Ausarbeitung UE07

1. Datenbankpakete (PL/SQL-Packages)

Anmerkung: Ein DEFAULT Wert für die Begrenzungen des Zeitintervalls (begin_date, end_date) ergibt für mich keinen Sinn, weswegen der Parameter n_count (für den ein DEFAULT Wert tatsächlich sinnvoll ist) ans Ende der Parameterliste verschoben wurde. Ich ändere sonst nie Schnittstellen, da ich einsehe, dass das unglaubliche Kompatiblitätsprobleme erzeugt. Ein statischer Wert als DEFAULT Wert erscheint mir bei Zeitintervallen aber nutzlos.

```
CREATE OR REPLACE PACKAGE top_customer_pkg AS
  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE) RETURN NUMBER;
  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE, begin_date IN DATE, end_date
IN DATE) RETURN NUMBER;
  PROCEDURE GetTopNCustomers(begin_date IN DATE, end_date IN DATE, n_count IN NUMBER
DEFAULT 10);
END;
/

CREATE OR REPLACE PACKAGE BODY top_customer_pkg AS

  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE) RETURN NUMBER IS
    film_count NUMBER;
  BEGIN
    SELECT COUNT(*) INTO film_count
    FROM customer
           INNER JOIN rental USING (customer_id)
           INNER JOIN inventory USING (inventory_id)
           INNER JOIN film USING (film_id)
    WHERE length >= 60 AND
      customer_id = cid;
    RETURN film_count;
  END;

  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE, begin_date IN DATE, end_date
IN DATE) RETURN NUMBER IS
    film_count NUMBER;
  BEGIN
    SELECT COUNT(*) INTO film_count
    FROM customer
           INNER JOIN rental USING (customer_id)
           INNER JOIN inventory USING (inventory_id)
           INNER JOIN film USING (film_id)
    WHERE length >= 60 AND
          customer_id = cid AND
          begin_date < rental_date AND
          rental_date < end_date;
    RETURN film_count;
  END;

  PROCEDURE GetTopNCustomers(begin_date IN DATE, end_date IN DATE, n_count IN NUMBER
DEFAULT 10) IS
    CURSOR customers IS
      SELECT first_name || ' ' || last_name AS name,
             top_customer_pkg.GetFilmCount(customer_id, begin_date, end_date) AS cnt
```

```
      FROM customer
      ORDER BY top_customer_pkg.GetFilmCount(customer_id, begin_date, end_date) DESC
      FETCH FIRST n_count ROWS ONLY;
  BEGIN
    FOR cust IN customers LOOP
      DBMS_OUTPUT.PUT_LINE(cust.name || ': ' || cust.cnt || ' films');
    END LOOP;
  END;

END;
/

-- 1.1
BEGIN
  DBMS_OUTPUT.PUT_LINE(top_customer_pkg.GetFilmCount(1));
END;
/
[2018-11-20 16:52:37] 31

-- 1.2
BEGIN
  top_customer_pkg.GetTopNCustomers(to_date('01.01.07', 'DD.MM.YY'), to_date('31.12.16',
'DD.MM.YY'));
END;
/
[2018-11-20 16:57:57] === Without n_count ===
[2018-11-20 16:57:58] ELEANOR HUNT: 44 films
[2018-11-20 16:57:58] KARL SEAL: 41 films
[2018-11-20 16:57:58] MARCIA DEAN: 38 films
[2018-11-20 16:57:58] RHONDA KENNEDY: 37 films
[2018-11-20 16:57:58] WESLEY BULL: 37 films
[2018-11-20 16:57:58] CLARA SHAW: 37 films
[2018-11-20 16:57:58] MARION SNYDER: 36 films
[2018-11-20 16:57:58] TAMMY SANDERS: 36 films
[2018-11-20 16:57:58] DAISY BATES: 35 films
[2018-11-20 16:57:58] SUE PETERS: 35 films
[2018-11-20 16:57:58] === With n_count ===
[2018-11-20 16:57:58] ELEANOR HUNT: 44 films
[2018-11-20 16:57:58] KARL SEAL: 41 films
[2018-11-20 16:57:58] MARCIA DEAN: 38 films
[2018-11-20 16:57:58] RHONDA KENNEDY: 37 films
[2018-11-20 16:57:58] CLARA SHAW: 37 films
```

