**PRACTICA 1**

**ALMACÉNS E MINARÍA DE DATOS**

**Alex Baquero Domínguez**

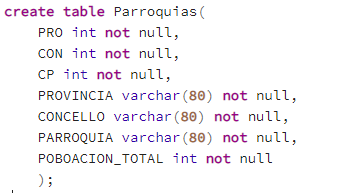
***alex.baquero@rai.usc.es***

**INDICE**

1. **Importación dos datos del Censo de población e vivendas 2001 do IGE…………………………………………………………………….…3**
2. **Cales son as 4 parroquias con máis habitantes da provincia de Pontevedra? ………………………………………………………………..4**
3. **Mostra, para cada provincia, os concellos coa población superior a 15000 habitantes……………………………………………...4**
4. **Instalación do *cstore\_fdw*………………………………………………….6**
5. **Consultas sobre ambas táboas e observación sobre os tempos. …………………………………………………………………………….8**
6. **Instalacion da BBDD Adventure Works………………………………9**
7. **Seccions do fichero install.sql e o seu propósito……………….10**
8. **Instalación da base de datos AdventureWorksDW2012 …,.11**
9. **Importación dos datos do Censo de población e vivendas 2001 do IGE.**

Antes de comenzar a facer nada, debemos instalar o postgresSQL. Para isto é necesario saber que ao longo desta práctica vamos a necesitar instalar o *cstore\_fdw*:

* O *cstore\_fdw* non é compatible con versions anteriores de PostgreSQL ou posteriores (requiere a versión de PostgreSQL de 9.3 a 12). De esta maneira, elexiremos o PostgreSQL12 para facer esta tarefa.
* O *cstore\_fdw* non é compatible con Windows. De esta maneira, elexiremos o sistema operativo Ubuntu para facer esta tarefa.

Creamos a nosa futura base de datos que tendrá una táboa chamada Parroquias da seguinte maneira:

Descargamos os datos do Censo de poboacion e vivendas 2001 do IGE a través de este enlace:

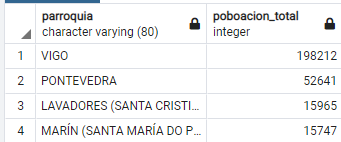
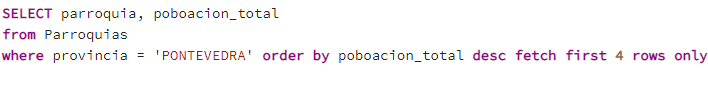
<http://www.ige.eu/igebdt/esqv.jsp?paxina=002001&c=-1&ruta=parroquias/parroquias.jsp>

Estos datos están almacenados nun archivo de Excel, o cal hai que transformar a formato *.csv* para poder copiar todos estos datos na nosa base de datos creada. Este hai que modifcalo para que este codificado a través de UTF-8 e eliminar o separador dos millares (os datos da poboacion serán números enteiros). Importamos os datos do .csv a través:



1. **Cales son as 4 parroquias con máis habitantes da provincia de Pontevedra?**

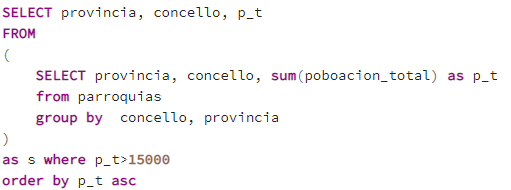
Seleccionamos as parroquias as cales teñan o carácter PONTEVEDRA na columna provincia e seleccionamos as primeiras catro filas, é dicir, as catro parroquias con mais poboacion.

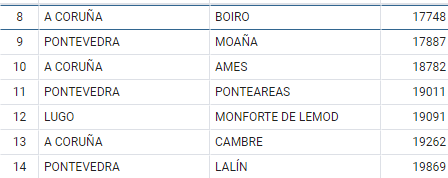
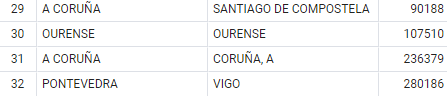


1. **Mostra, para cada provincia, os municipios coa poboación superior a 15000 habitantes.**

Facemos unha subconsulta que sume a poboacion das provincias para cada concello e para obter únicamente unha só fila por concello, agrupamos os datos por provincias e concellos.

Unha vez que temos a poboacion total por concellos a través da subconsulta, escollemos os concellos que teñan unha poboacion total superior a 15000 habitantes.

