EISTI Ecole Internationale des Sciences du Traitement de l'Information

M1 Adeo 2013-2014

Databases – PL/SQL

N. Nguyen – B. George – R. Chelouah

Exercises – Series 2
(Use of PL/SQL + Exceptions)

Introduction

Here, we'll take time to discover how to use PL/SQL with SQL requests, and how to handle exceptions, whether they are pre-defined by Oracle or user-defined.

0 – Easy exercises firtst

Exercise 0.1
Here, we're goint to learn how to use Oracle's pre-defined exceptions. First of all, let's take back Exercise 0.4 from Series 1. First, there is the CUSTOMERS table.

```
CREATE TABLE CUSTOMERS(
        ID INT NOT NULL,
        NAME VARCHAR (20) NOT NULL,
        AGE INT NOT NULL,
        ADDRESS CHAR (25),
        SALARY
        DECIMAL (18, 2),
        PRIMARY KEY (ID)
);
```

With some values inserted into the table :

```
INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (1, 'Ramesh', 32, 'Ahmedabad', 2000.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (2, 'Khilan', 25, 'Delhi', 1500.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (3, 'kaushik', 23, 'Kota', 2000.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (4, 'Chaitali', 25, 'Mumbai', 6500.00 );
```

```
INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (5, 'Hardik', 27, 'Bhopal', 8500.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (6, 'Komal', 22, 'MP', 4500.00 );
```

Write a program that displays a customer's name and salary from his ID, and in case this ID does not exist (example, ID = 8), catches a NO_DATA_FOUND exception by displaying an error message.

Your code should produce the following result :

```
No such customer !
PL/SQL procedure completed successfully
```

Exercise 0.2
Here, we're going to learn how to create and raise your own exceptions. Take the previous program and extend it by raising an exception in case the ID is invalid (lower than zero, for example).
use the IF conditional. Write a program that assigns to variable *a* value 100 (you can try several values), then checks whether *a* is inferior or superior to 1. In the first case, it displays that a is less than 20. In the second case, it displays that a is not less than 20. Finally, it displays the value of a.

Your code should produce the following result :

```
ID must be greater than zero !
PL/SQL procedure successfully completed.
```

1 – Let's mix SQL with PL/SQL

Here, we'll use the tables defined in annex A.1 and A.2.

Exercise 1
You'll notice that the *numpers* values from the PERSON table form a sequence from 1 to 21. Use a LOOP in which you'll place a request to copy/paste the couples *name/firstname* from the PERSON table to the CLIENT table.

Exercise 2
Write a program that gets the CLIENT who has the biggest primary key, and injects this client into the STAFF table.

Exercise 3
Open a 'current account' for every PERSON. Deposit an amount of cash that equals to *numpers* * 100.

Exercise 4
Open a 'supplement account' for every PERSON whose *numpers* is even. Deposit an amount of cash that equals to *numpers* * 100€. Transfer money from their 'current account' to their 'supplement account' so that only 500€ remain in the first one.

2 – Now, let's add exceptions to the mix !

We'll still use the tables defined in annex A.1 and A.2.

Exercise 1
First of all, you'll have to modify a little bit the code you previously wrote. Each time a SELECT … INTO is executed, you'll have to catch exceptions NO_DATA_FOUND and TOO_MANY_ROWS. Each time an INSERT INTO is executed, you'll catch exception DUP_VAL_ON_INDEX

Exercise 2
Make sure the important scripts from CLIENT tables are executed only once.

Exercise 3
Scripts that fill the OPERATION table won't work that way. Find error codes raised by these scripts, and catch them the most appropriate way.

Annex A.1 – The PERSON table

```
CREATE TABLE Person(
        numpers NUMBER PRIMARY KEY,
        name VARCHAR2(30) NOT NULL,
        firstname VARCHAR2(30),
        father REFERENCES Person(numpers),
        mother REFERENCES Person(numpers)
);

INSERT INTO Person VALUES (1, 'de Montmirail, dit le Hardi', 'Godefroy', NULL, NULL);
INSERT INTO Person VALUES (16, 'ET', NULL, NULL, NULL);
INSERT INTO Person VALUES (2, 'Nelson', 'Prince Rogers', 1, 16);
INSERT INTO Person VALUES (3, 'Bowie', 'David', NULL, NULL);
INSERT INTO Person VALUES (4, 'Mercury', 'Freddie', NULL, NULL);
INSERT INTO Person VALUES (13, 'Gore', 'Tipper', NULL, NULL);
INSERT INTO Person VALUES (5, 'George', 'Bart', 3, 4);
INSERT INTO Person VALUES (12, 'Cersei', 'Jamie', NULL, NULL);
INSERT INTO Person VALUES (6, 'Quatro', 'Suzi', NULL, NULL);
INSERT INTO Person VALUES (7, 'May', 'Brian', 4, 6);
INSERT INTO Person VALUES (8, 'Dijkstra', 'Edger', 2, 13);
INSERT INTO Person VALUES (9, 'Stark', 'Eddard', 4, 13);
INSERT INTO Person VALUES (10, 'Bach', 'Johann Sebastian', 9, 6);
INSERT INTO Person VALUES (17, 'Lannister', 'Cersei', NULL, NULL);
INSERT INTO Person VALUES (11, 'Lannister', 'Joffrey', 12, 17);
INSERT INTO Person VALUES (15, 'Socrates', NULL, NULL, NULL);
INSERT INTO Person VALUES (19, 'Stark', 'Catelyn', NULL, NULL);
INSERT INTO Person VALUES (20, 'Stark', 'Robb', 9, 19);
INSERT INTO Person VALUES (21, 'Lannister', 'Tyrion', 9, 17);
INSERT INTO Person VALUES (14, 'Snow', 'John', 9, 6);
INSERT INTO Person VALUES (18, 'Deacon', 'John', 7, 13);
```

