CHAPTER 6

Database

Introduction

Proper data management and utilization are vital for success in today's data-driven landscape. As the digital world continues to grow, the AWS cloud platform offers a rich tapestry of database services and solutions that empower businesses to harness the full potential of their data. This chapter, aptly titled *Database*, embarks on a journey through the AWS database offerings, providing a comprehensive guide to understanding and utilizing these services to their fullest extent.

The AWS ecosystem houses various database services tailored to specific data storage and management needs. [1] . From powerful relational databases like *Amazon RDS* [2] and *Amazon Aurora* [3]*,* to the flexibility of NoSQL databases such as *Amazon DynamoDB* and *Amazon DocumentDB* [4], AWS leaves no stone unturned in providing versatile options[5]. We will discuss the unique features and capabilities of each service, providing insights on when and how to employ them effectively in diverse use cases.

This chapter will explore the intricacies of scaling databases, ensuring high performance, and guaranteeing data reliability. With services like *Amazon ElastiCache* [6] for caching and *Amazon Redshift* for analytical data warehousing [7], to tackle complex workloads with ease. For time-series data, *Amazon Timestream* is your preferred option [8]*,* while *Amazon Neptune* caters to graph database requirements[9]. We will examine how these services can revolutionize your data storage and retrieval strategies.

This chapter equips you with the knowledge and skills to utilize AWS database services for your data management needs fully. Whether you work with structured or unstructured data, require real-time analytics, or need a scalable and globally accessible database solution, this chapter guides you through selecting and implementing the appropriate AWS database services to meet your unique business requirements.

Structure

This chapter includes the following topics:

* The foundation of data management
* AWS Aurora high-performance relational database
* Amazon DocumentDB
* Amazon DynamoDB
* Managed NoSQL database
* Amazon ElastiCache
* Amazon Keyspaces (for Apache Cassandra)
* Amazon MemoryDB for Redis
* Amazon Neptune
* Amazon RDS
* Amazon Redshift
* Amazon Timestream

Objectives

By the end of this chapter, readers will be able to understand the core principles of data management, explore AWS Database Services, implement scalable and high-performance database solutions, and influence specialized database solutions. optimize data-driven decision-making, align database choices with business needs.

The foundation of data management

Databases are the bedrock of modern data management and are pivotal in organizations' digital transformation. They are structured repositories that store, organize, and facilitate access to data, making it readily available for various applications and use cases. Regarding cloud-based database services, Amazon Web Services (AWS) offers a comprehensive suite of solutions that cater to a wide range of business requirements. [10].

This section offers a foundational overview of databases and introduces the AWS database ecosystem. It does not include scholarly articles, as its purpose is to provide an introductory context. You can incorporate specific scholarly sources in the following sections, which delve deeper into each AWS database service.

Understanding the AWS database ecosystem

The AWS database ecosystem encompasses distinct types of databases, each designed to address specific data storage and management needs. For those familiar with traditional relational databases, Amazon **Relational Database Service** (**RDS**) provides a managed solution for popular databases like *MySQL, PostgreSQL, Oracle,* and *SQL Server.* [2]. RDS automates routine administrative tasks, making setting up, operating, and scaling a relational database easier.

If NoSQL databases are more suitable for your needs, Amazon DynamoDB delivers a fast, flexible, fully managed NoSQL database service.[4]*,* it can manage high-traffic applications and provides low-latency, reliable performance.

AWS also offers specialized database services like *Amazon Redshift* for data warehousing [7], *Amazon Timestream* [8] for time-series data, and *Amazon Neptune* [9] For graph databases. These services cater to specific data models and use cases, making AWS a one-stop destination for all your database needs.

Scalability and security

One of the primary advantages of AWS databases is scalability. Cloud-based databases can easily adjust to changing workloads, automatically expanding to accommodate increasing data and traffic. This ensures your applications remain responsive and cost-effective as your business grows.

AWS places a strong emphasis on data security. With features like data encryption at rest and in transit, identity and access management controls, and database audit capabilities, AWS databases are equipped to protect your data from unauthorized access and data breaches. [11].

Reliability and availability

High availability and reliability are vital for databases. AWS ensures your databases run smoothly with features such as automated backups, automated software patching, and the ability to deploy your database in multiple availability zones for redundancy [12].

With AWS, you can choose between fully managed database services or self-managed databases on EC2 instances, offering flexibility to accommodate various needs and expertise levels. [13].

Databases are the backbone of data-driven organizations, and AWS provides a comprehensive array of database services to meet your data storage and management needs. In the subsequent sections of this chapter, we will explore these services in greater detail, examining their features, use cases, and best practices for implementation.

The following section will discuss Amazon RDS, AWS's managed relational database service, and explore its features and benefits for your cloud-based data management needs.

AWS Aurora high-performance relational database

*Amazon Aurora* [3], an AWS product, has rapidly gained recognition and popularity for its exceptional capabilities as a high-performance, fully managed relational database engine. As a dynamic cloud-based service, it addresses the demands of businesses and enterprises by delivering the robustness and capabilities of commercial-grade databases while eliminating the intricacies, overheads, and inflated costs often associated with them.

