



UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIAS DE COMPUTAÇÃO

Disciplina: SCC0240 - Base de Dados

Prof^ª: Elaine Parros M. de Sousa

Projeto – Jogos Paralímpicos

O código será entregue no escaninho do aluno: Eduardo Kiyoshi Kamikabeya.

Aluno

Eduardo Kiyoshi Kamikabeya

Henrique Pasquini Santos

Nº USP

9778536

8532252

Data de entrega: 21/06/2016

Descrição do Problema

Uma empresa organizadora de eventos foi contratada para gerenciar a logística dos Jogos Paralímpicos Rio 2016. Sua função é garantir o alojamento de todas as delegações paralímpicas, de maneira que elas fiquem próximas aos locais de suas modalidades, além de fornecer um transporte acessível a cada atleta, de acordo com suas necessidades.

Os jogos serão realizados em diferentes regiões da cidade. Cada região possui seus próprios centros de prática (arenas, estádios, etc.) e pontos turísticos. Algumas atrações podem pertencer a mais de uma região, como por exemplo o Pão de Açúcar que esta entre Copacabana, Botafogo e Flamengo . Um centro de prática é identificado pelo seu nome e definido pelo seu endereço e a região a que pertence.

Os membros das delegações são definidos por: nome, país, idade, sexo e passaporte/CPF. Todos os membros das delegações receberão um cartão de identificação, com um número identificador único por questões de segurança, e para facilitar o acesso aos centros de prática e aos hotéis nos quais estão hospedados. Os cartões são diferenciados para cada função da delegação (atleta recebem azul, comissão técnica recebem verde e equipe de apoio amarelo).

Os atletas possuem equipamentos que são definidos pelo seu dono, tipo(ex. Cadeira de roda adaptada, perna mecânica, etc.) e a quantidade, caso sejam transportadas aparelhos substitutos . Além disso é necessário, para o planejamento de transporte, armazenar seu volume e a data que o equipamento deve ser entregue em um centro de prática.

Uma modalidade é definida por: nome e classe(ambos únicos para cada modalidade) e suas regras. Uma delegação possui múltiplas modalidades e é identificada pelo seu país de origem, e composta por atleta(s), comissão técnica e equipe de apoio. Cada delegação possui vários veículos de transporte adaptados para o(s) seus atleta(s). Um veículo é identificado pela sua placa e deve possuir um tipo de veículo que identifica se ele transporta pessoas ou equipamentos de atletas. Caso o veículo transporte pessoas, deseja-se verificar seu número de lugares e as adaptações que possui. Sobre um veículo que transporta carga, a organização do evento deseja armazenar a carga máxima que ele consegue transportar. A equipe



