דוח הפרויקט שלב ג

יובל יפת 213938905

ג׳רמי תורג׳מן 1828264

Main Program

room_cleaning_management

This program combines the room efficiency function and task assignment procedure which will be explained later. It:

- Calls get_room_cleaning_efficiency to display room cleaning statistics
- Calls assign_cleaning_tasks to assign new cleaning tasks
- Provides an overview of room cleaning status and task assignment

CREATE OR REPLACE FUNCTION ... | CREATE OR REPLACE PROCEDUR ... | DECLARE v_efficiency_curso ... | select * from room

SQL | Output | Statistics

```oracle
DECLARE
    v_efficiency_cursor SYS_REFCURSOR;
    TYPE EfficiencyRecord IS RECORD (
        RID Room.RID%TYPE,
        Type Room.Type%TYPE,
        Total_Cleanings INT,
        Completion_Rate FLOAT,
        Inspection_Pass_Rate FLOAT
    );
    v_efficiency EfficiencyRecord;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Room Cleaning Management');
    DBMS_OUTPUT.PUT_LINE('-------------------------');

    -- Get room cleaning efficiency
    v_efficiency_cursor := get_room_cleaning_efficiency;
    DBMS_OUTPUT.PUT_LINE('Room Cleaning Efficiency:');
    DBMS_OUTPUT.PUT_LINE('RID | Type | Total Cleanings | Completion Rate | Inspection Pass Rate');
    LOOP
        FETCH v_efficiency_cursor INTO v_efficiency;
        EXIT WHEN v_efficiency_cursor%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE(
            v_efficiency.RID || ' | ' ||
            v_efficiency.Type || ' | ' ||
            v_efficiency.Total_Cleanings || ' | ' ||
            ROUND(v_efficiency.Completion_Rate, 2) || '% | ' ||
            ROUND(v_efficiency.Inspection_Pass_Rate, 2) || '%'
        );
    END LOOP;
    CLOSE v_efficiency_cursor;

    DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Assigning Cleaning Tasks...');
    assign_cleaning_tasks;

    DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Room Cleaning Management Completed.');
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error in room_cleaning_management: ' || SQLERRM);
```

40:2 | c##jerem@XE | [6:53:54 PM] Done in 0.095 seconds

Output:

function_1.sql | procedure_1.sql | mainProgram_1.sql | select * from room

SQL | Output | Statistics

Clear | Buffer size 20000 | Enabled

```
Room Cleaning Management
------------------------
Room Cleaning Efficiency:
RID | Type | Total Cleanings | Completion Rate | Inspection Pass Rate
179 | Single | 1 | 100% | 100%
242 | Single | 1 | 100% | 100%
87 | Suite | 1 | 100% | 100%
343 | Single | 1 | 100% | 100%
195 | Single | 1 | 100% | 100%
337 | Double | 2 | 100% | 100%
301 | Single | 1 | 100% | 100%
306 | Double | 1 | 100% | 100%
81 | Double | 1 | 100% | 100%
217 | Single | 1 | 100% | 100%
105 | Suite | 2 | 100% | 100%
304 | Double | 1 | 100% | 100%
279 | Double | 1 | 100% | 100%
222 | Double | 1 | 100% | 100%
371 | Double | 1 | 100% | 100%
315 | Double | 2 | 100% | 100%
375 | Suite | 1 | 100% | 100%
103 | Suite | 2 | 100% | 100%
356 | Single | 1 | 100% | 100%
257 | Double | 2 | 100% | 100%
244 | Suite | 3 | 100% | 66,67%
69 | Double | 3 | 100% | 66,67%
291 | Double | 3 | 100% | 66,67%
373 | Single | 2 | 100% | 50%
393 | Double | 4 | 100% | 50%
155 | Double | 2 | 100% | 50%
77 | Suite | 2 | 100% | 50%
188 | Single | 2 | 100% | 50%
107 | Double | 2 | 100% | 50%
5 | Double | 2 | 100% | 50%
43 | Single | 2 | 100% | 50%
117 | Suite | 2 | 100% | 50%
57 | Single | 2 | 100% | 50%
```

542:1 c##jerem@XE SQL script saved successfully

CREATE OR REPLACE FUNCTION ... | CREATE OR REPLACE PROCEDUR ... | DECLARE v_efficiency_curso ... | select * from room

