

Autonomous Database Administration Using Generative AI and Retrieval-Augmented Generation for Intelligent Enterprise Cloud Systems

The AIDAS Framework: Artificial Intelligence Driven Autonomous Database Administration System

Jipsa Mathew^{1*}; Thomas Philip²

¹M.S., Computer Science Senior Member of Technical Staff, Oracle Corporation,
Dallas, Texas, United States

²M.S., Computer Science Technical Consulting Engineering, Cisco Systems, Richardson,
Dallas, Texas, United States

Corresponding Author: Jipsa Mathew^{1*}

Publication Date: 2026/10/02

Abstract: Enterprise database environments have grown into sprawling, heterogeneous estates spanning on-premises systems, Oracle Cloud Infrastructure, Amazon Web Services, and Microsoft Azure, placing an unsustainable operational burden on human database administrators (DBAs). Traditional automation, rooted in rule-based monitoring and narrow machine learning models, addresses isolated tasks such as index tuning or anomaly scoring, but it cannot reason across the full administrative lifecycle, nor explain its decisions in terms a human operator can trust and act upon. This paper introduces the Artificial Intelligence Driven Autonomous Database Administration System (AIDAS), a novel conceptual framework that unifies generative artificial intelligence, retrieval-augmented generation (RAG), vector search, multi-agent large language model (LLM) orchestration, and the Model Context Protocol (MCP) into a single governed architecture for autonomous enterprise database administration. AIDAS is organized as a seven-layer pipeline spanning telemetry ingestion, enterprise knowledge retrieval, standardized tool access, multi-agent reasoning, governed execution, and continuous institutional-memory capture. Unlike prior LLM-for-database systems, which typically target a single task such as diagnosis or query rewriting, AIDAS is architected to intelligently automate monitoring, incident detection, root cause analysis, infrastructure troubleshooting, performance and query optimization, security monitoring, compliance validation, automated documentation, and decision support as a coherent whole, while remaining portable across heterogeneous cloud database platforms through an MCP-based integration fabric. The paper presents the system architecture, the enterprise RAG pipeline, the multi-agent decision workflow, a formal orchestration algorithm, an implementation strategy, and four enterprise use cases. Because AIDAS is presented as a design-science contribution rather than a deployed and empirically measured system, the performance discussion is framed conceptually against patterns reported in the cited AIOps and LLM-for-database literature, and empirical validation is explicitly identified as future work. Six original contributions are articulated and differentiated from existing enterprise AI database solutions, and the paper concludes with a candid discussion of limitations, including hallucination risk, MCP security exposure, and the governance requirements of autonomous change execution in production environments.

Keywords: *Autonomous Database Administration; Generative Artificial Intelligence; Retrieval-Augmented Generation; Large Language Models; Model Context Protocol; Multi-Agent Systems; AIOps; Cloud Database Management; Vector Search; Enterprise Knowledge Base.*

How to Cite: Jipsa Mathew; Thomas Philip (2026), Autonomous Database Administration Using Generative AI and Retrieval-Augmented Generation for Intelligent Enterprise Cloud Systems. *International Journal of Innovative Science and Research Technology* 11(7), 4013-4031. <https://doi.org/10.38124/ijisrt/26jul1270>

I. INTRODUCTION

Database administration has historically been a discipline of deep specialization, requiring practitioners to internalize the idiosyncrasies of specific database engines, storage subsystems, and workload patterns. Over the past decade, this discipline has been placed under mounting strain by three simultaneous pressures: the proliferation of cloud database platforms (Oracle Cloud Infrastructure, Amazon Web Services, and Microsoft Azure, frequently deployed together in hybrid and multi-cloud topologies), the exponential growth in transaction volume and schema complexity within enterprise information systems, and a persistent shortage of experienced database administrators relative to the scale of modern estates. The consequence is an operational model in which a small number of DBAs are responsible for thousands of database instances, a ratio that traditional tooling was never designed to support.

Artificial intelligence for IT operations (AIOps) emerged in response to this pressure, applying machine learning and deep learning to failure detection, anomaly scoring, and capacity forecasting. However, most AIOps tooling remains narrow: it produces alerts and metrics rather than diagnoses, and it cannot explain, in natural language, why an anomaly occurred or what corrective action is appropriate. The recent emergence of large language models (LLMs) capable of multi-step reasoning, tool invocation, and natural-language explanation has opened a new design space for database administration automation. Systems such as D-Bot and GaussMaster demonstrate that LLMs can read diagnostic documentation, reason over telemetry, and produce grounded diagnostic reports for individual database instances. Yet these systems, together with the broader corpus of LLM-for-AIOps research, remain fragmented: each addresses a single administrative task in isolation, few incorporate a standardized mechanism for tool and cloud-platform integration, and none combine diagnosis, optimization, security monitoring, compliance validation, documentation, and governed execution within one architecture that is portable across heterogeneous cloud database platforms.

This paper addresses that gap by proposing the Artificial Intelligence Driven Autonomous Database Administration System (AIDAS), a novel enterprise architecture that treats database administration as an end-to-end, continuously learning process rather than a collection of point solutions. AIDAS combines four technological pillars. First, generative AI and LLMs provide the reasoning substrate capable of natural-language diagnosis, explanation, and decision justification. Second, retrieval-augmented generation and vector search ground that reasoning in an enterprise's own runbooks, incident history, schema metadata, and compliance policy, mitigating the hallucination risk inherent in relying on a model's parametric knowledge alone. Third, the Model Context Protocol (MCP), an open standard for connecting AI systems to external tools and data sources, provides a uniform integration fabric across Oracle, AWS, and Azure control planes, eliminating the bespoke point-to-point

integrations that have historically made cross-cloud automation brittle. Fourth, a multi-agent orchestration layer decomposes the administrative workload across specialized agents, mirroring the way a real DBA team divides labor between monitoring, diagnosis, tuning, security, and documentation responsibilities, while a governed execution layer enforces human-in-the-loop approval for high-risk actions.

The remainder of this paper is organized as follows. Section 2 articulates the problem statement in greater detail. Section 3 identifies the specific research gap that AIDAS addresses, informed by a critical reading of the AIOps and LLM-for-database literature. Section 4 states the research objectives, and Section 5 enumerates six original contributions. Sections 6 and 7 present related work and a literature review. Section 8 describes the design-science methodology used to develop the framework. Section 9 presents the AIDAS architecture, RAG pipeline, agent workflow, and orchestration algorithm. Sections 10 through 13 discuss implementation strategy, enterprise use cases, performance evaluation, and comparative analysis. Sections 14 through 17 discuss advantages, limitations, future research directions, and conclusions.

➤ Problem Statement

The central operational problem facing enterprise database organizations is a widening gap between the scale of database estates and the human capacity available to administer them. This gap manifests in five concrete ways. First, monitoring fatigue: DBAs are exposed to a continuous stream of alerts from AWR/ASH snapshots, performance counters, and log aggregation systems, a volume that exceeds what a human triage process can absorb without significant delay or alert suppression. Second, root cause analysis (RCA) remains heavily dependent on tacit, individually held expertise; when an experienced DBA departs an organization, the institutional knowledge required to diagnose recurring incident patterns departs with them, a phenomenon documented extensively in the AIOps literature on failure management (Zhang et al., 2025). Third, cross-cloud fragmentation means that troubleshooting practices, tooling, and control-plane APIs differ across Oracle Cloud Infrastructure, AWS, and Azure, so automation built for one platform rarely transfers to another, forcing organizations to maintain parallel, platform-specific tooling. Fourth, compliance and security validation obligations, driven by frameworks such as the EU AI Act and GDPR as well as sector-specific regulation, impose continuous evidentiary burden on DBA teams that is rarely automated beyond periodic manual audit. Fifth, documentation debt accumulates because incident resolution knowledge is rarely captured systematically at the moment it is generated, leaving future incidents to be re-diagnosed largely from first principles.

Point-solution AI tools have begun to address individual facets of this problem. Vendor-native offerings such as Oracle Autonomous Database automate patching, indexing, and workload-aware tuning within the Oracle stack, while AWS Aurora and Azure SQL Database provide

comparable native performance-insight and auto-tuning features within their own platforms. Academic systems such as D-Bot (Zhou et al., 2024) and GaussMaster (Zhou et al., 2025) demonstrate that LLMs, grounded in diagnostic documentation through retrieval, can produce root-cause reports competitive with experienced human DBAs on individual diagnosis tasks. However, none of these systems is designed to operate as a single administrative layer across heterogeneous multi-cloud database estates, nor to unify monitoring, diagnosis, optimization, security, compliance, and documentation under one governed decision-making process. The problem this paper addresses is therefore not the absence of AI techniques applicable to database administration, but the absence of an integrating architecture capable of applying those techniques coherently, safely, and portably at enterprise scale.

➤ Research Gap

A critical reading of the literature surveyed in Sections 6 and 7 reveals four specific gaps that motivate the AIDAS framework.

- Task fragmentation. LLM-for-database research to date addresses individual administrative tasks: D-Bot and GaussMaster focus on diagnosis, RCRank (Ouyang et al., 2025) focuses on root-cause ranking for slow queries, and the learned query rewrite systems of Zhou et al. (2021, 2023) focus narrowly on query optimization. No published architecture integrates these task-specific capabilities into a single decision-making pipeline that

spans the full administrative lifecycle identified in Section 2.

- Absence of a standardized cross-cloud integration layer. Prior LLM-for-database systems are typically built against a single DBMS (commonly PostgreSQL, MySQL, or openGauss) using bespoke tool-calling interfaces. The Model Context Protocol, introduced by Anthropic in November 2024 (Anthropic, 2024), offers a standardized mechanism for connecting AI agents to external systems, but its application to unifying access across heterogeneous cloud database control planes for administrative automation remains largely unexplored in the published database systems literature.
- Governance and human-in-the-loop design specific to database change risk. Multi-agent orchestration frameworks such as AutoGen (Wu et al., 2023) and MetaGPT (Hong et al., 2023) provide general-purpose coordination patterns, but none of the surveyed literature proposes a risk-tiered arbitration mechanism specifically calibrated to the blast radius of database administrative actions, where an incorrect automated change can cause data loss or extended downtime.
- Institutional memory capture. Existing systems produce diagnostic output but rarely close the loop by converting each resolved incident into structured, retrievable knowledge that improves future retrieval-augmented reasoning, leaving the 'tribal knowledge' problem identified in Section 2 largely unaddressed.

Table 1 summarizes this gap analysis in relation to the specific evidence drawn from the literature.

