Banco de Dados

Aulas Práticas - Laboratório 3

Profª Cristina Verçosa Pérez Barrios de Souza
cristina.souza@pucpr.br

Tópicos

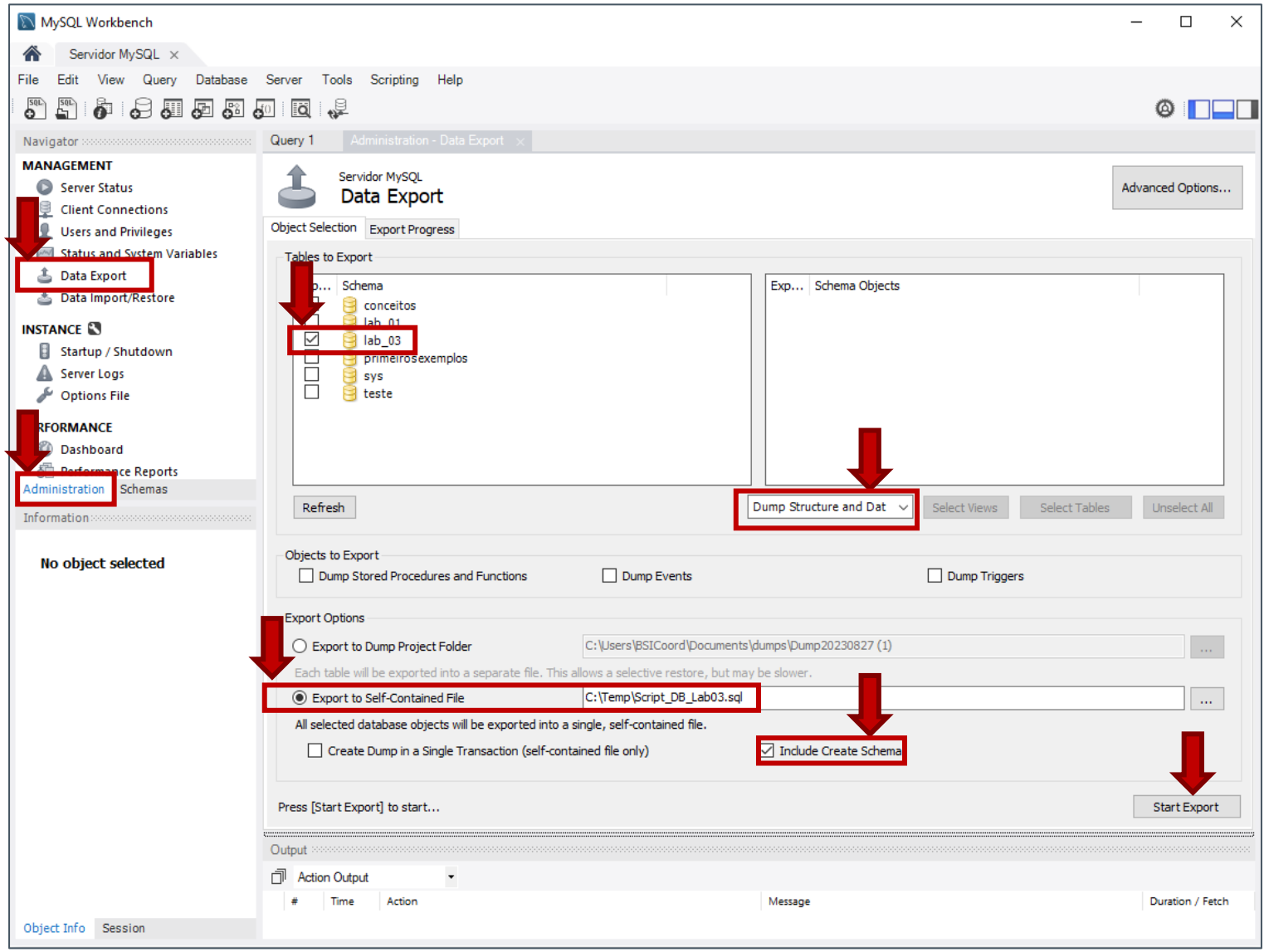
- Select
- Associações
- Subconsultas
- Subconsultas Correlacionadas
- Visualizações
- Funções de Agregação

SELECT

- Instrução **Select**
  - Comando SQL mais usado
  - Maneira fundamental de se recuperar dados

Prática 3.1: Criação de Database

- **Criação do Database LAB_03 completo**
  - Executar o script do material de apoio, que irá criar e povoar a database: **ScriptDB_Lab03.sql**
  - Observação: este **script SQL** completo do BD (esquema e dados) foi gerado com o **MySQL Workbench**:



Prática 3.2: Consulta com cálculo

› Execute o código abaixo:

RESPONDER:

1. O que faz a função **YEAR()**?
2. O que faz a função **MONTH()**?
3. O que faz a função **DAY()**?
4. O que faz a função **NOW()**?
5. Qual a sintaxe para o comando **CASE ...**?
6. Escreva o comando **SQL** para calcular a **sua idade em anos**, cuidando para o SQL verificar se já fez aniversário ou não.

