Team 7

IS 680 Database Management Systems Project

Auction Management System

Harshitha Manjunatha

Introduction:

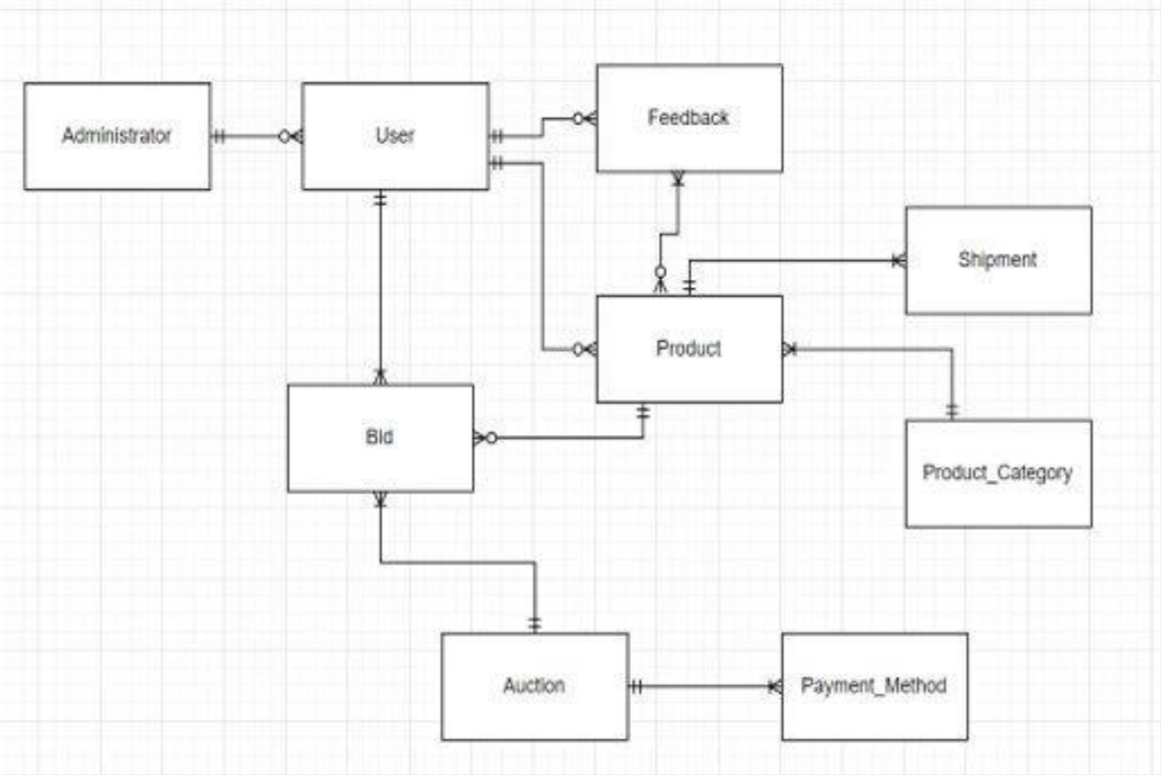
Auction System is basically carrying out the traditional auctioning process in an electronic form, using the Internet. It assists clients with buying or selling anything they want by posting advertisements and requesting bids and choosing the best price for their products. This process would enable clients to post their items for auction; interested bidders can create an account. The need for these kinds of systems could be to give bidders easy access to a reliable reuse platform. These kinds of systems are also beneficial as they allow you to place your bids for items not only in your locality but worldwide., requesting bids, and choosing the best, requesting bids, and choosing the best bid.

In the design of this database, we have included a conceptual model, a logical model, a data dictionary, the physical design of the database, and the application of some of the tasks expected from the database and the related SQL queries.

Problem Statement:

- The current auction system requires people attending auction sessions that are scheduled at specific times. Some would not make it because they were traveling from afar, and thus would miss the sessions the majority of the time.
- Furthermore, the auction houses are not for everyone; they are only for those who have purchased tickets to attend. This restricts people who do not have authorization, even if they may be interested in purchasing some products.
- For example, a person in California may not be aware of an auction taking place in Miami. As a result, the auctions will be attended by a small number of people, and the seller may be unable to obtain the expected price for the product.