2. Cursor mit FOR-UPDATE

2.1

```
CREATE TABLE top_customers (
```

```
  customer_id      NUMBER                          NOT NULL,
  nr_of_films      NUMBER                          NOT NULL,
  date_created     DATE          DEFAULT SYSDATE  NOT NULL,
  created_by       VARCHAR2(50) DEFAULT USER     NOT NULL,
  date_deactivated DATE          DEFAULT NULL,
  CONSTRAINT top_customers_pk PRIMARY KEY (customer_id),
  CONSTRAINT top_customers_fk FOREIGN KEY (nr_of_films)
    REFERENCES customer (customer_id)
);

CREATE OR REPLACE PACKAGE BODY top_customer_pkg AS

  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE) RETURN NUMBER IS
    film_count NUMBER;
  BEGIN
    SELECT COUNT(*) INTO film_count
    FROM customer
           INNER JOIN rental USING (customer_id)
           INNER JOIN inventory USING (inventory_id)
           INNER JOIN film USING (film_id)
    WHERE length >= 60 AND
          customer_id = cid;
    RETURN film_count;
  END;

  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE, begin_date IN DATE, end_date
IN DATE) RETURN NUMBER IS
    film_count NUMBER;
  BEGIN
    SELECT COUNT(*) INTO film_count
    FROM customer
           INNER JOIN rental USING (customer_id)
           INNER JOIN inventory USING (inventory_id)
           INNER JOIN film USING (film_id)
    WHERE length >= 60 AND
          customer_id = cid AND
          begin_date < rental_date AND
          rental_date < end_date;
    RETURN film_count;
  END;

  PROCEDURE GetTopNCustomers(begin_date IN DATE, end_date IN DATE, n_count IN NUMBER
DEFAULT 10) IS
    CURSOR customers IS
      SELECT customer_id,
             first_name || ' ' || last_name AS name,
             top_customer_pkg.GetFilmCount(customer_id, begin_date, end_date) AS cnt
      FROM customer
      ORDER BY top_customer_pkg.GetFilmCount(customer_id, begin_date, end_date) DESC
      FETCH FIRST n_count ROWS ONLY;
  BEGIN
    DELETE FROM top_customers;
    FOR cust IN customers LOOP
      INSERT INTO top_customers
          (customer_id, nr_of_films)
      VALUES
             (cust.customer_id, cust.cnt);
      DBMS_OUTPUT.PUT_LINE(cust.name || ': ' || cust.cnt || ' films');
    END LOOP;
  END;

END;
/
BEGIN
```

```
  top_customer_pkg.GetTopNCustomers(to_date('01.01.07', 'DD.MM.YY'), to_date('31.12.16',
'DD.MM.YY'));
END;
/
```

	customer_id	nr_of_films	date_created	created_by	date_deactivated
1	148	44	2018-11-20 17:03:31	S1710307099	<null>
2	526	41	2018-11-20 17:03:31	S1710307099	<null>
3	236	38	2018-11-20 17:03:31	S1710307099	<null>
4	137	37	2018-11-20 17:03:31	S1710307099	<null>
5	469	37	2018-11-20 17:03:31	S1710307099	<null>
6	144	37	2018-11-20 17:03:31	S1710307099	<null>
7	178	36	2018-11-20 17:03:31	S1710307099	<null>
8	75	36	2018-11-20 17:03:31	S1710307099	<null>
9	295	35	2018-11-20 17:03:31	S1710307099	<null>
10	197	35	2018-11-20 17:03:31	S1710307099	<null>

