

Projeto de Programas

Paulo Torrens

paulotorrens@gnu.org

Departamento de Ciência da Computação
Centro de Ciências e Tecnologias
Universidade do Estado de Santa Catarina

2020/1

Modelo Entidade-Relacionamento

- Ao projetar um software, é necessária consideração sobre o armazenamento das informações necessárias para a execução
- Em muitos projetos, especialmente os de maior porte, as informações são salvas através de um software externo, o **banco de dados**
- Embora existam outras variações, bancos de dados costumam funcionar através de **relacionamento** entre **entidades**
 - Entidades são representadas através de **tabelas**
 - Bancos que não usam o modelo relacional são tradicionalmente chamados de bancos NoSQL
 - A linguagem **SQL** (*Structured Query Language*) é uma linguagem padronizada para manipulação de dados relacionais

Modelo Entidade-Relacionamento

- Ao projetar um software, é necessária consideração sobre o armazenamento das informações necessárias para a execução
- Em muitos projetos, especialmente os de maior porte, as informações são salvas através de um software externo, o **banco de dados**
- Embora existam outras variações, bancos de dados costumam funcionar através de **relacionamento** entre **entidades**
 - Entidades são representadas através de **tabelas**
 - Bancos que não usam o modelo relacional são tradicionalmente chamados de bancos NoSQL
 - A linguagem **SQL** (*Structured Query Language*) é uma linguagem padronizada para manipulação de dados relacionais

Modelo Entidade-Relacionamento

- Ao projetar um software, é necessária consideração sobre o armazenamento das informações necessárias para a execução
- Em muitos projetos, especialmente os de maior porte, as informações são salvas através de um software externo, o **banco de dados**
- Embora existam outras variações, bancos de dados costumam funcionar através de **relacionamento** entre **entidades**
 - Entidades são representadas através de **tabelas**
 - Bancos que não usam o modelo relacional são tradicionalmente chamados de bancos NoSQL
 - A linguagem **SQL** (*Structured Query Language*) é uma linguagem padronizada para manipulação de dados relacionais

Modelo Entidade-Relacionamento

- Ao projetar um software, é necessária consideração sobre o armazenamento das informações necessárias para a execução
- Em muitos projetos, especialmente os de maior porte, as informações são salvas através de um software externo, o **banco de dados**
- Embora existam outras variações, bancos de dados costumam funcionar através de **relacionamento** entre **entidades**
 - Entidades são representadas através de **tabelas**
 - Bancos que não usam o modelo relacional são tradicionalmente chamados de bancos NoSQL
 - A linguagem **SQL** (*Structured Query Language*) é uma linguagem padronizada para manipulação de dados relacionais

Modelo Entidade-Relacionamento

- Ao projetar um software, é necessária consideração sobre o armazenamento das informações necessárias para a execução
- Em muitos projetos, especialmente os de maior porte, as informações são salvas através de um software externo, o **banco de dados**
- Embora existam outras variações, bancos de dados costumam funcionar através de **relacionamento** entre **entidades**
 - Entidades são representadas através de **tabelas**
 - Bancos que não usam o modelo relacional são tradicionalmente chamados de bancos NoSQL
 - A linguagem **SQL** (*Structured Query Language*) é uma linguagem padronizada para manipulação de dados relacionais

Modelo Entidade-Relacionamento

- Ao projetar um software, é necessária consideração sobre o armazenamento das informações necessárias para a execução
- Em muitos projetos, especialmente os de maior porte, as informações são salvas através de um software externo, o **banco de dados**
- Embora existam outras variações, bancos de dados costumam funcionar através de **relacionamento** entre **entidades**
 - Entidades são representadas através de **tabelas**
 - Bancos que não usam o modelo relacional são tradicionalmente chamados de bancos NoSQL
 - A linguagem **SQL** (*Structured Query Language*) é uma linguagem padronizada para manipulação de dados relacionais

Modelo Entidade-Relacionamento

- Uma forma de representar modelos relacionais é através do **diagrama ER**
- Conceitualmente, há três tipos de informação necessárias para a representação de um modelo:
 - **Entidades**, representando objetos de interesse para o negócio, como Cliente, Livro, Empréstimo, etc
 - **Atributos**, que são campos de uma entidade, salvando uma informação relevante a ela; por exemplo, um Cliente deve ter um nome, e-mail, endereço, etc
 - **Relacionamentos**, que são formas de se “navegar” dentro dos dados, e representam relações entre as entidades existentes; um relacionamento geralmente é representado por um verbo: um Cliente efetua um Empréstimo, um Empréstimo contém um Livro, etc

