DBMS FINAL DEADLINE

Team Name- Null-Pointers
Domain Name- Karigar Bazaar

Team Number-27
1. Adarsh Kumar(2018213)
2. Tanya Sanjay Kumar (2018109)
3. Shreeya Garg (2018415)
4. Gauri Shankar(2018035)
5. Mohd Huzaifa(2018158)

OUR PROJECT:

Our project is an e-commerce application which primarily focuses on reducing the gap between Karigars and an open Global market.

The idea is to remove the middle man so that the Karigars can get full credit for their work and save them from being exploited by the middle man.

Our application can be used by sellers to sell their products, customers to buy the product of their choice by comparing the prices by other sellers, provides an employment opportunity as a delivery man, opens the market to bulk buyers, and exposes sellers to fests and government schemes.

STAKEHOLDERS:

1) Seller-

This application will provide the sellers with a global platform to showcase their talent and sell their products on the global level.

Benefits :

1) They get a bigger market and thus, many potential buyers. Less competition in the market due to greater market size, and thus more profit.

2) Sellers below the poverty line (those who are self-employed and unstable) will be provided benefits from the government / NGOs after associating with our application.
3) Sellers will also be able to bid their items in order to maximize their profit.
4) Various flea markets and other fests will be organized and the sellers will be notified about them. They can choose these fests too as a medium to expand their market.
5) This application basically cuts the middleman. This provides a direct link between the seller and the buyer. Sellers cannot be exploited this way.

Queries:

1. Sellers can know their competition in the market.
   ➔ They can do so by getting the count of other sellers who sell the same product and also their selling price. They can find these details in the product table using the seller id, product name and price.
2. Sellers can update their prices or products.
   ➔ They can update the product table and modify the values in prices or add another column for new products.
3. Sellers can know the details of the government schemes they have to access to.
   ➔ They can do so by finding the details about govt schemes from govt table.
4. Sellers can know about upcoming fests to participate in.
   ➔ They can do so by finding details about fests from fest table.
5. Sellers can also bid their products.
   ➔ We have provided the sellers with the bidding option. They can opt for that to get maximum price for their product.

2) Customer-

Customers will use this application to buy Arti-crafts like paintings, pots, decorative items, wall hangings, Bed table, chairs, hand made sweaters, woollen socks, woollen sweatshirts and food items like pickles, pappad, cakes, biscuits, etc at a reasonable rate.

Ways a customer can use this application-

1)Customers can directly buy from the producer of the item rather than dealing with the retailer. This will ensure that customers pay the right price of the item.

2)Customers can have innovative and cheap hand made items at this store

3)Customers can have doorstep delivery of their favourite items. Customers can shop without going to a store.

4)Customers can compare items of different sellers on an online platform. They can compare the quality and price of different items across India while sitting at home.

5)Customers can also get the list of items arranged according to the prices so that they can scroll and look for the items lying in a particular range of price favourable to them.

6)Customers can also see different sellers and what all products each seller is offering on our online store.

Queries-

1. The customer can add as many items in cart of a particular category.They can order items of varied kinds in the cart.As soon as an item is added in the cart it gets deleted from the available products.Each customer has its own cart when he logs in the program.
2. You can delete the items from the cart too and as soon as you delete an item from the cart then the item gets added in the available product list.
3. When we proceed the program displays the cart with all the products in it.The cart has all the items which the customer has chosen to buy.The final

payable amount is displayed too.All details of the selected items is shown in the cart.

4. The customer can also see all the previous transactions he/she has done so far in our program.
5. To ease the process of selecting items to buy,We have provided two more options.
    a. The customer can view all the available products arranged according to the price in decreasing order.This will allow him to choose the range of price favourable to him and see all items lying in that particular price range.
    b. The customer can view items grouped according to seller id so that he can see all the items of a seller of his/her choice.

