

CUESTIONES SQL**(1 punto)**

- 1.- Cuando se realiza una venta directa a un cliente no habitual, no hace falta almacenar todos sus datos, por lo que el valor de codcli en la factura correspondiente es igual a NULL. Según éste comentario, ¿la siguiente sentencia SQL responde a la consulta “facturación del año 1995, sin tener en cuenta descuentos e iva, que incluya las ventas directas y las ventas realizadas a los clientes de la provincia de Castellón?”. Razona la respuesta sin ambigüedad. (0.5 puntos)

```
SELECT SUM(cant*precio)
FROM   lineas_fac lin, facturas fac, clientes cli
WHERE  lin.codfac = fac.codfac
AND    ((fac.codcli = cli.codcli AND cli.codpostal like '12%')
OR     (fac.codcli IS NULL)) AND to_char (fecha,'YY') = '95';
```

No, las facturas que tengan como un valor de codcli igual a NULL, se van a acumular tantas veces como clientes hayan en la tabla CLIENTES.

- 2.- ¿Qué errores produce la siguiente consulta?. Explica, sin ambigüedad, porqué se produce cada uno. (0.5 puntos)

```
SELECT nombre, descrip, SUM(cant)
FROM   clientes cli, facturas fac, lineas_fac lin, articulos art
WHERE  cli.codpue = pue.codpue AND cli.codcli = fac.codcli
AND    fac.codfac = lin.codfac AND lin.codart = art.codart
AND    EXISTS ( SELECT *
                  FROM   pueblos pue, provincias pro
                  WHERE  pue.codpro = pro.codpro
                        AND  UPPER(pro.nombre) = 'CASTELLON')
GROUP BY codcli, codart
HAVING UPPER(nombre) like '%GARCIA%'
AND    UPPER(descrip) like 'BOMBILLAS%';
```

Los atributos nombre y descrip que aparecen en el SELECT y en el HAVING, deberían aparecer también en el GROUP BY.

Existe ambigüedad en los atributos codcli y codart del GROUP BY. Sería necesario poner un prefijo que indicara a que tabla se refieren.

La referencia externa (cli.codpue = pue.codpue) debería aparecer en la subconsulta y no en la consulta, lo que genera un error porque el alias pue no está definido en el FROM.

Películas que se encuentran en todos los idiomas disponibles en el video club.

CUESTIONES SQL**(1 punto)**

- 1.- La información almacenada en la base de datos para los clientes no es completa, pudiendo existir clientes de los cuales no se tiene constancia de la localidad en la que viven. Según este comentario, ¿la siguiente sentencia SQL responde a la consulta “facturación del primer trimestre del año 1996 de los clientes de la provincia de Castellón?”. Razona la respuesta sin ambigüedad. (0.5 puntos)

```
SELECT SUM(precio * cant)
FROM facturas fac, lineas_fac lin, clientes cli, pueblos pue, provincias prv
WHERE prv.nombre = 'CASTELLON' AND prv.codpro = pue.codpro
AND pue.codpue = cli.codpue AND cli.codcli = fac.codcli
AND fac.codfac = lin.codfac
AND TO_NUMBER(TO_CHAR (fecha,'YY')) = 96
AND TO_NUMBER(TO_CHAR (fecha,'Q')) = 1;
```

No, la facturación de los clientes que tengan como valor de codpue NULL no aparecerán en el resultado, aunque sean de Castellón.

- 2.- ¿Qué error podría aparecer al ejecutar la siguiente consulta?. Explica, sin ambigüedad, porqué se produce. (0.5 puntos)

```
SELECT art.descrip
FROM articulos art
WHERE 0 < ( SELECT fac.dto
            FROM facturas fac, lineas_fac lin
            WHERE fac.codfac = lin.codfac
            AND lin.codart = art.codart
            AND MONTHS_BETWEEN (SYSDATE, fecha) < 24 );
```

El resultado de la subconsulta puede tener más de una fila, lo que provocaría un error al compararlo con el número 0.

Para resolver el error sería necesario incluir ALL ó ANY a continuación de <.

