LLM As DBA

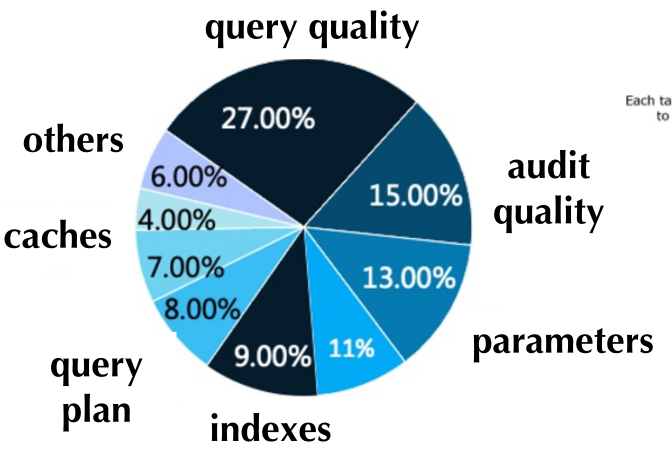
数据库虚拟人

Database Stability is Critical

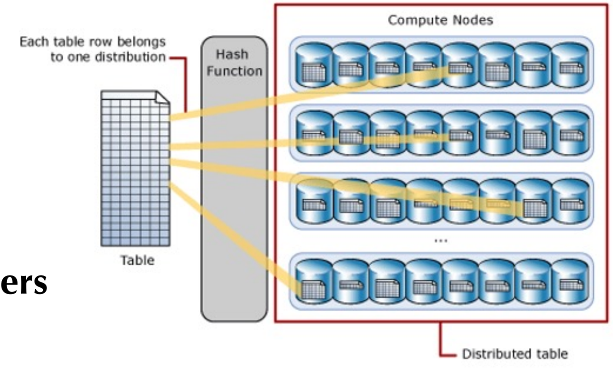
“When selecting database for financial institutions, three most crucial factors are: **Stability**, **Security**, and **Performance**, accounting for 80.4%, 72.6%, and 59.6% respectively.”

Increasing Pressure on Database Operations and Maintenance

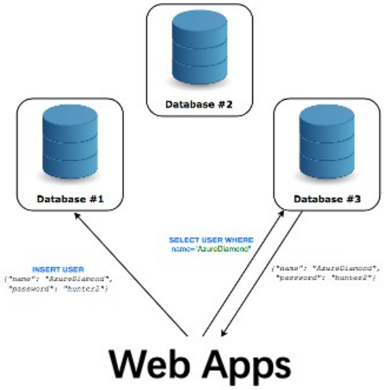
- **Numerous Documents:** A single database product provides over **10,000+ pages** of materials (e.g., operational guides and instructions). It's tricky for junior DBAs to fully grasp all of this.
- **Significant Operational Pressure:** The number of cloud database instances is massive, but there's a shortage of DBAs.
- **Complex Issues：** Many urgent problems are interconnected, making it hard to respond quickly, resulting in economic losses.



Various Root Causes



1M+ Cloud Instances

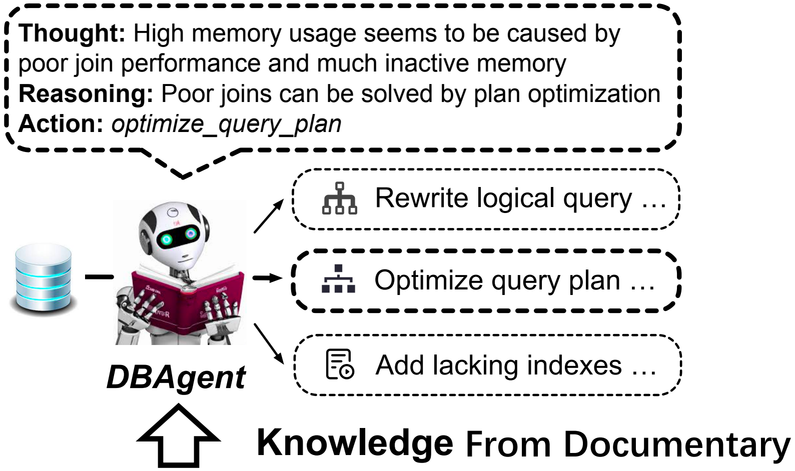

Database #2
Database #1
Database #3
SELECT USER WHERE name="AzureDiamond"
INSERT USER {"name": "AzureDiamond", "password": "hunter2"}
{"name": "AzureDiamond", "password": "hunter2"}
Web Apps

Correlated Online Issues

Vision: LLM As DBA

- **Goal:** Utilize LLM as an "intelligent DBA assistant", learning from human operational experience and diagnose root causes.
- **Benefits:** Save manpower; Enhance proficiency in using maintenance tools; Improve the ability to generalize operational knowledge.

Other methods like *expert systems* are hard to reuse similar cases



Thought: High memory usage seems to be caused by poor join performance and much inactive memory
Reasoning: Poor joins can be solved by plan optimization
Action: *optimize_query_plan*

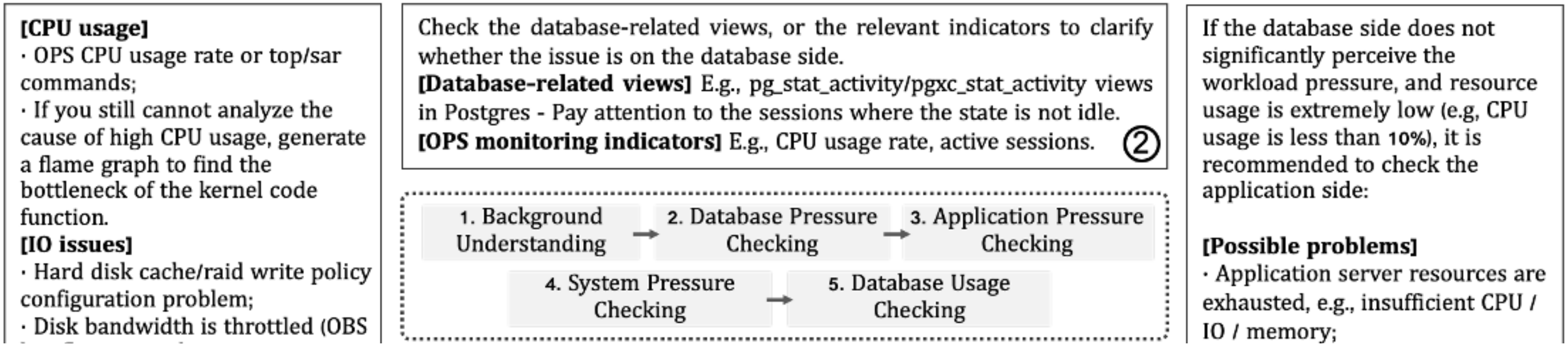
Rewrite logical query …

Optimize query plan …

Add lacking indexes …

DBAgent

Knowledge From Documentary



[CPU usage]
· OPS CPU usage rate or top/sar commands;
· If you still cannot analyze the cause of high CPU usage, generate a flame graph to find the bottleneck of the kernel code function.
[IO issues]
· Hard disk cache/raid write policy configuration problem;
· Disk bandwidth is throttled (OBS

