



Quem se prepara, não para.



# Banco de Dados

2º período

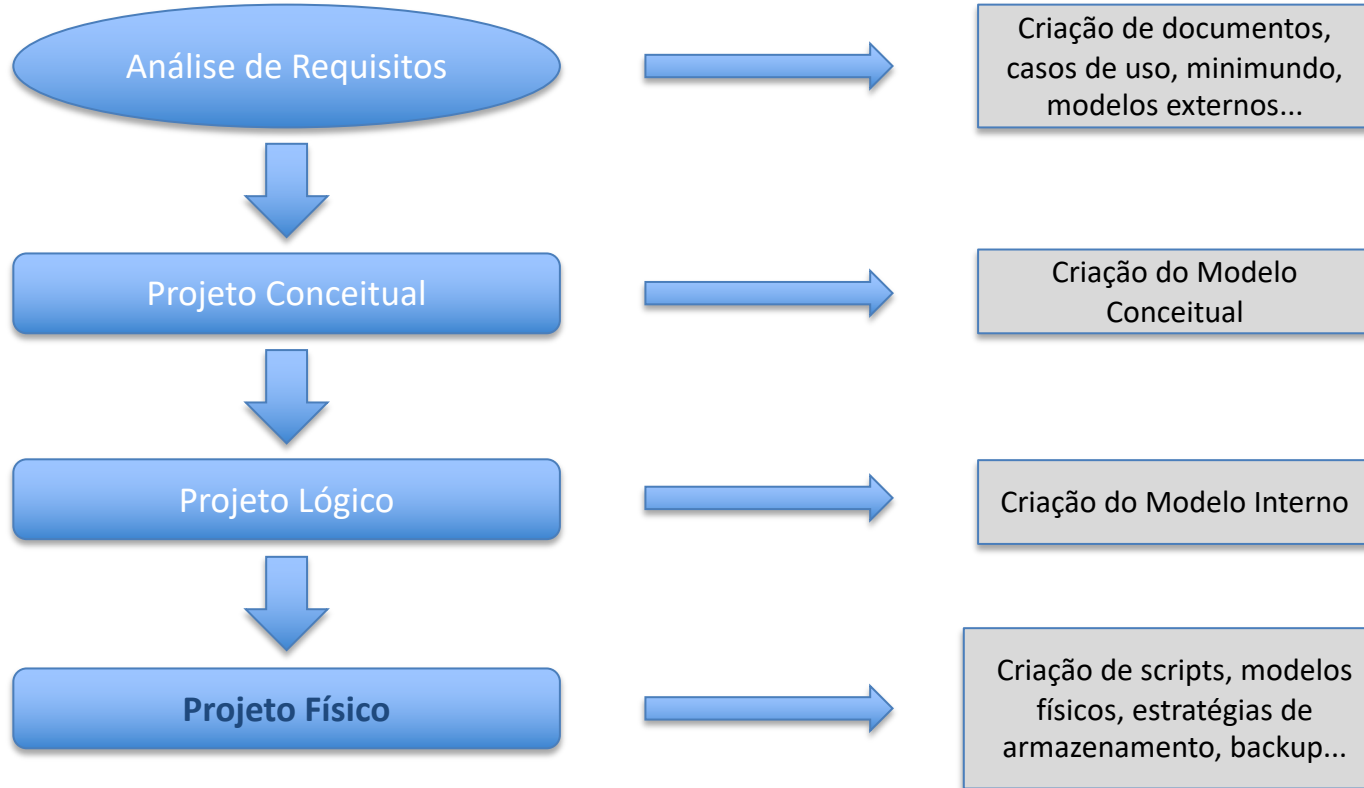
Prof. Dr. João Paulo Aramuni



# Projeto Físico de Bando de Dados

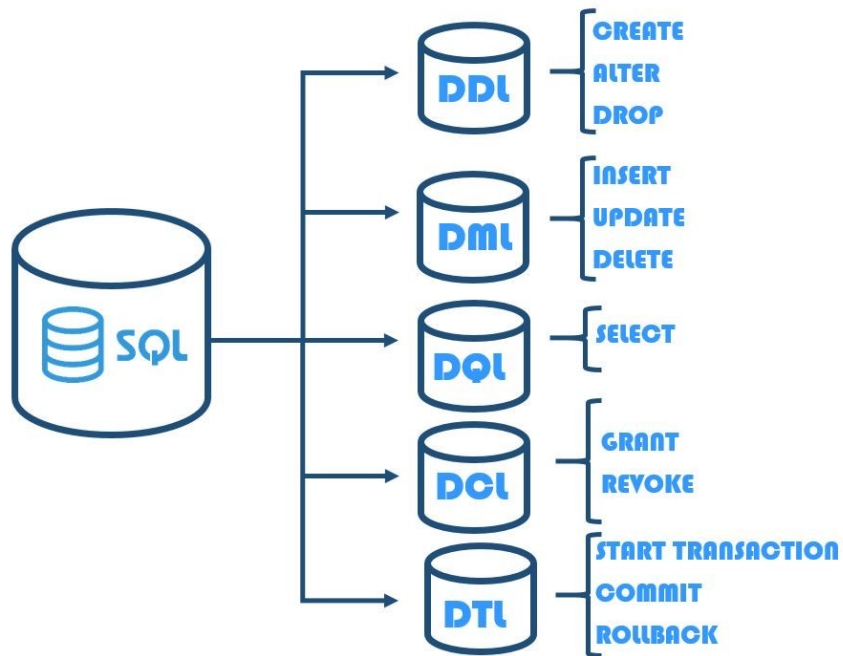
Aula 13

# Projeto Físico de BD



# Projeto Físico de BD

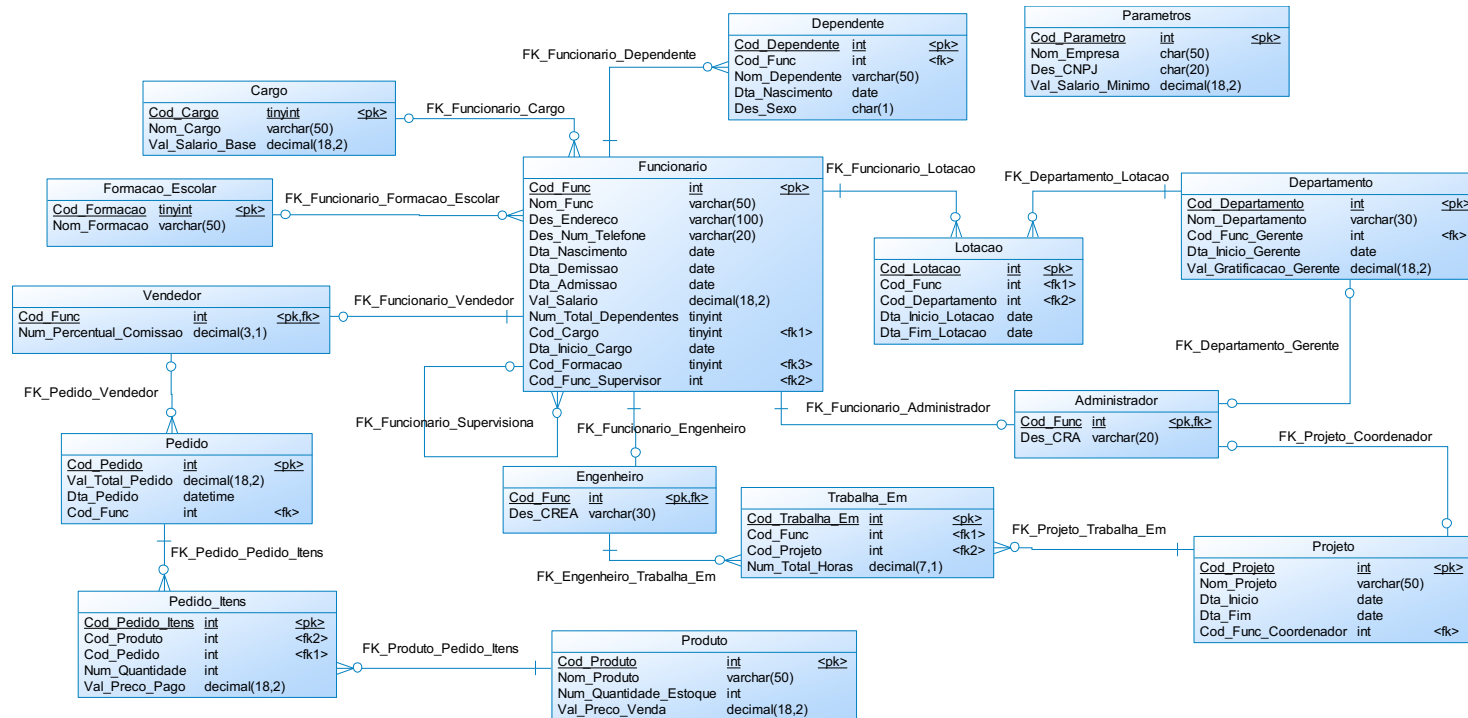
## Divisão da Linguagem SQL



# Projeto Físico de BD

Vamos trabalhar com o BD do DER abaixo.

Execute o script de criação do banco de dados BD\_RH.



# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

Revisão da última aula:

Mostre o nome dos projetos não encerrados e o nome dos respectivos coordenadores dos projetos, se existir.

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## **OPERADORES DE JUNÇÃO :**

INNER JOIN e LEFT OUTER JOIN juntos

Autorelacionamento

Tabelas duplicadas no FROM

## **FUNÇÕES AGREGADAS:**

COUNT, SUM, AVG, MAX, MIN

## **CLÁUSULAS:**

GROUP BY / HAVING



# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## OPERADORES DE JUNÇÃO :

INNER JOIN e LEFT OUTER JOIN juntos

Recupere o nome dos Projetos com pelo menos um funcionário trabalhando e o nome do coordenadores do projeto, se existir.

Recupere o nome dos departamentos com pelo menos um funcionário lotado nele e o nome dos gerentes do departamento. Mostre todos departamentos, mesmo que não tenha gerente definido. Ordene os dados pelo nome do departamento.

