MÓDULO 1 - EXTRA

JDBC

UNIPDS
EDUCAÇÃO

Definindo um MySQL com Docker Compose

```yaml
services:
 db.cardapio:
   image: mysql:9
   restart: always
   environment:
     MYSQL_ROOT_PASSWORD: senha123
   ports:
     - '3306:3306'
   volumes:
     - mysql.cardapio:/var/lib/mysql
volumes:
 mysql.cardapio:
```

Define um MySQL 9

Volume para manter dados ao reiniciar container

Executar com o comando: `docker compose up`

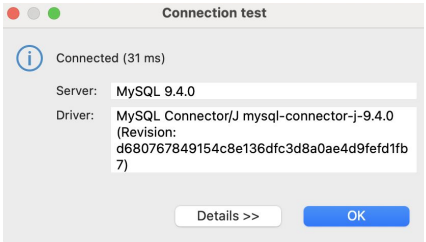
Instalando e Configurando o DBeaver

DBeaver Community é free e muito usado para manipular diferentes DBs.

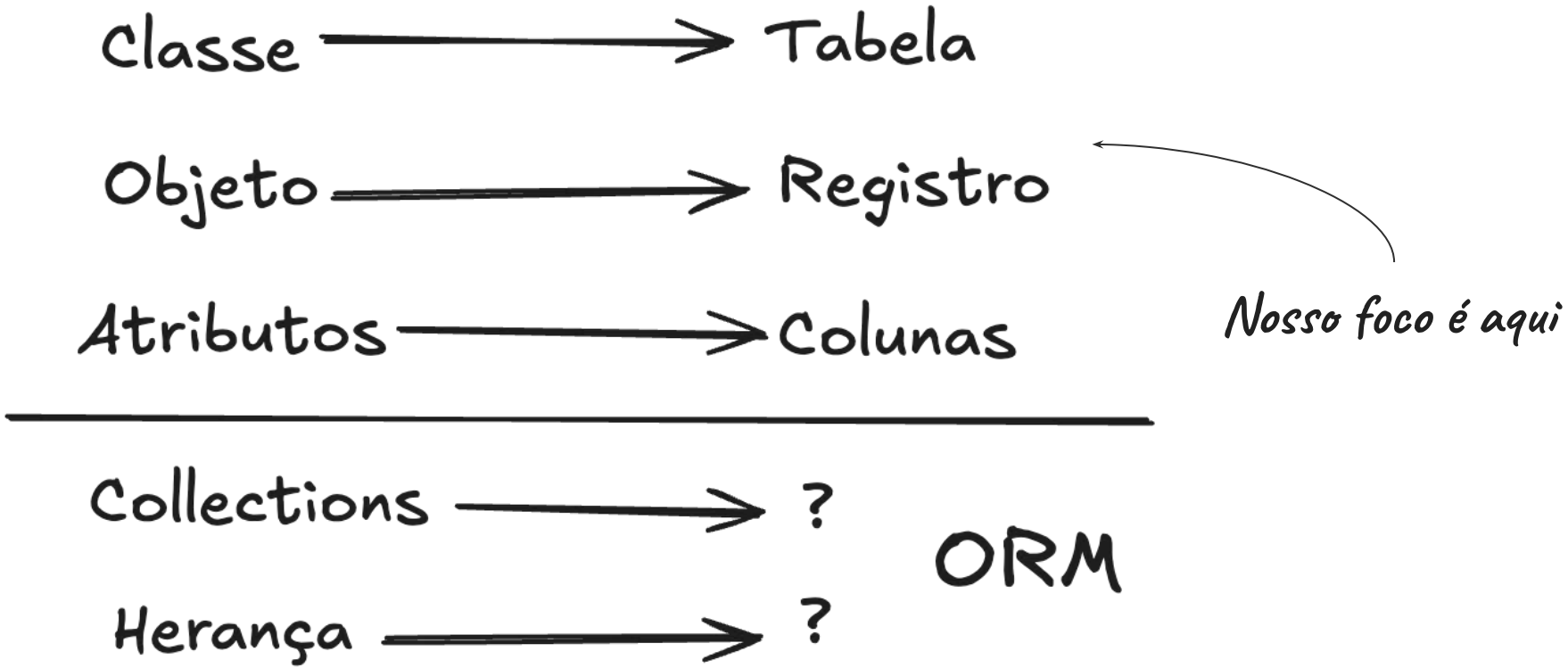
Baixe o DBeaver Community no link: https://dbeaver.io/download/