3). Delivery Guy-

There would be a delivery guy who would pick up the products from sellers and deliver them to the customers. Whenever a customer order something, corresponding order entry would be placed in the Orders Table. This table is updated on a daily basis. The delivery guy would lookup in this table and deliver the items.

Ways of using the application.(Queries)

1). The delivery guy can get the details of a delivery like date, time, total Priceof the cart/order ,customer ids etc.

➔ For this query, he needs to access deliveries table from our database. the attributes and data types for deliveries table are shown below.

2). Delivery Guy can also get the list of deliveries arranged in the sorted order of time and date,and on the basis of delivery_status..

➔ For this query also, he needs to access deliveries table and order them by time and date attributes.

3). He can mark the Delivery status true for a particular delivery and check delivery status of a particular item too.

➔ For this query, he needs to access and modify the delivery status attribute of deliveries table.

4). He can also view his own details like no of deliveries done, joining date, salary, bonus (for fast delivery), rating, and other personal details. Password would be hidden until he clicks on show button.

➔ For this query, he needs to access delivery guy table. Its attributes and data types are mentioned below.

5). Delivery guy can see the deliveries assigned to him and from that for any delivery he can see the pickup locations of sellers and drop location of customer/bulkbuyer.

➔ He can view his rating by accessing rating attribute from delivery guy table.

4) ORGANISERS-

Organizers will organize fest by inviting different artists, vendors, and self-employed workers. They can also call skillful artists by looking at their earnings and rating. They can look for possible sponsors in the way of organising fests and thereby will be able to create a huge market which will benefit both seller and customers. As customers can enjoy fests and also buy the needful whereas sellers can maximize their selling and hence profit. Organisers can also earn in this way by keeping a fest ticket which will be charged from sellers as well as customers.

Ways by which organizers can use this database

.

1) Organizers can see data of different vendors for making their event joyful.
    ➔ They can go to sellers table and with seller id they can check what type of products they do sell from going to products table
2) Organizers can also look for sponsors in the database from artists whose salary is high and wishes to promote their product.
    ➔ They can go to sellers table and see who all are above poverty line and can communicate with them for sponsorship possibilities.
3) Organizers can get contact number of artists for the exhibition of their products.
    ➔ They can see the products they made by going to products table with seller id as foreign key and if it is artistic and popular they can contact them for exhibition

4) Organisers can see their profile.
    ➔ They can simply see the organiser list tuple matched with their IDs.

5) Organisers can add next fest organised by them to be made publicly available so as to attract maximum participants in the fest.
    ➔ They will give required information for fests tuple and it will get inserted into fests table.

5) BULK BUYERS-

Bulk Buyers can use this application to buy goods directly from the Sellers in large quantities and sell at their stores. This even helps the “Karigars” as they can receive big payments at once after selling a large number of their products in one go.

Why will Bulk Buyers use our Application

1) They get to select a large variety of goods for their stores as we have a large number of “Karigars” specialising in different arts and forms.

2) They get to compare prices for similar products for different sellers
3) They don’t have to worry about the hustle of delivery as we provide shipping directly to them with our verified Delivery Guys.

4) Can **bid** on Exclusive items(for more info see Innovation)
5) Can view fests where Karigars come to showcase their items, this will help them stay up-to-date with the latest trends in the market!
6) The convenience of keeping track of what all they've ordered, which will also help in placing future orders.

Queries:

1. Search the products
   ➔ Simply see the table Products
2. Get orders delivered directly, hassle-free
   ➔ One of our Delivery Guy(from table Delivery Guy) is assigned the respective Delivery from the table Deliveries using the attribute-bk_id and r-id, then the Delivery Guys gets the required address of both the Seller and the Bulk Buyers using s_id and r_id from the table Sellers and Bulk Buyers respectively on the time as mentioned in the attribute delivery_time in table Deliveries
3. For **Bidding**
   ➔ Select an item(from Bidding table) to bid on, see the Base Price and Highest bid along with their current bid and then they can bid again, all bids are stored in Cust_bidding table
   ➔ Get to know if he won the bid by getting a pop-up!
4. To view Nation-wide fest!
   ➔ Just click on "see fests" link and using table fest will show all current fest
5. Keeping track of what they've ordered
   ➔ Show only the orders placed by the respective Bulk Buyer

INNOVATION : (BONUS)

We have implemented a special feature in our application after discussing with sir: **The Bidding System**. Sometimes Karigars take a lot time to make their painting, etc. so they should be paid well for it.

