Fundamentos de Banco de Dados

Msc. Maiara Coelho
Instituto de Computação
UFAM

Conceitos Básicos

- **Dado**: elemento básico de informação, está em algum tipo de registro. Ex: nome, idade.
- **Informação**: conjunto de dados que possuem um significado. Ex: endereço de entrega.
- **Banco de Dados (BD):** coleção de dados inter relacionados, persistentes e acessíveis que representa um subconjunto de dados pertencentes a um domínio de aplicação.

Conceitos Básicos

- **Banco de Dados Relacional**
  - Organizados em tabelas permitindo o relacionamento entre as mesmas.
  - Padrão adotado mundialmente, maior velocidade de acesso aos dados e menor espaço de armazenamento.
- **Banco de Dados Não-Relacional**
  - Lidam bem com dados desestruturados (e-mails, dados multimídia e dados provenientes de mídias sociais).

Conceitos Básicos

- **Banco de Dad**
  - Organizad onamento entre as mesmas.
  - Padrão ad ade de acesso aos dados e me
- **Banco de Dac**
  - Lidam be s (e-mails, dados multimídia ociais).

Modelos de Dados

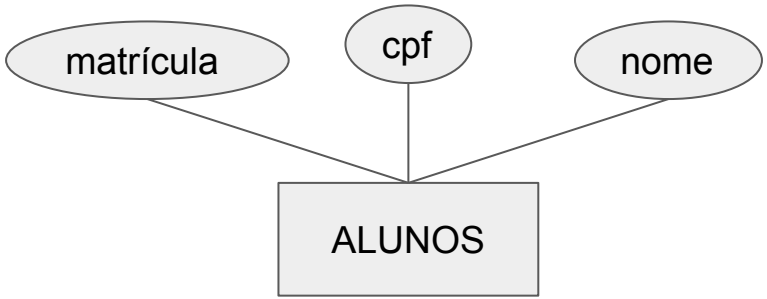
- Um modelo de dados é uma coleção de ferramentas conceituais para a descrição de dados, relacionamentos, semântica de dados e restrições de consistência.
- O MER (Modelo-Entidade-Relacionamento) é um modelo de dados conceitual de alto nível.
- Entidade - Relacionamento - Associação entre entidades ou com a mesma.
- Atributo - Características únicas para diferentes entidades.

Modelagem de Dados: Entidade

- Entidade - Representação abstrata de um objeto do mundo real sobre o qual desejamos guardar informações. Deve ser identificada de modo único.

Modelagem de Dados: Atributo

- Atributo - Características de uma entidade.



Modelagem de Dados: Atributos Chave

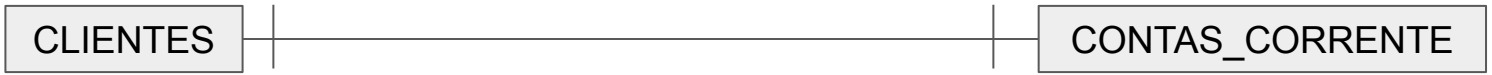
- Chave - Tem a função de identificar o elemento dentro do conjunto. Uma chave pode ter diversas classificações:
  - Chave Primária - ÚNICA para cada elemento da entidade.
  - Chave Candidata - ÚNICO mas, está sendo usado como atributo comum.
  - Chave Estrangeira - Chave primária em outra entidade.
  - Superchave - Formado pela análise conjunta das chaves Primária e Candidata.
  - Chave Composta - Quando a chave é formada por mais de um atributo ao mesmo tempo.

Modelagem de Dados: Tipos de Atributos

- Monovalorado: Valor único. Ex: número da rua.
- Composto: Pode ser referenciado de duas formas. Ex.: Endereço, composto por rua, numero, cidade, CEP.
- Multivalorado: Um atributo que possui mais de um valor para a entidade. Ex: o atributo telefone ou e-mail podem possuir mais de um valor atribuído.
- Determinante: Identificador único.

