1. Listar número de cliente, apellido y nombre, total comprado por el cliente 'Total del Cliente', la cantidad de órdenes de compra solicitadas por el cliente 'OCs del Cliente' y la cantidad de órdenes de compra solicitadas por todos los clientes 'Cant. Total OC' para los clientes cuyo promedio comprado por OC supere al promedio comprado por OC general y hayan solicitado por lo menos 2 órdenes y el zipcode comience con 94.

```
SELECT c.customer_num,

       c.lname,

       c.fname,

       SUM(i.quantity * i.unit_price) AS 'Total del Cliente',

       COUNT(DISTINCT o.order_num) AS 'Cantidad de OCs del Cliente',

       (SELECT COUNT(o.order_num) FROM orders o) AS 'Cantidad Total de OCs'

FROM customer c

    JOIN orders o ON (c.customer_num = o.customer_num)

    JOIN items i ON (o.order_num = i.order_num)

WHERE c.zipcode LIKE '94%'

GROUP BY c.customer_num, c.lname, c.fname

HAVING COUNT(DISTINCT o.order_num) >= 2

   AND (SUM(i.quantity * i.unit_price) / COUNT(DISTINCT i.order_num)) >

    (SELECT (SUM(i2.quantity * i2.unit_price)) / (COUNT(DISTINCT i2.order_num))

     FROM items i2)
```

2.

a. Se requiere crear una tabla temporal #ABC_Productos un ABC de Productos ordenado por cantidad de venta en u$, los datos solicitados son: Nro. de Stock, Código de fabricante, descripción del producto, Nombre de Fabricante, Total del producto pedido 'u$ por Producto', Cant. de producto pedido 'Unid. por Producto', para los productos que pertenezcan a fabricantes que fabriquen al menos 10 productos diferentes.

```
SELECT p.stock_num,
       p.manu_code,
       tp.description,
       m.manu_name,
       SUM(i.quantity * i.unit_price) AS 'USS por Producto',
       SUM(i.quantity) AS 'Unidad por Producto'
INTO #ABC_products
FROM products p
    JOIN product_types tp ON (p.stock_num = tp.stock_num)
    JOIN manufact m ON (p.manu_code = m.manu_code)
    JOIN items i ON (p.stock_num = i.stock_num AND p.manu_code = i.manu_code)
WHERE p.manu_code IN (SELECT p2.manu_code--, COUNT(p2.stock_num)
                      FROM products p2
                      GROUP BY p2.manu_code
                      HAVING COUNT(p2.stock_num) >= 10)
GROUP BY p.stock_num, p.manu_code, tp.description, m.manu_name
```

b. Listar los datos generados en la tablas #ABC_Productos ordenados por 'u$ por Producto', stock_num y manu_code de mayor a menor por precio y de menor a mayor por stock_num y manu_code.

```
SELECT [USS por Producto], stock_num, manu_code
FROM #ABC_products
ORDER BY [USS por Producto] DESC, stock_num, manu_code
```

3. En función a la tabla temporal generada en el punto 2, obtener un listado que detalle por cada producto existente en #ABC_Producto, la descripción del producto, el mes en el que fue solicitado, el cliente que lo solicitó (en formato 'Apellido, Nombre'), la cantidad de órdenes de compra 'Cant OC por mes', la cantidad de producto solicitado 'Unid Producto por mes' y el total en u$ solicitado 'u$ Producto por mes'.

- Sólo se deberán mostrar los clientes que vivan en el estado con mayor cantidad de clientes.
- Deberá estar ordenado por mes y producto.

```
SELECT MONTH(o.order_date) AS 'Mes',

       o.customer_num,

       tt.stock_num,

       tt.description,

       COUNT(DISTINCT i.order_num) AS 'OCs por mes',

       SUM(i.quantity) AS 'Cantidad de Productos por mes',

       SUM(i.quantity * i.unit_price) AS 'USS Productos por mes'

FROM #ABC_products tt

    JOIN items i ON (tt.stock_num = i.stock_num AND tt.manu_code = i.manu_code)

    JOIN orders o ON (o.order_num = i.order_num)

    JOIN customer c ON (c.customer_num = o.customer_num)

WHERE c.state IN (SELECT TOP 1 c2.state

                  FROM customer c2

                  GROUP BY c2.state

                  ORDER BY COUNT(customer_num) DESC)

GROUP BY tt.stock_num, tt.description, MONTH(o.order_date), o.customer_num

ORDER BY Mes, stock_num
```

4. Dado los productos con nro de stock 5,6 y 9 del fabricante 'ANZ' listar de a pares los clientes que hayan solicitado el mismo producto, siempre y cuando, el primer cliente haya solicitado más cantidad del producto que el 2do cliente.

