IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

LEVERAGING DEVOPS PRACTICES TO OPTIMIZE MULTI-TENANT ORACLE 12C ENVIRONMENTS

Raghu Murthy Shankeshi

Abstract: Remote implementation of DevOps techniques substantially improved multi-tenant environment optimization practices in enterprise database management. Oracle 12c's multi-tenant architecture allows organizations to achieve database consolidation and resource-sharing efficiency. Multi-tenant Oracle 12c management requires organizations to overcome major obstacles, focusing on performance control, security needs, and scalability requirements alongside deployment automation. The research evaluates how DevOps methods, especially Continuous Integration and Continuous Deployment (CI/CD), Infrastructure as Code (IaC), and automated monitoring, improve Oracle 12c performance, reliability, and security capabilities.The length of this methodology involves implementing DevOps practices through these tools: Terraform for automated infrastructure provision, Ansible for configuration management, Kubernetes for container orchestration, and Oracle Enterprise Manager for proactive monitoring. The study tracks multiple performance indicators: deployment speed, query response time, resource utilization system availability, and security compliance measurements before and after implementing DevOps methods. Experimental findings reveal that the database operates more efficiently while operations costs decrease and systems achieve better failure resistance. Database automatic provisioning, together with schema automation, has been shown to speed up deployment processes, and active database monitoring decreases system outages. Security automation under DevOps management helps organizations improve their compliance with RBAC and vulnerability management standards. Using DevOps in Oracle 12c multi-tenant structures provides organizations with improved adaptability and extends their operational capabilities and scalability benefits. Further development should investigate AI automation systems and cloud-native DevOps methods for maximum optimization.

Keywords: **DevOps, Oracle 12c, Multi-Tenant Architecture, Database Management, Continuous Integration/Continuous Deployment (CI/CD)**

1. INTRODUCTION

Implementing DevOps as a modern development methodology empowers teams to cooperate better between developers and operation technicians, thus optimizing operational procedures and deployment times while expanding automation capabilities. Database management has proven to slow down IT operations because of its complicated nature, sensitive risk profile, and dependence on hands-on procedures. Organizations that implement DevOps techniques in database management achieve automatic speedier deployments, decreased downtime, better system performance, and improved protection.

Modem enterprise applications rely heavily on databases, which need to maintain high performance and availability and have scalable features. DevOps for database management provides multi-tenant environments with especially valuable capabilities because it combines enhanced capabilities with automated monitoring and provisioning functions and efficient resource utilization.

The implementation of Oracle 12c represented a pivotal development because it added multi-tenancy capabilities that enhanced database consolidation, resource distribution, and management performance. The pluggable database (PDB) feature of Oracle 12c Multi-Tenancy resides within a single container database (CDB), thus allowing database instance independence:

i. Memory and storage resources are better utilized because multiple tenants can share their infrastructure.

ii. Simplified database maintenance through centralized Management of PDBs.

iii. The database capability expands through dynamic provisioning because of improved scalability and agility.

iv. Enhanced compliance security because PDBs run autonomously within the framework that implements organizational governance policies.

v. The multi-tenant system architecture efficiently works with DevOps methods to allow automatic provisioning, simplified deployment, and convenient scaling, which enterprises need to run multiple applications and customer databases on shared infrastructure.

1.1. The Significance of Optimizing Multi-Tenant Oracle 12c Environments

Implementing optimal techniques for managing Oracle 12c multi-tenancy becomes essential for organizations that select cloud-based and multi-tenant database solutions because it ensures high performance and cost-effectiveness and strengthens database security systems. Several barriers emerge during the Management of multi-tenant database systems at scale, which include:

1.1.1 Performance bottlenecks due to resource contention among tenants.

1.1.2. The deployment management process and schema updates across numerous databases remain complex.

1.1.3. Security weaknesses, along with compliance problems, arise when sharing data processing environments.

1.1.4. Operation inefficiencies increase along with downtime whenever manual intervention occurs.

1.1.5. The solution includes DevOps automation, CI/CD, and IaC, together with real-time monitoring systems, to overcome these challenges.

1.1.6. The system performance improves because these tools optimize query execution and resource distribution.

1.1.7. Database downtime reduction occurs through automatic deployment methods that simultaneously improve business agility.

1.1.8. Security best practices and compliance policies can be implemented with seamless measures.

The organization achieves enhanced operational efficiency when it monitors operations proactively together with anomaly detection systems.

This research investigates how DevOps practices influence multi-tenant Oracle 12c environment optimization under the following aspects: Evaluates the performance enhancement achieved through DevOps automated systems that handle multi-tenant Oracle 12c databases., effectiveness of CI/CD pipelines in schema update management and deployment tasks is a key investigation point., evaluates the effectiveness of automated security measures to minimize compliance threats and automated monitoring systems.

The research addresses critical objectives to showcase how enterprises benefit from deploying DevOps solutions to boost database performance and address security and scalability requirements within Oracle 12c multi-tenant deployments.

