IPS Instituto Politécnico de Setúbal
Escola Superior de Tecnologia de Setúbal

Base de Dados

Lei

Projeto- Sistema de FAQ acerca de compras online (1ºFase)

“ShopSmart”

Ano letivo: 2022/2023

Bernardo Vaz **Tiago Ramada**

202200278 **202200354**

Docente: Claúdio Sapateiro

Turma: 8 (Quinta-feira ás 10:30/12:30)

Índice

Sumário

Compreendendo que o mundo da tecnologia (e não só) está em evolução e com ela a inteligência artificial, sendo que são cada vez mais populares as plataformas de pesquisa relacionado com os mais diversos temas, como grupo decidimos desenvolver uma estrutura direcionada para a compra/venda de produtos online, que com o desenvolvimento da tecnologia e a maior aderência da mesma (compra/venda de produtos online) é cada vez maior.

É o objetivo desta 1º fase do projeto de base de dados, realizarmos uma definição do nosso problema tal como o levantamento, análise e especificação de requisitos para a mesma, apresentação do diagrama Entidade-Relação (DER) e do modelo relacional e ainda a 1ª conversão para o modelo relacional (MR).

Definição do domínio do problema

Tal como é descrito no enunciado do projeto, pedem-nos que seja desenvolvida um sistema de dados que permita uma pesquisa personalizada acerca de uma área temática a nossa escolha, e como dito no sumário decidimos que o nosso sistema responderá acerca de compra/venda de produtos online seja de forma geral, seja por categorias.

Para que tudo isto seja possível, vamos ter que guardar algumas informações acerca do utilizador e da sua sessão previamente iniciada, para que se possa fazer um histórico das pesquisas realizadas, e das questões e respetivas respostas obtidas, de modo a possibilitar ao utilizador uma funcionalidade de feedback (opcional) para com a sua satisfação acerca da plataforma. Para ainda que o sistema seja evolutivo guarda-se o mesmo feedback, e se vários utilizadores forneçam feedback negativo acerca das questões e respostas obtidas, as mesmas devem ser revistas por um utilizador com privilégios admin (explicado mais a frente como se obtém o mesmo).

Entidades

Identificamos como entidades e respetivas chaves as seguintes:

(a sublinhado encontram-se as chaves primarias, a *itálico* as chaves estrangeiras)

- **Questão** (ID_questao, texto_questao, idioma, data_entrada, data_última_atualizacao, *ID_categoria*, *ID_utilizador*);

- **Resposta** (ID_resposta, texto_resposta, data_aprovação, estado, *ID_questao*, *ID_utilizador*);

- **Utilizador** (ID_utilizador, nome, email, data_nascimento, género, password, nacionalidade, admin);

- **Sessão** (ID_sessão, token_identificador, data_hora_início, data_hora_fim, IP, tipo_dispositivo);

- **Administração** (*ID_sessão, admin*);

- **Feedback** (*ID_utilizador*, *ID_sessão*, *ID_resposta*, classificação);

- **Tag** (ID_tag, nome);

- **Categoria** (ID_categoria, nome_categoria, *ID_subcategoria*);

- **Subcategoria** (ID_subcategoria, nome_subcategoria, *ID_categoria*);

- **Pesquisa** (Tipo_pesquisa, *ID_questao*, *ID_resposta*, *ID_categoria*);

Modelo Entidade-Relação

- Utilizador - Resposta ()
- Utilizador - Questão()
- Questão - Resposta()
- Resposta - Feedback()
- Tag - Resposta()
- Tag - Questão()
- Categoria - Subcategoria()
- Categoria - Questão()
- Sessão - Utilizador()
- Utilizador - Administração()
- Utilizador - Feedback()
- Utilizador - Pesquisa()
- Pesquisa - Questão()
- Pesquisa - Tag()
- Pesquisa - Categoria()

Modelo Entidade-Relação (continuação)

Cada sessão terá um utilizador que a iniciará, tendo esta um ID que a identifica tal como a data de inicio/fim da mesma, guardando a informação do tipo dispositivo utilizado como também do IP e do seu token. O utilizador terá acesso então, a fazer pesquisas acerca das questões, tags ou categorias pretendidas na qual obterá questão/questões e resposta/respostas de acordo com o número de tags presentes na questão. Após o mesmo o utilizador terá acesso a fornecer um feedback acerca da resposta/s obtida/s.

