

Unit 04 Problem Set Submission Form

Overview

Your Name	Jainish Savaliya
Your SU Email	jsavaliy@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 games and justify your reasoning. For each, include the SQL as a screenshot with the output of the query.

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

The screenshot shows the SQL Server Enterprise Manager interface. The query editor contains the following SQL statement:

```
select * from vb_users where user_zip_code like '133%'
```

The Results pane displays the following data:

user_id	user_email	user_firstname	user_lastname	user_zip_code
9	weveyzing@mail.org	Martin	Eyezing	13244
12	rovlight@mail.org	Ray	Ovlight	13607
14	tanott@mail.org	Ty	Anott	13244
18	ppincher@mail.org	Penny	Pincher	13212
19	ostuff@mail.org	Oliver	Stuffission	13219
20	herchief@mail.org	Hank	Erchief	13290

- Find all the users from the state of New York. print their names and emails along with their city, state and zip code. Sort by city, then user's last /first name.

The screenshot shows the SQL Server Enterprise Manager interface. The query editor contains the following SQL statement:

```
select * from vb_users where user_zip_code like '133%'
SELECT u.user_lastname + ' ' + u.user_firstname as name, u.user_email,
uz.zip_city, uz.zip_state, uz.zip_code
from vb_users u
join vb_zip_codes as uz on uz.zip_code = u.user_zip_code
and uz.zip_state = 'NY'
order by uz.zip_city ASC , name ASC
```

The Results pane displays the following data:

name	user_email	zip_city	zip_state	zip_code
OvlightRay	rovlight@mail.org	ALEXANDRIA BAY	NY	13607
MossPete	pmoss@mail.org	ALFRED	NY	14802
DababbiCarrie	cdababbi@mail.org	BUFFALO	NY	14264
RheeVictor	vrhee@mail.org	LAKE PLACID	NY	12946
Abov-DuresstRose	rabovdu@mail.org	NEW YORK	NY	10027
MoniOtto	omoni@mail.org	NEW YORK	NY	10017
OfeueSeymour	sofeue@mail.org	NEW YORK	NY	10006
AnottTy	tanott@mail.org	SYRACUSE	NY	13244
ErchiefHank	herchief@mail.org	SYRACUSE	NY	13290
EyezingMartin	weveyzing@mail.org	SYRACUSE	NY	13244
PincherPenny	ppincher@mail.org	SYRACUSE	NY	13212
StuffissionOliver	ostuff@mail.org	SYRACUSE	NY	13219

- High Priced Items. Return the id, name, type, and reserve of items which have not been sold and have a reserve of 250 or higher. Sort the output so that the largest reserve items are first.

The screenshot shows the SQL Server Enterprise Manager interface. The query editor displays the following SQL code:

```

1 select * from vb_users where user_zip_code like '13%'
2 SELECT u.user_lastname + ' ' + u.user_firstname as name, u.user_email,
3 u.zip_city, u.zip_state, u.zip_code
4 from vb_users u
5 join vb_zip_codes as uz on uz.zip_code = u.user_zip_code
6 and uz.zip_state = 'NY'
7 order by u.zip_city ASC , name ASC
8 select item_id, item_name, item_type, item_reserve
9 from vb_items
10 where item_sold = 0 and item_reserve >= 250
11 order by item_reserve DESC
12 GO

```

The Results pane shows the following data:

item_id	item_name	item_type	item_reserve
27	Ark of the Covenant	All Other	1000000.00
28	Superbowl XLIV tickets	Tickets	750.00
24	Old Diamond Ring	Jewelry	599.99
14	Kleenex used by Dr. Dre	Collectables	500.00
18	Antique Desk	Antiques	250.00

- 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.

