Administering MySQL

SkySQL Ab Training

Percona Live MySQL Conference & Expo

Jens Bollmann
jens@skysql.com

10APR2012

Introduction

Administering MySQL

Welcome

Introduction

This class is for anyone who wants to administer a MySQL database server. We will:

- Install and configure MySQL
- Learn about the available tools and utilities
- Choose and configure storage engines
- Backup, restore and upgrade databases
- Handle crash recovery and fail over
- Implement database replication
- Cover security, monitoring and troubleshooting

Who is SkySQL Ab?

Introduction

SkySQL Ab is the alternative source for software, services and support for MySQL related technologies.

- Founded by key people from MySQL AB
- Funded by OnCorps Capital, David Axmark, Open Ocean, Founders and SkySQL employees
- Backed by Product Engineering Monty Program Ab
- Most of the team are ex-MySQL AB employees, with more joining all the time!
- Operating in 13 countries so far...

Course Structure

Introduction

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
10. Views
11. Partitioning
12. Security
13. Programming
14. Replication
15. Monitoring
16. Troubleshooting
17. Optimization
18. High Availability
19. Conclusion

Installation

Administering MySQL

Course Structure

Installation

1. Introduction
2. **Installation**
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Distributions

Installation

MySQL 5.1	• Oracle backed previous GA release (still common)
MySQL 5.5	• Oracle backed GA release
MariaDB 5.1	• Drop in replacement for MySQL 5.1 • Includes some MySQL 5.5 patches • Aria, XtraDB, PBXT, FederatedX
MariaDB 5.2	• Additional features deemed stable • OQGRAPH, SphinxSE
Percona Server	• Drop in replacement for 5.1 with XtraDB

Upgrades

Installation

SkySQL

Backup!

- Review change logs for both major and minor version upgrades
- Major version upgrade is safest with a data dump & reload
- Minor version upgrade can use the mysql_upgrade tool
- Check: Do your apps and libraries need updating too?
- Check: How much downtime is required?
- Replication can be used to reduce downtime by upgrading the slave first, promoting it, then upgrading the master

Test the entire process somewhere else first!

Client Utilities

Administering MySQL

Course Structure

Client Utilities

1. Introduction
2. Installation
3. **Client Utilities**
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

mysql

Client Utilities

- Command line tool (CLI)
- Requires only a shell session, and works well over SSH
- Manually execute any SQL statement
- Display mysqld settings and status counters
- Use as interactive session, or pipe SQL via the shell

```
shell> mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.1.50 MySQL Community Server (GPL)

mysql> SHOW DATABASES;
```

mysql

Client Utilities

SkySQL

Many mysql cli options exist. Just a few:

--user=username -u username	MySQL username
-- password[=password], -p[password]	MySQL password
--host=hostname -h hostname	Hostname, FQDN, or IP of the MySQL server
--port=portno -P portno	MySQL port number (defaults to 3306)
--execute=statement -e statement	Connect, execute the supplied statement and disconnect

mysqladmin

Client Utilities

- Command line tool (CLI)
- Requires only a shell session, and works well over SSH
- Manage user accounts, passwords and permissions
- Display mysqld settings and status counters
- View and kill active connections
- Ping or shutdown mysqld
- Create and drop databases

```
shell> mysqladmin processlist
shell> mysqladmin -u root -p create employees
shell> mysqladmin extended-status
```

mysqldump

Client Utilities

SkySQL

- Flexible tool to dump schema and/or data to:
  - SQL text file, tab delimited, xml formats
- Useful for backups, or transferring data between servers
- Requires locks for consistency. May affect other traffic
- Dump all databases or specific objects
- Optimize imports with extended inserts, temporarily disabling indexes, and ignoring foreign key checks

```
shell> mysqldump --all-databases > backup.sql
shell> mysqldump --no-data employees > schema.sql
shell> mysqldump -u backup -p -h production \
    | mysql -u devteam -h development
```

Other CLI Tools

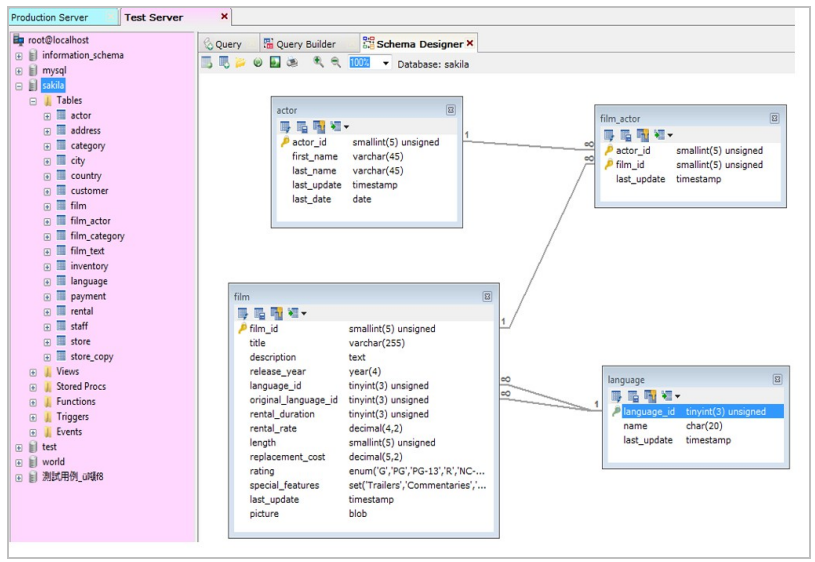
Client Utilities

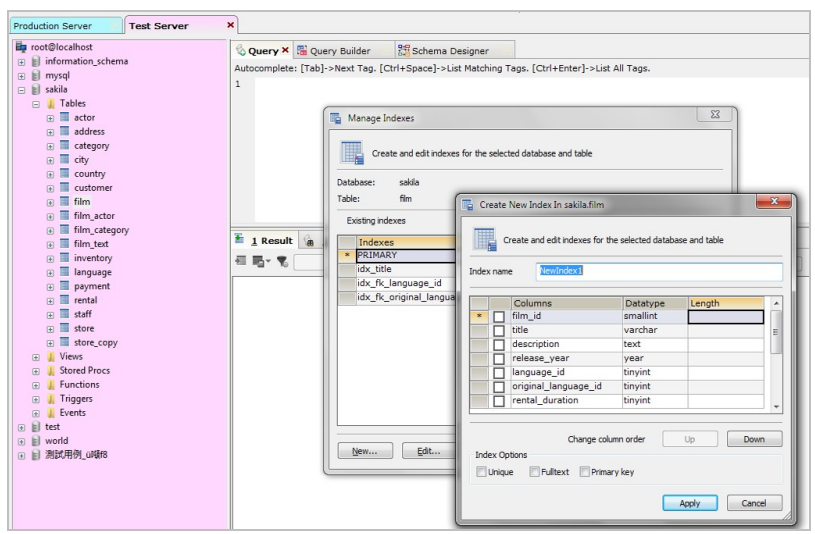
- mysql.server
- mysqld_safe
- mysqld_multi
- mysql_fix_privilege_tables
- mysql_install_db
- mysql_upgrade
- mysql_tz_to_sql
- my_print_defaults
- mysqlslap
- myisampack
- mysqlbinlog
- mysqldumpslow
- mysqlcheck
- myisamchk
- perror
- innochecksum

SkySQL Visual Editor

Client Utilities

- Supported by WebYob Inc.
- GUI application (MS Windows)
- Schema and query construction
- Optimization and profiling
- User account management
- Connect via SSL or SSH tunnel
- Data migration and cleansing
- Task and backup scheduling
- Schema sync and diff tools
- Notifications

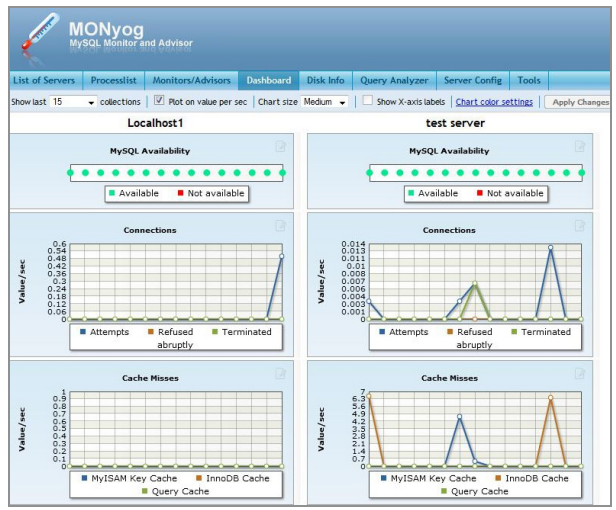


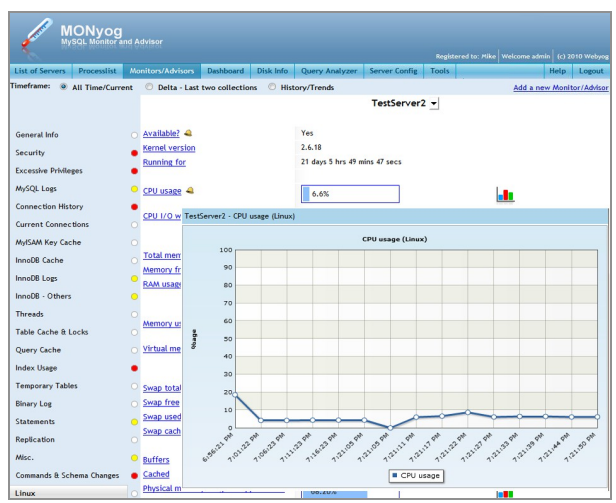


SkySQL Enterprise Monitor

Client Utilities

- A user friendly, cross platform, GUI
- No server-side agent required
- Manages multiple servers in one place
- Track and alert on many metrics:
  - monitor error log / security vulnerabilities
  - replication lag / excessive resource usage
  - slow queries / deadlocks
- Allows programming custom advisors
- Detailed history / trend analysis

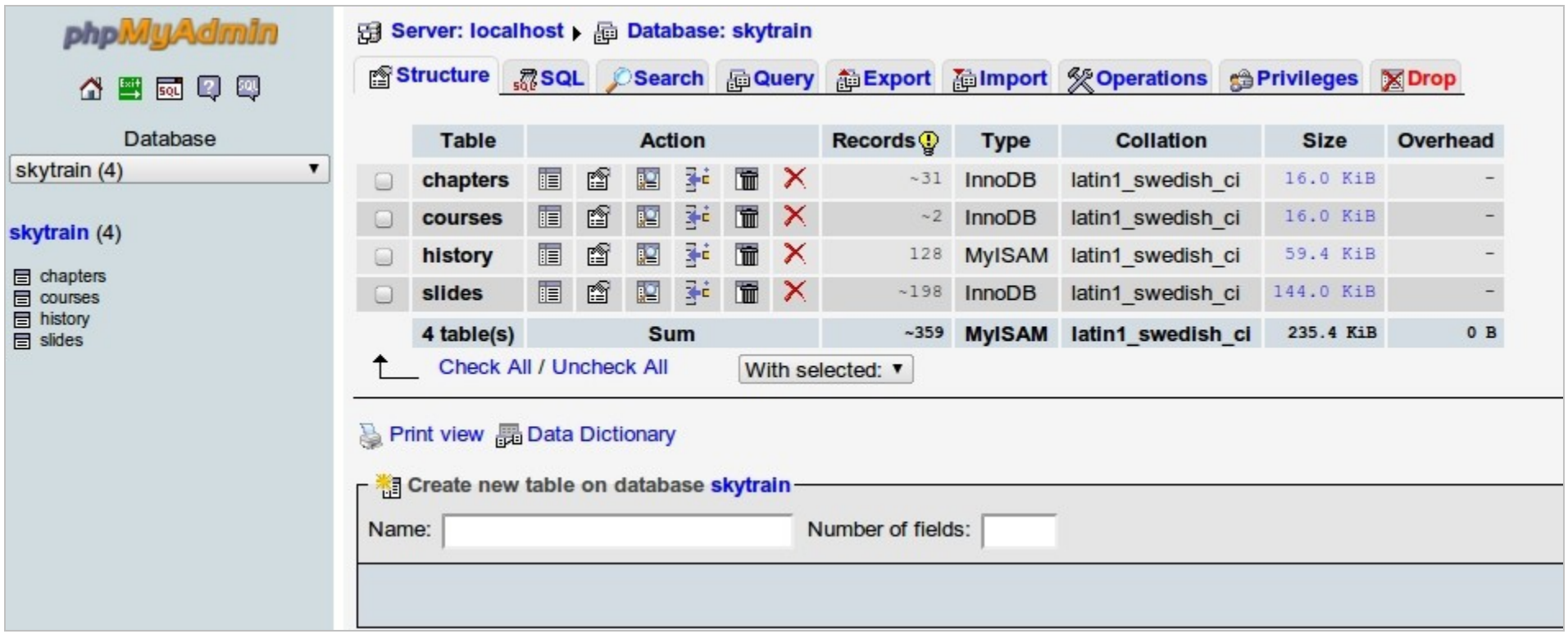




Backed by a SkySQL Ab partner: *Webyog, Inc*

phpMyAdmin

Client Utilities



- Web based. Available or installable on shared hosting
- Easy to use and full featured
- Well established with a large community

