BIMU3064

Veritabanı Yönetim Sistemleri

ÖDEV 5

Abdulkadir Azmanoğlu
1306130092

ID	name	Gender	parentID
1	Ali	Erkek	Null
2	Ayşe	Kadın	Null
3	Zeynep	Kadın	1
4	Mustafa	Erkek	1
5	Cafer	Erkek	4
6	Mithat	Erkek	5
7	Nermin	Kadın	1
8	Elif	Kadın	5
9	Senem	Kadın	6

1- Java+PostgreSQL

```
package com.abdulkadirazm.iuce;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Scanner;

public class JavaPostgreSql {
    private static int id;
    private static Scanner input;
    private static int choice;
    private static String idStr = "ID";
    private static String nameStr = "NAME";
    private static String genderStr = "GENDER";
    private static String parentIdStr = "PARENT_ID";

    public static void main(String[] args) {
        choice = 0;
        input = new Scanner(System.in);

        menu();
        choice = input.nextInt();

        while (choice != 0) {

            switch (choice) {

                case 1:
                    System.out.println("ID :");
                    id = input.nextInt();
                    findGens(id);
                    break;
                case 2:
                    System.out.println("ID :");
                    id = input.nextInt();
                    findDescendants(id);
                    break;
                case 0:
                    choice = 0;
                    break;
                default:
                    System.out.println("Please,Enter 1 or 2");
                    break;

            }

            menu();
            choice = input.nextInt();
        }
    }

    private static void findGens(int use_parent) {

        try (Connection connection =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/odev", "postgres",
"postgres")) {
            System.out.println("Connected to PostgreSQL database!");
            Statement statement = connection.createStatement();
            String ROM = "\"" + id + "\"";
            ResultSet resultSet = statement.executeQuery(
                    "WITH RECURSIVE t\n" +
                    "    AS\n" +
                    "    (\n" +
                    "        SELECT * \n" +
                    "          FROM person p\n" +
                    "         WHERE p.id = "+ use_parent +"\n" +
                    "         UNION ALL\n" +
                    "        SELECT person.*\n" +
                    "          FROM person\n" +
                    "          JOIN t prev ON (person.id = prev.parentID )\n" +
                    "\t)\n" +
                    "SELECT * FROM t;\n");

            System.out.printf("%-2.2s %-8.10s  %-10.10s %-2.9s%n", idStr, nameStr,
genderStr, parentIdStr);
            System.out.println("----------------------------------");
            String s = String.valueOf(use_parent);
            while (resultSet.next()) {
                if (!resultSet.getString("id").equals(s))
                    System.out.printf("%-2.2s %-8.10s  %-10.10s %-2.5s%n",
resultSet.getString("id"), resultSet.getString("name"), resultSet.getString("gender"),
resultSet.getString("parentID"));

            }
        } catch (SQLException e) {
            System.out.println("Connection failure.");
            e.printStackTrace();
        }
    }

    private static void findDescendants(int start_id) {
        try (Connection connection =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/odev", "postgres",
"postgres")) {
            System.out.println("Connected to PostgreSQL database!");
            Statement statement = connection.createStatement();
            String ROM = "\"" + id + "\"";
            ResultSet resultSet = statement.executeQuery(
                    "WITH RECURSIVE t\n" +
                    "    AS\n" +
                    "    (\n" +
                    "        SELECT * \n" +
                    "          FROM person p\n" +
                    "         WHERE p.id = " + start_id + "\n" +
                    "         UNION ALL\n" +
                    "        SELECT next.*\n" +
                    "          FROM t prev\n" +
                    "          JOIN person next ON (next.parentID = prev.id)\n" +
                    "    )\n" +
                    "SELECT * FROM t;\n");
            System.out.printf("%-2.2s %-8.10s  %-10.10s %-2.9s%n", idStr, nameStr,
genderStr, parentIdStr);
            System.out.println("----------------------------------");
            String s = String.valueOf(start_id);
            while (resultSet.next()) {
                if (!resultSet.getString("id").equals(s))
                    System.out.printf("%-2.2s %-8.10s  %-10.10s %-2.9s%n",
resultSet.getString("id"), resultSet.getString("name"), resultSet.getString("gender"),
resultSet.getString("parentID"));
            }
        } catch (SQLException e) {
            System.out.println("Connection failure.");
            e.printStackTrace();
        }
    }



    private static void menu() {
        System.out.println("Menu");
        System.out.println("0: Çıkış");
        System.out.println("1: Soy ağacı sorgula");
        System.out.println("2: Soyundan gelenleri sorgula");
        System.out.println("Enter choice ?");
    }
}
```

```
Menu
0: Çıkış
1: Soy ağacı sorgula
2: Soyundan gelenleri sorgula
Enter choice ?
1
ID :
5
Connected to PostgreSQL database!
ID NAME      GENDER     PARENT_ID
----------------------------------
4  Mustafa   Erkek      1
1  Ali       Erkek      null

Menu
0: Çıkış
1: Soy ağacı sorgula
2: Soyundan gelenleri sorgula
Enter choice ?
2
ID :
5
Connected to PostgreSQL database!
ID NAME      GENDER     PARENT_ID
----------------------------------
6  Mithat    Erkek      5
8  Elif      Kadın      5
9  Senem     Kadın      6
```