Check the database-related views, or the relevant indicators to clarify whether the issue is on the database side.
[Database-related views] E.g., pg_stat_activity/pgxc_stat_activity views in Postgres - Pay attention to the sessions where the state is not idle.
[OPS monitoring indicators] E.g., CPU usage rate, active sessions. ②

1. Background Understanding → 2. Database Pressure Checking → 3. Application Pressure Checking
4. System Pressure Checking → 5. Database Usage Checking

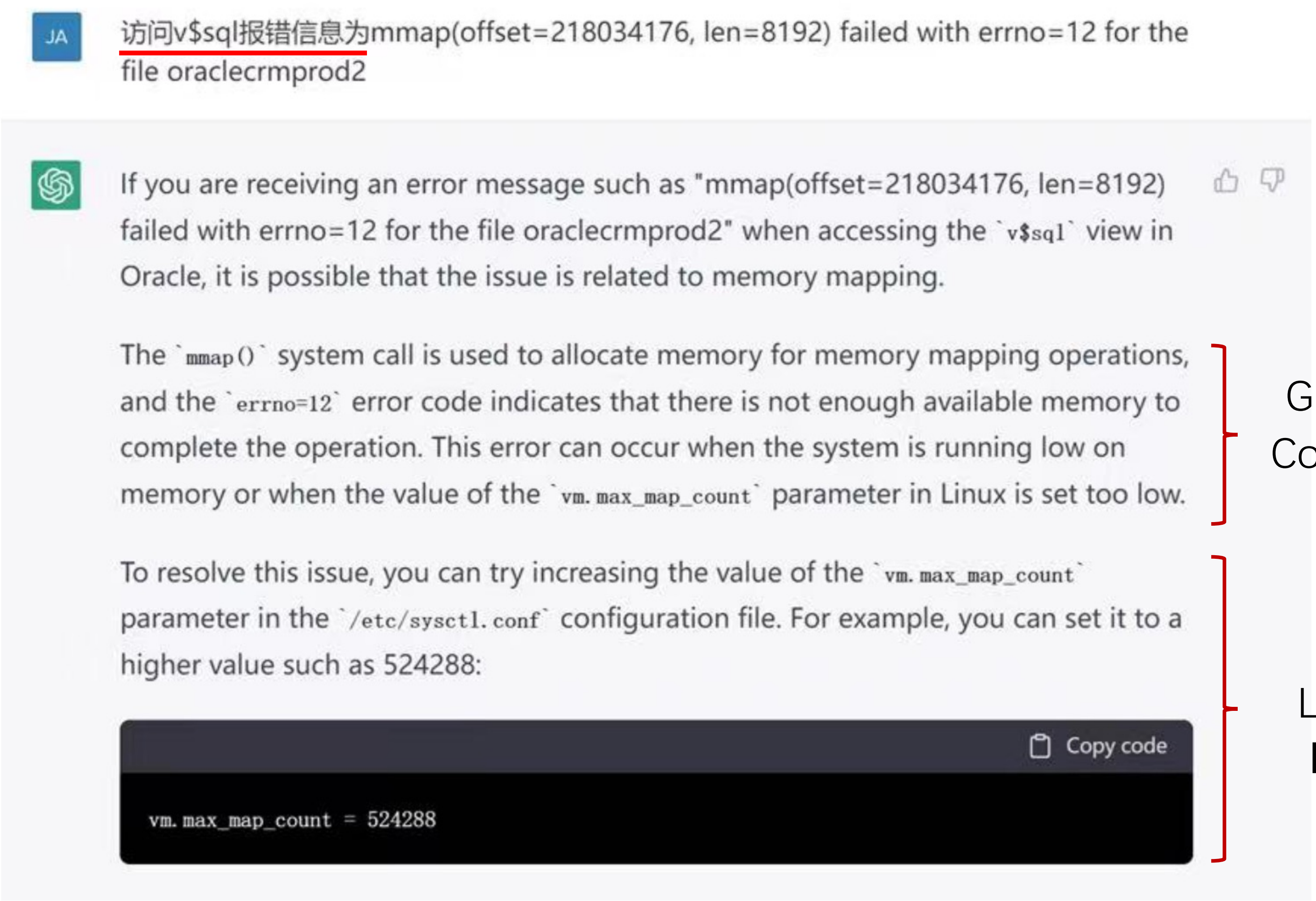
If the database side does not significantly perceive the workload pressure, and resource usage is extremely low (e.g, CPU usage is less than 10%), it is recommended to check the application side:

[Possible problems]
· Application server resources are exhausted, e.g., insufficient CPU / IO / memory;

First Period: Q&A Assistant

- First Period

Question ⇒ DBAgent



访问v$sql报错信息为mmap(offset=218034176, len=8192) failed with errno=12 for the file oraclecrmprod2

If you are receiving an error message such as "mmap(offset=218034176, len=8192) failed with errno=12 for the file oraclecrmprod2" when accessing the `v$sql` view in Oracle, it is possible that the issue is related to memory mapping.

The `mmap()` system call is used to allocate memory for memory mapping operations, and the `errno=12` error code indicates that there is not enough available memory to complete the operation. This error can occur when the system is running low on memory or when the value of the `vm.max_map_count` parameter in Linux is set too low.

To resolve this issue, you can try increasing the value of the `vm.max_map_count` parameter in the `/etc/sysctl.conf` configuration file. For example, you can set it to a higher value such as 524288:

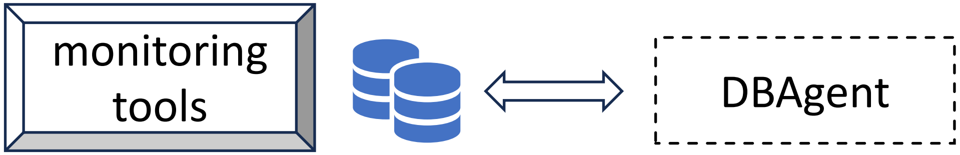
```
vm.max_map_count = 524288
```

General Concepts

Lack of **Basis**

Second Period: Diagnosis Assistant

- Second Period


monitoring tools

DBAgent

[Chief DBA]: The anomaly started at 1691897340 and ended at 1691897430.
[CPU Agent]: Based on the CPU relevant metric values, … . I suggest reviewing system logs and identifying any recent changes or updates. We should also analyze other relevant metrics such as memory usage, disk I/O, network traffic, and check for any abnormal behavior from specific processes or applications.