This is a special feature where any seller can bid an exclusive item and keep a base price and bulk buyers all over the world can compete with each other to place a higher bid. When a seller places an item for bidding, the bulk buyers can view the item and the highest bid placed on it till now along with his current bid. Then the buyer can place a higher bid if they want to.

When seller wants to end a bid, he/she can just press stop bidding to end bid of a particular item. The highest bidder at this point wins the item at that price and all Bulk Buyers get to know who won the bid.

ADVANCED FEATURES:

EMBEDDED SQL QUERIES:

QUERY1. CUSTOMER WANTS TO SEE ALL PRODUCTS OF A CERTAIN CATEGORY WITH AT MAX PRICE CUSTOMER HAS TO OFFER:

```
EXEC SQL BEGIN DECLARE SECTION;

INT CREDIT_AMOUNT;

EXEC SQL END DECLARE SECTION;

EXEC SQL

    DECLARE C CURSOR FOR

    SELECT p_id, p_cost from products where p_cost < :CREDIT_AMOUNT;
```

QUERY2. GOVERNMENT WANTS TO ADD SOME AMOUNT TO SELLERS BELOW POVERTY LINE.

```
EXEC SQL BEGIN DECLARE SECTION;
Int BONUS_AMOUNT;
EXEC SQL END DECLARE SECTION;
BONUS_AMOUNT = 1000 ;   // can be any variable
EXEC SQL
declare c cursor for
select * from SELLER where bpl_status = 'Y'
for update;
EXEC SQL
update seller set seller_account= seller_account + BONUS_AMOUNT;
where current of c;
EXEC SQL close c;
```

QUERY 3. WHEN SELLER WANTS TO UPDATE PRICE OF EVERY ITEM BY 5%

```
EXEC SQL BEGIN DECLARE SECTION;
Int Updated_Amt_perct;
EXEC SQL END DECLARE SECTION;
Updated_Amt_perct = 5 ;   // can be any variable
EXEC SQL
declare c cursor for
select * from products where s_id = 123
```

for update;

EXEC SQL

update products set p_cost= p_cost + p_cost*(:Updated_Amt_perc)/100t;

where current of c;

EXEC SQL close c;

QUERY4. WHEN SELLER WANTS TO UPDATE QUANTITY OF A CERTAIN PRODUCT

EXEC SQL BEGIN DECLARE SECTION;

Int Update_Amt_quantity;

EXEC SQL END DECLARE SECTION;

Updated_Amt_quanity = 15 ; // can be any variable

EXEC SQL

declare c cursor for

select * from products where s_id = 123 and p_id = 23;

for update;

EXEC SQL

update products set p_quantity= p_quantity + :Updated_Amt_quantity;

where current of c;

EXEC SQL close c;

QUERY5. IF ORGANISER WANTS TO INCREASE ENTRY FEE OF A CERTAIN FEST

```
EXEC SQL BEGIN DECLARE SECTION;

Int Updated_Fee;

EXEC SQL END DECLARE SECTION;

Updated_Fee = 15 ;   // can be any variable

EXEC SQL

declare c cursor for

select * from Fest where Fest_id = 123 and s_id = 23;

for update;

EXEC SQL

update fest set Entry_fee = :Updated_Fee;

where current of c;

EXEC SQL close c;
```

ADVANCED AGGREGATE SQL QUERIES-:

1)Ranking function implementation-:

In the ranking function we arrange the queries based on rank and we take out ranks based on a feature.

