IutBook

1. For today's task we will be implementing different parts of social media called IutBook.
2. Like the previous lab you will get a helper file where you can find all the necessary codes for the tables and also your solution should work with the values given in the helper file.
3. In this lab our main focus will be on how to implement triggers. **You can also use explicit cursors for your ease.**
4. **Showing the defined error codes are optional** (you can just print some messages using *dbms_output.put_line()* but showing the defined error codes will get you bonus marks).

Task 1

Make a table called UserInfo (take help from the helper file). Now you need to associate a trigger with the insert command of UserInfo so that if anyone inserts a row with only name then it will automatically allocate userId.
Let's say we do this,
INSERT INTO UserInfo (name) VALUES ('atiq');
INSERT INTO UserInfo (name) VALUES ('rahman');
Then UserInfo will look like,

userId	name
1	atiq
2	rahman

userId can be from 1 to 999999.
Hint : use *create sequence* command.

Task 2

You need to implement a procedure called **makeFriends** which takes two user ids as parameters and inserts these two user ids to a table called **FrndList**. Let's say we execute makeFriends with 1 and 3, then we need to insert two rows into FrndList with 1,3 as userId1,userId2 and 3,1 as userId1,userId2. Because if 1 is friend of 3 then 3 is also friend of 1. After the insertions FrndList looks like this,

userId1	userId2
1	3
3	1

The task is to associate a trigger with the insertion command of FrndList which will show an error code ORA-01718 (Hint : use PRAGMA EXCEPTION_INIT) if the user ids are invalid (the user ids are not present in UserInfo table) else insert those valid ids into FrndList.

Task 3

Make a table called **Message** (take help from the helper file). Associate a trigger with the insertion command of Message which will show an error code ORA-00123 if sender and receiver are not friends else do the insertion.

Task 4

Implement a procedure called **unfriend** which will take two user ids as parameters and delete two rows from FrndList (there are two rows for users 1 & 3 like 1,3 as userId1,userId2 and 3,1as userId1,userId2). Associate a trigger with the delete command which will remove the messages of those two friends from the message table. If the user ids are not friends then show an error code ORA-00009.

Task 5

Make two new tables called **Post** and **PostNotification** (take help from the helper file). Associate two triggers with the insertion command of Post, first trigger will delete all the previous notifications of the user (who is giving the post) from the PostNotification table and second trigger will insert new values to PostNotification. Let's say we have PostNotification like,

userId	frndId	postId
1	2	3

Then if user 1 is trying to insert value to Post like,
Insert into Post (postId,userId,text) values (4,1,'I need friends.');
After the above insert the PostNotification will look like,

userId	frndId	postId
1	2	4

So we are removing notifications of previous posts and inserting notifications with the new post id. And we use FrndList to get the frndIds.