Para configurar o **MySQL 9** no **DBeaver** Community:

- Ir em "**Database > New Database Connection**"
- Selecionar o **MySQL** e manter Server Host como localhost, Port como 3306 e User como root
- Modificar a **Password** colocando a senha definida no Docker Compose (**senha123**)
- Ao clicar em "**Test Connection...**" deve aparecer para editarmos o Driver da primeira vez
  - Na aba "**Libraries**", remova o mysql-connector-j da **RELEASE 8.x.x**
  - Clique em **"Add Artifact"** e na aba "**Declare Artifact Manually**" define groupId como "**com.mysql**", artifactId como "**mysql-connector-j**" e version como "**9.4.0**"
- Faça o Download e teste a conexão novamente. Ocorrerá um erro de "Public Key Retrieval"
- Na aba "**Driver Properties**", habilite a propriedade "**allowPublicKeyRetrieval**"
- Agora a conexão com o MySQL 9 deve funcionar!
- Conecte em **"localhost:3306"**
- Clique em "**Create > Database**" e crie o database "**cardapio**"



Representando um Objeto em um BD Relacional



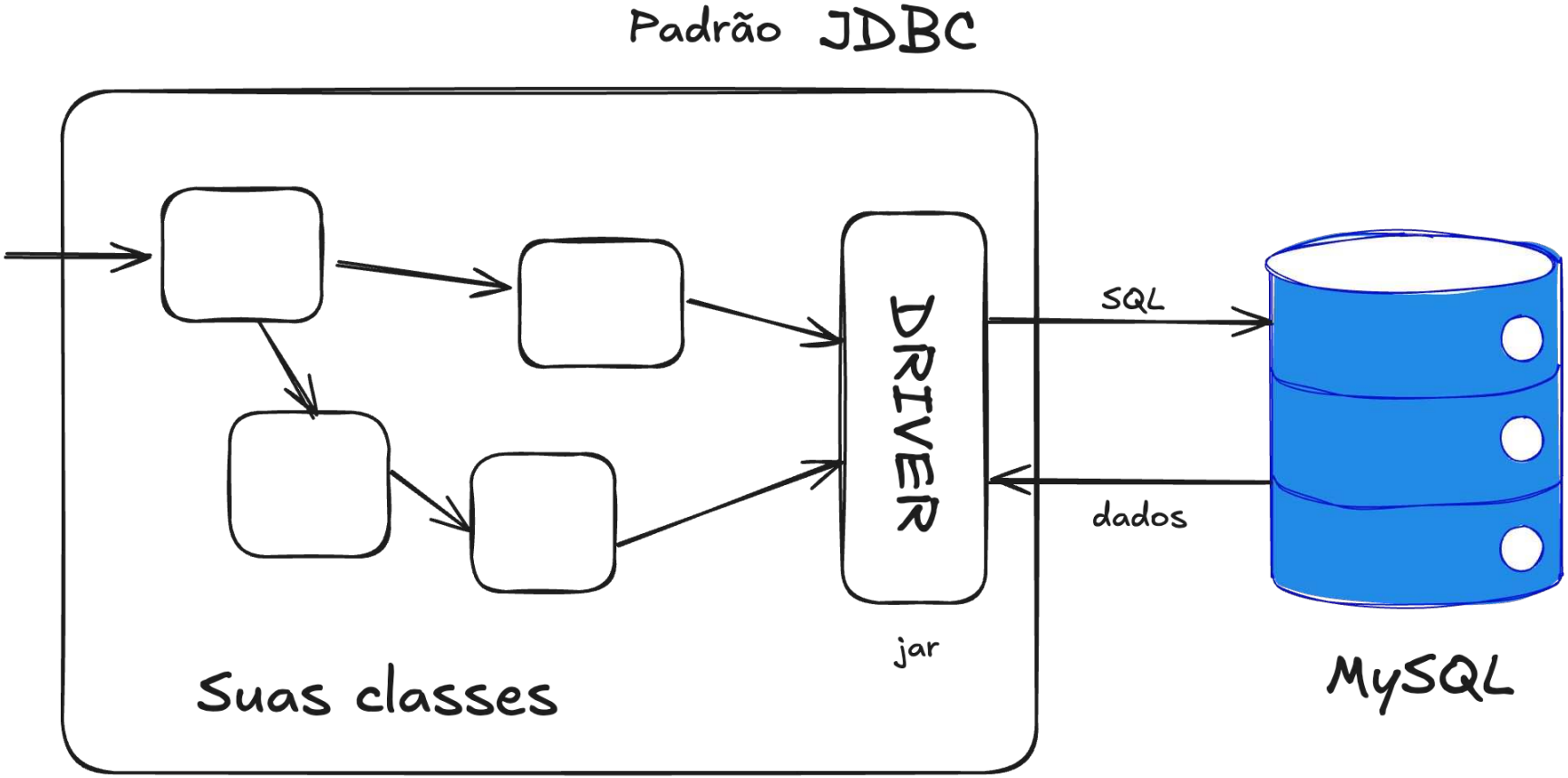
Criando a tabela e os itens de cardápio

```sql
create table item_cardapio (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    nome VARCHAR(100) NOT NULL,
    descricao VARCHAR(1000),
    categoria ENUM('ENTRADAS', 'PRATOS_PRINCIPAIS', 'BEBIDAS', 'SOBREMESA') NOT NULL,
    preco DECIMAL(9, 2) NOT NULL,
    preco_promocional DECIMAL(9, 2)
);
INSERT INTO item_cardapio (nome, descricao, categoria, preco, preco_promocional) VALUES
