Aula 6 – BD1
Modelo Relacional

Profa. Elaine Faria
UFU - 2018

Introdução

- Modelo Relacional
  - Proposto por Codd em 1970
  - Revolucionou a área de banco de dados
  - É o modelo dominante → base para os SGBDs líderes de mercado
    - DB2, Oracle, Sybase, Access, SQLServer e FoxBase
  - Simples e elegante
  - Embasado na teoria de conjuntos

Introdução

- Banco de Dados do Modelo Relacional
  - É uma coleção de uma ou mais relações
  - Cada relação é uma tabela com linhas (tupla, registro) e colunas (atributos, campos)
  - Possibilita o uso de linguagens de alto nível para consultar os dados

- Vantagens do Modelo Relacional
  - Representação de dados simples e facilidade para expressar consultas complexas

SQL

- *Structured Query Language*
- Linguagem mais usada para criar, manipular e consultar SGBD relacionais
- DDL - *Data Definition Language*
  - Linguagem de definição de dados
  - Linguagem padrão para criar, manipular e consultar dados em um SGBD relacional

Modelo Relacional

- Representação dos dados → relação

- Uma relação consiste de
  - Esquema da relação
    - Descreve os cabeçalhos de colunas das tabelas
    - Especifica o **nome** da relação, o nome de cada **campo** e o **domínio** de cada campo
      - Domínio: descrito pelo **nome de domínio** e possui um conjunto de **valores** associados
  - Instância da relação
    - É uma tabela , conjunto de **tuplas** (**registros, linha**)
      - Cada tupla tem o mesmo número de campos

Exemplo

- Esquema de uma relação

  Alunos (id_aluno: `string`, nome:`string`, login: `string`, idade: `integer`, média: `real`)

  → Conjunto de valores associados ao domínio `string`: todas as strings de caracteres

Exemplo

Campos (ou atributos ou colunas)

Nome dos campos

id_aluno	nome	login	idade	média
50000	João	joao@ufu	19	3.3
53666	Maria	maria@ufu	18	3.4
53688	Pedro	pedro@comp	18	3.2
53650	Pedro	pedro@mat	19	3.8
53831	Ana	ana@feelt	11	1,8
53832	Tony	tony@mec	12	2.0

Tuplas (registros, linhas)

Características das Tabelas

- Cada coluna tem um nome distinto
- Cada domínio possui valor atômico (indivisível)

- O valor *null* deve ser utilizado quando um atributo não possui valor ou seu valor não é conhecido

- Cada linha é distinta e representa uma tupla
- A ordem das colunas/linhas é irrelevante

- Uma *n*-tupla representa uma tupla que possui *n* valores (*n* é chamado de *grau da relação*)

Esquema de relação

- Domínio de um campo→ *tipo* do campo
- Formalmente

  R(f_1: D1, ... f_n:Dn) → esquema da relacão

  Para cada f_1:1 ≤ i ≤ n, seja Dom_i: o conjunto de valores associados ao domínio D_i

  Cada tupla com n campos

  { <f_1: d_1, ..., f_n:d_n>| $d_1 \in Dom_1$, ..., $d_n \in Dom_n$}

  <> → uma tupla {} → conjunto de tuplas

Esquema de relação

- Nível (aridade, grau) de uma relação
  - Número de campos (atributos)
- Cardinalidade de uma instância da relação
  - Número de tuplas
- Banco de dados relacional
  - Coleção de relações com nomes distintos
- Esquema do banco de dados relacional
  - Coleção de esquemas para as relações no banco de dados
- Instância de um banco de dados relacional
  - Coleção de instâncias das relações (uma por esquema)

Criando e modificando relações usando SQL

- SQL
  - DDL (*Data Definition Language*)
    - Subconjunto do SQL que permite a criação, eliminação e modificação de tabelas