(a)

```
USE LAB_03;

SELECT
        nome,               -- campo / coluna da tabela
        dt_nascimento,      -- campo / coluna da tabela
        DATE_FORMAT(dt_nascimento, '%d/%d/%Y') AS 'Aniversário',  -- formata data em dia/mês/ano com 4 dígitos
        (
        YEAR(NOW()) - YEAR(dt_nascimento) - -- vai SUBTRAIR de 0 ou 1, dependendo se já fez aniversário ou não
        CASE
            WHEN (MONTH(NOW()) * 100 + DAY(NOW())) > (MONTH(dt_nascimento) * 100 + DAY(dt_nascimento))
            THEN 0   -- Valor de retorno para ser subtraído = 0
            ELSE 1   -- Valor de retorno para ser subtraído = 1
        END
    ) AS Idade              -- AS é a indicação de apelido de exibição para a coluna recém calculada
FROM Empregado;
```

CÁLCULO DA IDADE com SUBTRAÇÕES:
Idade = Ano(dataAtual) – **Ano**(dataNascimento) –
Se ((mês * 100 + dia da data atual) **>** (mês * 100 + dia da data de aniversário) **)**
Então 0 -- diminui **zero** pois já fez aniversário, logo o cálculo já está correto
Senão 1 -- diminui **um** pois ainda não fez aniversário

ASSOCIAÇÕES

- Associações - ou **JOINS**
  - Retorna uma tabela virtual, que associa tabelas
  - Traz apenas as linhas das tabelas onde há **correspondência** entre as chave primária e chave estrangeira das tabelas envolvidas
- Tipos
  - **JOIN** (também chamado de **INNER JOIN**)
    - Retorna apenas as linhas das respectivas tabelas que satisfaziam a condição de igualdade
  - **OUTER JOIN**
    - preservam algumas (ou todas) as linhas que não satisfazem a condição de igualdade
      - FULL OUTER JOIN: preserva os dados das tabelas à esquerda e à direita do JOIN (**apenas no MS SQL Serever; MySQL não tem este comando**)
      - LEFT OUTER JOIN: preserva os dados da tabela à esquerda do JOIN
      - RIGHT OUTER JOIN: preserva os dados da tabela à direita do JOIN

Prática 3.3: Associação JOIN ou INNER JOIN

- Execute no Database **LAB_03**:

(a)

```
SELECT *
FROM Empregado AS E, Departamento AS D  -- PRODUTO CARTESIANO
WHERE E.ID_depto = D.ID_depto           -- Condição para retorno
ORDER BY E.nome;


SELECT *
FROM Empregado AS E INNER JOIN Departamento AS D -- JOIN ou INNER JOIN
ON (E.ID_depto = D.ID_depto)        -- Condição para retorno
ORDER BY E.nome;
```

A instrução **JOIN** produz o mesmo resultado que **PRODUTO CARTESIANO**.

É possível usar **INNER JOIN** (associação interna) no lugar de **JOIN**, basicamente para diferenciá-la de instrução **OUTER JOIN**.

RESPONDER:

1. Houve diferença no resultado dos comandos de **SELECT** do exercício?
2. Substitua no **2º SELECT** o "**INNER JOIN**" por apenas "**JOIN**". Qual a diferença entre esses dois comandos?

Prática 3.3: Associação JOIN ou INNER JOIN

- Execute no Database **LAB_03**:

RESPONDER:
1. Quantas e quais tabelas estão envolvidas nas consultas?
2. É possível retirar alguma das tabelas indicadas nos comandos e obter o mesmo resultado? Qual tabela podemos retirar?