SQL | Output | Statistics

Clear | Buffer size 20000 | Enabled

```
Assigned task 501 for room 302 (Suite) to Howie Statham
Assigned task 502 for room 308 (Double) to Neil McGinley
Assigned task 503 for room 310 (Double) to Juan Gill
Assigned task 504 for room 312 (Single) to Susan Voight
Assigned task 505 for room 315 (Double) to Winona O'Neal
Assigned task 506 for room 319 (Suite) to Winona Short
Assigned task 507 for room 328 (Double) to Judd Soul
Assigned task 508 for room 335 (Single) to Merrill Simpson
Assigned task 509 for room 337 (Double) to Chrissie Wright
Assigned task 510 for room 338 (Single) to Sander Harmon
Assigned task 511 for room 341 (Double) to John Doe
Assigned task 512 for room 342 (Single) to Alice Johnson
Assigned task 513 for room 343 (Single) to Charlie Brown
Assigned task 514 for room 345 (Single) to Emma Garcia
Assigned task 515 for room 348 (Single) to Madeleine Barry
Assigned task 516 for room 349 (Double) to Woody Cassel
Assigned task 517 for room 351 (Double) to Lonnie Downie
Assigned task 518 for room 353 (Double) to Jet Chappelle
Assigned task 519 for room 354 (Suite) to Larry Imbruglia
Assigned task 520 for room 358 (Single) to Amy Orton
Assigned task 521 for room 364 (Single) to Ahmad Mortensen
Assigned task 522 for room 366 (Double) to Demi Blackmore
Assigned task 523 for room 376 (Single) to Tyrone Deschanel
Assigned task 524 for room 377 (Single) to Victoria Bale
Assigned task 525 for room 378 (Single) to Sandra Holeman
Assigned task 526 for room 380 (Double) to Andrea Fender
Assigned task 527 for room 381 (Double) to Natacha Mirren
Assigned task 528 for room 382 (Suite) to Rosco Arquette
Assigned task 529 for room 383 (Suite) to Rene Fiennes
Assigned task 530 for room 385 (Double) to Sal Herndon
Assigned task 531 for room 390 (Suite) to Tea Gertner
Assigned task 532 for room 396 (Suite) to Veruca Torres
Assigned 131 cleaning tasks.

Room Cleaning Management Completed.
```

542:1 | c##jerem@XE | [6:53:54 PM] Done in 0.095 seconds

Database before execution:

SQL Output Statistics

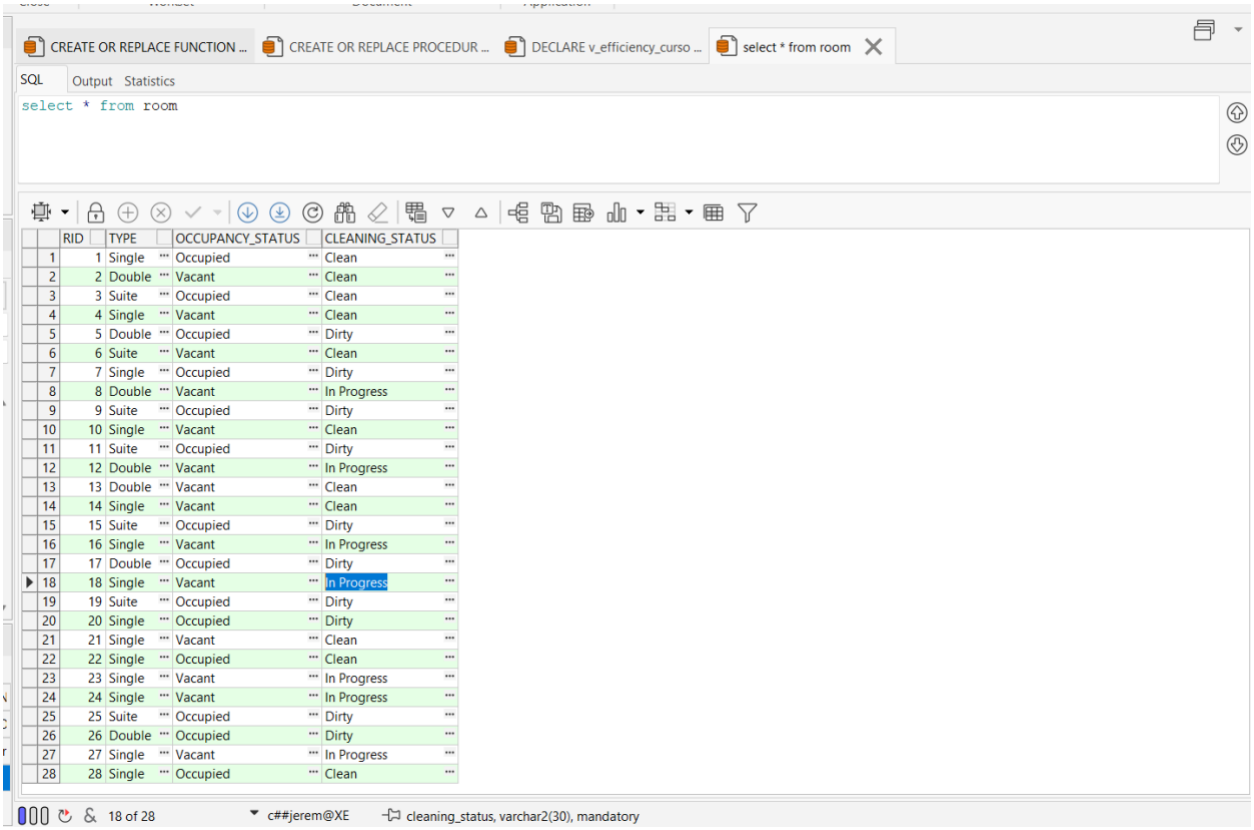
```
select * from room
```

	RID	TYPE	OCCUPANCY_STATUS	CLEANING_STATUS
1	1	Single	Occupied	Clean
2	2	Double	Vacant	Dirty
3	3	Suite	Occupied	Clean
4	4	Single	Vacant	Clean
5	5	Double	Occupied	Dirty
6	6	Suite	Vacant	Clean
7	7	Single	Occupied	Dirty
8	8	Double	Vacant	Dirty
9	9	Suite	Occupied	Dirty
10	10	Single	Vacant	Clean
11	11	Suite	Occupied	Dirty
12	12	Double	Vacant	Dirty
13	13	Double	Vacant	Clean
14	14	Single	Vacant	Clean
15	15	Suite	Occupied	Dirty
16	16	Single	Vacant	Dirty
17	17	Double	Occupied	Dirty
18	18	Single	Vacant	Dirty
19	19	Suite	Occupied	Dirty
20	20	Single	Occupied	Dirty
21	21	Single	Vacant	Clean
22	22	Single	Occupied	Clean
23	23	Single	Vacant	Dirty
24	24	Single	Vacant	Dirty
25	25	Suite	Occupied	Dirty
26	26	Double	Occupied	Dirty
27	27	Single	Vacant	Dirty
28	28	Single	Occupied	Clean

18 of 28 c##jerem@XE cleaning_status, varchar2(30), mandatory

Database after execution:

RID 18 changed from “Dirty” to “In Progress”



	RID	TYPE	OCCUPANCY_STATUS	CLEANING_STATUS
1	1	Single	Occupied	Clean
2	2	Double	Vacant	Clean
3	3	Suite	Occupied	Clean
4	4	Single	Vacant	Clean
5	5	Double	Occupied	Dirty
6	6	Suite	Vacant	Clean
7	7	Single	Occupied	Dirty
8	8	Double	Vacant	In Progress
9	9	Suite	Occupied	Dirty
10	10	Single	Vacant	Clean
11	11	Suite	Occupied	Dirty
12	12	Double	Vacant	In Progress
13	13	Double	Vacant	Clean
14	14	Single	Vacant	Clean
15	15	Suite	Occupied	Dirty
16	16	Single	Vacant	In Progress
17	17	Double	Occupied	Dirty
18	18	Single	Vacant	In Progress
19	19	Suite	Occupied	Dirty
20	20	Single	Occupied	Dirty
21	21	Single	Vacant	Clean
22	22	Single	Occupied	Clean
23	23	Single	Vacant	In Progress
24	24	Single	Vacant	In Progress
25	25	Suite	Occupied	Dirty
26	26	Double	Occupied	Dirty
27	27	Single	Vacant	In Progress
28	28	Single	Occupied	Clean

Function: get_room_cleaning_efficiency

This function calculates and returns the cleaning efficiency for each room. It uses a REF CURSOR to return a result set containing:

- Room ID and Type
- Total number of cleaning tasks for each room
- Completion rate of cleaning tasks
- Inspection pass rate

It joins the Room, Cleaning_Task, and Inspection_Logs tables to gather this information. The results are ordered by completion rate and inspection pass rate in descending order.

