Migrating SQL Server from On-Premise to Infrastructure as a Service on Cloud

Suhas Hanumanthaiah

Independent Research

Abstract:
The migration of Microsoft SQL Server databases from on-premise environments to cloud-based Infrastructure as a Service (IaaS) platform has become a strategic imperative for modern enterprises seeking agility, scalability, and cost-efficiency. This transformation leverages cloud computing's pay-as-you-go model, enabling faster deployment, global reach, and enhanced resource optimization. This paper presents a comprehensive exploration of migration strategies, automation frameworks, and DevOps integration practices to support large-scale SQL Server transitions to leading cloud IaaS providers such as Microsoft Azure, Amazon Web Services (AWS), and Google Cloud Platform (GCP). The analysis includes a comparison of native migration tools (e.g., Azure Database Migration Service, AWS DMS, GCP DMS), infrastructure-as-code solutions (Terraform, PowerShell), and database DevOps tools (Liquibase, dbForge, Redgate) to automate, streamline, and secure the migration lifecycle. Furthermore, the study examines the integration of SQL Server database changes into CI/CD pipelines, significantly reducing manual intervention and enabling consistent deployments across environments. Emphasis is placed on performance benchmarking, licensing models, cost optimization, and high availability architectures. Security concerns, including encryption, access control, and compliance requirements, are also thoroughly addressed. This research aims to equip IT leaders, database administrators, and enterprise architects with practical insights and frameworks to execute seamless, resilient, and cost-effective SQL Server migrations while maintaining operational excellence and business continuity.

1. INTRODUCTION

The migration of SQL Server databases from on-premise infrastructure to cloud-based Infrastructure as a Service (IaaS) represents one of the most significant technological transformations organizations undertake in their digital journey. Cloud computing has emerged as the backbone of modern economy by offering subscription-based services anytime, anywhere following a pay-as-you-go model, which has instigated shorter establishment times for start-ups, creation of scalable global enterprise applications, better cost-to-value associativity for scientific high-performance computing, and different invocation/execution models for pervasive ubiquitous applications [1].

Taking on-premises SQL Server databases to the cloud opens a world of benefits such as scalability, flexibility, and often, reduced costs [2]. However, the journey requires meticulous planning and execution [2]. This comprehensive guide explores the strategic, technical, and operational aspects of migrating SQL Server workloads to cloud IaaS platforms, with particular emphasis on automation tools and CI/CD integration for large-scale enterprise migrations.

2. UNDERSTANDING IAAS FOR SQL SERVER MIGRATION

2.1 IaaS Fundamentals and SQL Server Deployment Options

Infrastructure as a Service offers a flexible environment to deploy SQL Server by provisioning virtual machines (VMs) where users have complete control over operating systems, middleware, and database configurations. This environment grants the freedom to optimize SQL Server instances for specific workloads, customize configuration settings, and apply preferred patching or security measures. However, this flexibility

comes with increased administrative overhead, as users are responsible for OS updates, backups, monitoring, and scaling strategies.
Within IaaS models, provisioning virtual machines entails selecting appropriate instance types, balancing CPU, memory, and storage configurations to meet the performance requirements of SQL Server workloads. Administrative control over the environment enables businesses to maintain legacy applications or complex database configurations that might not be supported in managed services.
Licensing for SQL Server in IaaS presents complexity, often dictated by the cloud provider's policies, licensing mobility terms, and usage patterns. Licensing options include Bring Your Own License (BYOL) or pay-as-you-go bundled licenses, each with cost implications depending on workload duration and scale.
Cloud providers such as AWS, Azure, and GCP offer preconfigured SQL Server images to simplify deployment within their IaaS offerings. This approach accelerates database provisioning while allowing extensive control. Nonetheless, organizations must carefully consider the balance between control and operational responsibility [20].

2.2 Performance Characteristics and Scalability in IaaS

The performance of SQL Server on IaaS depends heavily on the underlying hardware, VM type, and cloud provider infrastructure characteristics. Hardware heterogeneity, a common phenomenon in cloud environments, may lead to performance variability, necessitating benchmarking and ongoing monitoring to ensure consistency, especially for mission-critical databases. Studies have demonstrated that multitenancy and resource contention can introduce unpredictable latency, warranting careful capacity planning and workload distribution [21].
To address scalability and load balancing, dynamic resource management techniques including horizontal scaling of VMs, load balancers, and cluster configurations can be implemented. Advanced algorithms for dynamic load balancing in IaaS environments optimize VM selection and server utilization, thereby enhancing throughput and reducing latency while maintaining reliability across distributed deployments [22].
Real-world performance assessments emphasize that leveraging optimized database access methods and connection pooling, as well as adopting asynchronous programming models, can substantially improve SQL Server responsiveness in cloud-hosted environments. These enhancements directly impact throughput and CPU efficiency, which are critical for sustaining enterprise-grade workloads on cloud IaaS platforms [23].