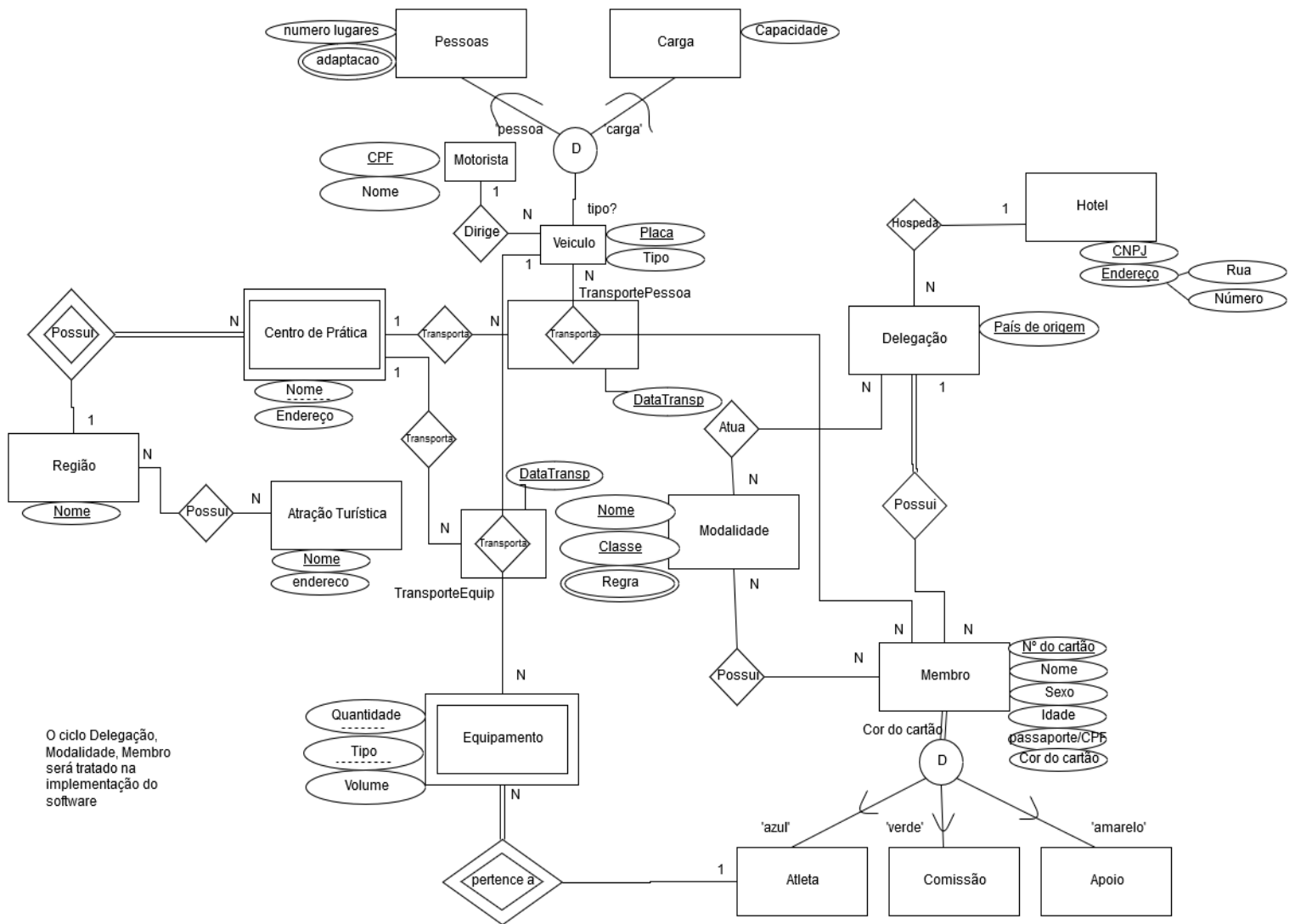
organizadora deseja pesquisar por essa informação, para o planejamento de transporte de equipamento de atletas. Cada veículo possui um motorista, sendo necessário armazenar suas informações: CPF, nome. O transporte de equipamentos e pessoas é realizado para os centros de prática e vice-versa, tanto para treinamentos, quanto para competições.

Os hotéis acomodam as delegações e sobre um hotel é definido unicamente no sistema por: endereço (rua e seu número) e CNPJ.

Para promover o turismo no país, as delegações também tem a possibilidade de pesquisar as atrações turísticas de uma região, assim deseja-se armazenar informações sobre os pontos turísticos de cada região. Uma atração turística é definida por: nome(único para cada atração) e endereço.

A comissão que organiza o transporte de jogos necessita pesquisar para cada veículo e data, quais membros vão ser transportados e o centro de prática destino. Também é necessário por questões de controle cadastrar as entregas de equipamentos a um centro de prática em determinada data. Só veículo de carga podem efetuar o transporte de equipamentos.

Modelo Entidade-Relacionamento



Discussão do Mapeamento MER-Relacional

Em algumas tabelas (CentrodePratica, Modalidade, Equipamento) foi optado criar “id’s” como chave primária, para facilitar as referências de chaves estrangeiras, que apontam apenas para uma chave simples ao invés de uma composta, diminuindo o número de colunas das tabelas.

