Antes de comenzar la clase una Reflexión:

La Nueva Generación De Padres De Familia.

>>Una Reflexión para Padres e Hijos.<<

Somos de las primeras generaciones de padres decididos a no repetir con los hijos los mismos errores que pudieron haber cometido nuestros progenitores.

Y en el esfuerzo de abolir los abusos del pasado, ahora somos los mas dedicados y comprensivos, pero a la vez lo más débiles e inseguros que ha dado la historia.

Lo grave es que estamos lidiando con unos niños, adolescentes y hasta estudiantes Universitarios mas "igualados", beligerantes y poderosos que nunca existieron.

Parece que en nuestro intento por ser los padres que quisimos tener, pasamos de un extremo al otro. Así que, somos los últimos hijos regañados por los padres y los primeros padres regañados por nuestros hijos.

Los últimos que le tuvimos miedo a nuestros padres y los primeros que tememos a nuestros hijos. Los últimos que crecimos bajo el mando de los padres y los primeros que vivimos bajo el yugo de los hijos.

Lo que es peor, los últimos que respetamos a nuestros padres, y los primeros que aceptamos que nuestros hijos no nos respeten.

En la medida que el permisivismo reemplazó al autoritarismo, los términos de las relaciones familiares han cambiado en forma radical, para bien y para mal.

En efecto, antes se consideraban buenos padres a aquellos cuyos hijos se comportaban bien, obedecían sus ordenes y los trataban con el debido respeto. Y buenos hijos a los niños que eran formales y veneraban a sus padres.

Pero en la medida en que las fronteras jerárquicas entre nosotros y nuestros hijos se han ido desvaneciendo, hoy los buenos

padres son aquellos que logran que sus hijos los amen, aunque poco los respeten.

Y son los hijos quienes ahora esperan el respeto de sus padres, entendiendo por tal que les respeten sus ideas, sus gustos, sus apetencias, sus formas de actuar y de vivir. Y que además les patrocinen lo que necesitan para tal fin.

Como quien dice, los roles se invirtieron, y ahora son los papás quienes tienen que complacer a sus hijos para ganárselos, y no a la inversa, como en el pasado.

Esto explica el esfuerzo que hoy hacen tantos papás y mamás por ser los mejores amigos de sus hijos y parecerles “muy cool” a sus hijos.

Sé a dicho que los extremos se tocan, y si el autoritarismo del pasado llenó a los hijos de temor hacia sus padres, la debilidad del presente los llena de miedo y menosprecio al vernos tan débiles y perdidos como ellos.

Los hijos necesitan percibir en la niñez que estamos a la cabeza de sus vidas como lideres, capaces de sujetarlos cuando no se pueden contener y de guiarlos mientras no saben para donde van.

Si bien el autoritarismo aplasta, el permisivismo ahoga

Solo una actitud firme y respetuosa les permitirá confiar en nuestra idoneidad para gobernar sus vidas mientras sean menores, porque vamos adelante liderándolos y no atrás cargándolos y rendidos a su voluntad.

Es así como evitaremos que las nuevas generaciones se ahoguen en el descontrol y hastío en el que se esta hundiendo la sociedad que parece ir a la deriva, sin parámetros, ni destino.

CONSULTANDO VARIAS TABLAS (JOIN).

La técnica para Consultar información contenida en varias Tablas, se conoce en términos de SQL como JOIN. Se puede decir entonces que: Un JOIN es la forma que utiliza el SELECT para combinar filas(Registros) de dos o más tablas.

JOIN SIMPLE

Cuando se hace un SELECT de dos o mas Tablas, un JOIN Simple, tiene la capacidad de generar Información mediante la Relación de Columnas(Campos) comunes en las Tablas relacionadas.

Sintaxis

```
SELECT   [ table. ]  column, [table,] column. . .
FROM  table1, table2. . .
WHERE  [   table1.  ]  column  =  [ table2. ] column
```

Ejemplo

Mostrar el Código de Cliente, Nombre de Cliente, Número de Facturas asociadas a cada Cliente, Fecha de cada Factura y Monto Total de Cada Factura.

