Continuing with the previous set of exercises, this time using JOINS.

1. List employee names and their department names; only include employees who are currently assigned to a department.

First, let's take a look at the structure of our two tables, employees and departments:

```
MariaDB [mynewpaltz]> select * from employees;
+----+---------+----------+---------------+------------+
| id | name    | salary   | department_id | manager_id |
+----+---------+----------+---------------+------------+
|  1 | Alice   | 60000.00 |             2 |         10 |
|  2 | Bob     | 55000.00 |             2 |         10 |
|  3 | Charlie | 50000.00 |             1 |       NULL |
|  4 | David   | 75000.00 |             4 |          5 |
|  5 | Emma    | 90000.00 |             4 |       NULL |
|  6 | Frank   | 47000.00 |             1 |          3 |
|  7 | Grace   | 52000.00 |             3 |       NULL |
```

We can see employees has employee ids, names, salary, department_id, and manager_id.

```
MariaDB [mynewpaltz]> select * from departments;
+----+-----------------+------------+
| id | department_name  | manager_id |
+----+-----------------+------------+
|  1 | HR              |          3 |
|  2 | Engineering     |         10 |
|  3 | Marketing       |          7 |
|  4 | Finance         |          5 |
|  5 | Sales           |          8 |
|  6 | IT              |         12 |
|  7 | Customer Support |         14 |
+----+-----------------+------------+
```

And departments has id, department_name, and manager_id.

We want to list all employees currently assigned to a department. By using an `INNER JOIN`, we can select only the employees who's department id is contained in the departments table.

```
select employees.id, employees.name, departments.department_name from
employees INNER JOIN departments ON
employees.department_id = departments.id;
```

```
+----+---------+-----------------+
| id | name    | department_name |
+----+---------+-----------------+
|  3 | Charlie | HR              |
|  6 | Frank   | HR              |
| 16 | Paul    | HR              |
| 23 | Wendy   | HR              |
|  1 | Alice   | Engineering     |
|  2 | Bob     | Engineering     |
| 10 | Jack    | Engineering     |
| 17 | Quincy  | Engineering     |
| 24 | Xander  | Engineering     |
|  7 | Grace   | Marketing       |
| 15 | Olivia  | Marketing       |
| 22 | Victor  | Marketing       |
|  4 | David   | Finance         |
|  5 | Emma    | Finance         |
| 18 | Rachel  | Finance         |
|  8 | Hank    | Sales           |
|  9 | Ivy     | Sales           |
| 26 | Zach    | Sales           |
| 11 | Karen   | IT              |
| 12 | Laura   | IT              |
| 20 | Tracy   | IT              |
| 13 | Mike    | Customer Support |
| 14 | Nina    | Customer Support |
| 21 | Uma     | Customer Support |
+----+---------+-----------------+
24 rows in set (0.001 sec)
```

2. List employee names and their department names including those who are not assigned a department.

To show this, we can simply alter the query from 1 to use a left join, outputting all employees, even if they aren't in a database. I'm also renaming `employees` to `e` and `departments` to `d`, just to compress the query.

```
SELECT e.id, e.name, d.department_name FROM employees e LEFT JOIN
departments d ON e.department_id = d.id;
```

```
+----+---------+-----------------+
| id | name    | department_name |
+----+---------+-----------------+
|  1 | Alice   | Engineering     |
|  2 | Bob     | Engineering     |
|  3 | Charlie | HR              |
|  4 | David   | Finance         |
|  5 | Emma    | Finance         |
|  6 | Frank   | HR              |
|  7 | Grace   | Marketing       |
|  8 | Hank    | Sales           |
|  9 | Ivy     | Sales           |
| 10 | Jack    | Engineering     |
| 11 | Karen   | IT              |
| 12 | Laura   | IT              |
| 13 | Mike    | Customer Support |
| 14 | Nina    | Customer Support |
| 15 | Olivia  | Marketing       |
| 16 | Paul    | HR              |
| 17 | Quincy  | Engineering     |
| 18 | Rachel  | Finance         |
| 19 | Steve   | NULL            |
| 20 | Tracy   | IT              |
| 21 | Uma     | Customer Support |
| 22 | Victor  | Marketing       |
| 23 | Wendy   | HR              |
| 24 | Xander  | Engineering     |
| 25 | Yvonne  | NULL            |
| 26 | Zach    | Sales           |
+----+---------+-----------------+
26 rows in set (0.001 sec)
```

We can see that the query is successful, outputting `NULL` for employees who do not belong to any department.

