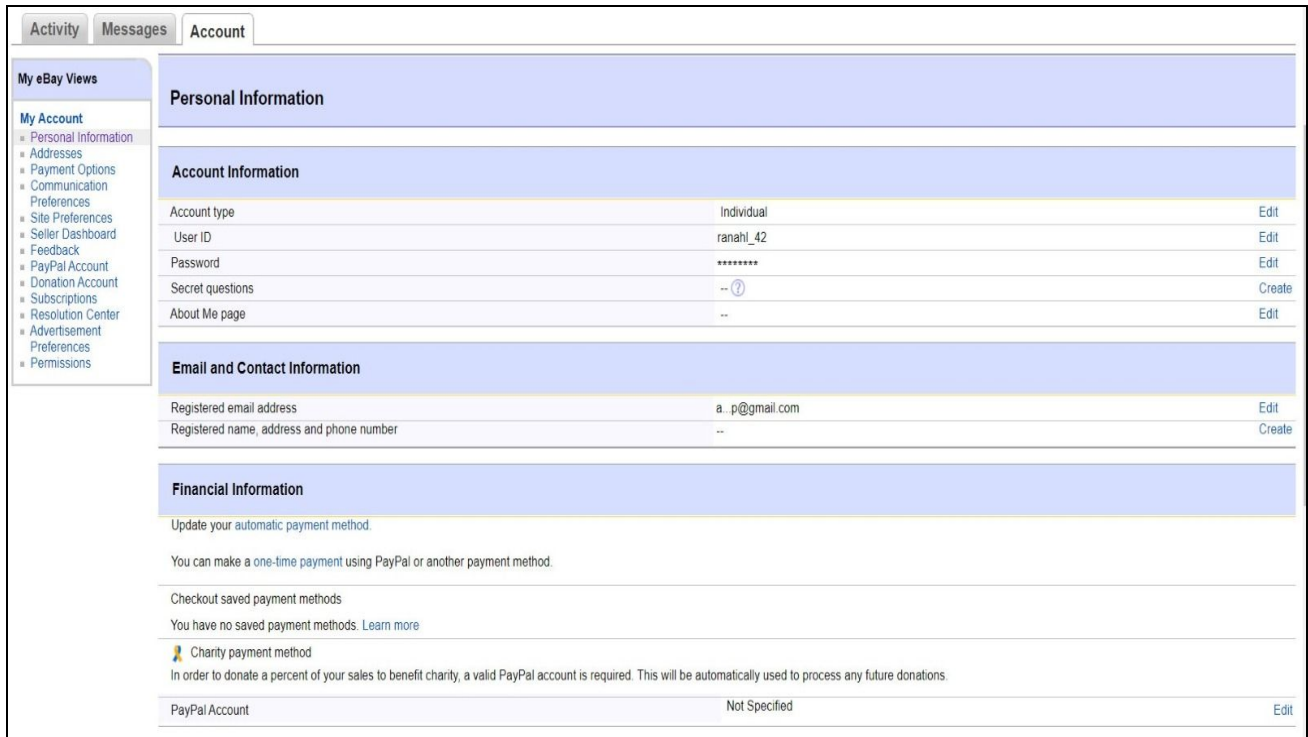
The database of ebay is fragmented at each level. It maintains separate databases for User, Item, Transaction, Product, Account and Feedback. Over 1000 logical databases on ~400 physical hosts.

1.2 Requirements Gathering

Services provided by eBay:

1. Account creation

Every user who wish to purchase or sell products online on ebay has to create an account with basic details such as name, age, ID, Address, contact number etc.



The screenshot displays the eBay 'My Account' page. At the top, there are tabs for 'Activity', 'Messages', and 'Account'. A left sidebar lists 'My eBay Views' including 'My Account', 'Personal Information', 'Addresses', 'Payment Options', 'Communication Preferences', 'Site Preferences', 'Seller Dashboard', 'Feedback', 'PayPal Account', 'Donation Account', 'Subscriptions', 'Resolution Center', 'Advertisement Preferences', and 'Permissions'. The main content area is divided into several sections: 'Personal Information', 'Account Information', 'Email and Contact Information', and 'Financial Information'. The 'Account Information' section contains a table with fields for Account type, User ID, Password, Secret questions, and About Me page. The 'Email and Contact Information' section contains a table with fields for Registered email address and Registered name, address and phone number. The 'Financial Information' section contains text about updating payment methods and a table for the PayPal Account.

Personal Information		
Account Information		
Account type	Individual	Edit
User ID	ranah1_42	Edit
Password	*****	Edit
Secret questions	-- ?	Create
About Me page	--	Edit

