```
--1. Obtener los nombres y salarios de los empleados con más de 1000 euros de
salario por orden alfabético.
SELECT nombre, ape1, ape2, salario
FROM empleado
WHERE salario > 1000
ORDER BY nombre ASC;

--2. Obtener el nombre de los empleados cuya comisión es superior
-- al 20% de su salario.
SELECT nombre, ape1, comision, salario
FROM empleado
WHERE comision > salario * 0.1; -- PROBAMOS 10% POR QUE CON 20 NO HAY

SELECT nombre, ape1, nvl(comision, 0), salario  -- Para ver los valores si
comision es null sacamos 0 en el listado
FROM empleado
WHERE nvl(comision, 0) > salario * 0.1;
--3. Obtener el código de empleado, código de departamento, nombre y sueldo
--total en pesetas de aquellos empleados cuyo sueldo total (salario más
--comisión) supera los 1800 euros. Presentarlos ordenados por código de
--departamento y dentro de éstos por orden alfabético.
select codemple, coddpto, nombre,
       round((salario + nvl(comision, 0)) / 0.006, 2) AS sueldoTotalPesetas, --
mil pesetas 6 euros
       round((salario+nvl(comision,0))*166.386,2) SueldoTotalPesetas --1 euro,
equivale a 166,386 pesetas
FROM empleado
WHERE salario + nvl(comision, 0) > 1800
ORDER BY coddpto,
         nombre ASC;

--4. Obtener por orden alfabético los nombres de empleados cuyo salario
--igualen o superen en más de un 5% al salario de la empleada ‘MARIA JAZMIN’.
SELECT nombre, ape1, salario
FROM empleado
WHERE salario >= ( SELECT salario * 1.05
                   FROM empleado
                   WHERE ape1 = 'JAZMIN'
                         AND nombre = 'MARIA'
                 )
ORDER BY 2, 1;
--5. Obtener un listado ordenado por años en la empresa con los nombres, y
--apellidos de los empleados y los años de antigüedad en la empresa
SELECT nombre, ape1, ape2, round((sysdate - fechaingreso) / 365.20, 2)       AS
"AntigÃ¼edad",
trunc(months_between(sysdate, fechaingreso) / 12) "Antiguedad"
FROM empleado
ORDER BY 5;

--6. Obtener el nombre de los empleados que trabajan en un departamento con
--presupuesto superior a 50.000 euros. Hay que usar predicado cuantificado:
SELECT nombre,ape1
FROM empleado
WHERE coddpto IN ( SELECT coddpto
```

```
                  FROM dpto
                  WHERE presupuesto > 50000
                );
SELECT nombre,ape1
FROM empleado
WHERE coddpto = SOME ( SELECT coddpto
                       FROM dpto
                       WHERE presupuesto > 50000
                     );


--7. Obtener los nombres y apellidos de empleados que más cobran en la
--empresa. Considerar el salario más la comisión
select
FROM empleado
WHERE nvl(comision, 0) + salario >= ALL ( SELECT nvl(comision, 0) + salario
                                          FROM empleado
                                        );
SELECT nombre, ape1, ape2
FROM empleado
WHERE salario + nvl(comision, 0) = ( SELECT MAX(salario + nvl(comision, 0))
                                     FROM empleado
                                   );

--8. Obtener en orden alfabético los nombres de empleado cuyo salario es
--inferior al mínimo de los empleados del departamento 1.
SELECT ape1, nombre
FROM empleado
WHERE salario < ( SELECT MIN(salario)
                  FROM empleado
                  WHERE coddpto = 1
                )
ORDER BY ape1, nombre;

SELECT ape1, nombre, salario
FROM empleado
WHERE salario < all ( SELECT salario
                  FROM empleado
                  WHERE coddpto = 1
                )
ORDER BY ape1, nombre;


SELECT ape1, nombre, nvl(comision, 0) + salario -- si condideramos sueldo total
FROM empleado
WHERE nvl(comision, 0) + salario <  ( SELECT min(nvl(comision, 0) + salario)
                                      FROM empleado
                                    );

--9. Obtener los nombre de empleados que trabajan en el departamento del
--cuál es jefe el empleado con código 1.
SELECT nombre, ape1
FROM empleado
WHERE coddpto = ( SELECT coddpto
```

```
                  FROM dpto
                  WHERE codemplejefe = 1
                );

--10. Obtener los nombres de los empleados cuyo primer apellido empiece por
--las letras p, q, r, s.
