Unit 05 Problem Set Submission Form

Overview

Your Name	Hongdi Li
Your SU Email	hli248@syr.edu

Instructions

Put your name and SU email at the top. Answer these questions all from the lab. When asked to include screenshots, please follow the screen shot guidelines from the first lab.

Remember as you complete the problem sets it is not only about getting it right / correct. We will discuss the answers in class so it's important to articulate anything you would like to contribute to the discussion in your answer:

- If you feel the question is vague, include any assumptions you've made.
- If you feel the answer requires interpretation or justification provide it.
- If you do not know the answer to the question, articulate what you tried and how you are stuck.

This how you receive credit for answering questions which might not be correct.

Questions

Answer these questions using the problem set submission template. You will need to consult the logical model in the overview section for details. For any screenshots provided, please follow the guidelines for submitting a screenshot.

Write the following as SQL queries. If the query is ambiguous, fill in the gaps yourself and justify your reasoning. For each, include the SQL as a screenshot with the output of the query.

1. How many item types are there? Perform an analysis of each item type. For each item type, provide the count of items in that type, the minimum, average, and maximum item reserve prices for that type. Sort the output by item type.

```sql
--ID:hli248
--Q1:
select item_type, min(item_reserve) as min_reserve,max(item_reserve) as max_reserve,avg(item_reserve) as avg_reserve, count(item_type) as item_sum
    from vb_items
    group by item_type
    order by item_type
```

Results Messages

	item_type	min_reserve	max_reserve	avg_reserve	item_sum
1	All Other	0.99	10000000.00	2500004.86	4
2	Antiques	9.00	250.00	81.5833	6
3	Books	4.50	10.99	8.48	3
4	Collectables	5.00	500.00	105.3828	14
5	Electronics	15.00	15.00	15.00	1
6	Jewelry	6.95	599.99	303.47	2
7	Sporting Goods	12.50	12.75	12.625	2
8	Tickets	5.00	750.00	377.50	2

2. 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 min, max and average item reserve over each item type so that the current item reserve can be compared to these values.

```sql
--ID:hli248
--Q2:
select item_name, item_type, item_reserve, min(item_reserve) as min_reserve,max(item_reserve) as max_reserve,avg(item_reserve) as avg_reserve
    from vb_items
    where item_type='Antiques' or item_type='Collectables'
    group by item_name, item_type, item_reserve
```

Results Messages

	item_name	item_type	item_reserve	min_reserve	max_reserve	avg_reserve
1	a Toaster	Antiques	20.00	20.00	20.00	20.00
2	Alf Alarm Clock	Collectables	5.00	5.00	5.00	5.00
3	Antique Desk	Antiques	250.00	250.00	250.00	250.00
4	Autographed Mik Jagger Poster	Collectables	75.00	75.00	75.00	75.00
5	Brass French Press	Antiques	45.50	45.50	45.50	45.50
6	Carlos Villalba BobbleHead	Collectables	49.95	49.95	49.95	49.95
7	case of vintage tube socks	Antiques	9.00	9.00	9.00	9.00
8	Dukes Of Hazard ashtray	Collectables	149.99	149.99	149.99	149.99
9	Dusty Vase	Antiques	100.00	100.00	100.00	100.00
10	Farrah Fawcet poster	Collectables	50.00	50.00	50.00	50.00
11	Joe Montanna Figurine	Collectables	200.00	200.00	200.00	200.00
12	Kleenex used by Dr. Dre	Collectables	500.00	500.00	500.00	500.00
13	Mike Fudge BobbleHead	Collectables	49.95	49.95	49.95	49.95
14	Original Coke Bottle from 19…	Antiques	65.00	65.00	65.00	65.00
15	PacMan Fever lunchbox	Collectables	29.99	29.99	29.99	29.99

3. Write a query to include the names, counts (number of ratings) and average seller ratings (as a decimal) of users. For reference, User Carrie Dababbbi has 4 seller ratings and an average rating of

