Prashant Bharti Roll No. 202251102

Indian Institute of Information Technology Vadodara

Database Management Systems Laboratory Assignment

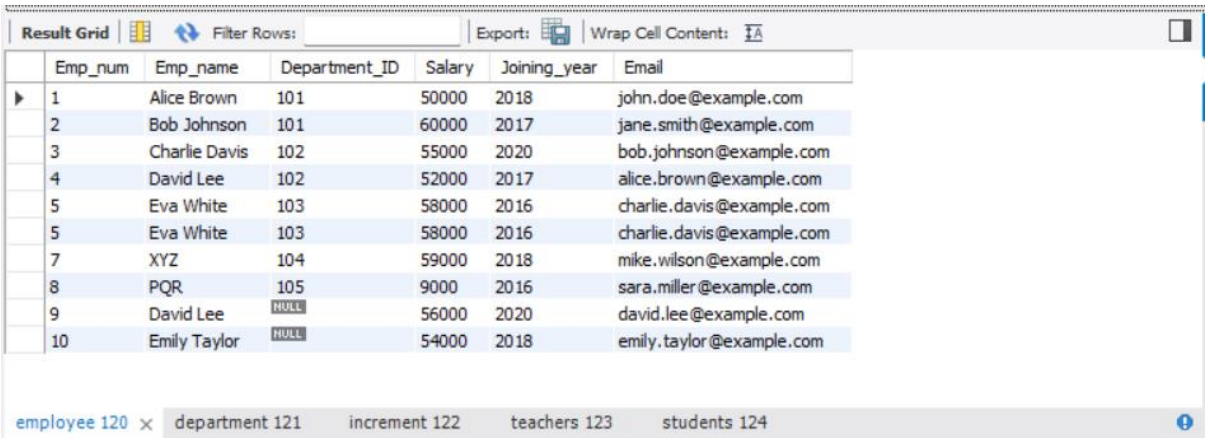
Week 4

-by PRASHANT BHARTI

(202251102)

Consider the following tables.

1. Employee (Emp_num, Emp_name, Department_ID, Salary, Joining_year, Email)

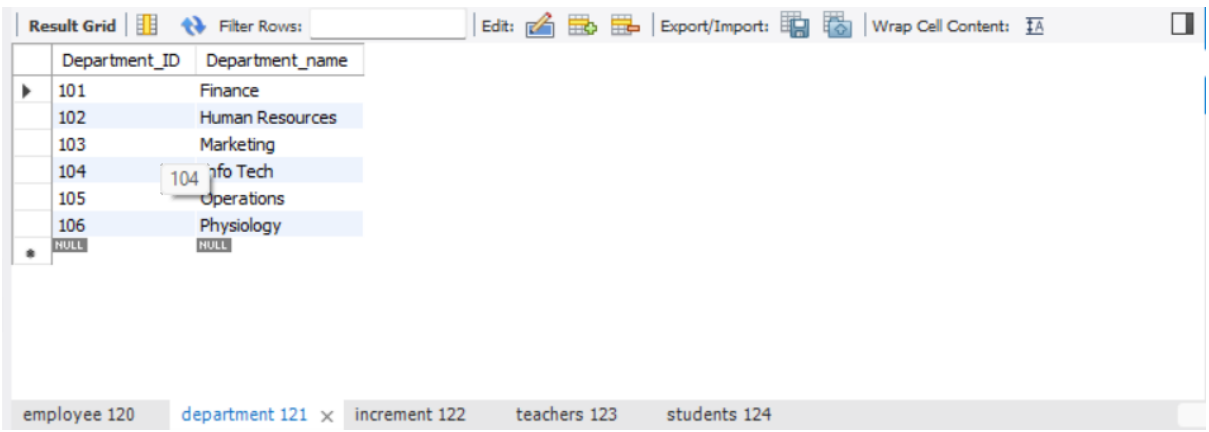


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
1	Alice Brown	101	50000	2018	john.doe@example.com
2	Bob Johnson	101	60000	2017	jane.smith@example.com
3	Charlie Davis	102	55000	2020	bob.johnson@example.com
4	David Lee	102	52000	2017	alice.brown@example.com
5	Eva White	103	58000	2016	charlie.davis@example.com
5	Eva White	103	58000	2016	charlie.davis@example.com
7	XYZ	104	59000	2018	mike.wilson@example.com
8	PQR	105	9000	2016	sara.miller@example.com
9	David Lee	NULL	56000	2020	david.lee@example.com
10	Emily Taylor	NULL	54000	2018	emily.taylor@example.com