2.3 Cost Implications and Optimization Strategies with IaaS

Cost remains a fundamental consideration in IaaS for SQL Server deployments. Total Cost of Ownership (TCO) models reveal that while cloud-based IaaS can offer significantly lower upfront capital expenditures compared to on-premises infrastructure, operational expenses—including licensing, storage, and bandwidth—may accumulate rapidly depending on usage patterns and data volumes [20].
Cost optimization strategies include leveraging reserved instances, which offer discounted rates in exchange for commitment to longer-term usage, and optimizing licensing by selecting the appropriate SQL Server edition or considering alternatives such as PostgreSQL to reduce expenses. Additionally, choice of operating system, such as preferring Linux-based environments over Windows, when possible, can influence licensing and support costs.
Moreover, continuous cost management requires careful analysis of workload patterns and rightsizing of instances to avoid over-provisioning. Tools that provide price-performance recommendations and identify over-provisioned resources have demonstrated tangible savings by enabling more efficient capacity allocation, thus balancing cost against required performance levels [24].

3. PLATFORM AS A SERVICE (PAAS) FOR SQL SERVER

3.1 PaaS Characteristics in Cloud SQL Server Deployment

Platform as a Service abstracts infrastructure management further than IaaS by providing fully managed SQL Server environments where the cloud provider assumes responsibility for OS patching, database backups, and

automatic scaling. This paradigm shifts operational responsibilities from database administrators to the provider, enabling development teams to focus on application logic and innovation.
The managed nature of PaaS environments includes automated software updates, built-in high availability configurations, and integrated monitoring. PaaS reduces the complexity involved in provisioning and maintaining database servers and enables faster time-to-market for applications relying on SQL Server backend services.
This abstraction often translates to improved reliability and simplified administration, though it requires trust in the provider's controls and may impose limitations on customization and access to certain system-level features. Businesses adopting PaaS benefit from elastic scaling features that can adjust compute and storage resources in response to workload fluctuations—ideal for web applications with variable user activity [24],[25].

3.2 Platform Offerings: Azure SQL Database, Amazon RDS, Google Cloud SQL

Each of the leading cloud providers offers PaaS solutions for SQL Server with distinct feature sets and integration capabilities.
Azure SQL Database and Azure SQL Managed Instance provide fully managed relational database services, incorporating transparent data encryption, advanced threat protection, and AI-driven performance tuning. These services support multi-model data and elastic pools for cost-efficient resource sharing across databases.
Amazon Relational Database Service (RDS) for SQL Server automates backups, replication, scaling, and failover, offering a range of licensing options, including BYOL. It integrates closely with AWS monitoring and security services, providing scalable performance suitable for various workloads [20].
Google Cloud SQL offers a straightforward managed SQL Server service optimized for smaller-scale deployments. While it supports core SQL Server features, it lacks some advanced capabilities present in Azure and AWS offerings. However, Cloud SQL integrates seamlessly with Google's AI and monitoring suite, providing value-added analytics and operational insights [26],[21].

3.3 Performance, Scalability, and Limitations in PaaS

PaaS services offer elastic scaling, enabling databases to adjust resource allocation dynamically. However, for high-intensity transactional workloads or large-scale data warehousing, PaaS solutions may exhibit scaling limitations compared to customizable IaaS environments.
Latency and transactional throughput in PaaS are largely dependent on the internal architecture of the managed service. While managed SLAs assure uptime and resilience, some applications with stringent performance requirements might find platform-imposed limitations restrictive.
Moreover, the SLA-backed business continuity features such as automated failover and geo-replication contribute to operational stability, yet understanding the nuances of these mechanisms is vital to architect robust systems.
Cutting-edge SQL Server enhancements leveraging AI for autonomous management further improve reliability and optimization within PaaS, signaling an ongoing evolution toward self-managing databases [21], [24].

4. CLOUD PROVIDER OPTIONS FOR SQL SERVER IAAS

4.1 Azure's IaaS for SQL Server: Capabilities and Innovations

Azure offers extensive native support for SQL Server in its IaaS portfolio, facilitating hybrid cloud integration using technologies like Azure Arc, which enables management of on-premises, multi-cloud, and edge environments. This solution bridges traditional database management with cloud operations, facilitating unified control across deployment types.
Azure's IaaS offerings benefit from native compatibility with SQL Server's latest features, including containerization support and the accommodation of microservices-based architectures. These advances pave the way for highly scalable, modular applications that leverage cloud elasticity with SQL Server's reliability.

Licensing models on Azure accommodate hybrid arrangements, allowing BYOL while supporting features like Always Encrypted and Row-Level Security that enhance security in cloud environments.

4.2 AWS IaaS for SQL Server: Strengths and Cost Optimization

AWS provides SQL Server instances through EC2 with a variety of instance types optimized for different workloads. The lift-and-shift migration strategy, initially deploying existing databases to EC2 instances, allows businesses to rapidly migrate without significant code alterations. Gradual adoption of PaaS elements, such as Elastic Beanstalk or Lambda, can follow to reduce infrastructure management burdens.

