4) Administração de Grupos, Usuários e Privilégios

4.1) Gerenciando Grupos
4.2) Gerenciando Usuários
4.3) Gerenciando Permissões
4.4) Controle de acesso a Objetos
4.5) GRANT e REVOKE para Objetos
4.6) Acesso Remoto

DCL (Data Control Language) é formado por um grupo de comandos SQL, responsáveis pela administração dos usuários, dos grupos e dos privilégios.

Dois usos importantes deste assunto: organização (dividir os bancos e esquemas pelos setores de empresas e instituições) e segurança (cada usuário somente com suas permissões restritivas).

Garantindo (GRANT) Privilégios

http://pgdocptbr.sourceforge.net/pg82/sql-grant.html

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | REFERENCES | TRIGGER }
[,...] | ALL [ PRIVILEGES ] }
ON [ TABLE ] nome_da_tabela [, ...]
TO { nome_do_usuário | GROUP nome_do_grupo | PUBLIC } [, ...] [ WITH GRANT OPTION ]
```

```
GRANT { { USAGE | SELECT | UPDATE }
[,...] | ALL [ PRIVILEGES ] }
ON SEQUENCE nome_da_seqüência [, ...]
TO { nome_do_usuário | GROUP nome_do_grupo | PUBLIC } [, ...] [ WITH GRANT OPTION ]
```

```
GRANT { { CREATE | CONNECT | TEMPORARY | TEMP } [,...] | ALL [ PRIVILEGES ] }
ON DATABASE nome_do_banco_de_dados [, ...]
TO { nome_do_usuário | GROUP nome_do_grupo | PUBLIC } [, ...] [ WITH GRANT OPTION ]
```

```
...
GRANT { CREATE | ALL [ PRIVILEGES ] }
ON TABLESPACE nome_do_espaço_de_tabelas [, ...]
TO { nome_do_usuário | GROUP nome_do_grupo | PUBLIC } [, ...] [ WITH GRANT OPTION ]
```

GRANT role [, ...] TO nome_do_usuário [, ...] [WITH ADMIN OPTION]

GRANT pode ser utilizado para conceder privilégios a um usuário sobre um objeto de banco de dados e para tornar um usuário membro de um papel (antigo grupo). A palavra chave PUBLIC indica que os privilégios devem ser concedido para todos os papéis, inclusive aos que vierem a ser criados posteriormente. PUBLIC pode ser considerado como um grupo definido implicitamente que sempre inclui todos os papéis.

Se for especificado WITH GRANT OPTION quem receber o privilégio poderá, por sua vez, conceder o privilégio a terceiros. Sem a opção de concessão, quem recebe não pode conceder o privilégio. A opção de concessão não pode ser concedida para PUBLIC.

Não é necessário conceder privilégios para o dono do objeto (geralmente o usuário que o criou), porque o dono possui todos os privilégios por padrão (Entretanto, o dono pode decidir revogar alguns de seus próprios privilégios por motivo de segurança). O direito de remover um objeto, ou de alterar a sua definição de alguma forma, não é descrito por um privilégio que possa ser concedido; é inerente ao dono e não pode ser concedido ou revogado.

Os privilégios possíveis são:

SELECT
Permite consultar (SELECT) qualquer coluna da tabela, visão ou seqüência especificada. Também permite utilizar o comando COPY TO (exportar). Para as seqüências, este privilégio também permite o uso da função currval.

INSERT
Permite inserir (INSERT) novas linhas na tabela especificada. Também permite utilizar o comando COPY FROM (importar).

UPDATE
Permite modificar (UPDATE) os dados de qualquer coluna da tabela especificada. Os comandos SELECT ... FOR UPDATE e SELECT ... FOR SHARE também requerem este privilégio (além do privilégio SELECT). Para as seqüências, este privilégio permite o uso das funções nextval e setval.

DELETE
Permite excluir (DELETE) linhas da tabela especificada.

REFERENCES
Para criar uma restrição de chave estrangeira é necessário possuir este privilégio, tanto na tabela que faz referência quanto na tabela que é referenciada.

TRIGGER
Permite criar gatilhos na tabela especificada (Consulte o comando CREATE TRIGGER).

CREATE
Para bancos de dados, permite a criação de novos esquemas no banco de dados.
Para esquemas, permite a criação de novos objetos no esquema. Para mudar o nome de um objeto existente é necessário ser o dono do objeto e possuir este privilégio no esquema que o contém.

Para espaços de tabelas (tablespaces), permite a criação de tabelas e índices no espaço de tabelas, e permite a criação de bancos de dados possuindo este espaço de tabelas como seu espaço de tabelas padrão (Deve ser observado que revogar este privilégio não altera a colocação dos objetos existentes).

CONNECT
Permite ao usuário se conectar ao banco de dados especificado. Este privilégio é verificado no estabelecimento da conexão (além de serem verificadas as restrições impostas por pg_hba.conf).

TEMPORARY
TEMP
Permite a criação de tabelas temporárias ao usar o banco de dados.

EXECUTE
Permite utilizar a função especificada (internas) e qualquer operador implementado utilizando a função. Este é o único tipo de privilégio aplicável às funções (Esta sintaxe funciona para as funções de agregação também).

USAGE
Para as linguagens procedurais, permite o uso da linguagem especificada para criar funções nesta linguagem. Este é o único tipo de privilégio aplicável às linguagens procedurais.

Para os esquemas, permite acessar os objetos contidos no esquema especificado (assumindo que os privilégios requeridos para os próprios objetos estejam atendidos). Essencialmente, concede a quem recebe o direito de "procurar" por objetos dentro do esquema. Sem esta permissão ainda é possível ver os nomes dos objetos, por exemplo consultando as tabelas do sistema. Além disso, após esta permissão ter sido revogada os servidores existentes poderão conter comandos que realizaram anteriormente esta procura, portanto esta não é uma forma inteiramente segura de impedir o acesso aos objetos.

Para as seqüências este privilégio permite a utilização das funções currval e nextval.

ALL PRIVILEGES
Concede todos os privilégios disponíveis de uma só vez. A palavra chave PRIVILEGES é opcional no PostgreSQL, embora seja requerida pelo SQL estrito.

Revogando (REVOKE) Privilégios

http://pgdocptbr.sourceforge.net/pg82/sql-revoke.html