CUESTIONES SQL**(1 punto)**

- 1.- ¿La siguiente sentencia SQL responde a la consulta “artículos que han aparecido más de una vez en una factura”? Razona la respuesta sin ambigüedad. (0.5 puntos)

```
SELECT DISTINCT art.codart, art.descrip
FROM  articulos art, lineas_fac lin, facturas fac
WHERE  art.codart = lin.codart AND lin.codfac = fac.codfac
GROUP BY  art.codart, fac.codfac
HAVING  SUM(lin.cant * lin.precio) <> MAX (lin.cant * lin.precio)
ORDER BY  2 ;
```

Sí.

- 2.- ¿Qué errores produce la siguiente consulta?. Explica, sin ambigüedad, porqué se produce cada uno. (0.5 puntos)

```
SELECT  nombre
FROM  pueblos pue, articulos art
WHERE  EXISTS
      ( SELECT  *
        FROM  facturas fac, lineas_fac lin, clientes cli
        WHERE  fac.codfac = lin.codfac AND lin.codart = art.codart
              AND  cli.codcli = fac.codcli AND cli.codpue = pue.codpue )
GROUP BY  codpue, codart
HAVING  SUM(cant) > 10 AND UPPER(descrip) LIKE 'CLAVO%' ;
```

Los atributos nombre y descrip que aparecen en el SELECT y en el HAVING, deberían aparecer también en el GROUP BY.

La condición (SUM(cant) > 10) no puede aparacer en el HAVING ya que el atributo cant no aparece ni en pueblos ni en artículos.

CUESTIONES SQL**(0.5 puntos)**

- 1.- La siguiente sentencia SQL intenta obtener "los clientes que han comprado más de 10 artículos diferentes en _____". Indica el periodo de tiempo relacionado con la consulta, así como si la consulta responde realmente al enunciado. (0.5 puntos)

```
SELECT cli.nombre
FROM clientes cli, facturas fac, lineas_fac lin
WHERE cli.codcli = fac.codcli
      AND fac.codfac = lin.codfac
      AND to_char(fac.fecha,'YYQ') = '964'
GROUP BY cli.codcli, cli.nombre
HAVING COUNT(*) > 10
ORDER BY 1;
```

El periodo de tiempo corresponde con el último trimestre de 1996.

La consulta cuenta el número de líneas que aparece en todas las facturas de un cliente, pero no el número de artículos. Para ello sería necesario modificar la función de grupo del HAVING, quedando la siguiente consulta. .

```
SELECT cli.nombre
FROM clientes cli, facturas fac, lineas_fac lin
WHERE cli.codcli = fac.codcli
      AND fac.codfac = lin.codfac
      AND to_char(fac.fecha,'YYQ') = '964'
GROUP BY cli.codcli, cli.nombre
HAVING COUNT(DISTINCT art.codart) > 10
ORDER BY 1;
```

CUESTIONES SQL**(0.5 puntos)**

- 1.- La siguiente sentencia SQL intenta obtener "los provincias, ordenadas alfabeticamente, que tienen más de 100 clientes y más de 10 pueblos con clientes". Indica si aparece algún error en la consulta, y si ésta responde realmente al enunciado. Razona la respuesta. (0.5 puntos)

```
SELECT  prv.nombre
FROM    provincias prv, pueblos pue, clientes cli
WHERE   prv.codpro = pue.codpro
        AND   pue.codpue = cli.codpue
GROUP BY prv.codpro
HAVING  COUNT(codpue) > 10 AND COUNT(codcli) > 100
ORDER BY 1;
```

En el GROUP BY deben de aparecer los atributos del SELECT, y en la consulta prv.nombre no cumple esta norma.

Existe ambigüedad en la aparición del atributo codpue de la función de grupo COUNT del HAVING, ya que dicho atributo está definido en dos tablas. Por tanto, debería utilizarse un prefijo que resolviera dicho problema.

La primera condición del HAVING no cuenta valores de codpue diferentes, sino que en las dos condiciones se cuentan tuplas, que en este caso equivale a contar clientes. Para resolverlo debería aparecer la clausula DISTINCT en el COUNT.