SELECT nombre, ape1
FROM empleado
WHERE lower(substr(ape1, 1, 1)) IN ( 'p', 'q', 'r', 's' ); -- también
lower(ape1) like 'p%' or ... or lower(ape1) like 's%', aunque perdiese un poco
de legibilidad

SELECT nombre,
       ape1
FROM empleado
WHERE substr(ape1, 1, 1) BETWEEN 'P' AND 'S'; -- esta a mí no se me hubiese
ocurrido, obviamente vale porque son letras correlativas, mejor pasarle un lower
o un apper para asegurar que no distinguimos entre minusculas y mayúsculas

--11. Obtener los empleados cuyo nombre de pila contenga el nombre JUAN.


SELECT nombre,
       ape1
FROM empleado
WHERE nombre LIKE 'JUAN%' or nombre LIKE ' %JUAN% ' or nombre LIKE '% JUAN'  ;
-- empiece por, contenga o termine en, para que no tenga en cuenta JUANIN

SELECT *
FROM empleado
WHERE instr(nombre, 'JUAN') <> 0; -- otra opción


--12. Obtener los nombres de los empleados que viven en ciudades en las que
--hay algún centro de trabajo
SELECT nombre, ape1
FROM empleado
WHERE upper(localidad) IN ( SELECT upper(localidad)
                      FROM centro
                    );

--13. Obtener el nombre del jefe de departamento que tiene mayor salario de
--entre los jefes de departamento.
SELECT nombre,ape1, salario
FROM empleado
WHERE codemple IN ( SELECT codemplejefe
                    FROM dpto
                  )
      AND salario >= ALL ( SELECT salario --salarios de todos los jefes de
departamento
                     FROM empleado
                     WHERE codemple IN ( SELECT codemplejefe
                                         FROM dpto
                                       )
```

```
                        );

--14. Obtener en orden alfabético los salarios y nombres de los empleados cuyo
--salario sea superior al 60% del máximo salario de la empresa.
SELECT nombre, ape1, salario
FROM empleado
WHERE salario > ( SELECT MAX(salario) * 0.6
                  FROM empleado
                )
ORDER BY 2, 1;
SELECT nombre, ape1, salario
FROM empleado
WHERE salario > 0.6*( SELECT MAX(salario) -- más eficiente
                  FROM empleado
                )
ORDER BY 2, 1;
--15. Obtener en cuántas ciudades distintas viven los empleados
SELECT COUNT(DISTINCT localidad)
FROM empleado;

-- 16 El nombre y apellidos del empleado que más salario cobra
SELECT nombre, ape1, ape2
FROM empleado
WHERE salario = ( SELECT MAX(salario)
                  FROM empleado
                );


--17. Obtener las localidades y número de empleados de aquellas en las que viven
más
--de 3 empleados
SELECT localidad, count(*)
FROM empleado
HAVING count(*) > 3;


--18. Obtener para cada departamento cuántos empleados trabajan, la suma de sus
--salarios y la suma de sus comisiones para aquellos departamentoS en los que
hay
--algún empleado cuyo salario es superior a 1700 euros.
SELECT coddpto, SUM(salario), SUM(comision), COUNT(codemple)
FROM empleado
GROUP BY coddpto
HAVING coddpto = SOME ( SELECT coddpto
                        FROM empleado
                        WHERE salario > 1700
                      );
SELECT * FROM DPTO;

-- si queremos listar el nombre del departamento combinamos con dpto
SELECT e1.coddpto,denominacion,  SUM(e1.salario), SUM(e1.comision),
COUNT(e1.codemple)
FROM empleado e1 join dpto on e1.coddpto=dpto.coddpto
GROUP BY e1.coddpto, denominacion
```

```
HAVING e1.coddpto = SOME ( SELECT distinct e2.coddpto
                           FROM empleado e2
                           WHERE salario > 1700
                         ) order by 3 desc;

--19. Obtener el departamento que más empleados tiene

SELECT denominacion
FROM dpto, empleado
WHERE empleado.coddpto = dpto.coddpto
GROUP BY dpto.coddpto, denominacion
HAVING COUNT(empleado.codemple) = ( select max (conteo) from
                                        (SELECT COUNT(codemple)as conteo
                                         FROM empleado
                                         GROUP BY coddpto
                                        )
                                   );
SELECT denominacion, count(empleado.codemple)
FROM dpto, empleado
WHERE empleado.coddpto = dpto.coddpto
GROUP BY dpto.coddpto, denominacion
HAVING COUNT(empleado.codemple) >= ALL ( SELECT COUNT(codemple) -- si es mayor o
igual que todos es mayor o igual que el máximo, yo prefiero la anterior pues es
más legible y comparas sólo conun valor (eficiencia)
                                         FROM empleado
                                         GROUP BY coddpto
                                        );
SELECT dpto.coddpto,denominacion, count(*) NumEmpleados  -- Para comprobar la
anterior
FROM empleado join dpto on empleado.coddpto=dpto.coddpto
GROUP BY dpto.coddpto, denominacion
order by 3 desc;
--20. Obtener los nombres de todos los centros y los departamentos que se ubican
en
--cada uno,así como aquellos centros que no tienen departamentos.
