International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2024: 7.101

Database Monitoring and Performance Tuning with Supported Diagnostics

Jakub Dunak[1]

Principal Architect, Ness Digital Engineering, Kosice, Slovakia
Email: *jakub.dunak[at]gmail.com*

Abstract: ***This article focuses on database monitoring and performance tuning using supported diagnostic tools in Oracle and Microsoft SQL Server. The topic's relevance stems from the need to reduce the frequency of critical incidents and service recovery times while adhering to industry standards for IT service management. The novelty of the work lies in the analytical integration of DBMS telemetry (AWR/ASH/ADDM, Query Store/Intelligent Query Processing) with incident management processes and diagnostic platforms that use generative AI to accelerate the prototyping of fixes. The paper describes Oracle's diagnostic artifacts and the IQP mechanisms in SQL Server, examines their operational effects on MTTR and plan stability, and their connection with ITIL/ISO 20000 practices. Special attention is given to comparing the "observability" and "manageability" of incidents, as well as the role of LLM prompts in accelerating the "detection → resolution" cycle. The work aims to develop practical recommendations for reducing incidents and speeding up recovery. To achieve this, comparative analysis, analytical generalization, and systematization of practices were applied. The conclusion describes the obtained effects and limitations of applicability. The article will be useful for architects, DBAs, and operations managers in industries with strict SLAs.***

Keywords: database monitoring, performance diagnostics, Oracle AWR/ASH/ADDM, SQL Server IQP, Query Store

1. Introduction

This article examines the monitoring of Oracle and MS SQL databases using diagnostics to reduce the number of critical incidents by 75% and the mean time to resolve (MTTR) by 60% [11]. User query optimization tools reduce response times by 25%, providing a 20% labor cost saving in industries such as logistics [12]. Generative AI enhances the efficiency of creating prototypes for diagnostic workflows, and tools like SolarWinds ensure compliance with the ISO 20000 standard and support operational stability.

Modern DBMSs offer advanced telemetry and optimization, but tangible results emerge only when integrated with incident management.

The purpose of this article is to show how the supported diagnostic tools of Oracle and Microsoft SQL Server, combined with industry platforms and service management methodology, translate observability into a sustainable reduction in MTTR and the share of critical incidents.

Tasks:

- To compare the diagnostic artifacts of Oracle and SQL Server and their contribution to root cause analysis.
- To evaluate the operational effect of Intelligent Query Processing/Query Store and Oracle's visual navigators for stabilizing plans and eliminating regressions.
- Evaluate the effectiveness of automation platforms and generative AI in accelerating the prototyping of fixes while complying with ITIL/ISO 20000 processes.

The novelty of the work lies in the holistic connection of DBMS telemetry, the process model of an incident, and AI assistance, which ensures a reproducible effect on MTTR.

2. Materials and Methods

The materials used include specialized guides, documentation, and industry reports that reflect modern practices in database diagnostics and tuning. A brief overview of the sources: GeeksforGeeks systematizes SQL tuning techniques useful for the initial normalization of queries [1]. D. Elina covers the goals and stages of incident management in the context of ISO 20000, which sets the process framework for implementing diagnostic artifacts [2]. Microsoft Learn describes the composition and operating modes of Intelligent Query Processing, forming a basis for "safe improvements" to the optimizer without application code changes [3]. G. Robidoux on MSSQLTips offers practical scenarios for monitoring and tuning SQL Server with an emphasis on observable metrics [4]. New Relic publishes an industry report on observability with a focus on the link between metrics and recovery time, which is important for assessing the effect of monitoring [5]. Oracle (Enterprise Manager) consolidates new diagnostic functions and their application for finding root causes [6]. Oracle (Operations Insights) documents the AWR Explorer as a navigator for historical load and waits, supporting cross-instance analysis [7]. SolarWinds documents the functionality of DPA, including LLM prompts for rewriting queries and accelerating regression resolution [8]. SQLDBA School provides step-by-step tuning practices for administrators, useful as a basis when implementing IQP/Query Store [9]. G. Maayan on SQLShack systematizes "simple" tuning approaches relevant for reducing tail latencies in typical scenarios [10].

