

Realizar las siguientes consultas a la base de datos MyColinc. Para cada una de las consultas se ha de especificar tanto el código SQL como un pantallazo de los datos obtenidos en la ejecución de la consulta.

1. - Obtener la información de los pedidos realizados por el Empleado con DNI **72850874F**.

```

6 • select p.*
7   from Pedidos as p
8   inner join Empleados as e on (p.cod_empleado = e.cod_empleado)
9   where DNI = '72850874F';

```



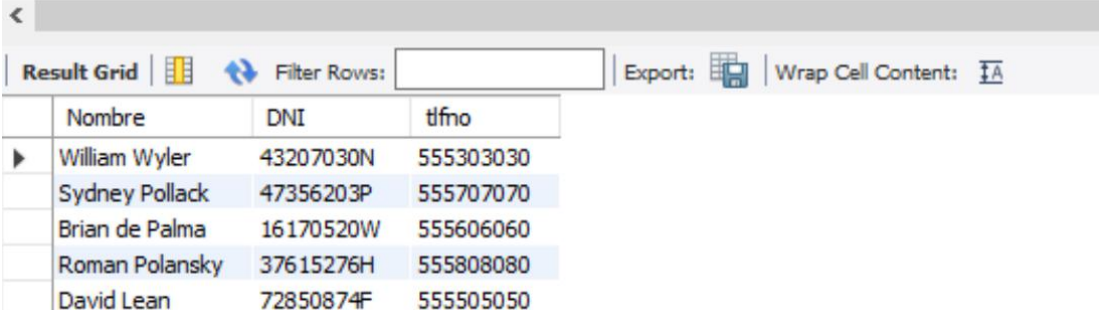
|   | cod_pedido | fecha_pedido | fecha_envio | fecha_entrega | estado   | descripcion | cod_empleado |
|---|------------|--------------|-------------|---------------|----------|-------------|--------------|
| ▶ | P_00002    | 2020-01-01   | 2020-01-02  | NULL          | ENVIADO  | NULL        | 3            |
|   | P_00008    | 2020-01-05   | 2020-01-06  | 2020-01-16    | DEVUELTO | NULL        | 3            |
|   | P_00013    | 2020-01-08   | NULL        | NULL          | ESPERA   | NULL        | 3            |

2.- Obtener el nombre, DNI y teléfono de los empleados con pedidos que aún no han sido enviados.

```

12 • select distinct e.Nombre, e.DNI, e.tlfno
13   from Empleados as e
14   inner join Pedidos as p on (e.cod_empleado = p.cod_empleado)
15   where estado != 'ENVIADO';

```



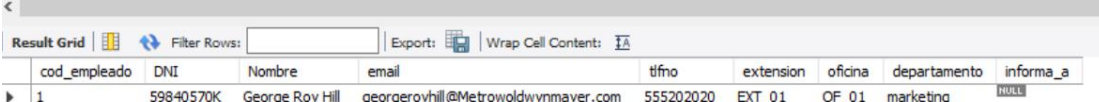
|   | Nombre         | DNI       | tlfno     |
|---|----------------|-----------|-----------|
| ▶ | William Wyler  | 43207030N | 555303030 |
|   | Sydney Pollack | 47356203P | 555707070 |
|   | Brian de Palma | 16170520W | 555606060 |
|   | Roman Polansky | 37615276H | 555808080 |
|   | David Lean     | 72850874F | 555505050 |

3.- Obtener la información de los empleados que no han realizado ningún pedido

```

18 • select e.*
19   from Empleados as e
20   left join Pedidos as p on (e.cod_empleado = p.cod_empleado)
21   where cod_pedido is null;

```



|   | cod_empleado | DNI       | Nombre          | email                               | tlfno     | extension | oficina | departamento | informa_a |
|---|--------------|-----------|-----------------|-------------------------------------|-----------|-----------|---------|--------------|-----------|
| ▶ | 1            | 59840570K | George Roy Hill | georgeroyhill@Metrowoldwynmayer.com | 555202020 | EXT_01    | OF_01   | marketing    | NULL      |

4.- Obtener la información de los pedidos que no tienen ningún empleado asociado.

```
24 • select p.*
25 from Pedidos as p
26 right join Empleados as e on (p.cod_empleado = e.cod_empleado)
27 where p.cod_empleado is null;
```

|   | cod_pedido | fecha_pedido | fecha_envio | fecha_entrega | estado | descripcion | cod_empleado |
|---|------------|--------------|-------------|---------------|--------|-------------|--------------|
| ▶ | NULL       | NULL         | NULL        | NULL          | NULL   | NULL        | NULL         |

