



*Serviço Nacional de Aprendizagem Industrial*

**PELO FUTURO DO TRABALHO**

# Modelagem de banco de dados relacional: Desvendando o SQL II

Rafael C. Ventura

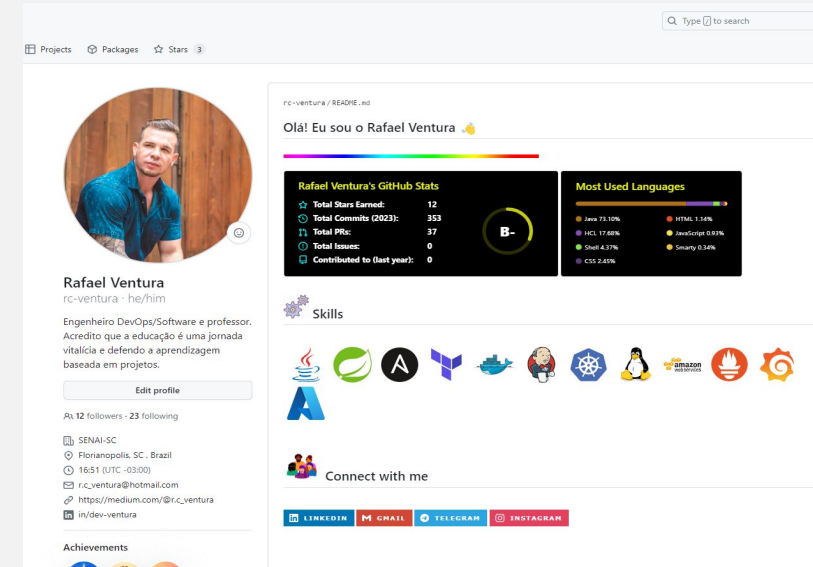
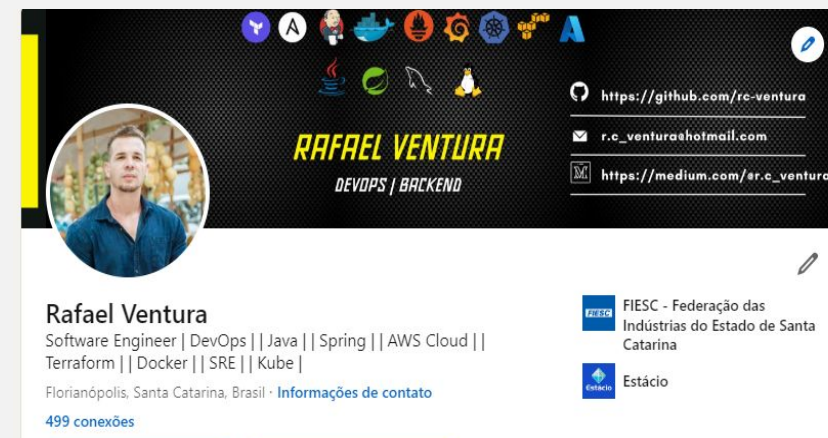
# Quem sou eu?

- Rafael C. Ventura
- Técnico em Segurança de Dados pela ANHANGUERA.
- Estudo Análise e Desenvolvimento de Sistemas na Estácio.
- Mestre em Educação pela PUC-RJ
- Leciono há mais de 10 anos.
- Experiência Cloud, IaC, DevOps
- email: rafael.ventura@edu.sc.senai.br




# Redes Sociais

- ❑ <https://www.linkedin.com/in/dev-ventura/>
- ❑ <https://github.com/rc-ventura>



# Entra 21

- ☐ Lógica com Java Script ( 4 encontros) 
- ☐ Banco de dados Relacionais (10 encontros)
- ☐ Metodologia Ágeis (1 encontro)
- ☐ Github (3 encontros)
- ☐ Programação Orientada à Objetos JS ( 10 encontros)
- ☐ React (15 encontros)
- ☐ Desenvolvimento do Trabalho de Conclusão do Curso ( 6 encontros)

# O que iremos aprender?

- Manipulando a cláusula UNION
- Manipulando sub consultas
- Criando uma view
- Começando uma transação
- Commit e Rollback

# Nossas referências

□ **RelaX** (<https://dbis-uibk.github.io/relax/landing>)

□ **SCALER** (<https://www.scaler.com/topics/sql/sql-query-execution-order/>)

# UNION

A cláusula UNION em SQL é usada para combinar os resultados de duas ou mais consultas em uma única tabela resultante. Cada consulta dentro do UNION deve ter o mesmo número de colunas e as colunas correspondentes devem ter tipos de dados compatíveis.

