ORACLE

MySQL REST Serviceでも使える JSON Relational Duality技術のご紹介

Open Source Conference 2024 Hokkaido
2024年6月29日

日本オラクル
MySQL Global Business Unit
MySQL Principal Solution Engineer
大塚 恒平 / Kohei Otsuka

アジェンダ

1. MySQL とは
2. MySQL で JSON を使う 3 つの方法
   2-1. なぜMySQL で JSON なのか？
   2-2. JSON データ型（ダイジェスト）
   2-3. MySQL ドキュメントストア（ダイジェスト）
3. MySQL REST Service と JSON Relational Duality
   3-1. MySQL Shell for VS Code
   3-2. MySQL REST Service
   3-3. JSON Relational Duality
4. 本セッションのまとめ
5. MySQL 製品 / サービス / コミュニティのご紹介

1. MySQL とは

MySQL とは
世界で最も普及しているオープンソースデータベース

デュアルライセンス
- コミュニティ版
- 商用版

Oracle が開発・提供
- 専任開発者
- バージョンリリース
- パッチ提供

MySQLを利用いただいているお客様

Social media	E-commerce	Finance	Tech	Car Manufacturers
facebook	Alibaba.com	VISA	HubSpot	TOYOTA
Twitter	airbnb	J.P.Morgan	GitHub	Volkswagen
LinkedIn	NETFLIX	citi	zendesk	TESLA
WeChat			New Relic	CAT

MySQLの新バージョンポリシー：LTSとInnovation Releases

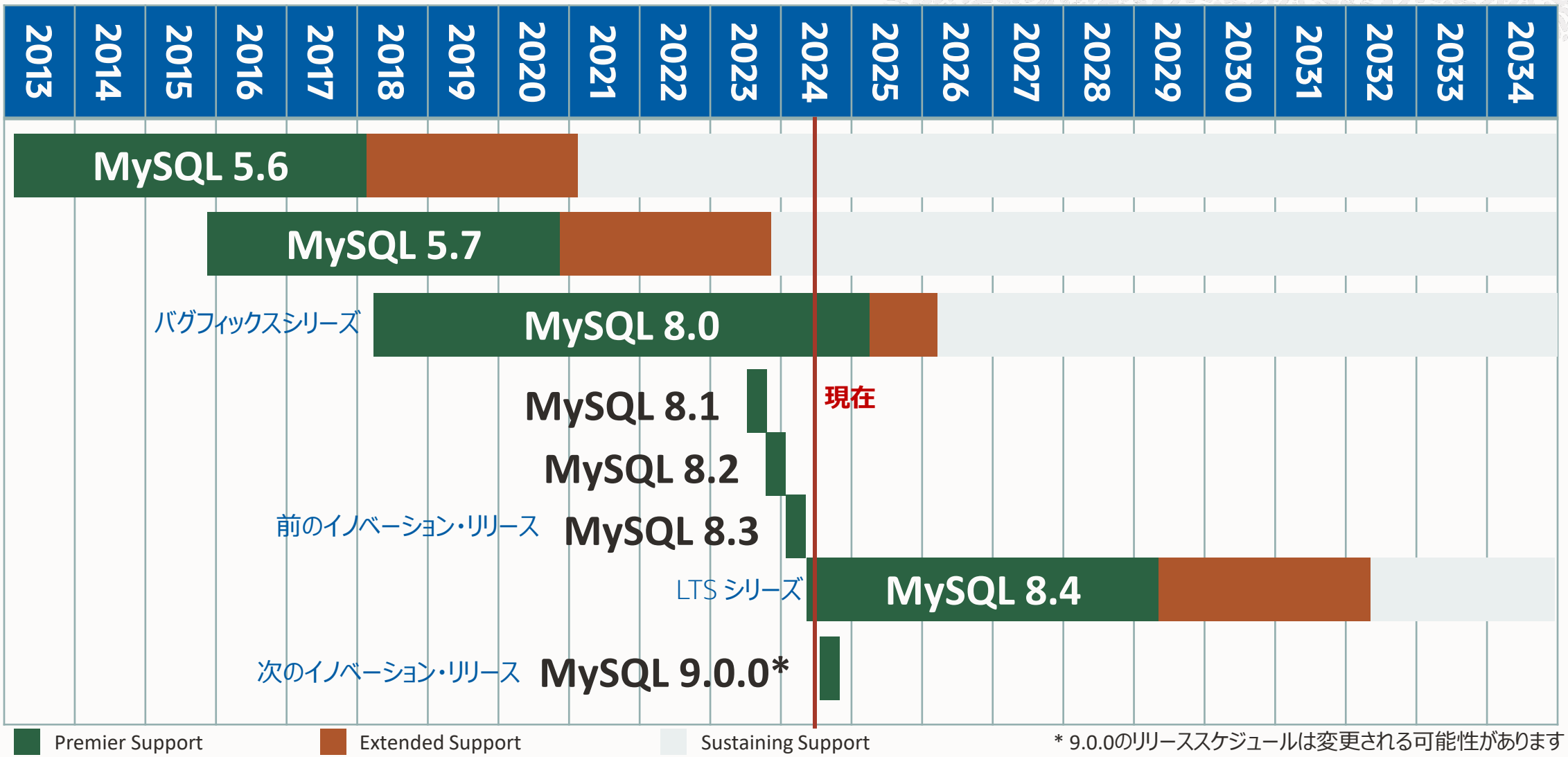
イノベーション・リリース

- **バグ修正と新機能追加を行うリリース**
  - MySQL 8.1, 8.2, 8.3, 9.0, 9.1 …
- **リリース方針**
  - バグ修正
  - セキュリティ・パッチ
  - 新機能追加
  - 機能やパラメータの非推奨化および削除
- **リリースサイクル**
  - 3ヶ月毎
  - 次バージョンのリリースでEOL
- **本番運用想定のテスト済み**

LTS(Long-Term Support)リリース

- **バグ修正のみを行うリリース**
  - MySQL 8.4
- **リリース方針**
  - バグ修正
  - セキュリティ・パッチ
  - バージョン間の互換性重視
- **リリースサイクル**
  - リリース後8年間サポート
  - 複数のLTSリリースをサポート予定
- **本番運用想定のテスト済み**

MySQL リリースとサポートのタイムライン



2. MySQL で JSON を使う 3 つの方法

2-1. なぜMySQL で JSON なのか？

リレーショナルとスキーマレスとの差とは？

- リレーショナル

```
SQL> SELECT * FROM pizza;
+------+------------------+-------+
| code | name             | price |
+------+------------------+-------+
| CLA  | Classic Pizza    | 400   |
| MAR  | Margherita Pizza | 500   |
+------+------------------+-------+
SQL> SELECT * FROM toppings;
+--------+------------+
| p_code | name       |
+--------+------------+
| CLA    | Pepperoni  |
| CLA    | Parmesan   |
| MAR    | Basil      |
| MAR    | Mozzarella |
+--------+------------+
```

テーブル（表形式）

表、カラム、行

- スキーマの適用が容易、長期的なアプリの変更管理に有用
- データに型や外部キーなどの制約を設定できる
- 正規化でデータ重複の削減
- トランザクション: ACID特性
- パワフルで標準化された SQL 言語

- スキーマレス（ドキュメント）

```
{
    "name": "Classic Pizza",
    "price": 400,
    "toppings": [ "Pepperoni", "Parmesan" ]
}
{
    "name": "Margherita Pizza",
    "price": 500,
    "toppings": [ "Basil", "Mozzarella" ],
    "options": [ { "name": "Olive", "price": 100 } ]
}
```

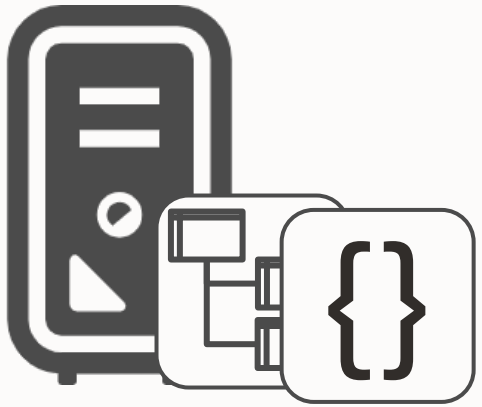
JSON

属性値のペアのコレクション

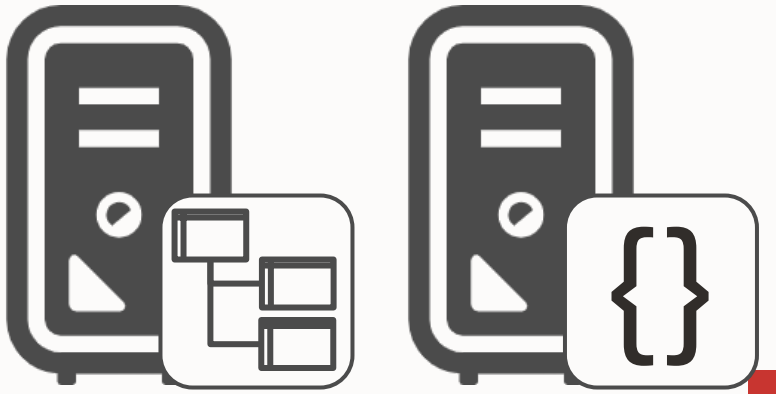
- ネストされる配列やオブジェクトのツリー構造 ([], {})
- スキーマレス: スキーマ設計、正規化、各種制約等の設計不要
- リレーショナルモデルではモデル化しにくいデータを柔軟に表現
- 迅速で容易な設計・プロトタイピング
- ORMを利用せずに、オブジェクトの永続化が可能

単一データベースで管理するか、複数データベースで管理するか？

- 単一データベース
  - より多くのメンバーが共通スキルで管理可能
  - 管理コストが抑制される
  - 少ないライブラリで開発可能
  - 少ないツールで管理可能
  - 容易なデータ連携
  - 運用及び分析を一緒に
  - SQL処理、CRUD処理共に可能



- 複数のデータベース
  - 追加のスキルが必要となり、管理・開発の複雑化
  - 管理コストが増加
  - 開発ライブラリの複数使い分けが必要
  - 管理ツールの複数使い分けが必要
  - データ連携に工数・コストがかかる
  - 運用と分析を別システムで処理



MySQL で JSON を扱う 3 つの方法

いずれもCommunity Editionで利用可能

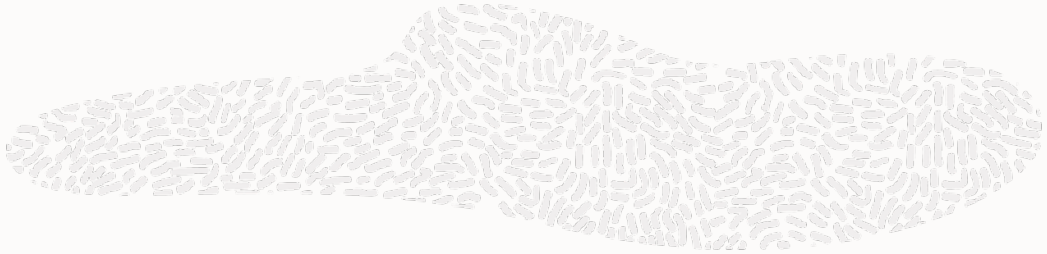
※ CRUD: CREATE（=INSERT）
READ（=SELECT）
UPDATE
DELETE

1. JSON データ型
   - MySQL テーブルの列に指定できるデータ型
   - JSON 関数を用いて属性を取得、さらに GENERATED カラムでインデックス付与
   - SQL を通じた操作: クラシックプロトコル（デフォルト 3306 ポート）で操作可能
2. MySQL ドキュメントストア
   - SQL ではなく、ドキュメントに対する CRUD（）処理
   - 新しい X プロトコル（デフォルト 33060 ポート）から、MySQL Shell や各言語の Connectors API から操作
   - 内部的には、JSON データ型カラムを含む特殊なテーブルの上に構築された機能
3. MySQL REST Service（& JSON Relational Duality）
   - MySQL REST Service（MRS）は、テーブルに対する CRUD 処理を HTTP 経由で行える機能
   - JSON Relational Duality は、外部制約を持ったリレーションテーブルの階層構造を、ネストされた JSON として CRUD できる MRS の拡張機能（完全に自由なスキーマの JSON ドキュメントは扱えない）
   - MySQL 自身ではなく MySQL Router（インターフェース実装）と MySQL Shell for VS Code（管理）が提供する機能（プレビュー版）

2-2. JSON データ型（ダイジェスト）

JSON データ型

MySQL 5.7.9、2015年10月 ～



- ネイティブJSONデータ型 (バイナリ形式)
- Insert時のJSON構文バリデーション機能
- 組み込みJSON関数(保存、検索、更新、操作)
- ドキュメントにインデックス設定し高速アクセス
- SQLとの統合を容易にする、新しいインライン構文
- utf8mb4の文字セットとutf8mb4_binの照合「