employee 120 × | department 121 | increment 122 | teachers 123 | students 124

2. Department (Department_ID, Department_name)



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Department_ID	Department_name
101	Finance
102	Human Resources
103	Marketing
104	nfo Tech
105	Operations
106	Physiology
NULL	NULL

104

employee 120 | department 121 × | increment 122 | teachers 123 | students 124

3. Increment (Emp_name, Salary_Increment, Emp_num)

Emp_name	Salary_Increment	Emp_num
Alice Brown	2000	1
Bob Johnson	2500	2
Charlie Davis	3000	3
David Lee	2500	4
Eva White	3000	5
Frank Wilson	2000	6
Grace Turner	3000	7
Eva White	3000	5
Frank Wilson	2000	6
Jack Taylor	2300	10

employee 120 department 121 increment 122 × teachers 123 students 124

4. Teachers (Teacher_ID, Teacher_name)

Result Grid Filter Rows: Export: Wrap Cell Content:

Teacher_ID	Teacher_name
1	John Smith
2	Emily Johnson
3	Michael Davis
4	Laura Wilson
5	Laura Wilson
6	Emma White
7	Emma White
8	Olivia Miller
9	Daniel Brown
10	Sophia Taylor
1	Mr. Anderson
2	Ms. Davis

employee 120 department 121 increment 122 teachers 123 × students 124

5. Students (Std_ID, Teacher_ID)

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:

Std_ID	Teacher_ID
100	0
101	1
102	2
103	3
104	4
105	5
106	6
107	7
108	8
NULL	NULL

employee 120 department 121 increment 122 teachers 123 students 124 ×

Creation:

```sql
create database week4;
use week4;
drop database week4;

create table Employee(
Emp_num int,
Emp_name varchar(20),
Department_ID int,
Salary int,
Joining_year int,
Email varchar(50)
);
create table Department(
Department_ID int,
Department_name varchar(20),
primary key (Department_ID)
);
create table Increment (
Emp_name varchar(20),
Salary_Increment int,
Emp_num int
);
create table Teachers (
Teacher_ID int,
Teacher_name varchar(20)
);
create table Students (
Std_ID int,
Teacher_ID int,
primary key (Std_ID)
);
```

Perform the following Queries.

1. Find the Second Highest Salary of an Employee.

```sql
-- 1 Find the Second Highest Salary of an Employee.
select *
from (select * from Employee
      order by salary desc
      limit 2 )  as e
order by salary asc
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
7	XYZ	104	59000	2018	mike.wilson@example.com

2. Find the duplicate rows in the table named Teachers.

```sql
-- 2 Find the duplicate rows in the table named Teachers.
select * from  teachers
where teachers.teacher_name in (select teacher_name
from teachers
group by teacher_name
having  count(teacher_name)>1);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Teacher_ID	Teacher_name
4	Laura Wilson
5	Laura Wilson
6	Emma White
7	Emma White

3. Fetch the monthly Salary of the Employee if the annual salary is given.

```sql
-- 3 Fetch the monthly Salary of the Employee if the annual salary is given.
SET SQL_SAFE_UPDATES =0;
UPDATE Employee
set salary=salary/12;

select  Emp_num,Emp_name,salary as monthly_salary
from employee;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	monthly_salary
1	Alice Brown	4167
2	Bob Johnson	5000
3	Charlie Davis	4583
4	David Lee	4333
5	Eva White	4833
5	Eva White	4833
7	XYZ	4917
8	PQR	750
9	David Lee	4667
10	Emily Taylor	4500

employee 181

Output

Action Output

#	Time	Action	Message	Duration / Fet
292	20:40:58	UPDATE Employee set salary=salary/12	10 row(s) affected Rows matched: 10 Changed: 10 War...	0.000 sec
293	20:41:01	select Emp_num,Emp_name,salary as monthly_salary fro...	10 row(s) returned	0.000 sec / 0.

4. Fetch the first record from the Employee table.