Dito isto, para nós cada sessão terá um utilizador, sendo que o utilizador poderá ou não fazer pesquisas, mas as mesmas terão sempre questões/respostas. O utilizador poderá fornecer um feedback de acordo com a sua satisfação com a resposta obtida de 0 a 10. Cada pesquisa vai dar acesso a uma questão/lista de questões (seja através de que tipo for a pesquisa) que estará numa categoria (mesmo que a pesquisa não tenha sido feita por categoria), que depois pode se direcionar para uma subcategoria especifica. As pesquisas podem ser feitas através de questões especificas que originam resposta através de o número de tags, por tags que obtém a questão/resposta com mais tags correspondentes ou categoria, que após indicar a categoria apresentará as questões/respostas dentro dessa categoria sendo possível ainda escolher uma subcategoria (caso exista). Já a administração que é quem fornece os privilégios admin aos utilizadores para estes gerirem as respostas, tem sempre pelo menos 1 admin de forma a manter o bom funcionamento.

Com base na nossa visão acerca deste modelo apresentamos os relacionamentos e restrições (incluindo participação e cardinalidade):

Utilizador – Questão (1…N)

- Um utilizador pode estar associado a N questões (ou não) (participação não obrigatória)
- Uma questão tem 1 utilizador (participação obrigatória)

Utilizador – Resposta (1…N)

- Um utilizador pode estar associado a N respostas (ou não) (participação não obrigatória)
- Uma resposta tem 1 utilizador (participação obrigatória)

Questão – Resposta (1…N)

- Uma questão pode estar associada a N respostas (participação obrigatória)
- Uma resposta tem 1 pergunta (participação obrigatória)

Resposta – Feedback (1…N)

- Uma resposta pode ter N feedbacks (ou não) (participação não obrigatória)
- Um feedback tem sempre uma resposta (participação obrigatória)

Tag – Resposta (N…N)

- Uma tag está associada a N uma respostas (participação obrigatória)
- Uma resposta pode ter N tags (participação obrigatória)

Tag – Questão (N…N)

- Uma tag está associada a N questões (participação obrigatória)
- Uma questão pode ter N tags (participação obrigatória)

Categoria – Subcategoria (1...N)

- Uma categoria pode ter N subcategorias (ou não) (participação não obrigatória)
- Uma subcategoria tem 1 categoria (participação obrigatória)

Categoria – Questão (1…N)

- Uma categoria tem N questões (participação obrigatória)
- Uma questão tem 1 categoria (participação obrigatória)

Sessão – Utilizador (1…1)

- Uma sessão tem 1 utilizador (participação obrigatória)
- Um utilizador tem 1 sessão (participação obrigatória)

Utilizador – Administração (N…1)

- Um utilizador pode ter 1 administração (ou não) (privilégio admin) (participação não obrigatória)
- Uma administração pode ter N utilizadores (com privilégio admin) (participação obrigatória)

Utilizador – Feedback (1…N)

- Um utilizador pode dar N feedbacks (ou não) (participação não obrigatória)
- Um feedback só pode ter 1 utilizador (participação obrigatória)

Utilizador – Pesquisa (1…N)

- Um utilizador pode fazer N pesquisas (ou não) (Participação não obrigatória)
- Uma pesquisa só pode ter 1 utilizador (participação obrigatória)

Pesquisa – Questão (N…1)

- Uma pesquisa vai ter 1 questão (participação obrigatória)

- Uma questão pode ter N pesquisas (participação não obrigatória)