Para ejecutar esta Instrucción es necesario invocar a varias Tablas en el FROM. Luego es necesario en el WHERE igualar las Columnas(Campos) que hacen la

Conexión entre las Tablas involucradas. En este caso, las Columnas que permiten la Conexión son: código y codcli, es decir cuando ellas dos son iguales se sabe que una Factura Y le pertenece a un Cliente X.

```
SQL> select codigo,nombre,numero,codcli,fecha,montotal
  2  from clientes,facturas
  3  where codigo=codcli
  4  and clientes.estatus='A'
  5  and facturas.estatus='A';

CODI NOMBRE                     NUME CODC FECHA      MONTOTAL
---- ------------------------- ---- ---- -------- ----------
0002 BECO                      1855 0002 07/01/04        4000
0003 EPA                       1852 0003 05/01/04        6000
0004 UCLA                      1851 0004 05/01/04        4500
0004 UCLA                      1854 0004 06/01/04        1000
0006 LOCATEL                   1857 0006 08/01/04        4500
0007 SIDETUR                   1853 0007 06/01/04        7500
0007 SIDETUR                   1856 0007 07/01/04        2000
0008 RENE DESSES               1859 0008 09/01/04       10000
0008 RENE DESSES               1861 0008 11/01/04        5000
0008 RENE DESSES               1862 0008 12/01/04        7500
0009 BANCOR                    1858 0009 09/01/04        7500

CODI NOMBRE                     NUME CODC FECHA      MONTOTAL
---- ------------------------- ---- ---- -------- ----------
0009 BANCOR                    1860 0009 10/01/04        2500

12 filas seleccionadas.
```

Es importante notar que si dos Tablas tienen Columnas con un mismo nombre, es necesario colocar un prefijo con el nombre de la Tabla para que ORACLE sepa a que columna se está haciendo referencia. Tal es el caso del campo estatus el cual se encuentra en ambas Tablas y posee el mismo nombre.

En la Consulta no es estrictamente necesario mostrar la Columna codcli, pero se hizo para validar que las columnas de conexión son efectivamente iguales.

De esta Manera, se puede combinar de una forma fácil la información de varias columnas a la vez(Dos, Tres, Cuatro, Cinco y todas las que se deseen en una sola Consulta).

PRODUCTO CARTESIANO.

Todo profesional con experticia en SQL, debe conocer este Concepto y tomar conciencia de los enormes problemas que su existencia puede ocasionar en cualquier Sistema. El Producto Cartesiano puede aparecer cuando se hacen consultas de varias Tablas y debe ser evitado en todo momento.

Concepto

De una manera sencilla, se puede definir el Producto Cartesiano de la siguiente manera: Se tiene dos Conjuntos A y B. El conjunto A={a,b,c}. Y el Conjunto B={1,2}. El Producto Cartesiano AxB se define como la relación del Primer Elemento del Conjunto A con cada uno de los Elementos del Conjunto B. Al establecerse esa relación se generarían las Tuplas [a,1], es decir, la Relación del Primer elemento del Conjunto A con el primer elemento del Conjunto B. Luego se generarían las Tuplas [a,2],[b,1], [b,2],[c,1],[c,2].

¿Como se genera un Producto Cartesiano en SQL?

Obviando la igualdad de las Columnas(Campos) que establecen la Relación entre las Tablas.

Un ejemplo de Producto Cartesiano entre las Tablas de Clientes y de Facturas sería, la siguiente Instrucción:

```
SQL> select codigo,nombre,numero,codcli,fecha,montotal
  2  from clientes,facturas
  3  where facturas.estatus='A'
  4  and clientes.estatus='A';

CODI NOMBRE                     NUME CODC FECHA      MONTOTAL
---- ------------------------- ---- ---- -------- ----------
0001 MAKRO                     1851 0004 05/01/04       4500
0004 UCLA                      1851 0004 05/01/04       4500
0005 EXITO                     1851 0004 05/01/04       4500
0006 LOCATEL                   1851 0004 05/01/04       4500
0002 BECO                      1851 0004 05/01/04       4500
0008 RENE DESSES               1851 0004 05/01/04       4500
0007 SIDETUR                   1851 0004 05/01/04       4500
0009 BANCOR                    1851 0004 05/01/04       4500
0010 UPEL                      1851 0004 05/01/04       4500
0003 EPA                       1851 0004 05/01/04       4500
0099 Regional Ligth            1851 0004 05/01/04       4500

CODI NOMBRE                     NUME CODC FECHA      MONTOTAL
---- ------------------------- ---- ---- -------- ----------
0098 Solera                    1851 0004 05/01/04       4500
0055 POLAR                     1851 0004 05/01/04       4500
0001 MAKRO                     1852 0003 05/01/04       6000
0004 UCLA                      1852 0003 05/01/04       6000
0005 EXITO                     1852 0003 05/01/04       6000
0006 LOCATEL                   1852 0003 05/01/04       6000
0002 BECO                      1852 0003 05/01/04       6000
0008 RENE DESSES               1852 0003 05/01/04       6000
0007 SIDETUR                   1852 0003 05/01/04       6000
0009 BANCOR                    1852 0003 05/01/04       6000
0010 UPEL                      1852 0003 05/01/04       6000

CODI NOMBRE                     NUME CODC FECHA      MONTOTAL
---- ------------------------- ---- ---- -------- ----------
0003 EPA                       1852 0003 05/01/04       6000
```