     ('Refresco do Chaves', 'Suco de limão que parece de tamarindo e tem gosto de groselha.',
'BEBIDAS', 2.99, NULL),
     ('Sanduíche de Presunto do Chaves', 'Sanduíche de presunto simples, mas feito com muito
amor.', 'PRATOS_PRINCIPAIS', 3.50, 2.99),
     ('Torta de Frango da Dona Florinda', 'Torta de frango com recheio cremoso e massa
crocante.', 'PRATOS_PRINCIPAIS', 12.99, 10.99),
     ('Pipoca do Quico', 'Balde de pipoca preparado com carinho pelo Quico.',
'PRATOS_PRINCIPAIS', 4.99, 3.99),
     ('Água de Jamaica', 'Água aromatizada com hibisco e toque de açúcar.', 'BEBIDAS', 2.50,
2.00),
     ('Churros do Chaves', 'Churros recheados com doce de leite, clássicos e irresistíveis.',
'SOBREMESA', 4.99, 3.99);
```

Selecionados tudo e contando os itens

```sql
SELECT id, nome, descricao, categoria, preco, preco_promocional FROM item_cardapio;

SELECT count(*) FROM item_cardapio;
```

O padrão JDBC e o Driver



Definindo o Driver no Gradle

Home » com.mysql » mysql-connector-j » 9.4.0

MySQL Connector/J » 9.4.0

MySQL Connector/J is a JDBC Type 4 driver, which means that it is pure Java implementation of the MySQL protocol and does not rely on the MySQL client libraries. This driver supports auto-registration with the Driver Manager, standardized validity checks, categorized SQLExceptions, support for large update counts, support for local and offset date-time variants from the java.time package, support for JDBC-4.x XML processing, support for per connection client information and support for the NCHAR, NVARCHAR ...

