## VBay online Auctions

First things first, before jumping into answering questions, I like to get familiar with the objects/tables and how they are connected.

A diagram of a database

Description automatically generated with medium confidence

Show all the tables to get familiar:

All bids

A screenshot of a computer

Description automatically generated

Item type look-up table

A screenshot of a computer

Description automatically generated

All Items

A screenshot of a computer

Description automatically generated

Rating type look-up tableA screenshot of a computer

Description automatically generated

All user ratings

A screenshot of a computer

Description automatically generated

All users

A screenshot of a computer

Description automatically generated

All user zip-codes

A screenshot of a data

Description automatically generated

Filter to only show collectables

A screenshot of a computer

Description automatically generated

Project to show **item\_name, item\_type, item\_reserve, and** **item\_soldamout** for collectables

A screenshot of a computer

Description automatically generated

Sort previous output by item name

A screenshot of a computer

Description automatically generated

Now let’s start looking at the seller’s information for the antiques. Let’s join **vb\_items** and **vb\_users** to determine who are the sellers of all items

\*\*output too large to display all results\*\*

A computer code with text

Description automatically generated with medium confidence

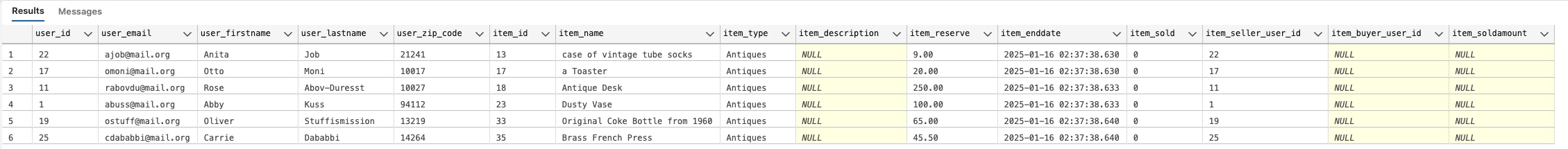
A screenshot of a computer

Description automatically generated

Filter rows to show sellers of antiques

A computer screen shot of a computer code

Description automatically generated



Project only the columns needed for previous output

A screenshot of a computer

Description automatically generated

Now let’s join the items table to the user’s table *twice* to show both buyers and sellers’ information for each item

\*\*wide output, two screenshots attached\*\*

select \*

from vb\_items as i

join vb\_users as s on s.user\_id = i.item\_seller\_user\_id

join vb\_users as b on b.user\_id = i.item\_buyer\_user\_id;

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Join users’ tables twice to show the buyers and sellers for each item sold labeled appropriately, along with the difference between **item\_reserve** amount and **item\_soldamount** to show the profit/loss margin

\*\*wide output, two screenshots attached\*\*

select s.user\_email as sellers\_email,

b.user\_email as buyers\_email,

i.item\_soldamount - i.item\_reserve as item\_margin,

i.\*

from vb\_items as i

join vb\_users as s on s.user\_id = i.item\_seller\_user\_id

join vb\_users as b on b.user\_id = i.item\_seller\_user\_id

where item\_sold >= 1;

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

--Sort with highest margin first

select s.user\_email as sellers\_email,

b.user\_email as buyers\_email,

i.item\_soldamount - i.item\_reserve as item\_margin,

i.\*

from vb\_items as i

join vb\_users as s on s.user\_id = i.item\_seller\_user\_id

join vb\_users as b on b.user\_id = i.item\_buyer\_user\_id

where i.item\_sold = 1

order by item\_margin DESC;

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

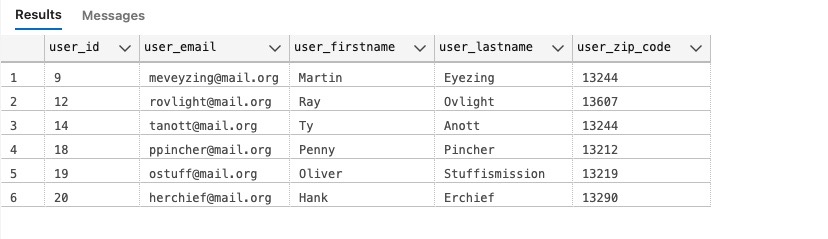
AI-generated content may be incorrect.

Sales would like to send mailings to users who live in a zip code that starts with ‘13’, for example 13244, so they can be notified of their new contact in that region.

select \*

from vb\_users

where user\_zip\_code like '13%';



Find all users from state of NY. Print their names, emails, city, state, and zip code.