(b)

```
SELECT E.nome AS Empregado, ES.nivel, S.nome
FROM   Empregado AS E, EmpSkill AS ES, Skill AS S  -- PRODUTO CARTESIANO
WHERE  E.ID_emp   = ES.ID_emp    AND               -- Condição para retorno
       S.ID_skill = ES.ID_skill
ORDER BY S.nome, E.nome;


SELECT E.nome AS Empregado, ES.nivel, S.nome       -- JOIN
FROM
  (  -- Primeiro, faz JOIN entre as tabelas Empregado e EmpSkill
     Empregado AS E INNER JOIN EmpSkill AS ES ON (E.ID_emp = ES.ID_emp)
     -- Depois, associa o resultado anterior com a tabela Skill
  )  INNER JOIN Skill AS S ON (S.ID_skill = ES.ID_skill)
ORDER BY S.nome, E.nome;
```

Prática 3.4: Associação LEFT OUTER JOIN

- Execute no Database **LAB_03**:

(a)

```
SELECT E.nome AS Empregado, ES.nivel, S.nome
FROM
  (  -- Primeiro, associa (LEFT OUTER JOIN) Empregado e EmpSkill
     Empregado AS E LEFT OUTER JOIN EmpSkill AS ES ON (E.ID_emp = ES.ID_emp)
     -- Depois, associa (LEFT OUTER JOIN) o resultado com Skill
  )  LEFT OUTER JOIN Skill AS S ON (S.ID_skill = ES.ID_skill)
ORDER BY S.nome, E.nome;
```

A instrução **LEFT OUTER JOIN** retorna as linhas, da **tabela à ESQUERDA** que não atendem à condição da associação / JOIN.

RESPONDER:

1. Para o **LEFT OUTER JOIN**, qual a **tabela à esquerda** do comando? E qual a **tabela à direita** do comando?
2. Quais dados foram apresentados, mesmo sem correspondência?

Prática 3.5: Associação RIGHT OUTER JOIN

- Execute no Database **LAB_03**:

(b)

```
SELECT E.nome AS Empregado, ES.nivel, S.nome
FROM
  (  -- Primeiro, associa (RIGHT OUTER JOIN) Empregado e EmpSkill
     Empregado AS E RIGHT OUTER JOIN EmpSkill AS ES ON (E.ID_emp = ES.ID_emp)
     -- Depois, associa (RIGHT OUTER JOIN) o resultado com Skill
  )RIGHT OUTER JOIN Skill AS S ON (S.ID_skill = ES.ID_skill)
ORDER BY S.nome, E.nome;
```

A instrução **RIGHT OUTER JOIN** retorna as linhas, da **tabela à direita** que não atendem à condição da associação / JOIN.

RESPONDER:

1. Para o **RIGHT OUTER JOIN**, qual a **tabela à direita** do comando? E qual a **tabela à esquerda** do comando?
2. Quais dados foram apresentados, mesmo sem correspondência?

A instrução **FULL OUTER JOIN** retorna **TODAS** as linhas, mesmo aquelas que não atendem à condição da associação / JOIN. Ela não existe no **MySQL** (deferente do **MS SQL Server**) Portanto, devemos **unir** os resultados de **LEFT OUTER** e **RIGHT OUTER**

Prática 3.5: Associação FULL OUTER JOIN

- Execute no Database **LAB_03**:

(a)

```
(
SELECT E.nome AS Empregado, ES.nivel, S.nome
FROM
  (  -- Primeiro, associa (LEFT OUTER JOIN) Empregado e EmpSkill
     Empregado AS E LEFT OUTER JOIN EmpSkill AS ES ON (E.ID_emp = ES.ID_emp)
     -- Depois, associa (LEFT OUTER JOIN) o resultado com Skill
  )  LEFT OUTER JOIN Skill AS S ON (S.ID_skill = ES.ID_skill)
)
UNION
(
SELECT E.nome AS Empregado, ES.nivel, S.nome
FROM
  (  -- Primeiro, associa (RIGHT OUTER JOIN) Empregado e EmpSkill
     Empregado AS E RIGHT OUTER JOIN EmpSkill AS ES ON (E.ID_emp = ES.ID_emp)
     -- Depois, associa (RIGHT OUTER JOIN) o resultado com Skill
  )RIGHT OUTER JOIN Skill AS S ON (S.ID_skill = ES.ID_skill)
)
ORDER BY nivel;
```

RESPONDER:

1. Quantos registros são retornados no **SELECT** do **LEFT OUTER JOIN**?
2. Quantos registros são retornados no **SELECT** do **RIGHT OUTER JOIN**?
3. Pesquise quais as condições para o coando **UNION** ser realizado. Dê um exemplo de uma utilização incorreta do **UNION** e sua respectiva correção explicada.
4. A operação de **UNION** do exercícios retornou quantos registros no total? Esse valor corresponde à soma dos resultados do **LEFT** e do **OUTER JOIN**? Sim ou não? Por que?