JSON データ型: サポートする値

- JSONで表現する全てのデータ型をサポート
  - 数値、文字列、Boolean (true / false)
  - オブジェクト {"キー" : "値"}、配列 [123456, "文字列", ...]
  - null
- 拡張
  - 日付 (date)、時刻、日付 (datetime)、タイムスタンプ、その他

```
SQL > show create table T_JSON_SUPPORT ¥G
*************************** 1. row ***************************
       Table: T_JSON_SUPPORT
Create Table: CREATE TABLE `T_JSON_SUPPORT` (
  `id` int NOT NULL AUTO_INCREMENT,
  `body` json DEFAULT NULL,
  `type` varchar(20) GENERATED ALWAYS AS (json_type(`body`))
VIRTUAL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci
1 row in set (0.0011 sec)
```

```
SQL > SELECT * FROM T_JSON_SUPPORT;
+----+--------------------------------+----------+
| id | body                           | type     |
+----+--------------------------------+----------+
|  8 | 1234567890                     | INTEGER  |
|  9 | NULL                           | NULL     |
| 10 | true                           | BOOLEAN  |
| 11 | "abcde"                        | STRING   |
| 12 | {"id": 5, "name": "Object"}    | OBJECT   |
| 13 | [-122.4220035, 37.80848009]    | ARRAY    |
| 14 | "2023-12-13"                   | DATE     |
| 15 | "2023-12-13 04:08:01.000000"   | DATETIME |
+----+--------------------------------+----------+
```

JSON データ型の作成

- 2通りの作り方

```
SQL > SET @a = JSON_OBJECT('p1', 1, 'p2', JSON_OBJECT('p3', 'val'));
Query OK, 0 rows affected (0.0009 sec)
SQL > SET @b = CAST('{"p2": {"p3": "val"}, "p1": 1}' AS JSON);
Query OK, 0 rows affected (0.0008 sec)
SQL > SELECT @a, @b, @a = @b;
+---------------------------------+---------------------------------+---------+
| @a                              | @b                              | @a = @b |
+---------------------------------+---------------------------------+---------+
| {"p1": 1, "p2": {"p3": "val"}} | {"p1": 1, "p2": {"p3": "val"}} |       1 |
+---------------------------------+---------------------------------+---------+
1 row in set (0.0009 sec)
```

p1を先に定義

p2を先に定義

一意になる

  - JSON_OBJECT関数で、属性と値を列挙して作成
  - JSONとしてValidな文字列を、CAST関数でJSONデータ型にキャストして作成
- 属性の並び順は、定義時の順序に関係なく正規化される

JSON データ型カラム内の属性によるインデックス付与

- JSON の属性値で GENERATED COLUMN を生成し、そのカラムにインデックスを付与

```
SQL > ALTER TABLE jsontable ADD COLUMN virt_p1 integer GENERATED ALWAYS AS (jdoc->>'$.p1') VIRTUAL;
Query OK, 700 rows affected (0.3825 sec)

Records: 700  Duplicates: 0  Warnings: 0
SQL > ALTER TABLE jsontable ADD INDEX virt_index_p1 ( virt_p1 );
Query OK, 0 rows affected (0.0572 sec)

Records: 0  Duplicates: 0  Warnings: 0
SQL > EXPLAIN SELECT * FROM jsontable WHERE jdoc->>'$.p1' > 10 AND jdoc->>'$.p1' < 30;
+----+-------------+----------+------------+-------+---------------+---------------+---------+------+------+----------+-------------+
| id | select_type | table    | partitions | type  | possible_keys | key           | key_len | ref  | rows | filtered | Extra       |
+----+-------------+----------+------------+-------+---------------+---------------+---------+------+------+----------+-------------+
|  1 | SIMPLE      | jsontable | NULL       | range | virt_index_p1 | virt_index_p1 | 5       | NULL |   19 |      100 | Using where |
+----+-------------+----------+------------+-------+---------------+---------------+---------+------+------+----------+-------------+
1 row in set, 1 warning (0.0052 sec)
Note (code 1003): /* select#1 */ select `jsontable`.`jsontable`.`ID` AS `ID`,`jsontable`.`jsontable`.`jdoc` AS
`jdoc`,`jsontable`.`jsontable`.`virt_p1` AS `virt_p1` from `jsontable`.`jsontable` where ((`jsontable`.`jsonta
ble`.`virt_p1` > 10) and (`jsontable`.`jsontable`.`virt_p1` < 30))
```

GENERATED カラムに
インデックス生成

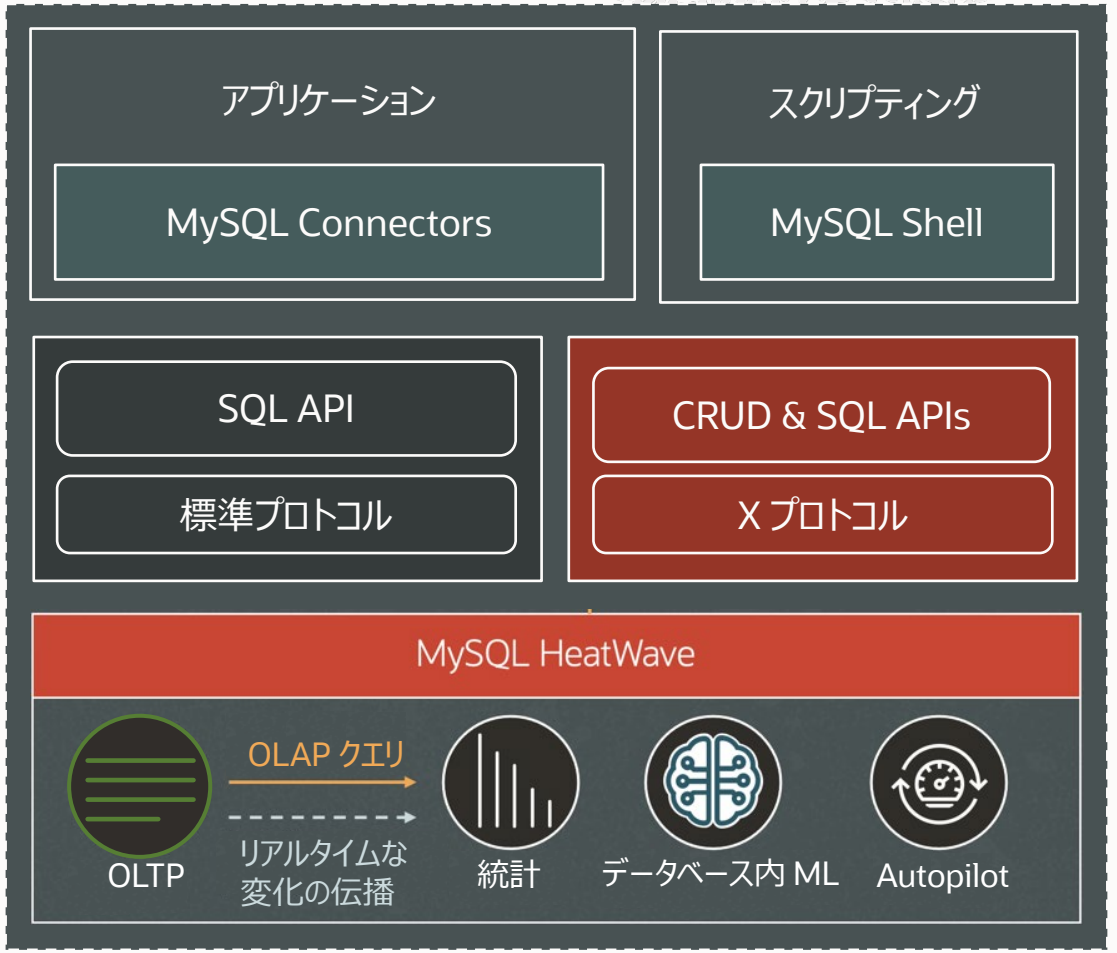
ASの後の演算結果で仮想的なカラム作成
- STORED: 挿入時に値計算、領域消費
- VIRTUAL: 評価時に値計算、領域消費しない
- 基本的にVIRTUALでよい

2-3. MySQL ドキュメントストア（ダイジェスト）

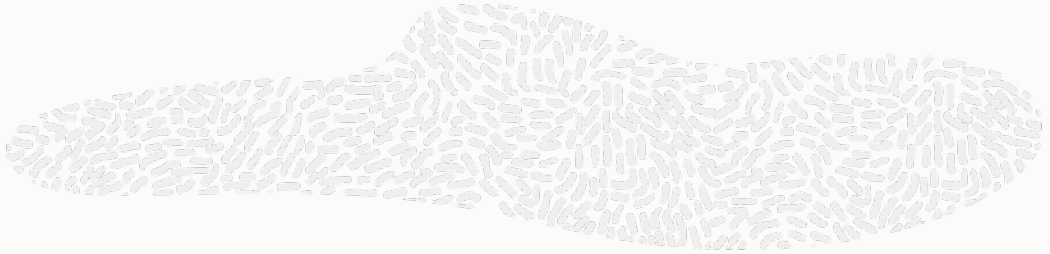
ドキュメントストア機能

MySQL 5.7.12、2016年4月 ～

- X プロトコル
  - MySQL をドキュメントストアとして用いるために、X プラグイン (mysqlx) により実装、デフォルト有効
- X DevAPI
  - SQL 処理とドキュメントに対する CRUD 処理
  - 各言語の MySQL Connectors により実装
- MySQL Shell (mysqlsh) (2017年4月GA)
  - コマンドラインクライアント (Javascript, Python, SQL)
- MySQL Shell for VSCode (2022年3月以降LA)
  - Visual Studio Code のプラグインとして動作する MySQL GUI (MySQL Shell、ノートブック I/F)

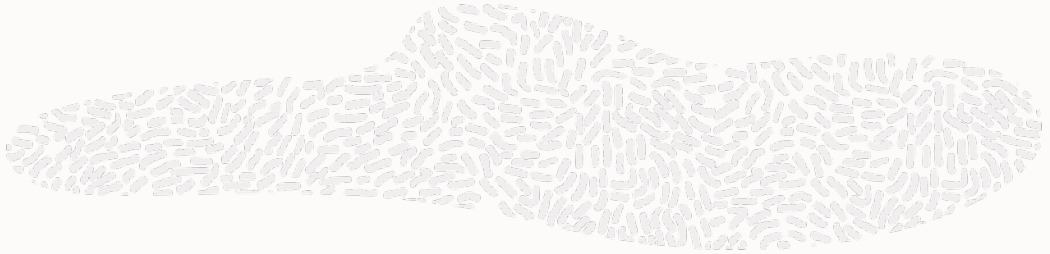


X プロトコルと X DevAPI



- X プラグインにより実装（デフォルトで有効化）
- X プロトコル
  - クライアントとサーバー間の柔軟な接続性
  - コマンドのパイプライン化に対応した非同期 API
  - TCP ポートとして 3306 ポートではなく 33060 ポートを利用（デフォルト）
  - 通信はラップされた protobuf 標準に則ったメッセージを使用
  - SQL とドキュメントに対する新しい CRUD API の両方をサポート
- X DevAPI
  - X プラグイン (MySQL) ⇔ X プロトコル ⇔ X DevAPI (ドライバー) の組み合わせで動作
  - ドキュメントとテーブルのコレクションに対しての CRUD 処理
  - NoSQL ライクな API 構文でドキュメントに対し CRUD 処理可能
  - 各言語の MySQL Connectors ドライバー、MySQL Shell、MySQL Shell for VSCode などで動作

MySQL Connectors での X DevAPI



- SQL、CRUD APIsの利用
- スキーマレスドキュメントおよびリレーショナルテーブルに対応

操作	ドキュメント	リレーショナル
Create	Collection.add()	Table.insert()
Read	Collection.find()	Table.select()
Update	Collection.modify()	Table.update()
Delete	Collection.remove()	Table.delete()

ドキュメントストアの仕組み
ドキュメントストア Collection の正体

```
MySQL  10.0.X.X:33060+ ssl  tpch_10g  SQL > SHOW CREATE TABLE LINEITEM_DOC;

 ......

| LINEITEM_DOC | CREATE TABLE `LINEITEM_DOC` (
  `doc` json DEFAULT NULL,
  `_id` varbinary(32) GENERATED ALWAYS AS (json_unquote(json_extract(`doc`,_utf8mb4'$._id'))) STORED NOT NUL
L,
  `_json_schema` json GENERATED ALWAYS AS (_utf8mb4'{"type":"object"}') VIRTUAL,
  PRIMARY KEY (`_id`),
  CONSTRAINT `$val_strict_37756BDA39FEEC09BB17E7305CECE8F5EC904E4D` CHECK (json_schema_valid(`_json_schema`,
`doc`)) /*!80016 NOT ENFORCED */
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |c
```