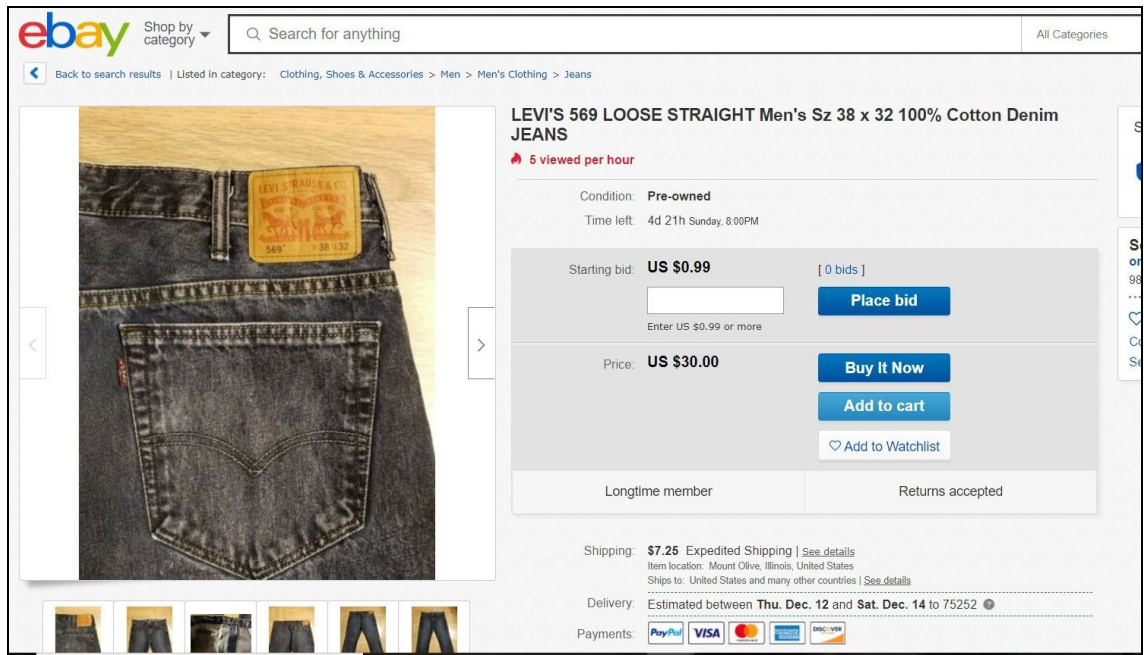
Email and Contact Information		
Registered email address	a_p@gmail.com	Edit
Registered name, address and phone number	--	Create

Financial Information		
Update your automatic payment method.		
You can make a one-time payment using PayPal or another payment method.		
Checkout saved payment methods		
You have no saved payment methods. Learn more		
Charity payment method		
In order to donate a percent of your sales to benefit charity, a valid PayPal account is required. This will be automatically used to process any future donations.		
PayPal Account	Not Specified	Edit

2. Buy product/ Bid on product

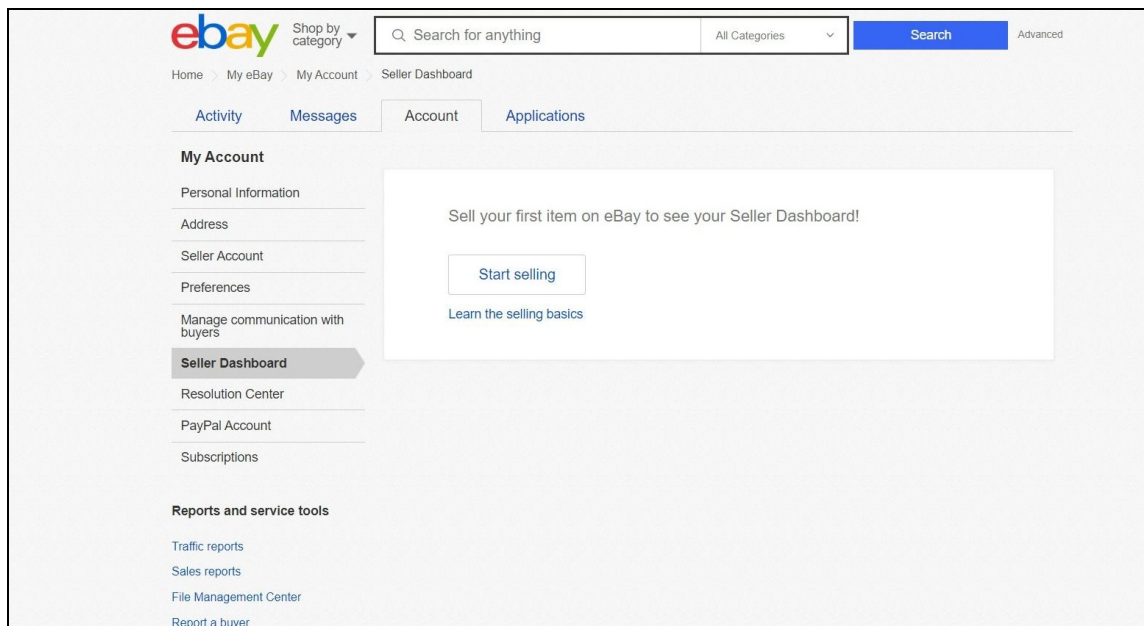
Every customer who wish to purchase a new or a refurbished product has a variety of choices to choose from. The product chosen can be added to the cart and later a transaction can be made to complete the purchase.

Ebay provides a feature that allows seller to post product for auction where multiple buyers can bid their price and then the buyer can decide to whom the product to be sold.





3. Sell product

Customer or shop owners who willing to sell products on ebay can post sell order with appropriate selling price, location, and features of product.



4. Purchase History

Each buyer can see the list of products that are purchased from different seller and see information related to previous orders.

<p>ORDER DATE Sep 06, 2015</p> <p>1 item sold by watchesplus11</p> 	<p>SPECIAL BRAND NEW G SHOCK WITH TAGS!! GA1000-9G Men's G-Aviation Black Classic (171917528093)</p> <p>Delivered on Thu, Sep 10 Tracking number: 9400109699937885734003</p>	<p>ORDER TOTAL US \$154.92</p> <p>Free shipping</p> <p>ITEM PRICE: US \$154.92</p> <p>Leave feedback</p> <p>View order details More actions ▼</p>
<p>ORDER DATE Aug 21, 2015</p> <p>1 item sold by hotnessfeet215</p> 	<p>Nike Air Max2 CB '94 Charles Barkley Black Purple OG 305440-006 MEN Size: 7.5-13 (221854085671)</p>	<p>ORDER TOTAL US \$144.95</p> <p>See description</p> <p>ITEM PRICE: US \$144.95</p> <p>View similar items</p> <p>View seller's other items More actions ▼</p>
<p>ORDER DATE Aug 21, 2015</p>		<p>ORDER TOTAL US \$45.99</p> <p>See description</p> <p>View similar items</p> <p>View seller's other items</p>

5. Payment

Ebay provides multiple options to complete payment that includes majority of banks and credit card service companies.