Architecture

Administering MySQL

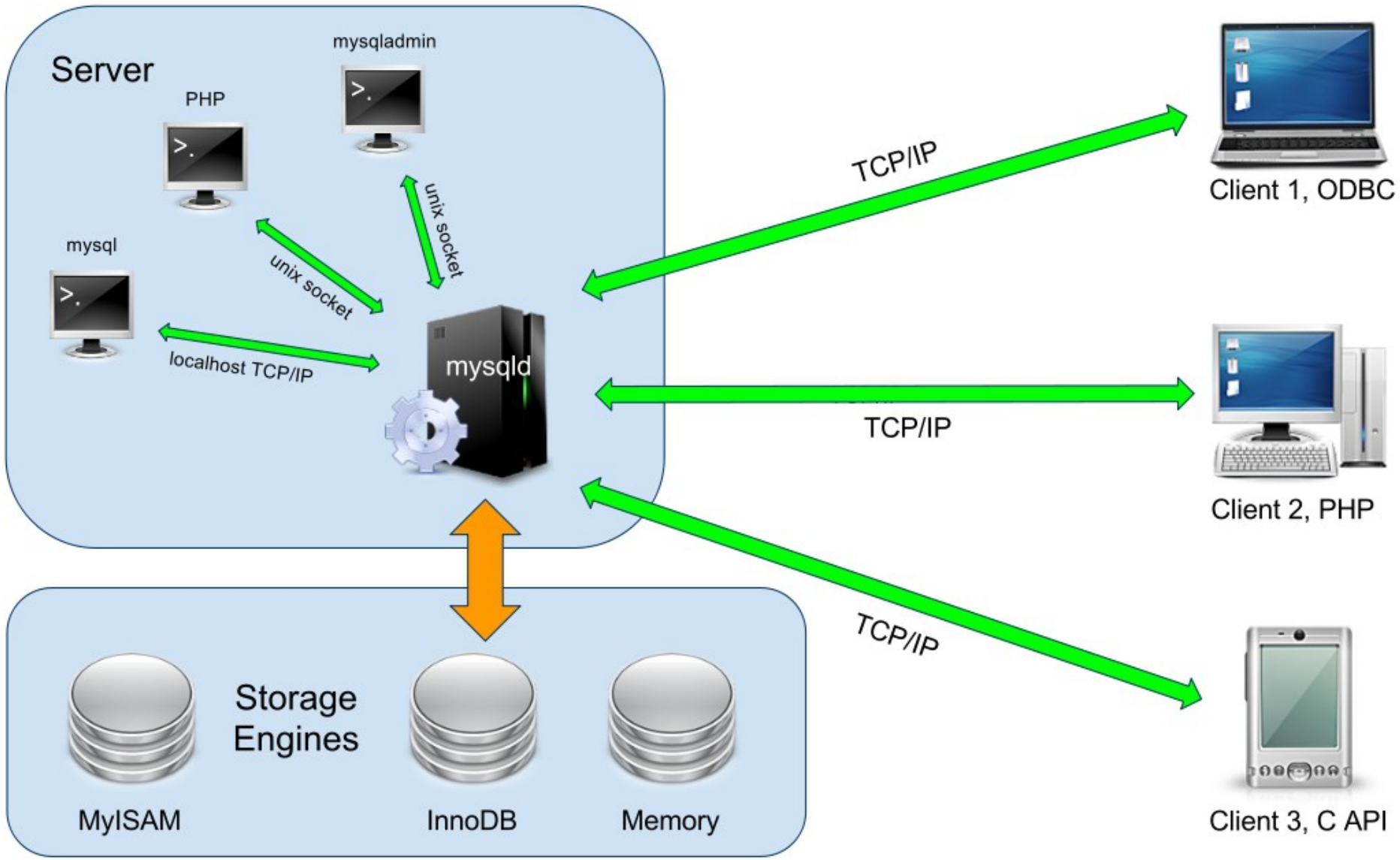
Course Structure

Architecture

1. Introduction
2. Installation
3. Client Utilities
4. **Architecture**
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Client, Server, Storage

Architecture



Client Connections

Architecture

- TCP/IP connections available on all platforms
  - Disable them with --skip-networking
- Socket files available on Unix platforms
  - Fastest communication solution for Unix
- Named pipe and shared memory available on Windows
  - --enable-named-pipe and --shared-memory respectively
- MySQL connections are light weight and cheap to open
  - Many people do use external connection pools, which is fine...
  - ...but in many cases they simply are not needed
- A global client connection limit can (and should) be set with max_connections=N

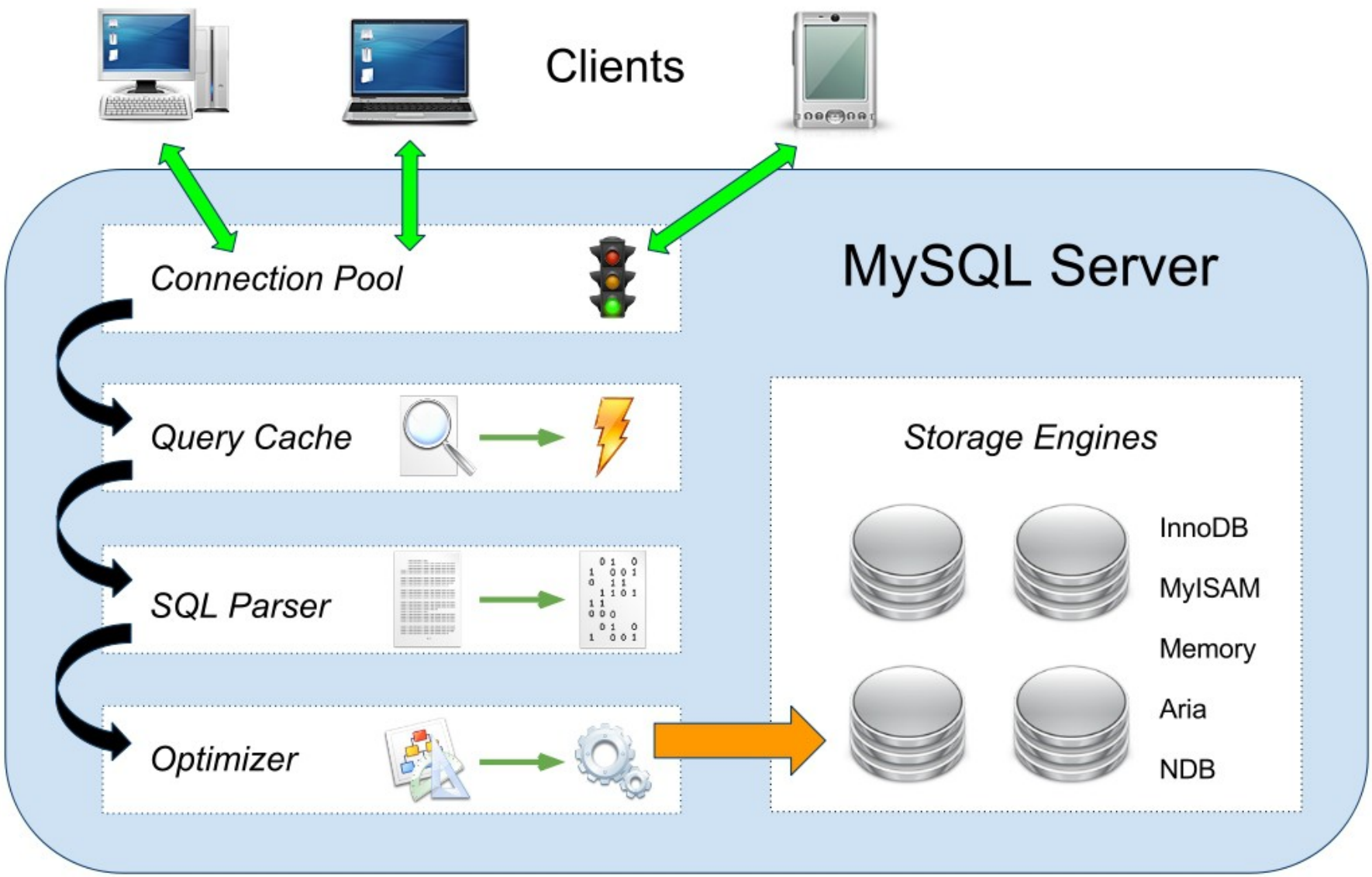
Client Libraries

Architecture

- Various libraries exist to connect to MySQL, depending on the client language, platform and environment
  - PHP, Perl, Python, Ruby, Lua, .NET, Scheme and many more
- Most of them are wrappers for the MySQL C API which speaks the MySQL network protocol
- Connectors for ODBC, NET and JDBC are available
- There is also an embeddable version of MySQL called libmysqld, which many well known apps wrap

mysqld Overview

Architecture



Connection Pool

Architecture

- mysqld's main thread listens for client connections
- Each incoming connection has a thread assigned to handle it, which remains active on the server for the duration
- Depending on configuration:
  - A thread can be reused from the *thread cache*
  - A new thread may be created on the fly
- User authentication is processed based on host IP or name, user account name and password
- Client specific buffers for session variables and network communications, are allocated

Query Cache

Architecture

- Stores SELECT queries and their result sets
- Re-issue the same query, even from a different client, and the cached result is returned
- Useful for high read, low write environments
- Changes to underlying table data cause relevant queries to be removed from the cache
- Can be controlled on a per query basis:

```
SELECT SQL_CACHE * FROM ...
SELECT SQL_NO_CACHE * FROM ...
```

SQL Parser

Architecture

- Converts text SQL statements to an internal binary format
- The Parser includes:
  - a *Lexical Scanner*
  - a *Grammar Rules Module*
- The basic steps are:
  1. Parse SQL into tokens, such as keyword, table, field, value, etc
  2. Apply grammar rules to check the statement is valid
  3. Construct a structure called a *parse tree* to be passed on to the Optimizer
- MySQL's parser is custom C code, built by hand for the best possible performance

Optimizer

Architecture

The Optimizer reads the *parse tree* and calculates the best *execution plan* to handle the query

- Find any appropriate indexes that may be used
- Balance cost of index access against a table scan
- Determine the table JOIN order
- Eliminate unnecessary tables or WHERE clauses
- Find indexes that can handle GROUP and ORDER BY
- ...and a whole lot more arcane stuff!

Configuration

Administering MySQL

Course Structure

Configuration

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. **Configuration**
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Config File Locations

Configuration

SkySQL

Unix

- /etc/my.cnf by default, but it is distro specific
- $MYSQL_HOME/my.cnf is checked too
- ~/.my.cnf is loaded by client apps

Windows

- WINDIR\my.ini, ie, C:\WINDOWS
- C:\my.ini
- INSTALLDIR\my.ini

Security Note! Who has read access to your config files?

Global Variables

Configuration

Connect to MySQL with the mysql client, and run:

```
msyql> SHOW GLOBAL VARIABLES;
```

GLOBAL variables are system wide settings...

They are drawn from the configuration file at startup. Some can be changed dynamically by a SUPER user:

```
mysql> SET GLOBAL
    -> tmp_table_size = 32*1024*1024;
```

Basic Variables

Configuration

Set a custom socket, port, and mysqld process id file:

```
socket    = /tmp/mysqld.sock
port      = 5150 # default is 3306
pid-file = /var/run/mysqld.pid
```

Manual installations need to know the installation directory, so they can resolve various relative file paths:

```
basedir   = /usr/mysql
```

Set the location for schema, data and various logs:

```
datadir   = /var/lib/mysql
```

Basic Variables 2

Configuration

SkySQL

Temporary disk space is required.

If the system temp is a tmpfs (or any RAM based) mount, a separate on disk location is needed for replication slaves:

```
tmpdir = /tmp
slave_load_tmpdir /tmp_not_tmpfs
```

Think about the default character set for data.