Here products are ranked in decreasing order by their prices.The highest price product is placed on the top and the lowest price product is placed at the bottom.This will help customers to find the products lying in their favourable range of prices.

SQL QUERY-:

```
select p_name, rank() over (order by p_cost desc) as cost_rank from products order by cost_rank;
```

2)Ranking function with partitioning-:

In this we rank the entries based on one feature and we partition the entries into group based on a another feature.Here I have partitioned based on seller id.This will help customers to see all products of a particular seller together.This will allow customer to view all the products of the favourite seller in one go.

SQL QUERY-:

```
select p_name,p_cost,p_quantity,s_id,rank() over (PARTITION BY s_id) as R from products order by s_id,p_cost;
```

DBMS CONCEPTS

INDEX CREATION

```
create index index1 on organiser(organising_group);

create index index2 on organiser(organiser_id);

create index index3 on products(p_id);

create index index4 on deliveries(o_id);

create index index5 on cart(cart_id);

create index index6 on delivery_guy(D_id);

create index index7 on products(p_id);
```

PROCEDURES:

PROCEDURE 1.

```
DELIMITER //
```

```
CREATE PROCEDURE get_maxID_Organiser(out ID1 int)
BEGIN
select max(Organiser_id) into ID1 from organiser;
END//
```

PROCEDURE 2

```
DELIMITER //
CREATE PROCEDURE SelectProducts()
select * from products
 END //
```

PROCEDURE 3

```
DELIMITER//
CREATE PROCEDURE SelectCart_id()
BEGIN
Select Cart_id from cart
END//
```

PROCEDURE 4

```
DELIMITER //
CREATE PROCEDURE SelectC_id()
BEGIN
Select C_id from customer
```

END//

PROCEDURE 5

DELIMITER//

CREATE PROCEDURE DeleteProducts()

BEGIN

DELETE FROM products;

END//

PROCEDURE 6

DELIMITER//

CREATE PROCEDURE SelectProducts()

select * from products;

END//

_DELIMITER //

CREATE PROCEDURE get_maxID_Organiser(out ID1 int)

BEGIN

select max(Organiser_id) into ID1 from organiser;

END//

PROCEDURE 7

```
DELIMITER //

CREATE PROCEDURE get_maxID_Fest(out ID1 int)

BEGIN

select max(Organiser_id) into ID1 from Fest;

END//
```

RELATIONAL ALGEBRAIC QUERIES:

QUERY 1) Displaying items from bidding table of a particular seller say seller with ID 1.

- Select bid_id,bid_name,bid_bp from bidding where s_id=1;

➔ ∏bid_id,bid_name,bid_bp(σ s_id=1 (bidding))

QUERY 2) Displaying product names and their price in the market by other sellers of all notebooks.

- Select p_name,p_cost from products where p_name like “%notebook%”;

➔ ∏p_name,p_cost(σ p_name like(“%notebook%”) (products))

QUERY3) Displaying seller details for organiser to contact them

- select s_name,s_address,s_pin,s_contactno,s_emailid from seller where s_bplstatus = 'N';

➔ ∏s_name,s_address,s_pin,s_contactno,s_emailid(σ s_bplstatus = ‘N’ (seller))

QUERY4) Displaying full details of organiser

- select * from organiser where Organising_group = '"+group_name+"'"

➔ ∏Organiser_id,Organsing_group,Total_event_organised,O_contactno,O_Adress,O_ratingemail d,password(Organiser)

➔

OTHER DBMS CONCEPTS:

We have implemented atomicity in our project and also ensures consistency of database.We have implemented indexing too. We have also implemented advanced aggregate sql functions like ranking and ranking and ranking with partitioning.