6. Shipment Tracker

Each product that is purchased from ebay has an order number that can be used for tracking the progress of shipment and delivery. This services it handled by third party.

7. Notifications

Once the user sets up the account and provide email id and phone number to the system, it will send notifications about user's order delivery or when there are offers available.

8. Customer care services

Customer care services provide help to the user and give suggestions and answer the question of customers.

2. System Design

2.1 Structure of Ebay

User:

Users are the integral unit of the system. Each user has a user ID, password, Email ID, Name, Address, Phone number and a payment info. User can either act as a buyer or a seller for a particular transaction of product purchase. Depending on the role of the user the additional attributes are considered for the user. For buyer search history, shipping address and list of items are considered additionally. Seller stores the shipping to address.

Account:

Each customer must have an account to purchase an item from ebay and account must be unique and only one account per customer. Account can be uniquely defined by its id and each account is secured with an encoded password which is helpful for security purpose. So, there is one to one relation between account and Customer.

Payment

There are multiple payment methods for seller to accept payment from as well as for buyer to make payment. Each payment has payment_id, Account, Type of payment(debit, credit, cheque using routing number). This payment is executed at the bank end by sending the transaction details to the bank.

Product

The product is the most important entity in any ecommerce platform. Our product in ER diagram is represented as product offer. The product offer contains product_id , seller_user_id as primary key which uniquely identifies what the seller wants to sell. This also determines other attributes of the product like product_name, date_of_expiry, category, keyword, description and foreign key is user_id.

Bank

The user buys a product with the help of credit/debit cards. The card is linked to his/her bank account. The primary key is Bank_id which uniquely identifies Bank attributes like Bank Name, Branch Name, Amount, Payment_Id and User_id. Foreign Key payment_id and user_id references Payment(payment_id and user_id)

Bid

The bidding option is an important feature of ebay. This allows the user to bid for a particular product. The primary key for Bid is Bid_no and Product_id which uniquely identifies the bid the user places. The product_id and seller_user_id is foreign key on Product_Offer.

Auction

Ebay also allows the sellers to place items on their platform for auction. The interested buyers place their bids and the buyers with the highest bid gets the product. The foreign key in Auction is product_id and seller_user_id which references Product Offer(product_id, seller_user_id)

Direct Buy

In addition to auction system ebay also allows users to directly buy the product. The seller can put the product for auction as well as places direct buy offer to the buyers. The Direct Buy entity has foreign key product_id and seller_user_id which references Product Offer(product_id, seller_user_id)

Buyer

Buyer is one of the most important entities in any ecommerce platform. Buyers is the one who buys products. The Buyer_User_Id is the primary key for the buying entity. It uniquely identifies the buyer, shippingtoaddress, following list and user_id

Seller

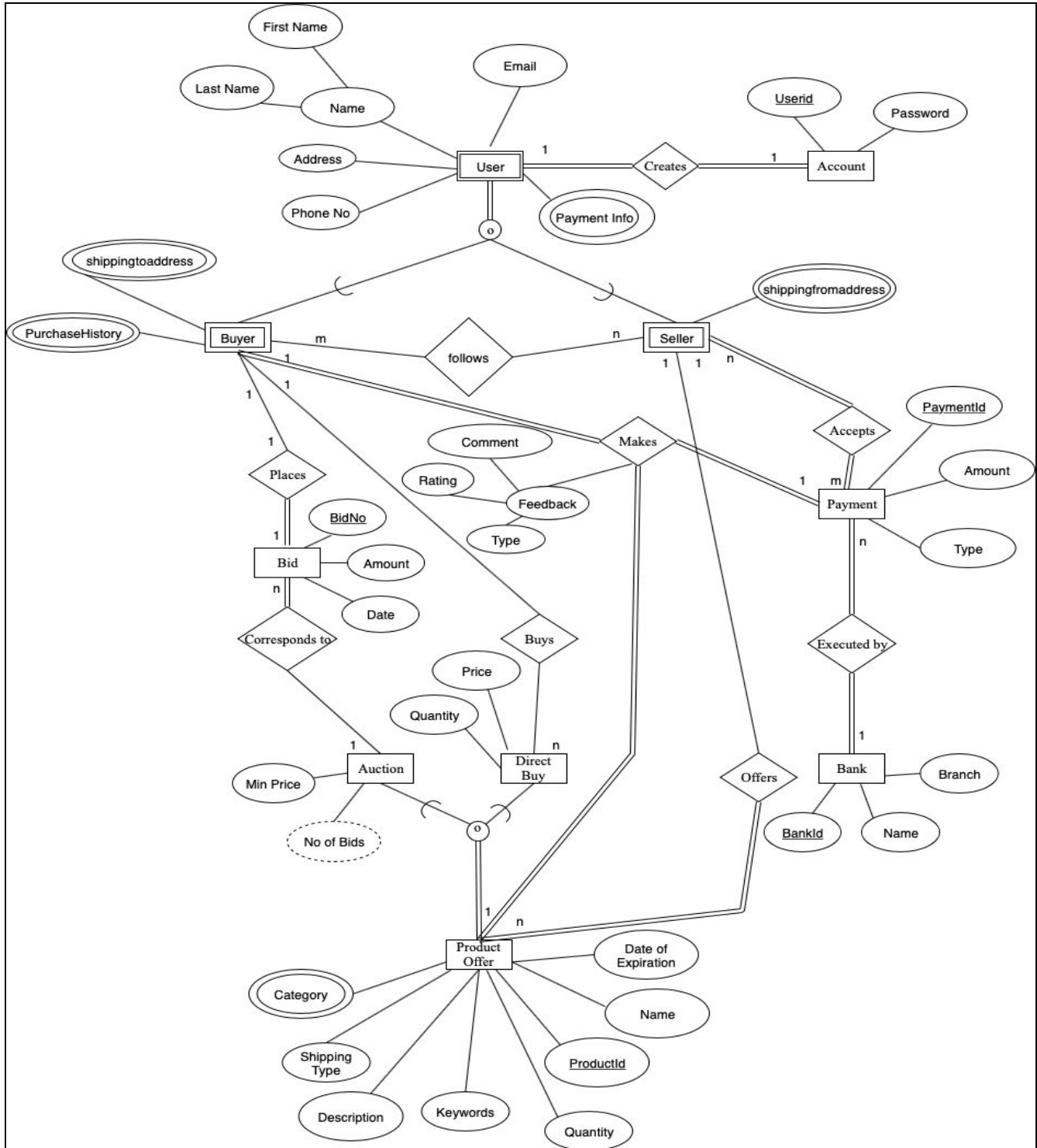
The seller is another important entity. Seller is the one who sells products to the buyer. Primary key is seller_id. The foreign key is seller_user_id which references Account(User_id)