The screenshot shows the SQL Server Enterprise Manager interface. The query editor displays the following SQL code:

```

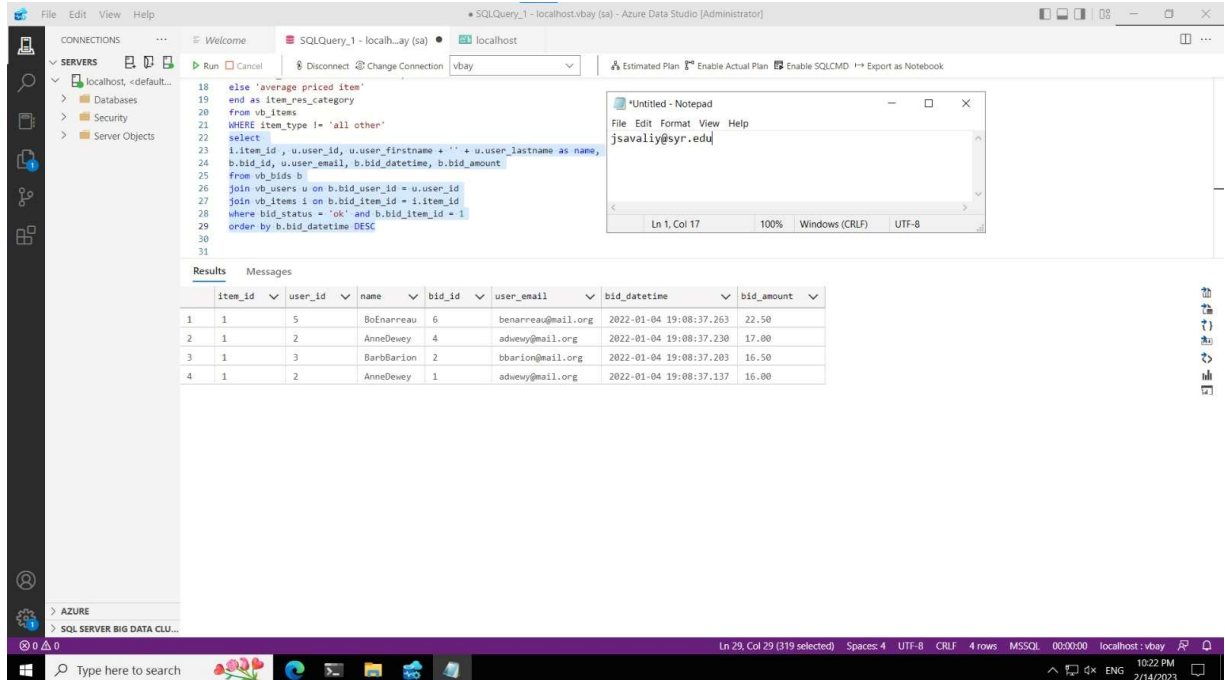
11 order by item_reserve DESC
12 GO
13
14 select item_id, item_name, item_type, item_reserve,
15 case
16 when item_reserve >= 250 then 'high priced item'
17 when item_reserve <= 50 then 'low priced item'
18 else 'average priced item'
19 end as item_res_category
20 from vb_items
21 WHERE item_type != 'all other'
22
23

```

The Results pane shows the following data:

item_id	item_name	item_type	item_reserve	item_res_category
2	Rare Mint Snow Globe	Collectables	30.50	low priced item
3	Smurf TV Tray	Collectables	25.00	low priced item
5	Alf Alarm Clock	Collectables	5.00	low priced item
6	Shatner's old Toupee	Collectables	199.99	average priced item
7	Slightly-damaged Golf Bag	Sporting Goods	12.75	low priced item
8	Some Beanie Babies, New with...	Collectables	99.99	average priced item
10	Your Watch, Please?	Jewelry	6.95	low priced item
11	Dukes Of Hazard ashtray	Collectables	149.99	average priced item
12	PacMan Fever lunchbox	Collectables	29.99	low priced item
13	case of vintage tube socks	Antiques	9.00	low priced item
14	Kleenex used by Dr. Dre	Collectables	500.00	high priced item
15	Farrah Fawcett poster	Collectables	50.00	low priced item
16	Pez dispensers	Collectables	10.00	low priced item
17	a Toaster	Antiques	20.00	low priced item
18	Antique Desk	Antiques	250.00	high priced item
19	SQL for Dummies	Books	10.99	low priced item
20	Mike Fudge Bobblehead	Collectables	49.95	low priced item

