Exercício no QGis: Agregação de dados dos hospitais para os distritos em BD GeoPackage

GeoPackage é um banco de dados (BD), derivado do BD SQLite, contido num arquivo com a extensão: **.gpkg** . Isso permite a execução de queries diretamente no QGis, assim como em outros softwares, sem a necessidade do uso de sistemas de bancos de dados de maior complexidade, como o PostgreSQL. Ou de BDs que tenham de ser acessados fora do ambiente operacional do QGis, como visto para os exemplos com o uso do Libre Office Base e o ms-Access.

Neste exercício será explorado o uso do GeoPackage para a agregação dos dados dos hospitais públicos e privados; a associação desses dados aos distritos nos quais esses hospitais se inserem, além de outros exemplos.

Criar um novo GeoPackage para os distritos do município de São Paulo

Carregar os dados no QGis

Menu Layer > Add Layer > Vector Layer

Selecionar o shape file CEMDistMSP criado anteriormente, a partir do CEMDist

O sistema de coordenadas já deve ser a projeção UTM23S sobre o Datum SIRGAS 2000, conforme indicado no arquivo CEMDistMSP.prj

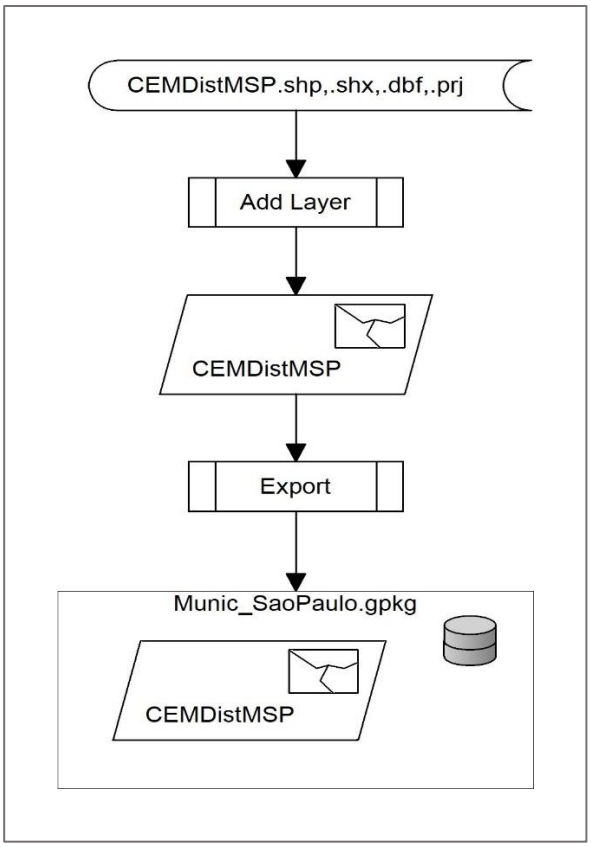
Encoding: **windows-1252**

Após clicar no botão Add e carregar o shapefile, verificar que o CRS corrente é o **EPSG:31983** (SIRGAS 2000/UTM 23S)

Abrir a tabela de atributos do *layer* e verificar que a acentuação esteja correta nos nomes dos distritos da coluna **nome**

Criar GeoPackage contendo os distritos carregados (Figura 1)

Figura 1 – árvore de expressões para a criação de novo GeoPackage a partir do arquivo de distritos do MSP



Clicar com o botão direito do mouse no *layer* CEMDistMSP e selecionar Export > Save Features As...

Na caixa de diálogo, selecionar / digitar:

Format: GeoPackage

FileName: navegar até a pasta desejada e digitar **Munic_SaoPaulo**, seguido do botão Save

LayerName: CEMDistMSP

CRS: EPSG:31983 - SIRGAS 2000/UTM 23S

Encoding: UTF-8 (default e fixo!)

Add saved file to map: [**v**]

Select fields to export and export options:

[ ] id

manter os demais

Geometry type: Automatic

Extent: [**v**] Current Layer

Layer options:

Description: Distritos do MSP

FID: fid (default)

Geometry_Name: geom (default)

Identifier: (default)

Spatial_Index: Yes

Clicar no botão OK para terminar

Remover o *layer* do shapefile, uma vez que foi gerada uma cópia dos distritos no novo GeoPackage.

Salvar o Projeto

Abrir tabela de atributes do novo *layer* e verificar se acentuação está correta.

Observar que, além da coluna **id**, oriunda do shapefile, foi acrescentada a coluna **fid**

Verificar também em **Properties** desse *layer* se o sistema de coordenadas está correto

Conectar ao GeoPackage recém-criado

No painel **Browser**, clicar com o botão direito em **Geopackage** e selecionar **New Connection...**

Selecionar o arquivo Munic_SaoPaulo.gpkg

Clicar com o botão direito do mouse no layer CEMDistMSP e selecionar Add Selected Layer(s) to Canvas

Remover o *layer* anteriormente referenciada, uma vez que se trata dos mesmos dados presentes no GeoPackage.

Salvar o Projeto

Acrescentar a tabela Hospitals ao GeoPackage recém-criado

Carregar Hospitals no QGis (apenas para verificar conteúdo)

Menu Layer > Add Layer > Add Delimited Text Layer…

Selecionar o arquivo Hospitals.csv

Layer name: HOSPITALS

Encoding: windows-1252

File format:

(*) Custom delimiters [v] Semicolon

Records and fields options:

Number of header lines to discard: 0 (default)

Geometry definition:

(*) No geometry (attribute only table)

Clicar no Botão Add

Abrir a tabela de atributos e verificar os dados – não há caracteres acentuados neste arquivo.

Os tipos de dados de leitos, médicos, enfermeiro(a)s, ambulâncias, etc. podem ser interpretados genericamente pelo QGis como sendo do tipo caracteres (string), por isso, pode-se criar um arquivo auxiliar, de mesmo nome do CSV, mas com terminação CSVT, antes de sua inserção no BD GeoPackage.

O conteúdo desse arquivo CSVT, corresponde ao cabeçalho do arquivo CSV, ou seja, para:

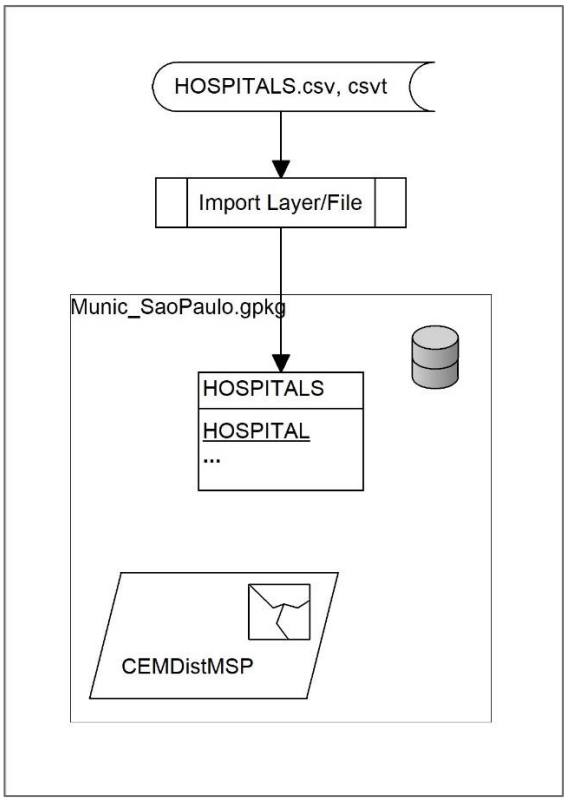
HOSPITAL;DISTRITO;NUM_LEITOS;NUM_ENFERM;NUM_MEDICO;
NUM_AMBULA;PUBLICO;NUM_P_MASC,NUM_P_FEM;TOTAL_PAC;COL_TESTE

Tem-se:

String,String,Integer, Integer, Integer,
Integer,String,Integer,Integer,Integer,Integer

Carregar Hospitals no GeoPackage (Figura 2)

Figura 2 – árvore de expressões para inclusão da tabela de hospitais dos distritos no GeoPackage criado anteriormente



No menu Database selecionar > DB Manager > GeoPackage > Munic_SaoPaulo

Clicar no botão Import Layer/File...