AWS cost optimization strategies are robust, including the use of reserved instances and spot pricing, as well as licensing optimization through Windows Server and SQL Server-based discounts. Furthermore, running Linux-based applications over AWS can reduce licensing charges where possible.

Understanding AWS-specific features and ecosystem capabilities is paramount to realize these cost benefits, underscoring the need for detailed pre-migration analysis and continuous financial control to avoid unexpected expenditures [20], [24].

4.3 GCP’s IaaS Capabilities for SQL Server Deployment

Google Cloud Platform supports SQL Server workloads on Compute Engine virtual machines. Despite being less extensive compared to Azure and AWS in terms of specialized SQL Server services, GCP emphasizes simplicity and integrates cloud-native AI and monitoring tools to enhance operational insights.

Performance predictability on GCP’s IaaS infrastructure has been subject to variability depending on the selected VM types and multitenant configurations, requiring users to benchmark workloads carefully.

Tooling around SQL Server on GCP is relatively nascent, with fewer ready integrations than AWS or Azure, yet it holds appeal for organizations prioritizing AI-driven analytics and wishing to diversify cloud deployments [21], [26], [27].

Feature	**Azure SQL VM**	**AWS EC2 SQL Server**	**Google Cloud SQL Server**
Licensing Model	BYOL or Pay-as-you-go	BYOL or License Included	Pay-as-you-go
High Availability	Always On Availability Groups	Multi-AZ deployments	Regional persistent disks
Backup & Recovery	Azure Backup Service	Automated backups	Point-in-time recovery
Monitoring	Azure Monitor	CloudWatch	Cloud Monitoring
Security	Azure Security Center	AWS Security Hub	Google Cloud Security
Cost Optimization	Azure Hybrid Benefit (up to 85% savings)	Reserved Instances	Committed use discounts
Migration Tools	Azure Migrate, DMA, DMS	AWS DMS, SCT	Database Migration Service
Automation	PowerShell, ARM Templates	CloudFormation, CLI	Deployment Manager

Table 1.1 Comparison of Cloud Providers Features for IaaS deployment of SQL Server

5. SECURITY CONSIDERATIONS ACROSS IAAS, PAAS, AND SAAS FOR SQL SERVER

5.1 Data Security and Encryption Mechanisms

Data security in cloud SQL Server deployments mandates robust encryption strategies. Providers implement symmetric algorithms such as AES and DES for encrypting large volumes of data efficiently, complemented by asymmetric cryptography methods like RSA, ECC for secure key exchange and digital signatures.
Protecting data both at rest and in transit is vital due to the vulnerabilities borne from remote storage. Cloud architectures adopt encryption at multiple layers, combining transport layer security with disk and database encryption to mitigate unauthorized access.
Continuous research emphasizes improving encryption algorithms and secure key management practices to address evolving threat landscapes and computational attacks [28], [29].

5.2 Identity Management, Access Controls, and Compliance

Effective identity management and access control frameworks are foundational in cloud security, including Role-Based Access Control (RBAC), federated identity systems, and stringent multi-tenancy controls. Cloud providers enforce shared responsibility models, delineating security duties between user organizations and providers.
SLA commitments reflect these models, with providers guaranteeing uptime and data protection, while customers manage data governance and credential security.
Additionally, compliance with sectoral standards (e.g., GDPR, HIPAA, PCI DSS) is facilitated through cloud-based auditing, data residency controls, and security certifications, allowing enterprises to meet regulatory obligations across jurisdictions [30],[31].

5.3 Security Risks and Mitigation Strategies in Cloud SQL Server

Cloud deployments inherently expose systems to risks such as unauthorized data access, injection attacks, or Distributed Denial of Service (DDoS). Implementing Intrusion Detection Systems (IDS) and anomaly detection mechanisms enhances proactive threat identification and response capabilities.
Advanced application delivery services secure data flow and provide cryptographic safeguards to ensure data confidentiality, integrity, and availability. Layered security architectures integrating encryption, IDS, and continuous monitoring form a robust defense posture for SQL Server workloads in cloud environments [32], [33].

6. PERFORMANCE, SCALABILITY, AND RELIABILITY ANALYSIS

6.1 Measuring and Benchmarking SQL Server Performance on Cloud Models

Evaluating SQL Server performance in cloud environments involves measuring metrics such as throughput, latency, reliability, and availability under varied workloads. Load balancing algorithms dynamically distribute query loads across VMs to optimize resource utilization and reduce bottlenecks.
Performance variability introduced by heterogeneous hardware and multitenancy poses challenges, necessitating detailed benchmarking and statistical analysis to identify optimal configurations.
Database access technologies, including different ORM layers or direct ADO.NET usage, further impact query efficiency and throughput in cloud-deployed SQL Server instances [21], [22].

