SQL_02 - Functions, Filtering and Subqueries

All

Name of the Problem	Type	Difficulty	Score	Status	Submissions	Asked In	Actions
Q1. Weighted avg rating	SQL	Hard	50.0/50	Solved	2 submissions	-	Solve
Q2. New Salary	SQL	Medium	50.0/50	Solved	1 submissions	-	Solve
Q3. Movies profit	SQL	Medium	50.0/50	Solved	1 submissions	-	Solve
Q4. Olympic Table	SQL	Easy	50.0/50	Solved	1 submissions	-	Solve
Q5. Movies (Not Boring)	SQL	Hard	50.0/50	Solved	1 submissions	-	Solve
Q6. 2012-2015	SQL	Medium	50.0/50	Solved	1 submissions	-	Solve

Q 6	Q7. Low Fat & Recyclable Products	SQL	Medium	50.0/50	Solved	1 submissions	-	Solve
Q 7	Q8. Problems that are low quality	SQL	Medium	50.0/50	Solved	1 submissions	-	Solve
Q 8	Q9. Letter 'n'	SQL	Medium	50.0/50	Solved	1 submissions	-	Solve
Q 9	Q10. Keywords	SQL	Hard	50.0/50	Solved	1 submissions	-	Solve
Q 10	Q11. Concatenate the Name and the Profession	SQL	Easy	50.0/50	Solved	1 submissions	-	Solve
Q 11	Q12. Movie titles & Tagline	SQL	Medium	50.0/50	Solved	1 submissions	-	Solve
Q 12	Q13. Salary(5000-10000)	SQL	Medium	50.0/50	Solved	1 submissions	-	Solve
Q 13								

Q1. Weighted avg rating

Question | Hints | Help Requests | Submissions | Discussions

Q1. Weighted avg rating SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

Calculate the weighted average rating from the columns **vote_count** and **vote_average** and save the column as "**Weighted_avg_rating**".

Display the **top 10** movies and **their rating** up to **two** decimals based on the newly created column.

- Use the given **movies** table.
- Return the columns `original_title`, `Weighted_avg_rating`
- Return the output ordered by `Weighted_avg_rating` in descending order and `original_title` in ascending order.

MySQL (MySQL 8.0)

```sql
/*
Note: Use the given formula to calculate a weighted average rating:
    (v/(v+m) * R) + (m/(m+v) * C )
Where,
    v is the number of votes for the movie - vote_count
    m is the minimum votes required, take m as 104.0
    R is the average rating of the movie - vote_average
    C is the mean vote across the whole report take c as 5.97
*/

SELECT
    original_title,
    ROUND((vote_count/(vote_count + 104.0) * vote_average) +
            (104.0/(104.0 + vote_count) * 5.97), 2)
            AS Weighted_avg_rating
FROM movies
ORDER BY Weighted_avg_rating DESC, original_title ASC
LIMIT 10;
```

Test Output

Test

Dataset description for movies table:

1) id - tmdb movie id
2) imdb_id - imdb movie id
3) popularity -A numeric quantity specifying the movie's popularity.
4) budget -The budget in which the movie was made.
5) revenue - The worldwide revenue generated by the movie.
6) original_title- The title of the movie
7) cast - The name of the lead and supporting actors.
8) homepage - A link to the homepage of the movie.
9) director - The name of the director of the movie
10) tagline - Movie's tagline.
11) keywords -The keywords or tags related to the movie.
12) overview -A brief description of the movie.
13) runtime -The running time of the movie in minutes.
14) genres -The genres of the movies
15) production_companies-The production house of the movie.
16) release_date -the date on which it was released.
17) vote_count -the count of votes received.
18) vote_average - average ratings the movie received.
19) release_year - the year on which it was released

Q2. New Salary

Question | Hints | Help Requests | Submissions | Discussions

Q2. New Salary SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Description:

Write a query to calculate the salary of all employees after an increment of **20%**. Save the newly calculated salary column as '**New_salary**'.

Note:

- Return the columns **emp_id**, **name**, **salary**, and '**New_salary**'.
- Order the output by the **emp_id** in ascending order.