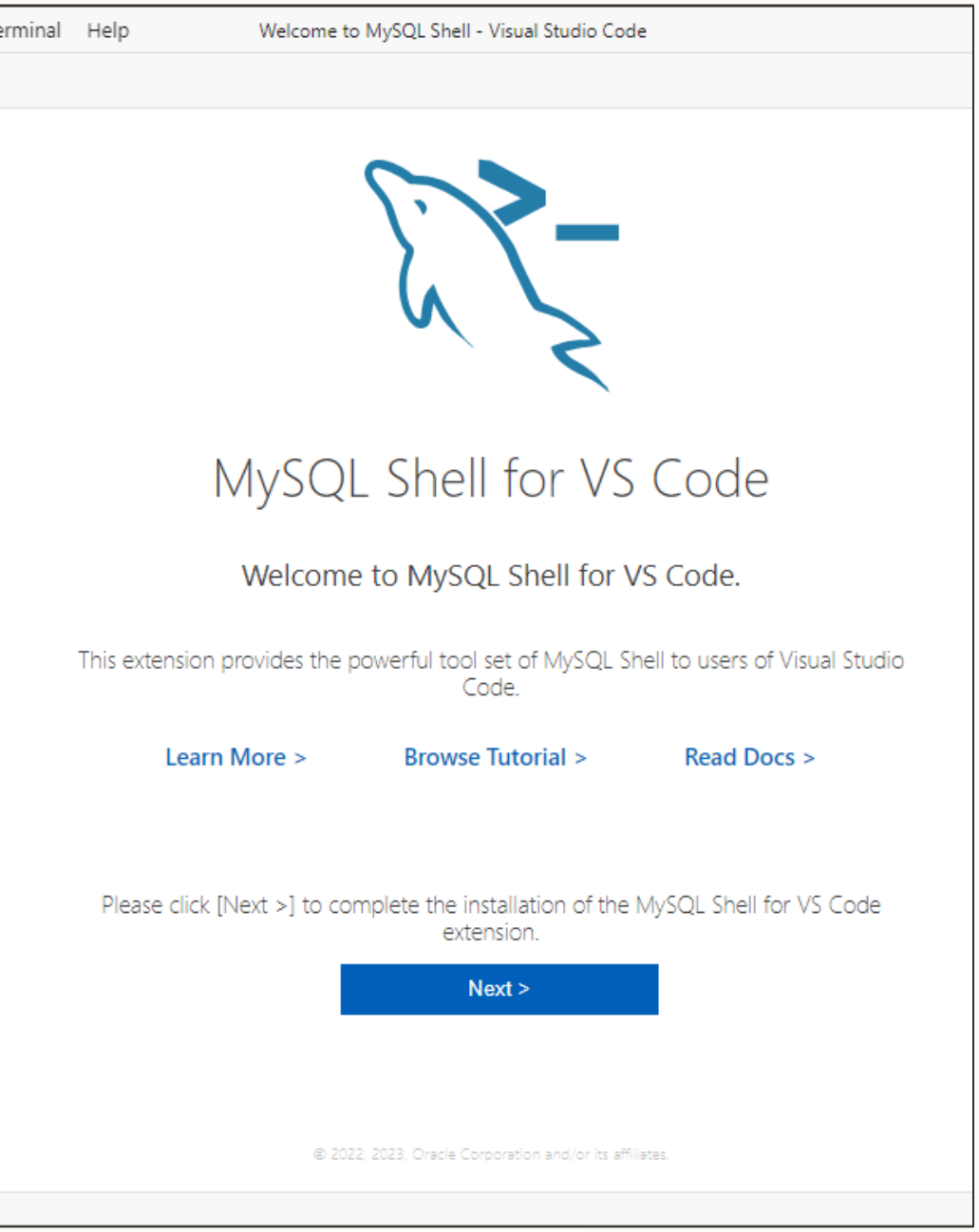
- Collectionの正体は特別な構造を持ったテーブル
  - JSON オブジェクトを保持する JSON 型カラム
  - JSON から ID の値を抜き出した STORED Generated Column
  - JSON 構文チェック用の JSON Schema 制約を含む VIRTUAL Generated Column

JSON データ型、MySQL ドキュメントストアの詳細

- 過去のセミナー、ウェビナー、ブログ資料などをご覧ください
  - オープンソースカンファレンス2024 Online Spring
    『JSONにJavaScript - Webフロントエンド技術とMySQL HeatWaveの2024最新事情』
    - 動画
    - 資料

3. MySQL REST Service と JSON Relational Duality

3-1. MySQL Shell for VS Code



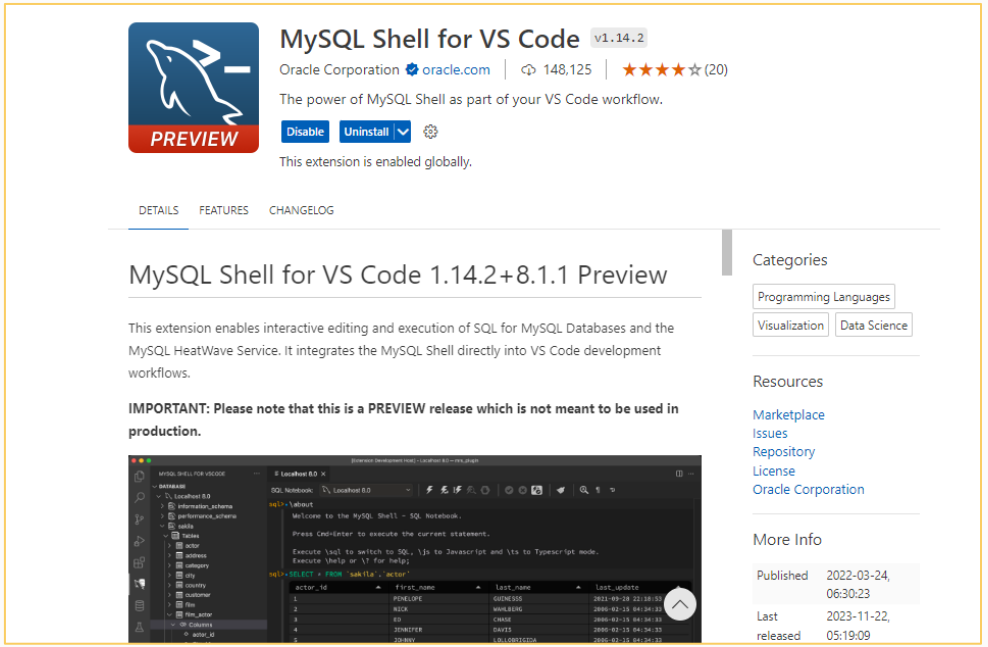
MySQL Shell for VS Code

MySQL Workbenchの利便性とMySQL Shellの機能を統合
https://github.com/mysql/mysql-shell-plugins

MySQL Shell for VS Code 1.14.2+8.1.1 Preview リリース

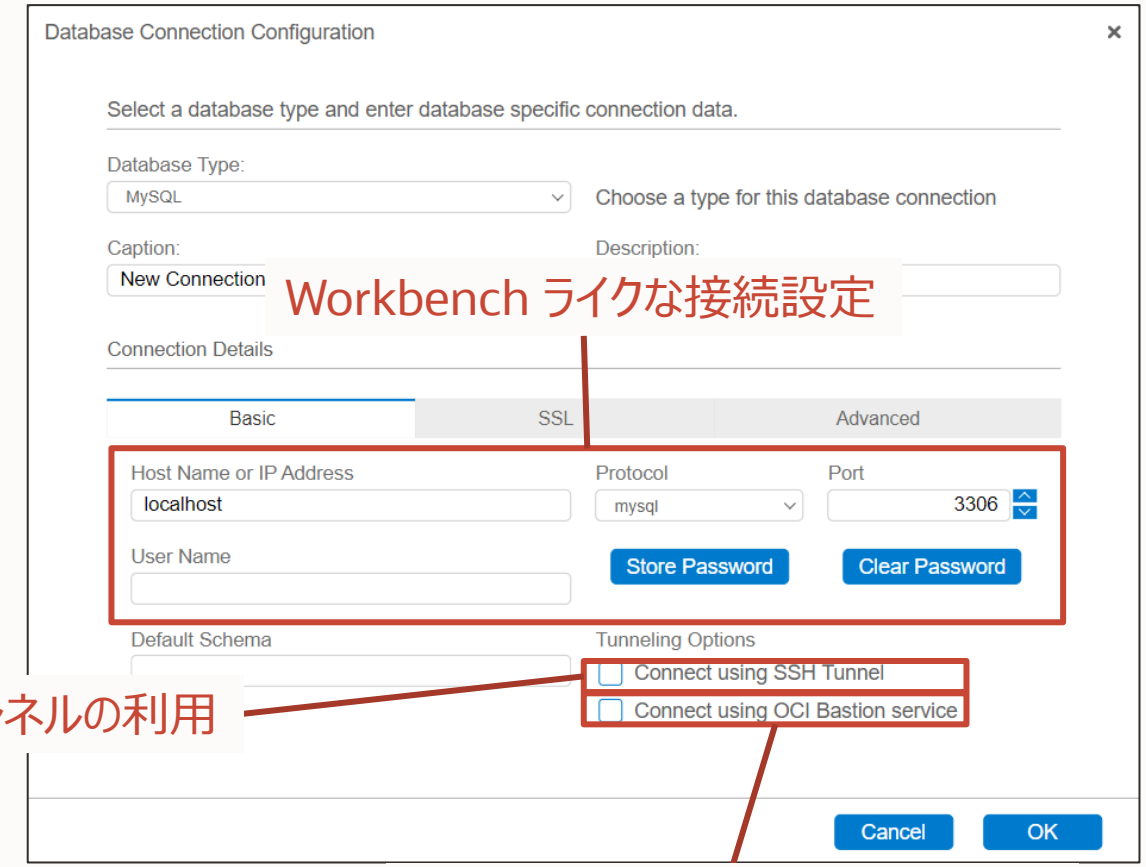
- コミュニティ向けのプレビュー版
- GPL v2 (OpenSSL関連の追加許諾あり)
- 主な機能
  - MySQLサーバーやMySQL HeatWaveのインスタンスへの接続を管理
  - Oracle Cloud Infrastructure (OCI) のMySQL HeatWaveや関連するサービスの管理
  - データベースに接続した上でDB Editorでの作業
    - DB Notebook インターフェース
    - MySQL Shell GUI Console
    - MySQL Workbench後継の管理ダッシュボード
- Visual Studio Code のマーケットプレイスから検索してインストール

OS	バージョン/環境	アーキテクチャ
macOS	11以上	arm64, x64
Windows	10以上	x64
Linux	glibc 2.12	arm64, x64



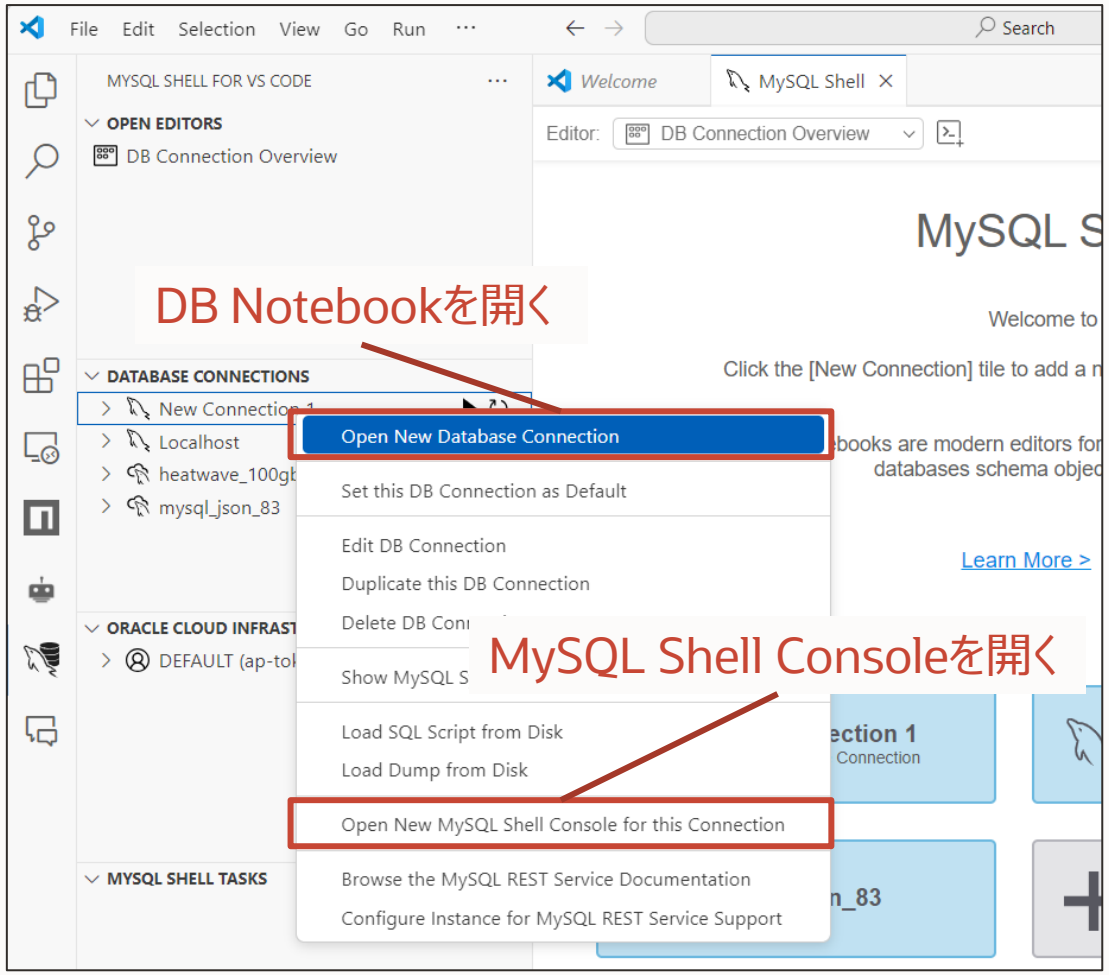
MySQL Serverへの接続

- MySQL Workbenchライクな接続設定
- SSHトンネルを通じた接続にも対応
- OCI への接続は、中継サーバである OCI Bastion（要塞）サービスの自動起動にも対応