2- Java+MongoDB

```
package com.abdulkadirazm.iuce;

import com.mongodb.*;
import com.mongodb.MongoClient;

import java.util.*;


public class JavaMongoDB {
    private static final String HOST = "localhost";
    private static final int PORT = 27017;
    private static final String DATABASE_NAME = "odev";
    private static final String COLLECTION_NAME = "person";
    private static MongoClient mongoClient;
    private static DB database;
    private static DBCollection person;
    private static PersonObj personObj;
    private static List<PersonObj> personList;
    private static List<PersonObj> parentList;
    private static List<PersonObj> children;
    private static List<PersonObj> DX;
    private static int id;
    private static Scanner input;
    private static int choice;
    private static String idStr;
    private static String nameStr;
    private static String genderStr;
    private static String parentIdStr;

    public JavaMongoDB() {
        personList = new ArrayList<>();
        parentList = new ArrayList<>();
        children = new ArrayList<>();
        idStr = "ID";
        nameStr = "NAME";
        genderStr = "GENDER";
        parentIdStr = "PARENT_ID";
        input = new Scanner(System.in);
        choice = 0;
    }

    public static void main(String[] args) {
        new JavaMongoDB();

        menu();
        choice = input.nextInt();

        while (choice != 0) {

            switch (choice) {

                case 1:
                    System.out.println("ID :");
                    id = input.nextInt();
                    String sId = String.valueOf(id);
                    findGens(sId);
                    System.out.printf("%-2.2s %-8.10s  %-10.10s %-2.9s%n", idStr,
nameStr, genderStr, parentIdStr);
                    System.out.println("----------------------------------");
                    for (PersonObj p : parentList) {
                        System.out.printf("%-2.2s %-8.10s  %-10.10s %-2.5s%n", p.id,
p.name, p.gender, p.parentId);
                    }
                    personList.clear();
                    parentList.clear();
                    break;
                case 2:
                    System.out.println("ID :");
                    id = input.nextInt();
                    String sId2 = String.valueOf(id);
                    findDescendants(sId2);
                    System.out.printf("%-2.2s %-8.10s  %-10.10s %-2.9s%n", idStr,
nameStr, genderStr, parentIdStr);
                    System.out.println("----------------------------------");
                    for (PersonObj child : children) {
                        System.out.printf("%-2.2s %-8.10s  %-10.10s %-2.5s%n",
child.id, child.name, child.gender, child.parentId);
                    }
                    personList.clear();
                    children.clear();
                    break;
                case 0:
                    choice = 0;
                    break;
                default:
                    System.out.println("Please,Enter 1 or 2");
                    break;

            }

            menu();
            choice = input.nextInt();
        }
    }

    private static void findDescendants(String id) {
        //personObj = getPersonById(id);

        getAllPersons();

        ArrayList<Pair> pairs = new ArrayList<Pair>();

        for (PersonObj person : personList) {
            pairs.add(new Pair(person.id.toString(), person.parentId.toString()));
        }

        Map<String, PersonObj> hm = new HashMap<>();

        for (Pair p : pairs) {

            //  ----- Child -----
            PersonObj mmdChild;
            if (hm.containsKey(p.getChildId())) {
                mmdChild = hm.get(p.getChildId());
            } else {
                mmdChild = new PersonObj();
                hm.put(p.getChildId(), mmdChild);
            }
            mmdChild.setId(p.getChildId());
            mmdChild.setParentId(p.getParentId());
            // no need to set ChildrenItems list because the constructor created a new
empty list


            // ------ Parent ----
            PersonObj mmdParent;
            if (hm.containsKey(p.getParentId())) {
                mmdParent = hm.get(p.getParentId());
            } else {
                mmdParent = new PersonObj();
                hm.put(p.getParentId(), mmdParent);
            }
```

```java
            mmdParent.setId(p.getParentId());
            mmdParent.setParentId("null");
            mmdParent.addChildrenItem(mmdChild);
        }

        // Get the root
        DX = new ArrayList<PersonObj>();
        for (PersonObj mmd : hm.values()) {
            if (mmd.getParentId().equals("null"))
                DX.add(mmd);
        }


        for (PersonObj mmd : DX) {
            if (mmd.id.equals(id)) {
                addChild(id, mmd.getChildrenItems());
            }
        }
    }

    private static void addChild(String parentID, List<PersonObj> mmd) {

        for (PersonObj md : mmd) {
            children.add(getPersonById(md.id));
            if (md.getChildrenItems().size() > 0) {
                addChild(md.id, md.getChildrenItems());
            }
        }
    }

    private static void addParent(PersonObj child, List<PersonObj> parentList) {