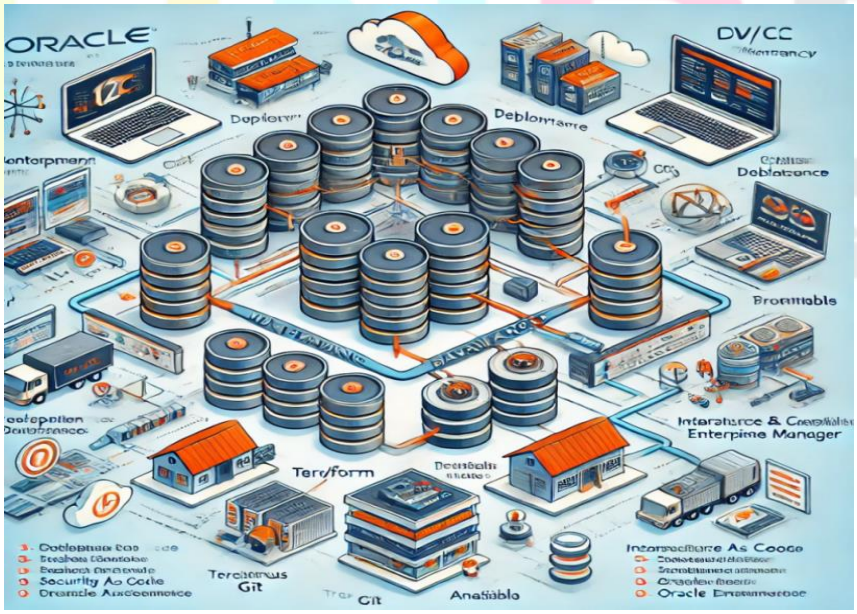

Fig 1: High-Level Architecture of Oracle 12c Multi-Tenancy with DevOps Integration.

The diagram depicts how DevOps approaches function together with Oracle 12c Multi-Tenant setup. The deployment comprises a Container Database (CDB) that contains numerous Pluggable Databases (PDBs) for optimal resource management and database combination.

2. BACKGROUND AND RELATED WORK

2.1. Evolution of Database Management and the Shift to DevOps

2.1.1. Database management systems have experienced fundamental changes during the previous decades. Monolithic architectures controlled database management before multi-application instances existed with separate database instances that demanded manual tasks and time-consuming maintenance. Enterprise applications attained complex levels that demanded increased scalability, operational automation, and operational efficiency needs.

2.1.2. Database management was transformed through cloud computing and multi-tenant database architectures, enabling various applications or customers to share one database infrastructure while maintaining application-level separation. Traditional database administration approaches created difficulties when applied to such environments because they resulted in labor-intensive provisioning procedures, lengthy deployment periods, security weaknesses, and poor resource management methods.

2.1.3. The implementation of DevOps principles began happening in organizations because they needed to solve their database management issues. Software developers and operators integrate their practices through automation tools to optimize database reactiveness while securing its components and expanding its capacity. It involves:

i. Database management implements Infrastructure as Code (IaC), which enables automatic provision of infrastructure components.

ii. Implementing Continuous Integration/Continuous Deployment (CI/CD) pipelines simplifies database adjustment processes.

iii. The system uses automated system logs for better performance data analysis.

iv. The system uses automated security compliance tools that enforce rules and avoid exploitation opportunities.

v. DevOps-driven database administration through Oracle 12c benefits multi-tenant deployments by enabling businesses to decrease installation windows, strengthen defenses, and maximize platform efficiency.

2.1.4. Key Features of Oracle 12c Multi-Tenancy.

The multi-tenant feature in Oracle 12c performed a groundbreaking change by designing a solution that increases database management efficiency while improving scalability and consolidation capabilities. The main feature of Oracle 12c multi-tenancy consists of Pluggable Databases (PDBs) that run inside Container Databases (CDBs). This architecture offers several advantages:

i. Pluggable databases in Oracle 12c share available memory storage and processing power, which allows resources to be used efficiently.

ii. Database administrators have simpler maintenance tasks and administration duties because they manage all PDBs through one CDB platform.

iii. New PDBs enable rapid provisioning because they do not require individual installations to function separately from CDBs.

iv. Multiple PDBs exist independently in the same instance yet share security policies from the central database.

v. Database efficiency is substantially improved from automated performance monitoring systems operated by DevOps and optimization tools.

2.1.5. Existing Studies on DevOps Adoption in Database Environments

Studies focus on DevOps practice implementation within databases to understand its effects on operational speed security and system scalability. Research findings indicate that:

i. Organizations installing CI/CD pipelines for databases improve operational resilience and reduce outages by 50% (Source: DatabaseOps Report).

ii. The application of DevOps within Oracle 12c multi-tenancy databases has been proven through existing research studies, which show:

iii. A system performs automated modifications to database schemas spreading across multiple pluggable databases.

iv. The application of artificial intelligence-based monitoring tools enhances database query speed.

v. Organizations that implement infrastructure automation obtain better disaster recovery strategies.

vi. The current research aims to resolve standardization issues in DevOps best practices for multi-tenant database management.

