



SQL - Banco de Dados I



Introdução

- **Structured Query Language**
- Linguagem padrão de acesso aos dados no modelo relacional
- Vários padrões existentes:
 - ANSI* SQL, SQL92 (a.k.a. SQL2), SQL99 (a.k.a. SQL3),

*American National Standard Institute

Introdução - Benefícios

- Custo de treinamento reduzido
- Produtividade
- Portabilidade
- Longevidade
- Reduz dependência para um tipo específico de SGBD
- Facilita a comunicação inter-sistemas

Introdução - Ambiente

- Catálogo (Dicionário)
 - Um banco de dados utilizado pelo SGBD para armazenar os dados dos objetos do banco de dados (tabelas, usuários, índices)

employee_id	first_name	last_name	nin	department_id
44	Simon	Martinez	HH 45 09 73 D	1
45	Thomas	Goldstein	SA 75 35 42 B	2
46	Eugene	Comelsen	NE 22 63 82	2
47	Andrew	Petculescu	XY 29 87 61 A	1
48	Ruth	Stadick	MA 12 89 36 A	15
49	Bary	Scardelis	AT 20 73 18	2
50	Sidney	Hunter	HW 12 94 21 C	6
51	Jeffrey	Evans	LX 13 26 39 B	6
52	Doris	Bemdt	YA 49 88 11 A	3
53	Diane	Eaton	BE 08 74 68 A	1
54	Bonnie	Hall	WW 53 77 68 A	15
55	Taylor	Li	ZE 55 22 80 B	1

Data

Column	Data Type	Description
employee_id	int	Primary key of a table
first_name	nvarchar(50)	Employee first name
last_name	nvarchar(50)	Employee last name
nin	nvarchar(15)	National Identification Number
position	nvarchar(50)	Current position title, e.g. Secretary
department_id	int	Employee department. Ref: Department
gender	char(1)	M = Male, F = Female, Null = unknown
employment_start_date	date	Start date of employment in organization.
employment_end_date	date	Employment end date. Null if employee is

Metadata

- Esquema
 - Estrutura que contém descrições dos objetos criados pelo usuário (também é armazenado no catálogo)

Introdução - Ambiente

- Data Definition Language (**DDL**)
 - Conjunto de comandos para gerenciar os objetos do BD
 - create table, create view, create index
- Data Manipulation Language (**DML**)
 - Conjunto de comandos para consultar e manter os dados do banco de dados
 - select, update, insert, ...
- Data Control Language (**DCL**)
 - Conjunto de comandos para controlar o banco de dados, definir privilégios, entre outros
 - create user, grant, revoke, ...

DDL

- Como criar banco de dados e tabelas:
 - Conexão psql (postgresql)
 - **psql -h <servidor> -U <usuario> -d <banco>**

```
dduarte@dduarte:~$ psql -U postgres -h localhost -d postgres
Password for user postgres:
psql (10.14 (Ubuntu 10.14-0ubuntu0.18.04.1))
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off)
Type "help" for help.
```

postgres=#

tipo de usuário
conectado: super usuário

nome do
banco

tipo de
espera:

pronto para receber um novo comando

DDL

- Como criar banco de dados e tabelas:
 - Conexão psql (postgresql)
 - **psql -h <servidor> -U <usuario> -d <banco>**

```
dduarte@dduarte:~$ psql -U dduarte -h localhost -d postgres
Password for user dduarte:
psql (10.14 (Ubuntu 10.14-0ubuntu0.18.04.1))
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off)
Type "help" for help.

postgres=> select
postgres->
```

tipo de usuário
conectado: normal

nome do
banco

tipo de
espera:

esperando a continuação do comando

DDL

- Criar um banco de dados
 - create database <nome>;
 - Exemplo: create database locadora;
- Como conectar:
 - \c <nome>, exemplo \c locadora

```
dduarte@dduarte:~$ psql -U postgres -h localhost -d postgres
Password for user postgres:
psql (10.14 (Ubuntu 10.14-0ubuntu0.18.04.1))
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off)
Type "help" for help.

postgres=# create database locadora;
CREATE DATABASE
postgres=# \c locadora
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off)
You are now connected to database "locadora" as user "postgres".
locadora=#
```

banco
conectado

DDL

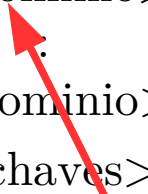
- Criação de tabelas

```
create table <nome> (  
    <atributo1> <dominio> <restrição nulo>,  
        :           :           :  
    <atributon> <dominio> <restrição nulo>,  
    <restrições de chaves>);
```