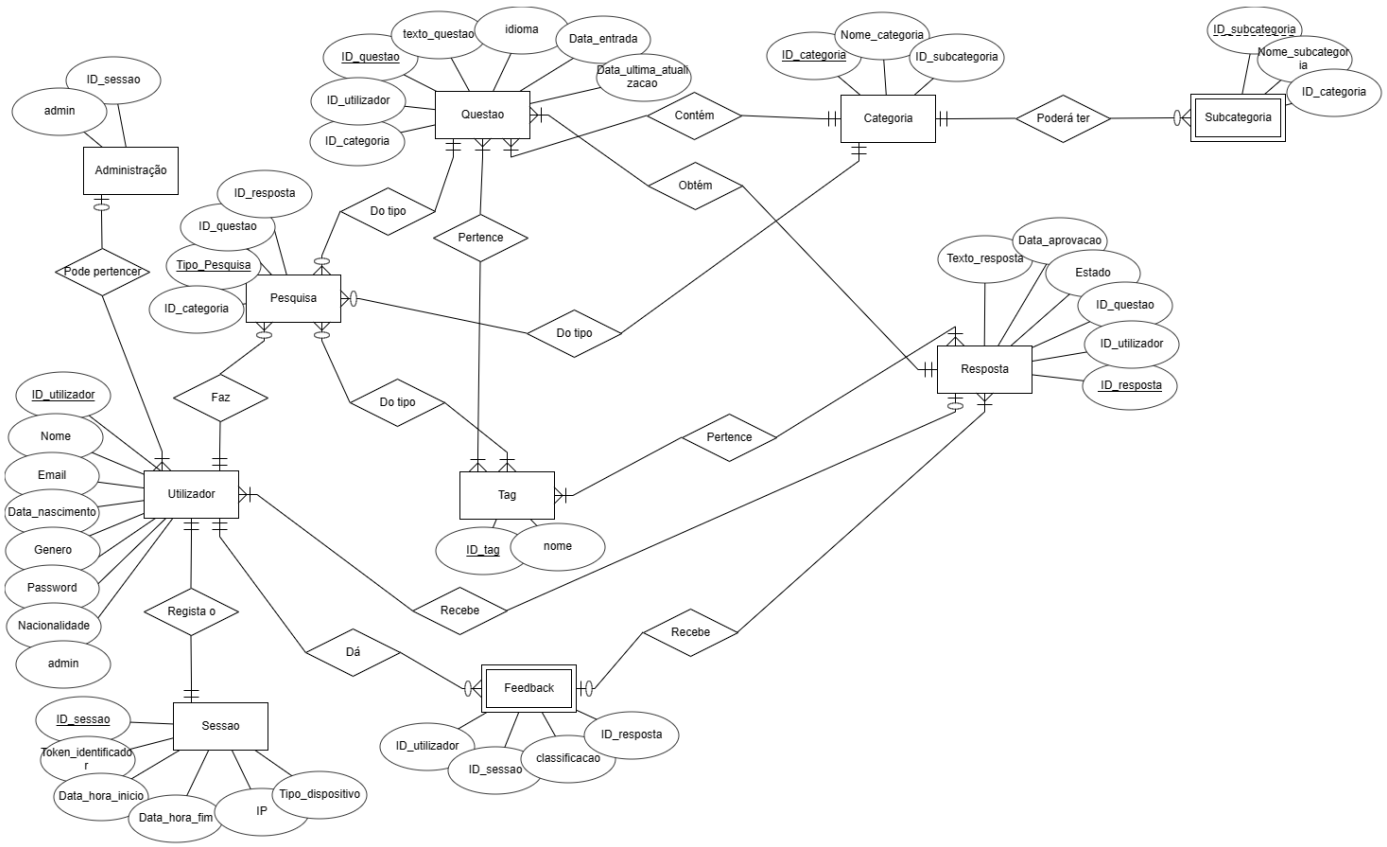
Pesquisa – Tag (N…N)

- Uma pesquisa pode ter N tags (participação obrigatória)
- Uma tag pode ter N pesquisas (participação não obrigatória)

Pesquisa – Categoria (N…1)

- Uma pesquisa tem 1 categoria (participação obrigatória)
- Uma categoria pode ter N pesquisas (participação não obrigatória)

Diagrama Entidade-Relação



Modelo-Relacional

Utilizador (ID_utilizador, nome, email, data_nascimento, género, password, nacionalidade, admin)

PK(ID_utilizador)

FK(ID_resposta -> Resposta(ID_resposta))

Sessão (ID_sessão, token_identificador, data_hora_início, data_hora_fim, IP, tipo_dispositivo)

PK(ID_sessão)

FK(ID_utilizador -> Utilizador(ID_utilizador))

Questão (ID_questao, texto_questao, idioma, data_entrada, data_última_atualizacao, ID_categoria, ID_utilizador)

PK(ID_questao)

FK(ID_resposta -> Resposta(ID_resposta),

ID_categoria -> Categoria(ID_categoria))

Resposta (ID_resposta, texto_resposta, data_aprovação, estado, ID_questão, ID_utilizador);

PK(ID_resposta)

FK(ID_utilizador -> Utilizador(ID_utilizador))

Administração (ID_sessão, admin);

PK(ID_sessão)

FK(ID_utilizador -> Utilizador(ID_utilizador))

