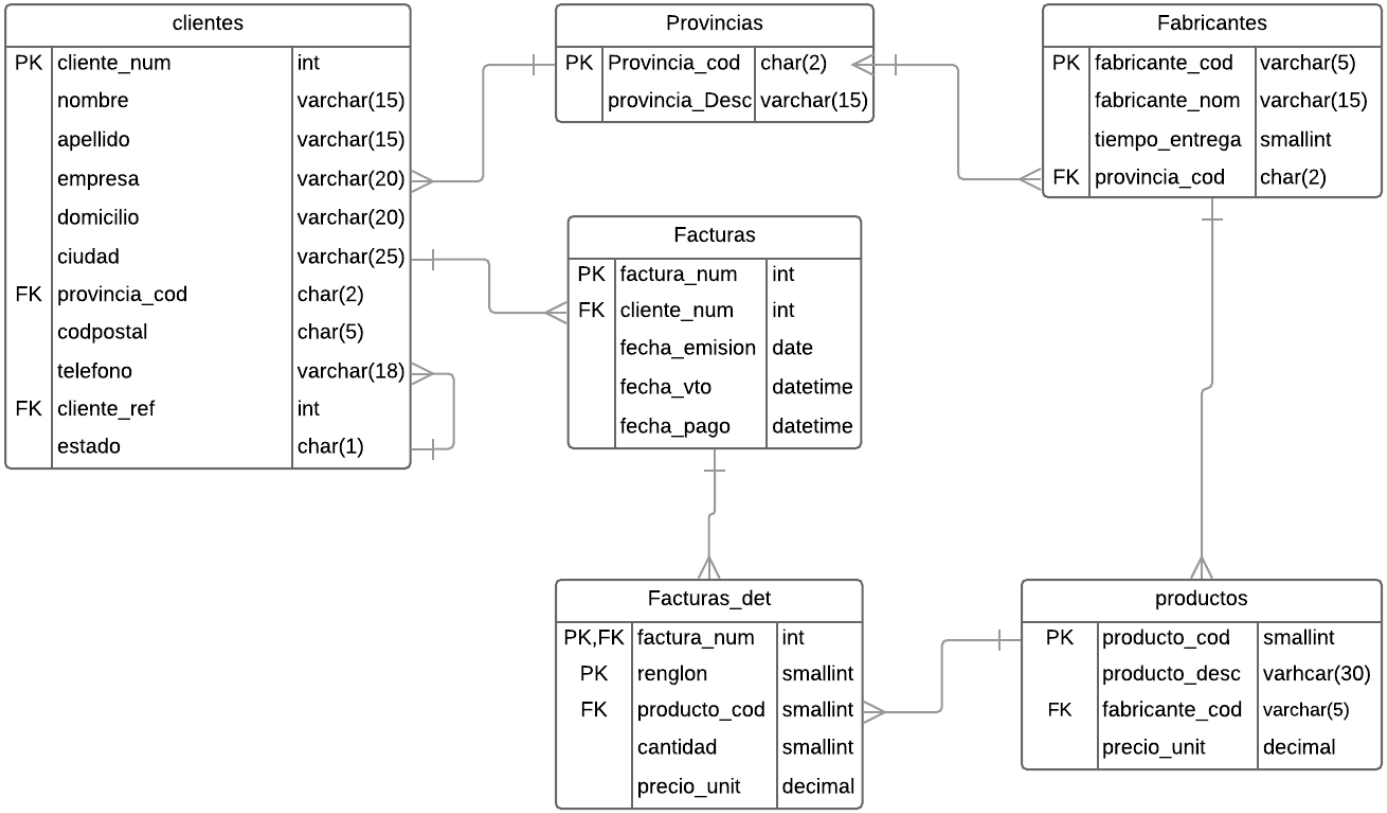
BASES DE DATOS

SQL - Subqueries

Agenda

- Introducción
- Subqueries
- WITH
- Queries recursivos

DER


clientes
PK cliente_num int
nombre varchar(15)
apellido varchar(15)
empresa varchar(20)
domicilio varchar(20)
ciudad varchar(25)
FK provincia_cod char(2)
codpostal char(5)
telefono varchar(18)
FK cliente_ref int
estado char(1)
Provincias
PK Provincia_cod char(2)
provincia_Desc varchar(15)
Fabricantes
PK fabricante_cod varchar(5)
fabricante_nom varchar(15)
tiempo_entrega smallint
FK provincia_cod char(2)
Facturas
PK factura_num int
FK cliente_num int
fecha_emision date
fecha_vto datetime
fecha_pago datetime
Facturas_det
PK,FK factura_num int
PK renglon smallint
FK producto_cod smallint
cantidad smallint
precio_unit decimal
productos
PK producto_cod smallint
producto_desc varhcar(30)
FK fabricante_cod varchar(5)
precio_unit decimal

