Câmpus
Anápolis de Ciências
Exatas e Tecnológicas
Henrique Santillo

Universidade
Estadual de Goiás

Curso: Sistemas de Informação
Disciplina : Banco de Dados II

SQL – Structure Query Language
DDL – Data Definition Language

Prof. M.e. Guiliano Rangel Alves

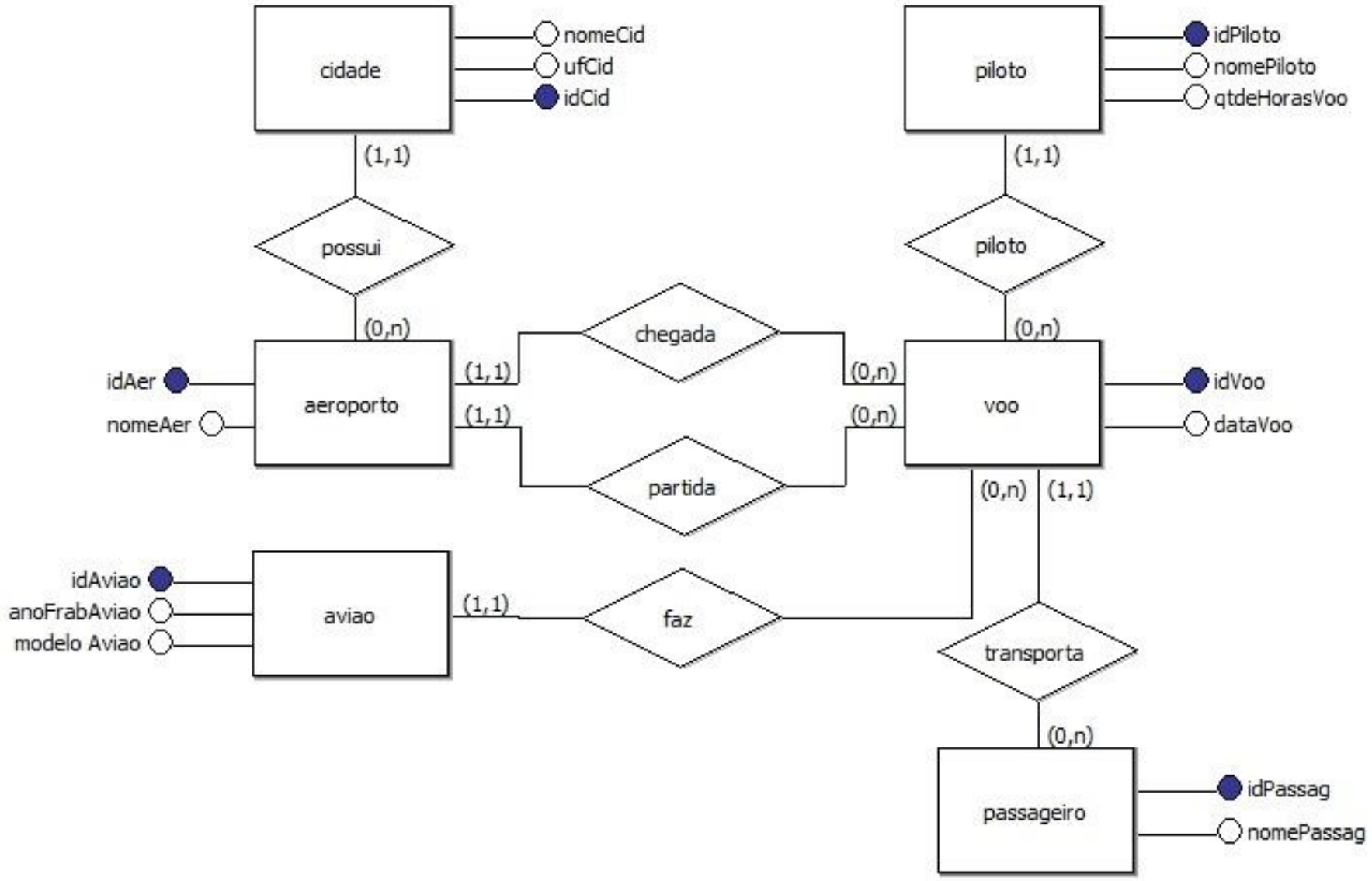
Sumário

- Conceitos
  - DDL – Data Definition Language
  - DML – Data Manipulation Language
  - DQL – Data Query Language