```
REVOKE [ GRANT OPTION FOR ]
{ { SELECT | INSERT | UPDATE | DELETE | REFERENCES | TRIGGER }
[,...] | ALL [ PRIVILEGES ] }
ON [ TABLE ] nome_da_tabela [, ...]
FROM { nome_do_usuário | GROUP nome_do_grupo | PUBLIC } [, ...]
[ CASCADE | RESTRICT ]
```

```
REVOKE [ GRANT OPTION FOR ]
{ { USAGE | SELECT | UPDATE }
[,...] | ALL [ PRIVILEGES ] }
ON SEQUENCE nome_da_seqüência [, ...]
FROM { nome_do_usuário | GROUP nome_do_grupo | PUBLIC } [, ...]
[ CASCADE | RESTRICT ]
```

```
REVOKE [ GRANT OPTION FOR ]
{ { CREATE | CONNECT | TEMPORARY | TEMP } [,...] | ALL [ PRIVILEGES ] }
ON DATABASE nome_do_banco_de_dados [, ...]
FROM { nome_do_usuário | GROUP nome_do_grupo | PUBLIC } [, ...]
[ CASCADE | RESTRICT ]
```

```
...
REVOKE [ GRANT OPTION FOR ]
{ { CREATE | USAGE } [,...] | ALL [ PRIVILEGES ] }
ON SCHEMA nome_do_esquema [, ...]
FROM { nome_do_usuário | GROUP nome_do_grupo | PUBLIC } [, ...]
[ CASCADE | RESTRICT ]
```