Foi criada uma tabela Adaptação pois não se sabe o número de adaptações de um veículo. Caso fosse escolhido um número fixo de atributos e esse necessitasse ser ultrapassado, poderia ocorrer problemas no transporte de atletas. O mesmo caso acontece com as regras de cada modalidade, onde não se sabe quantas regras ao certo tem cada modalidade e sua quantidade pode variar muito.

As agregações são identificadas pelas chaves das entidades participantes mais a data de transporte, pois um equipamento pode ser transportado pelo mesmo veículo em datas diferentes, quando em dias de treinamento/Competição diferentes.

O grupo optou por colocar os dados de motorista na tabela Veiculo, ao invés de criar uma nova tabela, pois isso torna a busca mais rápida, levando em conta que uma busca por informações de um motorista ocorre com menos frequência.

Optamos por criar uma tabela Hotel, ao invés de colocar seus atributos em delegação, pois como um Hotel pode ocupar mais de uma delegação, ocorreria a duplicação de dados de um mesmo hotel, possivelmente levando a inconsistências.

A especialização Veículo, foram criadas as tabelas de Veiculo, VeiculoPessoas e VeiculoCarga pois a tabela pai tem relacionamentos com outras entidades e as tabelas filhas possuem atributos próprios. Assim, concatenar as tabelas filhas com a pai geraria muitos campos nulos. Além disso, criar somente as tabelas filhas, geraria uma especialização total que não existe no domínio do problema.

Em relação a especialização Membro, o grupo optou por criar somente a tabela pai, tratando em software a verificação do tipo de membro, de que somente um atleta pode possuir equipamentos e garantir a especialização total, verificando que somente os tipo permitidos (Atleta, Comissão, Apoio) possam ser inseridos. Essa opção foi escolhida pois

somente a tabela pai possui atributos o que levaria a criação de três tabelas com os mesmos tipo de dados.

Implementação do SGBD

A implementação do banco de dados e suas funções foi feita na linguagem SQL, e o sistema que opera as funcionalidades do sistema foi implementado em Java, na plataforma NetBeans.

Requisitos:

Java instalado: https://www.java.com/pt_BR/download/

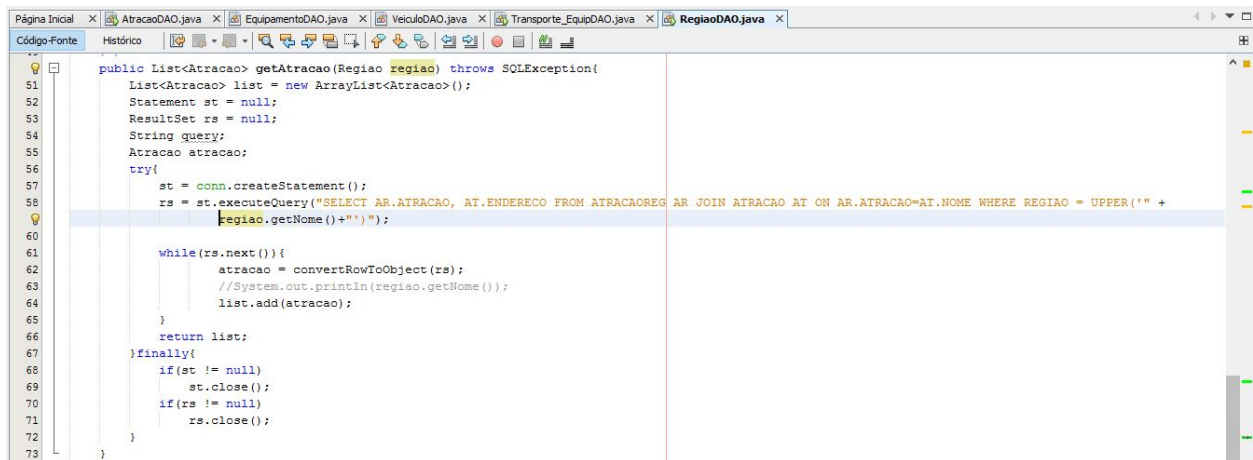
Java SE Development Kit 8 (JDK):