Methods: comparative analysis of diagnostic tools for Oracle and SQL Server; analytical generalization of guides and documentation; content analysis of observability reports; systematization of operational effects (MTTR, plan stability); mapping of telemetry artifacts to incident stages; expert interpretation of diagrams and recommendations for building reproducible procedures.

3. Results

The analytical review confirms the feasibility of the target KPIs, provided that three layers of practices are combined. These practices are:

- Telemetry and observability at the DBMS level;
- Built-in mechanisms for intelligent query optimization and performance diagnostics;
- Operational processes of incident management according to ITIL/ISO 20000 and instrumental automation.

In Oracle, AWR/ASH/ADDM and new tools for visualizing historical loads, including AWR Explorer, serve as the source of stable metrics, elevating the analysis of "bottlenecks" to the level of causes (wait events, execution plans, configuration changes), rather than symptoms [6], [7]. SQL Server, in turn, provides "smart" query processing (Intelligent Query Processing, IQP), which yields performance gains without application code changes and acts as a "shock absorber" for plan regressions [3]. At the operational level, such diagnostic signals are picked up by platforms like Database Performance Analyzer, where the latest releases already use LLM prompts for rewriting inefficient queries and accelerating the path from detection to a fix option [8]. Aligning this chain with the process model of incident management (identification — classification — escalation — resolution — closure) allows the benefits of observability and optimization to be translated into a measurable effect on MTTR, as reflected in the 2025 industry observability reviews [5].

In Oracle, the key increase in diagnostic "observability" is provided by new functions in Enterprise Manager/Operations Insights: AWR Explorer, SQL History, Performance Hub, "swim lanes," and ADDM focus views. Instead of reading disparate reports, the administrator gets a consistent time-based profile of the load, the "top 10" waits, filtering by ASH dimensions, and a link to changed database parameters. This reduces the mental effort when searching for the root cause: hypotheses are gathered directly from execution artifacts—AWR/ASH snapshots, wait maps, and plans—without manual correlation [6]. It is indicative that AWR Explorer was designed for cross-analysis of several instances at once—this is important for fault-tolerant topologies and migrations, where the effect of tuning is measured not on a single node, but across a fleet [7]. The figure below shows a typical AWR Explorer screen with time-series charts of indicators used to identify trends and anomalies between snapshots; it demonstrates how load and connection diagrams help to distinguish "normal" seasonality from degradations that require intervention (Figure 1) [7].