Por razones de espacio no se muestra la salida Completa, la cual ocuparía 4 páginas de este texto. Lo importante que hay que resaltar aquí, es que el haber obviado la igualdad entre las Columnas código y codcli, originó como se puede notar que el SQL relacione al Cliente 0001 MAKRO con la Factura 1851 lo cual no es correcto porque esa factura le corresponde al Cliente 0004 UCLA. Peor aún, SQL relacionó la Factura 1851 con cada uno de los Clientes registrados, como se puede notar. Luego la Factura 1852 también la relacionó con cada uno de los Clientes existentes y así sucesivamente. Es decir la salida es completamente incorrecta. Adicionalmente hay que notar que una salida correcta de 12 filas se convirtió en una salida incorrecta de muchas filas mas.

Aunado al Problema originado por la salida, se presenta otro problema sumamente grave: Los manejadores cuando hacen operaciones con Tablas montan la Información contenida en las mismas en Memoria Principal para optimizar la velocidad de búsqueda. El resultado de la combinación de las tablas también se monta en Memoria Principal. Quiere decir que si un usuario genera Producto Cartesiano constante, es capaz de agotar la Memoria RAM del equipo donde esté trabajando y traerle a ese Computador problemas de lentitud y hasta colapso en sus operaciones.

Imagine lo que pasaría si en vez de ser 10 Clientes, existiera una Tabla con 100 Mil Clientes. Y si esos Clientes en vez de tener 12 Facturas tuvieran 120 Mil

Facturas. El Producto cartesiano generado probablemente se habría consumido una porción muy importante de cualquier Computador.

MORALEJA FINAL: El Producto cartesiano debe ser evitado en lo posible para evitar salidas inadecuadas y Problemas de Velocidad de Respuesta.

¿Qué se debe hacer al Respecto?

Examinemos la siguiente Instrucción:

```
SQL> select codigo,nombre,numero,codcli,fecha,montotal
  2  from clientes,facturas
  3  where codigo=codcli
  4  and clientes.estatus='A'
  5  and facturas.estatus='A';

CODI NOMBRE                    NUME CODC FECHA      MONTOTAL
---- ------------------------- ---- ---- -------- ----------
0002 BECO                      1855 0002 07/01/04       4000
0003 EPA                       1852 0003 05/01/04       6000
0004 UCLA                      1851 0004 05/01/04       4500
0004 UCLA                      1854 0004 06/01/04       1000
0006 LOCATEL                   1857 0006 08/01/04       4500
0007 SIDETUR                   1853 0007 06/01/04       7500
0007 SIDETUR                   1856 0007 07/01/04       2000
0008 RENE DESSES               1859 0008 09/01/04      10000
0008 RENE DESSES               1861 0008 11/01/04       5000
0008 RENE DESSES               1862 0008 12/01/04       7500
0009 BANCOR                    1858 0009 09/01/04       7500

CODI NOMBRE                    NUME CODC FECHA      MONTOTAL
---- ------------------------- ---- ---- -------- ----------
0009 BANCOR                    1860 0009 10/01/04       2500

12 filas seleccionadas.
```

Note que aquí se le dice a ORACLE que restrinja la relación entre las Columnas al cumplirse que la columna código sea igual a la columna codcli. Pero, si no se establece esa igualdad, ORACLE trataría de relacionar todas las columnas de la Primera Tabla con cada una de las Columnas de la Segunda Tabla, como sucedió en el Ejemplo anterior.