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## OPERADORES DE JUNÇÃO :

### Autorelacionamento

O **autorelacionamento**, também conhecido como **autojunção** ou relação **recursiva**, é um conceito em bancos de dados relacionais que ocorre quando uma tabela se relaciona consigo mesma por meio de uma chave estrangeira (foreign key).

Em outras palavras, é uma relação na qual uma coluna da tabela faz referência a outra coluna dentro da mesma tabela.

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## OPERADORES DE JUNÇÃO :

### Autorelacionamento

Esse tipo de relacionamento é utilizado quando existe uma relação hierárquica, de dependência ou subordinação entre os registros da mesma entidade representada pela tabela. O autorelacionamento é útil quando queremos modelar situações em que cada registro da tabela tem uma conexão ou relação com outros registros da mesma tabela.

Exemplo:

```
SELECT f1.Nome AS Funcionario, f2.Nome AS Supervisor FROM Funcionarios f1  
LEFT JOIN Funcionarios f2 ON f1.Supervisor = f2.ID;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## OPERADORES DE JUNÇÃO :

### Autorelacionamento

Recupere o nome dos funcionários não demitidos com os respectivos cargos e o nome dos respectivos supervisores caso tenha.

Recupere o nome do departamento, nome do gerente e o nome dos funcionários supervisores dos gerentes. Mostre todos os departamentos, mesmo que não tenham gerentes definidos.