<http://www.oracle.com/technetwork/pt/java/javase/downloads/jdk8-downloads-2133151.html>

NetBeans instalado: <https://netbeans.org/downloads/>

Códigos

Classe: RegiaoDAO.java



```
1 public List<Atracao> getAtracao(Regiao regiao) throws SQLException{
2     List<Atracao> list = new ArrayList<Atracao>();
3     Statement st = null;
4     ResultSet rs = null;
5     String query;
6     Atracao atracao;
7     try{
8         st = conn.createStatement();
9         rs = st.executeQuery("SELECT AR.ATRACAO, AT.ENDERECO FROM ATRACAO REG AR JOIN ATRACAO AT ON AR.ATRACAO=AT.NOME WHERE REGIAO = UPPER('" +
10             regiao.getNome()+"')");
11     }
12     while(rs.next()){
13         atracao = convertRowToObject(rs);
14         //System.out.println(regiao.getNome());
15         list.add(atracao);
16     }
17     return list;
18 }finally{
19     if(st != null)
20         st.close();
21     if(rs != null)
22         rs.close();
23 }
```

A função getAtracao busca todas as atrações da região informada.

Classe: Transporte_PessoaDAO.java

```

Página Inicial x AtracaoDAO.java x EquipamentoDAO.java x VeiculoDAO.java x Transporte_EquipDAO.java x RegiaoDAO.java x Transporte_PessoaDAO.java x
Código-Fonte Histórico
public List<Transporte_Pessoa> getMembrosByVeiculo(Veiculo veiculo, String data) throws SQLException{
    List<Transporte_Pessoa> list = new ArrayList<Transporte_Pessoa>();
    Statement st = null;
    ResultSet rs = null;
    String query;
    Transporte_Pessoa tp;
    try{
        st = conn.createStatement();
        rs = st.executeQuery("SELECT CP.NOME, CP.ENDERECO, TP.VEICULO, M.NOME AS MEMBRO, TP.DATA\n" +
        "FROM CENTRO_DE_PRATICA CP JOIN TRANSPORTE_PESSOA TP\n" +
        "ON CP.ID = TP.CENTRO_DE_PRATICA JOIN\n" +
        "MEMBRO M ON M.N_CARTAO = TP.MEMBRO WHERE VEICULO = UPPER('" + veiculo.getPlaca() + "') AND TP.DATA = TO_DATE('" + data + "','dd/mm/yyyy')");
        while(rs.next()){
            tp = convertRowToObject(rs);
            //System.out.println(regiao.getNome());
            list.add(tp);
        }
        return list;
    }finally{
        if(st != null)
            st.close();
        if(rs != null)
            rs.close();
    }
}

```

A função `getMembrosByVeiculo` busca, a partir do `Veiculo` e de uma data, o nome e endereço de um centro de prática e o nome do membro a ser transportado.

Classe: VeiculoDAO.java

```

Código-Fonte Histórico
12 /**
13  *
14  * @author lenovo
15  */
16
17 public class VeiculoDAO {
18     private Connection conn;
19     public VeiculoDAO(){
20         conn = new Conexao().getConn();
21     }
22     public String getTipo(String placa) throws SQLException{
23         Statement st = null;
24         ResultSet rs = null;
25         String tipo = "";
26
27         try{
28             st = conn.createStatement();
29             rs = st.executeQuery("select tipo from veiculo where placa = upper('" + placa + "')");
30
31             while(rs.next()){
32                 tipo = rs.getString(1);
33             }
34             return tipo;
35         }finally{
36             if(st != null)
37                 st.close();
38             if(rs != null)
39                 rs.close();
40         }
41     }
42 }
43

```

A função `getTipo` retorna o tipo do veículo baseado em sua placa.