MySQL (MySQL 8.0)

```sql
/*
Steps to calculate the salary increment:
(1) Multiply the current salary by the percentage of the increment.
(2) Divide the result by 100.
(3) Then add the result to the current salary.
(4) Round off the 'New_salary'.
*/

SELECT
    emp_id,
    name,
    salary,
    ROUND(salary + (0.20 * salary), 0) AS New_salary
FROM employees
ORDER BY emp_id;
```

Sample Input:

Table: employees

emp_id	name	salary
1	Luis	6142
2	Den	11259
3	Alexander	5374
4	Shelli	12572
5	Sigal	6897

Sample Output:

emp_id	name	salary	New_salary
1	Luis	6142	7370
2	Den	11259	13511
3	Alexander	5374	6449
4	Shelli	12572	15086
5	Sigal	6897	8276

Q3. Movies profit

Question | Hints | Help Requests | Submissions | Discussions

Q3. Movies profit SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

Calculate the **profit percentage** using the columns '**budget**' and '**revenue**' and save the column as '**Profit_percentage**'.

- Return the columns '**original_title**' and '**Profit_percentage**'
- Display all the movies and their percentage up to two decimals based on the new column created.

MySQL (MySQL 8.0)

```sql
/*
Explanation:
Use the given formula to calculate the profit percentage:
    ((revenue-budget)/budget) * 100
*/

SELECT
    original_title,
    ROUND(((revenue - budget)/budget) * 100, 2) AS Profit_percentage
FROM movies;
```

Q4. Olympic Table

Question | Hints | Help Requests | Submissions | Discussions

Q4. Olympic Table SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

Write a SQL query to sort the `olympic` table according to the following rules:

- The country with more gold medals should come first.
- If there is a tie in the no. of gold medals, the country with more silver medals should come first.
- If there is a tie in the no. of silver medals, the country with more bronze medals should come first.
- If there is a tie in the no. of bronze medals as well, then the countries with the tie are sorted in ascending order lexicographically.

MySQL (MySQL 8.0)

```sql
/*
Explanation:
-- The tie between China and USA is broken by their lexicographical names.
-- Since "China" is lexicographically smaller than "USA", it comes first.
-- Israel comes before Egypt because it has more bronze medals.
*/

SELECT *
FROM olympic
ORDER BY gold_medals DESC,
         silver_medals DESC,
         bronze_medals DESC,
         country ASC;
```

Test Output Code's All Neat!

Test

Sample Input:

Table: `olympic`

country	gold_medals	silver_medals	bronze_medals
China	10	10	20
South Sudan	0	0	1
USA	10	10	20
Israel	2	2	3
Egypt	2	2	2

Sample output:

country	gold_medals	silver_medals	bronze_medals
China	10	10	20
USA	10	10	20
Israel	2	2	3
Egypt	2	2	2
South Sudan	0	0	1

Q5. Movies (Not Boring)

Question | Hints | Help Requests | Submissions | Discussions

Q5. Movies (Not Boring) SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

Write a query to report the movies with an **odd-numbered ID** and a description that is not "**boring**".

- Return the result table ordered by **`rating`** in **descending order.**

Sample Input:

Table: cinema

id	movie	description	rating
1	War	great 3D	8.9
2	Science	fiction	8.5
3	irish	boring	6.2
4	Ice song	Fantacy	8.6
5	House card	Interesting	9.1

MySQL (MySQL 8.0)

```
/*
Explanation:
-- We have three movies with odd-numbered IDs: 1, 3, and 5.
-- The movie with ID = 3 is boring so we do not include it in the answer.
*/

SELECT *
FROM cinema
WHERE (id % 2 != 0) AND (description NOT LIKE "boring")
ORDER BY rating DESC;

# Method 2
# Note: Here mod function can also be used inplace of %
SELECT *
FROM cinema
WHERE MOD(id, 2) = 1
    AND description != "boring"
ORDER BY rating DESC;
```

Test Output

Test

Q6. 2012-2015

Question | Hints | Help Requests | Submissions | Discussions

Q6. 2012-2015 SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

Find the details of the movies that are released **between** the years **2012-2015** i.e., (Including 2012 and 2015).

- Return the columns '**original_title**', '**genres**', '**vote_average**', and '**revenue**'.
- Return the result ordered by **original_title** in ascending order.

MySQL (MySQL 8.0)

```sql
SELECT
    original_title,
    genres,
    vote_average,
    revenue
FROM movies
WHERE release_year BETWEEN 2012 AND 2015
ORDER BY original_title;
```

Q7. Low Fat & Recyclable Products

Question | Hints | Help Requests | Submissions | Discussions

Q7. Low Fat & Recyclable Products SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

Write a query to find the ids of products that are both **low-fat and recyclable**.

- Return the result table ordered by `product_id` in ascending order.

