

Aprendendo a usar o MySQL



Trabalho do grupo 4

-

Eduardo Domingues dos Santos
Hettore Eduardo Barreto da Silva
Valdete Alves do Nascimento




Iniciando o banco de dados

Após a instalação do MySQL Server com todos os adicionais, incluindo o MySQL Workbench (Usamos o Windows), é necessário configurar o banco e criar as tabelas, para depois importar os dados passados na proposta do trabalho.

É interessante saber que os comandos do MySQL são divididos entre,

conforme a tabela ao lado:

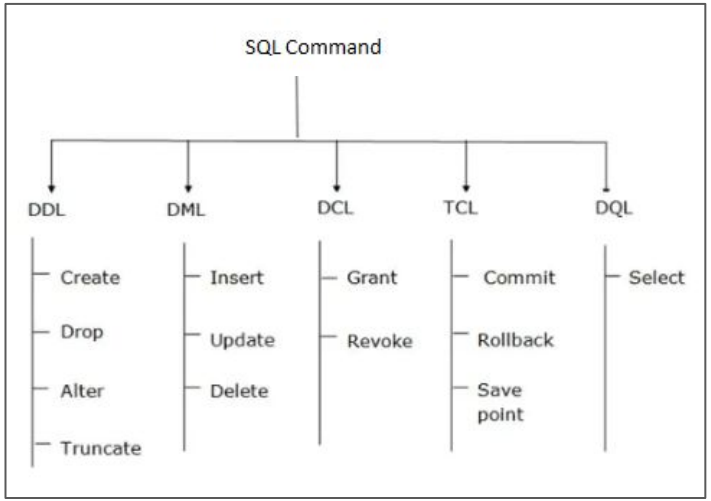
DDL - (Data Definition Language) - Linguagem de Definição de Dados

DML - (Data Manipulation Language) - Linguagem de Manipulação de Dados

DCL - (Data Control Language) - Linguagem de Controle de Dados

DTL - (Data Transaction Language) - Linguagem de transação de dados

DQL - (Data Query Language) - Linguagem de consulta de dados



Comando para criar o banco de dados

Usa-se CREATE DATABASE “NOME_DO_BANCO”.

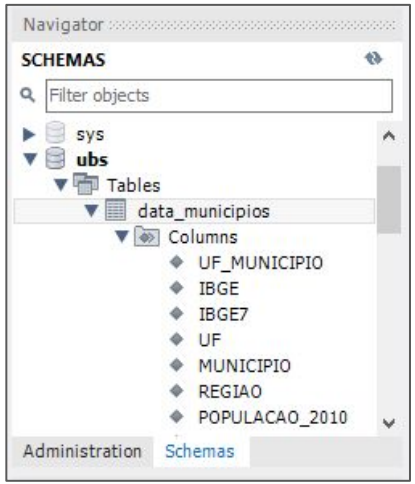
```
Limit to 1000 rows
1  CREATE DATABASE ubs; /*  Comando para criar o banco de dados  */
```

E após usa-se o comando USE “NOME_DO_BANCO”.

```
2
3  USE ubs; /*  Comando para usar/selecionar o banco de dados  */
```

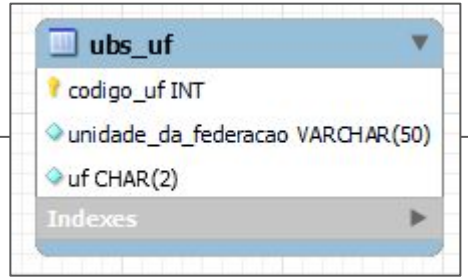
Na figura à direita mostra que quando é usado o comando USE “NOME_DO_BANCO”, o banco de dados que está nomeado como ubs fica em negrito para informar que é ele que vai ser usado, só após esse comando que será possível criar as tabelas.

Lembrete:
O MySQL não é case sensitive, salvo alguns casos por exemplo no linux onde o sistema força o MySQL a ser, porém os comandos em letra maiúscula são uma boa prática.