This can still be changed for certain databases and tables, or by specific client connections:

```
default-character-set = utf8
default-collation = utf8_general_ci
```

Basic Variables 3

Configuration

Consider the storage engine requirements. The default engine is MyISAM, but for ACID, it should be InnoDB:

```
default_storage_engine = InnoDB
```

Protect the server from overload by limiting client connections, and removing stale connections:

```
max_connect_errors = 10
max_connections    = 100
wait_timeout       = 3600
```

Basic Variables 4

Configuration

Set global caches to best suit the expected application traffic:

```
thread_cache_size = 100
table_cache       = 100
query_cache_limit = 1M
query_cache_size  = 32M
```

Error Log

Configuration

SkySQL

- Contains Startup / Shutdown / Error log messages. It is turned on by default
  - stderr is used on Unix, redirected to HOST.err in datadir by default
  - Some Unix packages redirect it to the system log instead
  - HOST.err in datadir is used on Windows, or the system event log
- Customize as follows:

```
log-error    = /your/preferred/path
log-warnings = 1 # or skip-log-warnings
```

- log-warnings is a numeric level
  - =2 increases verbosity (gives more frequent and varied warnings)
  - particularly useful for client or replication slave connection glitches

General Query Log

Configuration

- Contains all queries issued by all database clients
- Queries are logged in the order they are received and *not necessarily in order of execution*
- This log is turned off by default
  - Can be a disk I/O bottleneck on high traffic servers
  - Obviously, it could grow in size very quickly!
  - Contains raw queries, which could be a security issue
  - Default location is the datadir
- Activate as follows:

```
[mysqld]
log = /your/preferred/path
```

Slow Query Log

Configuration

SkySQL

- Contains queries that took longer than long_query_time seconds to execute. Turned off by default
  - Contains raw queries, which could be a security issue
  - Useful to activate during an application's early days, to monitor slow queries in real world traffic
  - Default location is in the datadir
- Activate as follows:

```
[mysqld]
log_slow_queries = /your/preferred/path
long_query_time  = 2
log-queries-not-using-indexes
log_slow_admin_statements
```

Binary Log

Configuration

- A precise and consistent log of write queries in the order they were executed
- Can be replayed later against an old database snapshot, to bring it back up to date
- Useful for recovery purposes, or replicating data to a Slave MySQL server for load balancing or backup
  - Replication is covered later in the course...
- Default location is in the datadir
  - Move to separate disk for security purposes
- Potential to grow very large, so they should be expired or rotated periodically

Storage Engines

Administering MySQL

Course Structure

Storage Engines

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. **Storage Engines**
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

There are many!

Storage Engines

Many different storage engines are available:

```
mysql> show engines;
+------------+----------+----------------------------------------------------------------+
| Engine     | Support  | Comment                                                        |
+------------+----------+----------------------------------------------------------------+
| MyISAM     | DEFAULT  | Default engine as of MySQL 3.23 with great performance         |
| MEMORY     | YES      | Hash based, stored in memory, useful for temporary tables      |
| InnoDB     | YES      | Supports transactions, row-level locking, and foreign keys     |
| BerkeleyDB | NO       | Supports transactions and page-level locking                   |
| BLACKHOLE  | YES      | /dev/null storage engine (anything you write disappears)       |
| EXAMPLE    | NO       | Example storage engine                                         |
| ARCHIVE    | YES      | Archive storage engine                                         |
| CSV        | YES      | CSV storage engine                                             |
| ndbcluster | DISABLED | Clustered, fault-tolerant, memory-based tables                 |
| FEDERATED  | YES      | Federated MySQL storage engine                                 |
| MRG_MYISAM | YES      | Collection of identical MyISAM tables                          |
| ISAM       | NO       | Obsolete storage engine                                        |
+------------+----------+----------------------------------------------------------------+
```

Engines can be mixed and matched on the same server, and within the same query too.

Overview

Storage Engines

- Storage engines are specified per table:

```
CREATE TABLE ... ENGINE = <name>;
ALTER TABLE ... ENGINE = <name>;
```

- The default storage is MyISAM in 5.0 and 5.1
  - --default-storage-engine
- Engines can control:
  - Storage medium (disk, memory, cluster, other?)
  - **A**tomicity **C**onsistency **I**solation **D**urability
  - Locking level (table, page, row, other?)
  - Special features (FULLTEXT, GIS, Clustered)
  - Some optimization

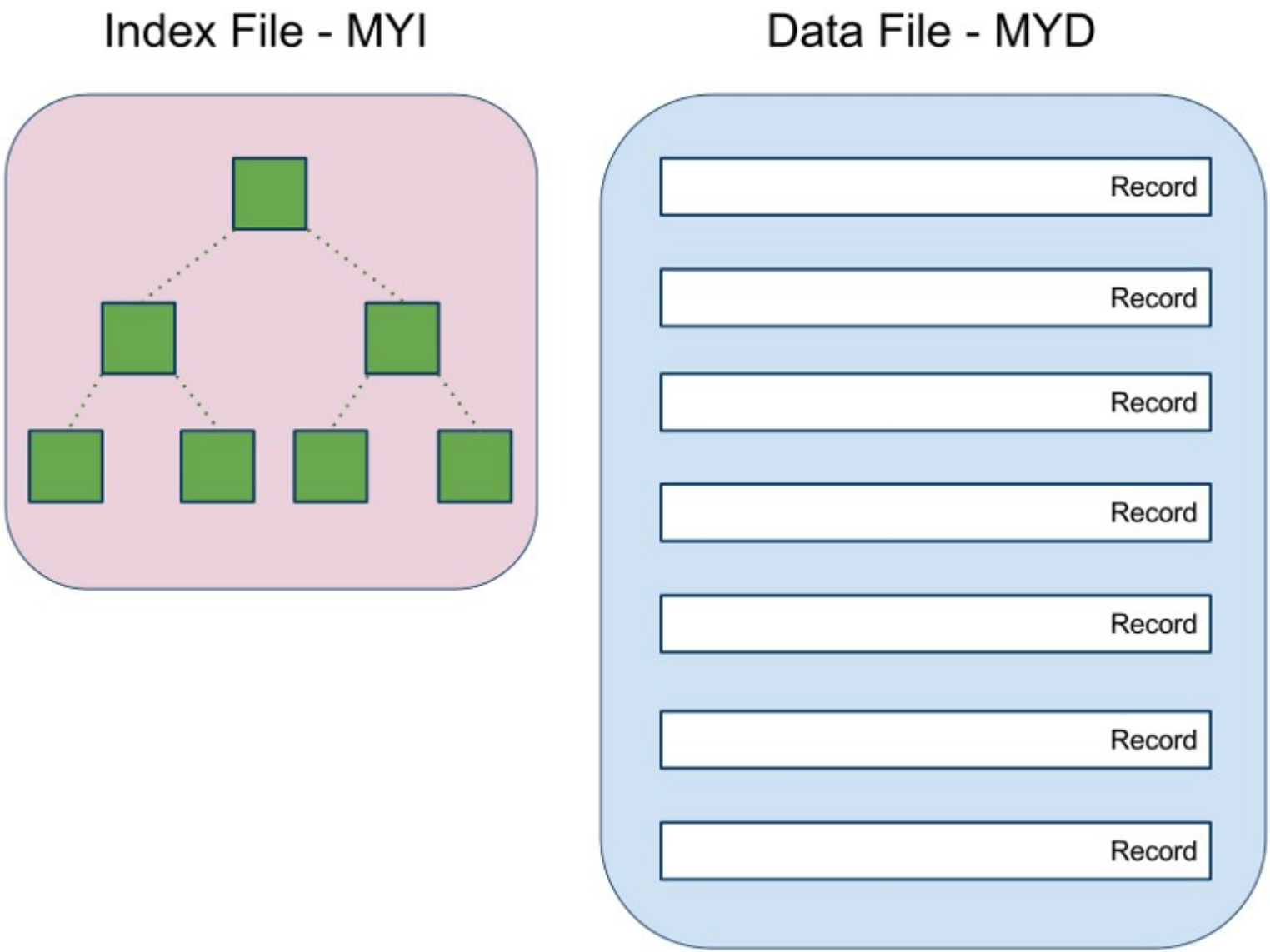
MyISAM

Storage Engines

- Non-transactional
- Suits high read / low write traffic loads
- Uses table level locking (one writer **OR** many readers)
- Can be configured to allow concurrent inserts:
  - SELECT can use a special read lock that lets INSERT to write to the end of the data files concurrently
- Poor crash recovery (relies on OS to flush data to disk)
- Indexes are cached by mysqld in key_buffer
- Data caching relies on the OS disk cache
- FULLTEXT search and GIS spacial data types

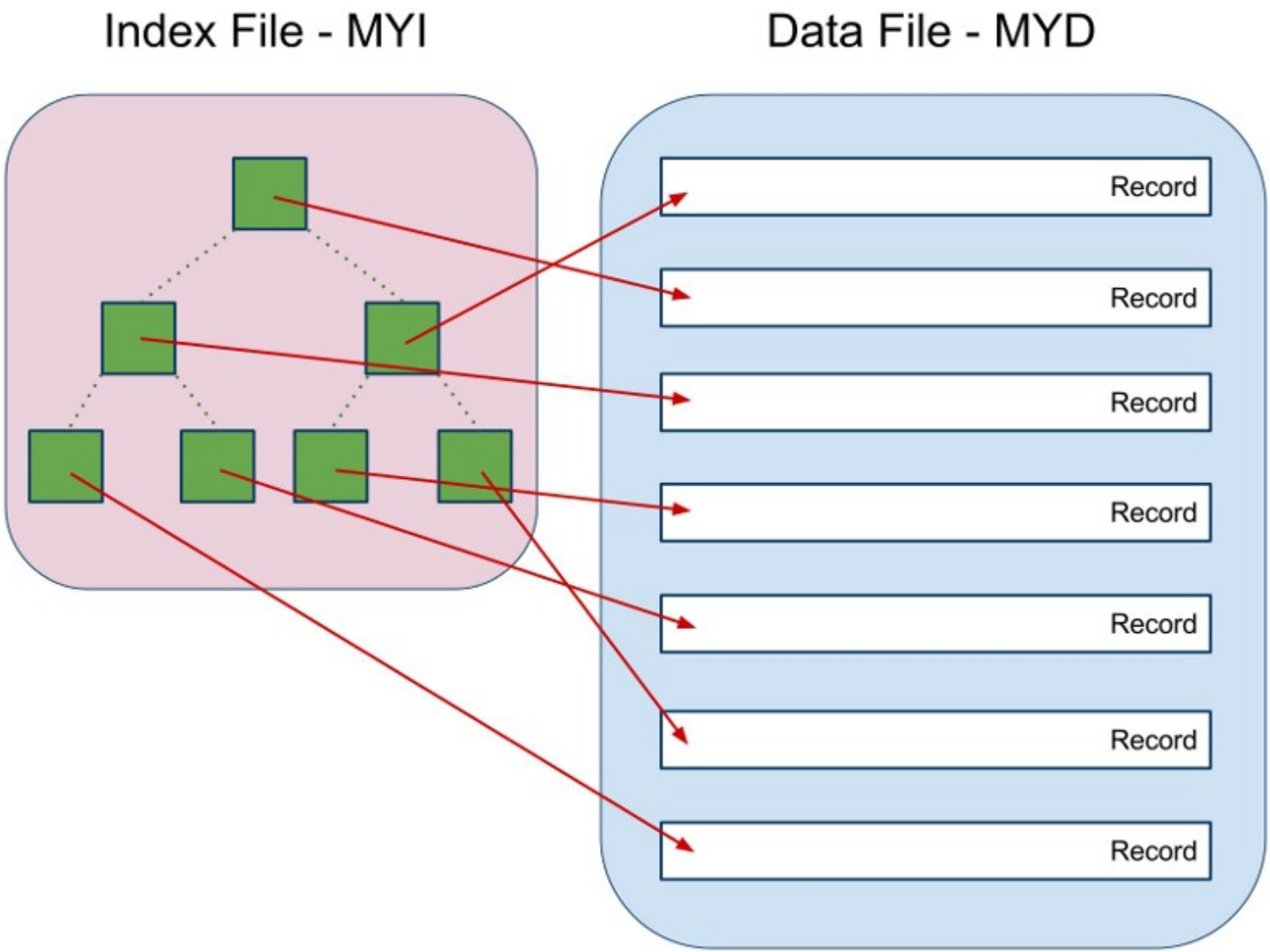
MyISAM Architecture

Storage Engines



MyISAM Architecture

Storage Engines



MyISAM Row Formats

Storage Engines

- Static
  - Fixed length data types (INT, CHAR, etc)
  - Less data file fragmentation
  - Heaviest disk space usage
- Dynamic
  - Fixed and variable length data types (VARCHAR, TEXT)
  - More data file fragmentation
  - More efficient disk space usage
- Compressed
  - Decompressed on the fly
  - Very light disk space usage
  - Read only

MyISAM Variables

Storage Engines

Implicit temporary table use MyISAM, so always give the engine *some* resources, even if it is never used explicitly!

