1. **Crear una vista que devuelva:**
   1. **Código y Nombre (manu\_code,manu\_name) de los fabricante, posean o no productos (en tabla stock) , cantidad de productos que poseen en tabla stock (cant\_producto) y la fecha de la última OC que contenga un producto suyo (ult\_fecha\_orden).**
      * **De los fabricantes que fabriquen productos sólo se podrán mostrar los dr**
      * **No se permite utilizar funciones definidas por usuario, ni tablas temporales, ni UNION.**
   2. **Realizar una consulta sobre la vista que devuelva manu\_code, manu\_name, cant\_producto y si el campo ult\_fecha\_orden posee un NULL informar ‘No Posee Órdenes’ si no posee NULL informar el valor de dicho campo.**
      * **No se puede utilizar UNION para el SELECT.**

-- 1a

-- Opción 1

CREATE VIEW vrecu1 AS

SELECT m.manu\_code,m.manu\_name,

count(stock\_num) cant\_productos,

(SELECT max(order\_date)

FROM orders o JOIN items i

ON o.order\_num=i.order\_num

AND i.manu\_code=m.manu\_code) ult\_compra

FROM manufact m LEFT JOIN products s

ON s.manu\_code = m.manu\_code

GROUP BY m.manu\_code,m.manu\_name

HAVING count(stock\_num)=0 OR count(stock\_num)>1

-- Opción 2

DROP VIEW vrecu1

CREATE VIEW vrecu1 AS

SELECT m.manu\_code,m.manu\_name,

count(distinct s.stock\_num) cant\_productos,

max(o.order\_date)ult\_compra

FROM manufact m

LEFT JOIN products s ON s.manu\_code = m.manu\_code

LEFT JOIN items i ON s.manu\_code = i.manu\_code AND s.stock\_num=i.stock\_num

LEFT JOIN orders o ON i.order\_num = o.order\_num

GROUP BY m.manu\_code,m.manu\_name

HAVING count(distinct s.stock\_num)=0

OR count(distinct s.stock\_num)>1

-- Opción 3

DROP VIEW vrecu1

CREATE VIEW vrecu1 AS

SELECT m.manu\_code,m.manu\_name,

count(distinct s.stock\_num) cant\_productos,

max(o.order\_date)ult\_compra

FROM manufact m

LEFT JOIN products s ON s.manu\_code = m.manu\_code

LEFT JOIN items i ON s.manu\_code = i.manu\_code AND s.stock\_num=i.stock\_num

LEFT JOIN orders o ON i.order\_num = o.order\_num

WHERE m.manu\_code IN

(SELECT m2.manu\_code

FROM manufact m2 JOIN products s2

ON (m2.manu\_code = s2.manu\_code)

GROUP BY m2.manu\_code

HAVING COUNT(\*) >1 OR COUNT(\*) = 0)

GROUP BY m.manu\_code,m.manu\_name

-- 1b

-- Inserto fila de prueba, la borro al final

INSERT INTO manufact VALUES ('PRU','Prueba',99,'CA',NULL,NULL)

-- Opcion 1 con CASE

select manu\_code,manu\_name,cant\_productos,

case when ult\_compra is null then 'No posee Productos'

when ult\_compra is not null then cast(ult\_compra as char) end

from vrecu1

-- Opcion 2 con COALESCE

-- falla por problemas de Casteo

select manu\_code,manu\_name,cant\_productos,

COALESCE(ult\_compra,'No posee Productos'

from vrecu1

-- Opcion 2 con COALESCE

select manu\_code,manu\_name,cant\_productos,

COALESCE(cast(ult\_compra as char),'No posee Productos')

from vrecu1

-- Borro la fila dummy

DELETE FROM manufact WHERE manu\_code='PRU'

1. **Desarrollar una consulta muestre un ABC de fabricantes que:**

**Liste el código de fabricante, el nombre del fabricante, la cantidad de órdenes de compra que contentan sus productos y la suma total los productos vendidos.**

**Se deberán tener en cuenta sólo los fabricantes cuyo código comience con A ó con N y posea 3 letras, y los productos cuya descripción posea el string “tennis” ó el string “ball” en cualquier parte del nombre.**

**Sólo se podrán mostrar los datos de los fabricantes cuyo total sea mayor que el total de ventas promedio de todos los fabricantes (Cantidad vendida / Cantidad de fabricantes que tuvieron productos vendidos).**

**La consulta deberá mostrar los registros ordenados por total vendido de mayor a menor.**

SELECT m.manu\_code,m.manu\_name,

COUNT(DISTINCT i.order\_num) cantidadOrdenes,

SUM(total\_price\*quantity) totalComprado,

(SELECT COUNT(\*) FROM manufact m2) cantFabricantes

FROM manufact m JOIN items i ON (m.manu\_code=i.manu\_code)

