345537708 Samuel Tapiro - 133467 Lior Tordjman - 346012065 Daniel Elbaz

Step C

1. Introduction:

This report documents the work completed for Step C of the Database Project. This step involves writing and testing PL/SQL programs on our database tables. The programs include functions and procedures, which are essential for performing specific operations on the database. The main programs demonstrate the use of these functions and procedures, ensuring they work as intended.

2. Functions and procedures:

The calculate_total_sales function calculates the total sales amount for a given customer ID by iterating through each sale record and summing the total amount.

```
CREATE OR REPLACE FUNCTION calculate_total_sales(customer_id IN NUMBER)
RETURN NUMBER IS
    total_sales NUMBER := 0;
    CURSOR sales_cursor IS
        SELECT TotalAmount FROM Sales WHERE CustomerID = customer_id;
BEGIN
    -- Loop through each sale to calculate the total amount
    FOR sale IN sales_cursor LOOP
        total_sales := total_sales + sale.TotalAmount;
    END LOOP;
    RETURN total_sales;
EXCEPTION
    WHEN OTHERS THEN
        -- Raise error if there is an issue
        RAISE_APPLICATION_ERROR(-20002, 'An error occurred: ' || SQLERRM);
END calculate_total_sales;
/
```

The insert_new_sale procedure inserts a new sale record into the Sales table. It takes the sale ID, sale date, total amount, customer ID, and employee ID as parameters.

```
CREATE OR REPLACE PROCEDURE insert_new_sale(sale_id IN NUMBER, sale_date IN
DATE, total_amount IN NUMBER, customer_id IN NUMBER, employee_id IN NUMBER)
IS
BEGIN
    -- Insert a new sale record into the Sales table
    INSERT INTO Sales (SaleID, DateOfSale, TotalAmount, CustomerID,
EmployeeID)
    VALUES (sale_id, sale_date, total_amount, customer_id, employee_id);
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN
        -- Raise error if there is an issue inserting the new sale
        RAISE_APPLICATION_ERROR(-20004, 'Error inserting new sale: ' ||
SQLERRM);
```

```
END insert_new_sale;
/
```

The get_product_inventory function returns a SYS_REFCURSOR that points to the inventory records for a specific product ID.

```
CREATE OR REPLACE FUNCTION get_product_inventory(product_id IN NUMBER)
RETURN SYS_REFCURSOR IS
    inventory_cursor SYS_REFCURSOR;
BEGIN
    -- Open a ref cursor to fetch the inventory records for a specific
product
    OPEN inventory_cursor FOR
    SELECT * FROM Inventory WHERE ProductID = product_id;

    -- Return the ref cursor to the caller
    RETURN inventory_cursor;
EXCEPTION
    WHEN OTHERS THEN
        -- Raise an application error if any other exception occurs
        RAISE_APPLICATION_ERROR(-20001, 'An error occurred: ' || SQLERRM);
END get_product_inventory;
/
```

The update_inventory_stock procedure updates the inventory quantity for a specific product ID.

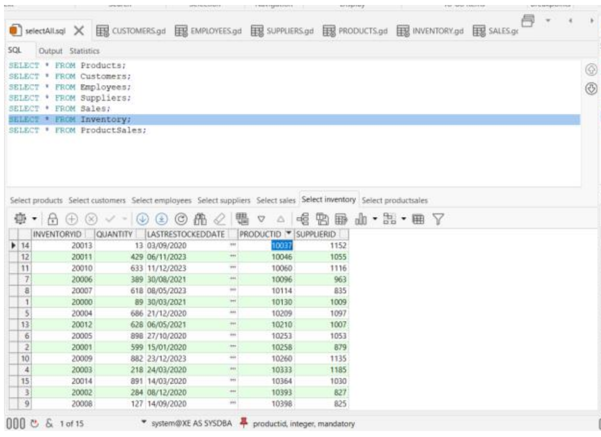
```
CREATE OR REPLACE PROCEDURE update_inventory_stock(product_id IN NUMBER,
new_quantity IN NUMBER) IS
BEGIN
    -- Update the inventory quantity for a specific product
    UPDATE Inventory
    SET Quantity = new_quantity
    WHERE ProductID = product_id;
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN
        -- Raise error if there is an issue updating the inventory stock
        RAISE_APPLICATION_ERROR(-20003, 'Error updating inventory stock: '
|| SQLERRM);
END update_inventory_stock;
/
```