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## OPERADORES DE JUNÇÃO :

Tabelas duplicadas no FROM

Tabelas duplicadas no FROM ocorrem quando você faz referência à mesma tabela mais de uma vez na cláusula FROM de uma consulta SQL. Você utiliza aliases (apelidos) para cada ocorrência da tabela para diferenciá-las, permitindo assim tratar cada referência da tabela como uma entidade distinta na consulta.

Essa técnica é útil quando você precisa realizar operações de junção entre diferentes partes da mesma tabela ou fazer comparações entre registros da mesma tabela.

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## OPERADORES DE JUNÇÃO :

Tabelas duplicadas no FROM

Exemplo:

```
SELECT f1.Nome AS Funcionario, f1.Cargo AS Cargo_Funcionario, f1.Salario AS  
Salario_Funcionario, f2.Nome AS Supervisor, f2.Cargo AS Cargo_Supervisor, f2.Salario AS  
Salario_Supervisor, f1.Salario - f2.Salario AS Diferenca_Salarial FROM Funcionarios f1  
LEFT JOIN Funcionarios f2 ON f1.SupervisorID = f2.ID;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## **FUNÇÕES AGREGADAS:**

**COUNT**, SUM, AVG, MAX, MIN

**COUNT:** A função COUNT é usada para determinar o número de linhas ou itens em um conjunto de dados. Ela conta o total de valores não nulos em uma coluna específica.