Input: selecionar o arquivo HOSPITALS.csv

Output table

Table: HOSPITALS

Options

Primary key: HOSPITAL

Encoding: UTF-8 (default) – neste caso não há problema por não haver caracteres acentuados

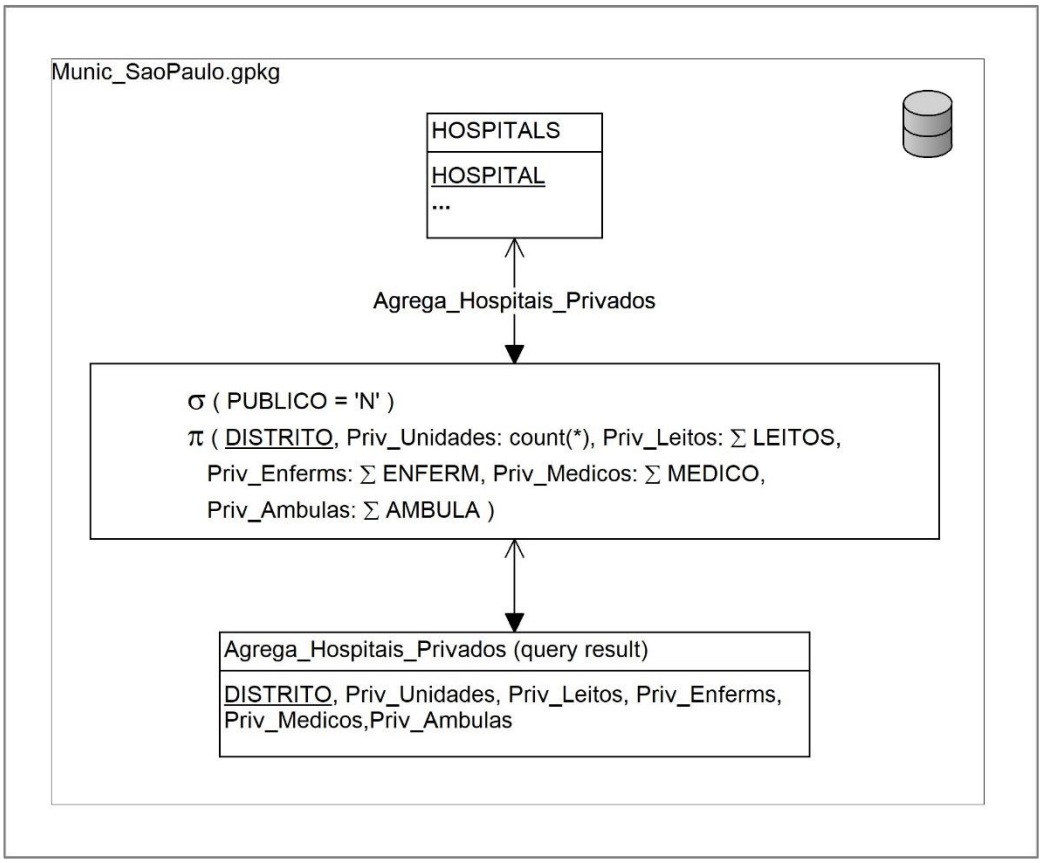
Clicar no botão OK

Observar na caixa de diálogo DB Manager que a tabela foi acrescida ao GeoPackage – em seguida fechar essa caixa e retornar à tela principal do QGis.

Observar que HOSPITALS também aparece no Browser sob o GeoPackage recém-conectado

Agregar os dados dos Hospitais <u>Privados</u> por distrito (Figura 3)

Figura 3 – árvore de expressões para a agregação dos dados dos hospitais privados no BD GeoPackage



No menu Database selecionar > DB Manager > GeoPackage > Munic_SaoPaulo

Clicar no botão **SQL Window** (ícone de uma chave inglesa sobre uma tabela)

Na aba Query que se abre, clicar no botão **Query Builder** (ícone SQL sobre rolo de papel)

Na caixa de diálogo SQL Query Builder, na **aba Data**, clicar no botão **Tables** e selecionar HOSPITALS

Observar, na parte esquerda, o termo “HOSPITALS”, no campo **Tables**

Clicar no botão **Columns** e selecionar as colunas abaixo indicadas, as quais vão se somando, na parte da esquerda, o campo **Columns** acima de Tables:

"HOSPITALS"."DISTRITO",

"HOSPITALS"."NUM_LEITOS",

"HOSPITALS"."NUM_ENFERM",

"HOSPITALS"."NUM_MEDICO",

"HOSPITALS"."NUM_AMBULA"

Copie manualmente a primeira linha dessa lista para o campo **Group By**, como o abaixo indicado:

"HOSPITALS"."DISTRITO"

Altere o conteúdo do campo **Columns** de modo a somar e dar nomes aos dados consolidados pela agregação, conforme indicado abaixo

"HOSPITALS"."DISTRITO",

Count(*) as Priv_Unidades,

sum("HOSPITALS"."NUM_LEITOS"**) as Priv_Leitos**,

sum("HOSPITALS"."NUM_ENFERM"**) as Priv_Enferms**,

sum("HOSPITALS"."NUM_MEDICO"**) as Priv_Medicos**,

sum("HOSPITALS"."NUM_AMBULA"**) as Priv_Ambulas**

No campo **Where**, preencha com "HOSPITALS"."PUBLICO" = 'N'

O valor 'N' é extraído das opções de valores para a coluna "HOSPITALS"."PUBLICO" na **aba Columns' values**

Clicar em OK

De volta à caixa de diálogo DB Manager, preencher o campo Name com Agrega_Hospitais_Privados e, em seguida, clicar no botão Save para dar nome à query.

Para criar a nova tabela virtual, a partir do resultado da Query, selecionar [**v**] Load as New Layer, e configurar como segue:

[**v**] Column with unique values: DISTRITO

[ ] Geometry column

Layer name (prefix): Agrega_Hospitais_Privados

Clicar no botão **Execute**

Clicar no botão **Load**

Importante: A query não está armazenada no GeoPackage, mas seu resultado é visível na lista de layers do QGis. Se fechar o QGis sem salvar o arquivo do projeto no qual se está trabalhando, as queries são perdidas.

Há 78 distritos com hospitais privados.

Salvar o Projeto.

Agregar os dados dos Hospitais <u>Publicos</u> por distrito

Com a query anterior ainda aberta na caixa de diálogo DB Manager

Altere o valor do campo **Name**, de: Agrega_Hospitais_Privados para: Agrega_Hospitais_**Publicos**

Edite manualmente o termo WHERE "HOSPITALS"."PUBLICO" = 'N' para WHERE "HOSPITALS"."PUBLICO" = 'S'

Também substituir o prefixo Priv_ por **Publ_** no campo **Columns**

"HOSPITALS"."DISTRITO",

Count(*) as **Publ**_Unidades,

sum("HOSPITALS"."NUM_LEITOS") as **Publ**_Leitos,

sum("HOSPITALS"."NUM_ENFERM") as **Publ**_Enferms,

sum("HOSPITALS"."NUM_MEDICO") as **Publ**_Medicos,

sum("HOSPITALS"."NUM_AMBULA") as **Publ**_Ambulas

Para criar a novo *layer*, a partir do resultado da Query, selecionar
[**v**] Load as New Layer, e configurar como segue:

[**v**] Column with unique values: DISTRITO

[ ] Geometry column

Layer name (prefix): Agrega_Hospitais_Publicos

Clicar no botão **Execute**

Clicar no botão **Load**

A query está armazenada no GeoPackage e seu resultado é visível na lista de layers do QGis