Sample Input:

Table: `products`

product_id	low_fats	recyclable
0	Y	N
1	Y	Y
2	N	Y
3	Y	Y
4	N	N

MySQL (MySQL 8.0)

```sql
/*
Explanation:
-- Only products 1 and 3 are both low-fat and recyclable.
*/

SELECT product_id
FROM products
WHERE (low_fats = "Y" AND recyclable = "Y")
ORDER BY product_id;
```

Test Output Code's All Neat!

Test

Q8. Problems that are low quality

Question | Hints | Help Requests | Submissions | Discussions | Your Score: 50 | Max Score:

Q8. Problems that are low quality SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

Write a query to report the IDs of the **low-quality** problems.

A problem is low-quality if the **like** percentage of the problem (the number of likes divided by the total number of votes) is strictly **less than 60%**.

- Return the result table ordered by `problem_id` in **ascending order**.

MySQL (MySQL 8.0)

```sql
-- Problem 11: (2677 / (2677 + 8659)) * 100 = 23.61503%
-- Problem 13: (2050 / (2050 + 4164)) * 100 = 32.99002%
Problems 7, 10, 11, and 13 are low-quality problems because their like percentages are less than 60%.
*/
SELECT problem_id
FROM problems
WHERE (likes / (likes + dislikes)) < 0.60
ORDER BY problem_id;
```

Sample Input:

Table: problems

problem_id	likes	dislikes
6	1290	425
11	2677	8659
1	4446	2760
7	8569	6086
13	2050	4164
10	9002	7446

Sample output:

problem_id
7
10
11
13

Q9. Letter 'n'

Question | Hints | Help Requests | Submissions | Discussions

Q9. Letter 'n' SQL Solved

Stuck somewhere?
Using hints is now penalty free

Check Now

Problem Description:

Write a query to find all the employees whose **first_name** ends with the letter **'n'.**

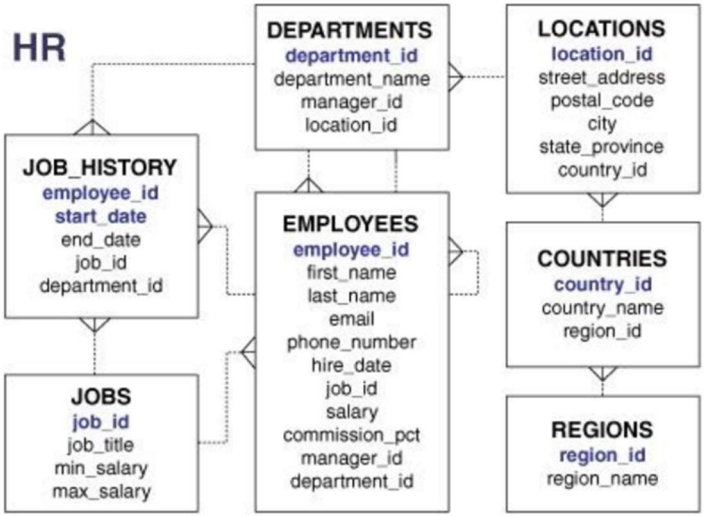
- Return the columns '**employee_id**', '**full_name**' (first name and last name separated by space), and '**phone_number**'.
- Return the output ordered by **employee_id** in ascending order.

MySQL (MySQL 8.0)

```sql
SELECT
    employee_id,
    CONCAT(first_name, " ", last_name) AS full_name,
    phone_number
FROM employees
WHERE first_name LIKE "%n"
ORDER BY employee_id;

# Method 2 - Using substr()
select employee_id,
concat(first_name, " ", last_name) "full_name", phone_number
from employees
where substr(first_name,-1,1) = "n"
order by employee_id;
```

Dataset Description:



Sample Input:

Table: employees

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	commission_pct	manager_id	department_id
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	25000	NULL	NULL	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1989-09-21	AD_VP	17000	NULL	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1993-01-13	AD_VP	17000	NULL	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1990-01-03	IT_PROG	9000	NULL	102	60

Sample Output:

employee_id	full_name	phone_number
100	Steven King	515.123.4567

Q10. Keywords

Question | Hints | Help Requests | Submissions | Discussions

Q10. Keywords SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

List down all the movies along with their details that have **keywords** like 'sport' or 'sequel' or 'suspense'.

Note:

1. Return the columns '**original_title**', '**director**', '**genres**', '**cast**', '**budget**', '**revenue**', '**runtime**', and '**vote_average**'.
2. Return the columns ordered by **original_title** in ascending order.