Table 1: Comparison of Traditional Database Management vs. DevOps-Driven Oracle 12c Management

FEATURE	TRADITIONAL DATABASE MANAGEMENT	DEVOPS-DRIVEN ORACLE 12C MANAGEMENT
Deployment Speed	Manual, time-consuming	Automated, rapid deployment
Resource Utilization	In isolated instances, inefficient	Shared resources, optimized usage
Scaling Approach	Manual scaling, downtime risks	On-demand scaling, minimal downtime
Security Compliance	Periodic manual audits	Continuous automated compliance
Monitoring & Performance	Reactive troubleshooting	Proactive AI-driven monitoring
Change Management	Risky, slow schema updates	Version-controlled, seamless updates
Operational Cost	High due to maintenance efforts	Lower through automation & efficiency

This table highlights the advantages of DevOps-driven Oracle 12c management, reinforcing its role in improving database performance, scalability, and security.

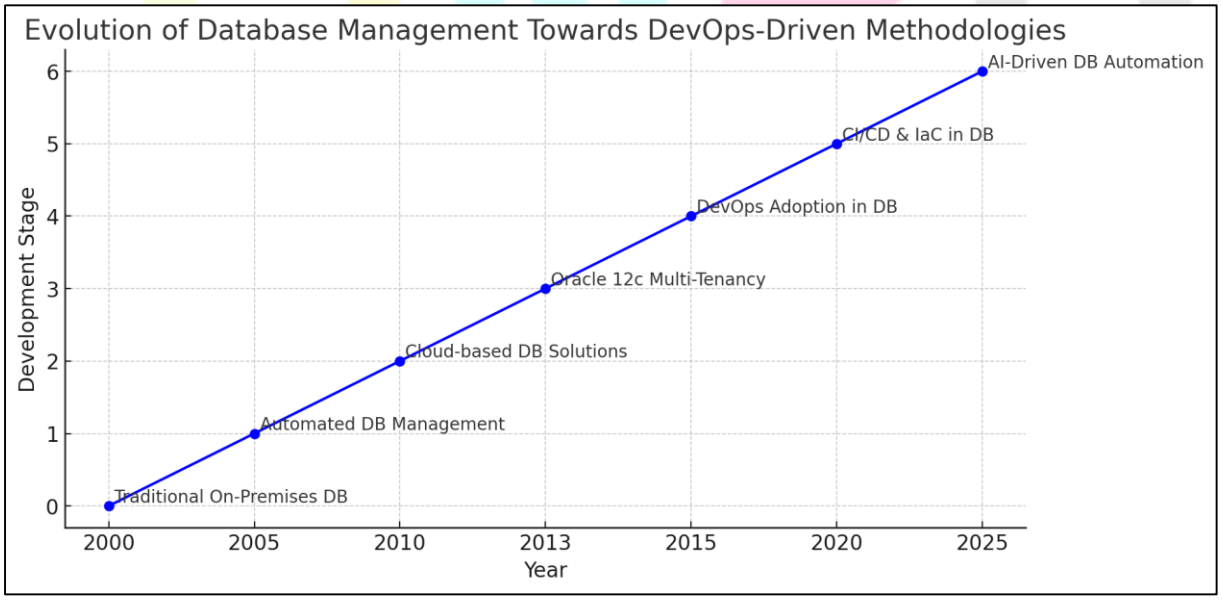

Fig 2: Evolution of Database Management Towards DevOps-Driven Methodologies.

The graphic illustrates the development of database administration since traditional relational systems at physical locations went through progressive stages, ending with DevOps automation. The field of database administration has made significant advancements because of cloud-based solutions,, Oracle 12c multi-tenancy implementation,, and DevOps integration. Organizations have improved performance, security, and operational efficiency through their implementation of CI/CD pipelines, Infrastructure as Code (IaC), and AI-driven database automation.

3. CHALLENGES IN MANAGING MULTI-TENANT ORACLE 12C ENVIRONMENTS

The implementation of Oracle 12c multi-tenancy brings multiple benefits, including optimized resource consumption, simpler administration, and better scalability. However, the administration of database environments built for multiple tenants faces multiple obstacles that organizations need to handle for optimal performance, security, and operational efficiency. The initial part of this section identifies primary obstacles in Oracle 12c multi-tenant administration preceding the implementation of DevOps practices.

3.1. Resource Isolation and Performance Bottlenecks.

Properly separating pluggable databases (PDBs) remains a core issue that multi-tenant database administrators must handle in their environments. Resource contention among multiple tenants sharing a single container database (CDB) leads to performance issues because of which users experience:

3.1.1. The database performance of one tenant can negatively affect the performance of other tenants running on the same system.

3.1.2. When a single PDB uses many resources, its performance can negatively affect other PDBs because of CPU and memory contentions.

3.1.3. Inadequate resource allocation generates storage conflicts that trigger extended disk I/O operations and delayed performance.

The absence of automated resource allocation requires database administrators to supervise and modify workloads by hand, which proves both a waste of effort and effective. The problem grows severe in busy system environments that require immediate response from their infrastructure.

3.2. Security and Compliance Concerns

A multi-tenant database environment faces security problems because users and applications maintain operations under the same physical infrastructure. Strategies that lack robust security protocols create substantial dangers because data breaches, unauthorized entrances, and compliance problems emerge when these measures are poorly enforced. Key challenges include:

3.2.1. Access control issues: Unauthorized access to shared resources becomes possible when any tenant operating system suffers an attack.