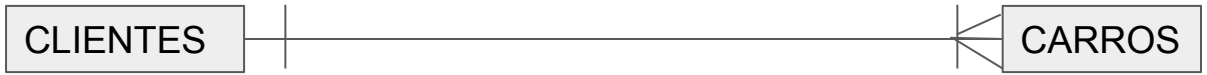
Modelagem de Dados: Relacionamento

- Relacionamento - Associação entre entidades ou com a mesma.
- Relacionamento 1 para 1 (1:1)



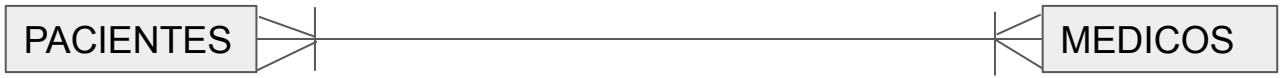
Modelagem de Dados: Relacionamento

- Relacionamento 1 para muitos (1:N)



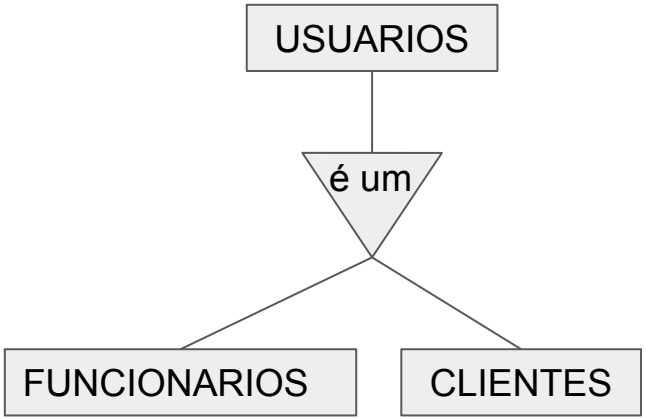
Modelagem de Dados: Relacionamento

- Relacionamento muitos para muitos (N:M)