Table 1 Research Gap Analysis

Gap Dimension	Evidence from Literature	How AIDAS Addresses the Gap
Task fragmentation	D-Bot, GaussMaster, and RCRank each target a single administrative task (diagnosis or RCA ranking) (Zhou et al., 2024; Zhou et al., 2025; Ouyang et al., 2025).	Seven-layer architecture unifies monitoring, RCA, optimization, security, compliance, and documentation under one orchestration layer.
Cross-cloud portability	LLM-for-database systems are typically bound to a single DBMS/testbed; MCP standardization is not yet applied to DBA automation.	MCP integration fabric (Section 9.1) abstracts OCI, AWS, and Azure control planes behind a common tool interface.
Risk-tiered governance	General multi-agent frameworks (AutoGen, MetaGPT) provide coordination but not database-specific risk gating.	Decision agent applies a confidence/risk threshold with mandatory human-in-the-loop escalation for high-blast-radius actions (Section 9.3).
Institutional memory capture	AIOps survey literature identifies tacit-knowledge loss as a persistent failure-management challenge (Zhang et al., 2025).	Documentation agent converts every resolved incident into indexed knowledge-base entries, closing the retrieval feedback loop (Section 9.2).

➤ Research Objectives

This paper pursues six research objectives:

- To design a unified, layered enterprise architecture (AIDAS) that automates the full database administration lifecycle rather than a single isolated task.
- To specify how retrieval-augmented generation, vector search, and an enterprise knowledge base can be structured specifically for database administration corpora, including runbooks, incident history, and schema/configuration metadata.

- To define a multi-agent orchestration model, including a formal decision algorithm, that decomposes administrative reasoning across specialized agents while enforcing human oversight proportional to action risk.
- To demonstrate how the Model Context Protocol can serve as a standardized integration fabric enabling one AIDAS deployment to operate across Oracle Cloud, AWS, and Azure without bespoke per-platform engineering.
- To articulate a phased implementation strategy and illustrate the framework through representative enterprise use cases.

- To critically position AIDAS against existing enterprise AI database solutions and vendor-native autonomous database offerings, identifying advantages, limitations, and a concrete agenda for empirical validation.

II. NOVEL CONTRIBUTIONS

This paper makes six original contributions to the enterprise AI and cloud database administration literature.

➤ *The AIDAS Layered Architecture*

Whereas prior work such as D-Bot (Zhou et al., 2024) and GaussMaster (Zhou et al., 2025) is architected around a single diagnostic loop, AIDAS proposes a seven-layer architecture (Section 9.1) that treats monitoring, retrieval-augmented reasoning, tool access, multi-agent decision-making, governed execution, and infrastructure control as distinct but continuously interacting layers. This is the first architecture, to the authors' knowledge, to integrate the full task list identified by enterprise stakeholders (monitoring, incident detection, RCA, troubleshooting, performance and query optimization, security monitoring, compliance validation, documentation, and decision support) within one coherent design.

➤ *MCP-Based Cross-Cloud Integration Fabric*

AIDAS applies the Model Context Protocol (Anthropic, 2024) as a standardized integration layer between the multi-agent reasoning layer and heterogeneous cloud database control planes. This contrasts with vendor-native autonomous database features, which operate only within their own stack (Oracle Autonomous Database within OCI, native tuning within AWS Aurora or Azure SQL Database), and with academic LLM-for-database systems, which are typically bound to a single test DBMS. The MCP fabric is, to the authors' knowledge, a novel application of the protocol to unified multi-cloud database administration.

➤ *Risk-Tiered Multi-Agent Decision Arbitration*

AIDAS introduces a decision-arbitration mechanism, detailed in Algorithm 1 (Section 9.4), that classifies proposed remediation actions by confidence and blast-radius risk and routes only low-risk, high-confidence actions to automated execution, escalating all others to human-in-the-loop review. This extends general-purpose multi-agent orchestration patterns (Wu et al., 2023; Hong et al., 2023) with a governance model calibrated specifically to the operational stakes of production database change management.

➤ *Enterprise RAG Design for Database Administration Corpora*

The AIDAS RAG pipeline (Section 9.2) combines hybrid dense-sparse retrieval, a schema and configuration knowledge graph, and a historical-incident memory store, extending general enterprise RAG design patterns (Gao et al., 2024; Packowski et al., 2024) into a domain-specific configuration tailored to the structure of DBA knowledge: runbooks, alert taxonomies, execution-plan history, and audit policy.

➤ *Closed-Loop Institutional Memory Capture*

A dedicated documentation agent converts every resolved incident into a structured, embedded knowledge-base entry, closing the loop between execution and retrieval. This directly targets the tacit-knowledge attrition problem identified in the AIOps literature (Zhang et al., 2025) and distinguishes AIDAS from systems that generate a diagnostic report but do not systematically feed that report back into the retrieval corpus for future reasoning.

➤ *Cross-Cloud Abstraction Enabling Vendor-Neutral Autonomy*

Because the MCP fabric abstracts cloud-specific control-plane calls behind a common tool interface, a single AIDAS deployment can, in principle, administer Oracle, AWS, and Azure database instances side by side, in contrast to vendor-native self-driving features that are confined to their own cloud. This positions AIDAS as a vendor-neutral layer above, rather than a replacement for, existing cloud-native automation.

➤ *Related Works*

Related work relevant to AIDAS falls into six clusters: rule-based and machine-learning AIOps, LLM-based database diagnosis systems, retrieval-augmented generation architectures, multi-agent LLM orchestration, tool-integration standards, and vendor-native autonomous database offerings.

• *Rule-Based and Machine-Learning AIOps*

AIOps, a term introduced by Gartner in 2016 (Prasad & Rich, 2018), applies machine learning and deep learning to operational telemetry for failure detection and diagnosis. Foundational self-driving database research, exemplified by the NoisePage system (Pavlo et al., 2021), demonstrated that a DBMS could predict its own workload behavior and adjust configuration without human guidance, using classical ML rather than generative reasoning. Query-tuning research such as QTune (Li et al., 2019) applied deep reinforcement learning to workload-aware configuration, and DBMind (Zhou et al., 2021) packaged multiple learned tuning components into a self-driving platform for openGauss. These systems established that autonomous database operation was technically feasible but remained constrained by the narrow, task-specific scope of individual learned models and by limited natural-language explainability.

• *LLM-Based Database Diagnosis Systems*

The emergence of instruction-following LLMs (Ouyang et al., 2022) capable of chain-of-thought reasoning (Wei et al., 2022) and tool invocation (Yao et al., 2023; Schick et al., 2023) enabled a new generation of database administration copilots. Zhou et al. (2023) proposed framing the DBA role itself as an LLM task ('LLM as DBA'), and D-Bot (Zhou et al., 2024) operationalized this idea with a tree-search reasoning method and collaborative multi-agent architecture that automatically acquires knowledge from diagnosis documents to produce well-founded root-cause reports within acceptable response times. GaussMaster (Zhou et al., 2025) extended this line of work into a broader LLM-based database copilot system. RCRank (Ouyang et

al., 2025) addressed a complementary problem, formulating root-cause ranking for slow queries as a multimodal machine-learning problem over query statements, execution plans, logs, and key performance indicators. Chat2Data (Zhao et al., 2024) combined RAG, vector databases, and LLMs into an interactive data-analysis system. Across this cluster, the broader AIOps-for-LLM survey by Zhang et al. (2025), which analyzed 183 papers published between 2020 and 2024, documents a clear trend: LLM techniques are being applied task-by-task to individual AIOps problems (log parsing, root cause analysis, incident response), with comparatively little work on integrating these capabilities into a single administrative system.

- *Retrieval-Augmented Generation Architectures*

Retrieval-augmented generation was introduced by Lewis et al. (2020) as a mechanism for grounding language model generation in retrieved documents, addressing the parametric-knowledge limitations of models such as those built on the Transformer architecture (Vaswani et al., 2017) and BERT-style encoders (Devlin et al., 2019). Subsequent work refined retrieval quality through dense passage retrieval (Karpukhin et al., 2020) and sentence-level embeddings (Reimers & Gurevych, 2019), and efficient similarity search at scale through approximate nearest-neighbor libraries such as FAISS (Johnson et al., 2021). More recent architectural variants, including graph-structured retrieval (Edge et al., 2024), recursive tree-organized retrieval (Sarathi et al., 2024), and lightweight hybrid retrieval (Guo et al., 2024), extend RAG beyond flat document chunking toward structure-aware retrieval, a direction highly relevant to database administration corpora, which are inherently hierarchical (schemas, tables, columns, configuration parameters). Packowski et al. (2024) examined enterprise RAG specifically from a content-design perspective, highlighting that retrieval quality in production settings depends as much on corpus curation as on model architecture, a finding this paper adopts directly in the design of the AIDAS knowledge layer (Section 9.2).

- *Multi-Agent LLM Orchestration*

Multi-agent frameworks such as AutoGen (Wu et al., 2023) and MetaGPT (Hong et al., 2023) demonstrated that decomposing complex tasks across specialized, communicating LLM agents improves reliability relative to single-agent prompting, particularly for software-engineering-style workflows. Survey work (Li et al., 2024) catalogs the rapidly growing space of coordination topologies, and more recent optimization-oriented approaches, including graph-based agent optimization (Zhuge et al., 2024) and automated agentic-system design (Hu et al., 2024), attempt to learn or search over orchestration structure rather than hand-designing it. Cost-aware routing research such as RouteLLM (Ong et al., 2025) addresses the practical concern that not every sub-task requires the most capable and expensive model, a consideration directly relevant to the economics of running AIDAS at enterprise scale (Section 10). None of this literature, however, addresses the specific governance requirements of production database change management,

which is the gap AIDAS's decision-arbitration mechanism targets (Section 5.3).

- *Tool-Integration Standards: The Model Context Protocol*

The Model Context Protocol, open-sourced by Anthropic in November 2024 (Anthropic, 2024), standardizes how LLM applications connect to external data sources and tools through a client-server architecture built on JSON-RPC. Within roughly a year of release, MCP achieved broad industry adoption, including donation of the protocol to the Linux Foundation's Agentic AI Foundation, and has been used to connect AI agents to systems ranging from developer tools to enterprise SaaS platforms. Security researchers have also documented meaningful risks in early MCP deployments, including under-authenticated servers and over-permissioned tool access (Descope, 2026), a concern this paper addresses directly in the AIDAS security design (Section 9.1 and Table 4). To the authors' knowledge, MCP has not previously been proposed as the integration fabric for a unified, cross-cloud database administration architecture.