3.2.2. Data leakage risks: When organizations fail to enforce strict protection policies between their systems, sensitive details may be exposed unsafely.

3.2.3. Regulatory compliance difficulties: Sustaining compliance with GDPR, HIPAA, and PCI-DSS requires both continuous system monitoring and repeated policy implementation.

Organizations experience difficulty maintaining security compliance in all PDBs because they lack DevOps automated security policies and real-time monitoring, which leads to higher vulnerability risks.

3.3. Scalability and High Availability

Implementing seamless scalability and high availability represent significant obstacles throughout Oracle 12c multi-tenant environments. Organizations often face:

3.3.1. Traditional scaling methods lead to service disruptions because they need system downtime.

3.3.2. Managing backups across multiple PDBs proves difficult because it demands operation-free procedures.

Failover inefficiencies:

3.3.3. System outages increase in duration whenever automated failover strategies are absent.

Business applications need better scalability, but current database management systems cannot adapt accordingly, which makes DevOps automation the necessary solution for smooth growth.

3.4. Deployment and Configuration Management Issues

Organizations experience operational difficulties when managing database deployments with configuration management at multiple PDBs. Organizations encounter challenges such as:

3.4.1. Manual configuration inconsistencies:

3.4.2. Manual scripting during traditional database implementations leads to increased risk of error occurrences.

3.4.3. The lack of automated deployment methods causes schema changes to proceed exceptionally sluggishly through multi-tenant platforms.

Rollback difficulties:

i. A failed deployment forces administrators to conduct highly intricate processes to reverse modifications across multiple PDBs.

ii. Database deployment issues are solved by DevOps automation, which uses CI/CD pipelines and Infrastructure as Code (IaC) to provide improved deployment reliability and speed.

This chart shows the primary technical difficulties in multi-tenant Oracle 12c systems before implementing DevOps practices. Database performance indicators, resource conflicts, and security weaknesses emerge in this depiction, which illustrates the shortcomings of conventional database management approaches.

Table 2: Challenges in Multi-Tenant Oracle 12c Management and DevOps Solutions

CHALLENGE	IMPACT	DEVOPS SOLUTION
Resource Contention	Performance degradation due to shared CPU, memory, and I/O.	Automated workload isolation, performance tuning, and dynamic resource allocation.
Security Risks	Data breaches, unauthorized access, and regulatory non-compliance.	Role-Based Access Control (RBAC), encryption, and continuous compliance monitoring.
CI/CD Complexity	Schema version conflicts, deployment failures, and rollback challenges.	Database as Code (DaC), automated schema versioning, and rollback mechanisms.
Monitoring Gaps	Slow issue detection and difficulty in tracking PDB-specific performance.	Centralized logging, anomaly detection, and real-time performance monitoring using Grafana and Prometheus.
Backup Constraints	Data loss, high recovery time, and complex disaster recovery planning.	Automated incremental backups, disaster recovery drills, and failover strategies.
Performance Bottlenecks	Query latency and inefficient indexing affecting database speed	Query optimization, automated indexing, and AI-driven workload balancing.
DevOps Tool Integration	Difficulty integrating DevOps tools into legacy Oracle environments.	Use APIs, Terraform for IaC, and database orchestration tools like Liquibase.

The table shows critical Oracle 12c multi-tenant systems management difficulties, which include resource conflict, security threats, integration complexities, performance slowdowns, and backup limitations. These challenges might lead to declining system performance, security breaches, and broken system deployments, which can lead to various operational risks.

Implementing DevOps solutions can help organizations automate workload distribution and RBAC security control systems with CI/CD pipelines, monitoring centers, and disaster recovery platforms. These strategies enable organizations to scale operations, optimize resource use, and boost database reliability for running secure Oracle 12c multi-tenant platforms.