insert into centro values (04,'Gran Via','Vigo'); -- insertamos un nuevo centro
para que se observe la diferencia entre outer e inner join
select * from dpto;
select * from centro;

SELECT c.codcentro, c.direccion, c.localidad, d.coddpto, d.denominacion
FROM centro c
LEFT JOIN dpto   d ON c.codcentro = d.codcentro; -- sintaxis del estándar

SELECT direccion, denominacion
FROM centro tc, dpto   td
WHERE tc.codcentro = td.codcentro (+)  -- sintaxis alternativa en oracle
ORDER BY 1, 2;

--21. Obtener el nombre del departamento de más alto nivel, es decir, aquel que
no
--depende de ningún otro.
select * from dpto;
```

```
select denominacion from dpto where coddptodepende is null;

--22. Obtener todos los departamentos existentes en la empresa y los empleados
(si
--los tiene) que pertenecen a él.
 insert into dpto values (08,'PRUEBAS',03,05,04,'F',40000);

SELECT denominacion, nombre, ape1, ape2
FROM dpto     td
LEFT JOIN empleado te ON td.coddpto = te.coddpto
ORDER BY 1;

--23. Obtener un listado en el que aparezcan todos los departamentos existentes
y el
--departamento del cual depende,si depende de alguno.
SELECT d1.coddpto, d1.denominacion,
   d2.coddpto DependeDe, d2.denominacion nombredpto
FROM dpto d1  left JOIN dpto d2 ON d1.coddptodepende = d2.coddpto;
select * from dpto;

SELECT td.denominacion, a.denominacion
FROM dpto td, dpto a
WHERE a.coddpto (+) = td.coddptodepende;

--24. Obtener un listado ordenado alfabéticamente donde aparezcan los nombres de

--los empleados y a continuación el literal "tiene comisión" si la tiene,y "no
tiene
--comisión" si no la tiene.
select nombre,ape1,ape2,nvl2(comision,'tiene comisiÃ³n','no tiene comisiÃ³n') as
comision -- la más sencilla con nvl2
from empleado
order by 1,2,3;

SELECT nombre, comision, decode(nvl(comision, 0), 0, 'No Tiene comisión', 'Tiene
comisión') tieneComision
FROM empleado
ORDER BY nombre ASC;

SELECT nombre, ape1, ape2, 'tiene comision'
FROM empleado
WHERE comision IS NOT NULL
UNION
SELECT nombre, ape1, ape2, 'no tiene comision'
FROM empleado
WHERE comision IS NULL
ORDER BY 4, 2;



SELECT     -- esta también es sencilla y legible (legibilidad=facilidad para
"leer" o entender el código)
    nombre,
    CASE
```

```
    WHEN comision IS NULL THEN
    'no tiene comision'
    ELSE
    'tiene comision'
    END comision
FROM
    empleado
ORDER BY
    nombre ASC;


--25. Obtener un listado de las localidades en las que hay centros
--y no vive ningún  empleado, ordenado alfabéticamente.
SELECT localidad
FROM centro c
WHERE upper(localidad) NOT IN ( SELECT DISTINCT localidad
                                  FROM empleado
                                )
ORDER BY localidad ASC;

SELECT upper(tc.localidad)
FROM centro tc
MINUS
SELECT upper(te.localidad)
FROM empleado te
ORDER BY 1;


 SELECT   -- opción de alumnado,  no me chista, poco eficiente, menos legible,
no la daría por buena
    upper(
        centro.localidad
    ) "localidad del centro"
FROM
    centro
    LEFT JOIN empleado ON upper(
        empleado.localidad
    ) = upper(
        centro.localidad
    )
GROUP BY
    centro.localidad
HAVING
    COUNT(empleado.localidad) < 1
ORDER BY
    centro.localidad ASC;



--26. Obtener un listado de las localidades en las que hay centros
-- y además vive al menos un empleado ordenado alfabéticamente.
