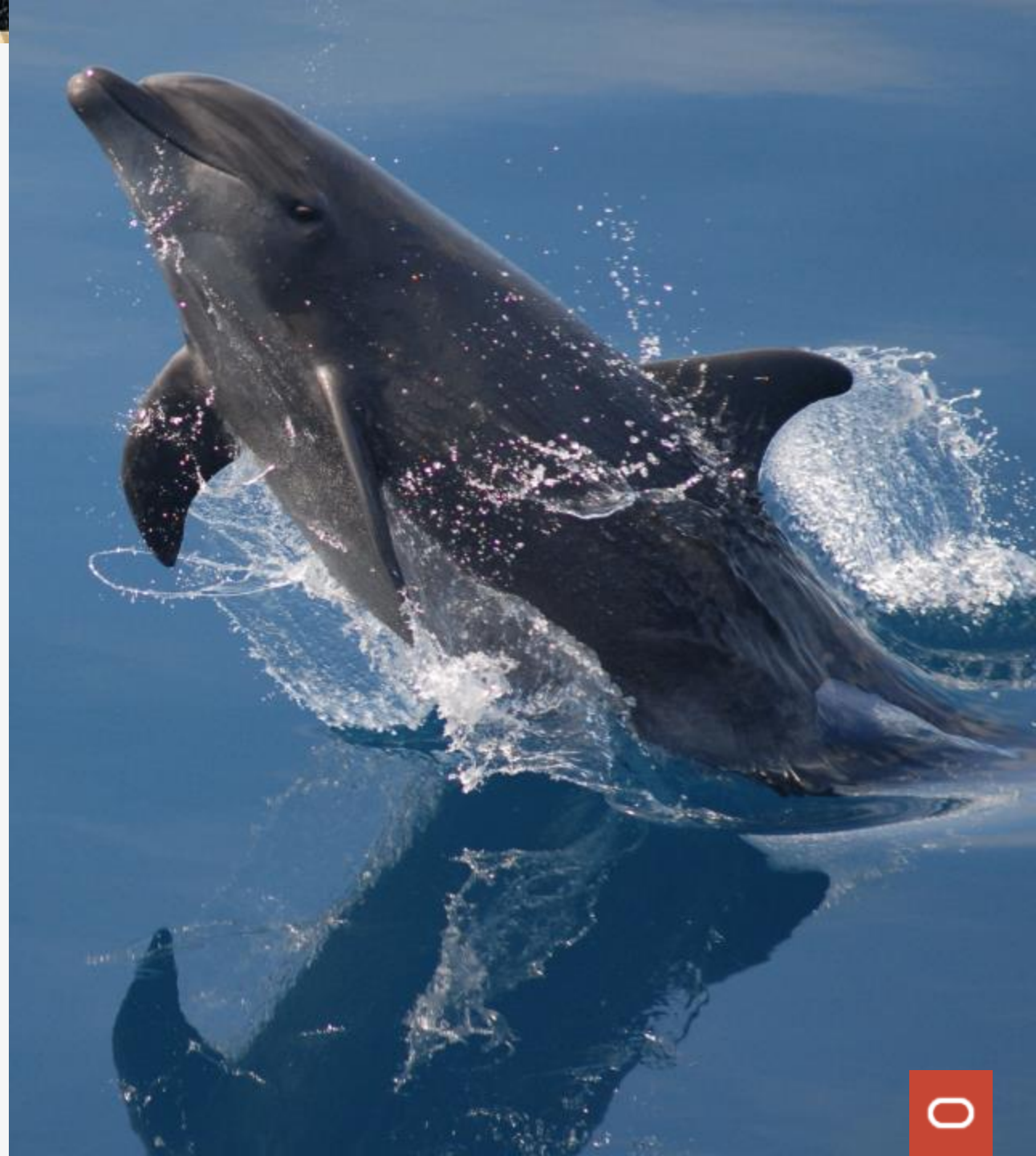
ORACLE

MySQL Enterprise Edition

The Complete Guide

Chandan Kumar
MySQL Solution Architect
MySQL Business Unit



DAY - 01

1. MySQL Architecture
2. NoSQL in MySQL – Document Store
3. MySQL Backup Tool
4. Monitoring Tool
5. MySQL Performance Tuning

Day 01:-10:30-05:00 PM

Safe harbor statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

Confidential – © 2019 Oracle Internal/Restricted/Highly Restricted

MySQL Enterprise Edition

The Highest Levels of MySQL Performance, Security & Uptime

Advanced Features

- Scalability
- High Availability
- Security
- Encryption + TDE
- Data Masking
- Audit
- Firewall

Management Tools

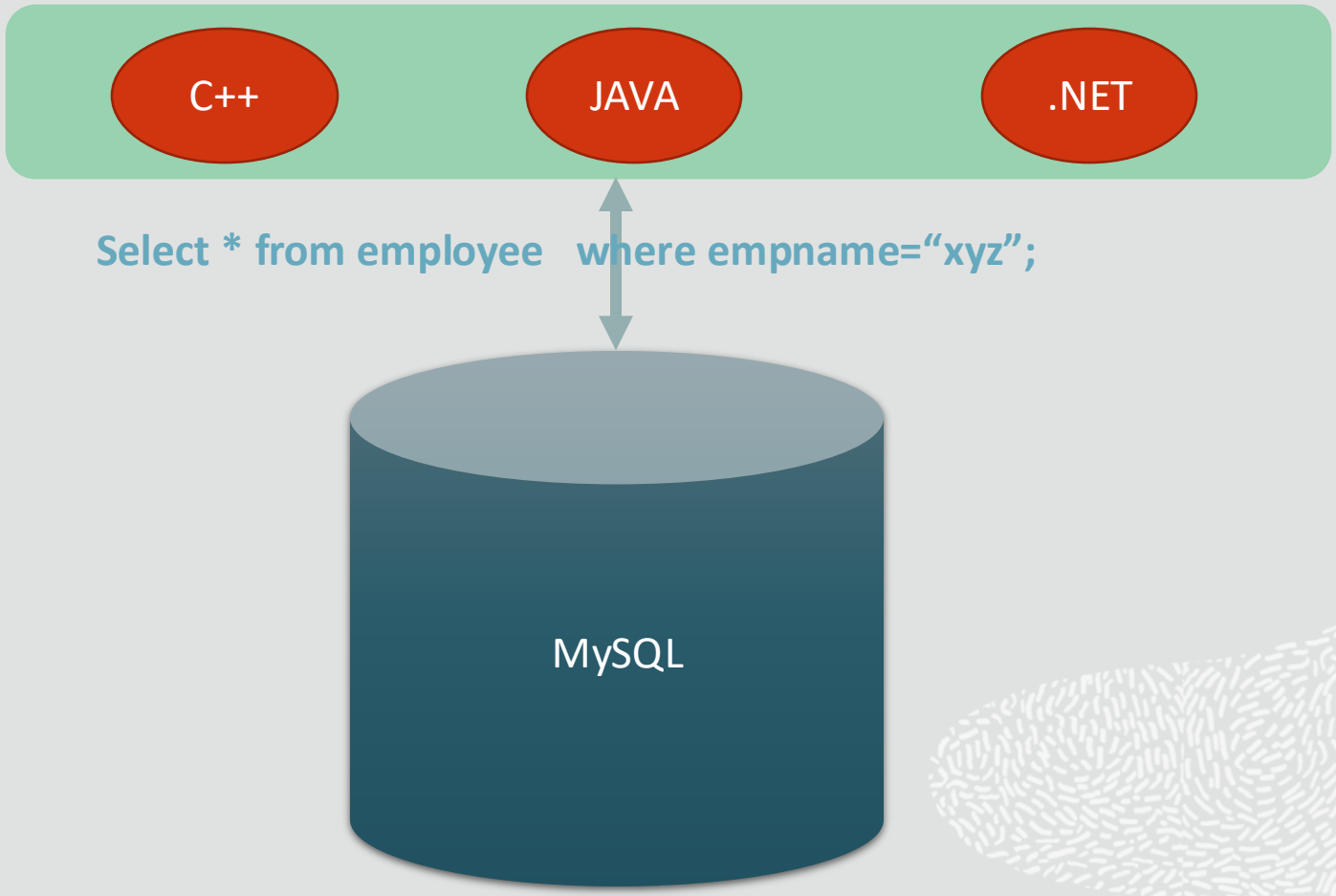
- Monitoring
- Backup
- Development
- Administration
- Migration

Support

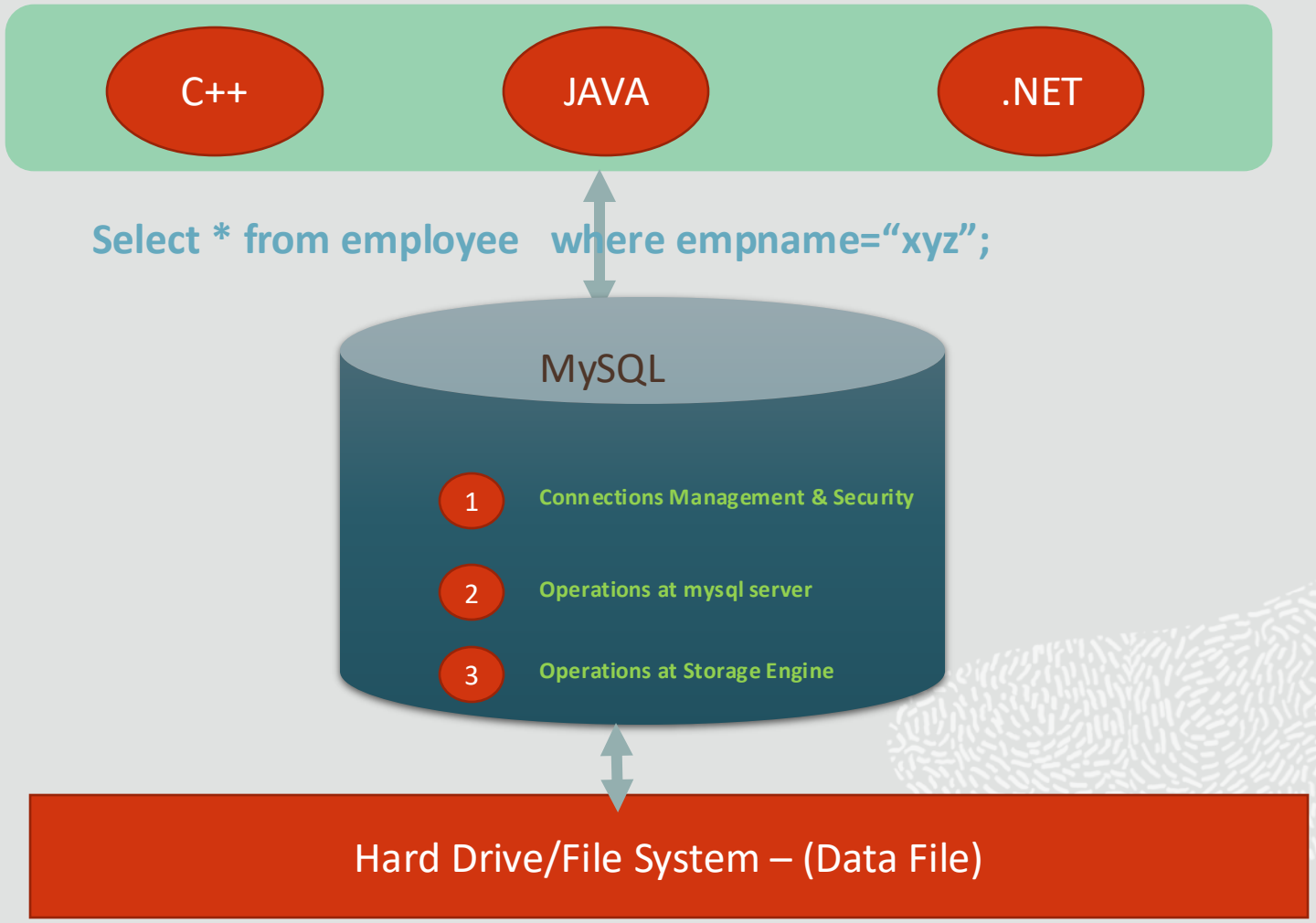
- Technical Support
- Consultative Support
- Oracle Certifications

MySQL Architecture Explain

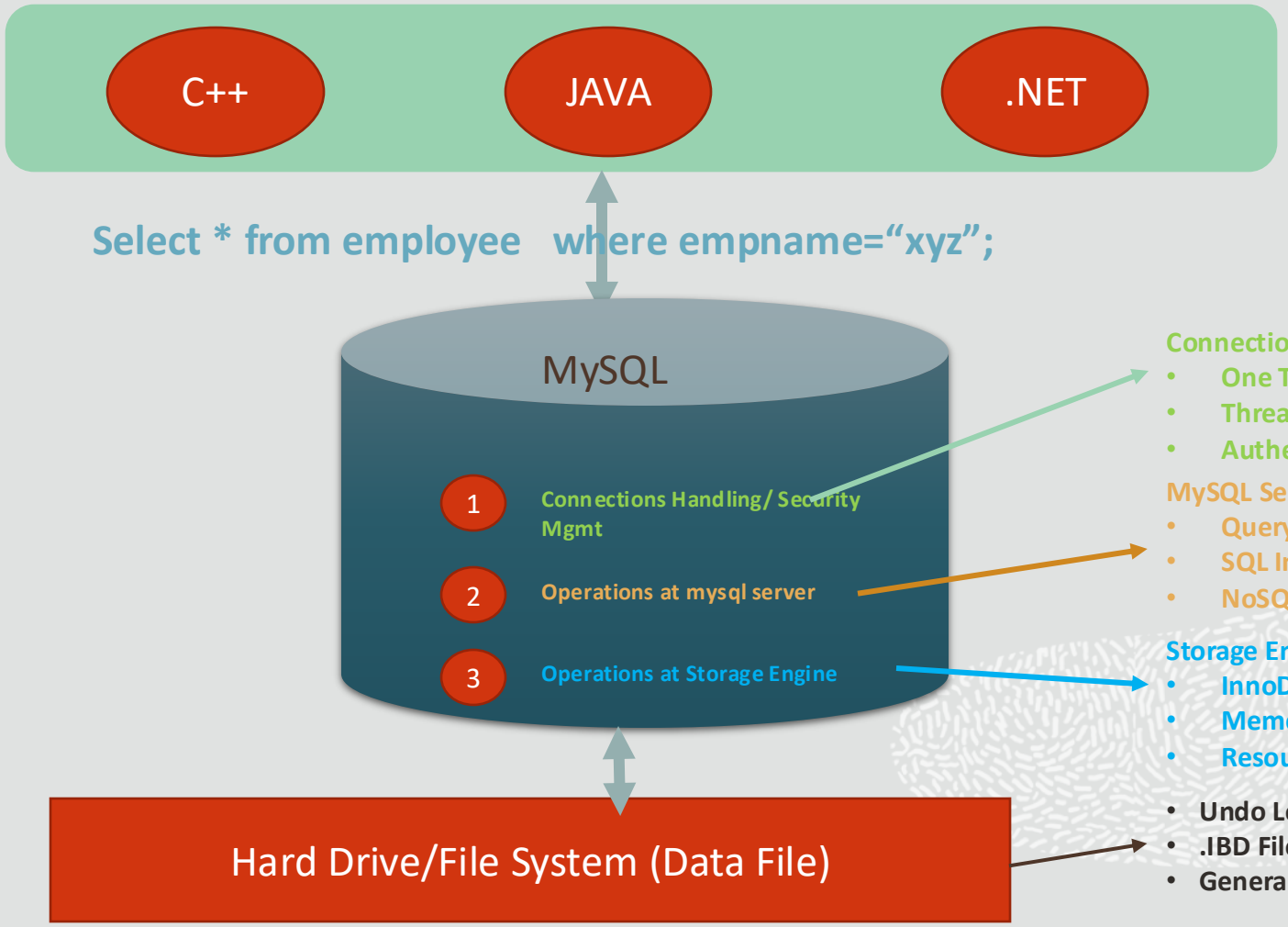
MySQL Architecture Explain



MySQL Architecture Explain



MySQL Architecture Explain



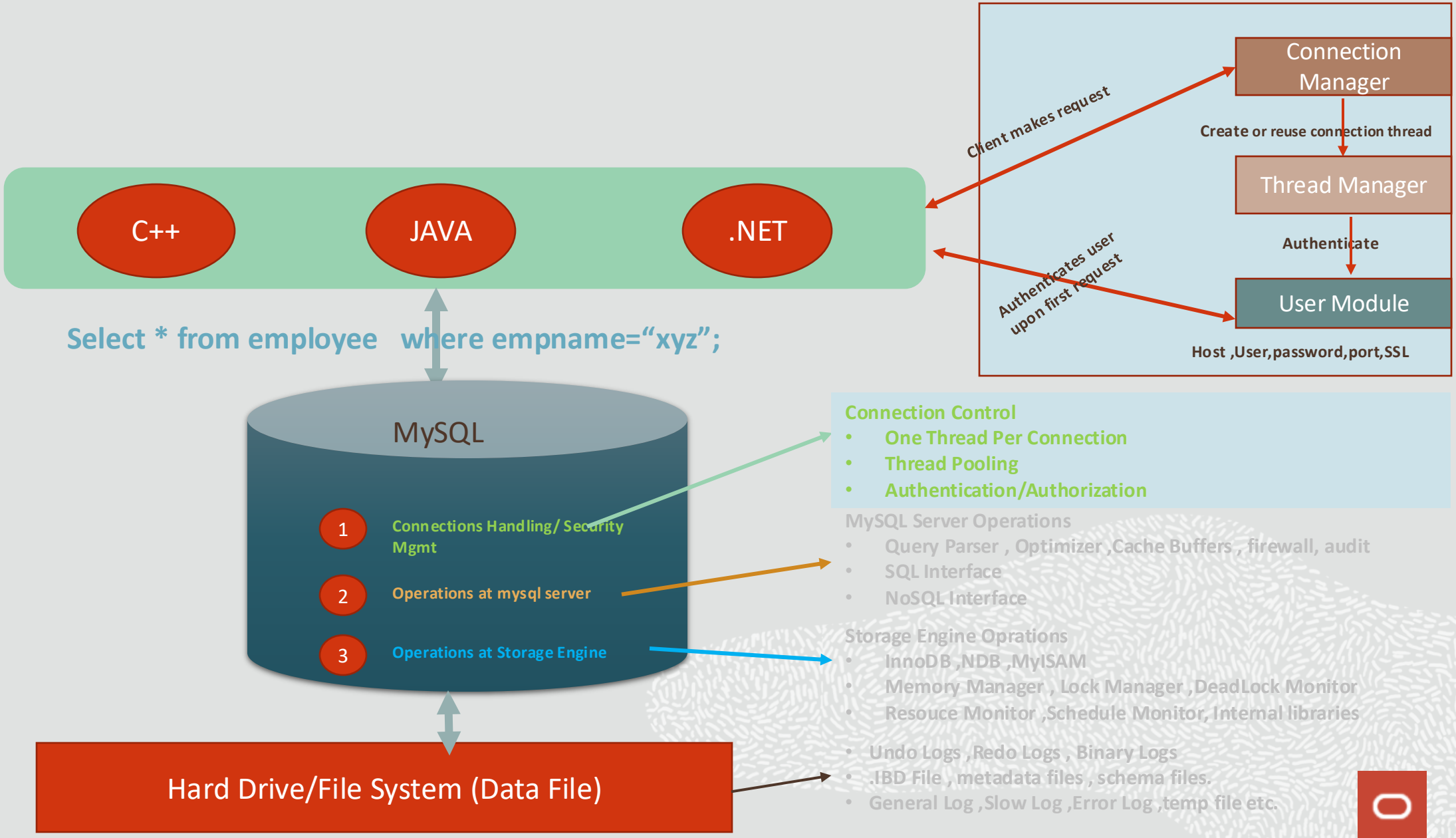
Connection Control
- One Thread Per Connection
- Thread Pooling
- Authentication/Authorization

MySQL Server Operations
- Query Parser , Optimizer ,Cache Buffers , firewall, audit
- SQL Interface
- NoSQL Interface

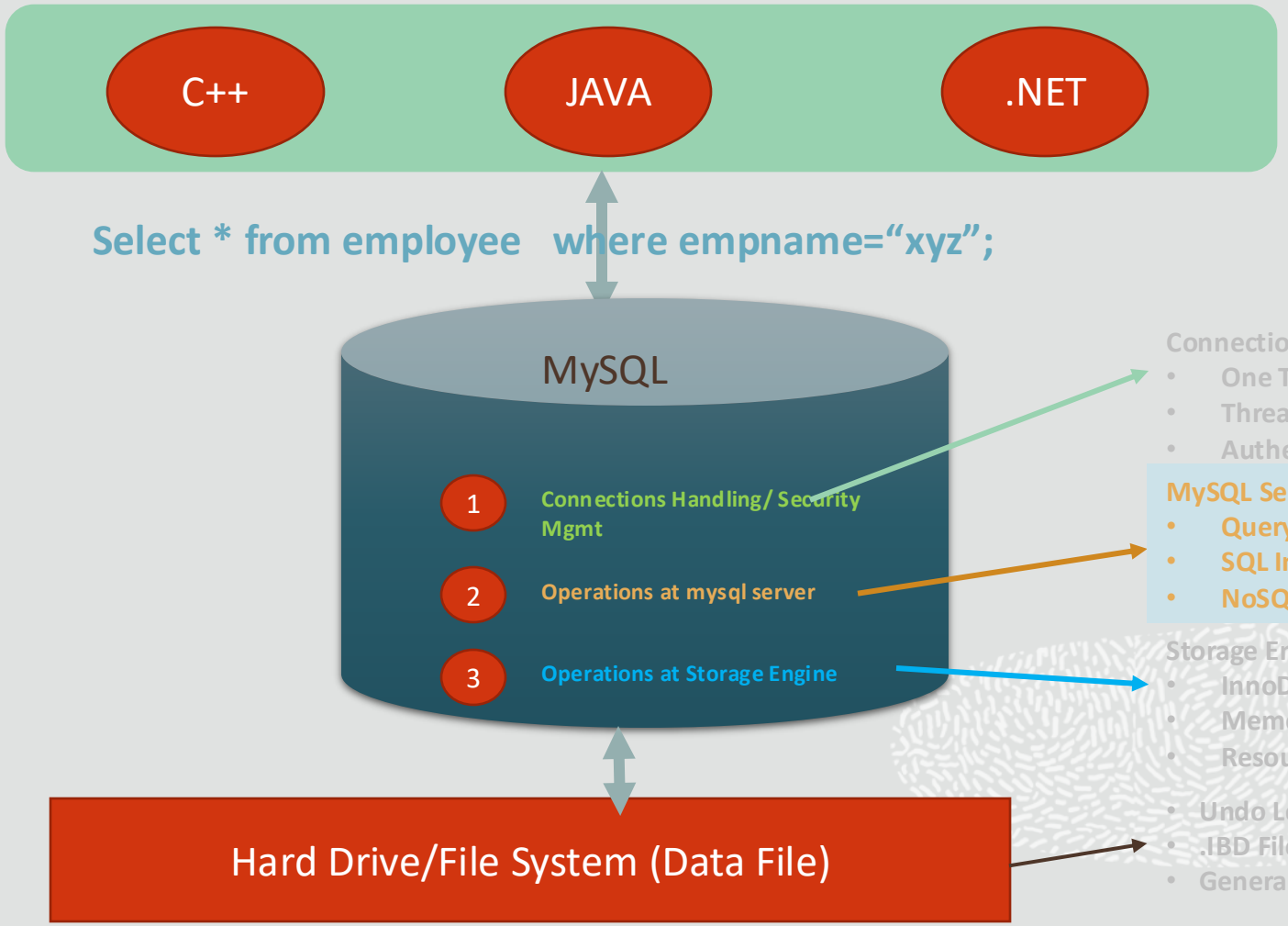
Storage Engine Oprations
- InnoDB ,NDB ,MyISAM
- Memory Manager , Lock Manager ,DeadLock Monitor
- Resouce Monitor ,Schedule Monitor, Internal libraries

- Undo Logs ,Redo Logs , Binary Logs
- .IBD File , metadata files , schema files.
- General Log ,Slow Log ,Error Log ,temp file etc.

MySQL Architecture Explain : Authentication/Authorization



MySQL Architecture Explain : Query Processing Steps



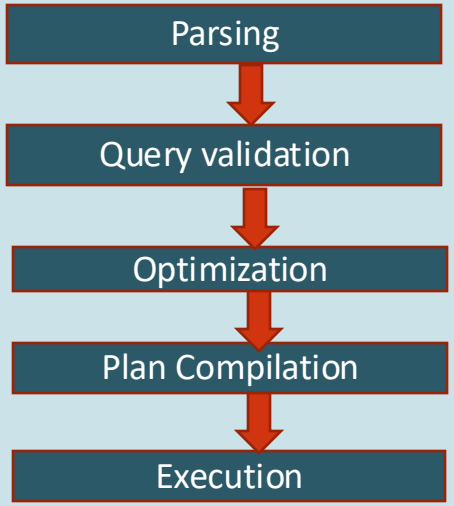
Connection Control
- One Thread Per Connection
- Thread Pooling
- Authentication/Authorization

MySQL Server Operations
- Query Parser , Optimizer ,Cache Buffers , firewall, audit
- SQL Interface
- NoSQL Interface

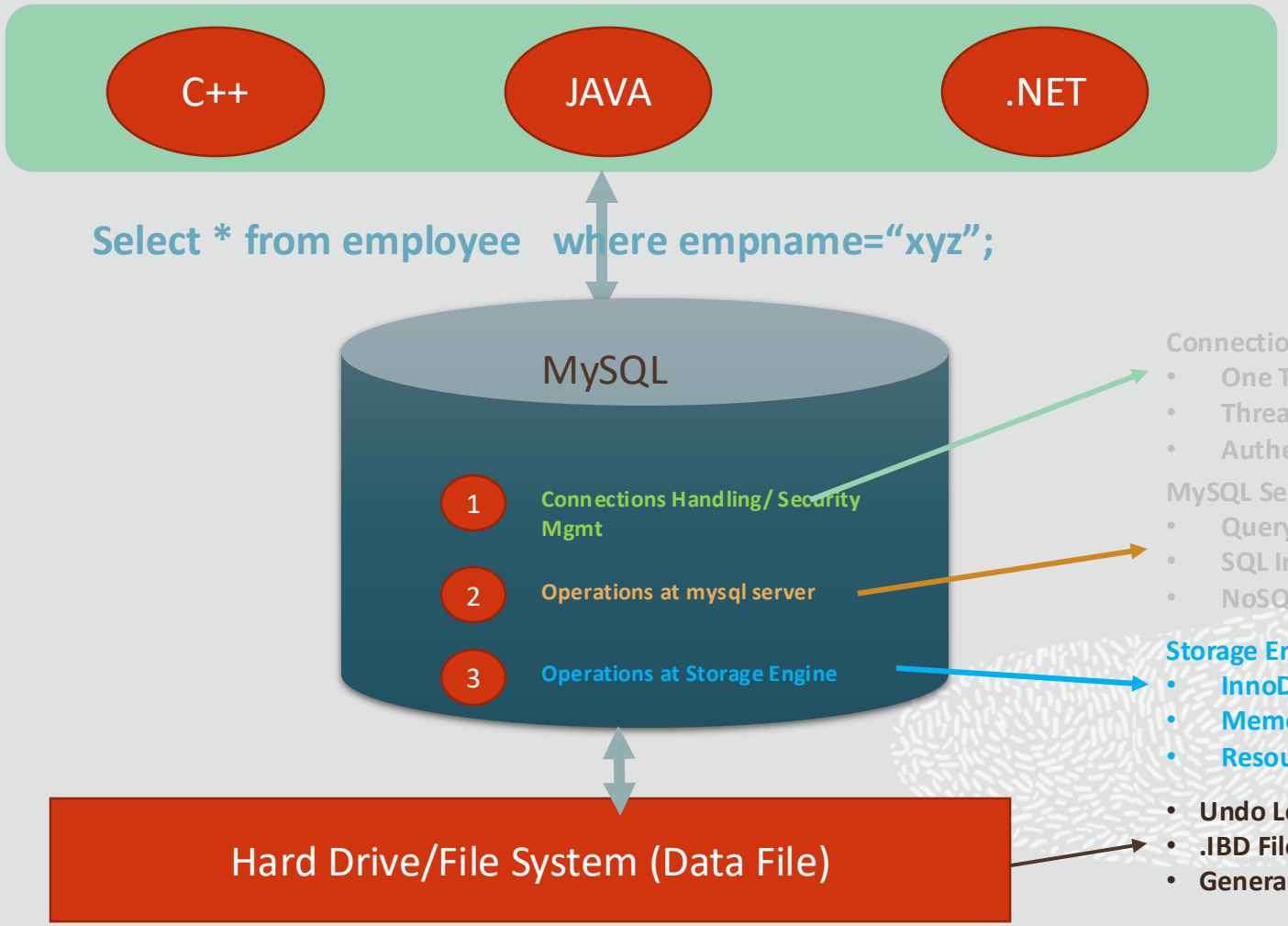
Storage Engine Oprations
- InnoDB ,NDB ,MyISAM
- Memory Manager , Lock Manager ,DeadLock Monitor
- Resouce Monitor ,Schedule Monitor, Internal libraries

- Undo Logs ,Redo Logs , Binary Logs
- .IBD File , metadata files , schema files.
- General Log ,Slow Log ,Error Log ,temp file etc.