DDL

- Criação de tabelas

```
create table <nome> (  
    <atributo1> <dominio> <restrição nulo>,  
    : : :  
    <atributon> <dominio> <restrição nulo>,  
    <restrições de chaves>);
```



integer – 4 bytes

real – 4 bytes (6 decimais)

number(tamanho, casas decimais) → (6,4) → 99.9999

varchar (n) – n é tamanho máximo da sequência

date – datas (ver formato configurado no banco)

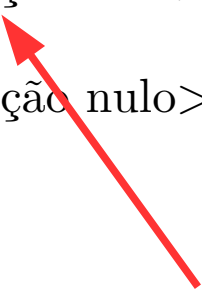
serial – incrementa automaticamente o atributo

DDL

- Criação de tabelas

```
create table <nome> (  
    <atributo1> <dominio> <restrição nulo>,  
    :           :           :  
    <atributon> <dominio> <restrição nulo>,  
    <restrições de chaves>);
```

not null ou **null** (default)



DDL

- Criação de tabelas

```
create table <nome> (  
    <atributo1> <dominio> <restrição nulo>,  
        :           :           :  
    <atributon> <dominio> <restrição nulo>,  
    <restrições de chaves>  
);
```



Onde se define as chaves primária, estrangeira e candidata:

constraint <nome> <tipo> <atributo> <referência>

Chave primária:

constraint pk_tabela **primary key** (atributo (s))

Chave estrangeira:

constraint fk_tabelas **foreign key** (atributo(s)) **references** tabela(chave)

Chave candidata:

constraint uk_atrib_tab **unique** (atributo(s))

DDL

- Banco de dados do exercício

cliente(codcli,cpf,nome,email,ender,ntel)

veiculo(placa,modelo,ano,cor,kilom)

tarifa(descr,valor)

locacao(placa(veiculo), codcli(cliente), dtloca, dtdevo, tarifa (tarifa))

```
create table cliente(  
  codcli integer not null,  
  cpf varchar(11) not null,  
  nome varchar(30) not null,  
  email varchar(30) not null,  
  ender varchar(40) not null,  
  ntel integer not null,  
  constraint pk_cliente primary key (codcli)  
);
```

DDL

- Banco de dados do exercício

cliente(codcli, cpf, nome, email, ender, ntel)

veiculo(placa, modelo, ano, cor, kilom)

tarifa(descr, valor)

locacao(placa(veiculo), codcli(cliente), dtloca, dtdevo, tarifa (tarifa))

```
create table cliente(  
  codcli integer not null,  
  cpf varchar(11) not null,  
  nome varchar(30) not null,  
  email varchar(30) not null,  
  ender varchar(40) not null,  
  ntel integer not null,  
  constraint pk_cliente primary key (codcli)  
);
```

e as outras chaves? cpf e email

DDL

- Banco de dados do exercício

cliente(codcli,cpf,nome,email,ender,ntel)

veiculo(placa,modelo,ano,cor,kilom)

tarifa(descr,valor)

locacao(placa(veiculo), codcli(cliente), dtloca, dtdevo, tarifa (tarifa))

```
create table cliente(  
    codcli integer not null,  
    cpf varchar(11) not null,  
    nome varchar(30) not null,  
    email varchar(30) not null,  
    ender varchar(40) not null,  
    ntel integer not null,  
    constraint pk_cliente primary key (codcli),  
    constraint uk_cpf_cliente unique (cpf),  
    constraint uk_email_cliente unique (email)  
);
```