5. Bidder list. Write a query which 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.



The screenshot shows the SQL Server Enterprise Manager interface. The query editor displays the following SQL code:

```

18 else 'average priced item'
19 end as item_res_category
20 from vb_items
21 WHERE item_type != 'all other'
22 select
23 i.item_id, u.user_id, u.user_firstname + ' ' + u.user_lastname as name,
24 b.bid_id, u.user_email, b.bid_datetime, b.bid_amount
25 from vb_bids b
26 join vb_users u on b.bid_user_id = u.user_id
27 join vb_items i on b.bid_item_id = i.item_id
28 where bid_status = 'ok' and b.bid_item_id = 1
29 order by b.bid_datetime DESC
30
31

```

The Results pane shows the following data:

	item_id	user_id	name	bid_id	user_email	bid_datetime	bid_amount
1	1	5	BoEnarreau	6	benarreau@mail.org	2022-01-04 19:08:37.263	22.50
2	1	2	AnneDewey	4	adewey@mail.org	2022-01-04 19:08:37.230	17.00
3	1	3	BarbBarion	2	bbarion@mail.org	2022-01-04 19:08:37.203	16.50
4	1	2	AnneDewey	1	adewey@mail.org	2022-01-04 19:08:37.137	16.00

6. The bad bidder list. Write 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.

SQLQuery_1 - localhost.vbay (sa) - Azure Data Studio [Administrator]

```

31 --6
32 select i.item_id, i.item_name, u.user_id, u.user_lastname + '
33 + u.user_firstname as name, u.user_email, b.bid_datetime, b.bid_id,
34 b.bid_amount, b.bid_status
35
36 from vb_bids b
37 join vb_users u on b.bid_user_id = u.user_id
38 join vb_items i on b.bid_item_id = i.item_id
39 where bid_status != 'ok'
40 order by name asc,
41 b.bid_datetime DESC
42
43 --7

```

Results

item_id	item_name	user_id	name	user_email	bid_datetime	bid_id	bid_amount	bid_status
11	Dukes Of Hazard ashtray	11	Abov-DuresstRose	rabovdu@mail.org	2022-01-04 19:08:37.460	22	100.00	low_bid
18	Antique Desk	3	BarionBarb	bbbarion@mail.org	2022-01-04 19:08:37.650	39	251.00	low_bid
1	Used Pink Bathrobe	2	DeweyAnne	adewey@mail.org	2022-01-04 19:08:37.213	3	16.50	low_bid
36	Autographed Mik Jagger Poster	1	KussAbby	abuss@mail.org	2022-01-04 19:08:38.040	74	95.00	low_bid
1	Used Pink Bathrobe	1	KussAbby	abuss@mail.org	2022-01-04 19:08:37.253	5	20.00	item_seller
2	Rare Mint Snow Globe	10	MelatorMary	mmelator@mail.org	2022-01-04 19:08:37.273	7	30.00	low_bid
6	Shatner's old Toupee	17	MoniOtto	omon@mail.org	2022-01-04 19:08:37.360	15	500.00	item_seller
6	Shatner's old Toupee	17	MoniOtto	omon@mail.org	2022-01-04 19:08:37.343	13	500.00	item_seller
16	Pez dispensers	16	MossPete	pmoss@mail.org	2022-01-04 19:08:37.610	36	10.00	low_bid

Ln 37, Col 45 Spaces: 4 UTF-8 CRLF 9 rows MSSQL 00:00:00 localhost: vbay 10:47 PM 2/14/2023

7. Produce a report of items which do not contain a bid. Include the item id, item name, item type, seller name item reserve.

SQLQuery_1 - localhost.vbay (sa) - Azure Data Studio [Administrator]

```

41 b.bid_datetime DESC
42
43 --7
44 select i.item_id, i.item_name,
45 i.item_type, u.user_firstname + ' ' + u.user_lastname as name,
46 i.item_reserve
47
48 from vb_items i
49 join vb_users u on i.item_seller_user_id = u.user_id
50 where i.item_id not in (select bid_item_id from vb_bids)
51
52 --8
53 select u.rating_by_user_id, u.user_firstname + '
54 + u.user_lastname as user_who_gave_rating, u.rating_for_user_id,
55 u.user_firstname + ' ' + u.user_lastname as user_the_rating_for