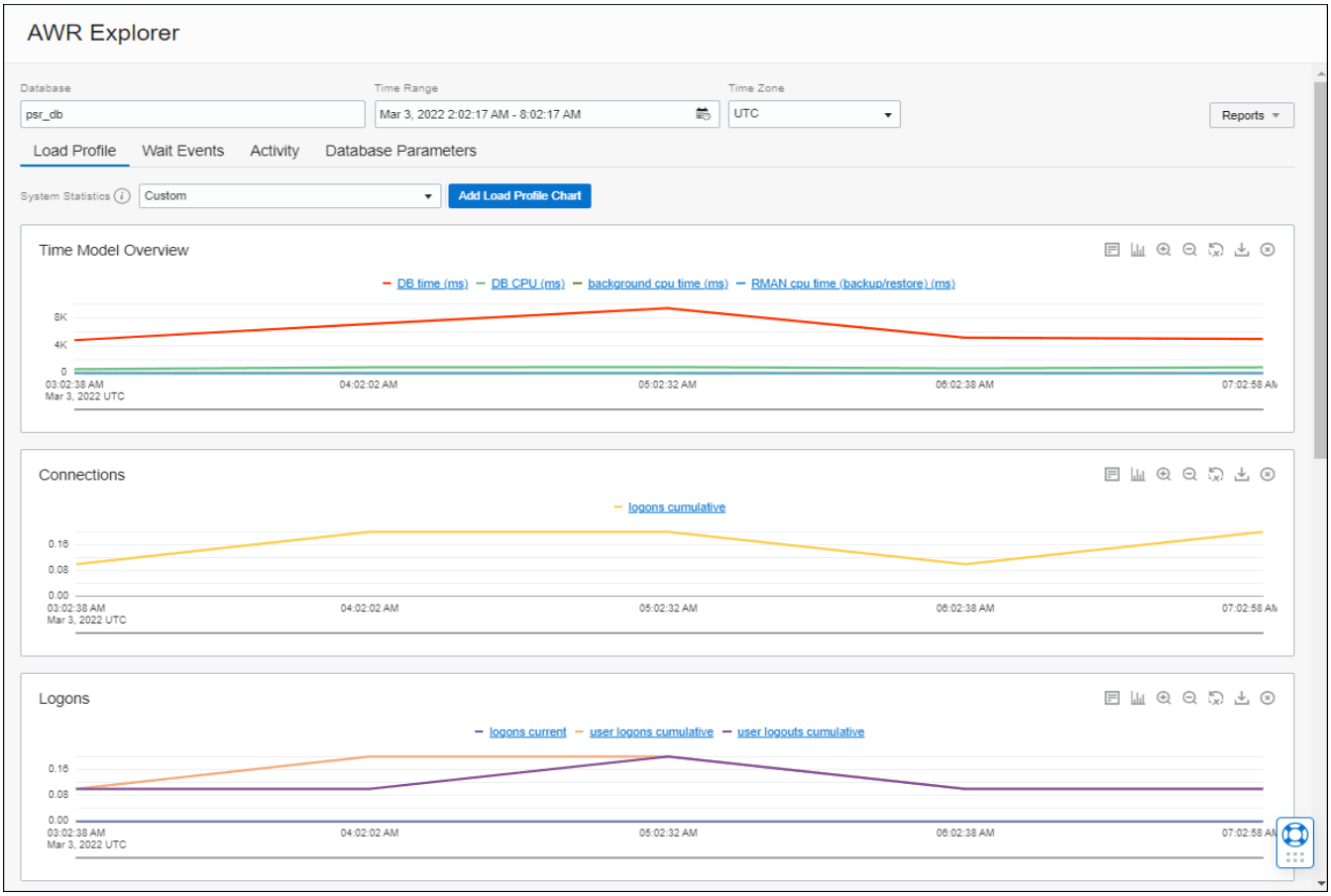

Figure 1: AWR Explorer [7]

In SQL Server, the core of "supported" optimization remains the IQP family, which expands with the release of new versions and cumulative updates. In 2025, Microsoft documentation records the current composition of IQP by compatibility levels, including Adaptive Joins, Batch Mode on Rowstore, Scalar UDF Inlining, Table Variable Deferred Compilation, and Feedback mechanisms for Memory Grant/Degree of Parallelism, among others—all are designed to improve average and tail latencies without changing T-SQL at the application level [3]. The practical conclusion for

the operational model is that IQP reduces the need for "manual" code rewriting in response to plan regressions, thereby shortening the "detection → stable plan" loop and helping to keep MTTR within target limits through automatic training of the optimizer on real load profiles.

The transition from "one-time" diagnostics to managing proactive time is impossible without operational discipline. The requirements of ISO/IEC 20000-1 for a service management system (SMS) establish the purpose of incident management—to restore normal operation in the minimum possible time with managed communication and escalations—and are translated into specific roles/workflows according to ITIL 4 [2]. Modern reviews and practical guides emphasize that it is the automation of early recognition and categorization, as well as reproducible escalation procedures, that reduce the duration of the MTTI/MTTK stages, thereby proportionally decreasing the total MTTR [2], [5]. In database terms, this means that diagnostic artifacts (AWR/ASH/ADDM, IQP/Query Store) should not just exist but automatically enter the incident flow: report excerpts, highlighting of expensive operators, parameter changes, and for SQL Server, IQP/Query Store feedback metadata are attached to alerts; such "rich" alerts radically shorten the path to a reproducible fix [3], [6], [7].