MySQL Shell for VS Codeの2つのコマンドラインインターフェース

DB NotebookとMySQL Shell Console



機能	DB Notebook	MySQL Shell Console
対応言語	SQL, JavaScript, TypeScript	SQL, Python, JavaScript
対話型実行	○	○
バッチ実行	○	○
対応API	N/A	AdminAPI, X DevAPI, ShellAPI
Xプロトコル対応	○	○
ユーティリティ	×	○
APIコマンドライン	N/A	N/A
出力フォーマット	×	×
ログ/デバグ	○	○
グローバルセッション	×	○

MySQL Shell for VS Code の詳細

- 過去のセミナー、ウェビナー、ブログ資料などをご覧ください
  - MySQL Innovation Day Tokyo 2023
    『MySQL ShellとMySQL Shell for VSCode』
    - 資料

3-2. MySQL REST Service

MySQL REST Service (MRS)

- MySQL Router（MRS プレビュー、こちらからダウンロード）上で動作、MySQL Shell for VS Code で設定
- MySQLデータへの高速で安全なHTTPSアクセス
- オンプレミス、MySQL HeatWave双方で利用可能

RESTful Webサービス

- テーブル、ビュー、プロシージャの自動REST化
- {JSON} による応答
- 結果のページング
- 開発者サポート (GUI、CLI、API)