6.2 Scalability Features Across Models and Providers

PaaS models generally provide automated scaling capabilities, achieving elastic resource allocation without human intervention. Conversely, IaaS requires manual or scripted scaling, giving granular control at the expense of operational complexity.
Horizontal scaling strategies, including read replicas and sharding, are supported variably across providers and deployment models, facilitating load distribution and resilience.
Cloud-native architectural patterns, such as microservices and containers, enhance scalability and enable rapid scaling, supported by cloud features tailored towards container orchestration and distributed databases [25], [23].

6.3 High Availability, Disaster Recovery and SLA Guarantees

High Availability (HA) is achieved through multi-AZ deployments, automated failover, and geo-replication to ensure near-zero downtime in critical SQL Server workloads. Backup and restore services integrated with managed platforms facilitate rapid disaster recovery.
Service Level Agreements (SLAs) provided by cloud vendors define expected availability and penalties for downtime, influencing enterprise risk management strategies.
Assessment of these features is essential for workload-critical applications demanding guaranteed uptime and rapid recovery capabilities [24].

7. COST MODELING AND TOTAL COST OF OWNERSHIP (TCO)

7.1 TCO Factors for IaaS, PaaS, and SaaS SQL Server Deployments

TCO analysis covers infrastructure expenses, licensing fees, personnel costs for operational overhead, and maintenance expenditures. Flexibility in IaaS can translate to higher administrative expenses, whereas PaaS and SaaS reduce management burdens but may have higher subscription fees.
Data storage fees, data transfer charges, and backup costs also influence overall TCO, particularly for data-intensive SQL Server applications.
Balancing cost with desired control and performance levels requires accurate modeling and consideration of workload duration, data growth, and peak resource utilization [20], [24].

7.2 Cost Optimization Strategies Across Cloud Providers

Effective strategies include commitment to reserved or spot instances, leveraging auto-scaling to align resources with demand, and choosing the right licensing approach (BYOL vs pay-as-you-go).
Organizations may also consider open-source alternatives or hybrid cloud configurations to reduce cost exposure while maintaining operational flexibility.
Continuous monitoring and rightsizing tools offered by providers assist in identifying cost drains and optimizing expenditures [24], [34].

7.3 Financial Implications of Migration and Vendor Lock-In

Initial cloud migration entails capital investment in analysis, refactoring, and potential downtime costs. Vendor lock-in risks can be mitigated by multi-cloud and hybrid strategies facilitating workload portability and resilience.
Conducting risk assessments focusing on financial exposure, contractual obligations, and exit strategies is prudent during cloud adoption planning [13], [26].

8. MIGRATION STRATEGIES AND APPROACHES

8.1. Lift-and-Shift Migration

The lift-and-shift approach involves moving existing SQL Server instances to cloud virtual machines with minimal modifications. This strategy is ideal for organizations that need to maintain existing configurations and have specific OS-level dependencies.

8.2. Hybrid Migration Approaches

Organizations can maximize existing investments in on-premises licenses by taking them to the cloud with Azure Hybrid Benefit and combine them with reserved instances for up to 85 percent savings [5]. Additionally, they can explore dev/test pricing, and enjoy up to three free years of Extended Security Updates [5].

8.3. Assessment and Planning Phase

Azure Migrate is designed to help identify and assess SQL data on VMware [3]. Azure Migrate offers recommendations for Azure SQL deployments, as well as monthly cost estimates and recommendations for target sizing [3].

9. MIGRATION TOOLS AND TECHNOLOGIES

9.1. Native SQL Server Tools

Traditional backup and restore methods remain fundamental for SQL Server migrations. These tools provide reliable, well-understood migration paths with minimal learning curves for database administrators.

9.2. Cloud-Specific Migration Tools

9.2.1. Azure Migration Tools

Data Migration Assistant is a desktop tool that can help achieve single-database SQL Server migrations to Azure SQL Database, allowing migration of both data and schema. [3] You can install Data Migration Assistant on an on-prem server or on a local machine with connectivity to the source databases.

Azure Database Migration Service (DMS) automatically migrates on-premises SQL Server instances to the managed Azure SQL Database service. [3] You can use it to automatically migrate using PowerShell, or manually.

9.2.2. AWS Migration Tools

AWS Database Migration Service supports SQL Server on IaaS or Azure SQL Database as PaaS as a migration source, though downtime will occur as it does not support CDC mode [8].

To migrate SQL Server databases from Azure SQL Managed Instance to AWS with minimal downtime, you can also leverage CloudBasic from the AWS Marketplace [9].

9.2.3 GCP Migration Tools

GCP Database Migration Service (DMS) is a managed service that simplifies the migration of SQL Server databases to Google Cloud managed database services like Cloud SQL. GCP Database Migration Service supports continuous replication to migrate SQL Servers from On prem or other cloud providers into GCP Cloud SQL.