3. Main programs:

The main1.sql script calls the update_inventory_stock procedure to update the inventory quantity for a specific product. It then calls the get_product_inventory function to retrieve and display the updated inventory details.

```
DECLARE
    v_product_id NUMBER := 10037;
    v_new_quantity NUMBER := 50;
    v_inventory_cursor SYS_REFCURSOR;
    v_inventory_record Inventory%ROWTYPE;
BEGIN
    -- Call the procedure update_inventory_stock
    update_inventory_stock(v_product_id, v_new_quantity);

    -- Call the function get_product_inventory
    v_inventory_cursor := get_product_inventory(v_product_id);

    LOOP
        FETCH v_inventory_cursor INTO v_inventory_record;
        EXIT WHEN v_inventory_cursor%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Product ID: ' || v_inventory_record.ProductID
||
                             ', Quantity: ' || v_inventory_record.Quantity
||
                             ', Last Restocked Date: ' ||
v_inventory_record.LastRestockedDate);
    END LOOP;

    CLOSE v_inventory_cursor;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

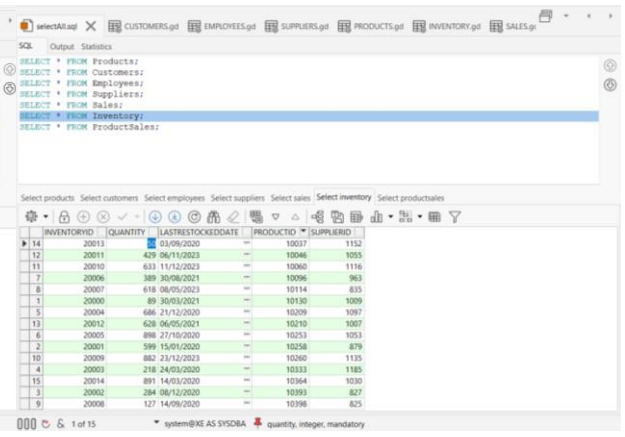
Screenshots



```
SELECT * FROM Products;
SELECT * FROM Customers;
SELECT * FROM Employees;
SELECT * FROM Suppliers;
SELECT * FROM Sales;
SELECT * FROM Inventory;
SELECT * FROM ProductSales;
```

	INVENTORYID	QUANTITY	LASTRESTOCKEDDATE	PRODUCTID	SUPPLIERID
14	20013	13	03/09/2020	10037	1152
12	20011	429	06/11/2023	10046	1055
11	20010	633	11/12/2023	10060	1116
7	20006	389	30/08/2021	10096	963
8	20007	618	08/05/2023	10114	835
1	20000	89	30/03/2021	10130	1009
5	20004	686	21/12/2020	10209	1097
13	20012	628	06/05/2021	10210	1007
6	20005	898	27/10/2020	10253	1053
2	20001	599	15/01/2020	10258	879
10	20009	882	23/12/2023	10260	1135
4	20003	218	24/03/2020	10333	1185
15	20014	891	14/03/2020	10364	1030
3	20002	284	08/12/2020	10393	827
9	20008	127	14/09/2020	10398	825

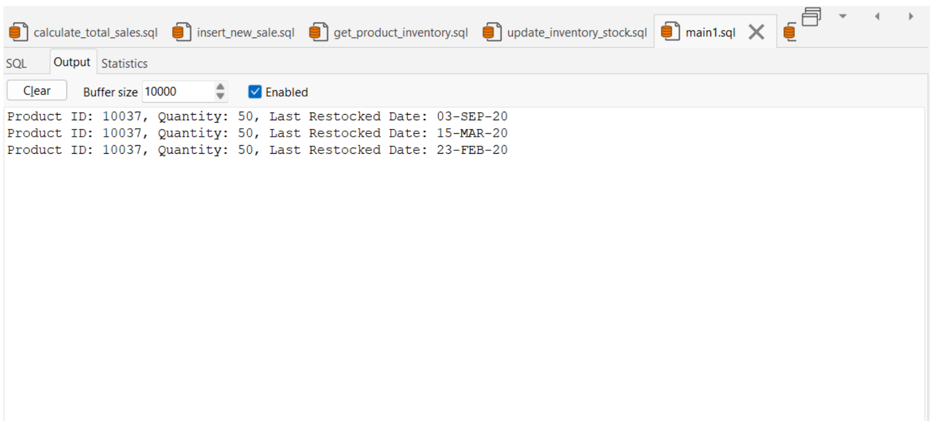
1 of 15 system@XE AS SYSDBA productid, integer, mandatory