```sql
-- 4 Fetch the first record from the Employee table.
select *
from Employee
limit 1 ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Teacher_ID	Teacher_name
4	Laura Wilson
5	Laura Wilson
6	Emma White
7	Emma White
4	Mrs. Johnson
6	Mrs. Johnson
7	Mr. Miller
8	Mr. Miller

teachers 126

5. Fetch the last record from the Department table. (EXCEPT works even it shows error)

```sql
-- 5 Fetch the last record from the Department table.
select *
from Department
order by Department_ID desc
limit 1;

 -- OR

 -- 5 Fetch the last record from the Department table.
select *
from Department
 except (select * from Department
        limit 5
           ) ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Department_ID	Department_name
106	Physiology

6. Display the first 5 Records from the Employee table.

```sql
-- 6 Display the first 5 Records from the Employee table.
select *
from Employee
      limit 5 ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
1	Alice Brown	101	50000	2018	john.doe@example.com
2	Bob Johnson	101	60000	2017	jane.smith@example.com
3	Charlie Davis	102	55000	2020	bob.johnson@example.com
4	David Lee	102	52000	2017	alice.brown@example.com
5	Eva White	103	58000	2016	charlie.davis@example.com

Employee 129

7. Get 3 Highest salaries records from the Employee table.

```sql
-- 7 Get 3 Highest salaries records from the Employee table.
select *
from employee
order by salary desc
limit 3 ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
2	Bob Johnson	101	60000	2017	jane.smith@example.com
7	XYZ	104	59000	2018	mike.wilson@example.com
5	Eva White	103	58000	2016	charlie.davis@example.com

8. Create a table with the same structure as the Employee table.

```sql
-- 8 Create a table with the same structure as the Employee table.
CREATE TABLE NewEmployees AS
SELECT *
FROM Employee
WHERE 1 = 0;
select * from newemployees;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email

9. Display the first 50% of records from the Employee table.

```sql
-- 9
SELECT *
FROM Employee
where emp_num < (select max(emp_num)/2 from employee)
ORDER BY Emp_num;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
1	Alice Brown	101	50000	2018	john.doe@example.com
2	Bob Johnson	101	60000	2017	jane.smith@example.com
3	Charlie Davis	102	55000	2020	bob.johnson@example.com
4	David Lee	102	52000	2017	alice.brown@example.com

10. Fetch only common records between tables Teachers and Students.

```
203     -- 10  Fetch only common records between tables Teachers and Students.
204 •   select *
205     from teachers
206     inner join students
207     on teachers.Teacher_ID=students.Teacher_ID;
208
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Teacher_ID	Teacher_name	Std_ID	Teacher_ID
1	John Smith	101	1
2	Emily Johnson	102	2
3	Michael Davis	103	3
4	Laura Wilson	104	4
5	Laura Wilson	105	5
6	Emma White	106	6
7	Emma White	107	7
8	Olivia Miller	108	8

11. Get information about Employees where an Employee is not assigned to the department.

```
209     -- 11
210 •   select *
211     from Employee
212     where department_id is null;
213
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
9	David Lee	NULL	56000	2020	david.lee@example.com
10	Emily Taylor	NULL	54000	2018	emily.taylor@example.com

12. Get distinct records from the Increment table without using distinct keyword.