VOLVIENDO AL "JOIN" SIMPLE

Con el uso de este JOIN, se puede obtener información adicional mediante la obtención de varias tablas.

Ejemplo con 3 Tablas.

Mostrar el Código del Cliente, Nombre del Cliente, Número de Factura, Fecha de la Factura, Código de cada Artículo de la Factura, Cantidad, Precio y Subtotal. Ordenado por Cliente.

```
SQL> select codigo,substr(nombre,1,10),numero,fecha,codart,cantidad,precio,
  2  (cantidad*precio) as subtotal
  3  from clientes, facturas,detfacturas
  4  where codigo=codcli
  5  and numero=numfac
  6  and clientes.estatus='A'
  7  and facturas.estatus='A'
  8  order by codigo;
```

```
CODI SUBSTR(NOM NUME FECHA    CODA   CANTIDAD     PRECIO    SUBTOTAL
---- ---------- ---- -------- ---- ---------- ---------- ----------
0002 BECO        1855 07/01/04 0002          1       2500       2500
0002 BECO        1855 07/01/04 0001          1       1500       1500
0003 EPA         1852 05/01/04 0001          4       1500       6000
0004 UCLA        1851 05/01/04 0002          1       2500       2500
0004 UCLA        1851 05/01/04 0003          2       1000       2000
0004 UCLA        1854 06/01/04 0001          1       1000       1000
0006 LOCATEL     1857 08/01/04 0005         20        200       4000
0006 LOCATEL     1857 08/01/04 0008          1        500        500
0007 SIDETUR     1853 06/01/04 0004          3       2500       7500
0007 SIDETUR     1853 06/01/04 0004          3       2500       7500
0007 SIDETUR     1856 07/01/04 0005          5        200       1000

CODI SUBSTR(NOM NUME FECHA    CODA   CANTIDAD     PRECIO    SUBTOTAL
---- ---------- ---- -------- ---- ---------- ---------- ----------
0007 SIDETUR     1856 07/01/04 0005          5        200       1000
0007 SIDETUR     1856 07/01/04 0006          4        100        400
0007 SIDETUR     1856 07/01/04 0006          4        100        400
0007 SIDETUR     1856 07/01/04 0007          1        600        600
0007 SIDETUR     1856 07/01/04 0007          1        600        600
0008 RENE DESSE  1859 09/01/04 0002          4       2500      10000
0008 RENE DESSE  1861 11/01/04 0004          1       2500       2500
0008 RENE DESSE  1861 11/01/04 0002          1       2500       2500
0008 RENE DESSE  1862 12/01/04 0006          4        100        400
0008 RENE DESSE  1862 12/01/04 0001          2       1500       3000
0008 RENE DESSE  1862 12/01/04 0004          1       2500       2500

CODI SUBSTR(NOM NUME FECHA    CODA   CANTIDAD     PRECIO    SUBTOTAL
---- ---------- ---- -------- ---- ---------- ---------- ----------
0008 RENE DESSE  1862 12/01/04 0008          2        500       1000
0008 RENE DESSE  1862 12/01/04 0007          1        600        600
0009 BANCOR      1858 09/01/04 0002          2       2500       5000
0009 BANCOR      1860 10/01/04 0006          5        100        500
0009 BANCOR      1860 10/01/04 0003          1       1000       1000
0009 BANCOR      1860 10/01/04 0008          2        500       1000
0009 BANCOR      1858 09/01/04 0004          1       2500       2500

29 filas seleccionadas.
```

Ejemplo con 4 Tablas.

