BANCO DE DADOS
DML

ANTONIO DAVID VINISKI
antonio.david@pucpr.br
PUCPR

AGENDA
MODELAGEM LÓGICA DE BANCO DE DADOS

- MySQL – DML
  - Operadores Aritméticos, Relacionais, Lógicos, auxiliares.
  - ORDER BY.
  - Verificação de caracteres.
  - Funções Agregadas / Informações agrupadas.
  - Qualificadores.

EXERCÍCIOS BANCO DE DADOS DO RESTAURANTE

EXERCÍCIOS

1. Inserir um registro de produto nas comandas 5, 6, 10 e 12.
    a) Escolha um produto da tabela produtos, verifique o valor do produto para ser inserido no registro, escolha um garçom para vincular ao registro.
2. Atualizar o valor do produto 8 para R$ 12,00.
3. Atualizar a quantidade do produto 2, registrado na comanda 2, pelo garçom com 5, para ser igual a 3.
4. Excluir o registro do produto 19 na comanda 2.
5. Criar a tabela Ingredientes, a qual está ligada ao produto por meio de uma relação muitos para muitos.
6. Inserir os ingredientes necessários para preparação do produto 4 ("Caipirinha de Limão") e vincular esses ingredientes.

OPERADORES ARITMÉTICOS

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- São responsáveis pela execução de operações matemáticas simples:

OPERADORES	
+	Adição
-	Subtração
*	Multiplicação
/	Divisão

```
SELECT quantidade * valor_produto
FROM restaurante.registro
WHERE comanda_id = 1;

SELECT 3 * 6;
```

Selecione a média dos produtos consumidos na comanda 1

OPERADORES RELACIONAIS

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- São utilizados quando precisamos fazer comparações entre dois valores:

OPERADORES	
>	Maior que
<	Menor que
=	Igual a
<>	Diferente de
>=	Maior ou igual a
<=	Menor ou igual a

```
SELECT * FROM restaurante.comanda
WHERE data_criacao >= '2022-09-01';
```

```
SELECT * FROM restaurante.cliente
WHERE data_criacao = '2022-09-19';
```

OPERADORES LÓGICOS - AND

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O operador lógico AND, ou E, deve ser usado em uma pesquisa que se deseja filtrar por dois ou mais valores.
- O AND, verifica ambas as cláusulas da comparação, e só retorna algum valor se as duas tiverem uma resposta verdadeira.
- Observe o exemplo:

```
SELECT * FROM restaurante.comanda WHERE data_criacao >= '2022-09-01'
AND id_cliente = 1;
```

- Esta pesquisa listará todos os registros de comandas que contém valores de data_hora maiores ou iguais a '2022-09-01', do cliente cujo id é igual a 4;

OPERADORES LÓGICOS - OR

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O operador lógico OR, ou OU, deve ser usado em uma pesquisa que se deseja filtrar por dois ou mais valores.
- O OR, verifica ambas as cláusulas da comparação, e retorna valores se qualquer um dos membros obtiver resultado.
- Observe o exemplo:

```
SELECT * FROM restaurante.comanda WHERE (data_hora >= '2022-09-01' AND
data_hora <= '2022-09-19') OR data_hora = '2022-08-16' ;
```

- Esta pesquisa listará todos os registros de comandas que contém valores de data_hora maiores ou iguais a '2022-09-01 11:29:35' e menores ou iguais a '2022-09-19 14:55:18', ou cuja campo data seja igual a '2022-08-16 23:59:59' ;

OPERADORES LÓGICOS - NOT

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O operador lógico NOT, ou NÃO, realiza uma pesquisa, excluindo valores determinados do resultado.
- Observe o exemplo:

```
SELECT * FROM restaurante.comanda WHERE NOT (data_hora >= '2022-09-16 12:09:23');
```

- Esta pesquisa listará todos os registros de comandas contém valores diferentes de data_hora >= '2022-09-16 12:09:23';

OPERADORES LÓGICOS – ORDER BY

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- O operador lógico ORDER BY, ou ORDENAR POR, simplesmente lista os registros, colocando-os em ordem de acordo com o campo solicitado.
- Observe o exemplo:

```
SELECT * FROM restaurante.pessoa ORDER BY data_nasc;
```