TABLES

1)Seller

Attribute name	Data Type	Special keys
s_id	int	Primary Key
s_name	varchar(50)	Not null
s_address	varchar(50)	Not null
s_pin	int	Not null
s_contactno	int	Not null
s_emailid	varchar(60)	Not null, Unique
s_bplstatus	char(1)	Check(s_bplstatus

		in(“Y”,”N”), Default(“N”)
s_password	varchar(20)	Not null

2)PRODUCTS

Attribute name	**Data type**	**Special Keys**
p_id	int	Primary Key
p_name	varchar(30)	Not Null
s_id	int	Foreign Key(references-Seller table(s_id))
p_cost	decimal(8,2)	Not Null
p_quantity	int	Not Null

3)CUSTOMER

Attribute name	**Data Type**	**Special keys**
C_id	int	Primary Key
C_name	varchar(20)	Not null
C_address	varchar(60)	Not null
C_pin	int	Not null
C_contact	int	Not null
C_email	varchar(30)	Not Null
C_password	varchar(20)	Not Null

4)Seller_Commision

Attribute name	Data Type	Special Keys
s_id	int	Foreign key(seller(s_id))
com_value	numeric(6,2)	Not Null
Dolp	date	Not Null

5)CART

Attribute name	Data Type	Special Keys
Cart_id	int	Not null
C_id	int	Not null
P_id	int	Primary Key
P_name	varchar(30)	Not null
p_cost	int	Not null
p_quantity	int	Not null
s_id	int	Not null

6)Bidding__

Attribute name	Data Type	Special Keys
P_id	int	Primary Key

C_id	int	Foreign Key - C_id reference Customer table
C_Bid	Numeric(10,2)	Not null

7) Delivery Guy

Attribute name	**Data Type**	**Special keys**
D_id	int	Primary key
D_name	varchar(20)	Not Null
D_rating	Numeric(2,2)	
D_NoDeliveries	int	Not Null
D_contact	int	Not Null
D_JoinDate	char(10)	Not Null //dd/mm/yyyy
D_bonus	Numeric(6,2)	Not Null
D_salary	Numeric(6,2)	Not Null
D_Addr	Varchar(60)	Not Null
D_password	Varchar(20)	Not Null

8)DELIVERIES

Attribute name	Data Type	Special Keys
O_id	int	Primary key
D_id	int	Foreign key (delivery guy)
c_id	int	Foreign key (customer)
r_id	int	Foreign key (Bulk Buyers)
Cart_id	int	Foreign key (cart)
bk_id	Int	Foreign key (Bulk Orders)
Delivery_status	Boolean	Not Null
delivery_time	char(5)	Not Null. //hh:mm

9)ORGANISER

Attribute name	Data Type	Special keys
Organiser_id	int	Primary key
Organising_group	Varchar(100)	Not null
Total_event_organised	int	Not null // can be 0

Ocontact_number	Varchar(14)	Not null
O_Address	Varchar(100)	Not null
Organiser_rating	Numeric(2,1)	Not null
email_id	Varchar(50)	Not null
password	varchar(50)	Not null

10)Fest

Attribute name	**Data Type**	**Special keys**
Fest_id	int	Primary Key
Organiser_id	int	Foreign key (referencing Organiser table Organiser_id)
fest_location	varchar(100)	Not null
Expected_crowd	int	
Entry_fee	int	Not null
Fest_date	varchar(10)	Not null
Special_Attraction	varchar(500)	

11)BULK BUYERS

Attribute name	**Data Type**	**Special keys**
r_id	int	Primary Key

r_name	varchar(50)	Not null
r_address	varchar(50)	Not null
r_pin	int	Not null
r_contactno	int	Not null
r_emailid	varchar(60)	Not null, Unique
r_account	numeric(10,2)	Unique
r_password	varchar(20)	Not Null

