Practice8

На странице представлена информация по практическому заданию 8.

> ⛔ При коммите действуют ограничения на заливаемые файлы: class, jar файлы из коммита исключить!
> В противном случае не сможете сделать коммит.

Задание

Дан код:

Click here to expand...

```
package ua.nure.your_last_name.Practice8;
import java.util.List;
import ua.nure.your_last_name.Practice8.db.DBManager;
import ua.nure.your_last_name.Practice8.db.entity.Group;
import ua.nure.your_last_name.Practice8.db.entity.User;
public class Demo {

 private static <T> void printList(List<T> list) {
  for (T element : list) {
   System.out.println(element);
  }
 }

 public static void main(String[] args) {
  // users  ==> [ivanov]; groups ==> [teamA]

  DBManager dbManager = DBManager.getInstance();

  // Part 1
  dbManager.insertUser(User.createUser("petrov"));
  dbManager.insertUser(User.createUser("obama"));
  printList(dbManager.findAllUsers());
  // users  ==> [ivanov, petrov, obama]

  System.out.println("============================");

  // Part 2
```

```
dbManager.insertGroup(Group.createGroup("teamB"));
dbManager.insertGroup(Group.createGroup("teamC"));
printList(dbManager.findAllGroups());
// groups ==> [teamA, teamB, teamC]

System.out.println("============================");

// Part 3
User userPetrov = dbManager.getUser("petrov");
User userIvanov = dbManager.getUser("ivanov");
User userObama = dbManager.getUser("obama");

Group teamA = dbManager.getGroup("teamA");
Group teamB = dbManager.getGroup("teamB");
Group teamC = dbManager.getGroup("teamC");

// method setGroupsForUser must implement transaction!
dbManager.setGroupsForUser(userIvanov, teamA);
dbManager.setGroupsForUser(userPetrov, teamA, teamB);
dbManager.setGroupsForUser(userObama, teamA, teamB, teamC);

for (User user : dbManager.findAllUsers()) {
 printList(dbManager.getUserGroups(user));
 System.out.println("~~~~~");
}
// teamA
// teamA teamB
// teamA teamB teamC

System.out.println("============================");

// Part 4

// on delete cascade!
dbManager.deleteGroup(teamA);

// Part 5
teamC.setName("teamX");
dbManager.updateGroup(teamC);

// Part 6
printList(dbManager.findAllGroups());
```

```
    // groups ==> [teamB, teamX]
  }
}
```

Создать и реализовать соответствующие типы таким образом, чтобы при запуске класса Demo отрабатывал а соответствующая функциональность.

Part 1

```
dbManager.insertUser(User.createUser("petrov"));
dbManager.insertUser(User.createUser("obama"));
printList(dbManager.findAllUsers());
```

Метод DBManager#insertUser должен модифицировать поле id объекта User.

Метод DBManager#findAllUsers возвращает объект java.util.List<User>.

Part 2

```
dbManager.insertGroup(Group.createGroup("teamB"));
dbManager.insertGroup(Group.createGroup("teamC"));
printList(dbManager.findAllGroups());
```

Метод DBManager#insertGroup должен модифицировать поле id объекта Group.

Метод DBManager#findAllGroups возвращает объект java.util.List<Group>.

Part 3

```
User userPetrov = dbManager.getUser("petrov");
User userIvanov = dbManager.getUser("ivanov");
User userObama = dbManager.getUser("obama");

Group teamA = dbManager.getGroup("teamA");
Group teamB = dbManager.getGroup("teamB");
Group teamC = dbManager.getGroup("teamC");

// method setGroupsForUser must implement transaction!
dbManager.setGroupsForUser(userIvanov, teamA);
dbManager.setGroupsForUser(userPetrov, teamA, teamB);
dbManager.setGroupsForUser(userObama, teamA, teamB, teamC);

for (User user : dbManager.findAllUsers()) {
    printList(dbManager.getUserGroups(user));
    System.out.println("~~~~~");
}
```

Метод DBManager#setGroupsForUser должен реализовывать транзакцию. Грамотно реализовать логику commit/rollback транзакции.

Метод DBManager#getUserGroups возвращает объект java.util.List<Group>.

Part 4

```
// on delete cascade!
dbManager.deleteGroup(teamA);
```

Метод DBManager#deleteGroup удаляет группу по имени. Все дочерние записи из таблицы users_groups также должны быть удалены. Последнее реализовать с помощью каскадных ограничений ссылочной целостности: ON DELETE CASCADE.

Part 5

```
teamC.setName("teamX");
dbManager.updateGroup(teamC);
```

Метод DBManager#updateGroup обновляет группу.

Замечания

1. В качестве базы данных использовать любую реляционную БД.

2. БД содержит три таблицы: users (id, login); groups (id, name); users_groups (user_id, group_id)
3. Изначально таблицы БД должны иметь некоторое наполнение (см. код Demo)
4. В корне создать каталог sql и сохранить в нем скрипт создания базы данных db-create.sql

---

Derby

Если вы будете использовать JavaDB (Apache Derby), то в данном архиве есть полезные скрипты, которые упрощают процесс запуска СУБД и выполнения скриптов: derby.zip