SELECT localidad
FROM centro c
WHERE upper(localidad) IN ( SELECT DISTINCT localidad
```

```
                            FROM empleado
                          )
ORDER BY localidad ASC;

SELECT DISTINCT upper(localidad)
FROM centro
INTERSECT
SELECT DISTINCT upper(localidad)
FROM empleado
ORDER BY 1 ASC;

-- no me chista la siguiente
SELECT localidad
FROM centro c
WHERE upper(localidad) NOT IN ( SELECT DISTINCT localidad
                                  FROM empleado
                                )
ORDER BY localidad ASC;

SELECT upper(tc.localidad)
FROM centro tc
MINUS
SELECT upper(te.localidad)
FROM empleado te
ORDER BY 1;

--27. Esta cuestión puntúa por 2. Se desea dar una gratificación por navidades
en
--función de la antigüedad en la empresa siguiendo estas pautas:
--1. Si lleva entre 1 y 5 años, se le dará 100 euros
--2. Si lleva entre 6 y 10 años, se le dará 50 euros por año
--3. Si lleva entre 11 y 20 años, se le dará 70 euros por año
--4. Si lleva más de 21 años, se le dará 100 euros por año
--28. Obtener un listado de los empleados,ordenado alfabéticamente,indicando
--cuánto le corresponde de gratificación.
select nombre, ape1, ape2,TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12)
antiguedad from empleado; -- veamos la antiguedad

--28. Obtener un listado de los empleados,ordenado alfabéticamente,indicando
--cuánto le corresponde de gratificación.
SELECT nombre, ape1, ape2, años_antiguedad, CASE
    WHEN años_antiguedad BETWEEN 1 AND 5   THEN
        100
    WHEN años_antiguedad BETWEEN 6 AND 10  THEN
        50 * años_antiguedad
    WHEN años_antiguedad BETWEEN 11 AND 20 THEN
        70 * años_antiguedad
    WHEN años_antiguedad >= 21             THEN
        100 * años_antiguedad
                                            END gratificacion
FROM ( SELECT nombre, ape1, ape2, trunc(months_between(sysdate, fechaingreso) /
12) años_antiguedad
-- una subselect en la cláusula from me permite simplificar la expresión de la
select externa, poniendo antiguedad en lugar de la expresión MONTHS_BETWEEN...
```

```
       FROM empleado
     )
ORDER BY nombre, ape2, ape2 ASC;


-- la union de consultas es otra opción, menos bonita que tampoco gana en
eficiencia, pero arroja el mismo resultado
SELECT nombre, ape1, ape2,TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12)
antiguedad, 100 AS gratificacion
FROM empleado
WHERE TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12) BETWEEN 1 AND 5

UNION

SELECT nombre, ape1, ape2, TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12)
antiguedad, 50 * TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12)
FROM empleado
WHERE TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12) BETWEEN 6 AND 10

UNION

SELECT nombre, ape1, ape2, TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12)
antiguedad,70 * TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12)
FROM empleado
WHERE TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12) BETWEEN 11 AND 20

UNION

SELECT nombre, ape1, ape2,TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12)
antiguedad, 100 * TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12)
FROM empleado
WHERE TRUNC(MONTHS_BETWEEN(SYSDATE,FECHAINGRESO)/12) >= 21
ORDER BY 2, 3, 1;


/*28 Obtener un listado de los empleados,ordenado alfabéticamente,indicando
cuánto le corresponde de gratificación.*/
SELECT
    nombre,
    ape1,
    ape2,fechaingreso,
    round(months_between( -- aquí podéis observar cómo se reptite la expresión
round(months_between..... ,por eso lo de la subselect en la cláusula from me
parece mejor opción
        sysdate, fechaingreso
    ) / 12) AS "años de antigüedad",
    CASE
    WHEN round(months_between(
        sysdate, fechaingreso
    ) / 12) BETWEEN 1 AND 5   THEN
    100
    WHEN round(months_between(
        sysdate, fechaingreso
```

```
    ) / 12) BETWEEN 6 AND 10  THEN
    ( round(months_between(
        sysdate, fechaingreso
    ) / 12) ) * 50
    WHEN round(months_between(
        sysdate, fechaingreso
    ) / 12) BETWEEN 11 AND 20 THEN
    ( round(months_between(
        sysdate, fechaingreso
    ) / 12) ) * 70
    ELSE
    round(months_between(
        sysdate, fechaingreso
    ) / 12) * 100
    END     gratificacion
FROM
    empleado
;



--29. Obtener a los nombres, apellidos de los empleados que no son jefes de
--departamento.
SELECT nombre, ape1, ape2
FROM empleado
WHERE codemple NOT IN ( SELECT DISTINCT codemplejefe
                          FROM dpto
                       );
```