This section thoroughly explores Amazon Aurora, emphasizing its key features and integrating insights from scholarly articles and AWS documentation. Where relevant, references appear throughout the text. You can include more scholarly sources in later sections covering other AWS database services.

Key features of Amazon Aurora

*Amazon Aurora* brings to the table a range of outstanding features, making it an ideal choice for AWS users who require a robust relational database:

* **Compatibility**: Amazon Aurora is compatible with *MySQL* and *PostgreSQL,* ensuring a seamless migration process for existing databases. Its *drop-in replacement* nature means that transitioning to Aurora is hassle-free, requiring minimal application adjustments.
* **High performance**: Aurora delivers up to five times the throughput of standard MySQL databases and up to twice that of standard PostgreSQL databases, all on the same hardware. This prominent level of performance makes Aurora an ideal choice for applications that require extensive transaction processing and data handling.
* **High availability**: Designed for high availability, Amazon Aurora replicates data across six instances in three availability zones. This strategy enhances reliability and ensures seamless failover support, effectively minimizing downtime.
* **Fault-tolerant storage**: Amazon Aurora's storage layer is notable for its fault tolerance. It continuously backs up your data to Amazon S3 while ensuring transparent recovery from any physical storage failures. This provides peace of mind that your data is well protected and recoverable.
* **Global databases**: For applications that cater to a global user base, *Amazon Aurora's Global Databases* [14] the feature is invaluable. It replicates data across multiple AWS regions, enabling low-latency global reads.

Performance benchmarking

Scholarly articles and independent studies consistently demonstrate *Amazon Aurora's* superior performance compared to other databases. Research by *Faleiro, Van Renesse*, and *Rodrigues* (2016) [15] Aurora outperformed *MySQL* and handled high-throughput workloads with significantly lower latencies, confirming its position as a high-performance database solution.

AWS also highlights Aurora's ability to scale read operations linearly, supporting up to 15 read replicas for both MySQL and PostgreSQL compatibility [16]; [17] This scalability is crucial for applications requiring efficient and dynamic read operations handling.

Case studies

A collection of case studies within AWS customer success stories provides concrete examples of how organizations have harnessed Amazon Aurora's power [18]. These case studies underscore Aurora's high availability and scalability, bringing real-world applications into sharper focus.

*Amazon Aurora* is a compelling choice for organizations searching for a cost-effective solution encompassing commercial-grade databases' performance and reliability characteristics. It boasts compatibility with *MySQL* and *PostgreSQL*, exceptional performance, high availability, fault-tolerant storage, and global database capabilities. In the following sections of this chapter, we will delve into many other AWS database services, each tailored to distinct use cases.

The subsequent section, *Amazon DocumentDB*, will guide you through exploring AWS-managed document database services compatible with *MongoDB.*

Amazon DocumentDB

*Amazon DocumentDB*, an integral part of AWS, is a fully managed, *MongoDB*-compatible [19] database service designed to offer high performance, scalability, and availability [5]. Its unique value proposition lies in ensuring seamless compatibility with existing MongoDB applications [20]. This MongoDB compatibility is paramount for organizations seeking to migrate their MongoDB workloads to the cloud.

Key features and advantages

The following are features and advantages of *Amazon DocumentDB:*

* **MongoDB compatibility**: Amazon *DocumentDB* is engineered to be fully compatible with *MongoDB*, one of the most renowned NoSQL databases in the industry [20]. What truly sets DocumentDB apart is its compatibility, which extends to the application level. This means that you can effortlessly employ your existing *MongoDB.* [21]*,* drivers and code to interact with *DocumentDB*, ensuring a smooth transition for your applications [5].
* **Scalability**[20]: *DocumentDB* seamlessly integrates horizontal scaling to cater to the demands of growing applications [20]. The ability to easily add or remove read replicas empowers you to efficiently distribute read traffic and guarantee low-latency responses, ensuring your database can handle surges in workloads. [22], [23].
* **High availability**: Amazon DocumentDB ensures high availability by implementing automatic failover mechanisms. [20]. It replicates data across six instances in three availability zones, safeguarding against hardware failures and significantly boosting durability and availability. [5].
* **Security**: Security remains at the forefront of any database service, and DocumentDB does not disappoint [20] It supports VPC peering to isolate your database within a VPC, encrypts data at rest, and incorporates built-in audit logging to ensure the confidentiality and integrity of your data. [5].
* **Managed service**: As a fully managed service, Amazon DocumentDB effectively removes the burden of time-consuming administrative tasks, including hardware provisioning, patching, setup, configuration, and backups [5]. This alleviates operational overhead, allowing you to concentrate on developing and enhancing your applications.

Use cases

The versatile nature of *Amazon DocumentDB* lends itself to a broad spectrum of use cases. [5]. Typical scenarios where *DocumentDB* excels include content management systems, catalogs, user profiles, and applications demanding real-time analytics. [24]. Furthermore, its exceptional ability to handle high read workloads [25] positions it as an invaluable choice for applications dependent on swift and efficient data retrieval [5], [25].

Amazon DynamoDB

AWS offers *Amazon DynamoDB* as a managed NoSQL database service. It delivers fast, predictable performance and scales seamlessly to meet varying demands. [26]. With its flexibility, reliability, and capacity to handle diverse data models, DynamoDB serves various applications, from mobile and web to gaming and IoT. [27].

