Avaliação - Módulo Bases de Dados - Upskill JavaDotnet-v2 - 2024

Tiago Barracha

Enunciado

a) (grupo) 20%

Deseja-se criar uma base de dados para gerir as atividades de manutenção de uma empresa.
A empresa está organizada em secções, que são identificadas por um código e um nome.
Há uma seção chamada "Manutenção" responsável por reparos e reformas em outras seções que está organizada por especialidades.
Cada funcionário da empresa tem um número identificativo e é caracterizado na Base de Dados por um BI, um nome e pelas secções da empresa onde trabalha (uma ou mais). Os funcionários que trabalham na secção de "Manutenção" só trabalham nesta secção e são caracterizados pela especialidade correspondente (eletricista, serralheiro, etc).
Os funcionários que trabalham nas outras secções da empresa enviam pedidos de manutenção para a secção de "Manutenção". Cada pedido tem um número sequencial identificativo e é caracterizado pela data e hora de envio, a secção que efetuou o pedido e o funcionário que o solicitou.
A secção de "Manutenção" organiza os pedidos que recebe por especialidade e elabora atividades, onde cada atividade pode satisfazer um ou mais pedidos, atribuindo-lhe um número e, nomeando um funcionário responsável para a sua execução (obrigatoriamente da especialidade da atividade) e estimando uma data de início e uma data fim para a sua execução. Deverá ainda definir o conjunto de funcionários a alocar a esta execução. Para análise de desvios deverão ser registadas as datas efetivas de realização de início e fim de execução da atividade.
Para a realização de uma atividade poderão ser utilizados vários materiais sendo importante conhecer a quantidade gasta na realização de cada atividade.
Cada material é identificado por um código, designação, unidade de medida e pertence a uma categoria de material (madeira, cabo elétrico, tinta, etc), Um material pode ser fornecido por um ou vários fornecedores. Para cada compra efetuada deverá registar-se a atividade a que se destina, o fornecedor, a quantidade adquirida, a data de aquisição e o preço. Cada fornecedor identifica-se pelo seu número de contribuinte, nome da empresa, morada e uma lista de telefones.

Com base nos requisitos descritos, esboce um modelo de dados relacional normalizado (modelo lógico), representando as principais entidades e os atributos mais significativos.
Deve ser claro no modelo quais são os atributos que constituem as chaves primárias e as chaves estrangeiras assim como as relações entre as entidades, e respectivas cardinalidades.

b) (grupo) 20%

Resolver a ficha de Exercícios SQL 4

c) (grupo) 20%

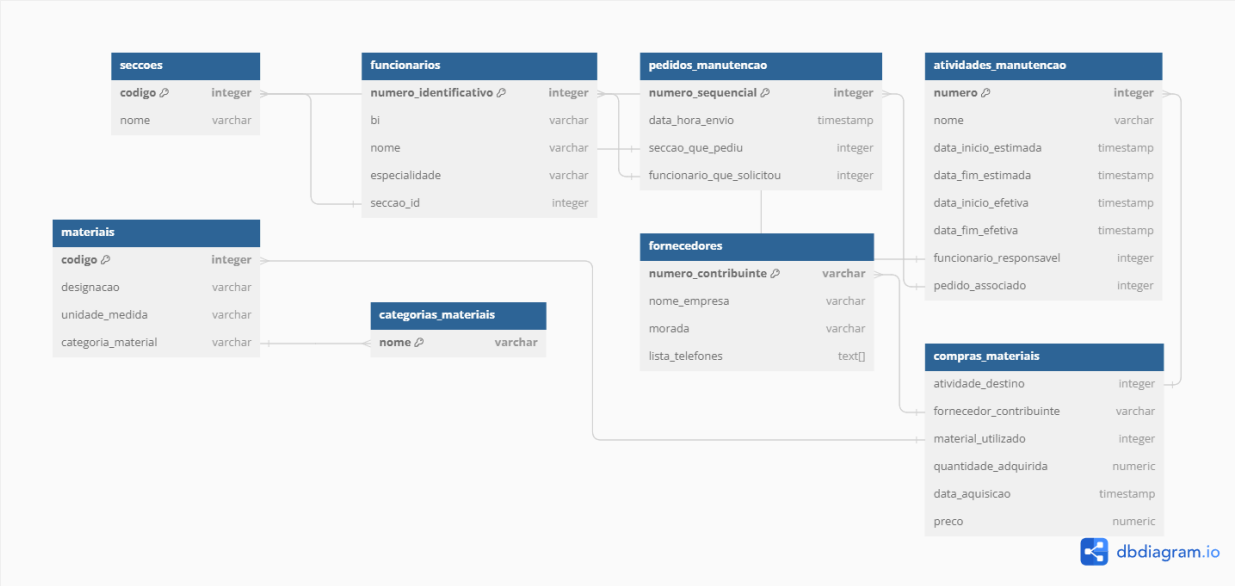
Projeto SNS - Desenvolver script do modelo físico

d) (individual) 40%

Projeto SNS
Implementar um caso de uso usando persistência em base de dados. (desconsiderar os casos de uso que já são dados como exemplo na framework disponibilizada)

Resolução

a) base de dados para gerir as atividades de manutenção de uma empresa.



