# Evaluating the Performance, Scalability, and Flexibility of Diverse Database Architecture

Suhair Amer
Department of Computer Science
Southeast Missouri State University
Cape Girardeau, Missouri, USA

Ankita Maharajan
Department of Computer Science
Southeast Missouri State University
Cape Girardeau, Missouri, USA

## ABSTRACT
This paper presents a comparative analysis of eleven database systems, including MySQL, NoSQL, Oracle, PostgreSQL, Teradata, and others, examining their performance, scalability, flexibility, and cost-effectiveness. The study categorizes the systems into relational, non-relational, and hybrid models to evaluate their suitability for modern data-driven applications. Findings reveal that Oracle and Teradata provide exceptional performance for large enterprise workloads, while PostgreSQL offers a balanced combination of reliability, adaptability, and open-source accessibility. NoSQL systems demonstrate strong scalability and flexibility for handling unstructured and large-scale data, whereas legacy platforms like Supra PDM and Versant maintain specialized roles in specific domains. Overall, the analysis emphasizes that effective database selection depends on aligning system capabilities with application requirements, data complexity, and organizational scale.

## Keywords
DBMS, security, Relational Databases, SQL, NoSQL

## 1. MYSQL DATABASE
MySQL is an open-source relational database management system (RDBMS) and it is free and open-source software under the terms of the GNU General Public License and is also available under a variety of proprietary licenses [1]. An example of a security issue is CVE-2014-4258, that it allows remote authenticated users to affect confidentiality, integrity, and availability via vectors related to SRINFOSC. Another example is CVE-2013-1492 that has unspecified impact and attack vectors, a different vulnerability than CVE-2012-0553. CVE-2003-0780 security issue allows attackers with ALTER TABLE privileges to execute arbitrary code via a long Password field [2]. There are solutions and methods to improve security performance that are listed on the official website of MySQL. When discussing security, it is important to consider protecting the entire server hosts (not just MySQL servers) from all types of applicable attacks: eavesdropping, change, playback, and denial of service. For example, no one (except the MySQL root account) should be allowed access the user table MySQL in the system database! [3]. It is important that when running a client program to connect to a MySQL server, and to use a password that is difficult to decipher. MySQL stores the password for the user account in the mysql.user system table. Therefore, one must not grant access to this table to any non-administrative accounts. Also, MySQL Enterprise Edition includes MySQL Enterprise Firewall, an application-level Firewall that allows database administrators to allow or reject SQL statement execution based on a white list of acceptable statement patterns [2]. MySQL has quite a few security features built into the database. The first set of major features MySQL has is through its functions, such as ENCRYPT, DES_ENCRYPT, AES_ENCRYPT, PASSWORD, OLD_PASSWORD, and ENCODE [4]. These functions allow for data to be encrypted through two different types (DES and AES), and for passwords to be encoded [4]. Another main security feature of MySQL is that it has its own user privilege system for access control [5]. This privilege system provides many features to MySQL's security as a whole. Another security feature is its ability to restrict access from certain users. This is done through the GRANT and REVOKE features of MySQL. The users can range from IP domains to specific computers. Because of this restriction, it becomes extremely hard for hackers to access the data inside the database. These main security features of MySQL make the database hard to hack and retrieve data [5]. MySQL has few security issues. The two main security issues of MySQL tend to coincide with the command LOAD DATA LOCAL. This would be a huge problem as it allows anybody to see what any person has seen on the server. The second issue with the LOAD DATA LOCAL command is dealing with a web environment [6].

## 2. NOSQL
A NoSQL database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. Such databases have existed since the late 1960s but did not obtain the "NoSQL" moniker until a surge of popularity in the early 21st century, triggered by the needs of Web 2.0 companies [7]. One security threat is posed by the nature of the distributed environments as they increase attack surface across several distributed nodes. This is observed, for example, when Malicious data gets propagated from a single compromised location. Protecting nodes, name servers and clients becomes difficult especially when there is no central management security point [8]. The NoSQL database is a highly scalable database that is basically just a giant repository of data. Unlike many other databases, NoSQL provides simple and lose control over its databases. The NoSQL database was initially developed to provide a faster and easier way to access data. The main problem with this is that NoSQL is not made to be secure [10]. NoSQL databases have many security issues because of their lack of security focus. One major security issue is that data is left unencrypted. This basically means that any hacker that can get access to the filesystem can easily read the data. Another major security issue is authentication. When NoSQL databases transmit passwords, they are in plaintext [11].

## 3. ORACLE DATABASE
The Oracle Database is a widely used and highly reliable RDBMS that supports both on-premises and cloud deployment. Cloud-based implementations improve scalability, reduce downtime, and enhance operational flexibility [12]. Due to its extensive use among major corporations, Oracle databases are frequent targets for cyberattacks, prompting Oracle to develop

advanced security features [13]. Key protections include Transparent Data Encryption (TDE), which encrypts entire databases or specific columns to safeguard sensitive information [14], and Data Redaction, which limits data exposure through unsecured applications [15]. The Oracle Key Vault securely manages and stores thousands of encryption keys across multiple databases [16]. Oracle's auditing services enable administrators to monitor activity by IP address, program, or time, with configurable audit policies and separation of duties [17]. The Database Firewall complements auditing by analyzing SQL traffic, detecting suspicious statements, and enforcing compliance with standards such as PCI DSS and HIPAA [18]. Another feature, the Virtual Private Database (VPD), dynamically adds security clauses to SQL queries to restrict access by user role or time period [19]. Oracle's evolution began with Oracle V2 (1979), the first commercial SQL-based RDBMS, progressing to versions such as Oracle 11g, designed for enterprise flexibility [20]. Despite strong security, vulnerabilities remain; for instance, the 2016 MICROS breach demonstrated persistent risks in large-scale Oracle systems [21][22].