2.1.1 Entities in Ebay:

- Account
- User
- Buyer
- Seller
- Bid
- Auction
- Direct Buy
- Product Offer
- Payment
- Bank

2.2 EER Diagram:

The relationship between the entities can be shown as per the following EER diagram



The requirements can be summarized/ derived from ERD as below -

1. Each user must have an account in order to perform any activity on ebay. Every user must have at most one account hence 1:1 relationship between Users and Account. Also every user has to have one account and each account must have one user, hence total participation of both entities in creates relationship.
2. Every seller accepts payment for purchase in order to complete the transaction. Each payment acceptance must involve both the entities(Payment and Seller) hence total participation. Also seller can accept multiple payments hence, many to many relationship.
3. Seller can offer multiple product offers and each offer must have exactly one seller, hence one to many relationship between Product offer and Seller. Offer must include a product but it is not mandatory that all sellers will offer product offer, hence total participation on Product_offer and partial participation on Seller.
4. Bank can execute multiple payments at a time and each payment must include exactly one bank for successful transaction, hence one to many relationship between Payment and Bank. Both the entities must include in transaction to complete the payment successfully, hence total participation of Bank and Payment.
5. Buyer may follow one or more Sellers and Seller can be followed by many buyers, hence many to many relationship between buyer and seller. Also it is not mandatory for Buyer to follow any Seller and also for Seller to follow Buyer, hence partial participation on both the entities.
6. One buyer can place only one bid on a product and each bid must have exactly one buyer, hence one to one relationship between Bid and Buyer. Also, Buyer may or may not place a bid but each bid must have a buyer, hence total participation of Bids and partial participation of Buyer.

7. Products that are put on offers by seller can either be purchased as direct-buy or by bidding in Auction. Hence direct_buy and Auction will have all attributes from product_offer.
8. Every bid must corresponds to an auction, hence total participation of Bid in relationship but an auction may or may not have any bid. Action may have multiple bids.

3. Relational Schema

3.1 Relational Schema in 3NF

Account

<u>Userid</u>	Password
---------------	----------

Primary Key: Userid

Ebay_user

<u>User_id</u>	Fname	Phone_no	Lname	Address	Payment_info	Email
----------------	-------	----------	-------	---------	--------------	-------

Primary Key: User_id

Foreign Key: User_id On Account(User_id)

Seller

<u>Shippingfromaddress</u>	<u>User_id</u>
----------------------------	----------------

Primary Key: (User_id, Shippingfromaddress)

Foreign Key: User_id On Account(User_id)

Buyer

<u>Shippingtoaddress</u>	<u>Userid</u>
--------------------------	---------------

Primary Key: (User_id, Shippingtoaddress)

Foreign Key: User_id On Account (User_id)

Purchase History

<u>Product_id</u>	<u>User_id</u>
-------------------	----------------

Primary Key: (User_id, Product_id)

Foreign Key: Product_id, User_id References Payment(Product_id, User_id)

Product Offer

<u>Product_id</u>	<u>Seller_user_id</u>	Pname	Date_of_exp	Keyword	Category	Description
-------------------	-----------------------	-------	-------------	---------	----------	-------------

Primary Key: Product_id, Seller_user_id

Foreign_key: Seller_user_id References Seller(User_id)

Bid

<u>Bid_no</u>	Amount	Date	<u>Product_id</u>	Seller_user_id
---------------	--------	------	-------------------	----------------

Primary_key: Bid_no,Product_id

Foreign Key: (Product_id,Seller_user_id) References Product_offer(Productid, Seller_user_id)

Payment

<u>Payment_id</u>	Amount	Payment_type	<u>User_id</u>	<u>Product_id</u>
-------------------	--------	--------------	----------------	-------------------

Primary_key: (Payment_id, User_id, Product_id)

Foreign Key: User_id On Buyer(User_id), Product_id References Product_offer(Product_id)

Bank

<u>Bank_id</u>	B_name	Branch	<u>Payment_id</u>	<u>User_id</u>
----------------	--------	--------	-------------------	----------------

Primary Key (Bank_id),

Foreign Key(Payment_id, User_id) References Payment(Payment_id, User_id)

Payment_info

<u>User_id</u>	Payment_info
----------------	--------------

Foreign Key(User_id) References Ebay_user(User_id)

Category

<u>Product_id</u>	Category
-------------------	----------

Foreign Key (Product_id) References Product_offer(Product_id)

Auction

Min_price	No_of_bids	<u>Product_id</u>	<u>Seller_user_id</u>
-----------	------------	-------------------	-----------------------

Primary Key: Seller_user_id, Product_id

Foreign Key: (Product_id,Seller_user_id) On Product_offer(Productid, Seller_user_id)