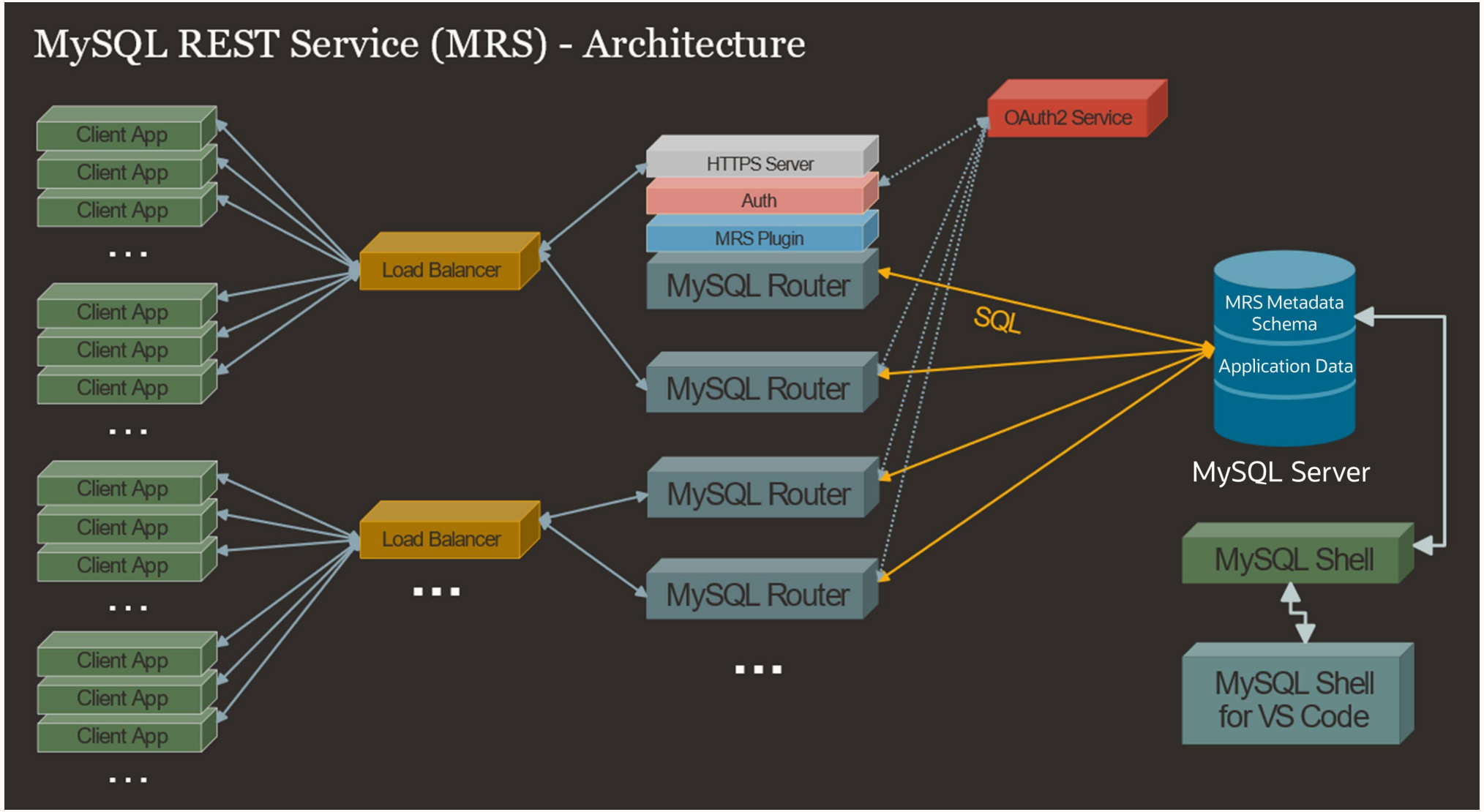
MySQL Shell for VS Code

- MRS管理用のGUIフロントエンド
- RESTfulなWebサービス作成
- 対話的なドキュメント
- CLIとスクリプト実行のサポート

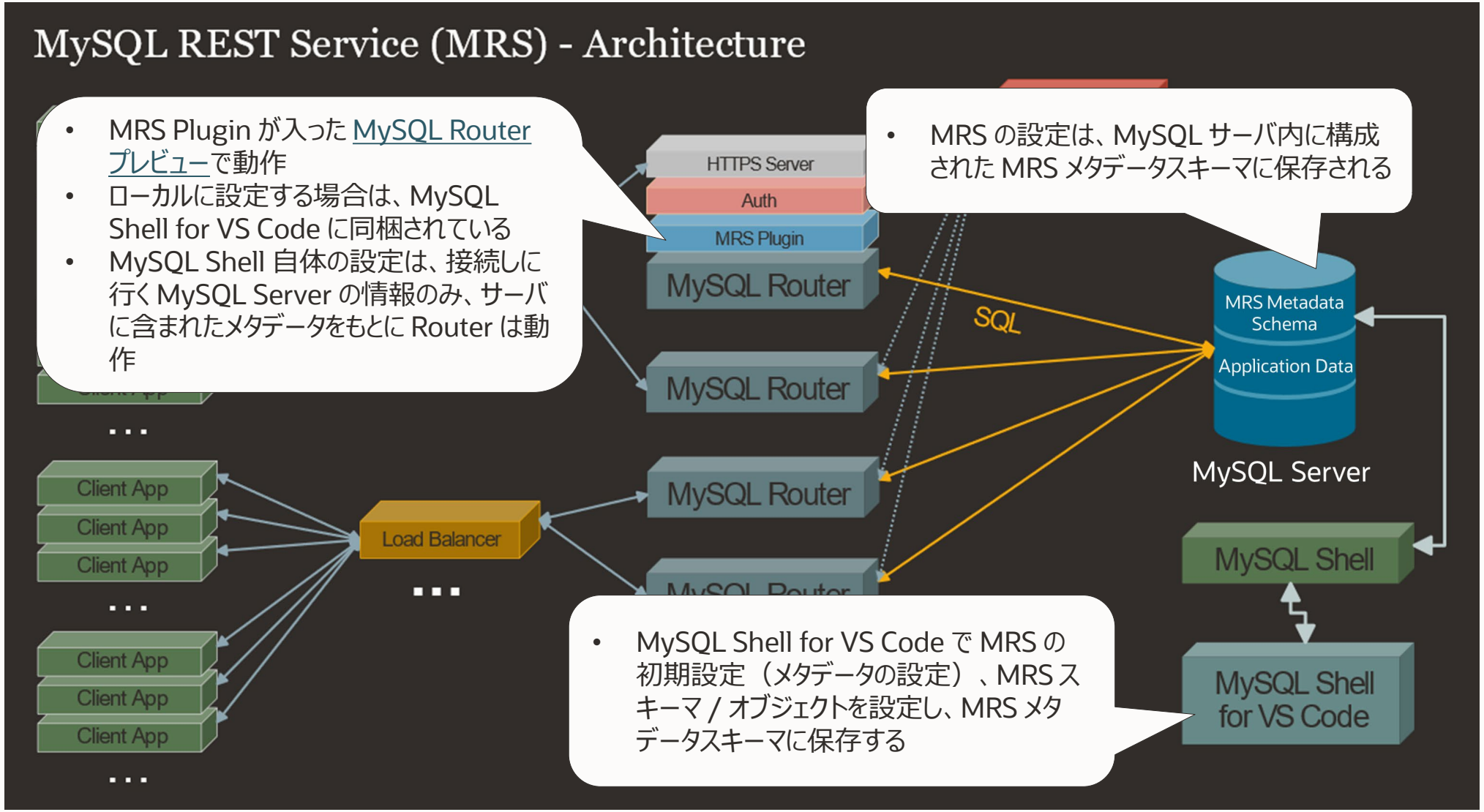
ユーザ管理機能の搭載

- 一般的なOAuth2認証をサポート
- ロール、グループ、階層管理の使用
- ユーザー管理GUI
- CLIとスクリプト実行のサポート

MySQL REST Service のアーキテクチャ

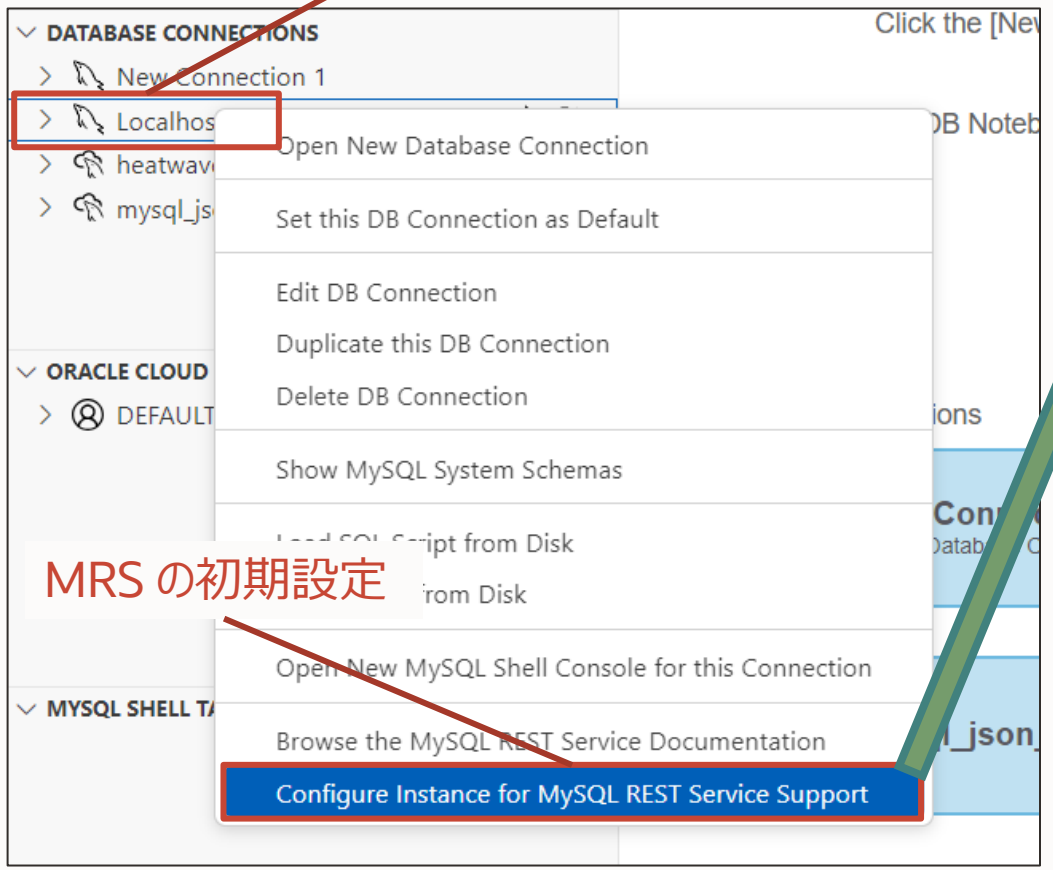


MySQL REST Service のアーキテクチャ



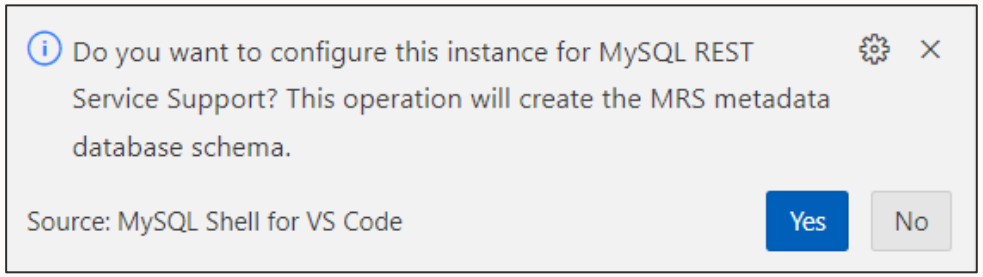
MySQL REST Serviceの初期設定

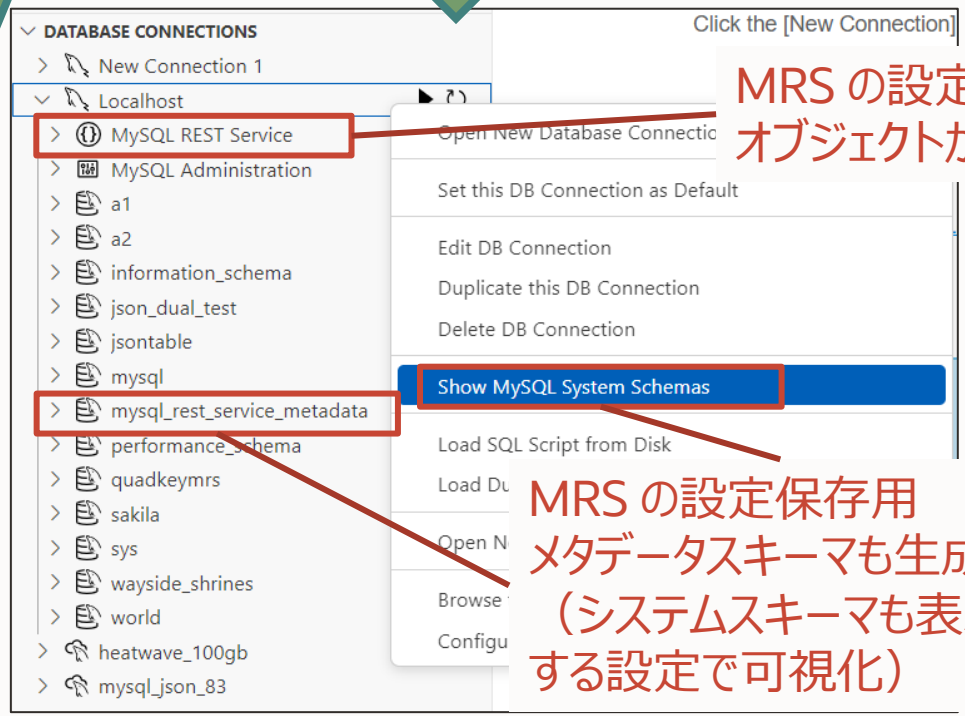
接続を右クリック



MRS の初期設定

MRS を設定して、MRS 用のメタデータスキーマを作成してよいかの確認

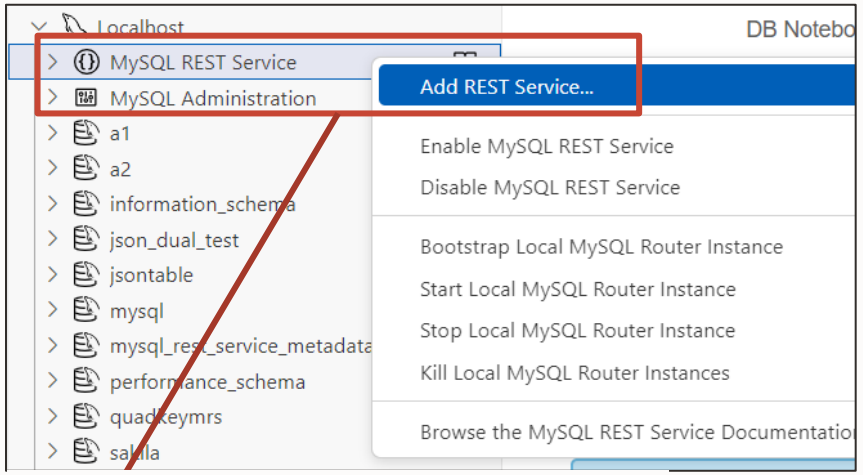




MRS の設定用オブジェクトができる

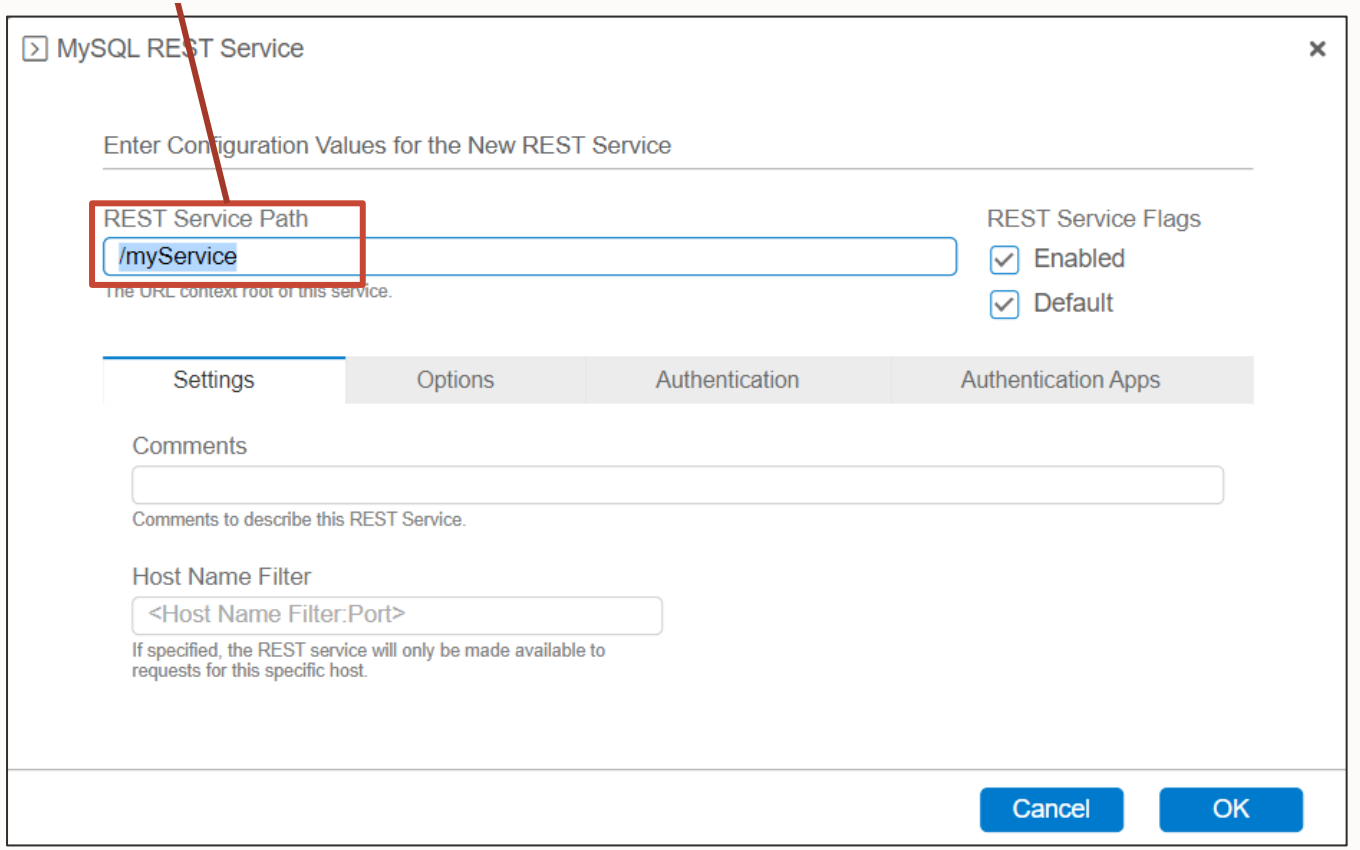
MRS の設定保存用メタデータスキーマも生成（システムスキーマも表示する設定で可視化）