# 4. POINT BASE NETWORK SERVER

PointBase Network Server is a Java based database server implementation of a relational database management system, or a Relational Database Management System (RDBMS). This means that to run this database, the user must have a JVM installed on their machine. PointBase can be run on several operating systems including Windows, Macintosh, and Linux. This database can be run from a text-based SQL command prompt or a GUI interface [23]. By default, created users in the PointBase RDBMS do not have any access privileges. The schema owner is the only individual who can grant privileges to view the schema and/or the data objects within it. Privileges can be granted so users can access tables, columns, and SQL procedures and functions. Giving new users no access by default is a good security measure. This prevents confidential data from being stolen by users who shouldn't be able to access it [24]. To give privileges to users, the GRANT statement must be used. If the privilege syntax is incorrect within the GRANT statement, an error will occur. Low-privilege users are not allowed to grant privileges that they do not hold themselves. This prevents privileges being given to those who do not require it. By default, the GRANT statement does not allow read or write privileges to be given to users. Users are only able to execute, not access the tables the SQL function uses [24]. PointBase makes use of different cryptography measures to ensure that the database is secure. There are two different forms of encryption: database encryption and client and server communication encryption. With database encryption, the pages within the database are each encrypted. This encryption can be used with the PointBase Embedded version and with the PointBase Embedded – Server Option [25]. When encryption is used between the client and the server, a random session key is generated by the client and then used in a symmetric stream cipher. As an added security measure, a new session key is generated every time a connection is dropped. Once the connection is established, all communications are encrypted – this also includes the database passwords and data. One drawback of the cryptography used is that there is no algorithm set by default. If the administrator wants to use cryptography within the database, they must set it manually [25].

One unique and convenient feature that PointBase servers contain is a graphical database management console, meaning clients that use PointBase can create and manage database schemas. This gives viewers of a database the logical overview of the entire database in a much more organized manner and graphically displays how the relations among data are associated. There are three different editions of PointBase for connecting to a database. The first is PointBase Micro Edition, which provides a single connection to a database from only one application running on the same JVM. Second is Embedded Edition, the version of Micro Edition that allows multiple connections to a database from a single application. The third is the Server Edition, which provides multiple connections to the database from multiple Java applications. Using the Server Edition of PointBase gives users the ability to access a database over a network. With numerous amounts of clients having convenient accessibility to a server over a network, there are many potential and almost guaranteed security issues/threats that must be regarded [26].

# 5. POSTGRESQL

PostgreSQL is a free open source object-relational database that originated in 1986 at the University of California at Berkeley. This database software can run on several operating systems including macOS, Windows, Solaris, BSD, and Linux. PostgreSQL uses and extends the SQL language and is highly extensible. Users can define their own data types, build custom functions, and write code from different programming languages without having to recompile the database. Aside from C and SQL, there are four procedural languages that can be used: pgSQL, Perl, Tcl, and [27]. There are measures in place to prevent server spoofing, which can only occur when the server is down. If spoofing were to take place, then the spoofed server could read all queries and passwords sent by clients. To prevent spoofing on a local connection, a Unix domain socket directory can be used. In addition, clients can use require peer to specify the required owner of the server. To prevent spoofing on TCP connections, SSL certificates should be used. SSL spoofing can be prevented whenever the server is configured to only accept hostssl connections [28]. PostgreSQL offers several different encryption options. To begin, all passwords are encrypted. By storing passwords such as hash values, which prevents the database administrator from having knowledge of the plaintext password. Certain columns can be set to be encrypted within the database. For example, if some data is sensitive but other data isn't then the sensitive data can be specifically encrypted rather than all data. Data partitions can also be encrypted. If the drive containing the database was stolen, this prevents the individual from gaining access to the data it stores. The only loophole in this scenario is if the file system is mounted. However, the file system can be set to where it requires an encryption key before access is granted. If the key is stored on the host machine, then the system can't be mounted without it [29].

Data sent over the network is encrypted through SSL connections. This includes all passwords, queries, and the data fetched. Administrators are able to select how secure they wish to carry out this feature. Some hosts can be set to use no encryption if there isn't any required. If the administrator prefers to use Stunnel or SSH, those options can be used to encrypt connections as well. As an extra security measure, man in the middle attacks can be prevented by requiring the client and server to provide SSL certificates to each other. This is harder to implement than by just requiring passwords, but it is a better verification of identity on both ends [29].