- Criando uma tabela em SQL
  - Usa a palavra *table* para denotar relação

Criando e modificando relações usando SQL

- CREATE TABLE
  - Comando usado para definir uma nova tabela

```
CREATE TABLE Alunos( id_aluno    CHAR(20),
                     nome        CHAR(30),
                     login       CHAR(20),
                     idade       INTEGER,
                     media       REAL)
```

- INSERT
  - Comando usada para inserir tuplas em uma tabela

```
INSERT
INTO Alunos(id_aluno,nome, login, idade, media)
VALUES ('53688', 'Smith', 'smith@ee', 18, 3.2)
```

Obs.: é possível omitir a lista de nomes de colunas na cláusula INTO

Criando e modificando relações usando SQL

- `DELETE`
  - Comando usado para eliminar tuplas

```
DELETE
FROM Alunos
WHERE  nome = ‘Smith’
```

- `UPDATE`
  - Modifica os valores de uma coluna em uma linha existente

```
UPDATE Alunos
SET  idade= idade + 1,media =media-1
WHERE id_aluno = ‘53688’
```

Criando e modificando relações usando SQL

- Cláusula `WHERE`
  - É a primeira a ser executada e determina quais linhas serão modificadas
- Cláusula `SET`
  - Determina como as linhas serão modificadas

```
UPDATE  Alunos A
SET  A.média = A.média -0.1
WHERE A.média >=3.3
```

Tipos de Dados de Atributos em SQL

- Numéricos
  - INTEGER
  - FLOAT
  - DOUBLE
- Cadeia de caracteres
  - CHAR(n)
  - VARCHAR(n)
- Booleano
- Data e tempo
  - Date (data)
  - Time (horário)
- *Timestamp*
  - Date + time + 6 posições para frações decimais de segundos

Valor *Default* e restrição de domínio

- `DEFAULT` <valor>
  - Usado para definir um valor *default* para um atributo, caso não seja fornecido nenhum valor explícito
  - Se não for especificada essa cláusula, o valor default será *null*

- `CHECK`
  - Usado para limitar os valores do atributo ou de seu domínio

    Ex: Numero `INTEGER CHECK` (Numero > 0 `AND` Numero < 21)

Restrições de Integridade sobre Relações

- Restrição de integridade
  - Condição especificada no esquema do BD e restringe os dados que podem ser armazenados em uma instância do BD
    - Se uma instância satisfaz todas as restrições de integridade → instância válida

Um SGB impõe restrições de integridade!

Restrições de Chave

- Declaração de que certo subconjunto mínimo dos campos de uma relação é um identificador único para uma tupla
  - Ex: 2 alunos não podem ter a mesma identificação

- Conjunto de campos que identifica uma tupla de acordo com uma restrição de chave → **chave candidata**
  - Ex: *id_aluno* na relação Alunos

Restrições de Chave

- Na definição de chave
  - Duas duplas distintas em uma instância não podem ter valores idênticos em todos os campos de uma chave
  - Nenhum subconjunto do conjunto de campos em uma chave é um identificador único para uma tupla
    - Ex: {id_aluno,nome} não é uma chave para Alunos, mas é uma **superchave**

Restrições de Chave

- Superchave
  - Subconjunto de atributos de R que identifique univocamente cada tupla
  - Combinação de valores não se repete para a superchave
  - Ex: Aluno = {Nome, Idade, Curso, Id_aluno}
    - SCH1(Aluno) = {Nome, Id_aluno, Idade}
    - SCH2(Aluno) = {Id_aluno, Nome}

- Chave
  - É uma superchave da qual não se pode retirar nenhum atributo e ainda preservar a propriedade de identificação unívoca

Restrições de Chave

- Chave Candidata
  - Pode existir mais de uma chave para uma mesma relação
  - Cada uma das chaves é chamada de **Chave Candidata**
    - CH1(Aluno) = {CPF}
    - CH2(Aluno) = {Id_aluno}
- Chave Primária
  - Escolhida entre as chaves candidatas (não nula)
  - É freqüentemente utilizada para acessos à relação
    - CH(Aluno) = {Id_aluno}

