Treinamento PostgreSQL - Aula 03

Eduardo Ferreira dos Santos

SparkGroup
Treinamento e Capacitação em Tecnologia
eduardo.edusantos@gmail.com
eduardosan.com

29 de Maio de 2013

Cronograma

Semana 1: 27 de Maio a 4 de Junho Administração de Dados

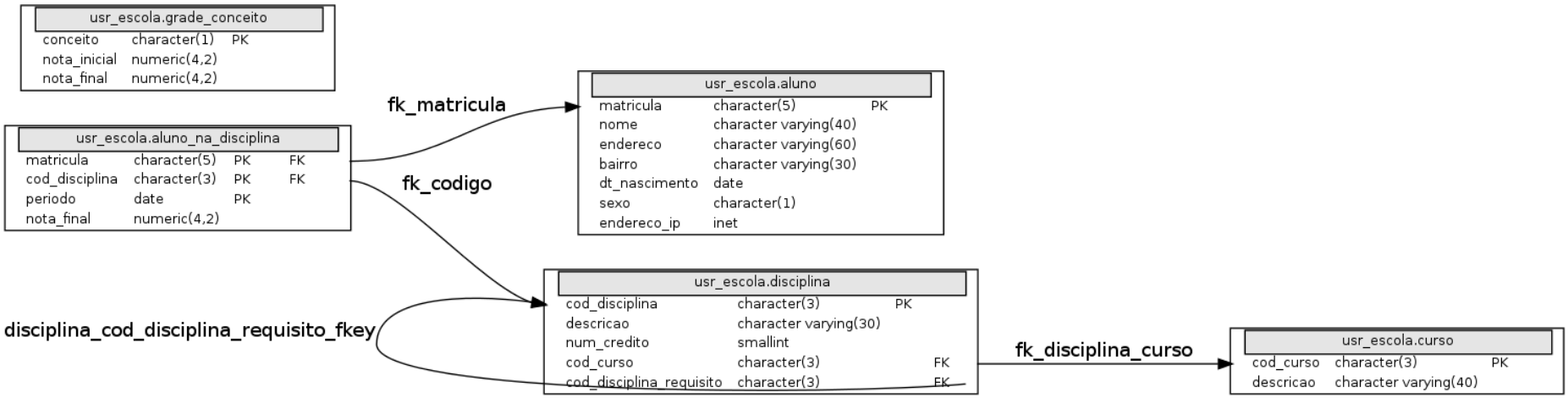
Semana 2: 5-11 de Junho Administração de Banco de Dados

Semana 3: 13-18 de Junho Alta disponibilidade

Semana 4: 19-24 de Junho Performance Tuning

Sumário

Definindo a base de dados



Modelo da base de dados

Relembrando a criação da base

Listing 1: Cria usuário, banco e SCHEMA

```
PostgreSQL> createuser usuario
Shall the new role be a superuser? (y/n) n
Shall the new role be allowed to create databases? (y/n) n
Shall the new role be allowed to create more new roles? (y/n) n

PostgreSQL> createdb -O usuario primeiro_banco
CREATE DATABASE

PostgreSQL> psql -U usuario primeiro_banco

psql (9.1.9)
Type "help" for help.

primeiro_banco=> CREATE SCHEMA usr_escola;
CREATE SCHEMA

primeiro_banco=>\q
quit
```

Permissões

Listing 2: Ajusta permissões

```
vim /etc/postgresql/9.0/main/pg_hba.conf

# A seguinte linha:
local   all             postgres    ident

# Altera para ficar assim:
local   all             postgres    ident
local   primeiro_banco          usuario         trust

logout

# Como root, reinicie o banco
/etc/init.d/postgresql restart
su - postgres
```

Cria estrutura da base

- Primeiro baixe os arquivos SQL:
  - Estrutura da base: `https://seacloud.cc/f/20677b5257/`
  - Carga inicial de dados: `https://seacloud.cc/f/20677b5257/`
- Em seguida execute os arquivos no banco de dados:

Listing 3: Cria base

```
PostgreSQL> psql -U usuario -f cria-banco.sql primeiro_banco
BEGIN
SET
psql:/tmp/cria-banco.sql:12: NOTA:  CREATE TABLE / PRIMARY KEY ácriar índice
      íimplcito "pk_aluno" na tabela "aluno"
CREATE TABLE
psql:/tmp/cria-banco.sql:20: NOTA:  CREATE TABLE / PRIMARY KEY ácriar índice
      íimplcito "pk_disciplina" na tabela "disciplina"
CREATE TABLE
psql:/tmp/cria-banco.sql:28: NOTA:  CREATE TABLE / PRIMARY KEY ácriar índice
      íimplcito "pk_aluno_na_disciplina" na tabela "aluno_na_disciplina"
CREATE TABLE
psql:/tmp/cria-banco.sql:33: NOTA:  CREATE TABLE / PRIMARY KEY ácriar índice
      íimplcito "pk_curso" na tabela "curso"
CREATE TABLE
psql:/tmp/cria-banco.sql:40: NOTA:  CREATE TABLE / PRIMARY KEY ácriar índice
      íimplcito "pk_conceito" na tabela "grade_conceito"
CREATE TABLE
ALTER TABLE
COMMIT
```

