IS5102

A3

Group I

190028058, 200010118,
200009834, 200033682

Table of Contents

Introduction

The purpose of this report is to compare SQLite with other common SQL implementations. The first section of this report will introduce SQLite, the framework through which SQL implementations will be compared, and the four SQL database technologies which have been selected to be compared to SQLite. Sections two to five will apply the framework discussed above to compare SQLite with four different common SQL implementations. The sixth and final section will combine the results from the comparative sections to provide the reader with an understanding of the relative strengths and weaknesses of SQLite and the specific contexts in which SQLite is an optimal choice.

History of SQLite

Richard Hipp created the first version of SQLite in the year 2000 (JavaTPoint, 2018). The philosophy which underpinned the creation of SQLite was that the software development community was lacking a database technology which was "light weight" (concretely defined as "less than 500Kb in size"), easy to maintain, and required little to no administration (Ibid.). The first version of SQLite shipped in August 2000 as part of the "GNU database manager" (Ibid.). As of November 2020 the most current version of SQLite is version 3.33.0 which was released on the 14th of August 2014 (SQLite, 2020). Twenty years after release SQLite has been adopted by large companies such as Apple, Google, Facebook, Adobe, and Microsoft while also ranking third on the explore group list of "most popular database technologies" (SQLite, 2020; explore group, 2019).

Databases Selected for Comparison

We have made the decision to compare the following databases to SQLite: Postgres, MySQL, Oracle Database, and HiveSQL. This set of databases was selected as it represents a set of popular databases which span the gamut from extensible open source software to closed-source monolithic enterprise database systems. This wide spectrum of databases was selected to assist the user in understanding the advantages and disadvantages of SQLite in a broader context.

Comparative Database Framework

To ensure methodological consistency across our comparisons of databases, we have developed the following comparative framework structure which will be applied in sections two through five:

1. Introduction to Postgres/Oracle DB/MySQL/HiveSQL.
2. Comparison of Supported Datatypes.
3. Advantages of Postgres/Oracle DB/MySQL/HiveSQL when compared to SQLite
4. Disadvantages of Postgres/Oracle DB/MySQL/HiveSQL when compared to SQLite.
5. When to consider using Postgres/Oracle DB/MySQL/HiveSQL instead of SQLite.
6. When to consider using SQLite instead of Postgres/Oracle DB/MySQL/HiveSQL.
7. Summary of Comparative Analysis.

The seven steps of the comparative framework will be applied to each of the four databases described above. This will enable a high-level comparison of comparisons in the sixth and final section of this report. Our aim is to conclude the report with a descriptive set of mental models which the reader can apply to determine which database system to implement in a given engineering scenario.

Structured Comparison of SQLite and PostgreSQL (Postgres)

Introduction to Postgres

Postgres is an “open source relational database management system [RDMBS]” which was developed around the key principles of “extensibility and standards compliance” (Fujitsu, 2020). This database technology began as a DARPA-funded project at the University of California Berkeley in 1986 (PostgreSQL, 2020). The Postgres project underwent three major revisions in 1994: it was renamed to Postgres95, SQL support was added, and it was declared to be open source software (Ibid.). In 1996 the project was renamed to PostgreSQL 6.0 which laid the foundation for the contemporary versions of the Postgres RDBMS (Ibid.). Shifting from

the past to the present, the value proposition of modern Postgres is structured around the core tenets of “reliability, data integrity, [a] robust feature set, [and] extensibility” in addition to the fact that Postgres is both open source and compatible with “all major operating systems” (PostgreSQL, 2020). The purpose of this section is to compare SQLite to Postgres in a structured manner. The aim of this comparison is to assist the user in developing an understanding of the relative advantages and disadvantages of SQLite when compared to Postgres.

Supported Data Types

The first dimension on which SQLite will be compared to Postgres is the breadth and variety of datatypes which are supported by both databases. In this context, SQLite falls short as it officially supports the following five datatypes: “null, integer, real, text, blob” (SQLite, 2020). By comparison, Postgres supports a significantly broader class of datatypes such as: “numeric, monetary, character, binary, date/time, Boolean, enumerated, geometric, network address, bit string, text search, UUID, XML, JSON, arrays, [and] composite” data types (PostgreSQL, 2020; TablePlus, 2018).

Advantages of Postgres

SQL Standards Compliance

A key advantage of Postgres is that it is highly compliant with existing SQL standards (Drake, 2019). According to the publicly available documentation for Postgres version 13, Postgres is compliant with “at least 170 […] out of 177 mandatory features required for full core conformance” with the SQL:2016 standard (PostgreSQL, 2020). By comparison, SQLite falters in the context of compliance as it does not implement basic SQL functions such as: “right and full outer join(s) […] alter table […] trigger(s) […] writing to views [and] grant and revoke” (SQLite, 2020).

Strong Foundational Security Through Authentication

The Computer Security Resource Center of the US National Institute of Standards and Technology (NIST) defines authentication as the process of "verifying the identity of a user […] often as a prerequisite to allowing access to resources in an information system" (CSRC, 2020). In the context of security and authentication, Postgres is superior to SQLite as it implements the following eleven types of authentication: "trust, password, GSSAPI, SSPI, Kerberos, Ident, peer, LDAP, RADIUS, certificate, [and] PAM" (PostgreSQL, 2020). By comparison, SQLite databases can be read by "anyone with access to the database file itself" (Leifer, 2018).