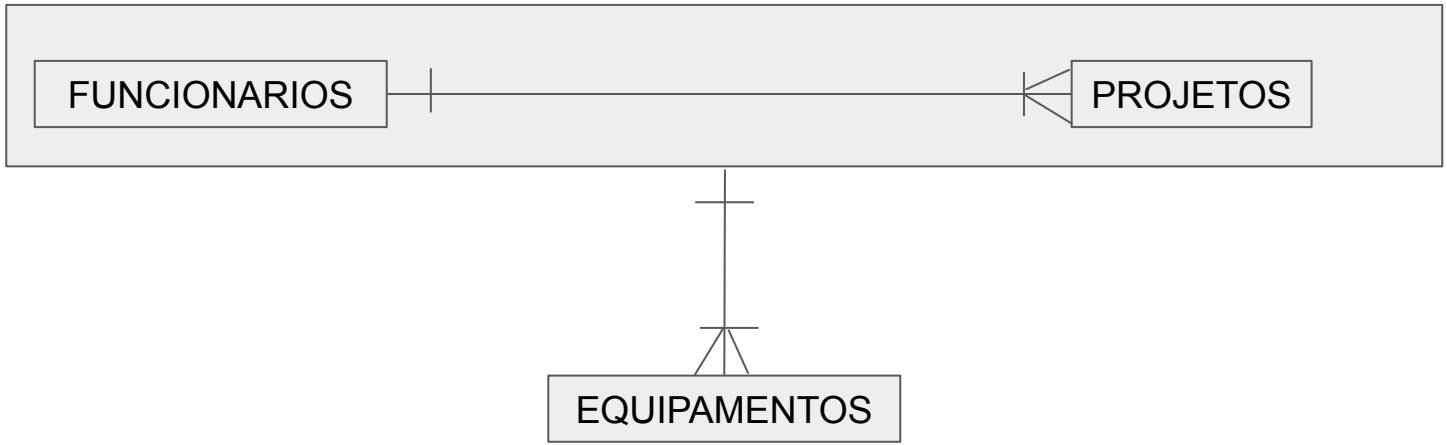
Modelagem de Dados: Relacionamento

- Generalização/Especialização: É um



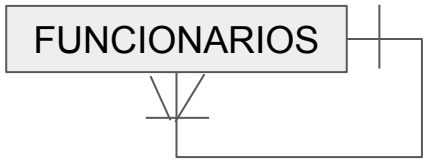
Modelagem de Dados: Relacionamento

- Agregação



Modelagem de Dados: Relacionamento

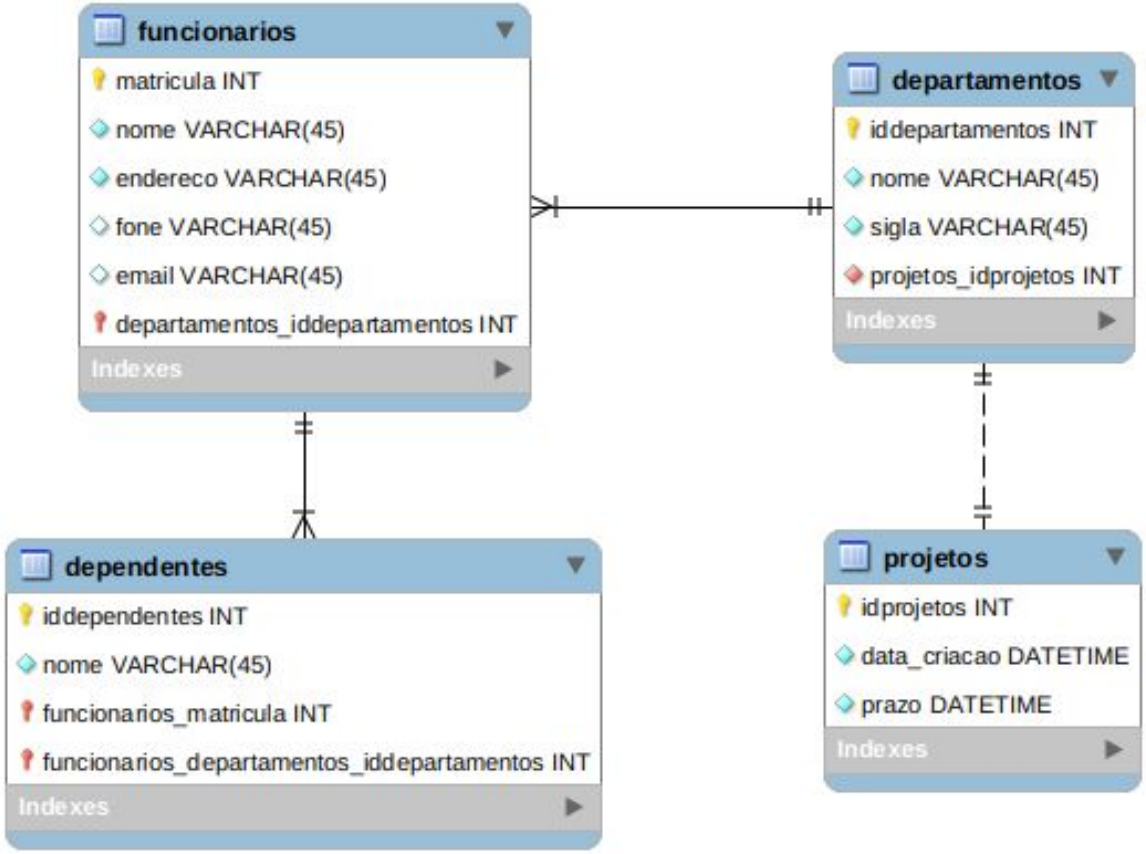
- Auto Relacionamento



Exemplo

Uma empresa tem vários funcionários que são lotados em departamentos. Um funcionário está em apenas um departamento de cada vez. Ao funcionário é dada a opção de ter dependentes. Cada departamento é responsável por um projeto da empresa. Os dados mais importantes de funcionário são: matrícula, nome, CPF, endereço e telefone. Os dependentes devem ter nome e código. Cada departamento tem código, sigla e nome completo. Os projetos possuem código, data de criação e prazo de execução. Pede-se que seja feito o diagrama de entidades-relacionamentos deste caso.

Exemplo ER no MySQL Workbench



SGBD

- **Sistema Gerenciador de Bancos de Dados (SGBD)**: consiste em um conjunto de programas para acessar e gerenciar os bancos de dados.

Linguagens de Banco de Dados

- **DDL – Data Definition Language** – Linguagem de Definição de Dados – Define estruturas. Comandos de Criação.
- **DML – Data Manipulation Language** – Linguagem de Manipulação de Dados – Linguagem que permite inserções, buscas, deleções e demais operações sobre os dados armazenados.
- **SQL - Structured Query Language -** Linguagem de Consulta Estruturada - Linguagem que une DDL e DML em uma só, dentre outras funcionalidades.

MySQL: Instalação

- $ sudo apt install mysql-server
- $ sudo systemctl start mysql.service
- $ sudo mysql_secure_installation
- $ mysql -u root -p
- $ systemctl status mysql.service
- $ mysql --version
- $ sudo snap install mysql-workbench-community

MySQL: Configuração e Criação do Banco

- Neste ponto, é importante que o banco de dados e o usuário MySQL já estejam criados no SGBD local.
- Caso não estejam, você pode usar comandos abaixo para criá-los, mantendo o usuário e senha conforme informado.
- Comandos DDL:
    - CREATE (criação de estrutura), ALTER (alterar estrutura) e DROP (permite remover ou excluir uma estrutura).