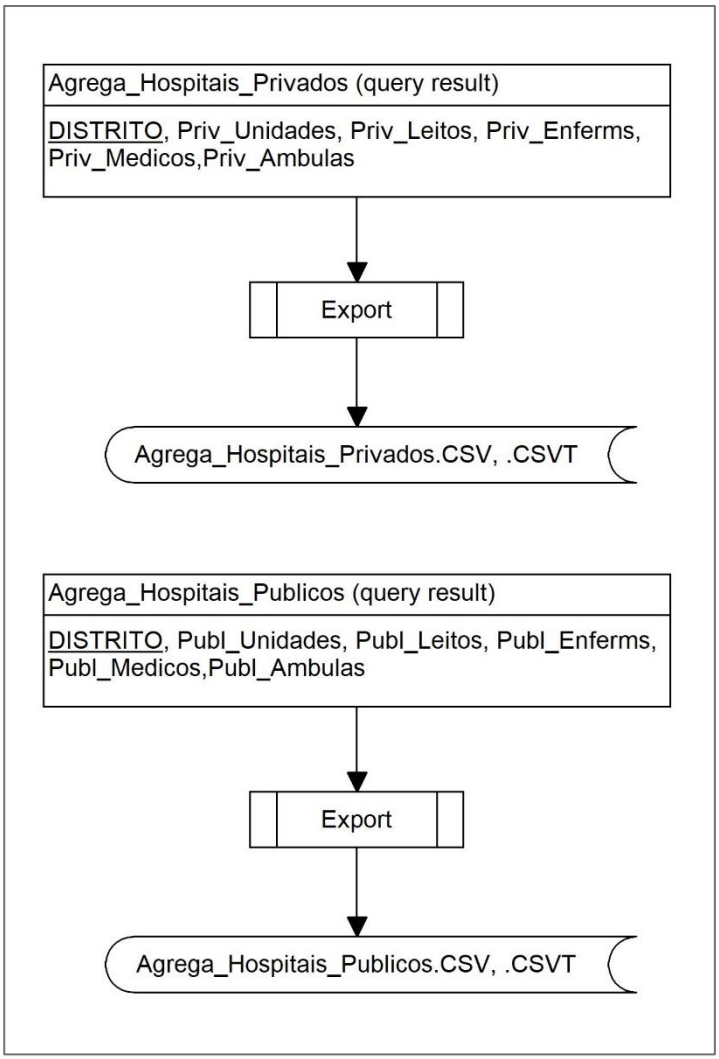
Há 46 distritos com hospitais públicos.

Importar os dados agregados nas queries para o GeoPackage

A forma encontrada para criar tabelas dos dados agregados, no BD GeoPackage, em substituição às queries armazenadas nesse BD, foi de exportar seus resultados para arquivos CSV (Figura 4), para depois importar esses arquivos no BD como tabelas (Figura 5).

Exportar cada "*layer*" para um arquivo CSV

Figura 4 – árvore de expressões para a exportação das tabelas com os dados agregados para arquivos CSV



Clicar com o botão direito do mouse no *layer*: Agrega_Hospitais_Publicos e selecionar Export > Save Features As...

Format: Coma Separated Value [CSV]

File Name: navegar até a pasta desejada e acrescentar o nome do arquivo CSV a ser gerado: Agrega_Hospitais_Publicos

Encoding: UTF-8

[ ] Add saved file to map

Geometry:

Geometry Type: **No geometry**

Layer Options:

Create CSVT: **Yes**
Geometry: <default>
Line format: <default>
Separator: **SEMICOLON**
String_Quoting: **IF_NEEDED**
Write BOM: No

Clicar em OK para gravar o arquivo

Fazer o mesmo para o Agrega_Hospitais_Publicos

Verificar na pasta escolhida se os arquivos CSV e CSVT foram gerados, assim como a integridade de seus conteúdos.

Por exemplo, ao abrir o arquivo CSVT, se os tipos dos dados estão corretos na versão do QGis usada na montagem deste exercício estava errado:

String,String,String,String,String,String

O correto é, e deve ser alterado para:

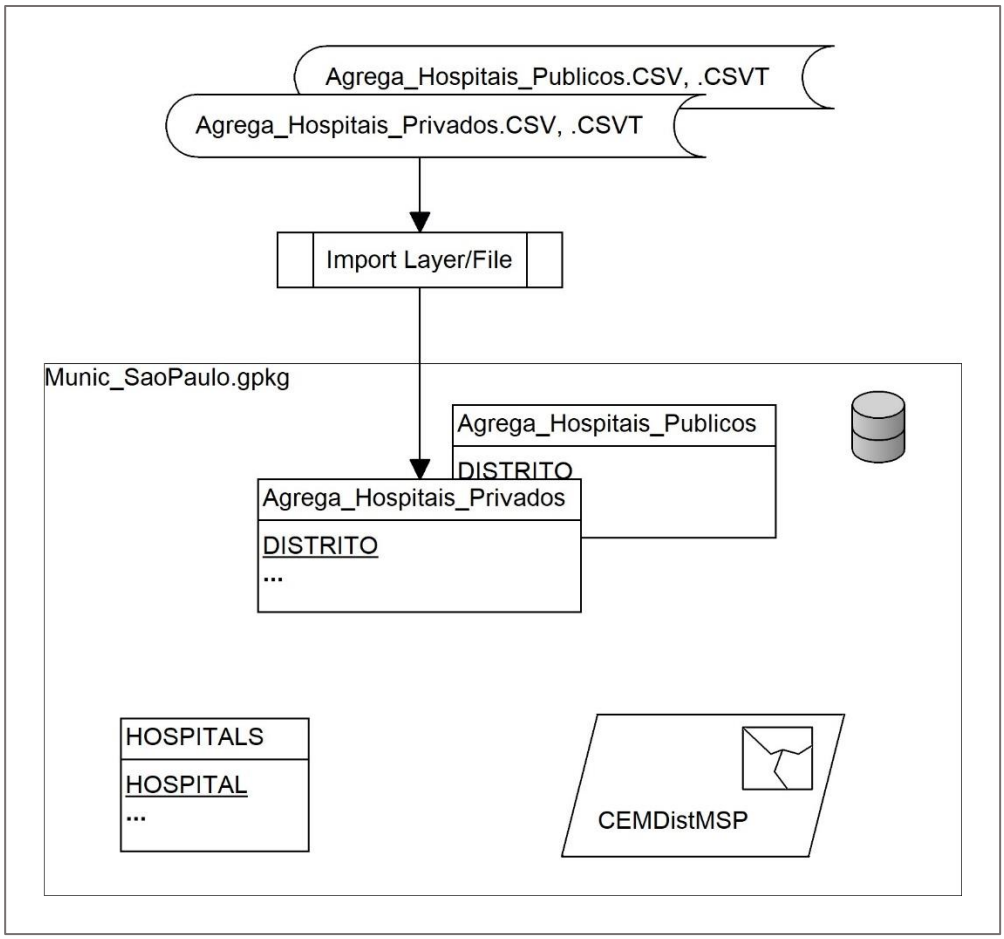
String,Integer,Integer,Integer,Integer,Integer

Remover os layers correspondentes às queries, cujos resultados acabam de ter sido salvos em arquivos CSV.

Salvar o Projeto

Carregar os arquivos CSV no GeoPackage

Figura 5 – árvore de expressões do carregamento dos arquivos CSV no BD GeoPackage



No menu Database selecionar > DB Manager > GeoPackage > Munic_SaoPaulo

Clicar no botão Import Layer/File...

Input: selecionar o arquivo Agrega_Hospitais_Publicos.csv

Output table:

Table: Agrega_Hospitais_Publicos

Options

Primary key: **DISTRITO**

Fazer o mesmo com o arquivo Agrega_Hospitais_Privados

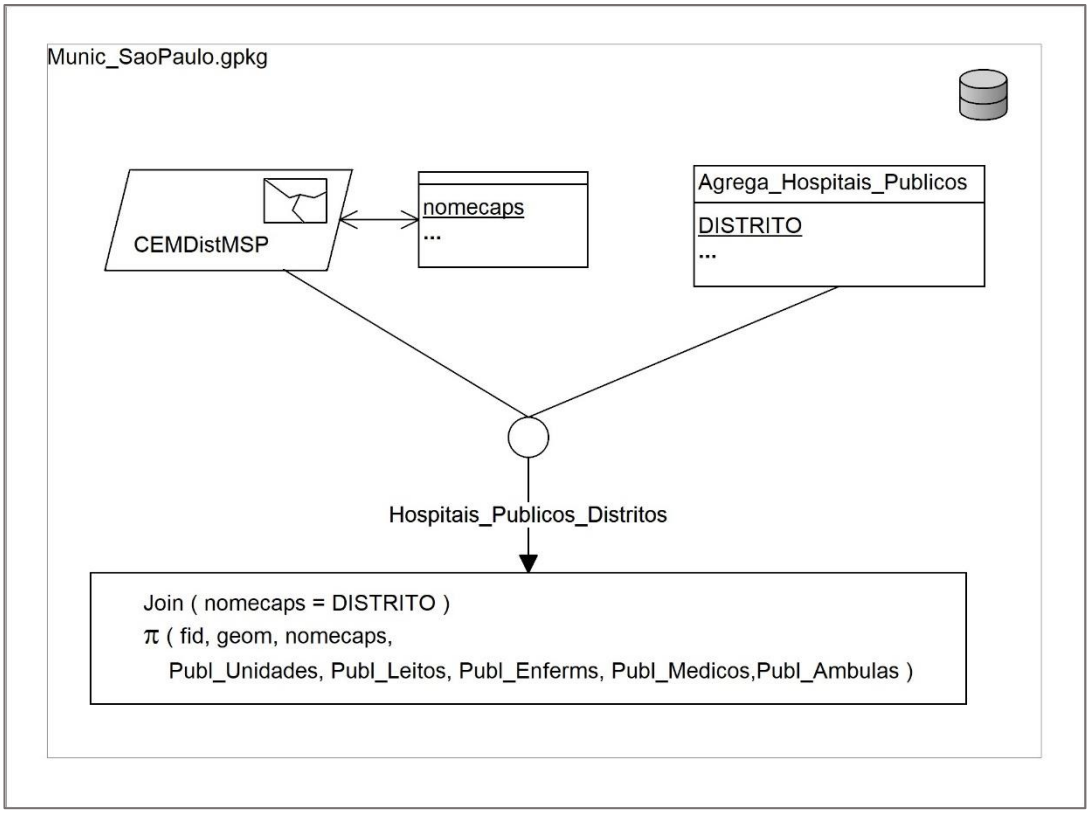
Salvar o Projeto

Join dos distritos com as tabelas de dados agregados

A associação dos dados dos hospitais públicos e privados aos respectivos distritos (em cujos limites eles estão inseridos) requer o cruzamento dos dados das respectivas tabelas.