```
SELECT * FROM Products;
SELECT * FROM Customers;
SELECT * FROM Employees;
SELECT * FROM Suppliers;
SELECT * FROM Sales;
SELECT * FROM Inventory;
SELECT * FROM ProductSales;
```

	INVENTORYID	QUANTITY	LASTRESTOCKEDDATE	PRODUCTID	SUPPLIERID
14	20013	50	03/09/2020	10037	1152
12	20011	429	06/11/2023	10046	1055
11	20010	633	11/12/2023	10060	1116
7	20006	389	30/08/2021	10096	963
8	20007	618	08/05/2023	10114	835
1	20000	89	30/03/2021	10130	1009
5	20004	686	21/12/2020	10209	1097
13	20012	628	06/05/2021	10210	1007
6	20005	898	27/10/2020	10253	1053
2	20001	599	15/01/2020	10258	879
10	20009	882	23/12/2023	10260	1135
4	20003	218	24/03/2020	10333	1185
15	20014	891	14/03/2020	10364	1030
3	20002	284	08/12/2020	10393	827
9	20008	127	14/09/2020	10398	825

1 of 15 system@XE AS SYSDBA quantity, integer, mandatory


calculate_total_sales.sql
insert_new_sale.sql
get_product_inventory.sql
update_inventory_stock.sql
main1.sql
SQL
Output
Statistics
Clear

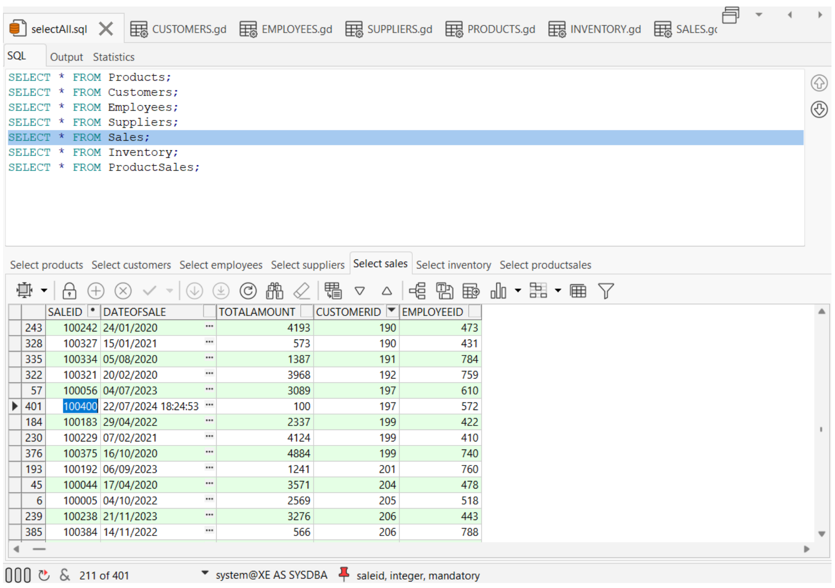
Buffer size 10000 ✓ Enabled

```
Product ID: 10037, Quantity: 50, Last Restocked Date: 03-SEP-20
Product ID: 10037, Quantity: 50, Last Restocked Date: 15-MAR-20
Product ID: 10037, Quantity: 50, Last Restocked Date: 23-FEB-20
```

The main2.sql script calls the insert_new_sale procedure to insert a new sale record. It then calls the calculate_total_sales function to calculate and display the total sales for a specific customer.

```
DECLARE
    v_sale_id NUMBER := 100400;
    v_sale_date DATE := SYSDATE;
    v_total_amount NUMBER := 100;
    v_customer_id NUMBER := 197;
    v_employee_id NUMBER := 572;
    v_total_sales NUMBER;
BEGIN
    -- Call the procedure insert_new_sale
    insert_new_sale(v_sale_id, v_sale_date, v_total_amount, v_customer_id,
v_employee_id);

    -- Call the function calculate_total_sales
    v_total_sales := calculate_total_sales(v_customer_id);

    DBMS_OUTPUT.PUT_LINE('Total sales for customer ' || v_customer_id || '
is ' || v_total_sales);
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

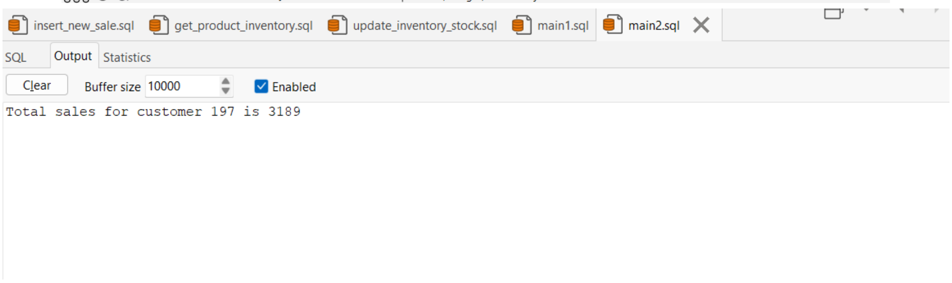
Screenshots



```
SELECT * FROM Products;
SELECT * FROM Customers;
SELECT * FROM Employees;
SELECT * FROM Suppliers;
SELECT * FROM Sales;
SELECT * FROM Inventory;
SELECT * FROM ProductSales;
```

	SALEID	DATEOFSALE	TOTALAMOUNT	CUSTOMERID	EMPLOYEEID
243	100242	24/01/2020	4193	190	473
328	100327	15/01/2021	573	190	431
335	100334	05/08/2020	1387	191	784
322	100321	20/02/2020	3968	192	759
57	100056	04/07/2023	3089	197	610
401	100400	22/07/2024 18:24:53	100	197	572
184	100183	29/04/2022	2337	199	422
230	100229	07/02/2021	4124	199	410
376	100375	16/10/2020	4884	199	740
193	100192	06/09/2023	1241	201	760
45	100044	17/04/2020	3571	204	478
6	100005	04/10/2022	2569	205	518
239	100238	21/11/2023	3276	206	443
385	100384	14/11/2022	566	206	788

211 of 401   system@XE AS SYSDBA   saleid, integer, mandatory



```
Total sales for customer 197 is 3189
```