Thummala sriramcharan
G02497386

Create the following table:

```
create table tenants
(
   tenant_ID int primary key,
   tenant_name varchar(20),
   Age int,
   income decimal(9,2),
   unit_number varchar(10),
   monthly_rent number
);
```

Task#1:: **5 points/question**

- Write a procedure 'display_tenant_info' that displays tenant-name, income, and their unit-number having monthly-rent > value, where value is passed as an argument to the procedure.
- Use bulk collect mechanism, and demonstrate bulk collect into nested-table collection using the following three approaches. Write separate procedures for each approach.

1. SELECT column(s) BULK COLLECT INTO collection(s) remaining SQL-Query

```
      CREATE OR REPLACE PROCEDURE
display_tenant_info_approach1(value NUMBER) IS
  TYPE TenantInfoTable IS TABLE OF VARCHAR2(1000); -- Adjust
the data type as needed
```

```
  tenant_names TenantInfoTable;
  tenant_incomes TenantInfoTable;
  tenant_unit_numbers TenantInfoTable;

BEGIN
  SELECT tenant_name, income, unit_number
  BULK COLLECT INTO tenant_names, tenant_incomes,
tenant_unit_numbers
  FROM tenants
  WHERE monthly_rent > value;

  -- Display the results
  FOR i IN 1..tenant_names.COUNT LOOP
    DBMS_OUTPUT.PUT_LINE('Tenant Name: ' || tenant_names(i));
    DBMS_OUTPUT.PUT_LINE('Income: ' || tenant_incomes(i));
    DBMS_OUTPUT.PUT_LINE('Unit Number: ' ||
tenant_unit_numbers(i));
  END LOOP;
END;
```

2. FETCH cursor BULK COLLECT INTO collection(s)

```
      CREATE OR REPLACE PROCEDURE
display_tenant_info_approach2(value NUMBER) IS
```

```
  TYPE TenantInfoTable IS TABLE OF VARCHAR2(2000); -- Adjust the data
type as needed
  tenant_names TenantInfoTable;
  tenant_incomes TenantInfoTable;
  tenant_unit_numbers TenantInfoTable;

  CURSOR c is
    SELECT tenant_name, income, unit_number
    FROM tenants
    WHERE monthly_rent > value;
BEGIN
  OPEN c;
  FETCH c BULK COLLECT INTO tenant_names, tenant_incomes,
tenant_unit_numbers;

  -- Display the results
  FOR i IN 1..tenant_names.COUNT LOOP
    DBMS_OUTPUT.PUT_LINE('Tenant Name: ' || tenant_names(i));
    DBMS_OUTPUT.PUT_LINE('Income: ' || tenant_incomes(i));
    DBMS_OUTPUT.PUT_LINE('Unit Number: ' || tenant_unit_numbers(i));
  END LOOP;

  CLOSE c;
END;
```

3. EXECUTE IMMEDIATE SQL-Query BULK COLLECT INTO collection(s)

```
CREATE OR REPLACE PROCEDURE
display_tenant_info_approach3(new_value NUMBER) IS
  TYPE TenantInfoTable IS TABLE OF VARCHAR2(1000); -- Adjust the data
type as needed
  new_tenant_names TenantInfoTable;
  new_tenant_incomes TenantInfoTable;
  new_tenant_unit_numbers TenantInfoTable;

BEGIN
  EXECUTE IMMEDIATE 'SELECT tenant_name, income, unit_number FROM
your_table WHERE monthly_rent > :new_value'
  BULK COLLECT INTO new_tenant_names, new_tenant_incomes,
new_tenant_unit_numbers
  USING new_value;

  -- Display the results
  FOR i IN 1..new_tenant_names.COUNT LOOP
    DBMS_OUTPUT.PUT_LINE('Tenant Name: ' || new_tenant_names(i));
    DBMS_OUTPUT.PUT_LINE('Income: ' || new_tenant_incomes(i));
    DBMS_OUTPUT.PUT_LINE('Unit Number: ' || new_tenant_unit_numbers(i));
```

```
   END LOOP;
END;
```

Task#2:
5 points/ question

- Write PL/SQL procedure 'update_rent' that updates the monthly-rent of tenants. The procedure should take income and percent-increase as input, and update the monthly-rent by the given percentage of only those tenants who are earning more than the given income-value.
- For example, update (100000, 0.05); should update rents of tenants by 5% who earn more than 0.1 million.

- Use the following three approaches and write separate procedures for each approach.

1. Technique where there is a *context switch for each update* between PL/SQL and SQL.

```
CREATE OR REPLACE PROCEDURE update_rent_approach1q(
   income_threshold NUMBER,
   percent_increase NUMBER
) IS
BEGIN
   FOR rec IN (SELECT tenant_id, monthly_rent, income FROM tenants
WHERE income > income_threshold) LOOP
      rec.monthly_rent := rec.monthly_rent + (rec.monthly_rent *
percent_increase);
      UPDATE tenants
      SET monthly_rent = rec.monthly_rent
      WHERE tenant_id = rec.tenant_id;
   END LOOP;
END;
```

2. Use a ***single SQL statement*** to perform the update.

```
CREATE OR REPLACE PROCEDURE update_rent_approach2(
    income_threshold NUMBER,
    percent_increase NUMBER
) IS
BEGIN
    UPDATE tenants
    SET monthly_rent = monthly_rent + (monthly_rent * percent_increase)
    WHERE income > income_threshold;
END;
```

3. perform the update using *bulk-collect and forall*.

```
  CREATE OR REPLACE PROCEDURE update_rent_approach3(
  income_threshold NUMBER,
  percent_increase NUMBER
) IS
  TYPE TenantInfoTable IS TABLE OF tenants%ROWTYPE;
  tenant_data TenantInfoTable;
BEGIN
  -- Bulk collect tenants into a collection
  SELECT *
  BULK COLLECT INTO tenant_data
  FROM tenants
  WHERE income > income_threshold;

  -- Update the rents in the collection
  FOR i IN 1..tenant_data.COUNT LOOP
    tenant_data(i).monthly_rent := tenant_data(i).monthly_rent + 
(tenant_data(i).monthly_rent * percent_increase);
  END LOOP;

  -- Bulk update using FORALL
  FORALL i IN 1..tenant_data.COUNT
    UPDATE tenants
    SET monthly_rent = tenant_data(i).monthly_rent
    WHERE tenant_id = tenant_data(i).tenant_id;
END;
```

Submission: copy your PL/SQL script-code to a doc/pdf file and submit it through the link made available on blackboard.