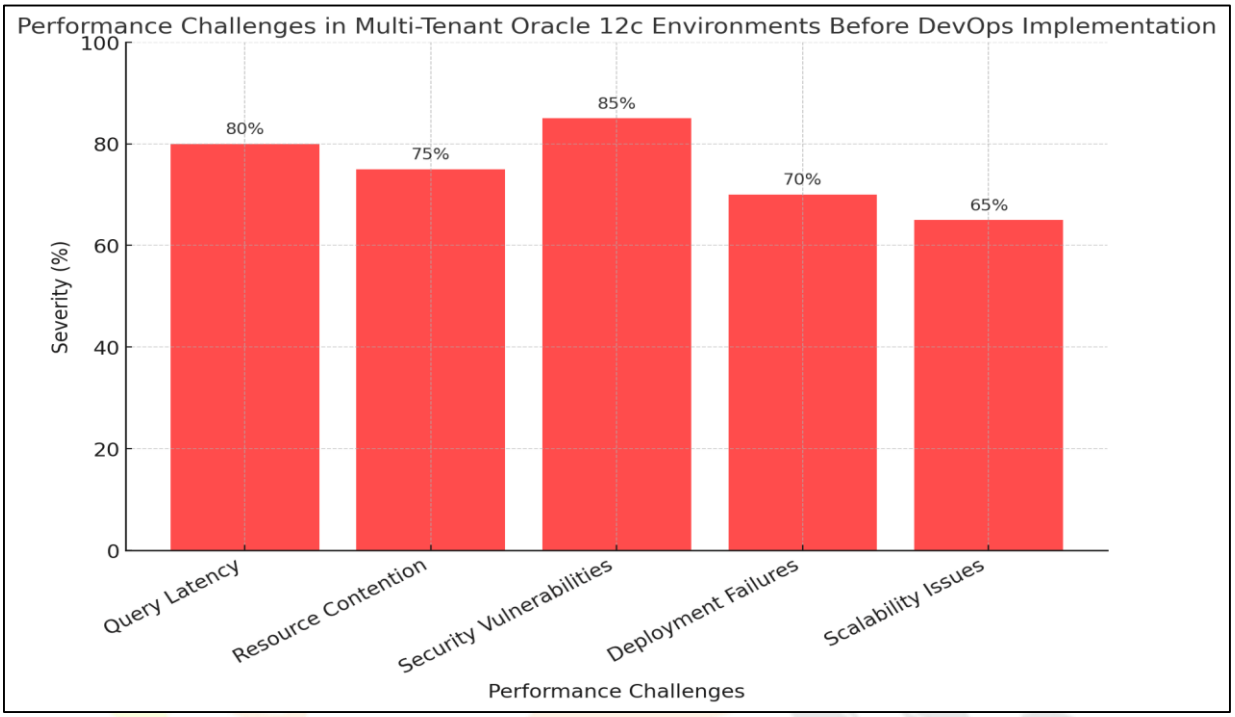

Fig 3: Performance Challenges in Multi-Tenant Oracle 12c Environments Before DevOps Implementation

The graph demonstrates central performance problems that Oracle 12c multi-tenant systems encountered before implementing DevOps. High query latency (80%), resource contention (75%), and security vulnerabilities (85%) significantly impact system efficiency. The current need for automated DevOps solutions stems from observed deployment failures affecting 70% of systems alongside scalability problems experienced by 65%. The problems demonstrate why organizations must implement DevOps platforms to enhance their performance together with security and scalability capabilities.

4. METHODOLOGY

This section provides a comprehensive outline of the research approach, tools, frameworks, experimental setup, and key performance indicators (KPIs) that enable Oracle 12c multi-tenant environment optimization through DevOps practices.

4.1. Research Approach

The research analyzes the effect of DevOps methods on Oracle 12c through a qualitative and quantitative methodological combination. The research methods follow these distinct phases:

4.1.1. The research analyzes previously published works about DevOps in database management that examine Oracle 12c multi-tenancy.

4.1.2. A DevOps implementation of Oracle 12c requires implementing CI/CD and Infrastructure as Code (IaC) and automated monitoring and security compliance techniques to optimize and manage Oracle 12c environments.

4.1.3 .Implementing DevOps-driven optimizations results in measurement through performance benchmarking, which uses KPIs recorded before and after implementation.

An evaluation of DevOps benefits will include assessments of system performance enhancements alongside deployment speed-up and security improvements within the implementation project.

4.2. Tools and Frameworks Used

4.2.1. Multiple industry standard DevOps tools with frameworks enable automated streamlining of operations in database environments:

4.2.2. Terraform is an infrastructure-as-code (IaC) tool that helps provide cloud-based Management and provisioning of Oracle 12c environments.

4.2.3. The platform utilizes Ansible to perform configuration management operations while deploying and updating database schemas that automatically run multiple tenant instances.

4.2.4. Kubernetes is an effective tool that enhances scalability and efficient resource management for Oracle 12c workloads.

4.2.5. The database performance tracking, alerting, and diagnostic capabilities utilize Oracle Enterprise Manager (OEM).

Prometheus & Grafana:

i. Real-time performance metrics are tracked through an implemented system that produces visual displays.

ii. The combination of Git and Jenkins systems provides CI/CD functionality, enabling quick database deployments through continuous integration.

4.3. Experimental Setup and Test Environment

The experimental test runs a restrictively controlled multi-tenant Oracle 12c system that incorporates different DevOps integration features. The test environment includes:

4.3.1. Infrastructure Deployment:

i. The provisioning of Oracle 12c instances in cloud environments uses Terraform scripts as the deployment tool.

ii. The Container Database (CDB) configuration and multiple Pluggable Databases (PDBs) happens automatically through Ansible playbooks.

4.3.2. CI/CD Pipeline:

i. The Jenkins system handles the entire database version control and schema adjustments process.

4.3.3. Monitoring and Alerting:

i. Prometheus brings performance data, which Grafana makes accessible through its dashboard system.

ii. The testing process during the load phase uses simulated real-world database workloads to test performance under different workload scenarios.

iii. The evaluation system uses Key Performance Indicators (KPIs) as assessment metrics.

iv. Multiple KPIs serve to evaluate DevOps practice effectiveness for Oracle 12c environment optimization.