MySQL: Configuração e Criação do Banco do Exemplo

- N ySQL já
- C á-los, m
- C ra) e

```
may@may-pc:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 23
Server version: 8.0.33-0ubuntu0.22.10.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database myapp;
Query OK, 1 row affected (0,01 sec)

mysql> create user 'myapp'@'localhost' identified by 'myAPP@2023'
    -> ;
Query OK, 0 rows affected (0,04 sec)

mysql> grant all privileges on myapp.* to 'myapp'@'localhost';
Query OK, 0 rows affected (0,03 sec)
```

SQL: Comandos DDL para o exemplo: Projetos

```
use mydb;

CREATE TABLE IF NOT EXISTS
`projetos` (
  `idprojetos` INT NOT NULL,
  `data_criacao` DATETIME NOT NULL,
  `prazo` DATETIME NOT NULL,
  PRIMARY KEY (`idprojetos`));
```

SQL: Comandos DDL para o exemplo: Departamentos

```
CREATE TABLE IF NOT EXISTS `departamentos` (
  `iddepartamentos` INT NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  `sigla` VARCHAR(45) NOT NULL,
  `projetos_idprojetos` INT NOT NULL,
  PRIMARY KEY (`iddepartamentos`),
  INDEX `fk_departamentos_projetos1_idx`
(`projetos_idprojetos` ASC) VISIBLE,
  CONSTRAINT `fk_departamentos_projetos1`
    FOREIGN KEY (`projetos_idprojetos`)
    REFERENCES `projetos` (`idprojetos`));
```

SQL: Comandos DDL para o exemplo: Funcionários

```
CREATE TABLE IF NOT EXISTS `funcionarios` (
  `matricula` INT NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  `endereco` VARCHAR(45) NOT NULL,
  `fone` VARCHAR(45) NULL,
  `email` VARCHAR(45) NULL,
  `departamentos_iddepartamentos` INT NOT NULL,
  PRIMARY KEY (`matricula`, `departamentos_iddepartamentos`),
  UNIQUE INDEX `email_UNIQUE` (`email` ASC) VISIBLE,
  INDEX `fk_funcionarios_departamentos_idx`
(`departamentos_iddepartamentos` ASC) VISIBLE,
  CONSTRAINT `fk_funcionarios_departamentos`
    FOREIGN KEY (`departamentos_iddepartamentos`)
    REFERENCES `departamentos` (`iddepartamentos`));
```

SQL: Comandos DDL para o exemplo: Dependentes

```
CREATE TABLE IF NOT EXISTS `dependentes` (
  `iddependentes` INT NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  `funcionarios_matricula` INT NOT NULL,
  `funcionarios_departamentos_iddepartamentos` INT NOT NULL,
  PRIMARY KEY (`iddependentes`, `funcionarios_matricula`,
`funcionarios_departamentos_iddepartamentos`),
  INDEX `fk_dependentes_funcionarios1_idx`
(`funcionarios_matricula` ASC,
`funcionarios_departamentos_iddepartamentos` ASC) VISIBLE,
  CONSTRAINT `fk_dependentes_funcionarios1`
    FOREIGN KEY (`funcionarios_matricula` ,
`funcionarios_departamentos_iddepartamentos`)
    REFERENCES `funcionarios` (`matricula` ,
`departamentos_iddepartamentos`));
```

SQL: Mostrar todas as tabelas de uma base de dados do Exemplo

SQL: Mostrar informações de uma única tabela do exemplo

```
mysql> describe departamentos;
+---------------------+-------------+------+-----+---------+-------+
| Field               | Type        | Null | Key | Default | Extra |
+---------------------+-------------+------+-----+---------+-------+
| iddepartamentos     | int         | NO   | PRI | NULL    |       |
| nome                | varchar(45) | NO   |     | NULL    |       |
| sigla               | varchar(45) | NO   |     | NULL    |       |
| projetos_idprojetos | int         | NO   | MUL | NULL    |       |
+---------------------+-------------+------+-----+---------+-------+
4 rows in set (0,01 sec)

mysql>
```

SQL: Comandos DML

- Linguagem que permite manipular os dados que estão nas estruturas (Tabelas).
- Principais comandos: SELECT, INSERT, UPDATE, DELETE
  - Alguns autores definem que o comando SELECT faz parte de uma subdivisão chamada DQL (Data Query Language – Linguagem de Consulta de Dados).