```

Results

item_id	item_name	item_type	name	item_reserve
4	Pet Rock	All Other	BarryDeHatchett	2.50
9	Tchotchkes	All Other	MaryMelator	0.99
10	Your Watch, Please?	Jewelry	RoseAbov-Duresst	6.95
12	PacMan Fever lunchbox	Collectables	CarrieDababbi	29.99
17	a Toaster	Antiques	OttoMoni	20.00
20	Mike Fudge Bobblehead	Collectables	MartinEyzeying	49.95
21	Carlos Villalba Bobblehead	Collectables	MartinEyzeying	49.95
25	Betamax Player	Electronics	SeymourOfewe	15.00
27	Ark of the Covenant	All Other	AbbyKuss	10000000.00
28	Superbowl XLIV tickets	Tickets	BarbBarion	750.00
30	Avatar 3D Two Tickets	Tickets	CarrieDababbi	5.00
35	Brass French Press	Antiques	CarrieDababbi	45.50

Ln 50, Col 58 Spaces: 4 UTF-8 CRLF 12 rows MSSQL 00:00:00 localhost: vbay 10:48 PM 2/14/2023

8. 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 rating comment. Include only ratings of sellers.

SQLQuery_1 - localhost.vbay (sa) - Azure Data Studio [Administrator]

```

51 --8
52 select ur.rating_by_user_id, u.user_firstname + '
53 + u.user_lastname as user_who_gave_rating, ur.rating_for_user_id,
54 ul.user_firstname + ' + ul.user_lastname as user_who_rating_for ,
55 ur.rating_value , ur.rating_comment , ur.rating_astype
56
57
58 from vb_user_ratings ur
59 join vb_users u on ur.rating_by_user_id= u.user_id
60 join vb_users ul on ur.rating_for_user_id = ul.user_id
61 where ur.rating_astype = 'seller'
62
63 --9
64 SELECT i.item_id, i.item_name, i.item_type, i.item_soldamount,

```

Results

	rating_by_user_id	user_who_gave_rating	rating_for_user_id	user_who_rating_for	rating_value	rating_comment	rating_astype
1	1	AbbyKuss	25	CarrieDababbi	5	Top Notch!	Seller
2	2	AnneDewey	25	CarrieDababbi	4	Good	Seller
3	3	BarbBarion	22	AnitaJob	3	Okay	Seller
4	4	BarryDeHatchett	1	AbbyKuss	4	Reliable	Seller
5	5	BoEnarreau	25	CarrieDababbi	5	Excellent!	Seller
6	6	GusTofvind	9	Martiney	3	Not Bad	Seller
7	25	CarrieDababbi	11	RoseAbou-Duresst	0	Horrible.	Seller
8	8	LesIsmoore	3	BarbBarion	5	Nice!	Seller
9	9	Martineyeyezing	1	AbbyKuss	5	The Best!	Seller
10	10	MaryMelator	9	Martineyeyezing	2	So-So	Seller
11	11	AbbyKuss	11	RoseAbou-Duresst	2	Needs Improvement	Seller
12	12	RayOvlight	3	BarbBarion	2	Bad Experience.	Seller
13	2	AnneDewey	11	RoseAbou-Duresst	1	Bad.	Seller
14	7	IsabelleGunninger	14	Tyanott	1	Poor	Seller
15	25	CarrieDababbi	1	AbbyKuss	4	Great!	Seller
16	16	PeteMoss	25	CarrieDababbi	5	Super Great!	Seller
17	17	OttoMoni	14	Tyanott	4	Great!	Seller

Ln 59, Col 51 Spaces: 4 UTF-8 CRLF 20 rows MSSQL 00:00:00 localhost.vbay 2/14/2023

