DSO 552: SQL Databases for Business Analysts

Final Exam (Exam 1)

You get 1 point for free. There are 10 questions, and you can choose to submit any 9 questions you'd like. We will not provide extra credit. If you do all 10 questions, we will pick the first 9 to grade.

Note: Make sure to submit your answers by the deadline on Blackboard. Late submissions will not be accepted.

Also make sure

- your column names match the sample output
- you do not `LIMIT 5` unless the question directly asks you to do so

Yelp Database

1. ***(1 point)*** **Which users tend to have the highest average rating per reviews? List the top 5 users in terms of star ratings on reviews. List only users with more than 2 total reviews.**

```sql
SELECT u.user_id, AVG(r.stars) AS average_star_rating
FROM users u
JOIN reviews r ON r.user_id = u.user_id
GROUP BY 1
HAVING COUNT(*) > 2
ORDER BY 2 DESC
```

Expected:

user_id	average_star_rating
5hfrKyck8DBN96EN9VczUw	5.000000
WLde9fTArPw_L6KkuZ60-w	5.000000
lMTmBNddR47i35jgOnIdIA	5.000000
I-4KVZ9lqHhk8469X9FvhA	5.000000
-LG_49tfkstPBzXB-C6R2g	5.000000
CKRzIkY1N0eFCDkShZi67A	5.000000
ZRnsUs9q6DbeJbOLjCY4ZA	4.333333
P9tsVSYF79fwRvMEjtTvWg	4.000000
Ryxj0u0AW3mRsRypdYli2A	4.000000
BRGnIsVAClYJ8nNZWSZPMg	3.666667
p-2tjwAQ_8pB5r552gQNFQ	3.666667
Jtj_c2qMRDMgno8tLSlCuQ	3.000000
wGCIzYUugIYOds35_Qzagw	2.666667
YVAFkLWAi0ZQewBHIfCMlw	2.500000
cgO4Bg_O4-a_ydEavF7Mhg	2.000000
2hMKTjdvDJXoAbXEmEWsMg	1.000000

2. ***(1 point)*** **Which category of business tend to have the highest ratings? Calculate the weighted average number of stars for a category - the sum of the review count x stars di-**

vided by the sum of review count. Show only categories that have more than 3 businesses. Sort by the weighted average number of stars.

```
WITH category_ratings AS (SELECT b.business_id, stars, review_count, unnest(string_to_array(categories,
FROM businesses b)

SELECT category,
SUM(stars * review_count) / SUM(review_count) AS weighted_average_stars,
COUNT(DISTINCT business_id) AS num_businesses
FROM category_ratings
GROUP BY 1
HAVING COUNT(DISTINCT business_id) > 3
ORDER BY 2 DESC
```

Expected:

category	weighted_average_stars	num_businesses
Health & Medical	4.278571	5
Local Services	4.090000	6
Shopping	3.998000	10
Professional Services	3.938272	4
Restaurants	3.625767	19
Sandwiches	3.572581	4
Event Planning & Services	3.504287	4
Food	3.311475	5

3. ***(1 point)*** **List all users that left both a tip and a review for a particular business. List the business' name, and the number of tips and reviews left for that particular business.**

```
SELECT DISTINCT u.name AS username, b.name AS businessname,
COUNT(DISTINCT t.business_id || t.date) AS tips_left,
COUNT(DISTINCT r.business_id ||  r.date) AS reviews_left
FROM users u
JOIN tips t ON t.user_id = u.user_id
JOIN reviews r ON r.user_id = u.user_id AND r.business_id = t.business_id
JOIN businesses b ON b.business_id = r.business_id
GROUP BY 1,2
ORDER BY 2
LIMIT 10;
```

Expected:

username	businessname	tips_left	reviews_left
Alicia	99 Cents Only	3	1
Fay	99 Cents Only	2	1
Matt	99 Cents Only	1	1
Mike	Bark 'N' Scratch	1	1
Mary	Burlington Coat Factory	1	1
Brent	Cholla Prime Steakhouse & Lounge	1	1
Comfy	Cholla Prime Steakhouse & Lounge	16	2
Jacqueline	Cholla Prime Steakhouse & Lounge	1	1
Jae And Nui	Cholla Prime Steakhouse & Lounge	1	1
Jolly	Cholla Prime Steakhouse & Lounge	1	1

4. ***(1 point)*** **List all users who have at least 5 fans that have left a review for `Burlington`**

Coat Factory (a business).

```
SELECT u.user_id, fans, u.name AS username, r.text, r.date
FROM users u
JOIN reviews r ON r.user_id = u.user_id
JOIN businesses b ON b.business_id = r.business_id
WHERE b.name = 'Burlington Coat Factory'
AND fans > 5;
```

Expected:

user_id	fans	username	text
hWDybu_KvYLSdEFzGrniTw	659	Bruce	This location has been turned over so many times and vacant for so
e9x9H_wGR430eCqvTgDgPw	41	Eren	Burlington Coat factory is easily one of my favorite places to shop b
l9i6eupjzPDyfVJoUrHyOw	7	Stefanie	This new location is just wonderful. It's a lot cleaner than the other
9eFi-CKKmdlwpoHVja9Y6g	21	Cameron	Burlington Coat factory is hit or miss. I've been here a handful of t
nyl_1VcRIAyI55bb_scpdw	61	Mary	I love Burlington Coat Factory. Years ago, when I lived in Rocheste