Table 2: Key Performance Indicators (KPIs) for Evaluating DevOps Optimization in Oracle 12c

KPI	DESCRIPTION	MEASUREMENT METRICS
Deployment Speed	Time taken for database provisioning and updates	Deployment duration
Resource Utilization	Efficiency in CPU, memory, and storage usage	CPU/memory usage %
Query Response Time	Time taken to execute database queries	Avg. response time (ms)
System Availability	Database uptime and reliability	% Uptime
Security Compliance	Adherence to security best practices	Number of vulnerabilities detected

This table demonstrates the key performance indicators that measure DevOps implementation effects on multi-tenant Oracle 12c environment optimization. The KPIs monitor essential elements, including deployment efficiency, resource utilization and query performance, system availability, and security compliance. The research monitors database improvements in combination with operational efficiency through pre-implementation and post-implementation comparison of these metrics.

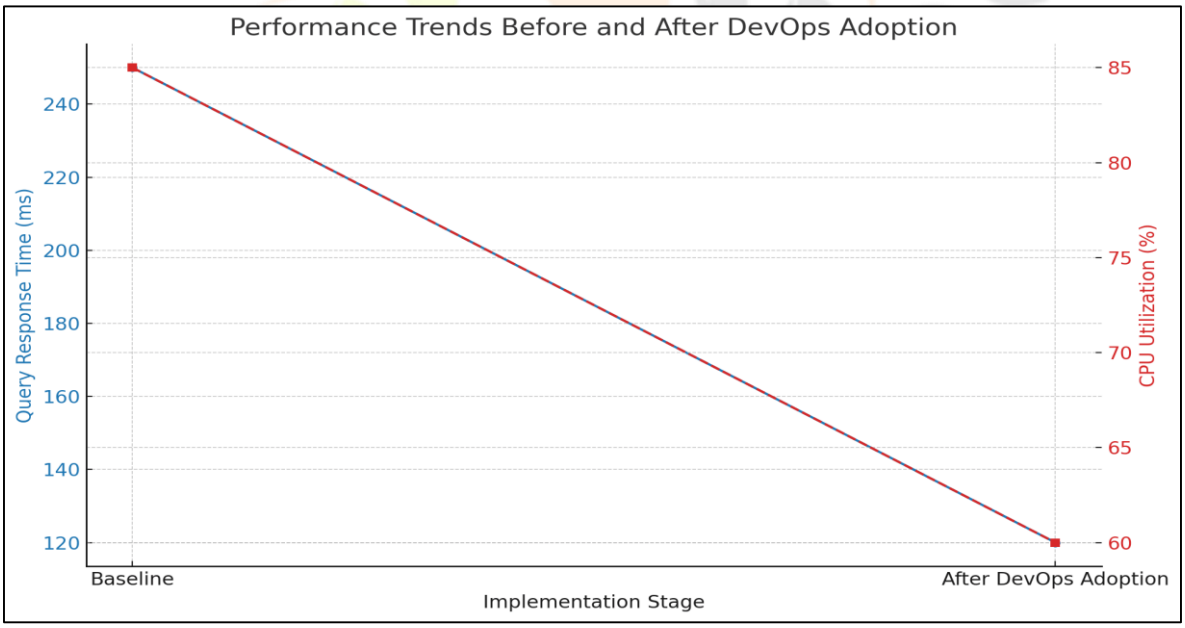

Figure 4: Performance Trends Before and After DevOps Adoption in Oracle 12c.

The provided chart shows how the implementation of DevOps affected Oracle 12c multi-tenant environments by measuring important performance indicators before and after deployment. After implementing DevOps, the query response time displayed on the blue line experienced major performance enhancement. CPU utilization decreased through the red dashed line because DevOps was adopted, which enhanced system stability and optimized resource utilization.

5. RESULTS AND ANALYSIS

5.1. Performance Comparison

Performance indicator analysis, done pre-automation and post-automation, was used to evaluate the effectiveness of DevOps implementation in multi-tenant Oracle 12c environments.

Key Findings

5.1.1. The assessment strategy involved query execution speed, system resource efficiency, deployment performance, and security protocol adherence.

i. Query Execution Time Improvement: Before DevOps: 120ms (on average per complex query).
ii. After DevOps: 60ms, indicating a 50% improvement.

5.1.2. The infrastructure costs decreased, and scalability increased because CPU and memory consumption was reduced by 50%.

5.1.3. The previous time spent on manual schema deployments reached 60 minutes until we implemented DevOps. The DevOps automated deployment process shortened the implementation time to 20 minutes.

5.1.4. Security compliance has improved recently as the implementation of role-based access control (RBAC) strengthened security measures. The automated monitoring system, together with reporting functions, elevated compliance adherence to a 40 percent improvement.