5.- Obtener la información del pago relacionado en el pedido que tiene como identificador **P\_00002**.

```
30 • select pa.*
31 from Pagos as pa
32 inner join Pedidos as pe on (pa.cod_pedido = pe.cod_pedido)
33 where pa.cod_pedido = 'P_00002';
```

|   | cod_pago | cod_cliente | cantidad | num_plazos | num_pago | cuantia_plazo | cod_pedido | pago_master |
|---|----------|-------------|----------|------------|----------|---------------|------------|-------------|
| ▶ | 2        | 1           | 1000     | 2          | 1        | 500           | P_00002    | NULL        |

6.- Obtener la información de los pedidos sin pagos asociados.

```
36 • select pe.*
37 from Pedidos as pe
38 left join Pagos as pa on (pe.cod_pedido = pa.cod_pedido)
39 where cod_pago is null;
```

|   | cod_pedido | fecha_pedido | fecha_envio | fecha_entrega | estado  | descripcion | cod_empleado |
|---|------------|--------------|-------------|---------------|---------|-------------|--------------|
| ▶ | P_00011    | 2020-01-07   | 2020-01-08  | NULL          | ENVIADO | NULL        | 6            |
|   | P_00012    | 2020-01-07   | 2020-01-08  | NULL          | ENVIADO | NULL        | 7            |
|   | P_00013    | 2020-01-08   | NULL        | NULL          | ESPERA  | NULL        | 3            |
|   | P_00014    | 2020-01-08   | NULL        | NULL          | ESPERA  | NULL        | 2            |
|   | P_00015    | 2020-01-08   | NULL        | NULL          | ESPERA  | NULL        | 2            |
|   | P_00016    | 2020-01-09   | NULL        | NULL          | ESPERA  | NULL        | 2            |

7.- Obtener el nombre de los empleados con pedidos con pagos asociados. El resultset obtenido debe eliminar repeticiones, además los nombres se deben mostrar de ordenados alfabéticamente de manera descendente.

```
42 • select distinct e.nombre
43 from Empleados as e
44 inner join Pedidos as pe on (e.cod_empleado = pe.cod_empleado)
45 inner join Pagos as pa on (pe.cod_pedido = pa.cod_pedido)
46 where pa.cod_pago is not null
47 order by e.nombre desc;
```

| Result Grid |                  | Filter Rows: | Export: | Wrap Cell Content: |
|-------------|------------------|--------------|---------|--------------------|
|             | nombre           |              |         |                    |
| ▶           | William Wyler    |              |         |                    |
|             | Sydney Pollack   |              |         |                    |
|             | Roman Polansky   |              |         |                    |
|             | David Lean       |              |         |                    |
|             | Cecil B. DeMille |              |         |                    |
|             | Brian de Palma   |              |         |                    |

8.- Obtener el nombre de los empleados con pedidos que no tienen pagos asociados. El resultset obtenido debe eliminar repeticiones, además los nombres se deben mostrar de ordenados alfabéticamente de manera ascendente.




```
50 • select distinct e.nombre
51 from Empleados as e
52 inner join Pedidos as pe on (e.cod_empleado = pe.cod_empleado)
53 left join Pagos as pa on (pe.cod_pedido = pa.cod_pedido)
54 where pa.cod_pago is null
55 order by e.nombre asc;
```

| Result Grid |                | Filter Rows: | Export: | Wrap Cell Content: |
|-------------|----------------|--------------|---------|--------------------|
|             | nombre         |              |         |                    |
| ▶           | Brian de Palma |              |         |                    |
|             | David Lean     |              |         |                    |
|             | Roman Polansky |              |         |                    |
|             | William Wyler  |              |         |                    |



12.- Obtener los pagos aplazados del pedido  
P\_00005.

```
80 • select pa.*
81 from Pagos as pa
82 inner join Pagos as ps on (pa.pago_master = ps.cod_pago)
83 inner join Pedidos as pe on (ps.cod_pedido = pe.cod_pedido)
84 where pe.cod_pedido = 'P_00005';
```

| <  |          |             |          |            |          |               |            |             |
|--|----------|-------------|----------|------------|----------|---------------|------------|-------------|
| Result Grid    Filter Rows: <input type="text"/>   Export:    Wrap Cell Content:  |          |             |          |            |          |               |            |             |
|  | cod_pago | cod_cliente | cantidad | num_plazos | num_pago | cuantia_plazo | cod_pedido | pago_master |
| ▶  | 10       | 3           | 2100.025 | 4          | 1        | 0             | NULL       | 9           |
|  | 11       | 3           | 2100.025 | 4          | 2        | 0             | NULL       | 9           |
|  | 12       | 3           | 2100.025 | 4          | 3        | 0             | NULL       | 9           |
|  | 13       | 3           | 2100.025 | 4          | 4        | 0             | NULL       | 9           |