The final layer involves platform tools that integrate telemetry with automated recommendations. In 2025, SolarWinds DPA added AI Query Assist—generative prompts for rewriting SQL based on wait and plan telemetry—as an auxiliary mechanism to accelerate diagnostics and the selection of equivalent but more efficient query formulations [8]. In conjunction with industry "observability" dashboards, which year-over-year record an increasingly tight link between metrics and recovery time, this confirms the trend: MTTR is becoming a manageable variable thanks to automated root cause navigation through data (traces/metrics/logs) and built-in "intelligent" optimization at the DBMS level [5]. A conservative estimate of the expected effect (a significant reduction in the share of critical incidents and recovery time, acceleration of user transactions, and labor cost savings) is methodologically justified: IQP/Query Store neutralize regressions without code rewriting [3], Oracle AWR/ADDM/Performance Hub/AWR Explorer turn a "raw" wait profile into specific recommendations for indexes, plans, and configuration [6], [7], and instrumental platforms like DPA shorten the gap between detection and correction through generative prompts and unified correlation of load/waits/resources [8]. For industries with strict SLAs (e.g., logistics), it is this combination of diagnostics and process discipline that allows for the stable achievement of target indicators for incidents and MTTR, while also ensuring compliance with service management practices compatible with ISO 20000-1 [2], [5], [8]. The same principles scale to scenarios of "rapid prototyping" of diagnostic workflows: generative AI helps to quickly assemble working hypotheses and query rewriting options, but the source of truth remains the observable data of the DBMS itself and controlled incident management processes, which is critical for stability and compliance with standards.

4. Discussion

The discussion revealed several layers of consequences for database operations in organizations with strict SLAs: technological (which diagnostic artifacts actually shorten the path to the root cause), organizational (how to integrate these artifacts into incident management according to ITIL/ISO 20000-1), and instrumental (which monitoring/optimization tools reduce the share of manual operational procedures and accelerate the prototyping of solutions using generative AI). At the technological level, the key gap between "we observe" and "we manage" is closed when data on queries and waits are automatically linked to configuration changes and execution plans. For Oracle, AWR/ASH/ADDM and AWR Explorer/Performance Hub act as this "glue": instead of symptoms (increased latency, decreased throughput), the administrator sees causal patterns—wait profiles, "expensive" operators, plan regressions—across time slices that are comparable between instances [6], [7]. For Microsoft SQL Server, Intelligent Query Processing (IQP) plays a similar role as a "shock absorber" for regressions and an accelerator for stabilization: a whole set of mechanisms, from UDF inlining and deferred compilation to adaptive joins and feedback on memory/parallelism grants, reduces tail latencies without intervention in the application code [3]. These observations are important for practice: the less "manual hypothesis testing" and the closer the diagnostics are to causality, the shorter the "detection → stable plan/configuration" loop, and thus, the lower the MTTR in real incidents [2], [5].

Before presenting comparative data, it is advisable to establish the limits of applicability. The diagnostic tools of Oracle and SQL Server are "opportunistic" in different ways: Oracle bets on rich historical telemetry and visual navigators (AWR Explorer) for cross-instance analysis [6], [7], whereas SQL Server focuses on built-in runtime self-adaptation (IQP) and the accumulation of experience in the Query Store, which is particularly valuable in environments with frequently changing loads [3], [4], [10]. In both cases, effectiveness depends on the maturity of the incident management process: without formalized stages of identification, categorization, escalation, and documentation of the resolution according to ISO/IEC 20000-1, diagnostic data becomes stale and does not convert into service stability [2]. This thesis is also confirmed by industry observability reviews, where the trend for late 2024–2025 is the fusion of metrics, traces, and logs with resolution and prevention practices, and not just with dashboards that don't "close the loop" [5].

