

Trabalho Experimental

Fase 2 – Trabalho C

Inserção de dados
Seleção de dados

Licenciatura em Engenharia Informática
Base de Dados

Paulo Nogueira Martins
Daniel Moreira Lopes Alexandre

Autores

Diogo António Costa Medeiros n.º 70633

Pedro Miguel Cunha da Silva n.º 70649

Rui João Barros Pinto n.º 70648

Vila Real, maio 2021

ÍNDICE

1. INTRODUÇÃO	3
2. ENQUADRAMENTO TEÓRICO	3
3. OBJETIVOS DO TRABALHO PRÁTICO	4
4. DESENVOLVIMENTO	5
4.1 INSERÇÃO DE REGISTOS	5
4.2 QUERIES DE SELEÇÃO DE DADOS	9
5 CONCLUSÃO	12
6 BIBLIOGRAFIA	12

1. INTRODUÇÃO

SQL foi desenvolvida pela IBM Corporation no final da década de 70. Foi adotada como um padrão nacional pelo Instituto Nacional Americano de Padrões (ANSI) em 1986 e pela Organização Internacional de Normalização (ISO) em 1987 (Kroenke & Auer, 2016).

SQL não é uma linguagem de programação completa, como o Java ou C#. É, sim uma sublinguagem de dados pois apenas contém as declarações necessárias para criar e processar dados e metadados de uma base de dados (Kroenke & Auer, 2016).

As declarações SQL dividem-se, habitualmente, em várias categorias, das quais se destacam:

- ❖ Declarações de linguagem de definição de dados (DDL) que são usadas na criação de tabelas, relações e outras estruturas;
- ❖ Declarações de linguagem de manipulação de dados (DML) que são usadas para consulta, inserção, modificação, e eliminação de dados.

2. ENQUADRAMENTO TEÓRICO

INSERT é uma declaração SQL que permite adicionar um ou mais registos a uma tabela de uma base de dados. Estas declarações seguem a seguinte sintaxe:

- ❖ INSERT INTO *tabela* (*coluna1*, [*coluna2*, ...]) VALUES (*valor1*, [*valor2*, ...])

Os valores inseridos pela declaração INSERT devem satisfazer todas as restrições presentes na respetiva tabela, tais como chaves (primárias e estrangeiras), restrições CHECK e restrições NOT NULL). Por outro lado, é possível inserir um valor por defeito desde que a respetiva coluna contenha uma restrição DEFAULT, através da omissão da mesma, (Insert (SQL), 2021).

A estrutura básica duma query SQL consiste de 3 cláusulas: SELECT, FROM e WHERE, sendo este último opcional.

Uma query recebe como input as relações listadas na cláusula FROM, opera sobre estas de acordo com o especificado nas cláusulas WHERE e SELECT e, de seguida, produz uma relação como resultado (Silberschatz, Korth, & Sudarshan, 2020).

Por vezes, pretende-se consultar todos os registos, sem restringir as colunas. Nestas situações, um asterisco ‘*’ pode ser usado para indicar que a consulta deve devolver todas as colunas das tabelas listadas (Select (SQL), 2021).

SELECT é a instrução mais complexa em SQL, com palavras-chave e cláusulas opcionais, i.e.:

- ❖ A cláusula FROM pode incluir subcláusulas JOIN opcionais para especificar as regras de junção de tabelas: FROM tabela1 [tipo] JOIN tabela2 [ON (condição)], onde o [tipo] (opcional) pode ser INNER, LEFT [OUTER], RIGHT [OUTER] ou CROSS (Microsoft Corporation, 2021).
- ❖ A cláusula WHERE inclui um predicado de comparação, que restringe as linhas retornadas pela consulta. Este pode ser composto por várias condições, que podem incluir operadores de comparação ou predicados do tipo LIKE, BETWEEN e IN, usando os operadores AND e OR.
- ❖ A cláusula GROUP BY projeta linhas com valores comuns num conjunto menor de linhas. GROUP BY é frequentemente usado em conjunto com funções de agregação SQL, tais como MAX, MIN, AVG, SUM ou COUNT, ou para eliminar linhas duplicadas de um conjunto de resultados. A cláusula WHERE é aplicada antes da cláusula GROUP BY.
- ❖ A cláusula HAVING inclui um predicado usado para filtrar linhas resultantes da cláusula GROUP BY. Como ela atua sobre os resultados desta cláusula, as funções de agregação podem ser usadas no predicado da cláusula HAVING.
- ❖ A cláusula ORDER BY identifica quais as colunas a usar para ordenar os dados resultantes e qual a ordem, crescente (ASC) ou decrescente (DESC).
- ❖ Sem uma cláusula ORDER BY, a ordem das linhas retornadas por uma consulta SQL é indefinida.
- ❖ A palavra-chave DISTINCT elimina dados duplicados.

3. OBJETIVOS DO TRABALHO PRÁTICO

Foi solicitada a inserção de registos nas tabelas da base de dados implementada na fase 1, bem como a criação de *queries* que respondessem a uma listagem de perguntas previamente definidas.

4. DESENVOLVIMENTO

4.1 INSERÇÃO DE REGISTOS

```
INSERT INTO NIFs (NIF, Nome, Apelido, Telefone)
```

```
VALUES
```

```
(100000001, 'Joaquim', 'Macedo', 976711074),  
(100000002, 'Maria', 'Ventura', 947089902),  
(100000003, 'Joana', 'Marques', 986058286),  
(100000004, 'Diogo', 'Antunes', 281876574),  
(100000005, 'Rui', 'Alexandre', 234686761),  
(100000006, 'Ricardo', 'Araujo', 260267139),  
(100000007, 'Judite', 'Pereira', 937466897),  
(100000008, 'Elizabete', 'Rodrigues', 917342571),  
(100000009, 'Joaquim', 'Fernandes', 967045322),  
(100000010, 'Filipa', 'Costa', 966841841),  
(100000011, 'João', 'Pedro', 937004002);
```

```
INSERT INTO CPs (CP, Localidade)
```

```
VALUES
```

```
('2840-167', 'Seixal'),  
( '4820-392', 'Fafe'),  
( '5000-081', 'Vila Real'),  
( '4000-011', 'Porto');
```

```
INSERT INTO Pessoas (ID, NIF, Morada, CP)
```

```
VALUES
```

```
(1001, 100000001, 'Rua Manuel 2', '5000-081'),  
(1002, 100000002, 'Rua Azevedo Pinto 23', '4820-392'),  
(1003, 100000003, 'Rua Capitao Marques 74 Andar 1', '2840-167'),  
(1004, 100000004, 'Rua Manuel 8', '5000-081'),  
(1005, 100000005, 'Rua Azevedo Pinto 9', '4820-392'),  
(1006, 100000006, 'Rua Capitao Marques 74 Andar 2', '2840-167'),  
(1007, 100000007, 'Rua Manuel 5', '5000-081'),  
(1008, 100000008, 'Rua Azevedo Pinto 49', '4820-392'),  
(1009, 100000009, 'Rua Capitao Marques 74 R/C', '2840-167'),
```

```
(1010, 100000010, 'Travessa do Agro Bom', '4820-392'),  
(1011, 100000011, 'Rua Fernandes Tomás', '4000-011');
```

```
INSERT INTO Pacientes(ID_Pac)
```

```
VALUES
```

```
(1001), (1002), (1003), (1004), (1005);
```

```
INSERT INTO Alergias(ID_Alerg, Tipo)
```

```
VALUES
```

```
(1, 'Alergia aos pólenes'),  
(2, 'Alergia aos ácaros'),  
(3, 'Alergia alimentar'),  
(4, 'Alergia a medicamento');
```

```
INSERT INTO Paciente_Alergia(ID_Pac, ID_Alerg)
```

```
VALUES
```

```
(1001, 3),  
(1002, 4),  
(1003, 2);
```

```
INSERT INTO Funcionarios(ID_Func, Salario)
```

```
VALUES
```

```
(1004, 1000),  
(1005, 900),  
(1006, 650),  
(1007, 1200),  
(1008, 600),  
(1009, 800),  
(1010, 650),  
(1011, 700);
```

```
INSERT INTO Medicos(ID_Med, Especialidade)
```

```
VALUES
```

```
(1004, 'Cardiologia'),  
(1009, 'Anestesiologia'),  
(1007, 'Anestesiologia'),
```

```

(1005, 'Neurologia');

INSERT INTO Enfermeiros(ID_Enf, Turno, Horas_Extra)
VALUES
(1006, 'Manhã', 0),
(1008, 'Tarde', 10),
(1010, 'Noite', 3);

INSERT INTO Auxiliares(ID_Aux, Antiguidade, Servico)
VALUES
(1006, 5, 'Ortopedia'),
(1008, 10, 'Geral'),
(1011, 25, 'Urgências');

INSERT INTO Descricoes(ID_Pac, Data_Inq, Descricao)
VALUES
(1001, '2020-01-21 09:00', NULL),
(1002, '2020-07-05 11:00', 'O paciente indicou alergia a
paracetamol'),
(1003, '2020-12-21 15:00', 'O paciente indicou alergia a ácaros');

INSERT INTO Inquerito(ID_Pac, Data_Inq, ID_Func)
VALUES
(1001, '2020-01-21 09:00', 1006),
(1002, '2020-07-05 11:00', 1008),
(1003, '2020-12-21 15:00', 1006);

INSERT INTO Info_Op(Data_Op, Duracao)
VALUES
('2021-04-23 08:30', 5),
('2021-04-15 10:30', 14),
('2021-05-18 14:30', NULL),
('2020-09-11 16:00', 9);

INSERT INTO Operar(ID_Op, ID_Med, ID_Enf, ID_Pac)