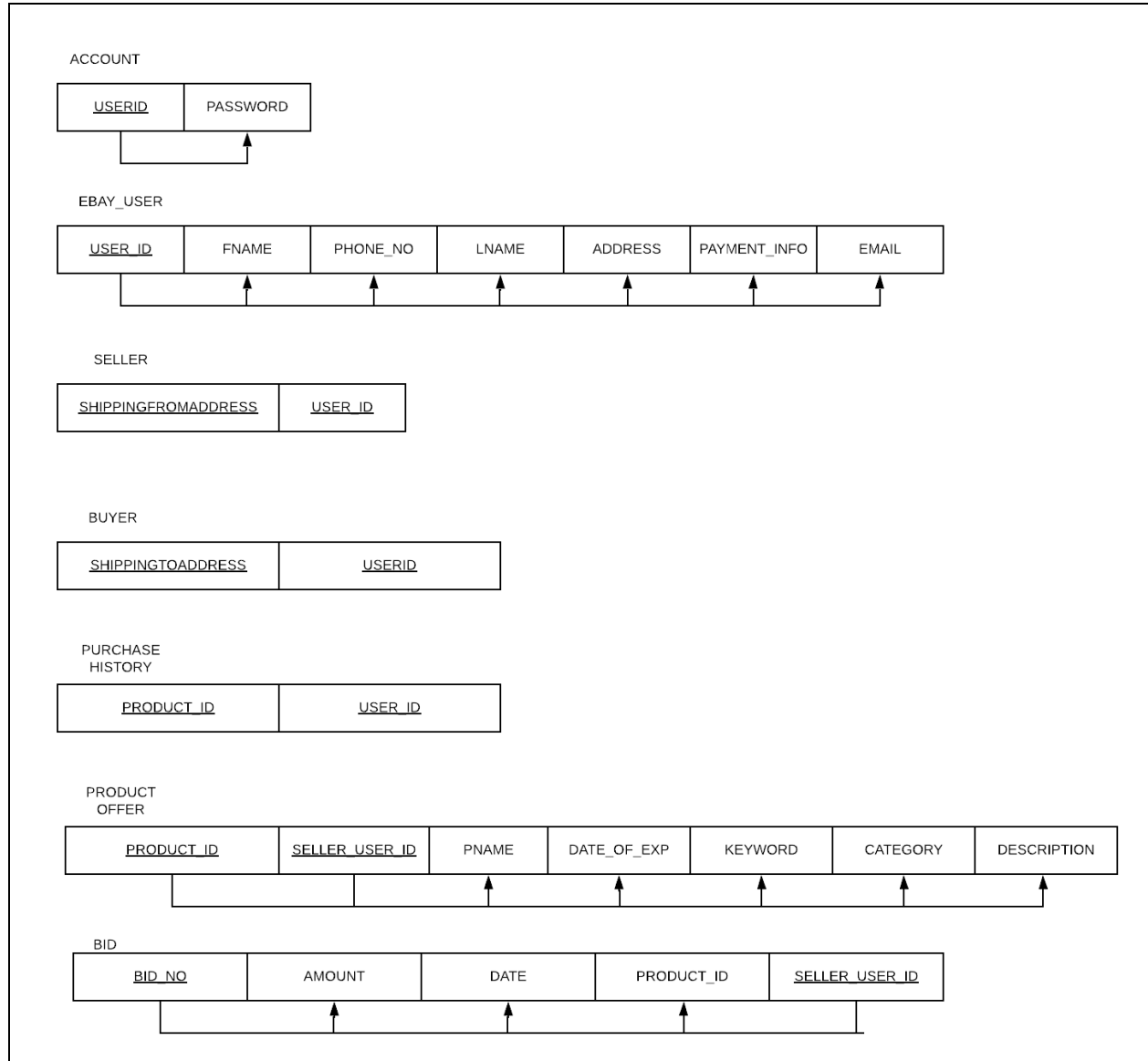
Direct Buy

Quantity	Price	<u>Product_id</u>	<u>Seller_user_id</u>
----------	-------	-------------------	-----------------------

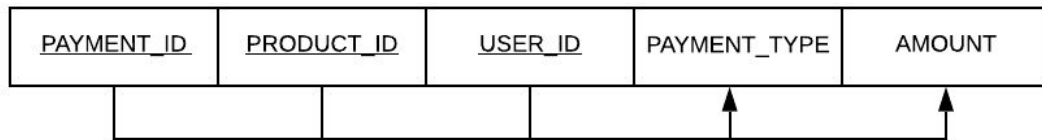
Primary Key: Seller_user_id, Product_id

Foreign Key: (Product_id,Seller_user_id) On Product_offer(Productid, Seller_user_id)

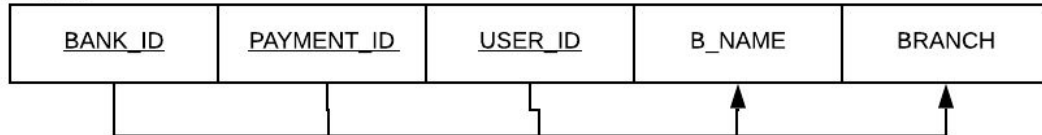
3.2 CONVERTING RELATIONAL SCHEMA TO BCNF