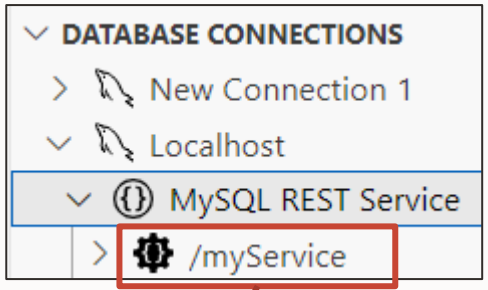
MySQL REST Service の REST ルートパスを設定



MRS オブジェクトを右クリック
Add REST Service を選択

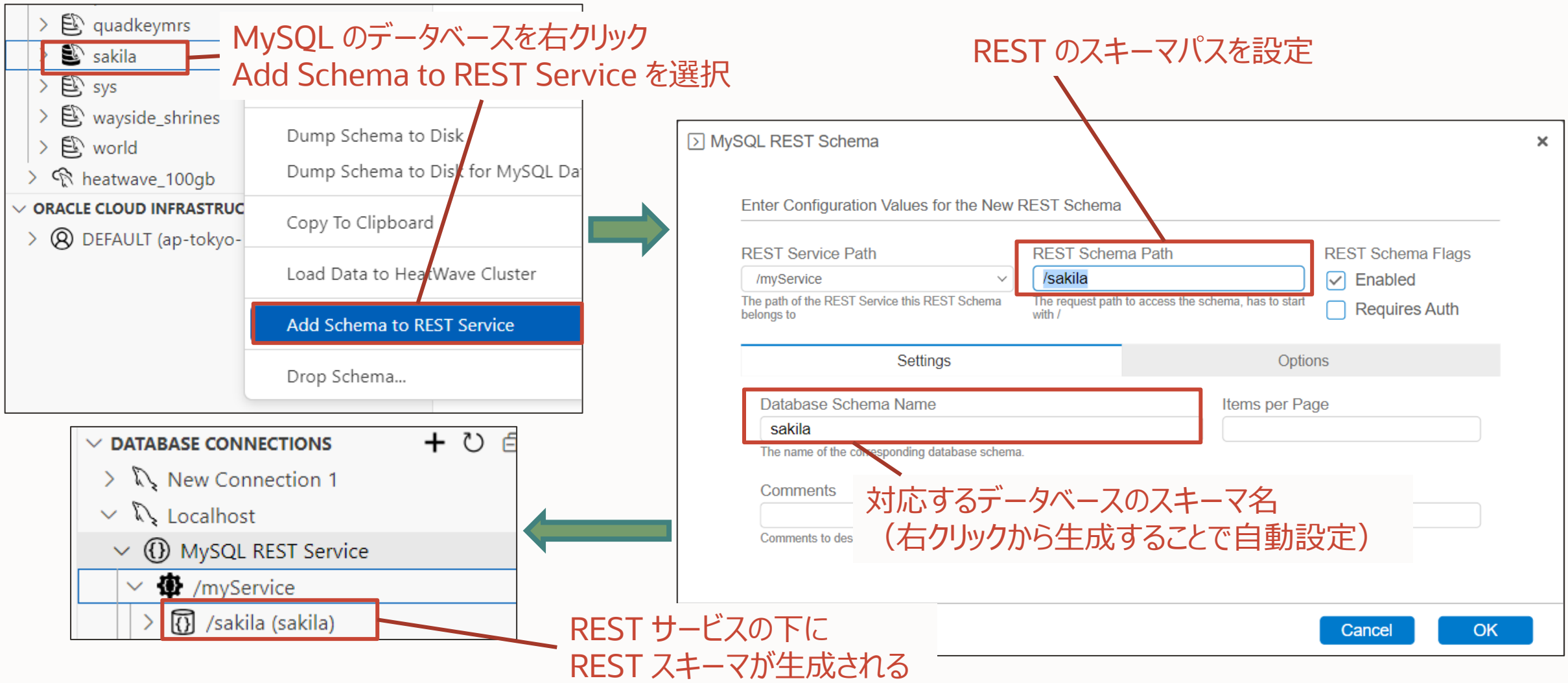
REST サービスのルートパスを
設定



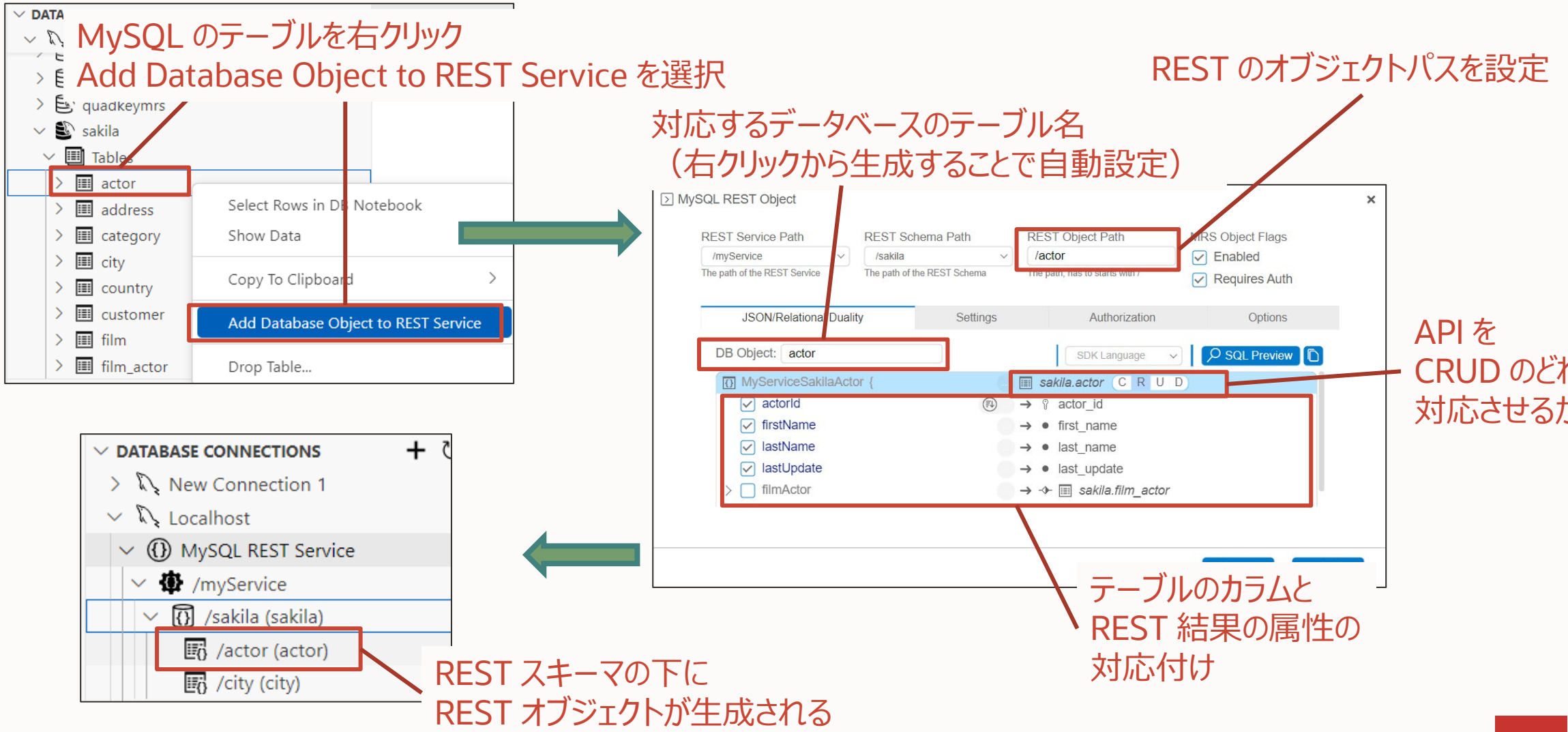


MRS オブジェクトの下に
REST サービスが生成される

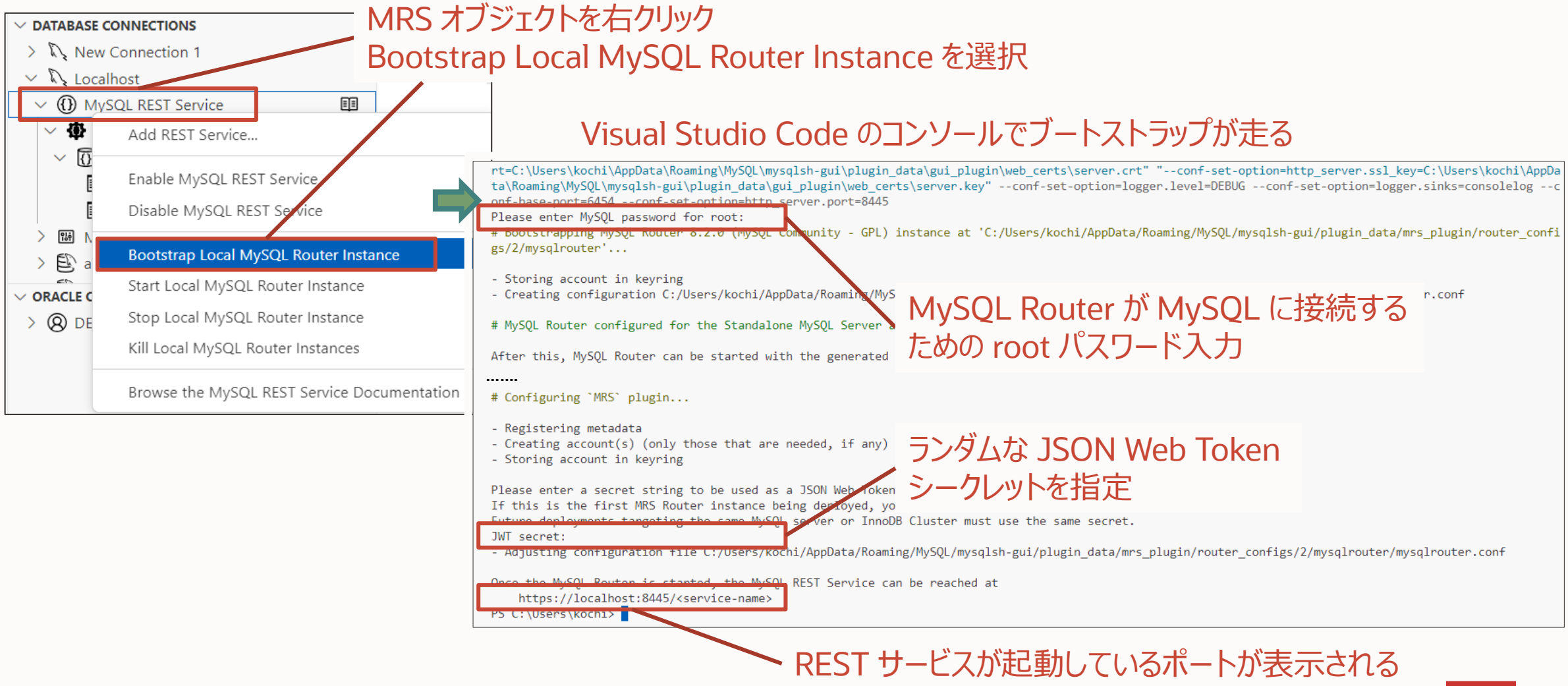
MySQL REST Service の REST スキーマを設定



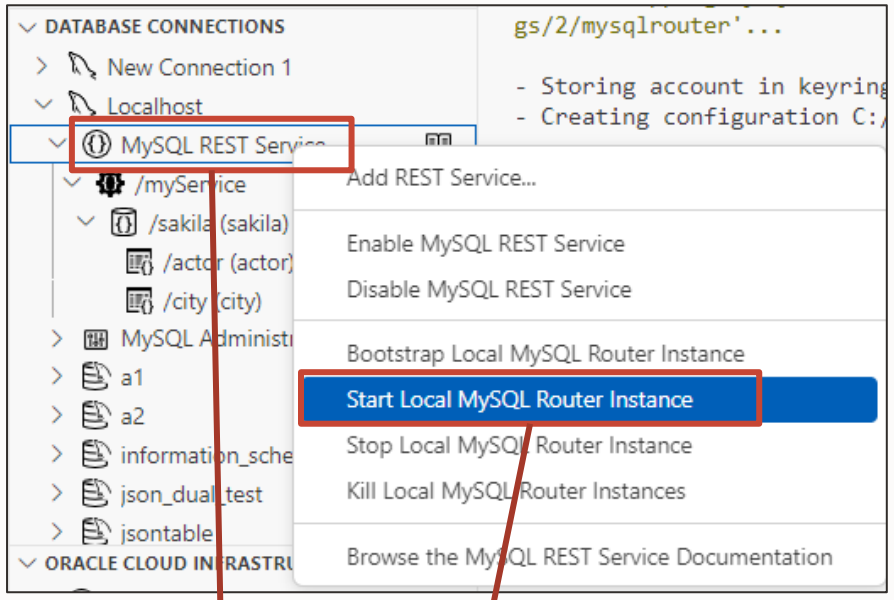
MySQL REST Service の REST データベースを設定



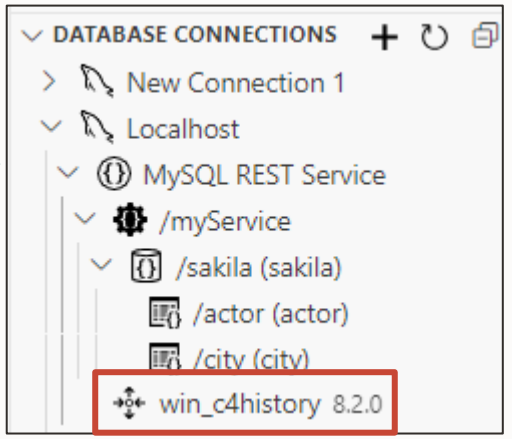
MySQL Shell for VS Code からのローカル MySQL Router 設定



ブートストラップ後の MySQL Router 起動



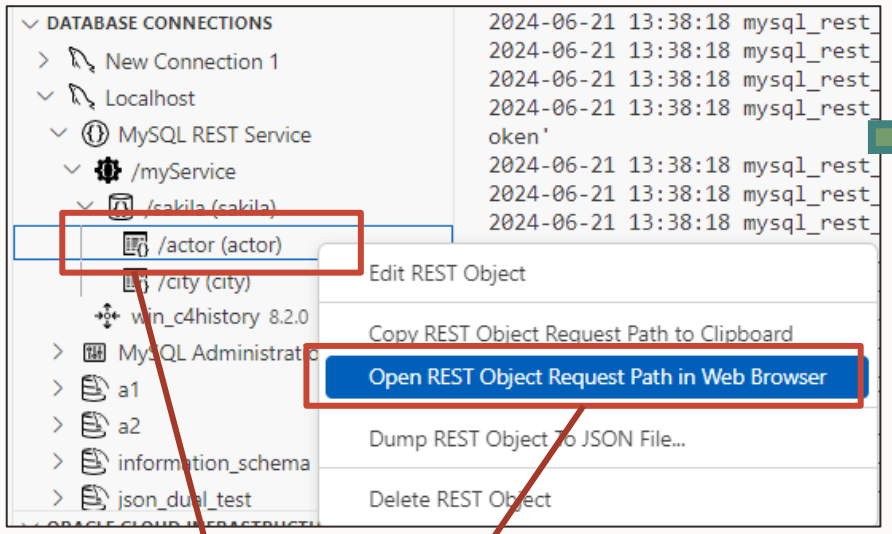
MRS オブジェクトを右クリック
Start Local MySQL Router Instance を選択



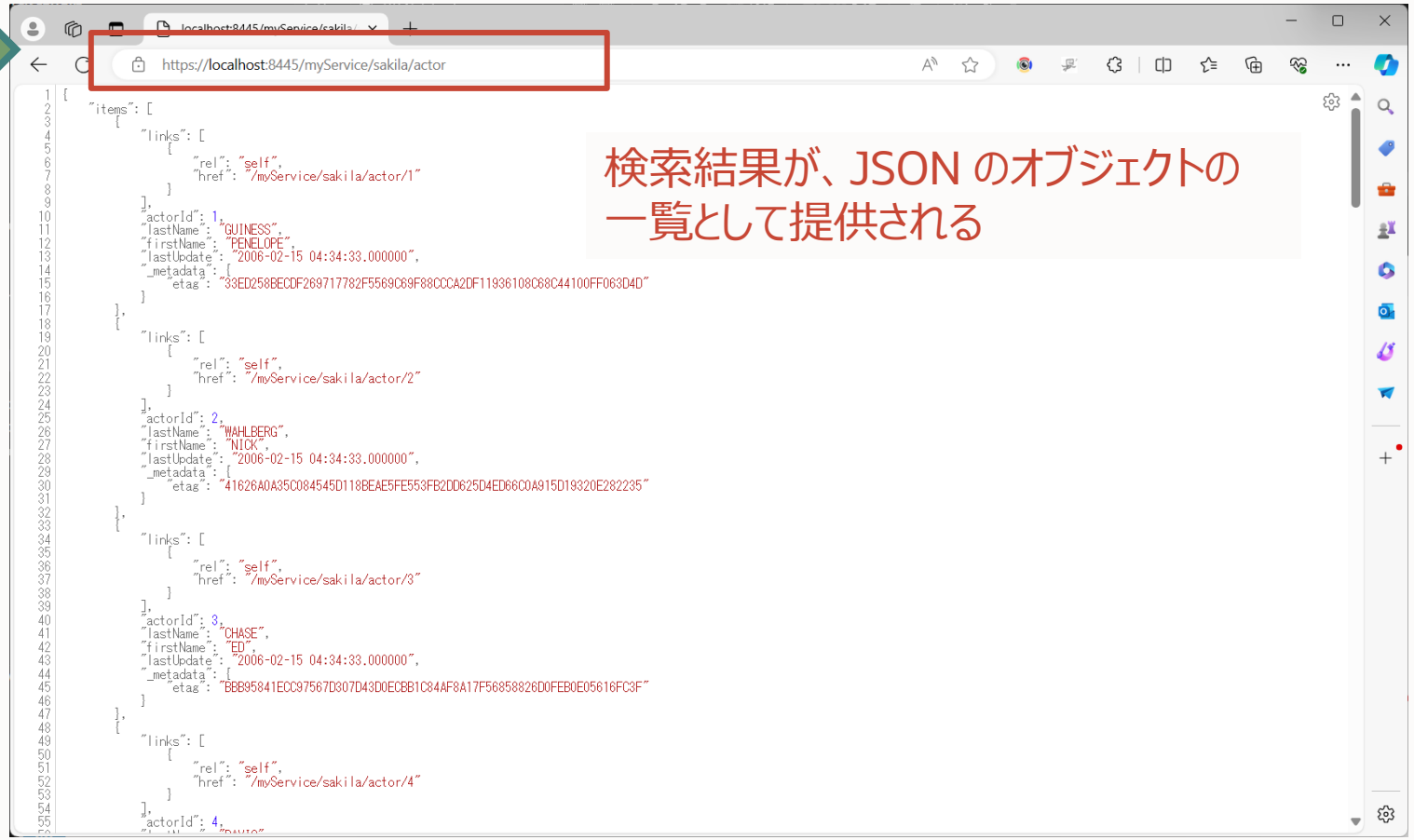
しばらく待つと、MySQL Router が起動し、
GUI 上にオブジェクトが生成される