```

VALUES

```
(1, 1004, 1006, 1002),  
(2, 1005, 1006, 1004),  
(3, 1004, 1008, 1002),  
(4, 1005, 1010, 1002);
```

INSERT INTO Local_Op(ID_Op, ID_Med, ID_Enf, ID_Pac, Data_Op, Local_Op)

VALUES

```
(1, 1004, 1006, 1002, '2021-04-23 08:30', 'Bloco B'),  
(2, 1005, 1006, 1004, '2021-04-15 10:30', 'Bloco A'),  
(3, 1004, 1008, 1002, '2021-05-18 14:30', 'Bloco D'),  
(4, 1005, 1010, 1002, '2021-05-01 16:00', 'Bloco C');
```

INSERT INTO Agendar(ID_Op, ID_Med, ID_Enf, ID_Pac, Data_Op, ID_Aux,
Data_Agend)

VALUES

```
(1, 1004, 1006, 1002, '2021-04-23 08:30', 1006, '2020-12-21 14:00'),  
(2, 1005, 1006, 1004, '2021-04-15 10:30', 1008, '2020-01-21 12:00'),  
(3, 1004, 1008, 1002, '2021-05-18 14:30', 1011, '2021-03-05 09:00'),  
(4, 1005, 1010, 1002, '2021-05-01 16:00', 1006, '2021-04-24 09:30');
```

INSERT INTO Preço_Pag(ID_Op, ID_Med, ID_Enf, ID_Pac, Preço)

VALUES

```
(1, 1004, 1006, 1002, 400),  
(2, 1005, 1006, 1004, 1000),  
(4, 1005, 1010, 1002, 800);
```

INSERT INTO Pagar(ID_Op, ID_Med, ID_Enf, ID_Pac, ID_Paciente, ID_Aux,
Data_Pag)