A. CONCEPTUAL MODEL



For the conceptual model, we have included **8** entities with appropriate relationships. We used multiple relationship types:

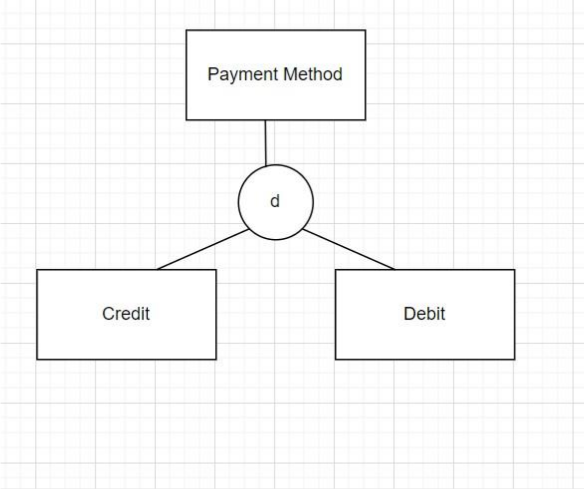
1. Mandatory one.
2. Optional one.
3. Mandatory many.
4. Optional many.

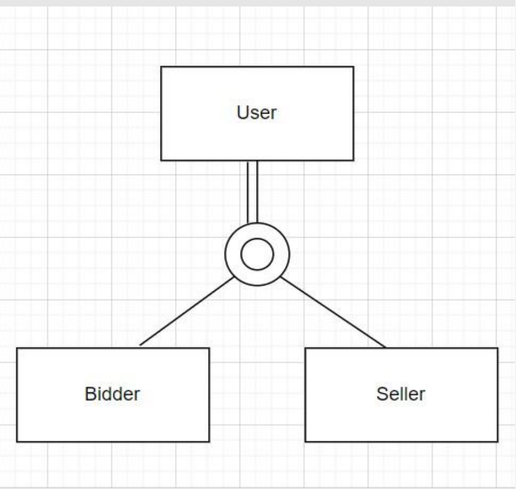
These relationships clearly show the link between entities and help us understand how different entities are connected.

Here, we can see that the majority of the entities are linked to the entity Product as product plays the key role in the entire model. For example, a one-to-many type of cardinality exists in the relationship between the entity user and the entity feedback in which the user can have many feedbacks posted on their names, but all the feedbacks are linked to only one user.

We have also included Supertype – Subtype identity transform to create an identifying relationship between a supertype entity and its subtype entities to simplify the model and improve the performance of the queries. In this model there are ___ types of constraints in the supertypesubtype relationships.

The subtype- supertype relationships in this model are as follows:





There are two types of constraints involved in the model:

Completeness constraint: It indicates whether there must be an explicit subtype for each possible instance of the supertype.