(matched knowledge) missing_index

[Memory Agent]: The root cause of the high memory usage could be disk spill and insufficient memory allocation. I suggest analyzing the actual resource usage during query execution, performing a thorough analysis of the query plans, and evaluating the system configuration settings related to memory allocation, such as 'work_mem' and 'shared_buffers.' More information about the sort spill count or hash spill count and the threshold values is needed to confirm if disk spill is the root cause. If the free memory is consistently low and close to the total memory capacity, consider increasing the memory allocation for the system.

(matched knowledge) disk_spill

Third Period: Optimization Assistant

- Third Period

monitoring tools

optimization tools

DBAgent

[Chief DBA]: The anomaly started at 1691897340 and ended at 1691897430.
[CPU Agent]: The anomaly is caused by high CPU due to the high number of running processes and the relatively high load averages, especially in the past 1 minute. In addition, slow queries also contribute to the high CPU usage. … 5. Add the recommended indexes: ['movie_info#movie_id', 'movie_companies#movie_id', 'keyword#keyword', 'aka_name#person_id'] to reduce cost from 46856716.92000003 to 31393898.919999983.

(matched knowledge) high number of running processes

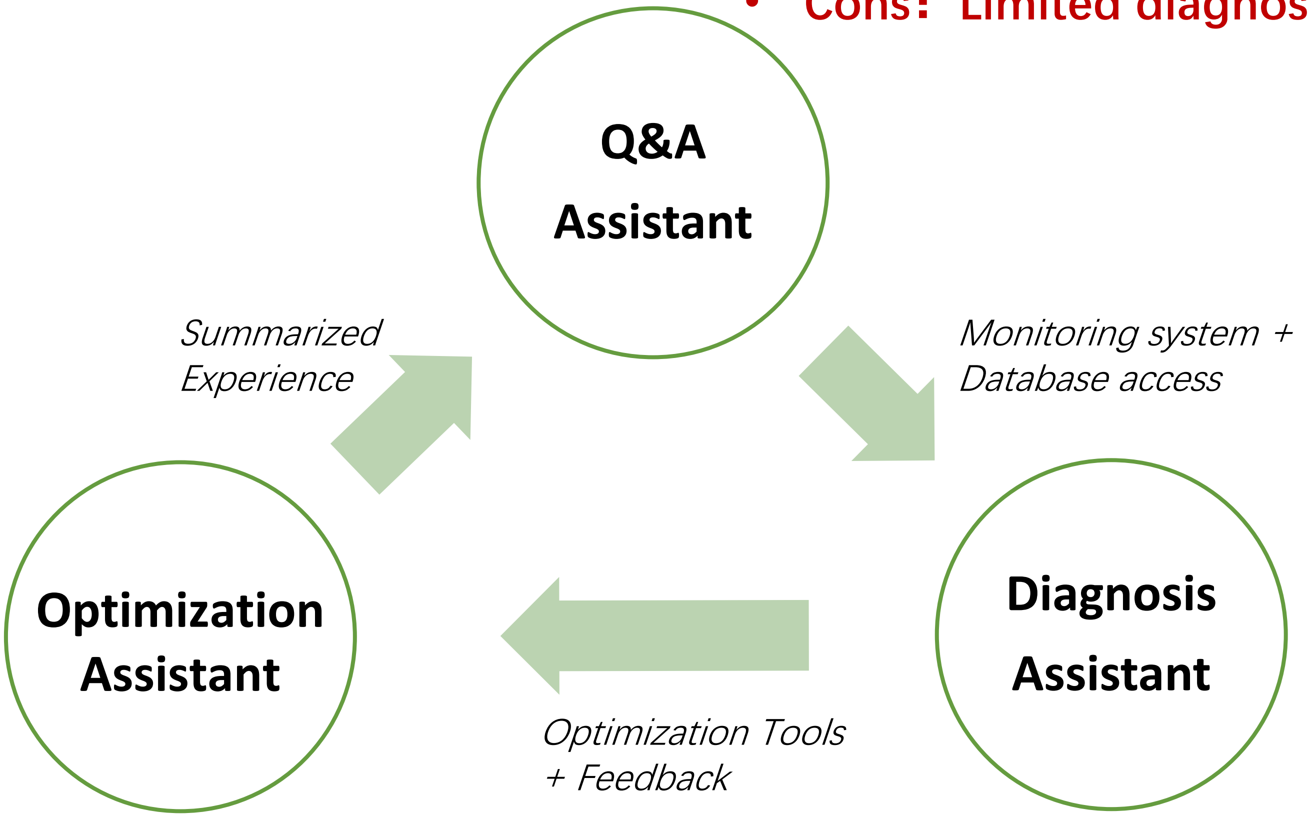
…

[Chief DBA]: The identified root causes of the anomaly are high CPU usage due to the high number of running processes and high load averages, slow queries, and high memory due to complex queries, insufficient memory allocation, and the possibility of disk spill.

(solution) Add the recommended indexes: ['movie_info#movie_id', 'movie_companies#movie_id', 'keyword#keyword', 'aka_name#person_id'] to reduce cost.