Link:
https://dbdiagram.io/d/65b0bedeac844320ae992a56

Code:

```
Table seccoes {
  codigo integer [primary key]
  nome varchar
}

Table funcionarios {
  numero_identificativo integer [primary key]
  bi varchar
  nome varchar
  especialidade varchar
  seccao_id integer
}
```

```
Table pedidos_manutencao {
  numero_sequencial integer [primary key]
  data_hora_envio timestamp
  seccao_que_pediu integer
  funcionario_que_solicitou integer
}

Table atividades_manutencao {
  numero integer [primary key]
  nome varchar
  data_inicio_estimada timestamp
  data_fim_estimada timestamp
  data_inicio_efetiva timestamp
  data_fim_efetiva timestamp
  funcionario_responsavel integer
  pedido_associado integer
}

Table materiais {
  codigo integer [primary key]
  designacao varchar
  unidade_medida varchar
  categoria_material varchar
}

Table categorias_materiais {
  nome varchar [primary key]
}

Table fornecedores {
  numero_contribuinte varchar [primary key]
  nome_empresa varchar
  morada varchar
  lista_telefones text[]
}
```

```
Table compras_materiais {
  atividade_destino integer
  fornecedor_contribuinte varchar
  material_utilizado integer
  quantidade_adquirida numeric
  data_aquisicao timestamp
  preco numeric
}

Ref: seccoes.codigo > funcionarios.seccao_id

Ref: seccoes.codigo > pedidos_manutencao.seccao_que_pediu

Ref: funcionarios.numero_identificativo >
pedidos_manutencao.funcionario_que_solicitou

Ref: funcionarios.numero_identificativo >
atividades_manutencao.funcionario_responsavel

Ref: pedidos_manutencao.numero_sequencial >
atividades_manutencao.pedido_associado

Ref: categorias_materiais.nome > materiais.categoria_material

Ref: fornecedores.numero_contribuinte >
compras_materiais.fornecedor_contribuinte

Ref: atividades_manutencao.numero > compras_materiais.atividade_destino

Ref: materiais.codigo > compras_materiais.material_utilizado
```