MySQL REST Service へのアクセス

REST API の URL が自動で開かれる
（2024 年 5 月の現行版では、ポートの指定に問題があり、
実際の動作ポートに関わらず 8443 で開かれる）

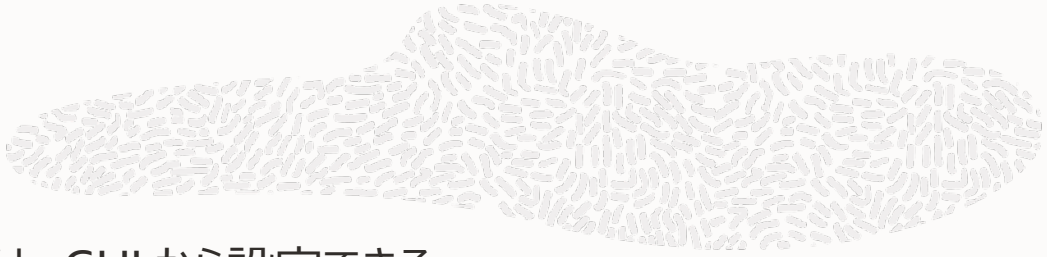


REST オブジェクトを右クリックし、
Open REST Object Request
Path in Web Browser を選択



検索結果が、JSON のオブジェクトの
一覧として提供される

MySQL REST Service の Tips（１）



- MySQL Shell for VS Code と同じマシンでの MySQL Router は、GUI から設定できる
- リモートのマシンに設定する場合には、ローカルで MySQL Shell for VS Code が生成した mysqlrouter.conf をコピーして中身を変更するのがおススメ
  - MySQL Shell for VS Code が生成する mysqlrouter.conf の場所（Windows の例）：
    - C:¥Users¥<ユーザ名>¥AppData¥Roaming¥MySQL¥mysqlsh-gui¥plugin_data¥mrs_plugin¥router_configs¥<設定番号>¥mysqlrouter¥mysqlrouter.conf
    - その他に、設定に用いられる秘密鍵である mysqlrouter.key などもこの URL にある
  - この mysqlrouter.conf をコピーし、中の設定を変更した後、MySQL Router MRS Preview をインストールしたサーバーの mysqlrouter.conf の設置場所に置く（以下、Windowsの例）
    - C:¥Program Files¥MySQL¥MySQL Router 8.2¥mysqlrouter.conf
    - C:¥Program Files¥MySQL¥MySQL Router 8.2¥mysqlrouter.ini
    - C:¥Users¥<ユーザ名>¥AppData¥Roaming/mysqlrouter.conf
    - C:¥Users¥<ユーザ名>¥AppData¥Roaming/mysqlrouter.ini
- MySQL Serverへの接続設定以外は設定ファイルに含まれないため、コピーした設定ファイルをバラまけば、分散構成も簡単に作成できる

MySQL REST Service の Tips（２）

- REST API での CRUD 処理テスト
  - READ（HTTP メソッドGET）以外（CREATE = POST、UPDATE = PUT、DELETE = DELETE）は動作確認が大変
  - POSTMAN アプリやブラウザ拡張、CURL/WGET などを活用する必要
- 引数などのテストだけであれば、TypeScript Client API を、Visual Studio for VS Code の DB Notebook（TypeScript）で試すことができる

```
ts> myService.sakila.actor.create({data: {firstName:"KOHEI", lastName:"OTSUKA"}});
{
  "links": [
    {
      "rel": "self",
      "href": "/myService/sakila/actor/201"
    }
  ],
  "actorId": 201,
  "lastName": "OTSUKA",
  "firstName": "KOHEI",
  "lastUpdate": "2024-05-20 22:28:19.000000",
  "_metadata": {
    "etag": "D12AF407EE592AC7E04D90F89E23ADA8B78AE3C02B94305EB5A97DFE8C6E0E2B"
  }
}
ts> myService.sakila.actor.delete({where:{lastName:'OTSUKA'}});
{
  "itemsDeleted": 1
}
```

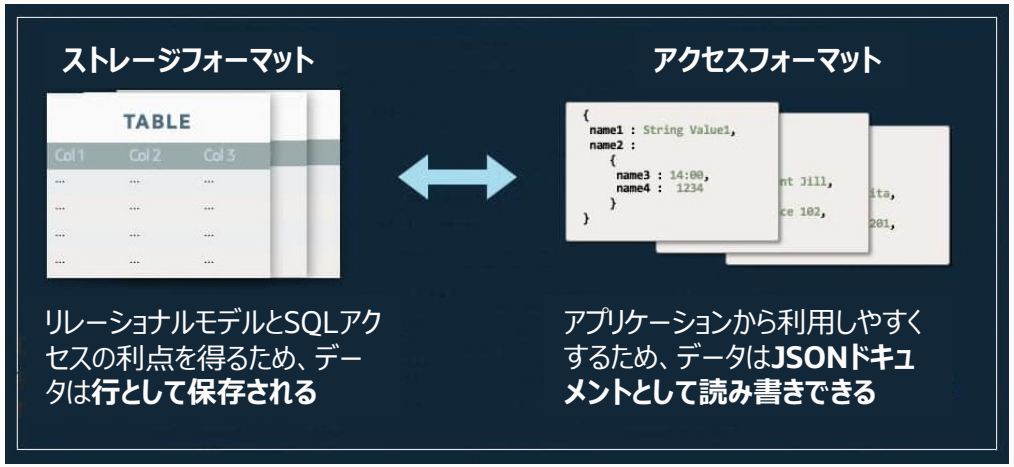
MySQL REST Service の詳細

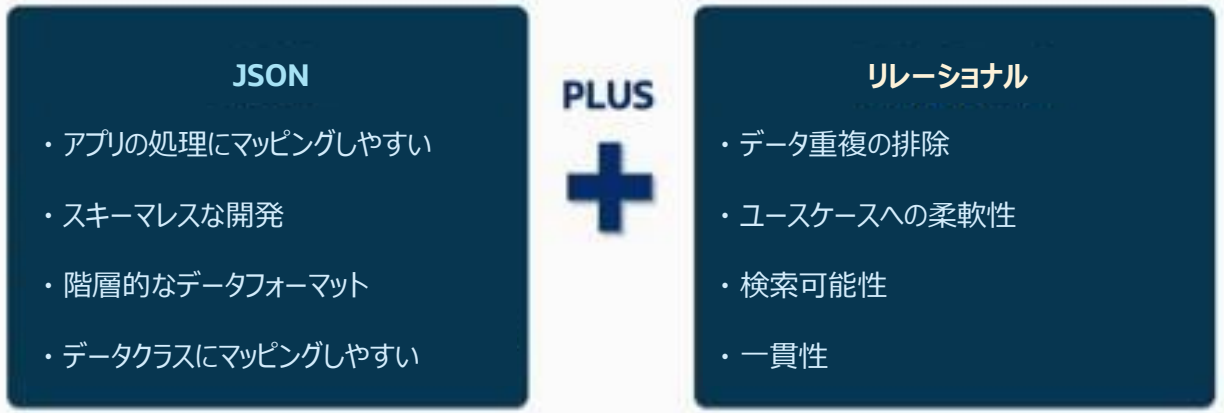
- まだ GA ではない、プレビュー状態のプロダクトです
  - ぜひ試してみていただきたいですが、プロダクトへの適用などは控えてください
  - バグや機能要望などは大歓迎しています
    https://bugs.mysql.com/
- 過去のセミナー、ウェビナー、ブログ資料などもご覧ください
  - The Oracle MySQL Japan Blog
    『MySQL REST Serviceの紹介』
    - 記事
  - MySQL REST Serviceドキュメント
    - MRS Developer's Guide (mysql.com)

3-3. JSON Relational Duality

JSON Relational Duality（JSONとリレーショナルの二面性）とは

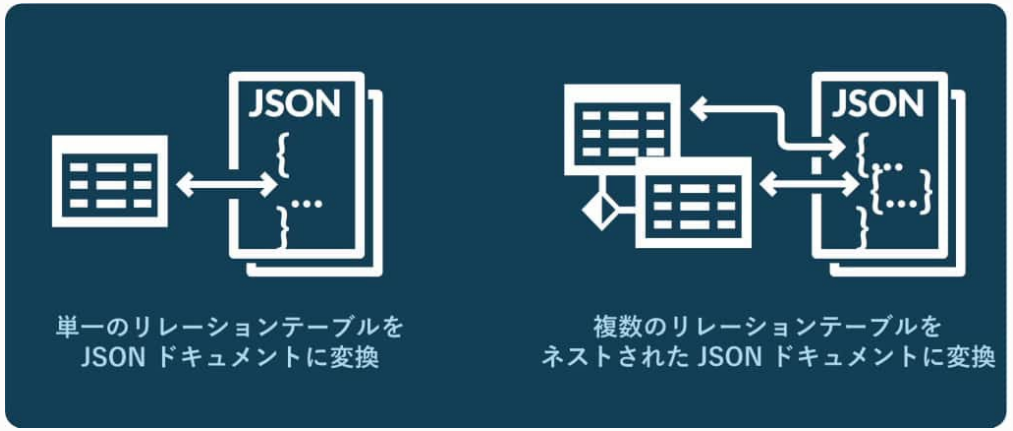
- Oracle Database 23c Free - Developer Release で導入された概念（ブログ）
- データは正規化されたリレーショナルテーブルに保存されるが、JSON ドキュメントとして CRUD 処理ができる
- テーブルとドキュメント、双方の特長をいいとこ取り
- 開発者はお気に入りのドライバを用いて、テーブル、ドキュメントの好きな形式でアクセスできる

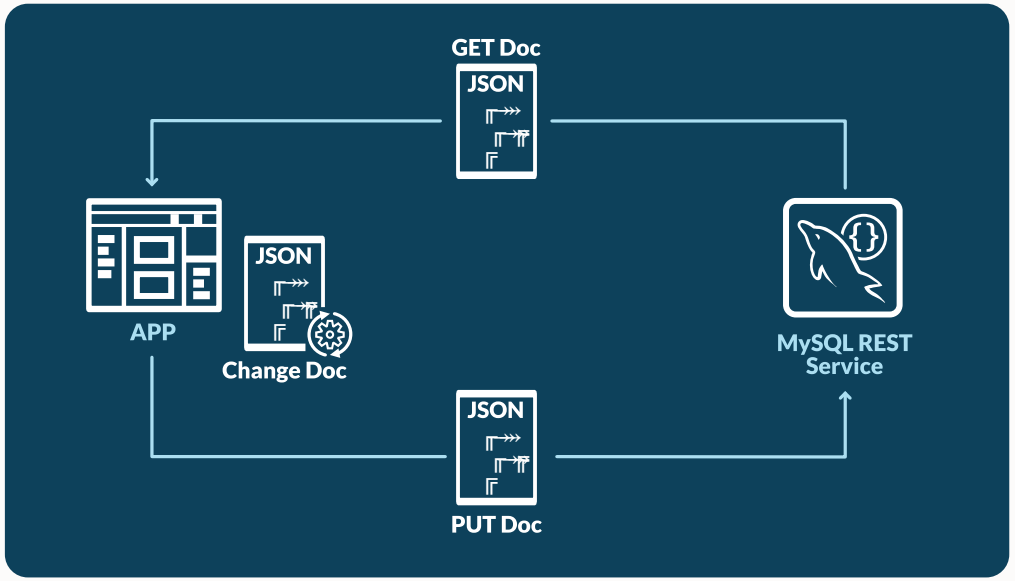




MySQL REST Service は JSON Relational Duality にも対応

- MySQL も MySQL REST Service で JSON Relational Duality に対応
- 単一のテーブルは単階層の JSON ドキュメントに、外部制約を持つリレーションテーブルはネストされた JSON ドキュメントに変換
- REST でのドキュメント管理ワークフロー
  - REST Duality ビューから GET でドキュメントを取得
  - ネストされた JSON ドキュメントへの変更を含め、必要な変更を加える
  - ドキュメントを REST Duality ビューに PUT で返す