Feedback (ID_utilizador, ID_sessão, ID_resposta, classificação)

PK(ID_utilizador, ID_resposta)

FK(ID_utilizador -> Utilizador(ID_utilizador))

ID_resposta -> Resposta(ID_resposta))

Tag (ID_tag, nome);

PK(ID_tag)

FK(Não tem)

Categoria (ID_categoria, nome_categoria, ID_subcategoria);

PK(ID_categoria)

FK(Não tem)

Subcategoria (ID_subcategoria, nome_subcategoria, ID_categoria)

PK(ID_subcategoria)

FK(ID_categoria -> Categoria(ID_categoria))

Pesquisa (Tipo_pesquisa, ID_questao, ID_resposta, ID_categoria);

PK(Tipo_pesquisa)

FK(ID_utilizador -> Utilizador(ID_utilizador)

ID_questao -> Questão(ID_questao)

ID_categoria -> Categoria(ID_Categoria))

Pertence(ID_tag, ID_questao)

PK(ID_tag, ID_questao)

FK(ID_tag -> Tag(ID_tag), ID_questao -> Questão(ID_questao))

Pertence(ID_tag, ID_resposta)

PK(ID_tag, ID_resposta);

FK(ID_tag -> Tag(ID_tag), ID_resposta -> Resposta(ID_resposta))

Do tipo(Tipo_pesquisa, ID_tag)

PK(Tipo_pesquisa, ID_resposta)

FK(Tipo_pesquisa -> Pesquisa(Tipo_pesquisa), ID_tag -> Tag(ID_tag))

Algumas perguntas de exemplo

-**Pesquisa do tipo questão**: *O que fazer caso não receba o produto?* (Encaixa-se na categoria - geral)

R: Contactar o vendedor através do site em uso, tentando saber o motivo do atraso.

-**Pesquisa do tipo tag**: *Como saber a fiabilidade de um vendedor?* (Tags: Fiabilidade e Vendedor) (Encaixa-se na categoria - geral)

R: Para verificar a fiabilidade de um vendedor pode-se no seu perfil, verificar os feedbacks dado por outros utilizadores.

R: O perfil do vendedor no site pode conter informações acerca da sua fiabilidade tais como, se é reconhecido pelo próprio site.

-Pesquisa do tipo categoria: (Categoria - Informática, Subcategoria - Computadores)

Q: *O que fazer se o computador vier com defeito?* (Tags: Defeito).

R: Pode prestar queixa ao vendedor acerca do defeito e pedir troca do mesmo ou reembolso.

Q: *Quanto tempo de garantia tem o meu computador?* (Tags: Tempo, Garantia)

R: Deve consultar as especificações da sua compra, na secção garantia procurar o tempo determinado.

Requisitos

- Temos 10 entidades.
- Pode-se obter uma lista de utilizadores através do nome, nacionalidade, gênero, e-mail e data de nascimento.
- Pode-se obter uma lista de questões e respetivos elementos através da pesquisa por categoria, pesquisa especifica ou pesquisa utilizando as tags.
- Podemos obter a lista de respostas a uma questão através da pesquisa especifica da questão que devolve as respostas á mesma ou através da pesquisa por tags que devolve as respostas com mais tags correspondentes com a questão.
- Pode-se obter uma lista de sessões através do tipo de dispositivo utilizado, do ID da sessão e pelas datas de início/fim.

Limitações

Algumas das limitações presentes á nossa 1ªfase são:

- Assumimos que todas as pesquisas irão ter questões e respostas associadas o que numa situação real não é estritamente necessário pois pode haver pesquisas que ainda não estivessem no sistema e tivessem de ser adicionadas.
- Acerca das categorias/subcategorias só temos em conta algumas já que sendo o tema abrangente poderia levar a um aumento enorme da complexidade desta base de dados. (preparação para segunda fase)

Inovações

Algumas inovações que consideramos presentes são:

- É possível escolher uma categoria e ter acesso logo as perguntas mais comuns acerca dessa categoria (como visto nas perguntas de exemplo).
- Apesar de só considerarmos algumas categorias/subcategorias seria relativamente fácil adicionar mais, contudo como dito nas limitações levaria a um aumento da sua complexidade.
- Temos diferentes tipos de pesquisa o que pode proporcionar ao utilizador uma melhor experiência.

Conclusão

Esta primeira fase deste projeto, serviu-nos como forma de consolidação das aprendizagens das aulas e estudo, desde a identificação de entidades e respetivos atributos, passando pelos relacionamentos entre as diferentes entidades, as suas participações e cardinalidades, até ao diagrama proposto.