Join dos distritos com a agregação dos dados dos hospitais públicos

Figura 6 – árvore de expressões do cruzamento dos distritos com os dados agregados dos hospitais públicos neles inseridos



A Figura 6 ilustra tal cruzamento, na forma de árvore de expressões. Observar que a geometria resultante está indicada indiretamente pelo atributo **geom** na lista do operador *Project* (π) da *query*.

No menu Database selecionar > DB Manager > GeoPackage > Munic_SaoPaulo

Se necessário selecionar a opção **Re-connect** com o botão esquerdo do mouse

Clicar no botão **SQL Window** (ícone de uma chave inglesa sobre uma tabela)

Na aba Query que se abre, clicar no botão **Query Builder** (ícone SQL sobre rolo de papel)

Na caixa de diálogo SQL Query Builder, na **aba Data**, clicar no botão **Tables** e selecionar (o layer) **CEMDistMSP** e (a tabela) **Agrega_Hospitais_Publicos**

Observar o preenchimento do campo Tables com:
"CEMDistMSP", "Agrega_Hospitais_Publicos"

Clicar no campo **Where** e montar a condição do Join conforme indicado abaixo:
"CEMDistMSP"."nomecaps" **=** "Agrega_Hospitais_Publicos"."DISTRITO"

No campo columns, selecionar os atributos essenciais da tabela **CEMDistMSP**:

"CEMDistMSP"."geom",

"CEMDistMSP"."nomecaps"

O atributo "CEMDistMSP"."geom" corresponde à representação geométrica de cada distrito a ser selecionado na query.

Adicione, à lista, as colunas essenciais da tabela **Agrega_Hospitais_Publicos**:

"Agrega_Hospitais_Publicos"."Publ_Unidades",

"Agrega_Hospitais_Publicos"."Publ_Leitos",

"Agrega_Hospitais_Publicos"."Publ_Enferms",

"Agrega_Hospitais_Publicos"."Publ_Medicos",

"Agrega_Hospitais_Publicos"."Publ_Ambulas"

Clique em OK para voltar ao DB Manager

De volta à caixa de diálogo DB Manager, preencher o campo Name com **Hospitais_Publicos_Distritos** e, em seguida, clicar no botão Save para dar nome à query

Para criar a novo *layer*, a partir do resultado da Query, selecionar
[**v**] Load as New Layer, e configurar como segue:

[**v**] Column with unique values: **nomecaps**

[**v**] Geometry column: **geom**

Layer name (prefix): Hospitais_Publicos_Distritos

Clicar no botão **Execute**

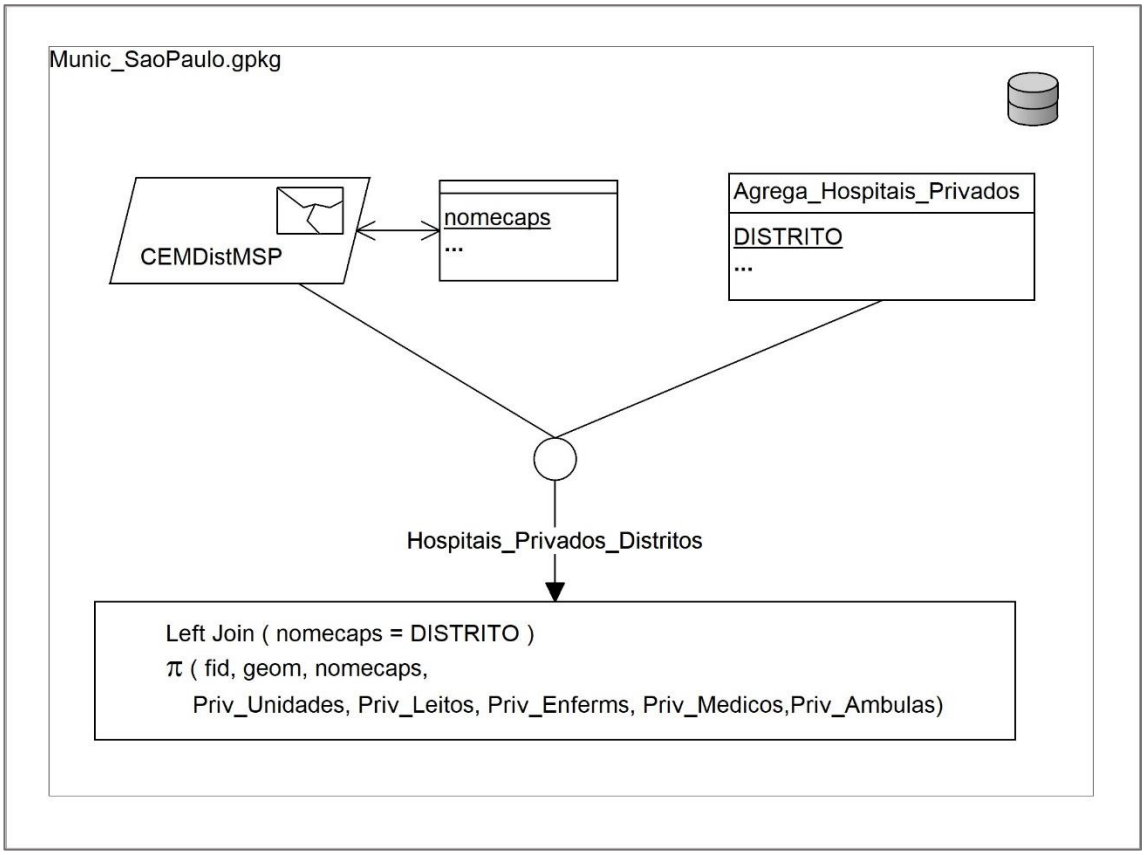
Clicar no botão **Load**

A query está armazenada no GeoPackage e seu resultado é visível na lista de layers do QGis

Salvar o Projeto

Observar que os distritos que não possuem hospitais públicos em seus territórios foram omitidos do resultado, mormente sua representação espacial. Isso poderia ser evitado se, ao invés do *Inner Join* (implícito na query acima) fosse usado um *Outer Join* (Figura 7).

Figura 7 - árvore de expressões do cruzamento dos distritos com os dados agregados dos hospitais privados neles inseridos



A interface do Query Builder sugere que apenas se possa executar queries com *inner join* implícitos, mas pode-se executar queries com clausulas explícitas de inner ou outer joins.

No caso, criar a query **Hospitais_Privados_Distritos**, com a seguinte expressão:

```
SELECT "CEMDistMSP"."geom",
"CEMDistMSP"."nomecaps",
"Agrega_Hospitais_Privados"."Priv_Unidades",
"Agrega_Hospitais_Privados"."Priv_Leitos",
"Agrega_Hospitais_Privados"."Priv_Enferms",
"Agrega_Hospitais_Privados"."Priv_Medicos",
"Agrega_Hospitais_Privados"."Priv_Ambulas"
FROM "CEMDistMSP" left join "Agrega_Hospitais_Privados"
   on "CEMDistMSP"."nomecaps" = "Agrega_Hospitais_Privados"."DISTRITO"
```

tendo o cuidado de fornecer os demais parâmetros:

[**v**] Column with unique values: **nomecaps**

[**v**] Geometry column: **geom**

Layer name (prefix): Hospitais_Privados_Distritos

Clicar no botão **Execute**

Clicar no botão **Load**

A query está armazenada no GeoPackage e seu resultado é visível na lista de layers do QGis. Verificar na lista de atributos do resultado que há diversos distritos sem dados agregados de hospitais (valores nulos).