```
REVOKE [ GRANT OPTION FOR ]
{ CREATE | ALL [ PRIVILEGES ] }
ON TABLESPACE nome_do_espaço_de_tabelas [, ...]
FROM { nome_do_usuário | GROUP nome_do_grupo | PUBLIC } [, ...]
[ CASCADE | RESTRICT ]
```

```
REVOKE [ ADMIN OPTION FOR ]
role [, ...] FROM nome_do_usuário [, ...]
[ CASCADE | RESTRICT ]
```

O comando REVOKE revoga, de um ou mais papeis, privilégios concedidos anteriormente. A palavra chave PUBLIC se refere ao grupo contendo todos os usuários, definido implicitamente.

Se for especificado GRANT OPTION FOR somente a opção de concessão do privilégio é revogada, e não o próprio privilégio. Caso contrário, tanto o privilégio quanto a opção de concessão serão revogados.
Se, por exemplo, o usuário A concedeu um privilégio com opção de concessão para o usuário B, e o usuário B por sua vez concedeu o privilégio para o usuário C, então o usuário A não poderá revogar diretamente o privilégio de C. Em vez disso, o usuário A poderá revogar a opção de concessão do usuário B usando a opção CASCADE, para que o privilégio seja, por sua vez, revogado do usuário C.

Usuários, grupos e privilégios

De fora do psql (no prompt do SO) utiliza-se comandos sem espaços: createdb, dropdb, etc.
De dentro do psql os comandos são formados por duas palavras separadas por espaço:
CREATE DATABASE, DROP DATABASE, etc (sintaxe SQL).

Em SQL:
CREATE USER (é agora um alias para CREATE ROLE, que tem mais recursos)

banco=# \h create role
Comando: CREATE ROLE
Descrição: define um novo papel (role) do banco de dados

Sintaxe:
CREATE ROLE nome [[WITH] opção [...]]

onde opção pode ser:

SUPERUSER | NOSUPERUSER
| CREATEDB | NOCREATEDB
| CREATEROLE | NOCREATEROLE
| CREATEUSER | NOCREATEUSER
| INHERIT | NOINHERIT
| LOGIN | NOLOGIN
| CONNECTION LIMIT limite_con
| [ENCRYPTED | UNENCRYPTED] PASSWORD 'senha'
| VALID UNTIL 'tempo_absoluto'
| IN ROLE nome_role [, ...]
| IN GROUP nome_role [, ...]
| ROLE nome_role [, ...]
| ADMIN nome_role [, ...]

| USER nome_role [, ...]
| SYSID uid

Caso não seja fornecido ENCRYPTED ou UNENCRYPTED então será usado o valor do parâmetro password_encryption do script postgresql.conf.

Criar Usuário
CREATE ROLE nomeusuario;

Criar um Usuário local com Senha:
Entrar no pg_hba.conf:
local all all 127.0.0.1/32 md5

Comentar as outras entradas para conexão local (caso existam).
Isso para usuário local (conexão via socket UNIX).

Criamos assim:
CREATE ROLE nomeuser WITH LOGIN ENCRYPTED PASSWORD '********';

Ao se logar:
psql -U nomeuser nomebanco.
CREATE ROLE nomeusuario VALID UNTIL 'data'

Excluindo Usuário
DROP ROLE nomeusuario;

Como usuário, na linha de comando:
Criar Usuário:
createuser -U postgres nomeusuario

Excluindo Usuário
dropuser -U postgres nomeusuario;

Criando Superusuário
CREATE ROLE nomeuser WITH LOGIN SUPERUSER ENCRYPTED PASSWORD '******';
-- usuário com poderes de super usuário

Alterar Conta de Usuário
ALTER ROLE nomeuser ENCRYPTED PASSWORD '******' CREATEUSER
-- permissão de criar usuários
ALTER ROLE nomeuser VALID UNTIL '12/05/2006';
ALTER ROLE fred VALID UNTIL 'infinity';
ALTER ROLE miriam CREATEROLE CREATEDB; -- poderes para criar bancos
Obs.: Lembrando que ALTER ROLE é uma extensão do PostgreSQL.

Listando todos os usuários do SGBD:
SELECT usename FROM pg_user;

A tabela pg_user é uma tabela de sistema (_pg) que guarda todos os usuários do PostgreSQL.

Também podemos utilizar:
\du ou \dg