SUBCONSULTAS

- **Aninhamento de Consultas**
  - Uma **subconsulta** é uma consulta que está aninhada dentro de uma instrução SELECT, INSERT, UPDATE ou DELETE ou em outra subconsulta.
  - É uma forma natural e eficiente de expressar critérios de cláusula WHERE em termos de resultados de outras consultas
  - Uma boa parte das associações (JOINs) pode ser expressa como uma subconsulta

Prática 3.6: Associação x Subconsulta

› Em sua database de trabalho, execute:

```sql
SELECT E.ID_emp, E.nome
FROM   Empregado AS E JOIN Departamento AS D
ON     (E.ID_depto = D.ID_depto)
WHERE  D.sigla = 'CTB' OR D.sigla ='VND'
ORDER BY E.nome;
```

(a)

```sql
SELECT E.ID_emp, E.nome
FROM   Empregado AS E
WHERE  E.ID_depto IN
(
     SELECT D.ID_depto
     FROM Departamento AS D
     WHERE D.sigla = 'CTB' OR D.sigla ='VND'
)
ORDER BY E.nome;
```

RESPONDER:

1. Qual a diferença entre os comando de **SELECT** passados?
2. Qual comando possui uma subconsulta?
3. Como funciona o comando **WHERE ... IN**?

Prática 3.6: Associação x Subconsulta

› Em sua database de trabalho, execute:

(b)

```
SELECT E.ID_depto, E.ID_emp, E.nome
FROM   Empregado AS E
WHERE  E.ID_depto NOT IN
(
     SELECT D.ID_depto
     FROM Departamento AS D
     WHERE D.sigla = 'CTB' OR D.sigla ='VND'
)
ORDER BY E.nome;



SELECT E.ID_depto, E.ID_emp, E.nome
FROM   Empregado AS E
WHERE  E.ID_depto <> ALL
(
     SELECT D.ID_depto
     FROM Departamento AS D
     WHERE D.sigla = 'CTB' OR D.sigla ='VND'
)
ORDER BY E.nome;
```

Experimente utilizar a expressão **= ANY** na subconsulta.

RESPONDER:

1. Qual a diferença entre os comando passados?
2. Como funciona o comando **WHERE ... <> ALL**?

SUBCONSULTAS CORRELACIONADAS

- **Utilização**
  - Comparar linhas em uma tabela com uma condição em uma tabela correspondente
  - Para cada linha que atende à consulta principal, uma nova subconsulta é executada e avaliada

Prática 3.7: Associação x Subconsulta

› Em sua database de trabalho, execute:

Experimente utilizar a expressão **= ANY** na subconsulta.

(a)

```
SELECT E.ID_depto, E.ID_emp, E.nome, E.dt_nascimento
FROM   Empregado AS E
WHERE  YEAR(E.dt_nascimento) >= 1998 AND
   E.ID_depto IN
(
     SELECT D.ID_depto
     FROM Departamento AS D, Empregado AS E1
     WHERE D.ID_depto = E1.ID_depto AND
          (D.sigla = 'CTB' OR D.sigla ='VND')
)
ORDER BY E.nome;
```

RESPONDER:

1. Como funcionou o comando **WHERE ... = ANY** nesta consulta?
2. Qual a diferença entre **WHERE ... = ANY** e **WHERE ... IN**.

VISUALIZAÇÕES

- Ou **VIEWs**
  - A visualização pode ser encarada como uma tabela virtual
  - De modo geral, é uma instrução SELECT nomeada que produz dinamicamente uma nova tabela virtual, ou um conjunto de resultados, que poderá ser utilizado posteriormente
  - Encobre a complexidade de certas consultas

Prática 3.8: View

RESPONDER:
1. Por que é interessante criar uma **VIEW**?
2. O que fica persistido no **BD** quando criamos uma **VIEW**?

