Hands-on Lab: Generative AI for Querying Databases

Estimated Effort: 30 mins

Introduction

Processed data saved in a database table can be accessed, based on your requirements, using queries. Because queries are an essential part of any data professional's workflow, writing *efficient* queries is a necessary skillset. In this lab, you will learn how you can leverage Generative AI platforms to create optimized queries for your data, provided you can give the model enough context.

Objective(s)

By the end of this lab, you'll be able to prompt a Generative AI model to create efficient queries for your data set.

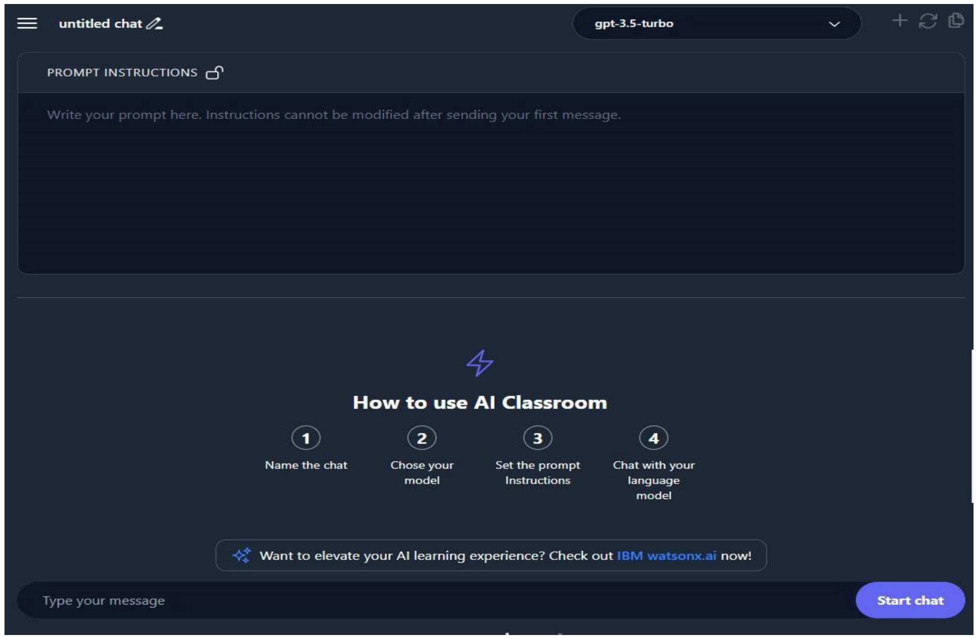
The data set

For the purpose of this lab, you are making use of the Heart Disease Data set from the UCI ML library, available publically under the CCA 4.0 International license.

You can download the data set and run the queries generated in this lab using any SQL querying system.

Giving the context

You might note that there is a section named `Prompt Instructions` on the Generative AI interface.


untitled chat
gpt-3.5-turbo
PROMPT INSTRUCTIONS
Write your prompt here. Instructions cannot be modified after sending your first message.
How to use AI Classroom
1 Name the chat
2 Chose your model
3 Set the prompt Instructions
4 Chat with your language model
Want to elevate your AI learning experience? Check out IBM watsonx.ai now!
Type your message
Start chat

If you provide the model with the description of your data, you can generate efficient and readily usable queries for fetching the data based on your requirements.

Paste the following text in the prompt instructions to give the model the appropriate context for the data.

```
We have a Heart Disease prediction dataset with a single table which has the following attributes.
1.  age - age in years
2.  gender- gender (1 = male; 0 = female)
3.  cp - chest pain type        -- Value 1: typical angina
        -- Value 2: atypical angina
        -- Value 3: non-anginal pain
        -- Value 4: asymptomatic
4.  trestbps - resting blood pressure (in mm Hg on admission to the hospital)
5.  chol - serum cholestoral in mg/dl
6.  fbs - (fasting blood sugar > 120 mg/dl)  (1 = true; 0 = false)
7.  restecg - resting electrocardiographic results        -- Value 0: normal
        -- Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
        -- Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
8.  thalach - maximum heart rate achieved
9.  exang - exercise induced angina (1 = yes; 0 = no)
10. oldpeak - ST depression induced by exercise relative to rest
11. slope - the slope of the peak exercise ST segment        -- Value 1: upsloping
        -- Value 2: flat
        -- Value 3: downsloping
12. ca - number of major vessels (0-3) colored by flourosopy
13. thal - 3 = normal; 6 = fixed defect; 7 = reversable defect
14. num (the predicted attribute) - diagnosis of heart disease (angiographic disease status)         -- Value 0: < 50% diameter narrowing
        -- Value 1: > 50% diameter narrowing
```

The model will now have enough context to generate SQL queries for your prompts.

