MySQL – Meetup Day - Brazil

JavaScript for MySQL HeatWave

Ana Paula Sales

MySQL HeatWave Solution Engineer

Ana.sales@oracle.com

Ana Paula Sales

- MySQL Solution Engineer at Oracle
- Bachelors in Science and Technology
- OCI Architect
- MySQL HeatWave Specialist

ORACLE Certified Foundations Associate

ORACLE Certified Associate MySQL HeatWave Implementation Rel 1

ORACLE Certified Associate Oracle Cloud Infrastructure Architect 2023

Microsoft CERTIFIED FUNDAMENTALS

ORACLE Certified Professional MySQL 8.0 Database Administrator

Ana.sales@oracle.com

/anasalesaraujo

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.

The materials in this presentation pertain to Oracle Health, Oracle, Oracle Cerner, and Cerner Enviza which are all wholly owned subsidiaries of Oracle Corporation. Nothing in this presentation should be taken as indicating that any decisions regarding the integration of any EMEA Cerner and/or Enviza entities have been made where an integration has not already occurred.

Forward-looking statements

This presentation 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.

Statements in this presentation relating to Oracle's future plans, expectations, beliefs, intentions, and prospects are "forward-looking statements" and are subject to material risks and uncertainties. A detailed discussion of these factors and other risks that affect our business is contained in Oracle's Securities and Exchange Commission (SEC) filings, including our most recent reports on Form 10-K and Form 10-Q under the heading "Risk Factors." These filings are available on the SEC's website or on Oracle's website at http://www.oracle.com/investor. All information in this presentation is current as of September 2023 and Oracle undertakes no duty to update any statement in light of new information or future events.

Some regulatory certifications or registrations to products or services referenced herein are held by Cerner Corporation. Cerner Corporation is a wholly owned subsidiary of Oracle. Cerner Corporation is an ONC-certified health IT developer and a registered medical device manufacturer in the United States and other jurisdictions worldwide.

The materials in this presentation pertain to Oracle Health, Oracle, Oracle Cerner, and Cerner Enviza which are all wholly owned subsidiaries of Oracle Corporation. Nothing in this presentation should be taken as indicating that any decisions regarding the integration of any EMEA Cerner and/or Enviza entities have been made where an integration has not already occurred.