Sort by city, then user’s last/first name.

select u.user\_email, u.user\_firstname, u.user\_lastname, z.zip\_city, z.zip\_state, z.zip\_code

from vb\_users as u

join vb\_zip\_codes as z on u.user\_zip\_code = z.zip\_code

where z.zip\_state = 'NY'

order by z.zip\_city, u.user\_lastname, u.user\_firstname;

A screenshot of a computer

AI-generated content may be incorrect.

High-priced items: Return the ID, name, type, and reserve of items that have not been sold and have a reserve of 250 or higher. Sort by city, then by user's last/first name

select item\_id, item\_name, item\_type, item\_reserve

from vb\_items

where item\_sold = 0

AND item\_reserve >= 250

order by item\_reserve DESC;

A screenshot of a computer

AI-generated content may be incorrect.

Reserve item categories. Include the ID, name, type, and reserve price of the item. Do not include items of type “All Other”. Create a category column based on item reserve price.  
When the item is 250 or more, it is a high-priced item.  
When the item is 50 or less, it is a low-priced item.  
Everything else is an average-priced item.

select item\_id, item\_name, item\_type, item\_reserve,

case when item\_reserve >= 250 then 'high-priced item'

when item\_reserve <=50 then 'low-priced item'

else 'average-priced item'

end as reserve\_item\_category

from vb\_items

where NOT item\_type = 'All Other';

A screenshot of a computer

AI-generated content may be incorrect.

Write a query that displays the valid user bids (bid status of ‘ok’) for a given **item\_id**. This  
would commonly be displayed on the website for the chosen item. You select the item ID to display and show the bid ID, bid user’s name, bid user email, bid date, and bid amount. Put the most recent bids at the top.

select b.bid\_id, b.bid\_item\_id, u.user\_firstname, u.user\_lastname, u.user\_email, bid\_datetime, b.bid\_amount, b.bid\_status

from vb\_bids as b

join vb\_users as u

on b.bid\_user\_id = u.user\_id

where b.bid\_status='ok'

order by bid\_datetime DESC;

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

The bad bidder list. Write a query to help the security audit team find fraudulent activity. For any bid that does not have a status of ‘ok’, include the date of the bid, name, email, and ID of the bidder and the name and ID of the item bid upon. Also include the amount of the bid and bid status. Sort the output by the user name (last, then first) and then by bid date for users with multiple bad bids.

select b.bid\_datetime,u.user\_firstname, u.user\_lastname, u.user\_email, u.user\_id, i.item\_name, i.item\_id, b.bid\_amount, b.bid\_status

from vb\_bids as b

join vb\_users as u

on b.bid\_user\_id = u.user\_id

join vb\_items as i

on b.bid\_item\_id = i.item\_id

where NOT bid\_status= 'ok'

order by u.user\_lastname, u.user\_firstname, b.bid\_datetime;

A screenshot of a computer

AI-generated content may be incorrect.

Product a report of items that do not contain a bid. Include the item ID, item name item type, seller’s name, and item reserve.

select i.item\_id,i.item\_name, i.item\_type,i.item\_reserve, b.bid\_status, concat(u.user\_firstname, ' ', u.user\_lastname) as seller\_name

from vb\_items i

left join vb\_bids b

on i.item\_id= b.bid\_item\_id

join vb\_users u

on i.item\_seller\_user\_id=u.user\_id

where b.bid\_status IS null;

A screenshot of a computer

AI-generated content may be incorrect.

Produce a list of seller ratings. Include the name of the user who gave the rating, the name of the user the rating was for, the rating value, and the rating comment. Include ratings of only sellers.

select concat(g.user\_firstname, ' ', g.user\_lastname) as rating\_giver, concat(t.user\_firstname, ' ', t.user\_lastname) as person\_rated,

r.rating\_value, r.rating\_comment, r.rating\_astype

from vb\_user\_ratings r

join vb\_users g

on r.rating\_by\_user\_id = g.user\_id

join vb\_users t

on r.rating\_for\_user\_id = t.user\_id

where r.rating\_astype='Seller';

A screenshot of a computer

AI-generated content may be incorrect.