CREATE OR REPLACE FUNCTION ... | CREATE OR REPLACE PROCEDUR ... | DECLARE v_efficiency_curso ... | select * from room

SQL | Output | Statistics

```oracle
CREATE OR REPLACE FUNCTION get_room_cleaning_efficiency
RETURN SYS_REFCURSOR
IS
    v_result SYS_REFCURSOR;
BEGIN
    OPEN v_result FOR
        SELECT
            r.RID,
            r.Type,
            COUNT(ct.TID) as Total_Cleanings,
            AVG(CASE
                WHEN ct.Status = 'Completed' THEN 1
                ELSE 0
            END) * 100 as Completion_Rate,
            AVG(CASE
                WHEN i.Inspection_Result = 'Pass' THEN 1
                ELSE 0
            END) * 100 as Inspection_Pass_Rate
        FROM Room r
        LEFT JOIN Cleaning_Task ct ON r.RID = ct.RID
        LEFT JOIN Inspection_Logs i ON r.RID = i.RID
        GROUP BY r.RID, r.Type
        ORDER BY Completion_Rate DESC, Inspection_Pass_Rate DESC;

    RETURN v_result;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error in get_room_cleaning_efficiency: ' || SQLERRM);
        RETURN NULL;
END get_room_cleaning_efficiency;
/
```

9:20 | c##jerem@XE | [6:31:07 PM] Done in 0.141 seconds

Procedure: assign_cleaning_tasks

This procedure assigns cleaning tasks to available staff members. It:

- Uses cursors to find dirty, vacant rooms and available staff
- Assign tasks to staff members, trying to distribute work evenly
- Updates the room status to 'In Progress' when a task is assigned
- Handles exceptions, such as when no staff is available

CREATE OR REPLACE FUNCTION ... | CREATE OR REPLACE PROCEDUR ... | DECLARE v_efficiency_curso ... | select * from room

SQL | Output | Statistics

```oracle
CREATE OR REPLACE PROCEDURE assign_cleaning_tasks
IS
    CURSOR c_dirty_rooms IS
        SELECT RID, Type
        FROM Room
        WHERE Cleaning_Status = 'Dirty'
        AND Occupancy_Status = 'Vacant'
        FOR UPDATE;

    CURSOR c_available_staff IS
        SELECT SM.SID, SM.First_Name, SM.Last_Name
        FROM Staff_Member SM
        LEFT JOIN (
            SELECT SID, COUNT(*) as Active_Tasks
            FROM Cleaning_Task
            WHERE Status = 'Pending'
            GROUP BY SID
        ) CT ON SM.SID = CT.SID
        WHERE CT.Active_Tasks IS NULL OR CT.Active_Tasks < 3
        ORDER BY CT.Active_Tasks NULLS FIRST;

    TYPE StaffRecord IS RECORD (
        SID Staff_Member.SID%TYPE,
        First_Name Staff_Member.First_Name%TYPE,
        Last_Name Staff_Member.Last_Name%TYPE
    );
    v_staff StaffRecord;
    v_task_id Cleaning_Task.TID%TYPE;
    v_assigned_count INT := 0;
    e_no_staff EXCEPTION;
BEGIN
    OPEN c_available_staff;
    FETCH c_available_staff INTO v_staff;
    IF c_available_staff%NOTFOUND THEN
        RAISE e_no_staff;
    END IF;
    CLOSE c_available_staff;
```

65:12 | c##jerem@XE | [6:33:48 PM] Done in 0.050 seconds

Main Program: staff_performance_and_supply_management

This program focuses on staff performance and supply management, the function and procedure will be explained later. It:

- Prompts for a review date
- Calls calculate_staff_performance to display staff performance metrics
- Calls update_cleaning_supplies to restock necessary supplies
- Gives a comprehensive view of staff efficiency and supply status

select * from cleaning_supply | DECLARE v_performance_curs ... | CREATE OR REPLACE PROCEDUR ... | CREATE OR REPLACE FUNCTION ...