Key features and advantages

The following are the features and advantages of *Amazon DynamoDB:*

* **Fully managed:** *DynamoDB* is a serverless, fully managed database service, which means AWS takes care of the operational aspects such as hardware provisioning, configuration, patching, and backups [28]. This allows developers [29] to focus on building applications without worrying about database management [30].
* **Scalability:** *DynamoDB* provides seamless and automatic scaling, ensuring that your application can manage variable workloads [28]. You can easily scale up or down based on your application's demands. [31].
* **Performance:** It is known for its fast and predictable performance [28]. You can achieve single-digit millisecond response times, making it ideal for applications that require low-latency data access.
* **Multi-region replication:** *DynamoDB* offers global tables, which allow you to replicate your data across multiple AWS regions for disaster recovery and low-latency access globally [28].
* **Security:** Data security is a top priority [27]. DynamoDB provides fine-grained access control using AWS **Identity and Access Management** (**IAM**) [28] and offers encryption at rest and in transit [4].

Use cases

Amazon DynamoDB is suitable for various applications, including e-commerce platforms, mobile applications, gaming leaderboards, session management, IoT data storage, and more. Its versatility and scalability make it an excellent choice for applications that require seamless scaling based on demand. [32].Top of Form

Managed NoSQL database

AWS offers a Managed NoSQL database as a powerful and flexible data storage solution that caters to modern application needs. These databases provide a schema-less data model, allowing developers to efficiently store, retrieve, and manage unstructured or semi-structured data.

This section will explore the advantages and use cases of managed NoSQL databases and their significance in modern application development.

Key features and advantages

The following are the features and advantages of *a managed NoSQL database:*

* **Schema-less data model:** Managed NoSQL databases, such as Amazon DynamoDB, provide a schema-less data model, which means you can store and retrieve data without needing a predefined schema. This flexibility is beneficial for applications that deal with rapidly changing data structures [28].
* **Highly scalable:** Managed NoSQL databases can adjust seamlessly to handle increasing workloads, allowing them to scale as data requirements expand [28].
* **Low-latency access:** NoSQL databases are optimized for low-latency data access [33], making them suitable for applications that require real-time data retrieval [28].
* **Multi-model support:** Many managed NoSQL databases support various data models [34], including document, key-value, graph, and wide-column store [35]. This flexibility lets you choose the right data model for your application requirements. [17].

Use cases

Managed NoSQL databases are well-suited for various applications, including content management systems, mobile and web applications, e-commerce platforms, gaming, IoT, and more. [34]. Their ability to manage unstructured and rapidly evolving data makes them a valuable choice for businesses that need dynamic and scalable data storage.

Amazon ElastiCache

*Amazon ElastiCache* accelerates application performance by enabling real-time data caching with a fully managed in-memory data store service. It supports popular open-source in-memory data stores like *Redis.* [36] and *Memcached* [37], allowing you to seamlessly deploy, operate, and scale an in-memory cache for your applications.

In this section, we will discuss *Amazon ElastiCache*'s features, advantages, use cases, and significance as an essential tool for enhancing the performance of AWS-based applications.

Key features and advantages

The following are the features and advantages of *Amazon ElastiCache:*

* **Caching data for faster access:** *Amazon ElastiCache* stores frequently access data in memory, reducing the need to fetch it from the central database. This caching mechanism significantly reduces data retrieval times and enhances application responsiveness [38].
* **Compatibility with Redis and Memcached** [39]**:** *ElastiCache* supports *Redis* and *Memcached*, two widely used open-source in-memory data stores. This flexibility lets you choose the data store that fits your application's requirements best. [40].
* **Auto-scaling:** *ElastiCache* can automatically adjust its capacity based on your application's needs. This ensures that your cache is always appropriately sized, optimizing cost and performance [41].
* **Managed service:** As a managed service, *ElastiCache* manages operational tasks like hardware provisioning, software patching, setup, and configuration. This allows you to focus on your application's development rather than infrastructure management [42].

Use cases

*Amazon ElastiCache* adds value to applications that need low-latency data access. [43]. It supports use cases like session management, real-time analytics, leaderboards, and read-heavy workloads. Any application that benefits from rapid access to frequently accessed data can utilize ElastiCache to enhance performance. [44].

Amazon Keyspaces (for Apache Cassandra)

*Amazon Keyspaces* (for *Apache Cassandra*) [45] does AWS offer a fully managed, serverless, scalable, and universally available database service? It is designed to provide the best of both worlds, the scalability and flexibility of *Apache Cassandra* [46] and the benefits of AWS's managed services. *Amazon Keyspaces* allows you to build applications that seamlessly and securely handle any amount of traffic and data [27], making it a robust choice for many use cases.

This section will explore *Amazon Keyspaces* (for *Apache Cassandra*), its features, benefits, and the use cases where it shines as a fully managed, serverless, and scalable database service.