For items that were sold, generate a report that includes the locations (city and state) of the buyer and seller. Include item ID, item name, item type, item sold amount, name of seller, seller’s city/state, name of buyer, and the buyer’s city/state.

select i.item\_id, i.item\_name, i.item\_type, i.item\_soldamount, concat(s.user\_firstname, ' ', s.user\_lastname) as seller\_name,

concat(sz.zip\_city,' ', sz.zip\_state) as seller\_city\_state, concat(b.user\_firstname, ' ', b.user\_lastname) as buyer\_name,

concat(bz.zip\_city, ' ', bz.zip\_state) as buyer\_city\_state

from vb\_items i

join vb\_users s

on i.item\_seller\_user\_id=s.user\_id

join vb\_users b

on i.item\_buyer\_user\_id=b.user\_id

join vb\_zip\_codes sz

on s.user\_zip\_code=sz.zip\_code

join vb\_zip\_codes bz

on b.user\_zip\_code=bz.zip\_code;

A screenshot of a computer

AI-generated content may be incorrect.

Users with no activity. Find the names and emails of any users who have never posted an item for bid or have never bought the item or have never placed a bid.

select u.user\_firstname, u.user\_lastname, u.user\_email, bi.item\_buyer\_user\_id,si.item\_seller\_user\_id, bd.bid\_user\_id

from vb\_users u

left join vb\_items si

on si.item\_seller\_user\_id=u.user\_id

left join vb\_items bi

on bi.item\_buyer\_user\_id=u.user\_id

left join vb\_bids bd

on bd.bid\_user\_id=u.user\_id

where bi.item\_buyer\_user\_id is null

and si.item\_seller\_user\_id is null

and bd.bid\_user\_id is null;

A screenshot of a computer

AI-generated content may be incorrect.

For all items, include the name of the item, the reserve price, the lowest bid, highest bid, and sold amount. Put items with the largest reserve first.

select item\_name, item\_reserve, min(bid\_amount) as min\_bid,

max(bid\_amount) as max\_bid, item\_soldamount

from vb\_items

join vb\_bids on item\_id=bid\_item\_id

where bid\_status = 'ok'

group by item\_name, item\_reserve, item\_soldamount

order by item\_reserve desc;

A screenshot of a computer

AI-generated content may be incorrect.

Show info with a count of bids including rows with no bids

select s.user\_email, s.user\_firstname, s.user\_lastname, count(\*) as bid\_counts

from vb\_users s

left join vb\_bids b

on b.bid\_user\_id=s.user\_id

where b.bid\_status = 'ok'

group by s.user\_email, s.user\_firstname, s.user\_lastname;

A screenshot of a computer

AI-generated content may be incorrect.

vBay! would like to classify their users based on the numbers of valid bids they have placed.

Low Activity = 0 or 1 bids  
Moderate Activity = 2 to 4 bids  
High Activity = 5 or more bids  
Then they would like to produce a report counting the number of users who fall into low, moderate, and  
high activity categories.

select s.user\_email, s.user\_firstname, s.user\_lastname, count(\*) as bid\_counts,

case when count(\*) between 0 and 1 then 'Low'

when count(\*) between 2 and 4 then 'Moderate'

else 'High' end as user\_bid\_activity

from vb\_users s

left join vb\_bids b

on b.bid\_user\_id=s.user\_id

where b.bid\_status = 'ok'

group by s.user\_email, s.user\_firstname, s.user\_lastname;

A screenshot of a computer

AI-generated content may be incorrect.

Produce Activity report from previous query by creating table using "WITH" statement to name query

with user\_bids as (

select s.user\_email, s.user\_firstname, s.user\_lastname, count(\*) as bid\_counts,

case when count(\*) between 0 and 1 then 'Low'

when count(\*) between 2 and 4 then 'Moderate'