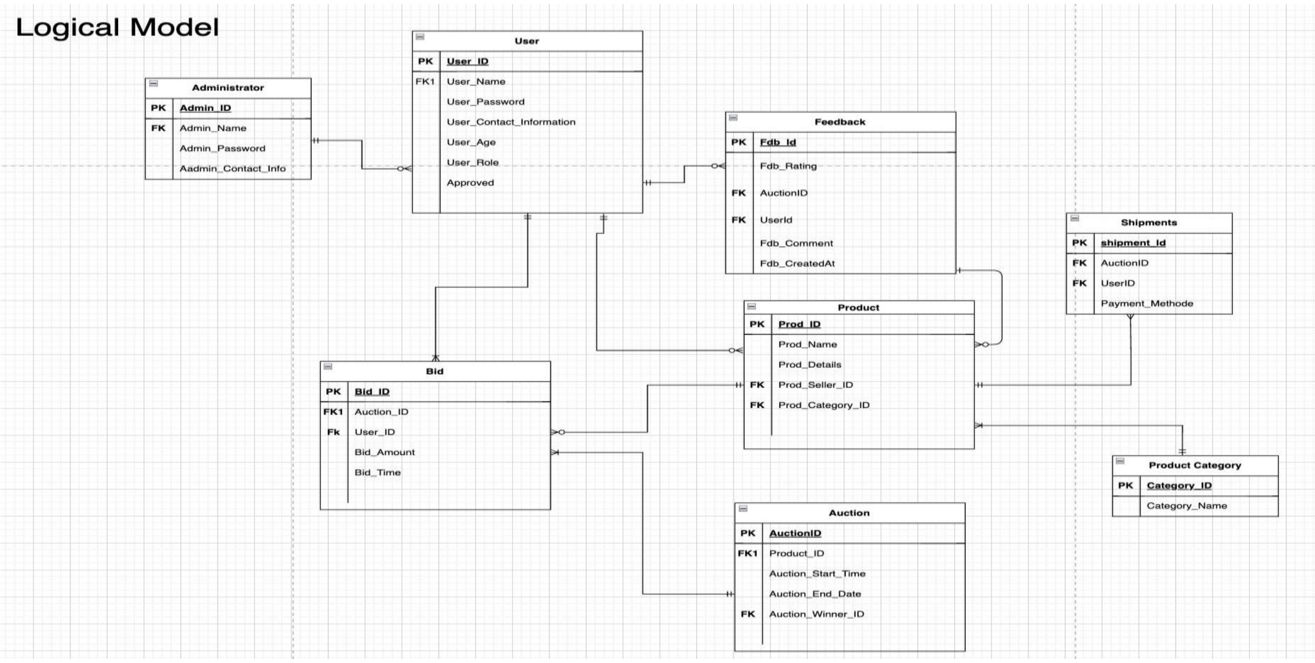
Disjointness constraint: It indicates whether a particular instance of a supertype could also be more than one of the subtypes.

Example:

For the payment Method entity, the method must be either credit or debit card. There is no other possibility. This is represented by single line coming down from the supertype entity type with disjoint(d).

And, for User entity the method should be both Bidder also and Seller also there cannot be only one methode. This is represented by double line coming down from the subtype entity type with overlapping(o).

B. LOGICAL MODEL



This is the complete logical model that we have created. However, as we can see the EER diagrams are difficult to read when there are too many entities and relationships.

In the logical model, we have specified the primary keys and foreign keys, which were not present in the conceptual model. Along with the keys, we also add the attributes that were not specified in the conceptual data model.

C. DATA DICTIONARY

Data dictionary contains all the attributes and their datatypes along with their function and importance

Entity: User

Attribute	Data Type	Range	Description
UserID (PK)	Integer	00001- 99999	Unique identifier for a user
Username	Varchar	Up to 255 characters	Username of the user
Password	Varchar	Up to 255 characters	The password of the user
Contact Information	Varchar	Up to 255 characters	Contact information of the user
Age	Integer	18 and above	Age of the user

Entity: Administrator

Attribute	Data Type	Range	Description
AdminID (PK)	Integer	00001- 99999	Unique identifier for an administrator
Username	Varchar	Up to 255 characters	Username of the administrator
Password	Varchar	Up to 255 characters	Password of the administrator
Contact Information	Varchar	Up to 255 characters	Contact information of the administrator

Entity: Product

Attribute	Data Type	Range	Description
ProductID (PK)	Integer	00001- 99999	Unique identifier for a product
ProductName	Varchar	Up to 255 characters	Name of the product
ProductDetails	Text	00001- 99999	Details about the product
SellerID (FK)	Integer	00001- 99999	Foreign key referencing User.UserID
CategoryID (FK)	Integer	00001- 99999	Foreign key referencing Product_Category.CategoryID

Entity: Auction

Attribute	Data Type	Range	Description
AuctionID (PK)	Integer	00001- 99999	Unique identifier for an auction