PostgreSQL was originally known as Ingres, a set of C code developed at the University of California at Berkeley in 1985. The code was later used to produce one of the first commercially used relational database servers, as well as, an object-relational database server known as Postgres. SQL capabilities were added to Ingres and Postgres95, the former version of PostgreSQL was

created. After many eager developers got involved with Postgres95's source code and many bugs and errors were assessed, Postgres95 became PostgreSQL in 1996. The name given is based on the SQL features and capabilities that it contains. The one great thing about open source software is the number of people that can help find and fix bugs that the source code contains, and that was a major issue with database servers like PostgreSQL [30]. Though having a software source code freely and openly available for developers and dedicated users provides many advantages for the software, it also poses many serious security threats due to its widescale availability [31]. From 2015 to 2017, 2228 open source software projects were selected by 360 different security teams in order to test many open source community projects for source code defects. The development languages for these projects included C, C++, C#, Java and many more. Out of the 257,835,574 lines of code detected, there were 2,626,352 source code defects. From the overall data gathered from these testing programs, it is determined that the security issues of OSS are extremely serious requiring security analysis on the source software to assess the security vulnerabilities before downloading and using the source code [31].

## 6. QUADBASE SQL SERVER

Quadbase is an advanced data analytics company that provides Database Management Systems (DBMS) for end users through an SQL server. This GUI (Graphic User Interface) based system is very user friendly, as it protects users from the complexity of databases by offering an easy-to-use user interface. The tools provided allow non-technical users to create queries, build charts, reports, and publish their results with little to no training [32]. Quadbase is a company that is based out of Santa Clara, CA. They specialize in databases and services that help companies to collect data. They democratize data into actionable insights through the Quadbase SQL Server [33]. The Quadbase SQL Server was released in 1989 and was meant to be an SQL Server tutorial that helped to teach more than the average tutorial. The server was a multiuser database management system that supported ANSI SQL and would run with any DOS 3.1 system in the local area network. The users are allowed to add and change records in the database. The biggest thing about this system was that it didn't require a large investment in hardware like other SQL servers did. It was able to share the files on the local network by utilizing existing applications, which meant less training and purchases for a company [34]. The Quadbase SQL Server shipped with three modes, exclusive, shareable, and read-only. Exclusive locked all but the Dquery users out of the files. Sharable caused Dquery users to lose their access when the file was locked by another user. Finally, read-only mode doesn't allow the data to be locked or updated, once it is in the database it will stay that way. When the server came out it cost $795, which was a very reasonable price for all that was offered in the package [34].

## 7. RED BRICK

The Red Brick DBMS is a commercial Relational Database Management System that is designed for data warehouse application, decision support, and querying. This DBMS is a software-only system that is compliant with ANSI specifications, meaning it complies with official features of the SQL language. Due to its architecture, one of Red Brick's most valued features is its high performance in high intensity query applications. To accomplish their performance and scalability, they use parallel processing and other special algorithms. Complex SQL SELECT statements are given most of the workload in these applications. Plus, they only read and do not update the database. This DBMS supports duplicate elimination, aggregation, and sorting. However, the typical Red Brick system can only support 50 to 500 user sessions at a time, which is not bad but would not be ideal for massive companies. Red Brick also maintains historical data for months and even years [35]. A major issue for all databases is the possibility of a Denial of Service (DoS) attack. There is no set way to combat these attacks, but there are many common practices one can do to reduce the frequency. First, one must make sure to reduce the available attack surface to reduce what is available to a potential attacker. This would include removing things not needed, such as unneeded user accounts, communication protocols and services, and unused database features. There is also a need to limit the resources available on a per-user basis, such as queries per minute, query timeouts, and throttling for memory and processors. While doing this cannot stop an attack, this will significantly lessen the impact of an attack on a database. Another important and essential piece of a database is a Database Activity Monitoring platform. These inspect incoming queries from end-users and check if they violate a set policy, and will examine user metadata to detect bad queries. Therefore, the database firewalls are an important security implementation. They can be used to block malicious queries since they sit between an application server and a database [36].

Red Brick is a database software creator that is most well-known for its data warehousing. Data warehousing is most frequently used by businesses to be able to collect a greater amount of information. They are essentially databases for databases. This allows a company to keep the information for a longer time which enables them to be able to see that data over a longer period, showing them things they may not be able to see in a typical database. In 2002 Red Brick was bought by Informix, who was then bought by IBM [37]. The Red Brick database was a majority of the time used for queries, decision support, and data warehouse. The company was very diverse in the database software that they offered. The thing that made their data warehouse software so popular was its algorithm and indexes. This was able to give the software a performance that was well above its competitors at the time [38]. According to IBM's website, the customer is responsible for the security of the software. Meaning that the Red Brick Warehouse system doesn't ship with built-in security. Because of this, the security of the data warehouse is limited by how much the company employing it knows about security. If the business doesn't take the proper security measures in both its hardware and the software of the Red Brick Warehouse it could cause the loss of a significant amount of data. Depending on the business they could even lose a substantial amount of personal data as well as transactional if the security of the warehouse was compromised [39]. Red Brick Warehouse allowed General Mills in 1993 to utilize the speed of the queries at the time. The marketing department become more effective by increasing productivity because they were able to pull up old commercials for review at their desktop, instead of having to go to a separate room to watch the film version of the commercial [40].