Criando as tabelas no banco de dados

Nossa primeira tabela será a ubs_uf que contém as colunas: codigo_uf, unidade_da_federacao e uf.

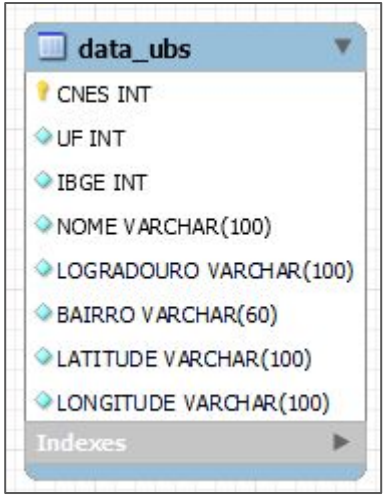



```
5  CREATE TABLE ubs_uf (
6  codigo_uf INT PRIMARY KEY NOT NULL,
7  unidade_da_federacao VARCHAR(50) NOT NULL,
8  uf CHAR(2) NOT NULL
9  ); /*  Comando para criar uma tabela já incluindo o título da coluna obs: tabela ainda sem conteúdo  */
```

Criando as tabelas no banco de dados

Nossa segunda tabela será a data_ubs que contém as colunas: CNES, UF, IBGE, NOME, LOGRADOURO, BAIRRO, LATITUDE e LONGITUDE.

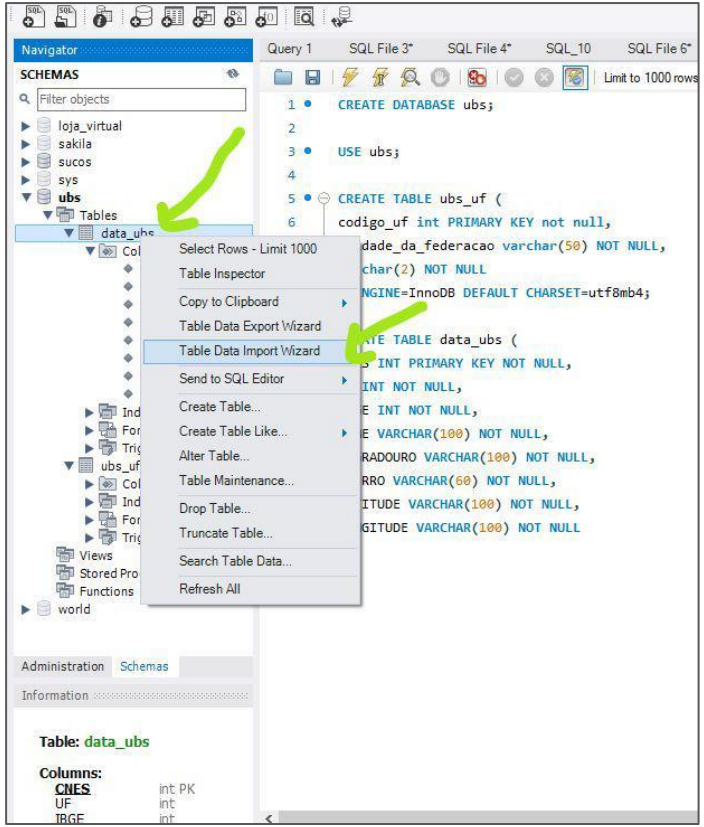
```
38  CREATE TABLE data_ubs (
39  CNES INT PRIMARY KEY NOT NULL,
40  UF INT NOT NULL,
41  IBGE INT NOT NULL,
42  NOME VARCHAR(100) NOT NULL,
43  LOGRADOURO VARCHAR(100) NOT NULL,
44  BAIRRO VARCHAR(60) NOT NULL,
45  LATITUDE VARCHAR(100) NOT NULL,
46  LONGITUDE VARCHAR(100) NOT NULL
47  );
```



Populando as tabelas

Os dados de cada tabela foram passados pelo professor e elas contém as informações necessárias para fazermos as consultas e responder as afirmações.