```
214     -- 12. Get distinct records from the Increment table without using distinct keyword.
215 •   select emp_num, Salary_Increment , emp_name
216     from increment
217     group by emp_num, Salary_Increment , emp_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_num	Salary_Increment	emp_name
1	2000	Alice Brown
2	2500	Bob Johnson
3	3000	Charlie Davis
4	2500	David Lee
5	3000	Eva White
6	2000	Frank Wilson
7	3000	Grace Turner
10	2300	Jack Taylor

13. Select all records from the Employee table whose name is ‘XYZ and ‘PQR’.

```
220     -- 13 Select all records from the Employee table whose name is 'XYZ and 'PQR'.
221 •   select *
222     from Employee
223     where Emp_name in ("XYZ","PQR");
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
7	XYZ	104	59000	2018	mike.wilson@example.com
8	PQR	105	9000	2016	sara.miller@example.com

14. Select all records from the Employee table where the name is not in 'XYZ' and 'PQR'.

```
225     -- 14 Select all records from the Employee table where the name is not in 'XYZ' and 'PQR'.
226 •   select *
227     from Employee
228     where Emp_name NOT in ("XYZ","PQR");
229
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
7	XYZ	104	59000	2018	mike.wilson@example.com
8	PQR	105	9000	2016	sara.miller@example.com

15. Write SQL query for the below scenario -

I/p: ABCDE

O/p:

A

B

C

D

E

```
230     -- 15 I/p: ABCDE
231     -- O/p:
232     -- A
233     -- B
234     -- C
235     -- D
236     -- E
237
238 •   SELECT SUBSTRING('ABCDE' FROM 1 FOR 1) AS Single_char
239     UNION ALL
240     SELECT SUBSTRING('ABCDE' FROM 2 FOR 1)
241     UNION ALL
242     SELECT SUBSTRING('ABCDE' FROM 3 FOR 1)
243     UNION ALL
244     SELECT SUBSTRING('ABCDE' FROM 4 FOR 1)
245     UNION ALL
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Single_char
A
B
C
D
E

16. Fetch all the records from employees whose joining year is 2017.

```
251     -- 16 Fetch all the records from employees whose joining year is 2017.
252 •   select *
253     from Employee
254     where Joining_year = 2017;
255
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
2	Bob Johnson	101	60000	2017	jane.smith@example.com
4	David Lee	102	52000	2017	alice.brown@example.com

17. Find the maximum salary offered by each department.

```
256     -- 17 Find the maximum salary offered by each department.
257 •   select  department_id, max(salary)
258     from Employee
259     group by department_id ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

department_id	max(salary)
101	60000
102	55000
103	58000
104	59000
105	9000
NULL	56000

18. Display the name of employees who joined in 2016 and whose salary is greater

than 10000.

```
260
261     -- 18 Display the name of employees who joined in 2016 and whose salary is greater than 10000.
262     select *
263     from Employee
264     where Employee.Joining_year = 2016
265           and Employee.salary>10000;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
5	Eva White	103	58000	2016	charlie.davis@example.com
5	Eva White	103	58000	2016	charlie.davis@example.com

19. Display the following using query -

O/p:

*

**

```
267     -- 19 Display the following using query -
268     -- O/p:
269     -- *
270     -- **
271     -- ***
272     SELECT REPEAT('*', numbers.n) AS Output
273     FROM (
274       SELECT ROW_NUMBER() OVER () AS n
275       FROM information_schema.columns
276     ) AS numbers
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Output
*
**
***

20. Add the email validation using only one query in the employee table.

```
280     -- 20 Add the email validation using only one query in the employee table.
281     update employee
282     set email = "invalid type"
283     where  email not like  "%@example.com" ;
284     select * from employee;
285
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
1	Alice Brown	101	50000	2018	john.doe@example.com
2	Bob Johnson	101	60000	2017	jane.smith@example.com
3	Charlie Davis	102	55000	2020	bob.johnson@example.com
4	David Lee	102	52000	2017	alice.brown@example.com
5	Eva White	103	58000	2016	charlie.davis@example.com
5	Eva White	103	58000	2016	charlie.davis@example.com
7	XYZ	104	59000	2018	mike.wilson@example.com
8	PQR	105	9000	2016	sara.miller@example.com
9	David Lee	NULL	56000	2020	david.lee@example.com
10	Emily Taylor	NULL	54000	2018	emily.taylor@example.com