Query processing steps



MySQL Architecture Explain : Storage Engine



InnoDB

MyISAM

NDB

Memory

Federated

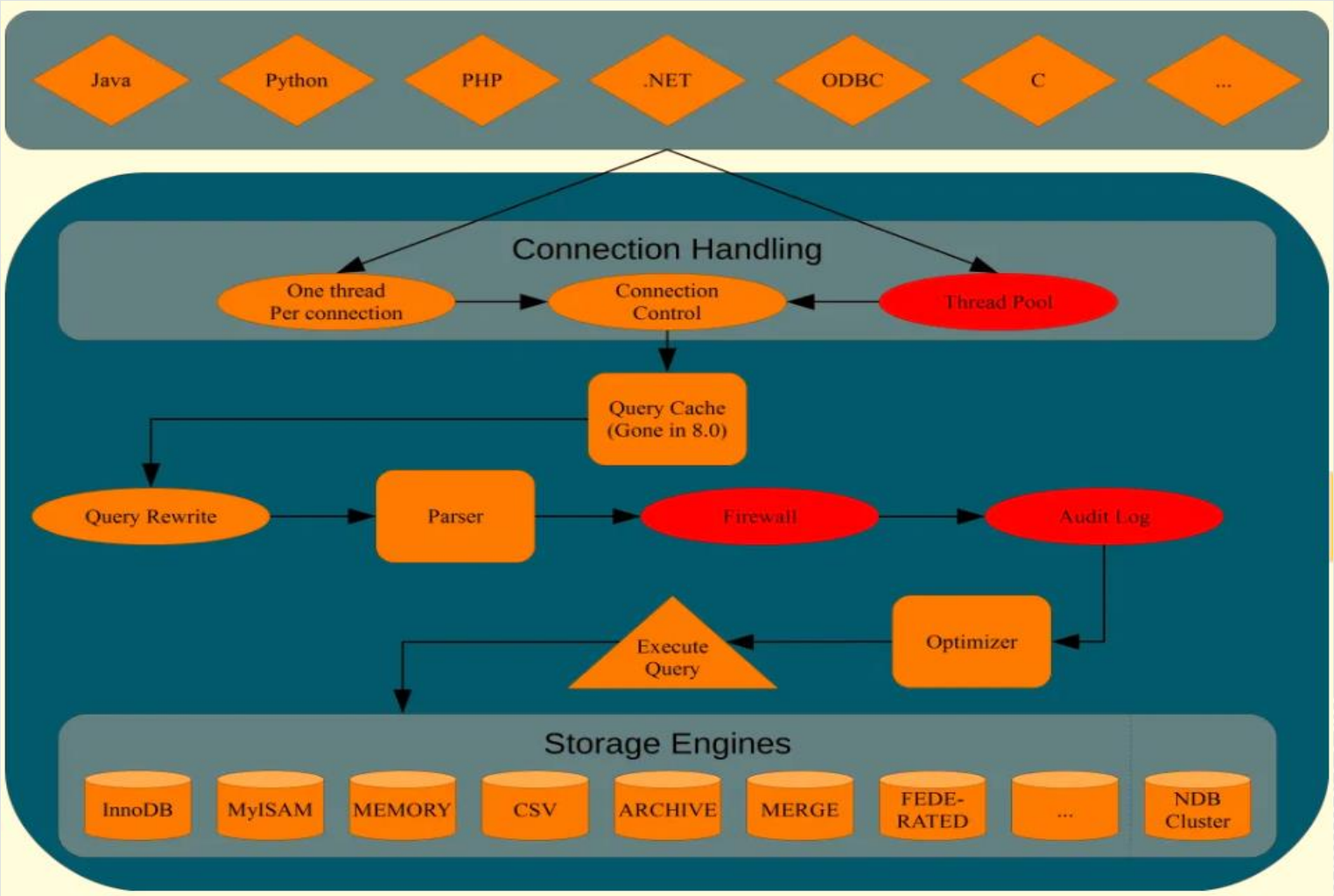
Connection Control
- One Thread Per Connection
- Thread Pooling
- Authentication/Authorization

MySQL Server Operations
- Query Parser , Optimizer ,Cache Buffers , firewall, audit
- SQL Interface
- NoSQL Interface

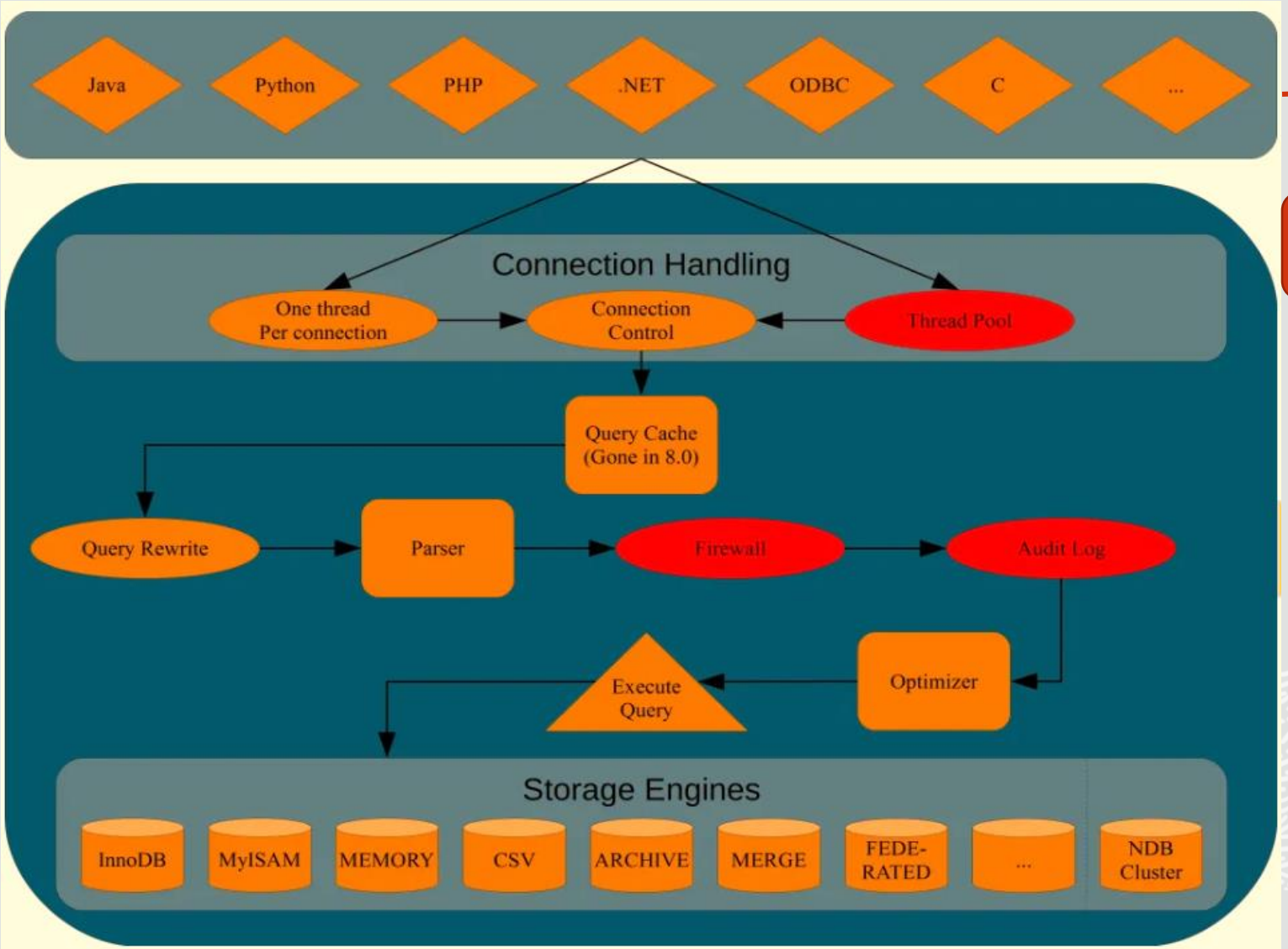
Storage Engine Oprations
- InnoDB ,NDB ,MyISAM
- Memory Manager , Lock Manager ,DeadLock Monitor
- Resouce Monitor ,Schedule Monitor, Internal libraries

- Undo Logs ,Redo Logs , Binary Logs
- .IBD File , metadata files , schema files.
- General Log ,Slow Log ,Error Log ,temp file etc.

MySQL Architecture Explain



MySQL Architecture Explain



First Layer/Top Layer

Connection Management
& Authentication/ Autherization

Parser , Firewall,Audit,Optimizer,Execution of query

InnoDB Storage Engine ~300 system

File System

MySQL Architecture Explain : Files & Database/I_S/P_S/sys schema

Data data directory
 # cd /var/lib/mysql

App

Insert into employee values (1);

Log file
-General log >> data directory >> Run Time
-Slow Query log >> data directory >> Run Time
-Error Log >> data directory
-Binary log

Schema=database in MySQL

-mysql
-information_schema(I_S) >> Global
-Performance_schema(P_S) >>Global
-Sys schema >> Global

User schema(employee ,sales,ddept)

Pages ~ 16 KB
----table1.ibd
----table2.ibd

Employee.ibd file

Don't delete manually

REDO LOG
- Recovery of uncomttied trx during failure
- .iblog01
- iblog02

MySQL

1 Connections Handling/ Security Mgmt

2 Operations at mysql server

3 Operations at Storage Engine

UNDO LOG
Undolog-01
Undolog-02
-rollback transaction

System table space(ibdata1) >global tablespace.ibd(pointer to the tables(metadata)

Binary Log (hold table data- Insert/update/delete)

mysql>show engine innodb status;

Select * from Performance_schema.event_stmt_history

Insformation_schema

INFORMATION_SCHEMA provides access to database metadata, information about the MySQL server such as the name of a database or table, the data type of a column, or access privileges.

```
mysql> SELECT table_name, table_type, engine
       FROM information_schema.tables
       WHERE table_schema = 'db5'
       ORDER BY table_name;
+------------+------------+--------+
| table_name | table_type | engine |
+------------+------------+--------+
| fk         | BASE TABLE | InnoDB |
| fk2        | BASE TABLE | InnoDB |
| goto       | BASE TABLE | MyISAM |
| into       | BASE TABLE | MyISAM |
| k          | BASE TABLE | MyISAM |
| kurs       | BASE TABLE | MyISAM |
| loop       | BASE TABLE | MyISAM |
| pk         | BASE TABLE | InnoDB |
| t          | BASE TABLE | MyISAM |
| t2         | BASE TABLE | MyISAM |
| t3         | BASE TABLE | MyISAM |
| t7         | BASE TABLE | MyISAM |
| tables     | BASE TABLE | MyISAM |
```

Pefromance_schema

The MySQL Performance Schema is a feature for monitoring MySQL Server execution at a low level.

```
mysql> SELECT * FROM performance_schema.setup_consumers;
+----------------------------------+---------+
| NAME                             | ENABLED |
+----------------------------------+---------+
| events_stages_current            | NO      |
| events_stages_history            | NO      |
| events_stages_history_long       | NO      |
| events_statements_current        | YES     |
| events_statements_history        | YES     |
| events_statements_history_long   | NO      |
| events_transactions_current      | YES     |
| events_transactions_history      | YES     |
| events_transactions_history_long | NO      |
| events_waits_current             | NO      |
| events_waits_history             | NO      |
| events_waits_history_long        | NO      |
| global_instrumentation           | YES     |
| thread_instrumentation           | YES     |
| statements_digest                | YES     |
+----------------------------------+---------+
```

```
mysql> SELECT EVENT_ID, TRUNCATE(TIMER_WAIT/1000000000000,6) as Duration, SQL_TEXT
       FROM performance_schema.events_statements_history_long WHERE SQL_TEXT like '%10001%';
+----------+----------+-----------------------------------------------------+
| event_id | duration | sql_text                                            |
+----------+----------+-----------------------------------------------------+
|       31 | 0.028310 | SELECT * FROM employees.employees WHERE emp_no = 10001 |
+----------+----------+-----------------------------------------------------+
```

Sys Schema

Sys schema objects can be used for typical tuning and diagnostics use cases.

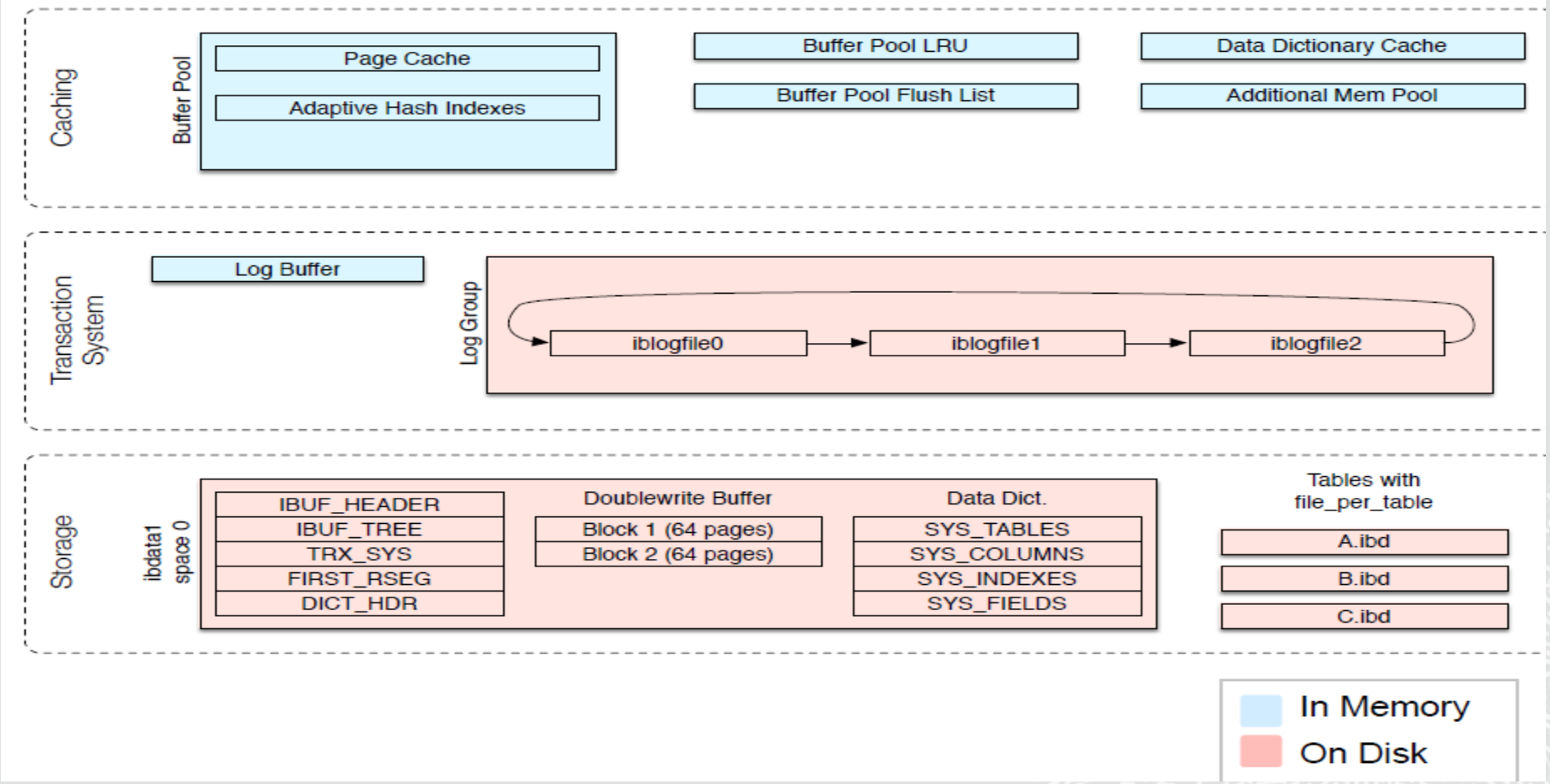
Sys schema, a set of objects that helps DBAs and developers interpret data collected by performance_schema.

```
mysql> SELECT * FROM sys.host_summary_by_file_io;
+------------+-------+------------+
| host       | ios   | io_latency |
+------------+-------+------------+
| localhost  | 67570 | 5.38 s     |
| background |  3468 | 4.18 s     |
+------------+-------+------------+
```

Break for 10 Minute: 12:20PM

Copyright © 2020 Oracle and/or its affiliates.

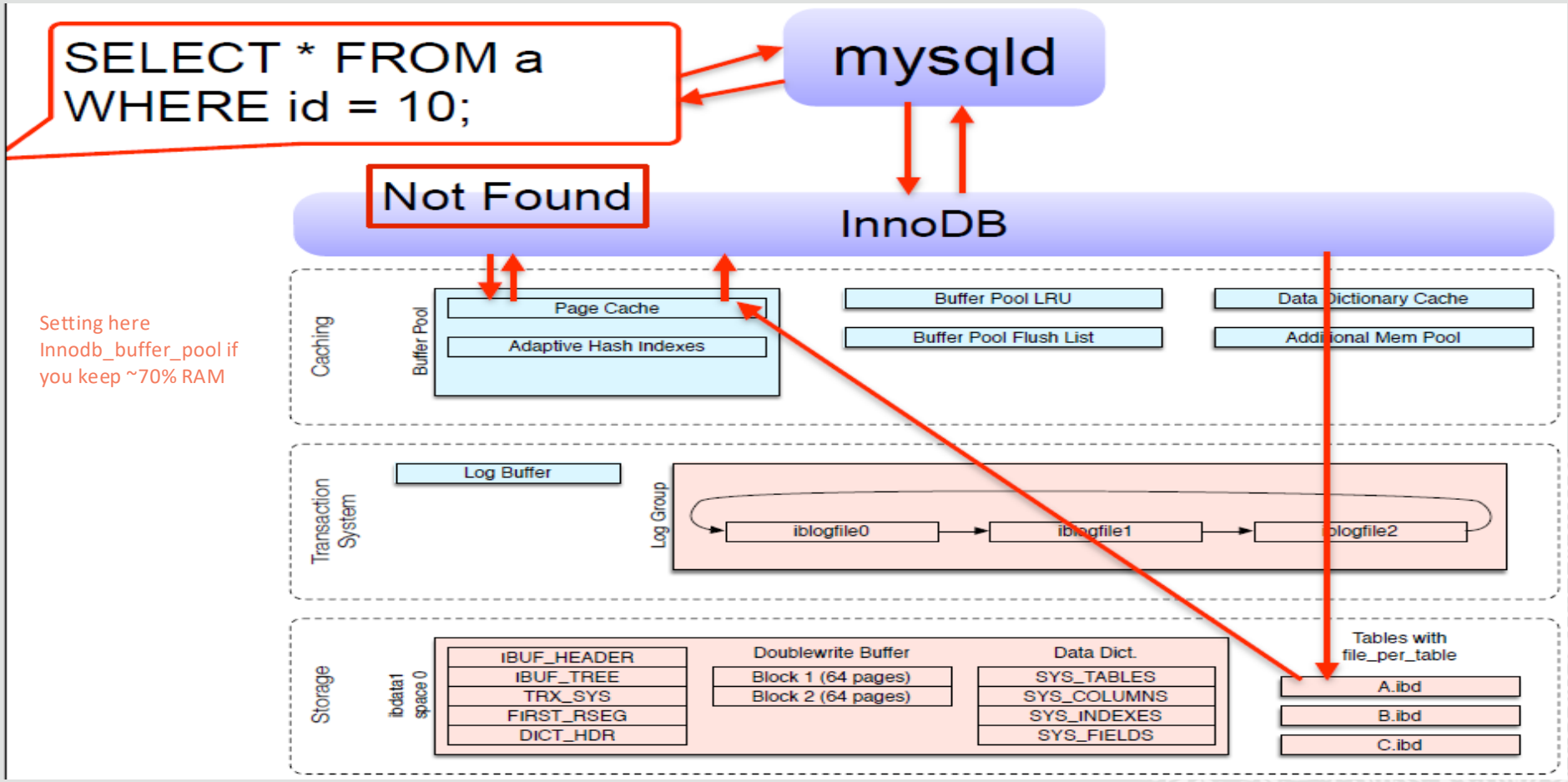
MySQL InnoDB Architecture



https://www.slideshare.net/morgo/inno-db-presentation?from_action=save

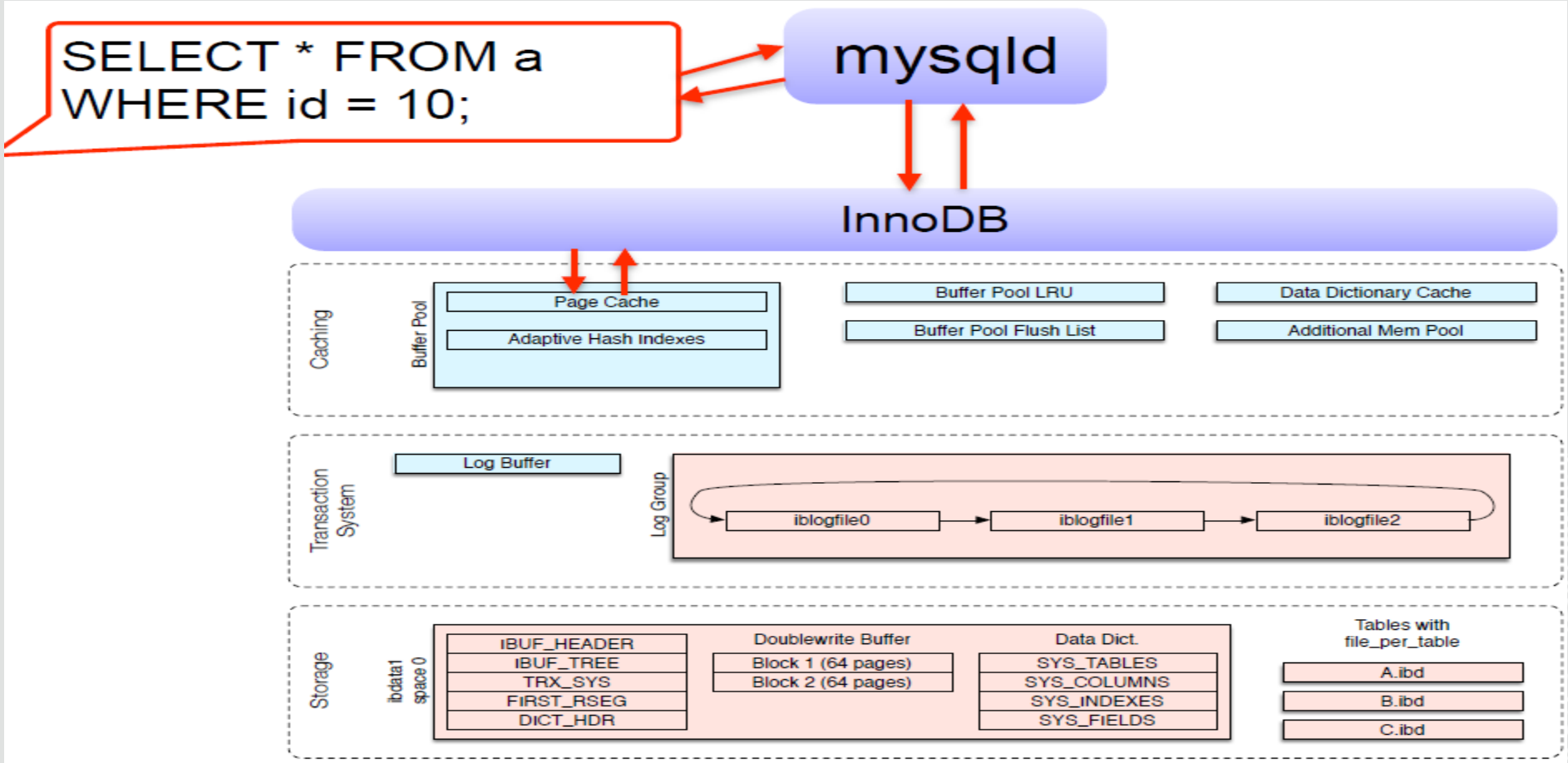
MySQL InnoDB Architecture

Pages not in the buffer pool

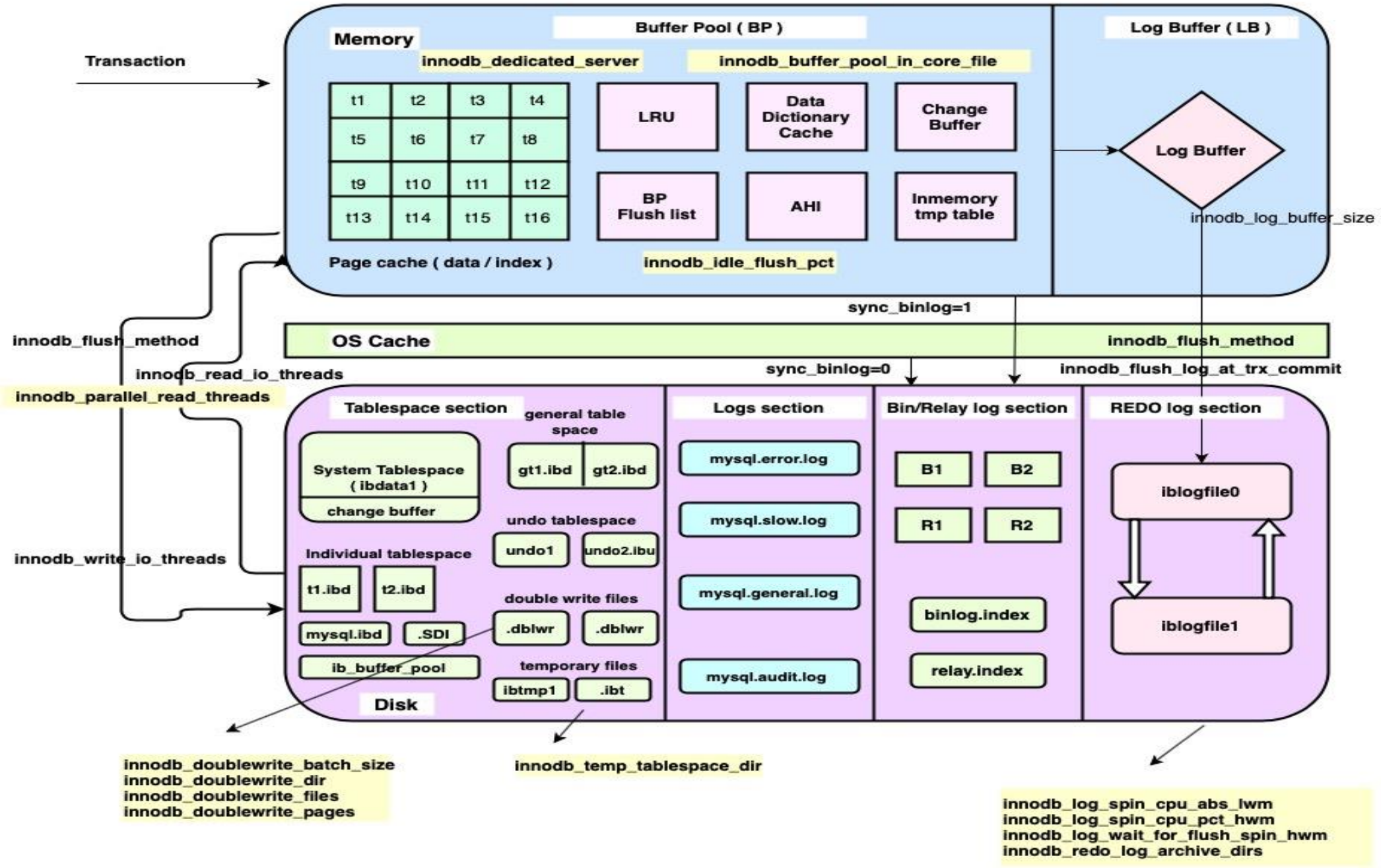


MySQL InnoDB Architecture

Pages in the buffer pool

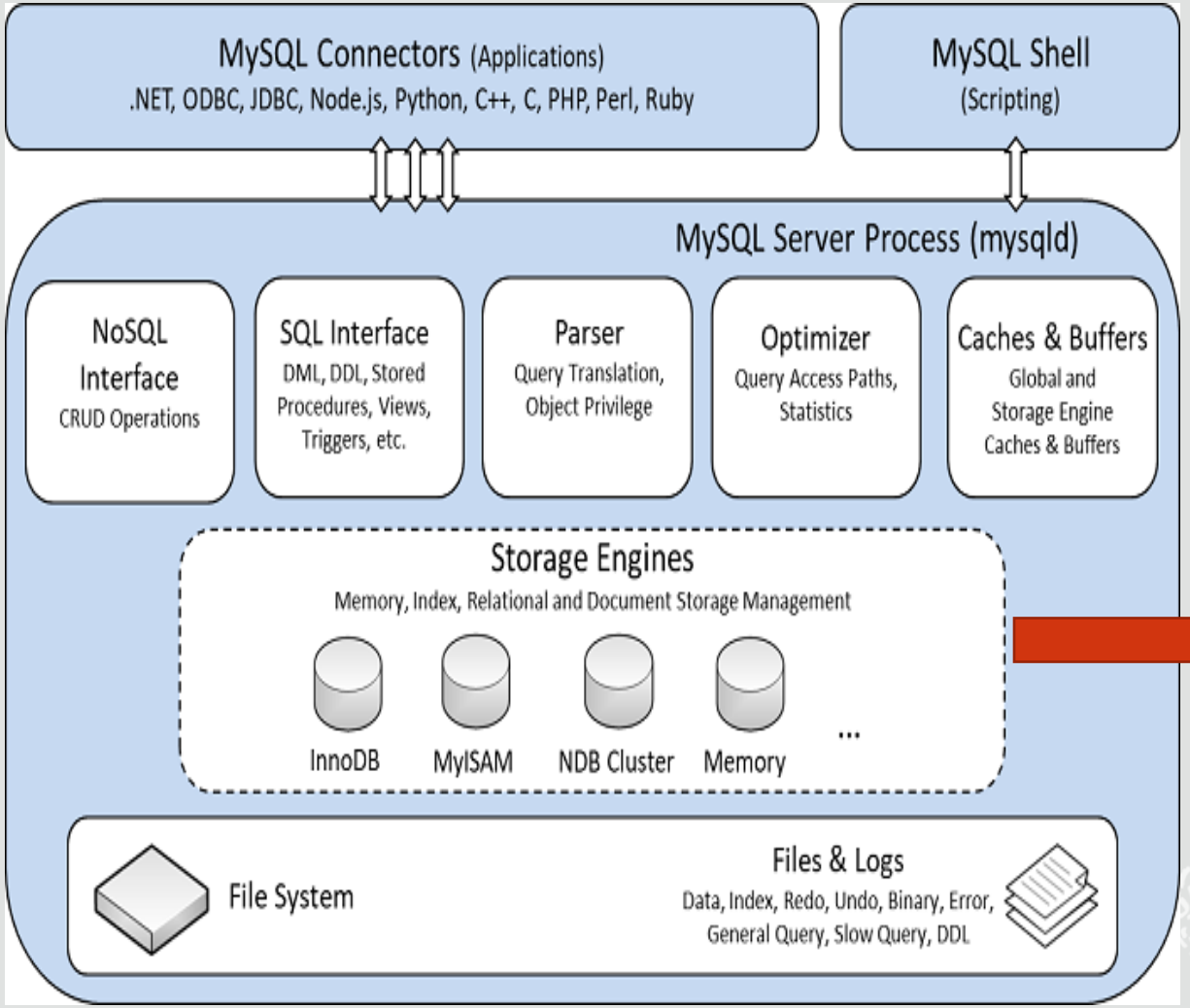


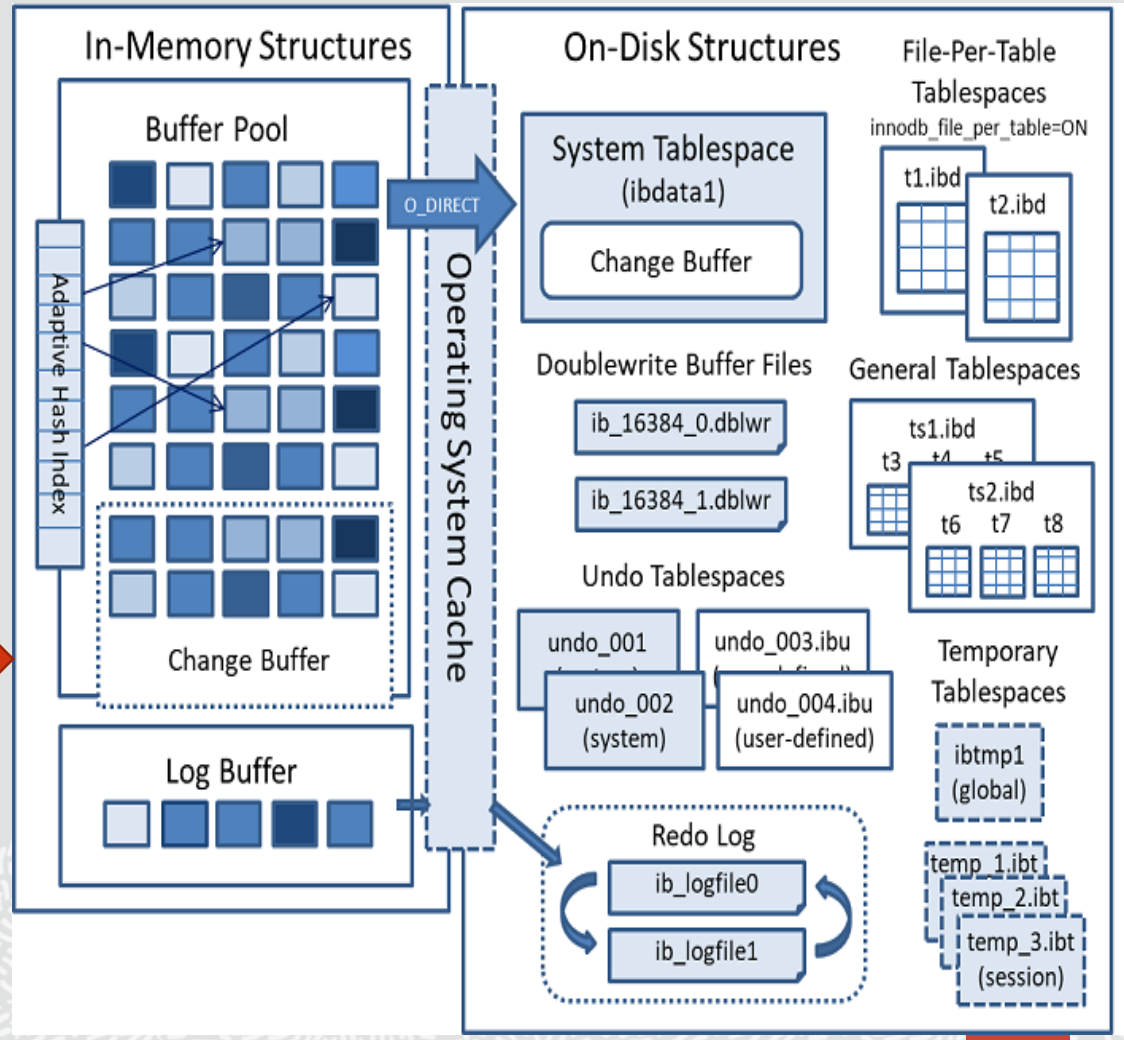
MySQL 8 InnoDB Architecture



MySQL 8.0 Architecture

References:https://dev.mysql.com/doc/refman/8.0/en/innodb-architecture.html





Copyright © 2020 Oracle and/or its affiliates.

