Datenbanksysteme (VU 4.0, 184.686)
Übungsteil, WS 2013/2014

Beispiel 3

Große Teile von Beispiel 3 basieren auf der im Zuge von Beispiel 2 erstellten Datenbank. Sie können entweder die Musterlösung als Basis für diese Aufgabe verwenden oder ihre eigene Lösung weiterverwenden. Die Musterlösung wird mit Ablauf der Abgabegespräche zu Beispiel 2 auf der Übungsseite der Lehrveranstaltung veröffentlicht.

Wenn Sie ihre eigene Lösung weiterverwenden wollen, kann es passieren, dass aufgrund anderer Namensgebung (für Tabellen und/oder Attribute) kleine Adaptierungen vorgenommen werden müssen. Konkret sind einige Codeblöcke im Java-Teil vorgegeben, welche auf die Bezeichnungen der Musterlösung abgestimmt sind. Diese müssten Sie unter Umständen anpassen, oder Sie passen ihre Lösung aus Beispiel 2 daran an.

Grundgerüst

Als Grundgerüst finden Sie auf der Übungsseite ein ZIP-Archiv mit einer grundlegenden Projektstruktur die Sie als Vorlage für diese Übung nutzen müssen. Weiters wird Ihnen in diesem Gerüst ein ant-Skript zur Verfügung gestellt (build.xml) um das Testen und die Abgabe der Übung zu vereinfachen: dbs-template.zip

Beachten Sie:

- Alle Dateien und Pfade beziehen sich auf diese Vorlage
- Genauere Instruktionen zur Verwendung der ant Targets finden Sie in den einzelnen Übungsbeschreibungen

Teil 1a - Java (Concurrency)

Beachten Sie bitte, dass die vorgegebenen Codezeilen, die markiert sind durch einen Kommentarblock (“Vorgegebener Codeteil”) und geschlossen werden durch eine Kommentarzeile aus Hashtags (#), nicht verändert werden dürfen.

Die ersten beiden Szenarien (`Szenario1.java`, `Szenario2.java`) sind so angelegt, dass es eine von Ihnen zu ergänzende Transaktion gibt und eine bereits existierende Transaktion, die Ihnen von uns zur Verfügung gestellt wird.

Sie müssen hierbei nur die Methode `runTransactionA()` vervollständigen, indem sie den in der Vorlage bereitgestellten Code ergänzen. Die bereitgestellte Methode `runTransactionB()` dient zur Überprüfung der Funktionalität und erzeugt bei Wahl des falschen Isolation-Levels ein Fehlverhalten.

Die Parameter der Szenarien 1 und 2 sind jeweils die folgenden:

1. "a" oder "b" für die Wahl der jeweiligen Transaktion
2. Servername
3. Port-Nummer
4. Datenbank-Name
5. Username (optional)
6. Passwort (optional)

Zum vereinfachten Testen haben wir Ihnen in diesem Semester auch ein ant-Skript zur Verfügung gestellt. Wenn Sie Ihre Verbindungs- und Benutzerdaten in diesem Skript hinterlegen, müssen Sie nur mehr die folgenden Aufrufe benutzen, um die Programme zu starten:

1. `ant run-szenario1-a` (Für Ihre eigene Implementierung von Szenario 1)
2. `ant run-szenario1-b` (Für die von uns bereitgestellte Transaktion zu Szenario 1)
3. `ant run-szenario2-a` (Für Ihre eigene Implementierung von Szenario 2)
4. `ant run-szenario2-b` (Für die von uns bereitgestellte Transaktion zu Szenario 2)