Management of Database Roles and Users

Postgres enables database administrators to "create multiple roles with specific sets of permissions" which individual users can subsequently be added to (Raja, 2019). This enables database administrators to clearly dictate both the specific tables and the read/write privileges which individual users, classes of users, or systems in the organisation have access to (Ibid.). By comparison, SQLite does not support the management of users and roles, and as such it is not possible to grant "permissions at the database level" (Copes, 2020). The lack of user management and authentication which are key disadvantages of SQLite are largely a by-product of the SQLite core architecture which will be explained in greater detail below (Ibid.).

Disadvantages of Postgres

Complexity of Setup

It is imperative to note that the fundamental architectures of SQLite and Postgres differ significantly. The architecture of SQLite is built around the principles of "embedded database(s)" (TablePlus, 2018). In concrete terms, this means that a SQLite database is a standalone .db file which is "server-less and can run within [an] app" (Ibid.). By comparison, Postgres is built around the tenets of the "client-server model" (Ibid.). This means that a Postgres database requires "a DB server to set up and run over the network" (Ibid.). The impact of this is that SQLite databases are significantly easier to set up, maintain, and

implement into an existing application when compared to Postgres. However, as put forth above, the flexibility afforded by SQLite does give rise to significant tradeoffs in the context of SQL compliance, functionality, and security.

Portability

A closely related disadvantage to the discussion above is the notion that Postgres lack in the context of portability when compared to SQLite (TablePlus, 2018). As a consequence of its serverless architecture, SQLite databases can be easily transferred between hosts or applications as they are entirely contained in one file (Ibid.). As an example: a developer can easily replace the SQLite database accessed by a Python script in production by simply using SFTP to transfer the new database.db file from their host environment to the production virtual machine. By comparison, the same process in Postgres would require the developer to export the existing Postgres database "to a file" and then upload this file to the Postgres database server which the production Python script was accessing (Ibid.). Although portability is presented as a disadvantage of Postgres when compared to SQLite, this may equally be seen as a strength of Postgres. As an example: in mission-critical systems the portability of a database may be a liability as opposed to a key advantage.

When to Consider using Postgres instead of SQLite

The official SQLite website suggests that developers should not consider using SQLite as a database solution for "client/server applications", "high-volume websites", "very large datasets", and for supporting those applications which require a high degree of concurrency (SQLite, 2020). As such, if the project which a developer is working on requires a database which is robust, supports a wide array of datatypes, is highly compliant with SQL standards, and adheres to advanced authentication and permissions management principles then Postgres may be the optimal choice of relational database management system.

When to Consider using SQLite Instead of Postgres

The official SQLite website suggests that developers should consider using SQLite as a database solution for “embedded devices and internet of things”, as an “application file format”, for simple “websites”, and for the purpose of “data analysis” (SQLite, 2020). As such, if the project which a developer is working on requires a database which is small, easy to set up and manage, highly portable, and does not require advanced security and permissions management then SQLite may be the optimal choice of relational database management system.

Summary

This section has formally introduced Postgres as a contemporary database technology. Given the relative and context-specific advantage and disadvantages of Postgres when compared to SQLite it is not possible to provide a definitive recommendation of one database system over the other. We encourage the reader to thoroughly consider the context within which a relational database management system will be used prior to making a decision regarding which database to implement.

Structured Comparison of SQLite and HIVE (HIVE)

Introduction to HIVE

The Apache Hive™ data warehouse software helps to read, write and manage large data sets residing in distributed storage and querying using SQL syntax. (apache, 2020) It is a data warehouse software project based on Apache Hadoop. HIVE is used to provide data query and analysis. (Venner, 2009) Hive provides query functions similar to SQL to query data stored in various databases and file systems integrated with Hadoop. Hive is open-sourced by Facebook to solve the data statistics of massive structured logs. The essence of HIVE is to transform HQL into MapReduce program

Context of HIVE

Hive was originally developed by Facebook, and later the Apache Software Foundation began to use it and further developed it as an open source under the name Apache Hive. It is widely used in many companies. For example, Amazon uses it in Amazon Elastic MapReduce. (tutorialspoint, 2020). Massive amounts of data are very expensive to maintain for traditional relational databases. The Hadoop distributed framework can use inexpensive machines to deploy distributed systems to store data on HDFS and perform calculations and analysis through MapReduce. However, the inconvenience caused by MapReduce makes programming very cumbersome. In most cases, each MapReduce program needs to include Mapper, Reducer, and a Drive. We need to package it as a jar package and update it on the cluster to run. If MapReduce is written and the project is already online, once the business logic changes, it may bring about large-scale changes to the code. Re-package, release the project will be very troublesome (XIAOYUZHOU, 2018). How to quickly perform statistical analysis operations on files on HDFS when a large amount of data is stored on HDFS? In this context, either we learn to generate MapReduce or use HIVE.

