Projeto de Bases de Dados

Parte 3

|  |  |  |  |
| --- | --- | --- | --- |
| Grupo 7 – Turno L03 – Segunda-feira 12:30 | | | |
| Docente: Duarte Alexandre Galvão | | | |
| Aluno | | Esforço (horas) | Contribuição  (%) |
| João Soares | 89475 | 21 | 33.3 |
| João Dinis | 89485 | 21 | 33.3 |
| Tiago Fonseca | 89542 | 21 | 33.3 |



Comandos de criação da base de dados

DROP TABLE IF EXISTS local\_publico CASCADE;

DROP TABLE IF EXISTS item CASCADE;

DROP TABLE IF EXISTS anomalia CASCADE;

DROP TABLE IF EXISTS anomalia\_traducao CASCADE;

DROP TABLE IF EXISTS duplicado CASCADE;

DROP TABLE IF EXISTS utilizador CASCADE;

DROP TABLE IF EXISTS utilizador\_qualificado CASCADE;

DROP TABLE IF EXISTS utilizador\_regular CASCADE;

DROP TABLE IF EXISTS incidencia CASCADE;

DROP TABLE IF EXISTS proposta\_de\_correcao CASCADE;

DROP TABLE IF EXISTS correcao CASCADE;

CREATE TABLE local\_publico (

latitude DECIMAL (9, 6) NOT NULL,

longitude DECIMAL(9, 6) NOT NULL,

nome VARCHAR(255) NOT NULL,

constraint pk\_local\_publico PRIMARY KEY (latitude, longitude)

);

CREATE TABLE item (

id SERIAL PRIMARY KEY,

descricao VARCHAR(255) NOT NULL,

localizacao VARCHAR(255) NOT NULL,

latitude DECIMAL (9, 6) NOT NULL,

longitude DECIMAL(9, 6) NOT NULL,

FOREIGN KEY (latitude, longitude) REFERENCES local\_publico(latitude, longitude) ON DELETE CASCADE

);

CREATE TABLE anomalia (

id SERIAL PRIMARY KEY,

zona VARCHAR(255) NOT NULL,

imagem VARCHAR(255) NOT NULL,

lingua VARCHAR(255) NOT NULL,

ts TIMESTAMP NOT NULL DEFAULT NOW(),

descricao VARCHAR(255) NOT NULL,

tem\_anomalia\_redacao BOOLEAN NOT NULL

);

CREATE TABLE anomalia\_traducao (

id INT NOT NULL PRIMARY KEY,

zona2 VARCHAR(255) NOT NULL,

lingua2 VARCHAR(255) NOT NULL,

FOREIGN KEY (id) REFERENCES anomalia(id) ON DELETE CASCADE

);

CREATE TABLE duplicado (

item1 INT NOT NULL,

item2 INT NOT NULL,

FOREIGN KEY (item1) REFERENCES item(id) ON DELETE CASCADE,

FOREIGN KEY (item2) REFERENCES item(id) ON DELETE CASCADE,

CONSTRAINT pk\_duplicado PRIMARY KEY (item1, item2),

CHECK (item1 < item2)

);

CREATE TABLE utilizador (

email VARCHAR(255) NOT NULL PRIMARY KEY,

password VARCHAR(255) NOT NULL

);

CREATE TABLE utilizador\_qualificado (

email VARCHAR(255) NOT NULL PRIMARY KEY,

FOREIGN KEY (email) REFERENCES utilizador(email) ON DELETE CASCADE

);

CREATE TABLE utilizador\_regular (

email VARCHAR(255) NOT NULL PRIMARY KEY,

FOREIGN KEY (email) REFERENCES utilizador(email) ON DELETE CASCADE

);

CREATE TABLE incidencia (

anomalia\_id INT NOT NULL PRIMARY KEY,

item\_id INT NOT NULL,

email VARCHAR(255) NOT NULL,

FOREIGN KEY (anomalia\_id) REFERENCES anomalia(id) ON DELETE CASCADE,

FOREIGN KEY (item\_id) REFERENCES item(id) ON DELETE CASCADE,

FOREIGN KEY (email) REFERENCES utilizador(email) ON DELETE CASCADE

);

CREATE TABLE proposta\_de\_correcao (

email VARCHAR(255) NOT NULL,

nro SERIAL,

data\_hora TIMESTAMP NOT NULL DEFAULT NOW(),

texto TEXT,

FOREIGN KEY (email) REFERENCES utilizador\_qualificado(email) ON DELETE CASCADE,

CONSTRAINT pk\_proposta\_de\_correcao PRIMARY KEY (email, nro)

);

CREATE TABLE correcao (

email VARCHAR(255) NOT NULL,

nro INT NOT NULL,

anomalia\_id INT NOT NULL,

FOREIGN KEY (nro, email) REFERENCES proposta\_de\_correcao(nro, email) ON DELETE CASCADE,

FOREIGN KEY (anomalia\_id) REFERENCES incidencia(id) ON DELETE CASCADE,

CONSTRAINT pk\_correcao PRIMARY KEY (email, nro, anomalia\_id)

);

Consultas em sql

1. TODO
2. TODO
3. TODO
4. TODO

Explicação da arquitetura da aplicação php

A aplicação php baseia-se numa arquitetura **Model-View**, ou seja, representamos as **tabelas** da base de dados através de modelos (**classes**). As páginas por sua vez utilizam estes modelos para interagir com a base de dados, permitindo assim um maior nível de **abstração** e **modulação**.

Existe uma classe “***Database***” (camada de mais baixo nível) que contém os métodos necessários para inserir, atualizar, pesquisar e eliminar entradas na base de dados com o objetivo de possibilitar um melhor escalonamento da aplicação. Contém também os “***queries***” que serão executados ao longo da aplicação.

Estruturalmente, os modelos são bastante semelhantes entre si:

* Constante, que define o nome da tabela;
* Métodos de procura por coluna(s) e/ou chave(s);
* Método que permite guardar um objeto na base de dados;
* Atributos públicos, que correspondem às colunas da tabela;

A interação com a base de dados é efetuada através destes modelos, que funcionam como um **proxy**.

Decidimos criar **dois** **ficheiros** para incluir em todas as páginas, os chamados “***includes***”, correspondentes ao “***header.php***” e ao “***footer.php***”. As páginas encontram-se dentro de uma diretoria “***pages***” que por sua vez está dividida em **sub-pastas** que contém as diferentes “***views***”. Por exemplo, a sub-pasta “*locais*” contém todos os ficheiros relacionados com os locais públicos.

Por fim, o ficheiro “***índex.php***” contém apenas a navegação para as páginas acima referidas.