

## EXPERIMENT 4

**Student Name:** Vishesh Aggarwal  
**Branch:** AIT-CSE (Cybersecurity)  
**Semester:** 4  
**Subject Name:** Database Management System

**UID:** 24BCY70145  
**Section/Group:** 24AIT\_KRG-G2  
**Subject Code:** 24CSH-298

### Experiment 4 – Data Analysis Using SQL and PL/SQL

#### Experiment

Experiment 4: Creating tables, inserting data, performing conditional queries, and using PL/SQL blocks to analyze schema violations and student grades. This experiment demonstrates table creation, updates, conditional logic, and ordering in Oracle SQL and PL/SQL.

#### Aim

The aim of this experiment is to practice working with Oracle SQL tables, using conditional logic to determine status and grades, and displaying results using SELECT queries and PL/SQL blocks.

#### Objective

- To create and populate tables in Oracle SQL.
- To use CASE statements for conditional evaluation of violation counts and student grades.
- To add and update columns based on conditions.
- To use PL/SQL anonymous blocks for status messages.
- To sort query results based on defined criteria.

#### Software Requirements

- Database: Oracle XE or Oracle Live SQL

#### Practical / Experiment Steps

1. Create a table schema\_violations with columns id, schema\_name, and violation\_count.
2. Insert data for various departments into the schema\_violations table.
3. Select violation status for each department using a CASE statement.

## EXPERIMENT 4

4. Add a new column `approval_status` to `schema_violations`.
5. Update `approval_status` based on violation count using a CASE statement.
6. Display the updated `schema_violations` table.
7. Execute a PL/SQL block to print a system status message based on a variable `v_count`.
8. Create a `students` table with columns `name` and `marks`.
9. Insert student data into the `students` table.
10. Display student grades using a CASE statement based on marks.
11. Order `schema_violations` by severity using a CASE statement in ORDER BY.

### Procedure of the Experiment

1. Open Oracle XE or Live SQL and connect to the database.
2. Create the `schema_violations` and `students` tables.
3. Insert sample data into both tables.
4. Execute SELECT queries with CASE statements to analyze violation and grade data.
5. Alter and update tables using conditional logic.
6. Write and execute a PL/SQL anonymous block for dynamic status messages.
7. Sort and retrieve data based on defined severity.
8. Observe outputs at each step and take screenshots for documentation.

### Input / Output Details

#### Input

- `schema_violations` table: `id`, `schema_name`, `violation_count`
- `students` table: `name`, `marks`
- PL/SQL block variable: `v_count`
- Conditional logic in SELECT and UPDATE statements

---

### Step-wise Output

## EXPERIMENT 4

### Step 1 – Create schema\_violations table

```

1  -- Create schema_violations table
2  CREATE TABLE schema_violations (
3      id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY
4      schema_name VARCHAR2(50),
5      violation_count NUMBER
6  );
7
8  -- Insert data

```

Query result    **Script output**    DBMS output    Explain Plan    SQL history



```

SQL> CREATE TABLE schema_violations (
      id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
      schema_name VARCHAR2(50),
      violation_count NUMBER...

```

Show more...

**ORA-00955: name is already used by an existing object**

<https://docs.oracle.com/error-help/db/ora-00955/>  
Error at Line: 4 Column: 0

### Step 2 – Insert data into schema\_violations

```

8  -- Insert data
9  INSERT INTO schema_violations (schema_name, violation_count) VALUES ('Finance', 0);
10 INSERT INTO schema_violations (schema_name, violation_count) VALUES ('HR', 2);
11 INSERT INTO schema_violations (schema_name, violation_count) VALUES ('Sales', 5);
12 INSERT INTO schema_violations (schema_name, violation_count) VALUES ('Security', 9);
13 INSERT INTO schema_violations (schema_name, violation_count) VALUES ('Admin', 1);
14
15 COMMIT;
16
17 -- Select with violation status
18 SELECT

