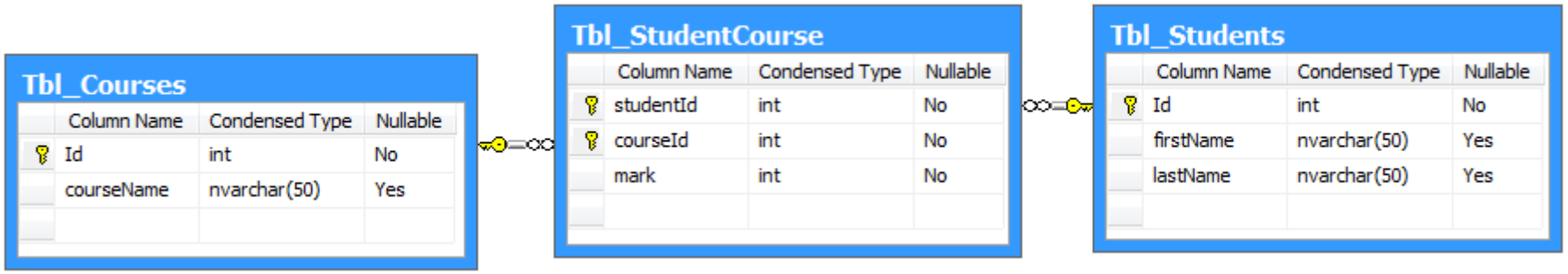
Praca domowa 09 – entity

Termin zwrotu : 25 maja godz. 23.00
Zadanie uznaje się za zaliczone, gdy praca oceniona zostanie na co najmniej 6 pkt.

W repozytorium danych przechowywane są wyniki (kolumna *mark*) uzyskane przez studentów w ramach wybranych przedmiotów. Poszczególne oceny mają charakter punktów w skali 0..100. Diagram odpowiedniego fragmentu bazy danych przedstawiono poniżej.



Należy stworzyć aplikację klienta `AppMain.java` odpowiadającą 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.

Rozwiązanie musi być oparte (dla potrzeb operowania danymi) o wykorzystanie *entity beans*. Dla potrzeb zadania zaprojektować można zgodną z potrzebami rozwiązania pewną ilość komponentów oraz niezbędnych interfejsów (nie wprowadza się dalszych ograniczeń co do ilości oraz nazewnictwa wykorzystanych komponentów).

Program ma być zapisany w pliku `AppMain.java` oraz pozostałych – wykonanych dla potrzeb zadania – elementach rozwiązania. Poszczególne elementy rozwiązania nie mogą korzystać z bibliotek zewnętrznych innych niż niezbędne moduły serwera (jak np. gf-client.jar, javaee.jar itp.).

Proces kompilacji musi być możliwy z użyciem komendy

```
javac -cp <app-server-modules> -Xlint AppMain.java *.java
```

Dla potrzeb dostępu do danych wykorzystane zostaną mechanizmy JPA. Niezbędny plik persistence.xml, który będzie używany w procesie testowania zadania (opisujący *persistence context*) podano poniżej :

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
  <persistence xmlns="http://java.sun.com/xml/ns/persistence" version="1.0">
      <persistence-unit name="myPersistence" transaction-type="JTA">
          <exclude-unlisted-classes>false</exclude-unlisted-classes>
          <jta-data-source>XXXXX</jta-data-source>
      </persistence-unit>
</persistence>
```

gdzie XXXXX jest zależne od instalacji (wskazuje na zasób JDBC specyfikujący źródło danych – bazę danych), a `myPersistence` jest nazwą kontekstu wykorzystywaną dla potrzeb zadania.

Uruchomienie programu winno być możliwe z użyciem komendy

```
java -cp <app-server-modules> AppMain <file>
```

Parametr `<file>` wskazuje na plik tekstowy zawierający w pierwszej linii nazwę analizowanego przedmiotu (*courseName*) a w linii kolejnej dane studenta (*firstName* oraz *lastName* rozdzielone przynajmniej jedną spacją). Wynik końcowy (w strumieniu wyjściowym nie powinny pojawiać się jakiekolwiek inne elementy – np. wydruki kontrolne) działania programu musi zawierać pojedynczą liczbę, a więc np.

```
Wynik : -12%
```

Wymagania :

- Klasa implementująca aplikację winna zostać zdefiniowane w pliku `AppMain.java`.
- W pliku README.pdf winien być zawarty opis mechanizm operowania danymi oraz algorytm wyznaczania wyniku.
- Proces obliczenia rozwiązania winien się kończyć w czasie nie przekraczającym 1 min (orientacyjnie dla typowego notebooka). Po przekroczeniu limitu czasu zadanie będzie przerywane, i traktowane podobnie jak w sytuacji błędów wykonania (czyli nie podlega dalszej ocenie).

Sposób oceny **:**

- 1 pkt – **Weryfikacja** : czy program jest skompletowany i spakowany zgodnie z ogólnymi zasadami przesyłania zadań.
- 1 pkt – **Kompilacja** : każdy z plików winien być kompilowany bez jakichkolwiek błędów lub ostrzeżeń (w sposób omówiony wyżej)
- 1 pkt – **Wykonanie** : program powinien wykonywać się bez jakichkolwiek błędów i ostrzeżeń (dla pliku danych wejściowych zgodnych z wyżej zamieszczoną specyfikacją) z wykorzystaniem omówionych wyżej parametrów linii komend
- 2 pkt – **README** : plik README.pdf dokumentuje w sposób kompletny i właściwy sposób zestawiania połaczenia
- 1 pkt – **Styl kodowania** : czy funkcji i zmienne posiadają samo-wyjaśniające nazwy ? Czy podział na funkcje ułatwia czytelność i zrozumiałość kodu ? Czy funkcje eliminują (redukują) powtarzające się bloki kodu ? Czy wcięcia, odstępy, wykorzystanie nawiasów itp. (formatowanie kodu) są spójne i sensowne ?
- 4 pkt – **Poprawność algorytmu** : czy algorytm został zaimplementowany poprawnie a wynik odpowiada prawidłowej (określonej zbiorem danych testowej) wartości.