b) Resolver a ficha de Exercícios SQL 4

```
-- ** eliminar tabelas se existentes **
-- CASCADE CONSTRAINTS para eliminar as restri??es de integridade das
chaves primárias e chaves únicas
-- PURGE elimina a tabela da base de dados e da "reciclagem"

DROP TABLE empregado                CASCADE CONSTRAINTS PURGE;
DROP TABLE empregadoEfetivo         CASCADE CONSTRAINTS PURGE;
DROP TABLE empregadoTemporario      CASCADE CONSTRAINTS PURGE;
DROP TABLE falta                    CASCADE CONSTRAINTS PURGE;
DROP TABLE ferias                   CASCADE CONSTRAINTS PURGE;
DROP TABLE avaliacao                CASCADE CONSTRAINTS PURGE;
DROP TABLE avaliacaoTemporario      CASCADE CONSTRAINTS PURGE;
DROP TABLE avaliacaoEfetivo         CASCADE CONSTRAINTS PURGE;
DROP TABLE departamento             CASCADE CONSTRAINTS PURGE;
DROP TABLE empregadoDepartamento    CASCADE CONSTRAINTS PURGE;

-- ** cria��o de tabelas **

CREATE TABLE empregado (
    idEmpregado             INTEGER GENERATED AS IDENTITY   CONSTRAINT
pkEmpregadoIdEmpregado          PRIMARY KEY,
    nome                    VARCHAR(40)                     CONSTRAINT
nnEmpregadoNome                 NOT NULL,
    dataNascimento          DATE                            CONSTRAINT
nnEmpregadoDataNascimento       NOT NULL,
    nrIdentificacaoCivil    INTEGER                         CONSTRAINT
ckEmpregadoNrIdentificacaoCivil  CHECK(REGEXP_LIKE(nrIdentificacaoCivil,
'^\d{6,}$'))
                                                            CONSTRAINT
ukEmpregadoNrIdentificacaoCivil  UNIQUE,
    nif                     INTEGER                         CONSTRAINT
nnEmpregadoNif                  NOT NULL
                                                            CONSTRAINT
ckEmpregadoNif                  CHECK(REGEXP_LIKE(nif, '^\d{9}$'))
                                                            CONSTRAINT
ukEmpregadoNif                  UNIQUE
);
```

```
CREATE TABLE empregadoEfetivo (
    idEmpregado         INTEGER         CONSTRAINT
pkEmpregadoEfetivoIdEmpregado   PRIMARY KEY,
    salarioMensalBase   NUMERIC(10,2)   CONSTRAINT
nnEmpregadoEfetivoSalarioMensalBase   NOT NULL
                                        CONSTRAINT
ckEmpregadoEfetivoSalarioMensalBase   CHECK (salarioMensalBase >= 500)
);

CREATE TABLE empregadoTemporario (
    idEmpregado INTEGER         CONSTRAINT
pkEmpregadoTemporarioIdEmpregado PRIMARY KEY,
    salarioHora NUMERIC(4,2)    CONSTRAINT
nnEmpregadoTemporarioSalarioHora NOT NULL
);

CREATE TABLE falta (
    idEmpregado     INTEGER,
    dataInicio      DATE,
    dataFim         DATE,
    justificacao    VARCHAR(50),

    CONSTRAINT pkFaltaIdEmpregadoDataInicio PRIMARY KEY (idEmpregado,
dataInicio),
    CONSTRAINT ckFaltaDataInicioDataFim     CHECK (dataFim>=dataInicio)
);

CREATE TABLE ferias (
    idEmpregado     INTEGER,
    dataInicio      DATE,
    dataFim         DATE        CONSTRAINT nnFeriasDataFim   NOT NULL,

    CONSTRAINT pkFeriasIdEmpregadoDataInicio    PRIMARY KEY (idEmpregado,
dataInicio),
    CONSTRAINT ckFeriasDataInicioDataFim        CHECK(dataFim>=dataInicio)
);

CREATE TABLE avaliacao (
    idAvaliacao VARCHAR(3)  CONSTRAINT pkAvaliacaoIdAvaliacao   PRIMARY
KEY,
```

```
    descricao   VARCHAR(15) CONSTRAINT nnAvaliacaoDescricao     NOT NULL
);

CREATE TABLE avaliacaoTemporario (
    idDepartamento  VARCHAR(5),
    idEmpregado     INTEGER,
    dataInicio      DATE,
    idAvaliacao     VARCHAR(3),

    CONSTRAINT pkAvaliacaoTemporarioIdEmpregadoDataInicio   PRIMARY KEY
(idDepartamento, idEmpregado, dataInicio)
);

CREATE TABLE avaliacaoEfetivo (
    idEmpregado INTEGER,
    ano         INTEGER     CONSTRAINT ckAvaliacaoEfetivoAno
CHECK(ano>=2015)    NOT NULL,
    idAvaliacao VARCHAR(3)  CONSTRAINT nnAvaliacaoEfetivoIdAvaliacao
NOT NULL,

    CONSTRAINT pkAvaliacaoEfetivoIdEmpregadoAno PRIMARY KEY (idEmpregado,
ano)
);

CREATE TABLE departamento (
    idDepartamento          VARCHAR(5)  CONSTRAINT
pkDepartamentoIdDepartamento PRIMARY KEY,
    idDepartamentoSuperior  VARCHAR(5),
    designacao              VARCHAR(50) CONSTRAINT
nnDepartamentoIdDesignacao   NOT NULL
);


CREATE TABLE empregadoDepartamento (
    idDepartamento  VARCHAR(5),
    idEmpregado     INTEGER,
    dataInicio      DATE,
    dataFim         DATE,
```

```
    CONSTRAINT pkEmpregadoDepartamentoIdDepartamentoIdEmpregadoDataInicio
PRIMARY KEY (idDepartamento, idEmpregado, dataInicio),