- Se deberá informar nro de stock, código de fabricante, Nro de Cliente y Apellido del primer cliente, Nro de cliente y apellido del 2do cliente.

```
SELECT DISTINCT i1.stock_num,
       i1.manu_code,
       c1.customer_num,
       c1.lname,
       c2.customer_num,
       c2.lname

FROM items i1
 JOIN orders o1 ON (i1.order_num = o1.order_num)
 JOIN customer c1 ON (o1.customer_num = c1.customer_num)
 JOIN items i2 ON (i1.stock_num = i2.stock_num AND i1.manu_code = i2.manu_code)
 JOIN orders o2 ON (i2.order_num = o2.order_num)
 JOIN customer c2 ON (o2.customer_num = c2.customer_num)

WHERE i1.stock_num IN (5, 6, 9)
  AND i1.manu_code = 'ANZ'
  AND (SELECT SUM(quantity)
       FROM items i11
       JOIN orders o11 ON (i11.order_num = o11.order_num)
       WHERE i11.stock_num = i1.stock_num
         AND i11.manu_code = i1.manu_code
         AND o11.customer_num = c1.customer_num)  >
          (SELECT SUM(quantity)
           FROM items i12
               JOIN orders o12 ON (i12.order_num = o12.order_num)
           WHERE i12.stock_num = i2.stock_num
             AND i12.manu_code = i2.manu_code
             AND o12.customer_num = c2.customer_num)

ORDER BY 1, 2
```

5. Se requiere informar en una consulta que devuelva sólo una fila, la siguiente información: La mayor cantidad de órdenes de compra solicitada por un cliente, Mayor total en u$ solicitado por un cliente y la mayor cantidad ítems de una OC solicitada por un cliente, la menor cantidad de órdenes de compra solicitada por un cliente, el menor total en u$ solicitado por un cliente y la menor cantidad ítems de una OC solicitados por un cliente.

- Los valores máximos y mínimos solicitados deberán corresponderse a los datos de clientes sumariados de todas las órdenes existentes, sin importar a que cliente corresponda el dato.

```
SELECT (SELECT TOP 1 COUNT(DISTINCT o.order_num)
        FROM customer c
        JOIN orders o ON (c.customer_num = o.customer_num)
        GROUP BY c.customer_num
        ORDER BY 1 DESC) AS 'Mayor cantidad de OC',

       (SELECT TOP 1 SUM(i.quantity * i. unit_price)
        FROM items i
        JOIN orders o ON (i.order_num = o.order_num)
        GROUP BY o.customer_num
        ORDER BY 1 DESC) AS 'Mayor total en dólares',

       (SELECT TOP 1 COUNT(o.order_num)
        FROM orders o
        JOIN items i ON (o.order_num = i.order_num)
        GROUP BY o.customer_num
        ORDER BY 1 DESC) AS 'Mayor cantidad de ítems de una OC',

       (SELECT TOP 1 COUNT(DISTINCT o.order_num)
        FROM customer c
        JOIN orders o ON (c.customer_num = o.customer_num)
        GROUP BY c.customer_num
        ORDER BY 1) AS 'Menor cantidad de OC',

       (SELECT TOP 1 SUM(i.quantity * i. unit_price)
        FROM items i
        JOIN orders o ON (i.order_num = o.order_num)
        GROUP BY o.customer_num
        ORDER BY 1) AS 'Menor total en dólares',

       (SELECT TOP 1 COUNT(o.order_num)
        FROM orders o
        JOIN items i ON (o.order_num = i.order_num)
        GROUP BY o.customer_num
        ORDER BY 1) AS 'Menor cantidad de ítems de una OC'
```

