SESI SENAI

UC Banco de Dados

Lista de comandos SQL

UC Banco de Dados

Linguagem de definição DDL

UC Banco de Dados

A linguagem de definição de dados, conhecida como **DDL** é um grupo de comandos dentro da linguagem SQL que é utilizada para **criação, alteração e exclusão** de objetos em um banco de dados, seus principais comandos são: **CREATE, ALTER, DROP e TRUNCATE.**

UC Banco de Dados

CREATE:

O comando CREATE é utilizado para a criação de estruturas de objetos do banco de dados, pode ser utilizado para situações como criação de um novo database, tabela e índice.

SESI SENAI

```
CREATE DATABASE veterinaria;


CREATE TABLE animal (
id_animal int primary key not null auto_increment,
nome_animal varchar(60) not null,
porte_animal varchar(30), raca varchar(30),
pelo varchar(30),
status int not null);
```

UC Banco de Dados

ALTER:

O comando ALTER é utilizado para alteração/modificação de estruturas de objetos do banco de dados, também pode ser utilizado para situações de modificação de um objeto existente no seu banco de dados como adicionar ou remover uma coluna em uma tabela.

UC Banco de Dados

```
ALTER TABLE animal ADD data_nasc  date;

ALTER TABLE animal DROP COLUMN pelo;

ALTER TABLE animal CHANGE porte_animal  porte_animal
VARCHAR(20) NOT NULL;

ALTER TABLE animal CHANGE porte_animal porte_animal
VARCHAR(30) NOT NULL;
```

UC Banco de Dados

DROP:

O comando DROP é utilizado para remoção de objetos e até mesmo de um banco de dados, portanto, é a maneira em que é feito a exclusão de tabelas, usuários, databases, entre outras estruturas. **Deve ser utilizado com cautela.**

UC Banco de Dados

DROP DATABASE veterinaria**;**
DROP TABLE animal**;**

UC Banco de Dados

TRUNCATE:

O comando TRUNCATE é utilizado para exclusão de todos os registros de uma tabela de forma imediata, ele é bem mais rápido que um comando **DELETE** pois não grava os dados que estão sendo removidos no log de transações. **Deve ser utilizado com cautela.**

UC Banco de Dados

TRUNCATE TABLE animal;

UC Banco de Dados

Linguagem de manipulação DML

UC Banco de Dados

A linguagem de manipulação de dados, mais conhecida como DML é um grupo de comandos dentro da linguagem SQL que é utilizada para inclusão, remoção e modificar o conteúdo das tabelas em um banco de dados, seus principais comandos são: **INSERT, UPDATE e DELETE**.

UC Banco de Dados

INSERT:

É o comando utilizado para inserir registros em uma tabela já existente no seu banco de dados.

UC Banco de Dados

INSERT INTO animal (id_animal, nome_animal, porte_animal, raca, status) **VALUES** (1, ‘Tonico das Neves’, ‘Médio’, ‘SRD’, 1);

UC Banco de Dados

UPDATE:

Comando utilizado para modificar as informações existentes em uma tabela, sejam dados individuais ou grupos de dados, portanto você pode atualizar um único registro ou vários dentro de uma tabela em seu banco de dados.

UC Banco de Dados

UPDATE animal **SET** porte_animal = ‘Grande’ WHERE id_animal = 1;

UC Banco de Dados

DELETE:

Comando utilizado para exclusão de dados de uma tabela em seu banco de dados.

UC Banco de Dados

DELETE FROM animal *WHERE id_animal = 1*;

SESI SENAI

UC Banco de Dados

Linguagem de consulta DQL

UC Banco de Dados

A linguagem de consulta de dados, mais conhecida como DQL é relacionada à um comando específico dentro da linguagem SQL, o comando mais utilizado de todos, seu objetivo é realizar consultas a dados pertencentes a uma tabela.

UC Banco de Dados

O comando SELECT é composto de várias cláusulas, operadores lógicos, relacionais e funções de agregação, isso possibilita a elaboração de consultas das mais simples até as mais avançadas.

Por meio desse simples comando, podemos retornar dados para outros comandos SQL e também para outras aplicações.

O comando SELECT é utilizado para consultar tabelas e dessa forma é possível obter informações específicas do banco de dados.

