8-Managing Data and Concurrency

● Manage Data Using DML

In this exercise, you will demonstrate transaction isolation and control. Use **two** SQL*Plus sessions, each connected as user SYSTEM Run the commands in the steps that follow in the two session s in the correct order.

Step	First session	Second session
1	create table t1 as select * from all_users;	
2	select count(*) from t1;	select count(*) from t1;
Results are the same in both sessions.		
3	delete from t1;	
4	select count(*) from t1;	select count(*) from t1;
Results differ because transaction isolation conceals the changes.		
5	rollback;	
6	select count(*) from t1;	select count(*) from t1;
Results are the same in both sessions.		
7	delete from t1;	
8	select count(*) from t1;	select count(*) from t1;
9	create view v1 as select * from t1;	
10	select count(*) from t1;	select count(*) from t1;
11	rollback;	
12	select count(*) from t1;	select count(*) from t1;
Oh dear! The DDL statement		

committed the DELETE, so it can't be rolled back.		
13	drop view v1;	
14	drop table t1;	

● Create PL/SQL Objects:

In this exercise, you will use SQL*PLUS to create PL/SQL objects, and execute them.

1. Connect as SYS or SYSDBA
2. Create or replace **a function** that take a number as input and print "even" or "odd"
   ```
   CREATE OR REPLACE FUNCTION function_name (arg_name type)
   RETURN return_att_type AS
   BEGIN
   // function body
   END function_name;
   /
   ```
3. Create or replace **a procedure** that take an n integer value and insert numbers to n in integers table
   ```
   CREATE OR REPLACE PROCEDURE procedure_name (arg_name type)
   AS
   BEGIN
   // procedure body
   END procedure_name;
   /
   ```
4. Create **a package** that group the previous function and procedure
   ```
   CREATE OR REPLACE PACKAGE package_name
   AS
   // backage body
   END package_name;
   /
   ```
5. Execute the package

● Detect and Resolve Lock Contention

In this exercise, you will first use SQL*Plus to cause a problem, and

detect and solve them

1. Using SQL*Plus, connect to your database in two sessions as user SYSTEM.
2. In your first session, lock all the rows in the INTEGERS table, which was created in Exercise **CREATE PL/SQL Objects**:
   select * from integers for update;
3. In your second session, attempt to update a row. The session will hang:
   update integers set c2='odder' where c1=1;
4.