12)ORDERS

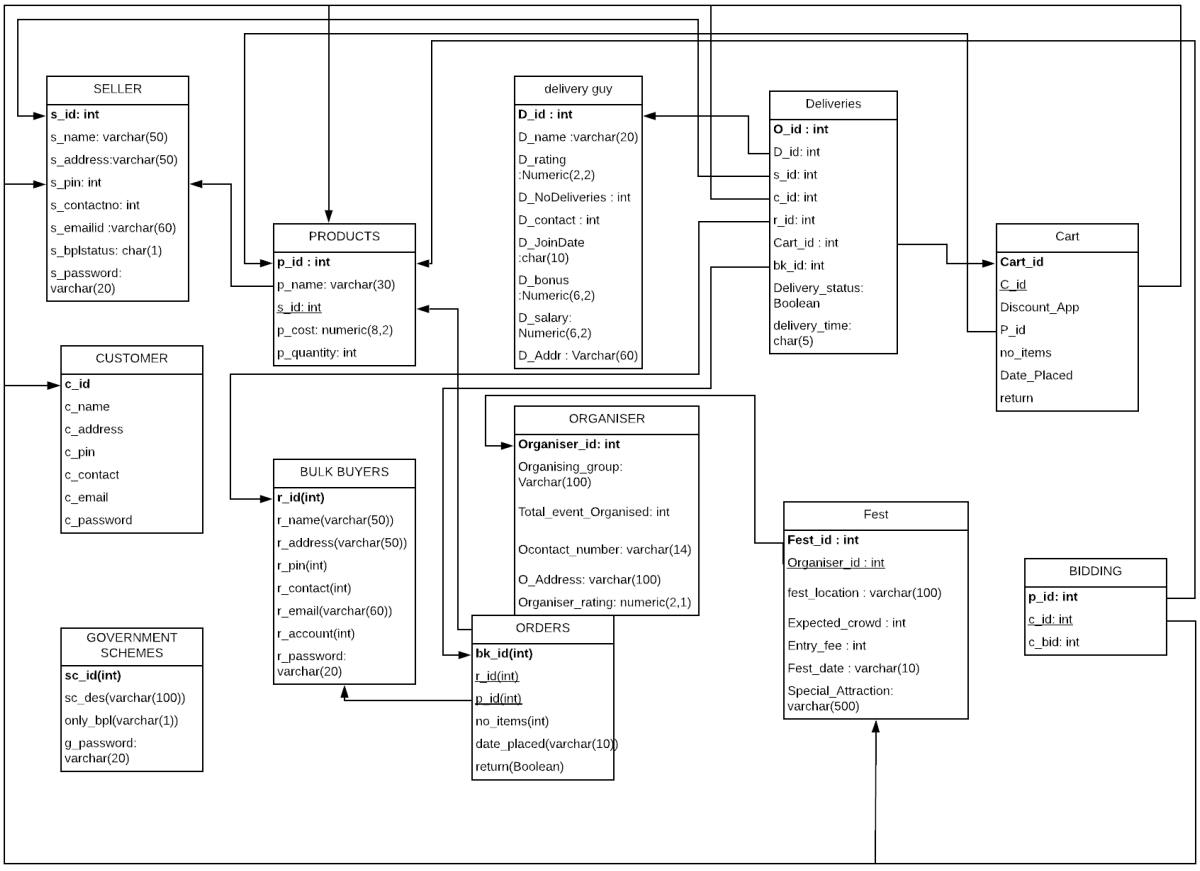
Attribute name	**Data Type**	**Special keys**
bk_id	int	Primary Key
r_id	int	Foreign Key(references-Retailers table(r_id))
p_id	int	Foreign Key(references-Products table(p_id)) Check(p_quantity>0)
no_items	int	Not null
date_placed	varchar(10)	Not null
Return	Boolean	