Exemplo:

```
SELECT COUNT(*) FROM nome_tabela;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## FUNÇÕES AGREGADAS:

COUNT, **SUM**, AVG, MAX, MIN

**SUM**: A função SUM é usada para calcular a soma de todos os valores em uma coluna numérica. Ela é útil para obter a soma total de um conjunto de valores.

Exemplo:

```
SELECT SUM(coluna_numerica) FROM nome_tabela;
```



# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## FUNÇÕES AGREGADAS:

COUNT, SUM, **AVG**, MAX, MIN

**AVG:** A função AVG (Average) é usada para calcular a média aritmética dos valores em uma coluna numérica. Ela fornece o valor médio do conjunto de dados.

Exemplo:

```
SELECT AVG(coluna_numerica) FROM nome_tabela;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## FUNÇÕES AGREGADAS:

COUNT, SUM, AVG, **MAX**, MIN

**MAX**: A função MAX é usada para encontrar o valor máximo (maior) em uma coluna numérica. Ela retorna o valor mais alto presente no conjunto de dados.

Exemplo:

```
SELECT MAX(coluna_numerica) FROM nome_tabela;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## **FUNÇÕES AGREGADAS:**

COUNT, SUM, AVG, MAX, **MIN**

**MIN:** A função MIN é usada para encontrar o valor mínimo (menor) em uma coluna numérica. Ela retorna o valor mais baixo presente no conjunto de dados.

Exemplo:

```
SELECT MIN(coluna_numerica) FROM nome_tabela;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## **FUNÇÕES AGREGADAS:**

COUNT, SUM, AVG, MAX, MIN

- Recupere o total de funcionários não demitidos.
- Recupere a soma dos salários dos funcionários não demitidos.
- Recupere o total de funcionários não demitidos, a soma dos salários, média dos salários, maior salário pago e menor salário pago.
- Recupere a quantidade, soma e média dos pedidos com valor total acima de 1000 reais.

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## CLÁUSULAS:

### **GROUP BY / HAVING**

O **GROUP BY** é uma cláusula usada para agrupar os resultados da consulta com base nos valores de uma ou mais colunas. Ele divide o conjunto de dados em grupos, onde cada grupo possui valores iguais nas colunas especificadas. As funções agregadas são então aplicadas a cada grupo individualmente.

```
SELECT departamento, COUNT(*) AS total_funcionarios  
FROM funcionarios GROUP BY departamento;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## CLÁUSULAS:

### **GROUP BY / HAVING**

O **GROUP BY** é uma cláusula usada para agrupar os resultados da consulta com base nos valores de uma ou mais colunas. Ele divide o conjunto de dados em grupos, onde cada grupo possui valores iguais nas colunas especificadas. As funções agregadas são então aplicadas a cada grupo individualmente.

```
SELECT cliente, SUM(valor) AS total_pedidos  
FROM pedidos GROUP BY cliente;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## CLÁUSULAS:

GROUP BY / **HAVING**

O **HAVING** é uma cláusula do SQL que é usada em conjunto com a cláusula GROUP BY para filtrar os resultados de uma consulta após a aplicação da agregação. Enquanto a cláusula WHERE é usada para filtrar os dados antes da agregação, a cláusula HAVING é usada para filtrar os resultados depois que a agregação foi realizada, ou seja, para filtrar os grupos resultantes.

```
SELECT cliente, SUM(valor) AS total_pedidos  
FROM pedidos GROUP BY cliente HAVING total_pedidos > 1000;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## **GROUP BY:**

Recupere o nome do funcionário e o total de dependentes que o funcionário possui.

## **GROUP BY + HAVING:**

Recupere o nome dos cargos com dois ou mais funcionários vinculados.



# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

Mostre o nome do cargo e o total de funcionários não demitidos vinculados ao cargo e a média salarial para os funcionários vinculados ao cargo.