****

**** ****   

1. **Instalación do *cstore\_fdw*.**

*Cstore\_fdw* é unha extensión de almacén de columnas de código aberto que reduce o espazo de almacenamento de datos e as E/S de disco para bases de datos PostgreSQL.

Antes de comezar coa instalación do *cstore\_fdw*, debemos instalar uns paquetes xa que este depende de protobuf-c para serializar y deserializar os metadatos da táboa:

Sudo apt-get install protobuf-c-compiler

Sudo apt-get install libprotobuf-c-dev

Instalamos o *cstore\_fdw* a través de este comando:

Sudo apt-get update

Sudo apt-get install postgresql-12-cstore-fdw

Antes de usar *cstore\_fdw*, agregamos shared\_preload\_libraries e postgresql.conf reiniciamos o postgres:

shared\_preload\_libraries = 'cstore\_fdw'

Entramos en postgres por liña de comando: sudo –i –u postgres

psql

Creamos una táboa e reiniciamos o servidor:

CREATE FOREING TABLE parroquias\_foreign

Sudo systemctl restart postgresql

Finalmente corremos ALTER EXTENSION cstore\_fdw UPDATE e iniciamos sesión en Postgres de novo. Executamos os seguintes comandos para crear una táboa foránea de almacenamento de columnas:

1. Cargar a extensión por primeira vez despois da instalación:

CREATE EXTENSION cstore\_fdw;

1. Crear un obxeto servidor:

CREATE SERVER cstore\_server FOREIGN DATA WRAPPER cstore\_fdw;

1. Crear táboa foránea

CREATE FOREIGN TABLE parroquias\_foreign

(

PRO INT,

CON INT,

CP INT,

PROVINCIA TEXT,

CONCELLO TEXT,

PARROQUIA TEXT,

POBOACION\_TOTAL INT

)

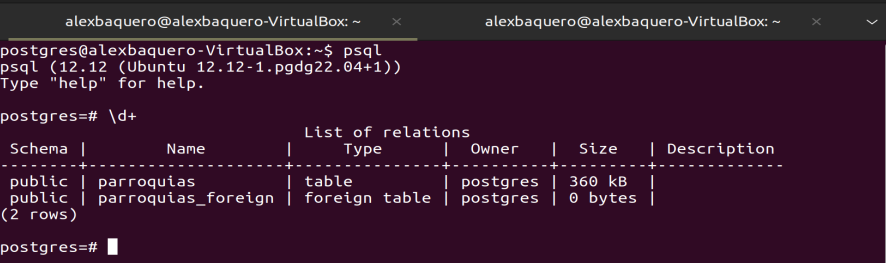
SERVER cstore\_server;

A continuación, cargamos datos na táboa:

\COPY parroquias\_foreign FROM ‘/tmp/PARROQUIAS.CSV ‘WITH CSV;

Mencionar que neste paso deron bastantes erros e tivemos que modifcar algunhas cousas do .csv como suprimir o título de cada columna, sustituir ; por un , e poner os todos textos entre comillas.

Procedemos a comparar o tamaño das táboas parroquias (táboa orixinal) e parroquias\_foreign (táboa foránea). Para iso, executamos \d+ para observar o tamaño das táboas:

****

Podemos ver que a táboa estranxeira ocupa menos que táboa orixinal. Observamos que a FDW pode traer ventaxas e ser moi útil, sobre todo nos casos en que se realice análise de datos:

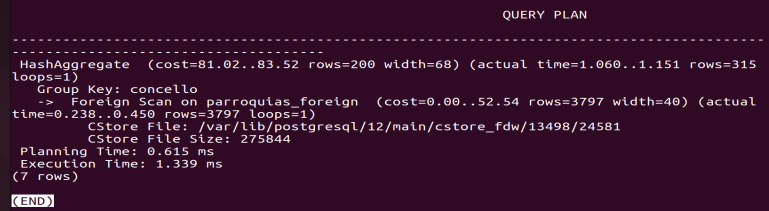
* Compresión: reduce significativamente o espazo en disco e a memoria.
* Proxección de columnas: só le as columnas necesarias para a consulta.

Dende logo, en ocasións, pode evitar moitos contratempos. Xa que implementa un mecanismo de almacenamento en columnas e utiliza o formato Optimized Row Columnar (ORC). Por suposto, estas vantaxes teñen un custo, xa que seguramente a carga é un pouco lenta porque ten que comprimir a información.