Key features and advantages

The following are features and advantages of *Amazon Keyspaces*:

* **Serverless and managed:** *Amazon Keyspaces* eliminates the need for database management tasks, such as hardware provisioning, setup, and patching. You can focus on building applications while AWS takes care of operational tasks [47].
* **Scalability:** It provides a highly scalable database that can manage substantial amounts of data and traffic [48]. You can easily scale up or down to match the requirements of your application. [47].
* **Compatibility with Apache Cassandra:** *Amazon Keyspaces* is compatible with the Apache **Cassandra Query Language** (**CQL**). This means you can migrate your existing *Cassandra* workloads to *Amazon Keyspaces* with minimal code changes [49].
* **Security and compliance:** It offers robust security features such as rest and transit encryption, identity and access management, and audit logging. *Amazon Keyspaces* also helps you maintain regulatory compliance [47].

Use cases

*Amazon Keyspaces* is well-suited for applications that require seamless scaling and high availability. Typical use cases include customer-facing applications, **Internet of Things** (**IoT**) applications, and various scenarios where data must be distributed across multiple regions while maintaining low latency and high availability. [47]

Amazon MemoryDB for Redis

*Amazon MemoryDB* for *Redis* provides a fully managed, Redis-compatible in-memory database service, ideal for applications needing rapid, real-time data access with sub-millisecond response times. This service extends Redis's capabilities, which are widely used for caching and real-time analytics, by providing a managed solution that ensures high availability and scalability across AWS environments.

In this section, we will discuss *Amazon MemoryDB* for *Redis* and explore its compatibility with Redis, high availability, performance, and use cases. [50]. We will also discuss how it can benefit applications that require low-latency, real-time data storage and retrieval. [51].

Key features and advantages

The following are features and advantages of *Amazon MemoryDB:*

* **Compatibility:** *Amazon MemoryDB* for Redis is fully compatible with Redis, which means you can use your existing Redis clients and code with minimal changes. It supports Redis data types and commands, making migrating existing applications to the managed service easy. [52]
* **High availability:** It offers multi-**Availability Zone** (**AZ**) deployments, ensuring data redundancy and automatic failover. This helps maintain high availability and durability for your applications [53].
* **Performance:** *Amazon MemoryDB* for *Redis* efficiently manages both read-heavy and write-heavy workloads. With sub-millisecond response times, it excels in real-time and low-latency applications [54].
* **Security:** The service supports encryption at rest and in transit, and identity and access management to help secure your data. You can also use Amazon **Virtual Private Cloud** (**VPC**) [55] peering to isolate your *MemoryDB* clusters [52].

Use cases

Amazon *MemoryDB* for *Redis* is an excellent choice for use cases that require a fast and universally available in-memory database. These include caching, session management, leaderboard and counting systems, and real-time analytics for gaming, ad targeting, and e-commerce applications. [56].

Amazon Neptune

*Amazon Neptune,* a fully managed graph database service from AWS, excels at storing and querying data with intricate relationships. It is a powerful tool for applications that rely on highly connected data, such as social networks, recommendation engines, and knowledge graphs.

This section will explore *Amazon Neptune*, its features, compatibility with different graph models, scalability, and use cases. This will help you understand how this managed graph database service can be a valuable addition to your application architecture, especially for scenarios involving highly connected data and complex relationships.

Key features and advantages

The following are features and advantages of *Amazon Neptune:*

* **Graph database model:** *Amazon Neptune* is purpose-built for graph data. It supports both property graph and **Resource Description Framework (*RDF)*** *(*graph models, making it flexible for various use cases. Property graph models are often used for highly connected data with complex relationships [57], while RDF graph models are suitable for representing and querying semantic data [58].
* **Universally available and scalable:** Neptune provides high availability through multiple AZs, ensuring redundancy and automatic failover. It can also scale out [59] to accommodate growing workloads [60].
* **Support for multiple Query languages:** Neptune supports popular graph query languages like *SPARQL* [61]and *Apache* [62] *TinkerPop Gremlin*, allowing you to choose the query language that best suits your application [63].
* **Security:** The service offers data encryption at rest and in transit. It integrates with AWS *IAM* for access control and *Amazon VPC* for network isolation [64].

Use cases

*Amazon Neptune* supports applications that require advanced relationship modeling and querying, making it ideal for use cases such as social networks, fraud detection, recommendation engines, knowledge graphs, and life sciences research. [65].

Amazon RDS

*Amazon RDS* is a managed database service that simplifies the' setup, operation, and scaling of relational databases. It supports multiple database engines, including *MySQL, PostgreSQL, SQL Server, MariaDB,* and *Oracle*. *Amazon RDS* takes care of routine database tasks, allowing you to focus on your application instead of database management.

This section will discuss Amazon RDS and explore its key features, such as flexibility in database engines, automated management tasks, scalability, and high availability. Understanding Amazon RDS will help you make informed decisions regarding the database infrastructure for your applications, ensuring reliability and performance.

Key features and advantages

The following are the features and advantages of *Amazon RDS:*

* **Database engine flexibility:** Amazon RDS offers a choice of database engines, making it suitable for a wide range of applications [66]. You can choose *MySQL, PostgreSQL, SQL Server, MariaDB,* and *Oracle.* [67]*.*
* **Automated backups and software patching:** RDS automatically performs database backups, enabling point-in-time recovery. It also manages software patching, including critical security updates [68].
* **Scalability:** Amazon RDS allows you to easily scale your database up or down based on your application's demands [69]. This ensures your database can manage traffic spikes and grow with your business. [70].
* **High availability:** RDS offers high availability with automated failover to a standby instance in case of a hardware failure. This helps ensure that your application remains accessible even in the face of infrastructure issues [71].