SQL: Comandos DML para o exemplo

- INSERT

```
insert into projetos(idprojetos,data_criacao,prazo)
values (1, '2023-03-01 10:00:00', '2023-03-01
10:00:00');
insert into departamentos(iddepartamentos, nome, sigla,
projetos_idprojet
os) values (1,'Alpha', 'A1', 1);
insert into departamentos(iddepartamentos, nome, sigla,
projetos_idprojetos) values (2,'Beta', 'B2', 1);
insert into departamentos(iddepartamentos, nome, sigla,
projetos_idprojetos) values (3,'Omega', 'O3', 1);
```

SQL: Comandos DML para o exemplo

- SELECT

```
mysql> select * from projetos;
+------------+---------------------+---------------------+
| idprojetos | data_criacao        | prazo               |
+------------+---------------------+---------------------+
|          1 | 2023-03-01 10:00:00 | 2023-03-01 10:00:00 |
+------------+---------------------+---------------------+
1 row in set (0,00 s
```

```
mysql> select * from departamentos;
+-----------------+-------+-------+----------------------+
| iddepartamentos | nome  | sigla | projetos_idprojetos  |
+-----------------+-------+-------+----------------------+
|               1 | Alpha | A1    |                    1 |
|               2 | Beta  | B2    |                    1 |
|               3 | Omega | O3    |                    1 |
+-----------------+-------+-------+----------------------+
3 rows in set (0,00 sec)
```

SQL: Comandos DML para o exemplo

- UPDATE

```
update departamentos set nome='Departamento
Pessoal' where iddepartamentos=1;
```

```
mysql> select * from departamentos;
+-----------------+-----------------------+-------+----------------------+
| iddepartamentos | nome                  | sigla | projetos_idprojetos  |
+-----------------+-----------------------+-------+----------------------+
|               1 | Departamento Pessoal  | A1    |                    1 |
|               2 | Beta                  | B2    |                    1 |
|               3 | Omega                 | O3    |                    1 |
+-----------------+-----------------------+-------+----------------------+
3 rows in set (0,00 sec)
```

SQL: Comandos DML para o exemplo

- WHERE

```
mysql> select nome from departamentos where iddepartamentos=1;
+----------------------+
| nome                 |
+----------------------+
| Departamento Pessoal |
+----------------------+
1 row in set (0,00 sec)
```

```
mysql> select nome from departamentos where nome='beta';
+------+
| nome |
+------+
| Beta |
+------+
1 row in set (0,00 sec)
```