SELECT - Subqueries

SELECT columna [,columna] Subquery

FROM tabla [JOIN tabla …] Subquery

[WHERE condición] Subquery

[GROUP BY columna [, columna]]

[HAVING condición] Subquery

[ORDER BY columna [, columna]]

SELECT - Subqueries

“

```
SELECT Producto_cod, producto_desc, fabricante_cod,
        (select fabricante_nom from fabricantes f
        where f.fabricante_cod = p.fabricante_cod) desc
  FROM productos p
```

SELECT - Subqueries

“

```
SELECT factura_num, fecha_emision
  FROM facturas f join (Select factura_num, sum(cantidad)
                          from facturas_det
                         group by facturas_num
                         having sum(cantidad) > 100) fd
                on f.factura_num = fd.factura_num
```

SELECT - Subqueries

“

```
SELECT factura_num, fecha_emision
   FROM facturas f
WHERE fecha_emisión = (select min(fecha_emision)
                       from facturas)
```

SELECT – ANY, ALL

“

SELECT [*columnas*]
FROM [*tablas*]
where *valorEscalar* **[condicion]** (**subquery)**

Condicion
[=, <>, !=, >, >=, <, <=] [ANY/SOME/ALL]

SELECT – ANY, ALL

fabricante_cod	tiempo_entrega	provincia_cod
DOTO	3	CO
EXPO	NULL	BA
GAMA	8	BA
HILO	7	BA
IMPO	NULL	BA
JAVA	4	BA
LACO	4	BA
ZICO	8	CO

```
SELECT fabricante_cod, tiempo_entrega
  FROM fabricantes
 where provincia_cod = 'CO' and
          tiempo_entrega < ANY
       (select tiempo_entrega from fabricantes
        where provincia_cod = 'BA')
```

fabricante_cod	tiempo_entrega
DOTO	3

SELECT – ANY, ALL

fabricante_cod	tiempo_entrega	provincia_cod
DOTO	3	CO
EXPO	NULL	BA
GAMA	8	BA
HILO	7	BA
IMPO	NULL	BA
JAVA	4	BA
LACO	4	BA
ZICO	8	CO

```
SELECT fabricante_cod, tiempo_entrega
  FROM fabricantes
 where provincia_cod = 'CO' and
       tiempo_entrega >= ALL
        (select tiempo_entrega from fabricantes
          where provincia_cod = 'BA')
```

Qué pasó ?

Resultado?

Messages

fabricante_cod	tiempo_entrega

SELECT – ANY, ALL

fabricante_cod	tiempo_entrega	provincia_cod
DOTO	3	CO
EXPO	NULL	BA
GAMA	8	BA
HILO	7	BA
IMPO	NULL	BA
JAVA	4	BA
LACO	4	BA
ZICO	8	CO

```
SELECT fabricante_cod, tiempo_entrega
  FROM fabricantes
 where provincia_cod = 'CO' and
       tiempo_entrega >= ALL
        (select tiempo_entrega from fabricantes
          where provincia_cod = 'BA' and
        tiempo_entrega is not null)
```

fabricante_cod	tiempo_entrega
ZICO	8

SELECT - Subqueries

“

```
SELECT factura_num, sum(cantidad)
   FROM facturas_det
GROUP BY factura_num
HAVING sum(cantidad) >
          (select sum(cantidad) /  count(distinct factura_num)
              from facturas_det)
```

SELECT – Queries correlacionados

“

```
SELECT cliente_num, nombre, apellido
  FROM clientes c
WHERE NOT EXISTS (Select 1 from facturas f
                  where f.cliente_num = c.cliente_num)
```

Selecciona aquellos clientes que no tienen facturas.

