Creating the Database Environment

Mullins chapter 2

Bjarte Kileng

HVL

January 10, 2018

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 2010 and 2011, popularity

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

Relational Open Source DBMSs (OSRDBMSs)

- Not very visible in the market share statistics (why?)
- 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.

Open source or commercial DBMS?

- Report from Gartner:

  > *Open source RDBMS have matured and today can be considered as a standard infrastructure choice for most new enterprise applications*

- Which relational DBMS is best for your company? – From the author of our book.

MySQL and MariaDB

- Fast.
- Many engines for different needs (MySQL, MariaDB).
- MySQL is owned by Oracle.
- For all practical purposes, MariaDB is a binary drop in replacement of the same MySQL version (ref).
- High market share (ref).

MySQL and MariaDB

We will be using MariaDB at 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?

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*

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

*Not focus for this course.

Mobile DBMS†

- 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.

†Not focus for this course.

DBMS Clustering

- DBMS distributed through multiple computing systems:
  - Working toghether as a single, high available system.
- MySQL – DBMS cluster.
- Two predominant architectures, *shared-nothing* and *shared-disk*.

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).

Understand the prerequisites

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

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

The DBMS uses many of the files concurrently. Use multiple storage devices.

Memory requirements

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

Means to configure the DBMS

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

Configuring MySQL and MariaDB

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

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

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

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

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

```
SET storage_engine=MyISAM;
```

Testing the installation

Verification

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

Upgrading DBMS Versions and Releases

- New revision are frequent:
  - MySQL 5.7 revisions.
- New versions are much less frequent:
  - July 1999 – MySQL 3.23.
  - December 2001 – MySQL 4.0.
  - December 2003 – MySQL 4.1.
  - July 2004 – MySQL 5.0.
  - November 2005 – MySQL 5.1.
  - December 2010 – MySQL 5.5.
  - April 2011 – MySQL 5.6.
  - April 2013 – MySQL 5.7.

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, we might have to fallback to an earlier version of the DBMS.

Review the fallback procedures provided by the DBMS.

Verification

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

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.