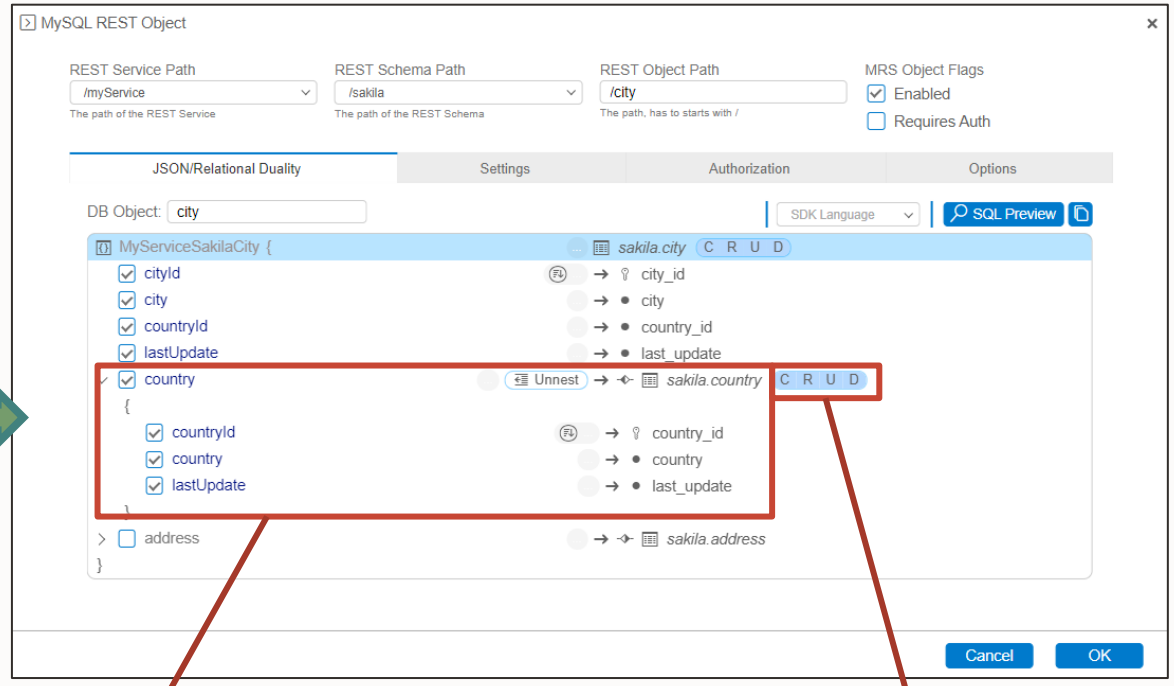




外部制約を持つテーブルを、REST オブジェクト作成で選ぶ

```
CREATE TABLE `country` (
  `country_id` smallint unsigned NOT NULL
AUTO_INCREMENT,
  `country` varchar(50) NOT NULL,
  `last_update` timestamp NOT NULL DEFAULT
CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`country_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `city` (
  `city_id` smallint unsigned NOT NULL AUTO_INCREMENT,
  `city` varchar(50) NOT NULL,
  `country_id` smallint unsigned NOT NULL,
  `last_update` timestamp NOT NULL DEFAULT
CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`city_id`),
  KEY `idx_fk_country_id` (`country_id`),
  CONSTRAINT `fk_city_country` FOREIGN KEY
(`country_id`) REFERENCES `country` (`country_id`) ON
DELETE RESTRICT ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```



リレーション先のテーブルの属性を、ネストされた JSON ドキュメントの属性として選択できる

リレーション先のテーブルが対応する CRUD は、個別に設定できる

ネストした JSON オブジェクトの取得例（REST）


```
{
    "items": [
        {
            "city": "Abu Dhabi",
            "links": [
                {
                    "rel": "self",
                    "href": "/myService/sakila/city/3"
                }
            ],
            "cityId": 3,
            "country": [
                "country": "United Arab Emirates",
                "countryId": 101,
                "lastUpdate": "2006-02-15 04:44:00.000000"
            ],
            "countryId": 101,
            "lastUpdate": "2006-02-15 04:45:25.000000",
            "_metadata": [
                "etag": "31ABC948ED5F707931D3F5CD5B3E351506ECD66221807688E3BE281AB7C78CF8"
            ]
        }
    ],
    "limit": 25,
    "offset": 0,
    "hasMore": false,
    "count": 1,
    "links": [
        {
            "rel": "self",
            "href": "/myService/sakila/city/"
        }
    ]
}
```

ネストした JSON オブジェクトの作成例（TypeScript Client API）

```
ts > myService.sakila.city.create({data: {city: "Tatebayashi", country: {country: "Gumma"}}});

    {
        "city": "Tatebayashi",
        "links": [
            {
                "rel": "self",
                "href": "/myService/sakila/city/602"
            }
        ],
        "cityId": 602,
        "country": {
            "country": "Gumma",
            "countryId": 110,
            "lastUpdate": "2024-05-21 02:10:38.000000"
        },
        "countryId": 110,
        "lastUpdate": "2024-05-21 02:10:38.000000",
        "_metadata": {
            "etag": "16976A78B2D8FB1BC5539338D4B3817C284D968D26A41BDC2753A386434378F7"
        }
    }
```

作成するネスト JSON オブジェクト
REST で作成する際は、これを POST する

JSON Relational Duality の詳細

- MySQL REST Serviceと同様、まだ GA ではない、プレビュー状態のプロダクトです
  - ぜひ試してみていただきたいですが、プロダクトへの適用などは控えてください
  - バグや機能要望などは大歓迎しています
    https://bugs.mysql.com/
- 過去のセミナー、ウェビナー、ブログ資料などもご覧ください
  - The Oracle MySQL Japan Blog
    『MySQL REST Service での JSON Relational Duality 機能の紹介』
    - 記事
  - MySQL REST Serviceドキュメント（JSON-Relational Duality Views）
    - MRS Developer's Guide: JSON-Relational Duality Views (mysql.com)

4. 本セッションのまとめ

本セッションのまとめ

MySQL REST Serviceでも使える JSON Relational Duality技術

- 新しいMySQLのバージョンモデル
  - 四半期に一度、イノベーションとセキュリティ・バグフィクスを提供する Innovation Release (IR)
    2023 年 7 月の 8.1 から 8.2、8.3、次は 2024 年 7 月
  - 2 年に一度、8 年間サポートを保証する Long Term Support (LTS)
    2024 年 4 月の 8.4 が最初の LTS
  - 8.0 のサポートは 2026 年 4 月まで
- MySQL での JSON の使い方
  - JSON データ型、MySQL ドキュメントストア、MySQL REST Service（with JSON Relational Duality）
- JSON Relational Duality
  - Oracle Database 23c Free - Developer Release で示された概念
  - テーブルとドキュメント、双方の特長をいいとこ取り
  - MySQL では MySQL REST Service の中で実現
  - MySQL REST Service は MYSQL Shell for VS Code と MySQL Router MRS Preview との組み合わせで実現
  - まだ GA ではないプレビューの機能、バグや機能要望は歓迎

5. MySQL 製品 / サービス / コミュニティのご紹介

柔軟なMySQLの利用方法

MySQLサーバーは全て共通のソースコードのためハイブリッド構成も可能

MySQLを自社で運用管理

オンプレミスでのMySQL

- バージョン選択や構成を最も柔軟に選択可能

IaaS上でのMySQL

- OCIのマーケットプレイスのイメージから簡単に環境構築

商用版MySQL

- コミュニティ版に加え、サポートやセキュリティに優れた商用版も

MySQLのマネージドサービス

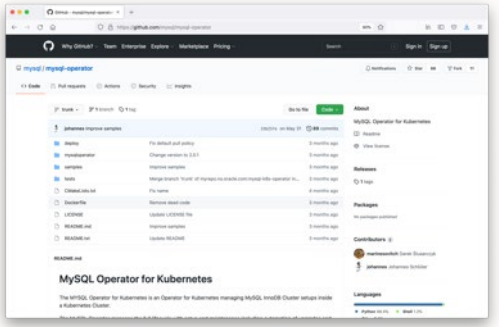
MySQL HeatWave Database Service

- MySQLチームが100%開発・提供するクラウド・サービス
- Amazon RDS (MySQL)の1/3以下のコスト
- データ損失なし、自動フェイルオーバーの高可用性機能をマネージドサービスで提供

クラウドネイティブなMySQL

MySQL Operator for k8s

- MySQL サーバーをKubernetes上に構築し運用管理



MySQL :: MySQL Operator for Kubernetes テクニカルアップデート

いずれの利用方法でもMySQL開発チームと連携した専門部隊によるサポートサービスをご利用いただけます※

※コミュニティ版を除く

Oracle Premier Support for MySQL

- 最大のMySQLのエンジニアリングおよびサポート組織
- MySQL開発チームによるサポート
- 29言語で世界クラスのサポートを提供
- メンテナンス・リリース、バグ修正、パッチ、アップデートの提供
- 24時間x365日サポート
- MySQL コンサルティング・サポート

Get immediate help for any MySQL issue, plus expert advice

開発チームと一体となったサポートサービス

⇒ 商用版MySQL サーバー及びMySQL HeatWave Database Serviceにより提供

年間サブスクリプション 74.9万円 (1サーバーあたり)

MySQLの最新情報配信

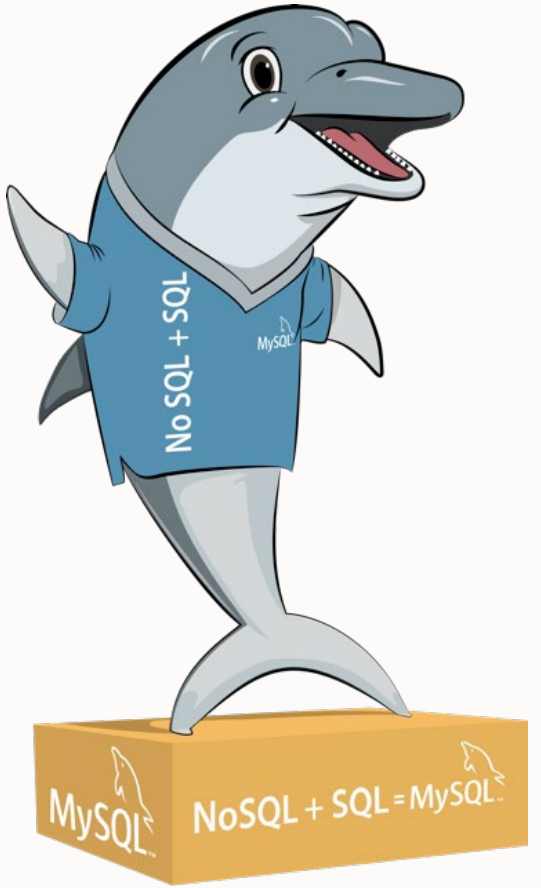
MySQLホームページ
http://www.mysql.com/jp

MySQL イベント
http://www.mysql.com/jp/news-and-events/

MySQLニュースレター 英語版&日本語版（月刊）
https://www.mysql.com/jp/news-and-events/newsletter/

The Oracle MySQL Japan ブログ
https://blogs.oracle.com/mysql-jp/

MySQL X（旧Twitter）日本語公式アカウント
@mysql_jp



MySQL コミュニティ

- MySQL コミュニティの紹介
- MySQL への貢献
  - Oracle Contribution Agreement (OCA)
- MySQL無償認証制度

MySQL コミュニティの紹介

- MySQL コミュニティへの貢献プロセスの運営
- MySQL ユーザーグループへの支援 **https://dev.mysql.com/community/mug/**
- 全世界でのサードパーティによるカンファレンスやイベントへの支援や参加 **https://dev.mysql.com/community/**
- 教育ビデオの作成
  - MySQL 短編動画 (MySQL Shorts)
  - MySQL 入門編シリーズ (MySQL 101 for Biginners)
    - **https://www.youtube.com/@mysql**
- MySQL RockStar プログラム
  - MySQLの利用促進に最も精力的に取り組んだ MySQL コミュニティ・メンバーへの表彰
  - 第1回: **https://blogs.oracle.com/mysql/post/mysql-rockstars-2022**
- MySQL ACE プログラム
  - MySQL プロジェクトでの ACE プログラムの運営
  - **https://ace.oracle.com/pls/apex/ace_program/r/oracle-aces/home**

MySQL への貢献

- MySQL オープンソースプロジェクトのコントリビューターコミュニティへの参加: **https://forums.oracle.com/ords/apexds/post/contributing-code-to-mysql-8037**
- コントリビュータになるために持つべきこと
  - MySQL の機能を変更/修正したい、あるいは新しい機能を追加したいといった要望
  - MySQL ソースコードのダウンロード **http://dev.mysql.com/downloads/**
  - bugs.mysql.com のアカウント **http://bugs.mysql.com** or
  - 有効な GitHub アカウント **https://github.com**
- Oracle Contribution Agreement (OCA) への署名 **https://oca.opensource.oracle.com/**
  - OCAは、コントリビューターとオラクルの両方を法的攻撃から保護する短い法的契約です。OCAに署名することにより、コントリビューターはオラクルがコントリビューターのコードをオラクル・ソフトウェアで使用することが法的に許可されていること、およびコントリビューターの知る限りにおいて、そのコードに特許的な問題がないことに同意することになります。

MySQL 無償認証制度

- MySQL コミュニティチームは、Oracle University および Oracle Academy と協力し、mylearn.oracle.com を介して、2ヶ月間の指定期間内に使用できる無料のトレーニングバウチャー/クレジットを受講者に提供します。
- ご興味のある方は、以下についての詳細をお知らせくだされば、MySQL コミュニティから連絡いたします。
  - 名前
  - 姓
  - Email アドレス
  - 居住国
- **https://education.oracle.com/**

連絡先

- **MySQL コミュニティとのコンタクト先一覧:**
- MySQL コミュニティページ, https://dev.mysql.com/community/
- MySQL Slack, https://mysqlcommunity.slack.com
- The Oracle MySQL ブログ, https://blogs.oracle.com/mysql/
- The Oracle MySQL Japan ブログ, https://blogs.oracle.com/mysql-jp/
- Planet MySQL, https://planet.mysql.com/
- LinkedIn, https://www.linkedin.com/groups/60715/
- ブログ, https://lefred.be/
- MySQL フォーラム, http://lists.mysql.com/
- ディスカッションフォーラム, http://forums.mysql.com


ORACLE