## 8. SUPRA PDM

The Supra PDM is NoSQL network-based system that was designed to be scalable, high-volume, with fast batch processing. It is a hierarchical management system that was developed by the company Cincom in the 1960s. It is widely regarded as the first commercial DBMS that was not bundled with hardware or other proprietary software. This DBMS is supported by non-mainframe platforms that include UNIX systems and Open/VMS, but is also supported by IBM mainframes. Supra PDM contains features such as database utilities that are used by coding with Utility Command Language (UCL) and SQL access support. Also, an efficient client Application Programming Interface (API) is vital

to provide simple, yet powerful functions that help an end user to navigate user-defined databases [41].
Supra PDM is a database management software for the mainframe that is distributed by Cincom Systems. The PDM stands for, "Physical Data Manager". It was first developed in the late 1960s under the name, "Total", and is known for being the first commercial database management system that was sold on its own, not with hardware or other software. Along with supporting mainframe it also supported UNIX. Supra PDM supports SQL access and has the ability to be accessed through a Microsoft Windows client [42]. Supra PDM is a modern database management software and is still supported by Cincom today. It is included in their SUPRA system for managing databases. It allows for the management of operational data and it works as a networked, hierarchical, database management system. It is able to be used by multiple clients on the same computer and it can also be accessed through TCP/IP from other computers. It also helps to reduce the overhead on the machines by reducing how much overhead it took to move data to the memory of applications. The database manager also gives access to SQL databases in real-time that are in the SUPRA system [42].

Supra PDM utilizes NoSQL for its system. With this comes a few possible security issues built into the system. By default, there is no user authentication enabled. This obviously can be enabled manually when setting the database up, but if this is something that is overlooked it could lead to a large number of problems, such as unauthorized users getting into the system or even just having access to information that is sensitive and should be hidden. Another possible security issue is the weak password storage. With a weak password system of storage, the passwords are at risk of being revealed by an attacker, giving them access to the system. Companies will have to be sure that they improve the hashing process to ensure that they are safe [43]. NoSQL doesn't support external encryption tools like LDAP. This means that the company cannot use proven encryption tools to help to protect passwords or even their data files. Although this does not directly cause a security issue, it can create one if someone is unfamiliar with the built-in encryption that is provided. Another security concern is that the data that is at rest is unencrypted. This means that when the data is not being used or transmitted it is viewable as plaintext. This could be a problem if an attacker is able to trick the system into believing that the data is at rest, giving them access to the data. Another security issue is that because the database management system uses a form of SQL it is vulnerable to an SQL injection. This malicious code could give an attacker easy access to potentially sensitive data [43].

# 9. TERADATA DATABASE

Teradata Database is a Relational Database Management System (RDMS) developed by Teradata Corporation [44]. Teradata database is often used by organization for mission critical systems. Teradata Database is able to scale large amounts of data and big projects. Teradata is also able to optimize and scale with the data used and has quick response time to queries. Uptime is also one of the main priorities for some enterprise data systems. In addition, some of the enterprise data systems focus on business drivers which can detract for IT security tasks. It provides tasks such as system patch management, secure configuration of systems before use, using strong passwords and having a good security policy in places to make sure the Teradata Database is secure. Some of these policies include the access control policy allowing certain users of the database to access certain files. The Teradata Database is managed by third parties' companies. This may complicate the delivery of important requirements needed for a security database. Like other databases, the DMBS Teradata database can have an insecure default configuration. It is also prone to attack vectors where actors can gain unauthorized access to business-critical resources [44]. This happens when a normal user account is granted all privileges or having an account with deciphered passwords. In addition, having easy policy can allow a low level user to get higher permission. In general, the security issues are insecure credentials such as default credentials, password attacks or hash cracking, secure use of privileged access, assessment of roles and privileges, Identification of sensitive data (especially in relation to GDPR, PCI, and other legislation), and assessment of patch-levels and security updates, assessment of user-defined functions and user capabilities, identification and analysis of application user credentials stored in the database, assessment of the use of encryption (for data at rest or in transit), and assessment of monitoring and logging [44].
One way they provide security is by defining a SPOOL SPACE for each database to prevent inappropriate database operations from exhausting unallocated space in the system, thus affecting the use of other users. There is separation between concept and operating system. When creating a user, permanent space is also defined and SPOOL space, and their meanings are exactly the same. The only difference between a user and a database is that the user has an account that can be used to log in to the system. Therefore, the user has a password and the database has no password [45]. Data Security It is important to implement appropriate controls to protect sensitive data. Data can be vulnerable when transmitted over non-secure networks or when appropriate access controls have not been enabled for stored data. The Teradata Database provides facilities to manage the encryption of sensitive data when transmitted over non-secure networks. Further, row and column-level security can be implemented readily using database views [45].

# 10. UNI DATA

UniData. UniData is a database management system made by Rocket. It is part of a bigger package of software for database management systems called Rocket U2. UniData can be run off Unix, Linux, and Windows operating systems. Rocket states that "UniData, a best-in-class of the MultiValue application platform, for scalable business applications that can evolve. As a stable, secure, fast data server engine with high availability and low administrative overhead, Rocket UniData is ideal for rapid development using modern tools. On-premises or in the cloud, UniData serves nearly a million users worldwide" [46]. The UniData 8.2 version enhanced the logging which helps with compliance with regulatory and SLA requirements. This version also supports Python language which allows users to use popular open source tools that are written in python. Lastly, it adds support for Rocket Discover. This is collaborative business intelligence solution to help business users get the most out of the UniData application data.