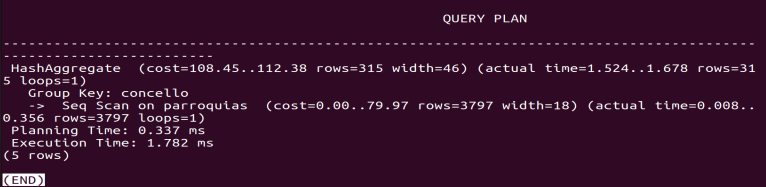
1. **Consultas sobre ambas táboas e observación sobre os tempos.**

Faremos unha consulta complexa onde existan cálculos con funcions de agregados e veremos o plan de execución e o tempo:

explain analyze select concello, min(cp), avg(poboacion\_total) from parroquias\_foreign group by concello;

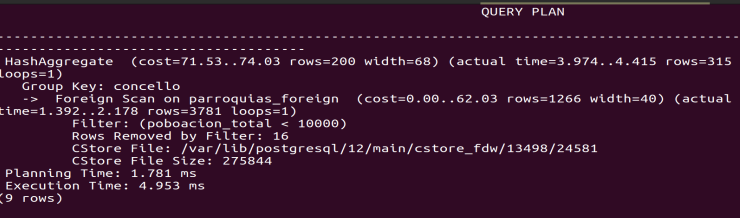


Se executamos a mesma consulta na táboa orixinal:

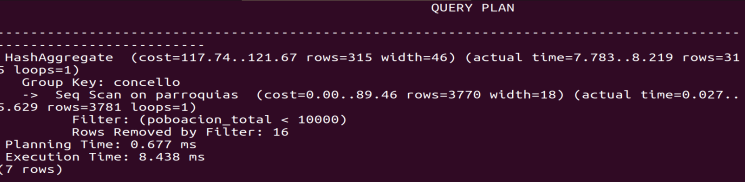


Se lle agregamos algún filtro á consulta:

explain analyze select concello, min(cp), avg(poboacion\_total) from parroquias\_foreign WHERE poboacion\_total<10000 group by concello;



Para a táboa orixinal é:



Incluso sacar o índice da táboa orixinal para a busca pode facer a busca na táboa foránea máis rápida, especialmente engadindo un filtro.

Co cstore\_fdw, a consulta completase máis rápido porque procesa menos datos: cada columna dividise en varios bloques e os índices de salto almacenan valores mínimos e máximos para cada un destes bloques. Se estes valores contradicesen co WHERE ao escanear a táboa, o bloque omitese por completo.

Podemos ver como elimina uns 16 rexistros na busca e isto é porque o cstore\_fdw crea coa opción block\_row\_count (número de filas por bloques de columna) é dicir, fai bloques de columnas onde almacena o mínimo e o máximo valor e así funciona con bloques de exclusión, onde só busca os bloques onde está a condición WHERE, os outros son ignorados.

Para usar os índices de salto de forma máis eficiente, cargamos os datos despois de ordenalos nunha columna que se usa na cláusula WHERE. Isto garante que haxa unha superposición mínima entre os bloques e que a probabilidade de que se salten sexa maior.

1. **Instalacion de la BBDD Adventure Works.**

Ofrécese un ficheiro Ruby para converter os CSV dispoñibles en CodePlex nun formato que Postgres poida usar, así como un script de Postgres para crear as táboas, cargar os datos, converter as columnas de identificadores xerárquicos, engadir claves primarias e externas e crear algunhas das vistas. Todo isto gárdase no script OLTP de Adventure Works 2014, que descargamos e descomprimimos.

Engadimos a este cartafol o update\_csvs.rb e install.sql. Instalamos os ruby a través da terminal, accedemos ao directorio onde se encontra o update\_csvs.rb cd /tmp/AdventureWorks e executamos:

ruby update\_csvs.rb

Con este comando, modificamos os CSV para que funcionen con Postgres.