Modelo Entidade-Relacionamento

- Uma forma de representar modelos relacionais é através do **diagrama ER**
- Conceitualmente, há três tipos de informação necessárias para a representação de um modelo:
 - **Entidades**, representando objetos de interesse para o negócio, como Cliente, Livro, Empréstimo, etc
 - **Atributos**, que são campos de uma entidade, salvando uma informação relevante a ela; por exemplo, um Cliente deve ter um nome, e-mail, endereço, etc
 - **Relacionamentos**, que são formas de se “navegar” dentro dos dados, e representam relações entre as entidades existentes; um relacionamento geralmente é representado por um verbo: um Cliente efetua um Empréstimo, um Empréstimo contém um Livro, etc

Modelo Entidade-Relacionamento

- Uma forma de representar modelos relacionais é através do **diagrama ER**
- Conceitualmente, há três tipos de informação necessárias para a representação de um modelo:
 - **Entidades**, representando objetos de interesse para o negócio, como Cliente, Livro, Empréstimo, etc
 - **Atributos**, que são campos de uma entidade, salvando uma informação relevante a ela; por exemplo, um Cliente deve ter um nome, e-mail, endereço, etc
 - **Relacionamentos**, que são formas de se “navegar” dentro dos dados, e representam relações entre as entidades existentes; um relacionamento geralmente é representado por um verbo: um Cliente efetua um Empréstimo, um Empréstimo contém um Livro, etc

Modelo Entidade-Relacionamento

- Uma forma de representar modelos relacionais é através do **diagrama ER**
- Conceitualmente, há três tipos de informação necessárias para a representação de um modelo:
 - **Entidades**, representando objetos de interesse para o negócio, como Cliente, Livro, Empréstimo, etc
 - **Atributos**, que são campos de uma entidade, salvando uma informação relevante a ela; por exemplo, um Cliente deve ter um nome, e-mail, endereço, etc
 - **Relacionamentos**, que são formas de se “navegar” dentro dos dados, e representam relações entre as entidades existentes; um relacionamento geralmente é representado por um verbo: um Cliente efetua um Empréstimo, um Empréstimo contém um Livro, etc

Modelo Entidade-Relacionamento

- Uma forma de representar modelos relacionais é através do **diagrama ER**
- Conceitualmente, há três tipos de informação necessárias para a representação de um modelo:
 - **Entidades**, representando objetos de interesse para o negócio, como Cliente, Livro, Empréstimo, etc
 - **Atributos**, que são campos de uma entidade, salvando uma informação relevante a ela; por exemplo, um Cliente deve ter um nome, e-mail, endereço, etc
 - **Relacionamentos**, que são formas de se “navegar” dentro dos dados, e representam relações entre as entidades existentes; um relacionamento geralmente é representado por um verbo: um Cliente efetua um Empréstimo, um Empréstimo contém um Livro, etc

Modelo Entidade-Relacionamento

- O intuito do diagrama ER é representar, **com níveis variados de precisão**, as necessidades das informações de um sistema
 - Muitas vezes, cabe ao programador ou DBA (*Database Administrator*) implementar o banco de dados baseando-se no diagrama
- Diagramas ER não apresentam um padrão efetivo, porém possuem algumas convenções que facilitam o entendimento
 - Entidades são representadas através de retângulos
 - Atributos são ou representados através de círculos conectados à entidade, ou apresentados dentro dela em forma similar a um diagrama UML
 - Relacionamentos são representados através de losangos conectados às entidades que relacionam

Modelo Entidade-Relacionamento

- O intuito do diagrama ER é representar, **com níveis variados de precisão**, as necessidades das informações de um sistema
 - Muitas vezes, cabe ao programador ou DBA (*Database Administrator*) implementar o banco de dados baseando-se no diagrama
- Diagramas ER não apresentam um padrão efetivo, porém possuem algumas convenções que facilitam o entendimento
 - Entidades são representadas através de retângulos
 - Atributos são ou representados através de círculos conectados à entidade, ou apresentados dentro dela em forma similar a um diagrama UML
 - Relacionamentos são representados através de losangos conectados às entidades que relacionam