UC Banco de Dados

SELECT * FROM animal

O comando select, selecionará, a instrução dada a seguir. Ao usarmos o caracter * selecionamos, ou seja, exibimos todos os campos (*id_animal, nome_animal, porte_animal, raca, status*). *Já o* comando **FROM** , indicará de qual tabela será recebido os valores.

UC Banco de Dados

SELECT nome_animal, raca FROM animal

Com a comando acima, iremos selecionar todos os registro da tabela animal, porem, visualizar somente os campos *nome_animal* e *raca*.

UC Banco de Dados

A cláusula WHERE consiste em um predicado (condição) envolvendo atributos das
tabelas que aparecem na cláusula FROM. A condição que segue a cláusula WHERE pode conter operadores de comparação.

SELECT * FROM animal WHERE raca = “Rottweiler”

A consulta acima, selecionará todos os registro de animais com raça Rottweiler.

UC Banco de Dados

A condição (ou predicado) que segue a clausula WHERE pode conter operadores de comparação.

= Igual

> Maior

< Menor

<> diferente

>= maior ou igual

<= menor ou igual

UC Banco de Dados

Exemplos:

SELECT * FROM aluno WHERE id_aluno > 15500;

SELECT * FROM aluno WHERE serie <> 3;

SELECT * FROM venda WHERE data_venda <= “2021-12-31”;

SESI SENAI

UC Banco de Dados

OPERADORES RELACIONAIS

- **BETWEEN** – Utilizado para especificar valores dentro de um intervalo fechado.

- **LIKE** – Utilizado na comparação de um modelo e para especificar registros de um banco de dados. "Like" + extensão % significa buscar todos resultados com o mesmo início da extensão.

- **IN** - Utilizado para verificar se o valor procurado está dentro de um intervalo. Ex.: idade IN (10,11,12,13).

UC Banco de Dados

Podemos refinar ainda mais nossas consultas com operadores lógicos

AND, OR e NOT.

AND = E
OR = OU
NOT = NÃO (Negação)

UC Banco de Dados

Exemplos:

SELECT * FROM aluno WHERE id_aluno > 15500 **AND** data_matricula > “2020-01-01”

SELECT * FROM aluno WHERE cidade = “Florianópolis” **OR** cidade = “São José”

UC Banco de Dados

O comando NOT pode facilitar a criação de consultas como vemos no exemplo abaixo:

SELECT * FROM aluno WHERE id_aluno <> 1030 **AND** id_aluno <> 2135 **AND** id_aluno <> 14175

UC Banco de Dados

O comando NOT pode facilitar a criação de consultas como vemos no exemplo abaixo:

```
SELECT * FROM aluno WHERE id_aluno <> 1030 AND id_aluno <> 2135
AND id_aluno <> 14175
```

```
SELECT * FROM aluno WHERE id_aluno NOT IN (1030, 2135, 14175)
```

Com o comando **NOT**, NÃO selecionamos os valores e com o comando **IN**, definimos um conjunto de valores.

UC Banco de Dados

O comando NOT pode facilitar a criação de consultas como vemos no exemplo abaixo:

SELECT * FROM aluno WHERE cidade <> ‘Joinville’ **AND** cidade <> ‘Mafra’

SELECT * FROM aluno WHERE cidade **NOT IN** (‘Joinville’, ‘Mafra’)

O operador NOT IN só pode substituir os operadores <> ou !=. Não pode substituir =, <, >, <=, >=, BETWEEN ou LIKE. Ele só encontrará e excluirá correspondências exatas. Valores duplicados na lista são ignorados.

UC Banco de Dados

FUNÇÃO DE AGREGAÇÃO

- AVG – Utilizada para calcular a média dos valores de um campo determinado;
- COUNT – Utilizada para devolver o número de registros da seleção;
- SUM – Utilizada para devolver a soma de todos os valores de um campo determinado;
- MAX – Utilizada para devolver o valor mais alto de um campo especificado;
- MIN – Utilizada para devolver o valor mais baixo de um campo especificado.