Classe: Transporte_EquipDAO.java

```

Página Inicial x AtracaoDAO.java x EquipamentoDAO.java x VeiculoDAO.java x Transporte_EquipDAO.java x RegiaoDAO.java x Transporte_PessoaDAO.java x
Código-Fonte Histórico
public float getCapacidadeAtual(Veiculo veiculo, String data) throws SQLException{
    float cargaAtual = 0;
    Statement st = null;
    ResultSet rs = null;
    String query;
    Regiao regiao;
    //query = "SELECT * FROM REGIAO:";
    try{
        st = conn.createStatement();
        rs = st.executeQuery("SELECT VC.PLACA, SUM(EQ.VOLUME) " +
"FROM TRANSPORTE_EQUIP TE JOIN EQUIPAMENTO EQ " +
"ON EQ.ID = TE.EQUIPAMENTO JOIN VEICULO_CARGA VC " +
"ON VC.PLACA = TE.VEICULO " +
"WHERE VC.PLACA = UPPER('"+ veiculo.getPlaca() + "') AND TE.DATA = " + data +
"GROUP BY VC.PLACA");
        while(rs.next()){
            cargaAtual = rs.getFloat(2);
        }
        return cargaAtual;
    }finally{
        if(st != null)
            st.close();
        if(rs != null)
            rs.close();
    }
}

```

A função `getCapacidadeAtual` busca o total de volume que o veículo planeja transportar em determinada data.

```

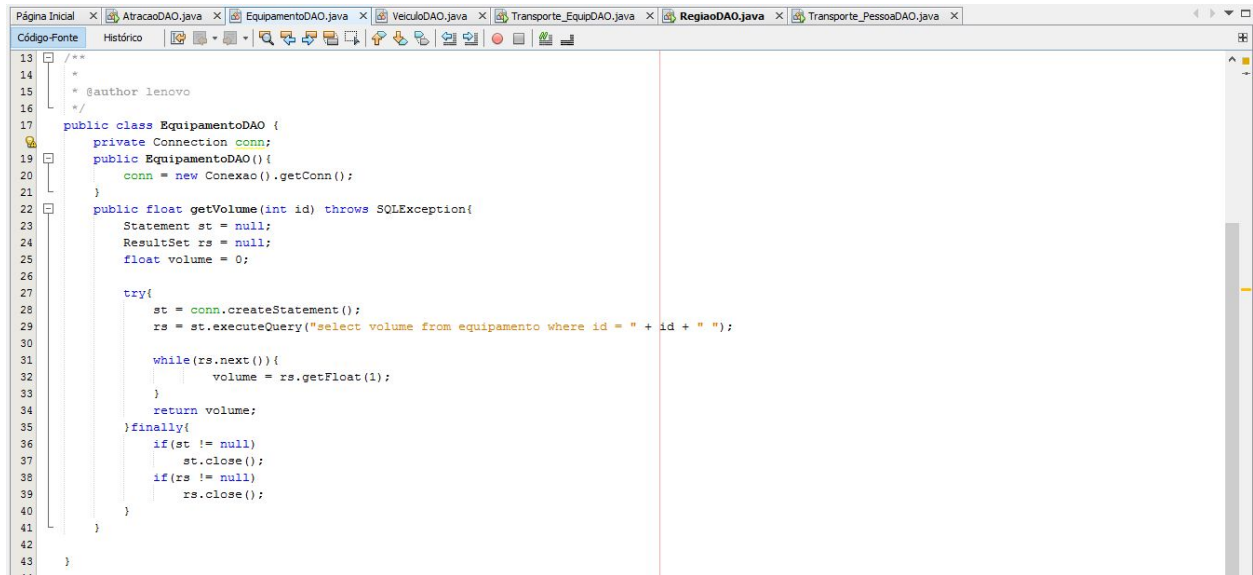
Página Inicial x AtracaoDAO.java x EquipamentoDAO.java x VeiculoDAO.java x Transporte_EquipDAO.java x RegiaoDAO.java x Transporte_PessoaDAO.java x
Código-Fonte Histórico
24 public boolean inserirEquipamento(Veiculo veiculo, Equipamento equip, CentroDePratica cp, String data) throws SQLException{
25     float cargaAtual = getCapacidadeAtual(veiculo, data);
26     float capacidade = 0;
27     Statement st = null;
28     ResultSet rs = null;
29     String query;
30     Regiao regiao;
31     EquipamentoDAO eDAO = new EquipamentoDAO();
32     equip.setVolume( eDAO.getVolume(equip.getId()));
33     try{
34         st = conn.createStatement();
35         rs = st.executeQuery("select capacidade from veiculo_carga where placa = upper('"+ veiculo.getPlaca()+"')");
36
37         while(rs.next()){
38             capacidade = rs.getFloat(1);
39         }
40         System.out.println(capacidade);
41         if(equip.getVolume() + cargaAtual < capacidade){
42             st.executeUpdate("INSERT INTO TRANSPORTE_EQUIP(CENTRO_DE_PRATICA, VEICULO, EQUIPAMENTO, DATA) VALUES('"
43                 + cp.getId() + "','"
44                 + veiculo.getPlaca() + "','"
45                 + equip.getId() + "','"
46                 + data + "')");
47             return true;
48         }else{
49             System.out.println("Capacidade ultrapassada");
50             return false;
51         }
52     }finally{
53         if(st != null)
54             st.close();
55         if(rs != null)
56             rs.close();
57     }
58 }
59