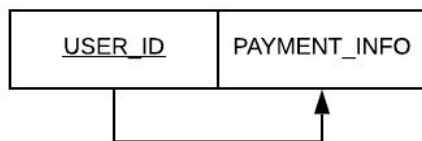
PAYMENT



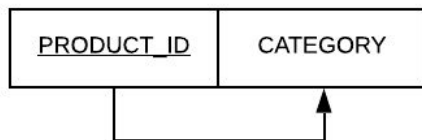
BANK



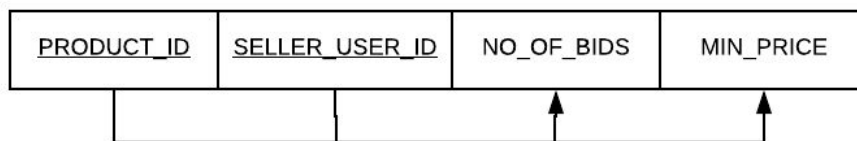
PAYMENT_INFO



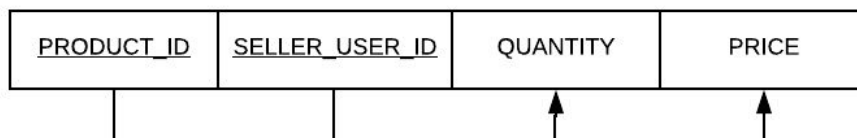
CATEGORY



AUCTION



DIRECT BUY



4. SQL CODE

4.1 SQL Statements to create Relations in DB and Add Constraints:

```
CREATE TABLE account
```

```
(  
    user_id    CHAR(20) NOT NULL,  
    password   VARCHAR(20) NOT NULL,  
    PRIMARY KEY(user_id)  
);
```

```
CREATE TABLE ebay_user
```

```
(  
    fname      CHAR(20) NOT NULL,  
    lname      CHAR(20) NOT NULL,  
    email      CHAR(20) NOT NULL,  
    address     VARCHAR(50) NOT NULL,  
    phone      INT NOT NULL,  
    payment_info VARCHAR(10),  
    user_id     CHAR(20) NOT NULL,  
    FOREIGN KEY(user_id) REFERENCES account(user_id),  
    PRIMARY KEY(user_id)  
);
```

```
CREATE TABLE buyer
```

```
(  
    purchase_hist    VARCHAR(50),  
    shippingtoaddress VARCHAR(50) NOT NULL,  
    buyer_user_id     CHAR(20) NOT NULL,  
    user_id           CHAR(20) NOT NULL,  
    PRIMARY KEY(buyer_user_id),  
    FOREIGN KEY(buyer_user_id) REFERENCES account(user_id)  
);
```

```
CREATE TABLE seller
```

```
(  
    seller_user_id     CHAR(20) NOT NULL,  
    shippingfromaddress VARCHAR(50) NOT NULL,  
    user_id            CHAR(20) NOT NULL,  
    PRIMARY KEY(seller_user_id),  
    FOREIGN KEY(seller_user_id) REFERENCES account(user_id)  
);
```

```

CREATE TABLE payment
(
    payment_id      CHAR(20) NOT NULL,
    amount          INT NOT NULL,
    type_of_payment VARCHAR(10) NOT NULL,
    user_id         CHAR(20) NOT NULL,
    PRIMARY KEY(payment_id, user_id),
    FOREIGN KEY(user_id) REFERENCES account(user_id)
);

```

```

CREATE TABLE product_offer
(
    productid      CHAR(20) NOT NULL,
    p_description   CHAR(20) NOT NULL,
    date_of_expiration INT NOT NULL,
    pname          VARCHAR(10) NOT NULL,
    quantity       INT,
    keyword         CHAR(20),
    shipping_type   VARCHAR(40) NOT NULL,
    category_offer  VARCHAR(40) NOT NULL,
    user_id        CHAR(20) NOT NULL,
    seller_user_id  CHAR(20) NOT NULL,
    price          INT NOT NULL,
    buyer_user_id   CHAR(20) NOT NULL,
    PRIMARY KEY(productid, seller_user_id),
    FOREIGN KEY(seller_user_id) REFERENCES seller(seller_user_id),
    FOREIGN KEY(buyer_user_id) REFERENCES buyer(buyer_user_id)
);

```

```

CREATE TABLE bid
(
    bidno          INT DEFAULT (0),
    amount         INT NOT NULL,
    bid_date       DATE NOT NULL,
    productid      CHAR(20) NOT NULL,
    seller_user_id CHAR(20) NOT NULL,
    PRIMARY KEY(bidno, productid),
    FOREIGN KEY(productid, seller_user_id) REFERENCES
product_offer(productid,
    seller_user_id)
);

```

```

CREATE TABLE auction
(
    min_price          INT NOT NULL,
    no_of_bids         INT NOT NULL,
    productid          CHAR(20) NOT NULL,
    seller_user_id     CHAR(20) NOT NULL,
    FOREIGN KEY(productid, seller_user_id) REFERENCES
product_offer(productid,
    seller_user_id)
);

```

```

CREATE TABLE direct_buy
(
    price              INT NOT NULL,
    quantity           INT NOT NULL,
    seller_user_id     CHAR(20) NOT NULL,
    productid          CHAR(20) NOT NULL,
    FOREIGN KEY(productid, seller_user_id) REFERENCES
product_offer(productid,
    seller_user_id)
);

```

```

CREATE TABLE bank
(
    bank_id            VARCHAR(10) NOT NULL,
    branch             VARCHAR(9) NOT NULL,
    bank_name          CHAR(10) NOT NULL,
    user_id            CHAR(20) NOT NULL,
    payment_id         CHAR(20) NOT NULL,
    PRIMARY KEY (bank_id),
    FOREIGN KEY(payment_id, user_id) REFERENCES payment(payment_id,
user_id)
);

```

```

CREATE TABLE follows
(
    seller_user_id     CHAR(20) NOT NULL,
    buyer_user_id      CHAR(20) NOT NULL,
    FOREIGN KEY(seller_user_id) REFERENCES seller(seller_user_id),
    FOREIGN KEY(buyer_user_id) REFERENCES buyer(buyer_user_id)
);

```



```

CREATE TABLE direct_buy
(
    price            INT NOT NULL,
    quantity         INT NOT NULL,
    seller_user_id   CHAR(20) NOT NULL,
    productid        CHAR(20) NOT NULL,
    buyer_user_id    CHAR(20) NOT NULL,
    PRIMARY KEY(buyer_user_id),
    FOREIGN KEY(buyer_user_id) REFERENCES account(user_id),
    FOREIGN KEY(productid, seller_user_id) REFERENCES
product_offer(productid,
    seller_user_id)
);