Für einen Test öffnen Sie am besten zwei Konsolenfenster. Im ersten Fenster rufen Sie zuerst Ihre eigene Implementierung auf (also z.B. mittels des Befehls "`ant run-szenario1-a`") und im zweiten Fenster die von uns bereitgestellte zugehörige Transaktion (z.B. "`ant run-szenario1-b`"). Sie sollten nun zwei Konsolenfenster sehen, die beide auf einen Befehl warten. Wenn Sie im Fenster mit der Transaktion A die Eingabetaste drücken, startet der von Ihnen programmierte Ablauf und sollte die erste Ausgabe anzeigen. Danach starten Sie Transaktion B ebenfalls mit der Eingabetaste. Die Transaktion B verändert nun den Datenbestand in der Datenbank. Wenn Sie nun wieder im anderen Konsolenfenster mit der Transaktion A die Enter-Taste betätigen, sollte im Erfolgsfall die korrekte zweite Ausgabe Ihres Programms erscheinen. Überprüfen Sie nun sorgfältig, ob sich Ihr Programm auch wirklich nach der Spezifikation verhält und keine unerwünschten Ergebnisse durch die Transaktion B entstehen. Durch eine andere Reihenfolge beim Drücken der Enter-Taste in den verschiedenen Programmen können Sie natürlich auch andere Ablaufmuster und Verschachtelungen der Transaktionen simulieren und testen.

Sie müssen anschließend beim Abgabegespräch Ihr Vorgehen bei der Erstellung und auch die Funktionsweise des Programms erklären können, insbesondere auch die Frage, warum Sie der Meinung sind, dass Ihr Programm die Aufgabenstellung optimal erfüllt.

Szenario 1:

Für den Jahresrückblick werden von der obersten Kommission Statistiken verlangt. Für die Erfüllung dieser Aufgabe benötigt der Koordinator der Kommission folgende Informationen:

1. Die durchschnittliche Anzahl der Studenten, die sich im Jahr 2012 zu Austauschprogrammen beworben haben,
2. sowie die durchschnittliche Anzahl der Studenten die sich allgemein zu Austauschprogrammen beworben haben und
3. das Verhältnis dieser beiden Werte

Diese Zahlenwerte sollen in einer Transaktion ermittelt werden.

Da diese Statistiken aus der Sicht des Koordinators nur eine untergeordnete Rolle spielen, ist es ihm ein dringendes Anliegen, dass der Echtzeitbetrieb so wenig wie möglich belastet wird. Daher sind etwaige minimale Abweichungen bei diesen Statistiken irrelevant.

Beachten Sie diese Anforderungen und den möglichen Zusammenhang der beiden Abfragen. Überlegen Sie sich, welches Isolation-Level hierfür ausreicht und begründen Sie ihre Entscheidung beim Abgabegespräch.

Ermitteln Sie die geforderten Informationen in den bereitgestellten Codeblöcken der Transaktion A in Szenario 1. Mit der zur Verfügung gestellten Transaktion B können Sie verschiedenste Verschachtelungen der einzelnen Befehle durchspielen und so gewollte, oder aber auch ungewollte Effekte des gewählten Isolation-Levels sehen.

Szenario 2:

Neben der oberen Statistik wurden von Firmen auch noch zusätzliche Statistiken angefordert. Einerseits sollen für die Firma mit der Firmennummer '10' der Name, das Land und das Jahr aller Austauschprogramme, gemeinsam mit der Anzahl der jeweiligen Praktika, welche die Firma zu dem jeweiligen Austauschprogramm angeboten hat ermittelt werden. Das Ergebnis soll nach der Anzahl aufsteigend sortiert werden.

In der gleichen Transaktion soll auch ermittelt werden wie viele Praktika zu Austauschprogrammen in den verschiedenen Ländern stattgefunden haben. Diese Daten sollen absteigend nach der Anzahl und aufsteigend nach dem Land sortiert werden.

Da der Koordinator die Daten gerne mobil abrufen möchte, soll die zu übertragende Datenmenge möglichst gering sein, d.h. es ist notwendig, dass von der Datenbank immer nur die jeweils notwendigen Tupel ausgelesen werden. Ebenso sollen alle Berechnungen und Sortierungen von der Datenbank übernommen werden. Ihre Java-Anwendung soll die abgerufenen Daten nur

mehr anzeigen.

Der Koordinator wünscht auch ausdrücklich, dass es nicht möglich sein soll, das Veränderungen in anderen Tabellen während der Laufzeit der Transaktion die Daten verfälschen können. Das heißt, dass die Daten aus der ersten Abfrage unbedingt konsistent mit denen aus der zweiten Abfrage sein müssen.