9. For items that were sold, generate a report which 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

SQLQuery_1 - localhost.vbay (sa) - Azure Data Studio [Administrator]

```

64 SELECT i.item_id, i.item_name, i.item_type, i.item_soldamount,
65 u_buyer.user_firstname + ' + u_buyer.user_lastname as buyer_name,
66 zc_buyer.zip_code + ' + zc_buyer.zip_state as buyer_city_state,
67 u_seller.user_firstname + ' + u_seller.zip_state as seller_city_state,
68 i.item_sold
69
70 from vb_items i
71 join vb_users u_buyer on item_buyer_user_id=u_buyer.user_id
72 join vb_users u_seller on item_seller_user_id = u_seller.user_id
73 join vb_zip_codes zc_buyer on u_buyer.user_zip_code = zc_buyer.zip_code
74 join vb_zip_codes zc_seller on u_seller.user_zip_code = zc_seller.zip_code
75
76 --10

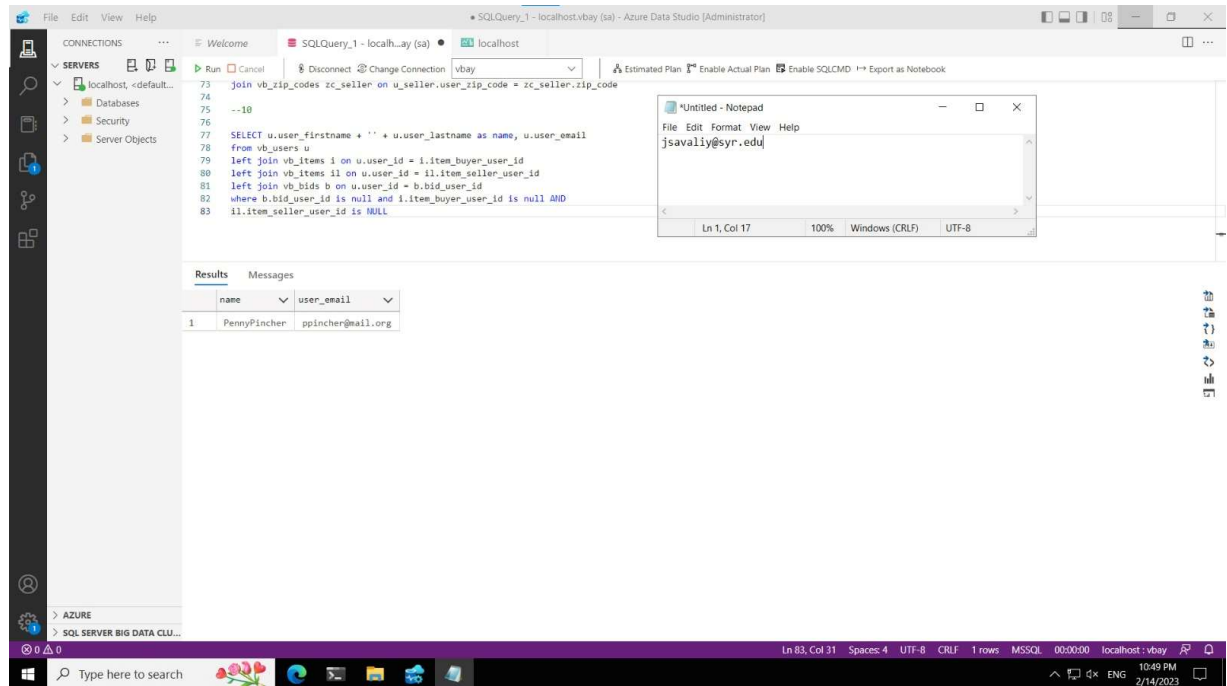
```

Results

	item_id	item_name	item_type	item_soldamount	buyer_name	buyer_city_state	seller_city_state	item_sold
1	1	Used Pink Bathrobe	All Other	22.50	BoEnarreau	ZEBULON,GA	AbbyCA	1
2	2	Rare Mint Snow Globe	Collectables	40.00	RoseAbou-Duresst	NEW YORK,NY	AnneAL	1
3	3	Smurf TV Tray	Collectables	26.00	LesIsmoore	CHATTANOOGA,TN	BarbAZ	1
4	16	Pez dispensers	Collectables	11.00	OttoMoni	NEW YORK,NY	VictorNY	1
5	22	Fuzzy Logic	Books	5.00	PeteMoss	ALFRED,NY	SeymourNY	1
6	36	Autographed Mik Jagger Poster	Collectables	100.00	AbbyKuss	SAN FRANCISCO,CA	CarrieNY	1

Ln 73, Col 75 Spaces: 4 UTF-8 CRLF 6 rows MSSQL 00:00:00 localhost.vbay 2/14/2023

10. 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.



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?
- **I gained knowledge of connecting tables by utilizing database schemas.**
2. What were the challenges or roadblocks (if any) you encountered on the way to completing it?
- **Requires much practice and revision of the principles.**
3. Were you prepared for this assignment? What can you do to be better prepared?
- **I have studied the joins' fundamentals; perhaps additional practice is needed.**
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.