Zaawansowane Techniki Programowania
Praca domowa 7
Dominik Freicher
12.12.2016

OPIS ZADANIA

Celem zadania jest implementacja aplikacji klienckiej AppMain, która odpowie na pytanie czy wskazany student o imieniu firstName i nazwisku lastName z przedmiotu o nazwie courseName uzyskał ocenę punktową poniżej czy powyżej mediany ocen dla tego przedmiotu. W przypadku, gdy ocena jest niższa od mediany dla wskazanego przedmiotu program winien zwrócić (wyprowadzić na standardowy strumień wyjściowy) wartość ujemną wyrażoną w procentach i określającą o ile procent wynik indywidualny jest niższy od mediany, w przypadku gdy ocena jest równa medianie – należy wyprowadzić wartość 0, gdy ocena uzyskana przez studenta jest wyższa od mediany – dodatnią wyrażoną w procentach i wskazującą o ile wynik indywidualny studenta jest wyższy od mediany.

OPIS MECHANIZMU WYSZUKIWANIA

Do zlokalizowania klasy DbManager oraz nawiązania z nią połączenia wykorzystany został interfejs JNDI (ang. Java Naming and Directory Interface). Pozwala on klientom na odkrywanie oraz wyszukiwania danych i obiektów na podstawie nazw.

Aby nawiązać połączenie należy utworzyć obiekt InitialContext:

```
InitialContext ctx = new InitialContext();
```

Następnie należy skorzystać z metody lookup, której implementacja umożliwia odnalezienie zasobu po nazwie:

```
IDbManager obj = (IDbManager) ctx.lookup("java:global/ejb-project/
DbManager!pl.jrj.db.IDbManager");
```

Ścieżka przestrzeni nazw ma postać:

```
java:global/ejb-project/DbManager!pl.jrj.db.IDbManager
```

DOSTĘP DO BAZY DANYCH

Dostęp do danych zrealizowany został przy pomocy mechanizmu JPA. Mechanizm ten jest odpowiedzialny za mapowanie relacyjno-obiektowe. W celu uzyskania obiektu EntityManager pozwalającego wykonywać operację na bazie, należy wykonać metodę createEntityManagerFactory. Metoda ta, jako parametr przyjmuje nazwę kontekstu (persistence-unit) zlokalizowanego w pliku persistence.xml.

Zaawansowane Techniki Programowania
Praca domowa 7
Dominik Freicher
12.12.2016

ALGORYTM WYZNACZANIA WYNIKU

1. Pierwszym krokiem jest pobranie danych z pliku wejściowego tj. imienia, nazwiska studenta oraz nazwy przedmiotu.
2. Następnie przy użyciu poniższego zapytania, uzyskujemy ocenę z kursu dla konkretnego studenta. Jako parametry użyte zostają wartości pobrane z pliku w kroku nr 1.

```
TypedQuery<Integer> q = em.createQuery("SELECT s.mark "
                    + "FROM StudentCourse s "
                    + "WHERE s.student.firstName = :studentFirstName "
                    + "AND s.student.lastName = :studentLastName "
                    + "AND s.course.courseName = :courseName",
            Integer.class);
```

3. Kolejnym krokiem jest wyznaczenie mediany ocen z kursu. W celu jej wyznaczenia z użyciem poniższego zapytania zwrócone zostają wszystkie uzyskane oceny z konkretnego kursu. Jako parametr użyta zostaje nazwa kursu pobrana z pliku w kroku nr 1.

```
TypedQuery<Integer> q = em.createQuery("SELECT s.mark "
                    + "FROM StudentCourse s "
                    + "WHERE s.course.courseName = :courseName",
            Integer.class);
```

4. Rezultat końcowy wyliczony zostaje w następujący sposób:
   - W przypadku nieparzystej liczby ocen mediana to wartość środkowa.
   - W przypadku parzystej liczby ocen mediana to średnia arytmetyczna pomiędzy dwiema środkowymi obserwacjami
   - Jeżeli ocena uzyskana przez studenta jest mniejsza od mediany, należy wykorzystać wzór:

$$result = \frac{\mathrm{Ost} - \mathrm{m}}{\mathrm{m}} * 100$$

gdzie: Ost – ocena studenta, m - mediana

5. Uzyskany wynik zwrócony zostaje w strumieniu wyjściowym jako pojedyncza liczba.

Zaawansowane Techniki Programowania
Praca domowa 7
Dominik Freicher
12.12.2016

STRUKTURY DANYCH

W skład projektu wchodzą klasy `AppMain.java` `IDbManager.java` `Course.java` `StudentCourse.java` `Student.java` `StudentCourseId.java`

```
AppMain.java
void registerWork()
void fetchingData(String fileName)
void calculateResult()
void printResult()
Integer getStudentGrade(EntityManager em)
List<Integer> getCourseGrades(EntityManager em)
```

Główna klasa aplikacji klienckiej, w metodzie main jako argument przekazana zostaje ścieżka do pliku, następnie wywołana zostaje metoda registerWork, której zadaniem jest zarejestrowanie zadania z użyciem metody register komponentu DbManager. Następnie wywołana zostaje metoda fetichingData, która pobiera dane z pliku. Kolejnym krokiem jest wywołanie metody calculateResult, która odczytuje dane ze źródła danych oraz wykonuje niezbędne obliczenia. Metoda getStudentGrade zwraca ocenę z przedmiotu dla konkretnego studenta. Metoda getCourseGrades zwraca wszystkie uzyskane stopnie z danego jako parametr przedmiotu. Metoda printResult odpowiedzialna jest za wyświetlenie rezultatu.

`IDbManager.java` – Interfejs zawierający deklaracje metody register
`boolean register(int hwork, String album)`

`Course.java` – Klasa zmapowana na podstawie tabeli bazy danych Tbl_Courses
`StudentCourse.java` – Klasa zmapowana na podstawie tabeli bazy danych Tbl_StudentCourse
`Student.java` – Klasa zmapowana na podstawie tabeli bazy danych Tbl_Students
`StudentCourseId.java` – Klasa reprezentująca klucz główny tabeli Tbl_StudentCourse

URUCHOMIENIE

Proces kompilacji jest możliwy z użyciem komendy:
`javac -cp <app-server-modules> -Xlint AppMain.java IDbManager.java *.java`

Uruchomienie programu możliwe jest z użyciem komendy:
`java -cp <app-server-modules> AppMain <file>`