- *Vendor-Native Autonomous Database Offerings*

Cloud providers have shipped increasingly sophisticated self-driving features within their own platforms. Oracle Autonomous Database automates patching, upgrading, scaling, and workload-aware tuning without administrator involvement (Oracle Corporation, 2023; Helskyaho et al., 2021), AWS Aurora offers machine-learning-based performance insights and optimization recommendations (Amazon Web Services, 2024), and Azure SQL Database provides automatic tuning based on query plan telemetry (Microsoft Azure, 2024). These offerings are mature and effective within their respective clouds, but they are, by design, vendor-locked: they cannot reason across a hybrid estate that spans Oracle, AWS, and Azure simultaneously, nor do they expose a natural-language, explainable decision-support interface to the human DBA. AIDAS is positioned as a vendor-neutral orchestration layer that can sit above such native automation, consuming its telemetry and, where appropriate, invoking its tuning actions through MCP-standardized tool calls, rather than as a replacement for it.

III. LITERATURE REVIEW

This section synthesizes the clusters introduced in Section 6 into a direct, critical comparison of existing enterprise AI database solutions, culminating in Table 1. Three observations emerge from this synthesis. First, task-specific LLM-for-database systems achieve strong results within their narrow scope: D-Bot's tree-search reasoning and multi-agent collaboration produce diagnosis reports comparable to experienced DBAs (Zhou et al., 2024), and RCRank's multimodal ranking improves root-cause identification for slow queries beyond single-modality baselines (Ouyang et al., 2025). These results validate the underlying premise of AIDAS, namely that LLMs grounded in retrieval can perform DBA-grade reasoning, but they do not by themselves constitute an enterprise administration

system, because each addresses only one stage of the administrative lifecycle described in Section 2.

Second, RAG architecture research has matured considerably beyond flat retrieval, with graph-structured (Edge et al., 2024) and recursive tree-organized (Sarathi et al., 2024) variants demonstrating improved multi-hop reasoning. Database administration knowledge is naturally hierarchical and relational (a slow query relates to an execution plan, which relates to a table, which relates to an index, which relates to a configuration parameter), suggesting that structure-aware retrieval is particularly well suited to this domain, a design choice incorporated into the AIDAS knowledge layer through its schema and configuration knowledge graph (Section 9.2).

Third, the multi-agent orchestration and MCP literatures have developed largely independently of the database-systems literature. Multi-agent frameworks provide general coordination primitives (Wu et al., 2023; Hong et al., 2023) that have not yet been specialized for the risk profile of database change management, and MCP has been adopted rapidly across developer tools and enterprise SaaS integrations (Anthropic, 2024) but not, to the authors' knowledge, formalized as the integration layer for a multi-cloud DBA automation architecture. This convergence gap, more than any single missing technique, is the central motivation for AIDAS.

Table 2 Comparison of Existing Enterprise AI Database Solutions

Dimension	D-Bot / GaussMaster	RCRank	Oracle Autonomous DB	AIDAS (Proposed)
Primary scope	Diagnosis / RCA for a single DBMS	Root-cause ranking for slow queries	Patching, tuning, scaling within OCI	Full DBA lifecycle across clouds
Multi-cloud portability	No (test DBMS-specific)	No	OCI only	Yes, via MCP integration fabric
RAG grounding	Yes (diagnosis documents)	Partial (multimodal signals)	No (rule/ML-based)	Yes (hybrid RAG + knowledge graph)
Multi-agent reasoning	Yes (diagnosis-focused)	No	No	Yes (six specialized agents, Section 9.1)
Standardized tool protocol (MCP)	No	No	No	Yes
Risk-tiered human-in-the-loop	Not formalized	Not formalized	No (fully automatic)	Yes (Algorithm 1)
Security & compliance validation	No	No	Partial (platform security)	Yes (dedicated agent)
Automated documentation / memory capture	Partial (report generation)	No	No	Yes (closed-loop knowledge base)

- *Note: Entries for D-Bot, GaussMaster, RCRank, and Oracle Autonomous Database summarize capabilities as reported in the cited primary sources; the AIDAS column reflects the proposed conceptual design and has not yet been empirically benchmarked (see Sections 12 and 15).*

IV. METHODOLOGY

➤ Research Methodology

This study adopts the Design Science Research (DSR) methodology to guide the development of the Artificial Intelligence-Driven Autonomous Database Administration System (AIDAS). Design Science Research is widely recognized within the information systems discipline as an appropriate research paradigm for creating innovative technological artifacts intended to address complex organizational problems while simultaneously contributing new scientific knowledge. Unlike explanatory research methodologies that primarily seek to validate hypotheses through statistical inference, DSR focuses on the systematic design, construction, refinement, and evaluation of artifacts

capable of solving practical problems while extending theoretical understanding.

The selection of DSR is particularly appropriate for this research because the primary objective is not to evaluate an existing autonomous database administration system, but rather to design and formally specify a novel enterprise framework that integrates Generative Artificial Intelligence (GenAI), Retrieval-Augmented Generation (RAG), multi-agent reasoning, Model Context Protocol (MCP), enterprise knowledge management, and governed automation into a unified architecture for intelligent database administration.

Enterprise database administration represents a socio-technical challenge involving heterogeneous cloud infrastructures, dynamic workloads, operational governance, security constraints, regulatory compliance, and continuous decision making. These characteristics cannot be adequately addressed using purely empirical or observational research approaches because no integrated solution currently exists that satisfies the identified requirements. Consequently, DSR provides the methodological foundation for

constructing an innovative artifact capable of addressing this research gap while remaining grounded in established theoretical knowledge.

Furthermore, Design Science Research has been extensively applied in enterprise systems engineering, decision support systems, artificial intelligence, cloud computing, cybersecurity, and information systems architecture because it supports iterative knowledge creation through artifact design and conceptual evaluation. The methodology therefore aligns closely with the objectives of this study, which seeks to contribute both a practical architectural framework and theoretical insights into autonomous enterprise database administration.

➤ *Justification for Using Design Science Research*

The decision to employ Design Science Research is motivated by three principal considerations.

First, the research addresses an engineering-oriented problem rather than a purely analytical problem. Existing enterprise database management solutions provide fragmented automation capabilities, including workload optimization, query tuning, anomaly detection, and isolated diagnostic assistance. However, the literature review demonstrates that no existing framework integrates intelligent monitoring, enterprise knowledge retrieval, multi-agent reasoning, governed decision making, cross-cloud interoperability, and institutional knowledge preservation into a unified autonomous architecture. Addressing this gap requires the creation of a new artifact rather than the evaluation of existing technologies.

Second, the proposed AIDAS framework represents a conceptual enterprise architecture composed of multiple interacting components rather than a standalone software implementation. The framework specifies architectural layers, information flows, decision mechanisms, governance policies, and interaction models that collectively describe how autonomous database administration can be achieved within heterogeneous enterprise environments. Such architectural contributions are characteristic outputs of Design Science Research.

Third, implementation and large-scale deployment of enterprise database automation systems require substantial organizational resources, production infrastructure, and long-term operational evaluation. Since the objective of the present study is to establish the conceptual and architectural foundations of AIDAS, conceptual validation provides an appropriate level of evaluation consistent with accepted DSR practices. Subsequent prototype implementation and empirical benchmarking constitute natural extensions of the proposed research rather than prerequisites for establishing the framework's scientific contribution.

Accordingly, Design Science Research provides a rigorous methodological foundation for systematically transforming identified organizational problems into a structured architectural solution supported by existing knowledge and validated through theoretical consistency.

➤ *Research Workflow*

The overall research workflow adopted in this study follows a structured sequence of interconnected phases that progressively transform identified enterprise challenges into a complete conceptual framework. Figure X illustrates the research workflow adopted throughout this study.

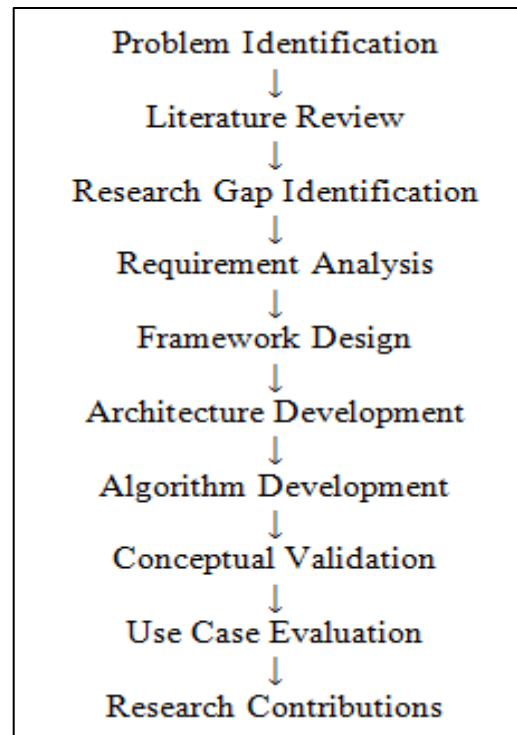


Fig 1 Research Workflow

The workflow begins with problem identification, where the growing complexity of enterprise database administration, increasing operational workloads, cloud heterogeneity, and knowledge management challenges are examined. This stage establishes the practical motivation for autonomous database administration and defines the operational problems requiring intelligent automation.

The second phase involves an extensive literature review covering six major research domains, namely autonomous database management, AIOps, Retrieval-Augmented Generation, Large Language Models, multi-agent systems, and Model Context Protocol. The literature review provides the theoretical foundation for the proposed framework while identifying existing architectural capabilities and technological limitations.

The literature synthesis subsequently enables research gap identification, where four major deficiencies are identified: fragmented task automation, lack of standardized cross-cloud interoperability, insufficient governance mechanisms for autonomous decision making, and inadequate institutional knowledge preservation. These gaps directly define the design objectives of AIDAS.

Based on these identified deficiencies, requirement analysis is conducted to determine the functional and non-functional capabilities expected from an enterprise-scale

autonomous database administration framework. Functional requirements include intelligent monitoring, root cause analysis, query optimization, security monitoring, compliance validation, documentation generation, and automated decision support. Non-functional requirements include explainability, scalability, interoperability, reliability, security, governance, extensibility, and enterprise adaptability.

The identified requirements then guide the framework design phase, during which the conceptual structure of AIDAS is developed. This stage determines the principal architectural components, defines their interactions, and specifies the continuous feedback mechanisms required for adaptive learning.

Following framework design, architecture development translates conceptual requirements into the seven-layer enterprise architecture presented in Section 9. Each architectural layer addresses a specific operational responsibility while maintaining clear communication interfaces with adjacent layers to ensure modularity, scalability, and maintainability.

The subsequent algorithm development phase formalizes autonomous decision making by defining the orchestration logic governing event processing, knowledge retrieval, multi-agent collaboration, risk assessment, automated execution, rollback procedures, and institutional knowledge updates. Formal algorithm specification ensures that governance policies remain explicit and reproducible rather than embedded within narrative descriptions.