```

A função `inserirEquipDAO` insere um transporte de equipamento. Toma como entradas o veículo, equipamento, centro de prática destino e a data do transporte. Caso o volume do

equipamento mais o total que o veículo planeja transportar em determinada data ultrapasse sua capacidade, não permite a inserção. Caso contrário insere o transporte.

Classe: EquipamentoDAO.java

A screenshot of a code editor window showing the source code for the EquipamentoDAO.java class. The code is written in Java and includes a package declaration, imports, and a class definition. The class has a private Connection attribute and a public method getVolume that takes an integer id and returns a float volume. The method uses a try-catch block to handle SQL exceptions and a while loop to retrieve the volume from the database. The code is formatted with syntax highlighting and line numbers.

```
13  /**
14   *
15   * @author lenovo
16   */
17  public class EquipamentoDAO {
18      private Connection conn;
19      public EquipamentoDAO(){
20          conn = new Conexao().getConn();
21      }
22      public float getVolume(int id) throws SQLException{
23          Statement st = null;
24          ResultSet rs = null;
25          float volume = 0;
26
27          try{
28              st = conn.createStatement();
29              rs = st.executeQuery("select volume from equipamento where id = " + id + " ");
30
31              while(rs.next()){
32                  volume = rs.getFloat(1);
33              }
34              return volume;
35          }finally{
36              if(st != null)
37                  st.close();
38              if(rs != null)
39                  rs.close();
40          }
41      }
42  }
43  }
```

A função getVolume retorna o volume do equipamento baseado em seu id.

Conclusão

Em geral, tivemos um bom proveito com o projeto, pois ele abrangeu todo o conteúdo da disciplina, ajudando no nosso aprendizado. Uma das grandes dificuldades foi durante o começo do projeto, onde tivemos que criar uma descrição de problema do zero, e que apresentou uma quantidade insuficiente de informações para a especificação do projeto. Também tivemos dificuldades na implementação na parte 3 do projeto, o qual exigiu bastante tempo mas, com as devidas correções e aprimoração do nosso conhecimento ao longo da disciplina, conseguimos implementar um sistema razoável, possuindo um objetivo focado (logística dos jogos paralímpicos).