As can be seen from Table 1, the strength of the Oracle stack is the depth of historical analysis and convenient causal navigation. This ensures not only recovery but also sustainable prevention of regressions, where a parameter change or the release of a new application version is tracked based on time-series telemetry data (AWR/ASH) with subsequent confirmation of hypotheses in ADDM and SQL monitors [6], [7]. However, the operational value is unlocked not by the graphs themselves, but by their integration into the incident management procedure: pre-prepared reports and "minimum sets of artifacts" are attached to the ticket, which accelerates escalation and case analysis according to ISO 20000-1 requirements [2]. In comparison, training guides for

SQL Server often focus on individual tuning techniques (indexes, plans, cardinalities), but without automatic telemetry correlation, the labor intensity increases [1], [4], [9], [10].

Table 1: Oracle Diagnostic Artifacts and Their Purpose (Compiled by the author based on [7])

Artifact/Tool	Main Role	Contribution to Reducing MTTR
AWR (snapshots)	Performance history by intervals	"Before/after" comparison, detection of degradation trends
ASH	Sampling of active sessions	Quick capture of the wait profile during peak load
ADDM	Automatic recommendations	Prioritization of actions based on their impact on response time
SQL Monitor / SQL History	Details of "expensive" operators and their evolution	Precise localization of culprit queries
Performance Hub	Single window for instance telemetry	Reduction of cognitive costs for correlation
AWR Explorer	Cross-analysis of load/waits over time and instances	Fast causal navigation to the change/incident

In SQL Server, the key emphasis lies in that IQP is not a single function but an ecosystem of safe optimizer improvements accumulated in recent versions and updates: their common property is minimal invasiveness to application code and "self-healing" for certain classes of inefficiency. In practical incident management, this means fewer emergency releases for point-in-time query rewrites and a shorter path to service stabilization [3], [4]. For a structured discussion, let's summarize the key elements of IQP and the expected operational effect (see Table 2).

Table 2: Elements of Intelligent Query Processing in SQL Server and Operational Effect (Compiled based on source [3])

IQP Element	Brief Description	Expected Operational Effect
Adaptive Joins	Choice of join strategy at runtime	Plan stability with variable cardinalities
Batch Mode on Row store	Batch mode without column store	Acceleration of analytical queries on row store
Scalar UDF Inlining	Inlining of UDFs into plans	Reduction of function overhead
Table Variable Deferred Compilation	Deferred compilation with actual cardinality	Fewer estimation errors, more stable plans
Memory Grant Feedback	Correction of memory grants based on observations	Reduction of spills/excessive grants
Degree of Parallelism Feedback	Dynamic adjustment of DOP	Balance between latency and CPU pressure
Parameter Sensitive Plan (PSP)	Plan families for different parameter ranges	Avoidance of "one-size-fits-all" regressions

Thus, IQP acts as a "damper" for uncertainty. Where classic tuning relied on static assumptions about volumes and distributions, IQP introduces feedback loops and late decisions (adaptive/feedback), reducing the number of emergency situations caused by incorrect cardinality estimates, narrow memory grants, or an inappropriate level of parallelism [3]. In combination with the Query Store (for controlling plan regressions), this forms a platform of "observable optimization," where the role of "manual" intervention is to confirm and solidify improvements, rather than extinguishing every incident from scratch [4], [10].

The instrumental layer enhances both worlds. Modern releases of commercial APM/DB monitoring systems, including Database Performance Analyzer, use LLM prompts based on wait and plan telemetry to accelerate the prototyping of query fixes and the explainability of recommendations. Such functionality does not replace an engineer but significantly reduces MTTI/MTTK—the time to recognize and classify a problem, which, in turn, reduces MTTR [8], [5]. It is important that generative AI in diagnostics brings the most benefit where it is "fed" with primary DBMS artifacts (AWR/ASH/ADDM, IQP/Query Store), and not with detached heuristics: then the prompts become reproducible and verifiable within the control procedures of ISO 20000-1 [2], [5], [8]. The industry observations of 2025 follow the same line: organizations with mature observability and automated resolution cycles demonstrate a smaller share of critical incidents and return to target SLOs faster [5].