JavaScript Applications with MySQL

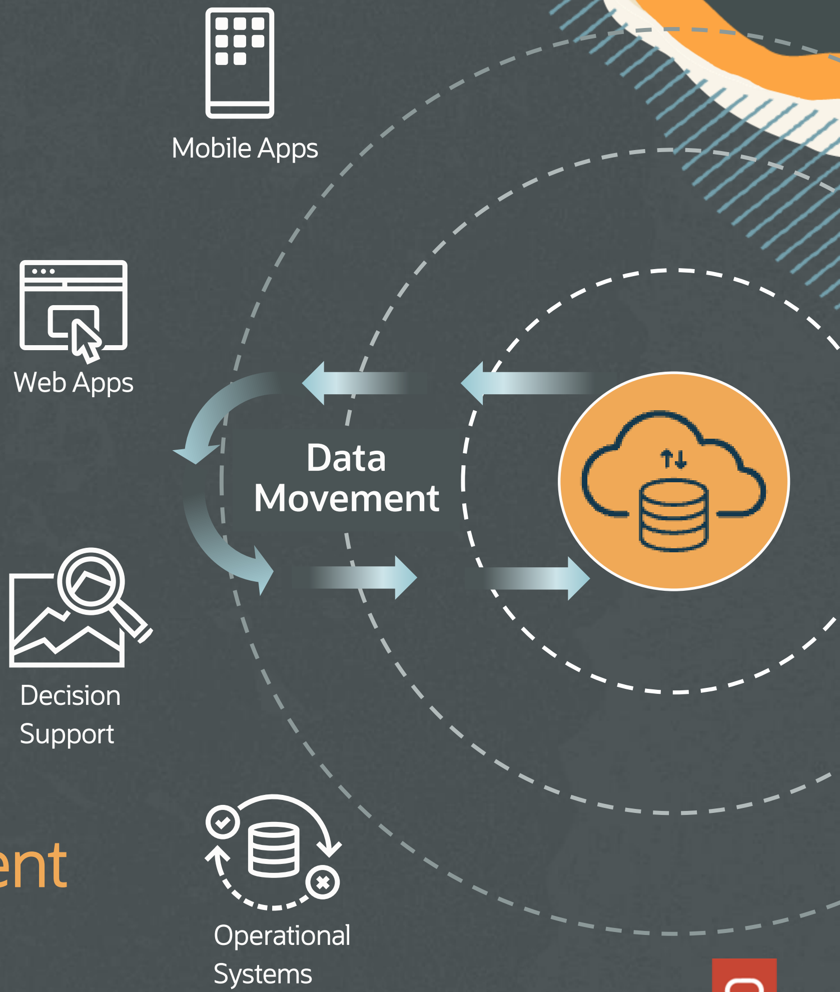
JavaScript applications are popular

- Powerful for light weight front-end and server-side applications
- Operates well with SQL servers, communicate database via connectors

How to handle data-intensive use cases?

- Data validation
- JSON & String processing, formatting
- Data cleaning, transformation
- …