5. ***(1 point)*** **Do users who have left at least one tip tend to have higher average star ratings than users who have not left any tips?**

```
WITH users_without_tips AS (
SELECT user_id FROM users WHERE user_id NOT IN (SELECT user_id FROM tips)),

users_with_tip AS (
SELECT user_id FROM users WHERE user_id IN (SELECT user_id FROM tips)
)

SELECT 'Users without tips' AS type_of_user, AVG(stars) AS avg_stars
FROM reviews
WHERE user_id IN (SELECT * FROM users_without_tips)
UNION
SELECT 'Users with tips', AVG(stars) AS avg_stars
FROM reviews
WHERE user_id IN (SELECT * FROM users_with_tip)
```

Expected:

type_of_user	avg_stars
Users without tips	3.566143
Users with tips	3.955307

6. ***(1 point)*** **Perform an analysis that shows the average star ratings of businesses that 1) accept credit cards or 2) offers take-out (`RestaurantsTakeOut` in `attributes`). If the attribute is not available for the business, default it to `False`. Sort by the average number of stars.**

```
WITH parsed_businesses AS (
    SELECT b.*,
           COALESCE(attributes:: JSON ->> 'BusinessAcceptsCreditCards', 'False') AS accepts_credit_cards
           COALESCE(attributes:: JSON ->> 'RestaurantsTakeOut', 'False')         AS offers_takeout
    FROM businesses b)

SELECT accepts_credit_cards, offers_takeout, AVG(stars) AS average_stars
FROM parsed_businesses p
```

```
GROUP BY 1, 2
ORDER BY 3 DESC
```

Expected:

accepts_credit_cards	offers_takeout	average_stars
True	False	3.625000
False	True	3.600000
False	False	3.437500
True	True	3.192308

Entertainment Agency Database

7. ***(1 point)*** **The agency has a theory that the first five engagements that entertainers book tend to be lower contract value than the later engagements. Produce an analysis that shows the average contract price of the first five engagements for an entertainer versus the average contract price of the 6th engagement and beyond.**

```
SELECT engagement_category, AVG(contractprice)
FROM (
         SELECT e.*,
                CASE
                    WHEN ROW_NUMBER() OVER (PARTITION BY e.entertainerid ORDER BY e.startdate, e.engage
                        THEN 'First Five Engagements'
                    ELSE '6th and Beyond Engagements' END
                    AS engagement_category
         FROM engagements e) t1
GROUP BY 1
```

Expected:

engagement_category	avg
6th and Beyond Engagements	1419.118
First Five Engagements	1136.250

8. ***(1 point)*** **For all engagements that have a contract price greater than the global average contract price across all engagements, list the average contract price (for these above average engagements), the most recent start date, and the number of engagements.**

```
SELECT AVG(contractprice)      AS avg_contract_price,
       MAX(startdate)          AS most_recent_start_date,
       COUNT(engagementnumber) AS num_engagements
FROM engagements
WHERE contractprice > (SELECT AVG(contractprice) FROM engagements)
```

Expected:

avg_contract_price	most_recent_start_date	num_engagements
2284.302	2018-03-04	43

9. ***(1 point)*** **Show the counts for each combination of musical style and entertainer member's gender. Only show combinations where there are more than 4 distinct entertainer members. Order by the number of members.**

```
SELECT gender, ms.stylename, COUNT(DISTINCT em.memberid) AS num_members
FROM entertainers e
         JOIN entertainer_members em ON em.entertainerid = e.entertainerid
         JOIN members m ON m.memberid = em.memberid
         JOIN entertainer_styles es ON es.entertainerid = e.entertainerid
         JOIN musical_styles ms ON ms.styleid = es.styleid
GROUP BY 1, 2
HAVING COUNT(DISTINCT em.memberid) > 4
ORDER BY 3 DESC
```

Expected:

gender	stylename	num_members
F	Standards	8
M	60's Music	8
F	Contemporary	5
F	Top 40 Hits	5
M	Classic Rock & Roll	5
M	Country	5

10. ***(1 point)*** **Show how much each customer has paid total to our agency - which is reflected in the `contractprice`. However, if a customer books 3 or more engagements in the same month, we charge only 90% of the contract price for the 3rd engagement that month and beyond (until the start of a new month). Do not include the agent Karen Smith's engagements at all here in this computation.**

```
WITH engagements_per_month AS (
    SELECT *,
           ROW_NUMBER()
           OVER (PARTITION BY customerid, DATE_TRUNC('month', startdate) ORDER BY startdate) AS engageme
    FROM engagements WHERE agentid <> 4),

     adjusted_engagements AS (
         SELECT *, CASE WHEN engagement_number >= 3 THEN .90 ELSE 1 END adjustment
         FROM engagements_per_month)

SELECT customerid, SUM(adjustment * contractprice) total_paid
FROM adjusted_engagements
GROUP BY 1
ORDER BY 2 DESC
```

Expected:

customerid	total_paid
10005	24635
10006	12970
10014	12669
10004	10618
10010	10470
10002	10393
10001	8855
10015	7080
10009	5690
10012	5357
10013	4580
10007	4485
10003	3130