2.2

```
CREATE OR REPLACE PACKAGE top_customer_pkg AS

  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE) RETURN NUMBER;
  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE, begin_date IN DATE, end_date
IN DATE) RETURN NUMBER;
  PROCEDURE GetTopNCustomers(begin_date IN DATE, end_date IN DATE, n_count IN NUMBER
DEFAULT 10);
  PROCEDURE DeactivateTopCustomers(n_films IN NUMBER);
END;
/

CREATE OR REPLACE PACKAGE BODY top_customer_pkg AS
  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE) RETURN NUMBER IS
    film_count NUMBER;
  BEGIN
    SELECT COUNT(*) INTO film_count
    FROM customer
           INNER JOIN rental USING (customer_id)
           INNER JOIN inventory USING (inventory_id)
           INNER JOIN film USING (film_id)
    WHERE length >= 60 AND
          customer_id = cid;
    RETURN film_count;
  END;

  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE, begin_date IN DATE, end_date
IN DATE) RETURN NUMBER IS
    film_count NUMBER;
  BEGIN
    SELECT COUNT(*) INTO film_count
    FROM customer
           INNER JOIN rental USING (customer_id)
           INNER JOIN inventory USING (inventory_id)
           INNER JOIN film USING (film_id)
    WHERE length >= 60 AND
          customer_id = cid AND
          begin_date < rental_date AND
          rental_date < end_date;
    RETURN film_count;
  END;
```

```
  PROCEDURE GetTopNCustomers(begin_date IN DATE, end_date IN DATE, n_count IN NUMBER
DEFAULT 10) IS
    CURSOR customers IS
      SELECT customer_id,
             first_name || ' ' || last_name AS name,
             top_customer_pkg.GetFilmCount(customer_id, begin_date, end_date) AS cnt
      FROM customer
      ORDER BY top_customer_pkg.GetFilmCount(customer_id, begin_date, end_date) DESC
      FETCH FIRST n_count ROWS ONLY;
  BEGIN
    DELETE FROM top_customers;
    FOR cust IN customers LOOP
      INSERT INTO top_customers
          (customer_id, nr_of_films)
      VALUES
             (cust.customer_id, cust.cnt);
      DBMS_OUTPUT.PUT_LINE(cust.name || ': ' || cust.cnt || ' films');
    END LOOP;
  END;

  PROCEDURE DeactivateTopCustomers(n_films IN NUMBER) IS
    CURSOR deactivat0r IS
      SELECT *
      FROM top_customers
      WHERE nr_of_films < n_films
    FOR UPDATE OF date_deactivated;
  BEGIN
    FOR x IN deactivat0r LOOP
      UPDATE top_customers
          SET date_deactivated = SYSDATE
      WHERE CURRENT OF deactivat0r;
    END LOOP;
  END;

END;
/

BEGIN
  top_customer_pkg.DeactivateTopCustomers(37);
END;
/
```

	customer_id	nr_of_films	date_created	created_by	date_deactivated
1	148	44	2018-11-20 17:03:31	S1710307099	<null>
2	526	41	2018-11-20 17:03:31	S1710307099	<null>
3	236	38	2018-11-20 17:03:31	S1710307099	<null>
4	137	37	2018-11-20 17:03:31	S1710307099	<null>
5	469	37	2018-11-20 17:03:31	S1710307099	<null>
6	144	37	2018-11-20 17:03:31	S1710307099	<null>
7	178	36	2018-11-20 17:03:31	S1710307099	2018-11-20 17:05:52
8	75	36	2018-11-20 17:03:31	S1710307099	2018-11-20 17:05:52
9	295	35	2018-11-20 17:03:31	S1710307099	2018-11-20 17:05:52
10	197	35	2018-11-20 17:03:31	S1710307099	2018-11-20 17:05:52

2.3

Die Session, in welcher der Prozeduraufruf zu erst abgesetzt wird, erhält ein Exklusivrecht für Operationen auf die Spalte “date_deactivated” (wie spezifiziert). Die zweite Session muss somit

warten bis dieses Exklusivrecht (mittels COMMIT) aufgehoben wird bevor sie Selber ein solches Recht erhalten kann.