```

```

CREATE TABLE feedback
(
    feedback_type    CHAR(20) NOT NULL,
    feedback_comment CHAR(20) NOT NULL,
    rating           INT DEFAULT(0),
    seller_user_id   CHAR(20) NOT NULL,
    productid        CHAR(20) NOT NULL,
    buyer_user_id    CHAR(20) NOT NULL,
    FOREIGN KEY(productid, seller_user_id) REFERENCES
product_offer(productid,
    seller_user_id),
    FOREIGN KEY(buyer_user_id) REFERENCES buyer(buyer_user_id)
);

```

```

CREATE TABLE accepts
(
    seller_user_id   CHAR(20) NOT NULL,
    buyer_user_id    CHAR(20) NOT NULL,
    FOREIGN KEY(seller_user_id) REFERENCES seller(seller_user_id),
    FOREIGN KEY(buyer_user_id) REFERENCES buyer(buyer_user_id)
);

```

```
CREATE TABLE purchase_history
(
    seller_user_id CHAR(20) NOT NULL,
    buyer_user_id CHAR(20) NOT NULL,
    productid CHAR(20) NOT NULL,
    PRIMARY KEY(buyer_user_id, productid),
    FOREIGN KEY(productid, seller_user_id) REFERENCES
product_offer(productid,
    seller_user_id)
);
```

5. PL-SQL CODE

5.1 Stored Procedures

A SQL Server stored procedure groups one or more Transact-SQL statements into a logical unit and is stored as an object in the Database Server. When a stored procedure is called at the first time, SQL Server creates an execution plan and stores it in the plan cache. In the subsequent executions of the stored procedure, SQL Server reuses the plan so that the stored procedure can execute very fast with reliable performance.

5.1.1 Average Rating of seller

Here, a stored procedure can be created to get the average rating of seller based on product sold by the seller. First name, Last Name and average rating of the seller can be found using a stored procedure.

```
CREATE PROCEDURE Get_avg_rating (seller_id      IN CHAR,  
                                   seller_fname OUT CHAR,  
                                   seller_lname OUT CHAR,  
                                   avg_rating   OUT INT)  
AS  
BEGIN  
    SELECT E.fname,  
           E.lname,  
           Avg (F.rating)  
    INTO   seller_fname, seller_lname, avg_rating  
    FROM   ebay_user E,  
           feedback F  
    WHERE  E.user_id = seller_id  
           AND F.seller_user_id = seller_id;  
END;
```

```
CREATE PROCEDURE GET_AVG_RATING
(
    SELLER_ID IN CHAR,
    SELLER_FNAME OUT CHAR,
    SELLER_LNAME OUT CHAR,
    AVG_RATING OUT INT
)
AS
BEGIN
    SELECT E.FNAME, E.LNAME, AVG(F.RATING) INTO SELLER_FNAME, SELLER_LNAME, AVG_RATING
    FROM EBAY_USER E, FEEDBACK F
    WHERE E.USER_ID = SELLER_ID AND F.SELLER_USER_ID = SELLER_ID;
END;
```

Script Output x

Task completed in 0.247 seconds

Procedure GET_AVG_RATING compiled

5.1.2 Purchase history

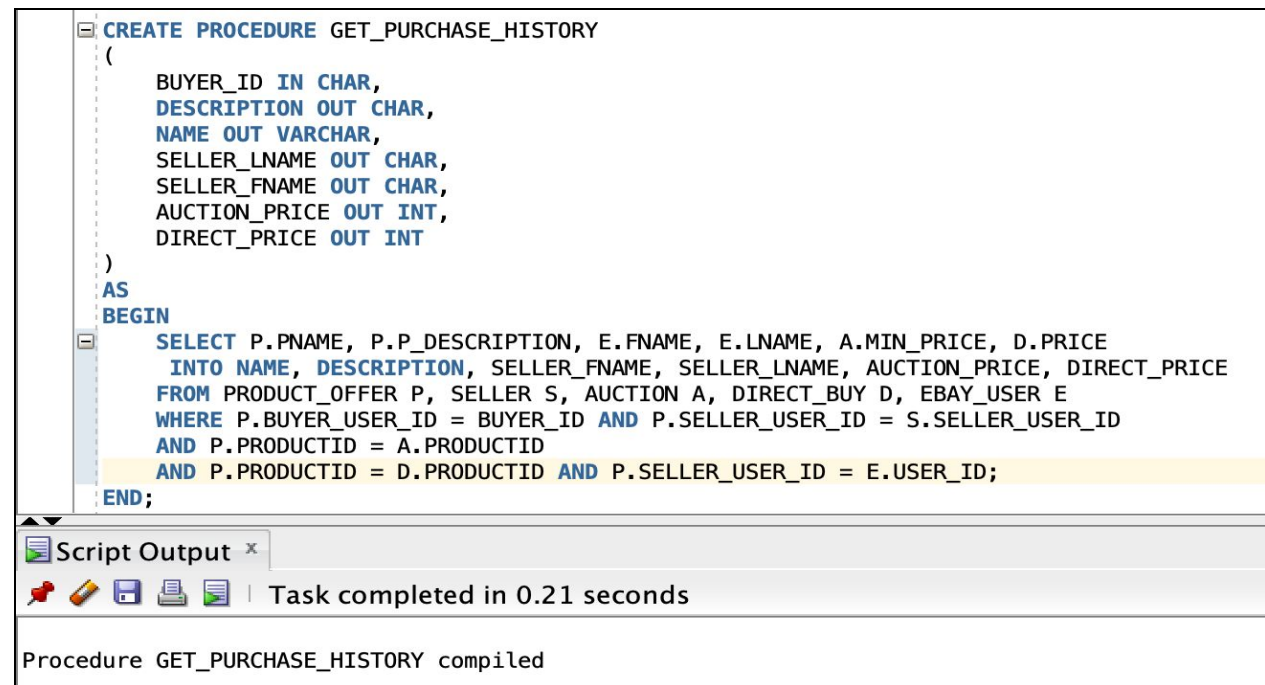
In this case, a stored procedure is created to get a purchase history of a buyer that includes Product Name, Product Description, First name of buyer, Last Name of Buyer, Minimum price of the product at which the product was sold and the actual price of the product.

```
CREATE PROCEDURE Get_purchase_history (buyer_id      IN CHAR,
                                       description    OUT CHAR,
                                       name           OUT VARCHAR,
                                       seller_lname    OUT CHAR,
                                       seller_fname    OUT CHAR,
                                       auction_price   OUT INT,
                                       direct_price    OUT INT)
AS
BEGIN
    SELECT P.pname,
           P.p_description,
           E.fname,
           E.lname,
           A.min_price,
           D.price
    INTO   name, description, seller_fname, seller_lname,
    auction_price, direct_price
    FROM   product_offer P,
           seller S,
           auction A,
```

```

        direct_buy D,
        ebay_user E
WHERE   P.buyer_user_id = buyer_id
        AND P.seller_user_id = S.seller_user_id
        AND P.productid = A.productid
        AND P.productid = D.productid
        AND P.seller_user_id = E.user_id;
END;