    CONSTRAINT ckEmpregadoDepartamentoDataInicioDataFim
CHECK(dataFim>=dataInicio)
);


-- ** chaves estrangeiras **

ALTER TABLE empregadoEfetivo    ADD CONSTRAINT
fkEmpregadoEfetivoIdEmpregado                          FOREIGN KEY
(idEmpregado)                                REFERENCES empregado
(idEmpregado);

ALTER TABLE empregadoTemporario ADD CONSTRAINT
fkEmpregadoTemporarioIdEmpregado                       FOREIGN KEY
(idEmpregado)                                REFERENCES empregado
(idEmpregado);

ALTER TABLE falta               ADD CONSTRAINT fkFaltaIdEmpregado
FOREIGN KEY (idEmpregado)                              REFERENCES
empregado (idEmpregado);

ALTER TABLE ferias              ADD CONSTRAINT fkFeriasIdEmpregado
FOREIGN KEY (idEmpregado)                              REFERENCES
empregadoEfetivo (idEmpregado);

ALTER TABLE avaliacaoTemporario ADD CONSTRAINT
fkAvaliacaoTemporarioIdEmpregado                       FOREIGN KEY
(idEmpregado)                                REFERENCES empregadoTemporario
(idEmpregado);
ALTER TABLE avaliacaoTemporario ADD CONSTRAINT
fkAvaliacaoTemporarioIdDepartamentoIdEmpregadoDataInicio FOREIGN KEY
(idDepartamento, idEmpregado, dataInicio)   REFERENCES
empregadoDepartamento (idDepartamento, idEmpregado, dataInicio);
ALTER TABLE avaliacaoTemporario ADD CONSTRAINT
fkAvaliacaoTemporarioIdAvaliacao                       FOREIGN KEY
(idAvaliacao)                                REFERENCES avaliacao
(idAvaliacao);
```

```
ALTER TABLE avaliacaoEfetivo    ADD CONSTRAINT
fkAvaliacaoEfetivoIdEmpregado                          FOREIGN KEY
(idEmpregado)                              REFERENCES empregadoEfetivo
(idEmpregado);
ALTER TABLE avaliacaoEfetivo    ADD CONSTRAINT
fkAvaliacaoEfetivoIdAvaliacao                          FOREIGN KEY
(idAvaliacao)                              REFERENCES avaliacao
(idAvaliacao);

ALTER TABLE departamento        ADD CONSTRAINT
fkDepartamentoidDepartamentoSuperior                   FOREIGN KEY
(idDepartamentoSuperior)                   REFERENCES departamento
(idDepartamento);

ALTER TABLE empregadoDepartamento ADD CONSTRAINT
fkEmpregadoDepartamentoIdDepartamento                FOREIGN KEY
(idDepartamento)                           REFERENCES departamento
(idDepartamento);
ALTER TABLE empregadoDepartamento ADD CONSTRAINT
fkEmpregadoDepartamentoIdEmpregado                   FOREIGN KEY
(idEmpregado)                              REFERENCES empregado
(idEmpregado);

-- ** guardar em DEFINITIVO as altera??es na base de dados, se a op??o
Autocommit do SQL Developer n?o estiver ativada **
-- COMMIT;


-- ** tabela Empregado **

INSERT INTO empregado (nome, dataNascimento, nrIdentificacaoCivil, nif)
VALUES ('Belmiro Cunha', TO_DATE('1985-01-13','yyyy-mm-dd'), 1111111,
111111111);
INSERT INTO empregado (nome, dataNascimento, nrIdentificacaoCivil, nif)
VALUES ('Luisa Coelho',  TO_DATE('1980-05-03','yyyy-mm-dd'), 2222222,
222222222);
INSERT INTO empregado (nome, dataNascimento, nrIdentificacaoCivil, nif)
VALUES ('Jo?o Pereira',  TO_DATE('1970-09-05','yyyy-mm-dd'), 3333333,
333333333);
```

```
INSERT INTO empregado (nome, dataNascimento, nrIdentificacaoCivil, nif)
VALUES ('Carlos Silva',  TO_DATE('1983-02-10','yyyy-mm-dd'), 4444444,
444444444);
INSERT INTO empregado (nome, dataNascimento, nrIdentificacaoCivil, nif)
VALUES ('Anibal Dias',   TO_DATE('1982-10-12','yyyy-mm-dd'), 5555555,
555555555);
INSERT INTO empregado (nome, dataNascimento, nrIdentificacaoCivil, nif)
VALUES ('Anibal Dias',   TO_DATE('1983-04-22','yyyy-mm-dd'), 6666666,
666666666);
INSERT INTO empregado (nome, dataNascimento, nrIdentificacaoCivil, nif)
VALUES ('Joana Freitas', TO_DATE('1995-03-15','yyyy-mm-dd'), 7777777,
777777777);

-- ** tabela Falta **

INSERT INTO falta (idEmpregado, dataInicio, dataFim, justificacao) VALUES
(2, TO_DATE('2015-05-15','yyyy-mm-dd'),
TO_DATE('2015-05-20','yyyy-mm-dd'), 'gripe');
INSERT INTO falta (idEmpregado, dataInicio, dataFim, justificacao) VALUES
(3, TO_DATE('2018-11-05','yyyy-mm-dd'),
TO_DATE('2018-11-15','yyyy-mm-dd'), 'apoio a familiares');
INSERT INTO falta (idEmpregado, dataInicio, dataFim, justificacao) VALUES
(5, TO_DATE('2018-08-15','yyyy-mm-dd'),
TO_DATE('2018-08-31','yyyy-mm-dd'), 'fratura');
INSERT INTO falta (idEmpregado, dataInicio, dataFim, justificacao) VALUES
(6, TO_DATE('2020-09-25','yyyy-mm-dd'),
TO_DATE('2020-09-28','yyyy-mm-dd'), 'gripe');

-- ** tabela EmpregadoEfetivo **