Especificando Restrições de Chave em SQL

- Declaração de que um subconjunto das colunas de uma tabela constituem uma chave → restrição `UNIQUE`
- Chave primária → `PRIMARY KEY`

```
CREATE TABLE Alunos(    id_aluno CHAR(20),
                        nome     CHAR(30),
                        login    CHAR(20),
                        idade    INTEGER,
                        cpf      CHAR(11),
                        media    REAL,
                        UNIQUE (cpf),
                        PRIMARY KEY (id_aluno))
```

Restrições de Chave Estrangeira

- Informações armazenadas em uma relação estão ligadas a informações de outra relação
  - Manutenção de dados consistentes

- RI mais comum envolvendo duas relações → chave estrangeira

Restrições de Chave Estrangeira

Matriculado(*id_aluno*: `string`, *id_disc*: `string`, *nota*: `string`)

- Garantir que apenas estudantes legítimos possam se matricular nas disciplinas
  - Valores do campo *id_aluno* da relação Matriculado deve aparecer no campo *id_aluno* na relação Alunos
  - O campo *id_aluno* de Matriculado é chamado **Chave Estrangeira** e se refere a Alunos

Restrições de Chave Estrangeira

- A chave estrangeira na relação de referência deve corresponder à chave primária (ou a um campo unique) da relação referenciada
  - Mesmo nro de colunas e tipos de dados compatíveis
  - Os nomes das colunas podem ser diferentes

- No exemplo matriculado
  - Todo valor de id_aluno que aparece em Matriculado aparece na coluna de chave primária de Alunos
  - Podem existir tuplas de Alunos que não sejam referenciadas a partir de Matriculado

Restrições de Chave Estrangeira

Chave Estrangeira

Chave Primária

id_disc	nota	id_aluno
Prog	C	53831
Materiais	B	53832
Calculo	A	53650
Historia	B	53666

id_aluno	nome	login	idade	media
50000	João	joao@ufu	19	3.3
53666	Maria	maria@ufu	18	3.4
53688	Pedro	pedro@comp	18	3.2
53650	Pedro	pedro@mat	19	3.8
53831	Ana	ana@feelt	11	1,8
53832	Tony	tony@mec	12	2.0

Restrições de Chave Estrangeira

- Uma chave estrangeira pode referenciar a mesma relação
  - Ex: relação Alunos com uma coluna chamada parceiro
    - Se o aluno não tiver um parceiro → usa-se null

A presença de null em um campo de chave estrangeira não viola a restrição de chave estrangeira

Restrições de Chave Estrangeira

FOREIGN KEY (atributos)

REFERENCES nome_relação (atributos)

[ON UPDATE [NO ACTION | CASCADE | SET NULL | SET DEFAULT]]

[ON DELETE [NO ACTION | CASCADE | SET NULL | SET DEFAULT]]

Especificando restrições de Chave Estrangeira em SQL

```
CREATE TABLE Matriculado(
        id_aluno CHAR(20),
        id_disc  CHAR(20),
        nota     CHAR(10),
        PRIMARY KEY (id_aluno,id_disc),
        FOREIGN KEY (id_aluno)
        REFERENCES Alunos (id_aluno))
```

id_disc também pode ser uma chave estrangeira referenciando uma tabela disciplinas

Verificando Restrições de Integridade

- RIs
  - São especificadas quando uma relação é criada e verificadas quando a mesma é modificada
  - Se um comando causa uma violação, ele é rejeitado
  - Toda violação é verificada no final da execução de cada instrução

- Exemplo

```
INSERT
INTO Alunos (id_aluno, nome, login, idade, media)
VALUES (null, ‘Mike’, ‘mike@ee’, 17,3.4)
```

