Tema: Gestão de Piscinas

Grupo 505:
André Baptista, up201505375
José Cunha, up201504880
Nelson Almeida, up201505394

Março de 2017

Descrição do projeto

Existe uma empresa que gere várias piscinas.

Cada piscina tem um nome, ID, morada, capacidade máxima de clientes e um stock de itens. Estes possuem um nome, um custo e um número de série único. São distribuídos por um fornecedor que define o custo dos mesmos. Os itens podem servir uma de duas funções: ser vendidos na loja da piscina ou utilizados durante as aulas.

A loja tem uma hora de abertura, de fecho e vários artigos para venda.

Com a piscina podem interagir várias pessoas, tendo cada uma um nome e um ID. Uma pessoa pode ser considerada cliente ou funcionária da piscina, sendo que os funcionários ou são professores, ou são empregados com uma determinada função. Uma pessoa pode ser cliente de várias piscinas contudo um funcionário trabalha apenas numa. Cada cliente tem associado uma mensalidade enquanto que os funcionários recebem um salário.

As ocorrências têm hora de início e fim, dia da semana em que se realizam e um ID que as identifica. Estas podem ser livres ou aulas, sendo neste caso monitorizadas por um professor. As aulas abordam uma determinada modalidade. As modalidades têm um nome e nível específicos. Para que um cliente esteja apto a frequentar uma aula deverá primeiro passar numa prova da modalidade do nível anterior.

Principais conceitos

Classes

Piscina

A classe Piscina é a classe principal desta base de dados, a classe sobre a qual tudo se desenrola.

Da Piscina é necessário registar o nome, ID, morada e capacidade máxima de clientes.

A sua chave primária é o ID.

Pessoa

É uma super-classe.

Cada Pessoa tem um nome e um ID.

A sua chave primária é o ID.

Cliente

É uma classe derivada da classe Pessoa.

Cada Cliente tem uma mensalidade.

Funcionário

É uma super-classe e simultaneamente uma classe derivada da classe Pessoa.

Cada Funcionário tem um salário.

Professor

É uma classe derivada da classe Funcionário.

Empregado

É uma classe derivada da classe Funcionário.

Cada Empregado tem uma função a desempenhar na piscina.

Função

Cada função tem um nome que será a sua chave primária.

Item

É uma super-classe.

Cada Item tem um nome, custo e número de série.

A sua chave primária é o número de série.

ItemUtilização

É uma classe derivada da classe Item.

Cada item de utilização tem associado uma condição que indica o seu o nível de desgaste.

ItemVenda

É uma classe derivada da classe Item.

Cada item de venda possui um preço.

Fornecedor

Cada Fornecedor tem um contacto que será a sua chave primária.

Loja

A Loja possui um nome, uma hora de abertura e de fecho.

A sua chave primária é o nome.

Ocorrência

É uma super-classe.

Cada Ocorrência tem um ID, uma hora de início e fim, duração e dia da semana.

A sua chave primária é o ID.

Aula

É uma classe derivada da classe Ocorrência.

Modalidade

Cada Modalidade possui um nome e um nível.

A sua chave primária é composta pelo nome e nível.

Prova

É uma classe de associação entre Cliente e Modalidade.

Cada Prova possui uma data e um resultado que indica se o Cliente foi aprovado a esta Modalidade.

Relações

Piscina-Cliente

Uma piscina possui vários clientes.

Cada cliente frequenta pelo menos uma piscina.

Piscina-Funcionário

Uma piscina possui vários funcionários.

Cada funcionário trabalha apenas numa piscina.

Piscina-Item

Uma piscina possui vários itens.

Cada item pertence apenas a uma piscina.

Piscina-Loja

É uma relação de um para um.

Cliente-Ocorrência

Um cliente pode participar em várias ocorrências.

Uma ocorrência pode englobar vários clientes.

Cliente-Modalidade

Um cliente pode ir a provas de várias modalidades.

Uma modalidade pode ser praticada por vários clientes.

Aula-Professor

Uma aula é dada por um só professor.

Um professor pode lecionar várias aulas.

Aula-Modalidade

Cada aula aborda uma modalidade.

Uma modalidade pode ser lecionada em várias aulas.

Loja-Empregado

Uma loja tem vários empregados.

Um empregado trabalha apenas numa loja.

