Problem solving on Subqueries, Case Statements and CTE

Relevel
by Unacademy

Instructions for the class

Instructions:

- We will use mode.com for this set of questions.

Caselet - 1

Caselet - 1

We will use **Tutorial.city_populations** data set for Caselet-1 questions

Question-1:

Write a query to return all the records where the city population is more than average population of dataset.

Caselet - 1

Answer-1:

```
SELECT
  *
FROM
  tutorial.city_populations
WHERE
  population_estimate_2012 > (SELECT AVG(population_estimate_2012)
FROM tutorial.city_populations)
```

Caselet - 1

Question-2:

Write a query to return all the records where the city population is more than the most populated city of Texas(TX) state

Caselet - 1

Answer-2:

```
SELECT
  *
FROM
  tutorial.city_populations
WHERE
  population_estimate_2012 > (SELECT MAX(population_estimate_2012)
FROM tutorial.city_populations WHERE state = 'TX')
```

Caselet - 1

Question-3:

Find the number of cities where population is more than the average population of Illinois(IL) state

Caselet - 1

Answer-3:

```
SELECT
  COUNT(city) AS num_cities
FROM
  tutorial.city_populations
WHERE
  population_estimate_2012 > (SELECT AVG(population_estimate_2012)
FROM tutorial.city_populations WHERE state = 'IL')
```

Caselet - 1

Question-4:

Write a query to add the additional column - percentage_population(city population/total population of dataset).

Caselet - 1

Answer-4

```
SELECT
 *,
 100.0 * population_estimate_2012/(SELECT SUM(population_estimate_2012)
FROM tutorial.city_populations) AS percentage_population
FROM
  tutorial.city_populations
```

Caselet - 1

Question-5:

Write a query to add the additional column - percentage_population_state(city population/total population of the state).

Caselet - 1

Answer-5:

```
SELECT
 a.*,
 100.0 * population_estimate_2012/state_population AS percentage_population
FROM
  tutorial.city_populations a
LEFT JOIN
  (
  SELECT
    state,
    SUM(population_estimate_2012) AS state_population
  FROM
    tutorial.city_populations
  GROUP BY
    state
  ) b
ON a.state = b.state
ORDER BY
  a.state
```

Caselet - 1

Question-6:

Write a query to add the additional column - population density. The column logic is:

- Population more than average - High
- Population less than or equal to average - Low

Caselet - 1

Answer-6:

```
SELECT
 *,
  CASE
    WHEN population_estimate_2012 > (SELECT AVG(population_estimate_2012)
FROM tutorial.city_populations)
  THEN
    'High'
  ELSE
    'Low'
  END AS population_density
FROM
  Tutorial.city_populations
```

Caselet - 2

Caselet - 2

We will use **Tutorial.oscar_nominees** for caselet-2 questions

Question-1:

Write a query to return the name of nominees who got more nominations than ‘Akim Tamiroff’. Solve this using CTE.

Caselet - 2

Answer-1:

```
WITH nominees AS (
  SELECT
    nominee,
    COUNT(*) AS nomination_count
  FROM
    tutorial.oscar_nominees
  GROUP BY
    nominee
)
SELECT
  nominee
FROM
  nominees
WHERE
  nomination_count > (SELECT COUNT(*) FROM tutorial.oscar_nominees
WHERE nominee IN ('Akim Tamiroff'))
```

Caselet - 2

Question-2:

Write a query to find the nominee name with the second highest number of oscar wins. Solve using subquery

Caselet - 2

Answer-2:

```
WITH wins AS (
  SELECT
    nominee,
    COUNT(*) AS num_wins
  FROM
      tutorial.oscar_nominees
  WHERE
    winner = true
  GROUP BY
    nominee
  ORDER BY
    num_wins DESC
)
SELECT
    nominee,
    num_wins
FROM
  wins
WHERE
  num_wins = (SELECT MAX(num_wins) FROM
wins WHERE num_wins < (SELECT
MAX(num_wins) FROM wins))
```

Caselet - 2

Question-3:

Write a query to create three columns per nominee

1. Number of wins
2. Number of loss
3. Total nomination

Caselet - 2

Answer-3:

```
SELECT
  nominee,
  SUM(CASE WHEN winner = true THEN 1 ELSE 0 END) AS num_wins,
  SUM(CASE WHEN winner = false THEN 1 ELSE 0 END) AS num_loss,
  COUNT(*) AS total_nomination
FROM
  tutorial.oscar_nominees
GROUP BY
  nominee
ORDER BY
  total_nomination DESC
```

Caselet - 2

Question-4:

Write a query to create two columns

- Win_rate: Number of wins/total wins
- Loss_rate: Number of loss/total wins

Caselet - 2

Answer-4:

```
SELECT
  movie,
  100.0 * SUM(CASE WHEN winner = true THEN 1 ELSE 0 END)/COUNT(*) AS win_rate,
  100.0 * SUM(CASE WHEN winner = false THEN 1 ELSE 0 END)/COUNT(*) AS loss_rate
FROM
  tutorial.oscar_nominees
GROUP BY
  Movie
```

Caselet - 2

Question-5:

Write a query to return all the records of the nominees who have lost but won at least once.

Caselet - 2

Answer-5:

```
SELECT * FROM  tutorial.oscar_nominees
WHERE
  nominee IN (SELECT DISTINCT nominee FROM tutorial.oscar_nominees WHERE winner = true)
  AND winner = false
```

Caselet - 2

Question-6:

Write a query to find the nominees who are nominated for both ‘actor in a leading role’ and ‘actor in supporting role’

Caselet - 2

Answer-6:

```
SELECT
DISTINCT nominee
FROM  tutorial.oscar_nominees
WHERE
  nominee IN (SELECT DISTINCT nominee FROM tutorial.oscar_nominees WHERE category IN ('actor
in a supporting role'))
  AND category IN ('actor in a leading role')
```

Caselet - 2

Question-7:

Write a query to find the movie which won more than average number of wins per winning movie.

Caselet - 2

Answer-7:

```
WITH movie_wins AS (
  SELECT
    movie,
    COUNT(*) AS num_wins
FROM
  tutorial.oscar_nominees
WHERE
  winner = true
GROUP BY
    movie
)
SELECT
  movie
FROM
  movie_wins
WHERE
  num_wins > (SELECT AVG(num_wins) FROM movie_wins)
```

Caselet - 2

Question-8:

Write a query to return the year which have more winners than year 1970

Caselet - 2

Answer-8:

```
WITH year_wins AS (
  SELECT
    year,
    COUNT(*) AS num_wins
FROM
  tutorial.oscar_nominees
WHERE
  winner = true
GROUP BY
    year
)
SELECT
  year
FROM
  year_wins
WHERE
  num_wins > (SELECT num_wins FROM year_wins WHERE year = 1970)
```

Caselet - 2

Question-9:

Write a query to return all the movies which have won oscars both in the actor and actress category.

Caselet - 2

Answer-9:

```
SELECT DISTINCT movie
FROM
  tutorial.oscar_nominees
WHERE
  winner = true
  AND lower(category) LIKE ('%actor%')
  AND movie IN ( SELECT DISTINCT movie FROM tutorial.oscar_nominees WHERE winner = true AND
lower(category) LIKE ('%actress%') )
```

Caselet - 2

Question-10:

Write a query to return the movie name which did not win a single oscar.

Caselet - 2

Answer-10:

```
SELECT DISTINCT movie
FROM
  tutorial.oscar_nominees
WHERE
  winner = false
   AND movie NOT IN ( SELECT DISTINCT movie FROM tutorial.oscar_nominees WHERE winner =
true)
```

Caselet - 3

Caselet - 3

We will be using **tutorial.patient_list** for Caselet-3 questions

Question-1:

Add two additional column in the dataset

- 'Age_category'
  - old_age: >60
  - mid_age: 30-60
  - young: < 30
- Bmi: 703*weight (lbs) /height (inches)^2

Caselet - 3

Answer-1:

```
SELECT
  *,
  CASE
    WHEN age > 60
      THEN 'old_age'
    WHEN age BETWEEN 30 AND 60
      THEN 'mid_age'
    ELSE 'young'
  END AS age_category,
  703.0 * weight_lbs/(height_inches * height_inches) AS BMI
FROM
  Tutorial.patient_list
```

Caselet - 3

Question-2:

Find the physician last_name who treats maximum mid_age patients.