- Esta pesquisa listará todos os registros de da tabela pessoa ordenados por data_nasc na ordem crescente;

OPERADORES AUXILIARES - BETWEEN

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- **BETWEEN**
  - Definição de intervalos de valores para a cláusula **WHERE**.
    - <expressão> [**NOT**] **BETWEEN** <mínimo> **AND** <máximo>.

```
SELECT * FROM restaurante.comanda
WHERE data_hora BETWEEN '2022-09-01' AND '2022-09-19';
```

```
SELECT * FROM restaurante.comanda
WHERE id BETWEEN 200 AND 456;
```

OPERADORES AUXILIARES - IN

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- **IN**
  - Em alguns casos não conseguimos definir um intervalo sequencial de valores, com isso podemos usar a cláusula IN para comparar uma lista de valores.
    - <expressão> [**NOT**] **IN** (valor1, valor2,..., valorN).

```
SELECT * FROM restaurante.comanda WHERE id IN (1,3,5,6,7);

SELECT * FROM restaurante.registro WHERE garcon_id NOT IN (3,10,26,43);
```

VERIFICAÇÃO DE CARACTERES

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- Para verificar sequência de caracteres dentro de um campo do tipo *string* (char, varchar, text), pode-se utilizar junto com a cláusula **WHERE** uma condição baseada no uso do operador **LIKE**.
- Exemplos:
  - 'A%' – começa com letra A.
  - '_A%' – segunda letra do nome A.
  - '%AN% - possui AN em qualquer posição.
  - 'A%S' – Começa com A e termina com S.

```
SELECT * FROM restaurante.pessoa WHERE nome LIKE 'M%';
```

VERIFICAÇÃO DE VAZIO

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- Uma ocorrência bastante útil é verificar a existência de campos que possuam valores em branco ou não.
- Para isso usa-se junto ao **WHERE** o operador **IS** [**NOT**] **NULL**.
- Exemplos:

SELECT * **FROM** restaurante.comanda **WHERE** pagamento **IS NOT NULL**;

CLÁUSULAS AUXILIARES – ORDER BY

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- ASC e DESC especificam o tipo de classificação e são, respectivamente, abreviações das palavras em inglês *ascending* e *descending*, ou seja, classificação crescente ou decrescente.
- Quando não especificamos nenhum , o padrão é ascendente
- Exemplo:

SELECT * **FROM** restaurante.pessoa **ORDER BY** data_nasc **DESC**, nome **ASC**;

EXERCÍCIOS

1. Selecionar as pessoas que nasceram depois de "1990-01-01", ordenadas por nome em ordem alfabética.
2. Selecionar as comandas cujo valor seja maior do que R$ 100,00, ordenadas do maior para o menor valor.
3. Selecionar as comandas registradas entre "2022-02-01" e "2022-09-02", ordenadas por data de criação.
4. Atualizar o valor do produto 10, aumentando 10%.
5. Atualizar o valor do produto 7, aumentando R$ 2,30.
6. Atualizar o valor dos produtos com valor maior que 40 reais, dando um desconto de 12%.
7. Inserir o pagamento referente ao valor da comanda 1;
8. Selecionar os registros que não estão ligados a nenhum pagamento de comissão.
9. Selecionar todas as pessoas cuja antepenúltima letra do nome é igual a "T".
10. Selecionar as pessoas que possuem as vogais "a", "e" e "o" no nome ou que tenham as vogais "i" e "u" .

FUNÇÕES AGREGADAS

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- **AVG**() – média aritmética.
- **MAX**() – Maior valor.
- **MIN**() - Menor valor.
- **SUM**() - Soma dos valores.
- **COUNT**() – Número de valores.
  - **ALL**- contagem dos valores não vazios.
  - **DISTINCT** – contagem dos valores não vazios e únicos.
- Exemplo:

```
SELECT AVG(valor _total) FROM restaurante.comanda;

SELECT SUM(valor _total) FROM restaurante.comanda;
```

INFORMAÇÕES AGRUPADAS

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- Obter a quantidade de comandas de cada cliente?
  - Para isso utilizamos a cláusula **GROUP BY**.
    - Agrupa os registros por valores únicos referentes aos campos agrupados.
  - Exemplos

```
SELECT id_cliente, COUNT(*) FROM restaurante.comanda GROUP BY id_cliente;
```

INFORMAÇÕES AGRUPADAS

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- Como ficaria a consulta do número de comandas e o valor médio das comandas de cada cliente?