Annex A.2 – The bank account database

```
DROP TABLE OPERATION;
DROP TABLE CLIENTACCOUNT;
DROP TABLE TYPECLA;
DROP TABLE TYPEOPERATION;
DROP TABLE STAFF;
DROP TABLE CLIENT;

CREATE TABLE CLIENT (
        numcli number,
        namecli varchar2(30),
        firstnamecli varchar2(30),
        address varchar2(60),
        tel varchar(10)
);

CREATE TABLE STAFF (
        numpers number,
        namepers varchar2(30),
        firstnamepers varchar2(30),
        manager number,
        salary number
);

CREATE TABLE TYPECLA (
        numtypecla number,
        nametypecla varchar2(30)
);

CREATE TABLE CLIENTACCOUNT (
        numcli number,
        numccl number,
        numtypecla number,
        dateccl date default sysdate not null,
        numpers number
);

CREATE TABLE TYPEOPERATION (
        numtypeoper number,
        nametypeoper varchar2(30)
);

CREATE TABLE OPERATION (
        numcli number,
        numccl number,
        numoper number,
        numtypeoper number,
        dateoper date default sysdate not null,
        amountoper number not null,
        libeloper varchar2(30)
);
```

```
ALTER TABLE CLIENT ADD (
        CONSTRAINT pk_client PRIMARY KEY (numcli),
        CONSTRAINT ck_telephone CHECK(LENGTH(tel)=10)
);

ALTER TABLE STAFF ADD (
        CONSTRAINT pk_staff PRIMARY KEY (numpers),
        CONSTRAINT ck_salary CHECK(SALARY >= 1254.28)
);

ALTER TABLE TYPECLA
ADD CONSTRAINT pk_typecla PRIMARY KEY (numtypecla);

ALTER TABLE TYPEOPERATION
ADD CONSTRAINT pk_typeoperation PRIMARY KEY (numtypeoper);

ALTER TABLE CLIENTACCOUNT ADD (
        CONSTRAINT pk_clientaccount PRIMARY KEY (numcli, numccl),
        CONSTRAINT fk_ccl_typecla FOREIGN KEY (numtypecla) REFERENCES
        TYPECLA (numtypecla),
        CONSTRAINT fk_ccl_client FOREIGN KEY (numcli) REFERENCES CLIENT
        (numcli),
        CONSTRAINT fk_ccl_staff FOREIGN KEY (numpers) REFERENCES     STAFF
(numpers)
);

ALTER TABLE OPERATION ADD (
        CONSTRAINT pk_operation PRIMARY KEY (numcli, numccl, numoper),
        CONSTRAINT fk_oper_ccl FOREIGN KEY (numcli, numoper) REFERENCES
        CLIENTACCOUNT (numcli, numccl),
        CONSTRAINT fk_oper_codeoper FOREIGN KEY (numtypeoper) REFERENCES
        typeoperation (numtypeoper),
        CONSTRAINT montant_operation CHECK(amountoper <> 0 AND amountoper >=
        -1000 AND amountoper <= 1000)
);

INSERT INTO TYPECLA VALUES ( (SELECT nvl(MAX(numtypecla), 0) + 1 FROM
TYPECLA ), 'Current Account');
INSERT INTO TYPECLA VALUES ( (SELECT nvl(MAX(numtypecla), 0) + 1 FROM
TYPECLA ), 'Supplement Account');
INSERT INTO TYPECLA VALUES ( (SELECT nvl(MAX(numtypecla), 0) + 1 FROM
TYPECLA ), 'PEL');
INSERT INTO TYPEOPERATION VALUES ( (SELECT nvl(MAX(numtypeoper), 0) + 1
FROM TYPEOPERATION ), 'cash deposit');
INSERT INTO TYPEOPERATION VALUES ( (SELECT nvl(MAX(numtypeoper), 0) + 1
FROM TYPEOPERATION ), 'punction');
INSERT INTO TYPEOPERATION VALUES ( (SELECT nvl(MAX(numtypeoper), 0) + 1
FROM TYPEOPERATION ), 'transfer'); INSERT INTO TYPEOPERATION VALUES
( (SELECT nvl(MAX(numtypeoper), 0) + 1 FROM TYPEOPERATION ), 'withdraw');
```