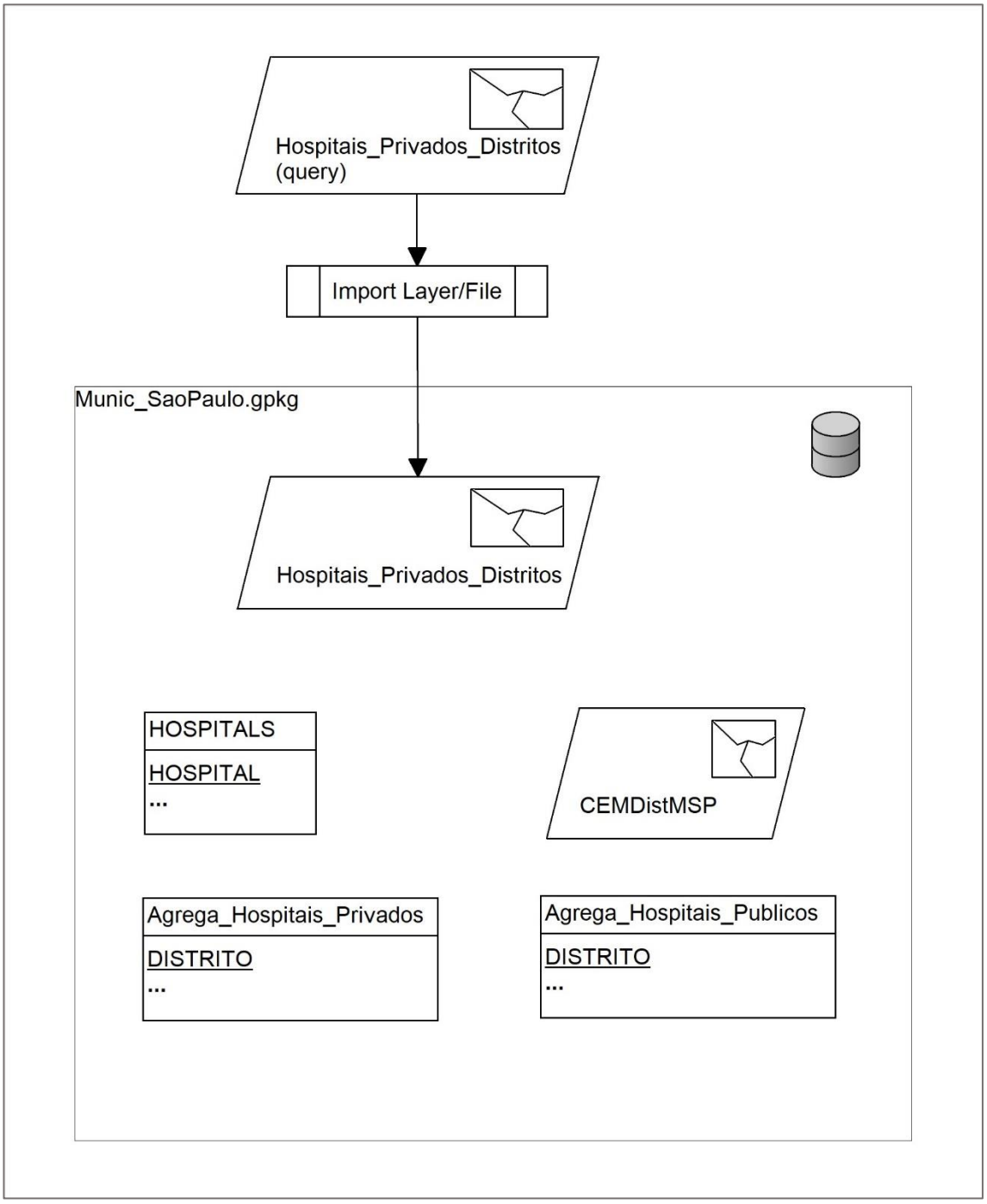
Salvar o Projeto

Uma vez executada a query acima, pode-se selecionar da lista e Deletar a query **Hospitais_Publicos_Distritos** assim como seu resultado das demais listas do QGis.

Ao invés de simplesmente repetir o processo para os dados agregados dos hospitais públicos, o que iria gerar uma cópia da geometria dos distritos, eles serão adicionados aos dados agregados dos hospitais públicos.

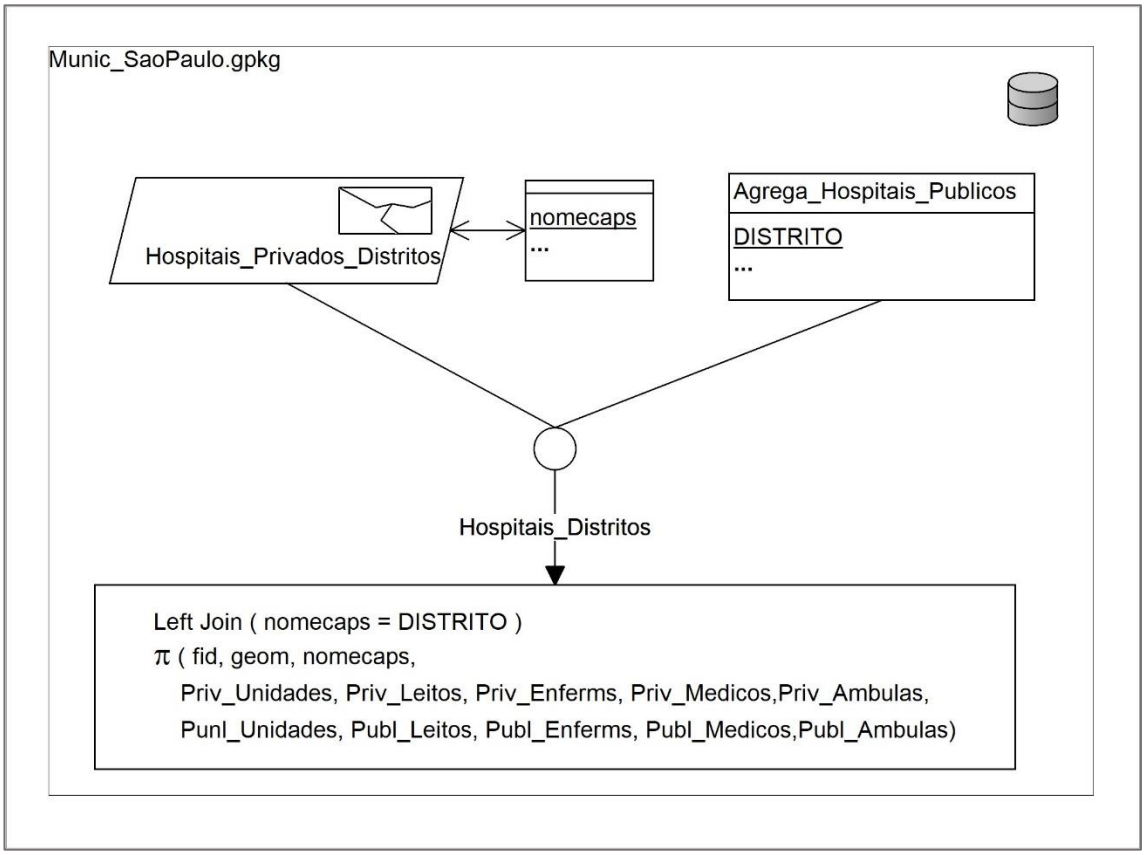
Mas, antes há de se carregar os dados da query **Hospitais_Privados_Distritos** como layer (features) no BD GeoPackage (Figura 8).

Figura 8 - árvore de expressões da importação da tabela de distritos
com dados agregados de hospitais públicos neles inseridos



A Figura 9 exibe o processamento para adicionar os dados agregados dos hospitais públicos ao layer (features) em uma nova query de nome **Hospitais_Distritos**.