```



The screenshot displays a database IDE interface. The main editor window shows the SQL code for creating a procedure named `GET_PURCHASE_HISTORY`. The procedure has six output parameters: `BUYER_ID` (CHAR), `DESCRIPTION` (CHAR), `NAME` (VARCHAR), `SELLER_LNAME` (CHAR), `SELLER_FNAME` (CHAR), `AUCTION_PRICE` (INT), and `DIRECT_PRICE` (INT). The body of the procedure is a `SELECT` statement that joins the `PRODUCT_OFFER`, `SELLER`, `AUCTION`, `DIRECT_BUY`, and `EBAY_USER` tables. The `WHERE` clause filters for a specific buyer and seller, and ensures the product is from an auction or direct buy. The last line of the procedure body is highlighted in yellow.

Below the editor, a "Script Output" window is open, showing a message: "Task completed in 0.21 seconds". At the bottom of the IDE, a status bar indicates "Procedure GET_PURCHASE_HISTORY compiled".

```

CREATE PROCEDURE GET_PURCHASE_HISTORY
(
    BUYER_ID IN CHAR,
    DESCRIPTION OUT CHAR,
    NAME OUT VARCHAR,
    SELLER_LNAME OUT CHAR,
    SELLER_FNAME OUT CHAR,
    AUCTION_PRICE OUT INT,
    DIRECT_PRICE OUT INT
)
AS
BEGIN
    SELECT P.PNAME, P.P_DESCRIPTION, E.FNAME, E.LNAME, A.MIN_PRICE, D.PRICE
    INTO NAME, DESCRIPTION, SELLER_FNAME, SELLER_LNAME, AUCTION_PRICE, DIRECT_PRICE
    FROM PRODUCT_OFFER P, SELLER S, AUCTION A, DIRECT_BUY D, EBAY_USER E
    WHERE P.BUYER_USER_ID = BUYER_ID AND P.SELLER_USER_ID = S.SELLER_USER_ID
    AND P.PRODUCTID = A.PRODUCTID
    AND P.PRODUCTID = D.PRODUCTID AND P.SELLER_USER_ID = E.USER_ID;
END;

```

Script Output x

Task completed in 0.21 seconds

Procedure GET_PURCHASE_HISTORY compiled

5.2 Triggers

SQL Server triggers are special stored procedures that are executed automatically in response to the database object, database, and server events.

5.2.1 Trigger to validate bid

```
CREATE OR replace TRIGGER validate_bid
After INSERT
on BID
for each row
DECLARE product_price number;
max_bid                number; BEGIN
    SELECT Max(amount)
    INTO    max_bid
    FROM    bid
    WHERE   :NEW.seller_user_id = seller_user_id;

    SELECT min_price
    INTO    product_price
    FROM    auction
    WHERE   :NEW.productid = auction.productid;

    IF :NEW.amount < product_price
    OR
    :NEW.amount < max_bid then
        raise_application_error( -20001, bid value should be greater
than minimum value! );
    ENDIF;
END;
```

CREATE OR REPLACE TRIGGER VALIDATE_BID

AFTER INSERT ON BID
FOR EACH ROW
DECLARE

PRODUCT_PRICE **NUMBER**;
MAX_BID **NUMBER**;

BEGIN

SELECT MAX(AMOUNT) **INTO** MAX_BID
FROM BID
WHERE :NEW.SELLER_USER_ID = SELLER_USER_ID;

SELECT AUCTION.MIN_PRICE **INTO** PRODUCT_PRICE
FROM AUCTION
WHERE :NEW.PRODUCTID = AUCTION.PRODUCTID;

IF :NEW.AMOUNT < PRODUCT_PRICE **OR** :NEW.AMOUNT < MAX_BID **THEN**
RAISE_APPLICATION_ERROR(-20001, 'BID_VALUE SHOULD BE GREATER THAN MINIMUM VALUE!');
END IF;
END;

 Script Output x

    | Task completed in 0.591 seconds

Trigger VALIDATE_BID compiled

5.2.2 Trigger to update number of bids

```
CREATE OR replace TRIGGER on_add_bid
  AFTER INSERT ON bid
  FOR EACH ROW
BEGIN
  UPDATE auction
  SET   no_of_bids = no_of_bids + 1
  WHERE seller_user_id = :NEW.seller_user_id;
END;
```