ProductID (FK)	Integer	00001-99999	Foreign key referencing Product.ProductID
BiddingStartTime	DateTime	Timestamp	Start time of bidding for the auction
BiddingEndTime	DateTime	Timestamp	End time of bidding for the auction
WinnerID (FK)	Integer	00001-99999	Foreign key referencing User.UserID

Entity: Feedback

Attribute	Data Type	Range	Description
FeedbackID (PK)	Integer	00001-99999	Unique identifier for a feedback
AuctionID (FK)	Integer	00001-99999	Foreign key referencing Auction.AuctionID
UserID (FK)	Integer	00001-99999	Foreign key referencing User.UserID
FeedbackDetails	Text		Details about the feedback(Description)

Entity: Bid

Attribute	Data Type	Range	Description
BidID (PK)	Integer	00001-99999	Unique identifier for a bid

AuctionID (FK)	Integer	00001-99999	Foreign key referencing Auction.AuctionID
UserID (FK)	Integer	00001-99999	Foreign key referencing User.UserID
BidAmount	Decimal		Amount of the bid

Entity: Product_Category

Attribute	Data Type	Range	Description
Category_ID (PK)	Integer	0001-99999	Unique identifier for a product category
CategoryName	Varchar	Up to 255 characters	Name of the product category

Entity : Shipment

Attribute	Data Type	Range	Description
ShipmentID (PK)	Integer	0001-99999	Unique identifier for a shipment
AuctionID (FK)	Integer	0001-99999	Foreign key referencing Auction.AuctionID
UserID (FK)	Integer	0001-99999	Foreign key referencing User.UserID

PURPOSE OF THE DATABASE:

As mentioned in the introduction, the aim of the auction system is to make the bidding process easier for both the seller and the buyer. We would need to create a database solution that will meet our business needs to automate and speed up a lot of day-to-day tasks that would otherwise take a lot of time if done manually.

The aim of creating a database solution is to develop a database application that will help solve the increasing problems of data management by facilitating efficient data storage and retrieval whenever needed. To do that, we need to make sure that the database is created correctly, so that the retrieval is efficient, and storage is correct.

D. PHYSICAL DESIGN

Having a physical design can make managing complex data simpler and improve query performance. If the tables are clearly created with that could be made from the main table. For example, the table User has an attribute Name, which could be split into FirstName and LastName. Denormalization is one of the theories which comes into picture. It can be referred to as a process of taking a normalized table and undoing the normalization to enhance the query performance. Even if normalization is a good way of handling a database to make its understanding better, it makes query processing complicated as it would have to refer to multiple tables making it more time consuming.

Since we are designing a database for a online auction system, we need our attributes to access the table which has just the relevant information. In our case, we need to make sure that the data in the database is up-to-date as people might view the items which are already auctioned.

We need to make sure that the data is stored in a proper data-type format. Our scenario requires that we use 3 different data types. For all the attributes that requires input as a number, such as UserID, AuctionID, SellerID, ProductID, we use the INT datatype. Similarly, for attributes relating to date we used the DATE datatype so that it maintains the readability of the information. Finally, we use the VARCHAR datatype for any attributes relating to names, addresses or payment types for variability of the responses.

The database for our system mainly depends on how many users we have who are willing to put their stuff out for auctions and how many buyers are willing to bid for those items.

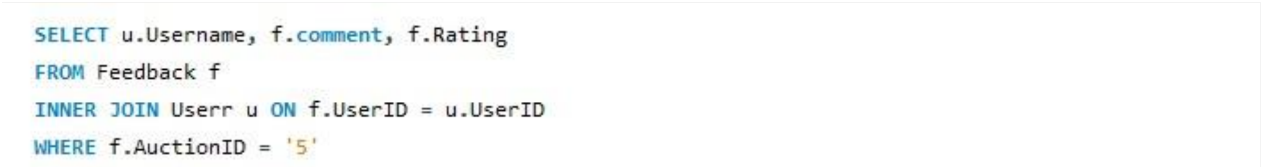
We have chosen to create our own database big enough to implement the queries to make sure it works efficiently and can add more data as time goes on. In a case with a large database, we can use indexed file organization method, where index IDs are mostly the primary keys, I.e. UserID, AuctionID, SellerID etc. This method enables the queries to locate and retrieve data more efficiently.