4.75.

```
91
92  --ID:hli248
93  --Q3:
94  select s.user_firstname +' '+ s.user_lastname as name,count(*) as number_of_rating,avg(cast(rating_value as decimal(10,3))) as avg_rating
95      from vb_user_ratings r
96      join vb_users s on r.rating_for_user_id=s.user_id
97      where r.rating_astype='Seller'
98      group by s.user_firstname,s. user_lastname
99
100
101
```

Results Messages

	name	number_of_rating	avg_rating
1	Rose Abov-Duresst	3	1.000000
2	Ty Anott	2	2.500000
3	Barb Barion	2	3.500000
4	Carrie Dababbi	4	4.750000
5	Martin Eyezing	2	2.500000
6	Les Ismoore	2	2.500000
7	Anita Job	1	3.000000
8	Abby Kuss	3	4.333333
9	Victor Rhee	1	4.000000

4. Create a list of “Collectable” item types with more than 1 bid. Include the name of the item and the number of bids making sure the item with the most bids appear first.

```
100
101  --ID:hli248
102  --Q4:
103  with c_item_bid as(
104  select i.item_name, count(*) as sum_of_bids
105      from vb_items i
106          join vb_bids b on i.item_id=b.bid_item_id
107          where i.item_type='Collectables'
108          group by i.item_name
109  )
110
111  select * from c_item_bid
112  where sum_of_bids>1
113  order by sum_of_bids desc
114
```

Results Messages

	item_name	sum_of_bids
1	Dukes Of Hazard ashtray	9
2	Autographed Mik Jagger Poster	6
3	Shatner's old Toupee	5
4	Rare Mint Snow Globe	3
5	Farrah Fawcet poster	3
6	Pez dispensers	2

5. 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. Here's an example showing the first 3 bids on item 11.

item_id	item_name	bid_order	bid_amount	bidder
11	Dukes Of Hazard ashtray	1	150.0000	Dan Delyons
11	Dukes Of Hazard ashtray	2	175.0000	Al Fresco
11	Dukes Of Hazard ashtray	3	200.0000	Carrie Dababbi

```
116
117   --ID:h1i248
118   --Q5:
119   select i.item_name,i.item_id,  RANK() OVER (ORDER BY b.bid_amount) bid_order , b.bid_amount, s.user_firstname+' '+ s.user_lastname as bidder
120       from vb_bids b
121       join vb_users s on b.bid_user_id=s.user_id
122       join vb_items i on i.item_id=b.bid_item_id
123           where b.bid_status='ok' and item_id='11'
124
```

Results Messages

	item_name	item_id	bid_order	bid_amount	bidder
1	Dukes Of Hazard ashtray	11	1	150.00	Dan Delyons
2	Dukes Of Hazard ashtray	11	2	175.00	Al Fresco
3	Dukes Of Hazard ashtray	11	3	200.00	Carrie Dababbi
4	Dukes Of Hazard ashtray	11	4	225.00	Gus Toffwind
5	Dukes Of Hazard ashtray	11	5	250.00	Isabelle Gunnering
6	Dukes Of Hazard ashtray	11	6	275.00	Dan Delyons
7	Dukes Of Hazard ashtray	11	7	300.00	Carrie Dababbi
8	Dukes Of Hazard ashtray	11	8	325.00	Isabelle Gunnering

6. Re-Write your query in the previous question to include the names of the next and previous bidders, like this example again showing the first 3 bids for item 11.