Because the framework has not yet been deployed within production environments, conceptual validation is employed to evaluate architectural consistency. Validation is achieved by systematically demonstrating how every architectural component addresses one or more deficiencies identified during the literature review and requirement analysis. This approach establishes logical coherence between research problems, proposed solutions, and expected operational outcomes.

The methodology concludes with use case evaluation, where representative enterprise scenarios illustrate how AIDAS operates across heterogeneous cloud environments. Rather than measuring implementation performance, these scenarios demonstrate the practical applicability, completeness, and internal consistency of the proposed framework.

➤ *Literature-Informed Framework Development*

The proposed AIDAS framework was derived through systematic architectural synthesis of the literature rather than through ad hoc system design. Each architectural component originates from specific theoretical findings identified during the literature review.

The telemetry ingestion layer is informed by research on AIOps and autonomous database monitoring, which

emphasizes continuous collection of operational telemetry as the foundation for intelligent infrastructure management.

The enterprise knowledge layer draws upon Retrieval-Augmented Generation research demonstrating that domain-specific retrieval substantially improves factual consistency while reducing hallucination in large language models. Consequently, AIDAS incorporates hybrid retrieval, enterprise vector databases, historical incident repositories, and structured knowledge graphs to support evidence-based reasoning.

The Model Context Protocol integration layer is derived from recent advances in standardized AI tool interoperability. Instead of implementing proprietary integrations for each cloud provider, MCP establishes a unified abstraction layer capable of exposing heterogeneous cloud services through standardized interfaces.

The multi-agent orchestration layer builds upon contemporary research showing that specialized collaborative agents outperform monolithic LLM architectures for complex reasoning tasks. Accordingly, monitoring, diagnosis, optimization, security, documentation, and governance are assigned to dedicated intelligent agents possessing clearly defined responsibilities.

Finally, the governed execution layer incorporates principles from enterprise IT governance, change management, and AI risk management by ensuring that autonomous actions remain proportional to operational risk through configurable human-in-the-loop decision mechanisms.

This literature-driven synthesis ensures that every architectural decision is theoretically grounded while collectively producing a framework that extends existing research beyond isolated AI-enabled database administration solutions.

➤ *Architectural Synthesis Process*

Architectural synthesis was conducted using a systematic decomposition approach in which enterprise database administration was first analyzed as a collection of operational processes rather than individual software functions.

Initially, major administrative activities—including monitoring, diagnosis, optimization, security management, compliance verification, documentation, and infrastructure control—were identified from existing literature and enterprise operational practices.

These activities were subsequently grouped according to their functional relationships and information dependencies. Components requiring continuous data acquisition were assigned to the telemetry layer, whereas reasoning-intensive functions were allocated to the multi-agent orchestration layer. Enterprise knowledge management responsibilities were consolidated within the

RAG layer, while execution-related activities were isolated within governed execution services.

The resulting layered architecture exhibits high functional cohesion while minimizing coupling between independent services. Such modularization facilitates future extensibility, allowing additional cloud providers, AI agents, retrieval strategies, or governance mechanisms to be incorporated without fundamental architectural redesign.

➤ *Conceptual Validation*

Since AIDAS is presented as a conceptual framework rather than an implemented software platform, conceptual validation constitutes an appropriate evaluation methodology within the Design Science Research paradigm.

Conceptual validation examines whether the proposed artifact adequately satisfies the design objectives established during requirement analysis and whether each architectural component logically contributes to solving the identified research problem.

Rather than reporting experimental performance metrics, conceptual validation evaluates internal consistency, architectural completeness, theoretical grounding, and practical applicability. Specifically, validation is achieved by mapping each framework component to the operational deficiencies identified in the literature, demonstrating traceability between research gaps, design decisions, and expected enterprise capabilities.

The illustrative enterprise use cases presented later in the paper further strengthen conceptual validation by demonstrating how autonomous monitoring, diagnosis, optimization, security management, documentation, and governance interact coherently within realistic enterprise scenarios. Such walkthrough-based validation is widely accepted in design-oriented information systems research when implementation remains outside the scope of the study.

➤ *Reliability and Validity*

The reliability of the proposed framework is supported through the systematic and transparent application of the Design Science Research process. Every stage of framework development—from problem identification to architectural synthesis and conceptual validation—follows a structured methodology grounded in peer-reviewed literature and established enterprise technologies. This systematic process enhances reproducibility by providing explicit documentation of design decisions and ensuring that future researchers can replicate or extend the framework using the same methodological principles.

Validity is established through multiple complementary perspectives. Construct validity is achieved by deriving architectural components directly from well-established concepts in AIOps, Retrieval-Augmented Generation, multi-agent systems, enterprise cloud computing, and autonomous database administration. Content validity is strengthened by ensuring that the

framework addresses all critical operational responsibilities identified during requirement analysis, including monitoring, diagnosis, optimization, security, governance, compliance, documentation, and knowledge management. Internal validity is demonstrated through the logical alignment between identified research gaps, design objectives, architectural components, and expected operational outcomes. Finally, external validity is supported by the framework's vendor-neutral design, which enables adaptation across heterogeneous enterprise cloud environments including Oracle Cloud Infrastructure, Amazon Web Services, Microsoft Azure, hybrid infrastructures, and future cloud platforms implementing standardized MCP interfaces.

Although empirical validation remains outside the scope of this study, the methodological rigor employed throughout the framework development process provides a robust conceptual foundation for subsequent prototype implementation, experimental benchmarking, and industrial deployment. Consequently, the proposed methodology establishes both the scientific credibility of AIDAS and its potential applicability to future enterprise database administration research and practice.

V. PROPOSED AIDAS FRAMEWORK

AIDAS is presented in four parts: the overall system architecture, the enterprise RAG pipeline that grounds agent reasoning, the multi-agent decision workflow, and a formal orchestration algorithm. A fifth subsection maps the architecture onto the ten automation targets identified by enterprise stakeholders (Section 9.5, Figure 4).

➤ *System Architecture*

Figure 1 depicts the AIDAS architecture as seven vertically integrated layers connected by a continuous feedback loop. The Enterprise Database Estate layer represents the heterogeneous population of managed instances: Oracle Database on Oracle Cloud Infrastructure, AWS RDS and Aurora, Azure SQL, and on-premises or hybrid deployments. The Telemetry and Ingestion layer collects AWR/ASH snapshots and performance counters, query and alert logs, audit trails, and streaming metrics from each managed instance, normalizing heterogeneous formats into a common event schema.

The Enterprise RAG and Knowledge layer, detailed further in Section 9.2, maintains the document and runbook corpus, the embedding pipeline, a hybrid dense-sparse vector store, the retriever, historical incident memory, and a schema and configuration knowledge graph. This layer is the primary mechanism by which AIDAS grounds LLM reasoning in enterprise-specific facts rather than relying solely on a model's parametric knowledge, directly mitigating the hallucination risk documented across the LLM literature.

The Model Context Protocol Integration Fabric sits between the knowledge layer and the orchestration layer, exposing a standardized set of MCP tools and resources for

each managed cloud platform: query-execution tools, configuration-read and configuration-write tools (the latter gated by the governance mechanism described below), log and metric query tools, and identity and access management tools. Because each cloud platform's control-plane API is wrapped behind an MCP server exposing a common tool schema, the agents in the orchestration layer above require no cloud-specific logic; adding support for a new cloud platform is a matter of implementing a new MCP server against the existing tool schema rather than re-engineering the reasoning layer.

The Multi-Agent Orchestration Layer comprises six specialized agents: a Monitoring Agent that performs continuous triage and anomaly classification; a Diagnostic/RCA Agent that performs RAG-grounded root-cause analysis; an Optimization Agent responsible for performance and query optimization; a Security and Compliance Agent responsible for security monitoring and

compliance validation; a Documentation Agent responsible for automated documentation and knowledge-base updates; and a Decision and Arbitration Agent that synthesizes the outputs of the other agents and applies the risk-tiered governance policy formalized in Algorithm 1. Below the orchestration layer, the Governed Execution Layer enforces a human-in-the-loop approval gate for actions that exceed the risk threshold, executes approved runbooks automatically for low-risk actions, and maintains rollback and safety controllers capable of reversing an automated action if post-execution verification fails. Finally, the Cloud Infrastructure Control Planes layer represents the actual OCI, AWS, Azure, and hybrid infrastructure being administered. A continuous feedback loop, shown on the right of Figure 2, routes outcomes from the execution and infrastructure layers back into telemetry ingestion and, via the Documentation Agent, into the knowledge layer, so that every administrative action becomes training signal for future reasoning.