E. SQL QUERIES

After creating the actual design of the database, to make it function, we will need to implement the database into SQL, for this application, the flavor of SQL that we chose is MYSQL

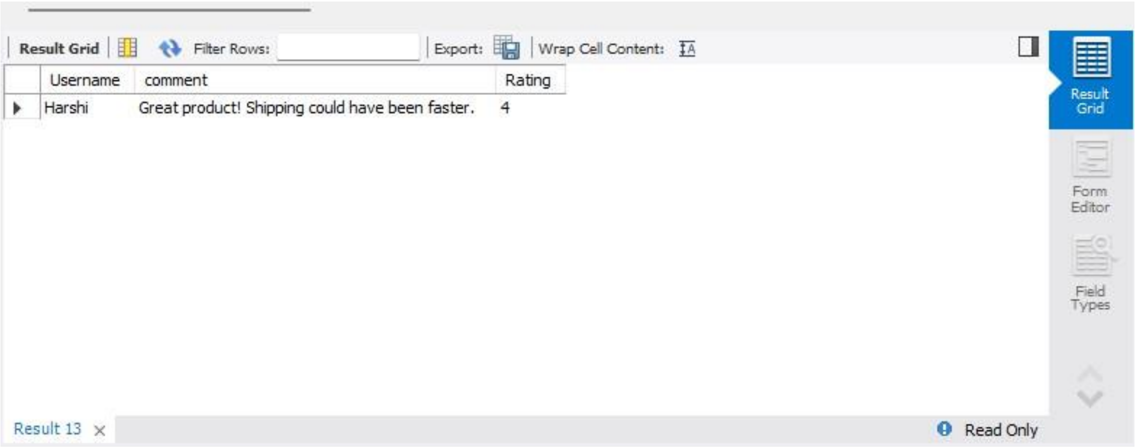
QUERY 1

Retrieve the username, comment and ratings for all the users who have provided feedback for the item which was auctioned at Auction_ID = 5



```
SELECT u.Username, f.comment, f.Rating
FROM Feedback f
INNER JOIN Userr u ON f.UserID = u.UserID
WHERE f.AuctionID = '5'
```

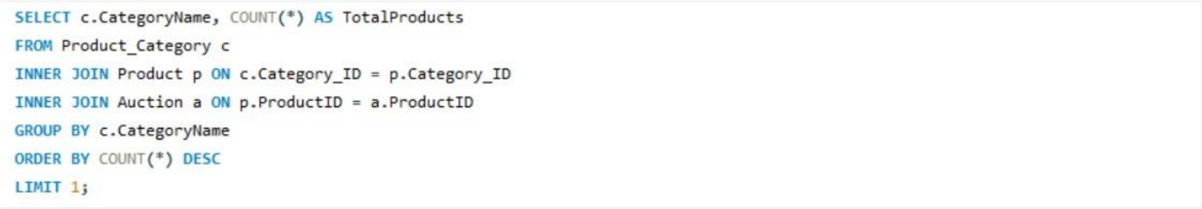
Hit:



Username	comment	Rating
Harshi	Great product! Shipping could have been faster.	4

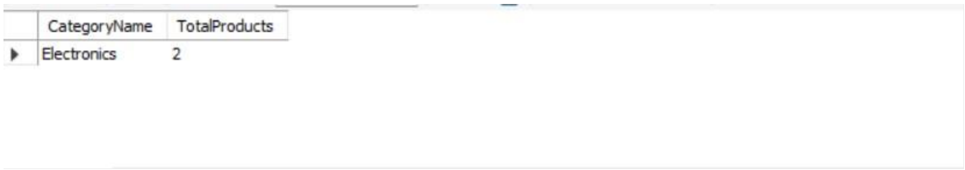
QUERY 2

Retrieve the product category with the highest number of products currently available for auction.



```
SELECT c.CategoryName, COUNT(*) AS TotalProducts
FROM Product_Category c
INNER JOIN Product p ON c.Category_ID = p.Category_ID
INNER JOIN Auction a ON p.ProductID = a.ProductID
GROUP BY c.CategoryName
ORDER BY COUNT(*) DESC
LIMIT 1;
```