```

Query result    **Script output**    DBMS output    Explain Plan    SQL history

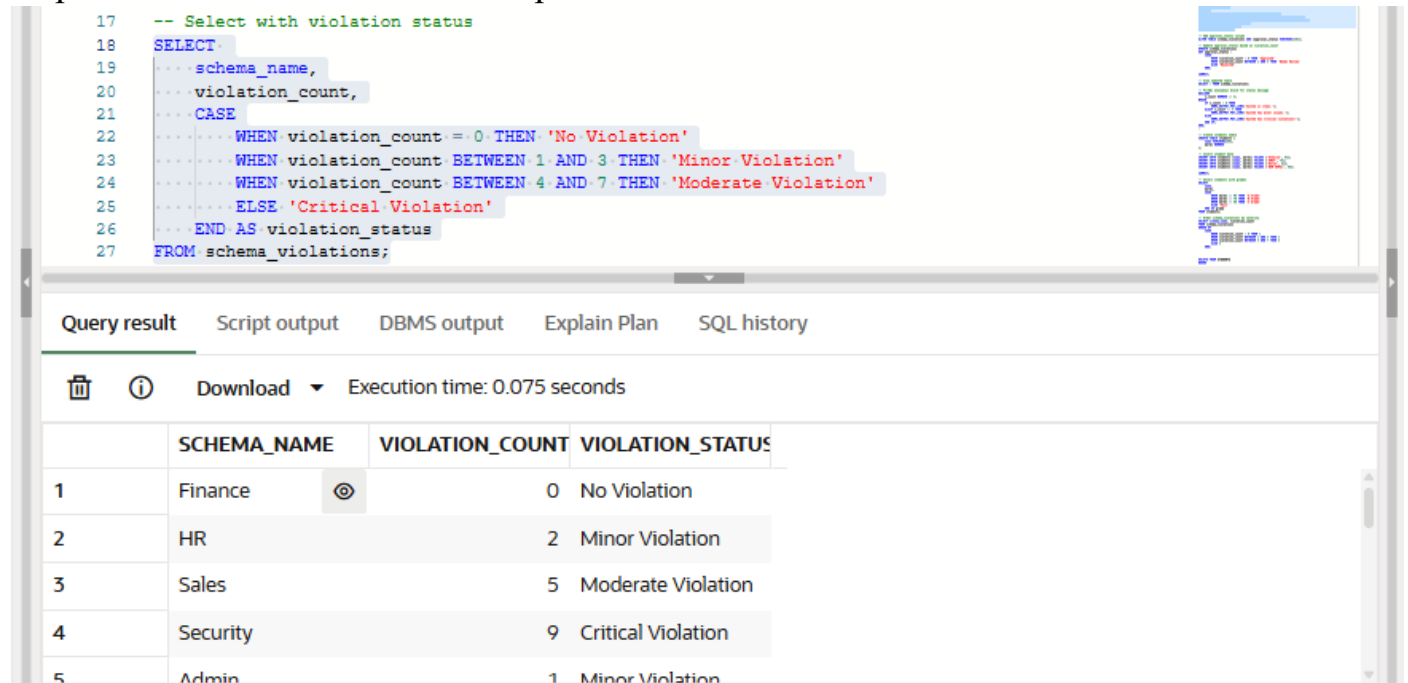


1 row inserted.

Elapsed: 00:00:00.000

## EXPERIMENT 4

### Step 3 – Violation status of each department



```

17  -- Select with violation status
18  SELECT
19      schema_name,
20      violation_count,
21      CASE
22          WHEN violation_count = 0 THEN 'No Violation'
23          WHEN violation_count BETWEEN 1 AND 3 THEN 'Minor Violation'
24          WHEN violation_count BETWEEN 4 AND 7 THEN 'Moderate Violation'
25          ELSE 'Critical Violation'
26      END AS violation_status
27  FROM schema_violations;
  
```

Query result   Script output   DBMS output   Explain Plan   SQL history

Download   Execution time: 0.075 seconds

	SCHEMA_NAME	VIOLATION_COUNT	VIOLATION_STATUS
1	Finance	0	No Violation
2	HR	2	Minor Violation
3	Sales	5	Moderate Violation
4	Security	9	Critical Violation
5	Admin	1	Minor Violation

schema\_name   violation\_count   violation\_status

Finance	0	No Violation
HR	2	Minor Violation
Sales	5	Moderate Violation
Security	9	Critical Violation
Admin	1	Minor Violation