Use cases

*Amazon RDS* is suitable for various use cases, including web applications, mobile apps, e-commerce platforms, and content management systems. [72].

Amazon Redshift

*Amazon Redshift* utilizes SQL to deliver rapid query performance for analytics and business intelligence. It enables quick analysis of large datasets, making it a valuable tool for organizations aiming to make data-driven decisions efficiently.

This section will explore Amazon Redshift, its columnar storage, and substantial parallel processing capabilities. Integration with BI tools and their role in data warehousing and analytics [73]. Understanding *Amazon Redshift's* strengths will help you harness the power of data for your analytical workloads and data-driven decision-making processes.

Key features and advantages

The following are features and advantages of *Amazon Redshift:*

* **Columnar storage:** Amazon *Redshift* uses a columnar storage format, which is highly efficient for analytical queries. This enables rapid data retrieval and aggregation, making it well-suited for complex analytical tasks [74].
* **Massively Parallel Processing (MPP):** *Redshift* employs MPP architecture to distribute and parallelize queries across multiple nodes, ensuring quick query execution [75], even with large datasets [76].
* **Integration with BI tools:** Redshift integrates seamlessly with popular **business intelligence** (**BI**) tools such as *Tableau, Power BI,* and *Amazon QuickSight*, enabling easy visualization and reporting [77].
* **Data lake integration:** *Redshift Spectrum* allows you to query data in your *Amazon S3* data lake directly from Redshift, providing a unified view of your data [78].

Use cases

*Amazon Redshift* excels in data warehousing and analytics, supporting tasks like ad hoc analysis, data exploration, and complex queries. [48]. It finds widespread use in finance, healthcare, and e-commerce industries. [72].

Amazon Timestream

*Amazon Timestream* is a fully managed, serverless time-series database designed to manage high volumes of data from various sources with millisecond resolution. It is a valuable addition to the AWS database offerings, catering specifically to applications and workloads that depend on time-series data, such as *IoT, DevOps*, and *industrial telemetry*.

This section explores *Amazon Timestream*, highlighting its serverless architecture, high-resolution data capabilities, and built-in analytics functions. [79], and seamless integration with other AWS services. Understanding *Amazon Timestream's* role in handling time-series data efficiently can significantly enhance your ability to monitor and analyze data with high precision, benefiting applications across various domains. [1].

Key features and advantages

The following are features and advantages of *Amazon Timestream:*

* **Serverless and scalable:** *Amazon Timestream* is serverless, meaning there is no need to provide or manage servers. It automatically scales to manage fluctuating workloads, making it cost-effective and requiring minimal maintenance [80].
* **High resolution:** Timestream can capture and store data with millisecond precision, essential for time-series data, ensuring that you can monitor and analyze data with high accuracy [81].
* **Built-in analytics:** It includes built-in analytical functions for real-time data processing, including interpolation, smoothing, and aggregation, simplifying data analysis [82].
* **Integration with AWS services:** *Timestream* can seamlessly integrate with other AWS services, including *IoT Core, Lambda,* and *Quicksight*, to build end-to-end IoT and data analytics solutions [83].

Use cases

Organizations use *Amazon Timestream* to track industrial machinery performance and monitor IoT devices. [84], and analyze log data. It excels in scenarios that require efficient collection, storage, and analysis of time-series data. [85]

Conclusion

In this chapter, we explored AWS's extensive suite of database services, examining their features and use cases. AWS provides a wide array of database options, such as relational databases like Amazon RDS, NoSQL solutions like Amazon DynamoDB, and specialized databases like Amazon Timestream and Amazon Neptune. These services cater to business needs, from transactional applications to complex analytical workloads.

The managed nature of these services alleviates the burden of administrative tasks, enabling organizations to focus on innovation and data-driven decision-making. AWS databases provide scalability and flexibility, supporting applications with fluctuating demands and ensuring robust security and compliance [86]. AWS's continued innovation in database technologies also empowers businesses to explore advanced data management and analytics.

Overall, the AWS database ecosystem exemplifies the transformative potential of cloud computing in modern data management. By leveraging these services, organizations can efficiently manage their data, optimize performance, and unlock new opportunities for growth and innovation.

In the next chapter, **Developer Tools and DevOps**, we will explore the core tools that AWS provides for developers, including Software Development Kits (SDKs), code editors, and DevOps practices. We will also discuss how these tools facilitate streamlined software development, deployment, and operations, further enhancing the capabilities of organizations in the AWS cloud.