Caselet - 3

Answer-2:

```
SELECT
  physician_last_name,
  COUNT(*) AS patient_count
FROM
(
SELECT
  *,
  CASE
    WHEN age > 60
      THEN 'old_age'
    WHEN age BETWEEN 30 AND 60
      THEN 'mid_age'
    ELSE 'young'
  END AS age_category
FROM
  tutorial.patient_list
) a
WHERE
  age_category = 'mid_age'
GROUP BY
  physician_last_name
ORDER BY
  patient_count DESC
LIMIT 1
```

Caselet - 3

Question-3:

Write a query to return the following for each category:

- Average age
- Max height
- Min weight
- Number of patients

Caselet - 3

Answer-3:

```
SELECT
  age_category,
  AVG(age) AS average_age,
  MAX(height_inches) AS max_height,
  MIN(weight_lbs) AS min_weight,
  COUNT(id) AS num_patients
FROM
(
SELECT
  *,
  CASE
    WHEN age > 60
      THEN 'old_age'
    WHEN age BETWEEN 30 AND 60
      THEN 'mid_age'
    ELSE 'young'
  END AS age_category,
    703.0   *   weight_lbs/(height_inches   *
height_inches) AS BMI
FROM
  tutorial.patient_list
) a
GROUP BY
  age_category
```

Caselet - 3

Question-4:

List all the records where bmi is less than average bmi. Solve using CTE.

Caselet - 3

Answer-4:

```
WITH cte_patient AS (
SELECT
  *,
  CASE
    WHEN age > 60
      THEN 'old_age'
    WHEN age BETWEEN 30 AND 60
      THEN 'mid_age'
    ELSE 'young'
  END AS age_category,
  703.0 * weight_lbs/(height_inches * height_inches) AS BMI
FROM
  tutorial.patient_list
)
SELECT
  *
FROM
  cte_patient
WHERE
  BMI < (SELECT AVG(BMI) FROM cte_patient)
```

Caselet - 4

Caselet - 4

We will be using **Tutorial.sales_performance** for Caselet-4 questions

Question-1:

Write a query to return all the records where sales_revenue is less than the average sales_revenue made by salesperson whose name starts with T. Output should not contain the records of salesperson whose name starts with T

Caselet - 4

Answer-1:

```
SELECT * FROM tutorial.sales_performance
WHERE
  sales_revenue < (SELECT AVG(sales_revenue) FROM tutorial.sales_performance
WHERE salesperson LIKE 'T%')
  AND salesperson NOT LIKE 'T%'
```

Caselet - 4

Question-2:

Write a query to find the record for salesperson with the second lowest sales_revenue.

Caselet - 4

Answer-2:

```
SELECT * FROM tutorial.sales_performance
WHERE
  sales_revenue = (SELECT MIN(sales_revenue) FROM tutorial.sales_performance
WHERE sales_revenue > (SELECT MIN(sales_revenue) FROM tutorial.sales_performance))
```

Caselet - 5

Caselet - 5

We will be using **Tutorial.playbook_users** for Caselet-5 questions

Question-1:

What percentage of users are in ‘pending’ state?

Caselet - 5

Answer-1:

```
SELECT
100.0 * SUM(CASE WHEN state = 'pending' THEN 1 ELSE 0 END )/COUNT(user_id) AS
percentage_pending
FROM
Tutorial.playbook_users
```

Caselet - 5

Question-2:

Find the language with the maximum ‘active’ state percentage.

Caselet - 5

Answer-2

```
SELECT
  language,
  100.0 * SUM(CASE WHEN state = 'active' THEN 1 ELSE 0 END )/COUNT(user_id) AS percentage_active
FROM
  tutorial.playbook_users
GROUP BY
  language
ORDER BY
  percentage_active DESC
LIMIT 1
```

Caselet - 5

Question-3:

Find the percentage of user(out of total dataset) per company.

Caselet - 5

Answer-3:

```
SELECT
  company_id,
  100.0 * COUNT(*)/(SELECT COUNT(user_id) FROM tutorial.playbook_users)  AS percentage_user
FROM
  tutorial.playbook_users
GROUP BY
  company_id
ORDER BY
  percentage_user DESC
```