Figura 9 - árvore de expressões do cruzamento dos distritos com os dados agregados também dos hospitais públicos neles inseridos



Para tanto, criar a query **Hospitais_Distritos**, com a seguinte expressão:

```
SELECT "Hospitais_Privados_Distritos"."geom",
"Hospitais_Privados_Distritos"."nomecaps",
"Hospitais_Privados_Distritos"."Priv_Unidades",
"Hospitais_Privados_Distritos"."Priv_Leitos",
"Hospitais_Privados_Distritos"."Priv_Enferms",
"Hospitais_Privados_Distritos"."Priv_Medicos",
"Hospitais_Privados_Distritos"."Priv_Ambulas",
"Agrega_Hospitais_Publicos"."Publ_Unidades"
"Agrega_Hospitais_Publicos"."Publ_Leitos",
"Agrega_Hospitais_Publicos"."Publ_Enferms",
"Agrega_Hospitais_Publicos"."Publ_Medicos",
"Agrega_Hospitais_Publicos"."Publ_Ambulas"
FROM "Hospitais_Privados_Distritos" left join "Agrega_Hospitais_Publicos"
  on "Hospitais_Privados_Distritos"."nomecaps" =
      "Agrega_Hospitais_Publicos"."DISTRITO"
```

tendo o cuidado de fornecer os demais parâmetros:

[**v**] Column with unique values: **nomecaps**

[**v**] Geometry column: **geom**

Layer name (prefix): Hospitais_Distritos

Clicar no botão **Execute**

Clicar no botão **Load**

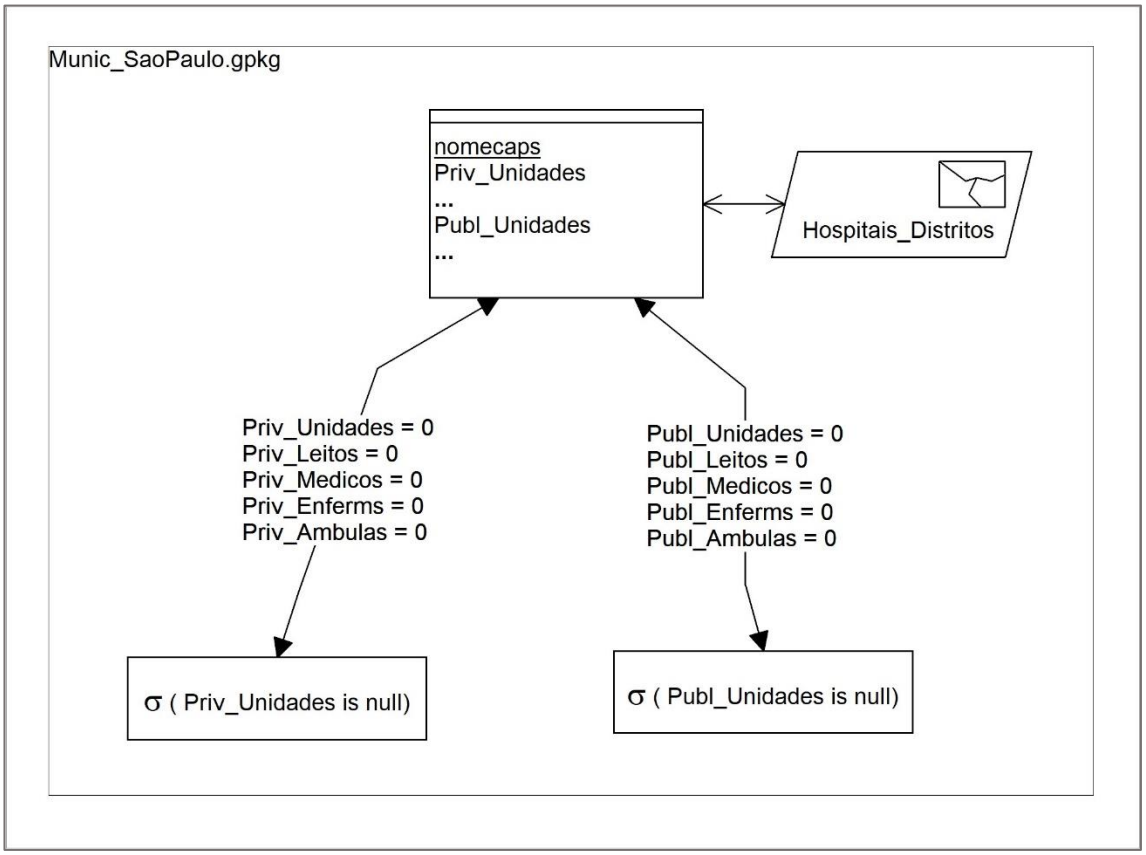
A query está armazenada no GeoPackage e seu resultado é visível na lista de layers do QGis. Verificar na lista de atributos do resultado que há diversos distritos sem dados agregados de hospitais (valores nulos).

Salvar o Projeto

Uma vez executada a query acima, pode-se selecionar da lista e Deletar a query **Hospitais_Privados_Distritos** assim como seu resultado das demais listas do QGis. O resultado da nova query, salvo como layer (features) no BD GeoPackage, contém os dados agregados, tanto dos hospitais públicos como dos privados.

Com o layer (features) Hospitais_Distritos salvo no BD GeoPackage Munic_SaoPaulo, pode-se gerar mapas temáticos indicativos do número de unidades, leitos, médicos, ambulâncias, etc. por distrito. Mas, a existência do valor *null* não é compreendida nesse processo e os distritos correspondentes não são exibidos.

Figura 10 - árvore de expressões da substituição de *null*s por zeros



Pode-se neste caso particular[1] substituir os *null*s por zeros, conforme indicado na Figura 10 e nas queries que seguem.

As queries correspondentes, são escritas diretamente na caixa de diálogo SQL Window (não no Query Builder):

```
update "Hospitais_Distritos"
set "Priv_Unidades" = 0,
"Priv_Leitos" = 0,
"Priv_Enferms" = 0,
"Priv_Medicos" = 0,
"Priv_Ambulas" = 0
WHERE "Priv_Unidades" is null
```

e

```
update "Hospitais_Distritos"
set "Publ_Unidades" = 0,
"Publ_Leitos" = 0,
"Publ_Enferms" = 0,
"Publ_Medicos" = 0,
"Publ_Ambulas" = 0
WHERE "Publ_Unidades" is null
```

Tal ajuste não poderia ser feito diretamente sobre a query, por isso os seus resultados terem sido salvos como layer (features) no BD GeoPackage.

[1] *Null* não necessariamente significa zero, mas ausência de informação.

Outros exemplos do uso do SQL Query Builder no DB Manager do QGis

(1) quais os distritos de CEMDist_MSP sem hospitais em seus territórios?

```
SELECT "CEMDistMSP"."nomecaps"
    FROM "CEMDistMSP" left join "HOSPITALS" on
    "CEMDistMSP"."nomecaps" = "HOSPITALS"."DISTRITO"
    WHERE "HOSPITALS"."DISTRITO" IS NULL
```

(2) encontrar o retângulo mínimo envolvente de CEMDist_MSP

```
SELECT Min(ST_MinX("CEMDistMSP"."geom")) as Min_X,
    Min(ST_MinY("CEMDistMSP"."geom")) as Min_Y,
    Max(ST_MaxX("CEMDistMSP"."geom")) as Max_X,
    Max(ST_MaxY("CEMDistMSP"."geom")) as Max_Y
    FROM "CEMDistMSP"
```

Obs: o SQL Query Builder lista cerca de 120 funções, inclusive espaciais, mas há questões levantadas em fóruns quanto ao funcionamento de diversas das espaciais, em especial em BDs GeoPackage

Geoprocessamento com GeoPackage

Um BD Geopackage[2] contém no mínimo as tabelas de metadados:

- **gpkg_contents** –lista das demais tabelas do BD, com as colunas:
    - table_name –nome da tabela
    - data_type – tipo de dados: “features”, “tiles”, “attributes” e outros, designados como “extensions”
    - identifier, e description – nomes definidos pelos usuários
    - last_change – última atualização no formato da ISO 8601 (RFC3339[3])
    - min_x, min_y, max_x, e max_y – extensão espacial do conteúdo
    - srs_id – sistema de coordenadas
- **gpkg_spatial_ref_sys** – Sistemas de coordenadas dos dados espaciais do BD, com as seguintes colunas:
    - srs_name, e description – nomes definidos pelos usuários
    - srs_id – identificador unívoco do sistema de coordenadas, e chave primária
    - organization – nome da organização que define o srs
    - organization_coodrsys_id – identificador para o srs dado pela organização que o define
    - definition – definição do srs no formato ‘well-known text’[4]

    Esta tabela deve conter ao menos três tuplas (linhas), com os códigos srs_id = 0 (srs geográfico indefinido), -1 (srs cartesiano indefinido) e 4326 (WGS84 em latitude e longitude).