Problem: Need to move data to client



Data Movement in the Cloud

Cost

- Cloud Egress Cost
- Developing Data Pipelines
- Maintaining Data Pipelines

Latency

- Serialization / Deserialization of Data
- Protocol and Connector Overhead

Security

- Unnecessary data transfer

Allow rich procedural programming capability inside database

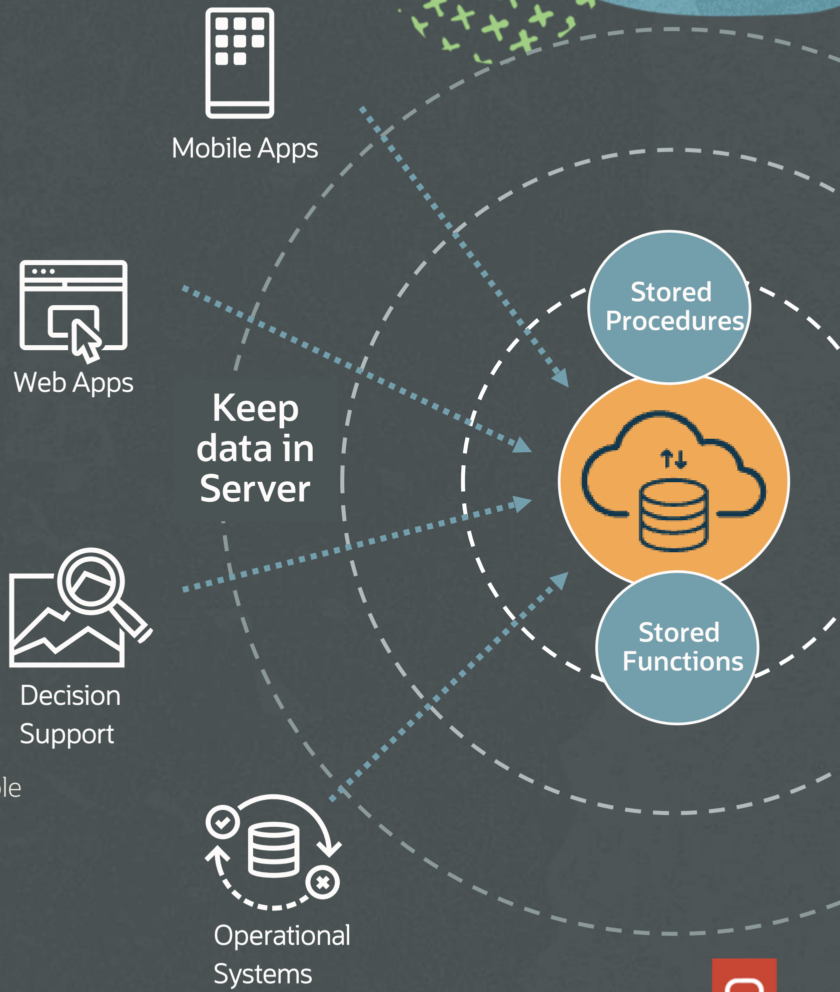
Procedural Program inside database

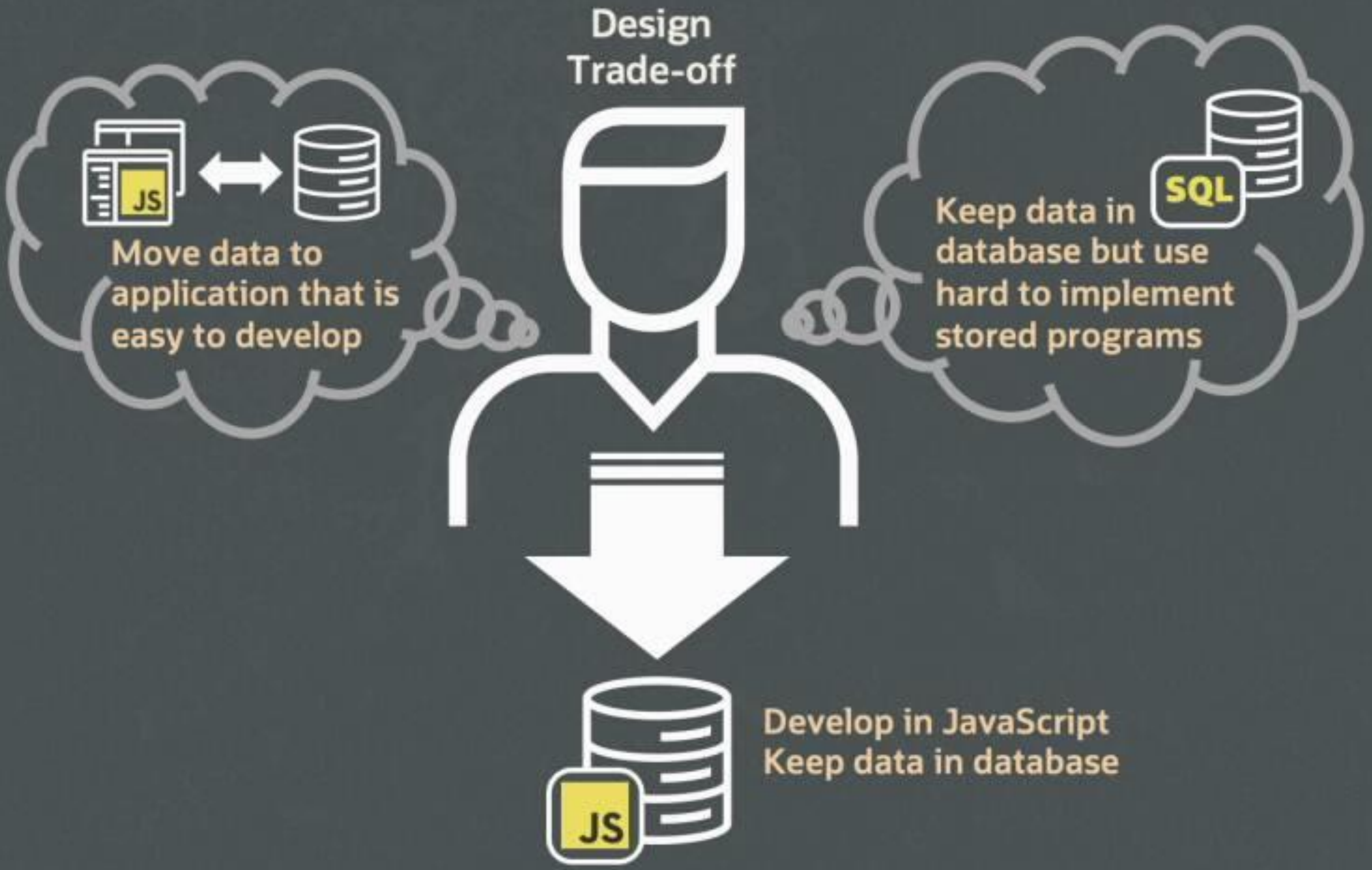
Handle data-intensive app functionality in stored programs

- Minimize data movement
- Reduce cost
- Improve cecurity
- Simplify complex ETL → ELT

Limitations in procedural SQL stored programs

- **Not expressive:** Hard to use, lacks basic constructs like containers (arrays, maps)
- **Not efficient:** challenging to optimize due to interpreted code
- **Insufficient development eco-system:** Editors, debuggers, testing frameworks, reusable 3rd party libraries





Design Trade-off

Move data to application that is easy to develop

Keep data in database but use hard to implement stored programs

Develop in JavaScript
Keep data in database

Introducing JavaScript for MySQL HeatWave (LA) on OCI, AWS, Azure

Execute JavaScript Stored Procedures and Stored Functions

Just like SQL Stored Programs, but now

- Focused on developer experience
- Designed for the cloud service
- Security at its core
- State-of-the-art optimizations

Why JavaScript?