```
SELECT C.NOM_CARGO, COUNT(F.COD_FUNC), AVG(F.VAL_SALARIO)  
FROM CARGO C LEFT OUTER JOIN FUNCIONARIO F  
ON C.COD_CARGO = F.COD_CARGO  
WHERE F.DTA_DEMISSAO IS NULL  
GROUP BY C.COD_CARGO, C.NOM_CARGO;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

Mostre o nome dos funcionários que trabalham em mais de 3 projetos.

```
SELECT F.NOM_FUNC, COUNT(T.COD_FUNC) FROM FUNCIONARIO F INNER JOIN  
TRABALHA_EM T ON F.COD_FUNC = T.COD_FUNC GROUP BY F.COD_FUNC, F.NOM_FUNC  
HAVING COUNT(T.COD_FUNC) > 3;
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## Subquery:

Uma subconsulta (subquery) é uma consulta SQL aninhada dentro de outra consulta. Ela permite que você utilize o resultado de uma consulta interna como entrada para uma consulta externa.

As subconsultas são bastante úteis para realizar operações complexas e realizar consultas mais sofisticadas em bancos de dados. A subconsulta é sempre executada primeiro, e o resultado dela é usado na consulta principal (externa) para realizar operações adicionais, como filtrar, comparar ou fazer uma correspondência com outras tabelas.

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## Subquery correlacionada:

Uma subconsulta correlacionada é aquela em que a subconsulta depende dos valores da consulta externa. Ou seja, a subconsulta é reexecutada para cada registro retornado pela consulta externa. Ela pode ser usada em cláusulas WHERE, HAVING ou JOIN para realizar filtragens e comparações mais complexas.

Exemplo:

```
SELECT nome FROM clientes WHERE idade > (SELECT AVG(idade) FROM clientes);
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## Subquery não correlacionada:

Uma subconsulta não correlacionada é aquela que pode ser executada independentemente da consulta externa, pois não depende dos valores da consulta externa. Ela é geralmente usada em cláusulas FROM ou JOIN para obter um conjunto de dados que será usado na consulta externa.

Exemplo:

```
SELECT nome, endereco FROM clientes WHERE cliente_id IN  
(SELECT cliente_id FROM pedidos WHERE valor_total > 1000);
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## **Subquery:**

Recupere o nome e salário dos funcionários que recebem o maior salário na empresa.

Recupere o nome e salário dos funcionários não demitidos que recebem salário acima da média salarial dos funcionários não demitidos.

## **IN / NOT IN:**

Recupere nome dos funcionários que não são gerentes.

Recupere o nome dos funcionários que não são vendedores, mas são coordenadores de projetos.

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## Subquery:

Recupere o nome dos funcionários que não são gerentes de departamentos e nem coordenadores de projetos.

```
SELECT NOM_FUNC FROM FUNCIONARIO  
WHERE COD_FUNC NOT IN  
    (SELECT COD_FUNC_GERENTE FROM DEPARTAMENTO WHERE  
        COD_FUNC_GERENTE IS NOT NULL) AND COD_FUNC NOT IN  
    (SELECT COD_FUNC_COORDENADOR FROM PROJETO WHERE  
        COD_FUNC_COORDENADOR IS NOT NULL);
```

# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## Desafio:

Recupere o nome dos projetos com soma total de horas trabalhadas pelos funcionários até 20 horas.

Recupere o nome dos funcionários que são coordenadores de projetos, gerentes de departamentos e supervisores ao mesmo tempo.

Recupere o nome dos funcionários que são coordenadores de projetos e o total de projetos que estes funcionários trabalham.



# Projeto Físico de BD

- **DQL** (*Data Query Language*): Linguagem de Consulta de Dados.

## Desafio:

Recupere o nome dos funcionários e o total de funcionários supervisionados que estes funcionários supervisionam.

Recupere o nome dos funcionários que são coordenadores de projeto mas não são gerentes de departamentos. Use duas subconsultas no WHERE.



Obrigado!

joao.aramuni@newtonpaiva.br