## EXPERIMENT 4

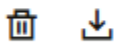
### Step 4 – Add approval\_status column

```

28
29 -- Add approval_status column
30 ALTER TABLE schema_violations ADD (approval_status VARCHAR2(20));
31
32 -- Update approval_status based on violation_count
33 UPDATE schema_violations
34 SET approval_status =

```

Query result Script output DBMS output Explain Plan SQL history



ORA-01430: column being added already exists in table

<https://docs.oracle.com/error-help/db/ora-01430/>  
Error at Line: 5 Column: 0

### Step 5 –

### Update approval\_status based on violation\_count

```

33 UPDATE schema_violations
34 SET approval_status =
35 ... CASE
36 ... WHEN violation_count = 0 THEN 'Approved'
37 ... WHEN violation_count BETWEEN 1 AND 5 THEN 'Needs Review'
38 ... ELSE 'Rejected'
39 ... END;
40
41 COMMIT;
42

```



Query result Script output DBMS output Explain Plan SQL history

Download Execution time: 0.075 seconds

	SCHEMA_NAME	VIOLATION_COUNT	VIOLATION_STATUS
	Finance	0	No Violation
	HR	2	Minor Violation
	Sales	5	Moderate Violation
	Security	9	Critical Violation
	Admin	1	Minor Violation

## EXPERIMENT 4

### Step 6 – View updated schema\_violations table

```

44 SELECT * FROM schema_violations;
45
46 -- PL/SQL anonymous block for status message
47 DECLARE
48     v_count NUMBER := 4;
49 BEGIN
50     IF v_count = 0 THEN
51         DBMS_OUTPUT.PUT_LINE('System is clean.');
```

```

48     ELSEIF v_count <= 5 THEN
49         DBMS_OUTPUT.PUT_LINE('System has minor issues.');
```

Query result    Script output    DBMS output    Explain Plan    SQL history				
Download    Execution time: 0.005 seconds				
ID	SCHEMA_NAME	VIOLATION_COUNT	APPROVAL_STATUS	
1	21 Finance	0	Approved	
2	22 HR	2	Needs Review	
3	23 Sales	5	Needs Review	
4	24 Security	9	Rejected	
5	25 Admin	1	Needs Review	

id	schema_name	violation_count	violation_status	approval_status
1	Finance	0	No Violation	Approved
2	HR	2	Minor Violation	Needs Review
3	Sales	5	Moderate Violation	Needs Review
4	Security	9	Critical Violation	Rejected
5	Admin	1	Minor Violation	Needs Review

### Step 7 – PL/SQL anonymous block for status message

Screenshot: s7.png

Output:

## EXPERIMENT 4

```

44 SELECT * FROM schema_violations;
45
46 -- PL/SQL anonymous block for status message
47 DECLARE
48     v_count NUMBER := 4;
49 BEGIN
50     IF v_count = 0 THEN
51         DBMS_OUTPUT.PUT_LINE('System is clean.');
```

```

48     IF v_count = 0 THEN
49         DBMS_OUTPUT.PUT_LINE('System is clean.');
```

Query result   Script output   DBMS output   Explain Plan   SQL history

Download   Execution time: 0.005 seconds

	ID	SCHEMA_NAME	VIOLATION_COUNT	APPROVAL_STATUS
1	21	Finance	0	Approved
2	22	HR	2	Needs Review
3	23	Sales	5	Needs Review
4	24	Security	9	Rejected
5	25	Admin	1	Needs Review

System has minor issues.