To migrate over SQL Server VMs over to GCP Compute Engine it is recommended to use Lift and Shift model of rehosting the VM on cloud.

Tool	Best For	Supported Sources	Supported Targets	Automation Level	Cost
Azure Database Migration Service	Azure migrations	SQL Server, MySQL, PostgreSQL	Azure SQL DB, MI, VM	High	Free
AWS Database Migration Service	Heterogeneous migrations	15+ database engines	AWS RDS, Aurora, EC2	High	Pay-per-use
GCP Database Migration Service	Heterogeneous migrations	Multiple database engines	GCP Cloud SQL, Alloy DB	High	Pay-per-use
Data Migration Assistant	Assessment and single DB	SQL Server	Azure SQL Database	Medium	Free

SQL Server Migration Wizard	Simple migrations	SQL Server	SQL Server	Low	Free
dbForge Studio	Development environments	SQL Server	SQL Server, Azure SQL	Medium	Commercial
Liquibase	Schema management	Multi-database	Multi-database	High	Open source/Commercial
Flyway	Version control	Multi-database	Multi-database	High	Open source/Commercial

Table 2: Migration Tools Comparison Chart

10. AUTOMATION TOOLS FOR LARGE-SCALE SERVER MIGRATIONS

10.1. Enterprise Automation Platforms

For organizations managing hundreds or thousands of SQL Server instances, automation becomes critical for successful migration execution. Cloud providers create a vendor-locked-in environment by offering proprietary and non-standard APIs, resulting in a lack of interoperability and portability among clouds. [11] To overcome this deterrent, solutions must be developed to exploit multiple clouds efficaciously.

10.2. Infrastructure as Code (IaC) Tools

Infrastructure deployment automation of Azure SQL can be done with PowerShell and CLI. [4] Useful examples can be found in the Azure PowerShell samples for Azure SQL Database and Azure SQL Managed Instance article.

10.2.1. Terraform for Database Migrations

Terraform can be used to create a CI/CD pipeline to migrate a Microsoft SQL Server database that's on premises, on a VM, or in another cloud environment to Amazon RDS on AWS [12]. Service automation via third party tools is also possible, including Azure SQL Managed Instance Terraform commands [4].

10.2.2. PowerShell and CLI Automation

Infrastructure deployment automation of Azure SQL can be done with PowerShell and CLI [4], providing scriptable interfaces for automating repetitive migration tasks across multiple database instances.

10.3. Database DevOps Tools

Leading database DevOps tools include DBmaestro for compliance and auditing, dbForge DevOps Automation for SQL Server for SQL Server CI/CD, Bytebase for database governance, and Toad DevOps Toolkit for continuous integration.

Additional tools include Flyway for simple database migrations, SQL Toolbelt Essentials for SQL Server management, ApexSQL for SQL Server environments, and Liquibase for database change automation.

Tool	Primary Focus	SQL Server Support	CI/CD Integration	Pricing Model
DBmaestro	Compliance & Auditing	Yes	Advanced	Enterprise
dbForge DevOps Automation	SQL Server CI/CD	Excellent	Native	Commercial
Bytebase	Database Governance	Yes	Good	Freemium
Toad DevOps Toolkit	Continuous Integration	Excellent	Good	Commercial
Liquibase	Change Management	Yes	Excellent	Open source/Commercial
Redgate SQL Change Automation	Automated Deployments	Excellent	Excellent	Commercial

Table 3: Database DevOps Tools Comparison

11. CI/CD INTEGRATION FOR DATABASE MIGRATION

11.1. Continuous Integration for Database Changes

A continuous integration and continuous deployment (CI/CD) pipeline is a series of established steps that developers must follow in order to deliver a new version of software. [14] CI/CD pipelines are a practice focused on improving software delivery throughout the software development life cycle via automation.
By automating CI/CD throughout development, testing, production, and monitoring phases of the software development lifecycle, teams are able to develop higher quality code, faster and more securely. [14] Automated testing also allows dependencies and other issues to be identified earlier in the software development lifecycle, saving time later.

11.2. Azure DevOps for Database CI/CD

You can use Azure DevOps Continuous Integration (CI) and Deployment (CD) Pipelines to fully embed automation within your Infrastructure-as-Code practices. [4] Building your database models and scripts can also be integrated through Database Projects with Visual Studio Code or Visual Studio. The use of Azure DevOps CI/CD pipelines will enable deployment of your Database Projects to an Azure SQL destination of your choice.
To organize the CI process, you'll need a GitHub account where you can upload the sample project to the repository, an account at Azure DevOps, and the dbForge extension installed. [15] You can select Browse marketplace, enter dbForge in the search box, select the dbForge extension from a list of search results, and install it.