Por todo lo dicho, la consulta quedaría como sigue:

```
SELECT  prv.nombre
FROM    provincias prv, pueblos pue, clientes cli
WHERE   prv.codpro = pue.codpro
        AND   pue.codpue = cli.codpue
GROUP BY prv.codpro, prv.nombre
HAVING  COUNT(DISTINCT pue.codpue) > 10 AND COUNT(codcli) > 100
ORDER BY 1;
```

CUESTIONES SQL**(0.5 puntos)**

- 1.- La siguiente sentencia SQL intenta obtener "el nombre y el número de facturas con un iva nulo que tiene cada uno de los clientes de Castellon que han hecho alguna compra". Indica si aparece algún error en la consulta, y si ésta responde realmente al enunciado. Razona la respuesta. (0.5 puntos)

```
SELECT cli.nombre, COUNT(*) - COUNT(fac.iva)
FROM provincias prv, pueblos pue, clientes cli, facturas fac
WHERE prv.codpro = pue.codpro
      AND pue.codpue = cli.codpue
      AND cli.codcli = fac.codcli
GROUP BY prv.codpro, cli.codcli
HAVING UPPER(prv.nombre) = 'CASTELLON'
ORDER BY 1;
```

En el GROUP BY deben de aparecer los atributos del SELECT y del HAVING, y en la consulta cli.nombre y prv. nombre no cumplen esta norma.

Aparte de estos errores, la consulta sí cumple el enunciado, quedando como sigue,

```
SELECT cli.nombre, COUNT(*) - COUNT(fac.iva)
FROM provincias prv, pueblos pue, clientes cli, facturas fac
WHERE prv.codpro = pue.codpro
      AND pue.codpue = cli.codpue
      AND cli.codcli = fac.codcli
GROUP BY prv.codpro, prv.nombre, cli.codcli, cli.nombre
HAVING UPPER(prv.nombre) = 'CASTELLON'
ORDER BY 1;
```

CUESTIONES SQL**(0.5 puntos)**

- 1.- La siguiente sentencia SQL intenta obtener, ordenadamente, "los clientes cuyos abonos durante el año pasado han sumado mas de 10 unidades, mostrando el número de unidades que han devuelto". Indica si aparece algún error en la consulta, y si ésta responde realmente al enunciado. Razona la respuesta. (0.5 puntos)

```
SELECT cli.nombre, -SUM (lin.cant)
FROM clientes cli, facturas fac, lineas_fac lin
WHERE cli.codcli = fac.codcli AND fac.codfac = lin.codfac
AND lin.cant < 0
AND TO_NUMBER(TO_CHAR (fac.fecha,'YY')) =
      TO_NUMBER(TO_CHAR (SYSDATE,'YY')) - 1
AND SUM (lin.cant) < -10
GROUP BY cli.codcli
ORDER BY 1;
```

Las funciones de grupo deben aparecer en el HAVING, mientras que en la consulta se define una condición con un SUM en el WHERE.

En el GROUP BY deben de aparecer los atributos del SELECT y del HAVING, y en la consulta cli.nombre no cumple esta norma.

```
SELECT cli.nombre, -SUM (lin.cant)
FROM clientes cli, facturas fac, lineas_fac lin
WHERE cli.codcli = fac.codcli AND fac.codfac = lin.codfac
AND lin.cant < 0
AND TO_NUMBER(TO_CHAR (fac.fecha,'YY')) =
      TO_NUMBER(TO_CHAR (SYSDATE,'YY')) - 1
GROUP BY cli.codcli, cli.nombre
HAVING SUM (lin.cant) < -10
ORDER BY 1;
```

CUESTIONES SQL**(0.5 puntos)**

- 1.- La siguiente sentencia SQL intenta obtener, ordenadamente, "los clientes que durante el año pasado realizaron más de 25 facturas en las que se incluían artículos cuya descripción contenía el texto BASE. Razona la respuesta. (0.5 puntos)

```
SELECT cli.codcli, cli.nombre, COUNT(fac.codfac)
FROM clientes cli, facturas fac, lineas_fac lin, articulos art
WHERE cli.codcli = fac.codcli AND fac.codfac = lin.codfac
      AND lin.codart = art.codart AND UPPER(art.descripcion) LIKE '%BASE%'
      AND TO_NUMBER(TO_CHAR(fac.fecha,'YY')) =
          TO_NUMBER(TO_CHAR(SYSDATE,'YY')) - 1
GROUP BY cli.codcli, fac.codfac
HAVING COUNT(fac.codfac) > 25
ORDER BY 2;
```

En el GROUP BY deben de aparecer los atributos del SELECT y del HAVING, y en la consulta cli.nombre no cumple esta norma.