SELECT – Queries correlacionados

“

```
SELECT cliente_num, nombre, apellido
   FROM clientes c
WHERE EXISTS (Select 1 from facturas f
                        where f.cliente_num = c.cliente_num)
```

Selecciona aquellos clientes que tienen facturas.

SELECT – Clausula WITH

Se utiliza para ejecutar ***queries*** descomponiéndolos en partes (subqueries) para hacerlos más sencillos.

A este tipo de queries se lo conoce como Common Table Expression (CTE)

```
WITH deptos AS ( select d.nro_depto, e.legajo
                   from departamentos d join empleados e
                             on d.nro_depto = e.nro_depto
                  where d.presupuesto > 10000
                    and e.FechaNacimiento > '20000101'
) select deptos.legajo, e2.nombre
    from deptos join empleados e2 on deptos.nro_depto = e2.nro_depto;
```

SELECT – Clausula WITH

```
with clientesABC as (select f.cliente_num, fd.producto_cod, sum(fd.cantidad) tot
                       from facturas f join facturas_det fd
                                         on f.factura_num = fd.factura_num
                     group by f.cliente_num, fd.producto_cod
                       having sum(fd.cantidad) > 40),
  fabricantesABC as (select fa.fabricante_cod, fd.producto_cod, sum(fd.cantidad) tot
                       from facturas_det fd JOIN productos p
                                         on fd.producto_cod = p.producto_cod
                                       JOIN fabricantes fa
                                         on p.fabricante_cod = fa.fabricante_cod
                     group by fa.fabricante_cod, fd.producto_cod
                       having sum(fd.cantidad) > 100)
 select coalesce(c.producto_cod, f.producto_cod) producto,
        c.cliente_num, c.tot, f.fabricante_cod, f.tot
   from clientesABC c full join fabricantesABC f ON c.producto_cod = f.producto_cod
```

¿Qué hace este query?

SELECT – Clausula WITH

```
with clientesABC as (select f.cliente_num, fd.producto_cod, sum(fd.cantidad) tot
                       from facturas f join facturas_det fd
                                         on f.factura_num = fd.factura_num
```

fabricantesABC

Results | Messages

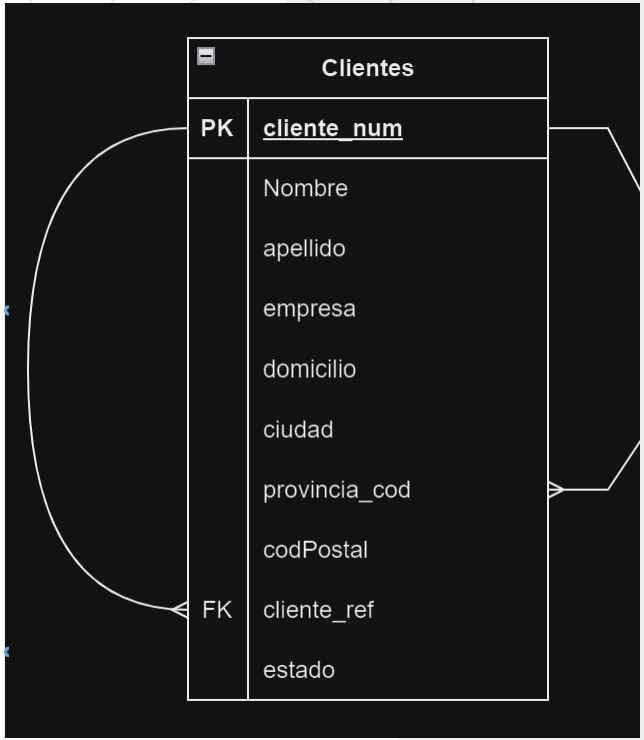
	producto	cliente_num	total	fabricante_cod	total
1	1003	114	50	NULL	NULL
2	1005	103	90	DOTO	195
3	1006	NULL	NULL	EXPO	188
4	1004	NULL	NULL	CASA	120

```
m(fd.cantidad) tot

 p.producto_cod

 fa.fabricante_cod
                     group by fa.fabricante_cod, fd.producto_cod
                    having sum(fd.cantidad) > 100)
 select coalesce(c.producto_cod, f.producto_cod) producto,
        c.cliente_num, c.tot, f.fabricante_cod, f.tot
   from clientesABC c full join fabricantesABC f ON c.producto_cod = f.producto_cod
```