There are a number of security issues with the Uni data database. The first security issue is a remote command execution vulnerability. This vulnerability allows a remote attacker exploit this issue to execute arbitrary commands with the system privileges. This vulnerability affects UniData database versions 7.2.12 and prior. Another vulnerability is Denial of service attack. This vulnerability is caused by the unirpc service listing on port 31438 and is affected by various denial of service regarding the access of invalid zones of memory. Despite this, Rocket states that they produce very effective products and have high-security standards. Data is secure wherever it is in transit or being stored on a Uni database. They employ many ways to secure data. First, SSL/TLS secures connections to the MultiValue server and encrypts data in transit for protection from snooping or

eavesdropping. Third parties can also validate and accredit cryptographic modules for FIPS 140-2 compliance [46]. They also use automatic data encryption by providing keys for accessing data on the database. If data is intercepted, attackers will not be able to decipher it. Uni data also offers other security features like role-based security and required DBA maintenance to implement and maintain these security features. In addition, audits within Unidata provides traceability and proof that actions are taken as needed to keep your database secure. Providing complete audit trails to external auditors. Auditing policies can be modified "on the fly" (by authorized personnel only) without restarting the database. One can trap and trace attacks or data misuse originating internally or externally to an organization when they occur [46]. It is important to have a recovery system in place. Rocket UniData HA/DR solution is designed to help recover from an unexpected outage or event that may cause a database to go offline. Using a publisher/multiple-subscriber model, UniData HA/DR replication is flexible, easy to understand, and easy to maintain. HA/DR solutions can even be configured with delays to protect from inadvertent operations mistakes [46].

# 11. VERSANT

The Versant Object Database (VOD) was created by Versant Corporation. The versant database enables developers using object-oriented language to transitionally store their data and information by allowing the object-oriented language to act as the Data Definition Language (DDL) for the database they are using. The Versant Object-Oriented Database can be customized for an application architecture and production environment based on the particular requirement [47]. Some of the features that Versant database can support are complex and high-performance data management, native object persistence, supports database distribution and partitioning. The versant database can support client enterprise applications and servers, but it is limited to how many users can access/use it at one time due to the hardware resources that are provided. This, however, is the reason why most companies that run a versant database put it in the cloud so that the database can scale as needed. The maximum cache size is 2048 shared memory segments. However, the overall database volume can be stored on multiple drives on a server. One can fully create their own Versant database with a list of tools that allows having a maintenance-free database engine to deploy to a mission-critical, multi-server deployment in a hosted environment. One can configure and instrument the Versant Object-Oriented Database to fulfill SLAs requirements. Some of these tools are Management center, Compact, FTS, Async Server, HA backup, and SQL access. A Versant database allows the use of multiple different types of object programming language and framework to help with development kits. Some of the object programming languages that can be used are Java/JPA and C++.

Some of the security issues of Versant database systems include the vulnerability that allows an unauthorized user to access the CaliberRM server via the versant_d service. This vulnerability only affects Versant version 7.0.1.3. This vulnerability was fixed by Versant and the patch is installed in later versions of CaliberRM (2008 and beyond). Users running any version of CaliberRM before 2008 can take one of the following steps to minimize/eliminate the threat to their CaliberRM Server [48]. First upgrade to CaliberRM 008 SP1, that is running Versant version 7.0.4.9 which has patched the vulnerability. Second, block access to port 5109 on your CaliberRM server. All of the Versant Clients used in the CaliberRM application are running from within the CaliberRM Server executable (caliber_srv.exe). If the database is running on the same server as your CaliberRM Server, there is no need to have the versant_d service exposed to the network. By simply blocking port 5109 with a firewall software or router, this will eliminate any this security risk. The CaliberRM Server will function properly with this port blocked. The only way an attacker would be able to exploit this vulnerability is if they were actually logged into the server [48].

# 12. COMPARISION AND ANALYSIS

Table 1 represents a comparative overview of the database systems discussed in this paper. It covers their type, key features, strengths, weaknesses, and typical use cases.

**Table 1: Comparative overview**.

| Database System | Type / Model | Open Source | Main Use Case | Key Strength |
|---|---|---|---|---|
| MySQL | Relational (RDBMS) | Yes | Web apps, CMS | Easy, popular, reliable |
| NoSQL | Non-relational | Varies | Big data, unstructured data | Scalable, flexible |
| Oracle Database | Relational (RDBMS) | No | Enterprise systems | Robust, secure, enterprise-grade |
| PointBase | Relational (Java) | No | Embedded Java apps | Lightweight, portable |
| PostgreSQL | Object-Relational | Yes | Complex apps, analytics | Advanced features, ACID |
| Quadbase SQL Server | Relational | No | Reporting, BI | Strong reporting tools |
| Red Brick | Data Warehouse | No | OLAP, analytics | Fast analytical queries |
| Supra PDM | Network/Relational | No | Legacy enterprise data | Flexible modeling |
| Teradata | Relational (MPP) | No | Big data warehousing | Massive scalability |
| UniData | MultiValue (NoSQL) | No | Retail, legacy apps | Nested data model |
| Versant | Object-Oriented | No | Real-time object systems | Object persistence |

Table 2 analyzes the database systems with regard to data model and architecture. The majority are SQL-based relational systems that are optimized for consistency and structure. Also, NoSQL, MultiValue, and Object-Oriented systems prioritize flexibility and performance for specialized tasks.