Temos presente um sentimento que fomos bem-sucedidos e que cumprimos para com os requisitos do enunciado e de que com este projeto ficamos a compreender melhor estes conceitos referidos no parágrafo acima.

2º Fase do Projeto

Para começar, será necessário executar os scripts fornecidos pela seguinte ordem:

1. **Create.sql** – que é o script que contém o código para a criação das tabelas e a associação de chaves estrangeiras.

2. **Logic.sql** – contém parte da logica das funcionalidades implementadas, como as views, storedProcedures, storedFunctions, e os triggers.

3. **populate.sql** – script que contém todas as entradas para as tabelas criadas.

4. **queries.sql** – script que contém a implementação das consultas à base de dados.

5. **results.sql** – script que contém a lógica pedida no tópico de resultados.

6. **test.sql** - script que contém a aplicação de todas as funcionalidades mencionadas no enunciado.

7. **test_triggers** - script que permite testar os dois triggers desenvolvidos.

Alterações feitas em relação a primeira fase do projeto

Na discussão da primeira fase com o nosso professor de laboratório, foram sugeridas algumas alterações, contudo decidimos não as concretizar apesar de sabermos que as mudanças sugeridas foram com o intuito de nos ajudar e facilitar a implementação desta segunda fase.

Vamos apresentar agora o conjunto de tabelas e os seus atributos, identificados com o tipo de dados utilizado para a logica:

Tabelas:

Tabela Utilizador:

ID_utilizador INT PRIMARY KEY auto_increment,

nome VARCHAR(50) NOT NULL,

email VARCHAR(100),

data_nascimento DATE,

genero VARCHAR(10),

password VARCHAR(100),

nacionalidade VARCHAR(50),

admin BOOLEAN,

Tabela Subcategoria:

ID_subcategoria INT PRIMARY KEY auto_increment,

nome_subcategoria VARCHAR(50)

Tabela Categoria:

ID_categoria INT PRIMARY KEY auto_increment,

nome_categoria VARCHAR(50),

ID_subcategoria INT

Tabela Questão:

ID_questao INT PRIMARY KEY auto_increment,

texto_questao TEXT,

idioma VARCHAR(50),

data_entrada DATE,

data_ultima_atualizacao DATE,

ID_categoria INT,

ID_utilizador INT

Tabela Resposta:

ID_resposta INT PRIMARY KEY auto_increment,

texto_resposta TEXT,

data_aprovacao DATE,

estado VARCHAR(50),

ID_questao INT,

ID_utilizador INT

Tabela Sessão:

ID_sessao INT PRIMARY KEY auto_increment,

token_identificador VARCHAR(100),

data_hora_inicio DATETIME,

data_hora_fim DATETIME,

IP VARCHAR(50),

tipo_dispositivo VARCHAR(50),

ID_utilizador INT

Tabela Administração:

ID_sessao INT PRIMARY KEY auto_increment,

admin BOOLEAN,

ID_utilizador INT

Tabela Feedback:

ID_utilizador INT auto_increment,

ID_sessao INT,

ID_resposta INT,

classificacao INT,

PRIMARY KEY (ID_utilizador, ID_resposta)

Tabela Tag:

ID_tag INT PRIMARY KEY auto_increment,

nome VARCHAR(50)

Tabela Pesquisa:

Tipo_pesquisa VARCHAR(50) PRIMARY KEY,

ID_questao INT,

ID_resposta INT,

ID_categoria INT,

ID_utilizador INT

Chaves estrangeiras:

Agora passamos ás chaves estrangeiras estabelecidas entre as tabelas:

Tabela Categoria- o atributo (ID_Subcategoria) referencia a tabela Subcategoria o atributo (ID_Subcategoria);

Tabela Questão- o atributo (ID_Categoria) referencia a tabela Categoria o atributo (ID_Categoria);

Tabela Resposta - o atributo (ID_questao) referencia a tabela Questão o atributo (ID_questao);

Tabela Resposta - o atributo (ID_utilizador) referencia a tabela Utilizador o atributo (ID_utilizador);

Tabela Sessão - o atributo (ID_utilizador) referencia a tabela Utilizador o atributo (ID_utilizador);

Tabela Administração - o atributo (ID_utilizador) referencia a tabela Utilizador o atributo (ID_utilizador);