Pros And Cons

- Pros： General, easy-to-deploy
- Cons： Limited diagnosis, hallucination



Q&A Assistant

Monitoring system + Database access

Diagnosis Assistant

Optimization Tools + Feedback

Optimization Assistant

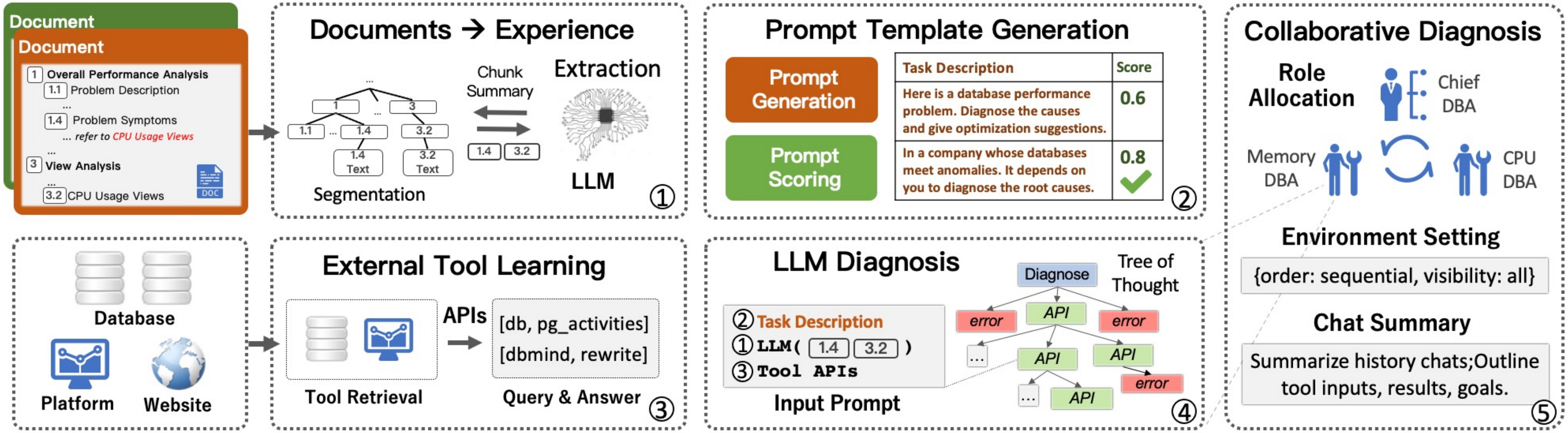
Summarized Experience

- Pros： Higher Diag Accuracy
- Cons： Model capability, deploy costs

- Pros： Finer-grained & founded diag
- Cons： Model capability, deploy costs

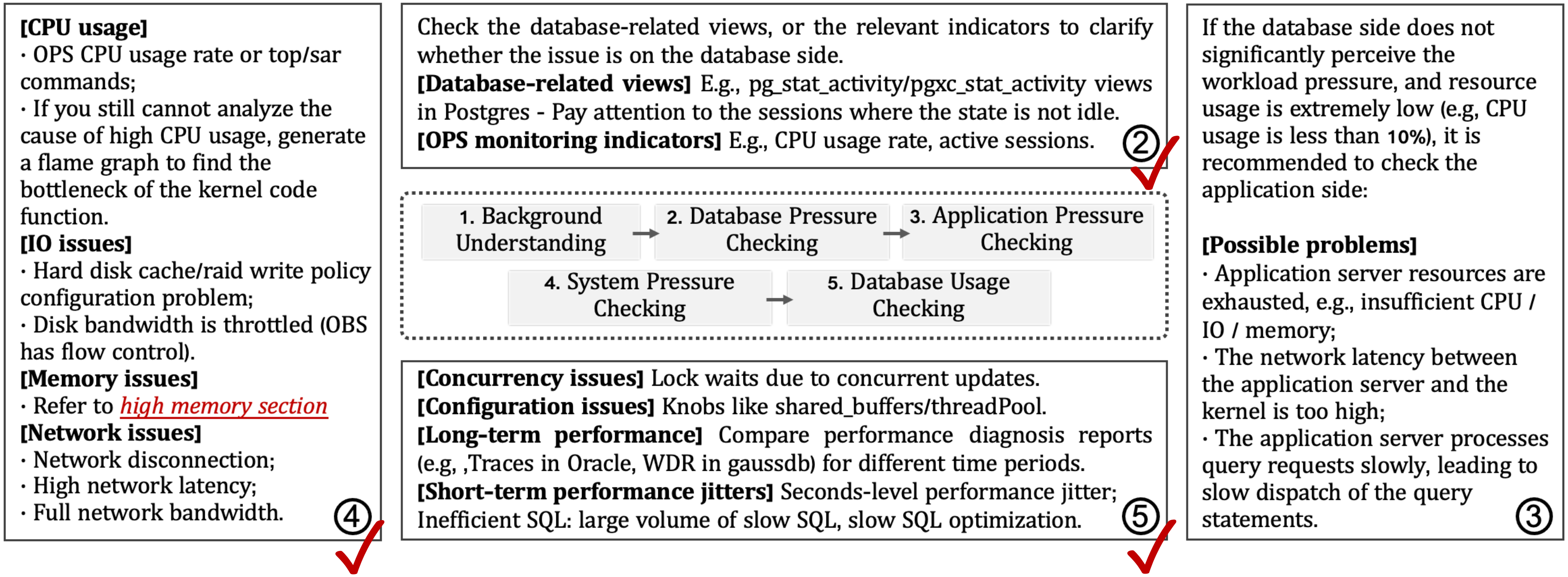
Framework

- **Observable system:** Original / processed data (e.g., logs, views, metrics).
- **Extraction of operational knowledge:** Segmentation of text blocks; generation of summary indexes; extraction of formalized knowledge.
- **Learning from external tools:** Learning API calling strategies; improving tool usage through tree-based searches.
- **Collaborative diagnostic optimization:** Chief DBA (diagnostic scheduling, finding summarization), sub-domain experts (e.g., resource anomaly, query optimization), conversation summarizers, users (providing effective feedback).



Typical Anomalies

1. **Background:** Customer expectations, business type, recent business changes, system changes, and etc.
2. **Database Kernel Pressure:** CPU usage of database server, database-related views, OPS-related metrics.
3. **Application-side Pressure:** Depletion of application server resources, high latency in application server and kernel networking.
4. **Database Resources:** CPU saturation, full I/O anomalies, insufficient memory, network anomalies.
5. **Database Utilization:** Concurrency, configurations, waits, non-optimal SQL.



[CPU usage]
- OPS CPU usage rate or top/sar commands;
- If you still cannot analyze the cause of high CPU usage, generate a flame graph to find the bottleneck of the kernel code function.

[IO issues]
- Hard disk cache/raid write policy configuration problem;
- Disk bandwidth is throttled (OBS has flow control).

[Memory issues]
- Refer to *high memory section*

[Network issues]
- Network disconnection;
- High network latency;
- Full network bandwidth.

④ ✓

Check the database-related views, or the relevant indicators to clarify whether the issue is on the database side.
[Database-related views] E.g., pg_stat_activity/pgxc_stat_activity views in Postgres - Pay attention to the sessions where the state is not idle.
[OPS monitoring indicators] E.g., CPU usage rate, active sessions.