La aparición de fac.codfac en el GROUP BY impide la posibilidad de contar número de facturas, ya que cada factura aparece en un grupo diferente. Por lo tanto fac.codfac se debe eliminar del GROUP BY.

Por último, para poder evaluar el número de facturas que cumplen la condición, resulta necesario incluir la cláusula DISTINCT, ya que de otro modo se obtiene el número de líneas de facturas que cumplen la condición.

De acuerdo a todo lo dicho, la consulta que cumple el enunciado es la siguiente:

```
SELECT cli.codcli, cli.nombre, COUNT(DISTINCT fac.codfac)
FROM clientes cli, facturas fac, lineas_fac lin, articulos art
WHERE cli.codcli = fac.codcli AND fac.codfac = lin.codfac
      AND lin.codart = art.codart AND UPPER(art.descripcion) LIKE '%BASE%'
      AND TO_NUMBER(TO_CHAR(fac.fecha,'YY')) =
          TO_NUMBER(TO_CHAR(SYSDATE,'YY')) - 1
GROUP BY cli.codcli, cli.nombre
HAVING COUNT(DISTINCT fac.codfac) > 25
ORDER BY 2;
```


CUESTIONES SQL**(0.5 puntos)**

- 1.- Verifica si la siguiente sentencia SQL obtiene, ordenadamente, "los artículos que durante el año pasado han tenido una venta mayor de 50 unidades". Razona la respuesta. (0.5 puntos)

```
SELECT  art.codart, art.descrip
FROM    facturas fac, lineas_fac lin, articulos art
WHERE   fac.codfac = lin.codfac AND lin.codart = art.codart
GROUP BY art.codart
HAVING  SUM(DISTINCT lin.cant) > 50
        AND  TO_NUMBER(TO_CHAR(fac.fecha,'YYYY')) =
              TO_NUMBER(TO_CHAR(SYSDATE,'YYYY'))-1
ORDER BY 1, 2;
```

En el GROUP BY deben de aparecer los atributos del SELECT y del HAVING, y en la consulta art.descrip y fac.fecha no cumplen esta norma.

La aparición de fac.fecha en el GROUP BY impediría la posibilidad de contar las ventas de un artículo, ya que las ventas de cada fecha aparecerían en un grupo diferente. Por lo tanto fac.fecha no se puede aparecer en el GROUP BY, y la condición asociada debería estar en el WHERE.

Por último, la cláusula DISTINCT en el SUM permite sumar valores distintos pero no todas las ventas, ya que si se venden 50 veces 1 artículos el resultado sería 1. Por tanto se debe eliminar esa cláusula.

De acuerdo a todo lo dicho, la consulta que cumple el enunciado es la siguiente:

```
SELECT  art.codart, art.descrip
FROM    facturas fac, lineas_fac lin, articulos art
WHERE   fac.codfac = lin.codfac AND lin.codart = art.codart
        AND  TO_NUMBER(TO_CHAR(fac.fecha,'YYYY')) =
              TO_NUMBER(TO_CHAR(SYSDATE,'YYYY'))-1
GROUP BY art.codart, art.descrip
HAVING  SUM(lin.cant) > 50
ORDER BY 1, 2;
```

