

Consolidando o seu conhecimento

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Na base **sucos_vendas**, abra um novo script MySQL.

2) Digite as três consultas abaixo:

```
SELECT A.NOME_DO_PRODUTO FROM tabela_de_produtos A;
```

```
SELECT A.NOME_DO_PRODUTO, C.QUANTIDADE  
FROM tabela_de_produtos A  
INNER JOIN itens_notas_fiscais C ON A.codigo_do_produto = C.codigo_do_produto;
```

```
SELECT A.NOME_DO_PRODUTO, YEAR(B.DATA_VENDA) AS ANO, C.QUANTIDADE  
FROM tabela_de_produtos A  
INNER JOIN itens_notas_fiscais C ON A.codigo_do_produto = C.codigo_do_produto  
INNER JOIN notas_fiscais B ON C.NUMERO = B.NUMERO;
```

```
SELECT A.NOME_DO_PRODUTO, YEAR(B.DATA_VENDA) AS ANO, SUM(C.QUANTIDADE) AS TOTAL  
FROM tabela_de_produtos A  
INNER JOIN itens_notas_fiscais C ON A.codigo_do_produto = C.codigo_do_produto  
INNER JOIN notas_fiscais B ON C.NUMERO = B.NUMERO  
GROUP BY A.NOME_DO_PRODUTO, YEAR(B.DATA_VENDA)  
ORDER BY A.NOME_DO_PRODUTO, YEAR(B.DATA_VENDA);
```

COPIAR CÓDIGO

3) Se você executar estas consultas, uma a uma, você verá que, a cada execução, o tempo de retorno das consultas passa a demorar cada vez mais. É que cada consulta vai exigindo mais processamento do banco de dados:

	Time	Action	Message	Duration / Fetch
7	17:32:42	SELECT A.NOME_DO_PR...	36 row(s) returned	0.000 sec / 0.000 sec
8	17:34:07	SELECT A.NOME_DO_PR...	1000 row(s) returned	0.000 sec / 0.000 sec
9	17:35:15	SELECT A.NOME_DO_PR...	Error Code: 1054. Unknown column 'A.NUMERO' in 'on clau...	0.000 sec
10	17:35:44	SELECT A.NOME_DO_PR...	1000 row(s) returned	0.000 sec / 0.000 sec
11	17:36:35	SELECT A.NOME_DO_PR...	153 row(s) returned	0.828 sec / 0.000 sec

4) Na linha de comando do Windows, acesse o diretório do MySQL:

```
cd\  
cd "Program Files"  
cd "MySQL"  
cd "MySQL 8.0"  
cd Bin
```

[COPIAR CÓDIGO](#)

5) Em seguida, acesse a interface de linha de comando do MySQL (a senha do usuário **root** será necessária):

```
mysql -uroot -p
```

[COPIAR CÓDIGO](#)

6) Já dentro da interface de linha de comando do MySQL, digite:

```
explain SELECT A.NOME_DO_PRODUTO FROM tabela_de_produtos A;
```

[COPIAR CÓDIGO](#)

7) Você verá alguns indicadores que refletem o custo de execução desta consulta.

8) Para visualizar em outro formato, digite:

```
explain format=JSON SELECT A.NOME_DO_PRODUTO FROM tabela_de_pro
```

[COPIAR CÓDIGO](#)

```
EXPLAIN: {
  "query_block": {
    "select_id": 1,
    "cost_info": {
      "query_cost": "3.75"
    },
    "table": {
      "table_name": "A",
      "access_type": "ALL",
      "rows_examined_per_scan": 35,
      "rows_produced_per_join": 35,
      "filtered": "100.00",
      "cost_info": {
        "read_cost": "0.25",
        "eval_cost": "3.50",
        "prefix_cost": "3.75",
        "data_read_per_join": "15K"
      },
      "used_columns": [
        "NOME_DO_PRODUTO"
      ]
    }
  }
}
```

Acima, você terá o plano de execução desta consulta e o parâmetro `cost_info` expressa o custo de resolução desta *query* (no caso acima, 3.75).

9) Veja o custo de outra consulta. Digite:

```
explain format=JSON SELECT A.NOME_DO_PRODUTO, C.QUANTIDADE FROM
```

[COPIAR CÓDIGO](#)

```
EXPLAIN: {
  "query_block": {
    "select_id": 1,
    "cost_info": {
      "query_cost": "76517.94"
    },
    "nested_loop": [
      {
        "table": {
          "table_name": "A",
          "access_type": "ALL",
          "possible_keys": [
            "PRIMARY"
          ],
          "rows_examined_per_scan": 35,
          "rows_produced_per_join": 35,
          "filtered": "100.00",
          "cost_info": {
            "read_cost": "0.25",
            "eval_cost": "3.50",
            "prefix_cost": "3.75",
            "data_read_per_join": "15k"
          },
          "cost_name": "f"
        }
      }
    ]
  }
}
```

Aqui, o custo da consulta, pelo plano de execução, passou a custar 76.517,94. Dezenas de vezes em relação à medição original.