INSERT INTO empregadoEfetivo (idEmpregado, salarioMensalBase) VALUES
(1,1500);
INSERT INTO empregadoEfetivo (idEmpregado, salarioMensalBase) VALUES
(2,700);
INSERT INTO empregadoEfetivo (idEmpregado, salarioMensalBase) VALUES
(3,1000);
INSERT INTO empregadoEfetivo (idEmpregado, salarioMensalBase) VALUES
(4,1000);

-- ** tabela EmpregadoTemporario **
```

```
INSERT INTO empregadoTemporario (idEmpregado, salarioHora) VALUES (5,7);
INSERT INTO empregadoTemporario (idEmpregado, salarioHora) VALUES (6,7);
INSERT INTO empregadoTemporario (idEmpregado, salarioHora) VALUES (7,5);

-- ** tabela Departamento **
INSERT INTO departamento (idDepartamento, idDepartamentoSuperior,
designacao) VALUES ('DIR', NULL, 'Dire??o');
INSERT INTO departamento (idDepartamento, idDepartamentoSuperior,
designacao) VALUES ('DRH', 'DIR', 'Departamento de Recursos Humanos');
INSERT INTO departamento (idDepartamento, idDepartamentoSuperior,
designacao) VALUES ('DSI', 'DIR', 'Departamento de Sistemas
Inform?ticos');
INSERT INTO departamento (idDepartamento, idDepartamentoSuperior,
designacao) VALUES ('DAU', 'DSI', 'Departamento de Apoio ao Utilizador');
INSERT INTO departamento (idDepartamento, idDepartamentoSuperior,
designacao) VALUES ('DMI', 'DSI', 'Departamento de Manuten??o
Inform?tica');

-- ** tabela EmpregadoDepartamento **

INSERT INTO empregadoDepartamento (idDepartamento, idEmpregado,
dataInicio, dataFim) VALUES ('DIR', 1, TO_DATE('2010-09-15','yyyy-mm-dd'),
NULL);
INSERT INTO empregadoDepartamento (idDepartamento, idEmpregado,
dataInicio, dataFim) VALUES ('DRH', 2, TO_DATE('2010-10-26','yyyy-mm-dd'),
NULL);
INSERT INTO empregadoDepartamento (idDepartamento, idEmpregado,
dataInicio, dataFim) VALUES ('DAU', 3, TO_DATE('2015-03-07','yyyy-mm-dd'),
TO_DATE('2019-09-09','yyyy-mm-dd'));
INSERT INTO empregadoDepartamento (idDepartamento, idEmpregado,
dataInicio, dataFim) VALUES ('DMI', 3, TO_DATE('2019-09-10','yyyy-mm-dd'),
NULL);
INSERT INTO empregadoDepartamento (idDepartamento, idEmpregado,
dataInicio, dataFim) VALUES ('DAU', 4, TO_DATE('2018-04-12','yyyy-mm-dd'),
NULL);
INSERT INTO empregadoDepartamento (idDepartamento, idEmpregado,
dataInicio, dataFim) VALUES ('DAU', 5, TO_DATE('2018-08-01','yyyy-mm-dd'),
TO_DATE('2018-08-31','yyyy-mm-dd'));
```

```
INSERT INTO empregadoDepartamento (idDepartamento, idEmpregado,
dataInicio, dataFim) VALUES ('DMI', 6, TO_DATE('2020-09-20','yyyy-mm-dd'),
TO_DATE('2020-09-30','yyyy-mm-dd'));
INSERT INTO empregadoDepartamento (idDepartamento, idEmpregado,
dataInicio, dataFim) VALUES ('DRH', 7, TO_DATE('2019-11-15','yyyy-mm-dd'),
TO_DATE('2019-12-31','yyyy-mm-dd'));
INSERT INTO empregadoDepartamento (idDepartamento, idEmpregado,
dataInicio, dataFim) VALUES ('DRH', 7, TO_DATE('2020-03-15','yyyy-mm-dd'),
TO_DATE('2020-04-15','yyyy-mm-dd'));

-- ** tabela Avaliacao **

INSERT INTO avaliacao (idAvaliacao, descricao) VALUES ('MB', 'MUITO BOM');
INSERT INTO avaliacao (idAvaliacao, descricao) VALUES ('B', 'BOM');
INSERT INTO avaliacao (idAvaliacao, descricao) VALUES ('S', 'SUFICIENTE');
INSERT INTO avaliacao (idAvaliacao, descricao) VALUES ('I',
'INSUFICIENTE');

-- ** tabela AvaliacaoTemporario **

INSERT INTO avaliacaoTemporario (idDepartamento, idEmpregado, dataInicio,
idAvaliacao) VALUES ('DAU', 5, TO_DATE('2018-08-01','yyyy-mm-dd'), 'S');
INSERT INTO avaliacaoTemporario (idDepartamento, idEmpregado, dataInicio,
idAvaliacao) VALUES ('DMI', 6, TO_DATE('2020-09-20','yyyy-mm-dd'), 'S');
INSERT INTO avaliacaoTemporario (idDepartamento, idEmpregado, dataInicio,
idAvaliacao) VALUES ('DRH', 7, TO_DATE('2019-11-15','yyyy-mm-dd'), 'MB');
INSERT INTO avaliacaoTemporario (idDepartamento, idEmpregado, dataInicio,
idAvaliacao) VALUES ('DRH', 7, TO_DATE('2020-03-15','yyyy-mm-dd'), 'MB');

-- ** tabela AvaliacaoEfetivo **

INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (1,
2015, 'MB');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (1,
2016, 'MB');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (1,
2017, 'MB');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (1,
2018, 'MB');
```

```
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (1,
2019, 'MB');

INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (2,
2015, 'MB');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (2,
2016, 'B');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (2,
2017, 'MB');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (2,
