

## EXPERIMENT 4

**Student Name:** Vinay Tiwari

**UID:** 24BDA70919

**Branch:** AIT-CSE

**Section/Group:** 24\_AIT\_KRG\_G2

**Semester:** 4

**Subject Name:** Database Management System

**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 schemaViolations.
5. Update approval\_status based on violation count using a CASE statement.
6. Display the updated schemaViolations 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 schemaViolations 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 schemaViolations 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

- schemaViolations 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

[ SQL Worksheet ]\* ▾      Aa 

```
1 CREATE TABLE schemaViolations (
2   id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
3   schema_name VARCHAR2(50),
4   violation_count NUMBER
5 );
```

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



```
schema_name VARCHAR2(50),
violation_count NUMBER...
```

Show more...

Table SCHEMA\_VIOLATIONS created.

Elapsed: 00:00:00.017

## EXPERIMENT 4

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

[ SQL Worksheet ]\* ▶ ⏪ ⏴ ⏵ ⏷ Aa ⏹

```
1 INSERT INTO schemaViolations (schema_name, violation_count) VALUES ('Finance', 0);
2 INSERT INTO schemaViolations (schema_name, violation_count) VALUES ('HR', 2);
3 INSERT INTO schemaViolations (schema_name, violation_count) VALUES ('Sales', 5);
4 INSERT INTO schemaViolations (schema_name, violation_count) VALUES ('Security', 9);
5 INSERT INTO schemaViolations (schema_name, violation_count) VALUES ('Admin', 1);
```

Query result    Script output    DBMS output    Explain Plan    SQL history

⠼ ⏹

```
SQL> INSERT INTO schema_violations (schema_name, violation_count) VALUES ('Admin', 1)
```

1 row inserted.

Elapsed: 00:00:00.001

## EXPERIMENT 4

### Step 3 – Violation status of each department

FreeSQL Worksheet Library 23a

[ SQL Worksheet ]\* ▶ ⌂ ⌂ ⌂ Aa ⌂

```
1  SELECT *
2    schema_name,
3    violation_count,
4    CASE
5      WHEN violation_count = 0 THEN 'No Violation'
6      WHEN violation_count BETWEEN 1 AND 3 THEN 'Minor Violation'
7      WHEN violation_count BETWEEN 4 AND 7 THEN 'Moderate Violation'
8      ELSE 'Critical Violation'
9    END AS violation_status
10   FROM schemaViolations;
```

Query result Script output DBMS output Explain Plan SQL history

Download Execution time: 0.073 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

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

[ SQL Worksheet ]\* ▶ ⏪ ⏴ ⏵ ⏷ Aa ⌂

```
1 -- Add approval_status column
2 ALTER TABLE schemaViolations ADD (approval_status VARCHAR2(20));
```

Query result    Script output    DBMS output    Explain Pl | core.util.apex\_layout.resize

✖ ↴

SQL> ALTER TABLE schemaViolations ADD (approval\_status VARCHAR2(20))

Table SCHEMA\_VIOLATIONS altered.

Elapsed: 00:00:00.025

## EXPERIMENT 4

Step 5 – Update approval\_status based on violation\_count

[ SQL Worksheet ]\* ▾ ▶ ⏪ ⏩ ⏴ ⏵ Aa └

```
1 UPDATE schemaViolations
2 SET approval_status =
3 CASE
4 WHEN violation_count = 0 THEN 'Approved'
5 WHEN violation_count BETWEEN 1 AND 5 THEN 'Needs Review'
6 ELSE 'Rejected'
7 END;
```

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

trash download

```
CASE
    WHEN violation_count = 0 THEN 'Approved'...
Show more...
```

5 rows updated.

Elapsed: 00:00:00.005

## EXPERIMENT 4

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

[ SQL Worksheet ]\* ▶ ⌂ ⌂ ⌂ Aa └

```

1  -- View updated table
2  SELECT * FROM schemaViolations;

```

Query result    Script output    DBMS output    Explain Plan    SQL history

Download ▾ Execution time: 0.004 seconds

ID	SCHEMA_NAME	VIOLATION_COUNT	APPROVAL_STATUS
1	Finance	0	Approved
2	HR	2	Needs Review
3	Sales	5	Needs Review
4	Security	9	Rejected

About Oracle | Contact Us | Legal Notices | Terms and Conditions | Your Privacy Rights | Delete Your FreeSQL Account

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

Output:

## EXPERIMENT 4

[ SQL Worksheet ]\* ▾



Aa



```
1 CREATE TABLE students (
2   name VARCHAR2(50),
3   marks NUMBER
4 );
```

Query result

Script output

DBMS output

Explain Plan

SQL history



```
name VARCHAR2(50),
marks NUMBER
)
```

Table STUDENTS created.

Elapsed: 00:00:00.011

## EXPERIMENT 4

Step 8 – Create students table

[ SQL Worksheet ]\* ▶ ⏪ ⏴ ⏵ Aa ⌂

```
1 CREATE TABLE students (
2   name VARCHAR2(50),
3   marks NUMBER
4 );
```

Query result    Script output    DBMS output    Explain Plan    SQL history

name VARCHAR2(50),  
marks NUMBER  
)

Table STUDENTS created.  
Elapsed: 00:00:00.011

Step 9 – Insert student data

Step 10 – Student grades using CASE statement

## EXPERIMENT 4

-- Insert student data

```
INSERT INTO students (name, marks) VALUES ('Meehul', 92);  
INSERT INTO students (name, marks) VALUES ('Dibyendu', 75);  
INSERT INTO students (name, marks) VALUES ('Vinay', 61);  
INSERT INTO students (name, marks) VALUES ('Kaif', 48);
```

**COMMIT;**

FreeSQL > Worksheet Library

[ SQL Worksheet ]\* ▶ ⌂ ⌂ Aa ⌂

```
9 -- Select students with grades
10 SELECT *
11   name,
12   marks,
13   CASE
14     WHEN marks >= 90 THEN 'A Grade'
15     WHEN marks >= 70 THEN 'B Grade'
16     WHEN marks >= 50 THEN 'C Grade'
17     ELSE 'Fail'
18   END AS grade
19 FROM students;
```

Query result Script output DBMS output Explain Plan SQL history

>Delete Download Execution time: 0.078 seconds

	NAME	MARKS	GRADE
1	Jay	92	A Grade
2	Sam	75	B Grade
3	sahil	61	C Grade
4	Dwayne	40	Fail

About Oracle Contact Us Legal Notices Terms and Conditions Your Privacy Rights Delete Your FreeSQL

name marks grade

Jay 92 A Grade

Sam 75 B Grade

Sahil 61 C Grade

## EXPERIMENT 4

name marks grade

Pranav 48 Fail

Step 11 – Schema violations ordered by severity

[ SQL Worksheet ]\* ▶ ⏪ ⏴ ⏵ Aa ┌

```
-- Order schemaViolations by severity
SELECT schema_name, violation_count
FROM schemaViolations
ORDER BY
CASE
WHEN violation_count = 0 THEN 1
WHEN violation_count BETWEEN 1 AND 3 THEN 2
WHEN violation_count BETWEEN 4 AND 7 THEN 3
ELSE 4
END;
```

Query result    Script output    DBMS output    Explain Plan    SQL history

trash info Download ▾ Execution time: 0.014 seconds

	SCHEMA_NAME	VIOLATION_COUNT
1	Finance	0
2	HR	2
3	Admin	1
4	Sales	5

schema\_name violation\_count

Finance 0

HR 2

Admin 1

## EXPERIMENT 4

schema\_name violation\_count

Sales 5

Security 9

---

### Learning Outcome

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

- 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.