MyISAM indexes are cached:

```
key_buffer_size = 32M
```

Sometime INSERT statements can run concurrently with SELECTs by writing to the end of the data file:

```
concurrent_insert = 1
```

Number of file handles available. Subject to OS limits:

```
open_files_limit = 1024
```

MyISAM Variables 2

Storage Engines

A buffer used for rebuilding indexes. Up to 4G

```
myisam_sort_buffer_size = 1G
```

MyISAM can automatically recover data and index files in the event of a crash:

```
myisam_recover_options = BACKUP, FORCE
```

MyISAM table scans use a dedicated buffer for each client:

```
read_buffer_size = 1M
```

MyISAM Notes

Storage Engines

- MyISAM is the default storage engine in MySQL 5.0 and 5.1, but *not in 5.5*, which defaults to InnoDB
- MyISAM used for *implicit temporary tables* on disk
  - Never reduce MyISAM resources (key_cache, etc) to zero

InnoDB

Storage Engines

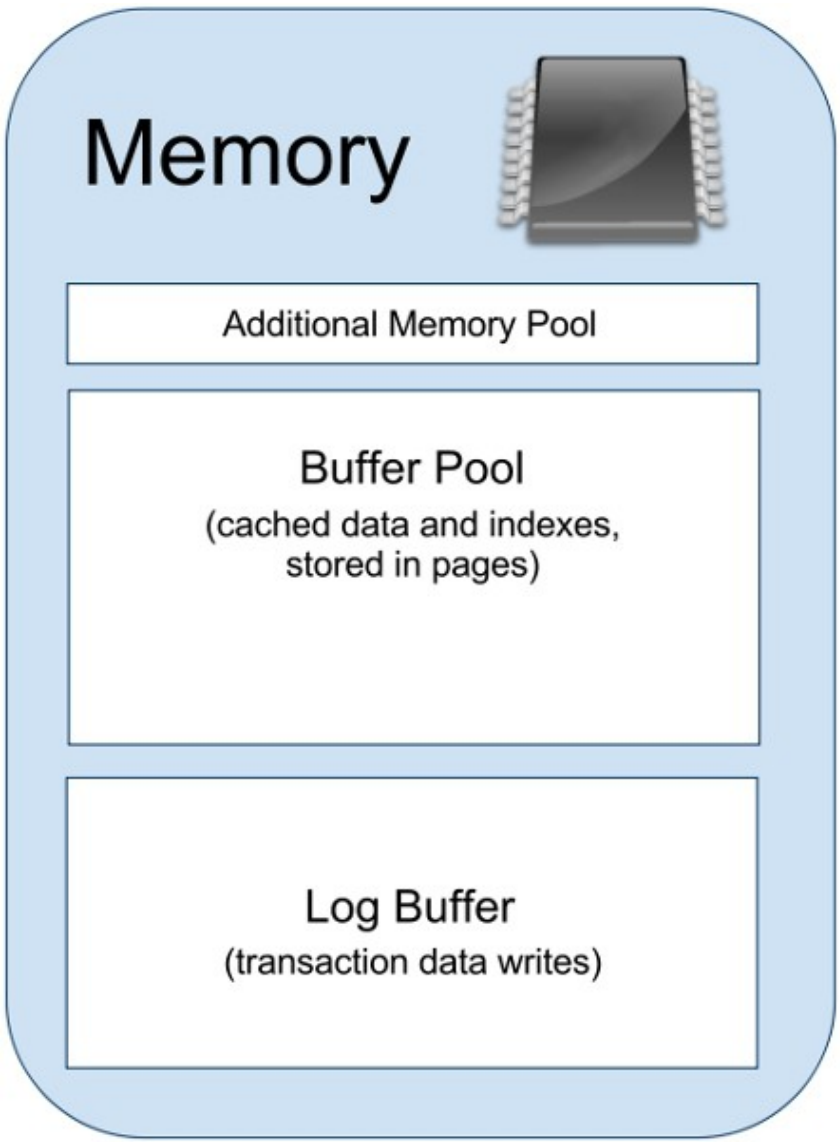
SkySQL

- Fully transactional, and ACID compliant
  - **A**tomicity **C**onsistency **I**solation **D**urability
- Uses row level locking
- Supports all four isolation levels
- Supports Foreign Keys and MVCC
- Both data and indexes cached by mysqld in a buffer pool
- Reliable crash recovery
- Can be slower than MyISAM but higher concurrency
- Uses a global table-space file and two redo log files
  - Optionally, configure individual table space files per table

InnoDB Architecture

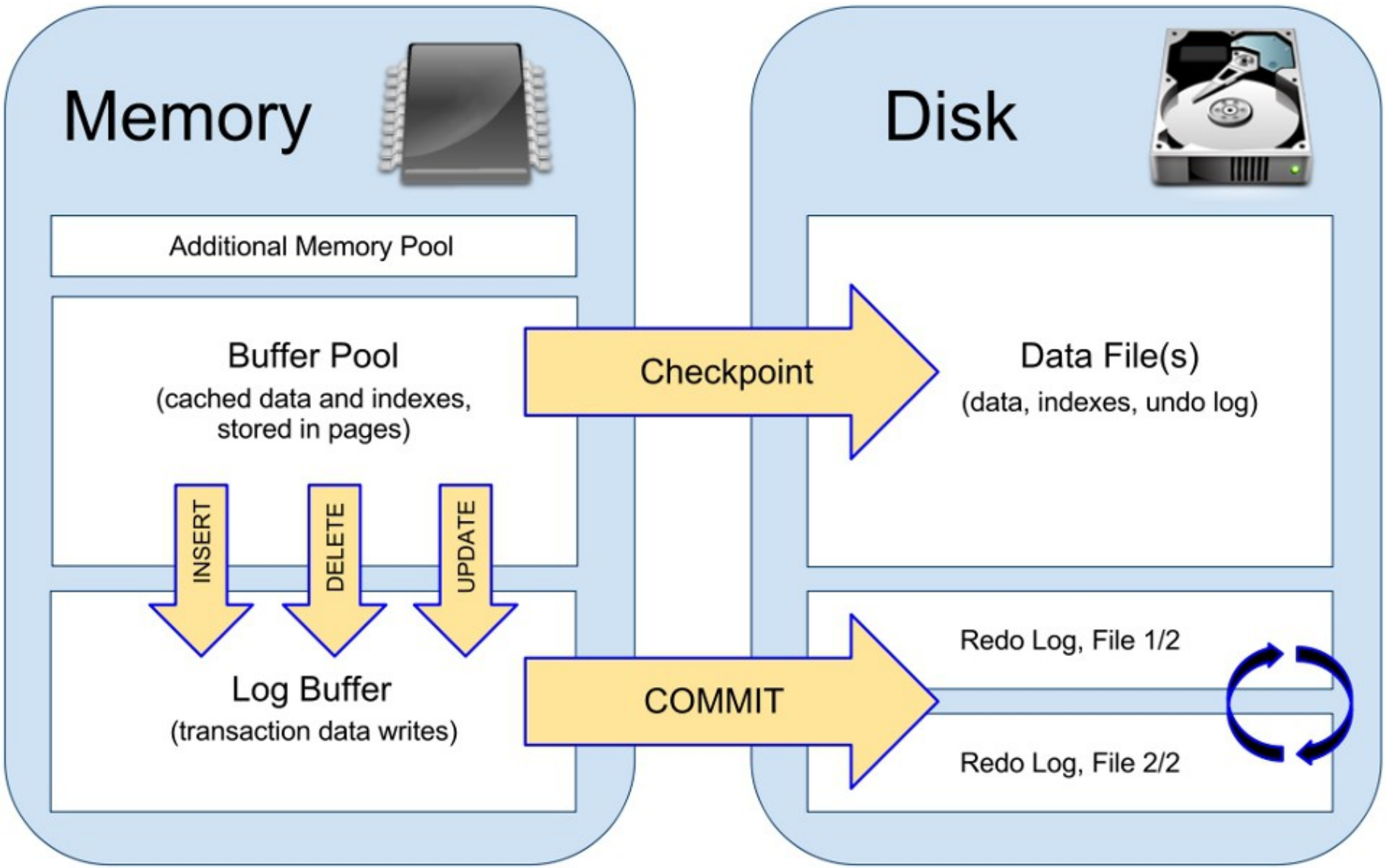
Storage Engines

SkySQL



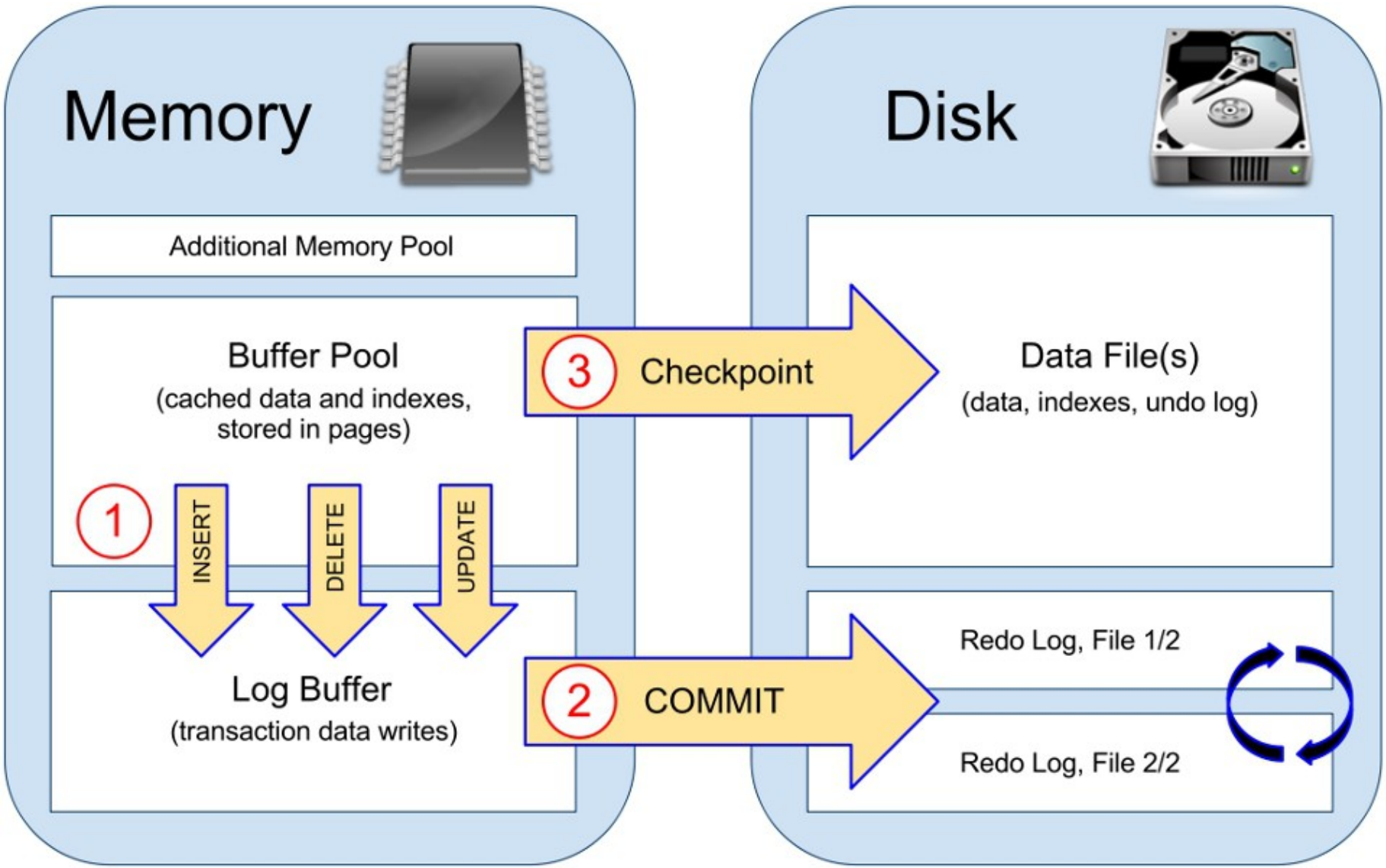
InnoDB Architecture

Storage Engines



InnoDB Architecture

Storage Engines



XtraDB

Storage Engines

- Maintained by Percona
- An enhanced version of InnoDB with patches for:
  - Buffer pool mutex split to increase query concurrency
  - Improved and configurable I/O (now also in InnoDB plugin)
  - Adaptive check-pointing to avoid blocking transactions
  - Faster crash recovery
- Better performance on multi-core systems and high workloads

Aria

Storage Engines

MyISAM replacement...