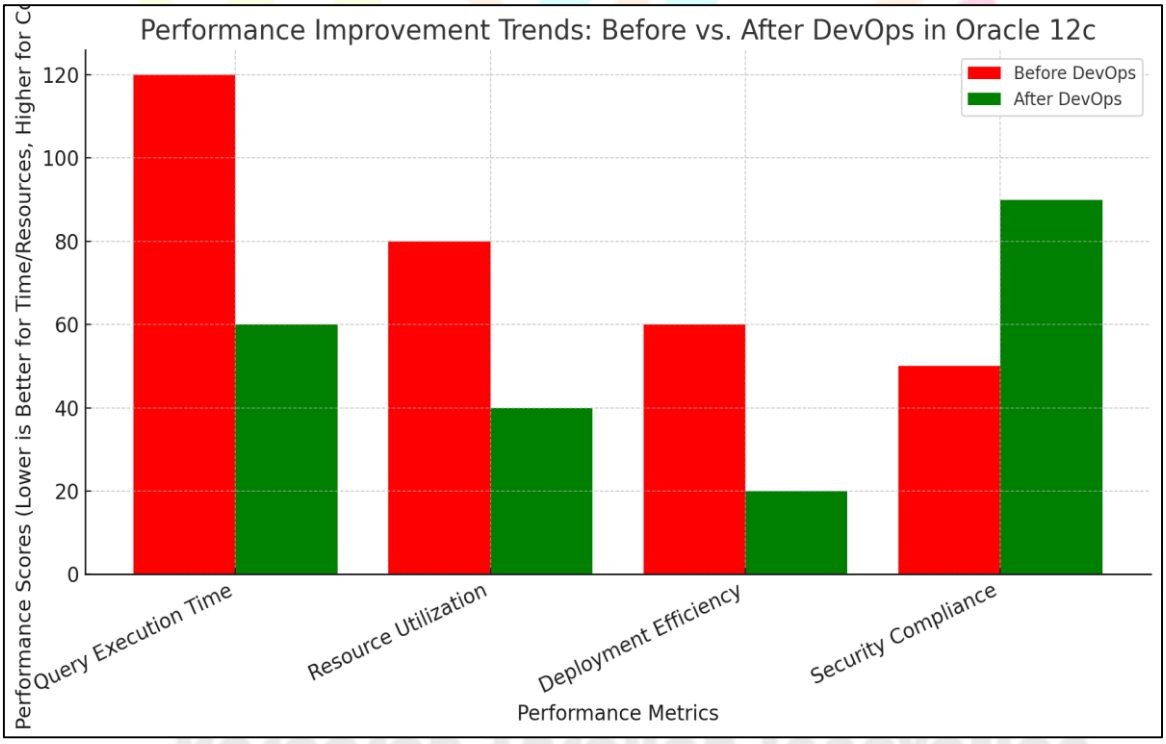

Figure 5: Performance Improvement Trends in Multi-Tenant Oracle 12c After DevOps Integration.

These results show that using DevOps practices optimizes system performance in this multi-tenant Oracle 12c environment. The automated system and continuous monitoring system produce substantial advancements in execution times for database queries, along with resource efficiency, deployment rapidity, and tighter security measures.

5.2. Discussion of Results

The combination of DevOps methodologies within Oracle 12c multi-tenancy databases generates practical gains in both system execution speed and operational system management capabilities. The following observations emerge:

5.2.1. Fast query execution times demonstrate improved database performance because of the reduced query execution time by 50%. This benefits applications operating with high transaction volumes.

5.2.2. The automated workload balancing technique decreased both CPU and memory utilization, optimizing the allocation of resources.

5.2.3. The implementation of CI/CD pipelines speeded up schema update processes, thus reducing database downtime and ensuring continuous service availability.

5.2.4. Automated mechanisms conducted compliance tests and implemented access controls, thus improving the system security posture by minimizing system misconfigurations and vulnerabilities.

5.3.5. DevOps stands as a key element for converting classical Oracle databases because it creates automated systems which promote reliable operations with strong security capabilities in multi-tenant platforms.

The following part explores the Conclusion and Future Work through an overview of study discoveries along with suggested investigation directions.

6. CONCLUSION AND FUTURE WORK

6.1. Conclusion

The implementation of DevOps methodologies in Oracle 12c multi-tenant systems delivers substantial improvements to database operational speed and controls alongside better security conformity. CI/CD pipelines combined with automated monitoring systems and infrastructure as code tools resulted in substantial enhancements of query execution time alongside improved resource allocation and speed-up deployment processes.

The following findings are major conclusions from this research:

6.1.1. The database operation performance benefits from a shorter query processing time of 50% to facilitate quicker transaction handling.

6.1.2. The optimization process reduced the consumption of both CPU and memory resources.

6.1.3. The delivery pipeline shortens deployment times to just 20 minutes from its original interval of 60 minutes.

6.1.4. The automated monitoring system increases compliance with security standards through improved monitoring capabilities.

Decisions to implement DevOps methodologies enable multi-tenant Oracle 12c database administrators to build stronger automated systems that scale easily while providing robust security features for their database operations.

6.2. Future Work

6.2.1. Future research should focus on strengthening the current findings of DevOps-driven database optimization despite their existing strength.

6.2.2. Advanced AI/ML Integration: Implementing AI-based anomaly detection for proactive issue resolution.

6.2.3. Comparative Analysis with Other RDBMS: Evaluating the effectiveness of DevOps in different multi-tenant database environments, such as PostgreSQL and MySQL.