MySQL Document Store

SQL + NoSQL = MySQL 8.0

Copyright © 2020 Oracle and/or its affiliates.

Document Oriented Databases

What is a Document in MySQL?

Document Oriented Databases

What is a Document?

A data structure that can represent complex information, similar to an Object

Structure of the data is part of the document, no uniform structure

JSON (=JavaScript Object Notation)

- Compact, popular and standardized
- Can be represented natively in many languages (JavaScript, Python etc)

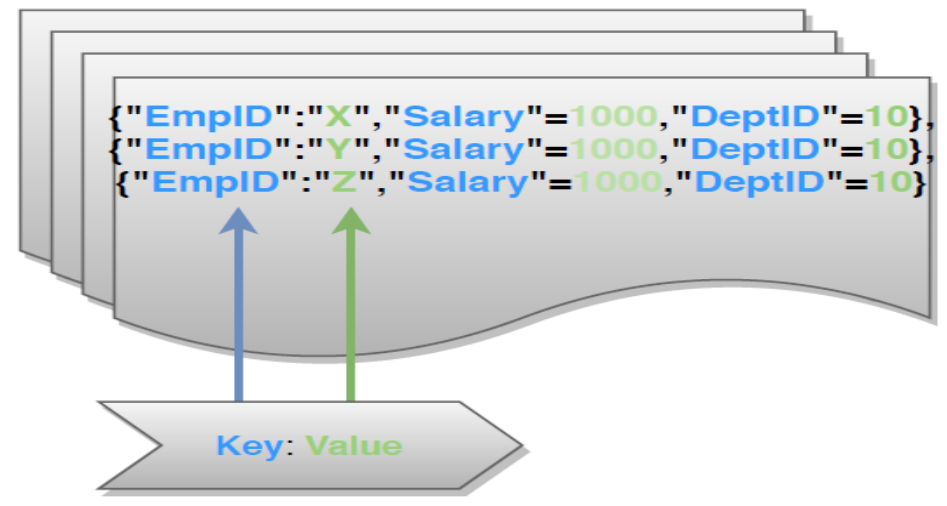
Other popular encoding formats are XML, YAML etc

JSON Document Example

```
{
    "_id": "IND",
    "Name": "India",
    "GNP": 211860,
    "IndepYear": 1947,
    "demographics": {
      "LifeExpectancy": 77.699,
      "Population": 1013662000
    },
    "geography": {
      "Continent": "Asia",
      "Region": "South & Central Asia",
      "SurfaceArea": 83859
    }
}
```

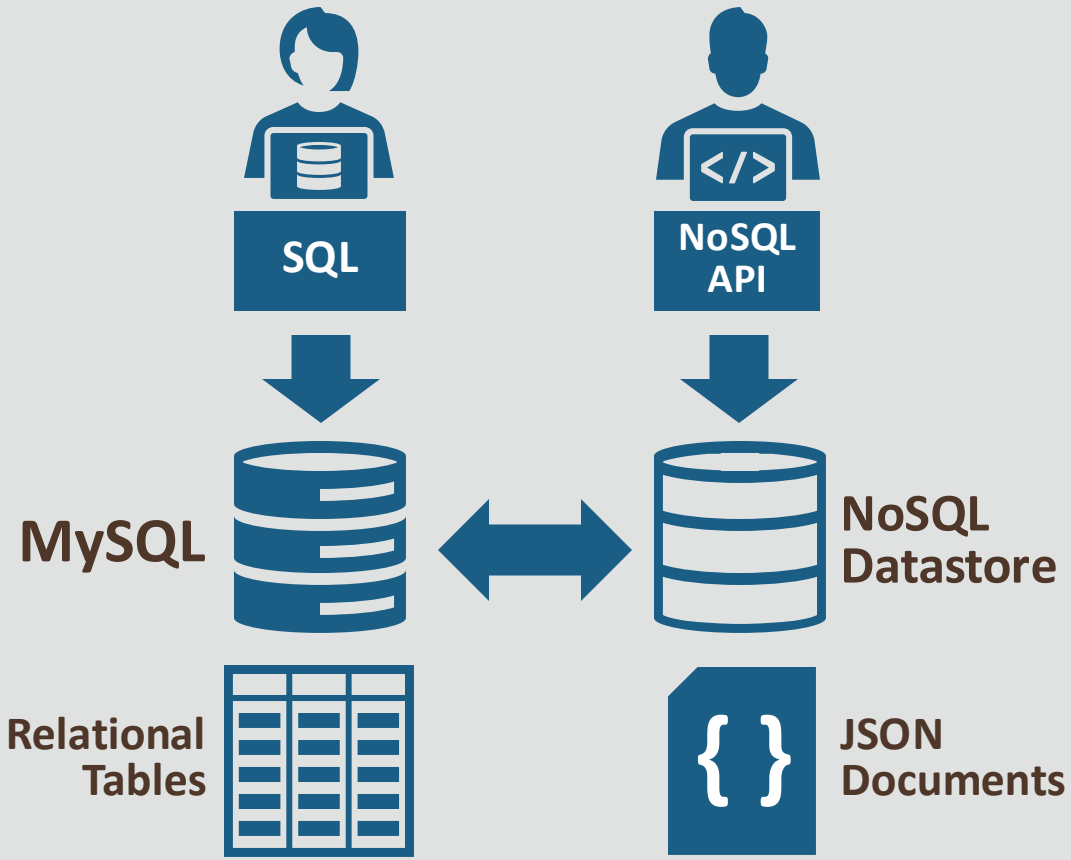
Why Json?

- JSON is very simple key - value format , Universal data structures
- More lightweight format to transport and process
- Based on subset of JavaScript Programming Language
- Support evolving in other languages.



- Combine MySQL's relational capabilities with JSON and you get best of both traditional SQL and NoSQL worlds. (SQL + NoSQL)

Pains running RDBMS plus NoSQL Datastore

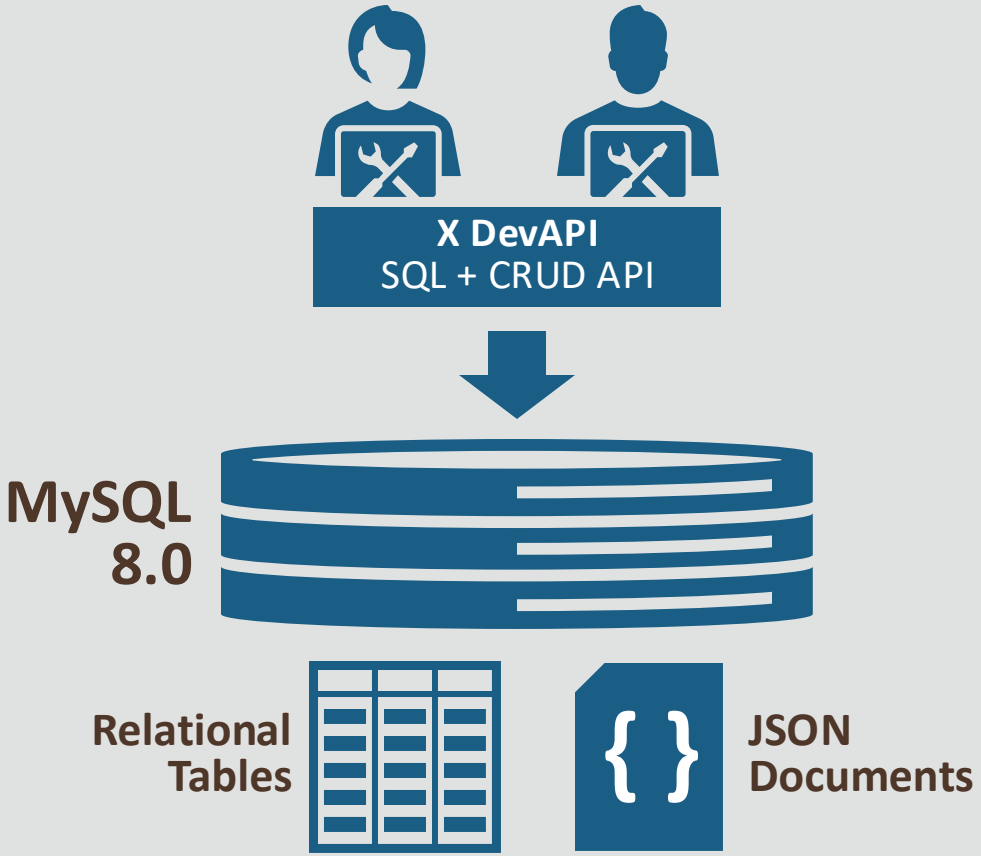


Developers
Required to learn multiple APIs

Data management
Difficult to keep data synchronization between tables and JSON documents

Operations
Required to manage multiple products with different tools

MySQL Document Store: SQL + NoSQL = MySQL

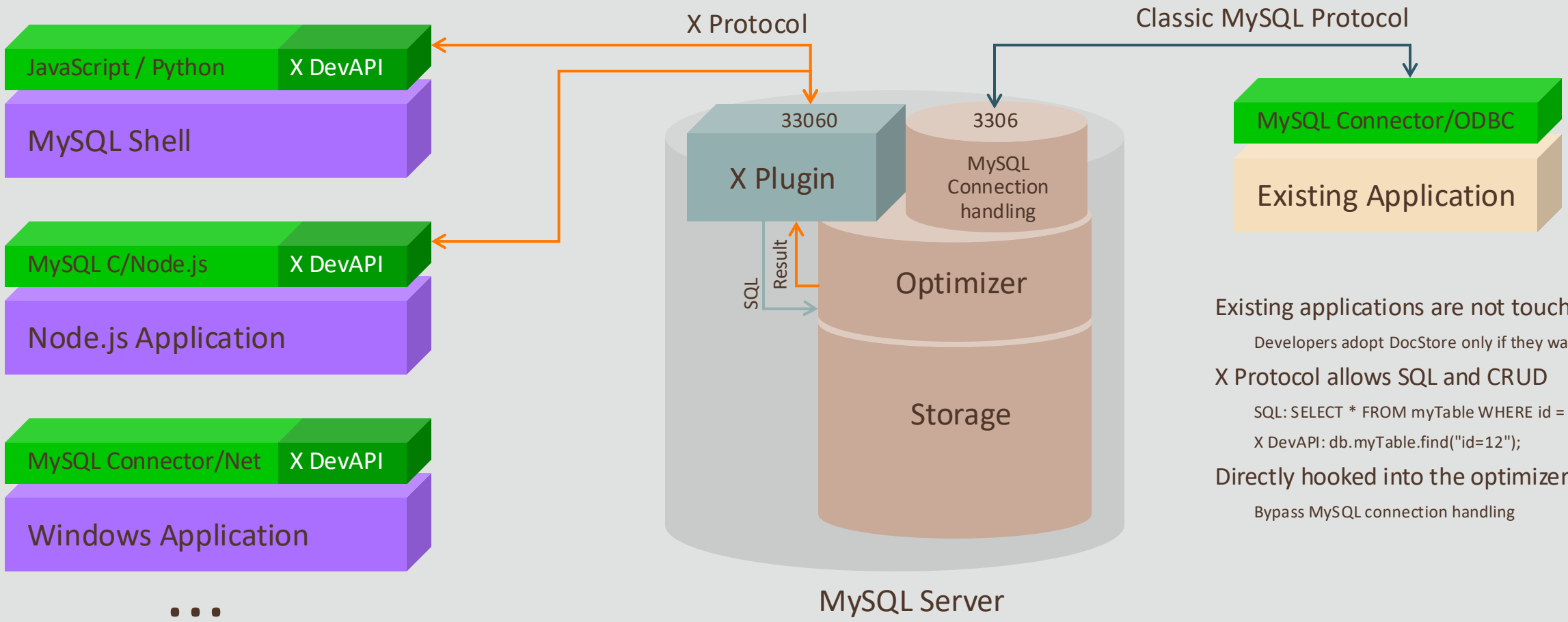


Flexibility for Developers
Unified API provides more flexibility

Reliable data management
Data is always in sync and you can “JOIN” tables and documents

Simplified Operations
Managing single database with unified management tool

MySQL Document Store – How it works



Existing applications are not touched

- Developers adopt DocStore only if they want to

X Protocol allows SQL and CRUD

- SQL: SELECT * FROM myTable WHERE id = 12;
- X DevAPI: db.myTable.find("id=12");

Directly hooked into the optimizer

- Bypass MySQL connection handling

How to use MySQL as NoSQL ?

Schema

- mysqlsh
- shell.connect("root@10.0.10.13")
- CREATE SCHEMA DocumentDB
- \use DocumentDB

Collection

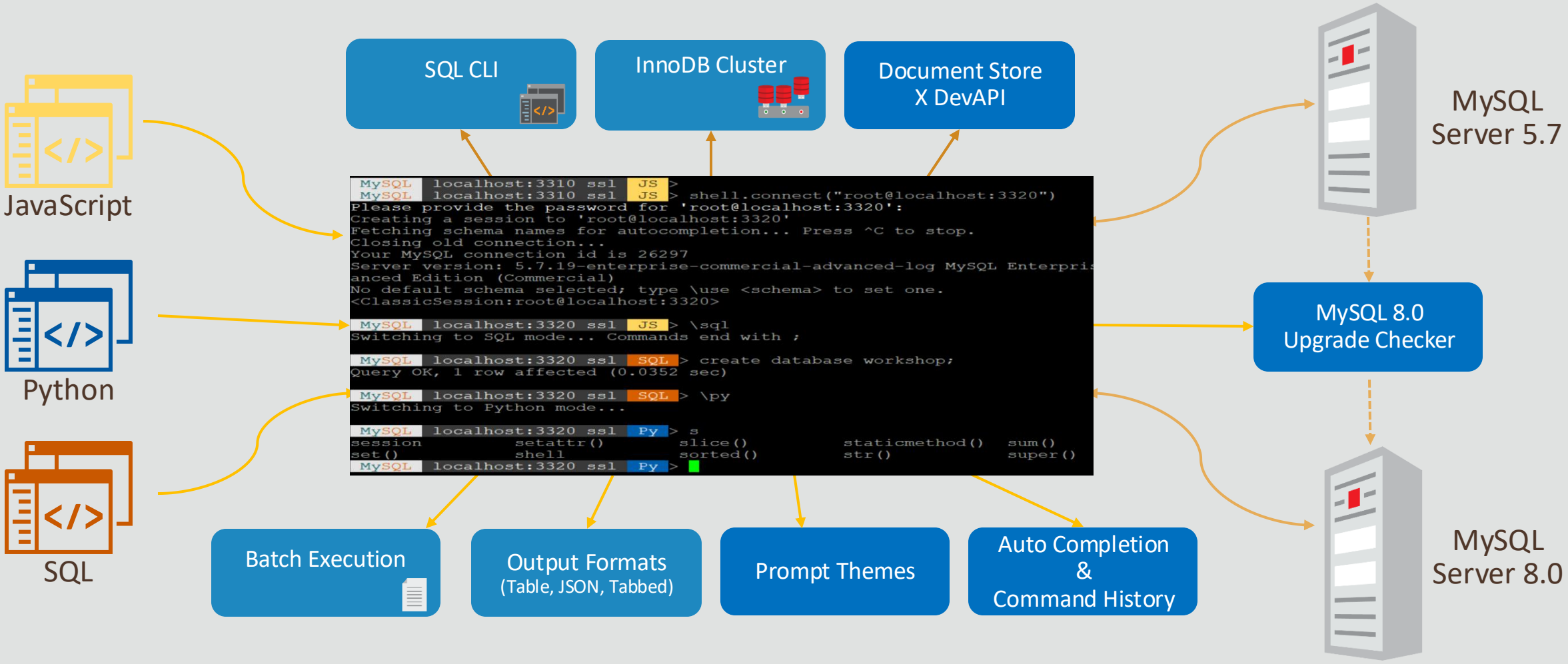
- db.createCollection("employee")
- db.getCollections()
- db.dropCollection("dept")

Document

- db.employee.add({"empname":"chandan","salary":1000,"dept":"HR"})
- db.employee.find()
- db.employee.find("empname='chandan'")
- db.employee.find(" empname='chandan' and dept='sales' ")
- db.employee.modify("empname='chandan'").set("empname","Radha")
- db.employee.remove("empname='Radha'")

Copyright © 2020 Oracle and/or its affiliates.

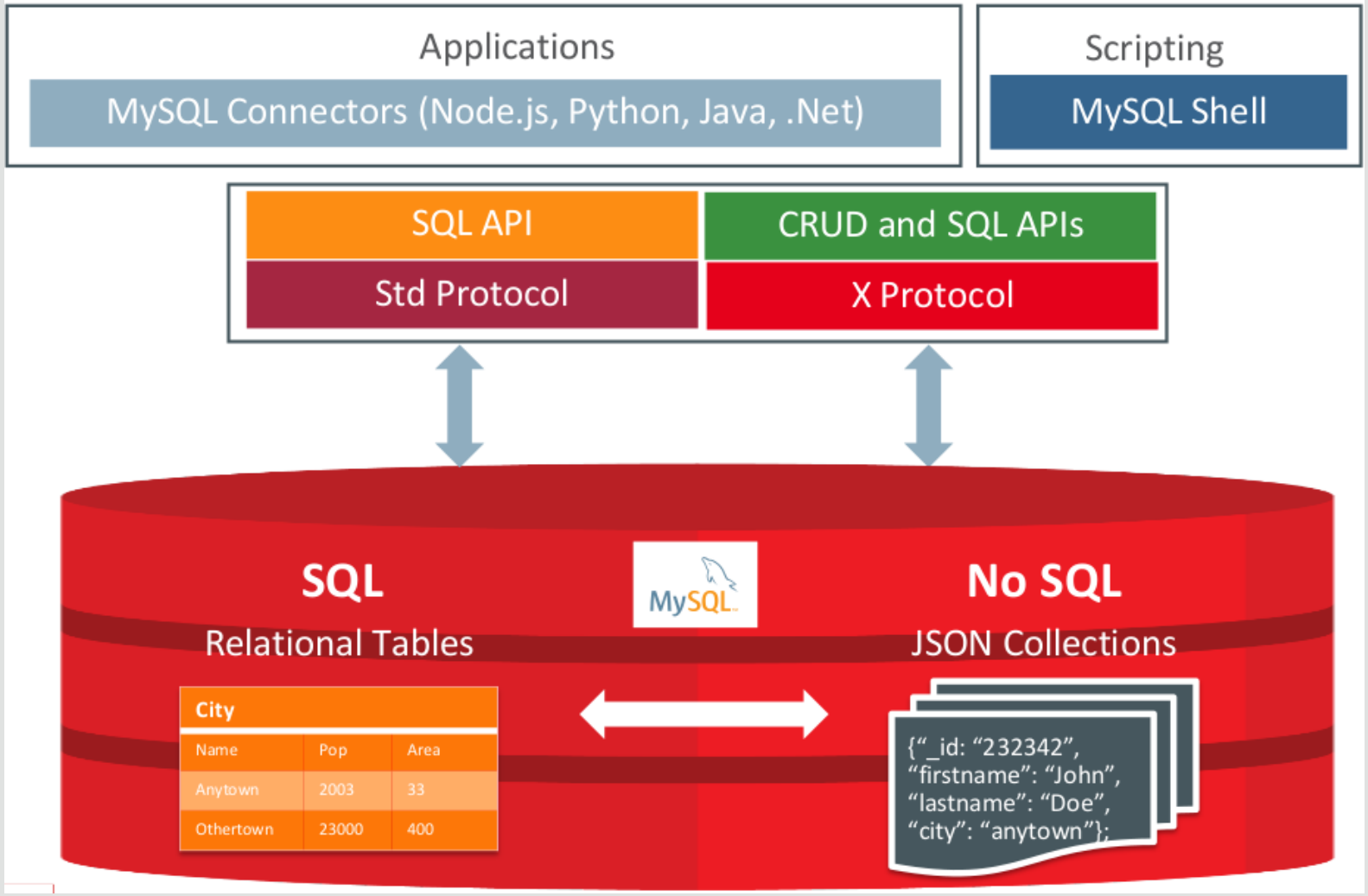
MySQL Shell 8.0



Lunch : back :2:30PM

Copyright © 2020 Oracle and/or its affiliates.

MySQL Document Store: Architecture

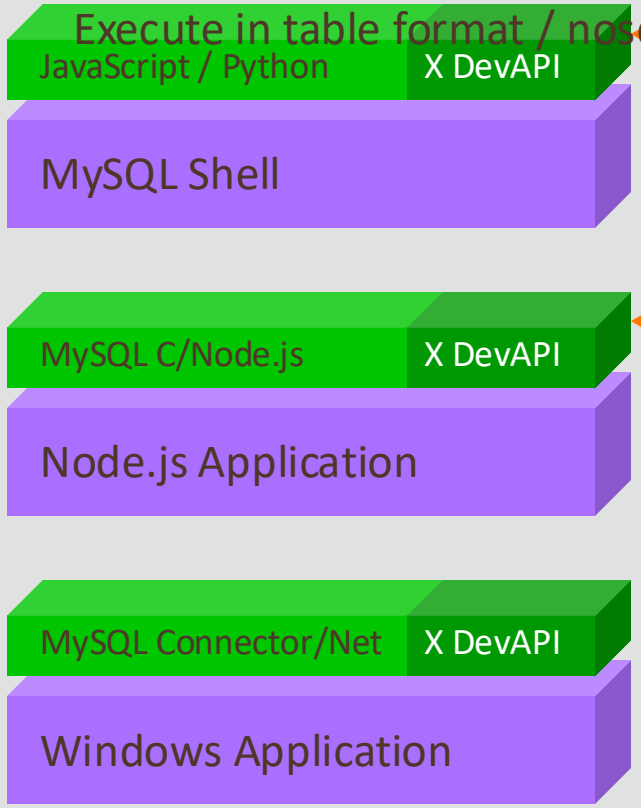


MySQL Document Store – How it works

Import mysqlx (table , json) XDEVAPI is treaty faster classic protocoal

Port 33060

Execute in table format / nosql



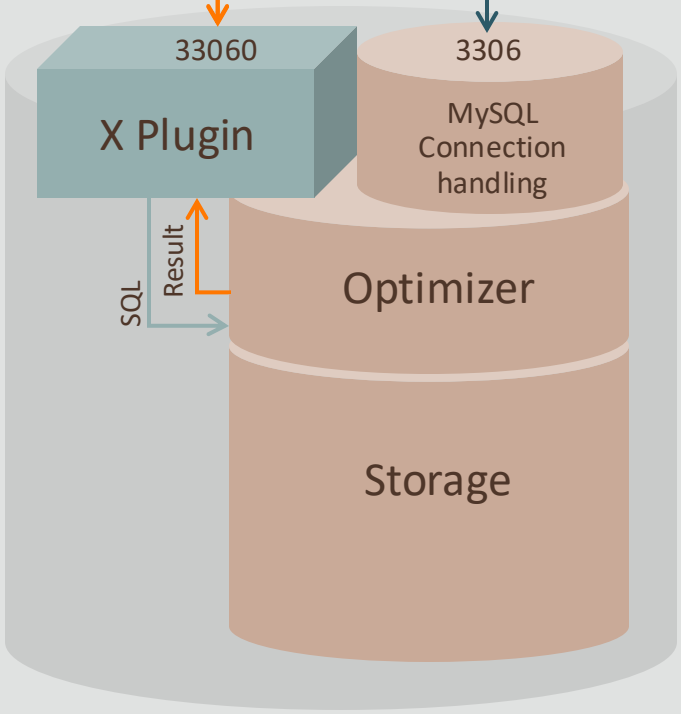
...

Classic 3306 port

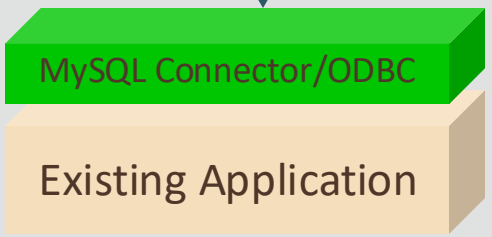
Import mysql

X Protocol

Classic MySQL Protocol



MySQL Server



Existing applications are not touched

- Developers adopt DocStore only if they want to

X Protocol allows SQL and CRUD

- SQL: SELECT * FROM myTable WHERE id = 12;
- X DevAPI: db.myTable.find("id=12");

Directly hooked into the optimizer

- Bypass MySQL connection handling

MySQL 8 - JSON Functions

JSON_ARRAY_APPEND()
JSON_ARRAY_INSERT()
JSON_ARRAY()
JSON_CONTAINS_PATH()
JSON_CONTAINS()
JSON_DEPTH()
JSON_EXTRACT()
JSON_INSERT()
JSON_KEYS()
JSON_LENGTH()

JSON_MERGE[_PRESERVE]()
JSON_OBJECT()
JSON_QUOTE()
JSON_REMOVE()
JSON_REPLACE()
JSON_SEARCH()
JSON_SET()
JSON_TYPE()
JSON_UNQUOTE()
JSON_VALID()

JSON_PRETTY()
JSON_STORAGE_SIZE()
JSON_STORAGE_FREE()
JSON_ARRAYAGG()
JSON_OBJECTAGG()
JSON_MERGE_PATCH()
JSON_TABLE()
JSON_OVERLAPS()
JSON Schema
JSON Array Indexes

Copyright © 2020 Oracle and/or its affiliates.