Tabela Administração - o atributo (ID_sessao) referencia a tabela Sessão o atributo (ID_sessao);

Tabela Feedback - o atributo (ID_utilizador) referencia a tabela Utilizador o atributo (ID_utilizador);

Tabela Feedback - o atributo (ID_sessao) referencia a tabela Sessão o atributo (ID_sessao);

Tabela Feedback - o atributo (ID_resposta) referencia a tabela Resposta o atributo (ID_resposta);

Tabela Pesquisa - o atributo (ID_utilizador) referencia a tabela Utilizador o atributo (ID_utilizador);

Tabela Pesquisa - o atributo (ID_questao) referencia a tabela Questão o atributo (ID_questao);

Tabela Pesquisa - o atributo (ID_resposta) referencia a tabela Resposta o atributo (ID_resposta);

Tabela Pesquisa - o atributo (ID_categoria) referencia a tabela Categoria o atributo (ID_categoria);

Na criação de todas estas foram colocadas as instruções: ON DELETE NO ACTION ON UPDATE NO ACTION; de modo a que estas quando sejam alteradas ou apagadas quando contiverem dados, não altere o conteúdo das outras tabelas relacionadas.

Entradas das tabelas

Para popularmos as nossas tabelas optamos por utilizar instruções insert, apesar de termos algumas stored procedures para o devido efeito e são elas:

Call newUser();

Call newCategory();

Call newFeedback();

Call newQuestion();

Critérios de consulta à base de dados

1.1->Ordenar por nacionalidade

```
SELECT * FROM Utilizador ORDER BY nacionalidade;
```

1.2 -> Ordenar por utilizadores do sexo feminino

```
SELECT * FROM Utilizador WHERE genero = 'Feminino';
```

1.3 -> Filtrar por utilizadores com contas ativas

```
SELECT * FROM Utilizador WHERE admin = true;
```

2.1 -> Ordenar por data de entrada na forma decrescente

```
SELECT * FROM Questao ORDER BY data_entrada DESC;
```

2.2 -> Filtrar por questões com o idioma Português

```
SELECT * FROM Questao WHERE idioma = "Português";
```

3.1 -> Filtrar por respostas aprovadas

```
SELECT * FROM Resposta WHERE estado = "Aprovada";
```

3.2-> Não implementado

4.1 -> Filtrar por sessões com dispositivos móveis

```
SELECT * FROM Sessao WHERE tipo_dispositivo = 'Móvel';
```

4.2 ->Filtrar por sessões em que a data é menor que:

SELECT * FROM Sessao WHERE data_hora_inicio < '2023-01-01 12:00:00';

4.3 -> Filtrar por sessões com um IP específico

SELECT * FROM Sessao WHERE IP = '192.168.0.1';

5 -> Lista de sessões de cada utilizador ordenadas por duração

SELECT ID_utilizador, data_hora_inicio, data_hora_fim, data_hora_fim - data_hora_inicio AS duracao FROM Sessao ORDER BY ID_utilizador, duracao;

6 -> Lista das questões que não foram pesquisadas no ano atual

SELECT * FROM Questao WHERE ID_questao NOT IN (SELECT DISTINCT ID_questao FROM Pesquisa WHERE YEAR(CURDATE()) = YEAR(NOW()));

7 ->Não implementado

8.1 -> Agrupar por tipo de pesquisa e calcular as estatísticas

SELECT Tipo_pesquisa, AVG(ID_questao) AS media_questao, MIN(ID_questao) AS min_questao, MAX(ID_questao) AS max_questao, STDDEV(ID_questao) AS desvio_questao FROM Pesquisa GROUP BY Tipo_pesquisa;

8.2 -> Filtrar por pesquisas realizadas por utilizadores com ID superior a 10

SELECT Tipo_pesquisa, AVG(ID_resposta) AS media_resposta, MIN(ID_resposta) AS min_resposta, MAX(ID_resposta) AS max_resposta, STDDEV(ID_resposta) AS desvio_resposta FROM Pesquisa GROUP BY Tipo_pesquisa;

9 -> Lista com o número médio, mínimo, máximo e desvio padrão da classificação das respostas por categoria de questão