Mostrar Código del Cliente, Nombre de Cliente, Número de Factura, Código de cada Artículo de la Factura, Descripción de Cada Artículo, Cantidad, Precio y Subtotal. Ordenado por Cliente.

```
SQL> select clientes.codigo,substr(nombre,1,10),numero,codart,substr(descripcion,1,10),
  2  cantidad,precio,(cantidad*precio) as subtotal
  3  from clientes, facturas,detfacturas,articulos
  4  where clientes.codigo=codcli
  5  and numero=numfac
  6  and codart=articulos.codigo
  7  and clientes.estatus='A'
  8  and facturas.estatus='A'
  9  and articulos.estatus='A'
 10  order by numero;
```

```
CODI SUBSTR(NOM NUME CODA SUBSTR(DES   CANTIDAD     PRECIO   SUBTOTAL

0004 UCLA       1851 0002 Enfriador           1       2500       2500
0004 UCLA       1851 0003 Dvd                 2       1000       2000
0003 EPA        1852 0001 Nevera              4       1500       6000
0007 SIDETUR    1853 0004 Cava Cuart          3       2500       7500
0007 SIDETUR    1853 0004 Cava Cuart          3       2500       7500
0004 UCLA       1854 0001 Nevera              1       1000       1000
0002 BECU       1855 0001 Nevera              1       1500       1500
0002 BECU       1855 0002 Enfriador           1       2500       2500
0007 SIDETUR    1856 0005 Ventilador          5        200       1000
0007 SIDETUR    1856 0005 Ventilador          5        200       1000
0007 SIDETUR    1856 0006 Ventilador          4        100        400

CODI SUBSTR(NUM NUME CODA SUBSTR(DES   CANTIDAD     PRECIO   SUBTOTAL
---- ---------- ---- ---- ---------- ---------- ---------- ----------
0007 SIDETUR    1856 0006 Ventilador          4        100        400
0007 SIDETUR    1856 0007 Licuadora           1        600        600
0007 SIDETUR    1856 0007 Licuadora           1        600        600
0006 LOCATEL    1857 0005 Ventilador         20        200       4000
0006 LOCATEL    1857 0008 Quemador d          1        500        500
0009 BANCOR     1858 0002 Enfriador           2       2500       5000
0009 BANCOR     1858 0004 Cava Cuart          1       2500       2500
0008 RENE DESSE 1859 0002 Enfriador           4       2500      10000
0009 BANCOR     1860 0003 Dvd                 1       1000       1000
0009 BANCOR     1860 0006 Ventilador          5        100        500
0009 BANCOR     1860 0008 Quemador d          2        500       1000

CODI SUBSTR(NOM NUME CODA SUBSTR(DES   CANTIDAD     PRECIO   SUBTOTAL
---- ---------- ---- ---- ---------- ---------- ---------- ----------
0008 RENE DESSE 1861 0002 Enfriador           1       2500       2500
0008 RENE DESSE 1861 0004 Cava Cuart          1       2500       2500
0008 RENE DESSE 1862 0006 Ventilador          4        100        400
0008 RENE DESSE 1862 0007 Licuadora           1        600        600
0008 RENE DESSE 1862 0004 Cava Cuart          1       2500       2500
0008 RENE DESSE 1862 0008 Quemador d          2        500       1000
0008 RENE DESSE 1862 0001 Nevera              2       1500       3000

29 filas seleccionadas.
```

JOIN EXTERNO

Un JOIN externo combina información de dos o más tablas, retornando inclusive aquellas filas que están en una tabla y que no están en la otra.

Ejemplo

Cuando se utilizó el JOIN SIMPLE para obtener a los Clientes con sus respectivas Facturas, puede notarse que el Cliente 0005 ÉXITO no aparece en la Salida, en

vista de que no posee actualmente Facturas asociadas. Si el usuario desea obtener la Información, y que salgan los clientes que no tengan facturas asociadas, se debe utilizar un JOIN EXTERNO.

```
SQL>  select codigo,nombre,numero,codcli,montotal from
  2   clientes, facturas
  3   where codigo=codcli(+)
  4   and clientes.estatus='A';

CODI NOMBRE                    NUME CODC   MONTOTAL
---- ------------------------ ---- ---- ----------
0001 MAKRO
0002 BECO                      1855 0002       4000
0003 EPA                       1852 0003       6000
0004 UCLA                      1851 0004       4500
0004 UCLA                      1854 0004       1000
0005 EXITO
0006 LOCATEL                   1857 0006       4500
0007 SIDETUR                   1853 0007       7500
0007 SIDETUR                   1856 0007       2000
0007 SIDETUR                   1853 0007       7500
0007 SIDETUR                   1856 0007       2000

CODI NOMBRE                    NUME CODC   MONTOTAL
---- ------------------------ ---- ---- ----------
0008 RENE DESSES               1859 0008      10000
0008 RENE DESSES               1861 0008       5000
0008 RENE DESSES               1862 0008       7500
0009 BANCOR                    1858 0009       7500
0009 BANCOR                    1860 0009       2500
0010 UPEL
0055 POLAR
0098 Solera
0099 Regional Ligth

20 filas seleccionadas.
```

Debe notarse que el Cliente 0005 Éxito, ahora si está presente en la salida ejecutada.