else 'High' end as user\_bid\_activity

from vb\_users s

left join vb\_bids b

on b.bid\_user\_id=s.user\_id

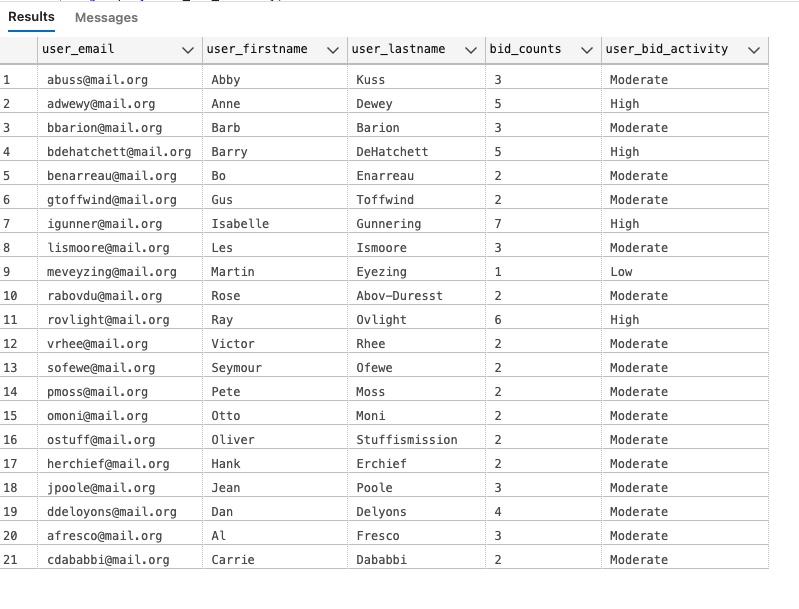
where b.bid\_status = 'ok'

group by s.user\_email, s.user\_firstname, s.user\_lastname

)

select \*

from user\_bids;

\*\*\*same result as above\*\*\*

Using previous query as a table, count how many users for each **user\_bid\_activity** category sorted by user\_count

with user\_bids as (

select s.user\_email, s.user\_firstname, s.user\_lastname, count(\*) as bid\_counts,

case when count(\*) between 0 and 1 then 'Low'

when count(\*) between 2 and 4 then 'Moderate'

else 'High' end as user\_bid\_activity

from vb\_users s

left join vb\_bids b

on b.bid\_user\_id=s.user\_id

where b.bid\_status = 'ok'

group by s.user\_email, s.user\_firstname, s.user\_lastname

)

select user\_bid\_activity, count(\*) as user\_count

from user\_bids

group by user\_bid\_activity

order by user\_count;

A screenshot of a computer

AI-generated content may be incorrect.

How many item types are there? For each item type, provide the count of items, in that type, the min, average and max item reserve prices for that type. Sort output by item type

select

item\_type,

count(item\_type) as count\_item\_type,

min(item\_reserve) as min\_item\_reserve,

AVG(item\_reserve) as average\_item\_reserve ,

max(item\_reserve) as max\_item\_reserve

from vb\_items

group by item\_type

order by item\_type;

A screenshot of a computer

AI-generated content may be incorrect.

Perform an analysis of each item in the “Antiques” and “Collectables” item types. For each item,  
display the name, item type, and item reserve. Include the minimum, maximum, and average item  
reserve over each item type so that the current item reserve can be compared to these values.

select item\_name, item\_type, item\_reserve,

min(item\_reserve) over (partition by item\_type) as min\_item\_reserve,

max(item\_reserve) over (partition by item\_type) as max\_item\_reserve,

avg(item\_reserve) over (partition by item\_type) as avg\_item\_reserve

from vb\_items

where item\_type in ('Antiques', 'Collectables');

A screenshot of a table

AI-generated content may be incorrect.

Write a query to include the names, counts (number of ratings), and average seller ratings (as a  
decimal) of users. For reference, User Carrie Dababbi has four seller ratings and an average rating of  
4.75.

select i.item\_name, count(\*) as count\_bids, i.item\_type

from vb\_items i

join vb\_bids b on

i.item\_id=b.bid\_item\_id

where item\_type = 'Collectables'

group by i.item\_name, i.item\_type

having count(\*) > 1

order by count(\*) desc;

A screenshot of a computer

AI-generated content may be incorrect.

Create a list of “Collectable” item types with more than one bid. Include the name of the item and the number of bids, making sure the item with the most bids appears first.

select i.item\_name, count(\*) as count\_bids, i.item\_type

from vb\_items i

join vb\_bids b on

i.item\_id=b.bid\_item\_id

where item\_type = 'Collectables'

group by i.item\_name, i.item\_type

having count(\*) > 1

order by count(\*) desc;

A screenshot of a computer

AI-generated content may be incorrect.

Generate a valid bidding history for any given item of your choice. Display the item ID, item name, a number representing the order the bid was placed,

the bid amount, and the bidder’s name.

select

i.item\_id, i.item\_name, row\_number () over (order by b.bid\_datetime) as bid\_order, b.bid\_amount,

u.user\_firstname + ' ' + u.user\_lastname as bidder

from vb\_bids b

join vb\_items i on

b.bid\_item\_id=i.item\_id

join vb\_users u on

b.bid\_user\_id = u.user\_id

where bid\_item\_id = 36;

A screenshot of a computer

AI-generated content may be incorrect.