UC Banco de Dados

FUNÇÃO DE AGREGAÇÃO

- AVG – Utilizada para calcular a média dos valores de um campo determinado:

SELECT **AVG** (idade) FROM aluno WHERE serie = 3 AND turma = ‘B’

SELECT **AVG** (valor_total) FROM venda WHERE id_cliente = 85

UC Banco de Dados

FUNÇÃO DE AGREGAÇÃO

- COUNT – Utilizada para devolver o número de registros da seleção:

SELECT **COUNT** (uf) FROM aluno WHERE uf = ‘SC’

SELECT **COUNT** (status_pgto) FROM venda WHERE status_pgto = ‘Aberto’

UC Banco de Dados

FUNÇÃO DE AGREGAÇÃO

- SUM – Utilizada para devolver a soma de todos os valores de um campo determinado:

SELECT **SUM** (peso) FROM entrega WHERE romaneio_entrega = 50

SELECT **SUM** (salario_bruto) FROM folha WHERE cargo = ‘Vendedor’ AND sexo = ‘M’

UC Banco de Dados

FUNÇÃO DE AGREGAÇÃO

- MAX – Utilizada para devolver o valor mais alto de um campo especificado:

SELECT **MAX** (peso) FROM aluno WHERE serie = 1 AND turma = ‘C’

SELECT **MAX** (salario_liquido) FROM folha WHERE cargo = ‘Consultora’ AND sexo = ‘F’

SESI SENAI

UC Banco de Dados

FUNÇÃO DE AGREGAÇÃO

- MIN – Utilizada para devolver o valor mais baixo de um campo especificado :

SELECT **MIN** (peso) FROM aluno WHERE serie = 9 AND turma = ‘A’ AND idade >= 14

SELECT **MIN** (salario_liquido) FROM folha WHERE cargo = ‘Gerente’ AND Depto = ‘Vendas’

SESI SENAI

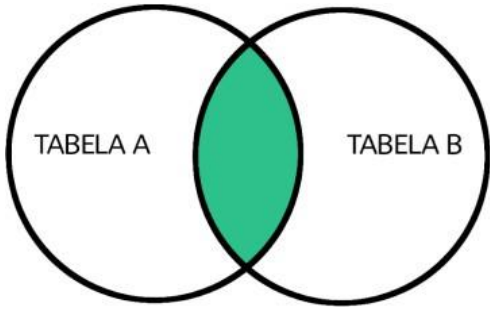
FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS

Uma cláusula JOIN em SQL, correspondente a uma operação de junção em álgebra relacional, combina colunas de uma ou mais tabelas em um banco de dados relacional.

Um JOIN é um meio de combinar colunas de uma (auto-junção) ou mais tabelas, usando valores comuns a cada uma delas. O SQL padrão ANSI especifica cinco tipos de JOIN:

INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN e CROSS JOIN.

UC Banco de Dados

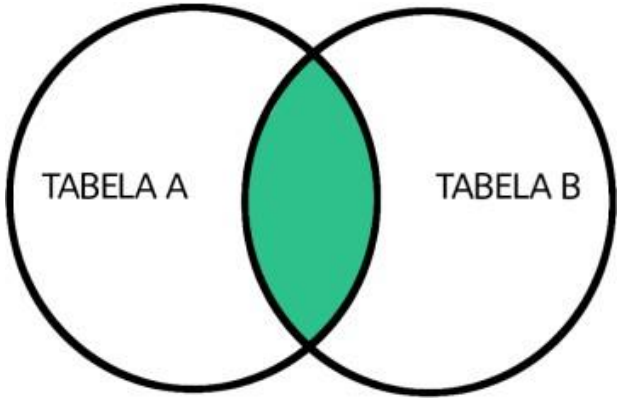


FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS

A cláusula **INNER JOIN** compara cada linha da tabela A com as linhas da tabela B para encontrar todos os pares de linhas que satisfazem a condição de junção. Se a condição de junção for avaliado como **TRUE**, os valores da coluna das linhas correspondentes das tabelas A e B serão combinados em uma nova linha e incluídos no conjunto de resultados.