② ✓

1. Background Understanding → 2. Database Pressure Checking → 3. Application Pressure Checking

4. System Pressure Checking → 5. Database Usage Checking

[Concurrency issues] Lock waits due to concurrent updates.
[Configuration issues] Knobs like shared_buffers/threadPool.
[Long-term performance] Compare performance diagnosis reports (e.g, ,Traces in Oracle, WDR in gaussdb) for different time periods.
[Short-term performance jitters] Seconds-level performance jitter; Inefficient SQL: large volume of slow SQL, slow SQL optimization.

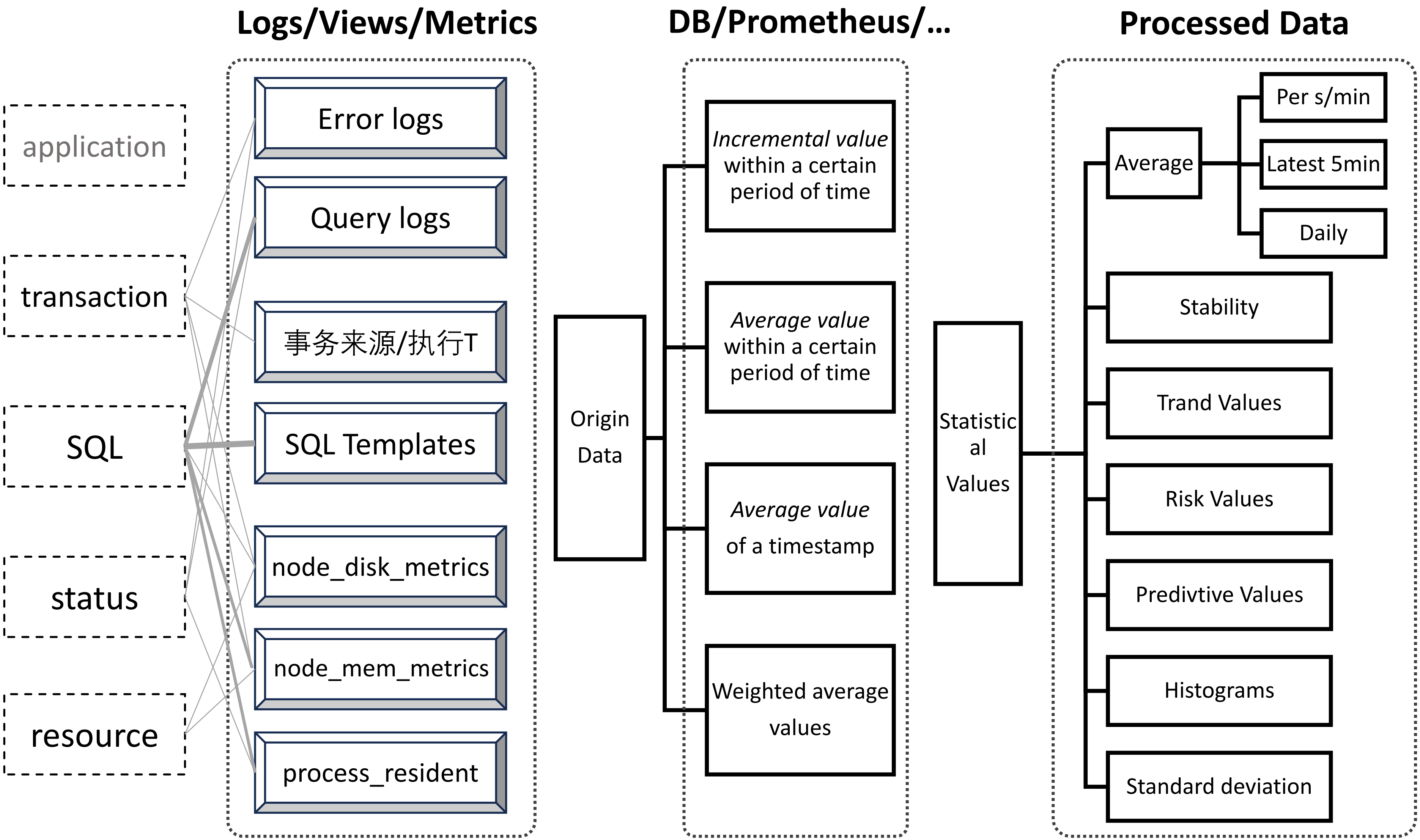
⑤ ✓

If the database side does not significantly perceive the workload pressure, and resource usage is extremely low (e.g, CPU usage is less than 10%), it is recommended to check the application side:

[Possible problems]
- Application server resources are exhausted, e.g., insufficient CPU / IO / memory;
- The network latency between the application server and the kernel is too high;
- The application server processes query requests slowly, leading to slow dispatch of the query statements.