CUESTIONES SQL**(0.5 puntos)**

- 1.- Verifica si la siguiente sentencia SQL obtiene, ordenadamente, "los artículos que sólo se han comprado en una única provincia, que también debe visualizarse". Razona la respuesta. (0.5 puntos)

```
SELECT  art.descripcion, prv.nombre PROVINCIA
FROM    articulos art, lineas_fac lin, facturas fac, clientes cli, pueblos pue,
        provincias prv
WHERE   art.codart = lin.codart AND lin.codfac = fac.codfac
        AND fac.codcli = cli.codcli AND cli.codpue = pue.codpue
        AND pue.codpro = prv.codpro
GROUP BY art.codart, art.descripcion
HAVING  COUNT(prv.codpro) = 1
ORDER BY 1;
```

La función de grupo COUNT(prv.codpro) obtiene el número de líneas de facturas en donde aparece el artículo, pero no el número de provincias. Para contar el número de provincias es necesario incluir la cláusula DISTINCT.

En el GROUP BY deben de aparecer los atributos del SELECT y del HAVING, y en la consulta prv.nombre no cumple esta norma.

La aparición de prv.nombre en el GROUP BY impediría la posibilidad de contar el número de provincias, ya que las ventas de cada provincia aparecerían en un grupo diferente. Por lo tanto prv.nombre no puede aparecer en el GROUP BY. La solución podría ser aplicar una función de grupo sobre dicho campo.

De acuerdo a todo lo dicho, la consulta que cumple el enunciado es la siguiente:

```
SELECT  art.descripcion, MAX(prv.nombre) PROVINCIA
FROM    articulos art, lineas_fac lin, facturas fac, clientes cli, pueblos pue,
        provincias prv
WHERE   art.codart = lin.codart AND lin.codfac = fac.codfac
        AND fac.codcli = cli.codcli AND cli.codpue = pue.codpue
        AND pue.codpro = prv.codpro
GROUP BY art.codart, art.descripcion
HAVING  COUNT(DISTINCT prv.codpro) = 1
ORDER BY 1;
```

CUESTIONES SQL**(0.5 puntos)**

- 1.- Verifica si la siguiente sentencia SQL obtiene, "el porcentaje del total de los pueblos donde se han realizado ventas durante el año en curso.". Razona la respuesta. (0.5 puntos)

```
SELECT  COUNT(DISTINCT cli.codpue) / COUNT(DISTINCT pue.codpue)
FROM    pueblos pue, clientes cli, facturas fac
WHERE   cli.codcli = fac.codcli
AND     TO_NUMBER(TO_CHAR(fac.fecha,'YYYY')) =
        TO_NUMBER(TO_CHAR(SYSDATE,'YYYY'));
```

Esta consulta sí cumple el enunciado.

CUESTIÓN SQL**(0.5 puntos)**

- 1.- Verifica si la siguiente sentencia SQL obtiene, "el número de pueblos que durante el año pasado tuvieron clientes que realizaron compras.". Razona la respuesta. (0.5 puntos)

```
SELECT COUNT (codpue)
FROM clientes cli
WHERE EXISTS
( SELECT fac.codcli
  FROM facturas fac
  WHERE fac.codcli = cli.codcli
    AND TO_NUMBER(TO_CHAR(fac.fecha,'YY')) =
      TO_NUMBER(TO_CHAR(SYSDATE,'YY')) - 1 )
```

No, ya que aparecen dos errores.

El primero se refiere al modo de contar los pueblos. COUNT(codpue) cuenta el número de clientes que cumplen la condición y no tienen un valor NULL en el campo codpue. Para contar pueblos es necesario incluir la cláusula DISTINCT.

El segundo hace referencia al modo de evaluar las fechas. Dado que estamos en el año 2000 y el año anterior es 1999, se debe de evaluar los cuatro dígitos del año para que la comparación sea adecuada.