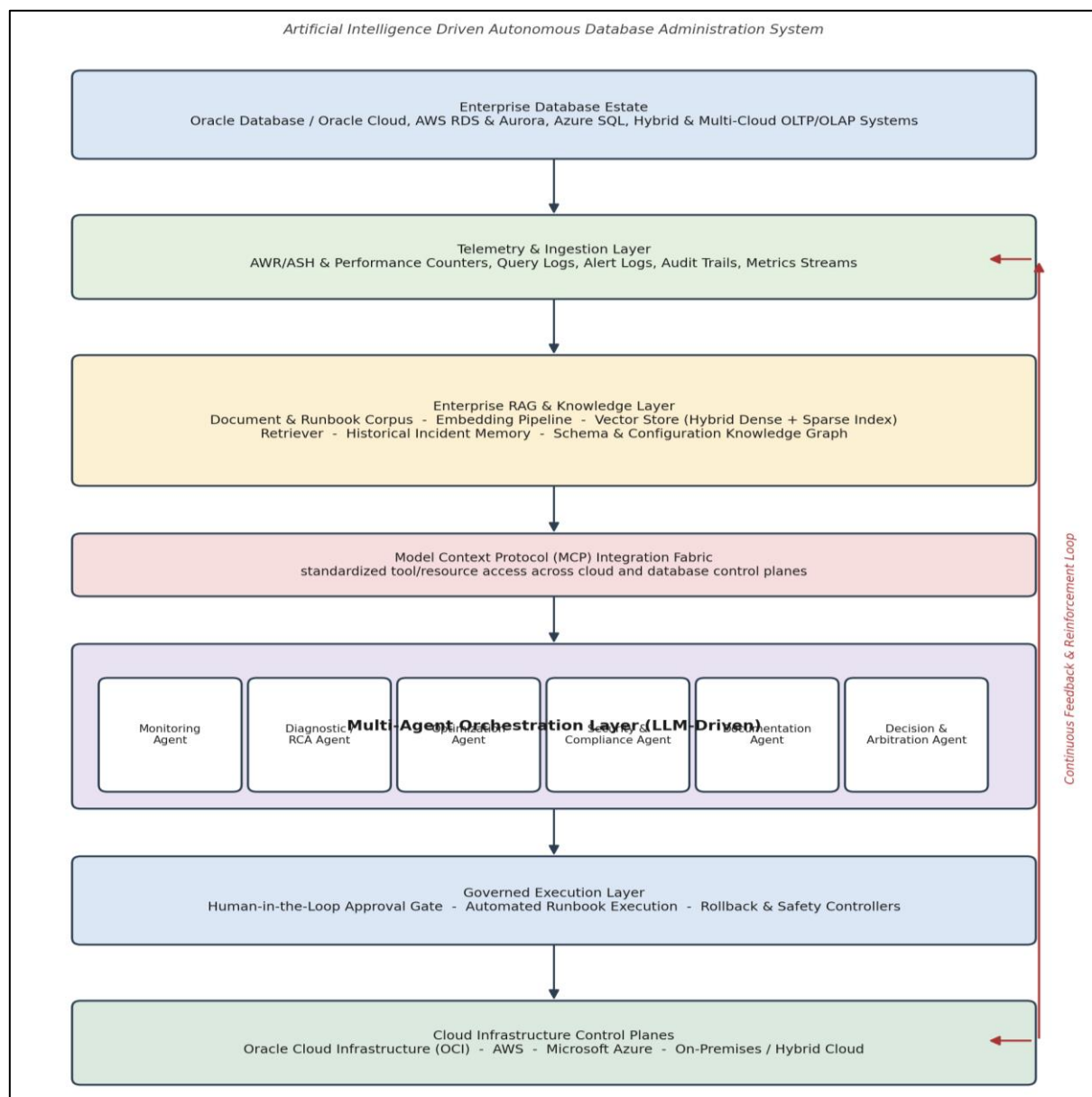


Fig 2 AIDAS Framework Architecture

➤ Enterprise RAG Pipeline

Figure 3 details the retrieval-augmented generation pipeline underlying the AIDAS knowledge layer. Four categories of source material are ingested continuously: DBA runbooks and vendor manuals, historical incident tickets, live telemetry (AWR/ASH snapshots and logs), and vendor documentation together with compliance policy documents. Each source is processed through a chunking and preprocessing stage that respects the structural boundaries specific to database documentation, for example segmenting by configuration parameter or by diagnostic rule rather than by naive fixed-length windows, consistent with the enterprise RAG content-design guidance of Packowski et al. (2024). Chunks are passed through an embedding model to produce dense vector representations, which are indexed in a hybrid vector store combining dense similarity search with sparse lexical indexing (BM25-style), a design choice motivated by evidence that hybrid retrieval

outperforms either method alone on domain-specific technical corpora (Gao et al., 2024).

At query time, a DBA question or an internally generated agent prompt is passed to a retriever that performs top-k retrieval followed by a re-ranking stage, before the retrieved passages are assembled into a context window alongside the original query. This assembled context is passed to the generator LLM, which produces a grounded answer. The historical-incident memory and schema/configuration knowledge graph shown in Figure 2 act as additional retrieval sources within this same pipeline, allowing the retriever to surface not only static documentation but also structurally related entities (a table's associated indexes, a query's historical execution-plan variants) and precedent from prior incidents of the same class, an approach informed by graph-structured and recursive retrieval research (Edge et al., 2024; Sarthi et al., 2024).

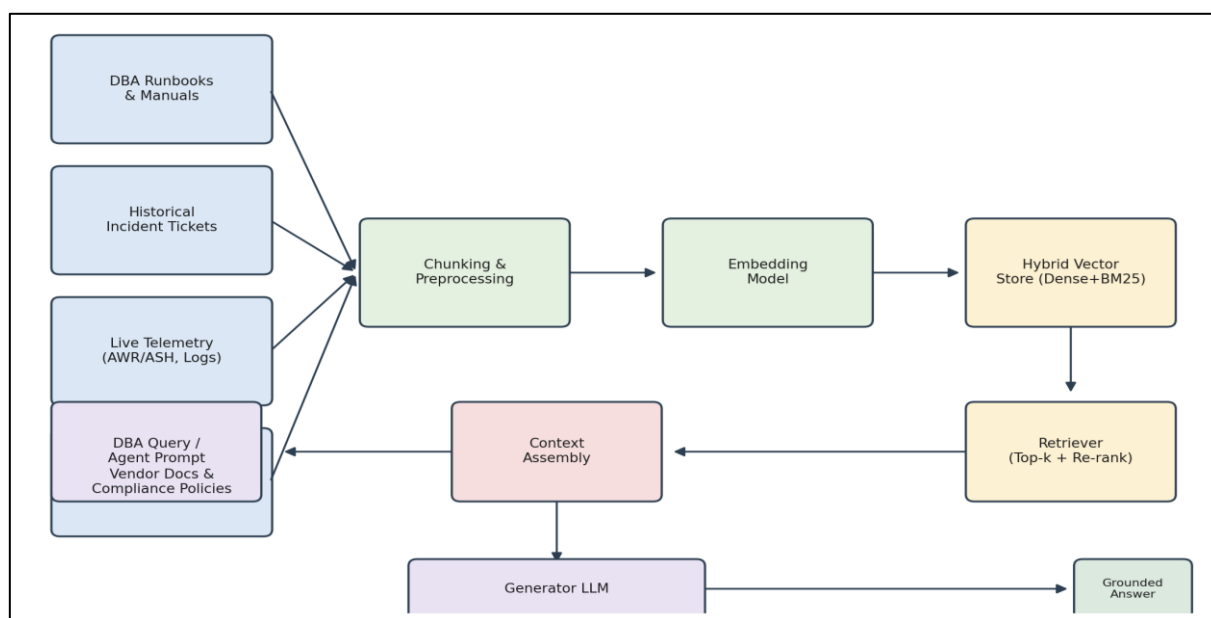


Fig 3 Enterprise RAG Pipeline for Database Administration

➤ AI Agent Decision Workflow

Figure 4 traces the decision workflow triggered by an incoming alert or event. The Monitoring Agent performs initial triage and classification, after which the Diagnostic Agent performs RAG-grounded root-cause analysis using the pipeline described in Section 9.2. The Optimization/Security Agent then generates one or more remediation candidates, each annotated with an estimated confidence score and a risk classification reflecting the operational blast radius of the proposed action (for example, an index rebuild during a maintenance window is lower risk than an online schema change during peak transaction volume).

The Decision Agent evaluates whether the proposed action's confidence exceeds a configurable threshold and its risk classification falls within an auto-executable tier. If both conditions are satisfied, the action proceeds directly to the

Execution Agent, which invokes the appropriate MCP tool to carry out the automated runbook or change action. If either condition fails, whether because confidence is insufficient or because the action falls into a high-risk category such as a destructive schema operation or a security-relevant configuration change, the workflow escalates to human-in-the-loop review, and a DBA must explicitly approve the action before execution proceeds. Following execution, a Verification Agent validates the outcome against expected health-check criteria; if verification fails, the rollback and safety controllers described in Section 9.1 are invoked. Finally, the Documentation Agent records the full incident lifecycle, including the original alert, the diagnostic reasoning, the action taken, and the verification outcome, as a new entry in the enterprise knowledge base, closing the continuous-learning feedback loop shown on the right of Figure 4.

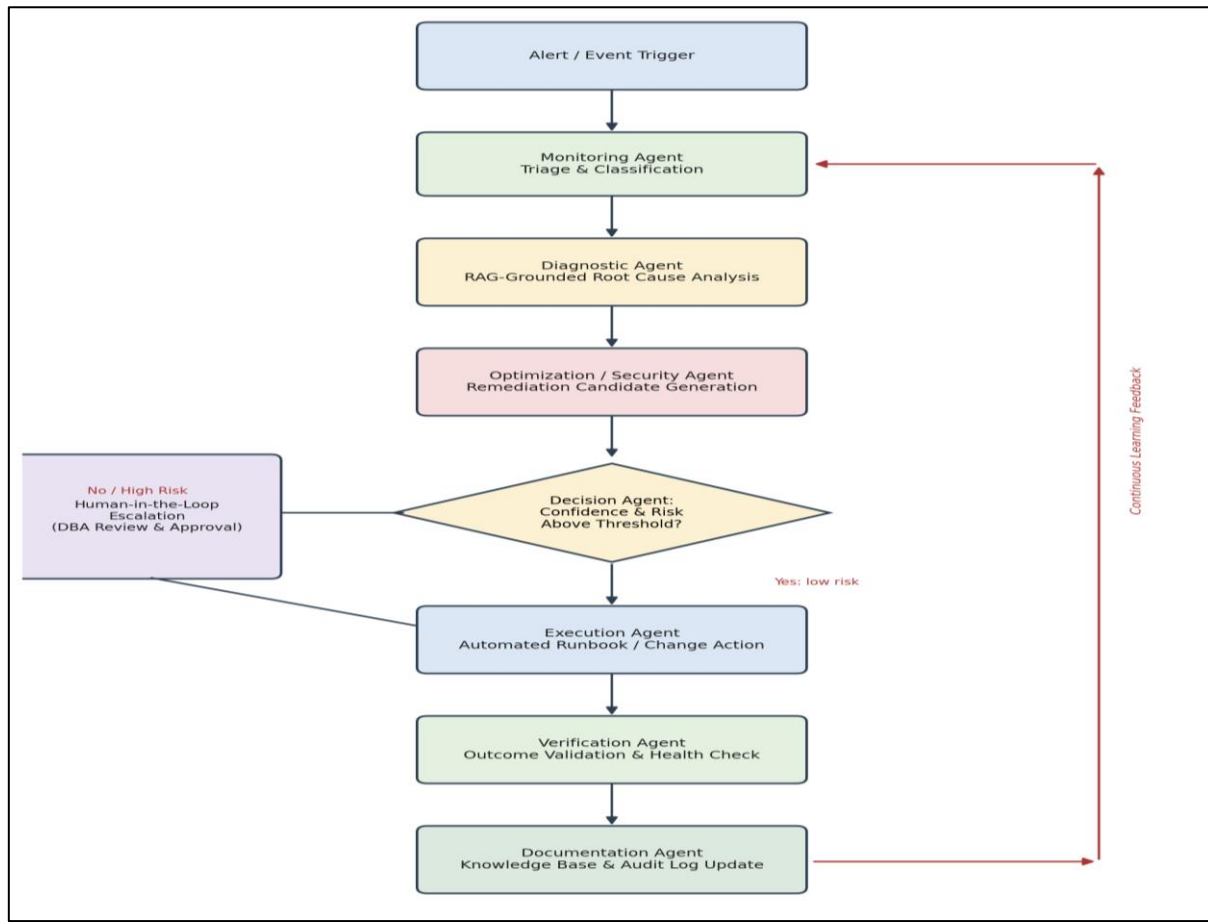


Fig 4 AI Agent Decision Workflow

➤ Algorithm

Algorithm 1 formalizes the decision workflow of Figure 3, in particular the risk-tiered arbitration mechanism central to Contribution 5.3.