Accedemos ao postgres na terminal e entramos no mesmo directorio mencionado anteriormente, pois ai tamen se encontra o noso ficheiro install.sql. Creamos a base de datos e as táboas, importamos os datos e configuramos as vistas e as claves:

psql -c "CREATE DATABASE \"Adventureworks\";"

psql -d Adventureworks < install.sql

Finalmente para observar se está todo correcto podemos executar no psql para conectarse a esta base de datos e ver a lista de táboas:

\c "Adventureworks"

\dt (humanresources|person|production|purchasing|sales).\*

1. **Seccions do fichero install.sql e o seu propósito.**

O principal propósito de install.sql é ter un ficheiro que cun simple comando na terminal podas crear as táboas, cargar os datos, converter as columnas de identificadores xerárquicos, engadir claves primarias e externas e crear algunhas das vistas. Este podemolo dividir nas seguintes seccions:

1. **Instalacion do AdventureWorks.** Esta primeira sección consiste nunha breve explicación de cómo podemos usar este script: descargar o zip, comandos para crear as táboas,importar os datos, mostrar as táboas**…** Todo esto está comentado a través do --
2. **Creacion de tipos de datos personalizados.** A través do commando CREATE DOMAIN creanse 6 datos: *OrderNumber, AccountNumber, Flag, NameStyle,Name* e *Phone.*
3. **Creacion dos esquemas cos seus datos e táboas** a través do commandos CREATE SCHEMA, CREATE TABLE
4. **Copiar os datos almacenados nos cvs nas suas correspondientes tablas** a través do commando SELECT … \copy … FROM ….
5. **Convertir de todo o hexágono nun fluxo de bits xerárquicos:** CREATE OR REPLACE FUNCTION
6. **Comentarios nas táboas e columnas** a través dos comandos COMMENT ON TABLE…, COMMENT ON COLUMN…
7. **Establecer as PRIMARY KEYS (chaves primarias) e as FOREIGN KEYS (chaves foráneas)** usando o ALTER TABLE
8. **Vistas:** aparece comentado unha consulta en MSSQL para ver a diferencia nas consultas orientadas a XML entre MSSQLServer e Postgres. Finalmente, executa na versión postgres que é mais recortada a través CREATE VIEW …AS…SELECT e clasifica os produtos e descripcions destes produtos por idioma formando unha vista materializada para que o rendimento poida ser mellor.
9. **Vistas de conveniencia**. Crea un esquema CREATE SCHEMA para varias vistas, agrupándoas.
10. **Instalación da base de datos AdventureWorksDW2012.**

Accedemos ao postgres na terminal e entramos no directorio cd /tmp/AdventureWorksDW,creamos a base de datos e as táboas, importamos os datos e configuamos as vistas e as claves:

psql -c "CREATE DATABASE \"AdventureworksDW\";"

psql -d AdventureworksDW < AdventureWorksDW2012.sql

Non é posible executar o .sql pois a versión que temos de postgres non admite ben o tipo money $ e os números que o teñan a coma dos millares. Para solucionalo:

* Executamos nano AdventureWorksDW2012.sql para editar o arquivo e sustituir o $ por un espacio.
* Executamos o comando sed que permite buscar e remplazar expresions e consiste en:

**sed 's/texto\_a\_buscar/texto\_a\_reemplazar/' <fichero\_a\_reemplazar >fichero\_nuevo**

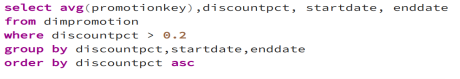
De esta maneira, poñemos na terminal o seguinte:

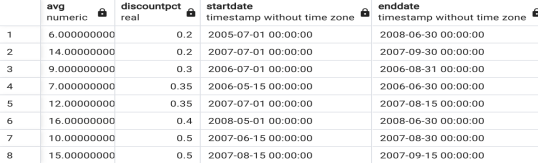
sed –i ‘s/\([0-9]\),\([0-9]\)/\1\2/g’/tmp/AdventureWorksDW/AdventureWorksDW2012.sql

Procedemos a facer consultas na base de datos para comprobar que esta todo correcto:









Finalmente, remataremos comparando a estructura da base de datos de AdventureWorks (apartado anterior) coa AdventureWorksDW (apartado actual):

* AdventureWorksDW ten 68 táboas e AdventureWorks ten 72 táboas. Ten cambios nas táboas debido que a DW ten esquemas para as sales, purchasaing, production, humanResources e person.
* AdventureWorksDW presenta un esquema e a AdventureWorks presenta quinto esquemas diferentes.

A AdventureWorksDW está deseñada para ser máis rápida; concretamente, para proporcionar un acceso rápido á recuperación e análise de datos. Permite un acceso, recompilación e análise de datos máis rápido para que se poida tomar decisións.