Aria 1.5

- crash-safe MyISAM alternative
- Activity log with full rollback/replay support
- Allows concurrent INSERT
- FULLTEXT, GIS
- Page cache

Aria 2.0

- MVCC and ACID
- Optional transaction support on per-table basis
- Concurrent UPDATE / DELETE
- Row level locking
- Group Commit

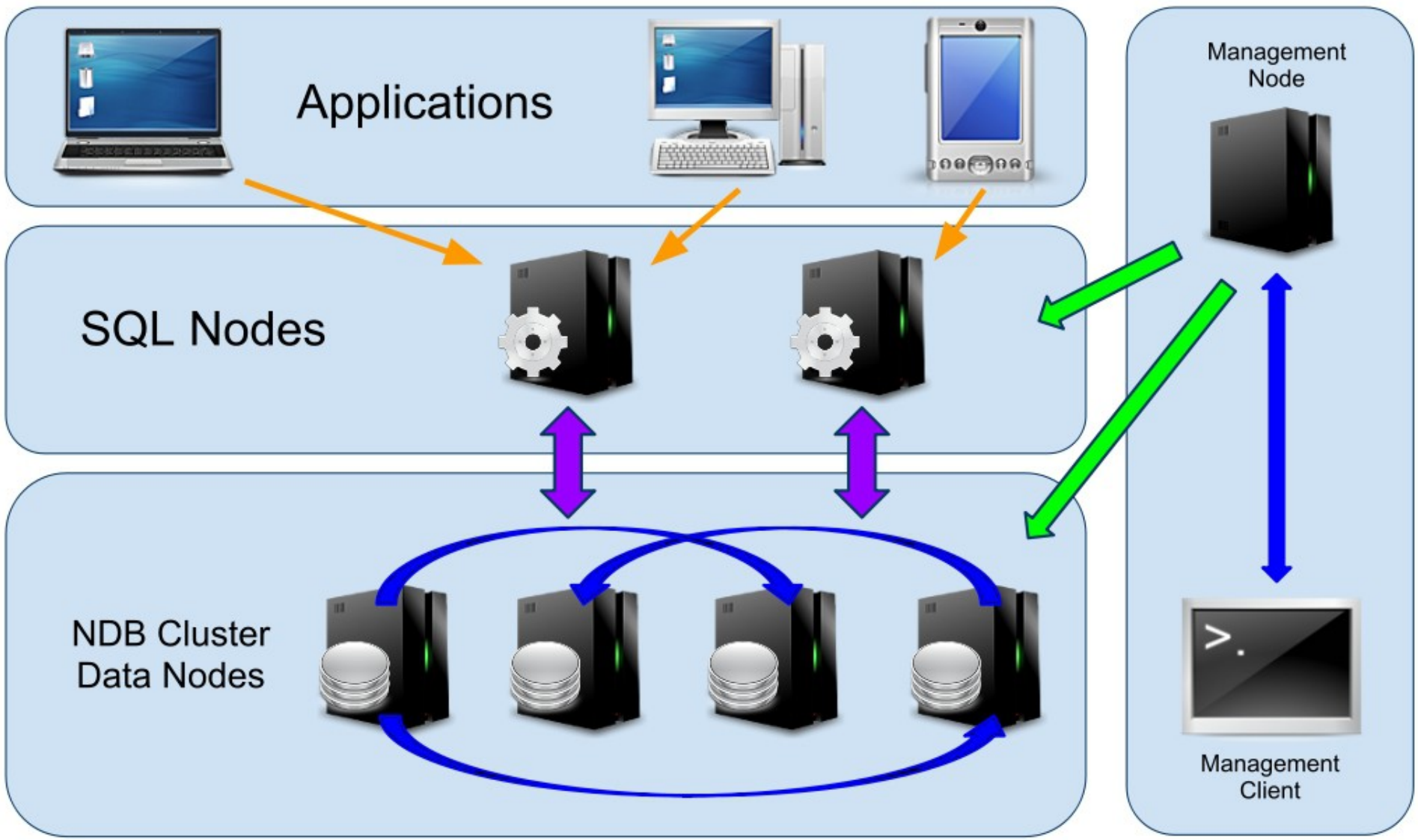
Cluster

Storage Engines

- Also called NDBCluster
  - NDB = **N**etwork **D**ata**b**ase
- A transactional clustered storage engine
  - Runs in memory across multiple servers
  - Allows multiple data (ndbd) and SQL (mysqld) nodes
  - Supports READ-COMMITTED only
- High Availability & Scalability
  - A minimum of three servers is required for HA
  - Individual machine outages do not stop the cluster
  - Backups and upgrades do not require downtime

Cluster Architecture

Storage Engines



Cluster Notes

Storage Engines

- Best Practices
  - Single row access by primary key is very fast
  - Joins can be slower, so keep them small & simple
  - Table scans hurt a lot, so index tables carefully

Memory

Storage Engines

- Structure on disk; data and indexes in memory -- *Fast!*
  - Data will not survive mysqld or system restart
  - Structure will survive
- Uses table level locking
- Allows fixed length records only
  - Variable length data types padded out to full width, so...
  - VARCHAR(100) becomes CHAR(100) behind the scenes!
- Some data types are not supported, such as:
  - TEXT, BLOB

Blackhole

Storage Engines

SkySQL

Supports everything! ... but does not store any data.

How could it possibly be useful?

- Replication. Queries are still written to the binary log
  - Use on a Master to replicate data without storing it locally
  - Use on a relay Slave for custom filtering rules
  - Use on a Slave to prevent storing replicated data locally
- Performance fixes:
  - Temporarily (and instantly) remove excessive log table activity
  - Benchmark to track down non-storage-engine bottlenecks
- On development servers:
  - Don't care about unit test data, just query execution?

Blackhole Architecture

Storage Engines

/dev/null

Archive

Storage Engines

Used for storing large amounts of data without indexes

- Uses the minimum possible memory footprint
- All data is compressed on disk with *zlib*
- Supports INSERT and SELECT statements only
- *Does not support* DELETE, UPDATE or REPLACE
- Useful for archived data or logging tables with frequent writes and infrequent reads
- No data caching except the OS disk cache
- Non-transactional

Others

Storage Engines

Many other storage engines exist!

AWSS3	BDB	CSV
IBMDB2I	ISAM	Infobright
mdbtools	memcache	MERGE
OQGRAPH	PBXT	Q4M
RitmarkFS	ScaleDB	SolidDB

Resource Usage

Administering MySQL

Course Structure

Resource Usage

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. **Resource Usage**
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Global Memory

Resource Usage

SkySQL

- Buffers that are allocated once at mysqld startup
- Some are storage engine dependent:
  - MyISAM: key_buffer_size
  - InnoDB: innodb_buffer_pool_size, innodb_additional_mem_pool_size, innodb_log_buffer_size
  - Storage engines may allocate other memory internally
- Some are general purpose:
  - table_cache
  - query_cache_size
  - thread_cache
  - permissions tables

Session Memory

Resource Usage

- Buffers allocated by each client connection as required
- Some are allocated more than once for joins and sorts
- Released when query is done or client session closed
- Persistent connections need to be reset somewhere!

thread_stack	net_buffer_length	max_allowed_packet
read_buffer_size	read_rnd_buffer_size	sort_buffer_size
join_buffer_size	max_heap_table_size	tmp_table_size
bulk_insert_buffer_size	myisam_max_sort_file_size	myisam_sort_buffer_size
binlog_cache_size		

Estimating Memory Usage

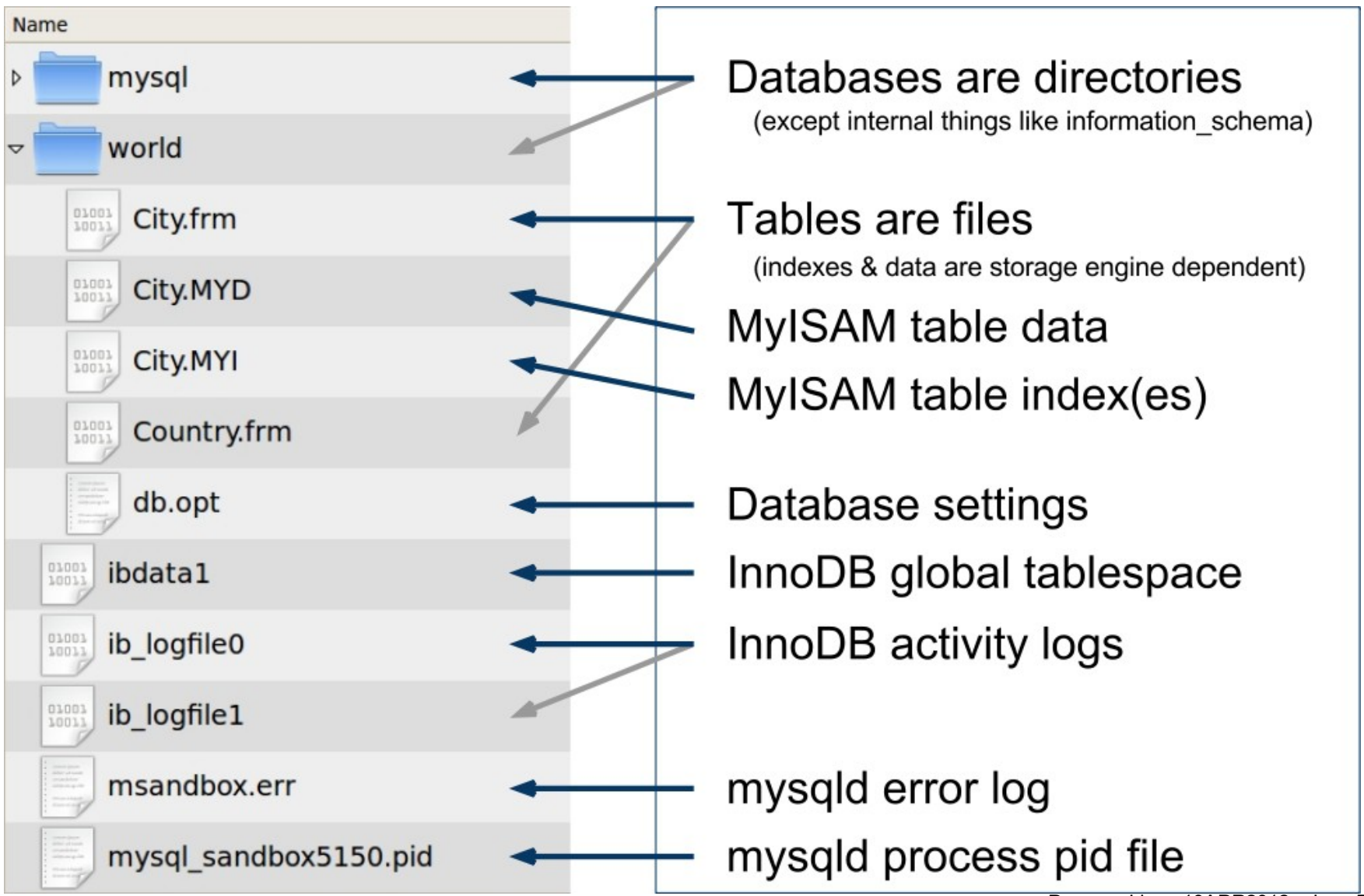
Resource Usage

footprint = sum(global_buffers) +
(max_connections * sum(session_buffers))

- This is the worst case maximum memory usage scenario:
  - The chances of all clients allocating all possible session buffers at the same time, is usually remote
  - Therefore it is quite normal practice to over commit memory
- The Linux "Out Of Memory" (OOM) killer is a risk if mysqld is operating near the system memory ceiling
  - Buy more memory or use a crash safe storage engine
- Always have system SWAP space, but try hard never to let MySQL start swapping during normal load