A principal finalidade do comando UNION é:

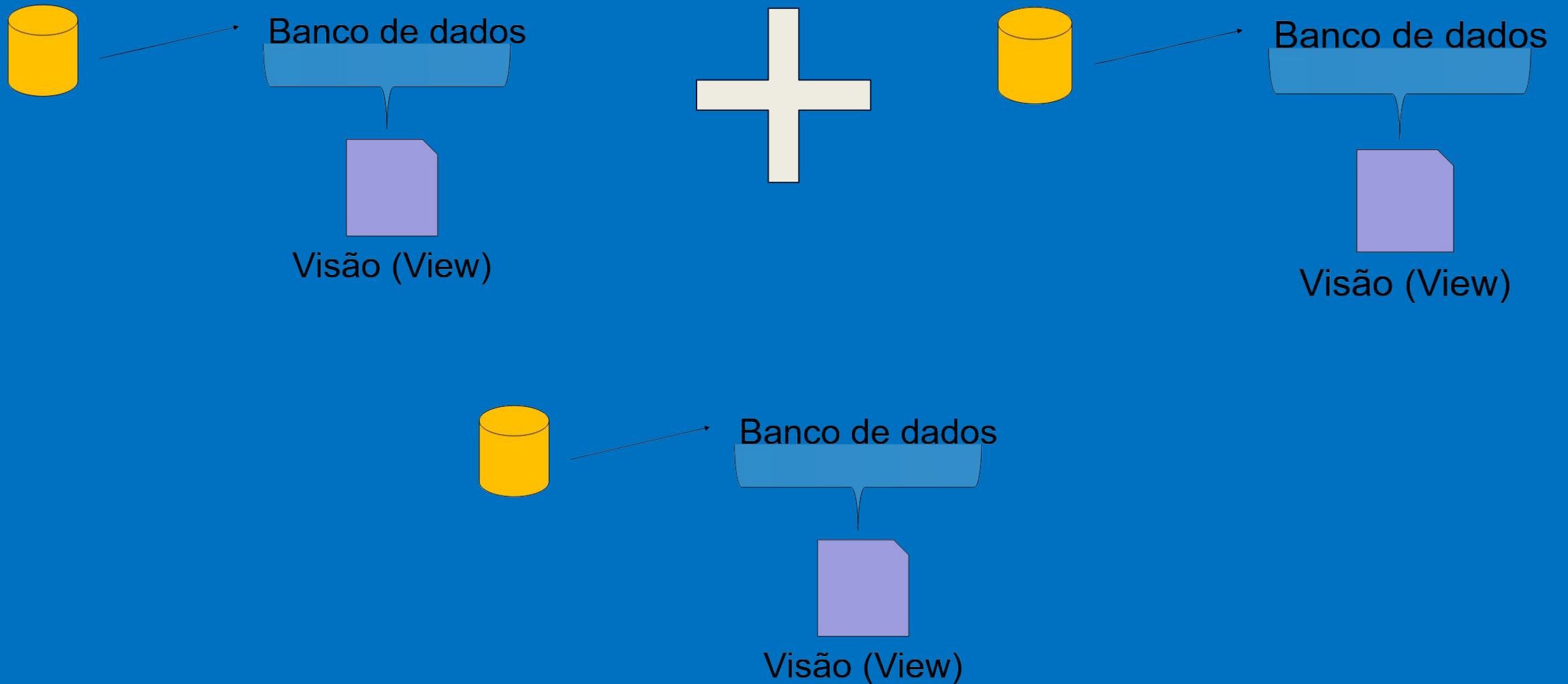
- ❖ **Combinar Resultados de Consultas Diferentes:** Você pode usar UNION quando deseja combinar os resultados de duas ou mais consultas que têm a mesma estrutura de colunas, mas são executadas em diferentes conjuntos de dados. Isso permite que você obtenha um único conjunto de resultados contendo dados de várias fontes.
- ❖ **Remover Duplicatas:** Por padrão, o UNION remove automaticamente registros duplicados dos resultados combinados. Isso é útil quando você tem consultas que podem retornar registros duplicados e deseja obter apenas entradas exclusivas.



