Ex.No.: 4

WORKING WITH CONSTRAINTS

1) Add a table-level PRIMARY KEY constraint to the EMP table on the ID column.The constraint should be named at creation. Name the constraint my_emp_id_pk.

```
alter table EMP1
add constraint my_emp_id_pk PRIMARY KEY(ID);
```

2) Create a PRIMAY KEY constraint to the DEPT table using the ID colum. The constraint should be named at creation. Name the constraint my_dept_id_pk.

```
alter table DEPT
add constraint my_dept_id_pk PRIMARY KEY(ID);
```

3) Add a column DEPT_ID to the EMP table. Add a foreign key reference on the EMP table that ensures that the employee is not assigned to nonexistent deparment. Name the constraint my_emp_dept_id_fk.

```
alter table emp
add DEPT_ID  Numbe(10);

alter table emp
add constraint my_emp_dept_id_fk FOREIGN KEY(DEPT_ID) references dept(ID);
```

4) Modify the EMP table. Add a COMMISSION column of NUMBER data type, precision 2, scale 2. Add a constraint to the commission column that ensures that a commission value is greater than zero.

```
alter table emp
add COMMISSION Number(2,2);

alter table emp
add CONSTRAINT commission_gt_zero CHECK(COMMISSION > 0);
```

Ex.No.: 5		CREATING VIEWS
Date:	23/08/2024	

1) Create a view called EMPLOYEE_VU based on the employee numbers, employee names and department numbers from the EMPLOYEES table. Change the heading for the employee name to EMPLOYEE.

```
create view EMPLOYEE_VU as
select employee_id , first_name || ' ' || last_name as "EMPLOYEE", department_id
from employees;
```

2) Display the contents of the EMPLOYEES_VU view.

```
select * from EMPLOYEE_VU;
```

Results | Explain | Describe | Saved SQL | History

EMPLOYEE_ID	EMPLOYEE	DEPARTMENT_ID
1	Justin Beiber	10
2	Emma Stone	15
3	Robert Downey	40
4	Scarlett Austin	45
5	Chris Evans	55
6	Mark Ruffalo	40
7	Chris Hemsworth	65
8	Jeremy Austin	70
9	Tom Holland	50

3) Select the view name and text from the USER_VIEWS data dictionary views.

```
select VIEW_NAME, TEXT
from USER_VIEWS
where VIEW_NAME = 'EMPLOYEE_VU';
```

VIEW_NAME	TEXT
EMPLOYEE_VU	select employee_id , first_name || ' ' || last_name as "EMPLOYEE", department_id from employees

1 rows returned in 0.04 seconds Download

4) Using your EMPLOYEES_VU view, enter a query to display all employees names and Department.

```
SELECT employee, department_id
```

FROM EMPLOYEE_VU;

EMPLOYEE	DEPARTMENT_ID
Emma Stone	15
Paul Rudd	30
Brie Larson	35
Elizabeth Olsen	90
Cate Austin	55
Jeff Goldblum	75
Robert Downey	40
Karen Gillan	95
Anthony Mackie	30
Sebastian Stan	75

More than 10 rows available. Increase rows selector to view more rows.

5) Create a view named DEPT50 that contains the employee number, employee last names and department numbers for all employees in department 50.Label the view columns EMPNO, EMPLOYEE and DEPTNO. Do not allow an employee to be reassigned to another department through the view.

```
CREATE VIEW DEPT50 AS
SELECT employee_id AS EMPNO,
      employee AS EMPLOYEE,
      department_id AS DEPTNO
FROM EMPLOYEE_VU
WHERE department_id = 50
WITH READ ONLY;
```

EMPNO	EMPLOYEE	DEPTNO
9	Tom Holland	50
13	Chris Austin	50
23	Benedict Cumberbatch	50

3 rows returned in 0.01 seconds Download

6) Display the structure and contents of the DEPT50 view.

Desc dept50;

Results Explain **Describe** **Saved SQL** History

Object Type **VIEW** Object **DEPT50**

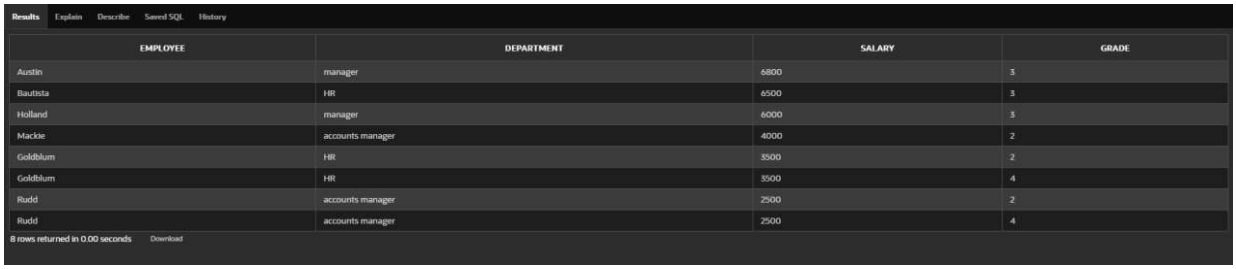
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT50	EMPNO	NUMBER	-	6	0	-	-	-	-
	EMPLOYEE	VARCHAR2	46	-	-	-	✓	-	-
	DEPTNO	NUMBER	-	4	0	-	✓	-	-

7) Attempt to reassign Matos to department 80.

```
UPDATE EMPLOYEES
SET department_id = 80
WHERE first_name = 'Matos';
```

8) Create a view called SALARY_VU based on the employee last names, department names, salaries, and salary grades for all employees. Use the Employees, DEPARTMENTS and JOB_GRADE tables. Label the column Employee, Department, salary, and Grade respectively.

```
CREATE VIEW SALARY_VU AS
SELECT e.last_name AS Employee,
      d.dept_name AS Department,
      e.salary AS Salary,
      j.grade_level AS Grade
FROM EMPLOYEES e
JOIN DEPARTMENT d
ON e.department_id = d.dept_id
JOIN JOB_GRADE j
ON e.salary BETWEEN j.lowest_sal AND j.highest_sal;
```



Results | Explain | Describe | Saved SQL | History

EMPLOYEE	DEPARTMENT	SALARY	GRADE
Austin	manager	6800	3
Bautista	HR	6500	3
Holland	manager	6000	3
Mackie	accounts manager	4000	2
Goldblum	HR	3500	2
Goldblum	HR	3500	4
Rudd	accounts manager	2500	2
Rudd	accounts manager	2500	4

8 rows returned in 0.00 seconds Download