6. Para los clientes del estado California(CA) con 4 o más OC cobradas en el 2015 mostrar el total cobrado de esas órdenes y el número de cliente. Solo se deberán mostrar las órdenes de compra cuya cantidad de items total supere a la orden de compra con mayor cantidad de ítems en el estado AZ en el mismo período.

```
SELECT c.customer_num,
       o.order_num,
       SUM(i.unit_price * i.quantity) totalOrdenPorCliente

FROM customer c
    JOIN orders o ON (c.customer_num = o.customer_num)
    LEFT JOIN items i ON (o.order_num = i.order_num)

WHERE c.state = 'CA'
  AND YEAR(o.order_date) = 2015
  AND c.customer_num IN (SELECT c1.customer_num
                         FROM customer c1
                         JOIN orders o1 ON (c1.customer_num = o1.customer_num)
                         WHERE YEAR(o1.order_date) = 2015
                         GROUP BY c1.customer_num
                         HAVING COUNT(*) >= 4)

GROUP BY c.customer_num, o.order_num

HAVING COUNT(i.item_num)  >
           (SELECT TOP 1 COUNT(i3.item_num)
            FROM orders o3
                JOIN items i3 ON (o3.order_num = i3.order_num)
                JOIN customer c3 ON o3.customer_num = c3.customer_num
            WHERE YEAR(o3.order_date) = 2015
              AND c3.state = 'AZ'
            GROUP BY o3.order_num
            ORDER BY COUNT(i3.item_num) DESC)

ORDER BY 1, 2
```

7. Se requiere listar para el Estado de California el par de clientes que sean los que suman el mayor monto en dólares en órdenes de compra, con el formato de salida: 'Código Estado', 'Descripción Estado', 'Apellido, Nombre', 'Apellido, Nombre', 'Total Solicitado'.

- El total solicitado contendrá la suma de los dos clientes.

```
SELECT TOP 1 c1.state,

    sname,

    c1.lname + ', ' + c1.fname,

    c2.lname + ', ' + c2.fname,

    SUM(i1.unit_price) + SUM(i2.unit_price)

FROM customer c1

    INNER JOIN orders o1 ON (c1.customer_num = o1.customer_num)

    INNER JOIN items i1 ON (o1.order_num = i1.order_num)

    INNER JOIN customer c2 ON (c1.state = c2.state AND c1.state = 'CA')

    INNER JOIN orders o2 ON (c2.customer_num = o2.customer_num)

    INNER JOIN items i2 ON (o2.order_num = i2.order_num)

    INNER JOIN state s ON (c1.state = s.state)

WHERE c1.customer_num < c2.customer_num

GROUP BY c1.state, sname, c1.lname, c1.fname, c2.lname, c2.fname

ORDER BY 5 DESC
```

8. Se observa que no se cuenta con stock suficiente para las últimas 5 órdenes de compra emitidas que contengan productos del fabricante 'ANZ'. La decisión tomada es asignarles productos en stock a la orden del cliente que más productos del fabricante 'ANZ', nos haya comprado.

Se solicita listar el número de OC, el código del cliente, la fecha de la orden y la fecha la orden modificada más el lead_time del fabricante más 1 día por preparación del pedido 'Fecha Modificada' salvo para el cliente al que le entregamos los productos en stock, en este caso, la fecha modificada deberá estar en NULL.