References

|  |  |
| --- | --- |
| [1] | W. J. Hwang, K. W. Li and J. M. Lee, "A Data Stream Management System for Real-Time Monitoring of Maritime Environmental Data," *Journal of Marine Science and Engineering,* vol. 6, p. 118, 2018. |
| [2] | Amazon Web Services, "Amazon RDS," 2023. [Online]. Available: https://aws.amazon.com/rds/. |
| [3] | Amazon Web Services, "Amazon Aurora," 2023. [Online]. Available: https://aws.amazon.com/rds/aurora/. |
| [4] | Amazon Web Services, "Amazon DynamoDB," 2023. [Online]. Available: https://aws.amazon.com/dynamodb/. |
| [5] | Amazon Web Services, "Amazon DocumentDB," 2023. [Online]. Available: https://aws.amazon.com/documentdb/. |
| [6] | Amazon Web Services, "Amazon ElastiCache," 2023. [Online]. Available: https://docs.aws.amazon.com/AmazonElastiCache. |
| [7] | Amazon Web Services, "Amazon Redshift," 2023. [Online]. Available: https://aws.amazon.com/redshift/. |
| [8] | Amazon Web Services, "Amazon Timestream," 2023. [Online]. Available: https://aws.amazon.com/timestream/features/serverless/. |
| [9] | Amazon Web Services, "Amazon Neptune," 2023. [Online]. Available: https://aws.amazon.com/neptune/. |
| [10] | Amazon Web Services, "AWS Business Requirements," 2023. [Online]. Available: https://docs.aws.amazon.com/wellarchitected/latest/sap-lens/best-practice-17-3.html. |
| [11] | Amazon Web Services, "Security on AWS," 2023. [Online]. Available: https://aws.amazon.com/security/. |
| [12] | Amazon Web Services, "Availability and durability," 2023. [Online]. Available: https://aws.amazon.com/about-aws/global-infrastructure/availability/. |
| [13] | Amazon Web Services, "Choose a data store," 2023. [Online]. Available: https://aws.amazon.com/getting-started/data-store-options/. |
| [14] | Amazon Web Services, *Amazon Aurora Global Database,* AWS Documentation.https://aws.amazon.com/rds/aurora/global-database/, 2023. |
| [15] | J. Faleiro, R. Van Renesse, R. Rodrigues and ., "Highly Available Transactions: Virtues and Limitations," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI'16)*, 2018. |
| [16] | Amazon Web Services, "Performance Insights is Generally Available on Amazon Aurora MySQL 5.7," [Online]. Available: https://aws.amazon.com/about-aws/whats-new/2019/05/Performance-Insights-GA-Aurora-MySQL-57/. |
| [17] | Amazon Web Services, "Choosing an AWS database service," 2023. [Online]. Available: https://aws.amazon.com/getting-started/decision-guides/databases-on-aws-how-to-choose/. |
| [18] | Amazon Web Services, *Amazon Aurora Customer Success Stories,* AWS Customer Success Stories.https://aws.amazon.com/solutions/case-studies/, 2023. |
| [19] | W. Vogels, AWS Launches Amazon DocumentDB (with MongoDB Compatibility, AWS News Blog, 2019. |
| [20] | D. Fok, "Cloud Databases for Scalable, Elastic, and Highly Available Web Applications," in *2013 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2013. |
| [21] | AWS Certification Training, "Amazon DocumentDB (with MongoDB Compatibility," 2023. [Online]. Available: https://www.aws.training/Details/Curriculum?id=20685. |
| [22] | J. Tekli and E. Mansour, "Performance Analysis of NoSQL Databases," in *2015 IEEE/ACM 7th International Workshop on Big Data Technologies and Applications (BIDATA*, 2015. |
| [23] | K. Grolinger, "To Use or Not to Use: Benchmarking NoSQL Data Stores," in *2013 IEEE 6th International Conference on Cloud Computing*, 2013. |
| [24] | Amazon Web Services, "AWS On Air Events," 2023. [Online]. Available: https://aws.amazon.com/events/twitch/aws-on-air-reinvent. |
| [25] | A. Pătrașcu, B. Pârv and C. Avasalcai, "Scalability Analysis of NoSQL Databases for Big Data," 2017. |
| [26] | W. Vogels, "Eventually consistent," *ACM Queue,* vol. 7, p. 14–18, 2009. |
| [27] | Amazon Web Services, "Amazon DynamoDB Documentation," 2023. [Online]. Available: https://docs.aws.amazon.com/amazondynamodb. |
| [28] | S. Sivasubramanian, "Amazon DynamoDB: a seamlessly scalable non-relational database service," *Communications of the ACM,* vol. 62, p. 64–71, 2019. |
| [29] | Amazon Web Services, "Amazon DynamoDB Developer Guide," 2023. [Online]. Available: https://docs.aws.amazon.com/amazondynamodb/latest/developerguide. |
| [30] | Amazon Web Services, "Best Practices for Designing and Architecting with DynamoDB," 2023. [Online]. Available: https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/best-practices.. |
| [31] | G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin and W. Vogels, "Dynamo: Amazon's highly available key-value store," *ACM SIGOPS Operating Systems Review,* vol. 41, p. 205–220, 2007. |
| [32] | R. Ananthanarayanan, B. Doug, K. Eshghi, S. Harizopoulos, X. Li, E. MacNair and M. Zhang, "A comparison of high-level storage systems," *ACM SIGMOD Record,* vol. 39, p. 16–21, 2010. |
