CPS610 - Assignment 1 (11 + 2 bonus marks)
Labs 1 / 2 / 3: Replicate database

Lab 1: Form your group and notify your TA (1 mark). Get started with the tasks for Lab 2 and Lab 3.

Lab 2: Database Replication Activity: Achieving Data Consistency Across Two Oracle Instances (5 marks + 1 bonus)

Objective: This activity will teach you how to implement a simple database replication mechanism, ensuring consistency between two Oracle database instances. You will use **SQL Developer** and two Oracle instances (already installed in our school servers) to replicate data and resolve potential inconsistencies.

Implementing and Testing Database Replication

Scenario: You have two database instances, Oracle 11 (DB1) and Oracle 12 (DB2). Both instances should maintain a table called EMPLOYEES, which stores employee information. The table's design (which columns, datatypes, etc) is up to you as long as they're the same in both instances. Your task is to implement a basic replication mechanism such that any changes made to the EMPLOYEES table in DB1 are reflected in DB2 and vice versa.

Steps:

1. Setup (Preparation):

- Download and run SQL Developer
- Make sure you can access the two Oracle Instances in our school servers. Ensure you can connect to both Oracle instances (which we are calling `DB1` and `DB2`). You have to be connected to the department's VPN to be able to connect from your laptop. Details about how to connect can be found in our Labs D2L folder, file: Oracle_Connection_Instructions.pdf.
- Create your `EMPLOYEES` Table in Both Instances

2. Task 1: Insert Replication

- Insert a row into the `EMPLOYEES` table in `DB1`.
- Write a script to replicate this data into `DB2`. Tip: use a combination of `DB_LINK` and any script (For example, `MERGE`).

- Create a Database Link in DB1 to connect to DB2. Here is an example of a code to create a DBLink. You need to replace username and password with your oracle username and password for Oracle 12.

```
CREATE DATABASE LINK db2
CONNECT TO username IDENTIFIED BY password
USING
'(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=oracle12c.scs.ryerson.
ca)(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=orcl12c)))';
```

- Replicate the Data. Here, you need a script to insert into EMPLOYEE in DB2 when something is inserted into EMPLOYEE in DB1. You also need to update EMPLOYEE in DB2 when something is updated into EMPLOYEE in DB1. You can achieve this by using MERGE in Oracle.

3. Task 2: Update and Delete Replication

- Update an Employee Record in `DB1`
- Replicate the Update to `DB2`. Use the same `MERGE` statement as in Task 1.
- Delete a Record in `DB1`
- Replicate the Deletion to `DB2`.Use a query to delete records in `DB2` that no longer exist in `DB1`.

4. Task 3 (Bonus): Automate Replication with Triggers

Create a TRIGGER in DB1 to replicate changes automatically. You need to complete this example code where you see the `........` for your trigger to be created successfully. Also, you need to replace the name of the DBLink I am using `db2` with the name you gave to your own DBLink to Oracle 12. You also need to replace the column names with the columns of the EMPLOYEE table you created.

Example:

```
Unset
CREATE OR REPLACE TRIGGER trg_replicate_to_db2
AFTER INSERT OR UPDATE OR DELETE ON EMPLOYEES
```

```
FOR EACH ROW
BEGIN
    -- INSERT and UPDATE replication
    IF INSERTING OR UPDATING THEN
        MERGE INTO EMPLOYEES@db2 d
        USING (SELECT :NEW.EMP_ID AS EMP_ID, :NEW.NAME AS NAME,
:NEW.POSITION AS POSITION, :NEW.SALARY AS SALARY FROM DUAL) s
        ON (d.EMP_ID = s.EMP_ID)
        WHEN MATCHED THEN
            ........
        WHEN NOT MATCHED THEN
            ........
    END IF;

    -- DELETE replication
    IF DELETING THEN
       ........
    END IF;
END;
/
```

LAB3: Implementing a Stored Procedure for Distributed Database Management. (5 marks + 1 bonus)

Objective: To build on the skills learned in database replication, you will now implement a stored procedure that automates data synchronization between two Oracle instances. You will gain experience in writing, testing, and executing stored procedures and further explore concepts related to distributed database management.

Distributed Synchronization Using Stored Procedures

Scenario: After successfully implementing replication, your team is tasked with creating a robust mechanism to synchronize changes between `DB1` and `DB2` in real-time. However, instead of relying on manual scripts or triggers, you will design and implement a stored procedure that handles data consistency between the two databases efficiently.