item_name	bid_order	bid_amount	prev_bidder	bidder	next_bidder
Dukes Of Hazard ashtray	1	150.0000	*NULL*	Dan Delyons	Al Fresco
Dukes Of Hazard ashtray	2	175.0000	Dan Delyons	Al Fresco	Carrie Dababbi
Dukes Of Hazard ashtray	3	200.0000	Al Fresco	Carrie Dababbi	Gus Toffwind

```
--ID:hli248
--Q6:
select i.item_name,i.item_id,  RANK() OVER (ORDER BY b.bid_amount) bid_order , b.bid_amount,
lag(s.user_firstname+' '+ s.user_lastname) over (ORDER BY b.bid_amount) prev_bidder,
s.user_firstname+' '+ s.user_lastname as bidder,
lead(s.user_firstname+' '+ s.user_lastname) over (ORDER BY b.bid_amount) prev_bidder
   from vb_bids b
   join vb_users s on b.bid_user_id=s.user_id
   join vb_items i on i.item_id=b.bid_item_id
       where b.bid_status='ok' and item_id='11'
```

ults Messages

item_name	item_id	bid_order	bid_amount	prev_bidder	bidder	prev_bidder
Dukes Of Hazard ashtray	11	1	150.00	*NULL*	Dan Delyons	Al Fresco
Dukes Of Hazard ashtray	11	2	175.00	Dan Delyons	Al Fresco	Carrie Dababbi
Dukes Of Hazard ashtray	11	3	200.00	Al Fresco	Carrie Dababbi	Gus Toffwind
Dukes Of Hazard ashtray	11	4	225.00	Carrie Dababbi	Gus Toffwind	Isabelle Gunnering
Dukes Of Hazard ashtray	11	5	250.00	Gus Toffwind	Isabelle Gunnering	Dan Delyons
Dukes Of Hazard ashtray	11	6	275.00	Isabelle Gunnering	Dan Delyons	Carrie Dababbi
Dukes Of Hazard ashtray	11	7	300.00	Dan Delyons	Carrie Dababbi	Isabelle Gunnering
Dukes Of Hazard ashtray	11	8	325.00	Carrie Dababbi	Isabelle Gunnering	*NULL*

7. 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 only include those users who have submitted more than 1 rating.

```
--ID:hli248
--Q7:
with test_rating as(
    select s.user_firstname +' '+ s.user_lastname as name,s.user_email,avg(cast(rating_value as decimal(10,3))) over () as avg_rating, r.rating_value,
    count(r.rating_by_user_id) as num_of_rating
        from vb_user_ratings r
        join vb_users s on r.rating_for_user_id=s.user_id
        where r.rating_by_user_id is not null
        group by s.user_firstname,s. user_lastname,s.user_email,r.rating_value
)
select * FROM test_rating
    where rating_value<avg_rating and num_of_rating>1
```

Results Messages

	name	user_email	avg_rating	rating_value	num_of_rating
1	Les Ismoore	lismoore@mail.org	3.130434	1	4
2	Gus Toffwind	gtoffwind@mail.org	3.130434	3	2
3	Les Ismoore	lismoore@mail.org	3.130434	3	3

8. 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

```
155  --ID:hli248
156  --Q8:
157  select s.user_firstname+' '+s.user_lastname as name,s.user_email ,
158      count(*) as total_bid, count(distinct bid_item_id) as total_item,
159      cast(count(*) as decimal(10,3))/cast(count(distinct bid_item_id) as decimal (10,3)) as KPI
160      from vb_items i
161          join vb_bids b on i.item_id=b.bid_item_id
162          join vb_users s on b.bid_user_id=s.user_id
163          where b.bid_status='ok'
164      group by user_email,user_firstname,user_lastname
165
```

Results Messages

	name	user_email	total_bid	total_item	KPI
1	Abby Kuss	abuss@mail.org	3	1	3.00000000000000
2	Anne Dewey	adwewy@mail.org	5	3	1.66666666666666
3	Barb Barion	bbarion@mail.org	3	2	1.50000000000000
4	Barry DeHatchett	bdehatchett@mail.org	5	1	5.00000000000000
5	Bo Enarreau	benarreau@mail.org	2	2	1.00000000000000
6	Gus Toffwind	gtoffwind@mail.org	2	2	1.00000000000000
7	Isabelle Gunner…	igunner@mail.org	7	2	3.50000000000000
8	Les Ismoore	lismoore@mail.org	3	3	1.00000000000000
9	Martin Eyezing	meveyzing@mail.org	1	1	1.00000000000000
10	Rose Abov-Dures…	rabovdu@mail.org	2	2	1.00000000000000
11	Ray Ovlight	rovlight@mail.org	6	3	2.00000000000000
12	Victor Rhee	vrhee@mail.org	2	2	1.00000000000000
13	Seymour Ofewe	sofewe@mail.org	2	1	2.00000000000000
14	Pete Moss	pmoss@mail.org	2	2	1.00000000000000