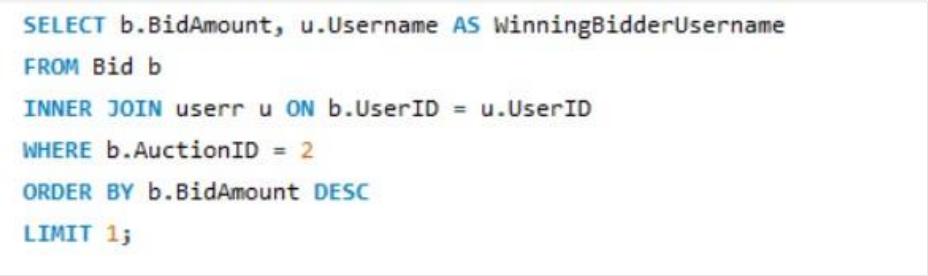
Hits:



CategoryName	TotalProducts
Electronics	2

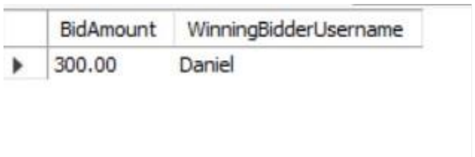
QUERY 3

Retrieve the details of the winning bid for a specific auction, including the bid amount and the username of the winning bidder.



```
SELECT b.BidAmount, u.Username AS WinningBidderUsername
FROM Bid b
INNER JOIN userr u ON b.UserID = u.UserID
WHERE b.AuctionID = 2
ORDER BY b.BidAmount DESC
LIMIT 1;
```

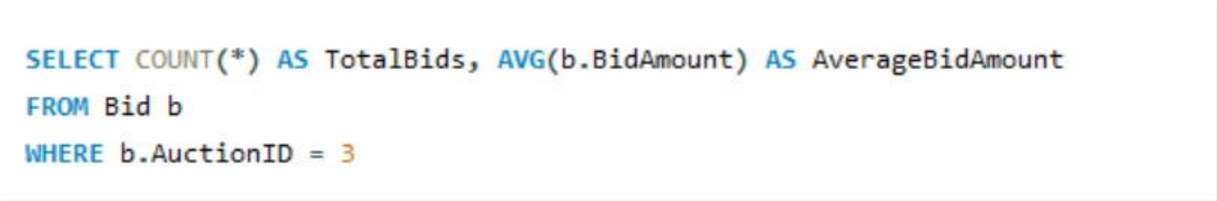
Hits:



BidAmount	WinningBidderUsername
300.00	Daniel

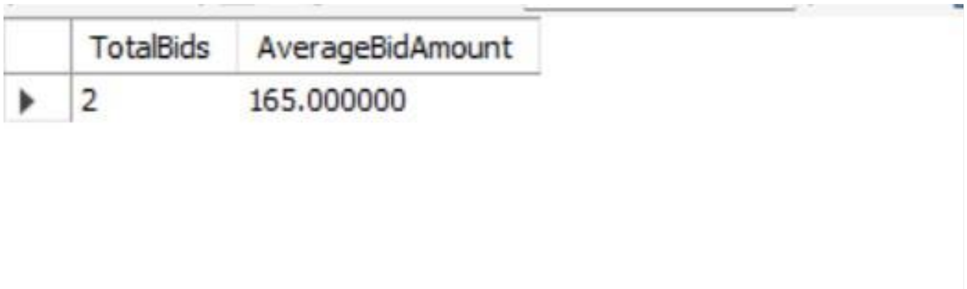
QUERY 4

Retrieve the total number of bids and the average bid amount for a specific auction



```
SELECT COUNT(*) AS TotalBids, AVG(b.BidAmount) AS AverageBidAmount
FROM Bid b
WHERE b.AuctionID = 3
```