13)GOVERNMENT SCHEMES

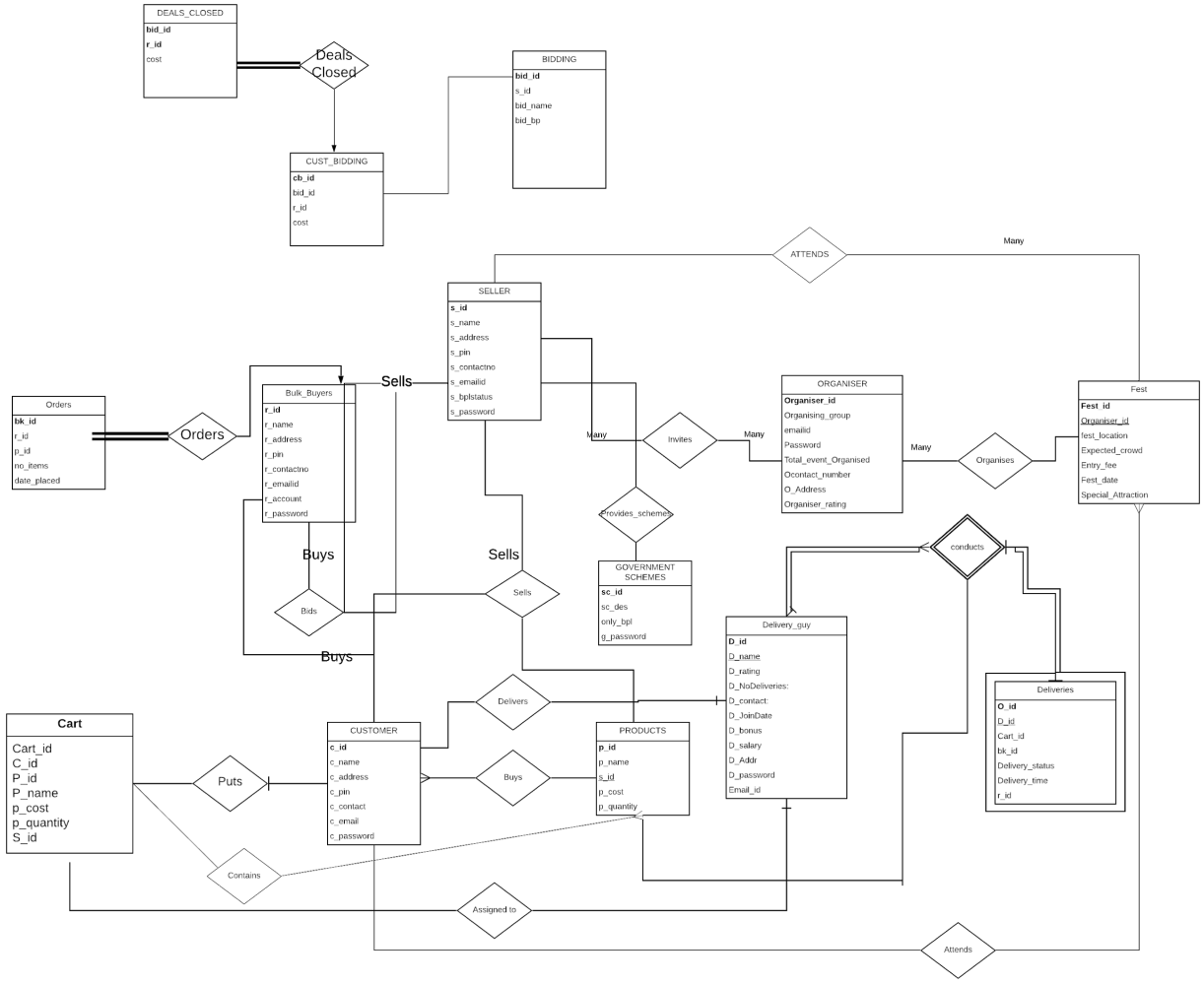
Attribute name	**Data Type**	**Special keys**

sc_id	int	Primary Key
sc_des	varchar(100)	Not null
only_bpl	varchar(1)	Not null Check(only_bpl in(“Y”,”N”), Default(“Y”)
g_password	varchar(20)	Not Null

SCHEMA:



ER DIAGRAM:



Weak entities: Deliveries , orders , deals_closed.