### Step 8 – Create students table

```

50 CREATE TABLE students (
51     name VARCHAR2(50),
52     marks NUMBER
53 );
54
55 -- Insert student data
56 INSERT INTO students (name, marks) VALUES ('Utkarsh', 92);
57 INSERT INTO students (name, marks) VALUES ('AMAY', 75);
58 INSERT INTO students (name, marks) VALUES ('Karan', 61);
59 INSERT INTO students (name, marks) VALUES ('RAM GOPAL', 48);
60
61 COMMIT;
```

```

50 CREATE TABLE students (
51     name VARCHAR2(50),
52     marks NUMBER
53 );
54
55 -- Insert student data
56 INSERT INTO students (name, marks) VALUES ('Utkarsh', 92);
57 INSERT INTO students (name, marks) VALUES ('AMAY', 75);
58 INSERT INTO students (name, marks) VALUES ('Karan', 61);
59 INSERT INTO students (name, marks) VALUES ('RAM GOPAL', 48);
60
61 COMMIT;
```

Query result   Script output   DBMS output   Explain Plan   SQL history

Download

Elapsed: 00:00:00.008

```

SQL> CREATE TABLE students (
      name VARCHAR2(50),
      marks NUMBER
    )
```

ORA-00955: name is already used by an existing object



## EXPERIMENT 4

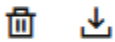
### Step 9 – Insert student data

```

55  -- Insert student data
56  INSERT INTO students (name, marks) VALUES ('Utkarsh', 92);
57  INSERT INTO students (name, marks) VALUES ('AMAY', 75);
58  INSERT INTO students (name, marks) VALUES ('Karan', 61);
59  INSERT INTO students (name, marks) VALUES ('RAM GOPAL', 48);
60
61  COMMIT;
--

```

Query result    Script output    DBMS output    Explain Plan    SQL history



Elapsed: 00:00:00.002

SQL> INSERT INTO students (name, marks) VALUES ('RAM GOPAL', 48)

1 row inserted.

### Step 10 –

### Student grades using CASE statement

```

63  -- Select students with grades
64  SELECT
65  ... name,
66  ... marks,
67  ... CASE
68  ...   WHEN marks >= 90 THEN 'A Grade'
69  ...   WHEN marks >= 70 THEN 'B Grade'
70  ...   WHEN marks >= 50 THEN 'C Grade'
71  ...   ELSE 'Fail'
72  ... END AS grade
73  FROM students;
74
75  -- Order schema_violations by severity
76  SELECT schema_name, violation_count
77  FROM schema_violations
78  ORDER BY
79  CASE
80  ...   WHEN violation_count = 0 THEN 1

```



Query result    Script output    DBMS output    Explain Plan    SQL history

Download    Execution time: 0.005 seconds

NAME	MARKS	GRADE
Utkarsh	92	A Grade
AMAY	75	B Grade
Karan	61	C Grade
RAM GOPAL	48	Fail
Utkarsh	92	A Grade

name    marks    grade

Vishesh 92    A Grade



## EXPERIMENT 4

name    marks    grade

Shreya    75    B Grade

Navneet 61    C Grade

RAM    48    Fail

Step 11 – Schema violations ordered by severity

```

76 SELECT schema_name, violation_count
77 FROM schema_violations
78 ORDER BY
79 ... CASE
80 ... WHEN violation_count = 0 THEN 1
81 ... WHEN violation_count BETWEEN 1 AND 3 THEN 2
82 ... WHEN violation_count BETWEEN 4 AND 7 THEN 3
83 ... ELSE 4
84 ... END;
85
86
87
88 DELETE FROM STUDENTS
89 WHERE
  
```



Query result    Script output    DBMS output    Explain Plan    SQL history

Download    Execution time: 0.005 seconds

SCHEMA_NAME	VIOLATION_COUNT
Finance	0
Finance	0
Finance	0
Finance	0
HR	2
Admin	1
Sales	5
Security	9

schema\_name violation\_count

Finance    0

HR    2

Admin    1

Sales    5

Security    9

### Learning Outcome

After completing this experiment, the student will be able to:

## EXPERIMENT 4

- Create and populate tables in Oracle SQL.
- Use CASE statements to evaluate conditions in queries.
- Update table data based on conditional logic.
- Write PL/SQL blocks for dynamic status messages.
- Sort query results using CASE statements in ORDER BY.
- Analyze data and assign grades or approval statuses automatically.
- Interpret step-wise outputs for better understanding of database operations.