Algorithm 1: AIDAS Incident Resolution Loop

Input: event stream E from Telemetry and Ingestion layer; confidence threshold τ ; risk-tier map R

Output: resolved incident record I , appended to enterprise knowledge base K

```

1: for each incoming event  $e$  in  $E$  do
2:    $c \leftarrow \text{MonitoringAgent.triage}(e)$ 
3:   if  $c$  is not anomalous then continue
4: end if
5:  $\text{ctx} \leftarrow \text{RAG Retrieve}(\text{query} = c, \text{sources} = \{\text{runbooks}, \text{incident memory}, \text{schema graph}\})$ 
6:  $\text{diagnosis} \leftarrow \text{DiagnosticAgent.analyze}(c, \text{ctx})$ 
7:  $\text{candidates} \leftarrow \text{OptimizationSecurityAgent.propose}(\text{diagnosis}, \text{ctx})$ 
8: for each action  $a$  in  $\text{candidates}$  do
9:    $(\text{confidence}, \text{risk}) \leftarrow \text{DecisionAgent.score}(a, \text{diagnosis})$ 
10:  if  $\text{confidence} \geq \tau$  and  $R[\text{risk}] = \text{AUTO\_EXECUTABLE}$  then
11:     $\text{result} \leftarrow \text{ExecutionAgent.execute}(a)$  // via MCP tool call
12:  else
13:     $\text{result} \leftarrow \text{HumanInLoop.escalate}(a, \text{confidence}, \text{risk}, \text{diagnosis})$ 
14:  end if
15:   $\text{verified} \leftarrow \text{VerificationAgent.check}(\text{result})$ 
16:  if not verified then
17:     $\text{RollbackController.revert}(a, \text{result})$ 
18:  end if
19: end for
20:  $I \leftarrow \text{DocumentationAgent.record}(e, \text{diagnosis}, \text{candidates}, \text{result}, \text{verified})$ 
21:  $K \leftarrow K \cup \{I\}$ 
22: end for
  
```

The algorithm makes explicit that human-in-the-loop escalation (line 13) is not an exceptional fallback but a first-class branch of the control flow, invoked whenever either the confidence or the risk-tier condition is not satisfied. This design reflects the governance principle articulated in Contribution 5.3: automation is extended only as far as the risk classification permits, and the threshold τ and risk-tier map R are configurable per organization, allowing more conservative enterprises to restrict automatic execution to a narrower set of low-risk actions.

➤ Enterprise Database Automation Process

Figure 5 maps the AIDAS architecture onto the eight process stages central to the enterprise automation scope defined for this framework: continuous monitoring, incident

detection, root cause analysis, infrastructure troubleshooting, performance and query optimization, security monitoring, compliance validation, and automated documentation. Each stage is not treated as an isolated module but as a station along a continuous process, with the shared Intelligent Decision Support layer, comprising RAG-grounded LLM reasoning, multi-agent arbitration, and the enterprise knowledge base, informing and being informed by every stage. This design ensures, for example, that a security finding surfaced during the Security Monitoring stage can influence the Compliance Validation stage's evidentiary record, and that a root-cause pattern discovered during RCA is immediately available to the Performance and Query Optimization stage for related instances.

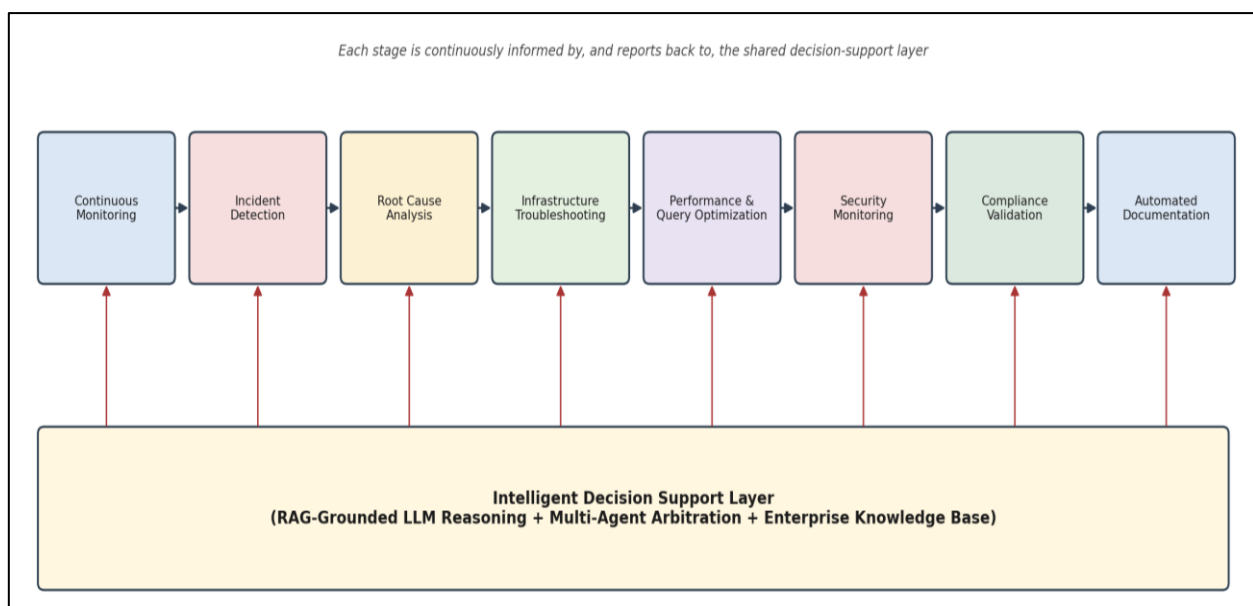


Fig 5 Enterprise Database Automation Process

VI. IMPLEMENTATION STRATEGY

A phased implementation strategy reduces the operational risk of introducing autonomous decision-making into production database environments. The strategy proposed here follows five phases, summarized in Table 5, moving from passive observation toward increasingly autonomous execution only as confidence in the system's diagnostic accuracy is established.

From a technology-stack perspective, the orchestration layer can be implemented using existing multi-agent frameworks such as AutoGen or MetaGPT-style role decomposition (Wu et al., 2023; Hong et al., 2023), adapted to the six agent roles defined in Section 9.1. The RAG layer requires a hybrid vector store supporting both dense embedding similarity and sparse lexical search, populated from existing DBA documentation, ticketing systems, and telemetry pipelines. The MCP integration fabric requires the implementation of MCP servers wrapping each cloud platform's administrative APIs (Oracle Cloud Infrastructure APIs, AWS RDS/Aurora APIs, Azure SQL Database APIs),

a scope of engineering effort comparable to building a conventional API adapter but with the advantage that, once built, the same servers are reusable across any MCP-compatible agent framework. Cost-aware model routing, following approaches such as RouteLLM (Ong et al., 2025), is recommended to direct routine monitoring and triage tasks to smaller, less expensive models while reserving the most capable models for complex diagnostic reasoning, given that LLM inference cost scales with the volume of telemetry processed.

Given the security concerns documented for early MCP deployments, including under-authenticated servers and over-permissioned tool access (Descope, 2026), the implementation strategy mandates OAuth-based authentication for every MCP server, least-privilege scoping of tool permissions (for example, separating read-only telemetry tools from configuration-write tools at the credential level), and comprehensive audit logging of every tool invocation, a requirement elaborated further in the security comparison of Table 3.

Table 3 Implementation Roadmap

Phase	Focus	Key Activities	Exit Criteria
1. Observation	Passive monitoring and knowledge-base seeding	Deploy telemetry ingestion; populate RAG corpus with existing runbooks and incident history; no automated actions	RAG retrieval accuracy validated against DBA-authored test questions
2. Advisory	Diagnostic and recommendation only	Diagnostic and Optimization Agents operate in shadow mode, producing recommendations for DBA review without execution rights	DBA acceptance rate of recommendations meets organizational threshold
3. Supervised automation	Human-in-the-loop execution	Decision Agent routes all actions through human approval regardless of confidence/risk score; Execution Agent restricted to approved actions only	Approval turnaround time and false-escalation rate meet targets
4. Bounded autonomy	Auto-execution for low-risk tier only	Risk-tier map R activated for a narrow, pre-agreed set of low-risk, reversible actions (e.g., statistics refresh, non-blocking index rebuild)	Rollback controller tested; zero unrecovered incidents in bounded scope
5. Expanded autonomy	Broader auto-executable scope with continuous audit	Risk-tier map expanded incrementally per organizational risk appetite; continuous compliance and security audit agent active	Sustained accuracy and safety metrics over an agreed observation period

➤ Enterprise Use Cases

The following illustrative scenarios trace representative incidents through the AIDAS architecture to demonstrate its internal coherence.

• Performance Degradation on Oracle Exadata (OCI)

A sustained increase in average query latency on an Oracle Exadata-based Autonomous Database instance triggers the Monitoring Agent, which classifies the anomaly as a potential resource contention event based on AWR snapshot deltas. The Diagnostic Agent retrieves relevant runbook sections and historical incidents of similar wait-event signatures through the RAG pipeline, identifying a likely cause in a recently introduced query plan regression. The Optimization Agent proposes a corrective action (a SQL plan baseline reversion), classified as low risk because it is reversible and non-blocking; given sufficient confidence, the Execution Agent applies the action automatically via the OCI MCP server, and the Verification Agent confirms latency recovery before the Documentation Agent records the resolved incident.

• Security Anomaly and Compliance Validation on AWS Aurora

An unusual pattern of privileged queries against a production Aurora cluster is flagged by the Security and Compliance Agent, which cross-references the activity against the organization's access policy retrieved from the compliance knowledge base. Because the action required, revoking or restricting a database role, carries a high-risk classification under the organization's risk-tier map, the Decision Agent escalates to human-in-the-loop review rather than executing automatically, and the Documentation Agent prepares an evidentiary summary suitable for compliance audit, referencing the specific policy clauses implicated.

• Cross-Cloud Query Optimization on Azure SQL and Hybrid Estate

A cost-optimization initiative requires identifying inefficient queries across both an Azure SQL Database deployment and an on-premises Oracle instance participating in a hybrid replication topology. Because both platforms are exposed behind MCP servers with a common tool schema, the Optimization Agent applies the same reasoning process to both environments, retrieving platform-specific execution-plan syntax through the RAG layer where needed, and produces a unified optimization report spanning both platforms without requiring platform-specific engineering in the reasoning layer itself.

