17) Módulo 2 - Aula Extra - BLOBs

17.1) BLOBs

Um grande objeto é um método de armazenamento (não um tipo de dados) que habilita o PostgreSQL a armazenar colunas grandes separadamente da tupla destinada a registrar os dados. Um objeto grande aparece como um OID dentro da tupla.
Uma coluna queimada (TOASTed column) é uma coluna onde o servidor automaticamente armazena valores grandes fora da tupla para vários tipos. O fato da coluna ser queimada é invisível à camada do SQL.

Objetos grandes não são necessários tanto quanto foram antes do recurso (TOAST) ser introduzido na versão 7.1. No entanto, objetos grandes podem ser acessados com interfaces cliente e servidor em C através de operações com arquivos do tipo Unix como open, close, lseek, read, e write. Objetos queimados (TOASTED) são sempre tratados como objetos inteiros.

O problema de objetos grandes é que eles não são armazenados separadamente de forma automática e devem ser manipulados separadamente dos outros valores no registro. Isto significa que você deve usar comandos especiais para inserir, atualizar e remover grandes objetos. O pg_dump também possui argumentos especiais para extrair e restaurar objetos grandes.

Caso você esteja armazenando imagens em um banco de dados onde esteja lendo-as e mostrando-as, então, provavelmente, o tipo bytea irá funcionar muito bem. A imagem, caso seja muito grande, deveria ser queimada (TOASTED). Se você possui uma aplicação de gerenciamento de imagens que manipula ou analisa o conteúdo das imagens, então o armazenamento de objetos grandes seria a melhor alternativa. Com o armazenamento de objetos grandes você poderia procurar por um pedaço específico, ler e gravar as alterações usando a interface em C. Se você utiliza uma simples coluna do tipo bytea (queimado), você seria obrigado a ler a imagem inteira e gravá-la também inteira, mesmo qua a alteração fosse em um só bit.

As interfaces em C também são úteis como funções do lado do servidor para análise de dados. Sua função em C do lado do servidor que reconhece uma imagem pode varrer e procurar por padrões ou aplicar transformações na imagem (morphing) sem necesitar transferir qualquer parte da imagem para o cliente. A interface cliente pgtcl, também suporta a funcionalidade de objetos grandes.

Para criar uma tabela que contenha um objeto grande, defina a coluna com o tipo OID. O OID será o identificador do objeto grande neste caso.

O próximo problema é como pegar um objeto grande do seu sistema de arquivo para ser armazenado dentro da tabela. O seu objeto grande se encontra do lado cliente ou servidor? Um arquivo do lado cliente é um arquivo que vive na máquina onde a parte cliente está rodando. Esta distinção é válida somente se o cliente estiver numa máquina diferente de onde estiver instalado o banco de dados. Um arquivo do lado servidor é um arquivo que reside na mesma máquina que o servidor de banco de dados.

Há uma diferença entre carregar arquivos do lado servidor e carregar arquivos do lado cliente. Para arquivos do lado servidor, você pode invocar a função lo_import() na sua instrução INSERT. Todo o trabalho é executado pelo servidor de banco de dados.

Para arquivos do lado cliente, o cliente tem que ter o trabalho de abrir o arquivo e enviá-lo para o servidor. Isto é feito no programa cliente. Isto também pode ser feito usando psql (que é um programa cliente especial) utilizando dois passos. Primeiro, importe o objeto grande e então insira

seu OID no devido registro da tabela utilizando o valor da variável :LASTOID do psql.

Quando você selecionar uma linha que contenha um objeto grande, o que você verá na coluna do objeto grande será um número OID. Para que acesse o conteúdo do objeto grande, você deve também extraí-lo para um arquivo ou abrí-lo e manipulá-lo através da sua interface cliente ou de uma função do servidor.

Para extrair a imagem da tabela mypictures para a máquina servidora, use lo_export(). Especifique a coluna do objeto grande e o arquivo destino de saída. O arquivo de destino de saída deve estar liberado para ser gravado pelo banco de dados do servidor e pertencerá ao dono do processo do banco de dados.

Para extrair a imagem de dentro de um arquivo na máquina cliente usando o psql, é preciso saber alguns truques. Sem o recurso de um shell script, você deve prestar atenção com um olho mágico no valor da coluna da figura e usá-lo na declaração \lo_export . (Se alguém souber como fazer isso somente com SQL e psql por favor, me avise!)

