LABORATÓRIO: EXEMPLO DE UTILIZAÇÃO DE BASE DE DADOS

O SQL (Structured Query Language) é uma linguagem de programação projetada especificamente para a criação, gestão e manipulação de bases de dados relacionais. Criada inicialmente na década de 1970, tornou-se o padrão de facto para interagir com sistemas de gestão de bases de dados relacionais (SGBDs). Esta linguagem é amplamente utilizada em SGBDs populares, como MySQL, PostgreSQL, SQL Server e Oracle, e desempenha um papel crucial em aplicações que dependem da gestão eficiente de dados estruturados.

Com o SQL, os utilizadores podem realizar uma ampla gama de operações que incluem desde a conceção inicial de estruturas de dados até tarefas complexas de análise e gestão. Entre as funcionalidades mais relevantes do SQL, destacam-se:

- **Criar estruturas de dados**: Permite a definição de tabelas, índices e relações entre os diferentes elementos de dados. Estas operações são fundamentais para garantir que a base de dados seja organizada e otimizada para as operações futuras.
- **Manipular dados**: Possibilita a inserção, atualização e eliminação de dados de forma eficiente. Estas operações são essenciais para manter a base de dados atualizada e em conformidade com as necessidades da aplicação ou do negócio.
- **Consultar informações**: Com a utilização de instruções SQL, é possível recuperar, organizar e filtrar dados de acordo com critérios personalizados. Este é um dos aspetos mais poderosos do SQL, uma vez que permite transformar grandes volumes de dados em informações úteis para a tomada de decisões.
- **Gerir permissões**: O SQL também inclui comandos para gerir a segurança dos dados, controlando o acesso de utilizadores a informações sensíveis. Esta funcionalidade é especialmente importante em ambientes empresariais, onde a proteção de dados é uma prioridade.

Neste laboratório, vamos explorar estas funcionalidades através de um cenário prático focado na gestão de operações navais. A nossa abordagem incluirá:

1. **Criação de esquemas de bases de dados**: Aprenderemos a definir tabelas, especificar os relacionamentos entre elas e configurar os elementos estruturais necessários para suportar os dados.
2. **Inserção de dados**: Praticaremos a utilização de comandos SQL para preencher as tabelas com informações relevantes, como dados de embarcações, tripulações, missões e operações.
3. **Consultas personalizadas**: Realizaremos consultas para recuperar informações específicas e relevantes para o contexto das operações navais. Por exemplo, será possível identificar quais embarcações estão alocadas a determinadas missões ou quais tripulações estão disponíveis para operações futuras.

4. **Gestão de permissões**: Por fim, iremos explorar os comandos que permitem gerir o acesso a dados críticos, garantindo que apenas utilizadores autorizados possam realizar determinadas operações.

CONFIGURAÇÃO DO AMBIENTE

Para executar instruções SQL, sem ter de instalar o servidor localmente consiste em recorrer a plataformas online que disponibilizam ambientes de teste para SQL. O serviço sqliteonline.com, apesar do nome, oferece a possibilidade de usar diferentes motores de bases de dados, incluindo MariaDB, o qual, na maioria dos casos, é compatível com comandos e funcionalidades de MySQL. Segue-se uma explicação detalhada dos passos necessários para efetuar o carregamento e a execução do ficheiro `sn.sql`:

1. **Escolher o motor de bases de dados** Assim que aceder a sqliteonline.com, procure, na parte superior ou lateral do ecrã, a secção de seleção de motores de bases de dados. É essencial selecionar a opção **MariaDB** (ou MariaDB (MySQL mode)), pois este motor é amplamente compatível com a sintaxe e as características habituais do MySQL. Ao selecionar MariaDB, a plataforma iniciará uma instância virtual de MariaDB, permitindo-lhe correr os comandos SQL que, em ambiente local, correria num servidor MySQL.
2. **Estabelecer a ligação ao motor** Depois de selecionar MariaDB, deverá visualizar um botão ou ligação similar a “Click to connect”. Ao clicar nesta opção, o serviço configura um ambiente de trabalho onde será possível executar scripts SQL, criar tabelas, inserir dados e efetuar consultas. Aguarde alguns instantes até que o ambiente esteja preparado; poderá observar uma mensagem de confirmação ou a disponibilização da consola interativa.