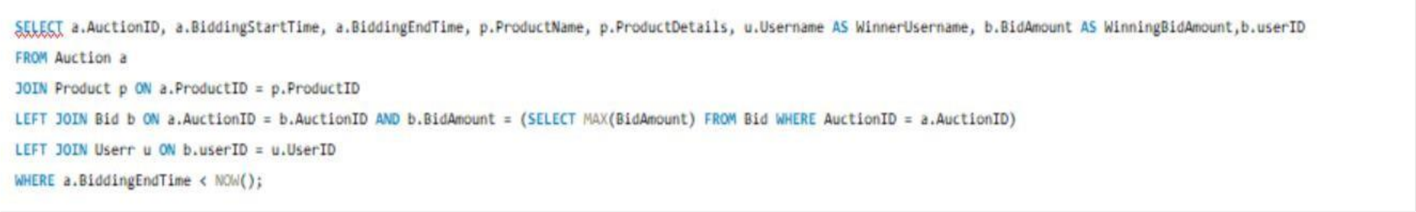
Hits:



TotalBids	AverageBidAmount
2	165.000000

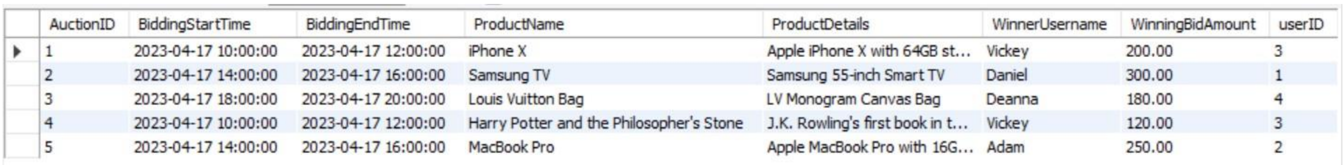
QUERY 5

Retrieve the details of all the auctions that have ended



```
SELECT a.AuctionID, a.BiddingStartTime, a.BiddingEndTime, p.ProductName, p.ProductDetails, u.Username AS WinnerUsername, b.BidAmount AS WinningBidAmount,b.userID
FROM Auction a
JOIN Product p ON a.ProductID = p.ProductID
LEFT JOIN Bid b ON a.AuctionID = b.AuctionID AND b.BidAmount = (SELECT MAX(BidAmount) FROM Bid WHERE AuctionID = a.AuctionID)
LEFT JOIN Userr u ON b.userID = u.UserID
WHERE a.BiddingEndTime < NOW();
```

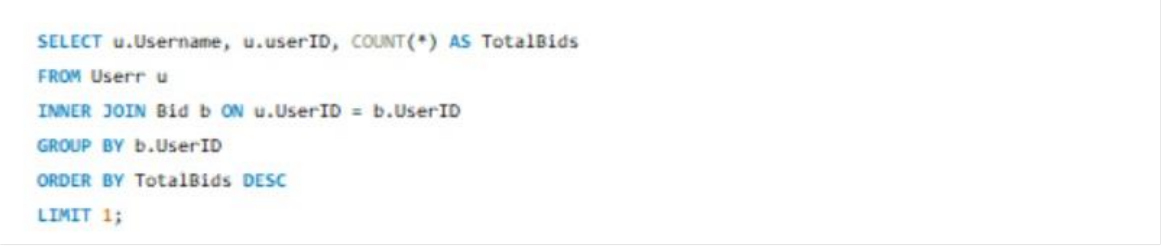
Hit:



	AuctionID	BiddingStartTime	BiddingEndTime	ProductName	ProductDetails	WinnerUsername	WinningBidAmount	userID
▶	1	2023-04-17 10:00:00	2023-04-17 12:00:00	iPhone X	Apple iPhone X with 64GB st...	Vickey	200.00	3
	2	2023-04-17 14:00:00	2023-04-17 16:00:00	Samsung TV	Samsung 55-inch Smart TV	Daniel	300.00	1
	3	2023-04-17 18:00:00	2023-04-17 20:00:00	Louis Vuitton Bag	LV Monogram Canvas Bag	Deanna	180.00	4
	4	2023-04-17 10:00:00	2023-04-17 12:00:00	Harry Potter and the Philosopher's Stone	J.K. Rowling's first book in t...	Vickey	120.00	3
	5	2023-04-17 14:00:00	2023-04-17 16:00:00	MacBook Pro	Apple MacBook Pro with 16G...	Adam	250.00	2