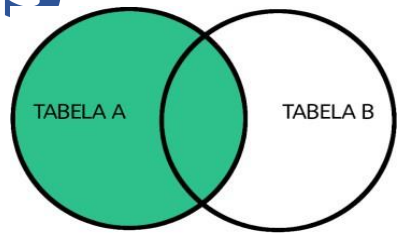
UC Banco de Dados

FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS



SELECT * FROM cliente **INNER JOIN** pedido WHERE cliente.id_cliente = pedido.id_cliente SELECT * FROM cliente c **INNER JOIN** pedido p WHERE c.id_cliente = p.id_cliente

UC Banco de Dados



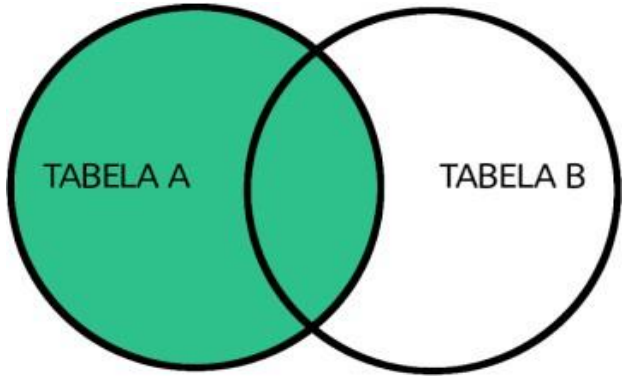
FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS

A cláusula **LEFT JOIN** é usado para retornar todos os registros da tabela esquerda, além dos registros da tabela à direita que têm valores em comum com a tabela esquerda. Para cada linha da tabela A, a consulta a compara com todas as linhas da tabela B. Se um par de linhas fizer com que a condição de junção seja avaliado como TRUE, os valores da coluna dessas linhas serão combinados para formar uma nova linha que será incluída no conjunto de resultados.

UC Banco de Dados

SESI SENAI

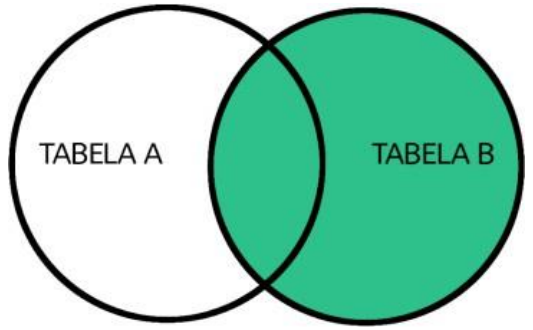
FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS



SELECT * FROM cliente c **LEFT JOIN** pedido p ON c.id_cliente = p.id_cliente

UC Banco de Dados

FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS

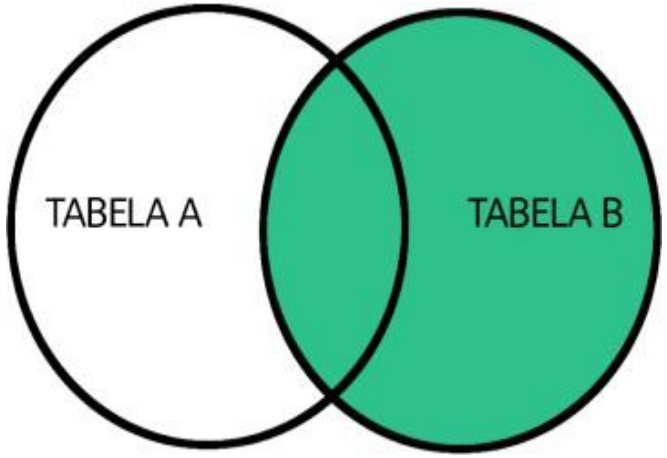


Com a cláusula **RIGHT JOIN**, podemos retornar todos os valores da tabela da direita juntamente com os registros em comum na tabela esquerda.

A RIGHT JOIN retorna um conjunto de resultados que inclui todas as linhas da tabela “direita” B, com ou sem linhas correspondentes na tabela “esquerda” A. Se uma linha na tabela direita B não tiver nenhuma linha correspondente da tabela “esquerda” A, a coluna da tabela “esquerda” A no conjunto de resultados será nula igualmente ao que acontece no LEFT JOIN.