③

Technology 1: Observable System



Logs/Views/Metrics

DB/Prometheus/…

Processed Data

application

transaction

SQL

status

resource

Error logs

Query logs

事务来源/执行T

SQL Templates

node_disk_metrics

node_mem_metrics

process_resident

Origin Data

Incremental value within a certain period of time

Average value within a certain period of time

Average value of a timestamp

Weighted average values

Statistical Values

Average

Per s/min

Latest 5min

Daily

Stability

Trand Values

Risk Values

Predivtive Values

Histograms

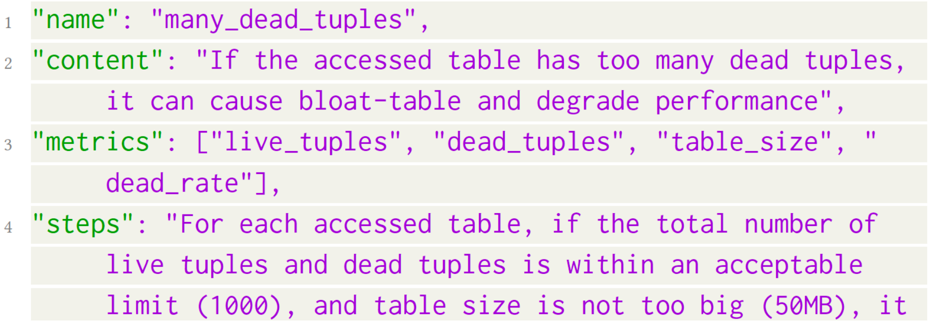
Standard deviation

Technology 2: Knowledge Extraction from Documents

Knowledge Extraction: Long text with interconnected sections

1. Semantic-based Text Segmentation

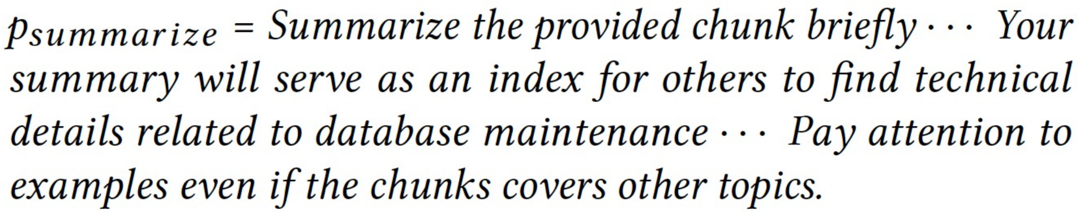
- Identify and split by chapter identifiers
- For chapters exceeding 1,000 tokens, use a tree-like split approach



```
"name": "many_dead_tuples",
"content": "If the accessed table has too many dead tuples,
    it can cause bloat-table and degrade performance",
"metrics": ["live_tuples", "dead_tuples", "table_size", "
    dead_rate"],
"steps": "For each accessed table, if the total number of
    live tuples and dead tuples is within an acceptable
    limit (1000), and table size is not too big (50MB), it
```

2. Summary Index Construction

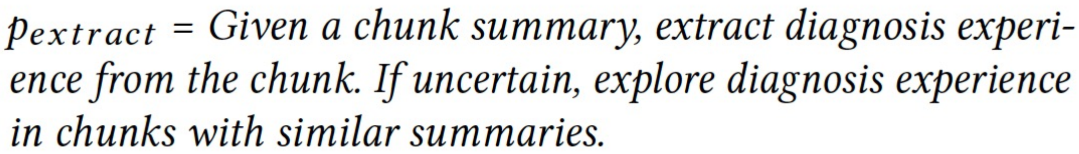
- The summary of a text block includes: 1) the content of the text block; 2) titles and summaries of child nodes



$p_{summarize}$ = *Summarize the provided chunk briefly ··· Your summary will serve as an index for others to find technical details related to database maintenance ··· Pay attention to examples even if the chunks covers other topics.*

3. Maintenance Experience Extraction

- LLM parses the content of each text block and compares it with the summaries of other similar blocks.



$p_{extract}$ = *Given a chunk summary, extract diagnosis experience from the chunk. If uncertain, explore diagnosis experience in chunks with similar summaries.*

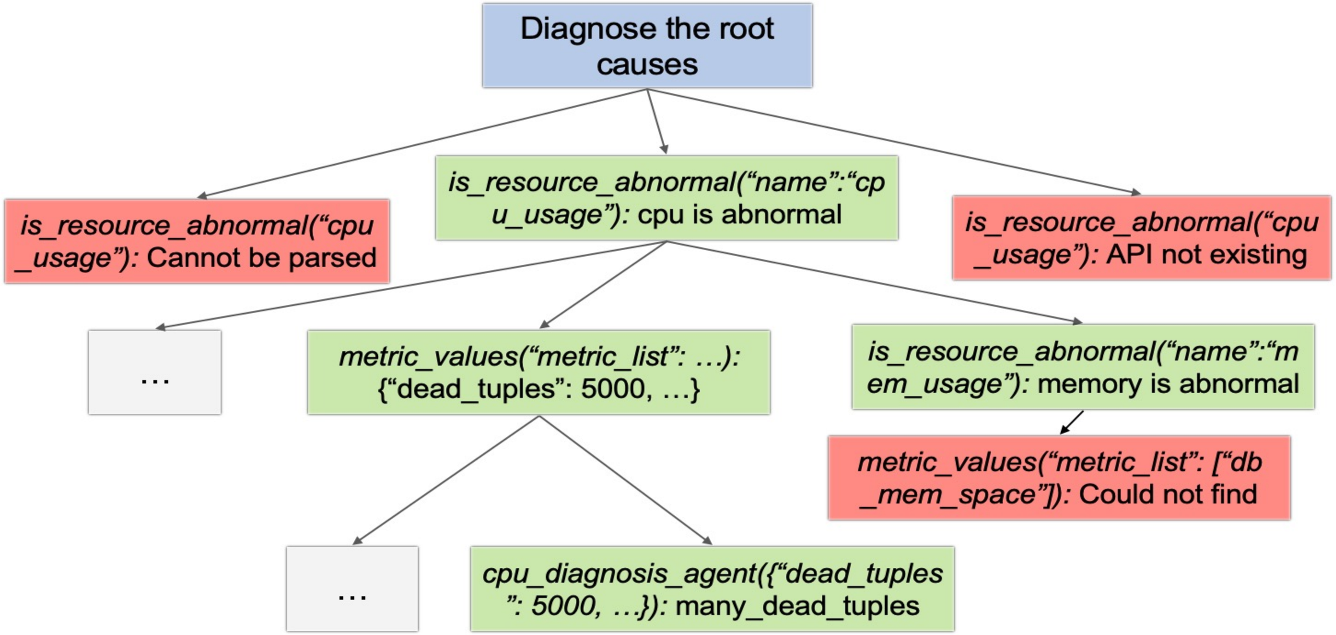
Technology 3: Tree-search-based Tool Calling

No tool usage / premature termination / incorrect APIs

1. Tool Retrival: BM25 / LLM Embedding / Dense Retrival.

2. Tree Search Algorithm

- **Initialization:** Root node - Diag request; Value calculation: $\mathrm{UCT}(v) = \frac{w(v)}{n(v)} + C \cdot \sqrt{\frac{\ln(N)}{n(v)}}$
- **Simulation:** E.g., from the current node, exploring child nodes that can yield new abnormal metrics.
- **Reflection:** If LLM determines the current path cannot provide useful information, decrease the UCT value (e.g., transitioning to the value of the parent node)