Difference Between HQL and SQL

HQL (Hibernate Query Language) is an object-oriented query language, it is a SQL-like query language. HQL is the most widely used retrieval method. HQL is an object-oriented query method. Hibernate is responsible for parsing HQL query statements, and then translates HQL query statements into corresponding SQL statements according to the mapping information in the object-relational mapping file. The subject in the HQL query statement is the class and its attributes in the domain model. (CSDN.MR YANG, 2017)

Supported Data Types

The breadth and variety of datatypes supported by SQLite are much small than HIVE. SQLite supports the following five datatypes: “null, integer, real, text, blob” (SQLite, 2020) comparing with HIVE supports: “Numeric types, Date types, String types, Misc types, Complex types, Column types including Integral types, strings, varchar, char, Timestamps, Dates, etc.” (apache, 2020)

Advantages of HIVE

SQL Standards Compliance

The first advantage of HIVE is the compliance of many versions of SQL. According to the publicly available documentation for HIVE, Hive provides standard SQL functionality, including many higher versions of SQL:2003, SQL:2011, and SQL:2016 features for analytics. (apache, 2020)

Easy to Learn

HIVE can help avoid to write MapReduce and reduce the learning cost of developers. The operation interface adopts SQL-like syntax to provide developers with the quick development capabilities.

Processing Big Data

Hive's advantage lies in processing a large amount of data. Hive aims to maximize scalability (scale out by dynamically adding more computers to the Hadoop cluster), performance, scalability, fault tolerance, and loose coupling of input formats. (apache, 2020) SQLite is committed to providing local data storage for individual applications and devices. SQLite emphasizes economy, efficiency, reliability, independence and simplicity. (SQLite, 2020) And it is mainly used in the single embedded device which cannot be seen as handle the problem about the very large dataset as it said in its official website: “An SQLite database is limited in size to 281 terabytes (247 bytes, 128 tibibytes). And even if it could handle larger databases, SQLite stores the entire database in a single disk file and many filesystems limit the maximum size of files to something less than this.” (SQLite, 2020)

User-defined Functions

Hive supports user-defined functions, and Users can realize their functions according to their needs. According to its public document, Hive SQL can also be extended to user code through user-defined functions (UDFs), user-defined collections (UDAFs) and user-defined table functions (UDTFs). (apache, 2020)

Disadvantages of HIVE

Processing Small Data

Hive's advantage lies in processing big data rather than in processing small data. SQLite works great as the database engine for most low to medium traffic websites and Embedded devices and the internet of things like cellphones, televisions, cameras, set-top boxes, watches, kitchen tools, automobiles, machine tools, aeroplanes, remote sensors, drones, medical devices, and all kinds of robots. (SQLite, 2020) Therefore, comparing with SQLite in processing small data, Hive emphasizes scalability and is inefficient.

Expression Ability is Limited

The iterative algorithm cannot be expressed in HIVE but can be found in SQLite. It also does not support UPDATE, non-equivalent connection, DELETE, INSERT, etc.

High Latency

Hive generates MapReduce jobs, which are usually not optimized enough. Therefore, HIVE has high latency and is not suitable for the real-time query. Since MapReduce is automatically generated, it is also difficult to optimize.

Complexity of Setup

An SQLite database is a standalone .db file which means it does not require any installation and configuration. Users just need to copy the SQLite library in their computer to create a database. And SQLite has both cross-platform and portability features. , SQLite uses the Public Domain protocol, users will not have copyright disputes when using SQLite. Comparing with SQLite, HIVE needs do much more work on the installation and environment configuration.

When to Consider Using HIVE instead of SQLite

Since Hive uses the SQL-like query language HQL (Hive Query Language), it is easy to understand Hive as a database. From a structural point of view, apart from both HIVE and RDBMS have similar query languages, they completely different in details. Comparing with that DBMS can be used in online applications, Hive is designed for data warehouse. Hive is very suitable for analysis and stores a large amount of data. Also, HIVE is easy to expand its storage capacity and computing power. Therefore, in occasions where real-time requirements are not high, we can use the HIVE to do complex statistical analysis.

When to Consider Using SQLite instead of HIVE

SQLite works well in Embedded devices and the internet of things, Application file format, most low to medium traffic websites, Cache for enterprise data, Server-side database, Data transfer format, Internal or temporary databases. (SQLite, 2020) In these fields, SQLite is more suitable.

Summary

Hive is a data warehouse software built on Hadoop. It can structure the stored data. It provides query statements similar to SQL HiveQL to analyze and process data. Hive converts HiveQL statements into a series of MapReduce jobs and executes them. Users can easily use the command line and JDBC program to connect to the HIVE. Currently, HIVE not only supports

the MapReduce computing engine but also supports two distributed computing engines, Spark and Tez. (XIAOYUZHOU, 2018) HIVE is often used for offline data analysis and processing.