Ubiquitous

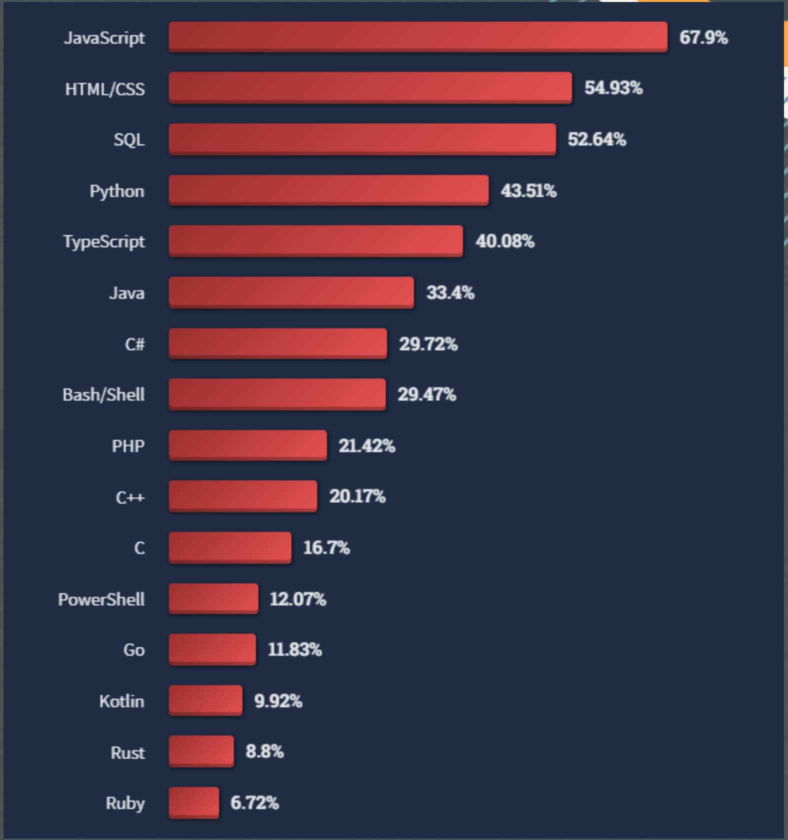
- One of the most used language by developers*
- > 98% of all web pages use JavaScript**

Multiple Runtimes

- Support in all major web browsers
- Massively used server-side runtimes
  - Node.js
  - Deno

Development Eco-system

- Npm package manager contains > 2 million free to use JavaScript packages***
- > 10 million users use the npm package manager



* Stack Overflow 2023 survey
** https://w3techs.com/technologies/details/cp-javascript
*** https://www.npmjs.com/

What is GraalVM

High Performance

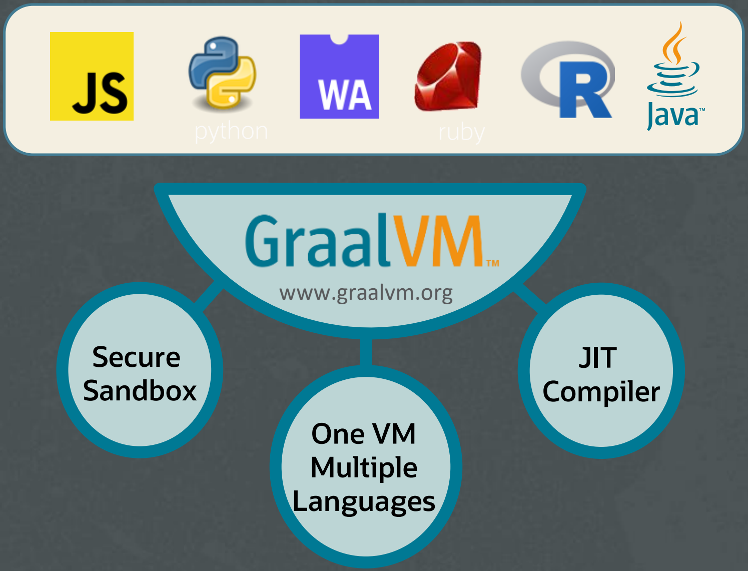
- Profile guided JIT compiler
- Advanced compiler optimizations, such as aggressive inlining and partial escape analysis