Carga inicial dos dados

Listing 4: Alimenta base

```
áííáííáííáííáíí
PostgreSQL> psql -U usuario -f cria-banco.sql primeiro_banco
BEGIN
SET
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
(..)
INSERT 0 1
INSERT 0 1
INSERT 0 1
COMMIT
```

Junções (JOIN)

- Junção de duas tabelas (reais ou derivadas) de acordo com as regras do tipo particular de JOIN:

  INNER JOIN A tabela resultado contém todos os resultados de ambas as tabelas que satisfazam a condição fornecida;

  OUTER JOIN A tabela resultado contém todas os resultados de ambas as tabelas, ainda que a condição não seja satisfeita;

  CROSS JOIN Produto cartesiano dos campos das duas tabelas.

INNER JOIN

Listing 5: Primeiro exemplo de INNER JOIN

```
SELECT d.cod_disciplina,
       d.descricao,
       c.cod_curso,
       d.cod_disciplina_requisito,
       c.descricao
FROM usr_escola.disciplina d
INNER JOIN usr_escola.curso c ON d.cod_curso = c.
    cod_curso
ORDER BY c.cod_curso;
```

INNER JOIN

Listing 6: Segundo exemplo de INNER JOIN

```
SELECT d.descricao,
       c.cod_curso,
       c.descricao,
       a.matricula
FROM usr_escola.disciplina d
INNER JOIN usr_escola.curso c ON d.cod_curso = c.
    cod_curso
INNER JOIN usr_escola.aluno_na_disciplina a ON d.
    cod_disciplina = a.cod_disciplina
ORDER BY c.cod_curso;
```

INNER JOIN

Listing 7: Terceiro exemplo de INNER JOIN

```
SELECT d.descricao,
       c.cod_curso,
       c.descricao,
       a.matricula,
       al.nome
FROM usr_escola.disciplina d
INNER JOIN usr_escola.curso c ON d.cod_curso = c.
    cod_curso
INNER JOIN usr_escola.aluno_na_disciplina a ON d.
    cod_disciplina = a.cod_disciplina
INNER JOIN usr_escola.aluno al ON al.matricula = a.
    matricula
ORDER BY c.cod_curso;
```

INNER JOIN

Listing 8: INNER JOIN com filtro

```
SELECT d.descricao,
       c.cod_curso,
       c.descricao,
       a.matricula,
       al.nome
FROM usr_escola.disciplina d
INNER JOIN usr_escola.curso c ON d.cod_curso = c.
    cod_curso
INNER JOIN usr_escola.aluno_na_disciplina a ON d.
    cod_disciplina = a.cod_disciplina
INNER JOIN usr_escola.aluno al ON al.matricula = a.
    matricula
WHERE c.cod_curso = 'MAT';
```

Considerações sobre performance

- Carregue o PGAdmin[1] e execute a consulta no banco de dados;
- Execute a consulta no modo Analyze;
- O que foi "custoso" para a consulta?

[1] Ferramenta de administração para bases de dados PostgreSQL: http://www.pgadmin.org/

Considerações sobre performance

- Carregue o PGAdmin[1] e execute a consulta no banco de dados;
- Execute a consulta no modo Analyze;
- O que foi "custoso" para a consulta?
- Aplique o filtro utilizando o exemplo 8. O que mudou?

[1] Ferramenta de administração para bases de dados PostgreSQL: `http://www.pgadmin.org/`

Considerações sobre performance

- Carregue o PGAdmin[1] e execute a consulta no banco de dados;
- Execute a consulta no modo Analyze;
- O que foi "custoso" para a consulta?
- Aplique o filtro utilizando o exemplo 8. O que mudou?
- Utilize o comando EXPLAIN ANALYZE;
- Copie e cole os resultados no site `http://explain.depesz.com/`
- Quais os procedimentos que mais consomem recursos do banco de dados?

[1] Ferramenta de administração para bases de dados PostgreSQL: `http://www.pgadmin.org/`

OUTER JOIN

Listing 9: Exemplo de OUTER JOIN

```
SELECT d.cod_disciplina,
       d.descricao,
       c.cod_curso,
       d.cod_disciplina_requisito
FROM usr_escola.disciplina d
FULL OUTER JOIN usr_escola.curso c ON d.cod_curso =
    c.cod_curso;
```

OUTER JOIN

Listing 10: Exemplo de OUTER JOIN com filtro

```
SELECT d.cod_disciplina,
       d.descricao,
       d.cod_disciplina_requisito,
       c.cod_curso
FROM usr_escola.disciplina d
FULL OUTER JOIN usr_escola.curso c ON d.cod_curso =
    c.cod_curso
WHERE c.cod_curso = 'DAN';
```