• Automated Documentation Following a Multi-Cloud Migration

Following a database migration from an on-premises system to Azure SQL Database, the Documentation Agent aggregates the sequence of configuration changes, schema adjustments, and validation checks performed throughout the migration into a structured runbook entry, which is embedded into the enterprise knowledge base. This entry subsequently improves the Diagnostic Agent's retrieval quality for any future incident referencing the migrated system, directly operationalizing the institutional-memory contribution described in Section 5.5.

➤ Performance Evaluation

AIDAS is presented in this paper as a conceptual, design-science framework and has not, at the time of writing, been implemented as a complete system and deployed against production workloads. Accordingly, this section does not report empirical benchmark results for AIDAS itself. Instead, it presents a conceptual performance projection, illustrated in Figure 5 and Table 3, derived from patterns reported for its constituent components in the cited literature, together with a discussion of what a rigorous empirical evaluation of AIDAS would require. This distinction between illustrative projection and measured

result is maintained deliberately throughout this section, consistent with the honest-reporting principle guiding this paper.

The projection in Figure 5 is grounded in three literature-reported findings. First, D-Bot's tree-search and multi-agent diagnostic architecture has been reported to produce well-founded root-cause reports within a time budget suitable for online operation, in contrast to manual diagnosis processes that DBAs describe as tedious and slow for large instance populations (Zhou et al., 2024). Second,

RCRank's multimodal root-cause ranking has been shown to improve identification accuracy over single-signal baselines for slow-query diagnosis (Ouyang et al., 2025). Third, self-driving database research has established that continuous, always-on monitoring achievable by an automated system structurally exceeds the practical coverage of periodic human review across large instance populations (Pavlo et al., 2021). Figure 6 aggregates these qualitative patterns into an illustrative comparison across five dimensions; the underlying values are conceptual placeholders on a relative qualitative scale and are explicitly labeled as such.

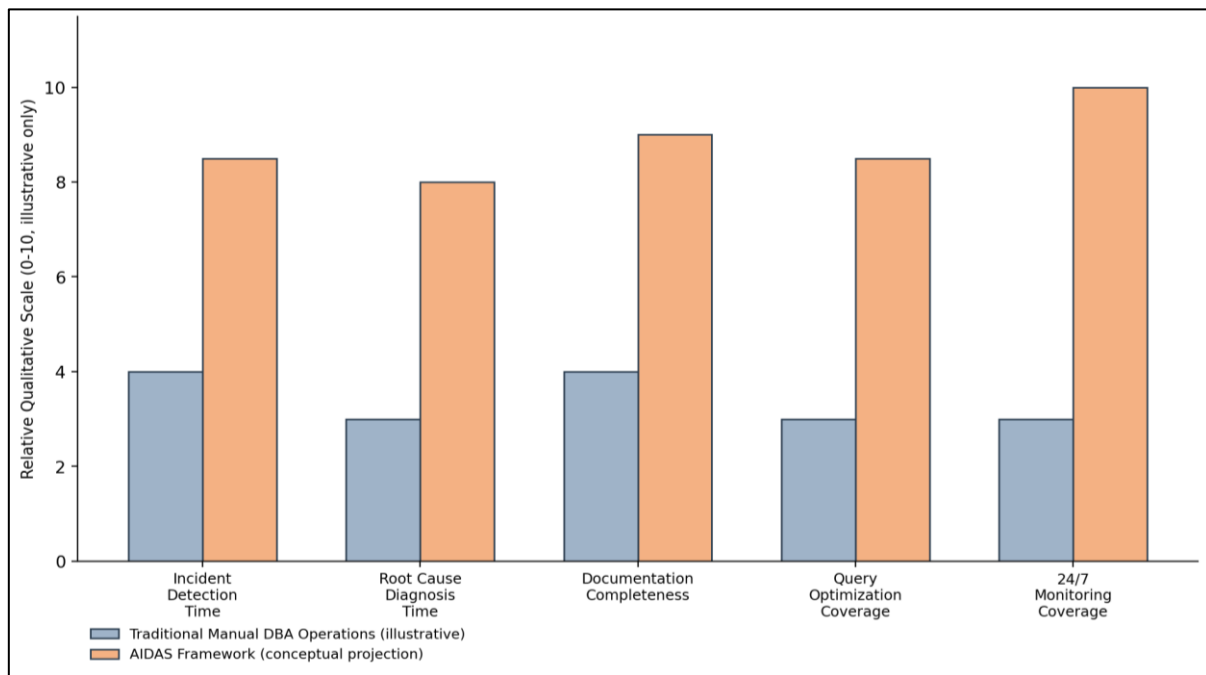


Fig 6 Conceptual Comparison of Traditional DBA Operations vs. the Proposed AIDAS Framework (Illustrative Projection)

Table 4 Performance Metrics (Illustrative Projection, Not Empirical Benchmark Data)

Metric	Traditional Manual DBA (Qualitative Baseline)	AIDAS (Conceptual Projection)	Basis in Literature
Incident detection latency	Dependent on alert-review cadence; can be delayed by on-call rotation and alert fatigue	Near-continuous, bounded primarily by telemetry ingestion latency	Self-driving DBMS continuous monitoring (Pavlo et al., 2021)
Root cause diagnosis time	Highly variable; dependent on individual DBA experience and availability of tribal knowledge	Reduced and more consistent, grounded in retrieved runbook and incident precedent	D-Bot reported diagnosis turnaround (Zhou et al., 2024); RCRank ranking accuracy (Ouyang et al., 2025)
Documentation completeness	Inconsistent; often deferred or skipped under operational pressure	Systematic, generated as a byproduct of every resolved incident	Institutional-memory / tacit-knowledge attrition findings (Zhang et al., 2025)
Cross-cloud consistency	Fragmented; separate tooling and practices per platform	Unified reasoning via MCP-standardized tool access	MCP standardization for cross-system integration (Anthropic, 2024)
24/7 monitoring coverage	Constrained by staffing and on-call capacity	Structurally continuous	AIOps failure-management literature (Zhang et al., 2025)

A rigorous empirical evaluation of AIDAS, identified in Section 16 as a priority for future work, would require an implemented system tested against a benchmark suite comparable to those recently proposed for evaluating autonomous cloud operations agents, together with a

controlled comparison against both traditional DBA workflows and existing point solutions such as D-Bot, under matched workloads and incident scenarios.

➤ Comparative Analysis

Table 1 (Section 7) positioned AIDAS against D-Bot/GaussMaster, RCRank, and Oracle Autonomous Database along architectural dimensions. This section extends that comparison to a security-specific analysis,

presented in Table 5, because the introduction of an MCP-based integration fabric and autonomous execution capability materially changes the security posture of database administration relative to both traditional tooling and existing LLM copilots.

Table 5 Security Comparison

Security Dimension	Traditional DBA Tooling	Generic LLM Copilot (Unstructured Tool Access)	AIDAS (Proposed Design)
Authentication to managed systems	Direct administrator credentials; individually audited	Often ad hoc API keys with broad scope	OAuth-based MCP server authentication with per-agent credential scoping
Authorization granularity	Role-based access control at the database level	Frequently coarse; tool access not separated by risk	Least-privilege MCP tool scoping; read-only vs. configuration-write separation
Audit trail	Native database audit logging	Inconsistent; dependent on ad hoc logging	Comprehensive logging of every MCP tool invocation, linked to the Documentation Agent record
Prompt-injection / malicious-tool exposure	Not applicable	Documented risk of under-authenticated or over-permissioned MCP servers (Descope, 2026)	Mandatory server authentication, permission scoping, and human-in-the-loop gating for high-risk actions
Data residency and governance	Enterprise-controlled	Variable; dependent on third-party model/tool provider	Configurable per organization; compliance agent validates against policy corpus
Change-management governance	Manual change-approval boards	Typically absent or bypassed	Formal risk-tiered arbitration (Algorithm 1) with mandatory escalation thresholds

This comparison highlights that the principal security advantage of AIDAS over an unstructured LLM copilot approach is not the use of LLMs per se, which introduces its own risks, but the disciplined application of MCP authentication and scoping standards together with the risk-tiered governance mechanism of Algorithm 1, both of which are necessary preconditions for safely extending any autonomous capability into production database change management.

➤ Advantages

- Unified administrative scope: a single architecture addresses monitoring, diagnosis, optimization, security, compliance, and documentation rather than requiring separate point tools for each function.
- Cross-cloud portability: the MCP integration fabric allows the same reasoning layer to operate across Oracle Cloud Infrastructure, AWS, and Azure, reducing duplicated engineering effort relative to platform-specific automation.
- Explainability: RAG grounding produces natural-language, citation-linked justifications for diagnostic conclusions and recommended actions, improving DBA trust and auditability relative to opaque ML-based anomaly scoring.
- Proportionate governance: the risk-tiered arbitration mechanism extends automation only as far as an organization's risk appetite permits, avoiding an all-or-nothing choice between full manual control and full automation.

- Institutional memory retention: the closed-loop documentation mechanism directly addresses the tacit-knowledge attrition problem that traditional tooling leaves largely unaddressed.
- Extensibility: because cloud-specific logic is isolated within MCP servers, adding a new database platform or cloud provider does not require re-engineering the multi-agent reasoning layer.

VII. LIMITATIONS

This paper's contribution is conceptual and architectural; several limitations follow directly from that status and from the underlying technologies AIDAS builds upon.

- Absence of empirical validation: AIDAS has not been implemented as a complete system and evaluated against production workloads. The performance projections in Section 12 are illustrative and derived from component-level literature findings, not measurements of the integrated system, and should not be read as evidence of achieved performance.
- Hallucination and reasoning-error risk: despite RAG grounding, LLM-based agents can still produce incorrect diagnoses or recommend inappropriate actions, particularly for incident patterns underrepresented in the retrieval corpus; the risk-tiered human-in-the-loop mechanism mitigates but does not eliminate this risk.
- MCP security maturity: the Model Context Protocol is a relatively young standard, and documented

vulnerabilities in early deployments, including under-authenticated servers and over-permissioned tool access (Desclope, 2026), mean that production deployment of the AIDAS integration fabric requires careful security engineering beyond what the protocol enforces natively.