> Note: The prompt instructions are editable only until the first prompt is executed. Thereafter, you cannot change the instructions. If a mistake occurs and you want to start over, you can generate a fresh chat and add the correct prompt instructions before generating the SQL queries.

Prompts for Data Querying

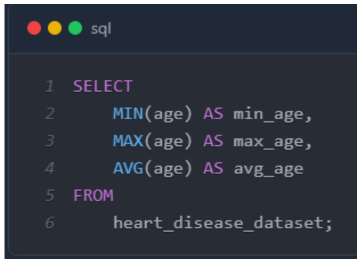
Age Distribution

Consider the following prompt.

```
Write an SQL query to find the minimum, maximum, and average age of patients in the dataset.
```

You can expect to the see the following query in response.

```
SELECT
    MIN(age) AS min_age,
    MAX(age) AS max_age,
    AVG(age) AS avg_age FROM
heart_disease_prediction_dataset;
```



```sql
SELECT
    MIN(age) AS min_age,
    MAX(age) AS max_age,
    AVG(age) AS avg_age
FROM
    heart_disease_dataset;
```

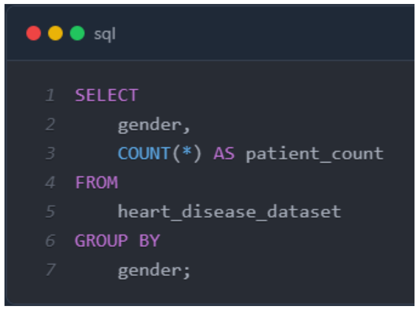
Gender Analysis

Consider the following prompt.

```
Write and SQL query to count the number of male and female patients in the dataset.
```

You can expect to the see the following query in response.

```
SELECT
gender,
    COUNT(*) AS patient_count FROM
    heart_disease_prediction_dataset
GROUP BY
gender;
```



```
sql
SELECT
    gender,
    COUNT(*) AS patient_count
FROM
    heart_disease_dataset
GROUP BY
    gender;
```

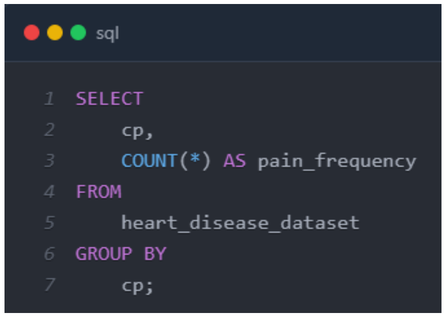
Chest Pain Type Frequency

Consider the following prompt.

```
Write an SQL query to determine the frequency of each type of chest pain (typical angina, atypical angina, non-anginal pain, asymptomatic) among patients."
```

You can expect the following query to be generated.

```
SELECT
cp,
    COUNT(*) AS pain_frequency FROM
    heart_disease_prediction_dataset
GROUP BY
cp;
```



```
sql
SELECT
    cp,
    COUNT(*) AS pain_frequency
FROM
    heart_disease_dataset
GROUP BY
    cp;
```

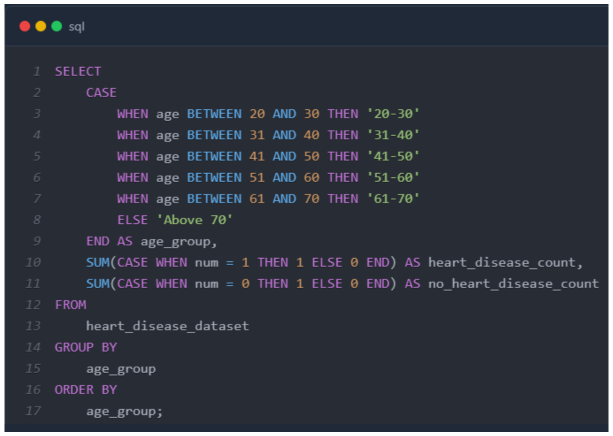
Age Group Analysis and Target Variable

Consider the following prompt.

```
Write an SQL query to investigate the distribution of the target variable (presence or absence of heart disease) within different age groups (e.g., 20-30, 30-40, etc.).
```