SELECT – Recursivos

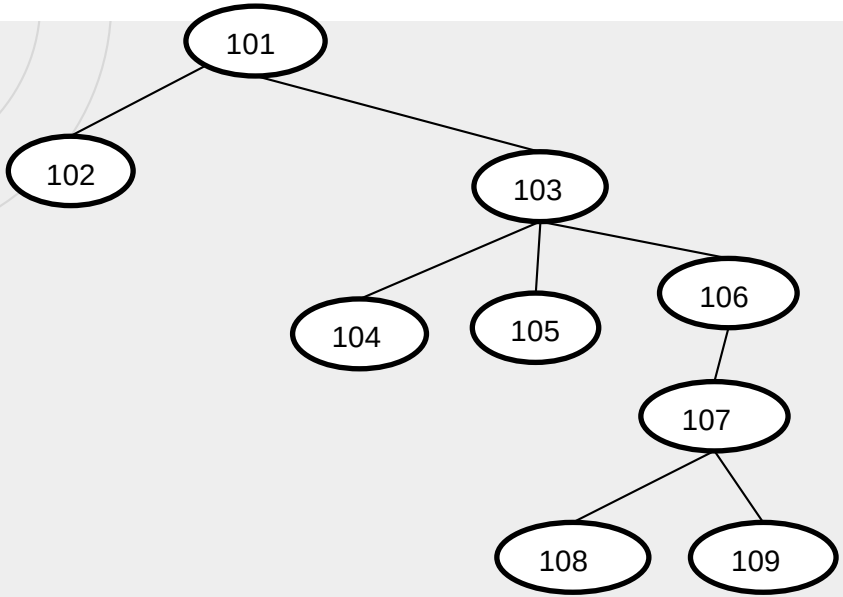


En ocasiones se necesita consultar información que está organizada en forma jerárquica.

Posee una relación recursiva la cual puede ser de varios niveles.

SELECT – Recursivos

En la figura se pueden ver las relaciones existentes entre clientes:



SELECT – Recursivos

Dado el Referente 101 necesitamos obtener todos sus referidos.

```
WITH CTERecursivo AS -- CTE common table expression
( -- SELECT RAIZ (ROOT)
 SELECT a.cliente_num, a.nombre, a.apellido, a.cliente_ref, r.nombre, r.apellido,
        0 as LEVEL, RIGHT('000000' + CAST(a.cliente_num AS varchar(MAX)), 6) AS sort
   FROM clientes a LEFT JOIN clientes r ON (a.cliente_ref = r.cliente_num)
  WHERE a.cliente_num = 101
UNION ALL -- SELECTs RECURSIVOS (Resto del Arbol)
 SELECT a.cliente_num, a.nombre, a.apellido, a.cliente_ref, c.nombre, c.apellido,
LEVEL + 1, c.sort+'/'+RIGHT('000000' + CAST(a.cliente_num AS varchar(20)), 6) AS sort
 FROM clientes a JOIN CTERecursivo C ON c.cliente_num = a.cliente_ref
) -- SELECT que muestra resultado del WITH
SELECT replicate('-', level * 4) + cast(clientE_num as varchar) as arbol, *
  FROM CTERecursivo ORDER BY sort;
```

SELECT – Recursivos

arbol	cliente...	nombre	apellido	cliente_ref	nombre	apellido	LEVEL	sort
101	101	Horacio	De Marco	NULL	NULL	NULL	0	000101
----102	102	Omar	Ramos	101	Horacio	De Marco	1	000101/000102
----103	103	Marta	Smith	101	Horacio	De Marco	1	000101/000103
--------104	104	Olga	Masa	103	Marta	Smith	2	000101/000103/000104
--------105	105	Edua...	Ansel	103	Marta	Smith	2	000101/000103/000105
--------106	106	Javier	Silva	103	Marta	Smith	2	000101/000103/000106
----------107	107	Ana	Bequi	106	Javier	Silva	3	000101/000103/000106/...
----------108	108	Carla	Vlck	107	Ana	Bequi	4	000101/000103/000106/...
----------109	109	Aldo	Lagos	107	Ana	Bequi	4	000101/000103/000106/...

Ejercicios