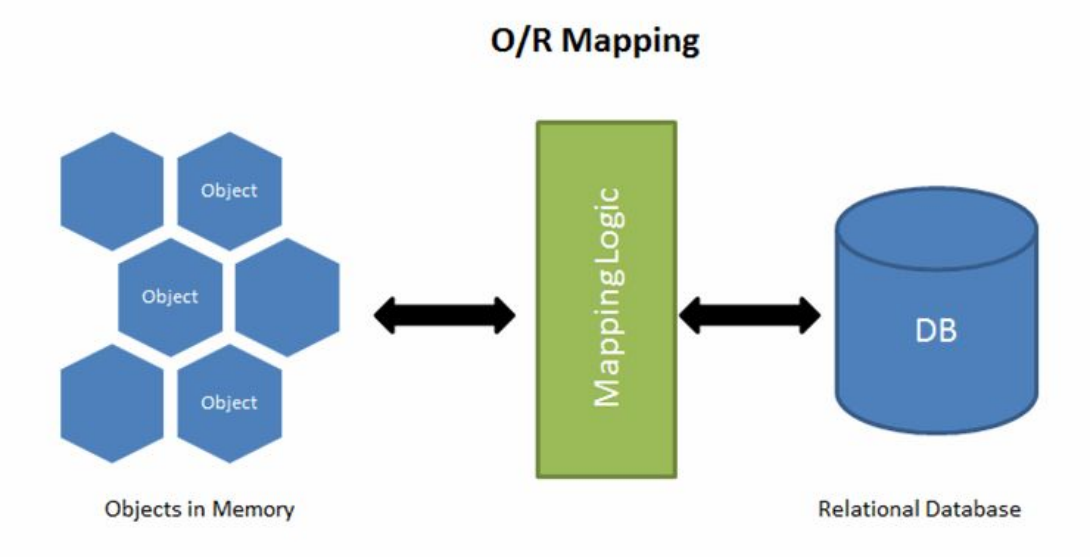
SQL: Comandos DML para o exemplo

- LIKE

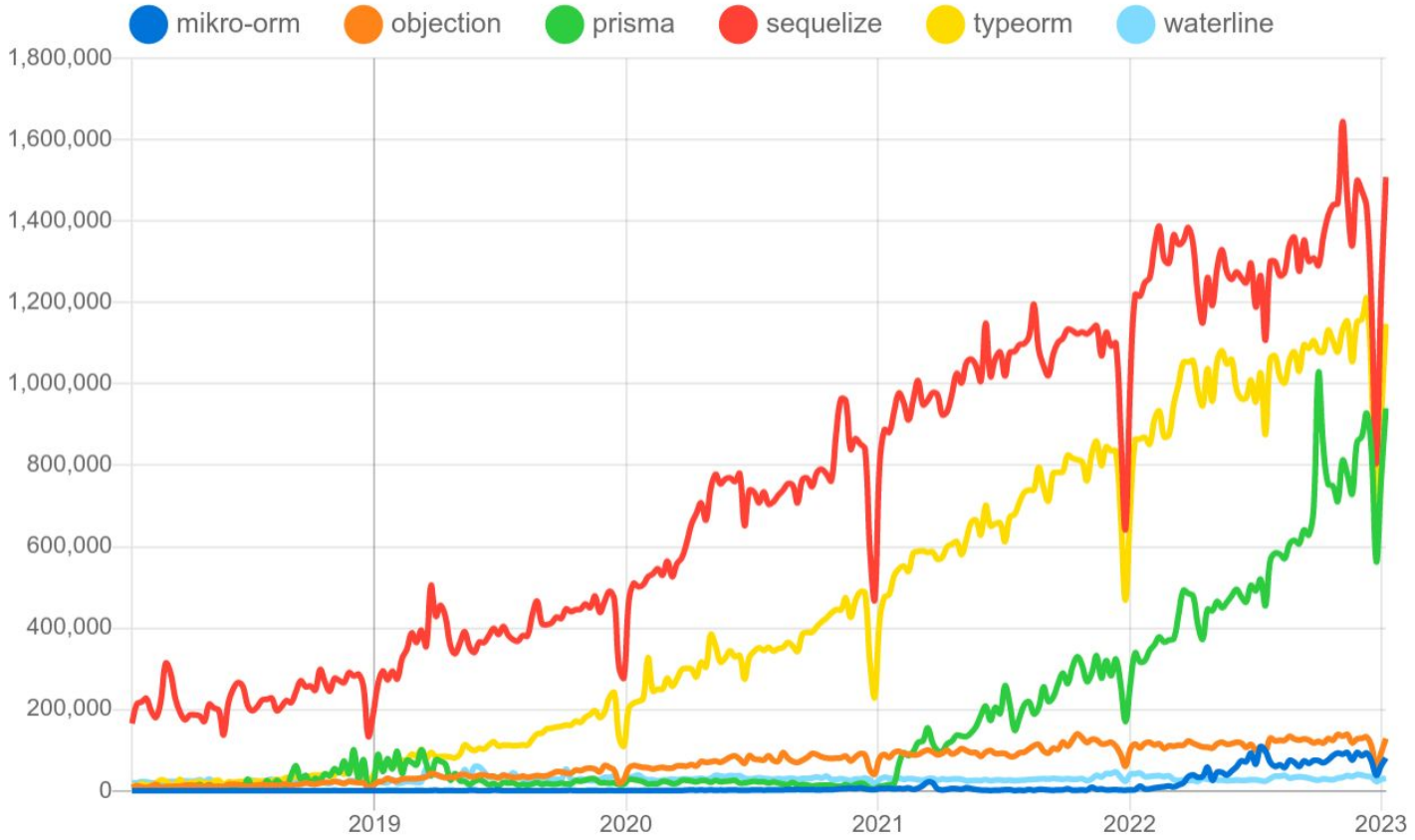
```
mysql> select nome from departamentos where nome like '%ta%';
+----------------------+
| nome                 |
+----------------------+
| Departamento Pessoal |
| Beta                 |
+----------------------+
2 rows in set (0,00 sec)
```