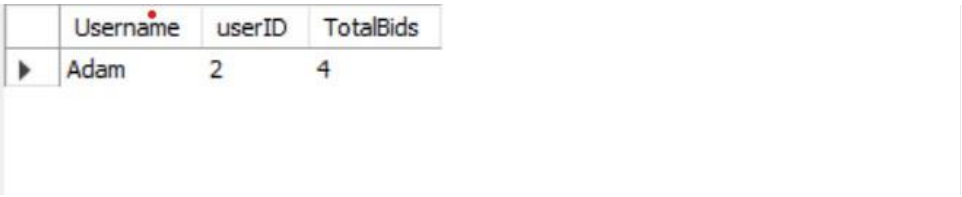
QUERY 6

Retrieve username, userID and total number of bids for a user with highest number of bids.



```
SELECT u.Username, u.userID, COUNT(*) AS TotalBids
FROM Userr u
INNER JOIN Bid b ON u.UserID = b.UserID
GROUP BY b.UserID
ORDER BY TotalBids DESC
LIMIT 1;
```

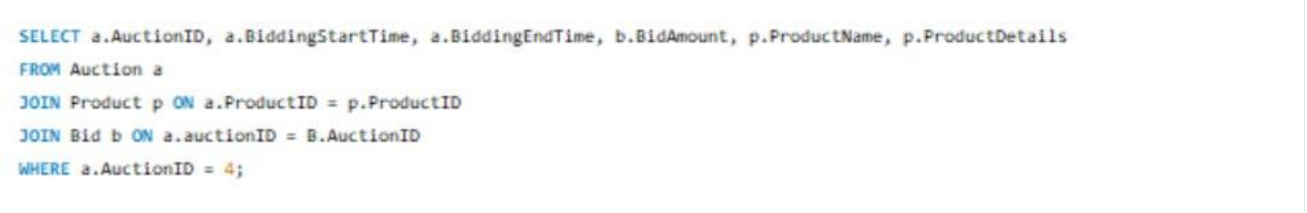
Hits:



	Username	userID	TotalBids
▶	Adam	2	4

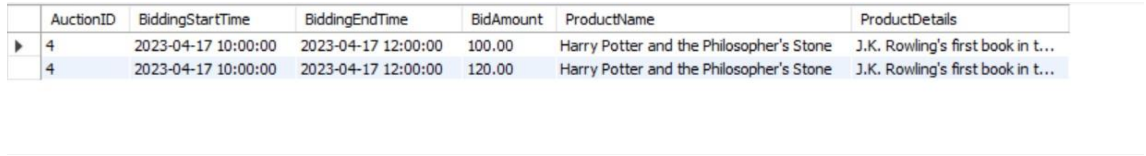
QUERY 7

Retrieve all the bids presented for a particular auction



```
SELECT a.AuctionID, a.BiddingStartTime, a.BiddingEndTime, b.BidAmount, p.ProductName, p.ProductDetails
FROM Auction a
JOIN Product p ON a.ProductID = p.ProductID
JOIN Bid b ON a.auctionID = B.AuctionID
WHERE a.AuctionID = 4;
```

Hit:



	AuctionID	BiddingStartTime	BiddingEndTime	BidAmount	ProductName	ProductDetails
▶	4	2023-04-17 10:00:00	2023-04-17 12:00:00	100.00	Harry Potter and the Philosopher's Stone	J.K. Rowling's first book in t...
	4	2023-04-17 10:00:00	2023-04-17 12:00:00	120.00	Harry Potter and the Philosopher's Stone	J.K. Rowling's first book in t...