Structured Comparison of SQLite and MySQL

Introduction to MySQL

MySQL is a lightweight relational database management system developed by the Swedish MySQL AB company and currently belongs to the Oracle company (Oracle, 2020). At present, MySQL is widely used in small and medium-sized websites on the Internet. Due to its small size, high speed, low total cost of ownership, open source, and free, the development of small and medium-sized websites generally chooses Linux + MySQL as the website database (Hongyu, 2019). MySQL is a relational database management system (Oracle, 2020). The relational database stores data in different tables instead of putting all data in a large warehouse, which increases speed and flexibility.

Supported Data Types

MySQL supports a variety of types, which can be roughly divided into three categories: numeric, date/time, and string (character) types (Oracle, 2020).

MySQL supports all standard SQL numeric data types.

These types include strict numeric data types (INTEGER, SMALLINT, DECIMAL, and NUMERIC), and approximate numeric data types (FLOAT, REAL, and DOUBLE PRECISION).

The keyword INT is a synonym for INTEGER, and the keyword DEC is a synonym for DECIMAL.

The BIT data type saves bit field values and supports MyISAM, MEMORY, InnoDB and BDB tables.

As an extension of the SQL standard, MySQL also supports the integer types TINYINT, MEDIUMINT, and BIGINT.

MySQL Features

MySQL is a widely used database with the following characteristics (Zhuque, 2019):

1. Written in C and C++ and tested with a variety of compilers to ensure the portability of the source code.

2. Support AIX, FreeBSD, HP-UX, Linux, Mac OS, Novell Netware, OpenBSD, OS/2 Wrap, Solaris, Windows and other operating systems.

3. APIs are provided for multiple programming languages. Programming languages include C, C++, Python, Java, Perl, PHP, Eiffel, Ruby, Tcl, etc (Junyan, et al., 2009).

4. Support multi-threading, make full use of CPU resources.

5. Optimized SQL query algorithm, effectively improving query speed.

6. It can be used as a separate application in the client-server network environment and can also be embedded into other software as a library to provide multi-language support. Common encodings such as GB 2312 and BIG5 in Chinese and Shift_JIS in Japanese Etc. can be used as data table name and data column name (Junyan, et al., 2009).

7. Provide multiple database connection methods such as TCP/IP, ODBC and JDBC.

8. Provide management tools for managing, checking, and optimizing database operations.

9. Can handle large databases with tens of millions of records.

Context of MySQL

MySQL is designed for speed and reliability, but at the cost of fully following standard SQL. MySQL developers have been working hard to adhere to standard SQL more strictly, but it still lags behind other SQL implementations. It is true, but it comes with a variety of SQL modes and extensions that make it closer to compliance. Unlike applications that use SQLite, applications that use a MySQL database access it through a separate background resident process. Because the server process is located between the database and other applications, it can better control who has access to the database (Denton & Peace, 2003).

MySQL has inspired a large number of third-party applications, tools, and integrated libraries, which extend the functionality of MySQL and make it easier to use. Among these third-party tools, phpMyAdmin, DBeaver and HeidiSQL are widely used.

Advantages of MySQL

Compared with large databases such as Oracle, DB2, SQL Server, etc. MySQL has its own shortcomings, such as small scale and limited functions (the functions and efficiency of MySQL Cluster are relatively poor), but this does not reduce it in the slightest (Zhuque, 2019). For general individual users and small and medium-sized enterprises, MySQL provides more than enough functions, and because MySQL is open-source software, it can greatly reduce the total cost of ownership. At present, the popular website architecture method on the Internet is LAMP (Linux+Apache+MySQL+PHP), which uses Linux as the operating system, Apache as the Web server, MySQL as the database, and PHP as the server-side script interpreter.

Disadvantages of MySQL

Known limitations: Because MySQL is designed to speed and ease of use, rather than fully complying with the SQL standard, it has certain functional limitations. For example, lack of support for the FULL JOIN clause (Kala, 2020).

License and proprietary features: MySQL are a dual-licensed software, a free and open-source community edition licensed under GPLv2, and several paid commercial editions released under a proprietary license. Therefore, some functions and plug-ins can only be used in the proprietary version.

The development speed has slowed down: Since the MySQL project was acquired by Sun Microsystems in 2008 and subsequently acquired by Oracle in 2009, users have been complaining that the development process of DBMS has slowed down significantly, because the community no longer has institutions to quickly respond to problems and implement them change (Kala, 2020).

When to Consider Using MySQL instead of SQLite

Distributed operation: MySQL's support for replication makes it an excellent choice for setting up distributed databases, such as primary-secondary or primary-primary architecture (Dawodi, et al., 2019).

Websites and web applications: MySQL provides support for many websites and applications on the Internet. This is largely due to the ease of installing and setting up the MySQL database, as well as its overall speed and scalability in the long run.

Expected future growth: MySQL's support for replication helps promote horizontal expansion. In addition, upgrading to the commercial version of MySQL is a relatively simple process, such as a MySQL cluster that supports automatic sharding, which is another horizontal scaling process.

When to Consider Using SQLite instead of MySQL

According to the official SQLite documentatation:

“SQLite is not directly comparable to client/server SQL database engines such as MySQL, Oracle, PostgreSQL, or SQL Server since SQLite is trying to solve a different problem. Client/server SQL database engines strive to implement a shared repository of enterprise data. They emphasize scalability, concurrency, centralization, and control. SQLite strives to provide local data storage for individual applications and devices. SQLite emphasizes economy, efficiency, reliability, independence, and simplicity. SQLite does not compete with client/server databases” (SQLite, 2020).