SQL | Output | Statistics

```oracle_plsql
DECLARE
    v_performance_cursor SYS_REFCURSOR;
    TYPE PerformanceRecord IS RECORD (
        SID Staff_Member.SID%TYPE,
        Staff_Name VARCHAR2(61),
        Total_Tasks INT,
        Completed_Tasks INT,
        Completion_Rate FLOAT,
        Inspection_Pass_Rate FLOAT
    );
    v_performance PerformanceRecord;
    v_review_date DATE;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Staff Performance and Supply Management');
    DBMS_OUTPUT.PUT_LINE('----------------------------------------');

    -- Get user input for review date
    v_review_date := TO_DATE('&review_date', 'YYYY-MM-DD');

    -- Calculate staff performance
    v_performance_cursor := calculate_staff_performance(v_review_date);
    DBMS_OUTPUT.PUT_LINE('Staff Performance for ' || TO_CHAR(v_review_date, 'YYYY-MM-DD') || ':');
    DBMS_OUTPUT.PUT_LINE('SID | Name | Total Tasks | Completed Tasks | Completion Rate | Inspection Pass Rate');
    LOOP
        FETCH v_performance_cursor INTO v_performance;
        EXIT WHEN v_performance_cursor%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE(
            v_performance.SID || ' | ' ||
            v_performance.Staff_Name || ' | ' ||
            v_performance.Total_Tasks || ' | ' ||
            v_performance.Completed_Tasks || ' | ' ||
            ROUND(v_performance.Completion_Rate, 2) || '% | ' ||
            ROUND(v_performance.Inspection_Pass_Rate, 2) || '%'
        );
    END LOOP;
    CLOSE v_performance_cursor;

    DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Updating Cleaning Supplies...');
```

1:1 | c##jerem@XE | [7:14:23 PM] Done in 0.012 seconds

Close | Worksheet | Document | Application

select * from cleaning_supply | SQL Window | CREATE OR REPLACE PROCEDUR ... | CREATE OR REPLACE FUNCTION ...

SQL | Output | Statistics

```oracle_plsql
DECLARE
    v_performance_cursor SYS_REFCURSOR;
    TYPE PerformanceRecord IS RECORD (
        SID Staff_Member.SID%TYPE,
        Staff_Name VARCHAR2(61),
        Total_Tasks INT,
        Completed_Tasks INT,
        Completion_Rate FLOAT,
        Inspection_Pass_Rate FLOAT
    );
    v_performance PerformanceRecord;
    v_review_date DATE;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Staff Performance and Supply Mana
    DBMS_OUTPUT.PUT_LINE('--------------------------------

    -- Get user input for review date
    v_review_date := TO_DATE('&review_date', 'YYYY-MM-DD');

    -- Calculate staff performance
    v_performance_cursor := calculate_staff_performance(v_review_date);
    DBMS_OUTPUT.PUT_LINE('Staff Performance for ' || TO_CHAR(v_review_date, 'YYYY-MM-DD') || ':');
    DBMS_OUTPUT.PUT_LINE('SID | Name | Total Tasks | Completed Tasks | Completion Rate | Inspection Pass Rate');
    LOOP
        FETCH v_performance_cursor INTO v_performance;
        EXIT WHEN v_performance_cursor%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE(
            v_performance.SID || ' | ' ||
            v_performance.Staff_Name || ' | ' ||
            v_performance.Total_Tasks || ' | ' ||
            v_performance.Completed_Tasks || ' | ' ||
            ROUND(v_performance.Completion_Rate, 2) || '% | ' ||
            ROUND(v_performance.Inspection_Pass_Rate, 2) || '%'
        );
    END LOOP;
    CLOSE v_performance_cursor;

    DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Updating Cleaning Supplies...');
```

Variables

Name	Value
review_date	

OK | Cancel | Clear

46:2 | c##jerem@XE | Initializing...