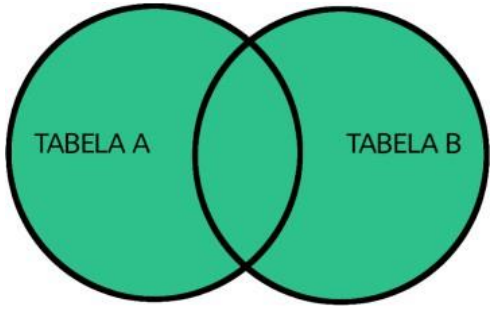
UC Banco de Dados

FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS



SELECT * FROM cliente c **RIGHT JOIN** pedido p ON c.id_cliente = p.id_cliente

UC Banco de Dados



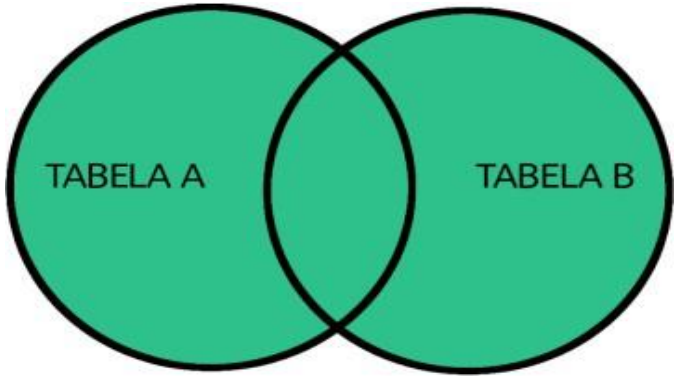
FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS

A cláusula FULL JOIN retorna todas as linhas das tabelas unidas, correspondidas ou não, ou seja, você pode dizer que a FULL JOIN combina as funções da LEFT JOIN e da RIGHT JOIN. FULL JOIN é um tipo de junção externa, por isso também é chamada junção externa completa.

Quando não existem linhas correspondentes para a linha da tabela esquerda, as colunas da tabela direita serão nulas. Da mesma forma, quando não existem linhas correspondentes para a linha da tabela direita, a coluna da tabela esquerda será nula.

SESI SENAI

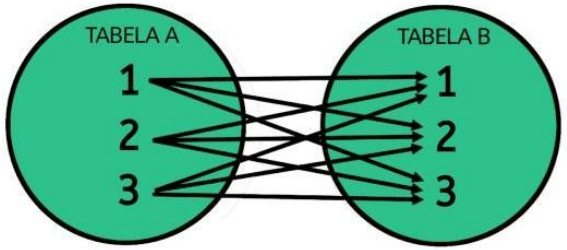
FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS



SELECT * FROM cliente c **FULL JOIN** pedido p WHERE c.id_cliente = p.id_cliente

UC Banco de Dados

FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS

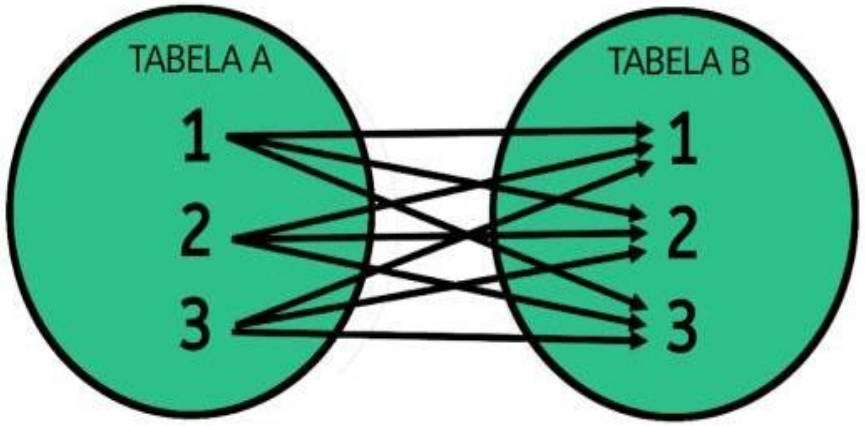


Com o comando SQL CROSS JOIN é possível fazer um produto cartesiano entre as tabelas.
Isso significa que para cada linha da tabela esquerda ele vai retornar todas as linhas da tabela direita, ou vice-versa.

Porém, para isso é preciso que ambas tenham o campo em comum, para que a ligação exista entre as duas tabelas.

UC Banco de Dados

FUNÇÃO DE JUNÇÃO DE DADOS DE TABELAS



SELECT * FROM cliente c **CROSS JOIN** pedido p WHERE c.id_cliente = p.id_cliente