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

```sql
--ID:hli248
--Q9:
select distinct FIRST_VALUE(s.user_firstname+' '+ s.user_lastname) over (partition by item_name order by bid_amount desc) as name,
FIRST_VALUE(bid_amount) over (partition by item_name order by bid_amount desc) as highest_bid,
i.item_name
    from vb_bids b
    join vb_users s on b.bid_user_id=s.user_id
    join vb_items i on i.item_id=b.bid_item_id
        where b.bid_status='ok' and item_sold=0
```

Results Messages

	name	highest_bid	item_name
1	Al Fresco	11.00	Client/Server Survival Guide
2	Al Fresco	11.00	SQL for Dummies
3	Dan Delyons	202.00	Shatner's old Toupee
4	Gus Toffwind	70.00	Original Coke Bottle from 19…
5	Isabelle Gunnering	22.00	Ten Speed Bike
6	Isabelle Gunnering	325.00	Dukes Of Hazard ashtray
7	Jean Poole	106.00	Dusty Vase
8	Les Ismoore	5.01	Alf Alarm Clock
9	Les Ismoore	250.00	Some Beanie Babies, New with…
10	Martin Eyezing	205.00	Joe Montanna Figurine
11	Otto Moni	601.00	Old Diamond Ring
12	Ray Ovlight	255.00	Antique Desk
13	Ray Ovlight	515.00	Farrah Fawcet poster
14	Ray Ovlight	1000.00	Kleenex used by Dr. Dre
15	Seymour Ofewe	14.50	Slightly-damaged Golf Bag

10. 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.

```sql
--ID:hli248
--Q10:
with test as(
select distinct s.user_firstname +' '+ s.user_lastname as name,count(rating_for_user_id) over(partition by rating_for_user_id) as number_of_rating,
avg(cast(rating_value as decimal(10,3))) over(partition by rating_for_user_id)as global_avg_rating,
avg(cast(rating_value as decimal(10,3))) over(partition by rating_astype)as avg_rating
    from vb_user_ratings r
    join vb_users s on r.rating_for_user_id=s.user_id
    where r.rating_astype='Seller'
)
select name, number_of_rating, global_avg_rating, avg_rating, (avg_rating -global_avg_rating) as diff from test
```

Results Messages

	name	number_of_rating	global_avg_rating	avg_rating	diff
	Abby Kuss	3	4.333333	3.200000	-1.133333
	Anita Job	1	3.000000	3.200000	0.200000
	Barb Barion	2	3.500000	3.200000	-0.300000
	Carrie Dababbi	4	4.750000	3.200000	-1.550000
	Les Ismoore	2	2.500000	3.200000	0.700000
	Martin Eyezing	2	2.500000	3.200000	0.700000
	Rose Abov-Duresst	3	1.000000	3.200000	2.200000
	Ty Anott	2	2.500000	3.200000	0.700000
	Victor Rhee	1	4.000000	3.200000	-0.800000

Reflection

Use this section to reflect on your learning. To achieve the highest grade on the assignment you must be as descriptive and personal as possible with your reflection.

1. What are the key things you learned through the process of completing this assignment?

2. What were the challenges or roadblocks (if any) you encountered on the way to completing it?

3. Were you prepared for this assignment? What can you do to be better prepared?

4. Now that you have completed the assignment rate your comfort level with this week's material. This should be an honest assessment: (choose one)

   4 ==> I understand this material and can explain it to others.
   3 ==> I understand this material.
   2 ==> I somewhat understand the material but sometimes need guidance from others.
   1 ==> I understand very little of this material and need extra help.