Creating the Database Environment

Mullins chapter 2

Bjarte Wang-Kileng

HVL

January 23, 2025

Western Norway University of Applied Sciences

Outline

Outline

Choosing a DBMS

- Minimise the number of DBMSs.
- Choose one DBMS, or one DBMS for each platform/OS.
- Choose a vendor with large market share:
  - Missing functionality compared to others might soon be implemented.
- If vendor has small market share:
  - Fewer developers.
  - Long time before errors are corrected.
  - Can have more undiscovered errors.
  - What about the future of the DBMS?

The three large

Market share by popularity

- Oracle:
  - Linux, Solaris, Windows.
- IBM DB2:
  - Linux, UNIX, Windows, z/OS, iSeries.
- Microsoft SQL server:
  - Windows, Linux.

Relational Open Source DBMSs (OSRDBMSs)

- MySQL – Owned by Oracle.
- MariaDB – Community-developed fork of MySQL.
- PostgreSQL.
- BerkeleyDB – Owned by Oracle.
- PostgreSQL and MySQL compared:
  - MariaDB vs. MySQL vs. PostgreSQL Comparison.

MySQL and MariaDB

- Fast.
- Many engines for different needs (MySQL, MariaDB, MySQL pluggable Storage Engines (SE) | FromDual).
- MySQL is owned by Oracle.
- For all practical purposes, MariaDB is a binary drop in replacement of the same MySQL version (ref).
  - But, implementation differences continue to grow with each new MariaDB version.
- High market share (ref).
  - January 2025, MySQL is number two, Oracle.

MySQL and MariaDB

We will use MariaDB in the lab.

Factors when choosing a DBMS

- OS support.
- Organisation:
  - Government, finanical, healt etc. are more conservative than universities, dot-coms and web companies.
  - Conservative – Oracle, DB2.
  - Liberal – MariaDB, MySQL, PostgreSQL, MongoDB.
- Benchmark tests.
  - Might not be representative for production database system.
  - TPC Benchmarks
- Scalability.

More factors when choosing a DBMS

- Available software tools:
  - Query tools.
  - Analysis tools.
  - Administration tools.
  - Backup and recovery tools.
  - Performance monitoring tools.
  - Connectors
- Knowledge available in the organisation.
- Total cost:
  - License for DBMS.
  - License for supporting tools.
  - Cost of programming.
  - Support and administration.
  - Cost of hardware.

And more factors when choosing a DBMS

- Release schedule:
  - Cutting edge features – Fast cycle is good.
  - Conservative organisation:
    - Slow cycle is good.
    - Frequent changes can be difficult to support.
    - Frequent changes can cause the organisation to use an outdated DBMS-version.
- Experiences of others?
  - How is support? Do they respond well?
  - Are there any problems?
  - Quality of new releases?
  - Many bug fixes?

Outline

Levels of DBMS Architecture

- Enterprise DBMS.
- Departmental DBMS.
- Personal DBMS.
- Mobile DBMS.

Enterprise DBMS

- High scalability.
- High performance.
- Support very large databases.
- Large numbers of concurrent users.
- Multiple types of applications.
- Mainframe or high-end server.
- Including all “bells and whistles” available from the DBMS vendor.
- Multiprocessor support and support for parallel queries.

Departmental DBMS

- Dedicated servers.
- Used by work group (small to medium sized) within organisation.

Personal DBMS[1]

- Single user.
- Low- to medium-powered PC.
- Microsoft Access, SQLite, FileMaker.
- Personal versions of other DBMSs.
- Only for small-scale projects.

[1] Not focus for this course.

Mobile DBMS[2]

- Specialized version of departmental or enterprise DBMS.
- For remote users witout permanent network connection.
- Database access on laptops and handheld devices.
- Requires later synchronisation to DBMS in organisation.

[2] Not focus for this course.

DBMS Clusters

- DBMS distributed through multiple computing systems:
  - Working toghether as a single, high available system.
- Two predominant architectures, *shared-nothing* and *shared-disk*.
- MySQL – DBMS cluster.
  - The MySQL NDB cluster is not supported by MariaDB, see e.g. NDB in MariaDB.
- MySQL and MariaDB – Galera cluster.
  - DBMS cluster through replication.
  - See also What is MariaDB Galera Cluster.