employee 164

21. Display 1 to 100 Numbers with a query.

```
286  -- 21 Display 1 to 100 Numbers with a query.
287  WITH RECURSIVE NumberSequence AS (
288    SELECT 1 AS Number
289    UNION ALL
290    SELECT Number + 1
291    FROM NumberSequence
292    WHERE Number < 100
293  )
294  SELECT Number
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Number
1
2
3
4
5
6
7
8
9
10

Result 165

90
91
92
93
94
95
96
97
98
99
100

Result 180

22. Find the count of duplicate rows from the table employee.

```
299  -- 22  Find the count of duplicate rows from the table employee.
300  SELECT Emp_num, Emp_name, Department_ID, Salary, Joining_year, Email, COUNT(*) AS DuplicateCount
301  FROM Employee
302  GROUP BY Emp_num, Emp_name, Department_ID, Salary, Joining_year, Email
303  HAVING COUNT(*) > 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email	DuplicateCount
5	Eva White	103	58000	2016	charlie.davis@example.com	2

23. Remove duplicate rows from the table employee?

```
305  -- 23 Remove duplicate rows from the table employee?
306  select distinct *
307  from employee;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email
1	Alice Brown	101	50000	2018	john.doe@example.com
2	Bob Johnson	101	60000	2017	jane.smith@example.com
3	Charlie Davis	102	55000	2020	bob.johnson@example.com
4	David Lee	102	52000	2017	alice.brown@example.com
5	Eva White	103	58000	2016	charlie.davis@example.com
7	XYZ	104	59000	2018	mike.wilson@example.com
8	PQR	105	9000	2016	sara.miller@example.com
9	David Lee	NULL	56000	2020	david.lee@example.com
10	Emily Taylor	NULL	54000	2018	emily.taylor@example.com

24. Apply the following Query on the Teachers and Students Table -

- Apply Inner Join Query and write the concept of it.

```
309  -- 24
310  -- INNER JOIN is used to concatenate that rows of two tables that have common attribute value
311  select *
312  from teachers
313  inner join students
314  on teachers.Teacher_ID=students.Teacher_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Teacher_ID	Teacher_name	Std_ID	Teacher_ID
1	John Smith	101	1
2	Emily Johnson	102	2
3	Michael Davis	103	3
4	Laura Wilson	104	4
5	Laura Wilson	105	5
6	Emma White	106	6
7	Emma White	107	7
8	Olivia Miller	108	8

Result 168

- Apply Left Outer Join Query. Write the concept of it.

```
316  -- LEFT OUTER JOIN is used concatenate all values of left table to the common values of shared
317  -- atrribute of right table. Values of right table not common with the shared attribute of left
318  -- one are taken null.
319  select *
320  from teachers
321  left join students
322  on teachers.Teacher_ID=students.Teacher_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Teacher_ID	Teacher_name	Std_ID	Teacher_ID
1	John Smith	101	1
2	Emily Johnson	102	2
3	Michael Davis	103	3
4	Laura Wilson	104	4
5	Laura Wilson	105	5
6	Emma White	106	6
7	Emma White	107	7
8	Olivia Miller	108	8
9	Daniel Brown	NULL	NULL
10	Sophia Taylor	NULL	NULL

- Apply Right Outer Join Query. Write the concept of it.

```
324  -- RIGHT OUTER JOIN is used concatenate all values of right table to the common values of shared
325  -- atrribute of left table. Values of left table not common with the shared attribute of right
326  -- one are taken null.
327  select *
328  from teachers
329  right join students
330  on teachers.Teacher_ID=students.Teacher_ID;
331
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Teacher_ID	Teacher_name	Std_ID	Teacher_ID
NULL	NULL	100	0
1	John Smith	101	1
2	Emily Johnson	102	2
3	Michael Davis	103	3
4	Laura Wilson	104	4
5	Laura Wilson	105	5
6	Emma White	106	6
7	Emma White	107	7
8	Olivia Miller	108	8