Modelo Entidade-Relacionamento

- O intuito do diagrama ER é representar, **com níveis variados de precisão**, as necessidades das informações de um sistema
 - Muitas vezes, cabe ao programador ou DBA (*Database Administrator*) implementar o banco de dados baseando-se no diagrama
- Diagramas ER não apresentam um padrão efetivo, porém possuem algumas convenções que facilitam o entendimento
 - Entidades são representadas através de retângulos
 - Atributos são ou representados através de círculos conectados à entidade, ou apresentados dentro dela em forma similar a um diagrama UML
 - Relacionamentos são representados através de losangos conectados às entidades que relacionam

Modelo Entidade-Relacionamento

- O intuito do diagrama ER é representar, **com níveis variados de precisão**, as necessidades das informações de um sistema
 - Muitas vezes, cabe ao programador ou DBA (*Database Administrator*) implementar o banco de dados baseando-se no diagrama
- Diagramas ER não apresentam um padrão efetivo, porém possuem algumas convenções que facilitam o entendimento
 - Entidades são representadas através de retângulos
 - Atributos são ou representados através de círculos conectados à entidade, ou apresentados dentro dela em forma similar a um diagrama UML
 - Relacionamentos são representados através de losangos conectados às entidades que relacionam

Modelo Entidade-Relacionamento

- O intuito do diagrama ER é representar, **com níveis variados de precisão**, as necessidades das informações de um sistema
 - Muitas vezes, cabe ao programador ou DBA (*Database Administrator*) implementar o banco de dados baseando-se no diagrama
- Diagramas ER não apresentam um padrão efetivo, porém possuem algumas convenções que facilitam o entendimento
 - Entidades são representadas através de retângulos
 - Atributos são ou representados através de círculos conectados à entidade, ou apresentados dentro dela em forma similar a um diagrama UML
 - Relacionamentos são representados através de losangos conectados às entidades que relacionam

Modelo Entidade-Relacionamento

- O intuito do diagrama ER é representar, **com níveis variados de precisão**, as necessidades das informações de um sistema
 - Muitas vezes, cabe ao programador ou DBA (*Database Administrator*) implementar o banco de dados baseando-se no diagrama
- Diagramas ER não apresentam um padrão efetivo, porém possuem algumas convenções que facilitam o entendimento
 - Entidades são representadas através de retângulos
 - Atributos são ou representados através de círculos conectados à entidade, ou apresentados dentro dela em forma similar a um diagrama UML
 - Relacionamentos são representados através de losangos conectados às entidades que relacionam

Modelo Entidade-Relacionamento

- Alguns atributos especiais podem ser classificados como **chaves primárias**, geralmente sublinhados no diagrama
 - A ideia de uma chave primária é **identificar** um elemento daquela entidade, sendo um valor único; por exemplo, um Cliente pode ser identificado por um UUID ou por seu CPF
- Relacionamentos possuem uma **cardinalidade**
 - Ao se criar uma relação entre entidades A e B, é necessário informar quantos A se relacionam a B, e quantos B se relacionam a A
 - Cada lado de uma relação possui uma cardinalidade, podendo ser **opcional** (zero ou um), **única** (apenas um), **múltipla** (zero ou mais), etc
 - Um para muitos, muitos para muitos, etc
 - Por exemplo, tendo **Cliente 1** \longleftrightarrow **N Empréstimo** como uma relação, um cliente pode efetuar múltiplos empréstimos, mas cada empréstimo registrado é atrelado a apenas um cliente

Modelo Entidade-Relacionamento

- Alguns atributos especiais podem ser classificados como **chaves primárias**, geralmente sublinhados no diagrama
 - A ideia de uma chave primária é **identificar** um elemento daquela entidade, sendo um valor único; por exemplo, um Cliente pode ser identificado por um UUID ou por seu CPF
- Relacionamentos possuem uma **cardinalidade**
 - Ao se criar uma relação entre entidades A e B, é necessário informar quantos A se relacionam a B, e quantos B se relacionam a A
 - Cada lado de uma relação possui uma cardinalidade, podendo ser **opcional** (zero ou um), **única** (apenas um), **múltipla** (zero ou mais), etc
 - Um para muitos, muitos para muitos, etc
 - Por exemplo, tendo **Cliente 1** \longleftrightarrow **N Empréstimo** como uma relação, um cliente pode efetuar múltiplos empréstimos, mas cada empréstimo registrado é atrelado a apenas um cliente