→ Viola a restrição de chave primária: Valor null no campo chave

Verificando Restrições de Integridade

- Exemplo

```
INSERT
INTO Alunos (id_aluno, nome, login, idade, media)
VALUES (‘53688’, ‘Mike’, ‘mike@ee’, 17,3.4)
```

→ Viola a restrição de chave primária: Já existe o aluno id_aluno 53688

```
UPDATE Alunos A
SET A.id_aluno = ‘50000’
WHERE A.id_aluno = ‘53688’
```

→ Viola a restrição de chave primária: Já existe o aluno id_aluno 50000

Verificando Restrições de Integridade

- **Verificação da integridade referencial**
  - Exemplo: tabelas Matriculado e Alunos com restrição de chave estrangeira id_aluno (Alunos)

```
INSERT
INTO Matriculado (id_disc, nota, id_aluno)
VALUES ('Hindi101', 'B', '51111')
```

→ Viola a integridade referencial: Não há uma tupla em Alunos com id_aluno 51111

Exclusões de tuplas de Matriculado não violam a integridade referencial

Verificando Restrições de Integridade

- **Verificação da integridade referencial**
  - Inserções de tuplas em Alunos não violam a integridade referencial. Exclusões podem causar violações
  - Atualizações em Matriculado ou em Alunos que alteram o valor id_aluno podem violar a integridade referencial

Verificando Restrições de Integridade

- **Verificação da integridade referencial**
  - O que fazer se:

    Uma linha de Matriculado é inserida com um valor na coluna id_aluno que não aparece em nenhuma linha de Alunos
    - O comando `INSERT` é rejeitado

Verificando Restrições de Integridade

- **Verificação da integridade referencial**
  - O que fazer se:

    Uma linha de Alunos é excluída
    - Excluir todas as linhas de Matriculado que referenciam a linha de Alunos excluída
    - Proibir a exclusão da linha de Alunos, caso uma linha de Matriculado a referencie
    - Configurar a coluna id_aluno com o valor de id_aluno de algum aluno padrão
    - Configurar a coluna id_aluno como null
      - Conflito: id_aluno faz parte da chave primária de Matriculado

Verificando Restrições de Integridade

- **Verificação da integridade referencial**
  - O que fazer se o valor da chave primária de uma linha de Alunos for atualizada?
    - Opções semelhantes ao caso da exclusão

- A SQL permite escolher qualquer uma dentre 4 opções citadas em `DELETE` E `UPDATE`

Verificando Restrições de Integridade

```
CREATE TABLE Matriculado(
                    id_aluno CHAR(20),
                    id_disc CHAR(20),
                    nota     CHAR(10),
                    PRIMARY KEY (id_aluno,id_disc),
                    FOREIGN KEY (id_aluno)
                    REFERENCES Alunos
                          ON DELETE CASCADE
                          ON UPDATE NO ACTION)
```

A opção padrão é `NO ACTION`

Verificando Restrições de Integridade

- `NO ACTION`
  - A ação deve ser rejeitada
  - Pode ser omitida
- `CASCADE`
  - Se uma linha de Alunos for excluída todas as linhas de Matriculado que se referem a ela também serão excluídas
- `SET DEFAULT`
  - Se uma linha de Alunos for excluída troca-se a matrícula para um aluno padrão
  - O aluno padrão é especificado como parte da definição do campo id_aluno em Matriculado
    - id_aluno `CHAR(20) DEFAULT` '53666'
- `SET NULL`
  - Permite o uso de *null* como padrão

Constraints: Create Table

- É importante definir as *constraints* associadas à tabela
  - Check
  - Not Null
  - Unique
  - Primary Key
  - Foreign Key

Constraints: Create Table

- Check

```
CREATE TABLE produto (
    nro_produto integer,
    nome text,
    preco numeric CHECK (preco > 0)
);
CREATE TABLE produto (
    nro_produto integer,
    nome text,
    preco numeric CONSTRAINT p_preco CHECK (preco > 0)
);
```