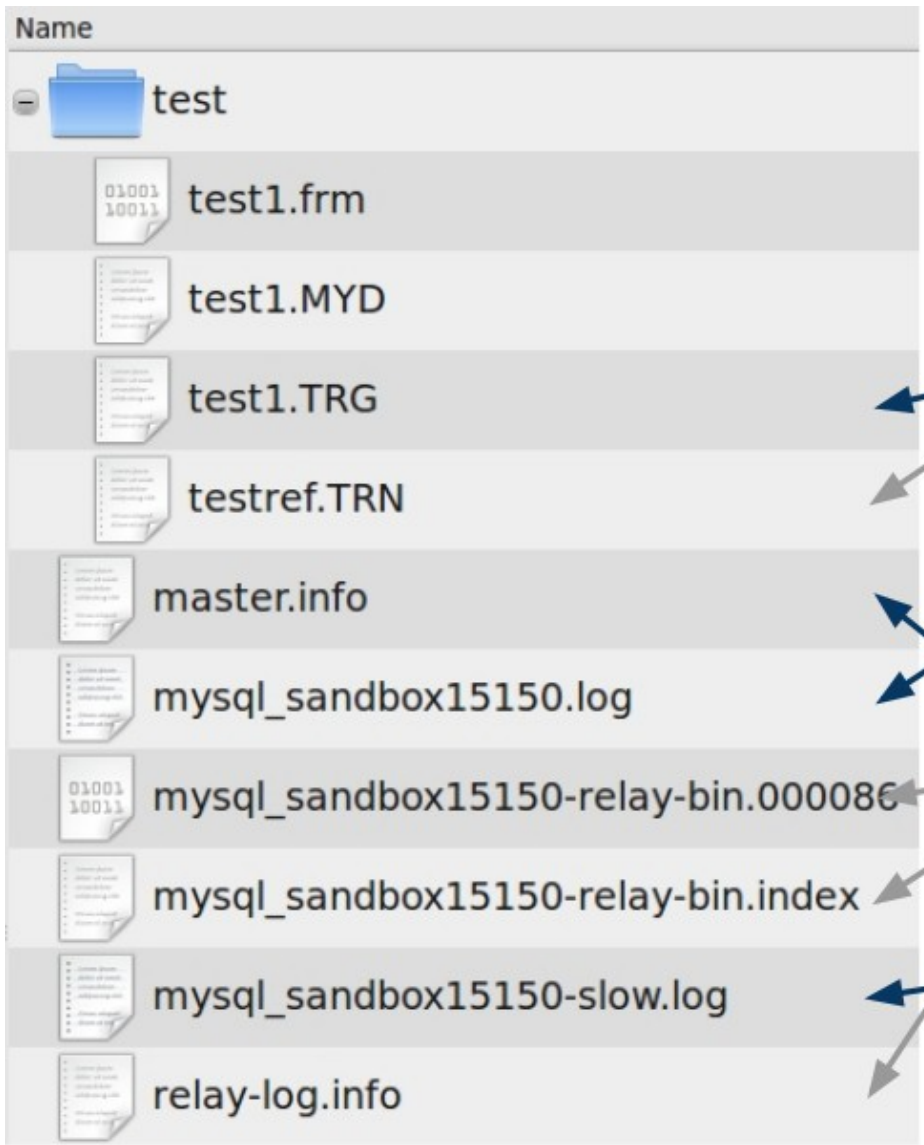
Disk Data Directory

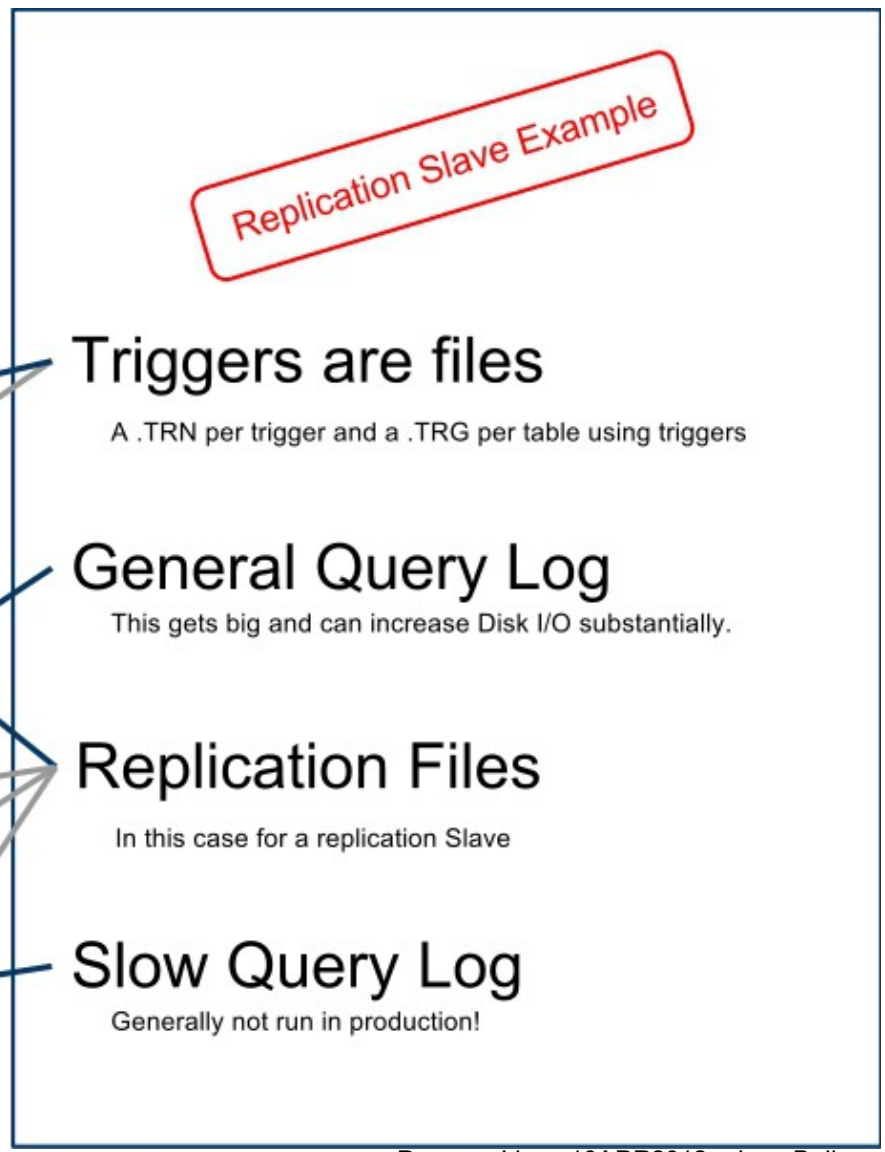
Resource Usage



Disk Data Directory 2/2

Resource Usage





Temporary Disk Space

Resource Usage

- MySQL uses temporary disk space for:
  - Large implicit temporary tables, used for resolving queries
  - Some sort operations (ORDER BY, GROUP BY)
- tmpdir defaults to system temporary location
  - Be careful - some Unix platforms mount /tmp as tmpfs
  - Not necessarily bad if your traffic / dataset is suitable
- slave_load_tmpdir for replicating LOAD DATA INFILE
  - Defaults to the same location as tmpdir
  - Slave exports from binlog to file here, then LOAD DATA INFILE
  - This one really should never be tmpfs!

Maintenance

Administering MySQL

Course Structure

Maintenance

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. **Maintenance**
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Tables

Maintenance

OPTIMIZE TABLE

- Defragment data files and sort indexes for MyISAM
- Simply maps to a full table rebuild for InnoDB

ANALYZE TABLE

- Check and store index cardinality values
- Useful to run periodically but infrequently

CHECK TABLE

- Check table for errors
- Arguments: QUICK, FAST, CHANGED, MEDIUM, EXTENDED

REPAIR TABLE

- Try to fix a corrupt table
- Arguments: QUICK, EXTENDED, USE_FRM

Log Files

Maintenance

SkySQL

- Log files need to be rotated periodically
  - The General and Binary logs can take up a *lot* of disk space!
  - So can the Slow Query Log (beware --log-queries-not-using-indexes)
- To rotate normal logs on (General and Slow):
  1. Move the existing files manually
  2. Then issue a FLUSH LOGS or mysqladmin flush-logs (causes the server to close and re-open/re-create log files)
- It is advisable to include logs in backups if practical
  - If problems arise. the General and Slow logs can provide a useful forensic resource and/or historical performance information
  - The Binary logs should be synchronized with backups
- *None of the above applies to the mysqld error log!*
  - To rotate the error log: stop mysqld, move the log, restart mysqld

Binary Log

Maintenance

- FLUSH LOGS triggers a Binary Log roll-over
  - Binary log file use incrementally numbered extensions
- Binary logs can be set to auto expire in my.cnf:

```
expire_logs_days = 7
```

  - Ensure enough time is left for all Slaves to read the files!
- Alternatively, binary log files can be purged explicitly:

```
PURGE BINARY LOGS TO 'hostname-bin.010';
PURGE BINARY LOGS BEFORE '2011-01-01 10:00:0';
```

- Inspect current binary logs with:

```
SHOW BINARY LOGS;
```

Backups

Administering MySQL

Course Structure

Backups

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. **Backups**
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Principles

Backups

- Even with a HA solution in place, you still need backups
  - For example, HA cannot protect against user error
- Consider both nightly snapshots and real time backup
  - For snapshots: mysqldump, InnoDB hot backup or LVM
  - For (near to) real time: Replication
- Store backups in multiple places:
  - Onsite for fast access and offsite for security
- Include both data and config files in your backup
  - And logs too? They can be useful for later forensics
  - Synchronize binary logs with backup files!

Test your backup recovery! How long does it take?

Logical

Backups

- Produce text files with SQL statements that can be replayed to rebuild the database
- Allows you to dump specific tables, whole databases, schema or data only
- An SQL dump is storage engine independent, and can be recovered to a *different* engine or used for migration
- The process can be slow and requires locks. Use a local drive rather than dumping across the network
- Multiple options exist:
  - mysqldump
  - SELECT INTO OUTFILE

Logical 2/2

Backups

- Recovery is a simple as reloading the SQL text file

```
$ cat backup.sql | mysql
```

```
mysql> source backup.sql
```

- Consider your character sets. mysqldump outputs UTF8 text files

Physical

Backups

- Produces a binary copy of your data
- Often faster than dumping to SQL text file
- Can only be recovered to the same storage engine, and is no use for migrations
- In the event of file corruption, errors remain unseen and affect the backup too
- Multiple options exist:
  - Manual data directory file copy (requires mysqld to be stopped; locking tables is not sufficient for some storage engines)
  - LVM on Linux to take a volume snapshot (data will be complete and consistent, but will still trigger InnoDB recovery after restore)
  - InnoDB Hot Backup tool (commercial) or XtraBackup

Replication

Backups

- Use MySQL built in replication to move data to a slave server (close to real time, but technically asynchronous)
- Recovery can be as fast and simple as switching application traffic to the slave server
  - Then restore the primary server offline and switch back
  - Also possible to replay binary logs, with mysqlbinlog tool, for more complex recovery situations
- Allows other backups methods to be executed on the slave server without increasing load on the primary server
- Allows the use of other replication features

Replication will be covered in detail later in the course...

NDB

Backups

- Instruct each data node to create a local data snapshot
- Can run online without noticeable affect to the cluster
- Requires using NDB, which is not necessarily a simple drop in replacement (depends on the application and traffic)

Data Types

Administering MySQL

Course Structure

Data Types

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. **Data Types**
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Overview

Data Types

- MySQL supports integer, string, decimal, temporal, spacial and custom data types
- The mysql client is a good place to start investigation:

```
mysql> help int;
Name: 'INT'
Description:
INT[(M)] [UNSIGNED] [ZEROFILL]

A normal-size integer. The signed range is -2147483648 to 2147483647.
The unsigned range is 0 to 4294967295.

URL: http://dev.mysql.com/doc/refman/5.1/en/numeric-type-overview.html
```

- For efficiency, use the most precise suitable data type
  - Be sure the chosen data type will store all possible values required by the application
  - MySQL sometimes silently truncates / rounds. Check sql_mode

Choosing Data Types

Data Types

- Use the most appropriate type
  - For example: store numbers in a numeric field, not in a VARCHAR
- Use accurate types where required
  - DECIMAL vs DOUBLE
- Use the smallest useful type
  - On-disk size != in-memory size
  - Variable length fields are often padded out
- Use NOT NULL if practical
  - Making a field NULL-able uses slightly more disk and memory (depends on storage engine)
- Use PROCEDURE ANALYSE()

Column Attributes

Data Types

- As you might expect, most types allow default values to be specified in the CREATE TABLE statement:

```
CREATE TABLE people
    (name VARCHAR(20) DEFAULT 'unknown');
```

- Columns can be NULL, unless defined NOT NULL
  - NULL means "no value" or "not applicable" or "unknown"
  - NULL is distinct and different to an empty string or number zero
  - NOT NULL reduces storage requirements in some engines
  - But, don't avoid NULL if it is appropriate to the application

Integers

Data Types

TINYINT SMALLINT MEDIUMINT INTEGER BIGINT

- Use UNSIGNED if it makes sense for the app
- INT(N) specifies display precision, not storage precision
- sql_mode defines how out of range values are handled
  - *strict* mode causes an error to be thrown
  - ... otherwise the value is silently clipped to fit!
- Actual size/precision on disk is storage engine dependent
  - Eg: InnoDB uses INTEGER for MEDIUMINT anyway!

AUTO_INCREMENT

Data Types

AUTO_INCREMENT is an alternative to a *sequence.*

- LAST_INSERT_ID() returns the latest auto_inc value generated by the current client connection
- SERIAL is a useful synonym for:

```
BIGINT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE
```

- MyISAM auto_inc counters never decrease unless they are manually reset, or the counter value wraps
- InnoDB prepares auto_inc counters at mysqld startup:

```
SELECT MAX(auto_inc_field) FROM table
```

  - Deleted rows could mean auto_inc values are reused - watch out!