11.3. Database Schema Management in CI/CD

A tool like Liquibase gets rid of the separate, manual effort, and allows for a consistent artifact that can be deployed through the pipeline. [16] Automating database change management involves systematically facilitating database schema and data changes across the development lifecycle. This practice ensures that database modifications are consistently applied, tracked, and documented, facilitating seamless migrations and deployments. Through automation, teams can streamline the process of integrating database changes into their CI/CD pipelines, reducing manual effort, mitigating risks associated with manual errors, and enhancing collaboration among development, operations, and database professionals.

11.4. CI/CD Pipeline Components

Tools like Liquibase or Flyway handle database migrations automatically. [17] The pipeline might produce an artifact (like a zip file with SQL scripts). This artifact is versioned, ensuring the same migrations are used throughout the environments. The pipeline can automatically deploy to a staging environment.
Deployments should include monitoring steps. [17] Teams watch logs, error rates, or data integrity checks. If something fails, the pipeline can run a rollback script. Tools like Liquibase store rollback instructions.

11.5. Automated Testing in Database CI/CD

When configuring database CI/CD tasks, you can specify folders that store update scripts. [15] Unit tests comprise conventional SQL scripts that are stored separately from the database. To add unit tests to the database, you will need the dbForge DevOps Automation for SQL Server Execute task, that executes SQL scripts from a folder or a file.

12. BEST PRACTICES AND CONSIDERATIONS

12.1. Pre-Migration Planning

Successful large-scale SQL Server migrations require comprehensive planning that addresses technical, operational, and business requirements. The Business and technical evaluation section covers cost saving, licensing, minimizing migration risk, business continuity, security, workloads and architecture, performance and similar business and technical evaluation questions [4].

12.2. Security and Compliance

With built-in security controls, layers of protection, and advanced threat protection, organizations can rest assured that their data is stored in a more secured and compliant environment [5]. This is particularly important for enterprises with strict regulatory requirements.

12.3. Performance Optimization

Organizations can ensure peak productivity with evergreen SQL offerings that never need to be patched or upgraded, and AI-powered features to optimize performance and security [5]. This reduces ongoing maintenance overhead while improving performance.

12.4. Cost Management

Organizations can maximize existing investments in on-premises licenses by taking them to the cloud with Azure Hybrid Benefit and combine them with reserved instances for up to 85 percent savings [5]. Proper cost planning is essential for large-scale migrations.

12.5. Continuous Monitoring and Optimization

Database DevOps has come of age and is seen as a key technical practice which can contribute to the successful implementation of DevOps, thereby eliminating the database bottleneck and having the releases faster and easier [18].

12.6. Risk Mitigation Strategies

Continuous delivery for database migrations is often a manual process. [19] Adopting continuous integration and continuous delivery (CI/CD) for database migrations can help automate and reduce risks.

13. IMPLEMENTATION ROADMAP

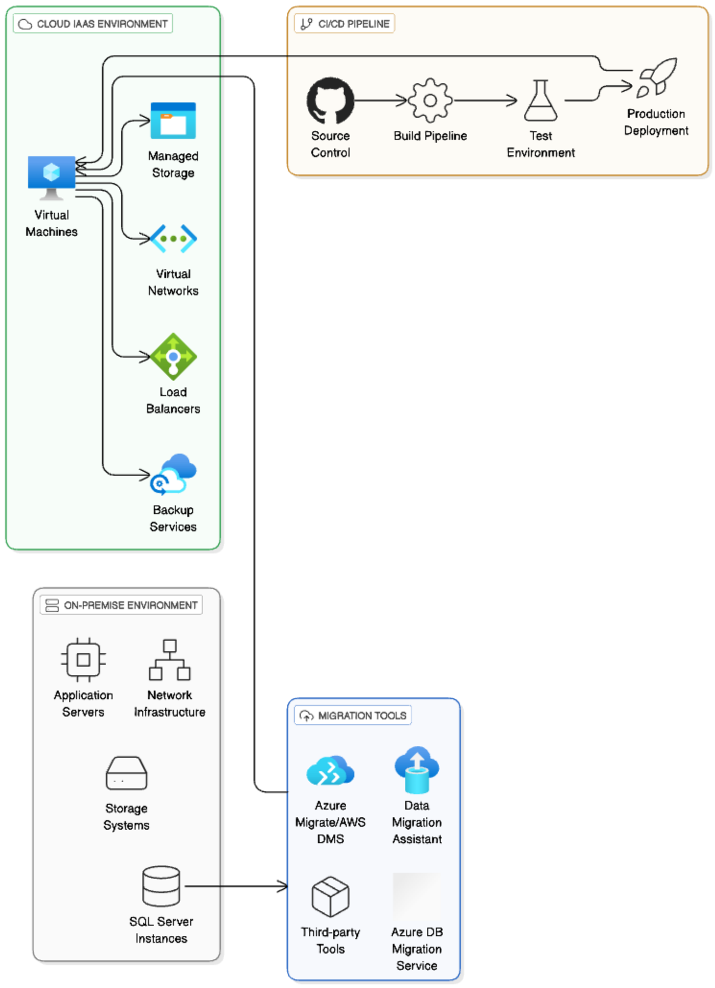

Fig. 1: Recommended Migration Architecture Diagram

Phase 1: Assessment and Planning
- Inventory existing SQL Server instances
- Assess application dependencies
- Evaluate network connectivity requirements
- Determine compliance and security requirements