MySQL (MySQL 8.0)

```sql
SELECT
    original_title,
    director,
    genres,
    cast,
    budget,
    revenue,
    runtime,
    vote_average
FROM movies
WHERE LOWER(keywords) LIKE "%sport%" OR
    LOWER(keywords) LIKE "%sequel%" OR
    LOWER(keywords) LIKE "%suspense%"
ORDER BY original_title;
```

Q11. Concatenate the Name and the Profession

Question | Hints | Help Requests | Submissions | Discussions | Your Score: 50 | Max Score

Q11. Concatenate the Name and the Profession

SQL Solved

Stuck somewhere?
Using hints is now penalty free

Check Now

Problem Statement:

Write a SQL query to display each person's name followed by the first letter of their profession enclosed in parentheses.

Note: Return the result table ordered by `person_id` in descending order.

MySQL (MySQL 8.0)

```sql
/*
Explanation:
Note that there should not be any white space between the name and the first letter of the profession.
*/

SELECT
    person_id,
    CONCAT(NAME, "(", UPPER(SUBSTRING(profession, 1, 1)), ")") AS name
FROM person
ORDER BY person_id DESC;
```

Sample Input:

Table: person

person_id	name	profession
1	Alex	Singer
3	Alice	Actor
2	Bob	Player
4	Messi	Doctor
6	Tyson	Engineer
5	Meir	Lawyer

Sample Output:

person_id	name
6	Tyson(E)
5	Meir(L)
4	Messi(D)
3	Alice(A)
2	Bob(P)
1	Alex(S)

Q12. Movie titles & Tagline

Question | Hints | Help Requests | Submissions | Discussions

Q12. Movie titles & Tagline SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

Write a SQL query to create a new column named "**Title**" in the **movies** table by concatenating the movie **title** with thecorresponding **tagline**, separated by a **hyphen (-)**.

MySQL (MySQL 8.0)

```sql
/*
Sample Explanation:
For example, the Title for a movie named "Vacation"
with the tagline "What could go wrong?"
should be displayed as "Vacation-What could go wrong?".
*/

SELECT
    CONCAT(original_title, "-", tagline) AS Title
FROM movies;
```

Q13. Salary(5000-10000)

Question | Hints | Help Requests | Submissions | Discussions

Q13. Salary(5000-10000) SQL Solved

Stuck somewhere?
Using hints is now penalty free
Check Now

Problem Statement:

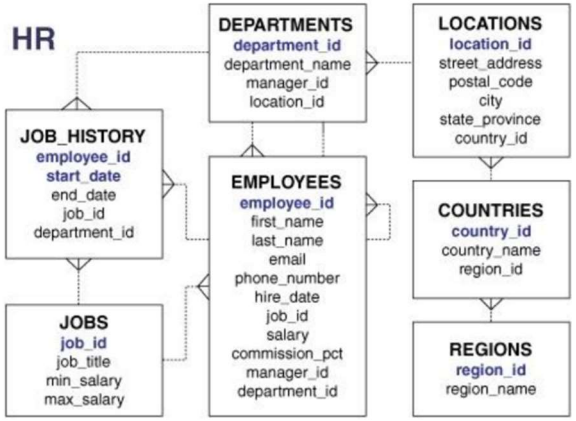
Display the details of the employees who work in departments **50,10,** or **80** and whose salary is between **5000** to **10000** and also where employees have **no commission_pct**(commission percentage).

- Return the result ordered by **employee_id** in ascending order.

MySQL (MySQL 8.0)

```sql
SELECT
    employee_id,
    first_name,
    last_name,
    salary
FROM employees
WHERE (department_id IN (50, 10, 80))
    AND (salary BETWEEN 5000 AND 10000)
    AND (commission_pct IS NULL)
ORDER BY employee_id;
```

Dataset Description:



Sample Input:

Table: employees

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	commission_pct	manager_id	department_id
115	Alexander	Khoo	AKHOO	515.127.4562	1995-05-18	PU_CLERK	3100	NULL	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1997-12-24	PU_CLERK	2900	NULL	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1997-07-24	PU_CLERK	2800	NULL	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1998-11-15	PU_CLERK	2600	NULL	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1999-08-10	PU_CLERK	2500	NULL	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1996-07-18	ST_MAN	8000	NULL	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1997-04-10	ST_MAN	8200	NULL	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1995-05-01	ST_MAN	7900	NULL	100	50

Sample Output:

employee_id	first_name	last_name	salary
120	Matthew	Weiss	8000
121	Adam	Fripp	8200
122	Payam	Kaufling	7900