Als Ausgabe soll Ihr Programm jeweils Listen der Ergebnisse der beiden Abfragen ausgeben. Achten Sie darauf, dass Sie unerwünschtes Verhalten vermeiden, aber die Datenbank nicht durch ein zu restriktives Isolation-Level unnötig blockieren.

Setzen Sie nun wieder die geforderten Kommandos in die Codeblöcke von Transaktion A in Szenario 2. Auch in diesem Szenario gibt es wieder eine vorgegebene Transaktion B, welche gewisse Effekte bei der parallelen Ausführung hervorbringt.

Hinweis:

Im Rahmen des Abgabegesprächs wird nicht nur die Korrektheit der Lösungen, sondern vor allem auch das Verständnis überprüft. Sie sollten beim Abgabegespräch in der Lage sein, zu erklären wie sich die Wahl des Isolation-Levels auf das Programmverhalten auswirkt. Zudem ist ein gewisses Basiswissen im Bereich Concurrency gefordert, dazu zählt zum Beispiel das Verständnis von Transaktionen (sprich z.B. ACID), die Charakteristika von Isolation-Levels, oder welche Probleme die einzelnen Isolation-Levels beheben (z.B. Dirty Read).

Teil 1b: Java (PL/pgSQL)

In der dritten Datei (`Szenario3.java`) sollen Sie die Methode calcAndPrintGehaelter() entsprechend der folgenden Angabe implementieren. Sie können dieses Programm entweder mittels der Kommandozeilen-Argumente oder wiederum direkt mittels des ant-Skripts (“ant run-szenario3”) starten.

Die Argumente für den Aufruf dieses Szenarios sind die folgenden:

1. Servername
2. Port-Nummer
3. Datenbank-Name
4. Username (optional)
5. Passwort (optional)

Auch hier müssen Sie beim Abgabegespräch in der Lage sein, detaillierte Fragen zum Verständnis der Anwendung und Ihres Vorgehens bei der Implementierung zu beantworten.

Szenario 3: (Gehaltsliste)

Für eine Statistik soll für jeden Mitarbeiter das aktuelle Gehalt berechnet und ausgegeben werden.

Die folgenden Aktionen sollen von der Prozedur ausgeführt werden:

1. Erstellen Sie ein Prepared Statement, das die Stored Function "`f_calc_salary(varchar, date)`" aus Beispiel 2 verwendet, um für einen übergebenen Mitarbeiter zum aktuellen Datum das Gehalt zu berechnen.
2. Benutzen sie das oben beschriebene Prepared Statement um für jeden Mitarbeiter das aktuelle Gehalt zu berechnen:
3. Geben Sie zum Abschluss alle Mitarbeiter mit ihrer SVNR, dem aktuellen Datum, sowie dem aktuellen Gehaltsanspruch aus.

Teil 2 - Abfragen

Die folgenden Aufgabenstellungen basieren auf der Datenbank, welche Sie für Beispiel 2 des Übungsteils zu erstellen hatten. Sie können entweder auf Ihrer Datenbank weiterarbeiten, oder die Musterlösung des zweiten Beispiels verwenden, welche auf der LVA-Homepage zum Download bereitsteht (allerdings erst nach den Abgabegesprächen zu Beispiel 2). Stellen Sie allerdings in jedem Fall in Ihrer Abgabe die Create-, Insert- und Drop-Befehle jener Datenbank bereit, auf der Sie arbeiten. Ersetzen Sie dazu im Verzeichnis `resources` in der Vorlage die Dateien.

Stellen Sie sicher, dass die Daten, mit denen Sie testen, der Spezifikation entsprechen (d.h. Kardinalitäten berücksichtigen usw.). Sie müssen keine Fälle berücksichtigen, die der Spezifikation widersprechen.