Depois de criar as tabelas usamos a ide para importar os dados que recebemos em arquivo .csv, basta clicar com o direito na tabela e depois em table data import wizard e seguir o passo a passo. Repetir o processo na segunda tabela.



Fazendo as consultas dos dados

Para consultar os dados de cada tabela dentro do banco de dados devemos seguir uma estrutura como o exemplo a seguir:

Aqui conseguimos ver o comando e o resultado!

O comando SELECT é usado para

fazer consultas.

```
18 •  SELECT * FROM ubs_uf; /*  Comando para mostrar todo o conteúdo de uma tabela  */
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

codigo_uf	unidade_da_federacao	uf
11	Rondônia	RO
12	Acre	AC
13	Amazonas	AM
14	Roraima	RR
15	Pará	PA
16	Amapá	AP
17	Tocantins	TO

ubs_uf 120 ×

E também a saída que o workbench dá.

```
186  16:55:00  SELECT data_ubs.uf AS UF, ubs_uf.unidade_da_federacao AS ESTADO,  (SELECT COUNT(CNES)  ...  27 row(s) returned
```

Fazendo as consultas dos dados

Também existe a possibilidade de usarmos filtros para trazer informações específicas como por exemplo:

```
22 •  SELECT COUNT(*) as TOTAL_DE_LINHAS FROM ubs_uf; /*  Comando para contar a  quantidade de linhas dentro de uma tabela  */
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TOTAL_DE_LINHAS
27

```
67 •  SELECT COUNT(*) as TOTAL_UBS_NORDESTE FROM data_ubs WHERE uf BETWEEN 21 AND 29;
68    /*  Faz a contagem de todas as linhas cadastradas que tenham uf=21;22;23;24;25;26;27;28 e 29  */
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TOTAL_UBS_NORDESTE
15463

```
34 •  SELECT * FROM ubs_uf where unidade_da_federacao = 'Rondônia';
```

Result Grid | Filter Rows: | Edit: | Export/Import:

codigo_uf	unidade_da_federacao	uf
11	Rondônia	RO
NULL	NULL	NULL

Outros comandos

Além dos comandos que vimos até agora existem vários outros, exemplo:

```
11 • INSERT INTO ubs_uf (
12     codigo_uf,
13     unidade_da_federacao,
14     uf) VALUES (52, 'Goiás', 'GO');
15     /* Comando para inserir dados manualmente, necessário informar o título da coluna e depois passar os valores */
```

```
17 • UPDATE ubs_uf SET uf='SP' WHERE codigo_uf=35;
18     /*  Comando para atualizar um registro especifico usando o WHERE  */
```

```
146 • ALTER TABLE ubs_uf ADD PRIMARY KEY (codigo_uf);
147     /*  Comando para adicionar um campo como chave primária  */
```

```
30 • TRUNCATE TABLE data_ubs;
31     /* Apaga todas as linhas de uma tabela */
```

```
149 • ALTER TABLE ubs_uf
150     CHANGE uf CIDADE int;
151     /* Comando para alterar o nome e o tipo de dado */
```

```
40 • DELETE FROM ubs_uf WHERE codigo_uf='52';
41     /*  Comando para apagar uma linha que tenha um conteúdo especifico usando o where  */
```

```
33 • DROP TABLE data_ubs;
34     /*  Comando para apagar uma tabela inteira  */
```

```
153 • ALTER TABLE ubs_uf
154     RENAME COLUMN codigo_uf TO codigo_uff;
155     /* Comando para renomear o nome da coluna */
```

Respondendo as afirmações

Após concluir todo o ambiente, ou seja: criar o banco, criar as tabelas e importar o conteúdo para as tabelas, podemos começar a responder as afirmações.

Hipótese - 1

O estado de São Paulo (SP) possui um número de UBSs maior que o somatório de todas as UBSs dos estados da região nordeste.

Conseguimos verificar de algumas formas:

```
86 •  SELECT COUNT(*) AS TOTAL_UBS_SP FROM data_ubs WHERE UF=35;
87    /*  Faz a contagem de quantas linhas cadastradas que tenham uf=35 (São Paulo)  */
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TOTAL_UBS_SP
5031

```
89 •  SELECT COUNT(*) AS TOTAL_UBS_NORDESTE FROM data_ubs WHERE uf BETWEEN 21 AND 29;
90    /*  Faz a contagem de todas as linhas cadastradas que tenham uf=21;22;23;24;25;26;27;28 e 29  */
```

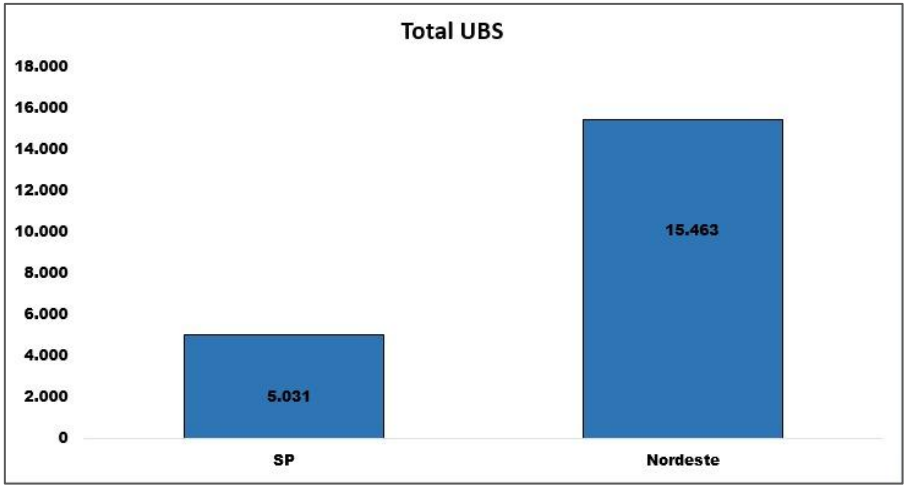
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TOTAL_UBS_NORDESTE
15463

Respondendo as afirmações

Resposta final da Hipótese 1:

```
93 •  SELECT uf.uf AS Estado_ou_Região, count(dt.CNES) AS Total_UBS
94    FROM data_ubs AS dt
95    INNER JOIN ubs_uf AS uf
96    ON uf.codigo_uf = dt.uf
97    WHERE uf.uf = 'sp'
98    UNION
99    SELECT 'Nordeste', SUM(Total_UBS) FROM (SELECT uf.uf, count(dt.CNES) AS Total_UBS
100   FROM data_ubs AS dt
101   INNER JOIN ubs_uf AS uf
102   ON uf.codigo_uf = dt.uf
103   WHERE dt.uf BETWEEN 21 AND 29
104   group by UF) AS Q;
```



A partir dessa consulta conseguimos trazer o total de Unidades Básicas de Saúde do estado de São Paulo e na segunda linha o total de todo o Nordeste e conseguimos perceber que a afirmação é <u>falsa</u>.

Result Grid | Filter Rows:

Estado_ou_Região	Total_UBS
SP	5031
Nordeste	15463

Respondendo as afirmações

Hipótese - 2

A maioria das UBSs, nos respectivos estados, estão localizadas nas regiões centrais das cidades (use como base os bairros intitulados como CENTRO).