Além dessas duas (e das tabelas de dados do usuário listadas em gpkg_contents), outras tabelas fazem parte dos metadados de um BD GeoPackage como indicado a seguir.

[2] https://www.geopackage.org/guidance/getting-started.html
[3] https://www.ietf.org/rfc/rfc3339.txt
[4] https://wiki.gis.com/wiki/index.php/Well-known_text

Ilustração do conteúdo do BD criado neste exercício

visualizado com o uso do SGBD SQLITE 3
após a sua criação e inserção do layer (feature) CEMDistMSP e da tabela (attributes) HOSPITALS

```
C:\Users\Marcelo\Documents\AUT5826\teste_TV5_e_QGIS>sqlite3

SQLite version 3.40.0 2022-11-16 12:10:08

sqlite> .open Munic_SaoPaulo.gpkg

sqlite> .tables

CEMDistMSP
HOSPITALS
gpkg_contents
gpkg_extensions
gpkg_geometry_columns
gpkg_ogr_contents
gpkg_spatial_ref_sys
gpkg_tile_matrix
gpkg_tile_matrix_set
rtree_CEMDistMSP_geom
rtree_CEMDistMSP_geom_node
rtree_CEMDistMSP_geom_parent
rtree_CEMDistMSP_geom_rowid
```

Indexação espacial

```
sqlite> .mode column

sqlite> select * from rtree_CEMDistMSP_geom;

id  minx          maxx          miny       maxy
--  ------------  ------------  ---------  ---------
40  313384.03125  320052.65625  7403382.5  7411588.5
7   315349.53125  321061.71875  7387866.5  7391788.5
11  316293.40625  321229.78125  7378191.5  7383690.5
10  316329.96875  323765.8125   7371742.5  7381222.0
4   317137.90625  322359.90625  7382610.5  7387991.0
```

```
sqlite> select * from rtree_CEMDistMSP_geom_node;
```

```
nodeno  data
------  ----
1
2
3
```

```
sqlite> select * from rtree_CEMDistMSP_geom_parent;
```

```
nodeno  parentnode
------  ----------
2       1
3       1
```

```
sqlite> select * from rtree_CEMDistMSP_geom_rowid;
```

```
rowid  nodeno
-----  ------
1      3
2      3
3      3
4      3
5      3
6      3
...
11     3
12     2
13     3
14     2
15     3
16     3
17     3
18     3
19     2
20     2
...
```

Observa-se que a árvore R-Tree possui a raiz (node 1) e dois ramos apenas (nodes 2 e 3) para os retângulos envolventes mínimos dos distritos.

Tabelas dos metadados

```
sqlite> select * from gpkg_contents;
```

table_name	data_type	identifier	description	last_change	min_x	min_y	max_x	max_y	srs_id
CEMDistMSP	features	CEMDistMSP	Distritos do MSP	2022-11-18T19:53:28.736Z	313384.0	7343730.0	360614.0	7416150.0	31983
HOSPITALS	attributes	HOSPITALS		2022-11-18T22:53:16.776Z					0

```
sqlite> select * from gpkg_spatial_ref_sys;
```

srs_name	srs_id	organization	organization_coordsys_id	definition	description
Undefined cartesian SRS	-1	NONE	-1	undefined	undefined cartesian coordinate reference system
Undefined geographic SRS	0	NONE	0	undefined	undefined geographic coordinate reference system
WGS 84 geodetic	4326	EPSG	4326	GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",0.0174532925199433,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4326"]]	longitude/latitude coordinates in decimal degrees on the WGS 84 spheroid
SIRGAS 2000 / UTM zone 23S	31983	EPSG	31983	PROJCS["SIRGAS 2000 / UTM zone 23S",GEOGCS["SIRGAS 2000",DATUM["Sistema_de_Referencia_Geocentrico_para_las_AmericaS_2000",SPHEROID["GRS 1980",6378137,298.257222101,AUTHORITY["EPSG","7019"]],TOWGS84[0,0,0,0,0,0,0],AUTHORITY["EPSG","6674"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",0.0174532925199433,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4674"]],PROJECTION["Transverse_Mercator"],PARAMETER["latitude_of_origin",0],PARAMETER["central_meridian",-45],PARAMETER["scale_factor",0.9996],PARAMETER["false_easting",500000],PARAMETER["false_northing",10000000],UNIT["metre",1,AUTHORITY["EPSG","9001"]],AXIS["Easting",EAST],AXIS["Northing",NORTH],AUTHORITY["EPSG","31983"]]	

```
sqlite> select * from gpkg_extensions;
```

table_name	column_name	extension_name	definition	scope
CEMDistMSP	geom	gpkg_rtree_index	http://www.geopackage.org/spec120/#extension_rtree	write-only

```
sqlite> select * from gpkg_geometry_columns;
```

table_name	column_name	geometry_type_name	srs_id	z	m
CEMDistMSP	geom	POLYGON	31983	0	0

```
sqlite> select * from gpkg_ogr_contents;
```

table_name	feature_count
CEMDistMSP	96
HOSPITALS	371

Esquema do banco de dados com uso de create

```
sqlite> .schema
```

metadados

```
CREATE TABLE gpkg_spatial_ref_sys (srs_name TEXT NOT NULL, srs_id INTEGER NOT NULL PRIMARY KEY, organization TEXT NOT NULL, organization_coordsys_id INTEGER NOT NULL, definition  TEXT NOT NULL, description TEXT);

CREATE TABLE gpkg_contents (table_name TEXT NOT NULL PRIMARY KEY, data_type TEXT NOT NULL, identifier TEXT UNIQUE, description TEXT DEFAULT '',
                            last_change DATETIME NOT NULL DEFAULT (strftime('%Y-%m-%dT%H:%M:%fZ','now')), min_x DOUBLE, min_y DOUBLE, max_x DOUBLE, max_y DOUBLE, srs_id INTEGER,
                            CONSTRAINT fk_gc_r_srs_id FOREIGN KEY (srs_id) REFERENCES gpkg_spatial_ref_sys(srs_id));

CREATE TABLE gpkg_ogr_contents(table_name TEXT NOT NULL PRIMARY KEY, feature_count INTEGER DEFAULT NULL);

CREATE TABLE gpkg_geometry_columns (table_name TEXT NOT NULL, column_name TEXT NOT NULL, geometry_type_name TEXT NOT NULL, srs_id INTEGER NOT NULL, z TINYINT NOT NULL, m TINYINT NOT NULL,
                                    CONSTRAINT pk_geom_cols PRIMARY KEY (table_name, column_name), CONSTRAINT uk_gc_table_name UNIQUE (table_name),
                                    CONSTRAINT fk_gc_tn FOREIGN KEY (table_name) REFERENCES gpkg_contents(table_name), CONSTRAINT fk_gc_srs FOREIGN KEY (srs_id) REFERENCES gpkg_spatial_ref_sys (srs_id));

CREATE TABLE gpkg_tile_matrix_set (table_name TEXT NOT NULL PRIMARY KEY, srs_id INTEGER NOT NULL, min_x DOUBLE NOT NULL, min_y DOUBLE NOT NULL, max_x DOUBLE NOT NULL, max_y DOUBLE NOT NULL,
                                   CONSTRAINT fk_gtms_table_name FOREIGN KEY (table_name) REFERENCES gpkg_contents(table_name), CONSTRAINT fk_gtms_srs FOREIGN KEY (srs_id) REFERENCES gpkg_spatial_ref_sys (srs_id));

CREATE TABLE gpkg_tile_matrix (table_name TEXT NOT NULL, zoom_level INTEGER NOT NULL, matrix_width INTEGER NOT NULL, matrix_height INTEGER NOT NULL, tile_width INTEGER NOT NULL, tile_height INTEGER NOT NULL,
                               pixel_x_size DOUBLE NOT NULL, pixel_y_size DOUBLE NOT NULL, CONSTRAINT pk_ttm PRIMARY KEY (table_name, zoom_level),
                               CONSTRAINT fk_tmm_table_name FOREIGN KEY (table_name) REFERENCES gpkg_contents(table_name));

CREATE TRIGGER 'gpkg_tile_matrix_zoom_level_insert' BEFORE INSERT ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'insert on table ''gpkg_tile_matrix'' violates constraint: zoom_level cannot be less than 0') WHERE (NEW.zoom_level < 0); END;

CREATE TRIGGER 'gpkg_tile_matrix_zoom_level_update' BEFORE UPDATE of zoom_level ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'update on table ''gpkg_tile_matrix'' violates constraint: zoom_level cannot be less than 0') WHERE (NEW.zoom_level < 0); END;

CREATE TRIGGER 'gpkg_tile_matrix_matrix_width_insert' BEFORE INSERT ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'insert on table ''gpkg_tile_matrix'' violates constraint: matrix_width cannot be less than 1') WHERE (NEW.matrix_width < 1); END;

CREATE TRIGGER 'gpkg_tile_matrix_matrix_width_update' BEFORE UPDATE OF matrix_width ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'update on table ''gpkg_tile_matrix'' violates constraint: matrix_width cannot be less than 1') WHERE (NEW.matrix_width < 1); END;

CREATE TRIGGER 'gpkg_tile_matrix_matrix_height_insert' BEFORE INSERT ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'insert on table ''gpkg_tile_matrix'' violates constraint: matrix_height cannot be less than 1') WHERE (NEW.matrix_height < 1); END;
```

```
CREATE TRIGGER 'gpkg_tile_matrix_matrix_height_update' BEFORE UPDATE OF matrix_height ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'update on table ''gpkg_tile_matrix'' violates constraint: matrix_height cannot be less than 1') WHERE (NEW.matrix_height < 1); END;

CREATE TRIGGER 'gpkg_tile_matrix_pixel_x_size_insert' BEFORE INSERT ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'insert on table ''gpkg_tile_matrix'' violates constraint: pixel_x_size must be greater than 0') WHERE NOT (NEW.pixel_x_size > 0); END;

CREATE TRIGGER 'gpkg_tile_matrix_pixel_x_size_update' BEFORE UPDATE OF pixel_x_size ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'update on table ''gpkg_tile_matrix'' violates constraint: pixel_x_size must be greater than 0') WHERE NOT (NEW.pixel_x_size > 0); END;

CREATE TRIGGER 'gpkg_tile_matrix_pixel_y_size_insert' BEFORE INSERT ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'insert on table ''gpkg_tile_matrix'' violates constraint: pixel_y_size must be greater than 0') WHERE NOT (NEW.pixel_y_size > 0); END;

CREATE TRIGGER 'gpkg_tile_matrix_pixel_y_size_update' BEFORE UPDATE OF pixel_y_size ON 'gpkg_tile_matrix' FOR EACH ROW
      BEGIN SELECT RAISE(ABORT, 'update on table ''gpkg_tile_matrix'' violates constraint: pixel_y_size must be greater than 0') WHERE NOT (NEW.pixel_y_size > 0); END;


CREATE TABLE sqlite_sequence(name,seq);

CREATE TABLE gpkg_extensions (table_name TEXT,column_name TEXT,extension_name TEXT NOT NULL,definition TEXT NOT NULL,scope TEXT NOT NULL,CONSTRAINT ge_tce UNIQUE (table_name, column_name, extension_name));
```