Constraints: Create Table

- Not-null

```
CREATE TABLE produto (
    nro_produto integer NOT NULL,
    nome text NOT NULL,
    preco numeric NOT NULL CHECK (preco > 0)
);
```

Constraints: Create Table

- Unique

```
CREATE TABLE produto (
    nro_produto integer UNIQUE,
    nome text,
    preco numeric
);
CREATE TABLE produto (
    nro_produto integer,
    nome text,
    preco numeric,
    UNIQUE (nro_produto)
);
```

Constraints: Create Table

- Unique

```
CREATE TABLE produto (
    nro_produto integer CONSTRAINT ser_differente UNIQUE,
    nome text,
    preco numeric
);
```

Constraints: Create Table

- Primary Key

```
CREATE TABLE produto (
    nro_produto integer PRIMARY KEY,
    nome text,
    preco numeric
);
CREATE TABLE example (
    a integer,
    b integer,
    c integer,
    PRIMARY KEY (a, c)
);
```

Constraints: Create Table

- Foreign Key

```
CREATE TABLE produto (
    nro_produto integer PRIMARY KEY,
    nome text,
    preco numeric
);
CREATE TABLE pedido (
    id_pedido integer PRIMARY KEY,
    nro_produto integer REFERENCES produto (nro_produto),
    quantidade integer
);
```

Constraints: Create Table

- Foreign Key

```
CREATE TABLE pedido(
    id_pedido integer PRIMARY KEY,
    nro_produto integer REFERENCES produto,
    quantidade integer
);
```

na ausência de uma lista de colunas, a chave primária da tabela referenciada é usada como a coluna a ser referenciada

Constraints: Create Table

- Foreign Key

```
CREATE TABLE t1 (
  a integer PRIMARY KEY,
  b integer,
  c integer,
  FOREIGN KEY (b, c) REFERENCES other_table (c1, c2)
);
```

também é possível associar um nome a *foreign key*

Apagando linhas

- `DELETE`

  remoção de tuplas de uma relação

  ```
  DELETE
  FROM Aluno
  WHERE nota = 0
  ```

Destruindo/Alterando Tabelas

- `DROP TABLE`
  - Destruir uma tabela → excluir todas as linhas e remover as informaçõ̃es de definição da tabela
  - Ex: `DROP TABLE` Alunos `CASCADE`
    - Destrói a tabela Alunos e todas as visões ou restrições de integridade associadas também serão eliminadas

Destruindo/Alterando Tabelas

- `ALTER TABLE`
  - Modifica a estrutura de uma tabela
  - Ex: `ALTER TABLE` Alunos

    `ADD COLUMN` nome-familia CHAR(10)
    - A coluna nome-familia é adicionada à tabela Alunos e todas as linhas existentes são preenchidas com valores *null* para essa coluna

Destruindo/Alterando Tabelas

- `ALTER TABLE`
  - Ex: `ALTER TABLE` Alunos
    `DROP` media
    - elimina a coluna media existente
  - Ex: `ALTER TABLE` Alunos
    `ALTER` média `SET DEFAULT 0.0`
    - Altera o valor *default* da coluna média para 0.0

Destruindo/Alterando Tabelas

- `ALTER TABLE`
  - Ex: `ALTER TABLE` Alunos `ALTER [COLUMN]` média `TO` médiageral
    - Altera o nome da coluna média para médiageral

- `ALTER TABLE`
  - Ex: `ALTER TABLE` Alunos `ALTER [COLUMN]` média `TYPE FLOAT`
    - Modifica o tipo de dado da coluna média

Referências

- R. Elmasri e S. B. Navathe, Sistema de Banco de Dados, 6ª edição, Pearson, 2011.

- R. Ramakrishnan e J. Gehrke, *Database Management Systems,* 3a Edição, McGraw-Hill, 2003.