Phase 2: Tool Selection and Setup
- Choose appropriate migration tools based on scale and requirements
- Set up CI/CD pipelines for database changes
- Configure monitoring and alerting systems
- Establish backup and recovery procedures

Phase 3: Pilot Migration
- Select pilot databases for initial migration
- Test migration procedures and tools
- Validate performance and functionality
- Refine processes based on lessons learned

Phase 4: Large-Scale Migration
- Execute migrations in batches
- Monitor performance and system health

- Implement automated rollback procedures
- Provide ongoing support and optimization

14. CONCLUSION

Migrating SQL Server from on-premise to cloud IaaS represents a significant transformation that requires careful planning, appropriate tooling, and robust automation. Azure is the cloud service that knows SQL Server migration best [5], while AWS and other cloud providers offer compelling alternatives with their own unique advantages.

The key to successful large-scale migrations lies in leveraging automation tools and CI/CD pipelines that can handle the complexity and scale of enterprise database environments. By automating CI/CD throughout development, testing, production, and monitoring phases of the software development lifecycle, teams are able to develop higher quality code, faster and more securely [14].

Through automation, teams can streamline the process of integrating database changes into their CI/CD pipelines, reducing manual effort, mitigating risks associated with manual errors, and enhancing collaboration among development, operations, and database professionals [16].

Organizations embarking on this journey should focus on comprehensive planning, appropriate tool selection, risk mitigation strategies, and continuous optimization to ensure successful migration outcomes. The investment in proper automation and CI/CD integration will pay dividends in reduced operational overhead, improved reliability, and enhanced ability to respond to changing business requirements.

As cloud computing continues to evolve, recent technological developments and paradigms such as serverless computing, software-defined networking, Internet of Things, and processing at network edge are creating new opportunities for computing, though they also pose several challenges that need new approaches and research strategies [1]. Organizations that establish strong foundations in database migration and automation will be well-positioned to take advantage of these emerging opportunities.

REFERENCES:

[1] R. Buyya et al., "A Manifesto for Future Generation Cloud Computing," Association for Computing Machinery, 2018. https://doi.org/10.1145/3241737
[2] T. D. Hub, "Migrating SQL Server On-Prem to the Cloud: A Guide to AWS, Azure, and Google Cloud - The DBA Hub," internet, n.d..
[3] Netapp, "Migrate SQL Server to Azure: Options, Tools, and a Quick Tutorial," internet, n.d..
[4] Microsoft, "Migrating SQL Server Workloads FAQ - Azure SQL | Microsoft Learn," internet, n.d..
[5] Microsoft, "SQL Server Migration to Azure | Microsoft Azure," internet, n.d..
[6] Amazon, "Migrating Microsoft SQL Server databases to the AWS Cloud - AWS Prescriptive Guidance," internet, n.d..
[7] Amazon, "Migrate an on-premises Microsoft SQL Server database to Amazon EC2 - AWS Prescriptive Guidance," internet, n.d..
[8] Microsoft, "Migrating SQL server Database to the AWS cloud. - Microsoft Q&A," internet, n.d..
[9] A. W. Services, "Migrate SQL Server database from Azure SQL Managed Instance to AWS | Microsoft Workloads on AWS," internet, n.d..
[10] A. W. Services, "Migrating SQL Server databases from Microsoft Azure to AWS in near-real time with CloudBasic | Amazon Web Services," internet, n.d..
[11] B. S et al., "Efficient Middleware for the Portability of PaaS Services Consuming Applications among Heterogeneous Clouds.," 2022. https://doi.org/10.3390/s22135013
[12] Amazon, "Set up a CI/CD pipeline for database migration by using Terraform - AWS Prescriptive Guidance," internet, n.d..
[13] S. Achar, "Enterprise SaaS Workloads on New-Generation Infrastructure-as-Code (IaC) on Multi-Cloud Platforms," None, 2021. https://doi.org/10.18034/gdeb.v10i2.652
[14] R. Hat, "What is a CI/CD pipeline?," internet, n.d..
[15] Devart, "Build CI/CD pipelines in Azure DevOps," internet, n.d..