You can expect the following output.

```
SELECT
    CASE
        WHEN age BETWEEN 20 AND 30 THEN '20-30'
        WHEN age BETWEEN 31 AND 40 THEN '31-40'
        WHEN age BETWEEN 41 AND 50 THEN '41-50'
        WHEN age BETWEEN 51 AND 60 THEN '51-60'
        WHEN age BETWEEN 61 AND 70 THEN '61-70'
        ELSE 'Above 70'
    END AS age_group,
    SUM(CASE WHEN num = 1 THEN 1 ELSE 0 END) AS heart_disease_count,
    SUM(CASE WHEN num = 0 THEN 1 ELSE 0 END) AS no_heart_disease_count FROM
    heart_disease_prediction_dataset
GROUP BY
age_group
ORDER BY
age_group;
```



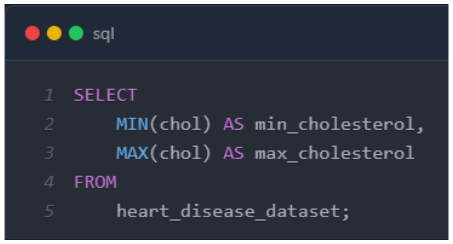
```
sql
SELECT
    CASE
        WHEN age BETWEEN 20 AND 30 THEN '20-30'
        WHEN age BETWEEN 31 AND 40 THEN '31-40'
        WHEN age BETWEEN 41 AND 50 THEN '41-50'
        WHEN age BETWEEN 51 AND 60 THEN '51-60'
        WHEN age BETWEEN 61 AND 70 THEN '61-70'
        ELSE 'Above 70'
    END AS age_group,
    SUM(CASE WHEN num = 1 THEN 1 ELSE 0 END) AS heart_disease_count,
    SUM(CASE WHEN num = 0 THEN 1 ELSE 0 END) AS no_heart_disease_count
FROM
    heart_disease_dataset
GROUP BY
    age_group
ORDER BY
    age_group;
```

Practice Prompts

Try to generate queries for the data set for the following prompts:

1. Cholesterol Range:

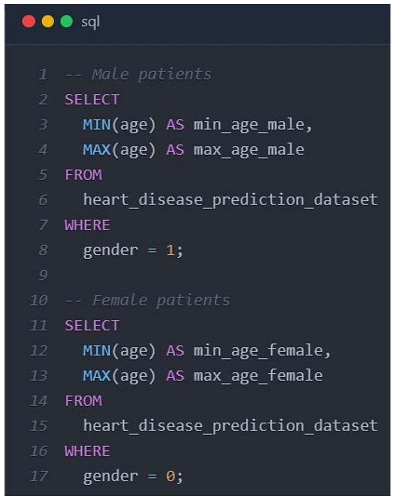
   `Find the range of cholesterol levels among patients (minimum, maximum).`



```sql
SELECT
    MIN(chol) AS min_cholesterol,
    MAX(chol) AS max_cholesterol
FROM
    heart_disease_dataset;
```

2. Age Range and Gender Analysis:

   `Determine the age range (youngest and oldest) for male and female patients separately.`

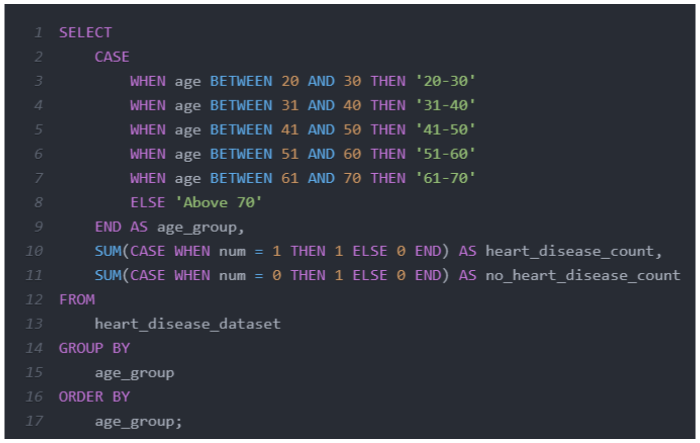


```sql
-- Male patients
SELECT
  MIN(age) AS min_age_male,
  MAX(age) AS max_age_male
FROM
  heart_disease_prediction_dataset
WHERE
  gender = 1;

-- Female patients
SELECT
  MIN(age) AS min_age_female,
  MAX(age) AS max_age_female
FROM
  heart_disease_prediction_dataset
WHERE
  gender = 0;
```