10) Veja o custo de mais uma consulta. Digite:

```
explain format=JSON SELECT SELECT A.NOME_DO_PRODUTO, YEAR(B.DA
```

COPIAR CÓDIGO

O custo aumenta mais ainda (260.242,51).

```
EXPLAIN: {
  "query_block": {
    "select_id": 1,
    "cost_info": {
      "query_cost": "260242.51"
    },
    "nested_loop": [
      {
        "table": {
          "table_name": "B",
          "access_type": "ALL",
          "possible_keys": [
            "PRIMARY"
          ],
          "rows_examined_per_scan": 87768,
          "rows_produced_per_join": 87768,
          "filtered": "100.00",
          "cost_info": {
            "read_cost": "289.00",
            "eval_cost": "8776.80",
            "prefix_cost": "9065.80",
            "data_read_per_join": "6M"
          },
          "cost_name": "f"
        }
      }
    ]
  }
}
```

11) Você pode acompanhar o plano de execução pelo Workbench. Execute a consulta:

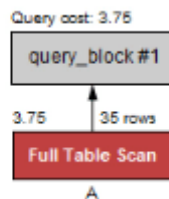
```
SELECT A.NOME_DO_PRODUTO FROM tabela_de_produtos A;
```

[COPIAR CÓDIGO](#)

12) Exiba o resultado através da opção **Execution Plan**:

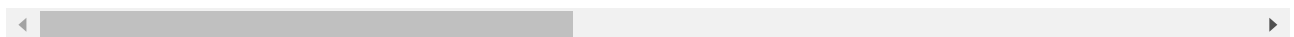


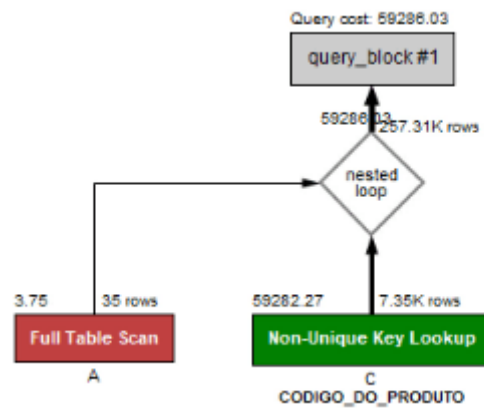
Você terá:



13) Comparar as outras consultas mais complexas. Primeiramente, execute:

```
SELECT A.NOME_DO_PRODUTO, C.QUANTIDADE FROM tabela_de_produtos
```

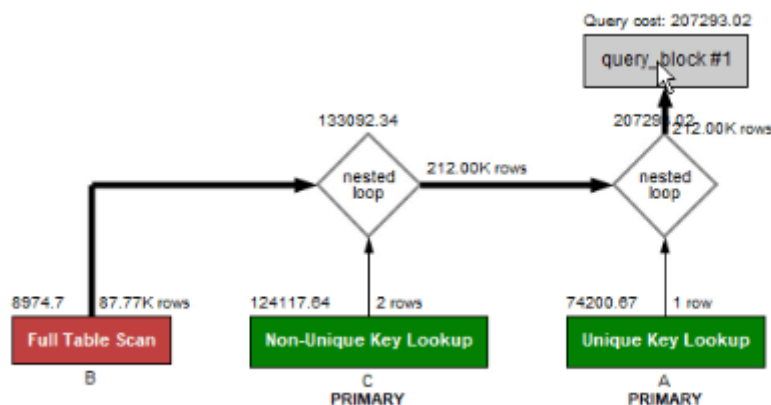
[COPIAR CÓDIGO](#)



14) Já na outra consulta mais complexa, digite:

```
SELECT SELECT A.NOME_DO_PRODUTO, YEAR(B.DATA_VENDA) AS ANO, C.(
```

COPIAR CÓDIGO



15) Quando você vê retângulos verdes, significa que a consulta utilizou algum tipo de índice. Quando você cria chaves primárias e estrangeiras, automaticamente, índices são criados e eles são usados nas consultas. Veja como isso acontece, primeiro criando três novas tabelas, com os comandos abaixo:

```
CREATE TABLE `itens_notas_fiscais2` (
  `NUMERO` int(11) NOT NULL,
  `CODIGO_DO_PRODUTO` varchar(10) NOT NULL,
  `QUANTIDADE` int(11) NOT NULL,
```

```
`PRECO` float NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_a:  
  
CREATE TABLE `notas_fiscais2` (  
  `CPF` varchar(11) NOT NULL,  
  `MATRICULA` varchar(5) NOT NULL,  
  `DATA_VENDA` date DEFAULT NULL,  
  `NUMERO` int(11) NOT NULL,  
  `IMPOSTO` float NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_a:
```

```
CREATE TABLE `tabela_de_produtos2` (  
  `CODIGO_DO_PRODUTO` varchar(10) NOT NULL,  
  `NOME_DO_PRODUTO` varchar(50) DEFAULT NULL,  
  `EMBALAGEM` varchar(20) DEFAULT NULL,  
  `TAMANHO` varchar(10) DEFAULT NULL,  
  `SABOR` varchar(20) DEFAULT NULL,  
  `PRECO_DE_LISTA` float NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_a:
```

[COPIAR CÓDIGO](#)

Estas tabelas são semelhantes às existentes, mas sem chaves primárias e estrangeiras.