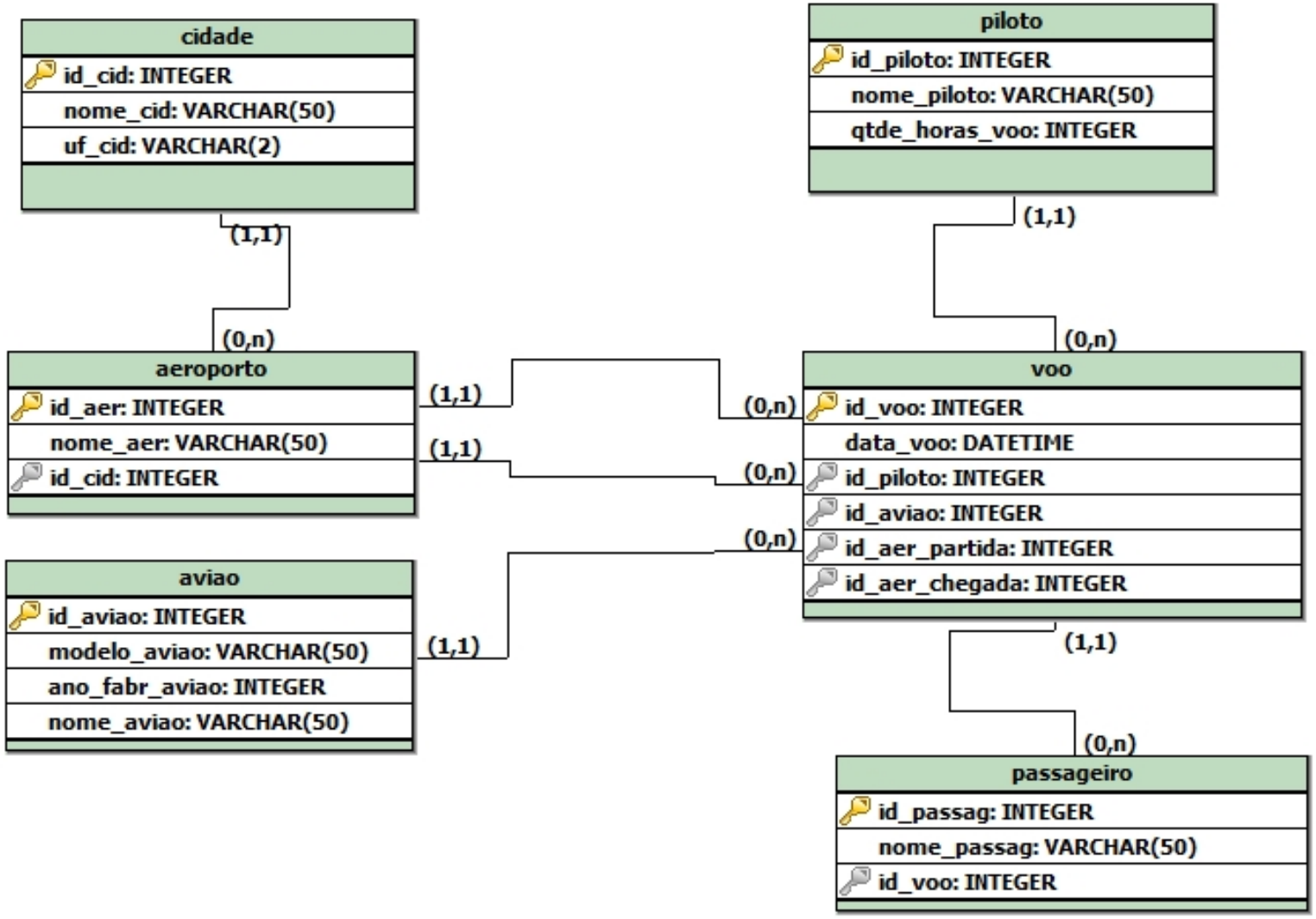
Conceito

- SQL – Structure Query Language
  - Linguagem de quarta geração para auxiliar os programadores de aplicação para criar modelos de interface com usuário e formatar dados para relatórios.
  - Disponível na maioria do Banco de Dados comerciais.