3. Age Group Analysis and Target Variable:

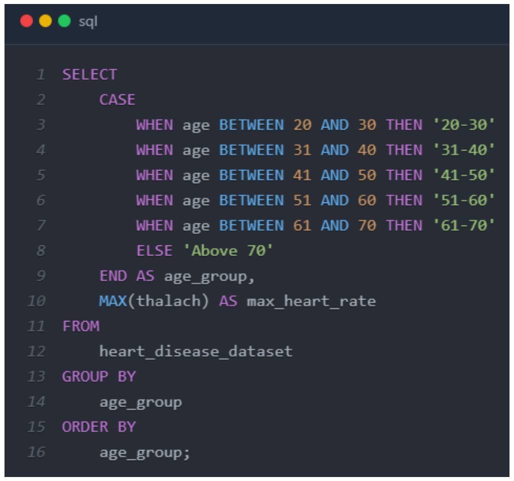
   `Investigate the distribution of the target variable (presence or absence of heart disease) within different age groups (e.g., 20-30, 30-40, etc.).`



```sql
SELECT
    CASE
        WHEN age BETWEEN 20 AND 30 THEN '20-30'
        WHEN age BETWEEN 31 AND 40 THEN '31-40'
        WHEN age BETWEEN 41 AND 50 THEN '41-50'
        WHEN age BETWEEN 51 AND 60 THEN '51-60'
        WHEN age BETWEEN 61 AND 70 THEN '61-70'
        ELSE 'Above 70'
    END AS age_group,
    SUM(CASE WHEN num = 1 THEN 1 ELSE 0 END) AS heart_disease_count,
    SUM(CASE WHEN num = 0 THEN 1 ELSE 0 END) AS no_heart_disease_count
FROM
    heart_disease_dataset
GROUP BY
    age_group
ORDER BY
    age_group;
```

4. Maximum Heart Rate by Age Group:

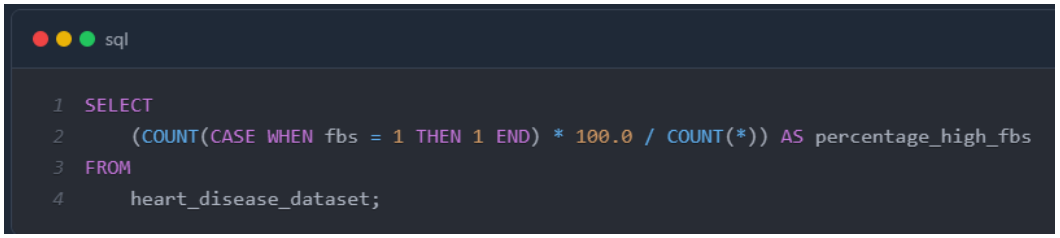
```
Find the maximum heart rate achieved during exercise for different age groups (e.g., 30-40, 40-50, etc.).
```



```sql
SELECT
    CASE
        WHEN age BETWEEN 20 AND 30 THEN '20-30'
        WHEN age BETWEEN 31 AND 40 THEN '31-40'
        WHEN age BETWEEN 41 AND 50 THEN '41-50'
        WHEN age BETWEEN 51 AND 60 THEN '51-60'
        WHEN age BETWEEN 61 AND 70 THEN '61-70'
        ELSE 'Above 70'
    END AS age_group,
    MAX(thalach) AS max_heart_rate
FROM
    heart_disease_dataset
GROUP BY
    age_group
ORDER BY
    age_group;
```

5. Percentage of Patients with High Blood Sugar:

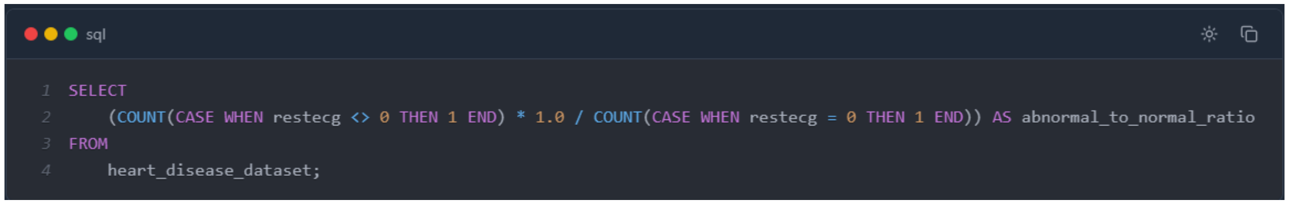
```
Calculate the percentage of patients with fasting blood sugar greater than 120 mg/dl.
```



```sql
SELECT
    (COUNT(CASE WHEN fbs = 1 THEN 1 END) * 100.0 / COUNT(*)) AS percentage_high_fbs
FROM
    heart_disease_dataset;
```