Para se familiarizar com o SQLiteOnline, recomendo assistir ao seguinte vídeo:

- Assista ao vídeo no YouTube

CRIAÇÃO DAS TABELAS

Copie e execute o seguinte código para criar as tabelas no seu banco de dados:

```sql
-- Apagar tabelas existentes para evitar conflitos
DROP TABLE IF EXISTS Operacao;
DROP TABLE IF EXISTS Missao;
DROP TABLE IF EXISTS Tripulacao;
DROP TABLE IF EXISTS Embarcacao;

-- Criar a tabela Embarcacao
CREATE TABLE Embarcacao (
 Id INT AUTO_INCREMENT PRIMARY KEY,
 Nome VARCHAR(100) NOT NULL,
 Tipo VARCHAR(50),
 Capacidade INT,
 Estado VARCHAR(50)
);

-- Criar a tabela Tripulacao
CREATE TABLE Tripulacao (
 Id INT AUTO_INCREMENT PRIMARY KEY,
 Nome VARCHAR(100) NOT NULL,
 Patente VARCHAR(50),
 Especializacao VARCHAR(50),
 Embarcacao_Id INT,
 FOREIGN KEY (Embarcacao_Id) REFERENCES Embarcacao(Id)
);

-- Criar a tabela Missao
CREATE TABLE Missao (
 Id INT AUTO_INCREMENT PRIMARY KEY,
 Objetivo VARCHAR(255),
 Area VARCHAR(100),
 Data_Inicio DATE,
 Data_Fim DATE
);

-- Criar a tabela Operacao
CREATE TABLE Operacao (
 Id INT AUTO_INCREMENT PRIMARY KEY,
 Missao_Id INT,
 Data DATE,
 Resultado VARCHAR(100),
 Observacoes TEXT,
 FOREIGN KEY (Missao_Id) REFERENCES Missao(Id)
);
```

INSERÇÃO DE DADOS

Execute o seguinte código para inserir registros fictícios:

```
-- Inserir dados na tabela Embarcacao
INSERT INTO Embarcacao (Nome, Tipo, Capacidade, Estado)
VALUES
('Navio Alpha', 'Fragata', 200, 'Operacional'),
('Navio Beta', 'Submarino', 50, 'Manutenção'),
('Navio Gamma', 'Corveta', 100, 'Operacional');

-- Inserir dados na tabela Tripulacao
INSERT INTO Tripulacao (Nome, Patente, Especializacao, Embarcacao_Id)
VALUES
('João Silva', 'Capitão', 'Navegação', 1),
('Maria Oliveira', 'Tenente', 'Comunicação', 1),
('Carlos Ferreira', 'Sargento', 'Engenharia', 2);

-- Inserir dados na tabela Missao
INSERT INTO Missao (Objetivo, Area, Data_Inicio, Data_Fim)
VALUES
('Patrulha Costeira', 'Atlântico Norte', '2023-01-10', '2023-01-20'),
('Operação de Resgate', 'Mar Mediterrâneo', '2023-02-15', '2023-02-18');

-- Inserir dados na tabela Operacao
INSERT INTO Operacao (Missao_Id, Data, Resultado, Observacoes)
VALUES
(1, '2023-01-12', 'Sucesso', 'Patrulha realizada com sucesso.'),
(1, '2023-01-18', 'Sucesso', 'Nenhuma anomalia detectada.'),
(2, '2023-02-16', 'Sucesso', 'Tripulantes resgatados em segurança.');
```

CONSULTAS SELECT

CONSULTAS BÁSICAS

1. Listar todas as embarcações

```sql
SELECT * FROM Embarcacao;
```

2. Exibir o nome e a patente dos membros da tripulação

```sql
SELECT Nome, Patente FROM Tripulacao;
```

3. Obter os detalhes das missões

```sql
SELECT * FROM Missao;
```

CONSULTAS COM FILTROS

1. Exibir embarcações que estão operacionais

```sql
SELECT Nome, Tipo FROM Embarcacao WHERE Estado = 'Operacional';
```

2. Listar operações realizadas na missão com ID 1

```sql
SELECT * FROM Operacao WHERE Missao_Id = 1;
```

CONSULTAS COM JUNÇÃO (JOIN)

1. Listar os nomes das embarcações e seus capitães

```sql
SELECT e.Nome AS Embarcacao, t.Nome AS Capitao
FROM Embarcacao e
JOIN Tripulacao t ON e.Id = t.Embarcacao_Id
WHERE t.Patente = 'Capitão';
```

---

2. Exibir as operações realizadas com o objetivo das missões correspondentes

```
SELECT m.Objetivo, o.Data, o.Resultado
FROM Operacao o
JOIN Missao m ON o.Missao_Id = m.Id;
```

CONSULTAS COM FUNÇÕES AGREGADAS

1. Contar o número de tripulantes em cada embarcação

```
SELECT e.Nome AS Embarcacao, COUNT(t.Id) AS Total_Tripulantes
FROM Embarcacao e
LEFT JOIN Tripulacao t ON e.Id = t.Embarcacao_Id
GROUP BY e.Nome;
```

2. Listar as missões com duração superior a 5 dias

```
SELECT Objetivo, DATEDIFF(Data_Fim, Data_Inicio) AS Duracao
FROM Missao
WHERE DATEDIFF(Data_Fim, Data_Inicio) > 5;
```

QUESTÕES

1. CRIAÇÃO DE ESTRUTURAS DE DADOS

Questões:

1. Os campos definidos para cada tabela representam adequadamente os dados pretendidos?
2. Existem relacionamentos claros entre as tabelas? Eles refletem corretamente as interdependências do problema?
3. Como poderiam ser criadas restrições adicionais para garantir integridade dos dados, como valores obrigatórios ou relações válidas?