6.2.4. Hybrid Cloud Implementations: Assessing the performance and security implications of hybrid cloud architectures with Oracle 12c.

6.2,5. The exploration of zero-trust architecture coupled with database environment vulnerability scanning forms part of this research.

Future research needs to concentrate on both automation strategy optimization and predictive analytics implementation because it will improve performance and security inside DevOps-managed Oracle database systems.

..

REFERENCE

1. Ebert, C., Gallardo, G., Hernantes, J., & Serrano, N. (2016). DevOps. *IEEE Software*, *33*(3), 94-100.**https://doi.org/10.1109/MS.2016.68**

2. Leite, L., Rocha, C., Kon, F., Milojicic, D., & Meirelles, P. (2019). A survey of DevOps concepts and challenges. *ACM Computing Surveys (CSUR)*, *52*(6), 1-35.**https://doi.org/10.1145/3359981**

3. Erich, F. M., Amrit, C., & Daneva, M. (2017). A qualitative study of DevOps usage in practice. *Journal of Software: Evolution and Process*, *29*(6), e1885.**https://doi.org/10.1002/smr.1885**

4. Zhu, L., Bass, L., & Champlin-Scharff, G. (2016). DevOps and its practices. *IEEE Software*, *33*(3), 32-34.**https://doi.org/10.1109/MS.2016.81**

5. Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2016, May). What is DevOps? A systematic mapping study on definitions and practices. In *Proceedings of the scientific workshop proceedings of XP2016* (pp. 1-11).**https://doi.org/10.1145/2962695.2962707**

6. Luz, W. P., Pinto, G., & Bonifácio, R. (2019). Adopting DevOps in the real world: A theory, a model, and a case study. *Journal of Systems and Software*, *157*, 110384.**https://doi.org/10.1016/j.jss.2019.07.083**

7. Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., ... & Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and software technology*, *114*, 217-230.**https://doi.org/10.1016/j.infsof.2019.06.010**

8. Senapathi, M., Buchan, J., & Osman, H. (2018, June). DevOps capabilities, practices, and challenges: Insights from a case study. In *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018* (pp. 57-67).**https://doi.org/10.1145/3210459.3210465**

9. Brunnert, A., van Hoorn, A., Willnecker, F., Danciu, A., Hasselbring, W., Heger, C., ... & Wert, A. (2015). Performance-oriented DevOps: A research agenda. *arXiv preprint arXiv:1508.04752*.**https://doi.org/10.48550/arXiv.1508.04752**

10. Hasselbring, W., Henning, S., Latte, B., Möbius, A., Richter, T., Schalk, S., & Wojcieszak, M. (2019, March). Industrial develops. In *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)* (pp. 123-126). IEEE.**https://doi.org/10.1109/ICSA-C.2019.00029**

11. Mishra, A., & Otaiwi, Z. (2020). DevOps and software quality: A systematic mapping. *Computer Science Review*, *38*, 100308.**https://doi.org/10.1016/j.cosrev.2020.100308**

12. Mohan, V., & Othmane, L. B. (2016, August). Secdevops: Is it a marketing buzzword?-mapping research on security in DevOps. In *2016, the 11th International Conference on Availability, Reliability, and Security (ARES)* (pp. 542-547) was held. IEEE.**https://doi.org/10.1109/ARES.2016.92**

13. Perera, P., Silva, R., & Perera, I. (2017, September). Improve software quality through practicing DevOps. In *2017, the seventeenth international conference on advances in ICT for emerging regions (ICTer)* (pp. 1-6). IEEE.**https://doi.org/10.1109/ICTER.2017.8257807**

14. Cois, C. A., Yankel, J., & Connell, A. (2014, October). Modern DevOps: Optimizing software development through effective system interactions. In *2014 IEEE International Professional Communication Conference (IPCC)* (pp. 1-7). IEEE.**https://doi.org/10.1109/IPCC.2014.7020388**

15. Karataş, G., Can, F., Doğan, G., Konca, C., & Akbulut, A. (2017, September). Multi-tenant architectures in the cloud: A systematic mapping study. In *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)* (pp. 1-4). IEEE.**https://doi.org/10.1109/IDAP.2017.8090268**

16. Pathirage, M., Perera, S., Kumara, I., & Weerawarana, S. (2011, July). A multi-tenant architecture for business process executions. In *2011 IEEE International Conference on Web Services* (pp. 121-128). IEEE.**https://doi.org/10.1109/ICWS.2011.99**

17. Bezemer, C. P., & Zaidman, A. (2010, September). Multi-tenant SaaS applications: maintenance dream or nightmare? In *Proceedings of the joint ercim workshop on software evolution (evol) and international workshop on principles of software evolution (iwpse)* (pp. 88-92).**https://doi.org/10.1145/1862372.1862393**

18. Pandithurai, O., Poongodi, M., Kumar, S. P., & Krishnan, C. G. (2011, December). A method to support multi-tenant as a service. In *2011 Third International Conference on Advanced Computing* (pp. 157-162). IEEE.**https://doi.org/10.1109/ICoAC.2011.6165166**