Projeto de Bases de Dados - Parte 3

TURNO L04 - PEDRO MANUEL MOREIRA VAZ ANTUNES DE SOUSA

GRUPO 5

Nome(número)	Horas(percentagem)
Pedro Lamego(89526)	12 horas [33.(3)%]
Beatriz Martins(89498)	12 horas [33.(3)%]
Miguel Crespo(90334)	12 horas [33.(3)%]

Ano Letivo 2019/2020

1. Comandos de criação da base de dados

```
DROP TABLE IF EXISTS duplicado;
DROP TABLE IF EXISTS correcao;
DROP TABLE IF EXISTS proposta_de_correcao;
DROP TABLE IF EXISTS incidencia CASCADE;
DROP TABLE IF EXISTS utilizador_qualificado;
DROP TABLE IF EXISTS utilizador_regular;
DROP TABLE IF EXISTS utilizador;
DROP TABLE IF EXISTS anomalia_traducao;
DROP TABLE IF EXISTS anomalia;
DROP TABLE IF EXISTS item;
DROP TABLE IF EXISTS local_publico;

create table local_publico (
        latitude float not null,
        longitude float not null,
        nome varchar(100) not null,
        primary key (latitude, longitude),
        check(-90 <= latitude and latitude <= 90),
        check(-180 <= longitude and longitude <= 180)
);

create table item (
        id SERIAL,
        descricao varchar(1024) not null,
        localizacao varchar(31) not null,
        latitude float not null,
        longitude float not null,
        primary key (id),
        foreign key (latitude, longitude)
                references local_publico(latitude, longitude) ON DELETE CASCADE,
        check(-90 <= latitude and latitude <= 90),
        check(-180 <= longitude and longitude <= 180)
);

create table anomalia (
        id SERIAL,
        zona box not null,
        imagem bytea not null,
        ts timestamp not null,
        lingua varchar(40) not null,
        descricao varchar(1024) not null,
        tem_anomalia_redacao boolean not null,
        primary key (id)
);
```

```
create table anomalia_traducao (
	id integer not null,
	zona2 box not null,
	lingua2 varchar(40) not null,
	primary key (id),
	foreign key (id)
		references anomalia ON DELETE CASCADE
);

create table duplicado (
	item1 integer not null,
	item2 integer not null,
	primary key(item1, item2),
	foreign key (item1)
		references item ON DELETE CASCADE,
	foreign key (item2)
		references item ON DELETE CASCADE,
	check(item1 < item2)
);

create table utilizador (
	email varchar(50) not null,
	password varchar(30) not null,
	primary key (email)
);

create table utilizador_qualificado (
	email varchar(50) not null,
	primary key (email),
	foreign key (email)
		references utilizador ON DELETE CASCADE
);

create table utilizador_regular(
	email varchar(50) not null,
	primary key (email),
	foreign key (email)
		references utilizador ON DELETE CASCADE
);

create table incidencia (
	anomalia_id integer not null,
	item_id integer not null,
	email varchar(50) not null,
	primary key (anomalia_id),
	foreign key (anomalia_id)
		references anomalia ON DELETE CASCADE,
	foreign key (item_id)
```

```
                references item ON DELETE CASCADE,
        foreign key (email)
                references utilizador ON DELETE CASCADE
);

create table proposta_de_correcao(
        email varchar(50) not null,
        nro integer not null,
        data_hora timestamp not null,
        texto varchar(1024) not null,
        primary key (email, nro),
        foreign key (email)
                references utilizador_qualificado ON DELETE CASCADE
);

create table correcao (
        email varchar(50) not null,
        nro integer not null,
        anomalia_id integer not null,
        primary key (email, nro, anomalia_id),
        foreign key (email, nro)
                references proposta_de_correcao(email, nro)
                ON DELETE CASCADE,
        foreign key (anomalia_id)
                references incidencia ON DELETE CASCADE
);
```

2. Consultas em SQL

1.
```
with count_places as(
        select localp.nome, count(localp.nome)
        from local_publico as localp natural join item as it(item_id, descricao, localizacao,
        latitude, longitude) natural join incidencia as inc
        group by localp.nome),
 max_count as(
        select MAX(count)
        from count_places)
select count_p.nome
from count_places as count_p, max_count as max_c
where max_c.max = count_p.count;
```

2.
```
with count_users as(
        select  ureg.email, count(ureg.email)
        from utilizador_regular as ureg natural join incidencia as inc natural join
        anomalia_traducao as anomtrad natural join anomalia as anom
        where anom.ts between '2019-01-01 00:00:00' AND '2019-6-30 23:59:59'
        group by ureg.email),
```

```
        max_count as (
                select MAX(count)
                from count_users
        )
        select count_u.email
        from count_users as count_u, max_count as max_c
        where max_c.max = count_u.count;
```