- Operational and inference cost: continuous multi-agent LLM reasoning over high-volume telemetry carries non-trivial computational cost; cost-aware routing strategies (Section 10) mitigate but do not eliminate this consideration, and a full cost-benefit analysis requires empirical deployment data not yet available.
- Change-management risk concentration: any framework capable of autonomous configuration or schema changes concentrates risk at the point of execution; the rollback and safety controllers described in Section 9.1 are a necessary but not sufficient mitigation, and organizations adopting AIDAS would need to validate rollback reliability empirically before expanding the auto-executable risk tier.
- Data governance across jurisdictions: operating a single reasoning layer across Oracle Cloud, AWS, and Azure in a multinational enterprise raises data residency and cross-border transfer considerations under regulations such as GDPR, which must be addressed at the MCP server and knowledge-base configuration level rather than assumed to be handled automatically.
- Model and knowledge drift: both the underlying LLMs and the enterprise knowledge base require ongoing maintenance; without active curation, retrieval quality and diagnostic accuracy may degrade as database platforms, versions, and organizational practices evolve.

➤ Future Research

- Empirical pilot deployment: implement a reference AIDAS instance against a controlled, non-production database environment and measure diagnostic accuracy, resolution time, and false-escalation rate against a matched traditional DBA workflow, addressing the primary limitation identified in Section 15.
- Benchmark suite development: adapt or extend existing autonomous-cloud-operations benchmark methodologies to the specific task set defined in Section 9.5, enabling standardized comparison across AIDAS and prior systems such as D-Bot and RCRank.
- Formal safety and red-team evaluation: subject the MCP integration fabric and the risk-tiered arbitration mechanism to adversarial testing, including prompt-injection and tool-poisoning scenarios documented in the MCP security literature (Desclope, 2026).
- Fine-tuning versus prompting trade-off analysis: empirically compare retrieval-augmented prompting against task-specific fine-tuning for each of the six agent roles, extending the broader LLM4AIOps discussion of method selection (Zhang et al., 2025).
- Cost-aware multi-agent routing: extend routing approaches such as RouteLLM (Ong et al., 2025) into a database-administration-specific cost-quality trade-off model, informed by the criticality tiers defined in Algorithm 1.

- Cross-cloud compliance automation certification: investigate formal certification pathways for the Security and Compliance Agent's evidentiary output, in collaboration with regulatory and audit stakeholders, to establish AIDAS-generated compliance records as admissible audit evidence.

VIII. CONCLUSION

This paper introduced AIDAS, a novel conceptual architecture for autonomous enterprise database administration that unifies generative AI, retrieval-augmented generation, vector search, the Model Context Protocol, and risk-governed multi-agent orchestration into a single, cross-cloud framework. Unlike prior LLM-for-database systems, which address individual administrative tasks in isolation, AIDAS is designed to span the full administrative lifecycle, monitoring, incident detection, root cause analysis, infrastructure troubleshooting, performance and query optimization, security monitoring, compliance validation, and automated documentation, while remaining portable across Oracle Cloud Infrastructure, AWS, and Azure through a standardized integration fabric. Six original contributions were articulated: the layered AIDAS architecture, the MCP-based cross-cloud integration fabric, risk-tiered multi-agent decision arbitration, an enterprise RAG design tailored to database administration corpora, closed-loop institutional memory capture, and a vendor-neutral abstraction over existing cloud-native automation. The paper has been explicit throughout that its contribution is architectural and conceptual rather than empirically validated, and it has identified a concrete agenda, beginning with a controlled pilot deployment and extending through formal safety evaluation, for establishing AIDAS's practical performance and security characteristics. As enterprise database estates continue to grow in scale and heterogeneity, frameworks that combine explainable, RAG-grounded reasoning with disciplined, risk-proportionate governance, rather than either unconstrained automation or purely manual operation, represent a promising and, this paper argues, necessary direction for the future of enterprise database administration.

REFERENCES

- [1]. Amazon Web Services. (2024). Amazon Aurora machine learning-based performance insights and optimization. AWS Documentation.
- [2]. Anthropic. (2024). Introducing the Model Context Protocol. Anthropic News. <https://www.anthropic.com/news/model-context-protocol>
- [3]. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
- [4]. Desclope. (2026). What is the Model Context Protocol (MCP) and how it works. Desclope Security Research.
- [5]. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional

- transformers for language understanding. *Proceedings of NAACL-HLT 2019*, 4171–4186.
- [6]. Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S., & Larson, J. (2024). From local to global: A graph RAG approach to query-focused summarization. *arXiv:2404.16130*.
- [7]. European Parliament and Council. (2016). Regulation (EU) 2016/679 on the protection of natural persons with regard to the processing of personal data (General Data Protection Regulation). *Official Journal of the European Union*.
- [8]. European Parliament and Council. (2024). Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (AI Act). *Official Journal of the European Union*.
- [9]. Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Guo, Q., Wang, M., & Wang, H. (2024). Retrieval-augmented generation for large language models: A survey. *arXiv:2312.10997*.
- [10]. Guo, Z., Xia, L., Yu, Y., Ao, T., & Huang, C. (2024). LightRAG: Simple and fast retrieval-augmented generation. *arXiv preprint*.
- [11]. Helskyaho, H., Kraft, J., & Koning, J. (2021). Oracle Autonomous Database for machine learning. *Oracle machine learning research reports*.
- [12]. Hong, S., Zheng, X., Chen, J., Cheng, Y., Wang, J., Zhang, C., Wang, Z., Yau, S. K. S., Lin, Z., Zhou, L., Ran, C., Xiao, L., & Wu, C. (2023). MetaGPT: Meta programming for a multi-agent collaborative framework. *arXiv:2308.00352*.
- [13]. Hu, S., Lu, C., & Clune, J. (2024). Automated design of agentic systems. *arXiv:2408.08435*.
- [14]. Johnson, J., Douze, M., & Jégou, H. (2021). Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3), 535–547.
- [15]. Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W. (2020). Dense passage retrieval for open-domain question answering. *Proceedings of EMNLP 2020*, 6769–6781.
- [16]. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459–9474.
- [17]. Li, G., Zhou, X., Li, S., & Gao, B. (2019). QTune: A query-aware database tuning system with deep reinforcement learning. *Proceedings of the VLDB Endowment*, 12(12), 2118–2130.
- [18]. Li, X., et al. (2024). A survey on LLM-based multi-agent systems: Workflow, infrastructure, and challenges. *Vicinagearth*, 1, 9.
- [19]. Microsoft Azure. (2024). Azure SQL Database automatic tuning documentation. *Microsoft Learn*.
- [20]. National Institute of Standards and Technology. (2023). Artificial Intelligence Risk Management Framework (NIST AI 100-1). *U.S. Department of Commerce*.
- [21]. Ong, I., Almahairi, A., Wu, V., Chiang, W. L., Wu, T., Gonzalez, J. E., Kadous, M. W., & Stoica, I. (2025). RouteLLM: Learning to route LLMs from preference data. *International Conference on Learning Representations*.
- [22]. Oracle Corporation. (2023). Self-driving database: Autonomous Database Oracle 19c. *Oracle Product Documentation*.
- [23]. Ouyang, B., Zhang, Y., et al. (2025). RCRank: Multimodal ranking of root causes of slow queries in cloud database systems. *Proceedings of the ACM on Management of Data*.
- [24]. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35, 27730–27744.
- [25]. Packowski, S., Halilovic, I., Schlotfeldt, J., & Smith, T. (2024). Optimizing and evaluating enterprise retrieval-augmented generation (RAG): A content design perspective. *arXiv:2410.12812*.
- [26]. Pavlo, A., Butrovich, M., Joshi, A., Ma, L., Menon, P., Van Aken, D., Lee, L., & Salakhutdinov, R. (2021). Make your database system dream of electric sheep: Towards self-driving operation. *Proceedings of the VLDB Endowment*, 14(12), 3211–3221.
- [27]. Prasad, P., & Rich, C. (2018). Innovation insight for AIOps. *Gartner Research*.
- [28]. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. *Proceedings of EMNLP-IJCNLP 2019*, 3982–3992.
- [29]. Roy, D., Zhang, X., Bhave, R., Bansal, C., Las-Casas, P., Fonseca, R., & Rajmohan, S. (2024). Exploring LLM-based agents for root cause analysis. *arXiv:2403.04123*.
- [30]. Sarthi, P., Abdullah, S., Tuli, A., Khanna, S., Goldie, A., & Manning, C. D. (2024). RAPTOR: Recursive abstractive processing for tree-organized retrieval. *arXiv:2401.18059*.
- [31]. Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36.
- [32]. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
- [33]. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35.
- [34]. Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A. H., White, R. W., Burger, D., & Wang, C. (2023). AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv:2308.08155*.
- [35]. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct:

- Synergizing reasoning and acting in language models. International Conference on Learning Representations.
- [36]. Zhang, L., et al. (2025). A survey of AIOps in the era of large language models. arXiv:2507.12472.
- [37]. Zhao, X., Zhou, X., & Li, G. (2024). Chat2Data: An interactive data analysis system with RAG, vector databases and LLMs. Proceedings of the VLDB Endowment.
- [38]. Zhou, X., Chai, C., Li, G., & Sun, J. (2022). Database meets artificial intelligence: A survey. IEEE Transactions on Knowledge and Data Engineering, 34(3), 1096–1116.
- [39]. Zhou, X., Jin, L., Sun, J., Zhao, X., Yu, X., Li, S., Wang, T., Li, K., & Liu, L. (2021). DBMind: A self-driving platform in openGauss. Proceedings of the VLDB Endowment, 14(12), 2743–2746.
- [40]. Zhou, X., Li, G., Chai, C., & Feng, J. (2021). A learned query rewrite system using Monte Carlo tree search. Proceedings of the VLDB Endowment, 15(1), 46–58.
- [41]. Zhou, X., Li, G., & Liu, Z. (2023). LLM as DBA. arXiv:2308.05481.
- [42]. Zhou, X., Li, G., Sun, Z., Liu, Z., Chen, W., Wu, J., Liu, J., Feng, R., & Zeng, G. (2024). D-Bot: Database diagnosis system using large language models. Proceedings of the VLDB Endowment, 17(10), 2514–2527.
- [43]. Zhou, X., Li, G., Wu, J., Liu, J., Sun, Z., & Zhang, X. (2023). A learned query rewrite system. Proceedings of the VLDB Endowment, 16(12), 4110–4113.
- [44]. Zhou, W., Sun, J., Zhou, X., Li, G., Liu, L., Wu, H., & Wang, T. (2025). GaussMaster: An LLM-based database copilot system. arXiv:2506.23322.
- [45]. Zhu, J. P., Cai, P., Niu, B., Ni, Z., Xu, K., Huang, J., Wan, J., Ma, S., Wang, B., Zhang, D., et al. (2024). AutoTQA: Towards autonomous tabular question answering through multi-agent large language models. Proceedings of ICDE 2024.
- [46]. Zhuge, M., Wang, W., Kirsch, L., et al. (2024). GPTSwarm: Language agents as optimizable graphs. Proceedings of the International Conference on Machine Learning (ICML).