Modelo de Entidade e Relacional



Modelo Relacional



Data Definition Language (DDL)

- Linguagem de Definição de Dados
- Permite a especificação de um conjunto de relações e informações sobre cada relação, incluindo:
- O schema (esquema) para cada relação.
- O domínio de cada valor associado com cada atributo.
- Restrição de Integridade.
- O conjunto de índices para ser mantido para cada relação.
- Informação sobre segurança e autorização para cada relação.
- A estrutura física de cada relação no disco

Tipos de domínios em SQL

- **char(n)**. Texto de tamanho fixo, com tamanho especificado pelo usuário (n).
- **varchar(n)**. Texto de tamanho variável com tamanho máximo especificado pelo usuário (n).
- **int**. Inteiro (um subconjunto finito dos números inteiros que é dependente da plataforma).
- **smallint**. Inteiro pequeno (sub conjunto do domínio dos inteiros - *integer*, dependente da máquina).
- **numeric(p,d)**. Número real, ponto fixo, com a especificação do usuário de *p* dígitos com *n* dígitos à direita do ponto decimal.

Tipos de domínios em SQL

- **real, double precision**. Ponto flutuante e precisão dupla com a precisão dependente de máquina.
- **float(n)**. Número de ponto flutuante, com a precisão especificada pelo usuário (n).
- **date**. Data contendo 4 dígitos para ano, mês e dia.
- **time**. Hora do dia em horas, minutos e segundos.
  - Valores nulos são permitidos em todos os tipos de domínios.
  - Declarando para ser **not null** proíbe-se valores nulos para o atributo.
  - Construtor **create domain** definindo em SQL-92, cria um tipo definido pelo usuário.

CREATE TABLE

- Uma relação SQL é definida utilizado o comando **create table.**

create table r (A_1 D_1,A_2 D_2,...,A_n D_n,

h integrity-constraint $_1$,

...,

h integrity- constraint $_k$)

  - r é o nome da relação
  - cada A_i é um nome de atributo da relação r.
  - D_i é um tipo de dado do domínio para o atributo A

CREATE TABLE - Exemplo

```
CREATE TABLE Cidade (
  idCid INTEGER NOT NULL,
  nomeCid VARCHAR(30),
  ufCid VARCHAR(2),
  PRIMARY KEY(idCid)
);
```

```
CREATE TABLE Aeroporto (
  idAer INTEGER NOT NULL,
  idCid INTEGER NOT NULL,
  nomeAer VARCHAR(30),
  PRIMARY KEY(idAer),
  FOREIGN KEY(idCid)
    REFERENCES Cidade(idCid)
);
```

CREATE TABLE – **Restrição de Integridade**

- **not null**
- **primary key** (A_1,...,A_n)
- Exemplo: Declarar idCid como a chave primária para a tabela Cidade
- A declaração **primary key** sobre um atributo automaticamente assegura que ele não será nulo (**not null**) em SQL-92.

DROP TABLE

- O comando **drop table** exclui toda informação sobre a tabela do banco de dados.
- Exemplo:
  - Drop table Cidade

ALTER TABLE

- O comando **alter table** é utilizado para alterar atributos de uma tabela existente:
  - Adicionar atributo: add
  - Excluir atributo: drop

- Obs: Alguns bancos de dados não aceitam a alteração do nome ou tipo de dado do atributo, sendo necessário excluir e adicionar o novo atributo

ALTER TABLE - Add

- Todas as tuplas da relação recebem o valor nulo para o novo atributo.
- O formato do comando **alter table** é:

  alter table r **add column** A D
  - onde A é o nome do novo atributo da tabela;
  - D é o domínio de A.
- Exemplo:
  - ALTER TABLE Aeroporto ADD COLUMN EnderAer VARCHAR(80) NULL;

ALTER TABLE - Drop

- O comando **alter table** pode também ser utilizado para excluir atributos da relação.

  alter table r **drop column** A

  - onde A é o nome da atributo da relação r.
- Exemplo:
  - ALTER TABLE Aeroporto DROP COLUMN EnderAer;

Restrição de Integridade

- Restrição de Integridade assegura contra danos acidentais sobre o banco de dados, assegurando que mudanças autorizadas sobre o banco de dados não resulte em perda da consistência dos dados.
- Restrição de domínio é a forma mais elementar de restrição.
- Eles testam os valores inseridos no banco de dados e testam as consultas para garantir que as comparações fazem sentido

Restrição de Integridade

- A cláusula **check** definida em SQL-92, permite que os domínios sejam restritos.(04/09)
- Exemplo:
  - Utilizar a cláusula **check** para assegurar que o valor da quantidade de horas de vôo deve ser maior ou igual 10 horas

```
CREATE TABLE Piloto (
  idPiloto INTEGER NOT NULL,
  nomePiloto VARCHAR(50),
  QtdeHorasVoo INTEGER NOT NULL,
  PRIMARY KEY(idPiloto),
  check (QtdeHorasVoo >=10)
);
```

Integridade Referencial

- Chave primária e estrangeira pode ser especificada como parte do comando SQL **create table.**
- A cláusula **primary key** do comando **create table** inclui uma lista de atributos que formam a chave primária.
- A cláusula **foreign key** inclui uma lista de atributos que formam a chave estrangeira e o nome da relação referenciada pela chave primária.