16) Inclua dados nestas tabelas, executando:

```
INSERT INTO itens_notas_fiscais2 SELECT * FROM itens_notas_fisc  
INSERT INTO notas_fiscais2 SELECT * FROM notas_fiscais;  
INSERT INTO tabela_de_produtos2 SELECT * FROM tabela_de_produto
```

[COPIAR CÓDIGO](#)

17) Observe o plano de execução da consulta original:

```
SELECT A.NOME_DO_PRODUTO, C.QUANTIDADE  
FROM tabela_de_produtos A INNER JOIN itens_notas_fiscais C  
ON A.codigo_do_produto = C.codigo_do_produto;
```

[COPIAR CÓDIGO](#)

Aqui, o custo foi de 76517.94.

18) Já executando a consulta com as tabelas sem chaves primárias e estrangeiras:

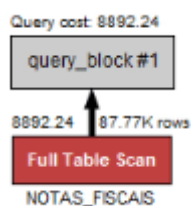
```
SELECT A.NOME_DO_PRODUTO, C.QUANTIDADE  
FROM tabela_de_produtos2 A INNER JOIN itens_notas_fiscais2 C  
ON A.codigo_do_produto = C.codigo_do_produto;
```

[COPIAR CÓDIGO](#)

O custo sobe para 769497.31.

19) Veja a influência do índice. Execute e veja o plano de execução:

```
SELECT * FROM NOTAS_FISCAIS WHERE DATA_VENDA = '20170101'
```

[COPIAR CÓDIGO](#)

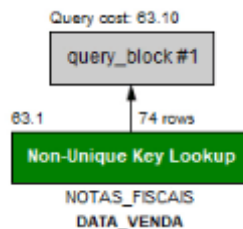
20) Crie o índice, executando:


```
ALTER TABLE NOTAS_FISCAIS ADD INDEX (DATA_VENDA);
```

[COPIAR CÓDIGO](#)

21) Veja o plano de execução novamente:

```
SELECT * FROM NOTAS_FISCAIS WHERE DATA_VENDA = '20170101'
```

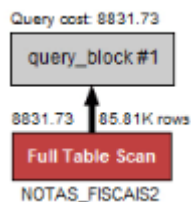
[COPIAR CÓDIGO](#)

22) Agora, apague o índice:

```
ALTER TABLE NOTAS_FISCAIS DROP INDEX DATA_VENDA;
```

[COPIAR CÓDIGO](#)

23) E execute novamente:



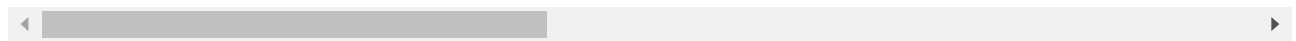
24) Há outra ferramenta, chamada **mysqlslap**, que simula acessos concorrentes a uma consulta. Você executa-o através da linha de comando. Logo, vá para:

```
cd\  
cd "Program Files"  
cd "MySQL"  
cd "MySQL 8.0"  
cd Bin
```

[COPIAR CÓDIGO](#)

25) Execute:

```
MYSQLSLAP -uroot -p --concurrency=100 --iterations=10 --create
```

[COPIAR CÓDIGO](#)

26) Você terá:

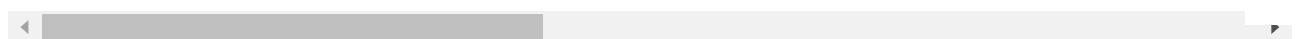
```
Average number of seconds to run all queries: 0.548 seconds  
Minimum number of seconds to run all queries: 0.203 seconds  
Maximum number of seconds to run all queries: 1.281 seconds  
Number of clients running queries: 100  
Average number of queries per client: 1
```

[COPIAR CÓDIGO](#)

A melhor execução da consulta retornou resultados em 0.548 segundos e a pior 1.281.

27) Use as tabelas sem chaves primárias e estrangeiras. Execute:

```
MYSQLSLAP -uroot -p --concurrency=100 --iterations=10 --create
```

[COPIAR CÓDIGO](#)

28) Você terá:

Average number of seconds to run all queries: 2.628 seconds

Minimum number of seconds to run all queries: 2.312 seconds

Maximum number of seconds to run all queries: 3.422 seconds

Number of clients running queries: 100

Average number of queries per client: 1

COPIAR CÓDIGO