response\_Analysis\_23\_PA - > the document shows the process of achieving the right answer with the usage of prompts only. The final result has been evaluated as partially answered.

Initial prompt: (GPT3.5-turbo, temp:0,5, max-length:1200)  
  
SYSTEM: {

*Question ID: 23*

*Question: What are the main causes of bottlenecks?*

*Prompt: You are a skilled data analyst. Your role is to explain which steps are needed in order to answer the question asked by user. The solution would be written in MS SQL and would query database in order to directly find the answer. Analyze the question and describe in multiply steps how MSSQL query should be built in order to answer the question.*

*The analyst is working on following data*

*The event\_log data for the process is stored in the table:BPI\_Challenge\_2019*

*columns:['case\_concept\_name', 'event\_concept\_name', 'event\_time\_timestamp' ]*

*column\_types:['nvarchar', 'nvarchar','datetime2' ].*

*In process mining, a case represents a single instance of a business process.*

*It has a unique ID and attributes like cycle time and costs.*

*Cases are used to analyze and improve processes .*

*ID of a case is stored in case\_concept\_name column.*

*Each case is composed of multiple events.*

*An event name, and identifier, is stored in event\_concept\_name.*

*The time of execution of the event is stored in event\_time\_timestamp.*

*Use minutes in datediff instead of seconds.*

*Use the FOR XML PATH() function to concatenate values instead of STRING\_AGG()*

*If not additionaly specified task duration is a time duration between an event and consecutive event.*

*bottleneck is expectionally long time between two events following each other. For description of bottleneck use event1 -> event2*

*keywords important for answering:*

*['bottlenecks: Bottlenecks are points in a process where the flow of work is slowed down or blocked altogether. \n In BPI\_Challenge\_2019, bottlenecks can be identified by analyzing cycle time and identifying events or activities that take an overly long time to complete.', '\n', 'causes: Causes are factors or events that generate a specific outcome in a process. \n In BPI\_Challenge\_2019, causes can be analyzed by examining various attributes such as case\_Spend\_area\_text, case\_Document\_Type, case\_Purch\_Doc\_Category\_name, case\_Item\_Type, and case\_Item\_Category.', '\n']*

*QUESTION:*

*What are the main causes of bottlenecks?*

*Write an answer in following order*

*STEPS:*

*here are the steps to acomplish the task. Do not add additional text or explenation outside of STEPS part. DO not add code outside of CODE part.*

*CODE:*

*```*

*here should be an example MS SQL Query*

*DO NOT USE '--' or any comments in SQL code. Put all comments into description!!*

*```*

*DESCRIPTION:*

*here is a description and reasoning behind the above approach.}*

USER : {

*What are the main causes of bottlenecks?*

}

ASSISTANT: {

STEPS:

1. Calculate the duration between consecutive events for each case.

2. Identify exceptionally long durations as bottlenecks.

3. Group by event pairs (event1 -> event2) to find the most common bottlenecks.

4. Calculate the average duration for each bottleneck event pair.

CODE:

```

WITH EventDurations AS (

SELECT

case\_concept\_name,

event\_concept\_name,

event\_time\_timestamp,

LEAD(event\_time\_timestamp) OVER (PARTITION BY case\_concept\_name ORDER BY event\_time\_timestamp) AS next\_event\_time

FROM

BPI\_Challenge\_2019

),

Bottlenecks AS (

SELECT

case\_concept\_name,

event\_concept\_name + ' -> ' + LEAD(event\_concept\_name) OVER (PARTITION BY case\_concept\_name ORDER BY event\_time\_timestamp) AS event\_pair,

DATEDIFF(minute, event\_time\_timestamp, next\_event\_time) AS duration

FROM

EventDurations

)

SELECT

event\_pair,

COUNT(\*) AS count,

AVG(duration) AS avg\_duration

FROM

Bottlenecks

WHERE

duration > (SELECT AVG(duration) \* 2 FROM Bottlenecks)

GROUP BY

event\_pair

ORDER BY

count DESC,

avg\_duration DESC

```

DESCRIPTION:

The above query first calculates the duration between consecutive events for each case using the LEAD window function in the EventDurations CTE. Then, it identifies exceptionally long durations (more than twice the average duration) as bottlenecks in the Bottlenecks CTE. Finally, it groups the data by event pairs (event1 -> event2) and calculates the count and average duration for each bottleneck event pair, ordering the results by the highest count and average duration.

}

USER:  
error from sql server: Started executing query at Line 1

Msg 8115, Level 16, State 2, Line 1

Arithmetic overflow error converting expression to data type int.