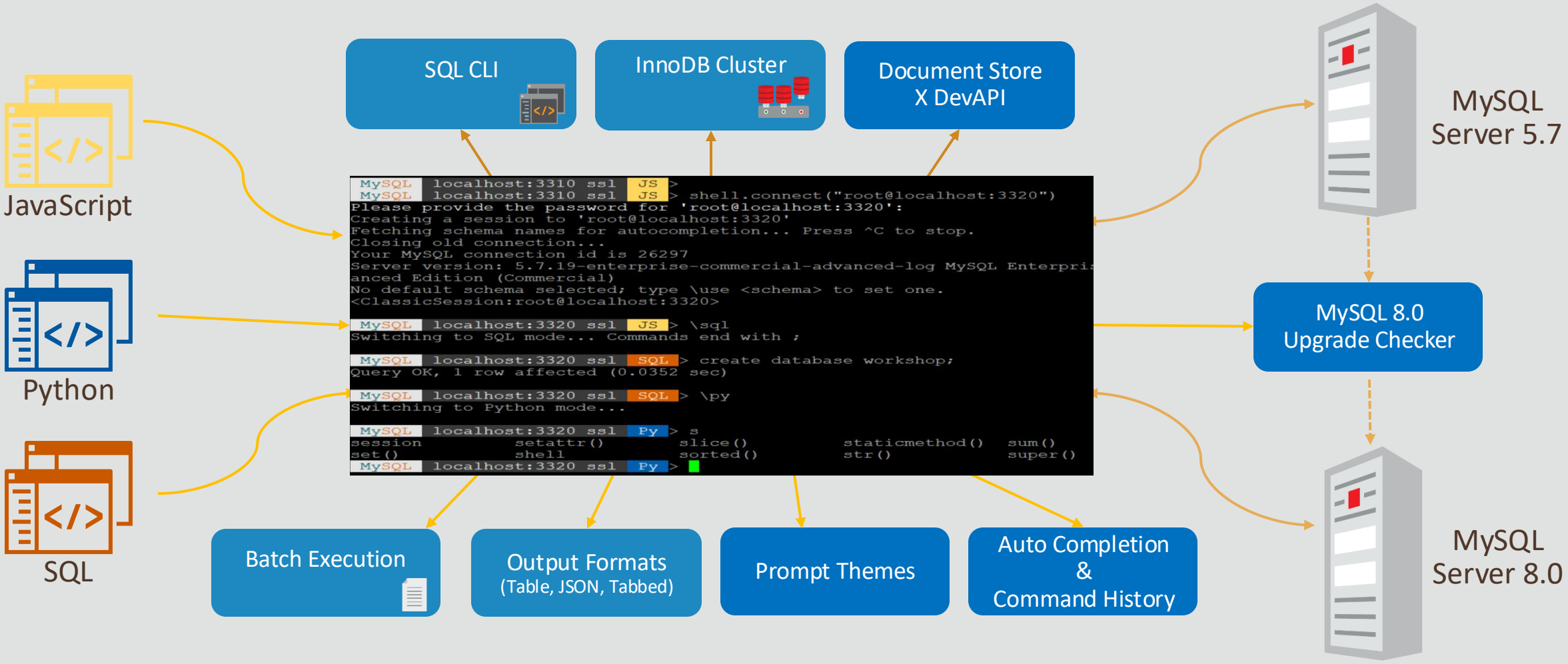
MySQL 8 - JSON_TABLE()

From JSON Document to SQL Table

```
mysql> SELECT emps.* FROM JSON_TABLE(@jsonempl, "$[*]" COLUMNS
(id INT PATH "$.id", name VARCHAR(45) PATH "$.name", age INT
PATH "$.age")) emps;
+------+------+------+
| id   | name | age  |
+------+------+------+
| 1    | John | 34   |
| 2    | Mary | 40   |
| 3    | Mike | 44   |
+------+------+------+
3 rows in set (0,00 sec)
```


Copyright © 2020 Oracle and/or its affiliates.

MySQL Shell 8.0



MySQL Document Store demo “cheat sheet”

```
js> session.createSchema('name')
js> \use doc
js> db.getCollections()
js> db.createCollection('myCollection')
js> db.getCollections()
js> db.myCollection.add({“param1":“value1", “param2":“value2”})
```

Create

```
js> db.myCollection.find()
js> db.myCollection.find().limit(1)
js> db.myCollection.find("_id = '00005af018430000000000000002'")
```

Read

```
js> db.myCollection.modify("_id = '1234'").set(“param",“value")
```

Update

```
js> db.myCollection.remove("_id = '1234'")
```

Delete

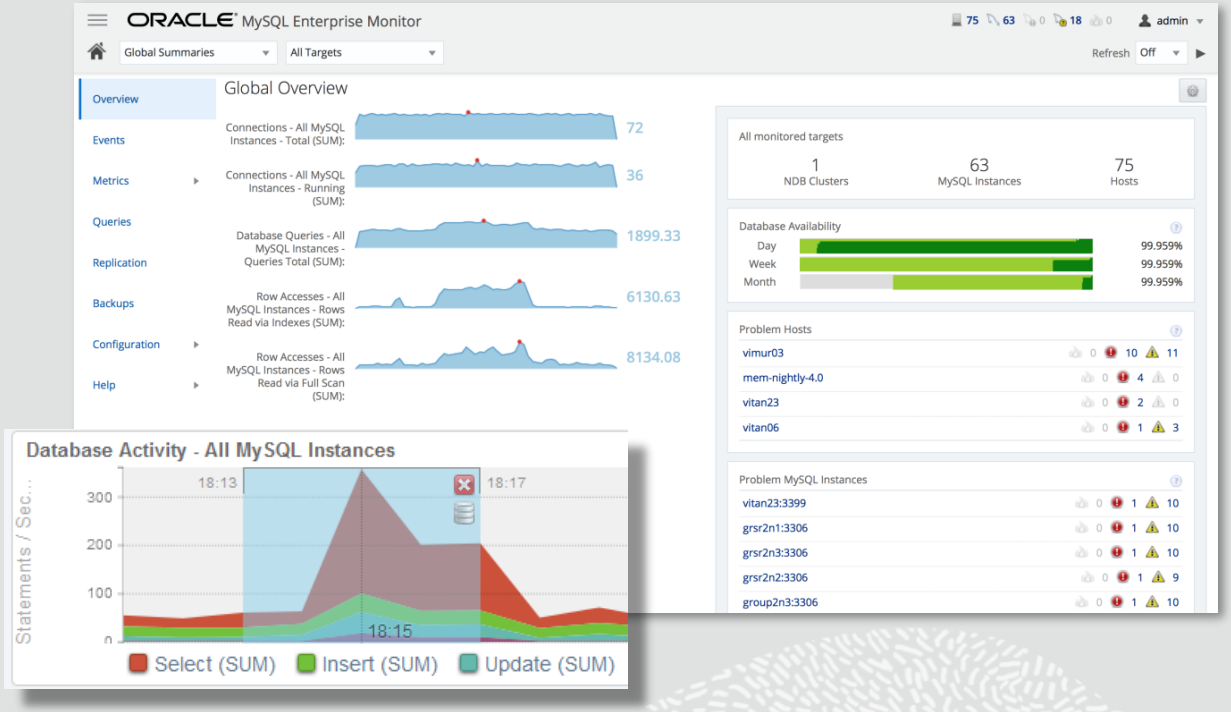
```
js> session.startTransaction()
js> …
js> session.rollback()
```

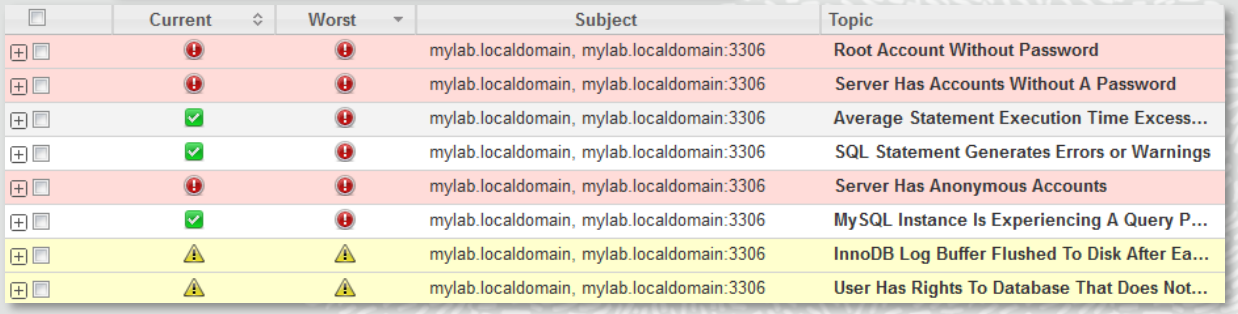
Sessions

Copyright @ 2019 Oracle and/or its affiliates. All rights reserved.

MySQL Enterprise Monitor

- Start monitoring MySQL in 10 minutes
- Real-time MySQL performance and availability monitoring
- Visually find & fix problem queries
- Disk monitoring for capacity planning
- Cloud friendly architecture
  - No agents required
- Role based access controls





Current	Worst	Subject	Topic
		mylab.localdomain, mylab.localdomain:3306	Root Account Without Password
		mylab.localdomain, mylab.localdomain:3306	Server Has Accounts Without A Password
		mylab.localdomain, mylab.localdomain:3306	Average Statement Execution Time Excess...
		mylab.localdomain, mylab.localdomain:3306	SQL Statement Generates Errors or Warnings
		mylab.localdomain, mylab.localdomain:3306	Server Has Anonymous Accounts
		mylab.localdomain, mylab.localdomain:3306	MySQL Instance Is Experiencing A Query P...
		mylab.localdomain, mylab.localdomain:3306	InnoDB Log Buffer Flushed To Disk After Ea...
		mylab.localdomain, mylab.localdomain:3306	User Has Rights To Database That Does Not...

MySQL Er

ORACLE MySQL Enterprise Monitor

Dashboards | Events | Query Analyzer | Reports & Graphs | Configuration | Refresh:

Browse Queries | Show / hide colum

Show 25 entries | Export data options... | Showing 1 to 25 of 1,197 entries | First Previous 1 2 3

Query	Database		Counts			QRTi	Latency (hh:mm:ss.ms)					
			Exec	Err	Warn		Total	Max	Avg	Locks	Avg History	Total
COMMIT (1)	mem		24,707	0	0	0.92	20:27.872	5.828	0.050	0.000		0
INSERT INTO `mem__quan`...o` ) , `hostTo`) , ... (1)	mem		6,903	0	0	0.79	18:16.538	17.784	0.159	6.699		7,356
INSERT INTO `mem__quan`...on` = IF (VALUES (... (1)	mem		6,985	0	0	0.86	10:54.856	8.940	0.094	8.301		7,332
INSERT INTO `mem__quan`...en` ) ) , `lastSeen`) (1)	mem		7,025	37	0	1.00	3:11.791	11.220	0.027	4.436		13,947
INSERT INTO `mem__instr...tency`) , `latency`) (1)	mem		740	0	0	0.77	1:48.459	8.745	0.147	0.147		1,386
SELECT `mysqlconne0_` ....asProc18_1191_0_` , ... (1)	mem		974	0	0	0.90	1:01.829	12.359	0.063	0.353		974
SELECT `mysqlserve0_` ...._` . `hostCache` AS ... (1)	mem		1,801	0	0	0.93	53.661	13.351	0.030	0.636		1,801
INSERT INTO `mem__instr...es`) , `diskWrites`) (1)	mem		26	0	0	0.79	44.838	14.954	1.725	0.070		26
UPDATE `mem__inventory`...n` = ? WHERE `hid` = ? (1)	mem		321	0	0	0.80	43.886	9.668	0.137	0.044		321
UPDATE `mem__config` . ... ? WHERE `user_id` = ? (1)	mem		321	0	0	0.71	40.788	10.607	0.127	0.038		321
CREATE TEMPORARY TABLE ...(`id` INT8 NOT NULL) (1)	mem		13	0	0	0.14	36.525	12.055	2.810	0.000		0
INSERT INTO `mem__instr...nedTableDefinitions`) (1)	mem		26	0	0	0.64	34.348	13.349	1.321	0.003		27
UPDATE `mem__inventory`...p` = ? WHERE `hid` = ? (1)	mem		416	0	0	0.81	33.469	13.060	0.080	0.042		416
INSERT INTO `mem__instr...hed`) , `notCached`) (1)	mem		26	0	0	0.61	32.509	11.782	1.250	0.003		27
SELECT * FROM (SELECT ...m_no_index_used` AS ... (1)	mem		14	0	0	0.00	29.832	9.602	2.131	0.008		7,459
INSERT INTO `mem__instr...s`) , `connections`) (1)	mem		25	0	0	0.82	29.462	14.294	1.178	0.005		26
INSERT INTO `mem__instr... ( `sent` ) , `sent`) (1)	mem		25	0	0	0.82	28.991	14.332	1.160	0.003		26
SELECT `agent0_` . `hid.... `hid` = ? FOR UPDATE (1)	mem		909	0	0	0.96	28.632	6.964	0.031	0.158		909
UPDATE `mem__events` . ...me` = ? WHERE `id` = ? (1)	mem		395	0	0	0.92	27.544	10.061	0.070	0.038		395

Copyright © 2005, 2013, Oracle and/or its affiliates. All rights reserved.

3.0.1.7150 - Cerberus.local (192.168.1.67) - Sep 16, 2013 11:50:28 am (Up Since: 38 mi

Query Analyzer Query Details

Canonical Query | **Example Query** | Explain Query | Graphs

The query with the longest execution time during the Time Span (usually the slowest but not always).

Sampled Query

truncated | full | formatted

```
SELECT
    mysqlserve0_.hid AS hid1124_0_, mysqlserve0_.`id` AS id2_1124_0_,
    mysqlserve0_.`lastContact` AS lastCont3_1124_0_,
    mysqlserve0_.`hasLastContact` AS hasLastC4_1124_0_,
    mysqlserve0_.`startTime` AS startTime5_1124_0_,
    mysqlserve0_.`hasStartTime` AS hasStart6_1124_0_,
    mysqlserve0_.`timestamp` AS timestamp7_1124_0_,
    mysqlserve0_.`capabilities` AS capabili8_1124_0_,
    mysqlserve0_.`hasCapabilities` AS hasCapab9_1124_0_,
    mysqlserve0_.`characterSet` AS charact10_1124_0_,
    mysqlserve0_.`hasCharacterSet` AS hasChar11_1124_0_,
    mysqlserve0_.`collation` AS collation12_1124_0_,
    mysqlserve0_.`hasCollation` AS hasColl13_1124_0_,
    mysqlserve0_.`connection` AS connection14_1124_0_,
    mysqlserve0_.`hasConnection` AS hasConn15_1124_0_,
    mysqlserve0_.`environment` AS environ16_1124_0_,
```

Execution Time
27,084 ms

Date
Sep 16, 2013 1:07:17 PM

User
service_manager

Thread ID
10,712

From Host
localhost

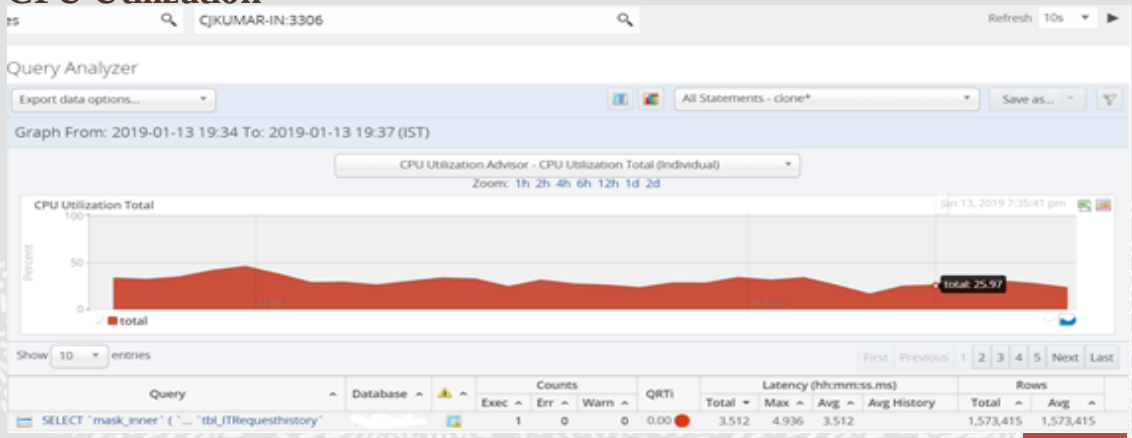
To Host

Source Location
None found.

Comments
None found.

Disk Operations

CPU Utilization



Query Analyzer Query Details

Canonical Query | **Example Query** | Explain Query | Graphs

The query with the longest execution time during the Time Span (usually the slowest but not always).

Sampled Query

truncated | full | formatted

```
SELECT
    mysqlserve0_.hid AS hid1124_0_, mysqlserve0_.`id` AS id2_1124_0_,
    mysqlserve0_.`lastContact` AS lastCont3_1124_0_,
    mysqlserve0_.`hasLastContact` AS hasLastC4_1124_0_,
    mysqlserve0_.`startTime` AS startTime5_1124_0_,
    mysqlserve0_.`hasStartTime` AS hasStart6_1124_0_,
    mysqlserve0_.`timestamp` AS timestamp7_1124_0_,
    mysqlserve0_.`capabilities` AS capabili8_1124_0_,
    mysqlserve0_.`hasCapabilities` AS hasCapab9_1124_0_,
    mysqlserve0_.`characterSet` AS charact10_1124_0_,
    mysqlserve0_.`hasCharacterSet` AS hasChar11_1124_0_,
    mysqlserve0_.`collation` AS collation12_1124_0_,
    mysqlserve0_.`hasCollation` AS hasColl13_1124_0_,
    mysqlserve0_.`connection` AS connection14_1124_0_,
    mysqlserve0_.`hasConnection` AS hasConn15_1124_0_,
    mysqlserve0_.`environment` AS environ16_1124_0_,
```

Execution Time
27,084 ms

Date
Sep 16, 2013 1:07:17 PM

User
service_manager

Thread ID
10,712

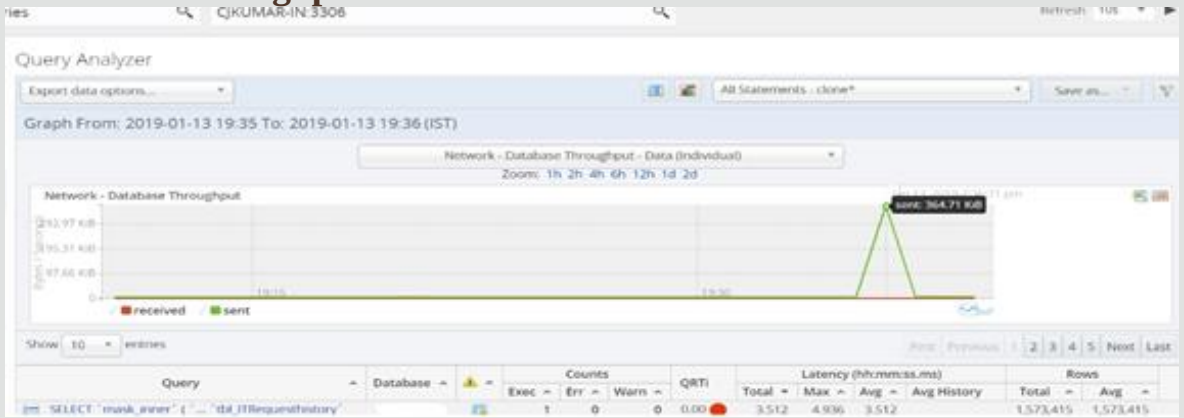
From Host
localhost

To Host

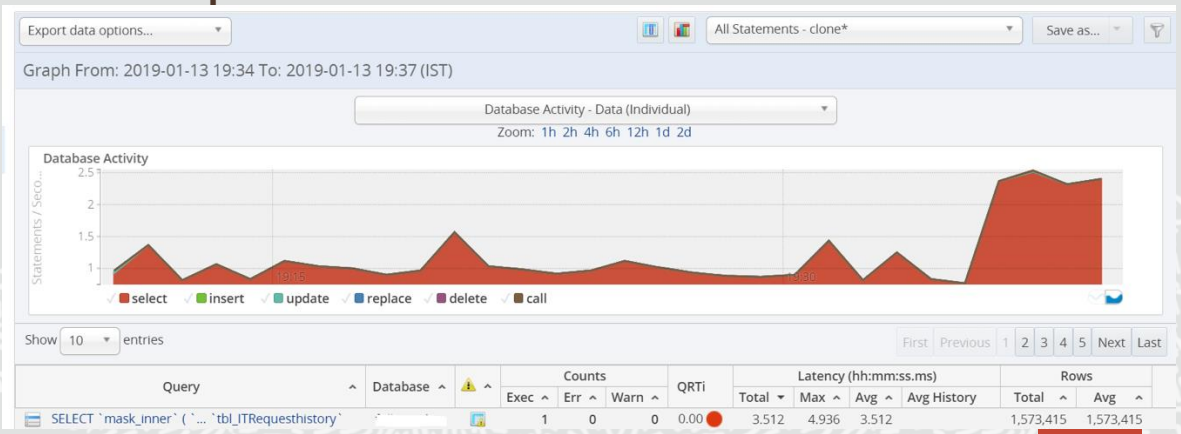
Source Location
None found.

Comments
None found.

Network Throughput

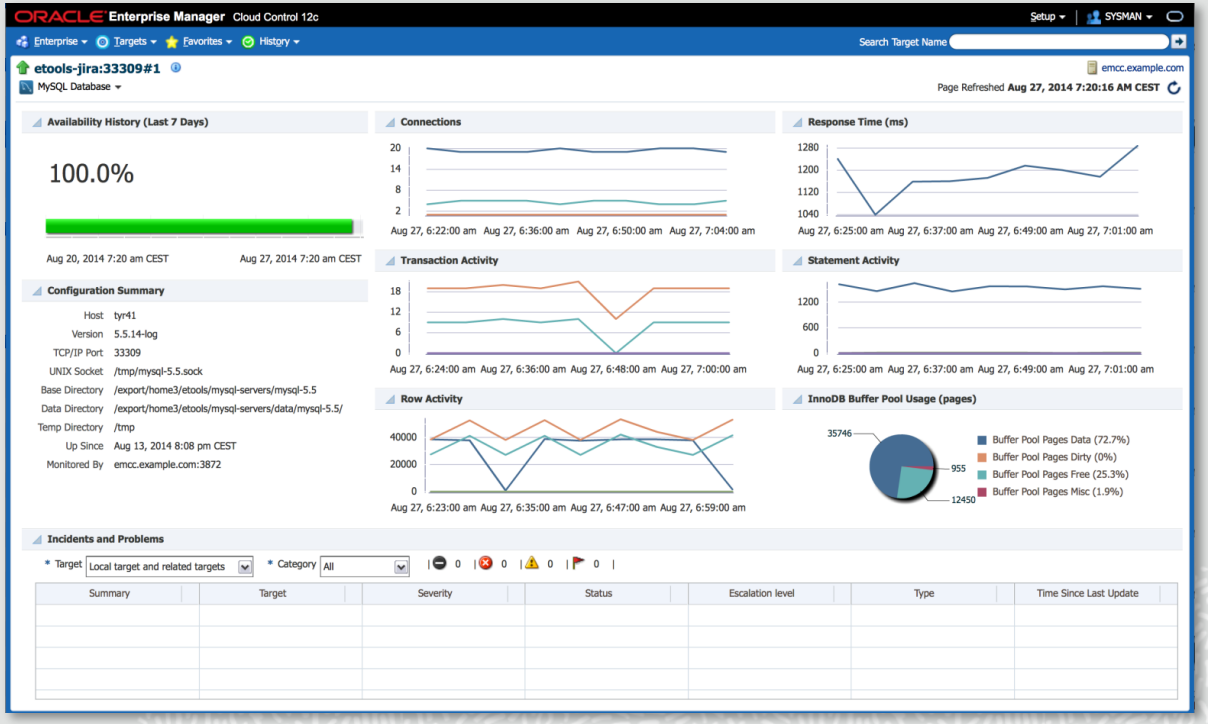


Database Operations



MySQL Enterprise Plugin for Enterprise Manager

- Availability monitoring
- Performance monitoring
- Configuration monitoring
- All available metrics collected
  - Allowing for custom threshold based incident reports
- MySQL auto-detection

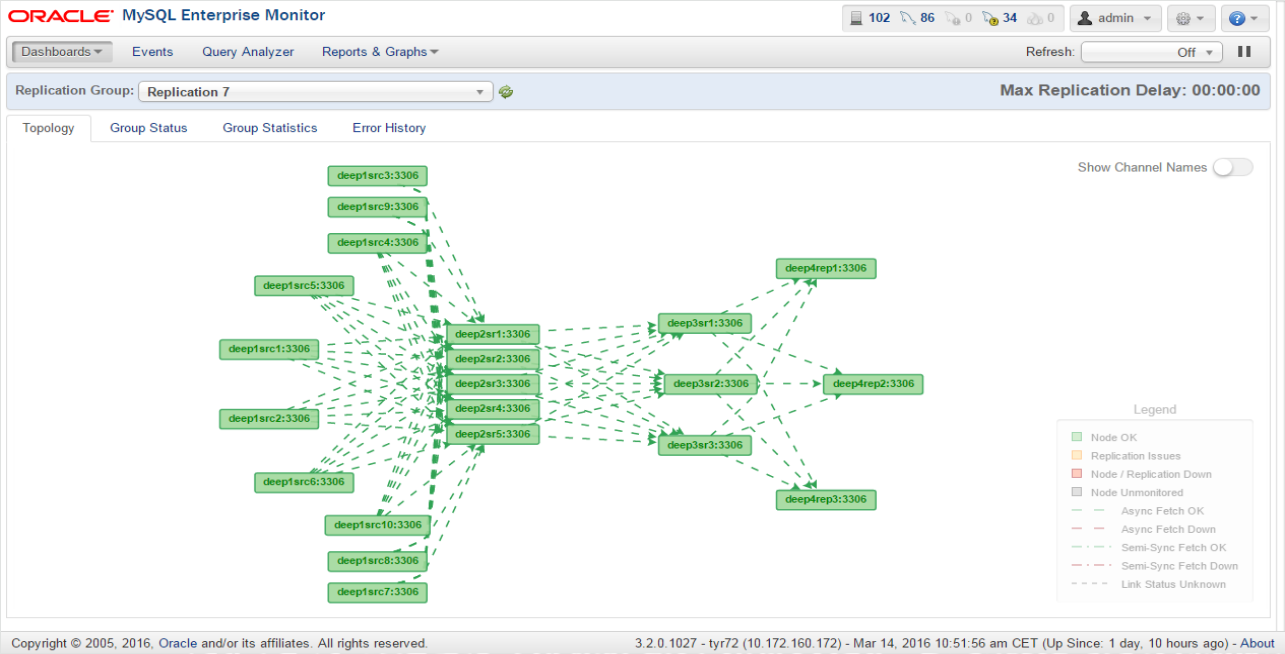


MySQL Enterprise Monitor : Replication Dashboard

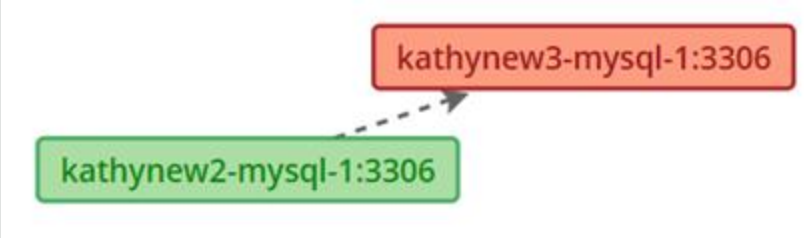
- Auto-discovers replication topology
  - High level overviews of all instances
  - User centric views on individual sets
- Master/Slave performance monitoring
- Replication advisor
- Best practice replication advice

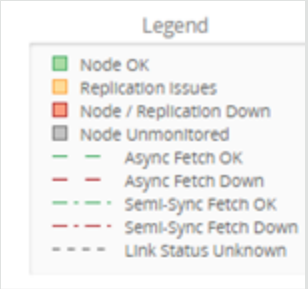
"I use the MySQL Enterprise Monitor every day to monitor and keep tabs on our MySQL databases. Quick one stop shopping for keeping tabs on them."
-Wes Homer,
Sr System and Network Administrator

InsideView



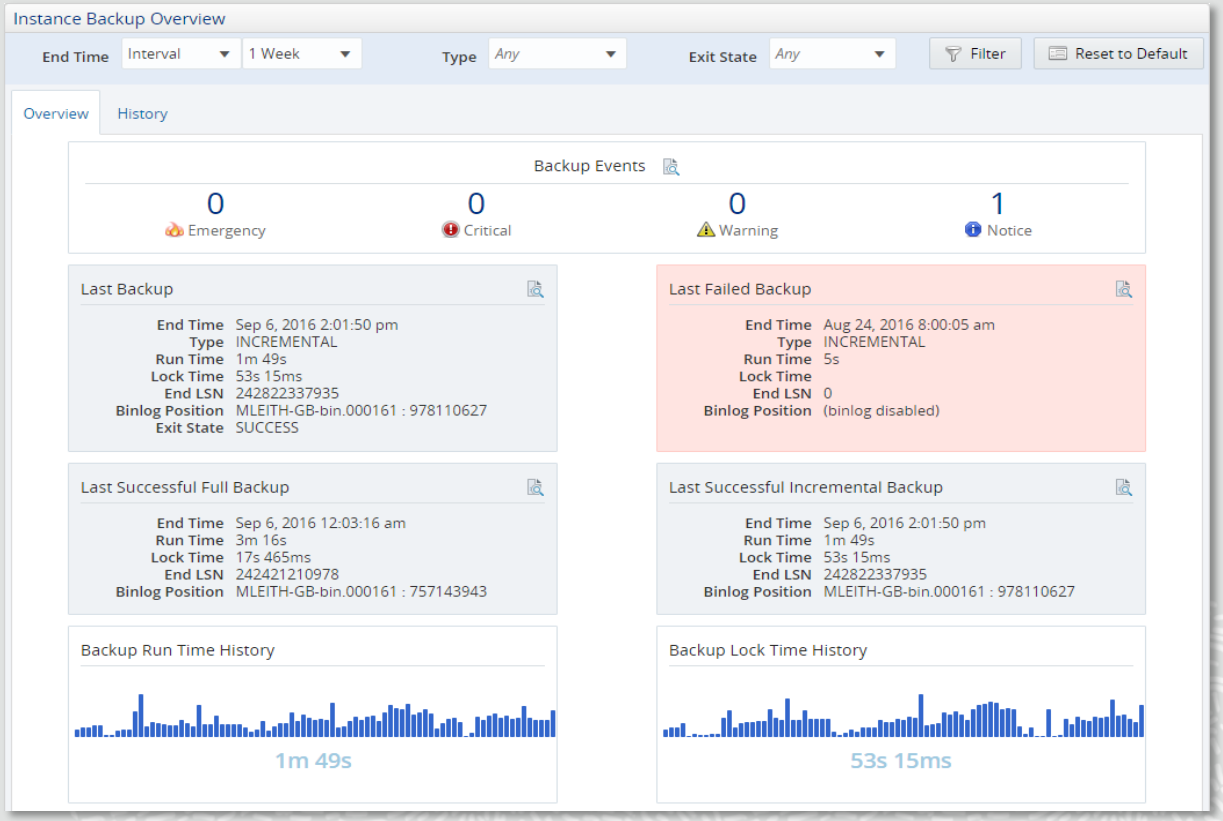
Replication Topology – Master Slave Monitoring (Red Slave is Down)





