DES3UE Datenbanksysteme WS 2018 Übung 7

Abgabetermin: 21.11.2018, 13:30 Uhr

☐ **DES31UE Niklas** **Name** _____Niklas Vest_____ **Aufwand in h** _5_____
☐ **DES32UE Niklas**
☒ **DES32UE Traxler** **Punkte** _____ **Kurzzeichen Tutor** _____

Ziel dieser Übung ist die Vertiefung von PL/SQL in Paketen, dem Cursor-Konzept und Behandlung von Exceptions.

1. Datenbankpakete (PL/SQL-Packages) (10 Punkte – 4+6 Pkt)

Mit Hilfe von PL/SQL-Packages können Sie zusammengehörige PL/SQL-Typen, Variablen, Datenstrukturen, Exceptions und Unterprogramme in einer Bibliothek zusammenfassen. Packages bestehen normalerweise aus zwei Komponenten (Spezifikation und Body), die separat in der Datenbank gespeichert werden. Das Package selbst kann nicht aufgerufen, parametrisiert oder verschachtelt werden.

```
CREATE OR REPLACE PACKAGE top_customer_pkg AS
    …
END;
/
```

```
CREATE OR REPLACE PACKAGE BODY top_customer_pkg AS
    …
END;
/
```

1. Erstellen Sie ein Datenbankpaket top_customer_pkg mit der Funktion GetFilmcount (übergeben Sie eine Customer-Id als Parameter) und speichern Sie die Package Spezifikation und den Package Body gemeinsam ab. Erstellen Sie einen Testaufruf (zB für customer_id = 100). GetFilmcount soll aus der Sakila Datenbank die Anzahl der geliehenen Filme ermitteln, dazu werden nur Filme gezählt, die mindestens 60 Minuten lang sind.

2. Implementieren Sie nun im Package top_customer_pkg eine zusätzliche Prozedur GetTopNCustomers, die drei Eingabeparameter verlangt:

   - die Anzahl der Kunden n_count, um eine Top-N Liste der n_count besten Kunden ausgeben zu können
   - begin_date und
   - end_date, um die Auswahl von Datensätzen auf den Bereich begin_date <= rental_date <= end_date einschränken zu können.

   D.h. es sollen die Anzahl (n_count) Kunden ausgegeben werden, die die meisten Filme zwischen begin_date und end_date ausgeliehen haben. Wird keine Anzahl an Kunden angegeben, so sollen default-mäßig die Top-10-Kunden ausgegeben werden. Geben Sie auch für die anderen Parameter Default-Werte an, um einen Aufruf ohne Parameter zu ermöglichen.

   Verwenden Sie für die zu erstellende Prozedur eine CURSOR FOR LOOP, die Prozedur DBMS_OUTPUT.PUT_LINE und erstellen Sie zur Ermittlung der Film-Anzahl eine Erweiterung der bereits vorhandenen Funktion (verwenden Sie dazu Overloading/Überladen mit begin_date

und end_date). Sortieren Sie nach der Anzahl absteigend. Geben Sie die Package Spezifikation, den Package Rumpf sowie den Testaufruf (mit und ohne Parameter) an.

Beispielausgabe

```
...
The top 10 customers from 01.01.07 to 31.12.16 are:
ELEANOR HUNT: 44 films
…
```

2. Cursor mit FOR-UPDATE (10 Punkte – 1.-2. je 3 Pkt und 3.-4. je 2 Pkt)

Wenn mehrere Sessions für eine einzelne Datenbank vorhanden sind, besteht die Möglichkeit, dass die Zeilen einer bestimmten Tabelle aktualisiert wurden, nachdem Sie den Cursor geöffnet haben. Sie sehen die aktualisierten Daten nur, wenn Sie den Cursor erneut öffnen. Es ist daher günstiger, die Zeilen zu sperren, bevor Sie Zeilen aktualisieren oder löschen. Sie können zum Sperren der Zeilen die FOR-UPDATE-Klausel in der Cursor-Abfrage verwenden.

1. Erstellen Sie eine Tabelle top_customers, die customer_id und die Anzahl der Filme enthält. Speichern Sie in der Tabelle auch den User und das Datum an dem der jeweilige Datensatz eingefügt wurde. Eine weitere Spalte enthält das Datum an dem der Datensatz ‚deaktiviert' wurde (dh. die Person nicht mehr zu den Top Customers gehört), für aktive Datensätze ist dieser Eintrag NULL.
   Erweitern Sie Ihre Prozedur GetTopNCustomers dahingehend, dass zusätzlich zur Ausgabe mit DMBS_OUTPUT auch Datensätze in die Tabelle top_customers eingefügt werden.

2. Fügen Sie dem Datenbankpaket top_customer_pkg eine weitere Prozedur DeactivateTopCustomers hinzu, die Datensätze in der Tabelle top_customers mit weniger als einer angegebenen Anzahl an Filmen deaktiviert (das aktuelle Datum einträgt). Testen Sie Ihre Prozedur.

   Hinweise: Verwenden Sie beim Cursor die FOR-UPDATE und beim UPDATE die WHERE CURRENT OF-Klausel. Geben Sie die Package Spezifikation, den Package Body sowie den Testaufruf an.

3. Öffnen Sie eine zweite Datenbank-Session (Strg+Shift+N) und führen Sie die Prozedur DeactivateTopCustomers mit den gleichen Parametern in jeder Session aus, ohne ein COMMIT auszuführen. Wählen Sie die Parameter so, damit zumindest ein Satz aus top_customers selektiert wird. Was passiert in der zweiten Session? Führen Sie in der ersten ein COMMIT aus und beschreiben Sie die Auswirkungen.

4. Erweitern Sie nun den Cursor in der Prozedur DeactivateTopCustomers um die NOWAIT-Klausel. Wiederholen Sie den Test und erläutern Sie den Unterschied. Geben Sie die Package Spezifikation, den Package Rumpf sowie den Testaufruf an.

3. EXCEPTIONS **(4 Punkte –** 1. 2 Pkt und 2.-3. je 1 Pkt**)**

Diese Aufgabe behandelt die Verwendung von vordefinierten Exceptions.

1. Erstellen Sie eine PL/SQL-Prozedur mit dem Parameter name_part (vom Typ customer.last_name), die anhand eines angegebenen Namens(-teils) jene Kunden auswählt, deren Namen damit beginnen. Hinweis: Verwenden Sie keinen expliziten Cursor und vergleichen Sie mit LIKE. Wenn der an die Prozedur übergebene Namensteil nur eine Zeile zurückgibt, fügen Sie in die Tabelle messages den Namen des Kunden (Vor- und Nachname und customer_id) ein. Die Tabelle messages soll aus einer Spalte results vom Typ VARCHAR2(100) bestehen. Testen Sie die Prozedur anhand folgender Fälle: a) Keine Zeile wird selektiert. b) Eine Zeile wird selektiert. c) Mehrere Zeilen werden selektiert. Was stellen Sie fest?

2. Wenn der eingegebene Name keine Zeilen zurückgibt, behandeln Sie die Exception mit einem entsprechenden Exception Handler, und fügen Sie in die Tabelle messages die Meldung „No customer found where last name begins with <name>“ ein.

3. Behandeln Sie beliebige weitere Exceptions mit einem entsprechenden Exception Handler, und fügen Sie in die Tabelle messages die Meldung „An undefined error occurred“ ein.