Diagnose the root causes

is_resource_abnormal(“cpu_usage”): Cannot be parsed

is_resource_abnormal(“name”:“cpu_usage”): cpu is abnormal

is_resource_abnormal(“cpu_usage”): API not existing

…

metric_values(“metric_list”: …): {“dead_tuples”: 5000, …}

is_resource_abnormal(“name”:“mem_usage”): memory is abnormal

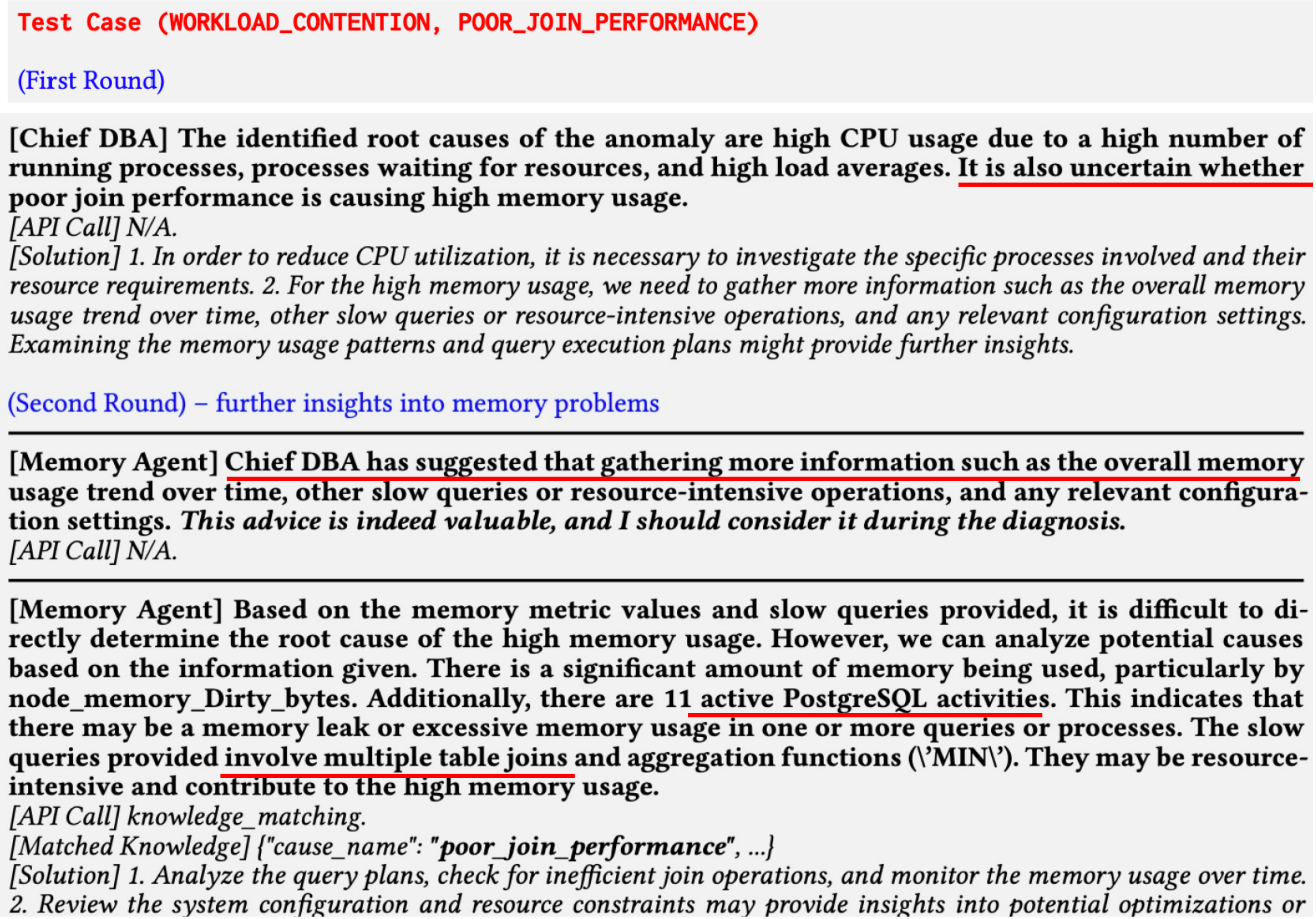
metric_values(“metric_list”: [“db_mem_space”]): Could not find

…

cpu_diagnosis_agent({“dead_tuples”: 5000, …}): many_dead_tuples

Technology 4: Multi-Model Consultation Mechanism

Chat mechanism is crucial for complex problems（1 LLM vs 3 LLMs)



`Test Case (WORKLOAD_CONTENTION, POOR_JOIN_PERFORMANCE)`

(First Round)

[Chief DBA] The identified root causes of the anomaly are high CPU usage due to a high number of running processes, processes waiting for resources, and high load averages. It is also uncertain whether poor join performance is causing high memory usage.
[API Call] N/A.
[Solution] 1. In order to reduce CPU utilization, it is necessary to investigate the specific processes involved and their resource requirements. 2. For the high memory usage, we need to gather more information such as the overall memory usage trend over time, other slow queries or resource-intensive operations, and any relevant configuration settings. Examining the memory usage patterns and query execution plans might provide further insights.

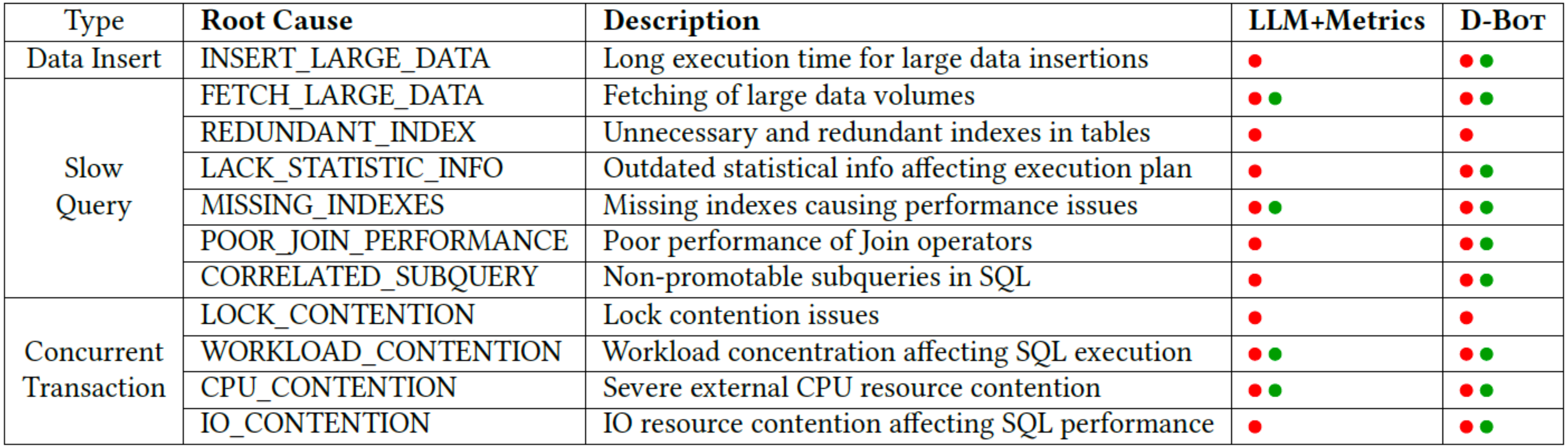
(Second Round) – further insights into memory problems

[Memory Agent] Chief DBA has suggested that gathering more information such as the overall memory usage trend over time, other slow queries or resource-intensive operations, and any relevant configuration settings. *This advice is indeed valuable, and I should consider it during the diagnosis.*
[API Call] N/A.