**Table 2: Data model and architecture analysis**

| Category | Databases | Analysis |
|---|---|---|
| **Relational (SQL)** | MySQL, Oracle, PostgreSQL, Quadbase, Red | These databases enforce structured schemas with ACID compliance and SQL |

| | | |
|---|---|---|
| | Brick, Teradata, PointBase | queries. They are strong for transactional systems (OLTP) and structured data. |
| **Non-Relational (NoSQL)** | NoSQL (generic), UniData | Schema-free, flexible for semi-structured or unstructured data. Excellent scalability but weaker transactional integrity. |
| **Object-Oriented** | Versant | Integrates directly with object-oriented programming models. Excellent for applications with complex relationships but niche in use. |
| **Network/ Hierarchical** | Supra PDM | Legacy models allowing flexible relationships but require specialized knowledge and are less compatible with modern tools. |

Table 3 analyzes the database systems with regard to performance and scalability. It is observed that Teradata dominates in massive parallel analytics. Oracle and PostgreSQL balance high reliability with moderate scalability. Finally, NoSQL offers linear scalability but sacrifices strict consistency.

**Table 3: performance and scalability analysis**

| Type | Best Performers | Notes |
|---|---|---|
| **Transactional Performance (OLTP)** | Oracle, PostgreSQL, MySQL | Designed for concurrent, consistent transactions. |
| **Analytical Performance (OLAP / Big Data)** | Teradata, Red Brick | Optimized for large-scale data warehousing and analytics. |
| **Real-Time / Embedded** | PointBase, Versant | Lightweight or direct object access makes them suitable for embedded or real-time environments. |
| **Horizontal Scalability** | NoSQL systems, Teradata | Designed to scale across distributed architectures. |

Table 4 analyzes the database systems with regard to flexibility and data complexity handling. For applications dealing with unstructured or hierarchical data, NoSQL, Versant, or UniData outperform traditional RDBMS. However, PostgreSQL offers a hybrid compromise — relational structure with NoSQL-like JSON support.

**Table 4: flexibility and data complexity handling analysis**

| Database | Schema Flexibility | Complex Data Support | Comment |
|---|---|---|---|
| MySQL | Low | Moderate | Good for structured tables only. |
| PostgreSQL | Moderate | High | Supports JSON, arrays, and spatial data. |
| Oracle | Moderate | High | Extensive data type support. |
| NoSQL | Very High | Very High | Ideal for dynamic and nested data. |
| UniData | High | Moderate | MultiValue structure allows nested attributes. |
| Versant | High | Very High | Excellent for deeply linked object models. |

Table 5 describes the enterprise maturity and support. For modern, mission-critical deployments, Oracle, PostgreSQL, and MySQL remain the safest investments. Teradata dominates large-scale analytics, while legacy or niche systems (Supra, Versant, Red Brick) are mostly used in existing enterprise setups, not new deployments.

**Table 5 enterprise maturity and support**

| Database | Maturity | Vendor Support | Ecosystem Strength |
|---|---|---|---|
| Oracle | 5/5 | Excellent | Enterprise standard |
| MySQL | 4/5 | Strong (Oracle + community) | Widely supported |
| PostgreSQL | 4/5 | Strong open-source | Rapidly growing |
| Teradata | 4/5 | Enterprise-grade | Big data ecosystems |
| Red Brick | 2/5 | Legacy | Limited updates |
| Supra PDM | 2/5 | Legacy | Narrow support |
| UniData | 2/5 | Niche | Limited developer base |
| Versant | 2/5 | Specialized | Small ecosystem |
| PointBase | 1/5 | Obsolete | Minimal support |
| NoSQL (e.g., MongoDB, Cassandra) | 4/5 | Active vendors | Cloud-native adoption |
| Quadbase | 2/5 | Moderate | Focused on BI |

Table 6: describes cost and accessibility. Open-source systems dominate cost-effectiveness and accessibility, while proprietary options like Oracle or Teradata target mission-critical enterprise systems with high reliability needs.

**Table 6: cost and accessibility**

| Category | Examples | Cost Profile | Typical Users |
|---|---|---|---|
| **Open Source** | MySQL, PostgreSQL, many NoSQL systems | Free (optional enterprise versions) | Academia, startups, SMBs |
| **Commercial / Licensed** | Oracle, Teradata, Quadbase | High cost | Large enterprises |
| **Legacy / Proprietary** | Supra, UniData, | Moderate to high | Specialized industries |

| | | | |
|---|---|---|---|
| | Versant, Red Brick | | |
| **Embedded** | PointBase | Low cost | Mobile and device vendors |

For General Web / Business Apps, MySQL or PostgreSQL are the best choice because it provides balance between performance and simplicity. For Enterprise Data Management, Oracle, is the best choice because it provides enterprise-grade performance, strong tools. For Big Data / Analytics, Teradata is the best choice because it provides MPP architecture for massive analytics. For Flexible or Unstructured Data, NoSQL systems is the best choice because it provides schema-less, scalable, and distributed. For Embedded Systems, PointBase is the best choice because it is provides lightweight Java integration. For Legacy Data Management, Supra PDM, UniData are the best choice because they are still used in certain manufacturing/retail environments. For Object-Oriented Systems, Versant is the best choice because it has specialized use in engineering or telecom.

Figure 1 is a Visual comparison chart showing ratings (1–5) for performance, scalability, flexibility, and cost-effectiveness across the database systems discussed in this paper. It is observed that Oracle and Teradata lead in performance and scalability. PostgreSQL balances all factors best. NoSQL excels in scalability and flexibility. MySQL is strong overall and very cost-effective. Legacy systems (Supra, Versant, etc.) show moderate but uneven strengths.