Lösen Sie die folgenden Probleme mittels SQL:

1. Geben Sie SVNR, Name sowie die Matrikelnummer jener Studenten aus, welche sich für alle Austauschprogramme beworben haben.
2. Wählen Sie per Hand eine Kommission aus, die anderen Kommissionen untergeordnet ist. Schreiben Sie eine Anfrage, die diese Kommission ausgibt, sowie rekursiv alle übergeordneten Kommissionen. Geben Sie für jede Kommission ID, Standort, Name, die ID der übergeordneten Kommission sowie die Stufe der Kommission in der Hierarchie aus. Die Stufe der jeweiligen Kommission ergibt sich dabei aus der Stufe der untergeordneten Kommission plus 1, wobei die von Ihnen per Hand ausgewählte Kommission logischerweise mit Stufe 0 beginnt. Zu der von Ihnen ausgewählten Kommission soll es in der Datenbank mindestens zwei Ebenen übergeordneter Kommissionen geben.
3. Geben Sie die PRNR, ABTNR, FANR, sowie den Aufgabenbereich und das Start- und End-Datum (von, bis) von jeden Praktika aus, die für die wenigsten Austauschprogramm ausgeschrieben sind, aber für welche die meisten Studenten ge-"shortlisted" sind. Sortieren Sie das Ergebnis zunächst absteigend nach von- und bis-Datum und anschließend

aufsteigend nach dem Aufgabenbereich.

Abgabe

Um ein abgabefertiges Archiv `dbs-exercise3-ws13.zip` zu erzeugen, führen Sie folgenden Befehl aus:

- `ant zip`

Diese ZIP-Datei ist bis spätestens 01.12.2013 um 23:59 im CourseManager abzugeben. Es wird stets die zuletzt hochgeladene Version Ihrer Lösung gewertet.

Führen Sie keine Änderung an der "Testumgebung" aus Teil 1 (`DBConnector.java`, Main-Methoden, usw.) durch, da ansonsten ihre Abgabe im Rahmen des Abgabegesprächs möglicherweise nicht ausgeführt werden kann. Gleiches gilt für die zur Verfügung gestellten Skripts aus Teil 2.

Abgabegespräche

Die Verteilung der Punkte erfolgt nach folgendem Schlüssel:

1. Datei `Szenario1.java`: 4
2. Datei `Szenario2.java`: 4
3. Datei `Szenario3.java`: 4
4. Datei `queries.sql`: 3

Im Rahmen des Kontrollgespräches wird nicht nur die Korrektheit, sondern vor allem das Verständnis der Konzepte überprüft. Durch die Übung sollen sowohl Ihre praktische Problemlösungskompetenz als auch das theoretische Wissen über Datenbanksysteme gefördert werden. Sie müssen daher bei den Abgabegesprächen in der Lage sein, nicht nur Ihre Beispiele zu erklären, sondern ebenfalls zeigen, dass Sie die bisher in der Vorlesung behandelte Theorie zu diesen Beispielen ausreichend verstanden haben. Dies soll Ihnen die Vorbereitung für die Prüfung erleichtern und so können Sie Ihr Wissen während der Abgabegespräche selbst testen und gegebenenfalls vertiefen.

Die volle Punktezahl gibt es nur, wenn die Beispiele korrekt gelöst wurden und die Lösung einwandfrei erklärt werden kann. Nicht selbstständig gelöste Abgaben werden mit 0 Punkten bewertet!

Erscheinen Sie in Ihrem eigenen Interesse **pünktlich** zum Abgabegespräch, da andernfalls nicht garantiert werden kann, dass Ihre gesamte Lösung in der verbleibenden Zeit beurteilt werden kann.

Bringen Sie bitte Ihren Studentenausweis zur Abgabe mit. Eine Abgabe ohne Ausweis ist nicht möglich.

Hinweise zur Verwendung von psql

Folgende Befehle können für Ihre Arbeit mit der interaktiven SQL-Shell psql von PostgreSQL hilfreich sein:

1. \?: Listet alle psql-internen Befehle samt Erklärung auf.
2. \i <dateiname>: Führt das Skript <dateiname> aus. Beispiel: \i create.sql
3. \o <dateiname>: Lenkt die Ausgabe in eine Datei mit dem Namen <dateiname>um. Lässt man den Parameter <dateiname> weg, so wird dieses Verhalten wieder abgestellt. Beispiel: \o index.txt