Modelo Entidade-Relacionamento

- Alguns atributos especiais podem ser classificados como **chaves primárias**, geralmente sublinhados no diagrama
 - A ideia de uma chave primária é **identificar** um elemento daquela entidade, sendo um valor único; por exemplo, um Cliente pode ser identificado por um UUID ou por seu CPF
- Relacionamentos possuem uma **cardinalidade**
 - Ao se criar uma relação entre entidades A e B, é necessário informar quantos A se relacionam a B, e quantos B se relacionam a A
 - Cada lado de uma relação possui uma cardinalidade, podendo ser **opcional** (zero ou um), **única** (apenas um), **múltipla** (zero ou mais), etc
 - Um para muitos, muitos para muitos, etc
 - Por exemplo, tendo **Cliente 1** \longleftrightarrow **N Empréstimo** como uma relação, um cliente pode efetuar múltiplos empréstimos, mas cada empréstimo registrado é atrelado a apenas um cliente

Modelo Entidade-Relacionamento

- Alguns atributos especiais podem ser classificados como **chaves primárias**, geralmente sublinhados no diagrama
 - A ideia de uma chave primária é **identificar** um elemento daquela entidade, sendo um valor único; por exemplo, um Cliente pode ser identificado por um UUID ou por seu CPF
- Relacionamentos possuem uma **cardinalidade**
 - Ao se criar uma relação entre entidades A e B, é necessário informar quantos A se relacionam a B, e quantos B se relacionam a A
 - Cada lado de uma relação possui uma cardinalidade, podendo ser **opcional** (zero ou um), **única** (apenas um), **múltipla** (zero ou mais), etc
 - Um para muitos, muitos para muitos, etc
 - Por exemplo, tendo **Cliente 1** \longleftrightarrow **N Empréstimo** como uma relação, um cliente pode efetuar múltiplos empréstimos, mas cada empréstimo registrado é atrelado a apenas um cliente

Modelo Entidade-Relacionamento

- Alguns atributos especiais podem ser classificados como **chaves primárias**, geralmente sublinhados no diagrama
 - A ideia de uma chave primária é **identificar** um elemento daquela entidade, sendo um valor único; por exemplo, um Cliente pode ser identificado por um UUID ou por seu CPF
- Relacionamentos possuem uma **cardinalidade**
 - Ao se criar uma relação entre entidades A e B, é necessário informar quantos A se relacionam a B, e quantos B se relacionam a A
 - Cada lado de uma relação possui uma cardinalidade, podendo ser **opcional** (zero ou um), **única** (apenas um), **múltipla** (zero ou mais), etc
 - Um para muitos, muitos para muitos, etc
 - Por exemplo, tendo **Cliente 1** \longleftrightarrow **N Empréstimo** como uma relação, um cliente pode efetuar múltiplos empréstimos, mas cada empréstimo registrado é atrelado a apenas um cliente

Modelo Entidade-Relacionamento

- Alguns atributos especiais podem ser classificados como **chaves primárias**, geralmente sublinhados no diagrama
 - A ideia de uma chave primária é **identificar** um elemento daquela entidade, sendo um valor único; por exemplo, um Cliente pode ser identificado por um UUID ou por seu CPF
- Relacionamentos possuem uma **cardinalidade**
 - Ao se criar uma relação entre entidades A e B, é necessário informar quantos A se relacionam a B, e quantos B se relacionam a A
 - Cada lado de uma relação possui uma cardinalidade, podendo ser **opcional** (zero ou um), **única** (apenas um), **múltipla** (zero ou mais), etc
 - Um para muitos, muitos para muitos, etc
 - Por exemplo, tendo **Cliente 1** \longleftrightarrow **N Empréstimo** como uma relação, um cliente pode efetuar múltiplos empréstimos, mas cada empréstimo registrado é atrelado a apenas um cliente

Modelo Entidade-Relacionamento

- Alguns atributos especiais podem ser classificados como **chaves primárias**, geralmente sublinhados no diagrama
 - A ideia de uma chave primária é **identificar** um elemento daquela entidade, sendo um valor único; por exemplo, um Cliente pode ser identificado por um UUID ou por seu CPF
- Relacionamentos possuem uma **cardinalidade**
 - Ao se criar uma relação entre entidades A e B, é necessário informar quantos A se relacionam a B, e quantos B se relacionam a A
 - Cada lado de uma relação possui uma cardinalidade, podendo ser **opcional** (zero ou um), **única** (apenas um), **múltipla** (zero ou mais), etc
 - Um para muitos, muitos para muitos, etc
 - Por exemplo, tendo **Cliente 1** \longleftrightarrow **N Empréstimo** como uma relação, um cliente pode efetuar múltiplos empréstimos, mas cada empréstimo registrado é atrelado a apenas um cliente

Modelo Entidade-Relacionamento (exemplo feito em sala)