2.4

```
CREATE OR REPLACE PACKAGE BODY top_customer_pkg AS

  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE) RETURN NUMBER IS
    film_count NUMBER;
  BEGIN
    SELECT COUNT(*) INTO film_count
    FROM customer
           INNER JOIN rental USING (customer_id)
           INNER JOIN inventory USING (inventory_id)
           INNER JOIN film USING (film_id)
    WHERE length >= 60 AND
          customer_id = cid;
    RETURN film_count;
  END;

  FUNCTION GetFilmCount(cid IN customer.customer_id%TYPE, begin_date IN DATE, end_date
IN DATE) RETURN NUMBER IS
    film_count NUMBER;
  BEGIN
    SELECT COUNT(*) INTO film_count
    FROM customer
           INNER JOIN rental USING (customer_id)
           INNER JOIN inventory USING (inventory_id)
           INNER JOIN film USING (film_id)
    WHERE length >= 60 AND
          customer_id = cid AND
          begin_date < rental_date AND
          rental_date < end_date;
    RETURN film_count;
  END;

  PROCEDURE GetTopNCustomers(begin_date IN DATE, end_date IN DATE, n_count IN NUMBER
DEFAULT 10) IS
    CURSOR customers IS
      SELECT customer_id,
             first_name || ' ' || last_name AS name,
             top_customer_pkg.GetFilmCount(customer_id, begin_date, end_date) AS cnt
      FROM customer
      ORDER BY top_customer_pkg.GetFilmCount(customer_id, begin_date, end_date) DESC
      FETCH FIRST n_count ROWS ONLY;
  BEGIN
    DELETE FROM top_customers;
    FOR cust IN customers LOOP
      INSERT INTO top_customers
          (customer_id, nr_of_films)
      VALUES
             (cust.customer_id, cust.cnt);
      DBMS_OUTPUT.PUT_LINE(cust.name || ': ' || cust.cnt || ' films');
    END LOOP;
  END;

  PROCEDURE DeactivateTopCustomers(n_films IN NUMBER) IS
    CURSOR deactivat0r IS
      SELECT *
      FROM top_customers
```

```
      WHERE nr_of_films < n_films
    FOR UPDATE OF date_deactivated NOWAIT;
  BEGIN
    FOR x IN deactivat0r LOOP
      UPDATE top_customers
      SET date_deactivated = SYSDATE
      WHERE CURRENT OF deactivat0r;
    END LOOP;
  END;

END;
/

-- Execute this boi in a second session
BEGIN
  top_customer_pkg.DeactivateTopCustomers(37);
END;
/
[61000][54] ORA-00054: resource busy and acquire with NOWAIT specified or timeout expired
ORA-06512: at "S1710307099.TOP_CUSTOMER_PKG", line 54
ORA-06512: at "S1710307099.TOP_CUSTOMER_PKG", line 59
ORA-06512: at line 2
```

3. EXCEPTIONS

Anmerkung: Die spezifikation zur Tabelle “messages” war sehr verwirrend. Gleichzeitig soll die Tabelle nur eine Spalte haben, jedoch drei verschiedene Informationen pro Datensatz enthalten. Ich habe diese Informationen also einfach in eine Zeichenkette zusammengefasst in der Hoffnung, dass eben das gefragt war.

```
CREATE TABLE messages (
  results VARCHAR2(100) NOT NULL
);

CREATE OR REPLACE PROCEDURE FindCustomer(name_part IN customer.last_name%TYPE) IS
  cust customer%ROWTYPE;
BEGIN

  SELECT * INTO cust
  FROM customer
  WHERE last_name LIKE name_part || '%';

  IF SQL%FOUND THEN
    INSERT INTO messages VALUES (cust.first_name ||  ' ' || cust.last_name || ' ' ||
cust.customer_id);
  END IF;

EXCEPTION

  WHEN NO_DATA_FOUND THEN
    INSERT INTO messages VALUES ('No customer found where last name begins with ' ||
name_part);

  WHEN OTHERS THEN
    INSERT INTO messages VALUES ('An undefined error occurred');

END;
/
```

```
BEGIN
  -- just 1 line
  FindCustomer('HUNTE');
  -- multiple lines
  FindCustomer('HUNT');
  --  no lines
  FindCustomer('JHIMHDIUSDFMH');
  FOR message IN (SELECT * FROM messages) LOOP
    DBMS_OUTPUT.PUT_LINE(message.results);
  END LOOP;
END;
/
```

	results
1	CHARLOTTE HUNTER 130
2	An undefined error occurred
3	No customer found where last name begins with JHIMHDIUSDFMH