Conseguimos verificar assim:

```
115 •  SELECT uf AS Estado, COUNT(*) AS total_ubs_centro FROM data_ubs WHERE UF = 11 AND bairro = 'centro';
116    /* Consulta a sigla do estado e retorna com a condição de ser 11 que é Rondônia e ser bairro centro */
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Estado	total_ubs_centro
11	41

```
118 •  SELECT uf AS CODIGO_UF, COUNT(*) AS total_ubs_estado FROM data_ubs WHERE UF = 11;
119    /* Consulta a sigla do estado e retorna o total de ubs do estado */
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

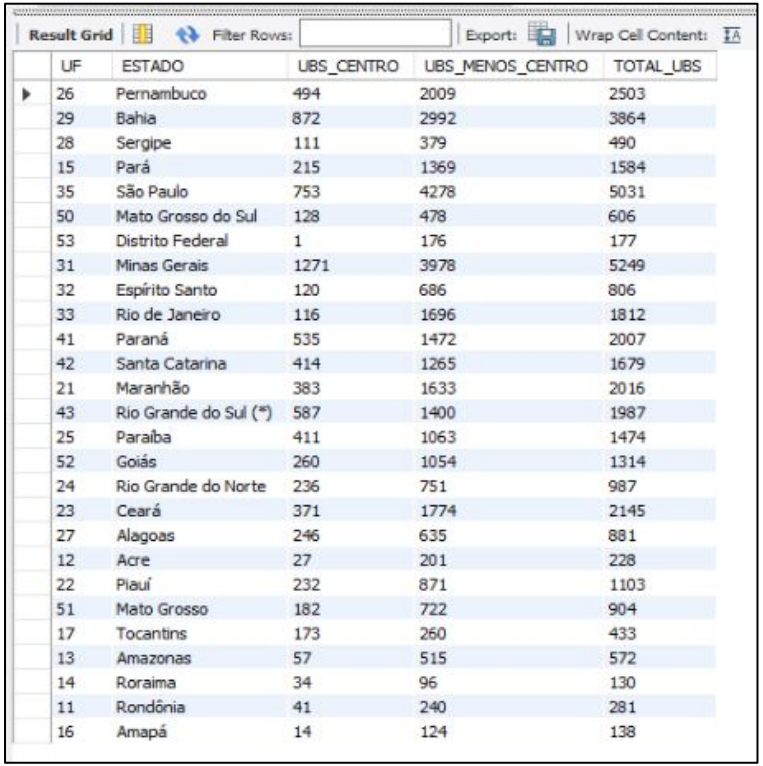
CODIGO_UF	total_ubs_estado
11	281

Respondendo as afirmações

Resposta final da afirmação 2:

```sql
SELECT data_ubs.uf AS UF, ubs_uf.unidade_da_federacao AS ESTADO,
       (SELECT COUNT(CNES)
        FROM data_ubs
        WHERE data_ubs.bairro = "centro" AND data_ubs.uf = ubs_uf.codigo_uf) AS UBS_CENTRO,
       COUNT(data_ubs.CNES) AS UBS_MENOS_CENTRO,
       (SELECT COUNT(CNES)
        FROM data_ubs
        WHERE data_ubs.uf = ubs_uf.codigo_uf) AS TOTAL_UBS
FROM data_ubs
INNER JOIN ubs_uf ON data_ubs.uf = ubs_uf.codigo_uf
WHERE data_ubs.bairro <> "centro"
GROUP BY data_ubs.uf, ubs_uf.codigo_uf;
```

A partir dessa consulta conseguimos visualizar todos os 27 estados, o número total de UBS que está cadastrada com o bairro centro, o total de USB sem contar o bairro centro e o total de UBS por estado.