SQLite should be able to perform outstandingly in small and medium-sized website application scenarios because according to Hongyu’s essay SQLite has these advantages (Hongyu, 2019):

1. Compared with MySQL, it is more completely free, and there are no restrictions on use.

2. Very compact, no configuration is required to support SQLite in PHP5 and above.

3. No need to purchase database service separately, no server process, zero configuration cost.

4. The entire database is stored in a single file, data import, export, backup and recovery are all copied files, and the maintenance difficulty is zero.

5. The reading speed is fast, and the speed is faster when the amount of data is not very large. More importantly, it saves a database remote connection without complicated authorization verification and can be operated after opening it (Junyan, et al., 2009).

Summary

In general, MySQL and SQLite do not belong to the same category of database language design ideas. So, their application scenarios are not very the same. For most cases, it is necessary to specify the relevant database according to the needs of the user. But not blindly choose one database.

Structured Comparison of SQLite and Oracle Database (Oracle DB)

Introduction

ORACLE is a large relational database based on a high-level structured query language (SQL) that manipulates large collections of regular data in a language that facilitates logical management (Oracle, 2020). It is widely used in areas such as enterprise data processing, management information systems and e-commerce. Because of its superior performance in terms of data security and integrity control, as well as its ability to interoperate data across operating systems and hardware platforms, more and more users are willing to use the Oracle database management system as a processing system for application data. As a general-purpose database system with a complete range of data management functions it includes storing large amounts of data, concurrency control, defining and manipulating data, integrity control, security control, interfacing with high-level languages, fault recovery, etc (Oracle, 2020). Oracle Database is also a distributed database system which supports a variety of distributed functions, particularly Various Internet processing (Alpati, 2005).

The main features of the Oracle database are:

1. Easy and safe backup. On the utility side it provides a strong ability to back up data.

2. Distributed processing. Distributed processing gives users access to data distributed on different computers in diverse cities.

3. Parallel access. Parallel queries enable the user to make use of queries on a single computer with several central processing units.

4. Cross-platform. Oracle is available on every relevant platform.

Advantages

Grouping Transactions

When the load on a database increases to a certain level, its read and write performance becomes very challenging and expansion of the database needs to be considered. Oracle DB can process multiple transactions in the same batch, which makes it more scalable than other SQL versions (Oracle, 2020). Most sequential versions of SQL can only be expanded vertically, which is more costly than horizontal expansion as it increases server memory and hard disk capacity (Sullivan, 2019). An oracle database can have multiple instances to access the database on shared storage, which is able to improve processing efficiency (Oracle, 2020).

Simultaneous Management of a Database by Multiple Servers

In addition to batch processing of transactions, Oracle DB also offers a distributed approach to improving database performance. In an application environment, multiple servers can use and manage the same database in order to spread the workload of each server. The hardware requires at least two servers, and a shared storage device. Two types of software are also required, one for clustering and the other for the RAC (Real Application Clusters), component of the Oracle database. In addition, all servers should be of the same sort of OS. according to the load balancing configuration policy, when a client sends a request to the listener of a service, this server, according to the load balancing policy, will send the request to the local RAC component to process and may also be sent to another server's RAC component to

process the request, after processing the request, the RAC will be sent through the Clustering software is used to access the shared storage devices (Oracle, 2020).

In terms of logical structure, each node participating in the cluster has an individual instance, which accesses the same database. The nodes communicate with each other via the communication layer of the clustering software. In order to reduce IO consumption, there is a global caching service, so that each database instance retains an identical copy of the database cache (Sullivan, 2019). The use of the RAC function on the same database with multiple servers can significantly increase the processing power of the server over the database.

Portability

Oracle DB can run on all major platforms (including Windows) and can be ported between them. It also supports most industry standards. Its open strategy allows customers to choose the most suitable solution, as well as full support for developers (Oracle, 2020). Installing oracle DB on a Unix server, for example, allows users to benefit from the reliability of Unix while maintaining the standardised SQL instructions.

Database Security

Oracle's security measures cover three main areas: User identification and authentication, Authorisation and checking mechanisms, and Auditing techniques (the use of which can be flexibly chosen by the user). In Oracle, the outermost layer of security is for the user to identify their name, which is then verified by the system, Oracle allows the user to be identified three times and if this is not done, the system automatically quits. Oracle's privileges include system privileges and database object privileges of two types, using a decentralised authorisation mechanism, i.e., the DBA is responsible for granting and reclaiming system privileges, and each user grants and reclaims privileges to the database objects they create. In Oracle, auditing is divided into user level auditing and system level auditing. User level audits are audits that can be set up by any Oracle user and are primarily user audits of the database tables or views that the user has created, recording all successful and/or unsuccessful access

requests by all users to these tables or views as well as various types of SQL operations (Oracle, 2020).