Rewrite your query in the previous question to include the names of the next and previous bidders, like this example,

again showing the first three bids for item 11.

select

i.item\_id, i.item\_name,

row\_number () over (order by b.bid\_datetime) as bid\_order,

b.bid\_amount,

lag(u.user\_firstname + ' ' + u.user\_lastname) over (order by b.bid\_datetime) as prev\_bidder,

u.user\_firstname + ' ' + u.user\_lastname as bidder,

lead(u.user\_firstname + ' ' + u.user\_lastname) over (order by b.bid\_datetime) as next\_bidder

from vb\_bids b

join vb\_items i on

b.bid\_item\_id=i.item\_id

join vb\_users u on

b.bid\_user\_id = u.user\_id

where bid\_item\_id = 36;

A screenshot of a computer

AI-generated content may be incorrect.

Find the names and emails of the users who give out the worst ratings (lower than the overall average rating) to either buyers or sellers

(no need to differentiate whether the user rated a buyer or seller), and include only those users who have submitted more than one rating.

with rating\_info as (

select u.user\_firstname + ' ' + u.user\_lastname as rater, u.user\_email, r.rating\_value, count(\*) as count\_of\_ratings,

avg(r.rating\_value) over () as avg\_overall\_rating

from vb\_user\_ratings r

join vb\_users u

on r.rating\_by\_user\_id = u.user\_id

group by u.user\_firstname + ' ' + u.user\_lastname, u.user\_email, r.rating\_value

having count(\*) >1

)

select \*

from rating\_info

where rating\_value < avg\_overall\_rating

order by rating\_value;

A screenshot of a computer

AI-generated content may be incorrect.

Produce a report of the KPI (key performance indicator) user bids per item. Show the user’s name and email, total number of valid bids,

total count of items bid upon, and then the ratio of bids to items. As a check, Anne Dewey’s bids per item ratio is 1.666666.

select

u.user\_firstname + ' ' + u.user\_lastname as bidder\_name, u.user\_email, b.bid\_status,

count(\*) as bid\_count,

count(distinct i.item\_name) as count\_items,

cast(count(\*) as decimal(3,2))/cast(count(distinct i.item\_name) as decimal (3,2)) as ratio\_bid\_to\_item

from vb\_users u

join vb\_bids b

on u.user\_id=b.bid\_user\_id

join vb\_items i

on b.bid\_item\_id = i.item\_id

where bid\_status = 'ok'

group by u.user\_firstname + ' ' + u.user\_lastname,u.user\_email, b.bid\_status;

A screenshot of a computer

AI-generated content may be incorrect.

Among items not sold, show highest bidder name and the highest bid for each item. Make sure to include only valid bids.

with bid\_info as (

select i.item\_id, i.item\_name, u.user\_firstname + ' ' + u.user\_lastname as high\_bidder, b.bid\_amount,

row\_number() over(partition by i.item\_id order by b.bid\_amount desc) as bid\_ranking

from vb\_bids b

join vb\_items i

on b.bid\_item\_id=i.item\_id

join vb\_users u

on u.user\_id=b.bid\_user\_id

where i.item\_sold = 0 and b.bid\_status='ok'

)

select

item\_id, item\_name, high\_bidder, bid\_amount

from bid\_info

where bid\_ranking = 1;

A screenshot of a computer

AI-generated content may be incorrect.

Write a query with output similar to Question 3, but also includes the overall average seller

rating and the difference between each user’s average rating and the overall average. For reference, the overall average seller rating should be 3.2.

with avg\_seller\_rating as (

select u.user\_firstname + ' ' + u.user\_lastname as rater,

avg(cast(rating\_value as decimal(3,2))) over(partition by u.user\_firstname, u.user\_lastname) as avg\_seller\_rating,

avg(cast(rating\_value as decimal(3,2))) over () as overall\_avg\_seller\_rating

from vb\_user\_ratings r

join vb\_users u

on r.rating\_for\_user\_id=u.user\_id

where rating\_astype = 'Seller'

)

select distinct(rater), avg\_seller\_rating, overall\_avg\_seller\_rating,

avg\_seller\_rating-overall\_avg\_seller\_rating as diff\_from\_overall\_rate\_avg

from avg\_seller\_rating;

A screenshot of a computer

AI-generated content may be incorrect.