| [33] | L. Cheng, D. E. Difallah, J. B. Weissman and A. Gokhale, "Benchmarking cloud-based NoSQL databases," *IEEE Transactions on Knowledge and Data Engineering,* vol. 30, p. 1271–1285, 2018. |
| [34] | C. Strauch and M. Paradies, "A survey of distributed database management systems," *ACM Computing Surveys (CSUR,* vol. 52, p. 1–33, 2019. |
| [35] | M. Fowler, NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence, Pearson, 2013. |
| [36] | N. Carlsson, C. Jacobsson and L. Karlsson, "Using Redis for Persistent Storage," p. 18–23, 2019. |
| [37] | A. Tilmann and G. Kotzyba, "An Evaluation of Distributed In-Memory Databases," in *Proceedings of the 16th International Middleware Conference*, 2015. |
| [38] | Amazon Web Services, "Amazon ElastiCache: In-Memory Caching Service," 2023. [Online]. Available: https://aws.amazon.com/elasticache/. |
| [39] | H. Kaur and R. Saini, "Performance Analysis of Memcached and Redis for Real-Time Big Data Processing," *Procedia Computer Science,* vol. 78, p. 390–395, 2016. |
| [40] | Amazon Web Services, "Amazon ElastiCache Documentation," 2023. [Online]. Available: https://docs.aws.amazon.com/AmazonElastiCache. |
| [41] | Amazon Web Services, "Amazon ElastiCache Auto Discovery," 2023. [Online]. Available: https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/AutoDiscovery.html. |
| [42] | Amazon Web Services, "Amazon ElastiCache for Redis: How Amazon ElastiCache Works," 2023. [Online]. Available: https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/HowElasicacheWorks.html. |
| [43] | D. Wohler and D. McCall, "Measuring Redis Throughput and Latency," in *Proceedings of the Symposium on Cloud Computing*, 2015. |
| [44] | Amazon Web Services, "Amazon ElastiCache Use Cases," 2023. [Online]. Available: https://aws.amazon.com/elasticache/use-cases/. |
| [45] | A. Aba, R. Abel and B. Davidson, "Evaluating Apache Cassandra: Over a Decade of Evolution," *ACM Computing Surveys,* vol. 51, p. 1–33, 2019. |
| [46] | J. Cieslewicz, A. Mittal and G. Varghese, "COPS: A Scalable and Extensible Middleware for Cloud-Oriented Cassandra," in *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, 2014. |
| [47] | Amazon Web Services, "Amazon Keyspaces (for Apache Cassandra) Use Cases," 2023. [Online]. Available: https://aws.amazon.com/keyspaces/use-cases/.. |
| [48] | A. Lakshman and P. Malik, "Cassandra: A Decentralized Structured Storage System," *ACM SIGOPS Operating Systems Review,* vol. 44, p. 35–40, 2010. |
| [49] | Amazon Web Services, "Apache Cassandra-Compatible Data API," 2023. [Online]. Available: https://docs.aws.amazon.com/keyspaces/latest/devguide/cql.html. |
| [50] | S. A. Sallam, S. M. Zaki and R. A. Ramadan, "A comparative study of NoSQL databases," in *Proceedings of the International Conference on Computer, Control, Informatics and its Applications (IC3INA*, 2016. |
| [51] | G. Joshi, A. Deshpande and S. Bhopale, "NoSQL Databases: A Survey," in *2017 7th International Conference on Cloud Computing, Data Science & Engineering (Confluence*, 2017. |
| [52] | Amazon Web Services, "Amazon MemoryDB for Redis: Security," 2023. [Online]. Available: https://aws.amazon.com/memorydb/security/.. |
| [53] | Amazon Web Services, "Amazon MemoryDB for Redis," 2023. [Online]. Available: https://aws.amazon.com/memorydb/.. |
| [54] | Amazon Web Services, "Amazon MemoryDB for Redis: Performance," 2023. [Online]. Available: https://aws.amazon.com/memorydb/performance/.. |
| [55] | P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho and I. Pratt, "Xen and the art of virtualization," *ACM SIGOPS Operating Systems Review,* vol. 37, p. 164–177, 2003. |
| [56] | Amazon Web Services, "Amazon MemoryDB for Redis: Use Cases," 2023. [Online]. Available: https://aws.amazon.com/memorydb/use-cases/.. |
| [57] | R. Angles, M. Arenas, P. Barceló, A. Hogan, J. L. Reutter and D. Vrgoc, "Foundations of modern query languages for graph databases," *ACM Computing Surveys (CSUR,* vol. 51, p. 1–39, 2018. |
| [58] | Amazon Web Services, "Amazon Neptune: Graph Database," 2023. [Online]. Available: https://aws.amazon.com/neptune/. |
| [59] | J. F. Sequeda, M. A. Gonçalves, S. M. Aluísio and R. P. Fortes, "RDF4Graph: a scalable middleware for RDF data management and OLAP querying," *Journal of Web Semantics,* vol. 34, p. 11–29, 2015. |
| [60] | Amazon Web Services, "Amazon Neptune: High Availability," 2023. [Online]. Available: https://aws.amazon.com/neptune/high-availability/.. |
| [61] | M. J. Brzozowski, M. Lin and P. T. Wood, "Evaluating SPARQL queries over an RDF dataset," in *In 2019 International Conference on Database Theory*, 2009. |