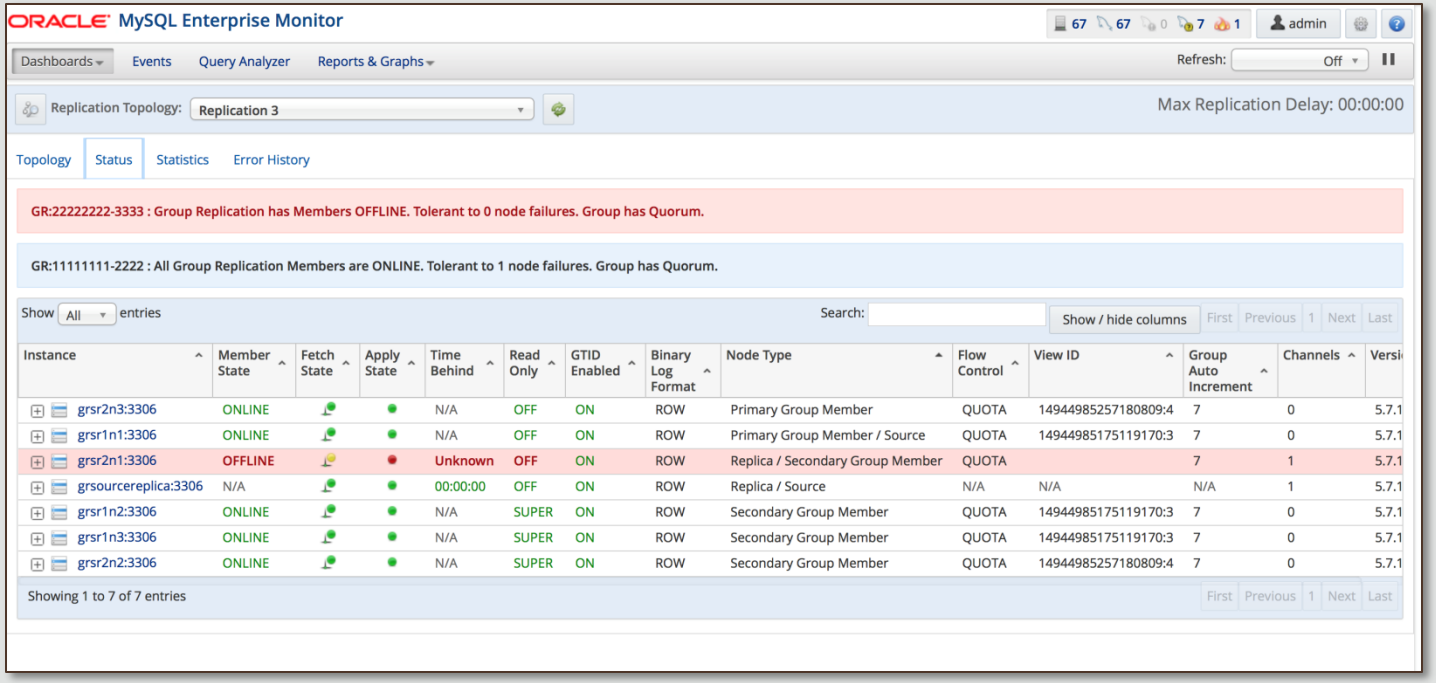
MySQL Enterprise Monitor: Backup Dashboard

- Monitor backup usage and health
  - Across your entire datacenter
- Drill into backup job details
  - Allowing for easy backup recovery
- Supports all backup types
- Alerting on significant events
  - Poor backup performance
  - Backup job failures
  - Out of date backups



Replication Monitoring: Group Status & Statistics

- Binary Logging
- GTIDs
- Group Replication
- Semi-Sync Replication
- Relay Logging
- Multi-threaded Replication
- Multi-source Replication



Instance	Member State	Fetch State	Apply State	Time Behind	Read Only	GTID Enabled	Binary Log Format	Node Type	Flow Control	View ID	Group Auto Increment	Channels	Versi
grsr2n3:3306	ONLINE			N/A	OFF	ON	ROW	Primary Group Member	QUOTA	14944985257180809:4	7	0	5.7.1
grsr1n1:3306	ONLINE			N/A	OFF	ON	ROW	Primary Group Member / Source	QUOTA	14944985175119170:3	7	0	5.7.1
grsr2n1:3306	OFFLINE			Unknown	OFF	ON	ROW	Replica / Secondary Group Member	QUOTA		7	1	5.7.1
grsourcereplica:3306	N/A			00:00:00	OFF	ON	ROW	Replica / Source	N/A	N/A	N/A	1	5.7.1
grsr1n2:3306	ONLINE			N/A	SUPER	ON	ROW	Secondary Group Member	QUOTA	14944985175119170:3	7	0	5.7.1
grsr1n3:3306	ONLINE			N/A	SUPER	ON	ROW	Secondary Group Member	QUOTA	14944985175119170:3	7	0	5.7.1
grsr2n2:3306	ONLINE			N/A	SUPER	ON	ROW	Secondary Group Member	QUOTA	14944985257180809:4	7	0	5.7.1

Best Practice Advisors

- Enforce MySQL best practices
- 14 Advisor categories
- 250+ Advisors
- Threshold-based alerts
  - Exponential moving averages
  - Rate change detection
- Expert problem resolution advice

"I definitely recommend the MySQL Enterprise Monitor to DBAs who don't have a ton of MySQL experience. It makes monitoring MySQL security, performance and availability very easy to understand and to act on."

Sandi Barr
Sr. Software Engineer
Schneider Electric

Security Administration

- Account management
- Firewall management
- Audit management
- Know when privileges change
- Get advice on best practices
- Monitor access problems

Current	Worst	Subject	Topic
		mylab.localdomain, mylab.localdomain:3306	Root Account Without Password
		mylab.localdomain, mylab.localdomain:3306	Server Has Accounts Without A Password
		mylab.localdomain, mylab.localdomain:3306	Average Statement Execution Time Excess...
		mylab.localdomain, mylab.localdomain:3306	SQL Statement Generates Errors or Warnings
		mylab.localdomain, mylab.localdomain:3306	Server Has Anonymous Accounts

mylab.localdomain, mylab.localdomain:3306 **Server Has Accounts Without A Password** 6 minutes ago

Topic: Server Has Accounts Without A Password

Categories: Security
Advisor: Server Has Accounts Without A Password
Current State: Open
Current Status: Critical
Worst Status: Critical
Auto-Closes by Default: Yes
Last Checked: Sep 3, 2013 1:17:40 PM
Worst Alarm Time: Sep 3, 2013 11:57:39 AM

Notes:

No notes provided.

Details:

Problem Description

Accounts without passwords are particularly dangerous because an attacker needs to guess only a username. Assigning passwords to all accounts helps prevent unauthorized users from accessing the system.

Advice

Assign a strong password to the following accounts on server **mylab.localdomain:3306**:

- ''@'localhost',
- ''@'mylab.localdomain',
- 'root'@'127.0.0.1',
- 'root'@'::1',
- 'root'@'mylab.localdomain'

A strong password should be at least 8 characters long, contain lowercase and uppercase characters, numbers and symbols, and not contain words found in a dictionary.

Recommended Action

SET PASSWORD FOR 'user_name'@'host_name' = PASSWORD('new_pass');

Links and Further Reading

Securing Your MySQL Installation
Securing a MySQL Server on Windows

Expression

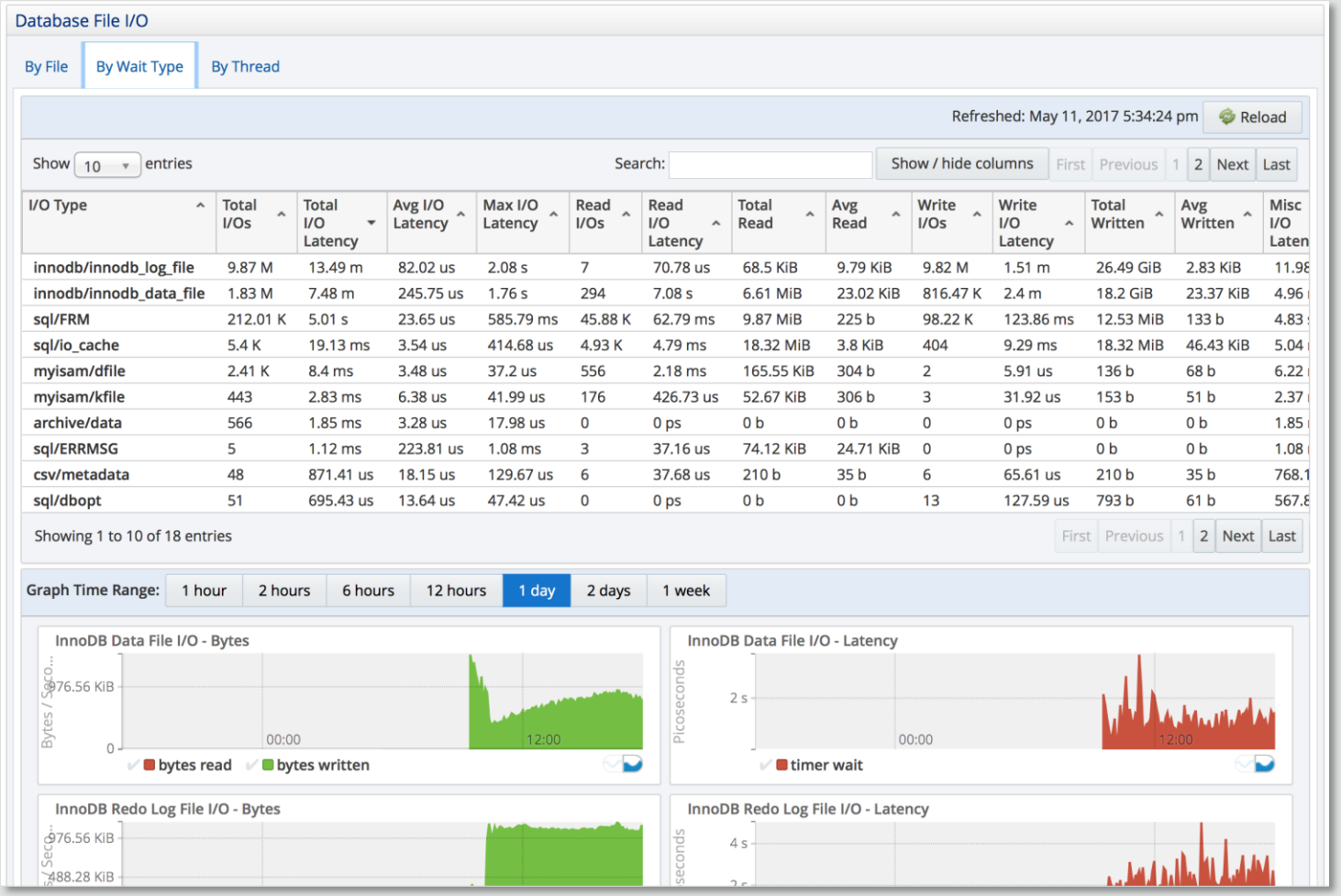
%user% != THRESHOLD

Evaluated Expression

- ''@'localhost',
- ''@'mylab.localdomain',
- 'root'@'127.0.0.1',
- 'root'@'::1',
- 'root'@'mylab.localdomain' != ''

Deep Insights: Database File IO Report

- Database activity by File
- See file IO hot spots
  - By file
  - By waits
  - By connection / thread
- Detailed breakdown of stats



I/O Type	Total I/Os	Total I/O Latency	Avg I/O Latency	Max I/O Latency	Read I/Os	Read I/O Latency	Total Read	Avg Read	Write I/Os	Write I/O Latency	Total Written	Avg Written	Misc I/O Laten
innodb/innodb_log_file	9.87 M	13.49 m	82.02 us	2.08 s	7	70.78 us	68.5 KiB	9.79 KiB	9.82 M	1.51 m	26.49 GiB	2.83 KiB	11.98
innodb/innodb_data_file	1.83 M	7.48 m	245.75 us	1.76 s	294	7.08 s	6.61 MiB	23.02 KiB	816.47 K	2.4 m	18.2 GiB	23.37 KiB	4.96
sql/FRM	212.01 K	5.01 s	23.65 us	585.79 ms	45.88 K	62.79 ms	9.87 MiB	225 b	98.22 K	123.86 ms	12.53 MiB	133 b	4.83
sql/io_cache	5.4 K	19.13 ms	3.54 us	414.68 us	4.93 K	4.79 ms	18.32 MiB	3.8 KiB	404	9.29 ms	18.32 MiB	46.43 KiB	5.04
myisam/dfile	2.41 K	8.4 ms	3.48 us	37.2 us	556	2.18 ms	165.55 KiB	304 b	2	5.91 us	136 b	68 b	6.22
myisam/kfile	443	2.83 ms	6.38 us	41.99 us	176	426.73 us	52.67 KiB	306 b	3	31.92 us	153 b	51 b	2.37
archive/data	566	1.85 ms	3.28 us	17.98 us	0	0 ps	0 b	0 b	0	0 ps	0 b	0 b	1.85
sql/ERRMSG	5	1.12 ms	223.81 us	1.08 ms	3	37.16 us	74.12 KiB	24.71 KiB	0	0 ps	0 b	0 b	1.08
csv/metadata	48	871.41 us	18.15 us	129.67 us	6	37.68 us	210 b	35 b	6	65.61 us	210 b	35 b	768.1
sql/dbopt	51	695.43 us	13.64 us	47.42 us	0	0 ps	0 b	0 b	13	127.59 us	793 b	61 b	567.8

Deep Insights: Lock Waits Report

- See row lock hot spots
- See blocking metadata locks
- Relate locks back to user connections and transactions
  - Understand behaviors
  - Remove bottlenecks
  - Unblock traffic

Refreshed: Sep 14, 2015 5:46:54 pm Reload

Show 10 entries Search: Show / hide columns First Previous 1 Next Last

Locked Table	Locked Index	Locked Type	Wait Started	Wait Age	Waiting Trx ID	Waiting Trx Started	Waiting Trx Age	Waiting Trx Rows Locked	Waiting Trx Rows Modified
`test`.`t`	GEN_CLUST_INDEX	RECORD	Sep 14, 2015 5:46:49 pm	5s	181998620	Sep 14, 2015 4:20:18 pm	1h 26m 36s	1	0
`test`.`t`	GEN_CLUST_INDEX	RECORD	Sep 14, 2015 5:46:52 pm	2s	181998745	Sep 14, 2015 4:20:22 pm	1h 26m 32s	1	0
`test`.`t`	GEN_CLUST_INDEX	RECORD	Sep 14, 2015 5:46:52 pm	2s	181998745	Sep 14, 2015 4:20:22 pm	1h 26m 32s	1	0

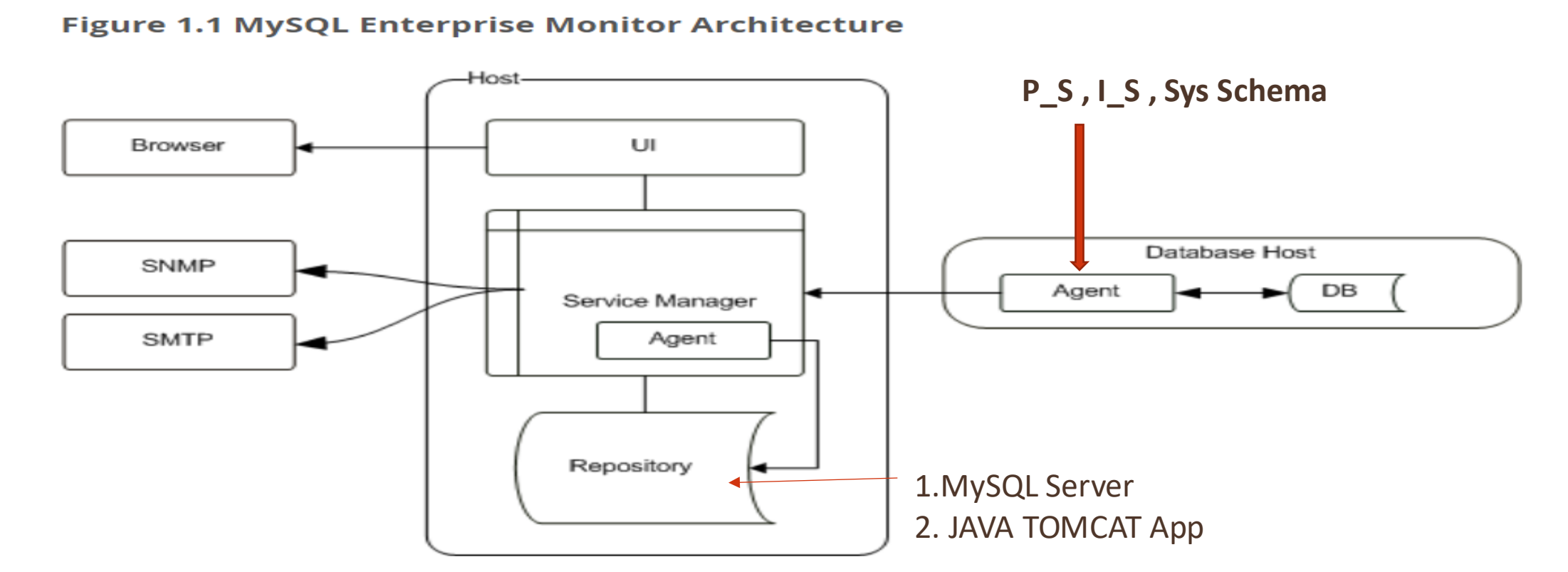
Showing 1 to 3 of 3 entries First Previous 1 Next Last

Solving Common Performance Issues

Causes of Performance Issues	MySQL Enterprise Monitor Helps
Queries doing table scans	See these queries immediately
Excessive temporary tables on disk	See the queries generating these
CPU spikes	Find out what caused it
Disk I/O saturation	Find out what caused it
Internal locking	Get alerts when locking issues occur
Hardware problems	Get alerts for disk and network problems
Database and schema changes	Get alerts when changes are made
New queries introduced	See what queries are added and when
Poor MySQL configuration	Get advice on configuration tuning

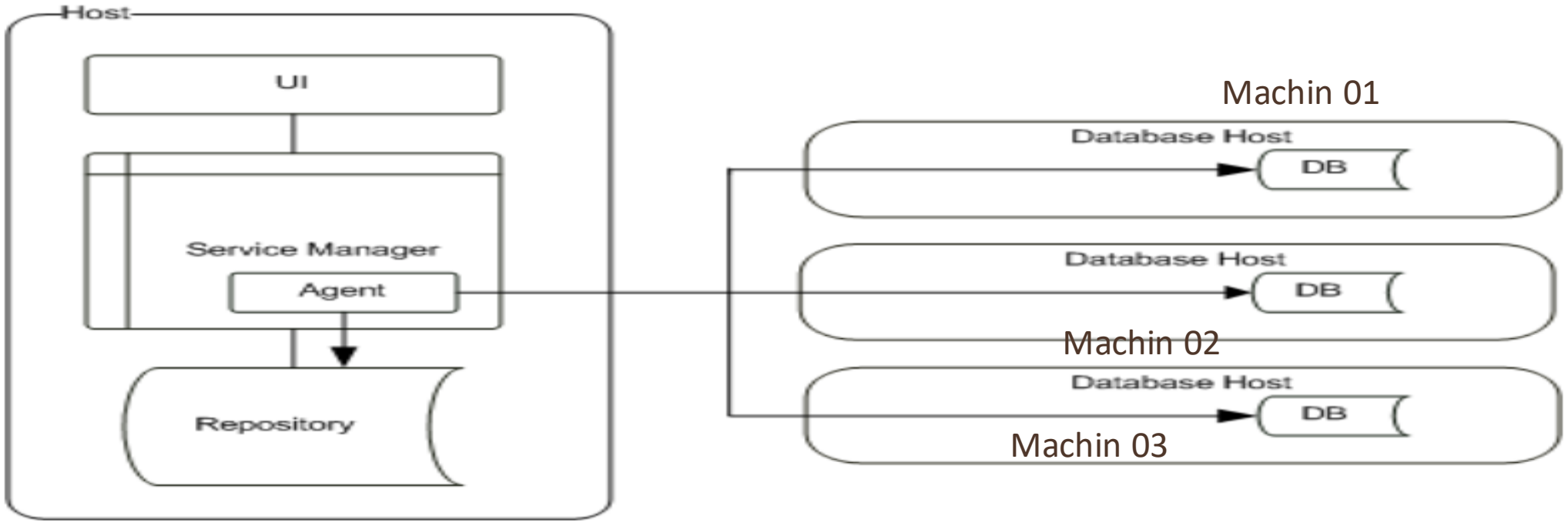
MySQL Enterprise Monitor : Architecture

1,SERVICE MGR
2. AGENT



MySQL Enterprise Monitor : Architecture

Figure 1.2 MySQL Enterprise Monitor Agentless Architecture



MySQL Enterprise Monitor : Where to Download

https://edelivery.oracle.com

☑	**MySQL Enterprise Edition 0.0.0.0.0**	
☑	MySQL Enterprise Monitor 8.0.17 for Microsoft Windows x64 (64-bit)	
☑	V982881-01.zip	MySQL Enterprise Monitor Service Manager 8.0.17 for Windows x86 (64-bit), 547.8 MB
☑	V982889-01.zip	MySQL Enterprise Monitor Agent 8.0.17 for Microsoft Windows x86 (64-bit), 228.7 MB

MySQL Enterprise Monitor : Installation Notes

Service Manager

- Step-1: cd /opt/packages/mem
- Step-2 :chmod +x mysqlmonitor-8.0.0.8131-linux-x86_64-installer.bin
- Step-3: ./opt/packages/mem/mysqlmonitor-4.0.0.5135-linux-x86_64-installer.bin
- To check MEM Monitor Running or not

  Verify: service mysql-monitor-server status

Agent

- Step-1: cd /opt/packages/mem
- Step-2: chmod +x mysqlmonitoragent-4.0.0.5135-linux-x86-64bit-installer.bin
- Step-./opt/packages/mem/mysqlmonitoragent-4.0.0.5135-linux-x86-64bit-installer.bin
- Step-4: /opt/mysql/enterprise/agent/etc/init.d/mysql-monitor-agent start
- To Check Agent Status

Verify: service mysql-monitor-agent status

MySQL Enterprise Monitor : Installation Notes

Service Manager

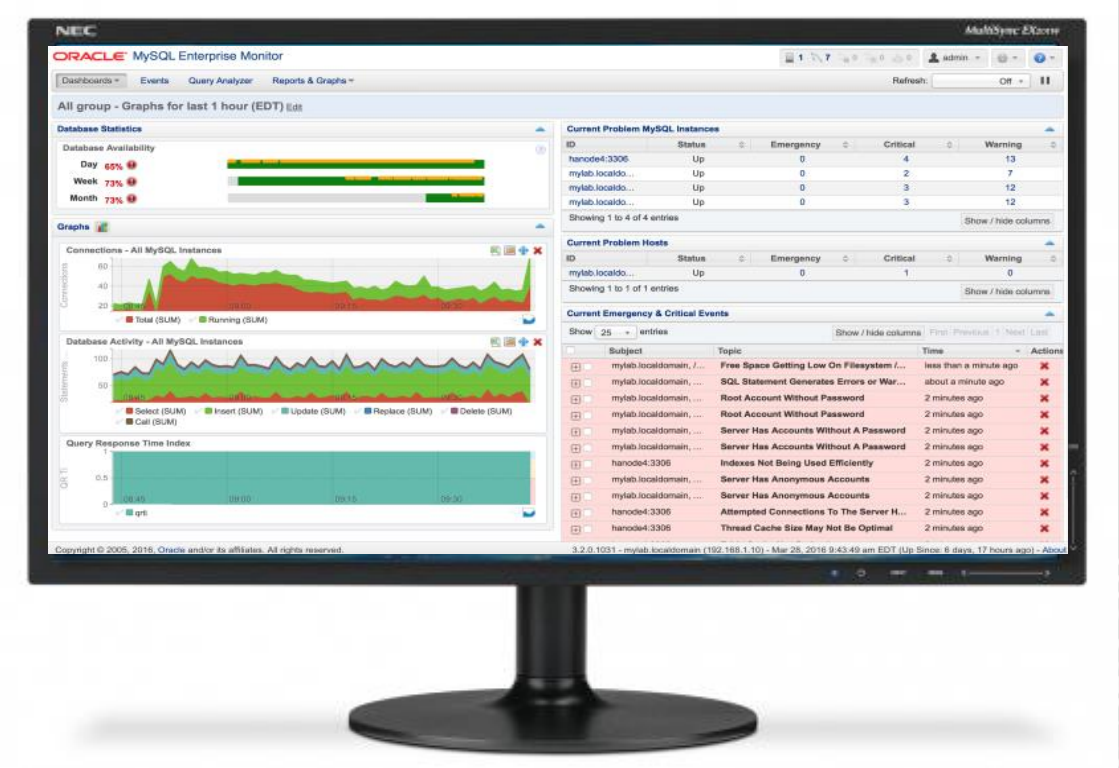
Installation Keywords	Value
Installation Directory	/opt/mysql/enterprise/monitor
Tomcat Server Port	18080
Tomcat SSL Port	18443
User Account	mysqlmem
Database Installation	(1) Using bundled MySQL database
Repository Username	service_manager
Password	<please key in the password> e.g. 123456
MySQL Database Port	13306
MySQL Database Name	mem
Use SSL	N

Agent

Installation Keywords	Value
Installation Directory	/opt/mysql/enterprise/agent
Host Name or IP Address	IP Address of Service Manager
Tomcat SSL Port	18443
Agent User Name	Same has to be all agents host
Agent Password	<Enter password of Agent>
MySQL Host or IP Address	IP Address of Host where MySQL is Running
Admin User/Password	mem/Mem@12345
After Agent Installation Please Start it by below commands	/opt/mysql/enterprise/agent/etc/init.d/mysql-monitor-agent start

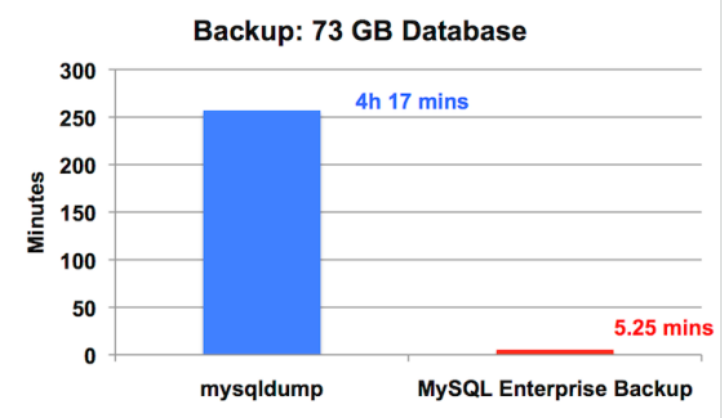
MySQL Enterprise Monitor : Installation & Configuration

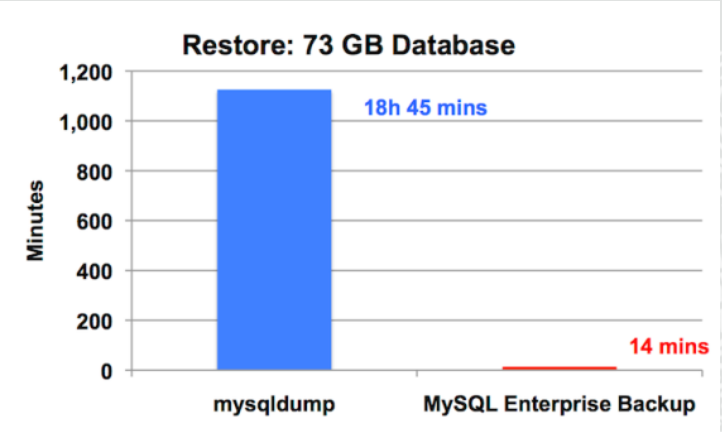
- Login to MySQL MEM Monitor
  - https://<IP Address of service manager>:18443
- Up and Running in 10 minutes
  - Auto-discovery of MySQL servers
  - Advisors are pre-configured
- Real-time performance monitoring
- Replication Monitoring
- Backup Monitoring
- **SLA Monitoring & Alerts/Notification**



MySQL Enterprise Backup

- Online, non-locking backup and recovery
  - Complete MySQL instance backup (data and config)
  - Partial backup and restore
- Direct **Cloud** storage backups
  - Oracle Storage Cloud, S3, etc.
- Incremental backups
- Point-in-time recovery
- Advanced compressed and encryption
- Backup to tape (SBT)
- Cross-Platform (Windows, Linux, Unix)





MySQL Enterprise Backup : How to Take Backup ?

Easy to use tool for backup and restore operations

***mysqlbackup* --port=3306 --host=127.0.0.1 --protocol=tcp --user=root --backup-dir=/home/mysql/backup/full --with-timestamp backup-and-apply-log**

command : backup-and-apply-log

Includes the initial phase of backup and the second phase, which brings the InnoDB tables in the backup up to date, including any changes made to the data while the backup was running

options :

--port, --host, –protocol, --user, --password (for connection to MySQL)

--backup-dir : The backup directory

--with-timestamp : The backup is stored in a subdirectory of the given backup-dir whose name is formed using current timestamp. Useful to maintain single backup-dir which stores many backup snapshots in it's subdirectories.

