

ORACLE



MySQL & NoSQL: The Best of Both Worlds

Ted Wennmark Principal Sales Consultant MySQL ted.wennmark@oracle.com

Safe Harbour 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, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

History of MySQL I

- 2001 MySQL 3.23 GA our first GA release ever!
- 2005 Oracle Corporation acquired Innobase OY
- 2008 Sun acquired MySQLAB for \$1 billion
- 2010 Oracle acquired Sun on 27 January
- 2010 MySQL 5.5 first Oracle release, great feedback from community!
- 2012 MySQL 5.6 "Best release ever"

History of MySQL II

- World's Most Popular Open Source Database
- Over 12 million product installations
- 65,000 downloads/day
- The "M" of the widely deployed LAMP stack
- MySQL Commercial Editions Available

Session Agenda

- NoSQL What are people looking for?
- RDBMS What advantages do they still have?
- How MySQL Delivers the Best of Both Worlds
 - MySQL Cluster
 - NoSQL attributes: Scale-out, performance, ease-of-use, schema flexibility, on-line operations
 - NoSQL APIs

ight © 2012, Oracle and/or its affiliates. All rights

reserved

- Key-Value store access to InnoDB (Memcached)
- What is coming with future releases



What NoSQL must deliver

- Massive scalability
 - No application-level sharding
- Performance
- High Availability/Fault Tolerance
- Ease of use
 - Simple operations/administration
 - Simple APIs
 - Quickly evolve application & schema





Still a role for the RDBMS?



- No best single solution fits all
- Mix and match

Scalability	
Performance	
НА	
Ease of use	
SQL/Joins	
ACID Transactions	

MySQL Cluster introduction

Scaling Reads & Writes	Auto-sharding + Multi-master		
	Transactional, ACID-compliant relational database		
99.999%	Shared-nothing design, no Single Point of Failure		
Availability	On-Line operations: Scale, Upgrade Schema, etc.		
Real-Time	High-load, real-time performance		
Responsiveness	Predictable low latency, bounded access times		
SQL & NoSQL APIs	Complex, relational queries + Key/Value Access		
	MySQL, Memcached, C++, Java, JPA, HTTP / REST		
Low TCO,	GPL & Commercial editions		
Open platform	Commodity hardware, management & monitoring tools		

MySQL Cluster Architecture



MySQL Cluster: Extensive Choice of NoSQL APIs



C++ example

NdbOperation *op = trx->getNdbOperation(myTable);

```
op->insertTuple();
op->equal("key", i);
op->setValue("value", &value);
```

trx->execute(NdbTransaction::Commit);



1.2 Billion UPDATEs per Minute



- NoSQL C++ API, flexaSynch benchmark
- 30 x Intel E5-2600 Intel Servers, 2 socket, 64GB
- ACID Transactions, with Synchronous Replication



13 Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Table T1



Data Node 1

Data Node 2



Data Node 3



Data Node 4





Data Node 4



ORACLE

15 Copyright © 2012, Oracle and/or its affiliates. All rights reserved.



Data Node 4













19 Copyright © 2012, Oracle and/or its affiliates. All rights reserved.



Table T1



Scalability	✓
Performance	
НА	~
Ease of use	
SQL/Joins	<
ACID Transactions	~



Scale-Out Reads & Writes on Commodity Hardware

SELECT Queries per Minute







- 8 x Commodity Intel Servers
 - 2 x 6-core processors 2.93GHz
 - x5670 processors (24 threads)
 - 48GB RAM
- Infiniband networking
- flexAsynch benchmark (NDB API)

On-line Schema changes

On-Line Operations

- Scale the cluster (add & remove nodes on-line)
- Repartition tables
- Upgrade / patch servers & OS
- Upgrade / patch MySQL Cluster
- Back-Up
- Evolve the schema on-line, in real-time





NoSQL Access to MySQL Cluster data



MySQL Cluster 7.1: ClusterJ/JPA



- New Domain Object Model Persistence API (ClusterJ) :
 - Java API
 - High performance, low latency
 - Feature rich
- JPA interface built upon this new Java layer:
 - Java Persistence API compliant
 - Implemented as an OpenJPA plugin
 - Uses ClusterJ where possible, reverts to JDBC for some operations
 - Higher performance than JDBC
 - More natural for most Java designers
 - Easier Cluster adoption for web applications

Java Access Performance



http://www.mysql.com/why-mysql/white-papers/mysql_wp_cluster_connector_for_java.php

NoSQL with Memcached (MySQL Cluster 7.2)



- Memcached is a distributed memory based hashkey/value store with no persistence to disk
- NoSQL, simple API, popular with developers
- MySQL Cluster already provides scalable, in-memory performance with NoSQL (hashed) access as well as persistence
 - Provide the Memcached API but map to NDB API calls
- Writes-in-place, so no need to invalidate cache
- Simplifies architecture as caching & database integrated into 1 tier
- Access data from existing relational tables