Según estos comentarios, la consulta quedaría como,

```
SELECT COUNT (DISTINCT codpue)
FROM clientes cli
WHERE EXISTS
( SELECT fac.codcli
  FROM facturas fac
  WHERE fac.codcli = cli.codcli
    AND TO_NUMBER(TO_CHAR(fac.fecha,'YYYY')) =
      TO_NUMBER(TO_CHAR(SYSDATE,'YYYY')) - 1 )
```

CUESTIÓN SQL**(0.5 puntos)**

- 1.- Verifica si la siguiente sentencia SQL obtiene, "los artículos que durante el año pasado siempre se vendieron con un descuento superior el 15%.". Razona la respuesta, indicando los errores que puedan aparecer y su solución. (0.5 puntos)

```
SELECT  art.descripcion
FROM    articulos art, lineas_fac lin, facturas fac
WHERE   art.codart = lin.codart AND lin.codfac = fac.codfac
        AND    TO_NUMBER(TO_CHAR (fac.fecha,'YY')) =
                TO_NUMBER(TO_CHAR (SYSDATE,'YY')) - 1
GROUP BY art.codart
HAVING  MIN (lin.dto) > 15
ORDER BY 1;
```

No, ya que aparecen varios errores.

El primer error es sintáctico, error que ocurre por la aparición del atributo art.descripcion en el SELECT, dado que dicho atributo no aparece en el GROUP BY. La solución es poner dicho atributo en el GROUP BY, ya que no genera ninguna incidencia en la generación de los grupos.

Los otros errores son semánticos, es decir, no generan un error de ejecución pero no permiten obtener la solución deseada.

El primero es que el atributo lin.dto acepta nulos, valores que se excluyen en el cálculo del MIN, y por tanto pueden aceptarse artículos no deseados.

El segundo hace referencia al modo de evaluar las fechas. Dado que estamos en el año 2000 y el año anterior es 1999, se debe de evaluar los cuatro dígitos del año para que la comparación sea adecuada.

Según estos comentarios, la consulta quedaría como,

```
SELECT  art.descripcion
FROM    articulos art, lineas_fac lin, facturas fac
WHERE   art.codart = lin.codart AND lin.codfac = fac.codfac
        AND    TO_NUMBER(TO_CHAR (fac.fecha,'YYYY')) =
                TO_NUMBER(TO_CHAR (SYSDATE,'YYYY')) - 1
GROUP BY art.codart, art.descripcion
HAVING  MIN (nvl(lin.dto,0)) > 15
ORDER BY 1;
```

CUESTIÓN SQL**(0.5 puntos)**

- 1.- Verifica si la siguiente sentencia SQL obtiene, "los clientes a los que se les ha aplicado más de un descuento diferente en las facturas del año pasado". Razona la respuesta, indicando los errores que puedan aparecer y su solución. (0.5 puntos)

```
SELECT cli.nombre
FROM clientes cli, facturas fac
WHERE cli.codcli = fac.codcli
AND TO_NUMBER(TO_CHAR (fac.fecha,'YY')) =
TO_NUMBER(TO_CHAR (SYSDATE,'YY')) - 1
GROUP BY cli.codcli
HAVING COUNT (fac.dto) > 1
ORDER BY 1;
```

No, ya que aparecen varios errores.

El primer error es sintáctico, error que ocurre por la aparición del atributo cli.nombre en el SELECT, dado que dicho atributo no aparece en el GROUP BY. La solución es poner dicho atributo en el GROUP BY, ya que no genera ninguna incidencia en la generación de los grupos.

Los otros errores son semánticos, es decir, no generan un error de ejecución pero no permiten obtener la solución deseada.

El primero es que el atributo fac.dto acepta nulos, valores que se excluyen en el cálculo del COUNT, por lo que resulta necesario utilizar la función NVL. Además en la función COUNT se debe incluir la cláusula DISTINCT para que sólo se cuenten los valores distintos de descuento.

El segundo hace referencia al modo de evaluar las fechas. Dado que estamos en el año 2000 y el año anterior es 1999, se debe de evaluar los cuatro dígitos del año para que la comparación sea adecuada.

Según estos comentarios, la consulta quedaría como,

```
SELECT cli.nombre
FROM clientes cli, facturas fac
WHERE cli.codcli = fac.codcli
AND TO_NUMBER(TO_CHAR (fac.fecha,'YYYY')) =
TO_NUMBER(TO_CHAR (SYSDATE,'YYYY')) - 1
GROUP BY cli.codcli, cli.nombre
HAVING COUNT (DISTINCT NVL(fac.dto,0)) > 1
ORDER BY 1;
```