DDL

- Banco de dados do exercício

cliente(codcli,cpf,nome,email,ender,ntel)

veiculo(**placa**,modelo,ano,cor,kilom)

tarifa(descr,valor)

locacao(placa(veiculo), codcli(cliente), dtloca, dtdevo, tarifa (tarifa))

create table veiculo(

placa varchar(10) not null,

modelo varchar(11) not null,

ano numeric(4,0) not null,

cor varchar(10) not null,

kilom numeric(6,0),

constraint pk_veiculo **primary key** (**placa**)

);

DDL

- Banco de dados do exercício

cliente(codcli,cpf,nome,email,ender,ntel)

veiculo(placa,modelo,ano,cor,kilom)

tarifa(descr,valor)

locacao(placa(veiculo), codcli(cliente), dtloca, dtdevo, tarifa(tarifa))

```
create table tarifa(  
    descr varchar(10) not null,  
    valor numeric(6,2) not null,  
    constraint pk_tarifa primary key (descr)  
);
```

DDL

- Banco de dados do exercício

locacao(placa(veiculo), codcli(cliente), dtloca, dtdevo, tarifa(tarifa))

```
create table locacao(  
  placa varchar(10) not null,  
  codcli integer not null,  
  dtloca date not null,  
  dtdevo date,  
  tarifa varchar(10) not null,  
  constraint pk_locacao primary key (placa,codcli,dtloca),  
  constraint fk_locacao_veiculo foreign key (placa) references veiculo(placa),  
  constraint fk_locacao_cliente foreign key (codcli) references cliente(codcli),  
  constraint fk_locacao_tarifa foreign key (tarifa) references tarifa(descr)  
);
```

```
create table veiculo(  
  placa varchar(10) not null,  
  modelo varchar(11) not null,  
  ano numeric(4,0) not null,  
  cor varchar(10) not null,  
  kilom numeric(6,0),  
  constraint pk_veiculo primary key (placa)  
);
```

```
create table tarifa(  
  descr varchar(10) not null,  
  valor numeric(6,2) not null,  
  constraint pk_tarifa primary key (descr)  
);
```

```
create table cliente(  
  codcli integer not null,  
  cpf varchar(11) not null,  
  nome varchar(30) not null,  
  email varchar(30) not null,  
  ender varchar(40) not null,  
  ntel integer not null,  
  constraint pk_cliente primary key (codcli),  
  constraint uk_cpf_cliente unique key (cpf),  
  constraint uk_email_cliente unique key (email)  
);
```

DML

- Inserir dados nas tabelas
 - comando insert:
 - insert into <tabela> (<atributos>) values (<valores>);
 - em <atributos> listar os atributos que receberão valores
- insert into tarifa (descr,valor) values ('blue',80);
- em grupo:
- insert into tarifa (descr,valor) values ('green',40), ('white',50);

```
create table tarifa(  
  descr varchar(10) not null,  
  valor numeric(6,2) not null,  
  constraint pk_tarifa primary key (descr)  
);
```

- * datas são inseridas como sequência de caracteres e obedecendo o formato configurado no SGBD. Por exemplo, '21/10/2020' ou '10/21/2020' ou '21-10-2020'.
- Em psql verifica-se o estilo da data com **show datestyle**;
 - Para alterar **set datestyle='novo estilo'**; , estilo pode ser: **ISO**, **DMY**

DML

- Inserir dados nas tabelas
 - Se as 3 tuplas anteriores existissem no banco de dados e o seguinte comando fosse executado:
 - `insert into tarifa (descr,valor) values ('blue',90.5);`
O seguinte erro seria invocado:
`ERROR: duplicate key value violates unique constraint "pk_tarifa"`
`DETAIL: Key (descr)=(‘blue’) already exists.`

DML

- Inserir dados nas tabelas
 - Se o programador tentar inserir a seguinte tupla na tabela locacao:

```
insert into locacao (placa,codcli,dtloca,tarifa)  
values ('MK-10','C22','15/10/2020','black');
```

e não existisse o valor **black** na tabela tarifa, teríamos o erro:

ERROR: insert or update on table "locacao" violates foreign key constraint
"fk_locacao_tarifa"

DETAIL: Key (tarifa)=(**black**) is not present in table "tarifa".