Ternary relation: conducts,sells , buys.

CREATE TABLE QUERIES:

1) SELLER TABLE:

   ->create table Seller(s_id int Primary key,s_name varchar(50) not null,s_address varchar(50) not null,s_pin int not null,s_contactno int not null,s_emailid varchar(60) not null unique,s_bplstatus char(1) check(s_bplstatus in ("Y","N")) default "N",s_password varchar(20) not null);

2) PRODUCTS TABLE:

   ->create table Products(p_id int primary key,D_id int not null,p_name varchar(30) not null,s_id int,p_cost numeric(8,2) not null,p_quantity int not null check(p_quantity>0), foreign key(s_id) references seller(s_id));

3) SELLER_COMMISION TABLE:

   -> create table Seller_Commision(s_id int ,com_value numeric(6,2) not null,Dolp date not null, foreign key(s_id) references Seller(s_id));

4) Delivery_guy

   → create table Delivery_Guy(

   D_id int primary key,

   D_name varchar(20) not null,

   D_rating numeric(2,2),

   D_NoDeliveries int not null,

   D_contact int not null,

D_JoinDate char(10) not null,

D_bonus numeric(6,2) not null,

D_salary numeric(6,2) not null,

D_Addr varchar(60) not null,

D_password varchar(20) not null

);

5) DELIVERIES

→ create table DELIVERIES(

O_id int primary key,

D_id int ,

s_id int,

c_id int,

r_id int,

Cart_id int,

bk_id int,

Delivery_status boolean not null,

delivery_time char(5) not null

);

6)Products Table-:

```
create table products(
                    p_id int primary key,
                    p_name varchar(30) ,
                    s_id int,
                    p_cost decimal(8,2),
                    p_quantity int,
                      );
```

7)Customer Table-:

```
create table customer(
                   C_id int primary key,
                    C_name varchar(20) ,
                    C_address varchar(60),
                    C_pin int,
                    C_contact int,
                    C_email varchar(30),
                    C_password varchar(20)
                     );
```

8)Cart table-:

```
create table cart(
                   Cart_id int primary key,
```

```
C_id int ,

P_id int,

P_name varchar(30),

p_cost int,

p_quantity int,

s_id int

);
```

Work distribution:

Adarsh: Back-end & GUI related to Bulk Buyers. Implemented various features for Bulk Buyer like purchasing items and Order management system. Including **Bonus implementation:** Bidding System where Bulk Buyers places the bid and competes with other bulk buyers and finally the winner is declared based on ranking. Schema diagram.

Tanya Kumar: I have done the front-end and back-end of Seller, implemented various features for the stakeholder like adding/removing items, checking his/her competition in the market, accessing the government schemes available for them and upcoming fests and also the login page for the application. I have also implemented the **bonus/innovative part** of the project : The bidding system. Also the ER diagram.

Shreeya:I have implemented the customers front-end and back end,I have implemented procedures and advanced aggregate sql functions.The customer can buy items,view previous orders and compare items.I have randomly assigned delivery guys to each order placed.After each successful order i have updated the products,cart and deliveries table.Made sign up page for customer.

Gauri Shankar: Have done front-end and back-end part of organiser stake-holder. Also implemented few database concepts like atomicity, indexing and written sql queries , procedures and embedded Queries.

Mohd Huzaifa: Handled All the functionalities of Delivery Guy. Managed deliveries whenever an order is made and Assigned Delivery guy to each of them. Implemented backend and front end for delivery guy stake-holder. Did the front end design(how it looks) part of the overall project. Also integrated different stakeholder's part into one complete project.Schema diagram.