

PostgreSQL



www.postgresql.org

André Luiz Fortunato da Silva
Analista de Sistemas
CIRP / USP
alf@cirp.usp.br

Características

- Licença BSD (aberto, permite uso comercial)
- Multi-plataforma (Unix, GNU/Linux, Windows...)
- Transações e “savepoints”
- Herança de tabelas
- Índices
- Tipo Serial (SEQUENCE simula auto-incremento)
- “Backup online” e “recovery”
- Tabelas temporárias
- Controle de acesso (usuário, banco e host)

Características

- Suporte a idiomas (no initdb)
- Objetos grandes (imagens, vídeos, etc.)
- Sub-consultas
- Integridade referencial
- Funções armazenadas
- Gatilhos
- Novos tipos de dados
- Esquemas (encapsula objetos em namespaces)
- Regras (rules, ações alternativas)

Características

- “Tablespaces” (outro local no sistema de arq.)
- Limpeza e otimização de consultas (VACUUM)
- Visões (é possível simular visões atualizáveis)
- Recuperação a partir de logs de transação (“Point in Time Recovery”)
- DB Link (contrib, comunicação entre servidores)
- Replicação (projetos Slony-I, pgPool e pgCluster)

Limites ???

- Tamanhos máximos:
 - banco de dados: ilimitado
 - tabela: 32 TB
 - Uma linha: 1,6 TB
 - Um campo: 1 GB
- Quantidades máximas:
 - linhas por tabela: ilimitado
 - colunas por tabela: de 250 a 1600 (depende do tipo)
 - índices por tabela: ilimitado

Aprofundando

Apoio:

- www.postgresql.org/docs
- www.postgresql.org.br
- pgfoundry.org
- gborg.postgresql.org

Treinamento:

- www.centercursos.com.br (Ribeirão Preto)
- www.dbexperts.com.br (São Paulo)

Vamos praticar um
pouco ??? ...

Interface modo texto psql:

psql -l (mostra os bancos)

psql template1 (acessa o banco)

Dentro do psql:

\? (help dos comandos internos do psql)

\h (comandos SQL disponíveis)

\h create database (sintaxe de um comando SQL)

Criando um banco de dados:

```
create database empresa  
  with encoding='LATIN1';
```

```
\|
```

```
\c empresa
```

Criando uma tabela:

```
create table funcs (  
id serial primary key not null,  
nome varchar(50) not null,  
tipo varchar(20) not null default 'junior',  
salario numeric(7,2) default 0  
);
```

```
\dt
```

```
\d funcs
```

```
insert into funcs (tipo) values ('senior');
```

```
insert into funcs (nome) values ('João');
```

Criando uma tabela relacionada:

```
create table fones (  
  tipo char(3),  
  nro varchar(8) not null,  
  func integer references funcs (id)  
    on delete restrict on update cascade  
);
```

```
\d fones
```

```
insert into funcs (nome) values ('Fulano');  
insert into funcs (nome) values ('Beltrano');  
insert into fones values ('res','12345',1);  
delete from funcs where id=1;  
delete from funcs where id=2;
```

Transação:

```
begin;  
update funcs set nome='Juca' where id=1;  
insert into funcs(nome) values ('Maria');  
commit;
```

(Se algum erro ocorrer antes do “commit”, o que foi feito a partir do “begin” é desfeito)

Tabela temporária:

```
create temporary table funcs2 as  
select nome, tipo from funcs;
```

```
\dt
```

```
select * from funcs2;
```

```
\q
```

```
psql empresa
```

```
\dt (tabela funcs2 não existe mais)
```

Visões:

```
create view vfunc as  
  select id,nome from funcs;
```

```
\dv  
select * from vfunc;
```

```
update vfunc set nome='Rita' where id=3;
```

```
create rule atualiza as  
  on update to vfunc do instead  
  update funcs set nome=new.nome  
  where id=new.id;
```

```
update vfunc set nome='Rita' where id=3;
```

Funções armazenadas:
create language plpgsql;

```
create function resultado (integer) returns text as  
'  
begin  
  if ($1 > 7) then  
    return "Aprovado";  
  else  
    return "Reprovado";  
  end if;  
end;  
'  
  
LANGUAGE plpgsql;  
  
select resultado(8);
```

Gatilhos:

```
create function insfone() returns trigger as
```

```
|
```

```
begin
```

```
  insert into fones values ('---','111',new.id);
```

```
  return new;
```

```
end;
```

```
|
```

```
language plpgsql;
```

```
create trigger tr_insfone
```

```
after insert on funcs
```

```
for each row execute procedure insfone();
```

Backup, faxina e análise:

```
~postgres/.pgpass
```

```
*:*:*:postgres:senha123
```

(formato: hostname:port:database:username:password)

No cron:

```
pg_dumpall -U postgres > meubkp.sql
```

```
vacuumdb -a -f -z -U postgres
```

(para único banco pg_dump...)

Restauração:

```
psql template1 < meubkp.sql
```

Segurança (grant):

```
create group gerentes;  
create group vendas;
```

```
create user fulano with password '123'  
  in group gerentes;  
create user siclano with password '456'  
  in group vendas;
```

```
grant all on funcs to group gerentes;  
grant select, insert on funcs to group vendas;
```

```
revoke all on funcs from group vendas;
```

```
\z (visualiza as permissões)
```

Segurança (acesso remoto):

/etc/postgresql/8.1/main/pg_hba.conf

```
# TYPE          DATABASE USER  CIDR-ADDRESS  METHOD
local          all                                     postgres
trust
local          all                                     all
              md5
host           all                                     127.0.0.1/32
              md5
host           empresa +gr1 143.10.20.0/24 md5
```

PostgreSQL



www.postgresql.org

Obrigado !!!

André Luiz Fortunato da Silva
Analista de Sistemas
CIRP / USP
alf@cirp.usp.br