- Exemplo:

```
SELECT id_cliente, COUNT(*), AVG(valor _total)  FROM restaurante.comanda
GROUP BY id_cliente;
```

- Observe que quando usamos **GROUP BY** a função **AVG**() retorna a média calculada para cada componente do grupo especificado, no caso “cliente_id".

INFORMAÇÕES AGRUPADAS

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- E se quiséssemos retornar somente quantidade de comandas e média de valores para os clientes que consumiram em média mais que R$ 200,00?
- Para isso utilizamos a cláusula **HAVING**
- Exemplo:

```
SELECT id_cliente, COUNT(*), AVG(valor _total)  FROM restaurante.comanda
GROUP BY id_cliente HAVING AVG(valor _total) > 200.0;
```

Note que a cláusula **HAVING** usada aqui tem o mesmo significado que **WHERE** nas consultas normais.

OBS: a cláusula "**WHERE**" não pode ser usada para restringir grupos que deverão ser exibidos.

QUALIFICADORES

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- **AS**
  - define um nome("alias") para uma coluna diferente do rótulo de coluna original.
- **ALL**
  - Especifica que a consulta deverá extrair todos os elementos indicados – é o padrão.
- **DISTINCT**
  - Faz com que o SQL ignore valores repetidos na tabela.

QUALIFICADORES - ALIAS

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- SQL permite dar um nome diferente de uma tabela, campo ou fórmula do nome real existente.
- Isso é conseguido com o comando AS utilizado junto com o **SELECT**.
- Quando pedimos para exibir o cliente_id, a quantidade de comandas (**COUNT**(*)) e a média do valor das comandas (**AVG**(valor_total)) para cada cliente. Podemos chamar as funções **COUNT**(*)) e **AVG**(valor _total) como "quantidade" e "Média Valor", respectivamente.

```
SELECT id_cliente, COUNT(*) AS "quantidade", AVG(valor) AS  "Média Valor"
FROM restaurante.comanda GROUP BY id_cliente HAVING AVG(valor_total) >
200.0;
```

QUALIFICADORES - DISTINCT

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- E se precisássemos identificar todos os valores registrados para um respectivo produto na tabela registro.

```
SELECT DISTINCT(valor) FROM restaurante.registro  WHERE produto_id = 2;
```

QUERIES ANINHADAS - SUBQUERY

LINGUAGEM DE MANIPULAÇÃO DE DADOS

- Uma *subquery* é um comando **SELECT** inserido em uma cláusula de um outro comando SQL.
- Pode-se desenvolver comandos sofisticados a partir de comandos simples, utilizando-se *subqueries*.
- Elas podem ser muito úteis quando for necessário selecionar linhas a partir de uma tabela com uma condição que dependa de dados da própria tabela.

QUERIES ANINHADAS - SUBQUERY

LINGUAGEM DE MANIPULAÇÃO DE DADOS

```
SELECT id_cliente,
        COUNT(*),
        SUM(valor)
FROM restaurante.comanda
WHERE restaurante.comanda.cliente_id IN (
                                        SELECT id FROM restaurante.cliente
                                        WHERE data_criacao > "2022-01-01")
GROUP BY id_cliente;
```

EXERCÍCIOS

1. Consultar valor máximo, mínimo e médio dos produtos cadastrados (Nomear os campos como Máximo, Mínimo e Média).
2. Consultar número de registros de cada um dos garçons.
3. Os valor total dos registros de cada garçom que efetuou mais do que 3 registros.
4. Consultar o valor total e o número de comandas já registradas.
5. Consultar as comandas que não possuem registro de produtos.
6. Consultar o id dos garçons que registraram mais do que R$ 120,00.
7. Consultar o valor total gasto e o número de comandas de cada um dos clientes.
8. Consultar os valores distintos de produtos no banco de dados.
9. Atualizar a quantidade disponível de cada um dos produtos diminuindo a quantidade dos produtos registrados.
10. Selecionar o valor médio das comandas e o valor médio dos produtos consumidos.