Floating and Fixed Point

Data Types

FLOAT DOUBLE DECIMAL NUMERIC REAL

- FLOAT and DOUBLE are approximate types, using 4 and 8 bytes IEEE storage format
- DECIMAL(M,D) means M digits total (precision), with D digits after the decimal point (scale)
- DECIMAL is an exact value type up to 65 digits precision, requiring 4 bytes storage for each multiple of nine digits
- NUMERIC is a synonym for DECIMAL
- REAL is a synonym for DOUBLE
  - Unless the REAL_AS_FLOAT SQL mode is active

String

Data Types

SkySQL

CHAR	VARCHAR	TINYTEXT
TEXT	MEDIUMTEXT	LONGTEXT

- CHAR(N), where N is the width in characters, not bytes
- VARCHAR becomes CHAR in implicit temporary tables and mysqld internal buffers, so keep them as short as possible
- TEXT types are not supported by the MEMORY engine
  - So implicit temporary tables may convert to MyISAM on disk!
- All string data types have a character set

Character Set & Collation

Data Types

- Character set may be specified at the global, schema, table or column level
- Multi-byte character sets increase disk storage and working memory requirements
  - Eg: UTF-8 requires up to 3 bytes per character, so for a CHAR(10) mysqld needs 30 bytes of memory to ensure the data fits
- Collations affect string comparison (character order)
- Collations can be changed for each query execution:

```
SELECT * FROM table1
   ORDER BY col1 COLLATE latin1_german2_ci;
```

Binary

Data Types

BINARY	VARBINARY	TINYBLOB
BLOB	MEDIUMBLOB	LONGBLOB

- BINARY and VARBINARY are case-insensitive CHAR and VARCHAR respectively
- Binary types have no character set or collation. Any comparison is always simple byte order
- Blobs are often used to store files in the database
  - Pro: Blobs are included in transactions, replication, backup
  - Con: Files are often faster and blobs inflate mysqld memory usage

Temporal

Data Types

SkySQL

DATE TIME DATETIME TIMESTAMP YEAR

- DATE range is *1000-01-01* to *9999-12-31* (YYYY-MM-DD)
- DATETIME is the same range with second precision: YYYY-MM-DD HH:mm:ss
- TIMESTAMP is internally equivalent to a Unix timestamp, a second count between 1970-01-01 and 2039-01-19
  - Many apps prefer to simply store UNIX_TIMESTAMP() values in an unsigned integer field
- TIME range is -838:59:59 to 838:59:59
- YEAR can be 4 digits (1901 - 2155) or 2 ([19]70 - [20]69)

Manipulating Dates

Data Types

There are a wide range of functions for date manipulation.

ADDDATE	ADDTIME	CONVERT_TZ	CURDATE	CURTIME
DATE_ADD	DATE_FORMAT	DATE_SUB	DATE	DATEDIFF
DAYNAME	DAYOFMONTH	DAYOFWEEK	DAYOFYEAR	EXTRACT
FROM_DAYS	FROM_UNIXTIME	GET_FORMAT	HOUR	LAST_DAY
MAKEDATE	MAKETIME	MICROSECOND	MINUTE	MONTH
MONTHNAME	NOW	PERIOD_ADD	PERIOD_DIFF	QUARTER
SEC_TO_TIME	SECOND	STR_TO_DATE	SUBTIME	SYSDATE
TIME_FORMAT	TIME_TO_SEC	TIME	TIMEDIFF	TIMESTAMP
TIMESTAMPADD	TIMESTAMPDIFF	TO_DAYS	UNIX_TIMESTAMP	UTC_DATE
UTC_TIME	UTC_TIMESTAMP	WEEK	WEEKDAY	WEEKOFYEAR
YEAR	YEARWEEK			

Other

Data Types

BIT ENUM SET

- BIT(N) allows storage of bitmaps up to 64 bits wide
  - Specify bitmaps with *b'1101101'* syntax
- An ENUM is a string object with a value chosen from a list of specified values

```
CREATE TABLE t1 (c1 ENUM('alpha','beta' 'gamma'));
```

- Storage is efficient, using only a 2 byte integer index into the list

- A SET is a string object that can have zero or more values from a list of specified values

Views

Administering MySQL

Course Structure

Views

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. **Views**
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

What are they?

Views

A VIEW is just an SQL statement represented as a table.

```
CREATE VIEW e AS
  SELECT emp_no, first_name, last_name FROM employees
  WHERE last_name LIKE 'e%';
```

```
mysql> SELECT * FROM e LIMIT 5;
+--------+------------+-----------+
| emp_no | first_name | last_name |
+--------+------------+-----------+
|  10021 | Ramzi      | Erde      |
|  10087 | Xinglin    | Eugenio   |
|  10122 | Ohad       | Esposito  |
|  10131 | Magdalena  | Eldridge  |
|  10145 | Akemi      | Esposito  |
+--------+------------+-----------+
```

Partitioning

Administering MySQL

Course Structure

Partitioning

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. **Partitioning**
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Overview

Partitioning

Split up a table so it is stored in multiple locations.

Why?

- If a table needs to be larger than a single disk or file system supports, partition it
- Easier maintenance. Work with single partitions (less data, fewer locks, faster operations) instead of the entire table
- Partition Pruning. Improve performance of those queries that need touch only specific partitions

Security

Administering MySQL

Course Structure

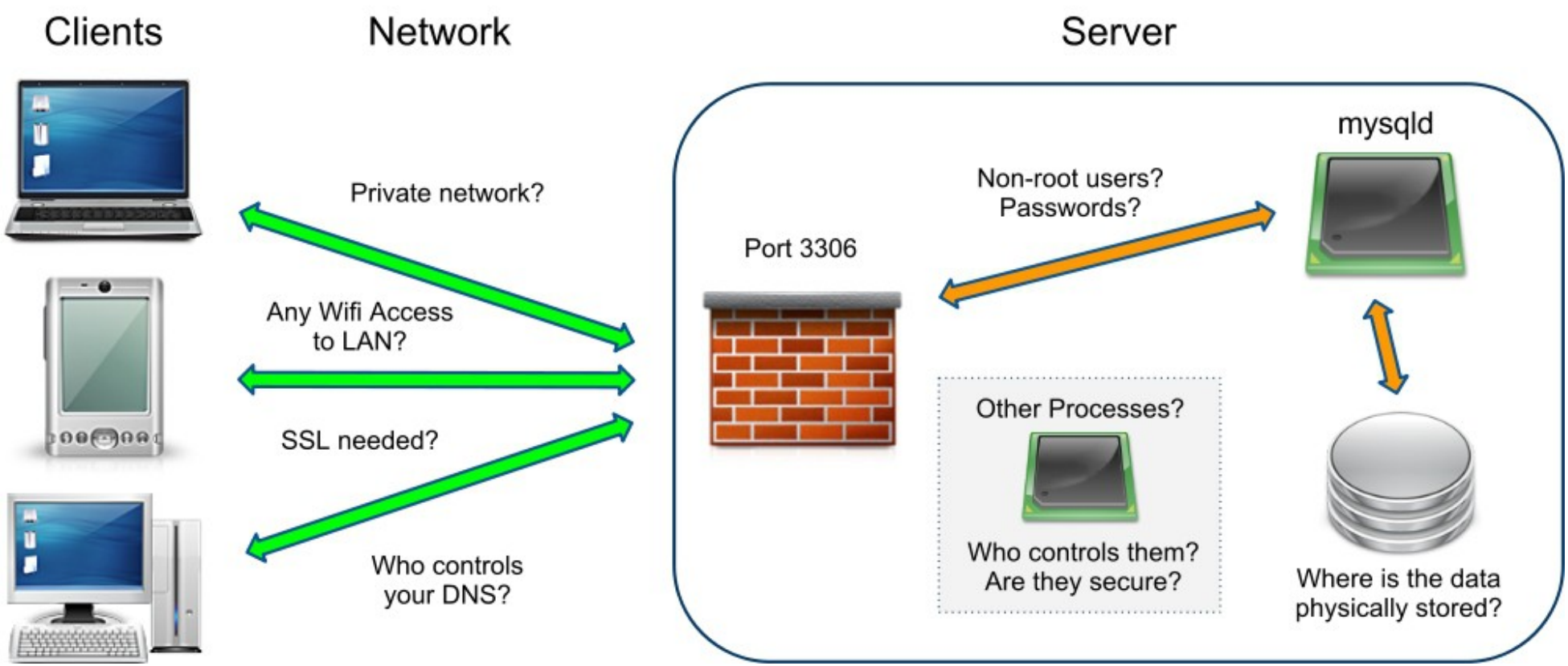
Security

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. **Security**
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Overview

Security

Security is a multi-layered challenge!



Network

Security

- Firewall
- If you have only local clients on a Unix server, disable network access and use the socket only
- Is your network private? MySQL traffic includes plain text queries, usernames, passwords, so consider SSL.
- Is there any company wifi access to the network?
- DNS - who controls it? Do you trust them? What happens if it goes down?
- User access control allows use of IP as well as hostname
- Reverse lookups still done unless --skip-name-resolve

Server

Security

SkySQL

- Don't run mysqld as root / Administrator
- Set the MySQL user root password and disable the anonymous access account
- Who has file level read/write access to the data directory?
- Are the logs in a secure location? Some logs have queries and data in plain text, and binlog can be dumped
- Are passwords visible in the OS process list?
  - mysql cli obfuscates passwords supplied on the command line. Do your app and scripts need to do the same?
- What other services run on the server? Who controls them?

MySQL Accounts

Security


```
CREATE USER 'User'@'Host' IDENTIFIED BY 'Password';
```

- Authentication is based on User, Host, and Password
- A MySQL *account* is both User and Host
  - *User* alone is a different account to *User@Host*
  - An empty string is a User wildcard
  - A % (percent symbol) string is a Host wildcard
- You can use hostnames or IPs for Host (check DNS!)
- *localhost* means the local socket on Unix

MySQL Accounts

Security

- CREATE USER is complemented by DROP USER and RENAME USER
- Passwords can be changed with:
  - `SET PASSWORD [FOR user] = PASSWORD('mypass')`
  - Omitting [FOR user] changes your own password

Permissions Tables

Security

- Access permissions are checked upon connection and for every query
- The *mysql* database holds the permissions tables, including all user accounts, hosts and privileges
  - The tables are loaded into memory at startup for fast access
  - It is possible to INSERT / DELETE directly on them
  - Need to use FLUSH PRIVILEGES to reload any manual changes
  - Restrict user access to the *mysql* database!
- The *mysql* database may be dumped and backed up just like any other database

Granting Privileges

Security


```
GRANT priv ON [obj] level TO 'User'@'Host'
```

Privilege levels are global, schema, table, routine or column:

ON *.*	Global
ON db.*	Database
ON [TABLE] db.table	Table
ON PROCEDURE db.routine	Routine
ON TABLE db.table	Column

GRANT can also replace CREATE USER to create an account and apply privileges in one step:

```
GRANT ... IDENTIFIED BY 'Password'
```

Available Privileges

Security

ALL [PRIVILEGES]	ALTER	ALTER ROUTINE
CREATE	CREATE ROUTINE	CREATE TEMPORARY TABLES
CREATE USER	CREATE VIEW	DELETE
DROP	EVENT	EXECUTE
FILE	**GRANT OPTION**	INDEX
INSERT	LOCK TABLES	PROCESS
REFERENCES	RELOAD	REPLICATION CLIENT
REPLICATION SLAVE	SELECT	SHOW DATABASES
SHOW VIEW	SHUTDOWN	SUPER
TRIGGER	UPDATE	USAGE

Reviewing Privileges

Security


```
SHOW GRANTS FOR 'User'@'Host'
```

- Note: SHOW GRANTS only displays grants for the explicitly listed 'User'@'Host'. Be sure you are checking the right one!
- To see your own grants, use:

```
SHOW GRANTS FOR CURRENT_USER;
```

- Alternatives to SHOW GRANTS include:
  - Querying the INFORMATION_SCHEMA tables
  - Querying the mysql database permissions tables directly

Revoking Privileges

Security


```
REVOKE priv ON [obj] level FROM User@Host
```

- The administrative user executing the REVOKE statement needs the GRANT OPTION privilege
- **priv** can be any privilege levels (they don't have to exactly match the user's privileges), or ALL PRIVILEGES
- **obj**, **level** and **User@Host** are the same as for GRANT
- While GRANT can create a user implicitly, REVOKE does not ever remove a user account. Use DROP USER
- Use SHOW GRANTS after a REVOKE to be sure it had the desired effect

User Limits

Security


```
GRANT ... WITH MAX_QUERIES_PER_HOUR 20
```

- User accounts may have some resource limits applied:

  MAX_QUERIES_PER_HOUR MAX_CONNECTIONS_PER_HOUR
  MAX_UPDATES_PER_HOUR MAX_USER_CONNECTIONS

- mysqld tracks these with per-user counters stored in the *mysql* database
- FLUSH USER_RESOURCES or FLUSH PRIVILEGES reset user counters, except MAX_USER_CONNECTIONS
- Counters reset to zero upon mysqld restart

Programming
Administering MySQL

Course Structure

Programming

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. **Programming**
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Stored Routines

Programming

- SQL:2003 syntax is used for stored routines
- Both procedures and functions are supported
- Databases are used as namespaces: CALL test.my_proc()
  - Routines are dropped along with the parent database
- Useful for:
  - Business logic accessible to more than one application
  - Isolating certain functionality at the database level for security
  - Libraries of common functions

```
CREATE FUNCTION say_hi (s CHAR(20))
   RETURNS CHAR(50) DETERMINISTIC
   RETURN CONCAT('Hi there, ',s,'!');
```

Replication

Administering MySQL

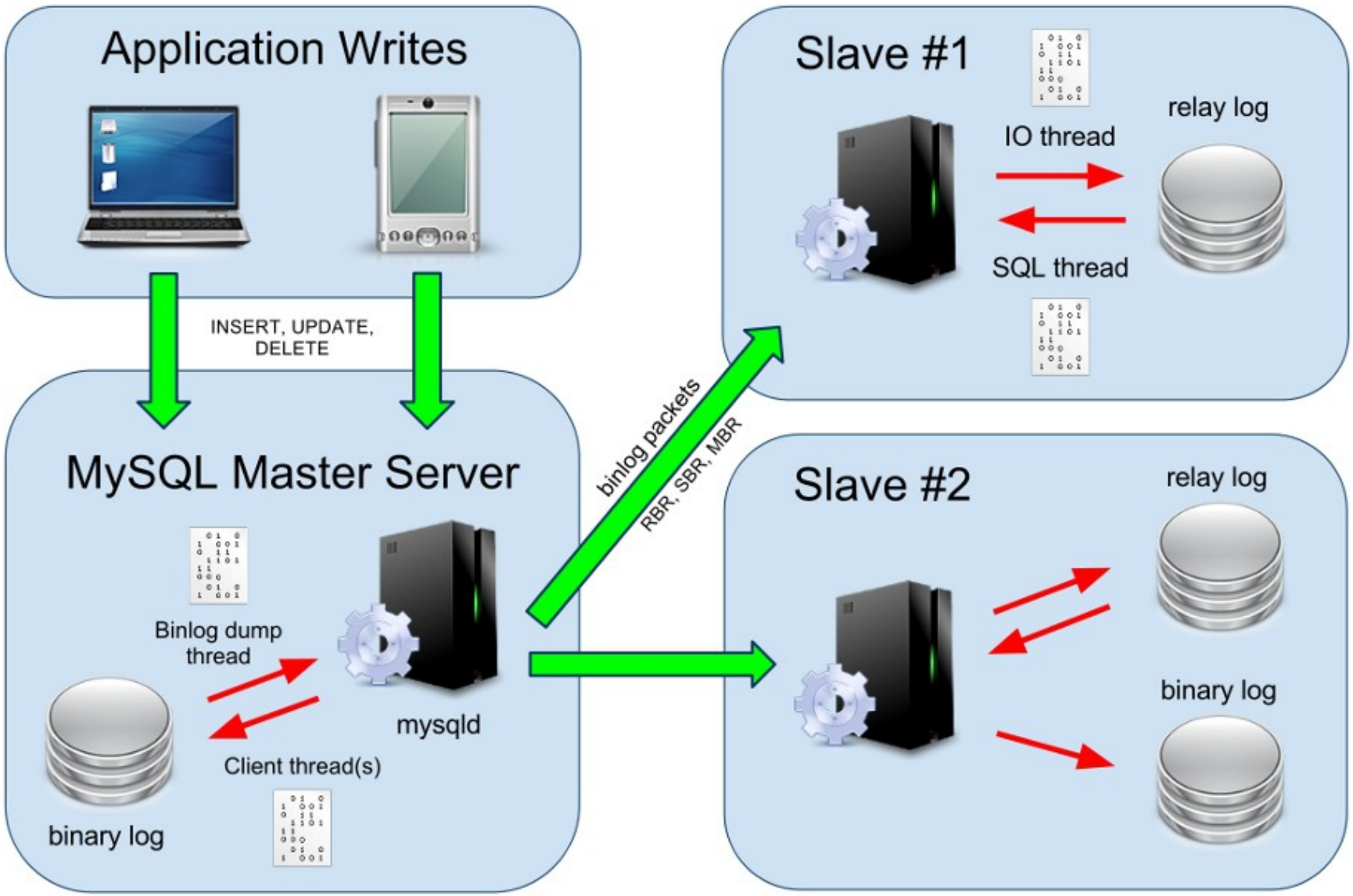
Course Structure

Replication

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. **Replication**
16. Monitoring
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

Architecture

Replication



Monitoring

Administering MySQL

Course Structure

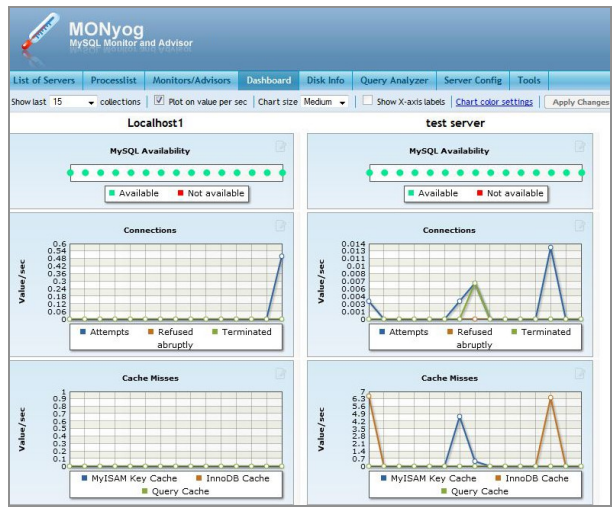
Monitoring

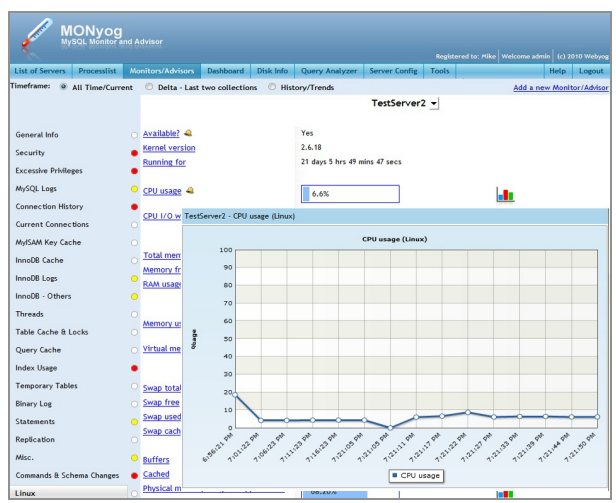
1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. **Monitoring**
17. Troubleshooting
18. Optimization
19. High Availability
20. Conclusion

SkySQL Enterprise Monitor

Monitoring

- A user friendly, cross platform, GUI
- No server-side agent required
- Manages multiple servers in one place
- Track and alert on many metrics:
  - monitor error log / security vulnerabilities
  - replication lag / excessive resource usage
  - slow queries / deadlocks
- Allows programming custom advisors
- Detailed history / trend analysis





Backed by a SkySQL Ab partner: *Webyog, Inc*

Troubleshooting

Administering MySQL

Server Status

Troubleshooting

SkySQL

- Is mysqld hitting a bottleneck? CPU / Disk / Memory / Net
- Collect system stats. Capture multiple snapshots over time, so you can identify any trends or patterns
  - A good monitoring package may do this for you through graphs or reports
- Is the problem unpredictable, or recurring regularly?
- Is anything else running on the server? Check cron too
- With multiple servers: are all affected, or just one?

Optimization

Administering MySQL

Course Structure

Optimization

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. **Optimization**
19. High Availability
20. Conclusion

Overview

Optimization

- Optimization is important
  - Have developers give careful thought to schema and indexes
  - But don't let them waste time over optimizing too early in the game
  - Most of the time you need to see real world dataset and traffic to optimize a database properly
- Optimizing for performance is about finding bottlenecks to handle more traffic or data per unit time
- Optimizing for efficiency is about getting the best use from hardware / bandwidth / money
- Often they are the same thing anyway!

Indexes

Optimization

- Indexing is the first option for performance optimization
- Just as an index allows someone to quickly look up the phone book, it also lets MySQL to resolve queries faster
- Indexes allow faster retrieval of matching rows, and in some case faster sorting of result sets
- Indexes can be added to single columns, or across multiple columns (latter are *composite indexes*)

```
CREATE TABLE people (
    id   INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(20) NOT NULL,
    age  TINYINT UNSIGNED,
    INDEX (name), INDEX (age, name) );
```

Query Analysis

Optimization

- Use the Slow Log to capture problem queries
- Use the mysqldumpslow utility to reduce the log to a manageable report (the slow log is often large)
- Use EXPLAIN to determine how MySQL executes the query and whether indexes are used:

```
mysql> EXPLAIN SELECT * FROM employees WHERE MONTH(birth_date) = 8;
+----+-------------+-----------+------+---------------+------+---------+------+--------+-------------+
| id | select_type | table     | type | possible_keys | key  | key_len | ref  | rows   | Extra       |
+----+-------------+-----------+------+---------------+------+---------+------+--------+-------------+
|  1 | SIMPLE      | employees | ALL  | NULL          | NULL | NULL    | NULL | 299587 | Using where |
+----+-------------+-----------+------+---------------+------+---------+------+--------+-------------+
```

- Use EXPLAIN EXTENDED and SHOW WARNINGS to see how MySQL rearranged a query before execution:

```
mysql> EXPLAIN EXTENDED SELECT ...;
mysql> SHOW WARNINGS;
```

Tables

Optimization

- Minimize table size on disk and in memory
- Archive table data when possible and appropriate
- Remove duplicate or unused indexes
  - No need to index a column if it is already the first field in another covering index
- Use appropriate data types. Smaller is better
  - Don't make variable length fields needlessly long.
  - They are compact on disk but may be expanded out to full width for some operations in memory
  - Use PROCEDURE ANALYSE() for field type guidance
- Consider sharding large tables across multiple servers, if the application can handle it

PT Course

Optimization

SkySQL

Get the best from your MySQL Database server.

Learn how to tune MySQL, its various storage engines and your application itself, for speed and efficiency.

- Choose the best storage engines to suit your app
- Configure MySQL for the best use of your hardware resources
- Design your schema, normalize it, and know when and where to de-normalize!
- Tune your queries with inside knowledge of the optimizer and the storage engines
- Design your app to handle scaling out with MySQL Replication or MySQL Cluster
- Learn how and when to *batch*, *shard* and *load balance*... and what they mean!

High Availability

Administering MySQL

Course Structure

High Availability

1. Introduction
2. Installation
3. Client Utilities
4. Architecture
5. Configuration
6. Storage Engines
7. Resource Usage
8. Maintenance
9. Backups
10. Data Types
11. Views
12. Partitioning
13. Security
14. Programming
15. Replication
16. Monitoring
17. Troubleshooting
18. Optimization
19. **High Availability**
20. Conclusion

Overview

High Availability

SkySQL

In the event of a crash, data should never be lost and users should never notice!

- A Highly Available system is one that can remain operational despite unforseen problems and outages
- Usually, HA means having multiple system redundancy in place at both software and hardware levels
- Software may be designed to write data to multiple physical devices or locations, in case one fails (software RAID, MySQL Replication, MySQL Cluster or DRBD)
- Hardware may be allocated to have a second database server acting as a hot spare, ready to take over instantly

DRBD & Heartbeat

High Availability

SkySQL

Distributed **R**eplicated **B**lock **D**evice

A Linux kernel module providing synchronous replication of a block device between two machines.

It's a hot spare server. If the primary fails, the secondary can take over seamlessly.

Since disk writes must traverse the network, DRBD can make MySQL slower. But, it offers true High Availability.

HA Course

High Availability

Make your MySQL environment bullet proof.

Learn how to setup MySQL to take advantage of many High Availability techniques and technologies.

- Replication
- DRBD and Heartbeat
- MySQL Cluster
- MySQL Proxy
- Monitoring and Fail Over
- ACID Storage Engines
- Virtualization and the Cloud
- Shared storage / SAN vs RAID

Conclusion

Administering MySQL

Thanks for attending!

Conclusion

- Questions?