Loja-ItemVenda

É uma relação de composição, sendo que o item de venda só existe se estiver no stock da loja.

Uma loja tem vários itens para venda.

Ocorrência-ItemUtilização

Uma ocorrência possui vários itens de utilização.

O mesmo item de utilização pode ser usado em várias ocorrências.

Item-Fornecedor

Um item tem um fornecedor.

Cada fornecedor possui vários itens.

Empregado-Função

Cada empregado tem uma única função a desempenhar na piscina.

Uma função é desempenhada por pelo menos um empregado.

Modelo Relacional e Dependências Funcionais (FD)

Piscina (ID, nome, morada, capacidade)

- {ID} -> {nome, morada, capacidade}
- {morada} -> {ID, nome, capacidade}
- ID: não pode haver duas piscinas com o mesmo ID (PRIMARY KEY);
- morada: duas piscinas não têm a mesma morada (UNIQUE);
- capacidade: deve ser um inteiro positivo (CHECK);

Pessoa (ID, nome)

- {ID} -> {nome}
- ID: não pode haver duas pessoas com o mesmo ID (PRIMARY KEY);

Cliente (ID -> Pessoa, nome, mensalidade)

- {ID} -> {nome, mensalidade}
- ID: não pode haver dois clientes com o mesmo ID (PRIMARY KEY, FOREIGN KEY);
- mensalidade: deve ser um inteiro positivo (CHECK);

Funcionário (ID -> Pessoa, nome, salário)

- {ID} -> {nome, salário}
- ID: não pode haver dois funcionários com o mesmo ID (PRIMARY KEY, FOREIGN KEY);
- salário: deve ser um inteiro positivo (CHECK);

Professor (ID -> Pessoa, nome, salário)

- {ID} -> {nome, salário}
- ID: não pode haver dois professores com o mesmo ID (PRIMARY KEY, FOREIGN KEY);
- salário: deve ser um inteiro positivo (CHECK);

Empregado (ID -> Pessoa, nome, salário)

- {ID} -> {nome, salário}
- ID: não pode haver dois empregados com o mesmo ID (FOREIGN KEY, UNIQUE);
- salário: deve ser um inteiro positivo (CHECK);

Função (nome)

- Apenas existe FD trivial - Viola a BCNF e 3NF
- nome: (PRIMARY KEY);

Item (nº série, nome, custo)

- {nºsérie} -> {nome, custo}
- nºsérie: não existem dois itens com o mesmo nºsérie (PRIMARY KEY);
- custo: deve ser um inteiro positivo (CHECK);

ItemUtilização (nºsérie -> Item, nome, custo, condição)

- {nºsérie} -> {nome, custo, condição}
- nºsérie: não existem dois itens de utilização com o mesmo nºsérie (PRIMARY KEY, FOREIGN KEY);
- custo: deve ser um inteiro positivo (CHECK);
- condição: um inteiro no intervalo [1,5] (CHECK);

ItemVenda (nºsérie -> Item, nome, custo, preço)

- {nºsérie} -> {nome, custo, preço}
- {nome, custo} -> {preço} - Viola a BCNF e 3NF
- nºsérie: não existem dois itens de venda com o mesmo nºsérie (PRIMARY KEY, FOREIGN KEY);
- custo: deve ser um inteiro positivo (CHECK);
- preço: deve ser um inteiro positivo (CHECK);

Fornecedor (contacto)

- Apenas existe FD trivial - Viola a BCNF e 3NF
- contacto: um inteiro com 9 dígitos (PRIMARY KEY, CHECK);

Loja (nome, hora de abertura, hora de fecho, piscina -> Piscina)

- {nome} -> {hora de abertura, hora de fecho, piscina}
- {piscina} -> {nome, hora de abertura, hora de fecho}
- nome: (PRIMARY KEY);
- piscina: não pode haver duas piscinas com o mesmo ID (FOREIGN KEY, UNIQUE);

Ocorrência (ID, hora de início, hora de fim, duração, dia da semana)

- {ID} -> {hora de início, hora de fim, duração, dia da semana}
- {hora de início, hora de fim} -> {duração} - Viola a BCNF e 3NF
- ID: não existem duas ocorrências com o mesmo ID (PRIMARY KEY);

Aula (ID -> Ocorrência, hora de início, hora de fim, duração, dia da semana)