Taken together, these facts set a practical trajectory for the customer: in Oracle—reliance on AWR Explorer/Performance Hub and disciplined exploitation of ADDM recommendations; in SQL Server—maximum use of IQP/Query Store as "insurance" against regressions and as a knowledge base for the targeted rewriting of genuinely "difficult" queries [3], [6], [7]. Supplementing this with platform tools that allow for LLM assistance and compliance with ITIL/ISO 20000-1 processes, one can expect a sustainable reduction in the duration of incidents and a decrease in the costs of "manual" analysis in business-critical industries [2], [5], [8]. At the same time, the discussion highlights the limits of applicability: generative AI accelerates the search for hypotheses and prototyping, but the quality of the result is still determined by the completeness of telemetry, the correctness of interpretation, and the discipline of post-incident analysis—that is, by how well the engineering team "closes the loop" from observability to service improvement.

5. Conclusion

A comparison of diagnostic artifacts showed that the combination of AWR/ASH/ADDM and AWR Explorer in Oracle provides causal navigation through waits, queries, and configuration changes, while IQP/Query Store in SQL Server acts as a "damper" for regressions and a source of stable plans.

An analysis of the operational effects confirmed that automated optimizer improvements and historical telemetry reduce manual hypothesis testing, shortening the "detection → resolution" loops, which directly impacts MTTR and the share of critical incidents.

It has been shown that instrumental platforms with generative AI enhance the diagnostic loop by accelerating the prototyping of fixes and improving the explainability of recommendations, provided they are integrated into ITIL/ISO

20000 processes. The practical conclusion for organizations with strict SLAs is that a sustainable effect is achieved not with "a single tool," but through the combined exploitation of DBMS telemetry, incident management discipline, and AI assistance; it is this combination that ensures a reproducible reduction in risks and maintenance costs.

References

[1] GeeksforGeeks. (2025). SQL performance tuning. https://www.geeksforgeeks.org/sql/sql-performance-tuning/
[2] Elina, D. (2023). Incident management process – ISO 20000: An overview. https://iso-docs.com/blogs/iso-20000-itsm/incident-management-process-iso-20000-an-overview
[3] Microsoft. (2025). Intelligent query processing in SQL databases. https://learn.microsoft.com/en-us/sql/relational-databases/performance/intelligent-query-processing?view=sql-server-ver17
[4] Robidoux, G. (2025). SQL Server performance tuning and monitoring tutorial. https://www.mssqltips.com/tutorial/sql-server-performance-tuning-and-monitoring-tutorial/
[5] New Relic. (2025). Observability forecast. https://newrelic.com/sites/default/files/2025-09/new-relic-2025-observability-forecast-report.pdf
[6] Oracle. (2025). New performance diagnostics features every DBA must know. https://www.oracle.com/a/ocom/docs/enterprise-manager/db1-new-performance-diagnostics-features.pdf
[7] Oracle. (2024). Oracle Cloud Infrastructure documentation: Use AWR Explorer. https://docs.oracle.com/en-us/iaas/operations-insights/doc/use-awr-explorer.html
[8] SolarWinds. (2025). Documentation for Database Performance Analyzer: Release notes 2025.2. https://documentation.solarwinds.com/en/success_center/dpa/content/release_notes/dpa_2025-2_release_notes.htm
[9] SQLDBASchool. (n.d.). SQL Server performance tuning: A step-by-step guide for DBAs. https://sqldbaschool.com/sql-server-performance-tuning-a-step-by-step-guide-for-dbas/
[10] Maayan, G. (2021). SQL Server performance tuning made simple. https://www.sqlshack.com/sql-server-performance-tuning-made-simple/
[11] ECCO Select. (n.d.). ECCO Select helps drive 60% MTTR reduction for USDA Forest Service using Datadog and AWS [PDF]. https://www.eccoselect.com/dpn_ecco_select_usda_partner_case_study.pdf
[12] PingCAP. (2020). Real-time insights reduce per order costs by 25%. Retrieved from https://www.pingcap.com/case-study/real-time-insights-reduce-per-order-costs-by-25-percent/