Respostas:

1. **Sim**, os campos definidos são adequados, mas podem ser expandidos se novos requisitos surgirem. Por exemplo, a tabela `Embarcação` poderia incluir um campo para localização atual.
2. **Sim**, as chaves estrangeiras refletem bem os relacionamentos (e.g., `Embarcacao_Id` em `Tripulacao`). No entanto, deve-se verificar se todas as relações necessárias estão presentes.
3. Restrições como `NOT NULL` para campos essenciais ou `UNIQUE` para IDs poderiam ser adicionadas para evitar dados inválidos ou duplicados.

2. INSERÇÃO DE DADOS

Questões:

1. Os dados inseridos seguem o formato correto definido na estrutura da tabela?
2. Existe duplicação de informações? Quais seriam os impactos dessa duplicação nos resultados das consultas?
3. Como as referências entre tabelas afetam a integridade das relações criadas?

Respostas:

1. **Sim**, os dados inseridos seguem os formatos esperados, mas deve-se garantir que valores como `Capacidade` sejam sempre positivos.
2. Não foram encontrados duplicados nas inserções iniciais. Caso houvesse, poderiam causar erros em consultas ou operações como atualizações e exclusões.
3. Referências válidas garantem que dados relacionados sejam consistentes, como garantir que uma tripulação esteja sempre associada a uma embarcação existente.

3. CONSULTAS PERSONALIZADAS

Questões:

1. Os resultados da consulta correspondem às expectativas? Caso contrário, qual parte do comando pode estar incorreta?
2. Como podem ser adicionados filtros ou junções para refinar ainda mais os resultados?
3. Os resultados fornecem uma visão útil para a tomada de decisão? Se não, o que poderia ser alterado na estrutura dos dados ou nas consultas?

Respostas:

1. **Sim**, os resultados correspondem às expectativas. Se não, deve-se verificar a lógica das consultas, como erros em condições ou junções.
2. Filtros podem ser adicionados para refinar os resultados, como listar apenas embarcações operacionais ou missões de uma região específica.
3. **Sim**, as consultas fornecem informações úteis. Caso contrário, seria necessário ajustar os campos ou criar novas tabelas para responder às perguntas do negócio.

4. IDENTIFICAÇÃO E CORREÇÃO DE PROBLEMAS

Questões:

1. Quais foram as possíveis causas para a existência de duplicados ou anomalias?
2. Quais as implicações dessas inconsistências na operação da base de dados?
3. Que restrições ou validações poderiam ter sido implementadas para evitar esses problemas?

Respostas:

1. A ausência de restrições como `UNIQUE` e `NOT NULL` pode permitir duplicados e anomalias.
2. Anomalias podem causar inconsistências em consultas e relatórios, além de complicar atualizações e exclusões.
3. Restrições como `UNIQUE` em IDs e validações para campos obrigatórios poderiam prevenir a inserção de registos inválidos.

5. ALTERAÇÕES NA ESTRUTURA

Questões:

1. As alterações implementadas corrigiram os problemas identificados? Se não, o que mais precisa ser ajustado?
2. Como as novas restrições impactam a flexibilidade da base de dados para novas operações?
3. Existem impactos negativos no desempenho ou na manutenção após as alterações realizadas?

Respostas:

1. **Sim**, as alterações eliminaram duplicados e corrigiram referências inválidas. Deve-se revisar periodicamente as tabelas para evitar novos problemas.
2. Restrições melhoram a consistência, mas podem limitar a flexibilidade se forem excessivamente rígidas.
3. Restrições adicionais podem impactar ligeiramente o desempenho em operações massivas, mas são compensadas pela melhoria na integridade dos dados.

6. IMPACTO GERAL

Questões:

1. Como esta operação ajuda a melhorar a consistência e a integridade dos dados?
2. Como a base de dados criada responde aos requisitos do problema apresentado?
3. O que poderia ser otimizado para melhorar a eficiência ou a facilidade de manutenção?

Respostas:

1. A operação elimina duplicados, corrige referências inválidas e assegura que os dados respeitem as relações definidas.
2. A estrutura reflete corretamente os requisitos, permitindo consultas eficazes e dados fiáveis.
3. Adicionar índices para acelerar consultas frequentes e documentar claramente as relações e restrições aplicadas pode melhorar a eficiência e a manutenção.