Shared-nothing

- Each computer has its own resources.
- Communication by passing messages through the network.
- Requests are routed to the computer that owns the resource.
- Scales well.

Shared-disk

- All computers share the same disk devices.
- Does not scale as well as shared nothing.
- For applications with only modest shared access to data.
- System with heavy data update is better implemented using *shared noting*.

Cloud database system

- Cost effective.
- Minimize database administration, mantenance cost and effort.
- Improve collaboration among partners, remote workers and mobile devices.
- But – can you trust the provider to store and manage your data?
- Examples of systems:
  - SQL – E.g. Amazon RDS (MySQL, Oracle, MS SQL and PostgreSQL), Microsoft SQL Azure (MS SQL), Heroku (PostgreSQL).
  - NoSQL – E.g. Amazon DynamoDB, Amazon SimpleDB, Google App Engine Datastore, Heroku (Cloudant, Couchbase Server, MongoDB and Redis), Google Firebase Realtime Database.

Outline

Understand the prerequisites

- OS.
- Libraries and related software.
- CPU version, speed, numbers.
- Storage (disk, tape).
- Memory (RAM).

Storage requirements

- The (primary) databases.
- System databases (administration, monitoring, tuning, testing, etc.).
- System catalogue (meta-data, ANSI – *Data Dictionary*, chapter 10).
- Indexes.
- Log files:
  - All changes to databases, problems, slow queries, errors.
- Configuration files.
- DBMS-specific work files.
- Temporary databases.
- System dump and error files.
- Grant-tables.
- Backup.

Use multiple storage devices

DBMS files

Use multiple storage devices as the DBMS works with many files concurrently.

Memory requirements

- DBMSs love memory.
- I/O is slow – much memory is used for data caches, e.g.:
  - Buffer pool – stores accessed data.
  - Program cache – stores "compiled" SQL.
- Sorted data.
- Locks.

Means to configure the DBMS

- GUI.
- Configuration files.
- OS commands and DBMS program switches.
- DBMS commands.

Configuring MySQL and MariaDB

- GUI – E.g. MySQL workbench.
- Command-line switches when starting the server.

```
/usr/bin/mysqld_safe --default-storage-engine=MyISAM
```

- Configuration files – "/etc/my.cnf", "/etc/my.cnf.d/", "~/.my.cnf".

```
[mysqld_safe]
default-storage-engine=MyISAM
```

- Options given to a running server with the «SET» command.

```
MariaDB [(none)]> SET storage_engine=MyISAM;
```

Testing the installation

Verification

After installation, run tests to verify that the DBMS has been properly installed and configured.

Outline

Upgrading DBMS Versions and Releases

- New revision are frequent:
  - MySQL 8.4 revisions.
- New versions are much less frequent, e.g.:
  - July 1999 – MySQL 3.23.
  - December 2001 – MySQL 4.0.
  - December 2003 – MySQL 4.1.
  - July 2004 – MySQL 5.0.

Upgrading – Pros

- New functionality.
- Might be required by 3'd party applications.
- Better performance.
- Better and faster support for a new release.
- Same version in test and production environments.

Upgrading – Cons

- Disruption to business operation.
- Might have to convert database structures.
- Previously supported features are removed.
- Cost (DBMS, planning, testing, deploying).
- Performance can suffer:
  - Performance tuning to existing DBMS might not work as well for the new version.
  - Changes in access paths to data.
  - Missing support from 3'd party applications.

Factors determining whether upgrading a DBMS

- Functionality.
- Complexity of the existing DBMS environment.
- Reputation of vendor.
- Support policy.
- Organisation style.
- DBA skills.
- Platform support.

Running a new version of the DBMS

Fallback

Do too bugs, fallback to an earlier version of the DBMS can be necessary.

Review the fallback procedures provided by the DBMS.

Verification

Implement procedures to verify that the DBMS release upgrade is satisfactory.

Outline

Standards and Procedures

- Standards:
  - Common practises.
  - Ensures consistency and effectiveness of the database environment.
- Procedures:
  - Defined step-by-step instructions for specific events, e.g. disaster recovery and backup.

Standards and Procedures

High level of standardisation reduces cost.

Examples of Database Standards

- Naming conventions.
- Roles and Responsibilities.
- Data administration standard.
- Database administration standards.
- System Administration Standards.
- Database Application Development Standards.
- Database Security Standards.
- Application Migration Standards.