Listando todos os grupos:
SELECT groname FROM pg_group;

Privilégios

Dando Privilégios a um Usuário
GRANT UPDATE ON nometabela TO nomeusuario;

Dando Privilégios a um Grupo Inteiro
GRANT SELECT ON nometabela TO nomegrupo;

Removendo Todos os Privilégios de Todos os Users de uma Tabela
REVOKE ALL ON nometabela FROM PUBLIC

O superusuário tem direito a fazer o que bem entender em qualquer banco de dados do SGBD.

O usuário que cria um objeto (banco, tabela, view, etc) é o dono do objeto.
Para que outro usuário tenha acesso ao mesmo deve receber privilégios.
Existem vários privilégios diferentes: SELECT, INSERT, UPDATE, DELETE, RULE, REFERENCES, TRIGGER, CREATE, TEMPORARY, EXECUTE e USAGE.
Os privilégios aplicáveis a um determinado tipo de objeto variam de acordo com o tipo do objeto (tabela, função, etc.).

O comando para conceder privilégios é o GRANT. O de remover é o REVOKE.

GRANT UPDATE ON contas TO joel;
Dá a joel o privilégio de executar consultas update no objeto contas.

GRANT SELECT ON contas TO GROUP contabilidade;

REVOKE ALL ON contas FROM PUBLIC;

Os privilégios especiais do dono da tabela (ou seja, os direitos de DROP, GRANT, REVOKE, etc.) são sempre inerentes à condição de ser o dono, não podendo ser concedidos ou revogados. Porém, o dono do objeto pode decidir revogar seus próprios privilégios comuns como, por exemplo, tornar a tabela somente para leitura para o próprio, assim como para os outros.

Normalmente, só o dono do objeto (ou um superusuário) pode conceder ou revogar privilégios para

um objeto.

Criação dos grupos
CREATE ROLE dbas;

CREATE ROLE dba1 ENCRYPTED PASSWORD 'dba1' CREATEDB CREATEROLE;

-- Criação dos Usuários do Grupo adm
CREATE ROLE andre ENCRYPTED PASSWORD 'andre' CREATEDB IN ROLE adm;
CREATE ROLE michela ENCRYPTED PASSWORD 'michela' CREATEDB IN ROLE adm;

O usuário de sistema (super usuário) deve ser um usuário criado exclusivamente para o PostgreSQL. Nunca devemos torná-lo dono de nenhum executável, como também devemos evitar o uso do super-usuário para administrar o SGBD. Para isso é recomendado criar um usuário com algumas restrições e utilizar o super-usuário somente se estritamente necessário.

Os nomes de usuários são globais para todo o agrupamento de bancos de dados, ou seja, podemos utilizar um usuário com qualquer dos bancos.

Os privilégios DROP, GRANT, REVOKE, etc pertencem ao dono do objeto não podendo ser concedidos ou revogados. O máximo que um dono pode fazer é abdicar de seus privilégios e com isso ninguém mais teria os mesmos e o objeto seria somente leitura para todos.

DICA
Exemplo: para permitir a um usuário apenas os privilégios de INSERT, UPDATE e SELECT e não permitir o de DELETE em uma tabela, use:

1) REVOKE ALL ON tabela FROM usuario;
2) GRANT SELECT,UPDATE,INSERT ON tabela TO usuario;

Mais detalhes:
http://pgdocptbr.sourceforge.net/pg80/user-manag.html
http://pgdocptbr.sourceforge.net/pg80/sql-revoke.html
http://pgdocptbr.sourceforge.net/pg80/sql-grant.html

Privilégios com o PGAdmin

ALL
Concede todos os privilégios disponíveis de uma só vez. A palavra chave PRIVILEGES é opcional

no PostgreSQL, embora seja requerida pelo SQL estrito.

TEMPORARY
TEMP

Permite a criação de tabelas temporárias ao usar o banco de dados.
CREATE
Para bancos de dados, permite a criação de novos esquemas no banco de dados.