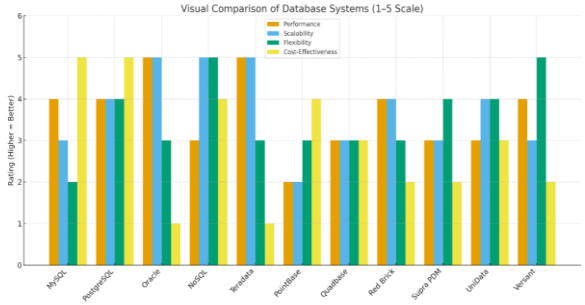


**Figure 1: Visual comparison chart showing ratings (1–5) for performance, scalability, flexibility, and cost-effectiveness**

## 13. CONCLUSION

From this study it is observed that for modern trends, PostgreSQL and cloud NoSQL systems (MongoDB, DynamoDB) are overtaking older proprietary platforms. Oracle and Teradata remain leaders for enterprise and analytics, but at higher operational costs. For legacy systems (Supra, Red Brick, Versant) highlight the evolution of database technology — from rigid network models to today's flexible, distributed architectures. Finally, for hybrid models (like PostgreSQL with JSON or Oracle with XML) now bridge the gap between structured and semi-structured data needs. Future work can extend this study by including emerging cloud-native and distributed databases to assess scalability and cost in real environments. Empirical benchmarking using real datasets could strengthen performance comparisons across workloads. Research may also explore AI-driven query optimization, hybrid relational–NoSQL architectures, and security or energy efficiency aspects. These directions will help refine database selection strategies and guide the development of more adaptive, intelligent, and sustainable data management systems.

## 14. REFERENCES

[1] MySQL. (n.d., para 1). In Wikipedia. Retrieved from https://en.wikipedia.org/wiki/MySQL

[2] Giacomo, M. D. (2005). MySQL: Lessons learned on a digital library. IEEE Software, 22(3), 10-13. Retreived from: doi:http://library.semo.edu:2275/10.1109/MS.2005.71

[3] MySQL. In MySQL. Retrieved from https://dev.mysql.com/doc/mysql-security-excerpt/5.7/en/security-guidelines.html

[4] George, B., & Valeva, A. (2006). A database security course on a shoestring. Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education - SIGCSE 06. doi: 10.1145/1121341.1121347

[5] Tomic, S. T., & Janata, P. (2007). Ensemble: A web-based system for psychology survey and experiment management. Behavior Research Methods, 39(3), 635-50. Retrieved from https://library.semo.edu:2443/login?url=https://library.semo.edu:4836/docview/204303959?accountid=38003

[6] Michael "Monty" Widenius, and David Axmark. MySQL Reference Manual : Documentation from the Source. Sebastopol, Oŕeilly & Assciates, 2002.

[7] NoSQL. (n.d., para1, para2, para3). In Wikipedia. Retrieved from https://en.wikipedia.org/wiki/NoSQL#cite_note-6

[8] Leonardo Aniello, Silvia Bonomi, Marta Breno, and Roberto Baldoni. 2013. Assessing data availability of Cassandra in the presence of non-accurate membership. In Proceedings of the 2nd International Workshop on Dependability Issues in Cloud Computing (DISCCO '13). Association for Computing Machinery, New York, NY, USA, Article 2, 1–6. https://doi.org/10.1145/2506155.2506157

[9] Lane, A. (2012, February 6). A Response To NoSQL Security Concerns. Retrieved from https://www.darkreading.com/application-security/database-security/a-response-to-nosql-security-concerns/d/d-id/1137044.

[10] Monger, M. D., Mata-Toledo, R., & Gupta, P. (2012). TEMPORAL DATA MANAGEMENT IN NOSQL DATABASES. Journal of Information Systems & Operations Management, 6(2), 1-7. Retrieved from https://library.semo.edu:2443/login?url=https://library.semo.edu:4836/docview/1346772131?accountid=38003

[11] L. Okman, N. Gal-Oz, Y. Gonen, E. Gudes and J. Abramov, "Security Issues in NoSQL Databases," *2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, Changsha, 2011, pp. 541-547.

[12] Oracle Database. (2019). Retrieved from https://www.oracle.com/database/

[13] Oracle Database Security. (2019). Retrieved from https://www.oracle.com/database/security/

[14] Oracle Advanced Security Transparent Data Encryption. (2018, March). Retrieved from https://www.oracle.com/technetwork/database/options/advanced-security/overview/advanced-security-tde-faq-2995212.pdf

[15] Advanced Security. (2019). Retrieved from https://www.oracle.com/database/technologies/security/advanced-security.html

[16] Oracle Key Vault. (2019). Retrieved from https://www.oracle.com/a/tech/docs/dbsec/okv/faq-security-key-vault-2019-05-01.pdf

[17] Database Auditing. (2019). Retrieved from https://www.oracle.com/database/technologies/security/db-auditing.html

[18] Audit Vault and Database Firewall. (2019). Retrieved from https://www.oracle.com/database/technologies/security/audit-vault-firewall.html

[19] Real Application Security and Virtual Private Database. (2019). Retrieved from https://www.oracle.com/database/technologies/security/virtual-private-db.html

