



GESTIÓN DE DATOS

Consultas con Joins.

ING. JAIME LLANOS BARDALES

Introducción

- Quiz



Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"



Poll Everywhere

Agenda

CONSULTAS MULTITABLA

- De tipo unión.
- De tipo inner join.
- De tipo left join.
- De tipo right join.
- De tipo full outer join.
- De tipo subconsulta.



Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"

Modelo Relacional



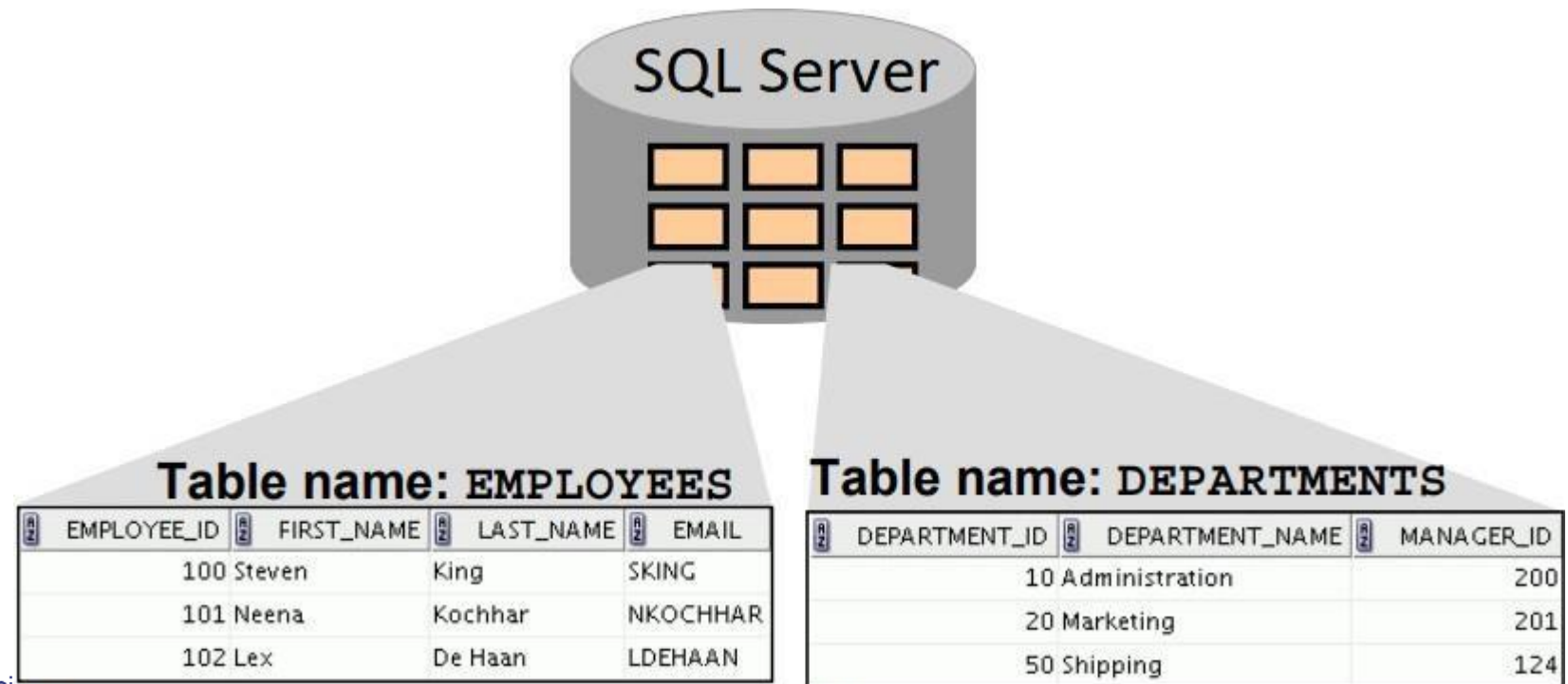
Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"

Una base de datos relacional es una colección de relaciones o tablas BI dimensionales.

