

# AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Course No: CSE4126

Course Title: Distributed Database Systems Lab

Spring 2021 | Lab Final Examination | Set B | Marks 30 | Time: 60 Minutes

1. Answer the following questions. [10]

- a. Define trigger. How many types are there? Explain the differences between them. [3]
- b. What is the structure of the DUAL table in SQLplus? Write down 3 examples of how it can be used. [3]
- c. Explain when the following predefined exceptions are raised: [3]
  - i. ACCESS\_INTO\_NULL
  - ii. VALUE\_ERROR
  - iii. NOT\_LOGGED\_ON
- d. What is the difference between IN parameter and IN-OUT parameter? [1]

2. Assume that, we have a table named "BOOK" that has four attributes :– [5]

- *bid* (type: int),
- *bookName* (varchar2),
- *price* (int)
- *copies* (int)

Also assume that, there are already three rows inserted in "BOOK" table as follows –

bid	bookName	price	copies
1	War and Peace	450	10
2	Macbeth	250	5
3	Harry Potter	350	20

Create five triggers (trig1, trig2, trig3, trig4, trig5) so that when we run the following queries they will work as expected:

**insert into book values (4,'A brief history of time',550,20);**

This will trigger trig4, trig5. Only trig4 will be triggered after insertion.

**update book set copies=copies-1 where bookname='Macbeth';**

This will trigger trig2, trig4.

**delete from book where price<300;**

This will trigger trig2, trig3. Only trig3 will be triggered before deletion.

**delete from book where bid=1;**

This will trigger trig2.

**update book set price=300 where bookname='Harry Potter';**

This will trigger trig1, trig2, trig4. Only trig1 will be triggered before insertion.

*Note that, inside triggers you have to print the trigger name. For example – when the trigger trig1 is executed it will print “trig1 activated”, for trig2 trigger - print “trig2 activated” and so on.*

3. Consider that you have two database tables named “Person” and “Details”, which belong to a Clinic Management System. Suppose, you have an .sql file which shows the description of the table as follows:

[10]

```
CREATE TABLE Person (pID int, pName VARCHAR2(50), dateOfBirth date,
height_in_meter number(3,2), weight_in_kg int, PRIMARY KEY (pID));
```

```
CREATE TABLE Details (pID int, BMI number(4,2), age number(2),
FOREIGN KEY (pID) REFERENCES Person(pid));
```

```
insert into Person values (1,'Ikhtiar','10-Jan-1980',1.54,55);
insert into Person values (2,'Uddin','10-Feb-1980',1.55,60);
insert into Person values (3,'Mohammad','10-Mar-1980',1.6,65);
insert into Person values (4,'Bakhtiyar','10-Apr-1980',1.52,55);
insert into Person values (5,'Khilji','10-May-1980',1.44,55);
```

Notice that no data has been inserted into the **Details** table yet. This is because all the data for this table needs to be derived/calculated from the **Person** table. Remember that, BMI(Body Mass Index) is found through dividing weight in kilogram by the square of height in meter. Also remember that, simply subtracting the date of birth from current date gives your age in days.

**Your tasks are as follows, to be written in PL/SQL:**

- You need to write a **function**, called “AgeCalculator”, that calculates the age of the Person. It takes in the **date of birth** as input from the main block and returns the **age**.
- You need to write a **procedure**, called “BMICalculator”, that calculates the BMI of a Person. It takes in **pID** and **age** as input from the calling block, calculates the BMI using a formula, and inserts a row into the **Details** table, with the required data.
- pID must be taken as user input in the main block.
- The function and the procedure need to be in a **package** called “Patients”

4. Suppose, you have a table named CURRENT\_STUDENTS and another named ALUMNI which contains the data of all students(current and graduated respectively) in a particular university. Now, write a **procedure** that updates the study year of the students when it is run at the starting of each year, i.e. when they are promoted to the upper year. Assume that the values of *currentStudyYear* can be within 1 to 4. When you find a 4th year student, add them to the ALUMNI table and delete them from the CURRENT\_STUDENTS table. Assume that all the students pass every year, so they all get promoted. The tables look as follows:

[5]

**CURRENT\_STUDENTS**

<b>student_id (int)</b>	<b>name (varchar2(50))</b>	<b>currentStudyYear (int)</b>
1	Rafiq	2
1501	Shafiq	4

**ALUMNI**

<b>student_id (int)</b>	<b>name (varchar2(50))</b>
1	Rafiq
1501	Shafiq