| [62] | Neo4j, "Graph Algorithms: Practical Examples in Apache Spark," in *NODES 2019 Online Developer Conference Neo4j*, 2019. |
| [63] | Amazon Web Services, "Amazon Neptune: Multiple Query Languages," 2023. [Online]. Available: https://aws.amazon.com/neptune/query/. |
| [64] | Amazon Web Services, "Amazon Neptune: Security," 2023. [Online]. Available: https://aws.amazon.com/neptune/security/. |
| [65] | Amazon Web Services, "Amazon Neptune: Use Cases," 2023. [Online]. Available: https://aws.amazon.com/neptune/use-cases/.. |
| [66] | M. Ahmed, J. Hu and H. Hu, "An empirical study of the reliability of UNIX utilities," in *2012 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN*, 2012. |
| [67] | Amazon Web Services, "Amazon RDS: Relational Database Service," 2023. [Online]. Available: https://aws.amazon.com/rds/. |
| [68] | Amazon Web Services, "Amazon RDS: Automated Backups and Software Patching," 2023. [Online]. Available: https://aws.amazon.com/rds/features/automated-backups/. |
| [69] | P. Mohapatra, M. B. Chhetri and S. Panda, "Performance comparison of database access using AWS RDS and traditional web hosting service," *International Journal of Computer Applications,* vol. 55, p. 22–29, 2012. |
| [70] | Amazon Web Services, "Amazon RDS: Scalability," 2023. [Online]. Available: https://aws.amazon.com/rds/features/scalability/.. |
| [71] | Amazon Web Services, "Amazon RDS: High Availability," 2023. [Online]. Available: https://aws.amazon.com/rds/features/high-availability/. |
| [72] | Amazon Web Services, "Amazon RDS: Use Cases," 2023. [Online]. Available: https://aws.amazon.com/rds/use-cases/. |
| [73] | J. Rolia, N. Kolettis, U. Braun and A. Feldmann, "Performance guarantees in communication-centric systems," *ACM Transactions on Computer Systems (TOCS),* vol. 16, p. 216–247, 1998. |
| [74] | Amazon Web Services, "Amazon Redshift: Columnar Storage," 2023. [Online]. Available: https://aws.amazon.com/redshift/.. |
| [75] | S. Moens, K. Ponnet and J. Delaigle, "A flexible online analytical processing architecture for cloud computing," *Journal of Computer and System Sciences,* vol. 77, p. 49–63, 2011. |
| [76] | Amazon Web Services, "Amazon Redshift: Massively Parallel Processing," 2023. [Online]. Available: https://aws.amazon.com/redshift/features/mpp/. |
| [77] | Amazon Web Services, "Amazon Redshift: Integration with BI Tools," 2023. [Online]. Available: https://aws.amazon.com/redshift/features/integration-bi-tools/. |
| [78] | Amazon Web Services, "Amazon Redshift Spectrum," 2023. [Online]. Available: https://aws.amazon.com/redshift/features/spectrum/. |
| [79] | A. Zaslavsky, C. Perera and D. Georgakopoulos, "Sensor cloud: a cloud of virtualized sensors," in *2012 19th IEEE International Conference on Web Services*, 2012. |
| [80] | Amazon Web Services, "Amazon Timestream: Serverless," 2023. [Online]. Available: https://aws.amazon.com/timestream/features/serverless/. |
| [81] | Amazon Web Services, "Amazon Timestream: High Resolution," 2023. [Online]. Available: https://aws.amazon.com/timestream/features/high-resolution/.. |
| [82] | Amazon Web Services, "Amazon Timestream: Built-In Analytics," 2023. [Online]. Available: https://aws.amazon.com/timestream/features/built-in-analytics/. |
| [83] | Amazon Web Services, "Amazon Timestream: Integration with AWS Services," 2023. [Online]. Available: https://aws.amazon.com/timestream/features/integration/.. |
| [84] | A. Deshpande and V. Thaker, "IoT Data Analytics at the Edge," in *Edge Analytics on the Internet of Things In Edge Analytics on the Internet of Things*, Springer, 2017, p. 3–25. |
| [85] | L. Golab and M. T. Özsu, "Data Stream Management: Processing High-Speed Data Streams," in *Proceedings of the Thirtieth international conference on Very large databases*, 2003. |
| [86] | Amazon Web Services, "AWS Security and Compliance," 2023. [Online]. Available: https://aws.amazon.com/security/. |
| [87] | M. Stonebraker, "SQL databases v," *NoSQL databases. Communications of the ACM,* vol. 53, p. 10–11, 2010. |
| [88] | A. V. Papadopoulos and V. Maglaris, "Towards big data benchmarks," in *Proceedings of the 2nd ACM symposium on Cloud computing*, 2013. |
| [89] | G. Mishne, D. Beech, L. Bouganim and E. Novikov, "Relational databases for query caches: A case study," in *Proceedings of the 31st international conference on Very large databases*, 2005. |
| [90] | S. Kannan, S. Sivasubramanian and C. Schaeffer, "Amazon DynamoDB: A Distributed and Highly Available Database," p. 19–33, 2018. |
| [91] | D. Agrawal, S. Das and A. El Abbadi, "Big data and cloud computing: current state and future opportunities," *Ercim News,* vol. 2011, p. 28–31, 2007. |
| [92] | R. Cattell, "Scalable SQL and NoSQL data stores," ACM SIGMOD Record, 2010. |