JOIN product\_types p ON (i.stock\_num=p.stock\_num)

WHERE (description LIKE '%tennis%' OR description LIKE '%ball%')

AND m.manu\_code LIKE '[AN]\_\_'

GROUP BY m.manu\_code,m.manu\_name

HAVING SUM(total\_price\*quantity) >

(select SUM(total\_price\*quantity)

/count(DISTINCT i.manu\_code) from items i)

ORDER BY 4 DESC

1. **Crear una vista que devuelva**

**Mostrar los datos (customer\_num,lname,company) de los clientes, posean o no órdenes de compra y la cantidad de órdenes de compra, la fecha de la última OC y el total en u$s (total\_price\*quantity)comprado y el total general Comprado por todos los clientes.**

**De los clientes que posean órdenes sólo se podrán mostrar los clientes que tengan alguna órden que posea productos que son fabricados por más de dos fabricantes.**

**Mostrar los clientes que posean menos de 5 órdenes de compra.**

**Ordenar el reporte primero por los clientes que tengan órdenes por cantidad de órdenes descendente y luego por los clientes que no tengan órdenes**

**No se permite utilizar funciones, ni tablas temporales.**

CREATE VIEW v\_parcial AS

select c.customer\_num, c.lname, c.company, sname,

null ultima\_compra,0 cantidad\_ordenes, 0 total\_ordenes,

(select sum(total\_price\*quantity) FROM items) total\_general

from customer c

inner join state s on c.state =s.code

where customer\_num not in (select distinct customer\_num from orders)

UNION

select c.customer\_num, c.lname, c.company, sname, MAX(order\_date),

count(distinct o.order\_num), sum(i.total\_price\*quantity),

(select sum(total\_price\*quantity) FROM items)

from customer c

join orders o on c.customer\_num=o.customer\_num

join items i on o.order\_num = i.order\_num

join state s on c.state =s.code

where c.customer\_num in

(select DISTINCT o2.customer\_num from orders o2

JOIN items i2 ON o2.order\_num=i2.order\_num

WHERE i2.stock\_num IN (SELECT stock\_num FROM products

GROUP BY stock\_num HAVING count(\*) >2))

group by c.customer\_num,c.lname,c.company,sname

having count(distinct o.order\_num) < 5

SELECT \* FROM v\_parcial

order by 6 DESC, 1

1. **Crear una vista que devuelva**

**El top 5 de los productos (description) que fueron más comprados en cada estado (state) con la cantidad vendida y su total vendido, teniendo en cuenta que solo se mostrará el estado en el que tuvo mayor cantidad de ventas ese mismo producto.**

**Ordenarlo por la cantidad vendida descendente.**

**No se permite utilizar funciones, ni tablas temporales.**

CREATE VIEW productMasComprados

(TipoProducto,Estado,CantVendida,TotalVendido)

AS

SELECT t.description, c.state,

SUM(i.quantity),

SUM(i.total\_price\*i.quantity)

FROM products s

JOIN items i ON (s.stock\_num = i.stock\_num)

JOIN product\_types t ON (s.stock\_num=t.stock\_num)

JOIN orders o ON (i.order\_num = o.order\_num)

JOIN customer c ON (o.customer\_num = c.customer\_num)

GROUP BY t.description, c.state

HAVING SUM(i.quantity)

= (SELECT TOP 1 SUM(i1.quantity)

FROM products s1

JOIN product\_types t1 ON (s1.stock\_num=t1.stock\_num)

JOIN items i1 ON (s1.stock\_num = i1.stock\_num)

JOIN orders o1 ON (i1.order\_num = o1.order\_num)

JOIN customer c1 ON (o1.customer\_num = c1.customer\_num)

WHERE t1.description = t.description

GROUP BY c1.state, t1.description

ORDER BY SUM(i1.quantity) DESC)

ORDER BY 1

SELECT TOP 5 \* FROM productMasComprados

order by cantVendida DESC

1. **Se quiere averiguar los customers que no posean órdenes de compra y aquellos cuyas últimas órdenes de compra superen el promedio de las anteriores. Se pide mostrar customer\_num, fname, lname, paid\_date y el precio total, de las órdenes que tengan la última fecha más reciente.**

**Ordenar por fecha de pago descendiente.**

**No se permite utilizar funciones, ni tablas temporales.**

VERSION 1:

SELECT c.customer\_num, c.fname, c.lname, o.paid\_date, SUM(i.total\_price)

FROM customer c JOIN orders o ON (c.customer\_num = o.customer\_num)

JOIN items i ON (o.order\_num = i.order\_num)