Para esquemas, permite a criação de novos objetos no esquema. Para mudar o nome de um objeto existente é necessário ser o dono do objeto e possuir este privilégio no esquema que o contém.
Para espaços de tabelas, permite a criação de tabelas e índices no espaço de tabelas, e permite a criação de bancos de dados possuindo este espaço de tabelas como seu espaço de tabelas padrão (Deve ser observado que revogar este privilégio não altera a colocação dos objetos existentes).

CONNECT
Permite ao usuário se conectar ao banco de dados especificado. Este privilégio é verificado no estabelecimento da conexão (além de serem verificadas as restrições impostas por pg_hba.conf).

WITH GRANT OPTION
As concessões e revogações também podem ser realizadas por um papel que não é o dono do objeto afetado, mas é membro do papel que é dono do objeto, ou é um membro de um papel que possui privilégios com WITH GRANT OPTION no objeto. Nestes casos, os privilégios serão registrados como tendo sido concedidos pelo papel que realmente possui o objeto, ou possui os privilégios com WITH GRANT OPTION.

Exercícios – Gerenciando grupos e usuários (roles)

create group e create user ainda podem ser utilizados mas recomenda-se create role ao invés.

create role dbas;

Criar Superusuário:
create role super2 superuser login connection limit 40 password 'super'
-- default é -1 (sem limites)

Estando usando o método ident e opção sameuser no pg_hba.conf, um usuário criado com login no psql, somente poderá acessar o postgresql se também for criado no sistema operacional.

sudo adduser super2
su - super2
psql -U super2 postgres

Criar usuário:
create role usuario2;

Criar Usuário
Na linha de comando do SO: createuser -U dba usuario

Comandos SQL:
create role usuario;
create role with login;
create role with login password 'usuario' valid until '2007-12-31';

Adicionar usuário ao grupo:

alter role adm role usuario2; // Adiciona o usuário2 no grupo adm
alter role adm role usuario3;
alter role adm role usuario4;

Excluir usuário:
drop role usuario2;

Remover apenas do Grupo, sem remover o usuário
revoke adm from usuario2;

Observação Importante
O comando "alter role ..." também pode ser utilizado para alterar praticamente todas as propriedades de usuários ou grupos, exceto para adicionar ou remover usuário de grupos, que agora ficam a cargo dos comandos GRANT e REVOKE.

Os usuários usuario2, usuario3 e usuario4 farão parte do grupo adm.

Listando todos os usuários do SGBD
SELECT usename FROM pg_user;

\du - listar usuários
ou
\dg - lista roles e respectivos grupos (Membro de ...)

Listar os grupos
SELECT groname FROM pg_group;

Excluir Usuário ou Grupo:
drop role nomerole;

Concedendo (GRANT) e Revogando (REVOKE) Privilégios

Após a criação de uma tabela execute:
\z clientes

Verá:
Privilégios de acesso ao banco dados "teste2"
Esquema | Nome | Tipo | Privilégios de acesso
---------+--------------+--------+---
public | clientes | tabela |
(1 linha)

Podemos conceder/revogar Privilégiosd de
- esquemas
    - funções
    - procedures
    - sequências
    - tabelas (todas ou individualmente)
    - views

Agora conceda alguns privilégios:

GRANT SELECT ON clientes TO PUBLIC; -- Todas as permissões
GRANT SELECT, UPDATE, INSERT ON clientes TO GROUP progs;

Execute novamente:
\z clientes

Veja agora o resultado:

Privilégios de acesso ao banco dados "teste2"
Esquema | Nome | Tipo | Privilégios de acesso