Traditional Memcached Architecture

Two levels of hashing









Cache miss (1): fetch from DB



Cache miss (2): manage cache



Data change (1): Write to DB





Data change (2): manage cache



34 Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

NoSQL with Memcached



- Flexible:
 - Deployment options
 - Multiple Clusters
 - Simultaneous SQL Access
 - Can still cache in Memcached server
 - Flat key-value store or map to multiple tables/columns

- set maidenhead 0 0 3 SL6 STORED
 - get maidenhead
 - VALUE maidenhead 0 3

- SL6
- END

Cluster & Memcached – Schema-Free



Cluster & Memcached - Configured Schema



Memcached with MySQL Cluster

Try it out

http://clusterdb.com/u/memcached



Memcached NoSQL Access with InnoDB



- Memcached as a plugin of MySQL Server; same process space, with very low latency access to data
- Memcapable: supports both memcached
 ascii protocol and binary protocol
- Support multiple columns: users can map multiple columns into "value"
- Optional local caching: "innodb-only", "cache-only", and "caching"
- Batch operations for performance
- Available from in MySQL 5.6

Performance



40 Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

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

InnoDB & Memcached - Configured Schema



Which API to use?

SQL

- Industry standard
- · Joins & complex queries
- · Relational model

Memcached

- simple to use API
- · key/value
- · driver for many languages
- · ideal as e.g. PHP proxy

ClusterJ

- simple to use Java API
- · Web & telco
- Object Relational Mapping
- native & fast access to MySQL Cluster

C++

- · experienced developer
- · ultra low latency / real-time

ORACLE

mod_ndb

- · REST/JSON
- HTML
- · using Apache





MySQL Cluster 7.3 DMR: Node.js NoSQL API



- Native JavaScript access to MySQL Cluster
 - End-to-End JavaScript: browser to the app and database
 - Storing and retrieving JavaScript objects directly in MySQL Cluster
 - Eliminate SQL transformation
- Implemented as a module for node.js
 - Integrates full Cluster API library within the web app
- Couple high performance, distributed apps, with high performance distributed database

What does an asynchronous API mean?



MySQL Cluster NoSQL API for Node.js

```
var nosql = require('mysql-js');
```

var annotations = new
nosql.Annotations();

annotations.mapClass(lib.Tweet,
 {'table' : 'tweet'});

```
var dbProperties =
    nosql.ConnectionProperties('ndb');
```

nosql.openSession(dbProperties, annotations, onSession);

- Modular connector with various back-end adapters:
 - ndb: low-level native access to MySQL Cluster
 - mysql: access to any MySQL server (translates operations to SQL statements)



MySQL Cluster NoSQL API for Node.js

```
var onSession = function(err,
session) {
```

```
if (err) {...} else {
```

```
var data = new
    lib.Tweet(user_args[0],
    user args[1]);
```

session.persist(data, onInsert, data);



MySQL Cluster NoSQL API for Node.js

```
var onInsert = function(err,
object) {
  console.log('onInsert.');
  if (err) {...} else {
    console.log('Inserted: ' +
    JSON.stringify(object));
```



Try Node.js example for yourself

https://github.com/mysql/mysql-js/tree/master/samples

	github			Signup and Pricing	Explore GitHub	Features	Blog Sign in
UBLIC	📃 mysql / mysq l	l-js		*	Star 3	پ Fork و	
	Code	Network	Pull Requests 0	Issues 0	Wiki		Graphs
	💡 branch: master 💌	Files Commits	Branches 1			Tags	Downloads

mysql-js / samples

name	age	message	history
Create.sql	11 days ago	first version [bo]	
🖹 delete.js	11 days ago	first version [bo]	
■ find.js	11 days ago	first version [bo]	
insert.js	11 days ago	first version [bo]	
■ lib.js	11 days ago	first version [bo]	

Industry Leaders Rely on MySQL





Today's web workloads demand more from databases

Performance, scale-out, simples access patterns & APIs

MySQL meets these needs while still delivering benefits of an ACID RDBMS

Next Steps

- Guide to MySQL and NoSQL Delivering the Best of Both Worlds
 - <u>http://mysql.com/why-mysql/white-papers/mysql-wp-guide-to-nosql.php</u>

- Evaluate MySQL Cluster 7.2
 - http://www.mysql.com/downloads/cluster/
- Bootstrap a Cluster
 - <u>https://edelivery.oracle.com/</u>
- Try Memcached API for InnoDB in 5.6
- http://www.mysql.com/downloads/

ORACLE

San Francisco, September 21-23 **Additional Day of Tutorials** Oracle.com/mysqlconnect

MySQL Connect - Birds-of-a-feather sessions

- Keynotes
- Conferences Sessions
- Tutorials
- Hands-on Labs
- Exhibition Hall
- Demo Pods
- Receptions





Copyright 2012, Oracle and/or its affiliates. All rights reserved.