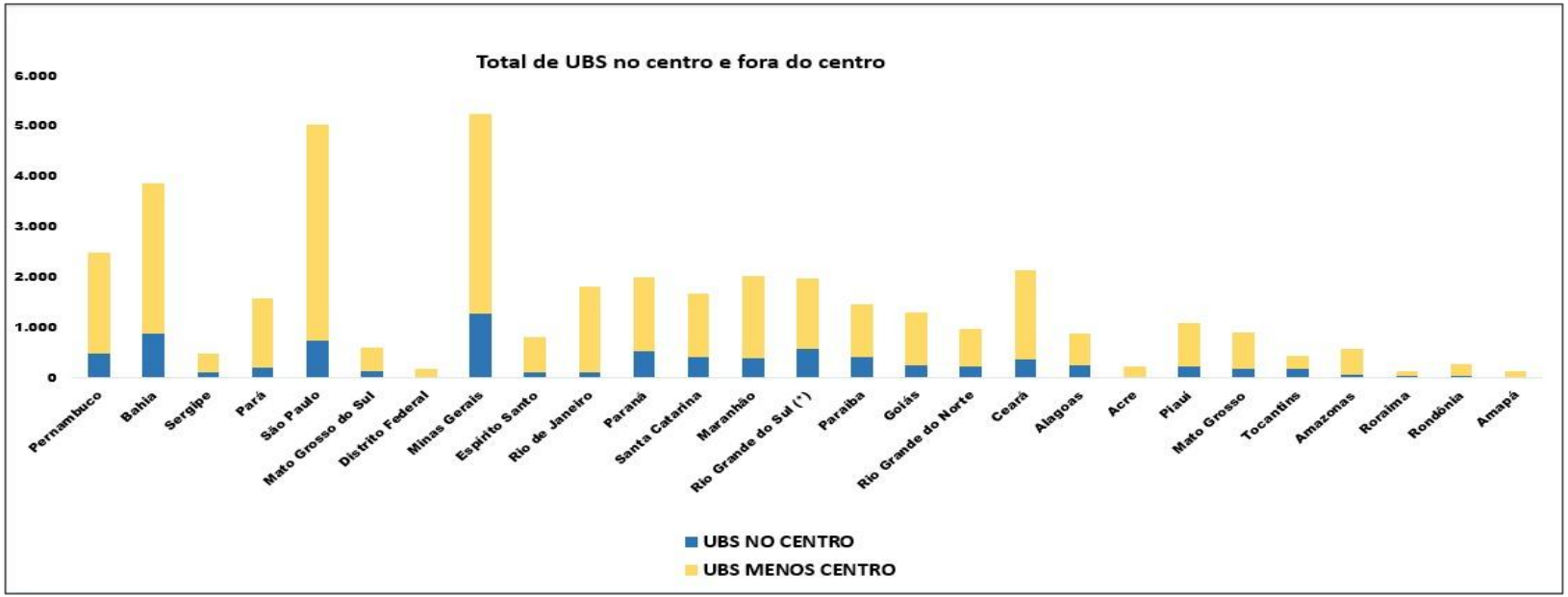
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

UF	ESTADO	UBS_CENTRO	UBS_MENOS_CENTRO	TOTAL_UBS
26	Pernambuco	494	2009	2503
29	Bahia	872	2992	3864
28	Sergipe	111	379	490
15	Pará	215	1369	1584
35	São Paulo	753	4278	5031
50	Mato Grosso do Sul	128	478	606
53	Distrito Federal	1	176	177
31	Minas Gerais	1271	3978	5249
32	Espírito Santo	120	686	806
33	Rio de Janeiro	116	1696	1812
41	Paraná	535	1472	2007
42	Santa Catarina	414	1265	1679
21	Maranhão	383	1633	2016
43	Rio Grande do Sul (*)	587	1400	1987
25	Paraíba	411	1063	1474
52	Goiás	260	1054	1314
24	Rio Grande do Norte	236	751	987
23	Ceará	371	1774	2145
27	Alagoas	246	635	881
12	Acre	27	201	228
22	Piauí	232	871	1103
51	Mato Grosso	182	722	904
17	Tocantins	173	260	433
13	Amazonas	57	515	572
14	Roraima	34	96	130
11	Rondônia	41	240	281
16	Amapá	14	124	138

Respondendo as afirmações

Melhorando a visualização dos dados:

Com esse gráfico que conseguimos a partir dos dados da consulta no MySQL verificar com mais clareza que a maioria das UBSs em relação ao total de cada estado não está localizado nos centros das cidades de seus respectivos estados. A hipótese 2 também é falsa.