[20] Docs.oracle.com. (2019). Introduction to Oracle Database. [online] Retreived from: https://docs.oracle.com/cd/E11882_01/server.112/e40540/intro.htm#CNCPT965

[21] Shaul, J., & Ingram, A. (2007). Practical Oracle Security : Your Unauthorized Guide to Relational Database Security. Rockland, MA: Syngress. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&db=e000xna&AN=227541&site=ehost-live

[22] Krebsonsecurity.com. (2019). Data Breach at Oracle's MICROS Point-of-Sale Division — Krebs on Security. Retreived from: https://krebsonsecurity.com/2016/08/data-s breach-at-oracles-micros-point-of-sale-division/

[23] PointBase Database Server. (n.d.). Retrieved from http://books.gigatux.nl/mirror/beaweblogic8.1/0672324873_ch10lev1sec2.html

[24] SQL Security and Privileges. (n.d.). Retrieved from http://www.cs.toronto.edu/~nn/csc309-20085/guide/pointbase/docs/html/htmlfiles/authorizationFIN.html

[25] PointBase System Guide. (2004). Retrieved, from https://docs.oracle.com/cd/E19253-01/817-7465/817-7465.pdf

[26] Artiges, M., & Marin, J. (2004). Bea WebLogic server 8.1 unleashed. Indianapolis, IN: Sams.

[27] About: What is PostgreSQL. (2019). Retrieved from https://www.postgresql.org/about/

[28] Preventing Server Spoofing. (2019). Retrieved from https://www.postgresql.org/docs/12/preventing-server-spoofing.html

[29] Encryption Options. (2019). Retrieved from https://www.postgresql.org/docs/12/encryption-options.html

[30] Momjian, B. (2001). PostgreSQL: introduction and concepts. Boston: Addison-Wesley.

[31] Li, Y., Ma, L., Shen, L., Lv, J., & Zhang, P. (2019). Open source software security vulnerability detection based on dynamic behavior features. PLoS ONE, 14(8), 1–14. Retreived from https://doi.org/10.1371/journal.pone.0221530

[32] Quadbase. (n.d.). Retrieved from https://www.quadbase.com/espressreport-es-description/.

[33] Quadbase. (n.d.). About Us. Retrieved from https://www.quadbase.com/about-us/.

[34] Mage, S. (1989, August 28). Quadbase Introduces DBMS Quadbase-SQL. InfoWorld, 19–19.

[35] Fernandez, P. (n.d.). Red Brick Warehouse. Retrieved from https://www.semanticscholar.org/paper/Red-Brick-Warehouse:-A-Read-Mostly-RDBMS-for-Open-Fernandez/943051a9caded9b0aa5ae17840ddd8ffc5ddfac6.

[36] Bristow, Paul. "Database DoS." Juniper Networks, 29 Jan. 2014, Retreived from: https://forums.juniper.net/t5/Security/Database-DoS-Part-3-Countermeasures-to-Help-Reduce-the-Impact-of/ba-p/227315.

[37] Informatica. (n.d.). Data Warehouse: What It Is, Meaning & Definition. Retrieved from http://www.informatica.com/services-and-training/glossary-of-terms/data-warehousing-definition.html.

[38] Fernandez, P.M. (1994). Red Brick Warehouse: A Read-Mostly RDBMS for Open SMP Platforms. SIGMOD Conference.

[39] IBM. (2004, April 13). IBM Red Brick Warehouse Server V6.30 enhances query performance, improves ease of use, and delivers new platform support. Retrieved from https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=ca&infotype=an&appname=iSource&supplier=897&letternum=ENUS204-064.

[40] Garside, W. (2000, March 9). Case Study: Redbrick database analysis. Retrieved from https://www.computerweekly.com/feature/Case-Study-Redbrick-database-analysis.

[41] SUPRA Server PDM for the Mainframe. (n.d.). Retrieved from https://www.cincom.com/pdf/DB041026-1.pdf.

[42] Cincom. (2013). Cincom SUPRA PDM for UNIX and Linux. Cincom SUPRA PDM for UNIX and Linux. Cincinnati, OH: Cincom.

[43] Perficient. (2015, June 23). NoSQL NoSecurity - Security issues with NoSQL Database. Retrieved from https://blogs.perficient.com/2015/06/22/nosql-nosecuity-security-issues-with-nosql-database/.

[44] Williams, B. (2018, July 20). Securing Teradata Database. Retrieved from https://www.nccgroup.trust/uk/about-us/newsroom-and-events/blogs/2018/july/securing-teradata-database/.

[45] Jim Browning and Adriaan Veldhuisen . Security Features in Teradata Database. Retrieved from https://www.teradataemea.com/campaign/pdf/EB1895.pdf

[46] Rocket UniData. (n.d.). Retrieved from https://www.rocketsoftware.com/products/rocket-unidata-0/rocket-unidata

[47] Versant Object Database. (2019). Retrieved from https://supportservices.actian.com/versant/vod

[48] There is a vulnerability issue in the Versant Database (Version 7.0.1.3) which allows unauthorized access to the CaliberRM Server via the versant_d service. Retrieved from https://community.microfocus.com/t5/Caliber-Knowledge-Base/There-is-a-vulnerability-issue-in-the-Versant-Database-Version-7/ta-p/1749280