Object Relational Mapper - ORM

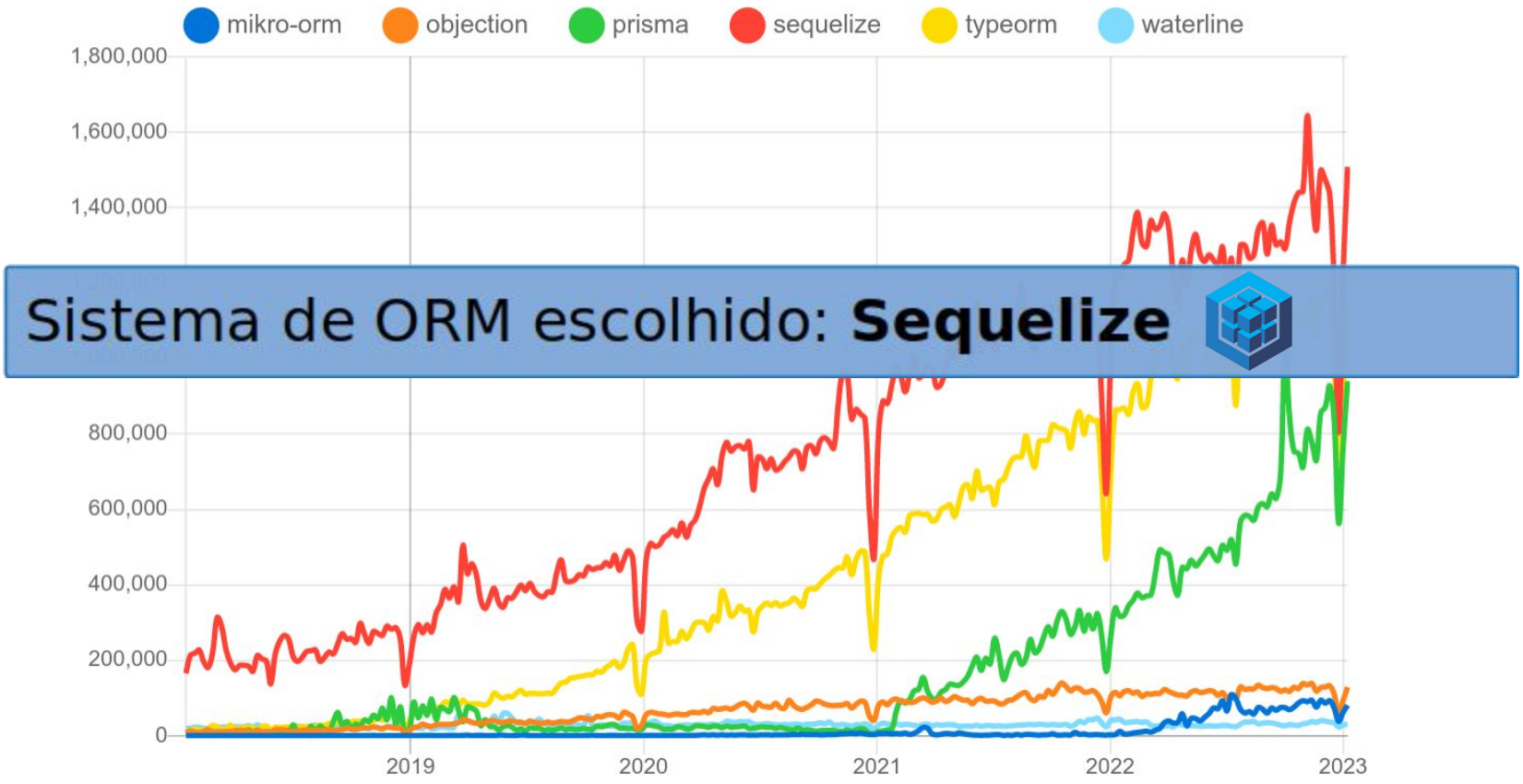
- ORM é uma técnica que permite consultar e manipular dados de um database usando o paradigma de orientação a objetos
- Desta forma, o acesso aos dados não é feito através da linguagem SQL, e sim através de objetos



Escolhendo o ORM



Escolhendo o ORM



Sequelize

- Use os comandos abaixo para instalar o Sequelize com suporte ao banco de dados MySQL

```
$ npm install --save sequelize mysql2
```

- Decorators para facilitar o uso do sequelize: Sequelize-typescript

```
npm install --save-dev @types/node @types/validator
npm install sequelize reflect-metadata sequelize-typescript
```