Graal.JS

- JavaScript Implementation based on ECMAScript 2023
- Competitive performance with V8 engine
- Implemented using Graal Polyglot Framework, that allows multiple languages inside the same VM

Virtual Machine

- Fully memory managed
- Secure sand box
- Support for developer tools



Secure Sandbox

One VM Multiple Languages

JIT Compiler

Eco-system of compiler technologies

Defining JavaScript Stored Programs

Simple Syntax

- LANGUAGE clause now allows JavaScript
- Extensible string quoting mechanism to enclose non-SQL language source
  - AS $$ … $$
  - AS $JavaScript$ … $JavaScript$

Function Environment

- No function redefinition in JavaScript required
- SQL argument identifiers directly available in JavaScript

Auto Type-Conversion

- Transparent MySQL ↔ JavaScript type conversion
- Supports all variations of INT, FLOATS, DATETIME, VARCHAR (full utf8mb4 support)

```
CREATE FUNCTION gcd_js (a INT, b INT)
RETURNS INT LANGUAGE JAVASCRIPT AS $$

  let [x, y] = [Math.abs(a), Math.abs(b)];
  while(y) [x, y] = [y, x % y];
  return x;

$$
```

JavaScript inside SQL

SELECT

- Use anywhere where SQL stored function can be used
- Expressions, Projection, WHERE clause, GROUP-BY, JOIN, ORDER BY, HAVING etc.

DMLs, DDLs, VIEWs

- Support inside all DMLs (e.g., INSERT, UPDATE, DELETE)
- DDLs including CREATE TABLE AS SELECT
- Support inside VIEWs

Interoperability

- Invoke JavaScript & SQL functions and procedures inside existing SQL stored functions or procedures
- Chain JavaScript & SQL stored functions together using input / output arguments

```
SELECT col1, col2, gcd_js(col1,col2)
FROM my_table
WHERE gcd_js(col1, col2) > 1
ORDER BY gcd_js(col1, col2);


CREATE TABLE gcd_table
AS SELECT gcd_js(col1,col2)
FROM my_table;
```

SQL inside JavaScript

Statement Types

- Simple SQL statements
- Prepared statements with bind parameters

Data Access API

- Execute SQL inside JavaScript using XDevAPI
- Seamless MySQL ↔ JavaScript type conversion for query results

Session State

- Continue transactions inside JavaScript
- Access all session state inside JavaScript such as session variables & temporary tables

```
CREATE PROCEDURE gen_random_age (IN row_count INT)
LANGUAGE JAVASCRIPT AS $$
  let insertStatement = mysql.getSession().prepare(
    "INSERT INTO my_table(age) VALUES ( ? )");
  for (var j = 0; j < row_count; j++) {
    var random_age = Math.trunc(Math.random() * 100);
    insertStatement.bind(random_age).execute();
  }
$$
```

```
CREATE PROCEDURE add_age (OUT age_sum INT)
LANGUAGE JAVASCRIPT AS $$
  age_sum = 0;
  let selectStatement = mysql.getSession().sql(
    "SELECT age FROM my_table");
  let result = selectStatement.execute(), row = null;
  while(row = result.fetchOne())
    age_sum += row[0];
$$
```

Debuggability

Standard Streams

- Access language standard output and error streams inside MySQL

Error Handling

- Translates unhandled JavaScript exceptions into MySQL errors
- Allow access to JavaScript stack traces in case of unhandled runtime error
- Translate MySQL errors and warnings into JavaScript exceptions while executing SQL statements inside JavaScript

```
CREATE PROCEDURE print_js()
LANGUAGE JAVASCRIPT AS $$
  console.log(“Hello World!”);
$$

CALL print_js();
SELECT mle_session_state(“stdout”);
Hello World!
```