WHERE o.paid\_date IN (SELECT MAX(o1.paid\_date) FROM customer c1 JOIN orders o1 ON (c1.customer\_num = o1.customer\_num)

WHERE c1.customer\_num = c.customer\_num)

GROUP BY c.customer\_num, c.fname, c.lname, o.paid\_date

HAVING SUM(i.total\_price) >= (SELECT AVG(i1.total\_price) FROM customer c1 JOIN orders o1 ON (c1.customer\_num = o1.customer\_num)

JOIN items i1 ON (o1.order\_num = i1.order\_num)

WHERE o.paid\_date >= o1.paid\_date AND c1.customer\_num = c.customer\_num)

UNION

SELECT c.customer\_num, c.fname, c.lname, o.paid\_date, SUM(i.total\_price)

FROM customer c LEFT JOIN orders o ON (c.customer\_num = o.customer\_num)

LEFT JOIN items i ON (o.order\_num = i.order\_num)

WHERE c.customer\_num NOT IN (SELECT customer\_num FROM orders)

GROUP BY c.customer\_num, c.fname, c.lname, o.paid\_date

ORDER BY o.paid\_date DESC

VERSION 2:

SELECT c.customer\_num, c.fname, c.lname, o.paid\_date, SUM(i.total\_price)

FROM customer c LEFT JOIN orders o ON (c.customer\_num = o.customer\_num)

LEFT JOIN items i ON (o.order\_num = i.order\_num)

WHERE (o.paid\_date IN (SELECT MAX(o1.paid\_date) FROM customer c1 JOIN orders o1 ON (c1.customer\_num = o1.customer\_num)

WHERE c1.customer\_num = c.customer\_num)) OR c.customer\_num NOT IN (SELECT customer\_num FROM orders)

GROUP BY c.customer\_num, c.fname, c.lname, o.paid\_date

HAVING SUM(i.total\_price) >= (SELECT AVG(i1.total\_price) FROM customer c1 JOIN orders o1 ON (c1.customer\_num = o1.customer\_num)

JOIN items i1 ON (o1.order\_num = i1.order\_num)

WHERE o.paid\_date >= o1.paid\_date AND c1.customer\_num = c.customer\_num) OR SUM(i.total\_price) IS NULL

ORDER BY o.paid\_date DESC

1. **Se desean saber los fabricantes que vendieron mayor cantidad de un mismo producto que la competencia con la cantidad vendida y su precio total. Tener en cuenta que puede existir un único producto que no sea fabricado por algún otro.**

**No se permite utilizar funciones, ni tablas temporales.**

SELECT m.manu\_code, m.manu\_name, t.description,

SUM(i.quantity), SUM(i.total\_price)

FROM manufact m JOIN products s ON (m.manu\_code = s.manu\_code)

JOIN items i ON (s.stock\_num = i.stock\_num)

JOIN product\_types t ON (s.stock\_num=t.stock\_num)

GROUP BY m.manu\_code, m.manu\_name, t.description

HAVING SUM(i.quantity) > (SELECT TOP 1 SUM(i1.quantity) FROM manufact m1 JOIN products s1 ON (m1.manu\_code = s1.manu\_code)

JOIN items i1 ON (s1.stock\_num = i1.stock\_num and m1.manu\_code = s1.manu\_code)

JOIN product\_types t1 ON (s1.stock\_num=t1.stock\_num)

WHERE (t1.description = t.description AND m1.manu\_code != m.manu\_code)

GROUP BY m1.manu\_code, m1.manu\_name, t1.description

ORDER BY 1 DESC)

OR (SELECT COUNT(\*) FROM products s2

JOIN product\_types t2 ON (s2.stock\_num=t2.stock\_num)

WHERE t2.description = t.description) = 1

Order by 3

* Para mi el anterior está mal. No trae el mayor fabricante de Baseball gloves por ej.
* El siguiente está bien, aunque No trae los Productos en los que hay varios Fabricantes con la misma cantidad máxima vendida

SELECT m.manu\_code, m.manu\_name, t.description,

SUM(i.quantity) cantidad, SUM(i.total\_price) total

FROM manufact m

JOIN items i ON (m.manu\_code = i.manu\_code)

JOIN product\_types t ON (i.stock\_num=t.stock\_num)

GROUP BY m.manu\_code, m.manu\_name, t.description

HAVING SUM(i.quantity) > coalesce((SELECT TOP 1 SUM(i2.quantity)

FROM manufact m2

JOIN items i2 ON (m2.manu\_code = i2.manu\_code)

JOIN product\_types t2 ON (i2.stock\_num = t2.stock\_num)

WHERE (t2.description = t.description AND m2.manu\_code != m.manu\_code)

GROUP BY m2.manu\_code, m2.manu\_name, t2.description ORDER BY 1 DESC), 0)

order by 3