Integridade Referencial – Exemplo

```
CREATE TABLE Aviao (
  idAviao INTEGER NOT NULL,
  nomeAviao VARCHAR(30),
  modeloAviao VARCHAR(20),
  anoFabrAviao INTEGER,
  PRIMARY KEY(idAviao)
);
```

```
CREATE TABLE Passageiro (
  idPassag INTEGER NOT NULL,
  Voo_idVoo INTEGER NOT NULL,
  nomePassag VARCHAR(50),
  PRIMARY KEY(idPassag),
  FOREIGN KEY(Voo_idVoo)
    REFERENCES Voo(idVoo)
);
```

Integridade Referencial – Exemplo

```
CREATE TABLE Voo (
  idVoo INTEGER NOT NULL,
  Aeroporto_partida_idAer INTEGER NOT NULL,
  Aeroporto_chegada_idAer INTEGER NOT NULL,
  Aviao_idAviao INTEGER NOT NULL,
  Piloto_idPiloto INTEGER NOT NULL,
  dataVoo DATE NULL,
  PRIMARY KEY(idVoo),
  FOREIGN KEY(Aviao_idAviao)  REFERENCES Aviao(idAviao),
  FOREIGN KEY(Piloto_idPiloto)  REFERENCES Piloto(idPiloto),
  FOREIGN KEY(Aeroporto_partida_idAer) REFERENCES
Aeroporto(idAer),
  FOREIGN KEY(Aeroporto_chegada_idAer) REFERENCES
Aeroporto(idAer)
);
```

Integridade Referencial - Ações em Cascata

- A cláusula **on delete cascade** faz com que se excluído uma tupla da tabela Cidade que resulta na violação da integridade referencial, exclua também as tuplas na tabela Aeroporto que se referem à cidade que foi excluída.
- O comando **on update cascade** funciona de forma similar.

Integridade Referencial - Ações em Cascata - Exemplo

```
CREATE TABLE Aeroporto (
  idAer INTEGER NOT NULL,
  idCid INTEGER NOT NULL,
  nomeAer VARCHAR(30),
  PRIMARY KEY(idAer),
  FOREIGN KEY(idCid)
    REFERENCES Cidade(idCid)
      ON DELETE CASCADE
      ON UPDATE CASCADE
```

Banco de Dados PostgreSQL

- Instalação:
  - Instale o banco de dados

PostgreSQL 9.4 ou superior(versão estável)

  - Instale a interface de gerenciamento

PGAdmin III ou IV

Portable disponibilizado professor

DML – Data Manipulation Language

- Insert
- Update
- Delete
- Select

Insert

- Incluindo Registros

```
insert into "CIDADE" ("IDCID", "NOMECID","UFCID")
        values (1,'Goiania','GO');

insert into "CIDADE" values (2,'Anapolis','GO')

insert into CIDADE values (3,'Brasilia','DF')
```

- OBS:
  - Cuidado com a dependência entre as tabelas e suas restrições de integridade referencial.

Update

- Alterando Registros

```
update CIDADE
set  NOMECID = 'Aparecida de Goiania'
where IDCID = 1
```

- OBS:
  - Cuidado! Se a condição não for informada, todos os registros serão alterados para o mesmo nome de cidade.

Delete

- Excluindo Registros

```
delete from CIDADE
where IDCID = 3
```

- OBS:
  - Cuidado! Se a condição não for informada, todos os registros da tabela serão excluídos.

Select

- Selecionado Registros
  - Uma consulta típica SQL tem a forma:

```
select A1, A2, ... , An
from r1, r2,...,rm
where P
```

    - A - representa atributos
    - r - representa relações
    - P - é um predicado

  - O resultado de uma consulta SQL é uma relação (tabela)

Exemplo

```
Select IDCID, NOMECID, UFCID
from CIDADE
```

```
Select IDCID, NOMECID, UFCID
from CIDADE
where IDCID = 1
```

Exercícios

- Insira dados em todas as tabelas do BD Aeroporto;
- Faça alterações nos dados do registro;
- Faça exclusões de registros.
- Experimente consultas básicas em cada tabela.

FIM