- Apply Full Outer Join Query. Write the concept of it.

```sql
-- FULL JOIN concatenate all rows of right and left table on basis of shared attribute, whether common
-- or not common.
select *
from teachers
left join students
on teachers.Teacher_ID=students.Teacher_ID
union
select *
from teachers
right join students
on teachers.Teacher_ID=students.Teacher_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Teacher_ID	Teacher_name	Std_ID	Teacher_ID
1	John Smith	101	1
2	Emily Johnson	102	2
3	Michael Davis	103	3
4	Laura Wilson	104	4
5	Laura Wilson	105	5
6	Emma White	106	6
7	Emma White	107	7
8	Olivia Miller	108	8
9	Daniel Brown	NULL	NULL
10	Sophia Taylor	NULL	NULL
NULL	NULL	100	0

25. Display the allocated departments with its department id.

```sql
-- 25 Display the allocated departments with its department id.(common department)
select department.department_name , department.Department_ID
from department
inner join employee
on employee.Department_ID = department.Department_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

department_name	Department_ID
Finance	101
Finance	101
Human Resources	102
Human Resources	102
Marketing	103
Marketing	103
Info Tech	104
Operations	105

26. Query to fetch employees associated with department names.

```sql
-- 26 Query to fetch employees associated with department names.
select Emp_name , Department_name, department.department_id
from employee
left join department
on employee.Department_ID = department.Department_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_name	Department_name	department_id
Alice Brown	Finance	101
Bob Johnson	Finance	101
Charlie Davis	Human Resources	102
David Lee	Human Resources	102
Eva White	Marketing	103
Eva White	Marketing	103
XYZ	Info Tech	104
PQR	Operations	105
David Lee	NULL	NULL
Emily Taylor	NULL	NULL

27. Query to fetch all departments with its associated employees.

```sql
-- 27 Query to fetch all departments with its associated employees.
select department.department_name,department.department_id,emp_name
from department
left join employee
on department.department_id=employee.department_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

department_name	department_id	emp_name
Finance	101	Bob Johnson
Finance	101	Alice Brown
Human Resources	102	David Lee
Human Resources	102	Charlie Davis
Marketing	103	Eva White
Marketing	103	Eva White
Info Tech	104	XYZ
Operations	105	PQR
Physiology	106	NULL

28. Joins between Employee, Department, and Increment Tables.

```sql
-- 28 Joins between Employee, Department, and Increment Tables.
select *
from employee
left join department
on department.department_id=employee.department_id
left join increment
on employee.emp_num=increment.emp_num
union
select *
from employee
right join department
on department.department_id=employee.department_id
right join increment
on employee.emp_num=increment.emp_num
;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Emp_num	Emp_name	Department_ID	Salary	Joining_year	Email	Department_ID	Department_name	Emp_name	Salary_Increment	Emp_num
1	Alice Brown	101	50000	2018	john.doe@example.com	101	Finance	Alice Brown	2000	1
2	Bob Johnson	101	60000	2017	jane.smith@example.com	101	Finance	Bob Johnson	2500	2
3	Charlie Davis	102	55000	2020	bob.johnson@example.com	102	Human Resources	Charlie Davis	3000	3
4	David Lee	102	52000	2017	alice.brown@example.com	102	Human Resources	David Lee	2500	4
5	Eva White	103	58000	2016	charlie.davis@example.com	103	Marketing	Eva White	3000	5
7	XYZ	104	59000	2018	mike.wilson@example.com	104	Info Tech	Grace Turner	3000	7
8	PQR	105	9000	2016	sara.miller@example.com	105	Operations	NULL	NULL	NULL
9	David Lee	NULL	56000	2020	david.lee@example.com	NULL	NULL	NULL	NULL	NULL
10	Emily Taylor	NULL	54000	2018	emily.taylor@example.com	NULL	NULL	Jack Taylor	2300	10
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Frank Wilson	2000	6
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Jack Taylor	2300	10

Result 179

THANK YOU

- - - [END] - - -