Progress…

mysql@virtual-22:~/demo/MEB

```
MySQL Enterprise Backup version 4.0.3 Linux-2.6.18-194.el5-x86_64 [2016/08/04]
Copyright (c) 2003, 2016, Oracle and/or its affiliates. All Rights Reserved.

 mysqlbackup: INFO: Starting with following command line ...
 /usr/local/meb/bin/mysqlbackup --port=3306 --host=127.0.0.1
        --protocol=tcp --user=root --password=xxxxx
        --backup-dir=/home/mysql/backup/full --with-timestamp
        backup-and-apply-log

 mysqlbackup: INFO:
 mysqlbackup: INFO: MySQL server version is '5.7.17-enterprise-commercial-advanced'
 mysqlbackup: INFO: MySQL server compile os version is 'linux-glibc2.5'
 mysqlbackup: INFO: Got some server configuration information from running server.

IMPORTANT: Please check that mysqlbackup run completes successfully.
           At the end of a successful 'backup-and-apply-log' run mysqlbackup
           prints "mysqlbackup completed OK!".

170120 16:44:29 mysqlbackup: INFO: MEB logfile created at /home/mysql/backup/full/2017-01-20_16-44-29/meta/MEB_2017-01-20.16-44-29_bac
kup_apply_log.log

--------------------------------------------------------------------
                       Server Repository Options:
--------------------------------------------------------------------
```

Progress…

mysql@virtual-22:~/demo/MEB

```
MySQL Enterprise Backup version 4.0.3 Linux-2.6.18-194.el5-x86_64 [2016/08/04]
Copyright (c) 2003, 2016, Oracle and/or its affiliates. All Rights Reserved.

 mysqlbackup: I
 /usr/local/meb
        --proto
        --backu
        backup-

 mysqlbackup: I
 mysqlbackup: I
 mysqlbackup: I
 mysqlbackup: I

IMPORTANT: Plea
           At t
           prin

170120 16:44:29
kup_apply_log.l

---------------

---------------
```

mysql@virtual-22:~/demo/MEB

```
-----------------------------------------------------------------------
                        Server Repository Options:
-----------------------------------------------------------------------
  datadir = /home/mysql/data/10-secure/data1/
  innodb_data_home_dir =
  innodb_data_file_path = ibdata1:12M:autoextend
  innodb_log_group_home_dir = /home/mysql/data/10-secure/data1/
  innodb_log_files_in_group = 2
  innodb_log_file_size = 50331648
  innodb_page_size = 16384
  innodb_checksum_algorithm = crc32
  innodb_buffer_pool_filename = ib_buffer_pool
  innodb_undo_directory = /home/mysql/data/10-secure/data1/
  innodb_undo_tablespaces = 0
  innodb_undo_logs = 128

-----------------------------------------------------------------------
                        Backup Config Options:
-----------------------------------------------------------------------
  datadir = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_data_home_dir = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_data_file_path = ibdata1:12M:autoextend
  innodb_log_group_home_dir = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_log_files_in_group = 2
```

Progress…

mysql@virtual-22:~/demo/MEB

```
--------------------------------------------------------------------
                      Backup Config Options:
--------------------------------------------------------------------
  datadir = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_data_home_dir = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_data_file_path = ibdata1:12M:autoextend
  innodb_log_group_home_dir = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_log_files_in_group = 2
  innodb_log_file_size = 50331648
  innodb_page_size = 16384
  innodb_checksum_algorithm = crc32
  innodb_buffer_pool_filename = ib_buffer_pool
  innodb_undo_directory = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_undo_tablespaces = 0
  innodb_undo_logs = 128

 mysqlbackup: INFO: Unique generated backup id for this is 14849018694067557

 mysqlbackup: INFO: Creating 14 buffers each of size 16777216.
170120 16:44:29 mysqlbackup: INFO: Full Backup operation starts with following threads
                1 read-threads    6 process-threads    1 write-threads
170120 16:44:29 mysqlbackup: INFO: System tablespace file format is Barracuda.
170120 16:44:29 mysqlbackup: INFO: Starting to copy all innodb files...
 mysqlbackup: INFO: Could not find binlog index file. binlogs will not be copied for this backup.
```

```
--------------------------------------------------------------------
  datadir = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_data_home_dir = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_data_file_path = ibdata1:12M:autoextend
  innodb_log_group_home_dir = /home/mysql/backup/full/2017-01-20_16-44-29/datadir
  innodb_log_files_in_group = 2
```

Progress…

mysql@virtual-22:~/demo/MEB

```
170120 16:44:29 mysqlbackup: INFO: Copying /home/mysql/data/10-secure/data1/sys/sys_config.ibd (Barracuda file format).
170120 16:44:29 mysqlbackup: INFO: Completing the copy of innodb files.
170120 16:44:29 mysqlbackup: INFO: Preparing to lock tables: Connected to mysqld server.
170120 16:44:29 mysqlbackup: INFO: Starting to lock all the tables...
170120 16:44:29 mysqlbackup: INFO: All tables are locked and flushed to disk
170120 16:44:29 mysqlbackup: INFO: Opening backup source directory '/home/mysql/data/10-secure/data1'
170120 16:44:29 mysqlbackup: INFO: Starting to backup all non-innodb files in
        subdirectories of '/home/mysql/data/10-secure/data1'
170120 16:44:29 mysqlbackup: INFO: Copying the database directory 'auditdb1'
170120 16:44:29 mysqlbackup: INFO: Copying the database directory 'mysql'
170120 16:44:29 mysqlbackup: INFO: Copying the database directory 'performance_schema'
170120 16:44:29 mysqlbackup: INFO: Copying the database directory 'sys'
170120 16:44:29 mysqlbackup: INFO: Completing the copy of all non-innodb files.
170120 16:44:29 mysqlbackup: INFO: A copied database page was modified at 2792613.
           (This is the highest lsn found on page)
           Scanned log up to lsn 2792736.
           Was able to parse the log up to lsn 2792736.
170120 16:44:30 mysqlbackup: INFO: All tables unlocked
170120 16:44:30 mysqlbackup: INFO: All MySQL tables were locked for 0.215 seconds.
170120 16:44:30 mysqlbackup: INFO: Reading all global variables from the server.
170120 16:44:30 mysqlbackup: INFO: Completed reading of all global variables from the server.
 mysqlbackup: INFO: Writing config file for server '5.7.17-enterprise-commercial-advanced'.
170120 16:44:30 mysqlbackup: INFO: Creating server config files server-my.cnf and server-all.cnf in /home/mysql/backup/full/2017-01-20
_16-44-29
```

Locking and Unlocking Stage for Non-InnoDB files backup

Progress…

mysql@virtual-22:~/demo/MEB

```
 Point-In-Time-Recovery will not be possible.
 If this is online backup then server may not have started with --log-bin.
 You may specify its location with --log-bin-index option.
 mysqlbackup: INFO: Using up to 100 MB of memory.
170120 16:44:30 mysqlbackup: INFO: ibbackup_logfile's creation parameters:
          start lsn 2792448, end lsn 2792736,
          start checkpoint 2792727.
 mysqlbackup: INFO: InnoDB: Doing recovery: scanned up to log sequence number 2792736.
 mysqlbackup: INFO: InnoDB: Starting an apply batch of log records to the database...
.
InnoDB: Progress in percent: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 3
8 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82
83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
 mysqlbackup: INFO: InnoDB: Setting log file size to 50331648.
 mysqlbackup: INFO: InnoDB: Setting log file size to 50331648.
170120 16:44:30 mysqlbackup: INFO: We were able to parse ibbackup_logfile up to
          lsn 2792736.
170120 16:44:30 mysqlbackup: INFO: The first data file is '/home/mysql/backup/full/2017-01-20_16-44-29/datadir/ibdata1'
          and the new created log files are at '/home/mysql/backup/full/2017-01-20_16-44-29/datadir'
170120 16:44:30 mysqlbackup: INFO: Apply-log operation completed successfully.
170120 16:44:30 mysqlbackup: INFO: Full backup prepared for recovery successfully.

mysqlbackup completed OK!
[mysql@virtual-22 MEB]$
```

MySQL Enterprise Backup : How to Restore Backup ?

Easy to use tool for backup and restore operations

***mysqlbackup* –defaults-file=<my.cnf>**
--backup-dir=/home/mysql/backup/full
copy-back

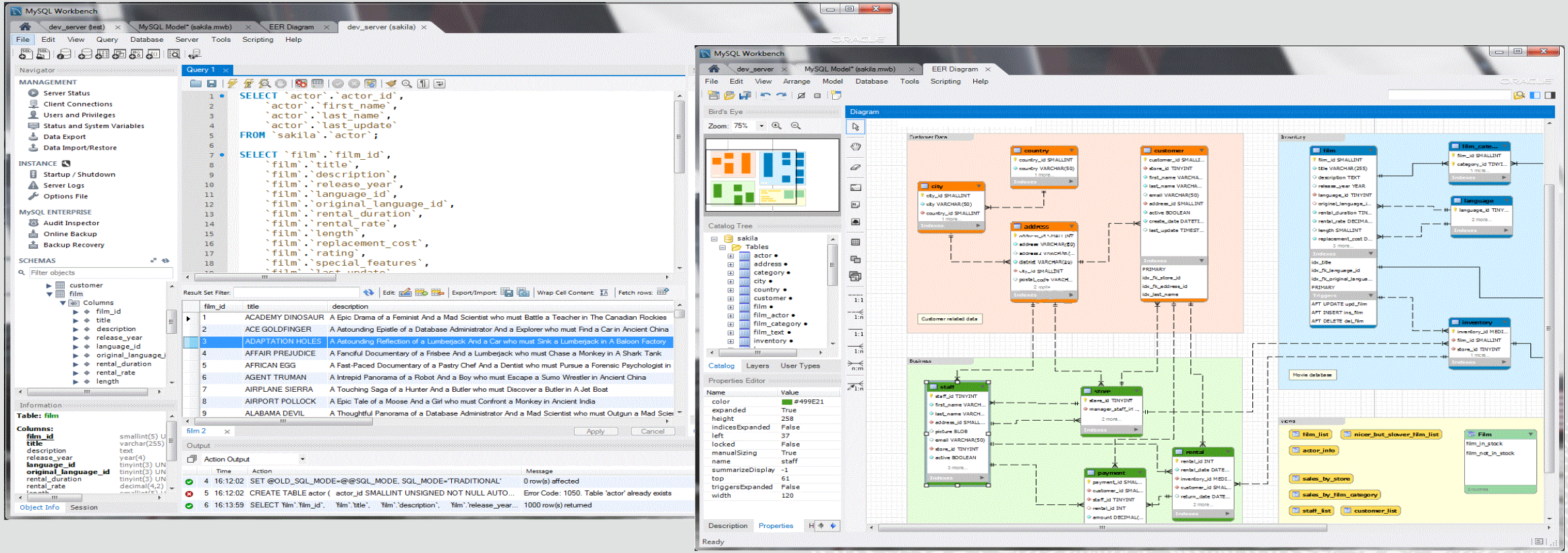
command : copy-back

options :

--defaults-file : define where the datadir and copy back copies data back to the folder

--backup-dir : The backup directory

Access Your Data and Data Model Using MySQL Workbench EE



MySQL Performance Tuning

Speed up MySQL Performance

Copyright © 2020 Oracle and/or its affiliates.

MySQL 8.0 Highlights

How to access MOS(My Oracle Support Knowledge Base)

ORACLE MY ORACLE SUPPORT | PowerView is Off | Chandan (Available) | (0)

Dashboard | Knowledge | Service Requests | Patches & Updates | Community | Certifications | Managed Cloud | CRM On Demand | More...

KM Search Results | Give Feedb

Knowledge Base | Archive | Documentation

performance | Search Additional Oracle Sites

MySQL Server | Any Product Version | Any Platform | Last updated - Any time

Results: Knowledge Base

- Apr 16, 2017 — Some Queries Show Degraded Performance After Upgrading MySQL Server (Doc ID 2223735.1)
  Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server
- Feb 1, 2017 — What are the Recommended Configuration of the Performance Schema? (Doc ID 2229601.1)
  Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server
- Sep 14, 2015 — Impact of InnoDB Buffer Pool Misses on Overal InnoDB Performance (Doc ID 2055625.1)
  Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server
- Mar 6, 2017 — Poor InnoDB performance when "History List" is long (Doc ID 1482378.1)
  Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server
- Nov 28, 2013 — Autosized Performance Schema Options in MySQL Server in MySQL 5.6 (Doc ID 1604225.1)
  Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server
- Dec 4, 2013 — When to Expect Performance Boost from Introducing Partitioning into Existing System (Doc ID 1605261.1)
  Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server
- Oct 28, 2015 — How To Use the MySQL Workbench Performance Reports? Sys Schema (Doc ID 2071907.1)
  Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server
- Mar 9, 2016 — How Does the sys Schema Affect Performance of MySQL Server? (Doc ID 2115053.1)
  Refine to All > Oracle Database Products > MySQL > MySQL Monitor > MySQL Enterprise Monitor
- Apr 2, 2017 — MySQL Performance Periodically Slows Down when using Certain RAID Hardware Controllers (Doc ID 1462128.1)
  Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server

How to access MOS(My Oracle Support Knowledge Base)

ORACLE MY ORACLE SUPPORT | PowerView is Off | Chandan (Available)

Dashboard | Knowledge | Service Requests | Patches & Updates | Community | Certifications | Managed Cloud | CRM On Demand | More...

KM Search Results

Knowledge Base | Archive | Documentation

replication lag | Search Additional Oracle Sites

Start typing to select Product | Any Product Version | Any Platform | Last updated - Any time

Results: Knowledge Base

Apr 11, 2018
MySQL Replication Falling Behind (Lag, Delay) Notes (Doc ID 2256128.1)
Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server

Apr 25, 2016
Setup Global Data Services With GoldenGate Hot standby (Doc ID 2130663.1)
Refine to All > Middleware > Data Integration > GoldenGate > Oracle GoldenGate

Oct 7, 2016
Oracle Key Manager (OKM) - Key Not Found During Restore Because Replication Operation is Running Very Slowly (Doc ID 2190877.1)
Refine to All > Sun Microsystems > Storage - Tape > Encryption (KMS) > Sun StorageTek Crypto Key Management System

Jan 8, 2016
Oracle Enterprise Manager Metric Extension for Replication s Minutes from Last Sync for Oracle ZFS Storage Appliance (Doc ID 2085813.1)
Refine to All > Sun Microsystems > Storage - Disk > Unified Storage > Sun ZFS Storage 7320

Oct 31, 2018
How to Determine Whether a Replication Slave is Behind in Execution to Its Master and if so How Much Behind? (Doc ID 1023278.1)
Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server

Feb 20, 2019
Master Note for MySQL Replication (Doc ID 2051432.1)
Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server

Oct 31, 2018
Replication Monitoring with the Performance Schema (Doc ID 2467891.1)
Refine to All > Oracle Database Products > MySQL > MySQL Server > MySQL Server

Aug 4, 2018
Catalog: Oracle Toolbox: Diagnostic and Analyzer Scripts - Data Collectors - Health Checks - Tools - Utilities - SRDC - All Products - Datab
Fusion - Middleware - Siebel - Sun - Peoplesoft (Doc ID 1987483.2)
Refine to All > Siebel > Customer Relationship Management > CRM - Enterprise Edition > Siebel Tools

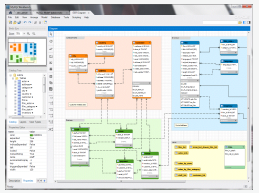
Golden Gate Replication Not Working Due To Lag Above Tolerance (Doc ID 2308680.1)
Refine to All > Oracle Cloud > Oracle Software Cloud > Oracle Retail Cloud > Oracle Retail Merchandising Foundation Cloud Service

Customer : FAQ on MySQL Performance

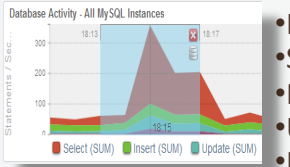
Performance

- Due to Bad SQL Queries
- Due to Bad MySQL Configurations
- Due to Concurrent Load
- Due to H/W Resources

MySQL Enterprise Edition



Workbench
- Model
- Data
- Audit Data
- User Management



Enterprise Monitor
- Identifies Vulnerabilities
- Security hardening policies
- Monitoring & Alerting
- User Monitoring
- Password Monitoring
- Schema Change Monitoring
- ackup Monitoring

Enterprise Thread Pool
- Attack minimization

Technical Support
Consultative Support

Knowledge Base Documents

Query Tuning advise

- Avoid **SubQuery/correlated** subquery. >> Convert to Joins .
- Avoid Cursors /while loop. >> Use temp table (index temp table) in conjunction with table joins
- Run **Explain Plan** and find the whether
  - Any index is there or not ?
  - Is this index is really right for this query. Use index hints (force index , use index, ignore index)
  - In MySQL 8.0:- **INVISIBLE INDEX , INVISIBLE Columns**
  - **Alter table INVISIBLE INDEX**
  - **Sql statements**
  - **Alter enable index;**
- Create index based on **Cardinality** of column.
- **Avoid MySQL Views with** table joining's because it will not use index for the table.

Query Tuning advise

- **[MySQL 8.0]**Use **Windows functions/CTE.** >> to rewrite query to improve the performance.
- **[MySQL 8.0]**Use **Lateral Derived Table.** >> to rewrite query to improve the performance.
- B-tree index **may become fragmented** with high OLTP load which hamper performance.
  - **OPTIMIZE TABLE** <tablename>; >>will actually reorganize the data inside the data file. Reclaim space.
  - **ANALYZE TABLE** <tablename>; >>will update the index statistics for the table.
- Try to have **only required columns on select queries**.

  - In most of scenario we observed that, select query fetching un-necessary columns even though application don't need it by avoiding this we can save more memory.

Understanding EXPLAIN PLAN

- **Cost: 4070 * 239 = 972,730**

```
mysql> explain SELECT * FROM city1 WHERE CountryCode IN (SELECT Code FROM country1 WHERE Code LIKE 'USA');
+----+--------------+-------------+------------+------+---------------+------+---------+------+------+----------+-----------------------------------------
--------+
| id | select_type  | table       | partitions | type | possible_keys | key  | key_len | ref  | rows | filtered | Extra
       |
+----+--------------+-------------+------------+------+---------------+------+---------+------+------+----------+-----------------------------------------
--------+
|  1 | SIMPLE       | <subquery2> | NULL       | ALL  | NULL          | NULL | NULL    | NULL | NULL |   100.00 | NULL
       |
|  1 | SIMPLE       | city1       | NULL       | ALL  | NULL          | NULL | NULL    | NULL | 4070 |    10.00 | Using where; Using join buffer (Block Nest
d Loop) |
|  2 | MATERIALIZED | country1    | NULL       | ALL  | NULL          | NULL | NULL    | NULL |  239 |    11.11 | Using where
       |
+----+--------------+-------------+------------+------+---------------+------+---------+------+------+----------+-----------------------------------------
--------+
3 rows in set, 1 warning (0.00 sec)
```

Does This Query Optimized ?

- **Look for table , type ,pssible_keys ,key**

O/P says- No Index , Subquery , Using Join Buffer

Table 1: 4070
Table 2: 239
1 *239 =239
1*239

Let's add index and see execution plan

Understanding EXPLAIN PLAN

- **Cost: 17 * 239 = 4063**

```
mysql> CREATE INDEX IX_city on city1(CountryCode);
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> explain SELECT * FROM city1 WHERE CountryCode IN (SELECT Code FROM country1 WHERE Code LIKE 'USA');
+----+--------------+-------------+------------+------+---------------+---------+---------+------------------+------+----------+-------------+
| id | select_type  | table       | partitions | type | possible_keys | key     | key_len | ref              | rows | filtered | Extra       |
+----+--------------+-------------+------------+------+---------------+---------+---------+------------------+------+----------+-------------+
|  1 | SIMPLE       | <subquery2> | NULL       | ALL  | NULL          | NULL    | NULL    | NULL             | NULL |   100.00 | NULL        |
|  1 | SIMPLE       | city1       | NULL       | ref  | IX_city       | IX_city | 3       | <subquery2>.Code |   17 |   100.00 | NULL        |
|  2 | MATERIALIZED | country1    | NULL       | ALL  | NULL          | NULL    | NULL    | NULL             |  239 |    11.11 | Using where |
+----+--------------+-------------+------------+------+---------------+---------+---------+------------------+------+----------+-------------+
3 rows in set, 1 warning (0.00 sec)

mysql>
```

Does This Query Optimized ?

- **Look for table , type ,pssible_keys ,key**

Query looks better than previous plan

But still , sub query , country1 is still not indexed

Let's add index and see execution plan

Understanding EXPLAIN PLAN

- **Cost: 17 * 1 = 17**

```
mysql> CREATE INDEX IX_country1 on country1(Code);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> explain SELECT * FROM city1 WHERE CountryCode IN (SELECT Code FROM country1 WHERE Code LIKE 'USA');
+----+--------------+-------------+------------+-------+---------------+-------------+---------+------------------+------+----------+-----------------------
--+
| id | select_type  | table       | partitions | type  | possible_keys | key         | key_len | ref              | rows | filtered | Extra
  |
+----+--------------+-------------+------------+-------+---------------+-------------+---------+------------------+------+----------+-----------------------
--+
|  1 | SIMPLE       | <subquery2> | NULL       | ALL   | NULL          | NULL        | NULL    | NULL             | NULL |   100.00 | NULL
  |
|  1 | SIMPLE       | city1       | NULL       | ref   | IX_city       | IX_city     | 3       | <subquery2>.Code |   17 |   100.00 | NULL
  |
|  2 | MATERIALIZED | country1    | NULL       | range | IX_country1   | IX_country1 | 3       | NULL             |    1 |   100.00 | Using where; Using in
x |
+----+--------------+-------------+------------+-------+---------------+-------------+---------+------------------+------+----------+-----------------------
--+
3 rows in set, 1 warning (0.00 sec)

mysql>
```

Does This Query Optimized ?

- **Look for table , type ,pssible_keys ,key**

Query looks better than previous plan

But still , sub query

Let's re-write query and see execution plan

Understanding EXPLAIN PLAN

- **Cost: 17 * 1 = 17**

```
mysql> explain  SELECT city1.* FROM city1 join country1 on city1.CountryCode =country1.Code  WHERE country1.Code LIKE 'USA';
+----+-------------+----------+------------+-------+---------------+-------------+---------+---------------------+------+----------+-----------------------
-+
| id | select_type | table    | partitions | type  | possible_keys | key         | key_len | ref                 | rows | filtered | Extra
 |
+----+-------------+----------+------------+-------+---------------+-------------+---------+---------------------+------+----------+-----------------------
-+
|  1 | SIMPLE      | country1 | NULL       | range | IX_country1   | IX_country1 | 3       | NULL                |    1 |   100.00 | Using where; Using ind
 |
|  1 | SIMPLE      | city1    | NULL       | ref   | IX_city       | IX_city     | 3       | world.country1.Code |   17 |   100.00 | NULL
 |
+----+-------------+----------+------------+-------+---------------+-------------+---------+---------------------+------+----------+-----------------------
-+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

Does This Query Optimized ?

- **Look for table , type ,pssible_keys ,key**

Query looks Perfact!!

Understanding EXPLAIN PLAN

- **Cost: 17 * 1 = 17**

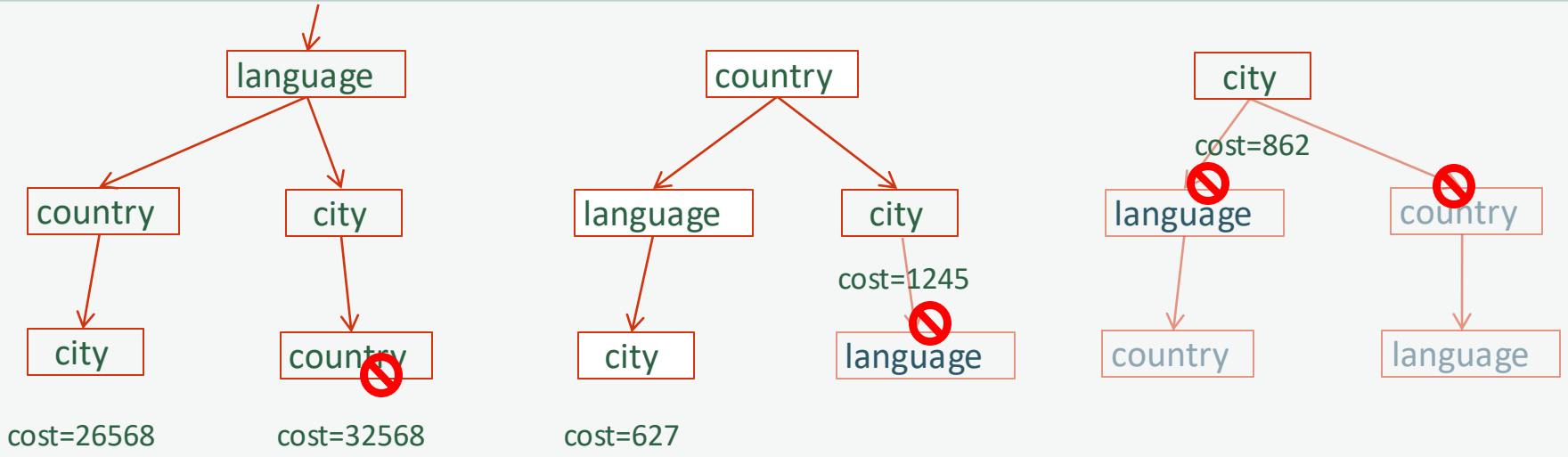
```
mysql> explain SELECT * FROM city1 WHERE CountryCode IN (SELECT Code FROM country1 WHERE Code LIKE 'USA');
+----+--------------+-------------+------------+-------+---------------+-------------+---------+------------------+------+----------+-----------------------
--+
| id | select_type  | table       | partitions | type  | possible_keys | key         | key_len | ref              | rows | filtered | Extra
  |
+----+--------------+-------------+------------+-------+---------------+-------------+---------+------------------+------+----------+-----------------------
--+
|  1 | SIMPLE       | <subquery2> | NULL       | ALL   | NULL          | NULL        | NULL    | NULL             | NULL |   100.00 | NULL
  |
|  1 | SIMPLE       | city1       | NULL       | ref   | IX_city       | IX_city     | 3       | <subquery2>.Code |   17 |   100.00 | NULL
  |
|  2 | MATERIALIZED | country1    | NULL       | range | IX_country1   | IX_country1 | 3       | NULL             |    1 |   100.00 | Using where; Using inde
x |
+----+--------------+-------------+------------+-------+---------------+-------------+---------+------------------+------+----------+-----------------------
--+
3 rows in set, 1 warning (0.00 sec)

mysql>
mysql> explain  SELECT city1.* FROM city1 join country1 on city1.CountryCode =country1.Code  WHERE country1.Code LIKE 'USA';
+----+-------------+----------+------------+-------+---------------+-------------+---------+--------------------+------+----------+-----------------------
-+
| id | select_type | table    | partitions | type  | possible_keys | key         | key_len | ref                | rows | filtered | Extra
 |
+----+-------------+----------+------------+-------+---------------+-------------+---------+--------------------+------+----------+-----------------------
-+
|  1 | SIMPLE      | country1 | NULL       | range | IX_country1   | IX_country1 | 3       | NULL               |    1 |   100.00 | Using where; Using index
 |
|  1 | SIMPLE      | city1    | NULL       | ref   | IX_city       | IX_city     | 3       | world.country1.Code |   17 |   100.00 | NULL
 |
+----+-------------+----------+------------+-------+---------------+-------------+---------+--------------------+------+----------+-----------------------
-+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

EXPLAIN – ACCESS TYPE

- *system*: the table contains excatly 1 row
- *const*: at most 1 row is matched for the table
- *eq_ref*: the table is the right-hand table in a join where the condition is on a PK or not null unique key.
- *ref*: the table is filtered by a nonunique secondary index.
- *ref_or_null*: the same as *ref* but the filtered column may also be `NULL`.
- *index_merge*: the Optimizer chooses a combination of two or more indexes to resolve a filter that includes an `OR` or `AND` between columns in different indexes.
- *fulltext*: use of a full text index to filter the table.
- *range*: this is used when an index is used to look up several values either in sequence or in groups.