- Listar toda la información ordenada por fecha modificada.

```
SELECT DISTINCT o1.order_num,
       o1.customer_num,
       o1.order_date,
       NULL AS 'Fecha Modificada'

FROM orders o1

    JOIN items i1 ON (o1.order_num = i1.order_num)

    JOIN (SELECT TOP 1 o2.customer_num,
                 SUM(unit_price) totcliente
          FROM orders o2
              JOIN items i2 ON (o2.order_num = i2.order_num)
          WHERE i2.manu_code = 'ANZ'
          GROUP BY o2.customer_num
          ORDER BY 2 DESC) sub1 ON (o1.customer_num = sub1.customer_num)

WHERE o1.order_num IN (SELECT DISTINCT TOP 5 o2.order_num
                       FROM orders o2
                            JOIN items i2 ON (o2.order_num = i2.order_num)
                       WHERE i2.manu_code = 'ANZ'
                       ORDER BY o2.order_num DESC)

UNION

SELECT DISTINCT o1.order_num,
       o1.customer_num,
       o1.order_date,
       o1.order_date + m1.lead_time

FROM orders o1

    JOIN items i1 ON (o1.order_num = i1.order_num)

    JOIN (SELECT TOP 1 o2.customer_num,
                 SUM(unit_price) totcliente
          FROM orders o2
            JOIN items i2 ON (o2.order_num = i2.order_num)
          WHERE i2.manu_code = 'ANZ'
          GROUP BY o2.customer_num
          ORDER BY 2 DESC) sub1 ON (o1.customer_num <> sub1.customer_num)

    JOIN manufact m1 ON (i1.manu_code = m1.manu_code)

WHERE o1.order_num IN (SELECT DISTINCT TOP 5 o2.order_num
                       FROM orders o2
                           JOIN items i2 ON (o2.order_num = i2.order_num)
                       WHERE i2.manu_code = 'ANZ'
                       ORDER BY o2.order_num DESC)

  AND i1.manu_code = 'ANZ'

ORDER BY 4
```

9. Listar el Número, nombre, apellido, estado, cantidad de Órdenes, monto total comprado por Cliente durante el año 2015 que no sean del estado de Florida.

Mostrar sólo aquellos clientes cuyo monto total comprado sea mayor que el promedio del monto total comprado por Cliente que no sean del estado Florida.

- Ordenado por total comprado en forma descendente.

```
SELECT c.customer_num,

       c.fname,

       c.lname,

       c.state,

       COUNT(distinct o.order_num) AS 'cant_ordenes',

       SUM(i.quantity * i.unit_price) AS 'monto_total'


FROM customer c

    JOIN orders o ON (c.customer_num = o.customer_num)

    JOIN items i ON (o.order_num = i.order_num)


WHERE c.state <> 'FL'

  AND YEAR(o.order_date) = 2015


GROUP BY c.customer_num, c.fname, c.lname, c.state


HAVING (SUM(i.quantity * i.unit_price))  >

     (SELECT SUM(i2.quantity * i2.unit_price) / COUNT(DISTINCT c2.customer_num)

      FROM orders o2

          JOIN items i2 ON (o2.order_num = i2.order_num)

          JOIN customer c2 ON (o2.customer_num = c2.customer_num)

      WHERE c2.state <> 'FL')


ORDER BY monto_total DESC
```

10. Seleccionar todos los clientes cuyo monto total comprado sea mayor al de su referente durante el año 2015.

Mostrar número, nombre, apellido y los montos totales comprados de ambos durante ese año.

Tener en cuenta que un cliente puede no tener referente y que el referente pudo no haber comprado nada durante el año 2015, mostrarlo igual.

```
SELECT c.customer_num,
       c.fname,
       c.lname,
       SUM(i.unit_price * i.quantity) MontoOrdenes,
       cr.customer_num,
       cr.fname,
       cr.lname,
       cr.totalRef

FROM customer c

    JOIN orders o ON (c.customer_num = o.customer_num)

    JOIN items i ON (o.order_num = i.order_num)

    LEFT JOIN (SELECT r.customer_num,
                      r.fname,
                      r.lname,
                      SUM(i2.unit_price * i2.quantity) totalRef
               FROM customer r
                    LEFT JOIN orders o2 ON (r.customer_num = o2.customer_num)
                    LEFT JOIN items i2 ON (o2.order_num = i2.order_num)
               WHERE YEAR(o2.order_date) = 2015
               GROUP BY r.customer_num, r.fname, r.lname) cr
                                ON (cr.customer_num = c.customer_num_referedBy)

WHERE YEAR(o.order_date) = 2015

GROUP BY c.customer_num, c.fname, c.lname, cr.customer_num, cr.fname, cr.lname,
cr.totalRef

HAVING SUM(i.unit_price * i.quantity) > COALESCE(cr.totalRef, 0)

ORDER BY MontoOrdenes DESC
```