Respondendo as afirmações

DESAFIO:

Observe nos dados das UBSs que existe uma coluna intitulada “IBGE”. Crie um relatório que liste todas as UBS de um respectivo município/distrito/subdistrito.

Analisando o desafio percebi a necessidade de fazer uma outra tabela que ajude a trazer o nome da cidade, ao invés de procurar pela sigla IBGE, com essa outra tabela fica mais fácil de entender a proposta.

```
54  CREATE TABLE data_municipios (
55      UF_MUNICIPIO VARCHAR(100) NOT NULL,
56      IBGE INT PRIMARY KEY NOT NULL,
57      IBGE7 INT NOT NULL,
58      UF CHAR(2) NOT NULL,
59      MUNICIPIO VARCHAR(100) NOT NULL,
60      REGIAO VARCHAR(50) NOT NULL,
61      POPULACAO_2010 LONG NOT NULL,
62      PORTE VARCHAR(50) NOT NULL,
63      CAPITAL VARCHAR(30)
64  );
```

Respondendo as afirmações

DESAFIO:

Assim seria o resultado pesquisando por fora o código do IBGE para filtrar a pesquisa, que ficaria mais difícil, porém também seria uma opção, mas quis ir mais a fundo e fazer a pesquisa pelo nome da cidade.

```
169 •  SELECT uf.uf as Estado, dt.IBGE as Cod_cidade, dt.NOME as Nome_da_UBS
170    FROM data_ubs AS dt
171    INNER JOIN ubs_uf AS uf
172    ON uf.codigo_uf = dt.uf
173    WHERE dt.IBGE = 355620;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Estado	Cod_cidade	Nome_da_UBS
SP	355620	CENTRO DE SAUDE VILA SANTANA
SP	355620	UBS REFORMA
SP	355620	UBS IMPERIAL
SP	355620	UBS BOM RETIRO
SP	355620	UBS PINHEIROS
SP	355620	UBS PARAISO
SP	355620	UBS MARACANA
SP	355620	UBS SAO MARCOS

Result 203

Output

Action Output

#	Time	Action	Message
301	10:14:26	SELECT uf.uf as Estado, dt.IBGE as Cod_cidade, dt....	14 row(s) returned

Respondendo as afirmações

DESAFIO:

Essa foi a forma mais amigável

tanto para filtrar e ler os resultados.

```
178 •  SELECT du.UF AS Cod_Estado, du.IBGE AS Cod_IBGE_cidade, dt.MUNICIPIO AS CIDADE, du.CNES AS CNES, du.NOME
179    FROM data_municipios AS dt
180    INNER JOIN data_ubs AS du
181    ON dt.IBGE = du.IBGE
182    WHERE dt.MUNICIPIO = 'Rio de janeiro';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Cod_Estado	Cod_IBGE_cidade	CIDADE	CNES	NOME
33	330455	Rio de Janeiro	30260	LONGEVICARE
33	330455	Rio de Janeiro	55344	CENTRO MEDICO SUBURBANA
33	330455	Rio de Janeiro	189200	SMS CF ENGENHEIRO SANITARISTA PAULO D AGUILA AP 33
33	330455	Rio de Janeiro	193089	SMS CF CRISTIANI VIEIRA PINHO AP 51
33	330455	Rio de Janeiro	199338	SMS CF ADV MARIO PIRES DA SILVA AP 33
33	330455	Rio de Janeiro	214949	SMS CF LOURIVAL FRANCISCO DE OLIVEIRA AP 40

Output

Action Output

#	Time	Action	Message
299	10:05:57	SELECT du.UF as Cod_Estado, du.IBGE as Cod_IB...	66 row(s) returned

FIM