3.
```
with Rio_Maior as (
        select nome, latitude, longitude
        FROM local_publico WHERE nome = 'Rio Maior'),
  Anomalia_years as(
        select id, extract(year from anomalia.ts) as years
        FROM anomalia)
  select distinct u.email, u.password
  FROM utilizador as u natural join incidencia as inc natural join item as it(item_id, descricao,
  localizacao, latitude, longitude), Anomalia_years as ay, Rio_Maior
  WHERE ay.years = '2019' AND inc.anomalia_id = ay.id AND it.latitude > Rio_Maior.latitude;
```

4.
```
with Rio_Maior as (
        select latitude, longitude
        from local_publico where nome = 'Rio Maior'
  ),
  join_corr_procorr as (
        select * from proposta_de_correcao natural join correcao natural join incidencia
  ),
  join_inc_user as (
        select utq.email, anomalia_id, item_id from incidencia as inc natural join
        utilizador_qualificado as utq
  ),
  except_f as (
        (select email,anomalia_id, item_id from join_inc_user) except (select email,
        anomalia_id, item_id from join_corr_procorr)
  )
  select distinct ex.email from item as it, except_f as ex, Rio_Maior as rm where it.id =
  ex.item_id and it.latitude < rm.latitude;
```

3. Explicação da arquitectura da aplicação PHP

A arquitectura da aplicação PHP é composta por um menu inicial(main.html) onde é possível escolher as operações a realizar: inserir, editar, remover, listar e registar.

Na operação inserir são requeridos campos ao utilizador através de forms que são validados e inseridos na base de dados após o utilizador carregar no botão “submeter”.

Na operação editar é listado o conteúdo através de um pedido à base de dados e o utilizador tem de inserir os dados necessários de forma a que consiga selecionar o que pretende editar. Após isso o utilizador tem apenas de preencher o campo com o novo valor e quando carregar no botão "submeter" o valor na base de dados será atualizado.

Na operação remover é listado o conteúdo através de um pedido à base de dados e o utilizador necessita de inserir os dados necessários de forma a que consiga selecionar o que pretende remover. Após carregar no botão "submeter" os valores serão eliminados da base de dados.

Na operação listar existe apenas uma página em que o utilizador não necessita de inserir dados para que seja possível listar o pretendido. Nas outras páginas o utilizador terá de o fazer e após isso terá de carregar no botão "submeter", fazendo-se um pedido à base de dados e exibindo esse pedido numa tabela com a informação requerida.

Na operação registar são feitos pedidos à base de dados para obter a informação que permite ao utilizador preencher os campos que depois de validados e após o utilizador carregar no botão "submeter" serão inseridos na base de dados.

Em todas as situações em que é necessário fazemos uso das funções de transaction de forma a garantir a atomicidade das operações.

4. Relação entre os diversos ficheiros

A relação entre os diversos ficheiros é a seguinte:

Inserir:

- O ficheiro "insert_place.php" implementa o form "form_insert_place" declarado no ficheiro "place.php"
- O ficheiro "insert_item.php" implementa o form "form_insert_item" declarado no ficheiro "item.php"
- O ficheiro "insert_anomaly_redaction.php" implementa o form "form_insert_anomaly" declarado no ficheiro "anomaly_redaction.php"
- O ficheiro "insert_anomaly_transaction.php" implementa o form "form_insert_anomaly_t" declarado no ficheiro "anomaly_transaction.php"
- O ficheiro "insert_proposal.php" implementa o form "form_insert_proposal" declarado no ficheiro "proposal.php"

Editar:

- O ficheiro "edit_proposal.php" implementa o form "form_choose_proposal" declarado no ficheiro "choose_proposal.php" e o ficheiro "edit_done.php" implementa o form "form_edit_proposal.php"

Remover:

- O ficheiro "remove_place_final.php" implementa o form "form_remove_place" declarado no ficheiro "remove_place.php"
- O ficheiro "remove_item_final.php" implementa o form "form_remove_item" declarado no ficheiro "remove_item.php"

- O ficheiro “remove_anomaly_final.php” implementa o form “form_remove_anomaly” declarado no ficheiro “remove_anomaly.php”
- O ficheiro “remove_proposal_final.php” implementa o form “form_remove_proposal” declarado no ficheiro “remove_proposal.php”

Listar:

- O ficheiro users.php
- O ficheiro “places_final.php” implementa o form “form_insert_places” declarado no “ficheiro anomalies_places.php”
- O ficheiro “degrees_final.php” implementa o form “form_insert_degrees” declarado no “anomalies_degrees.php”

Todos as páginas tem um botão que permite voltar ao menu inicial, seja este o botão “cancelar” ou o botão “home”.