2018, 'MB');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (2,
2019, 'MB');

INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (3,
2016, 'S');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (3,
2017, 'S');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (3,
2018, 'S');
INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (3,
2019, 'S');

INSERT INTO avaliacaoEfetivo (idEmpregado, ano, idAvaliacao) VALUES (4,
2019, 'MB');

-- ** tabela Ferias **

INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (1,
TO_DATE('2011-08-15','yyyy-mm-dd'), TO_DATE('2011-08-31','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (1,
TO_DATE('2012-08-15','yyyy-mm-dd'), TO_DATE('2012-08-31','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (1,
TO_DATE('2013-08-01','yyyy-mm-dd'), TO_DATE('2013-08-11','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (1,
TO_DATE('2014-08-15','yyyy-mm-dd'), TO_DATE('2014-08-31','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (1,
TO_DATE('2015-08-15','yyyy-mm-dd'), TO_DATE('2015-08-31','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (1,
TO_DATE('2016-08-15','yyyy-mm-dd'), TO_DATE('2016-08-31','yyyy-mm-dd'));
```

```
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (1,
TO_DATE('2017-08-15','yyyy-mm-dd'), TO_DATE('2017-08-31','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (1,
TO_DATE('2019-08-01','yyyy-mm-dd'), TO_DATE('2019-08-31','yyyy-mm-dd'));

INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (2,
TO_DATE('2014-03-01','yyyy-mm-dd'), TO_DATE('2014-03-25','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (2,
TO_DATE('2015-07-01','yyyy-mm-dd'), TO_DATE('2015-07-20','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (2,
TO_DATE('2016-06-01','yyyy-mm-dd'), TO_DATE('2016-06-30','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (2,
TO_DATE('2017-07-01','yyyy-mm-dd'), TO_DATE('2017-07-27','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (2,
TO_DATE('2018-05-01','yyyy-mm-dd'), TO_DATE('2018-05-31','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (2,
TO_DATE('2019-12-10','yyyy-mm-dd'), TO_DATE('2019-12-31','yyyy-mm-dd'));

INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (3,
TO_DATE('2016-06-01','yyyy-mm-dd'), TO_DATE('2016-06-30','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (3,
TO_DATE('2017-07-01','yyyy-mm-dd'), TO_DATE('2017-07-11','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (3,
TO_DATE('2018-03-01','yyyy-mm-dd'), TO_DATE('2018-03-18','yyyy-mm-dd'));
INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (3,
TO_DATE('2019-05-01','yyyy-mm-dd'), TO_DATE('2019-05-15','yyyy-mm-dd'));

INSERT INTO ferias (idEmpregado, dataInicio, dataFim) VALUES (4,
TO_DATE('2019-05-01','yyyy-mm-dd'), TO_DATE('2019-05-31','yyyy-mm-dd'));

-- ** guardar em DEFINITIVO as altera??es na base de dados, se a op??o
Autocommit do SQL Developer n?o estiver ativada **
-- COMMIT;

--A. Cl♦usulas Join
--1) Obter o produto cartesiano (i.e. CROSS JOIN) entre as tabelas
Empregado e EmpregadoEfetivo
select e.idEmpregado, ee.idEmpregado as idEmpregadoEfetivo
from empregado e
cross join empregadoEfetivo ee;
```

```sql
--2)Obter toda a informa��o dos empregados efetivos, por ordem alfab�tica
dos nomes
select e.*, ee.salarioMensalBase from empregado e join empregadoEfetivo ee
on e.idEmpregado = ee.idEmpregado
order by e.nome;

--3)Obter as faltas dos empregados efetivos
select e.*, f.* from empregado e join falta f on e.idEmpregado =
f.idEmpregado
join empregadoEfetivo ee on e.idEmpregado = ee.idEmpregado;

--4)Obter as faltas dos empregados tempor�rios
select e.nome, e.nrIdentificacaoCivil, et.idEmpregado, f.dataInicio,
f.dataFim, f.justificacao
from empregado e
join empregadoTemporario et on e.idEmpregado = et.idEmpregado
join falta f on et.idEmpregado = f.idEmpregado;

--5)Obter as avalia��es dos empregados tempor�rios

select distinct e.idEmpregado, e.nome, e.nrIdentificacaoCivil,
ed.idDepartamento, ed.dataInicio, ed.dataFim, a.descricao
from empregado e
join avaliacaotemporario at on e.idEmpregado = at.idEmpregado
join avaliacao a on at.idAvaliacao = a.idAvaliacao
join empregadodepartamento ed on e.idEmpregado = ed.idEmpregado
where e.idEmpregado in (select idEmpregado from empregadotemporario)
order by e.idEmpregado, ed.dataInicio;

--6)Obter o nome e o n�mero de identifica��o civil de todos empregados que
nunca faltaram

select
  e.nome,
  e.nrIdentificacaoCivil
from
  empregado e
```