Dados do usuário

```
CREATE TABLE IF NOT EXISTS "CEMDistMSP" ( "fid" INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, "geom" MULTIPOLYGON, "id" REAL, "area" REAL, "cemcod" REAL, "ibgecod" REAL,
                                          "sigla" TEXT(5), "nome" TEXT(30), "nomecaps" TEXT(30), "ibgepop00" REAL, "densidade" REAL, "munsigla" TEXT(8), "munnome" TEXT(30));

CREATE TRIGGER "trigger_insert_feature_count_CEMDistMSP" AFTER INSERT ON "CEMDistMSP" BEGIN UPDATE gpkg_ogr_contents SET feature_count = feature_count + 1 WHERE lower(table_name) = lower('CEMDistMSP'); END;

CREATE TRIGGER "trigger_delete_feature_count_CEMDistMSP" AFTER DELETE ON "CEMDistMSP" BEGIN UPDATE gpkg_ogr_contents SET feature_count = feature_count - 1 WHERE lower(table_name) = lower('CEMDistMSP'); END;

CREATE VIRTUAL TABLE "rtree_CEMDistMSP_geom" USING rtree(id, minx, maxx, miny, maxy)

/* rtree_CEMDistMSP_geom(id,minx,maxx,miny,maxy) */;

CREATE TABLE IF NOT EXISTS "rtree_CEMDistMSP_geom_rowid"(rowid INTEGER PRIMARY KEY,nodeno);

CREATE TABLE IF NOT EXISTS "rtree_CEMDistMSP_geom_node"(nodeno INTEGER PRIMARY KEY,data);

CREATE TABLE IF NOT EXISTS "rtree_CEMDistMSP_geom_parent"(nodeno INTEGER PRIMARY KEY,parentnode);

CREATE TRIGGER "rtree_CEMDistMSP_geom_insert" AFTER INSERT ON "CEMDistMSP" WHEN (new."geom" NOT NULL AND NOT ST_IsEmpty(NEW."geom"))
      BEGIN INSERT OR REPLACE INTO "rtree_CEMDistMSP_geom" VALUES (NEW."fid",ST_MinX(NEW."geom"), ST_MaxX(NEW."geom"),ST_MinY(NEW."geom"), ST_MaxY(NEW."geom")); END;

CREATE TRIGGER "rtree_CEMDistMSP_geom_update1" AFTER UPDATE OF "geom" ON "CEMDistMSP" WHEN OLD."fid" = NEW."fid" AND (NEW."geom" NOTNULL AND NOT ST_IsEmpty(NEW."geom"))
      BEGIN INSERT OR REPLACE INTO "rtree_CEMDistMSP_geom" VALUES (NEW."fid",ST_MinX(NEW."geom"), ST_MaxX(NEW."geom"),ST_MinY(NEW."geom"), ST_MaxY(NEW."geom")); END;

CREATE TRIGGER "rtree_CEMDistMSP_geom_update2" AFTER UPDATE OF "geom" ON "CEMDistMSP" WHEN OLD."fid" = NEW."fid" AND (NEW."geom" ISNULL OR ST_IsEmpty(NEW."geom"))
      BEGIN DELETE FROM "rtree_CEMDistMSP_geom" WHERE id = OLD."fid"; END;

CREATE TRIGGER "rtree_CEMDistMSP_geom_update3" AFTER UPDATE ON "CEMDistMSP" WHEN OLD."fid" != NEW."fid" AND (NEW."geom" NOTNULL AND NOT ST_IsEmpty(NEW."geom"))
      BEGIN DELETE FROM "rtree_CEMDistMSP_geom" WHERE id = OLD."fid"; INSERT OR REPLACE INTO "rtree_CEMDistMSP_geom" VALUES (NEW."fid",ST_MinX(NEW."geom"), ST_MaxX(NEW."geom"),ST_MinY(NEW."geom"), ST_MaxY(NEW."geom"));
      END;

CREATE TRIGGER "rtree_CEMDistMSP_geom_update4" AFTER UPDATE ON "CEMDistMSP" WHEN OLD."fid" != NEW."fid" AND (NEW."geom" ISNULL OR ST_IsEmpty(NEW."geom"))
      BEGIN DELETE FROM "rtree_CEMDistMSP_geom" WHERE id IN (OLD."fid", NEW."fid"); END;

CREATE TRIGGER "rtree_CEMDistMSP_geom_delete" AFTER DELETE ON "CEMDistMSP" WHEN old."geom" NOT NULL BEGIN DELETE FROM "rtree_CEMDistMSP_geom" WHERE id = OLD."fid"; END;

CREATE TABLE IF NOT EXISTS "HOSPITALS" ("fid" INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, "HOSPITAL" TEXT, "DISTRITO" TEXT, "NUM_LEITOS" INTEGER, "NUM_ENFERM" INTEGER, "NUM_MEDICO" INTEGER, "NUM_AMBULA" INTEGER,
                                        "PUBLICO" TEXT, "NUM_P_MASC" INTEGER, "NUM_P_FEM" INTEGER, "TOTAL_PAC" INTEGER, "COL_TESTE" INTEGER);

CREATE TRIGGER "trigger_insert_feature_count_HOSPITALS" AFTER INSERT ON "HOSPITALS" BEGIN UPDATE gpkg_ogr_contents SET feature_count = feature_count + 1 WHERE lower(table_name) = lower('HOSPITALS'); END;

CREATE TRIGGER "trigger_delete_feature_count_HOSPITALS" AFTER DELETE ON "HOSPITALS" BEGIN UPDATE gpkg_ogr_contents SET feature_count = feature_count - 1 WHERE lower(table_name) = lower('HOSPITALS'); END;
```