SELECT c.nome_categoria AS categoria, AVG(f.classificacao) AS media_classificacao, MIN(f.classificacao) AS min_classificacao, MAX(f.classificacao) AS max_classificacao, STDDEV(f.classificacao) AS desvio_classificacao FROM Feedback AS f JOIN Resposta AS r ON f.ID_resposta = r.ID_resposta JOIN Questao AS q ON r.ID_questao = q.ID_questao JOIN Categoria AS c ON q.ID_categoria = c.ID_categoria GROUP BY c.nome_categoria;

10-> Não implementado

11 -> Lista dos 5 utilizadores com mais pesquisas

SELECT u.nome, COUNT(*) AS total_pesquisas FROM Utilizador AS u JOIN Pesquisa AS p ON u.ID_utilizador = p.ID_utilizador GROUP BY u.nome ORDER BY total_pesquisas DESC LIMIT 5;

12 -> Consulta adicional recorrendo a, pelo menos, 3 tabelas

SELECT q.texto_questao AS questao, r.texto_resposta AS resposta, u.nome AS utilizador FROM Questao q JOIN Resposta r ON q.ID_questao = r.ID_questao JOIN Utilizador u ON r.ID_utilizador = u.ID_utilizador WHERE r.estado = 'Aprovada' LIMIT 10;

13 -> Não implementado

14 -> Não implementado

Registo de resultados

Call sp_criar_sessao();

Stored procedure que cria uma nova sessão a um utilizador fornecendo os dados necessários.

Call sp_adicionar_questao();

Stored procedure que serve para adicionar uma questão a lista de questões de uma dada sessão fornecida.

Call sp_remover_sessao();

Stored procedure que serve para remover a sessão fornecida por parâmetro, verificando as condições para a mesma.

Call sp_clonar_sessao();

Stored procedure que serve para criar uma nova sessão com uma cópia de dados de uma dada sessão.

Call sp_registar_resposta();

Stored procedure que serve para registar as respostas resultantes de uma pesquisa na respetiva sessão.

Remoção de dados

Call removeUser();

Stored procedure que remove um utilizador fornecido por parâmetro através do id de utilizador.

Call removeCategory();

Stored procedure que remove categoria através da passagem do id_categoria por parâmetro.

Call removeFeedback();

Stored procedure que permite remover o feedback, id_feedback fornecido por parâmetro.

Call removeQuestion();

Stored procedure que permite remover uma questão e todos os registos associados, recebe o id da questão por parâmetro.

Call removeAwnser();

Stored procedure que remove a resposta através do id_resposta fornecido por parâmetro.

Call removeSession();

Stored procedure que remove uma sessão fornecida por parâmetro e os registos associados.

Call removeSubcategory();

Stored procedure que remove uma subcategoria através do id_subcategoria fornecido por parâmetro.

Call removeSearch();

Stored procedure que permite remover uma pesquisa através dos ids fornecidos acerca da pesquisa, como o utilizador, categoria, resposta e questão.

Call removeTag();

Stored procedure que permite remover uma tag através do id_tag.

Stored functions

Temos 2 stored functions que são:

CalculaMediaFeedback();

Que serve para calcular a media da classificação do feedback fornecido de um dado utilizador, retornando assim em valor decimal o mesmo.

VerificaPermissaoAdmin();

Que serve para verificar se um dado utilizador tem as permissões de admin, retornando um tipo booleano.

Conclusão (2ºFase)

Começamos desde já por constatar que praticamente todos os materiais aprendidos nas aulas tiverem de ser utilizados, desde a criação das tabelas, popular as mesmas, criação da logica e implementação das consultas foi necessário aplicar todos estes materiais lecionados de extrema importância para a passagem da primeira fase que era mais teórica para aqui a implementação real da primeira fase.

Admitimos que sentimos dificuldades, e que todo este projeto envolveu bastante trabalho e aplicação da nossa parte, sendo que com isto dito gostaríamos de revelar que achamos que apesar das dificuldades sentidas, e ultrapassadas, das horas de trabalho até a concretização temos connosco presente um sentimento de que cumprimos o que era pretendido, obviamente com algumas limitações e prováveis falhas (o que é normal e expetável dado o nosso contexto académico) que nos serão apontadas na discussão de projeto, mesmo assim achamos que conquistámos um nível satisfatório.

Limitações(2ºFase)

Em termos das limitações não conseguimos implementar a nível da consulta á base de dados, os números 3.2, 7, 10, 13 e 14 sendo que preferimos dizer aqui que não conseguimos do que certamente entregar algo visivelmente errado ou com falhas, porque dado o nosso contexto de aprendizagem só assim aprendemos.