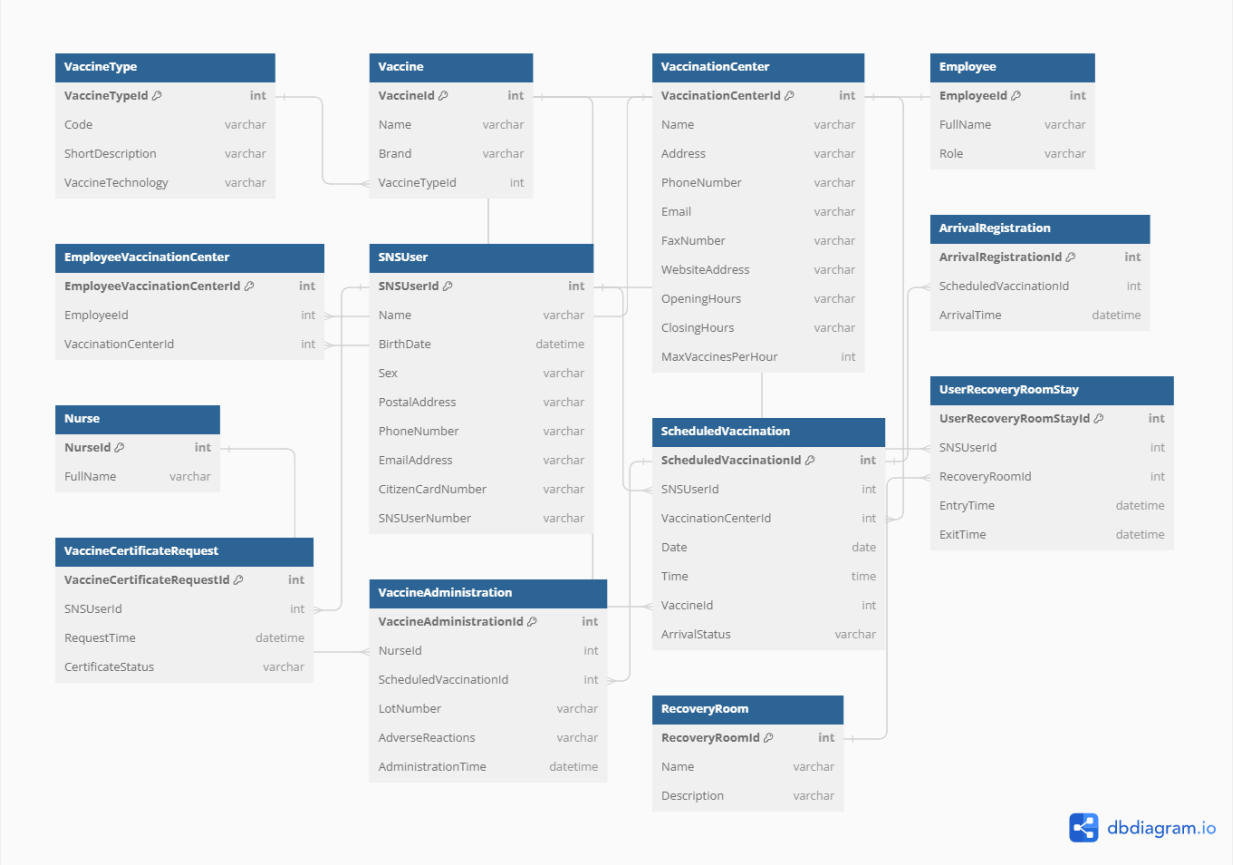
```sql
where
  not exists (
    select 1
    from falta f
    where f.idEmpregado = e.idEmpregado
  );


--7) Obter o identificador e a designa��o de todos os departamentos
juntamente com a designa��o do
--respetivo departamento do n�vel hier�rquico superior
select
  d1.idDepartamento,
  d1.designacao as designacaoDepartamento,
  d2.designacao as designacaoDepartamentoSuperior
from
  departamento d1
left join
  departamento d2 on d1.idDepartamentoSuperior = d2.idDepartamento
  order by
  d1.designacao desc;

--8)Obter pares de empregados distintos e com idades diferentes
select
  e1.idEmpregado as idEmpregado1,
  e1.nome as nome1,
  e2.idEmpregado as idEmpregado2,
  e2.nome as nome2
from
  empregado e1
join
  empregado e2 ON e1.idEmpregado < e2.idEmpregado
where
  abs(extract(year from current_date) - extract(year from
e1.dataNascimento)) !=
  abs(extract(year from current_date) - extract(year from
e2.dataNascimento));
```

```sql
--B)
--1) Obter o nome e o n◆mero de identifica◆◆o civil de todos os empregados
que tiveram pelo menos uma
--avalia◆◆o "MUITO BOM"

select
  e.nome,
  e.nrIdentificacaoCivil
from
  empregado e
where
  e.idEmpregado IN (
    select ae.idEmpregado
    from avaliacaoEfetivo ae
    join avaliacao a on ae.idAvaliacao = a.idAvaliacao
    where lower(a.descricao) = 'muito bom'
    union
    select at.idEmpregado
    from avaliacaoTemporario at
    join avaliacao a on at.idAvaliacao = a.idAvaliacao
    where lower(a.descricao) = 'muito bom'
  );

--2) Obter a descri◆◆o das avalia◆◆es que s◆o comuns a empregados efetivos
e tempor◆rios,
-- Seleciona as avalia◆◆es que s◆o comuns a empregados efetivos e
tempor◆rios
select distinct
  a.descricao
from
  avaliacao a
where
  exists (
    select 1
    from avaliacaoEfetivo ae
    where ae.idAvaliacao = a.idAvaliacao
  )
  and exists (
```

```
    select 1
    from avaliacaoTemporario at
    where at.idAvaliacao = a.idAvaliacao
  );



--3)Obter o nome e o n◆mero de identifica◆◆o civil dos empregados efetivos
que tiveram sempre classifica◆◆o
--◆MUITO BOM◆

select
  e.nome,
  e.nrIdentificacaoCivil
from
  empregado e
where
  e.idEmpregado in (
    select ae.idEmpregado
    from avaliacaoEfetivo ae
    where ae.idEmpregado not in (
      select ae_inner.idEmpregado
      from avaliacaoEfetivo ae_inner
      join avaliacao a on ae_inner.idAvaliacao = a.idAvaliacao
      where lower(a.descricao) <> 'muito bom'
  ));
--4)Obter o nome e o n◆mero de identifica◆◆o civil dos empregados efetivos
cujas f◆rias t◆m todas dura◆◆o
--superior a 15 dias
select
  e.nome,
  e.nrIdentificacaoCivil
from
  empregado e
where
  e.idEmpregado IN (
    select ef.idEmpregado
    from empregadoEfetivo ef

    where not exists (
```

```
    select 1
    from ferias f
    where f.idEmpregado = ef.idEmpregado
    and f.dataFim - f.dataInicio <= 15
  )

)order by e.nome;
```