- {ID} -> {hora de início, hora de fim, duração, dia da semana}
- {hora de início, hora de fim} -> {duração} - Viola a BCNF e 3NF
- ID: não existem duas aulas com o mesmo ID (PRIMARY KEY, FOREIGN KEY);

Modalidade (nome, nível)

- Apenas existe FD trivial - Viola a BCNF e 3NF
- nome: (PRIMARY KEY);
- nível: deve ser um inteiro positivo (PRIMARY KEY, CHECK);

Prova (IDcliente -> Cliente, [nomeMod, nívelMod] -> Modalidade, data, resultado)

- {IDcliente, nomeMod, nívelMod} -> {data, resultado}
- {IDcliente, nomeMod, data} -> {nívelMod, resultado}
- IDcliente: (PRIMARY KEY, FOREIGN KEY);
- nomeMod: (PRIMARY KEY, FOREIGN KEY);
- nívelMod: (PRIMARY KEY, FOREIGN KEY);

Atende (piscina -> Piscina, cliente -> Cliente)

- Apenas existe FD trivial - Viola a BCNF e 3NF
- piscina: (PRIMARY KEY, FOREIGN KEY);
- cliente: (PRIMARY KEY, FOREIGN KEY);

Trabalha (funcionário -> Funcionário, piscina -> Piscina)

- {funcionário} -> {piscina}
- funcionário: (PRIMARY KEY, FOREIGN KEY);
- piscina: (FOREIGN KEY);

Possui (item -> Item, piscina -> Piscina)

- {item} -> {piscina}
- item: (PRIMARY KEY, FOREIGN KEY);
- piscina: (FOREIGN KEY);

Participa (cliente -> Cliente, ocorrência -> Ocorrência)

- Apenas existe FD trivial - Viola a BCNF e 3NF
- cliente: (PRIMARY KEY, FOREIGN KEY);
- ocorrência: (PRIMARY KEY, FOREIGN KEY);

Leciona (aula -> Aula, professor -> Professor)

- {aula} -> {professor}
- aula: (PRIMARY KEY, FOREIGN KEY);
- professor: (FOREIGN KEY);

AulaModalidade (aula -> Aula, [nomeMod, nívelMod] -> Modalidade)

- {aula} -> {nomeMod, nívelMod}
- aula: (PRIMARY KEY, FOREIGN KEY);
- nomeMod: (FOREIGN KEY);
- nívelMod: (FOREIGN KEY);

Trabalha (<u>empregado</u> -> Empregado, loja -> Loja)

- {empregado} -> {loja}
- empregado: (PRIMARY KEY, FOREIGN KEY);
- loja: (FOREIGN KEY);

Vendido (<u>item</u> -> ItemVenda, loja -> Loja)

- {item} -> {loja}
- item: (PRIMARY KEY, FOREIGN KEY);
- loja: (FOREIGN KEY);

Utilizado (<u>item</u> -> ItemUtilização, <u>ocorrência</u> -> Ocorrência)

- Apenas existe FD trivial - Viola a BCNF e 3NF
- item: (PRIMARY KEY, FOREIGN KEY);
- ocorrência: (PRIMARY KEY, FOREIGN KEY);

Fornecido (<u>item</u> -> Item, fornecedor -> Fornecedor)

- {item} -> {fornecedor}
- item: (PRIMARY KEY, FOREIGN KEY);
- fornecedor: (FOREIGN KEY);

Desempenha (<u>empregado</u> -> Empregado, função -> Função)

- {empregado} -> {função}
- empregado: (PRIMARY KEY, FOREIGN KEY);
- função: (FOREIGN KEY);

Análise à Boyce-Codd Normal Form e 3rd Normal Form

De entre as dependências funcionais acima enumeradas a maioria não viola a BCNF dado que o lado esquerdo de todas estas FD representa uma key. Se tal acontece significa que também não violam a 3NF.

No nosso caso, todas as FD que violam a BCNF também violam a 3NF pois: ou são triviais ou, simultaneamente, o lado esquerdo não é (super)key e os atributos do lado direito não são todos primos, como é o caso da FD não trivial:

{hora de início, hora de fim} -> {duração}

Apesar disto, seria possível haver FDs que respeitassem a 3NF não respeitando a BCNF caso todos os atributos do lado direito da FD fossem primos mas o lado o esquerdo não fosse uma (super)key.

Diagrama de Classes