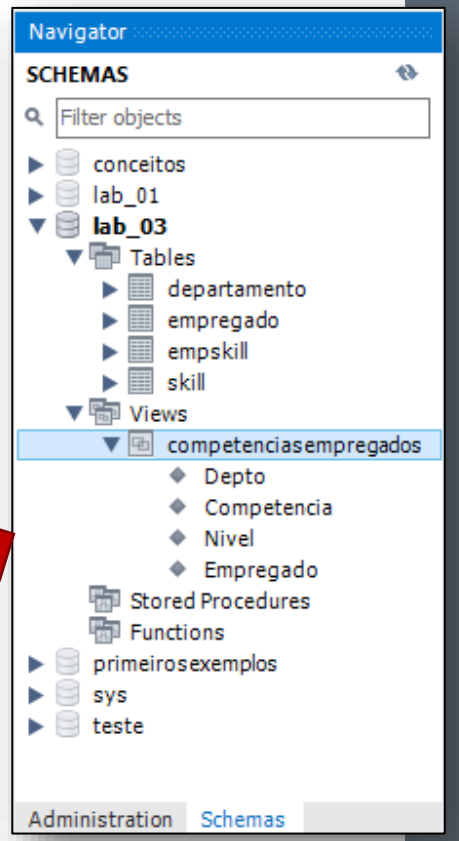
› Em sua database de trabalho, execute:

(a)

```
-- Apaga a VIEW se ela já existir
DROP VIEW IF EXISTS CompetenciasEmpregados;

CREATE VIEW CompetenciasEmpregados AS
(
     SELECT D.sigla AS Depto, S.nome AS Competencia, ES.nivel AS Nivel, E.nome AS Empregado
     FROM
       ((
         Empregado AS E INNER JOIN EmpSkill AS ES ON (E.ID_emp = ES.ID_emp)
       ) INNER JOIN Skill AS S ON (S.ID_skill = ES.ID_skill)
       ) INNER JOIN Departamento AS D ON (D.ID_depto = E.ID_depto)
);
```

No **MySQL Workbench** ...



Prática 3.8: View

› Em sua database de trabalho, execute:

(b)

```
SELECT *
FROM     CompetenciasEmpregados
ORDER BY Depto, Competência, Empregado;
```

RESPONDER:

1. Pesquise uma ou mais **vantagens** em se usar uma **VIEW**.
2. Pesquise uma ou mais **desvantagens** em se usar uma **VIEW**

Em vez de reformular a consulta com **duplo INNER JOIN** (vista no slide anterior), realizamos uma consultada à visualização (**VIEW**) já criada.

A visualização (**VIEW**) é muito útil quando precisamos de uma consulta complexa, recorrente e necessária em várias outras consultas.

FUNÇÕES DE AGREGAÇÃO

- Uma função de agregação executa um cálculo em um conjunto de valores e retorna um único valor.
- Com exceção de COUNT(*), as funções de agregação ignoram valores NULOS (NULL).
- As funções de agregação frequentemente são usadas com a cláusula GROUP BY da instrução SELECT.

SQL – Sintaxe Resumida

› Funções de Agregação, para o SELECT

SUM (N)	(retorna a soma dos valores em um grupo)
AVG (N)	(retorna a média dos valores em um grupo)
MIN (EXP)	(retorna o menor dos valores em um grupo)
MAX (EXP)	(retorna o maior dos valores em um grupo)
COUNT (EXP)	(retorna o total dos valores em um grupo)

Prática 3.9: Agregação

RESPONDER:

1. Qual a diferenças entre os comandos?

2. Descreva cada um dos resultados obtidos no segundo comando de **SELECT**, explicando os comandos de **AGREGAÇÃO** executados.

› Em sua database de trabalho, execute:

(a)

```
SELECT
COUNT(*)     AS 'Número de Empregados',
AVG(salario) AS 'Salário Médio',
MIN(salario) AS 'Menor Salário' ,
MAX(salario) AS 'Maior Salário' ,
SUM(salario) AS 'Total Salários'
FROM Empregado;
```

```
SELECT
COUNT(*)                              AS 'Número de Empregados',
CONVERT(AVG(salario), DECIMAL(8,2))   AS 'Salário Médio',
MIN(salario)                          AS 'Menor Salário' ,
MAX(salario)                          AS 'Maior Salário' ,
SUM(salario)                          AS 'Total Salários'
FROM Empregado;
```

Referência Bibliográfica

- Sistema de Banco de Dados
  - Abraham Silberschatz, Henry F. Korth, S. Sudaarshan

- Referência do SQL
  - Chapter 13 SQL Statements: https://dev.mysql.com/doc/refman/8.0/en/sql-statements.html
  - W3Schools: https://www.w3schools.com/mysql/mysql_drop_db.asp

- Documentação Técnica do MySQL
  - MySQL 8.0 Reference Manual
  - https://dev.mysql.com/doc/refman/8.0/en/