        PersonObj pPrev = new PersonObj();
        if (child.parentId.equals(""))
            return;
        personObj = getPersonById(child.parentId);
        for (PersonObj p : personList) {
            if (p.id.equals(personObj.id)) {
                parentList.add(personObj);
                pPrev = p;
            }
        }
        addParent(pPrev, parentList);
    }

    private static void findGens(String id) {
        personObj = getPersonById(id);

        getAllPersons();

        addParent(personObj, parentList);
    }

    public static void getAllPersons() {
        mongoClient = new MongoClient(HOST, PORT);
        database = mongoClient.getDB(DATABASE_NAME);
        person = database.getCollection(COLLECTION_NAME);
        DBCursor cursor = person.find();
        while (cursor.hasNext()) {
            DBObject next = cursor.next();
            personList.add(convert(next));
        }
    }

    private static PersonObj getPersonById(String id) {
        mongoClient = new MongoClient(HOST, PORT);
        database = mongoClient.getDB(DATABASE_NAME);
        person = database.getCollection(COLLECTION_NAME);
        DBObject query = new BasicDBObject("id", id);
        DBCursor cursor = person.find(query);
        return convert(cursor.one());
    }

    private static List<PersonObj> getPersonsByParentId(String parentId) {
        List<PersonObj> list = new ArrayList<>();
        mongoClient = new MongoClient(HOST, PORT);
        database = mongoClient.getDB(DATABASE_NAME);
        person = database.getCollection(COLLECTION_NAME);
        DBObject query = new BasicDBObject("parentID", Integer.parseInt(parentId));
        DBCursor cursor = person.find(query);

        while (cursor.hasNext()) {
            DBObject next = cursor.next();
            list.add(convert(next));
        }

        return list;
    }

    public static PersonObj convert(DBObject query) {
        PersonObj personObj = new PersonObj();
        personObj.setId(query.get("id") == null ? "" : query.get("id").toString());
        personObj.setName(query.get("name").toString());
        personObj.setGender(query.get("gender").toString());
        personObj.setParentId(query.get("parentID") == null ? "" :
query.get("parentID").toString());

        return personObj;
    }

    private static void menu() {
        System.out.println("Menu");
        System.out.println("0: Çıkış");
        System.out.println("1: Soy ağacı sorgula");
        System.out.println("2: Soyundan gelenleri sorgula");
        System.out.println("Enter choice ?");
    }

}

class Pair {
    private String childId;
    private String parentId;

    public Pair(String childId, String parentId) {
        this.childId = childId;
        this.parentId = parentId;
    }

    public String getChildId() {
        return childId;
    }

    public void setChildId(String childId) {
        this.childId = childId;
    }

    public String getParentId() {
        return parentId;
    }

    public void setParentId(String parentId) {
        this.parentId = parentId;
    }
```

```java
}

class PersonObj {
    private static final String HOST = "localhost";
    private static final int PORT = 27017;
    private static final String DATABASE_NAME = "odev";
    private static final String COLLECTION_NAME = "person";
    private static MongoClient mongoClient;
    private static DB database;
    private static DBCollection person;
    private static List<PersonObj> personList;
    String id = "";
    String name = "";
    String gender = "";
    String parentId = "";
    private List<PersonObj> childrenItems;

    public PersonObj() {
        this.id = "";
        this.name = "";
        this.parentId = "";
        this.childrenItems = new ArrayList<PersonObj>();
    }

    public String getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public String getGender() {
        return gender;
    }

    public String getParentId() {
        return parentId;
    }

    public void setId(String id) {
        this.id = id;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }

    public void setParentId(String parentId) {
        this.parentId = parentId;
    }

    public List<PersonObj> getChildrenItems() {
        return childrenItems;
    }

    public void setChildrenItems(List<PersonObj> childrenItems) {
        this.childrenItems = childrenItems;
    }

    public void addChildrenItem(PersonObj childrenItem) {
        if (!this.childrenItems.contains(childrenItem))
            this.childrenItems.add(childrenItem);
    }

    @Override
    public String toString() {
        return "PersonObj [Id=" + id + ", name=" + name + ", parentId="
               + parentId + ", childrenItems=" + childrenItems + "]";
    }
}
```

```
Menu
0: Çıkış
1: Soy ağacı sorgula
2: Soyundan gelenleri sorgula
Enter choice ?
1
ID :
5

ID NAME      GENDER     PARENT_ID
----------------------------------
4  Mustafa   Erkek      1
1  Ali       Erkek      null

Menu
0: Çıkış
1: Soy ağacı sorgula
2: Soyundan gelenleri sorgula
Enter choice ?
2
ID :
5

ID NAME      GENDER     PARENT_ID
----------------------------------
6  Mithat    Erkek      5
8  Elif      Kadın      5
9  Senem     Kadın      6
```