In addition to system-level security measures, Oracle also allows users to define special, more complex user-level security measures using database triggers. For example, specifying that the Student table can only be updated during working hours. In summary, Oracle provides a wide range of security measures and provides multi-level security checks that are independent of the security mechanisms of the operating system, and the data dictionary plays an important role in Oracle's security authorisation and checking and auditing techniques (Oracle, 2020).

Safe Data Backup

The backup technique of Oracle DB server, Oracle Data Guard, provides an effective, extensive, and highly available disaster recovery solution. Its easy-to-manage switching and failover capabilities allow roles to be switched between the primary and backup databases, minimising downtime for planned and unplanned outages of the primary database. With the Data guard in action, physical damage at the storage level on the primary database does not spread to the backup database, and similarly, logical corruption or user errors that would cause permanent damage to the primary database can be resolved. Finally, the recreated data needs to be validated before it is applied to the backup database. If the connection between the primary and backup databases is lost and the backup database is unable to receive the new data, once the connection is re-established, the data guard can automatically detect lost archive redo log files and then automatically transfer the lost archive redo logs to the backup database without any manual intervention to synchronise the two (Oracle, 2020).

Disadvantages of Oracle DB

Cost

The cost of Oracle DB is at a disadvantage compared to SQLite. Firstly, there are software licence fees for Oracle. As Oracle DB is a database management software for business applications, not only is it not free, but the licence fees are ten times higher than those for many SQL database management software (Reference*, 2020). Secondly, due to the highly

complex and specialised nature of oracle DB, finding a suitable database DB administrator can be several times more expensive than an SQLite administrator.

Difficulty

One of the main disadvantages of Oracle databases is their complexity. User-friendly versions such as SQLite are easy to install and set up with minimal customisation. The oracle database is much more difficult (Reference*, 2020). Using Oracle is not ideal if the user lacks the technical skills and know-how required to use an Oracle database. If an organisation or individual is looking for an easy-to-use database with basic functionality, oracle is not the first choice.

When to Consider using oracle instead of SQLite

Due to the multi-level security and high compatibility of Oracle DB, the advantages of handling large amounts of data and a full suite of data solutions, Oracle is better suited to large businesses. On the one hand, these companies have very large amounts of data, they are concerned about data security and efficient storage. Additionally, they want to be able to resolve problems quickly if they arise, and Oracle offers a range of services that meet their needs. On the other hand, these companies are well financed and can afford the high cost of Oracle.

Summary

In summary, Oracle offers a better ability of processing large-scale data, data security and platform compatibility than SQLite, however, its high price and complex operation hinder its popularity among small and medium-sized companies.

Conclusion

Overview

Our report has introduced the reader to the SQLite relational database management system and the ways in which SQLite is both inferior and superior to Postgres, HiveSQL, MySQL, and Oracle Database. The overarching theme which underpins all sections in this report report is the notion that it it not possible to establish an objective hierarchy of databases because the superiorty of databases is a fundamentally subjective concept. This subjectivity highlights the difficulty which is central to the process of selecting a database system for a project or information system. Selecting the optimal database therefore requires a thorough understanding of the requirements of the project or information system which the database will be implemented in. As such, the optimal database is the one which aligns most closely with the context and specific requirements on the project or information system which it will be implemented in. We hope that this report has provided the reader with the understanding, principles, and mental models needed to compare and assess the strengths and weaknesses of SQLite, Postgres, HiveSQL, MySQL, and Oracle Database in the context of selecting an optimal database technology for a project or information system. The following section will provide a brief summary which the reader may choose to reflect on throughout their decision-making process.

Research Summary: Comparing Databases At A Glance

Database Technology	Relative Advantages When Compared to SQLite	Relative Disadvantages When Compared to SQLite
SQLite	Small, portable, and easy to setup and maintain.	Limited datatypes, low relative compliance with SQL standards, lacking authentication features, and does not support user or role management.
Postgres	Wide range of datatypes, strong compliance with SQL standards, authentication features, and strong user and role management.	Difficult to set up and maintain in addition to issues related to ease of portability.
HiveSQL	Strong compliance with SQL standards, low learning curve, and highly capable of processing large amounts of data.	Not optimized for small datasets, does not support some SQL commands such as UPDATE, high latency issues, and is difficult to set up.

MySQL	Open source, low total cost of ownership, tightly integrated with existing development workflows through the LAMP stack.	Lacking SQL features such as FULL JOINs, some functions are restricted to proprietary licensed versions of MySQL, and the development speed of MySQL has slowed in recent years.
Oracle Database	Transaction grouping gives rise to greater efficiency, Oracle databases can be managed simulataneously by multiple servers, runs on all major operating systems, strong security through through Authentication-Authorization-Audit model, and strong data backup systems.	Oracle database is expensive, highly complex, and has a steep learning curve.

Bibliography

Alpati, S., 2005. *Expert Oracle Database 10g Administration.* 1st Edition ed. New York: Apress.

apache, 2020. *Home.* [Online]
Available at: https://cwiki.apache.org/confluence/display/Hive/Home
[Accessed 17 November 2020].

apache, 2020. *LanguageManual+Types.* [Online]
Available at: https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types
[Accessed 17 November 2020].