Sequelize

- Use os comandos abaixo para instalar o Sequelize com suporte ao banco de dados MySQL

```
$ npm install --save se                    sql2
```

tsconfig.ts

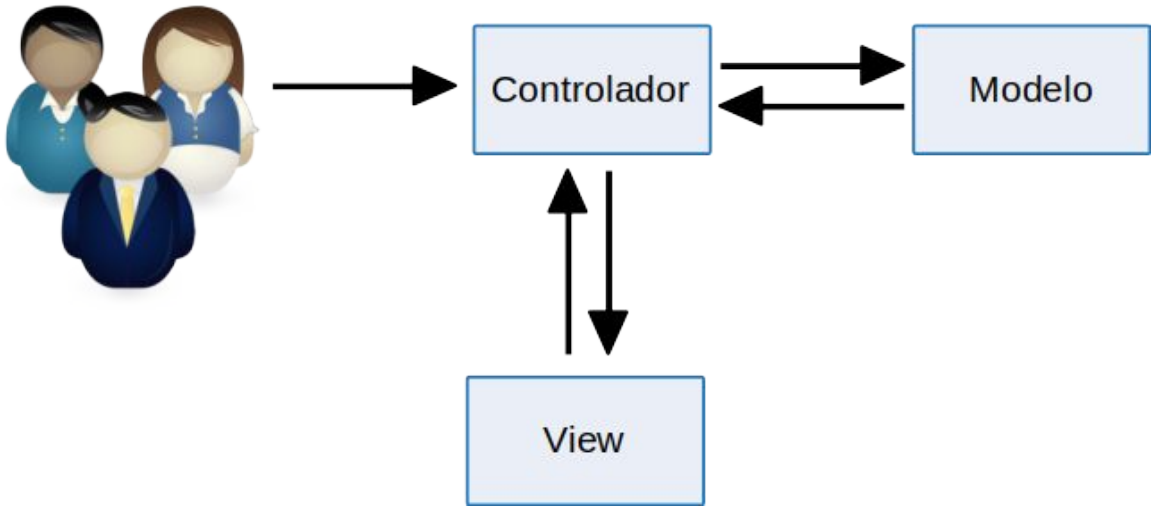
- De   uelize-typescript

```
"target": "es6",
"experimentalDecorators": true,
"emitDecoratorMetadata": true
```

```
npm ir                                                  es/validator
npm ir                                                  sequelize-typescript
```

Models

- Os modelos são responsáveis pela leitura e escrita dos dados presentes nas tabelas.
- Cada modelo representa uma tabela do banco de dados.
- Modelos representam a camada M do padrão MVC.



Models: Definindo um modelo

```
import { Table, Column, Model, HasMany }
from 'sequelize-typescript';

@Table
class Person extends Model {
  @Column
  name: string;

  @Column
  birthday: Date;

 }
```

Models: Definindo um modelo

```
@Table({
 paranoid: true,
 timestamps: true,
 indexes: [{
   unique: true,
   fields: ['description',
'initialDate', 'finalDate'],
 }],
});
```

```
@IsUUID('all')
 @PrimaryKey
 @Column({
   type: DataType.UUID,
   defaultValue:
DataType.UUIDV1,
 })
 id: string;
```

Models: Definindo um modelo

```
import {DataType} from
'sequelize-typescript';


@AllowNull(false)
@Column({
 type: DataType.TEXT,
})
description: string;
```

```
@AllowNull(false)
@IsDate
@Column({
 type: DataType.DATEONLY,
})
initialDate: number;
```

Exercício 1

1. Gere um sql do banco criado em sala de aula, entregar via colab, ainda hoje, dia 06 de junho de 2023.
    - mysqldump -u root -p mydb > mydbsql.sql

Exercício 2

1. Defina o que é Banco de Dados
2. Defina SGBD e dê exemplos
3. Defina DDL e DML.

Exercício 2

5. Faça o Diagrama Entidade-Relacionamentos do estudo de caso a seguir: Uma empresa pretende vender produtos pela internet através de uma loja virtual. Portanto, está precisando de um sistema para gerenciar os produtos, as vendas e seus futuros clientes. Cada produto tem uma categoria, que possui: o id e a descrição. Deve-se fazer um cadastro de clientes contendo: id, nome, endereço, email e datas de criação e alteração das suas informações. Os produtos devem ser gerenciados pelo sistema com os dados: id, descrição, preço, quantidade, categoria e datas de criação e alteração das suas informações. As vendas são geradas entre clientes e produtos. Onde um cliente compra vários produtos e um tipo de produto pode ser comprado por inúmeros clientes.

Exercício 2

6. Use os comandos DDL disponíveis no SQL para criação das tabelas da questão 5.

7. Use os comandos DML para “popular” e “explorar” as informações da base resultante da questão 6.

Obs: O exercício 2 pode ser entregue até dia 10 de junho à noite, em doc via colab.