3. List all department names and the employees assigned to them. List all departments, even if they
have no employees assigned to them.

For this, we can use a left join like the one from the first problem, but placing the departments table on the left instead.

```
SELECT d.department_name, e.name FROM departments d LEFT JOIN employees e ON
d.id = e.department_id;
```

```
+-----------------+---------+
| department_name | name    |
+-----------------+---------+
| HR              | Charlie |
| HR              | Frank   |
| HR              | Paul    |
| HR              | Wendy   |
| Engineering     | Alice   |
| Engineering     | Bob     |
| Engineering     | Jack    |
| Engineering     | Quincy  |
| Engineering     | Xander  |
| Marketing       | Grace   |
| Marketing       | Olivia  |
| Marketing       | Victor  |
| Finance         | David   |
| Finance         | Emma    |
| Finance         | Rachel  |
| Sales           | Hank    |
| Sales           | Ivy     |
| Sales           | Zach    |
| IT              | Karen   |
| IT              | Laura   |
| IT              | Tracy   |
| Customer Support | Mike   |
| Customer Support | Nina   |
| Customer Support | Uma    |
| IT SECURITY     | NULL    |
| QA              | NULL    |
+-----------------+---------+
26 rows in set (0.001 sec)
```

Here we can see all departments, even those without employees. We also could have altered the statements from 1 and 2 to use a right join.

4. List all employee names and their manager's names. List only those employees who have a manager.

Because we need to reference `name` twice, once for the employee, and once for manager, I'll use a self join.

```
SELECT e.name, a.name as manager FROM employees e INNER JOIN employees a ON e.manager_id = a.id;
```

```
MariaDB [mynewpaltz]> SELECT e.name, a.name as manager FROM employees
+--------+---------+
| name   | manager |
+--------+---------+
| Alice  | Jack    |
| Bob    | Jack    |
| David  | Emma    |
| Frank  | Charlie |
| Ivy    | Hank    |
| Karen  | Laura   |
| Mike   | Nina    |
| Olivia | Grace   |
| Paul   | Charlie |
| Quincy | Jack    |
| Rachel | Emma    |
| Tracy  | Laura   |
| Uma    | Nina    |
| Victor | Grace   |
| Wendy  | Charlie |
| Xander | Jack    |
| Zach   | Hank    |
+--------+---------+
17 rows in set (0.001 sec)
```

Looks great! We can see a list of all employees who have managers, and their respective manager.

5. List employees and their department names

We'll use a left join similar to earlier here:

```
SELECT e.name, d.department_name FROM employees e LEFT JOIN departments d ON e.department_id=d.id;
```

```
+---------+-----------------+
| name    | department_name |
+---------+-----------------+
| Alice   | Engineering     |
| Bob     | Engineering     |
| Charlie | HR              |
| David   | Finance         |
| Emma    | Finance         |
| Frank   | HR              |
| Grace   | Marketing       |
| Hank    | Sales           |
| Ivy     | Sales           |
| Jack    | Engineering     |
| Karen   | IT              |
| Laura   | IT              |
| Mike    | Customer Support |
| Nina    | Customer Support |
| Olivia  | Marketing       |
| Paul    | HR              |
| Quincy  | Engineering     |
| Rachel  | Finance         |
| Steve   | NULL            |
| Tracy   | IT              |
| Uma     | Customer Support |
| Victor  | Marketing       |
| Wendy   | HR              |
| Xander  | Engineering     |
| Yvonne  | NULL            |
| Zach    | Sales           |
+---------+-----------------+
26 rows in set (0.001 sec)
```

6. List name and salary of employee with highest salary

To find the highest salary, we'll use the `MAX` function:

```
SELECT name, MAX(salary) FROM employees;
```

```
MariaDB [mynewpaltz]> SELECT name, MAX(salary) FROM employees;
+-------+-------------+
| name  | MAX(salary) |
+-------+-------------+
| Alice |    90000.00 |
+-------+-------------+
1 row in set (0.001 sec)
```

We can see Alice has the highest salary, of 90000.00.

7. List name, department_name and and salary of employee with highest salary

We'll simply combine the query from 6 with a left join on department id.

```
SELECT name, MAX(salary), department_name FROM employees LEFT JOIN
departments ON employees.department_id=departments.id;
```

```
departments.id;
+-------+-------------+-----------------+
| name  | MAX(salary) | department_name |
+-------+-------------+-----------------+
| Alice |    90000.00 | Engineering     |
+-------+-------------+-----------------+
1 row in set (0.001 sec)
```

Nice, we can see Alice makes 90000.00 and is part of the engineering department.