VALUES

```
(1, 1004, 1006, 1002, 1002, 1008, '2021-04-30 14:00'),  
(2, 1005, 1006, 1004, 1004, 1008, '2021-04-20 09:00'),  
(4, 1005, 1010, 1002, 1002, 1011, DEFAULT);
```


4.2 QUERIES DE SELEÇÃO DE DADOS

```
-- 2.1 Qual o último inquérito realizado? [Paciente (Nome), Data,
Funcionário (Nome)]
```

```
SELECT N1.Nome Paciente, Data_Inq 'Data', N2.Nome Funcionário
FROM NIFs N1, NIFs N2, Pessoas P1, Pessoas P2, Pacientes,
Funcionarios, Inquerito
WHERE Data_Inq = (SELECT MAX(Data_Inq) FROM Inquerito)
AND Inquerito.ID_Pac = Pacientes.ID_Pac
AND Pacientes.ID_Pac = P1.ID
AND P1.NIF = N1.NIF
AND Inquerito.ID_Func = Funcionarios.ID_Func
AND Funcionarios.ID_Func = P2.ID
AND P2.NIF = N2.NIF;
```

```
-- 2.2. Quantos médicos existem de cada especialidade? [Especialidade,
N_Médicos]
```

```
SELECT Especialidade, COUNT(ID_Med) N_Médicos
FROM Medicos
GROUP BY Especialidade;
```

```
-- 2.3. Quais os dois enfermeiros que mais operações assistiram?
[Enfermeiro (Nome), N_Operações]
```

```
SELECT NIFs.Nome Enfermeiro, SQ1.N_Operações
FROM (SELECT TOP 2 ID_Enf, COUNT(Operar.ID_Op) N_Operações
FROM Operar
GROUP BY ID_Enf
ORDER BY N_Operações DESC) SQ1,
(SELECT ID_Enf, COUNT(Operar.ID_Op) N_Operações
FROM Operar
GROUP BY ID_Enf) SQ2,
Enfermeiros, Funcionarios, Pessoas, NIFs
WHERE NIFs.NIF = Pessoas.NIF
```

```

AND Pessoas.ID = Funcionarios.ID_Func
AND Funcionarios.ID_Func = Enfermeiros.ID_Enf
AND Enfermeiros.ID_Enf = SQ2.ID_Enf
AND SQ2.N_Operações = SQ1.N_Operações;

```

-- 2.4. Quais os pacientes que realizaram mais de 2 operações nos últimos 30 dias? Ordene-os alfabeticamente. [Pacientes (Nome), N_Operações]

```

SELECT NIFs.Nome Paciente, SQ1.N_Operações
FROM (SELECT Operar.ID_Pac, COUNT(Operar.ID_Op) N_Operações
      FROM Operar, Info_Op
      WHERE Operar.ID_Op = Info_Op.ID_Op
      AND DATEDIFF(DAY, Info_Op.Data_Op, GETDATE()) <= 30
      GROUP BY Operar.ID_Pac) SQ1,
      Pacientes, Pessoas, NIFs
WHERE SQ1.N_Operações > 2
AND SQ1.ID_Pac = Pacientes.ID_Pac
AND Pacientes.ID_Pac = Pessoas.ID
AND Pessoas.NIF = NIFs.NIF
ORDER BY Paciente;

```

-- 2.5. Quais os enfermeiros que também fazem de auxiliares? [Enfermeiros (nome)]

```

SELECT ID_Enf ID, NIFs.Nome Enfermeiro
FROM NIFs,
      (SELECT ID ID_Enf, NIF
      FROM Enfermeiros E, Funcionarios F, Pessoas P
      WHERE E.ID_Enf = F.ID_Func
      AND F.ID_Func = P.ID) Enfs,
      (SELECT ID ID_Aux
      FROM Auxiliares A, Funcionarios F, Pessoas P
      WHERE A.ID_Aux = F.ID_Func
      AND F.ID_Func = P.ID) Auxs
WHERE ID_Enf = ID_Aux