c) Projeto SNS - Modelo Físico



Link:
https://dbdiagram.io/d/65b0bedeac844320ae992a56

Code:

```
Table VaccineType {
  VaccineTypeId int [pk]
  Code varchar [unique]
  ShortDescription varchar
  VaccineTechnology varchar
}

Table Vaccine {
  VaccineId int [pk]
  Name varchar
  Brand varchar
  VaccineTypeId int [ref: > VaccineType.VaccineTypeId]
}

Table VaccinationCenter {
  VaccinationCenterId int [pk]
  Name varchar
  Address varchar
  PhoneNumber varchar
  Email varchar
  FaxNumber varchar
  WebsiteAddress varchar
  OpeningHours varchar
  ClosingHours varchar
  MaxVaccinesPerHour int
}

Table Employee {
  EmployeeId int [pk]
  FullName varchar
  Role varchar
}

Table EmployeeVaccinationCenter {
  EmployeeVaccinationCenterId int [pk]
  EmployeeId int [ref: > Employee.EmployeeId]
  VaccinationCenterId int [ref: > VaccinationCenter.VaccinationCenterId]
}
```

```
Table SNSUser {
  SNSUserId int [pk]
  Name varchar
  BirthDate datetime
  Sex varchar
  PostalAddress varchar
  PhoneNumber varchar
  EmailAddress varchar
  CitizenCardNumber varchar [unique]
  SNSUserNumber varchar [unique]
}

Table ScheduledVaccination {
  ScheduledVaccinationId int [pk]
  SNSUserId int [ref: > SNSUser.SNSUserId]
  VaccinationCenterId int [ref: > VaccinationCenter.VaccinationCenterId]
  Date date
  Time time
  VaccineId int [ref: > Vaccine.VaccineId]
  ArrivalStatus varchar
}

Table ArrivalRegistration {
  ArrivalRegistrationId int [pk]
  ScheduledVaccinationId int [ref: >
ScheduledVaccination.ScheduledVaccinationId]
  ArrivalTime datetime
}

Table Nurse {
  NurseId int [pk]
  FullName varchar
}

Table VaccineAdministration {
  VaccineAdministrationId int [pk]
  NurseId int [ref: > Nurse.NurseId]
  ScheduledVaccinationId int [ref: >
ScheduledVaccination.ScheduledVaccinationId]
  LotNumber varchar
```

```
  AdverseReactions varchar
  AdministrationTime datetime
}

Table RecoveryRoom {
  RecoveryRoomId int [pk]
  Name varchar
  Description varchar
}

Table UserRecoveryRoomStay {
  UserRecoveryRoomStayId int [pk]
  SNSUserId int [ref: > SNSUser.SNSUserId]
  RecoveryRoomId int [ref: > RecoveryRoom.RecoveryRoomId]
  EntryTime datetime
  ExitTime datetime
}

Table VaccineCertificateRequest {
  VaccineCertificateRequestId int [pk]
  SNSUserId int [ref: > SNSUser.SNSUserId]
  RequestTime datetime
  CertificateStatus varchar
}
```

d) Projeto SNS - Implementar um caso de uso usando persistência em base de dados. (desconsiderar os casos de uso que já são dados como exemplo na framework disponibilizada)

Uso de caso escolhido:

- US014 - As administrator, I want to register an employee. [**Priority**: Low]
- US015 - As administrator, I want to get a list of employees with a given function/role. [**Priority**: Medium]

Uso de caso aplicado com projecto git pda-base - branch pds-base.

Base Project Link: https://github.com/nc18961a/pds-base/tree/pds-base-simpleDB

PDA project Link: https://github.com/tbarracha/upskill-pds-software-development-project-Tiago
DB Project Link: https://github.com/tbarracha/Upskill-pda-SNS-sqlDatabaseConnection

SQL code to test implementation:

```
CREATE TABLE Facility (
    id INT PRIMARY KEY,
    name VARCHAR2(255),
    address_streetname VARCHAR2(255),
    address_postalcode VARCHAR2(255),
    address_cityname VARCHAR2(255),
    phone VARCHAR2(255),
    email VARCHAR2(255),
    fax VARCHAR2(255),
    website VARCHAR2(255),
    maxvaccinesperhour NUMBER
);

CREATE TABLE Employee (
    id INT,
    email VARCHAR(255) PRIMARY KEY NOT NULL,
    name VARCHAR(255),
    position VARCHAR(255),
    phone VARCHAR(255)
);


DROP TABLE Facility;
DROP TABLE Employee;

SELECT * FROM Facility;
SELECT * FROM Employee;
```