# SUB CONSULTAS

- Quando é necessário juntar duas views, isto é o resultado de duas consultas individuais agrupados em uma única view.
- Consultas aninhadas

# SUB CONSULTA



# SUB CONSULTAS

[https://drive.google.com/file/d/1w0ypocdmrVFYRfv7131t0fFKTJKYi\\_y7/view?usp=sharing](https://drive.google.com/file/d/1w0ypocdmrVFYRfv7131t0fFKTJKYi_y7/view?usp=sharing)

# VIEWS

□ **VIEWS** : É uma tabela lógica, resultado de uma consulta, que pode ser usada depois em qualquer outra consulta.

```
CREATE VIEW `VW_Nome_da_View` AS  
(Consulta)
```

```
REPLACE VIEW `VW_Nome_da_View` AS  
(Consulta)
```

# TRANSAÇÕES

- Uma unidade lógica de processamento que visa preservar a integridade e consistência dos dados.
- Tudo que eu fizer no meu banco de dados quando abrir uma transação ficará gravado em memória e não será persistente.

**START TRANSACTION**

**COMMIT;**

**ROLLBACK;**

# TRANSAÇÕES

- **START TRANSACTION** = Cria um ponto de estado do banco de dados.
- **COMMIT** = Conforma todas as operações entre o Start transaction e o comando Commit. Tudo será gravado no banco de dados de forma persistente.
- **ROLLBACK** = Tudo que foi feito entre o start transaction e o Rollback será descartado e os dados voltarão ao status de quando o start transaction foi executado.

# GROUP BY (SUM, COUNT, MAX, MIN, MEDIA)

Para funções de agregação podemos utilizar os comandos:

- **MAX:** a partir de um conjunto de valores é retornado o maior entre eles;
- **MIN:** analisa um grupo de valores e retorna o menor entre eles;
- **SUM:** calcula o somatório dos valores de um campo específico;
- **AVG:** realiza a média aritmética dos valores de uma determinada coluna;
- **COUNT:** contabiliza a quantidade de linhas selecionadas.

Nem sempre será necessário agrupar com Group By para usar essas cláusulas. Por exemplo Max e Min pode ser usado sem Group By

# DIFERENÇA

```
SELECT VENDAS.ID_VENDEDOR, VENDEDORES.NOME_VENDEDOR, VENDAS.QTD_VENDIDA  
FROM VENDAS, VENDEDORES  
WHERE VENDAS.ID_VENDEDOR = VENDEDORES.ID_VENDEDOR
```

```
SELECT VENDAS.ID_VENDEDOR, VENDEDORES.NOME_VENDEDOR, VENDAS.QTD_VENDIDA  
FROM VENDAS INNER JOIN VENDEDORES  
ON VENDAS.ID_VENDEDOR = VENDEDORES.ID_VENDEDOR
```



# GROUP BY (SUM, COUNT, MAX, MIN, AVG)

- 1- Qual a maior venda do clube do livro ?
- 2- Qual vendedor vendeu mais ?
- 3- A quantidade de livros que cada vendedor vendeu?
- 4- Que livro foi mais vendido?
- 5- Quais livros não foram vendidos ?
- 6- Existe algum livro que teve vendas e não faz mais parte dos livros disponíveis para comercialização? (Null em id do livro)
- 7 - Qual a média de vendas entre os vendedores?
- 8 - Qual foi o total de todas as vendas feitas ?
- 9 - Qual o maior pedido em termos de valor monetário feito?
- 10 - Quais os livros estão com um exemplar no estoque?



*Serviço Nacional de Aprendizagem Industrial*

**PELO FUTURO DO TRABALHO**

**0800 048 1212**     **sc.senai.br**

Rodovia Admar Gonzaga, 2765 - Itacorubi - 88034-001 - Florianópolis, SC