DML

- Consultas
 - Comando select
 - **select** [distinct] <lista atributos> **from** <tabela> **where** <filtro> **order by** <atributo>

- select **modelo, placa** from veiculo where **ano > 2019;**

modelo	placa
HR-V	XX-77

veiculo				
placa	modelo	ano	cor	kilom
MK-10	Audi Q3	2019	branco	45455
KK-10	Ka	2018	prata	76890
XX-77	HR-V	2020	vermelho	21888
TT-88	QQ	2018	branco	78000
JJ-34	HB20	2019	branco	45672
OK-01	Celta	2015	prata	98673

projeção

seleção

resultado

DML

- Consultas
 - Comando select
 - select [distinct] <lista atributos> from <tabela> where <filtro> order by <atributo>

```
select cor from veiculo;
```

```
cor
```

```
-----
```

```
branco  
prata  
vermelho  
branco  
branco  
prata
```

veiculo

placa	modelo	ano	cor	kilom
MK-10	Audi Q3	2019	branco	45455
KK-10	Ka	2018	prata	76890
XX-77	HR-V	2020	vermelho	21888
TT-88	QQ	2018	branco	78000
JJ-34	HB20	2019	branco	45672
OK-01	Celta	2015	prata	98673

```
select distinct cor from veiculo;
```

```
cor
```

```
-----
```

```
branco  
prata  
vermelho
```

DML

- Consultas
 - Exemplos cláusula **where**:

```
select * from veiculo;
```

like (% e _):

```
select placa from veiculo
       where placa like '%K%';
```

placa

MK-10
KK-10
OK-01

```
select placa from veiculo
       where placa like '_K-10'
```

placa

MK-10
KK-10

veiculo				
placa	modelo	ano	cor	kilom

MK-10	Audi Q3	2019	branco	45455
KK-10	Ka	2018	prata	76890
XX-77	HR-V	2020	vermelho	21888
TT-88	QQ	2018	branco	78000
JJ-34	HB20	2019	branco	45672
OK-01	Celta	2015	prata	98673

```
select placa, modelo from veiculo
       where ano in (2015,2020);
```

placa modelo

XX-77 HR-V
OK-01 Celta

```
select placa, modelo from veiculo
       where ano between 2015 and 2018;
```

placa modelo

KK-10 Ka
TT-88 QQ
OK-01 Celta

DML

- Consultas
 - Exemplos cláusula **where**:

veiculo				
placa	modelo	ano	cor	kilom
MK-10	Audi Q3	2019	branco	45455
KK-10	Ka	2018	prata	76890
XX-77	HR-V	2020	vermelho	21888
TT-88	QQ	2018	branco	78000
JJ-34	HB20	2019	branco	45672
OK-01	Celta	2015	prata	98673

```
select modelo, kilom from veiculo order by modelo asc;
```

modelo	kilom
Audi Q3	45455
Celta	98673
HB20	45672
HR-V	21888
Ka	76890
QQ	78000

```
select modelo, kilom from veiculo order by modelo desc;
```

modelo	kilom
QQ	78000
Ka	76890
HR-V	21888
HB20	45672
Celta	98673
Audi Q3	45455

DML

- Consultas
 - Postgres possui uma série de funções que podem ser utilizadas tanto na projeção quanto na seleção:
 - `lower()` / `upper()`: transforma string em minúsculo/maiúsculo
 - `select lower(nome) from cliente;`
 - `left/right` (string, # de caracteres)
 - `select left(nome,4) from cliente;` – retorna os 4 primeiros caracteres
 - `extract` (valor from atributo):
 - `select extract(day from dtloc) from locacao;` – extrai apenas o dia da locação
 - `length`(string): retorna o tamanho do string
 - `select modelo, length(modelo) from veiculo;`