[Memory Agent] Based on the memory metric values and slow queries provided, it is difficult to directly determine the root cause of the high memory usage. However, we can analyze potential causes based on the information given. There is a significant amount of memory being used, particularly by node_memory_Dirty_bytes. Additionally, there are 11 active PostgreSQL activities. This indicates that there may be a memory leak or excessive memory usage in one or more queries or processes. The slow queries provided involve multiple table joins and aggregation functions (\'MIN\'). They may be resource-intensive and contribute to the high memory usage.
[API Call] knowledge_matching.
*[Matched Knowledge] {"cause_name": "**poor_join_performance**", ...}*
[Solution] 1. Analyze the query plans, check for inefficient join operations, and monitor the memory usage over time. 2. Review the system configuration and resource constraints may provide insights into potential optimizations or

Experimental Performance

- **Anomalies: 11 single-cause anomalies + some multi-root-cause anomalies**
- **Baseline**： vanilla GPT-4 + relevant monitoring metrics
- Result Comparison: Achieve higher accuracy than baseline
  - Enhanced reasoning ability → explore finer-grained system performance metrics/views .
  - Utilize domain knowledge → avoid situations of “fabricated root causes”.
  - Learn to use tools → provide practical optimization suggestions.



Type	**Root Cause**	**Description**	**LLM+Metrics**	**D-Bot**
Data Insert	INSERT_LARGE_DATA	Long execution time for large data insertions	🔴	🔴🟢
Slow Query	FETCH_LARGE_DATA	Fetching of large data volumes	🔴🟢	🔴🟢
	REDUNDANT_INDEX	Unnecessary and redundant indexes in tables	🔴	🔴
	LACK_STATISTIC_INFO	Outdated statistical info affecting execution plan	🔴	🔴🟢
	MISSING_INDEXES	Missing indexes causing performance issues	🔴🟢	🔴🟢
	POOR_JOIN_PERFORMANCE	Poor performance of Join operators	🔴	🔴🟢
	CORRELATED_SUBQUERY	Non-promotable subqueries in SQL	🔴	🔴🟢
Concurrent Transaction	LOCK_CONTENTION	Lock contention issues	🔴	🔴
	WORKLOAD_CONTENTION	Workload concentration affecting SQL execution	🔴🟢	🔴🟢
	CPU_CONTENTION	Severe external CPU resource contention	🔴🟢	🔴🟢
	IO_CONTENTION	IO resource contention affecting SQL performance	🔴	🔴🟢

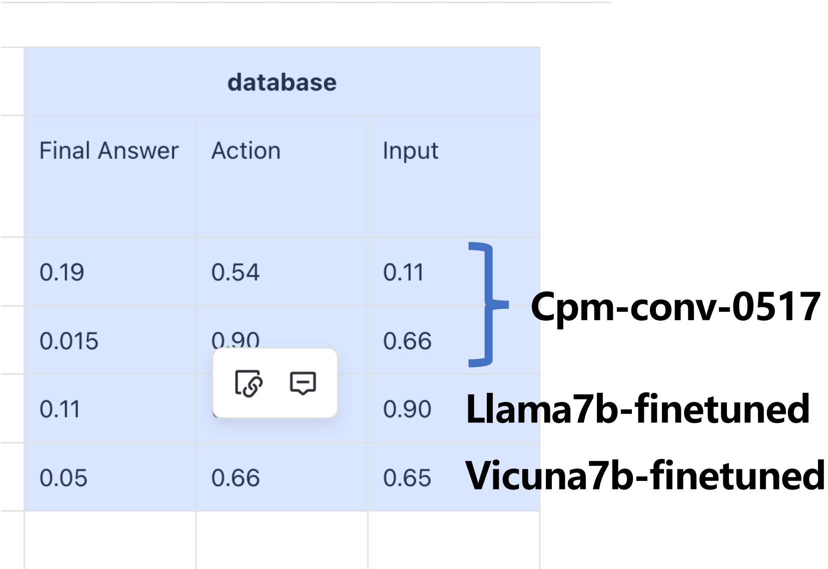
(🔴 : legal diagnosis results; 🟢 : accurate diagnosis results)

Experimental Performance

- **Data Insert**
  - **INSERT_LARGE_DATA**
  - **UPDATE_LARGE_TABLE**
  - **DELETE_LARGE_DATA**
- **Slow Query**
  - **FETCH_LARGE_DATA**
  - **REDUNDANT_INDEX**
  - **DISK_SPILL**
  - **LACK_STATISTIC_INFO**
  - **MISSING_INDEXES**
  - **POOR_JOIN_PERFORMANCE**
  - **STRING_MATCHING**
  - **COMPLEX_EXECUTION_PLAN**
  - **CORRELATED_SUBQUERY**
  - **POOR_AGGREGATION_PERFORMANCE**
- **Conccurent Transactions**
  - **LOCK_CONTENTION**
  - **WORKLOAD_CONTENTION**
  - **CPU_RESOURCE_CONTENTION**
  - **IO_RESOURCE_CONTENTION**
  - **TIMED_TASK_CONFLICT**

Training Local Private Models (todo)

- **Training Data: Document manuals + Diagnostic data**
  - Document Manuals: Over 80 pages of operational guide
  - Diagnostic Data: 4 script categories * Hyperparameters (read/write ratio, concurrency, etc.) * GPT-4 diagnostic cases
- **Open-source model fine-tuning: E.g., Llama2 (language comprehension), Vicuna (code comprehension), chatcpm (Chinese comprehension)"**



database			
Final Answer	Action	Input	
0.19	0.54	0.11	**Cpm-conv-0517**
0.015	0.90	0.66	
0.11	[illegible]	0.90	**Llama7b-finetuned**
0.05	0.66	0.65	**Vicuna7b-finetuned**

RougeL: Evaluate the similarity between two text

ROUGE-L = (LCS / R) * (LCS / C) * ((1 + beta^2) / (beta^2 * R + C))

- **LCS**: Length of the **longest common subsequence** between the candidate summary and reference text.
- **R**: **Total number of words** in the reference text.
- **C**: **Total number of words** in the candidate text.
- **beta**: A parameter that controls the importance of **precision** in the F1 score calculation. Typically, beta^2 is set to 1.