Warning: Null value is eliminated by an aggregate or other SET operation.

Total execution time: 00:00:00.624

Fix the code above

Do not apologize!!!!

ASSISTANT:

STEPS:

1. Calculate the duration between consecutive events for each case.

2. Identify exceptionally long durations as bottlenecks.

3. Group by event pairs (event1 -> event2) to find the most common bottlenecks.

4. Calculate the average duration for each bottleneck event pair.

CODE:

```

WITH EventDurations AS (

SELECT

case\_concept\_name,

event\_concept\_name,

event\_time\_timestamp,

LEAD(event\_time\_timestamp) OVER (PARTITION BY case\_concept\_name ORDER BY event\_time\_timestamp) AS next\_event\_time

FROM

BPI\_Challenge\_2019

),

Bottlenecks AS (

SELECT

case\_concept\_name,

event\_concept\_name + ' -> ' + LEAD(event\_concept\_name) OVER (PARTITION BY case\_concept\_name ORDER BY event\_time\_timestamp) AS event\_pair,

DATEDIFF\_BIG(minute, event\_time\_timestamp, next\_event\_time) AS duration

FROM

EventDurations

)

SELECT

event\_pair,

COUNT(\*) AS count,

AVG(CAST(duration AS bigint)) AS avg\_duration

FROM

Bottlenecks

WHERE

duration > (SELECT AVG(CAST(duration AS bigint)) \* 2 FROM Bottlenecks)

GROUP BY

event\_pair

HAVING

event\_pair IS NOT NULL

ORDER BY

count DESC,

avg\_duration DESC

```

DESCRIPTION:

The above query first calculates the duration between consecutive events for each case using the LEAD window function in the EventDurations CTE. Then, it identifies exceptionally long durations (more than twice the average duration) as bottlenecks in the Bottlenecks CTE. The DATEDIFF\_BIG function is used to avoid arithmetic overflow errors when calculating durations. The duration is casted to bigint to handle larger durations. Finally, it groups the data by event pairs (event1 -> event2) and calculates the count and average duration for each bottleneck event pair, ordering the results by the highest count and average duration. The HAVING clause is added to exclude NULL event pairs from the result.

USER:  
The count of returned objects is too high. Correct the code to return only 10 top elements.Do not apologize!!!

STEPS:

1. Calculate the duration between consecutive events for each case.

2. Identify exceptionally long durations as bottlenecks.

3. Group by event pairs (event1 -> event2) to find the most common bottlenecks.

4. Calculate the average duration for each bottleneck event pair.

5. Limit the results to the top 10 bottlenecks.

CODE:

```

WITH EventDurations AS (

SELECT

case\_concept\_name,

event\_concept\_name,

event\_time\_timestamp,

LEAD(event\_time\_timestamp) OVER (PARTITION BY case\_concept\_name ORDER BY event\_time\_timestamp) AS next\_event\_time

FROM

BPI\_Challenge\_2019

),

Bottlenecks AS (

SELECT

case\_concept\_name,

event\_concept\_name + ' -> ' + LEAD(event\_concept\_name) OVER (PARTITION BY case\_concept\_name ORDER BY event\_time\_timestamp) AS event\_pair,

DATEDIFF\_BIG(minute, event\_time\_timestamp, next\_event\_time) AS duration

FROM

EventDurations

)

SELECT TOP 10

event\_pair,

COUNT(\*) AS count,

AVG(CAST(duration AS bigint)) AS avg\_duration

FROM

Bottlenecks

WHERE

duration > (SELECT AVG(CAST(duration AS bigint)) \* 2 FROM Bottlenecks)

GROUP BY

event\_pair

HAVING

event\_pair IS NOT NULL

ORDER BY

count DESC,

avg\_duration DESC

```

DESCRIPTION:

The above query first calculates the duration between consecutive events for each case using the LEAD window function in the EventDurations CTE. Then, it identifies exceptionally long durations (more than twice the average duration) as bottlenecks in the Bottlenecks CTE. The DATEDIFF\_BIG function is used to avoid arithmetic overflow errors when calculating durations. The duration is casted to bigint to handle larger durations. Finally, it groups the data by event pairs (event1 -> event2) and calculates the count and average duration for each bottleneck event pair, ordering the results by the highest count and average duration. The HAVING clause is added to exclude NULL event pairs from the result. The TOP 10 clause is added to limit the results to the top 10 bottlenecks.

SQL:

**Answer evaluated as partially answered. The LLM response returns the main bottlenecks in process but does not explain the causes of them.**

A screenshot of a computer

Description automatically generated with medium confidence