Output:

mainProgram_2.sql | procedure_2.sql | function_2.sql

SQL | Output | Statistics

Clear | Buffer size 50000 | Enabled

```
Staff Performance and Supply Management
----------------------------------------
Staff Performance for 2024-03-01:
SID | Name | Total Tasks | Completed Tasks | Completion Rate | Inspection Pass Rate
295 | Geoff Braugher | 1 | 1 | 100% | 0%
258 | Phoebe Dunst | 1 | 1 | 100% | 0%
```

Database before execution:

SQL | Output | Statistics

```
select * from cleaning_supply
```

	SUID	NAME	TOTAL_QUANTITY	LOCATION
1	6	Trash Bags	160	Storage A
2	17	Floor Polish	130	Storage C
3	19	Broom	235	Storage A
4	20	White Vinegar	195	Storage C
5	25	Wood Polish	91	Storage C
6	28	Oven Cleaner	172	Storage C
7	33	Mop	213	Storage C
8	34	Detergent	285	Storage B
9	35	Stove Degreaser	340	Storage C
10	40	Carpet Cleaner	320	Storage C
11	49	Toilet Bowl Cleaner	0	Storage C
12	51	Fabric Softener	446	Storage B
13	54	Shoe Deodorizer	404	Storage B
14	61	Bathroom Deodorizer	336	Storage B
15	64	Carpet Cleaner	83	Storage A
16	69	Stove Degreaser	278	Storage B
17	70	Disinfectant	345	Storage C
18	74	Fabric Softener	339	Storage B
19	76	Essential Oils	101	Storage B
20	78	Rubber Gloves	341	Storage B
21	81	White Vinegar	203	Storage A
22	85	Paper Towels	359	Storage A
23	90	Shoe Deodorizer	307	Storage C
24	95	Baking Soda	276	Storage B
25	100	Broom	313	Storage B
26	101	Oven Cleaner	43	Storage B
27	102	Shoe Deodorizer	212	Storage A
28	103	Oven Cleaner	463	Storage A

Database after execution:

The total quantity of Item SUID 49 changed from 0 to 43.

select * from cleaning_supply | DECLARE v_performance_curs ... | CREATE OR REPLACE PROCEDUR ... | CREATE OR REPLACE FUNCTION ...

SQL | Output | Statistics

```
select * from cleaning_supply
```

	SUID	NAME	TOTAL_QUANTITY	LOCATION
1	6	Trash Bags	160	Storage A
2	17	Floor Polish	130	Storage C
3	19	Broom	235	Storage A
4	20	White Vinegar	195	Storage C
5	25	Wood Polish	91	Storage C
6	28	Oven Cleaner	172	Storage C
7	33	Mop	213	Storage C
8	34	Detergent	285	Storage B
9	35	Stove Degreaser	340	Storage C
10	40	Carpet Cleaner	320	Storage C
11	49	Toilet Bowl Cleaner	43	Storage C
12	51	Fabric Softener	446	Storage B
13	54	Shoe Deodorizer	404	Storage B
14	61	Bathroom Deodorizer	336	Storage B
15	64	Carpet Cleaner	83	Storage A
16	69	Stove Degreaser	278	Storage B
17	70	Disinfectant	345	Storage C
18	74	Fabric Softener	339	Storage B
19	76	Essential Oils	101	Storage B
20	78	Rubber Gloves	341	Storage B
21	81	White Vinegar	203	Storage A
22	85	Paper Towels	359	Storage A
23	90	Shoe Deodorizer	307	Storage C
24	95	Baking Soda	276	Storage B
25	100	Broom	313	Storage B
26	101	Oven Cleaner	43	Storage B
27	102	Shoe Deodorizer	212	Storage A
28	103	Oven Cleaner	463	Storage A

1:30 c##jerem@XE [7:14:49 PM] 28 rows selected in 0.017 seconds (more...)

Function: calculate_staff_performance

This function evaluates staff performance for a given date. It returns a REF CURSOR with:

- Staff ID and full name
- Total tasks assigned
- Number of completed tasks
- Task completion rate
- Inspection pass rate for their cleaned rooms

It uses a Common Table Expression (CTE) to calculate these metrics, joining the Staff_Member, Cleaning_Task, and Inspection_Logs tables. The results are ordered by completion rate and inspection pass rate.

select * from room | SQL Window | SQL Window | CREATE OR REPLACE FUNCTION ...