Copes, F., 2020. *SQLite User Permissions.* [Online]
Available at: https://flaviocopes.com/sqlite-user-permissions/
[Accessed 17 November 2020].

CSDN.MR YANG, 2017. *The difference between HQL and SQL.* [Online]
Available at: https://blog.csdn.net/qq_28633249/article/details/77884062
[Accessed 23 November 2020].

CSRC, 2020. *authentication.* [Online]
Available at: https://csrc.nist.gov/glossary/term/authentication
[Accessed 17 November 2020].

Dawodi, M., Hadi, M. & Baktash, J., 2019. *Facebook MySQL Performance vs MySQL Performance.* Vancouver, BC, International Conference and Workshop on Computing and Communication (IEMCON).

Denton, J. W. & Peace, A. G., 2003. Selection and Use of MySQL in a Database Management Course. *Journal of Information Systems Education,* 14(4), pp. 401-408.

Drake, M., 2019. *SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems.* [Online]
Available at: https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems
[Accessed 17 November 2020].

explore group, 2019. *The Most Popular Databases 2019.* [Online]
Available at: https://www.explore-group.com/blog/the-most-popular-databases-2019/bp46/
[Accessed 17 November 2020].

Fujitsu, 2020. *What is PostgreSQL?.* [Online]
Available at: https://www.postgresql.fastware.com/what-is-postgresql
[Accessed 16 November 2020].

Hongyu, 2019. *Different between SQLite and MySQL.* [Online]
Available at: https://www.cnblogs.com/cuihongyu3503319/p/10942966.html
[Accessed 23 November 2020].

JavaTPoint, 2018. *SQLite History.* [Online]
Available at: https://www.javatpoint.com/sqlite-history
[Accessed 17 November 2020].

Junyan, L., Shiguo, X. & Yijie, L., 2009. *Application Research of Embedded Database SQLite.* Chengdu, China, 2009 International Forum on Information Technology and Applications, IFITA.

Kala, 2020. *MySQL and Postgres and SQLite advantage disadvantage.* [Online]
Available at: https://kalasearch.cn/blog/sqlite-mysql-postgres-comparison
[Accessed 23 November 2020].

Leifer, C., 2018. *SQLite Database Authorization and Access Control with Python.* [Online]
Available at: https://charlesleifer.com/blog/sqlite-database-authorization-and-access-control-with-python/
[Accessed 17 November 2020].

Oracle, 2020. *Database Concepts.* [Online]
Available at: https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/
[Accessed 23 November 2020].

PostgreSQL, 2020. *19.3. Authentication Methods.* [Online]
Available at: https://www.postgresql.org/docs/9.1/auth-methods.html
[Accessed 17 November 2020].

PostgreSQL, 2020. *A Brief History of PostgreSQL.* [Online]
Available at: https://www.postgresql.org/docs/8.4/history.html
[Accessed 16 November 2020].

PostgreSQL, 2020. *About.* [Online]
Available at: https://www.postgresql.org/about/
[Accessed 16 November 2020].

PostgreSQL, 2020. *Appendix D. SQL Conformance.* [Online]
Available at: https://www.postgresql.org/docs/current/features.html
[Accessed 17 November 2020].

PostgreSQL, 2020. *Chapter 8. Data Types.* [Online]
Available at: https://www.postgresql.org/docs/9.5/datatype.html
[Accessed 16 November 2020].

Raja, Y., 2019. *Managing PostgreSQL users and roles.* [Online]
Available at: https://aws.amazon.com/blogs/database/managing-postgresql-users-and-roles/
[Accessed 17 November 2020].

Reference*, 2020. *What Are Some Advantages and Disadvantages of Oracle Database?.* [Online]
Available at: https://www.reference.com/world-view/advantages-disadvantages-oracle-database-dbec0904f3b40425
[Accessed 23 November 2020].

SQLite, 2020. *Appropriate Uses For SQLite.* [Online]
Available at: https://www.sqlite.org/whentouse.html
[Accessed 17 November 2020].

SQLite, 2020. *Datatypes In SQLite Version 3.* [Online]
Available at: https://www.sqlite.org/datatype3.html
[Accessed 16 November 2020].

SQLite, 2020. *History Of SQLite Releases.* [Online]
Available at: https://www.sqlite.org/chronology.html
[Accessed 17 November 2020].

SQLite, 2020. *SQL Features That SQLite Does Not Implement.* [Online]
Available at: https://www.sqlite.org/omitted.html
[Accessed 17 November 2020].

SQLite, 2020. *Well-Known Users of SQLite.* [Online]
Available at: https://www.sqlite.org/famous.html
[Accessed 27 November 2020].

Sullivan, D., 2019. *Advantages & Disadvantages of Oracle SQL.* [Online]
Available at: https://www.techwalla.com/articles/advantages-disadvantages-of-oracle-sql
[Accessed 23 November 2020].