```

```
AND Enfs.NIF = NIFs.NIF;
```

```
-- 2.6. Quais os pacientes que pagaram mais de 500€ em operações?  
[Pacientes(nome e apelido), Valor_Gasto]
```

```
SELECT CONCAT(NIFs.Nome, ' ', NIFs.Apelido) Paciente, Valor_Gasto  
FROM (SELECT ID_Paciente ID, SUM(Preco) Valor_Gasto
```

```
FROM Pagar, Preco_Pag
```

```
WHERE Pagar.ID_Op = Preco_Pag.ID_Op
```

```
AND Pagar.ID_Med = Preco_Pag.ID_Med
```

```
AND Pagar.ID_Enf = Preco_Pag.ID_Enf
```

```
AND Pagar.ID_Pac = Preco_Pag.ID_Pac
```

```
GROUP BY ID_Paciente) SQ1,
```

```
Pacientes, Pessoas, NIFs
```

```
WHERE SQ1.Valor_Gasto > 500
```

```
AND SQ1.ID = Pacientes.ID_Pac
```

```
AND Pacientes.ID_Pac = Pessoas.ID
```

```
AND Pessoas.NIF = NIFs.NIF;
```

```
-- 2.7. Qual o total de inquéritos realizado por cada funcionário no  
ano de 2020?[Funcionários (Nome), Total_Inquéritos]
```

```
SELECT NIFs.Nome Funcionário, SQ2.Total_Inquéritos
```

```
FROM (SELECT F.ID_Func, COUNT(SQ1.ID_Func) Total_Inquéritos
```

```
FROM Funcionarios F
```

```
LEFT JOIN (SELECT ID_Func
```

```
FROM Inquerito, Descricoes
```

```
WHERE Inquerito.ID_Pac = Descricoes.ID_Pac
```

```
AND Inquerito.Data_Inq =
```

```
Descricoes.Data_Inq
```

```
AND YEAR(Descricoes.Data_Inq) = 2020) SQ1
```

```
ON F.ID_Func = SQ1.ID_Func
```

```
GROUP BY F.ID_Func) SQ2,
```

```
Pessoas, NIFs
```

```
WHERE SQ2.ID_Func = Pessoas.ID
```

```
AND Pessoas.NIF = NIFs.NIF;
```

5 CONCLUSÃO

Face ao trabalho desenvolvido, crê-se ter atingido os objetivos definidos para esta etapa, nomeadamente a inserção dos dados pedidos bem como resposta à listagem de perguntas definida.

De referir que relativamente à pergunta 2.3, esta não prevê situações em que “existe mais do que um enfermeiro com o mesmo número de operações” pelo que se tomou a liberdade de adaptar o pedido.

6 BIBLIOGRAFIA

Insert (SQL). (19 de maio de 2021). Obtido de Wikipédia, a enciclopédia livre: [https://pt.wikipedia.org/wiki/Insert_\(SQL\)](https://pt.wikipedia.org/wiki/Insert_(SQL))

Kroenke, D. M., & Auer, D. J. (2016). *Database Processing: Fundamentals, Design, and Implementation*. Edinburgh Gate, Harlow, Essex CM20 2JE, England: Pearson Education.

Microsoft Corporation. (19 de maio de 2021). *Microsoft SQL documentation - SQL Server*. Obtido de Microsoft Docs: <https://docs.microsoft.com/en-us/sql/?view=sql-server-ver15>

Select (SQL). (19 de maio de 2021). Obtido de Wikipédia, a enciclopédia livre: [https://pt.wikipedia.org/wiki/Select_\(SQL\)](https://pt.wikipedia.org/wiki/Select_(SQL))

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). *Database System Concepts*. 2 Penn Plaza, New York, NY 10121: McGraw-Hill Education.