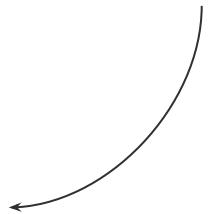
Categories	JDBC Drivers
Tags	database sql jdbc driver connector rdbms mysql connection
Organization	Oracle Corporation
HomePage	http://dev.mysql.com/doc/connector-j/en/
Date	Jul 29, 2025
Files	pom (3 KB) jar (2.5 MB) View All
Repositories	Central
Ranking	**#477 in MvnRepository (See Top Artifacts)** **#7 in JDBC Drivers**
Used By	**1,229 artifacts**

Maven | Gradle | SBT | Mill | Ivy | Grape | Leiningen | Buildr

Scope: Compile Format: Groovy Short

```
implementation 'com.mysql:mysql-connector-j:9.4.0'
```

Adicionar no `build.gradle`



```
implementation 'com.mysql:mysql-connector-j:9.4.0'
```

https://mvnrepository.com/artifact/com.mysql/mysql-connector-j/9.4.0

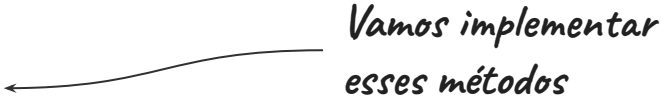
Refatorando código atual


```
public interface Database {
    List<ItemCardapio> listaItensCardapio();

    Optional<ItemCardapio> itemCardapioPorId(Long id);

    boolean removeItemCardapio(Long id);

    boolean alteraPrecoItemCardapio(Long id, BigDecimal novoPreco);

    int totalItensCardapio();

    void adicionaItemCardapio(ItemCardapio item);
}
```

Extrair interface

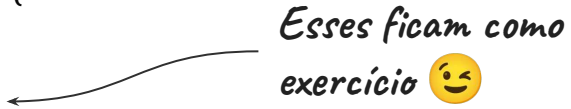
Implementando SQLDatabase

```
public class SQLDatabase implements Database {

    @Override
    public List<ItemCardapio> listaItensCardapio() {
        return List.of();
    }

    @Override
    public int totalItensCardapio() {
       return 0;
    }

    @Override
    public void adicionaItemCardapio(ItemCardapio item) {
    }

    //...
}
```



Implementando SQLDatabase - continuação

```
public class SQLDatabase implements Database {

    //...

    @Override
    public Optional<ItemCardapio> itemCardapioPorId(Long id) {
        throw new UnsupportedOperationException("TODO");
    }

    @Override
    public boolean removeItemCardapio(Long id) {
        throw new UnsupportedOperationException("TODO");
    }

    @Override
    public boolean alteraPrecoItemCardapio(Long id, BigDecimal novoPreco) {
        throw new UnsupportedOperationException("TODO");
    }


}
```



Implementando listagem com JDBC

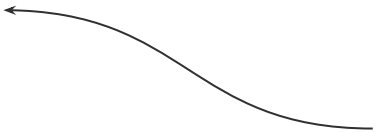
```
List<ItemCardapio> itensCardapio = new ArrayList<>();

String sql = "SELECT id, nome, descricao, categoria, preco, preco_promocional FROM item_cardapio";
try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cardapio",
                                                   "root", "senha123");
   PreparedStatement ps = conn.prepareStatement(sql);
   ResultSet rs = ps.executeQuery()){
   while (rs.next()) {
       long id = rs.getLong("id");
       String nome = rs.getString("nome");
       String descricao = rs.getString("descricao");
       String categoriaStr = rs.getString("categoria");
       BigDecimal preco = rs.getBigDecimal("preco");
       BigDecimal precoPromocional = rs.getBigDecimal("preco_promocional");

       ItemCardapio itemCardapio = new ItemCardapio(id, nome, descricao,
               ItemCardapio.CategoriaCardapio.valueOf(categoriaStr), preco, precoPromocional);

       itensCardapio.add(itemCardapio);
   }
} catch (SQLException e) {
   throw new RuntimeException(e);
}
```