[16] Liquibase, "Guide to Database Deployment Automation | Liquibase," internet, n.d..
[17] Talent500, "Automate Database Schema Changes in CI/CD Pipelines," internet, n.d..
[18] Azuredevopslabs, "Deploying Database changes with Redgate SQL Change Automation and Azure DevOps | Azure DevOps Hands-on-Labs," internet, n.d..
[19] A. W. Services, "Building a cross-account continuous delivery pipeline for database migrations | Amazon Web Services," internet, n.d..
[20] Y. Han, "Cloud Computing: Case Studies and Total Cost of Ownership," American Library Association, 2011. https://doi.org/10.6017/ital.v30i4.1871
[21] P. Leitner, J. Cito, "Patterns in the ChaosA Study of Performance Variation and Predictability in Public IaaS Clouds," Association for Computing Machinery, 2016. https://doi.org/10.1145/2885497
[22] M. Adhikari, T. Amgoth, "An Enhanced Dynamic Load Balancing Mechanism for Task Deployment in IaaS Cloud," None, 2018. https://doi.org/10.1109/GUCON.2018.8674932
[23] D. Yaganti, "Performance - Driven Design of Scalable Web APIs Using .Net Core 3.1, Entity Framework Core, and SQL Server on Azure App Services," International Journal of Science and Research (IJSR), 2021. https://doi.org/10.21275/sr210611095250
[24] J. Cahoon et al., "Doppler," Association for Computing Machinery, 2022. https://doi.org/10.14778/3554821.3554840
[25] M. Kumar, "Serverless Architectures Review, Future Trend and the Solutions to Open Problems," None, 2019. https://doi.org/10.12691/ajse-6-1-1
[26] A. R. Khot, "A Comparative Analysis of Public Cloud Platforms and Introduction of Multi-Cloud," None, 2020. https://doi.org/10.38124/ijisrt20sep234
[27] P. Pierleoni, R. Concetti, A. Belli, L. Palma, "Amazon, Google and Microsoft Solutions for IoT: Architectures and a Performance Comparison," Institute of Electrical and Electronics Engineers, 2019. https://doi.org/10.1109/access.2019.2961511
[28] R. V. Rao, K. Selvamani, "Data Security Challenges and Its Solutions in Cloud Computing," Elsevier BV, 2014. https://doi.org/10.1016/j.procs.2015.04.171
[29] Y. S. Abdulsalam, M. Hedabou, "Security and Privacy in Cloud Computing: Technical Review," Multidisciplinary Digital Publishing Institute, 2021. https://doi.org/10.3390/fi14010011
[30] M. Carroll, A. V. D. Merwe, P. Kotz, "Secure cloud computing: Benefits, risks and controls," None, 2011. https://doi.org/10.1109/issa.2011.6027519
[31] S. Mandal, D. A. Khan, "Comprehensive Survey of Security Issues &amp; Framework in Data-Centric Cloud Applications," Eastern Macedonia and Thrace Institute of Technology, 2020. https://doi.org/10.25103/jestr.141.01
[32] T. Chou, "Security Threats on Cloud Computing Vulnerabilities," None, 2013. https://doi.org/10.5121/ijcsit.2013.5306
[33] V. Chang et al., "A Survey on Intrusion Detection Systems for Fog and Cloud Computing," Multidisciplinary Digital Publishing Institute, 2022. https://doi.org/10.3390/fi14030089
[34] R. Al-Sayyed, W. Hijawi, A. M. Bashiti, I. Aljarah, N. Obeid, O. Adwan, "An Investigation of Microsoft Azure and Amazon Web Services from Users Perspectives," kassel university press, 2019. https://doi.org/10.3991/ijet.v14i10.9902

ABBREVIATIONS:

AKS - Microsoft Azure Kubernetes Service
API - Application Programming Interface
ARM - Azure Resource Manager
AWS - Amazon Web Services
BI - Business Intelligence
BLOB - Binary Large Objects
BYOL - Bring Your Own License

CD - Continuous Delivery/Continuous Deployment
CDC - Change Data Capture
CEO - Chief Executive Officer
CI - Continuous Integration
CI/CD - Continuous Integration/Continuous Delivery
CIO - Chief Information Officer
CLI - Command Line Interface
DevOps - Development and Operations
DMA - Data Migration Assistant - A desktop tool for assessing and migrating SQL Server databases to Azure
DMS - Database Migration Service - Cloud services that help migrate databases with minimal downtime
EC2 - Amazon Elastic Compute Cloud
EKS - Amazon Elastic Kubernetes Service
Git - A distributed version control system for tracking changes in source code
GKE - Google Kubernetes Engine
IaaS - Infrastructure as a Service
IaC - Infrastructure as Code
ITL - Information Technology Laboratory
NIST - National Institute of Standards and Technology
OS - Operating System
PaaS - Platform as a Service
RDBMS - Relational Database Management System
RDS - Amazon Relational Database Service
SaaS - Software as a Service
SBOM - Software Bills of Materials
SLA - Service Level Agreement
SLSA - Supply chain Levels for Software Artifacts
SMS - Short Message Service
SQL - Structured Query Language
SSDT - SQL Server Data Tools
TCO - Total Cost of Ownership
TFS - Team Foundation Server
TFVC - Team Foundation Version Control
VM - Virtual Machine
VSTS - Visual Studio Team Services