|  |
| --- |
| **COMP 3610 PL/SQL Exercises** |

**Objectives:**

* Design and implement a simple PL/SQL function, trigger, and stored procedure

# RESERVATION

R ID

START\_DATE

END\_DATE

C\_ID

V\_ID

TOTAL\_COST

# VEHICLE

V\_ID

VIN

V\_MAKE

COST\_PER\_DAY

# ARCHIVED\_RESERVATION

R ID

START\_DATE

END\_DATE

C\_ID

V\_ID

TOTAL\_COST

ADDED\_BY

ADDED\_DATE

# CLIENT

C\_ID

F\_NAME

L\_NAME

DOB

CITY

GENDER

**Problem 1 Age (Function)**

The vehicle reservation system should check the age of the person renting the car. The minimum age is 21 years. Create a function to calculate the age (in years) for a given C\_ID. The age is calculated as of today (SYSDATE on the server).

**Create a TRIGGER to prevent an addition of a reservation for a client who is not** yet 21. The age is calculated as of today (SYSDATE on the server). The vehicle reservation system should check the age of the person renting the car. The maximum age is 80 years. Create a TRIGGER to prevent an addition of a reservation for a client who is older than 80. The age is calculated as of today (SYSDATE on the server).

**Problem 2 Character-based Primary Key with a sequence number (TRIGGER)**

Client ids C\_ID are defined as a string of 10 characters. The first character is a letter ‘C’ followed by 9 characters, which are created from a sequence left padded by zeros. Create a trigger to add automatically a new primary key for an INSERT on the Client table.

**Problem 3 Archiving data (Stored Procedure)**

Create a stored procedure to archive the reservation data. The archived procedure is executed by the DBA on the first day of a new year. This procedure has no input parameters and displays message “total … # reservations archived.” The archiving procedure adds the old (previous year) reservations to the ARCHIVED\_RESERVATION table and deletes the old reservations from the RESERVATION table.

CREATE OR REPLACE FUNCTION calc\_age (p\_id IN VARCHAR)

RETURN number IS

v\_age number(3) := 0;

BEGIN

SELECT TRUNC(MONTHS\_BETWEEN(SYSDATE, dob)/12) INTO v\_age

FROM CLIENT

WHERE c\_id = p\_id;

RETURN v\_age;

END;

CREATE OR REPLACE TRIGGER age\_check\_BIR

BEFORE INSERT OR UPDATE

ON reservation

FOR EACH ROW

DECLARE

v\_age number(3) := 0;

BEGIN

v\_age number(3) := calc\_age(:NEW.c\_id);

IF (v\_age < 21 OR v\_age > 80) THEN

RAISE\_APPLICATION\_ERROR(-­20001, 'Client must be 21-80 years old'); END IF;

END;

CREATE OR REPLACE TRIGGER id\_generator\_BIR

BEFORE INSERT

ON CLIENT

FOR EACH ROW

BEGIN

:NEW.c\_id := 'C' || LPAD(seq\_c\_id.nextval),9,'0');

END;

CREATE OR REPLACE PROCEDURE ARCHIVE\_RES ()

IS

V\_count NUMBER(6): = 0;

BEGIN

INSERT INTO ARCHIVED\_RESERVATION( )

SELECT

FROM RESERVATION

WHERE EXTRACT(YEAR FROM END\_DATE) < EXTRACT(YEAR FROM SYSDATE);

-- SQL%ROWCOUNT - returns number of rows modified by the DML statement

V\_count := SQL%ROWCOUNT;

--

DBMS\_OUTPUT.PUT\_LINE (V\_count);

END;