Join Optimizer Illustrated

```
SELECT city.name AS capital, language.name
FROM city
  JOIN country ON city.country_id = country.country_id
  JOIN language ON country.country_id = language.country_id
WHERE city.city_id = country.capital
```



Influencing the Optimizer

When the optimizer does not do what you want

Add indexes

Force use of specific indexes:

- USE INDEX, FORCE INDEX, IGNORE INDEX

Force specific join order:

- STRAIGHT_JOIN

Adjust session variables

- optimizer_switch flags: set optimizer_switch="index_merge=off"
- Buffer sizes: set sort_buffer=8*1024*1024;
- Other variables: set optimizer_search_depth = 10;

Understanding Explain

EXPLAIN SELECT SUM(o_totalprice) FROM orders WHERE o_orderdate BETWEEN '1994-01-01' AND '1994-12-31';

id	select type	table	type	possible keys	key	key len	rows	filtered	Extra
1	SIMPLE	orders	ALL	i_o_orderdate	NULL	NULL	15000000	29.93	Using where

Cost Model Example

EXPLAIN SELECT SUM(o_totalprice) FROM orders
WHERE o_orderdate BETWEEN '1994-01-01' AND '1994-12-31';

id	select type	table	type	possible keys	key	key len	rows	filtered	Extra
1	SIMPLE	orders	ALL	i_o_orderdate	NULL	NULL	15000000	29.93	Using where

EXPLAIN SELECT SUM(o_totalprice) FROM orders
WHERE o_orderdate BETWEEN '1994-01-01' AND '1994-06-30';

Id	select type	table	type	possible keys	key	key len	rows	filtered	Extra
1	SIMPLE	orders	range	i_o_orderdate	i_o_order date	4	2235118	100.00	Using index condition

Cost Model Example: Optimizer Trace

join_optimization / row_estimation / table : orders / range_analysis

```
"table_scan": {
  "rows": 15000000,
  "cost": 3.12e6
} /* table_scan */,

"potential_range_indices": [
  {
    "index": "PRIMARY",
    "usable": false,
    "cause": "not_applicable"
  },
  {
    "index": "i_o_orderdate",
    "usable": true,
    "key_parts": [ "o_orderDATE", "o_orderkey" ]
  }

] /* potential_range_indices */,

...
```

```
"analyzing_range_alternatives": {

  "range_scan_alternatives": [
    {
      "index": "i_o_orderdate",
      "ranges": [ "1994-01-01 <= o_orderDATE <= 1994-12-31"
      ],
      "index_dives_for_eq_ranges": true,
      "rowid_ordered": false,
      "using_mrr": false,
      "index_only": false,
      "rows": 4489990,
      "cost": 5.39e6,
      "chosen": false,
      "cause": "cost"
    }

  ] /* range_scan_alternatives */,

  ...

} /* analyzing_range_alternatives */
```

Cost Model vs Real World

Measured Execution Times

	Data in Memory	Data on Disk	Data on SSD
Table scan	6.8 seconds	36 seconds	15 seconds
Index scan	**5.2 seconds**	2.5 hours	30 minutes

Force Index Scan:

```
SELECT SUM(o_totalprice)
FROM orders FORCE INDEX (i_o_orderdate)
WHERE o_orderdate BETWEEN '1994-01-01' AND '1994-12-31';
```

Performance Schema

Disk I/O

```
SELECT event_name, count_read, avg_timer_read/1000000000.0 "Avg Read Time (ms)",
        sum_number_of_bytes_read   "Bytes Read"
FROM performance_schema.file_summary_by_event_name
WHERE event_name='wait/io/file/innodb/innodb_data_file';
```

Table Scan

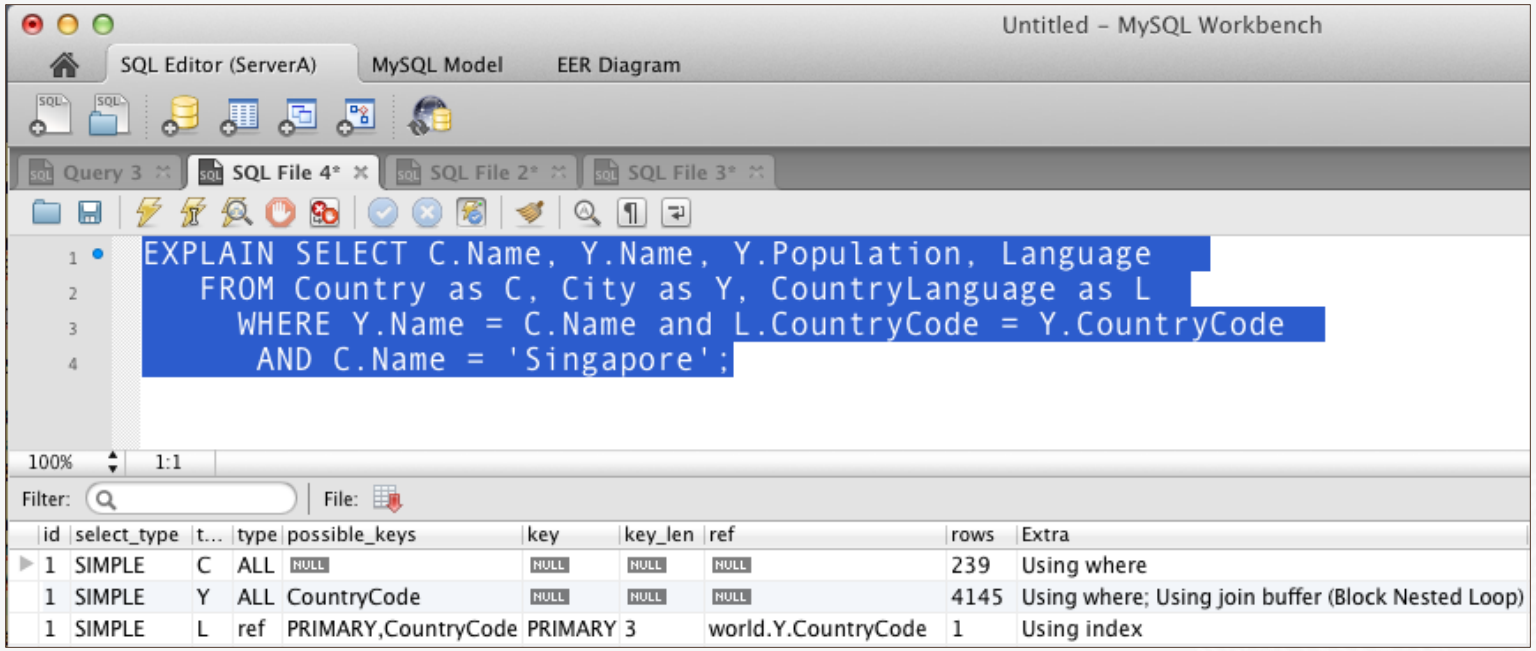
event_name	count_read	Avg Read Time (ms)	Bytes Read
wait/io/file/innodb/innodb_data_file	115769	0.0342	1896759296

Index Scan

event_name	count_read	Avg Read Time (ms)	Bytes Read
wait/io/file/innodb/innodb_data_file	2188853	4.2094	35862167552

What is Explain Plan ?

- It is the powerful tool for understanding and optimizing mysql SQL statements.



id	select_type	t...	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	C	ALL	NULL	NULL	NULL	NULL	239	Using where
1	SIMPLE	Y	ALL	CountryCode	NULL	NULL	NULL	4145	Using where; Using join buffer (Block Nested Loop)
1	SIMPLE	L	ref	PRIMARY,CountryCode	PRIMARY	3	world.Y.CountryCode	1	Using index

- **Cost: 239 * 4145 * 1 = 990655**

Does this Query is Optimized ?

Explain with Workbench

Untitled – MySQL Workbench

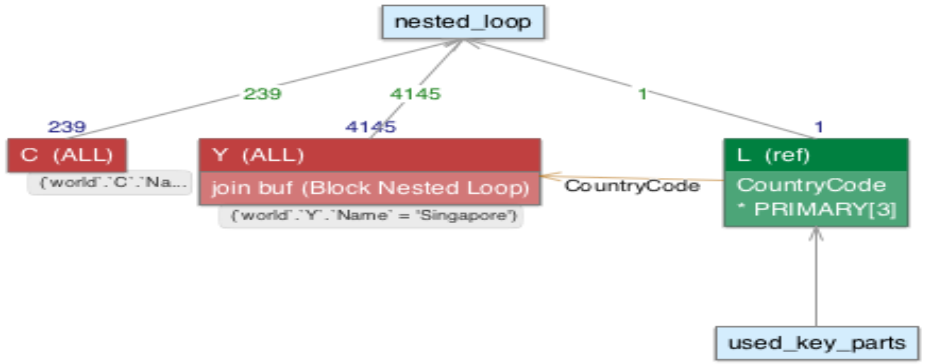
SQL Editor (ServerA) | MySQL Model | EER Diagram

Query 3 | SQL File 4* | SQL File 2* | SQL File 3*

```
SELECT C.Name, Y.Name, Y.Population, Language
   FROM Country as C, City as Y, CountryLanguage as L
     WHERE Y.Name = C.Name and L.CountryCode = Y.CountryCode
      AND C.Name = 'Singapore';
```

100% 32:4

Spacing:



Result 4 | Explain | Explain

SQL saved to snippets list.

Query Optimized

- The original cost was 239 * 4145 * 1 = 990,655
- The new cost is 1 * 1 * 1 = *1*

Untitled – MySQL Workbench

SQL Editor (ServerA) | MySQL Model | EER Diagram

Query 3 | SQL File 4* | SQL File 2* | SQL File 3*

```
EXPLAIN SELECT C.Name, Y.Name, Y.Population, Language
   FROM Country as C, City as Y, CountryLanguage as L
     WHERE Y.Name = C.Name and L.CountryCode = Y.CountryCode
      AND C.Name = 'Singapore';
```

100% 1:1

Filter: File:

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	C	ref	c2	c2	52	const	1	Using where; Using index
1	SIMPLE	Y	ref	CountryCode,c2	c2	35	const	1	Using index condition
1	SIMPLE	L	ref	PRIMARY,CountryCode	PRIMARY	3	world.Y.CountryCode	1	Using index

Type Column

Access or Join Types

Positive

- eq ref – unique key/primary to reference value
- const, system –turn part of query into constant
- Null – table or index not even accessed
- ref – match single value, non-unique index, ref_or_null = possible extra step
- range – WHERE .. BETWEEN, >

Possible Issue

- ALL table scan (depends on table size)
- INDEX (unless “using Index in EXTRA column”

http://dev.mysql.com/doc/refman/8.0/en/explain-output.htm

Extra Column

Positive	Possible Issue
▪ Using Index ▪ Using index for group by	▪ Using temporary ▪ Using filesort ▪ Using Where ▪ Good – Using Index

http://dev.mysql.com/doc/refman/8.0/en/explain-output.html#explain-extra-information

Performance Schema

Some useful tables

`events_statements_history`
`events_statements_history_long`

Most recent statements executed

`events_statements_summary_by_digest`

Summary for similar statements (same statement digest)

`file_summary_by_event_name`

Interesting event: **wait/io/file/innodb/innodb_data_file**

`table_io_waits_summary_by_table`
`table_io_waits_summary_by_index_usage`

Statistics on storage engine access per table and index

MySQL sys Schema

Example

statement_analysis: Lists a normalized statement view with aggregated statistics, ordered by the total execution time per normalized statement

```
mysql> SELECT * FROM sys.statement_analysis LIMIT 1\G
*************************** 1. row ***************************
query: INSERT INTO `mem__quan` . `nor ... nDuration` = IF ( VALUES ( ...
db: mem
full_scan: 0
exec_count: 1110067
err_count: 0
warn_count: 0
total_latency: 1.93h
max_latency: 5.03 s
avg_latency: 6.27 ms
lock_latency: 00:18:29.18
rows_sent: 0
rows_sent_avg: 0
rows_examined: 0
rows_examined_avg: 0
tmp_tables: 0
tmp_disk_tables: 0
rows_sorted: 0
sort_merge_passes: 0
digest: d48316a218e95b1b8b72db5e6b177788!
first_seen: 2014-05-20 10:42:17
```

Performance Schema -- Configuration

Enabling/Disabling Performance Schema

Within my.cnf add:

```
[mysqld]
performance_schema=on
```

Enable individual Instruments:

Within my.cnf add:

```
[mysqld]
--performance_schema_instrument='wait/synch/cond/%=counted'
```

off/false/0 = Disabled

on/true/1 = Enabled & Timed

counted = Enabled & Counted, rather than Timed

http://dev.mysql.com/doc/refman/5.5/en/performance-schema.html

Structured EXPLAIN

JSON format:

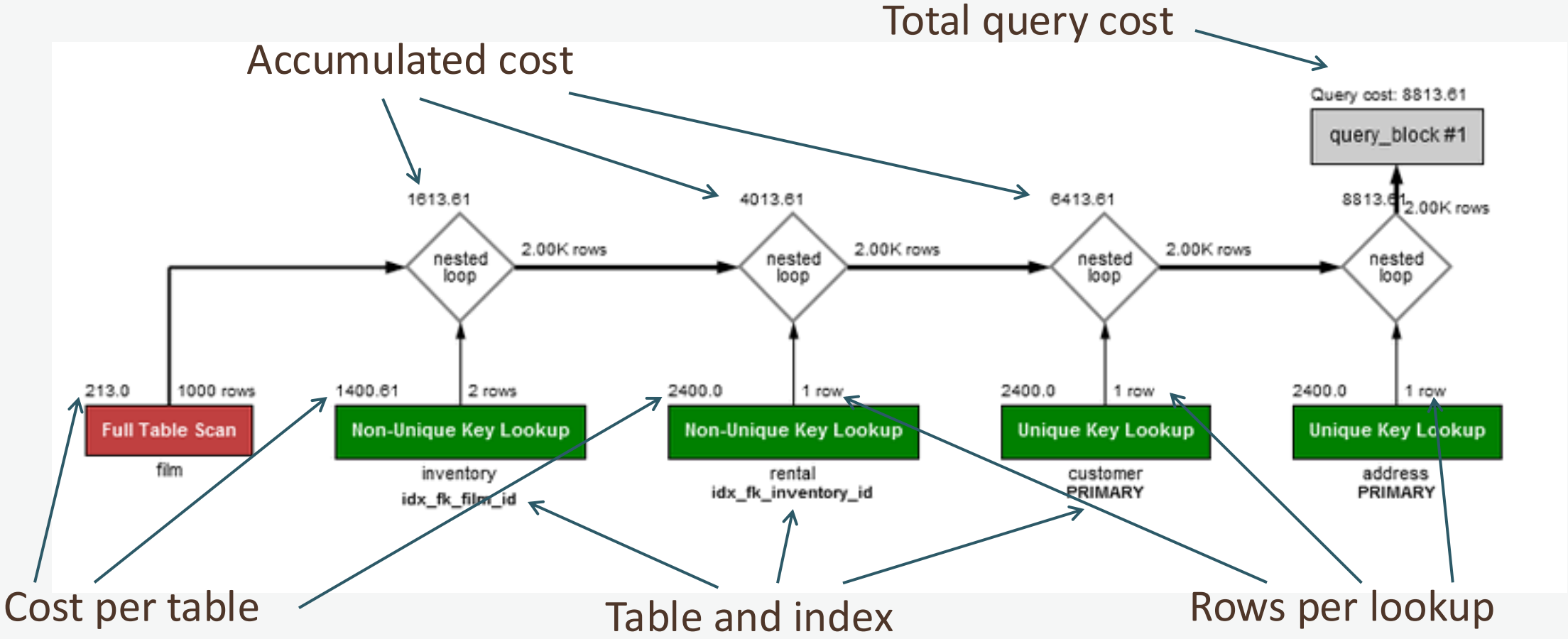
EXPLAIN FORMAT=JSON SELECT …

Contains more information:

- Used index parts
- Pushed index conditions
- Cost estimates
- Data estimates

```
EXPLAIN FORMAT=JSON
SELECT * FROM t1 WHERE b > 10 AND c > 10;
EXPLAIN
{
  "query_block": {
    "select_id": 1,
    "cost_info": {
      "query_cost": "17.81"
    },
    "table": {
      "table_name": "t1",
      "access_type": "range",
      "possible_keys": [
        "idx1"
      ],
      "key": "idx1",
      "used_key_parts": [
        "b"
      ],
      "key_length": "4",
      "rows_examined_per_scan": 12,
      "rows_produced_per_join": 3,
      "filtered": "33.33",
      "index_condition": "(`test`.`t1`.`b` > 10)",
      "cost_info": {
        "read_cost": "17.01",
        "eval_cost": "0.80",
        "prefix_cost": "17.81",
        "data_read_per_join": "63"
      },
      ........
      "attached_condition": "(`test`.`t1`.`c` > 10)"
    }
  }
}
```

Visual EXPLAIN (MySQL Workbench)



Most Common Queries

SQL File 1*

```
SELECT digest_text,count_star FROM performance_schema.events_statements_summary_by_digest
ORDER BY count_star DESC;
```

100% 1:2

Filter: Export:

digest_text	count_star
INSERT INTO t1 VALUES (...)	239378
SELECT intcol1 , charcol1 FROM t1	149949
DROP SCHEMA IF EXISTS `mysqlslap`	603
CREATE SCHEMA `mysqlslap`	600
CREATE TABLE `t1` (intcol1 INTEGER (?) , charcol1 VARCHARACTER (?))	600
SHOW TABLES	10
SELECT * FROM `setup_consumers`	8
SHOW SCHEMAS	5
SELECT SCHEMA ( )	5
SET `autocommit` = ?	5
SELECT * FROM `events_waits_history` WHERE `event_name` LIKE ?	5
SELECT `file_name` , `event_name` , `count_star` , `sum_timer_wait` FROM `file_summary_by_instance` ORDER BY...	4
SHOW SESSION VARIABLES LIKE ?	4
SELECT CURRENT_USER ( )	4
SELECT * FROM `events_statements_history` LIMIT ?	4
SELECT * FROM `setup_instruments` WHERE NAME LIKE ?	3

Last 10 Statements

SQL File 1*

```
SELECT thread_id,event_name,digest_text from performance_schema.events_statements_history LIMIT 10;
```

100% 65:1

Filter: Export: Fetch rows:

thread_id	event_name	digest_text
47054	statement/sql/select	SELECT intcol1 , charcol1 FROM t1
47054	statement/sql/select	SELECT intcol1 , charcol1 FROM t1
47054	statement/sql/insert	INSERT INTO t1 VALUES (...)
47054	statement/sql/select	SELECT intcol1 , charcol1 FROM t1
47054	statement/sql/insert	INSERT INTO t1 VALUES (...)
47055	statement/sql/select	SELECT intcol1 , charcol1 FROM t1
47055	statement/sql/select	SELECT intcol1 , charcol1 FROM t1
47055	statement/sql/insert	INSERT INTO t1 VALUES (...)
47055	statement/sql/select	SELECT intcol1 , charcol1 FROM t1
47055	statement/sql/insert	INSERT INTO t1 VALUES (...)

Files by File I/O

SQL File 1*

```sql
SELECT ps_helper.format_path(file_name) as File,
       count_read,
       ps_helper.format_bytes(sum_number_of_bytes_read) AS total_read,
       ps_helper.format_bytes(sum_number_of_bytes_write) AS total_write,
       ps_helper.format_bytes(sum_number_of_bytes_read + sum_number_of_bytes_write) AS total
  FROM performance_schema.file_summary_by_instance
 ORDER BY sum_number_of_bytes_read + sum_number_of_bytes_write DESC LIMIT 10;
```

100% 78:7

Filter: Export: Fetch rows:

File	count_read	total_read	total_write	total
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/ibdata1	530	10.25 MiB	981.14 MiB	991.39 MiB
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/mysqlslap/t1.ibd	0	0 bytes	109.80 MiB	109.80 MiB
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/mysql-bin.000014	2	120 bytes	83.94 MiB	83.94 MiB
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/ib_logfile1	0	0 bytes	78.30 MiB	78.30 MiB
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/ib_logfile0	6	68.00 KiB	77.48 MiB	77.55 MiB
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/mysql/innodb_index_stats.ibd	3	48.00 KiB	2.63 MiB	2.67 MiB
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/mysql/innodb_table_stats.ibd	3	48.00 KiB	2.53 MiB	2.58 MiB
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/mysql/proc.MYD	227	134.98 KiB	21.12 KiB	156.10 KiB
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/mysql/slave_master_info.ibd	4	64.00 KiB	0 bytes	64.00 KiB
/usr/local/mysql-5.6.7-rc-linux2.6-x86_64/data/mysql/slave_relay_log_info.ibd	4	64.00 KiB	0 bytes	64.00 KiB

Statements with Temporary Tables

SQL File 1*

```
SELECT ps_helper.format_statement(DIGEST_TEXT) AS query,
    count_star AS exec_count,
    sum_created_tmp_tables AS in_memory,
    sum_created_tmp_disk_tables AS on_disk
 FROM performance_schema.events_statements_summary_by_digest
  WHERE sum_created_tmp_tables > 0 ORDER BY sum_created_tmp_disk_tables DESC LIMIT 10;
```

100% 86:6

Filter: Export: Fetch rows:

query	exec_count	in_memory	on_disk
SHOW FULL FIELDS FROM `perform ... . `events_statements_history`	16	16	16
SHOW FULL FIELDS FROM `perform ... summary_global_by_event_name`	5	5	5
SHOW FULL FIELDS FROM `perform ... statements_summary_by_digest`	3	3	3
SHOW FULL FIELDS FROM `performance_schema` . `mutex_instances`	2	2	2
SHOW FULL FIELDS FROM `performance_schema` . `accounts`	2	2	2
CREATE VIEW `latest_file_io` A ... T NULL ORDER BY `timer_start`	1	1	1
CREATE VIEW `top_io_by_thread` ... SUM (`sum_timer_wait`) DESC	1	1	1
SHOW PROCEDURE STATUS	1	1	1
SHOW FULL FIELDS FROM `perform ... schema` . `setup_instruments`	1	1	1
SHOW PROCEDURE STATUS WHERE `Db` = ?	1	1	1

Resources

MySQL Training Course – MySQL Performance Tuning

http://education.oracle.com/pls/web_prod-plq-dad/ou_product_category.getPage?p_cat_id=159

View Performance Tuning Webinars

http://www.mysql.com/news-and-events/on-demand-webinars/

MySQL Performance Forum

http://forums.mysql.com/list.php?24

Download MySQL 8.0

http://www.mysql.com/downloads/mysql/

Try MySQL Enterprise Monitor:

http://www.mysql.com/trials/

Benchmarking MySQL

- Make your own
  - Can use general query log output
  - JMeter, LoadRunner, Visual Studio

mysqlslap http://dev.mysql.com/doc/refman/5.6/en/mysqlslap.html

supersmack http://vegan.net/tony/supersmack/

mybench http://jeremy.zawodny.com/mysql/mybench/

SysBench http://sysbench.sourceforge.net/

DBT2 http://osdldbt.sourceforge.net/

Indexing in MySQL

Faster access, Filtering and Sorting

Multiple Columns

CREATE INDEX IDX ON TAB1(col1,col2,col3) ;

Prefix Index:

CHAR, VARCHAR, BINARY, and VARBINARY

CREATE INDEX PRE ON TAB1(COL1(10)) ;

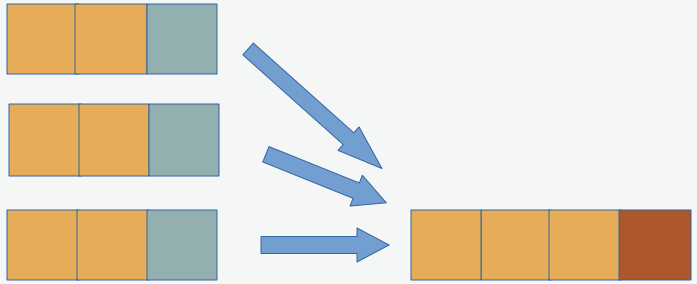
- Prefix is in bytes, not characters
- Very useful for large strings
- Works best when leading part of column is selective

MySQL 8 - Window Functions and CTEs(Common Table Expression)

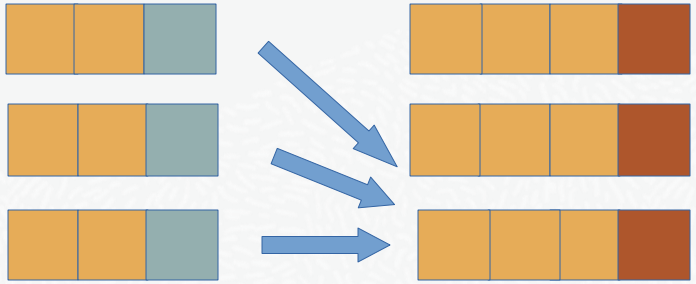
A window function performs a calculation across a set of rows that are related to the current row, similar to an aggregate function.

But unlike aggregate functions, a window function does not cause rows to become grouped into a single output row.

Window functions can access values of other rows “in the vicinity” of the current row.



Aggregate function



Window function

Copyright © 2020 Oracle and/or its affiliates.

Challenges:- Performance Bottleneck

[Customer From BFSI]:- On BOD , heavy data is getting populated from history table to main table to generate reports based on populated data. and once data moved to master table purge happened to history table on daily basis.

Below query tracked from ***slow query log,*** which is used on daily basis to move data from history table to main table.

How to tune below SQL Statement for optimal performance, suggest possible solutions?

Challenges:- Performance Bottleneck

```
INSERT INTO `tbl_ServiceRequestMaster_BeforeMySQL8`
        (
        ProblemId           ,Title              ,TechnicianName     ,
        CategoryId          ,OldState           ,CurrentState       ,
        LastUpdateTime      ,LocationId
        )

  SELECT DISTINCT
        A.RequestId           ,A.Title                ,A.ActionPerformedBy        ,
        A.CategoryId          ,A.OldStatename         ,A.NewStatename             ,
        A.LastUpdateTime      ,A.LocationId
  FROM  tbl_ServiceRequestHistory A ,(SELECT RequestId,CategoryId,MAX(LastUpdateTime) AS CurrentTime
                                      FROM tbl_ServiceRequestHistory
                                      GROUP BY RequestId,CategoryId
                                     )B
  WHERE A.RequestId=B.RequestId
    AND A.LastUpdateTime=B.CurrentTime
    AND A.CategoryId=B.CategoryId;
```

Understanding Query Pattern

Col1	Col2	Col3	Col4	Col5	Col6	Col7

Col1 → RequestId
Col3 → CategoryId
COl5 → LastUpdateTime

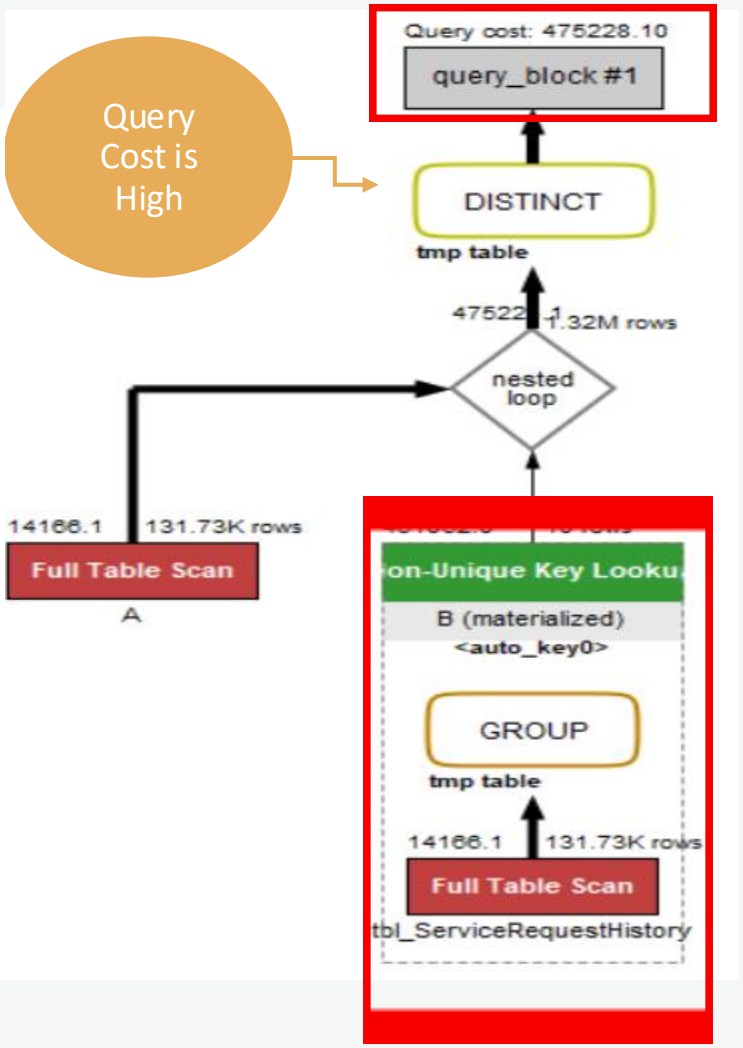
Col1	Col2	Col3	Col4	Col5	Col6	Col7

Max value record of *lastupdatetime column* based on *requested , categoryid*

Further Analysis…

Lets Generate Execution plan for the query

```
INSERT INTO `tbl_ServiceRequestMaster_BeforeMySQL8`
        (
        ProblemId           ,Title              ,TechnicianName     ,
        CategoryId          ,OldState           ,CurrentState       ,
        LastUpdateTime      ,LocationId
        )

  SELECT DISTINCT
        A.RequestId           ,A.Title              ,A.ActionPerformedBy        ,
        A.CategoryId          ,A.OldStatename       ,A.NewStatename             ,
        A.LastUpdateTime      ,A.LocationId
  FROM  tbl_ServiceRequestHistory A ,(SELECT RequestId,CategoryId,MAX(LastUpdateTime) AS CurrentTime
                                      FROM tbl_ServiceRequestHistory
                                      GROUP BY RequestId,CategoryId
                                     )B
  WHERE A.RequestId=B.RequestId
    AND A.LastUpdateTime=B.CurrentTime
    AND A.CategoryId=B.CategoryId;
```



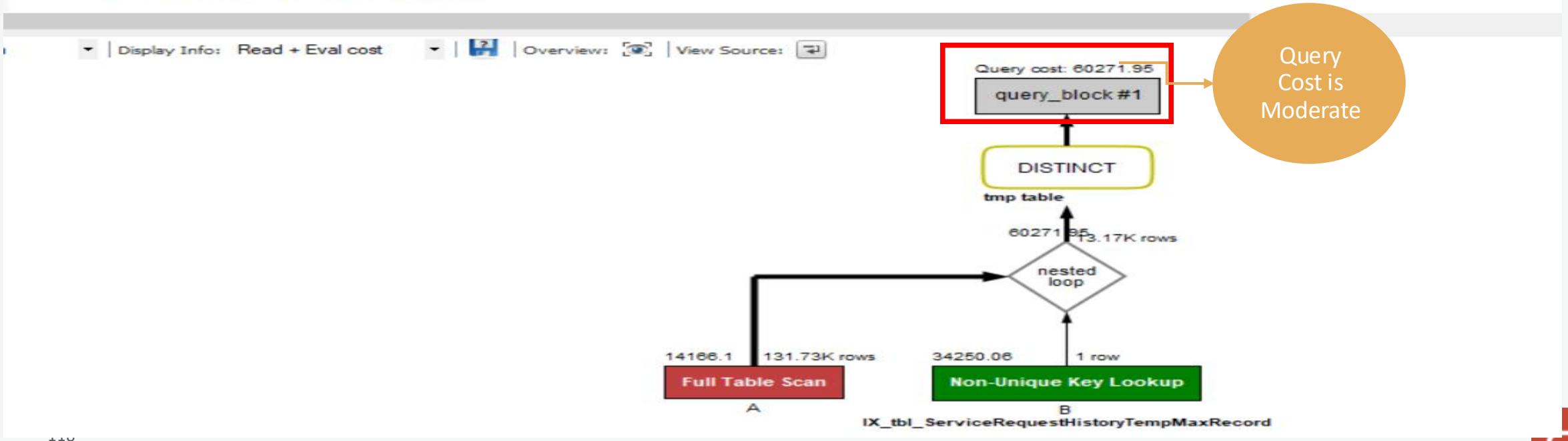
Possible Workaround

Step 1:-Create temporary table for derived table, add index on RequestId,CategoryId

Step 2:- See again explain plan to improve the performance of query.

```
  DROP TEMPORARY TABLE IF EXISTS tbl_ServiceRequestHistoryTempMaxRecord;
CREATE TEMPORARY TABLE tbl_ServiceRequestHistoryTempMaxRecord
AS
SELECT RequestId,CategoryId,MAX(LastUpdateTime) AS CurrentTime FROM tbl_ServiceRequestHistory GROUP BY RequestId,CategoryId;

CREATE INDEX IX_tbl_ServiceRequestHistoryTempMaxRecord On tbl_ServiceRequestHistoryTempMaxRecord(RequestId,CategoryId);
```

```
      SELECT DISTINCT
             A.RequestId              ,A.Title                    ,A.ActionPerformedBy          ,
             A.CategoryId             ,A.OldStatename             ,A.NewStatename               ,
             A.LastUpdateTime         ,A.LocationId
      FROM   tbl_ServiceRequestHistory A ,tbl_ServiceRequestHistoryTempMaxRecord B
      WHERE  A.RequestId=B.RequestId
        AND  A.LastUpdateTime=B.CurrentTime
        AND  A.CategoryId=B.CategoryId;
```



Best Solutions

```
INSERT INTO `tbl_ServiceRequestMaster_AfterMySQL8`
            (
            ProblemId              ,Title                  ,TechnicianName      ,
            CategoryId             ,OldState               ,CurrentState        ,
            LastUpdateTime         ,LocationId
            )
 SELECT
            A.RequestId               ,A.Title                    ,A.ActionPerformedBy          ,
            A.CategoryId              ,A.OldStatename             ,A.NewStatename               ,
            A.LastUpdateTime          ,A.LocationId

  FROM
    (
        SELECT  *,
                ROW_NUMBER() OVER (PARTITION BY RequestId,CategoryId ORDER BY LastUpdateTime DESC)as Rownum
          FROM tbl_ServiceRequestHistory
    ) A
 WHERE A.Rownum=1;
```

ROW_NUMBER() causes to generate unique row number for each partition (group by) based on ORDER BY clause. to decide which one row will be first and then when data of partition column will change again fresh unique row number will be generated per partition

Summary

Before

```
INSERT INTO `tbl_ServiceRequestMaster_BeforeMySQL8`
        (
        ProblemId           ,Title              ,TechnicianName     ,
        CategoryId          ,OldState           ,CurrentState       ,
        LastUpdateTime      ,LocationId
        )

  SELECT DISTINCT
        A.RequestId           ,A.Title              ,A.ActionPerformedBy        ,
        A.CategoryId          ,A.OldStatename       ,A.NewStatename             ,
        A.LastUpdateTime      ,A.LocationId
  FROM  tbl_ServiceRequestHistory A ,(SELECT RequestId,CategoryId,MAX(LastUpdateTime) AS CurrentTime
                                      FROM tbl_ServiceRequestHistory
                                      GROUP BY RequestId,CategoryId
                                     )B
  WHERE A.RequestId=B.RequestId
    AND A.LastUpdateTime=B.CurrentTime
    AND A.CategoryId=B.CategoryId;
```

After

```
INSERT INTO `tbl_ServiceRequestMaster_AfterMySQL8`
        (
        ProblemId           ,Title              ,TechnicianName     ,
        CategoryId          ,OldState           ,CurrentState       ,
        LastUpdateTime      ,LocationId
        )
 SELECT
        A.RequestId           ,A.Title              ,A.ActionPerformedBy        ,
        A.CategoryId          ,A.OldStatename       ,A.NewStatename             ,
        A.LastUpdateTime      ,A.LocationId

  FROM
   (
      SELECT  *,
              ROW_NUMBER() OVER (PARTITION BY RequestId,CategoryId ORDER BY LastUpdateTime DESC)a Rownum
        FROM tbl_ServiceRequestHistory
   ) A
 WHERE A.Rownum=1;
```

With Windowing you don’t need to compute aggregation statistics separately and then join /correlate back to underlying data, hence performance of query improved

Let’s Understand how window function works ?