6. Ratio of Patients with Resting Electrocardiographic Abnormality:

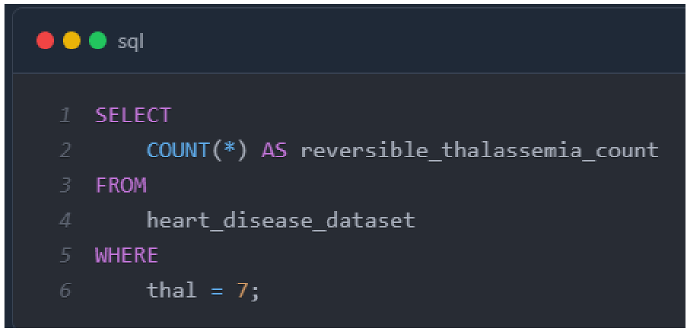
```
Find the ratio of patients with abnormal resting electrocardiographic results to those with normal results.
```



```sql
SELECT
    (COUNT(CASE WHEN restecg <> 0 THEN 1 END) * 1.0 / COUNT(CASE WHEN restecg = 0 THEN 1 END)) AS abnormal_to_normal_ratio
FROM
    heart_disease_dataset;
```

7. Number of Patients with Reversible Thalassemia:

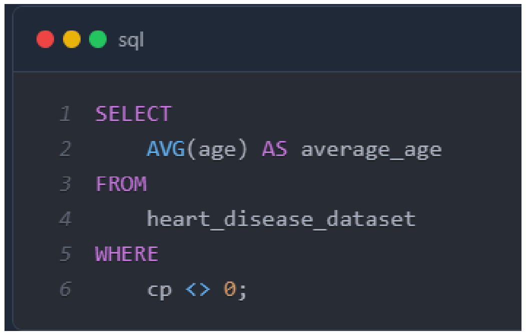
Count the number of patients with reversible thalassemia detected by thallium stress testing.



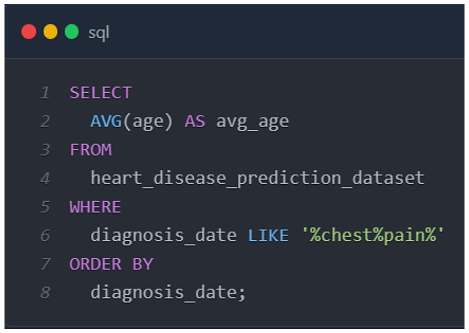
```sql
SELECT
    COUNT(*) AS reversible_thalassemia_count
FROM
    heart_disease_dataset
WHERE
    thal = 7;
```

8. Average Age of Patients with Chest Pain:

Calculate the average age of patients who experienced chest pain during diagnosis.



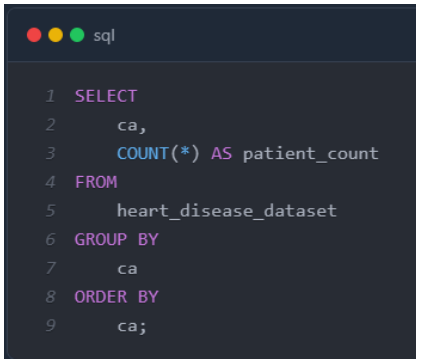
```sql
SELECT
    AVG(age) AS average_age
FROM
    heart_disease_dataset
WHERE
    cp <> 0;
```



```sql
SELECT
  AVG(age) AS avg_age
FROM
  heart_disease_prediction_dataset
WHERE
  diagnosis_date LIKE '%chest%pain%'
ORDER BY
  diagnosis_date;
```

9. Distribution of Patients by Number of Major Vessels:

Investigate the distribution of patients based on the number of major vessels colored by fluoroscopy (0-3).



```sql
SELECT
    ca,
    COUNT(*) AS patient_count
FROM
    heart_disease_dataset
GROUP BY
    ca
ORDER BY
    ca;
```

Conclusion

Congratulations on your successful completion of this lab!

You should now be able to use Generative AI to create efficient queries for retrieving relevant information from a database. Please note, that you need to provide the model with a detailed description of the attributes as context, to generate efficient and ready-to-use prompts.

Author(s)

Abhishek Gagneja