O código a seguir mostra a criação de uma tabela que contém um objeto grande. Então, carrega uma imagem a partir da máquina servidora e exporta uma cópia dela na máquina servidora. Então, carrega uma imagem a partir da máquina cliente e exporta uma cópia dela na máquina cliente.

```
--
-- Minha tabela de figuras.
--
CREATE TABLE mypictures (
   title TEXT NOT NULL primary key,
   picture OID);

-- A figura de Rosas Vermelhas está na Máquina Servidora
-- Carrega e exporta uma cópia na Máquina Servidora
--
INSERT INTO mypictures (title, picture)
    VALUES ('Red Roses', lo_import('/tmp/redroses.jpg'));
SELECT lo_export(picture, '/tmp/redroses_copy.jpg') FROM mypictures
    WHERE title = 'Red Roses';

--
-- A figura de Rosas Brancas está na Máquina Cliente
-- Carrega e exporta uma cópia na Máquina Cliente
--
\lo_import '/tmp/whiteroses.jpg'
INSERT INTO mypictures (title, picture) VALUES ('White Roses', :LASTOID);

SELECT * from mypictures;

--     title    | picture
-- -------------+---------
--  Red Roses   | 3715516
--  White Roses | 3715518
-- (2 rows)

\lo_export 3715518 '/tmp/whiteroses_copy.jpg'
```

Tudo isso funciona, no entanto, o que acontece quando você apaga uma linha ou atualiza um objeto grande? Apagando uma linha (registro) de uma tabela do modo usual apaga a linha, no entanto, o objeto grande é deixado pendente. Você pode dizer que o objeto largo ainda existe usando

\lo_list

no psql. Este comando lista todos os OIDs de objetos grandes conhecidos no sistema.

Por outro lado, às vezes o objeto grande não é deixado pendente porque outros registros referenciam a mesma imagem. Como os objetos grandes são, bem, grandes, pensou-se que seria melhor somente tratar os identificadores dos objetos, o OID. Aquele OID pode ser referenciado por outros registros se o desenvolvedor não quiser múltiplas cópias do mesmo objeto grande e aquele OID também pode estar envolvido em uma transação no mesmo ou em diferentes registros. Isto é um problema que o desenvolvedor deve resolver.

Neste exemplo, nós assumimos que quando um registro é removido, então, queremos também remover o objeto grande. Isto implica que nunca haverá outro registro se referenciando ao mesmo objeto grande.

Para fazer isso use uma regra (rule) para chamar a função lo_unlink() do servidor. Usando as interfaces de objetos grandes é possível gravar em objetos grandes e modificá-los. Quando você estiver usando somente SQL, então você precisa descartar (drop) e substituir o objeto grande. Isto também poderia ser feito por uma regra. Assim que você apontar um novo valor para o objeto grande, ele somente não aponta mais para o antigo. Mas faz isso somente se receber um novo valor.

```
CREATE RULE droppicture AS ON DELETE TO mypictures
DO SELECT lo_unlink( OLD.picture );

CREATE RULE reppicture AS ON UPDATE TO mypictures
DO SELECT lo_unlink( OLD.picture ) where OLD.picture <> NEW.picture;
```

Para verificar se aquela regra está funcionando, no psql selecione um registro da tabela e observe o registro o qual você quer remover. Use \lo_list para mostrar os objetos grandes. Descarte (remova) o registro de maneira usual. Selecionando da tabela, voce sabe que o registro da tabela foi removido e, usando \lo_list você pode ver que o objeto grande correspondente também foi removido.

```
=# select * from mypictures;
    title    | picture
-------------+---------
 White Roses | 3715592
 Red Roses   | 3715593
(2 rows)

=# update mypictures set picture=lo_import('/tmp/redroses_copy.jpg')
   where title='Red Roses';
 lo_unlink
-----------
         1
(1 row)

=# \lo_list
     Large objects
   ID    | Description
---------+-------------
 3715592 |
 3715598 |
(2 rows)
```

```
=# select * from mypictures;
    title    | picture
-------------+---------
 White Roses | 3715592
 Red Roses   | 3715598
(2 rows)
```

ATENÇÃO: Note que se você não apontar mais um objeto grande que está sendo referenciado por um registro existente E você possui estas regras no lugar, você não pode descartar (remover) o registro. Você deve descartar a regra, remover o registro e recriar a REGRA. A variável LO_TRANSACTION que determina a maneira pela qual se trata operações com objetos grandes não tem mais efeito para esta finalidade e **não existe mais na versão 7.4**.

Colaboradores: elein em varlena.com

Fonte: http://www.postgresql.org.br/Blobs_%28ou_como_armazenar_arquivos_dentro_do_banco%29