```
---------+--------------+--------+-------------------------------------------------------------
public | clientes | tabela | {usuario1=arwdRxt/miriam,=r/usuario1,"group todos=arw/usuario1"}
(1 linha)
```

As entradas mostradas pelo comando \z são interpretadas da seguinte forma:

=xxxx -- privilégios concedidos para PUBLIC
uname=xxxx -- privilégios concedidos para o usuário
group gname=xxxx -- privilégios concedidos para o grupo

r -- SELECT ("read")
w -- UPDATE ("write")
a -- INSERT ("append")
d -- DELETE
x -- REFERENCES
t -- TRIGGER
X -- EXECUTE
U -- USAGE
C -- CREATE
c -- CONNECT
T -- TEMPORARY
arwdxt -- ALL PRIVILEGES (para tabelas)

* -- opção de concessão para o privilégio precedente

/yyyy -- usuário que concedeu o privilégio

Se a coluna "Access privileges/Privilégios de acesso" estiver vazia para um determinado objeto, isto significa que o objeto possui os privilégios padrão. Os privilégios padrão sempre incluem todos os privilégios para o dono

Deve ser observado que as opções de concessão implícitas do dono não são marcadas na visualização dos privilégios de acesso. O * aparece somente quando as opções de concessão foram concedidas explicitamente para alguém.

Criar Usuário para Testes:

psql -U postgres

create role uteste;

alter role uteste with password 'uteste' login createdb createrole;

\c – uteste

create database bteste;
\c bteste

create table tabelat (codigo int primary key);

Quando um usuário cria um objeto ele é o dono deste objeto.

Se queremos ter segurança ao conceder privilégios a um usuário sobre um objeto, devemos antes remover todos os privilégios do mesmo sobre o objeto. Depois então conceder somente os privilégios desejados. Como no exemplo abaixo:

\c – postgres
revoke all privileges on tabelat from uteste;

\c bteste uteste
select * from tabelat; // receberemos um erro de permissão negada, já que rmovemos todos os privilégios do uteste sobre a tabela tabelat.

\c – portgres
grant select on tabelat to uteste;
\c – uteste
select * from tabelat; // Agora a consulta é permitida

Se quizermos conceder todos os privilégios existentes ao usuário usamos:
grant all privileges on tabelat to uteste;

Observar que quando existir um campo do tipo serial, temos que conceder também os privilégios para a sequência gerada pelo serial. Para saber o nome exato da sequência execute o comando \d. Geralmente o PG usa a regra ***serial_nomecampo_seq***.

Conceder os privilégios de SELECT, UPDATE e INSERT na tabela tabelat ao usuário uteste:
\c - postgres

grant select, update, insert on tabelat to uteste;
\z
Agora teste os privilégios concedidos e também o delete, não concedido:

\c – uteste
select * from tabelat; // Ok
insert into tabelat values (1); // Ok
update tabelat set codigo = 5 where codigo =1; // Ok
delete tabelat where codigo = 5; // Permissão negada

Revogando (REVOKE) Privilégios

Revogar o privilégio de inserção na tabela filmes concedido para todos os usuários:
REVOKE INSERT ON filmes FROM PUBLIC;

Revogar todos os privilégios concedidos ao usuário manuel sobre a visão vis_tipos:
REVOKE ALL PRIVILEGES ON vis_tipos FROM manuel;

Removendo privilégios de acesso a usuário em esquema
REVOKE CREATE ON SCHEMA public FROM PUBLIC
Com isso estamos tirando o privilégio de todos os usuários acessarem o esquema public.

-- Para que nenhum usuário tenha acesso à tabela use:
-- REVOKE ALL ON nometabela FROM PUBLIC;

Acesso Remoto

- Visualizar os scripts originais: postgresql.conf, pg_hba.conf e o pg_ident.conf do micro servidor (do professor) do PostgreSQL
- Tentar conectar ao PostgreSQL numa máquina remota (outro micro da sala), usando o PGAdmin e o psql
- Observar as mensagens de erro
- Criar um novo usuário no micro cliente
- Liberar no postgresql.conf, no pg_hba.conf apenas para o usuário 'postgres', para o banco 'dbapg', para o IP 192.168.x.y
- Tentar novamente
- Liberar para toda a rede interna

Aproveitar para fazer um tour pelo PGAdmin.
Observe que as tablespaces e as roles ficam fora dos bancos.