LEFT OUTER JOIN

Listing 11: Exemplo de LEFT OUTER JOIN

```
SELECT d.cod_disciplina,
       d.descricao,
       c.cod_curso,
       d.cod_disciplina_requisito
FROM usr_escola.disciplina d
LEFT OUTER JOIN usr_escola.curso c ON d.cod_curso =
    c.cod_curso;
```

LEFT OUTER JOIN

Listing 12: Exemplo de LEFT OUTER JOIN com filtro

```
SELECT d.cod_disciplina,
       d.descricao,
       c.cod_curso,
       d.cod_disciplina_requisito
FROM usr_escola.disciplina d
LEFT JOIN usr_escola.curso c ON d.cod_curso = c.
    cod_curso
WHERE c.cod_curso = 'DAN';
```

Exercício

Considerando o modelo da figura 5, construa uma consulta que traga a lista de alunos e a lista de disciplinas, incluindo os alunos que não estão matriculados em nenhuma disciplina.

Solução

Listing 13: Exercício 1

```
SELECT a.matricula AS m1,
       a.nome,
       al.matricula AS m2
FROM usr_escola.aluno a
LEFT OUTER JOIN usr_escola.aluno_na_disciplina al ON
       a.matricula = al.matricula;
```

Exercício

Adicione a informação do nome da disciplina ao exemplo anterior

Solução

Listing 14: Exercício 2

```
SELECT a.matricula AS m1,
       a.nome,
       al.matricula AS m2,
       d.descricao,
       al.cod_disciplina
FROM usr_escola.aluno a
LEFT OUTER JOIN usr_escola.aluno_na_disciplina al ON
       a.matricula = al.matricula
LEFT OUTER JOIN usr_escola.disciplina d ON al.
    cod_disciplina = d.cod_disciplina
```

Exercício

Troque o segundo LEFT OUTER JOIN por INNER JOIN e veja o que acontece.

Solução

Listing 15: Exercício 3

```
SELECT a.matricula AS m1,
       a.nome,
       al.matricula AS m2,
       d.descricao,
       al.cod_disciplina
FROM usr_escola.aluno a
LEFT OUTER JOIN usr_escola.aluno_na_disciplina al ON
       a.matricula = al.matricula
INNER JOIN usr_escola.disciplina d ON al.
     cod_disciplina = d.cod_disciplina
```

Introdução

- Objetivos:
  - Realizar operações com tipos de dados diferentes;
  - Comparar valores em diferentes formatos;
  - Conversão de dados;
  - Operadores complexos e dados abstratos.

Operadores de data

Listing 16: Conversão de dados

```
SELECT cod_disciplina, to_char(periodo, 'MM/YYYY')
    as mes
FROM usr_escola.aluno_na_disciplina;
```

Operadores de data

Listing 17: Operadores de data

```
SELECT cod_disciplina, age(periodo) as mes
FROM usr_escola.aluno_na_disciplina;

SELECT cod_disciplina, age(now(), periodo) as mes
FROM usr_escola.aluno_na_disciplina;
```

Operadores de data

Listing 18: Funções de data

```
SELECT clock_timestamp(),
    current_date,
    current_timestamp,
    localtime,
    localtimestamp,
    now(),
    statement_timestamp(),
    timeofday(),
    transaction_timestamp();
```

Adiciona tipo

Listing 19: Insere tipo inet

```
BEGIN;

ALTER TABLE usr_escola.aluno
ADD COLUMN endereco_ip inet;

UPDATE usr_escola.aluno
SET endereco_ip = '192.168.1.1'
WHERE matricula = '1';

UPDATE usr_escola.aluno
SET endereco_ip = '192.168.1.2'
WHERE matricula = '2';

UPDATE usr_escola.aluno
SET endereco_ip = '192.168.1.3'
WHERE matricula = '3';

UPDATE usr_escola.aluno
SET endereco_ip = '192.168.1.100'
WHERE matricula = '4';

UPDATE usr_escola.aluno
SET endereco_ip = '192.168.2.1'
WHERE matricula = '5';

END;
```

Funções

Listing 20: Funções básicas

```
SELECT abbrev(endereco_ip),
       broadcast(endereco_ip),
       family(endereco_ip),
       host(endereco_ip),
       netmask(endereco_ip)
FROM usr_escola.aluno;
```

Operadores

Listing 21: Operadores para endereço IP

```
SELECT matricula,
       nome
FROM usr_escola.aluno
WHERE endereco_ip = '192.168.1.1'::inet;

SELECT matricula,
       nome
FROM usr_escola.aluno
WHERE endereco_ip >= '192.168.2.1'::inet;

SELECT matricula,
       nome
FROM usr_escola.aluno
WHERE endereco_ip < '192.168.2.1'::inet;
```

Contato

Eduardo Ferreira dos Santos
Sparkgroup
Lightbase Consultoria em Software Público

eduardo.santos@lightbase.com.br
eduardo.edusantos@gmail.com

www.postgresql.org.br
www.eduardosan.com

+55 61 3347-1949