TablePlus, 2018. *SQLite vs PostgreSQL - Which database to use and why?.* [Online]
Available at: https://tableplus.com/blog/2018/08/sqlite-vs-postgresql-which-database-to-use-and-why.html#:~:text=SQLite%20supports%20only%20five%20types,that%20you%20can%20think%20of.
[Accessed 16 November 2020].

tutorialspoint, 2020. *Hive - Introduction.* [Online]
Available at: https://www.tutorialspoint.com/hive/hive_introduction.htm
[Accessed 22 November 2020].

Venner, J., 2009. In: *Pro Hadoop.* NEWYORK: Apress.

XIAOYUZHOU, 2018. *How to do HIVE optimization.* [Online]
Available at: https://blog.csdn.net/yu0_zhang0/article/details/81776459
[Accessed 17 November 2020].

Zhuque, 2019. *MySQL's advantage.* [Online]
Available at: http://www.codingbefore.com/article/aid/1576718782332
[Accessed 23 November 2020].

Individual Report

A3 Group I
Matriculation number: 200009834

Sub-topic researched

The part of my investigation is the differences and similarities between MySQL and SQLite. I divided the investigation into nine part based on the article framework we set.

1. The definition of MySQL.
2. The types of data that can be supported.
3. The characteristics of MySQL.
4. The context of MySQL
5. MySQL's advantages.
6. MySQL's disadvantages.
7. When to use MySQL instead of SQLite.
8. When to use SQLite instead of MySQL.
9. Summary.

How I went about carrying out this research

In our daily work, the types and sources of information we face are diverse, and the basic search engine data is already very rich, but how to find the information for ourselves requires pre-thinking, finishing in the process, and finally the brain Processing integration.

First of all, from the overall classification of data, it can be divided into: books, reports, periodicals, magazines, official websites, etc.

Secondly, I thought about the purpose of accessing information and what kind of information needs to be inquired: all data collection is for the purpose of solving problems. Therefore, before collecting information, I first clarified the purpose and requirements of the collected information. In my opinion, the more thorough the purpose and requirements are, the more accurate the collected information.

At the same time, in the search process, I pay attention to the principle of information access:

1. Accuracy: The information collected must be accurate, true and reliable. Of course, this principle is the most basic requirement for information collection.
2. Comprehensiveness: Only by collecting information extensively and comprehensively can the operating status of the target company be fully reflected.
3. Timeliness: Information can only be effective when it is provided to its users in a timely and rapid manner.
4. Ease of use: The collected information has an appropriate representation and is easy to use.

The resources that were used

The most important ones are Google Search, Google Scholar and IEEE. The official website of SQLite and MySQL are provided for navigation.

I wasted a long time in this part because I did not find the target problem awareness. In this part, I fully realize that I need to understand the origin of scientific research to find the problem (that is, what, why, how). Although it is possible to clarify things, in order to clarify ideas, problems are needed and a lot of time will be wasted.

Reflective of group working

I have the following feelings about this group cooperation:

1. Clear division of labor: each team member knows the overall goals and personal tasks.

2. Clear timeline: clearly defined tasks to be completed at each stage.

3. Timely communication: team members understand each other's task progress.

4. Effective communication: promptly raise and solve problems existing in the project.

I analyzed the reasons. It may be that all members of our team understand why we are doing this project, what we need to contribute to this project, and we all have a clear understanding of what problems the project has and how to improve it.

I have had a lot of bad team work experience before, someone could:

1. Obey on the surface and obstruct work in secret by delaying, perfunctory, and non-cooperative.
2. Treat cold and not express emotions.
3. Use satire well, but usually not just satire.
4. Don't give praise, pick others.
5. Frequent late and "forget" tasks.
6. They often break their promises about promises that can be easily fulfilled.

Therefore, most of the time I will try to avoid group cooperation, because the above behavior is destructive to the entire group. It will destroy the cohesion and productivity of the group. The work efficiency of a team will be reduced because of such people. Because a person’s sense of injustice can also cause each member to complain to each other, the final task is not completed very well.

However, I feel that this group cooperation was very successful. In the process of this group cooperation, everyone showed active handling of problems, team spirit, flexible mastery of the professional knowledge learned, and very serious responsibility for the work assigned by the group. Everyone has a strong learning ability, is good at cooperating with others, doing things right from the beginning to the end, having a strong sense of teamwork and sense of responsibility, and adhering to the principle of seeking things from facts.

This group work has brought me a lot. I can learn many qualities that I lack from the other three group members. It not only made me pay attention to the integration of various resources, but also how to handle the division of labor of group members. It also made me realize that hands-on practice, independent exploration, and cooperation and communication are one of the important ways of learning.

But in this group cooperation, I also have many shortcomings:

1. Determination of research goals. At the beginning, I lacked precise search for the selected target information, wasting a lot of time.

2. Improve writing skills. When writing the paper, many details were not fully noticed. This often leads to problems with the fluency of sentences, or sometimes problems with grammar. I am very grateful to the team members for helping me correct this mistake. I learned a lot from it.

This assignment allows me to consolidate the knowledge I have learned before, effectively apply the learning knowledge to specific cases, deepen my understanding, and at the same time, it also allows me to learn what cooperation is, and to listen to the opinions of others in the discussion. While conscientiously completing your own tasks, you must actively cooperate and assist others to take care of the overall situation.