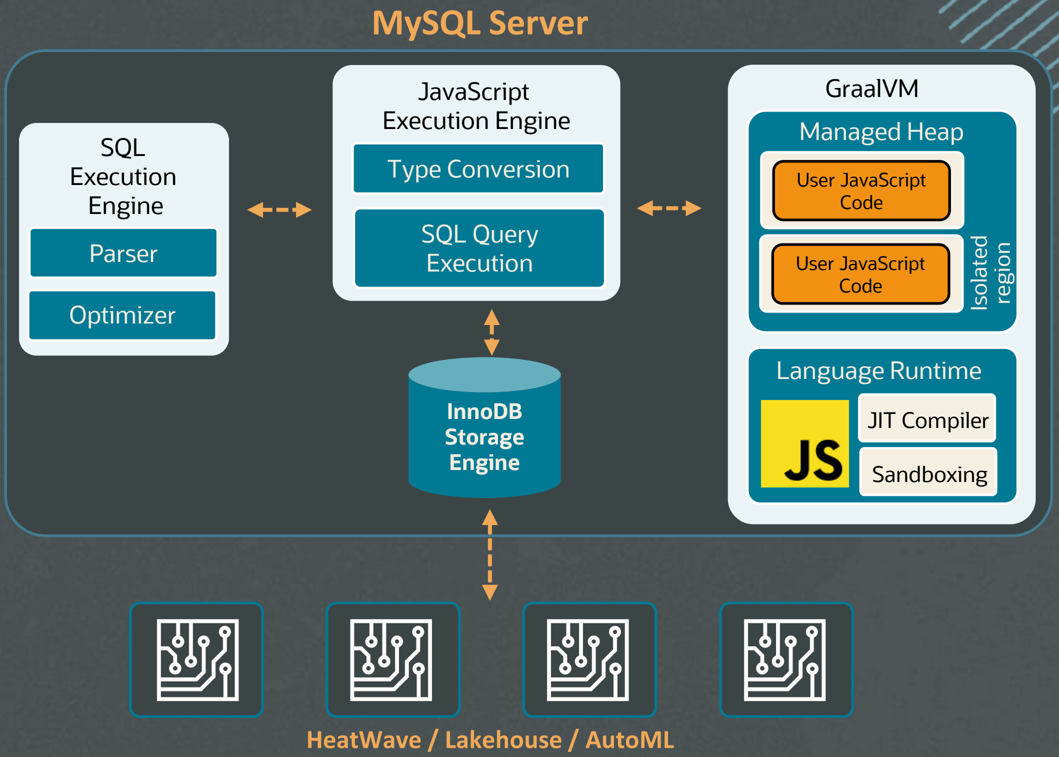
Cloud Integration

1. Cloud Centric Architecture
2. Resource Utilization
3. Security
4. Performance

Cloud-Centric Architecture

Works seamlessly with various server components configured on cloud service:

- InnoDB
- HeatWave
- HA / Replication solutions
- Enterprise Thread Pool
- MySQL AutoPilot
- HeatWave AutoML
- Auditing



Security

Code Isolation

Prevents visibility or interaction between any two different stored programs executed on GraalVM

Adds protection against JIT spraying and side-channel attacks.

Sand boxing

Each stored program runs inside GraalVM strict sandboxing policy that blocks any unauthorized access to

- File system
- Thread management
- Network access
- Native Access

MySQL Privileges

Uses MySQL Privileges for stored programs

SQL execution inside JavaScript uses DEFINER and INVOKER security context

Resource Utilization

Auto Configuration

- Memory and compute resources are configured based on the cloud instance shape

Resource Management

- Lazy allocation: resource utilization is zero if feature not used.
- Memory utilization is capped: benefits from GraalVM garbage collection
- Concurrency regulated by MySQL enterprise thread pool

Resource Monitoring

- Resource utilization available via MySQL status variable

Key Take Aways

Express complex logic in database using JavaScript

Push part of data-intensive application inside the database

Avoid vendor lock-in

Benefit from GraalVM Enterprise Edition optimizations at no additional cost

Integrate with MySQL cloud-only features seamlessly

Reduce data movement cost

Ana Paula Sales
MySQL Solution Engineer
Ana.sales@oracle.com

Thank YOU!