- **Example:- Sales Table**

```
1
2 ●  SELECT * FROM SalesTable;
```

sult Grid | Filter Rows: | Export:

EmployeeName	CountryName	CityName	Revenue
A	India	Mumbai	75000
B	India	Kolkata	50000
C	India	Delhi	25000
X	Pakistan	Lahore	50000
Y	Pakistan	Karachi	20000
Z	Pakistan	Peshawar	35000

SQL Code Before MySQL 8.0

Example:- Sales Table

```sql
SELECT * FROM SalesTable;
```

sult Grid | Filter Rows: | Export:

EmployeeName	CountryName	CityName	Revenue
A	India	Mumbai	75000
B	India	Kolkata	50000
C	India	Delhi	25000
X	Pakistan	Lahore	50000
Y	Pakistan	Karachi	20000
Z	Pakistan	Peshawar	35000

Find the maximum Revenue by Country

```sql
SELECT CountryName, MAX(Revenue) Revenue
  FROM SalesTable
  GROUP BY CountryName;
```

ult Grid | Filter Rows: | Export:

CountryName	Revenue
India	75000
Pakistan	50000

That’s Easy

SQL Code Before MySQL 8.0

Example:- Sales Table

```
SELECT * FROM SalesTable;
```

sult Grid | Filter Rows: | Export:

EmployeeName	CountryName	CityName	Revenue
A	India	Mumbai	75000
B	India	Kolkata	50000
C	India	Delhi	25000
X	Pakistan	Lahore	50000
Y	Pakistan	Karachi	20000
Z	Pakistan	Peshawar	35000

Find the 1st Top SalesPerson for each country

```
SELECT EmployeeName, CountryName, CityName, Revenue FROM SalesTable
 WHERE (CountryName,Revenue) IN (
                                 SELECT CountryName , MAX(Revenue)Revenue
                                   FROM SalesTable GROUP BY CountryName
                                );
```

sult Grid | Filter Rows: | Export: | Wrap Cell Content:

EmployeeName	CountryName	CityName	Revenue
A	India	Mumbai	75000
X	Pakistan	Lahore	50000

Little Complicated

SQL Code Before MySQL 8.0

Example:- Sales Table

```sql
SELECT * FROM SalesTable;
```

sult Grid | Filter Rows: | Export:

EmployeeName	CountryName	CityName	Revenue
A	India	Mumbai	75000
B	India	Kolkata	50000
C	India	Delhi	25000
X	Pakistan	Lahore	50000
Y	Pakistan	Karachi	20000
Z	Pakistan	Peshawar	35000

Find the 2nd Top SalesPerson for each country

```sql
SELECT EmployeeName, CountryName, CityName, Revenue FROM SalesTable
WHERE (CountryName,Revenue)
   IN (
         SELECT CountryName,MAX(Revenue)Revenue  FROM SalesTable

          WHERE (CountryName,Revenue) NOT IN
                                  (
                                    SELECT CountryName , MAX(Revenue)Revenue
                                      FROM SalesTable GROUP BY CountryName
                                  )
           GROUP BY CountryName
   );
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

EmployeeName	CountryName	CityName	Revenue
B	India	Kolkata	50000
Z	Pakistan	Peshawar	35000

Hmm…More Complicated!

Problems

Difficult to generalize Top-N Per Country

Lots of Sub-queries or temporary tables or derived tables.

Impact on Performance.

Difficult to read, write and execute

Re-write query with Window Functions

Let’s go back with SalesTable

```
SELECT * FROM SalesTable;
```

sult Grid | Filter Rows: | Export:

EmployeeName	CountryName	CityName	Revenue
A	India	Mumbai	75000
B	India	Kolkata	50000
C	India	Delhi	25000
X	Pakistan	Lahore	50000
Y	Pakistan	Karachi	20000
Z	Pakistan	Peshawar	35000

Windows Function In Action-MySQL 8.0

Before MySQL 8.0

Find the 1st Top SalesPerson for each country

```
SELECT EmployeeName, CountryName, CityName, Revenue FROM SalesTable
 WHERE (CountryName,Revenue) IN (
                                 SELECT CountryName , MAX(Revenue)Revenue
                                   FROM SalesTable GROUP BY CountryName
                                );
```

sult Grid | Filter Rows: | Export: | Wrap Cell Content:

EmployeeName	CountryName	CityName	Revenue
A	India	Mumbai	75000
X	Pakistan	Lahore	50000

In MySQL 8.0

Find the 1st Top SalesPerson for each country

```
SELECT EmployeeName, CountryName, CityName, Revenue FROM
(
    SELECT *, DENSE_RANK() OVER (PARTITION BY CountryName ORDER BY Revenue DESC) AS Ranks
    FROM SalesTable
)A WHERE Ranks=1;
```

ult Grid | Filter Rows: | Export: | Wrap Cell Content:

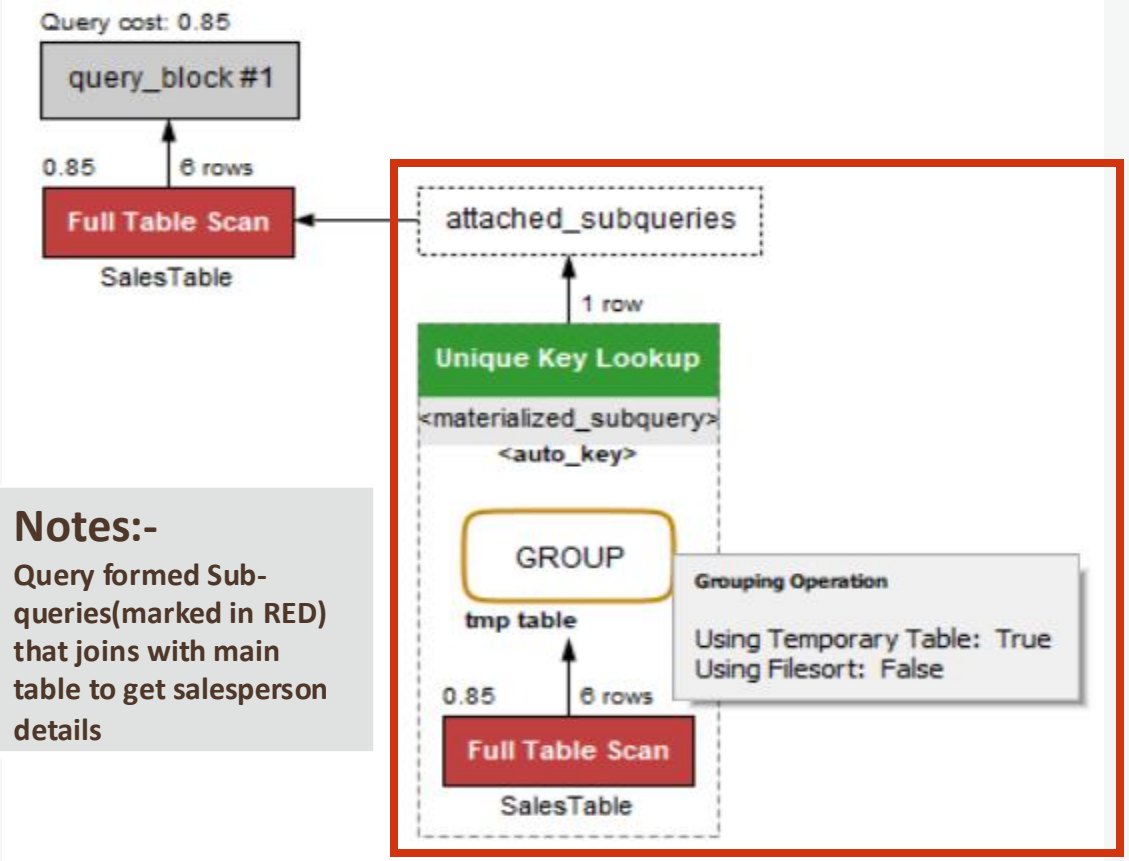
EmployeeName	CountryName	CityName	Revenue
A	India	Mumbai	75000
X	Pakistan	Lahore	50000

Window Functions

Window Function In Action-MySQL 8.0

Before MySQL 8.0

Find the 1st Top SalesPerson for each country

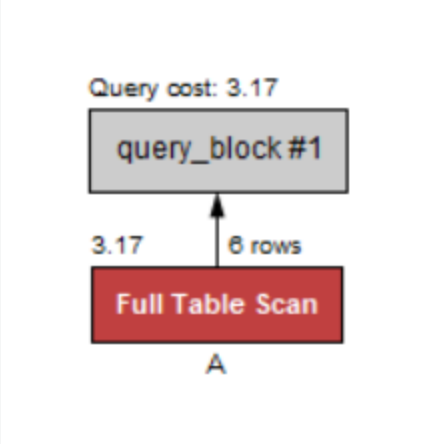


Notes:-

Query formed Sub-queries(marked in RED) that joins with main table to get salesperson details

In MySQL 8.0

Find the 1st Top SalesPerson for each country



Notes:-

Due to window functions on the fly computation for all records ,avoid extra scanning of underlying data

Window Function In Action-MySQL 8.0

Before MySQL 8.0

Find the 2nd Top SalesPerson for each country

```
SELECT EmployeeName, CountryName, CityName, Revenue FROM SalesTable
WHERE (CountryName,Revenue)
   IN (
         SELECT CountryName,MAX(Revenue)Revenue  FROM SalesTable

          WHERE (CountryName,Revenue) NOT IN
                                 (
                                   SELECT CountryName , MAX(Revenue)Revenue
                                     FROM SalesTable GROUP BY CountryName
                                  )
           GROUP BY CountryName
   );
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

EmployeeName	CountryName	CityName	Revenue
B	India	Kolkata	50000
Z	Pakistan	Peshawar	35000

In MySQL 8.0

Find the 2nd Top SalesPerson for each country

```
SELECT EmployeeName, CountryName, CityName, Revenue FROM
(
    SELECT *, DENSE_RANK() OVER (PARTITION BY CountryName ORDER BY Revenue DESC) AS Ranks
    FROM SalesTable
)A WHERE Ranks=2 ;
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

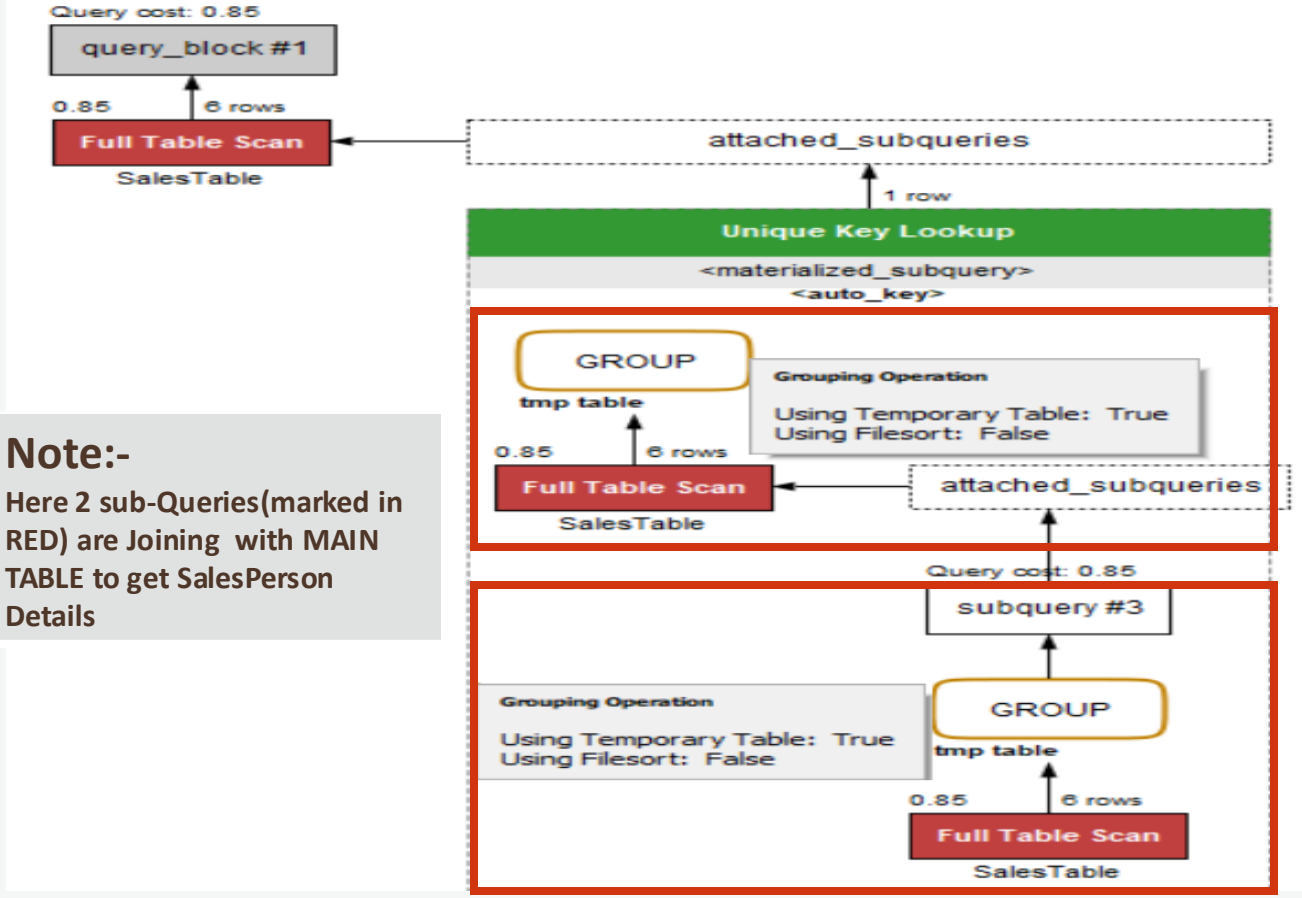
EmployeeName	CountryName	CityName	Revenue
B	India	Kolkata	50000
Z	Pakistan	Peshawar	35000

Window Functions

Window Function In Action-MySQL 8.0

Before MySQL 8.0

Find the 2nd Top SalesPerson for each country

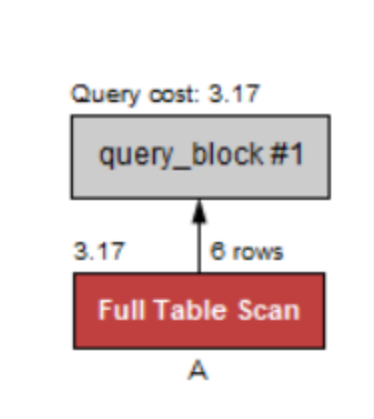


Note:-

Here 2 sub-Queries(marked in RED) are Joining with MAIN TABLE to get SalesPerson Details

In MySQL 8.0

Find the 2nd Top SalesPerson for each country



Note:-

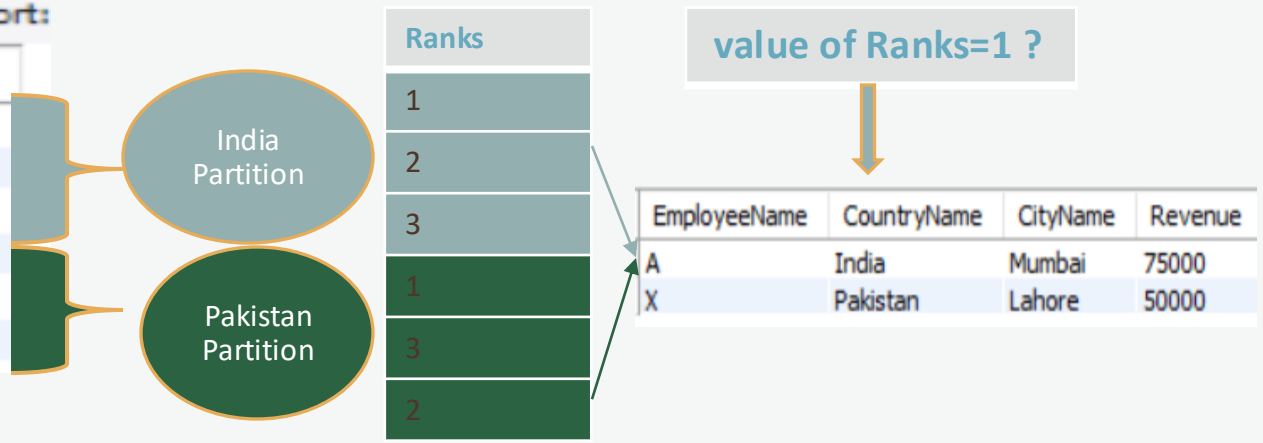
Due to window functions on the fly computation for all records ,which avoids extra scanning of underlying data

How it work?

```
SELECT * FROM SalesTable;
```

sult Grid | Filter Rows: | Export:

EmployeeName	CountryName	CityName	Revenue
A	India	Mumbai	75000
B	India	Kolkata	50000
C	India	Delhi	25000
X	Pakistan	Lahore	50000
Y	Pakistan	Karachi	20000
Z	Pakistan	Peshawar	35000



Let’s understand the query

```
SELECT EmployeeName, CountryName, CityName, Revenue FROM
(
    SELECT *,DENSE_RANK() OVER (PARTITION BY CountryName ORDER BY Revenue DESC) AS Ranks
    FROM SalesTable
)A WHERE  Ranks=1;
```

What are Window functions

Ranking Functions

- ROW_NUMBER()
- RANK()
- DENSE_RANK()
- NTILE()
- PERCENT_RANK()

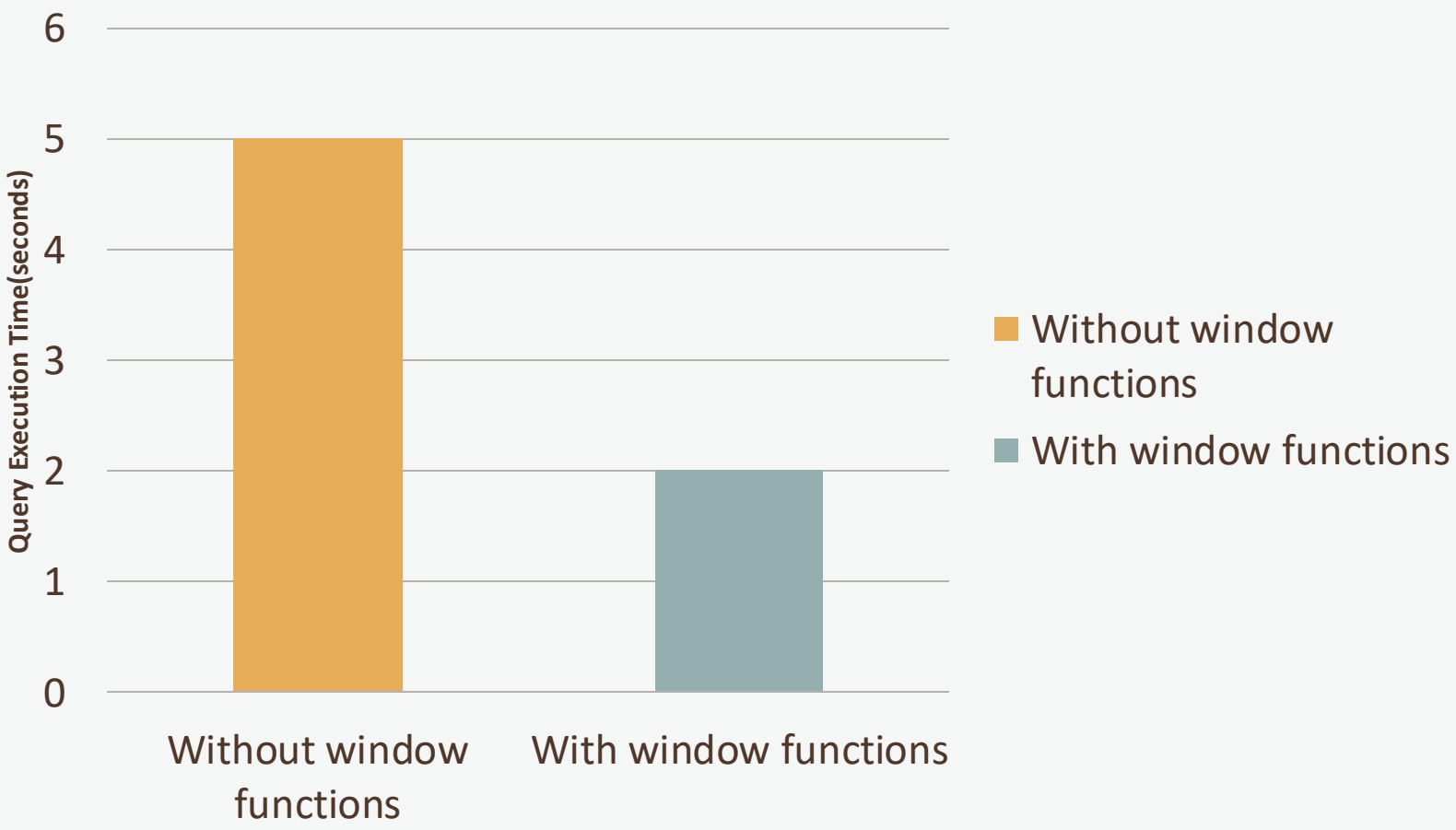
Analytics Functions

- FIRST_VALUE()
- LAST_VALUE()
- NTH_VALUE()
- LAG()
- LEAD()
- CUME_DIST()

Aggregate Functions

- COUNT()
- MAX()
- MIN()
- SUM()
- AVG()
- BIT_OR(),BIT_AND(), BIT_XOR()
- VARIANCE(),STDDEV()

Query Performance Comparison



Use case # 02 :-

Improving SQL Performance with Lateral Derived Tables

Glimpse of Lateral Derived Table

LATERAL keywords allows the sub-SELECT to refer to columns of FROM items that appear before it FROM clause.

Without **LATERAL** ,each sub-select is evaluated independently and so can't cross reference any other FROM item.

As of **8.0.14** version , a derived table can refer to(depend on) columns of preceding tables in same FROM clause.

Part of SQL:1999 standard

```
SELECT ActorName
 FROM actor A , (  SELECT actor_id
                   FROM film_actor
                   WHERE actor_id=A.actor_id
                 )B;
```

A is outside table

Error Code: 1054. Unknown column 'A.actor_id' in 'where clause

```
SELECT A.ActorName ,B.film_id
 FROM actor A ,LATERAL (SELECT actor_id ,film_id
                        FROM film_actor
                        WHERE actor_id=A.actor_id
                       )B;
```

Why do we need Lateral co-relation?

Pros

Performance Improvement

- On the fly correlation among inner & outer most tables.
- Simple execution plan.

Easily consolidate bigger business logic into simpler one.

Aggregation across several tables.

Cons

LATERNAL can't use with UDFs.

If lateral derived table references an aggregate function, the function's aggregation query can't be the one that owns the FROM clause in which the lateral derived table occurs.

Use cases for Lateral Derived table

Join a “Top-N” query to a normal table.

Ex:- find each actor and their top 10 best movies.

Reuse calculated values in same query.

Useful to avoid duplicated subquery , temporary tables.

Real Time Use Cases :- Calculate Computed columns with Lateral Joins

Let us consider stock market table

```
5 •  SELECT * FROM tbl_StockData;
```

esult Grid | Filter Rows: | Export: | Wrap

Symbol	Lastupdatetime	ClosePrice	Value	Volume
TATA	2018-01-09 11:44:22	240.34	24528000	102200
INFY	2015-02-19 10:14:22	740.04	149784096	202400
CTS	2018-06-07 10:44:27	540.37	75219504	139200
PNB	2018-03-03 15:04:22	140.86	56513032	401200
SBI	2018-01-09 11:44:22	310.74	157296588	506200

We wanted to calculate

- Average Traded Price
- '% change'
- Negative Trend
- Positive Trend

Formula:-

Average Traded Price = Value/Volume

'% change' = ROUND((Value/Volume)/ClosePrice,2)

Negative Trend =(Value/Volume) - (1-ClosePrice *((Value/Volume)/ClosePrice)

Positive Trend =(Value/Volume) + (1+ClosePrice *((Value/Volume)/ClosePrice)

Let's Compute Desired Results?

Disclaimer: Above formula is only for information purposes only

Method #1 :- Basic Query

```
Select Symbol , ClosePrice ,Value, Volume ,
 ROUND((Value/Volume),2) as 'Average Traded Price',
 ROUND(((Value/Volume)/ClosePrice),2) as '% change',
 IF((Value/Volume) - (1-ClosePrice *((Value/Volume)/ClosePrice) )<500,1,0) as 'Negative Trend',
 IF((Value/Volume)  + (1+ClosePrice *((Value/Volume)/ClosePrice) )>500,1,0) as 'Positive Trend'
From tbl_StockData;
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

Symbol	ClosePrice	Value	Volume	Average Traded Price	% change	Negative Trend	Positive Trend
TATA	240.34	24999100	102200	244.61	1.02	1	0
INFY	740.04	149774056	202400	739.99	1.00	0	1
CTS	540.37	74219434	139200	533.19	0.99	0	1
PNB	140.86	56413902	401200	140.61	1.00	1	0
SBI	310.74	153296578	506200	302.84	0.97	0	1

What a mess! This way too hard to read ,duplication of business logic etc hence impact on performance.

Method #02 :- Rewrite query with Subquery

Removing Duplication of business logic by adding subqueries.

```
select Symbol , ClosePrice ,Value, Volume,
ROUND(`% change`,2) as '% change',
  ROUND(atp,2) as 'Average Traded Price',
  IF((Atp-(1- ClosePrice*`% change`))<500,1,0) as 'Negative Trend',
  IF((Atp + (1+ClosePrice*`% change`))>500,1,0) as 'Positive Trend'
from (
  select *,
   (atp/ClosePrice) as '% change'
  from (
    select *,
      (Value/Volume) as atp
    From tbl_StockData
)AverageTradedPrice
)FinalTradedPrice;
```

Step 1

Step 2

Step 3

Above query is more efficient because it only does each stage of calculations once. But despite removing all duplication , it's arguably even harder to read.
What's next ?- There is better way with LATERAL Joins.

ult Grid | Filter Rows: | Export: | Wrap Cell Content:

Symbol	ClosePrice	Value	Volume	% change	Average Traded Price	Negative Trend	Positive Trend
TATA	240.34	24999100	102200	1.02	244.61	1	0
INFY	740.04	149774056	202400	1.00	739.99	0	1
CTS	540.37	74219434	139200	0.99	533.19	0	1
PNB	140.86	56413902	401200	1.00	140.61	1	0
SBI	310.74	153296578	506200	0.97	302.84	0	1

Method #03:- Calculate Computed columns with **Lateral Joins(only from 8.0)**

```
select Symbol , ClosePrice ,Value, Volume,
 ROUND(Atp,2) as 'Average Traded Price',
 ROUND(`% change`,2)'% change',
 IF(`Negative Trend`<500,1,0)'Negative Trend',
 IF(`Positive Trend`>500,1,0)'Positive Trend'
from tbl_StockData,
LATERAL (select (Value/Volume) as Atp) AverageTradedPrice,
LATERAL (select (atp/ClosePrice) as '% change') changes,
LATERAL (select Atp-(1- ClosePrice*`% change`) as 'Negative Trend') Negative_Trend,
LATERAL (select Atp + (1+ClosePrice*`% change`) as 'Positive Trend') Positive_Trend
```

Step 1

Step 2

Final Calculation

Easier to read business logic . Here each joined table computes a single value and then reusing them for the next calculations.
It is impossible without LATERAL JOINS
Much Cleaner, easier query

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

Symbol	ClosePrice	Value	Volume	Average Traded Price	% change	Negative Trend	Positive Trend
TATA	240.34	24999100	102200	244.61	1.02	1	0
INFY	740.04	149774056	202400	739.99	1.00	0	1
CTS	540.37	74219434	139200	533.19	0.99	0	1
PNB	140.86	56413902	401200	140.61	1.00	1	0
SBI	310.74	153296578	506200	302.84	0.97	0	1

ORACLE