Steps:

1. Setup (Preparation)

- Use the same two Oracle instances (`DB1` and `DB2`) with the `EMPLOYEES` table created in the previous assignment.
- Ensure the database link between `DB1` and `DB2` is operational.

2. Task 1: Create a Stored Procedure for Synchronization

Requirements for the Stored Procedure:

1. The stored procedure should:
   - Check for new, updated, or deleted rows in `EMPLOYEES` in `DB1`.
   - Synchronize the changes to `DB2`.
2. The synchronization process should:
   - Insert new rows from `DB1` into `DB2`.
   - Update rows in `DB2` that have been modified in `DB1`.
   - Delete rows in `DB2` that no longer exist in `DB1`.

Example:

```
Unset
CREATE OR REPLACE PROCEDURE sync_employees_to_db2 AS
BEGIN
   ...
END;
/
```

Test the stored procedure: Insert, update, and delete rows in DB1 and call the stored procedure:

```
Unset
BEGIN
    sync_employees_to_db2;
END;
```

3. Task 2: Add Logging for Synchronization

Enhance the stored procedure to log changes:

- Create a table SYNC_LOG in DB1 to track synchronization operations. This table should have the following attributes: LOG_ID, SYNC_TIMESTAMP, OPERATION_TYPE, EMP_ID.
- Now, update the stored procedure to log each operation, by inserting a new line into SYNC_LOG of OPERATION_TYPE = 'INSERT' each time an insert into EMPLOYEES happens. Do the same for 'UPDATE' and 'DELETE' operations: you will **INSERT** a new line into SYNC_LOG each time an update or delete operation is performed on the EMPLOYEES table.

4. Task 3: Automate the Synchronization (Bonus)

- Schedule the stored procedure to run every 5 minutes using Oracle's DBMS_SCHEDULER.

Deliverables: To be presented during your lab 3 session (by A1 deadline). Everything should be READY and presented to the TA during your LAB3 session.

1. The SQL Scripts for Lab2:
    - Table creation and the table object created in both instances. 1 marks.
    - Data insertion, updating, and deletion. 1 marks.
    - Replication using script. 1 marks.

    And explanations to:

    - How replication was achieved. 1 marks.
    - Challenges faced and how they were addressed. 1 marks.

    - **Bonus Lab2(Optional):** Demonstrate triggers to automate replication. 1 marks.

2. The SQL scripts for Lab3:

    - Creating the stored procedure. 1 marks.
    - Enhancing it with logging. 1 marks.
    - Test cases. 1 marks.

    And explanations to:

    - How the procedure was implemented and tested. 1 marks.
    - How logs help monitor the synchronization process and challenges faced and how they were overcome. 1 marks.

    - **Bonus Lab3 (Optional)** Automating using `DBMS_SCHEDULER`. 1 marks.

Assignment 1 grade out of: 1 (lab1) + 5 (lab2) + 5 (lab3) + 2 Bonus marks.

Submission Guidelines:

- **You must submit your assignment before attending the presentation Lab.**
- Submit the files in the corresponding Assignment folder on the D2L link:
    - Assessments > Assignments > Assignment 1

- **Each group must submit a single compressed `.zip` file, named using their group number:**
  Filename Format: `CPS610_A1_GroupX.zip` (Replace **X** with your group number)
- **In D2L submission comments, include:**
  - Student Number - Full names of all group members

Submission Structure (Inside the .zip File)

Each `.zip` file should have the following organized structure:

CPS610_A1_GroupX.zip

- **README.pdf** (A brief summary of the implementation, team members, and any special instructions, such as file names inside zip folder)
- **Lab_Report.pdf** (Group formation details, initial steps, Explanation of the replication process, challenges faced, and solutions, Explanation of the stored procedure, logging, testing, and automation)
- **Lab2/**
  - **lab2_schema.sql** (SQL scripts for table creation in DB1 & DB2)
  - **lab2_replication.sql** (SQL scripts for insert/update/delete replication using DB_LINK and MERGE)
  - **lab2_triggers.sql** (Bonus: SQL scripts for triggers to automate replication)
- **Lab3/**
  - **lab3_stored_procedure.sql** (SQL script for stored procedure implementation)
  - **lab3_logging.sql** (SQL script for logging mechanism)
  - **lab3_scheduler.sql** (Bonus: SQL script for automating with DBMS_SCHEDULER)
  - **lab3_test_cases.sql** (SQL scripts for testing the stored procedure)