SQL | Output | Statistics

```oracle_plsql
CREATE OR REPLACE FUNCTION calculate_staff_performance(p_date IN DATE)
RETURN SYS_REFCURSOR
IS
    v_result SYS_REFCURSOR;
BEGIN
    OPEN v_result FOR
        WITH staff_tasks AS (
            SELECT
                ct.SID,
                COUNT(*) as Total_Tasks,
                SUM(CASE WHEN ct.Status = 'Completed' THEN 1 ELSE 0 END) as Completed_Tasks,
                AVG(CASE
                    WHEN i.Inspection_Result = 'Pass' THEN 1
                    ELSE 0
                END) as Inspection_Pass_Rate
            FROM Cleaning_Task ct
            LEFT JOIN Inspection_Logs i ON ct.RID = i.RID
            WHERE TRUNC(i.Inspection_Date) = TRUNC(p_date)
            GROUP BY ct.SID
        )
        SELECT
            sm.SID,
            sm.First_Name || ' ' || sm.Last_Name as Staff_Name,
            NVL(st.Total_Tasks, 0) as Total_Tasks,
            NVL(st.Completed_Tasks, 0) as Completed_Tasks,
            CASE
                WHEN NVL(st.Total_Tasks, 0) > 0 THEN
                    ROUND(NVL(st.Completed_Tasks, 0) / NVL(st.Total_Tasks, 0) * 100, 2)
                ELSE 0
            END as Completion_Rate,
            NVL(ROUND(st.Inspection_Pass_Rate * 100, 2), 0) as Inspection_Pass_Rate
        FROM Staff_Member sm
        LEFT JOIN staff_tasks st ON sm.SID = st.SID
        ORDER BY Completion_Rate DESC, Inspection_Pass_Rate DESC;

    RETURN v_result;
EXCEPTION
    WHEN OTHERS THEN
```

42:2 | c##jerem@XE | [7:12:18 PM] Done in 0.016 seconds

Procedure: update_cleaning_supplies

This procedure manages the restocking of cleaning supplies. It:

- Identifies supplies with low quantities
- Restocks them with a random amount between 20 and 50 units
- Calculates the total cost of restocking
- Limits restocking to 5 items per run to prevent overstocking

select * from room | SQL Window | CREATE OR REPLACE PROCEDUR ... | CREATE OR REPLACE FUNCTION ...

SQL | Output | Statistics

```oracle_plsql
CREATE OR REPLACE PROCEDURE update_cleaning_supplies
IS
    CURSOR c_low_supplies IS
        SELECT SUID, Name, Total_Quantity, Location
        FROM Cleaning_Supply
        WHERE Total_Quantity < 10
        FOR UPDATE;

    v_restock_amount INT;
    v_total_restocked INT := 0;
    v_total_cost FLOAT := 0;
    e_restock_limit_reached EXCEPTION;
BEGIN
    FOR supply IN c_low_supplies LOOP
        v_restock_amount := FLOOR(DBMS_RANDOM.VALUE(20, 51)); -- Random restock between 20 and 50

        UPDATE Cleaning_Supply
        SET Total_Quantity = Total_Quantity + v_restock_amount
        WHERE CURRENT OF c_low_supplies;

        v_total_restocked := v_total_restocked + 1;
        v_total_cost := v_total_cost + (v_restock_amount * 2.5); -- Assuming $2.5 per unit

        DBMS_OUTPUT.PUT_LINE('Restocked ' || supply.Name || ' with ' || v_restock_amount || ' units in ' || supply.Location);

        IF v_total_restocked >= 5 THEN
            RAISE e_restock_limit_reached;
        END IF;
    END LOOP;

    IF v_total_restocked = 0 THEN
        DBMS_OUTPUT.PUT_LINE('All supplies are adequately stocked.');
    ELSE
        DBMS_OUTPUT.PUT_LINE('Total items restocked: ' || v_total_restocked);
        DBMS_OUTPUT.PUT_LINE('Total restock cost: $' || ROUND(v_total_cost, 2));
    END IF;

    COMMIT;
```

48:2 | c##jerem@XE | [7:12:31 PM] Done in 0.033 seconds