SubQuery

Los SubQuerys permiten realizar comparaciones entre valores de una misma Tabla. Por ejemplo una persona puede indagar acerca de los trabajadores que ganan un sueldo superior al suyo.

Ejemplo

Listar los Clientes cuyo Límite de Crédito es superior al Cliente BECO (Código=0002).

```
SQL> select codigo, nombre,limcre
  2  from clientes
  3  where limcre > (select limcre from clientes where codigo='0002'
  4                  and estatus='A')
  5  order by codigo;

CODI NOMBRE                         LIMCRE
---- ------------------------- ----------
0003 EPA                             2800
0005 EXITO                           7800
0006 LOCATEL                         7000
0007 SIDETUR                         7500
0007 SIDETUR                         7500
0008 RENE DESSES                     9500
0009 BANCOR                          9900
0098 Solera                          9000
0099 Regional Ligth                  8000

9 filas seleccionadas.
```

UNION, INTERSECT y MINUS

Combinan el resultado de dos o mas Querys en uno solo. Es necesario tomar en cuenta que:

1. Las Columnas que forman parte de los diferentes SELECT deben ser del mismo tipo.
2. La cantidad de Columnas deben ser iguales.

3. Si se utiliza la cláusula ORDER BY, se debe especificar el número de la columna.
4. En los tres operadores existe un DISTINCT implícito.

UNION

Permite integrar información de dos tablas en un mismo listado de manera consecutiva. Al respecto, retorna los valores que se encuentran en cualquiera de las tablas involucradas. Obedece a la operación de UNION de conjuntos(“Los que se encuentran en A o se encuentran en B”).

Ejemplo

Listar a los Clientes y a los Proveedores en una misma salida. Ahora bien, se debe recordar que los proveedores fueron creados a partir de la tabla de clientes utilizando un Query y una secuencia. Los Proveedores son:

CODI	NOMBRE	LIMCRE	E
1	SIDETUR	7500	A
2	MAKRO		A
3	UCLA	1100	A
4	EXITO	7800	A
5	LOCATEL	7000	A
6	BECO	2500	A
7	RENE DESSES	9500	A
8	SIDETUR	7500	A
9	BANCOR	9900	A
10	UPEL	1100	A
11	EPA	2800	A

CODI	NOMBRE	LIMCRE	E
12	Regional Ligth	8000	A
13	Solera	9000	A
14	POLAR	2300	A

Al realizar la operación de UNION nos quedaría:

```
SQL> select codigo,nombre
  2  from clientes
  3  union
  4  select codigo,nombre
  5  from proveedores;
```

Y la salida correspondiente sería:

```
CODI NOMBRE
---- ----------------
0001 MAKRO
0002 BECO
0003 EPA
0004 UCLA
0005 EXITO
0006 LOCATEL
0007 SIDETUR
0008 RENE DESSES
0009 BANCOR
0010 UPEL
0055 POLAR

CODI NOMBRE
---- ----------------
0098 Solera
0099 Regional Ligth
1    SIDETUR
10   UPEL
11   EPA
12   Regional Ligth
13   Solera
14   POLAR
2    MAKRO
3    UCLA
4    EXITO

CODI NOMBRE
---- ----------------
5    LOCATEL
6    BECO
7    RENE DESSES
8    SIDETUR
9    BANCOR
```

INTERSECT

Retorna aquellas Filas que se encuentran en la 1ª y en la 2ª. tabla. Para notar la diferencia vamos a agregar un nuevo proveedor., el cual será BECO código 0002 (Ya BECO existe como Proveedor con código 6), pero es necesario colocarlo con el mismo código de cliente para probar la intersección.

```
SQL> insert into proveedores
  2  select *
  3  from clientes
  4  where codigo='0002';

1 fila creada.

SQL> select * from proveedores;

CODI NOMBRE                         LIMCRE E
---- ------------------------- ---------- -
1    SIDETUR                          7500 A
2    MAKRO                                 A
3    UCLA                             1100 A
4    EXITO                            7800 A
5    LOCATEL                          7000 A
6    BECO                             2500 A
7    RENE DESSES                      9500 A
8    SIDETUR                          7500 A
9    BANCOR                           9900 A
10   UPEL                             1100 A
11   EPA                              2800 A

CODI NOMBRE                         LIMCRE E
---- ------------------------- ---------- -
12   Regional Ligth                   8000 A
13   Solera                           9000 A
14   POLAR                            2300 A
0002 BECO                             2500 A
```

