

DML – Inserir, Atualizar e Deletar

DDL é como você se refere a **comandos SQL** que tem como função **manipular dados de tabelas**. Por exemplo: *inserir, buscar, filtrar, remover*:

O valor do PK não precisa ser explícito, caso ele seja auto incrementável

```
INSERT INTO products (name, price, coffee_origin)
VALUES ('ESPRESSO', 8.74, 'BRAZIL');
```

Os valores são correspondentes aos campos da tabela

É bem óbvio que o critério de WHERE vai impactar diretamente na quantidade de linhas afetadas

```
UPDATE products
SET name = 'MACCHIATO'
WHERE id = 2;
```

Update em uma coluna e sem "alias"

```
UPDATE products as p
SET p.price = 2.99, p.coffee_origin = 'ETHIOPIA'
WHERE p.name = 'AMERICANO';
```

Update em mais colunas usando "alias"

A sintaxe de deleção das linhas é bem simples **"DELETE FROM <> WHERE <>"**, essas operações também podem ter mais critérios encadeados usando **AND, LIKE, etc...**

DML – Consultas Básicas

Consultas tem o objetivo de retornar dados específicos, os **critérios são os mais variados possíveis**. Da pra ordenar, selecionar o maior valor, somar, usar operadores lógicos e etc...:

É muito comum juntar vários filtros de consulta com "AND", o "LIKE" é uma espécie de comparador de String usando "%" como um coringa da comparação

```
SELECT * FROM products p
WHERE p.price >= 3.00 AND p.price <= 8.00
AND p.coffee_origin LIKE '%ZIL';
```

Isso poderia ser um BETWEEN

id	name	price	coffee_origin
1	MACHIATO	3,74	BRAZIL

Invés de usar "==" para Strings, pode usar uma forma mais semântica como IN() e NOT IN()

```
SELECT * FROM customers c
WHERE c.last_name IN ('Taylor', 'Armstrong');
```

Enter a SQL expression to filter results (using AND, OR, NOT, etc.)

id	first name	last name	gender	phone number
5	Sarah	Taylor	F	01176348290
6	Katie	Armstrong	F	01145787353
14	John	Taylor	M	[NULL]

```
SELECT * FROM orders o
WHERE o.order_time BETWEEN '2017-1-1' AND '2017-1-7';
```

Um BETWEEN entre datas é elegante

```
SELECT * FROM products p
ORDER BY p.price ASC
LIMIT 3;
```

```
SELECT * FROM customers c
ORDER BY c.first name DESC
LIMIT 5;
```

```
SELECT * FROM orders o
WHERE o.customer_id = 1
ORDER BY o.order time DESC
LIMIT 10;
```

Uma ordenação pode ser feita quase que por qualquer atributo, tanto CRESCENTE ou DECRESCENTE

Dá pra LIMITAR a quantidade de resultados

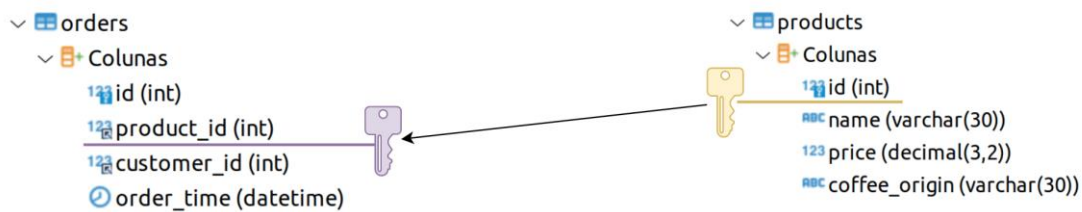
O **distinct** não entrou em exemplo, mas o uso dele é mto fácil também. É só aplicar o DISTINCT em uma coluna específica e você vai ter todos os valores distintos da coluna, também dá pra usar como filtro em where e etc...

DML – Consultas em múltiplas Tabelas

A sintaxe básica que te permite juntar dados de múltiplas tabelas é o "JOIN", obviamente **só dá pra juntar tabelas que possuem uma relação através das chaves**.

INNER JOIN: Retorna os dados que dão "match" nas duas (pode ser mais, é só ir dando JOIN) tabelas, esse match é relacionado a tabela 2 possuir valor atrelado a um campo chave na tabela 1 e

vice versa:



Obviamente critérios de seleção podem ser aplicados. Só vão ser trazidos os dados onde a cláusula de join for verdadeira

```
SELECT p.name, p.price, o.order_time FROM orders o
INNER JOIN products p ON o.product_id = p.id
WHERE p.price >= 6.00
ORDER BY o.order_time;
```

A junção acontece nesse ponto

LEFT JOIN: Traz **TODOS OS DADOS** da tabela na esquerda (pode entender como a primeira tabela referenciada) e os **dados que combinam da tabela direita**. É tipo tu pegar todos os dados da tabela 1 e os que dão o match na tabela 2.

RIGHT JOIN: É igual o left join, a diferença é que a tabela que vai retornar todos os dados é a 2 (da direita).