Testando listagem JDBC

```
Database database = new SQLDatabase();

List<ItemCardapio> listaItensCardapio = database.listaItensCardapio();
listaItensCardapio.forEach(System.out::println);
```

Implementando total com JDBC

```
String sql = "SELECT count(*) FROM item_cardapio";

try (Connection conn =
        DriverManager.getConnection("jdbc:mysql://localhost:3306/cardapio",
                                    "root", "senha123");
   PreparedStatement ps = conn.prepareStatement(sql);
   ResultSet rs = ps.executeQuery()){

   int count = 0;
   if (rs.next()) {
       count = rs.getInt(1);
   }
   return count;

} catch (SQLException e) {
   throw new RuntimeException(e);
}
```



Se colocar 0, lança java.sql.SQLException: Column Index out of range, 0 < 1

Testando total JDBC

```
Database database = new SQLDatabase();

//...

int total = database.totalItensCardapio();
System.out.println(total);
```

Implementando adição com JDBC

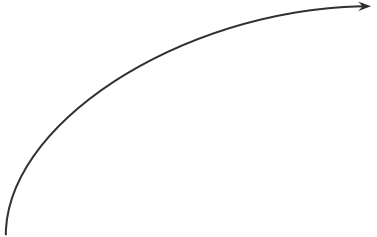
```
String sql = "INSERT INTO item_cardapio (id, nome, descricao, categoria, preco,
preco_promocional) VALUES (?, ?, ?, ?, ?, ?)";

try (Connection conn =
            DriverManager.getConnection("jdbc:mysql://localhost:3306/cardapio",
                                                    "root", "senha123");
    PreparedStatement ps = conn.prepareStatement(sql)) {

       ps.setLong(1, item.id());
       ps.setString(2, item.nome());
       ps.setString(3, item.descricao());
       ps.setString(4, item.categoria().name());
       ps.setBigDecimal(5, item.preco());
       ps.setBigDecimal(6, item.precoPromocional());

       ps.execute();

} catch (Exception e) {
   throw new RuntimeException(e);
}
```

Testando adição com JDBC

```
Database database = new SQLDatabase();

var novoItemCardapio = new ItemCardapio(10L, "Tacos de Carnitas",
        "Incríveis tacos recheados com carne tenra",
        ItemCardapio.CategoriaCardapio.PRATOS_PRINCIPAIS,
        new BigDecimal("25.9"), null);

database.adicionaItemCardapio(novoItemCardapio);
```

Usando JDBC no nosso servidor

```
private static final Database database =
                                    new InMemoryDatabase()
                                    new SQLDatabase();
```



Mudar metade de uma linha muda completamente o que o código faz! O poder de OO e do Polimorfismo! 😮

Melhorias a serem implementadas

- Separar o que é conexão do BD do que é específico de uma tabela
  - ConnectionFactory vs ItemCardapioDAO
- Verificar como SQLDatabase se comporta com múltiplos usuários
- Usar Pool de Conexões como HikariCP

DESAFIO

Detalhamento, remoção e alteração de preço na API de Itens de Cardápio.

Implemente os seguintes endpoints no nosso Servidor HTTP com Socket usando a nossa classe `Database` como banco de dados.

- `GET /itens-cardapio/{id}` - Detalha os dados de um item do cardápio
- `DELETE /itens-cardapio/{id}` - Remove um item do cardápio
- `PATCH /itens-cardapio/{id}` - Altera o preço de um item do cardápio enviando o novo preço no corpo da requisição

Teste usando o `curl`.