Al realizar la intersección, nos queda:

```
SQL> select codigo,nombre
  2  from clientes
  3  intersect
  4  select codigo,nombre
  5  from proveedores;

CODI NOMBRE
---- -------------------------
0002 BECO
```

Beco es la única fila que se encuentra en ambas tablas.

MINUS

Retorna aquellas Filas que se encuentran en la Primera Tabla y no están en la segunda tabla.

Ejemplo

Listar los Artículos que no han sido facturados. Para ello se listan todos los Artículos registrados en esa tabla, excepto aquellos que aparecen en la Tabla DetFacturas. Por la manera como se encuentran cargadas de información la tablas, todos los artículos han sido facturados. En consecuencia, se necesita ingresar un nuevo artículo.

```
SQL> insert into articulos
  2  values('0009','Porta Vasos', 140, 10000,'A');

1 fila creada.

SQL> select * from articulos;

CODI DESCRIPCION                    EXISTENCIA      COSTO E
---- ------------------------------ ---------- ---------- -
0009 Porta Vasos                           140      10000 A
0001 Nevera                                230        500 A
0002 Enfriador                             110       1630 A
0003 Dvd                                   615        420 A
0004 Cava Cuarto Pequeña                     4       1300 A
0005 Ventilador Mediano                    700         53 A
0006 Ventilador Pequeño                    518         29 A
0007 Licuadora                             200        459 A
0008 Quemador de Cds                       133        391 A
```

Los artículos facturados son:

```
SQL> select * from detfacturas;

NUMF CODA   CANTIDAD     PRECIO
---- ---- ---------- ----------
1851 0002          1       2500
1851 0003          2       1000
1852 0001          4       1500
1853 0004          3       2500
1854 0001          1       1000
1855 0002          1       2500
1855 0001          1       1500
1856 0005          5        200
1856 0006          4        100
1856 0007          1        600
1857 0005         20        200

NUMF CODA   CANTIDAD     PRECIO
---- ---- ---------- ----------
1858 0002          2       2500
1858 0004          1       2500
1859 0002          4       2500
1860 0006          5        100
1860 0003          1       1000
1860 0008          2        500
1861 0004          1       2500
1861 0002          1       2500
1862 0007          1        600
1862 0006          4        100
1862 0008          2        500

NUMF CODA   CANTIDAD     PRECIO
---- ---- ---------- ----------
1862 0001          2       1500
1862 0004          1       2500
1857 0008          1        500
```

Por ende al realizar el MINUS nos queda:

```
SQL>  select codigo
  2   from articulos
  3   MINUS
  4   select codart
  5   from detfacturas;

CODI
----
0009
```

Porque el Artículos cuyo código es 0009 es el único que no ha sido facturado.

PRACTICA III

1. Analice las desventajas del Producto Cartesiano y explique porque es inconveniente.
2. Elabore una Consulta que traiga el código y nombre de un cliente, el código y la descripción de los productos que le han sido facturados, la existencia que tenía ese producto antes de la elaboración de esa factura.
3. Muestre las Facturas que sean mas antiguas que la facturas Número 1856.
4. Elabore una Consulta que especifique los Clientes que no poseen facturas.

DESPUÉS DE ESTA CLASE TANNNNNNN!!!!!! LARGA: Algo de RELAX.

En esta oportunidad, la Viejita fue a la cárcel del pueblo dispuesta a entrar a la visita conyugal.

A su llegada, comienza la siguiente conversación:

Vigilante: Disculpe Abuela que desea?

Viejita: Vengo a una Visita Conyugal.

Vigilante Extrañado: Pero a quién visitará si usted no tiene ningún esposo o novio en esta cárcel.

Viejita: No importa Mijo a cualquiera!!!!!!!!!!!!!!