Por ejemplo, es posible que desee almacenar información sobre todos los empleados de su empresa. En una base de datos relacional, puede crear varias tablas para almacenar diferentes datos sobre sus empleados, una tabla de departamentos, una tabla de salarios, y una tabla con los empleados

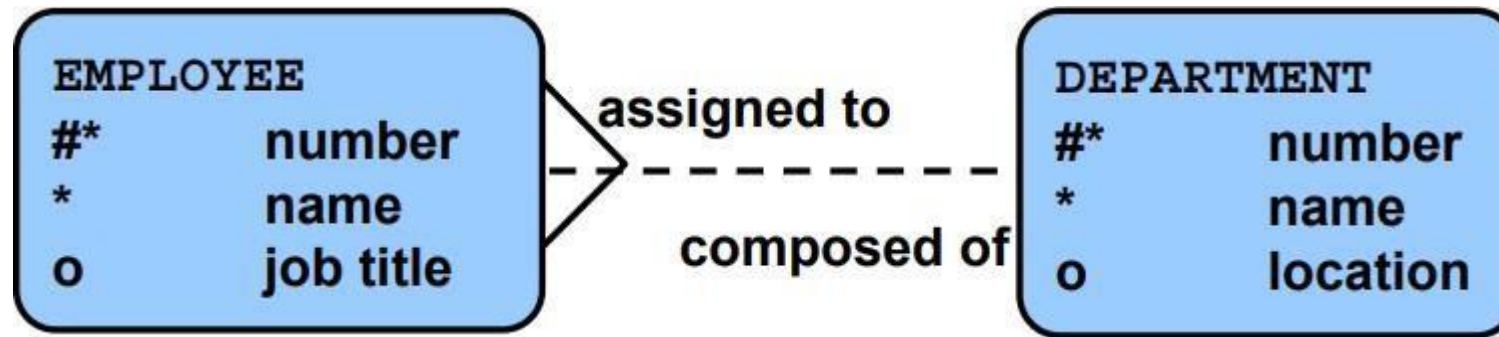
Propósito

- Comunicar
- Categorizar
- Describir
- Especificar
- Investigar
- Evolucionar
- Analizar



Modelo Entidad Relación

Las tablas (entidades) se relacionan entre si:



Escenario:

- *Asignar uno o mas empleados a un departamento*
- *Algunos departamentos no tiene asignados empleados*

Modelo Entidad Relación



Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"

Capacidades SQL

Proyección: Seleccionar las columnas. Unas o muchas

Selección: Obtener filas de una tabla que cumplen una condición.

Join: Utilizado para obtener data de diferentes tablas

Proyeccion

Tabla 1

Seleccion

Tabla 1

Tabla 1

Join

Tabla 2



Joins



Cuando se requiere obtener información de dos o mas tablas, relacionadas entre si:

- ID de los empleados que existen en la tabla de empleados
- ID de los departamentos que existen en la tabla de empleados y departamentos.
- Nombres de los departamentos que existen en la tabla departamentos

EMPLOYEES

	EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
1	200	Whalen	10
2	201	Hartstein	20
3	202	Fay	20
...			
18	174	Abel	80
19	176	Taylor	80
20	178	Grant	(null)

DEPARTMENTS

	DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
1	10	Administration	1700
2	20	Marketing	1800
3	50	Shipping	1500
4	60	IT	1400
5	80	Sales	2500
6	90	Executive	1700
7	110	Accounting	1700
8	190	Contracting	1700

↓ ↓

	EMPLOYEE_ID	DEPARTMENT_ID	DEPARTMENT_NAME
1	200	10	Administration
2	201	20	Marketing
3	202	20	Marketing
4	124	50	Shipping

SQL INNER JOIN

Table: Customers

customer_id	first_name
1	John
2	Robert
<u>3</u>	David
4	John
<u>5</u>	Betty

Table: Orders

order_id	amount	customer
1	200	10
2	500	<u>3</u>
3	300	6
4	800	<u>5</u>
5	150	8



customer_id	first_name	amount
3	David	500
5	Betty	800

Combinaciones de datos internas / unión izquierda



Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"

SQL LEFT JOIN

Table: Customers

customer_id	first_name
1	John
2	Robert
<u>3</u>	David
4	John
<u>5</u>	Betty

Table: Orders

order_id	amount	customer
1	200	10
2	500	<u>3</u>
3	300	6
4	800	<u>5</u>
5	150	8

customer_id	first_name	amount
1	John	
2	Robert	
3	David	500
4	John	
5	Betty	800

Combinaciones de datos / unión derecha



Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"

SQL RIGHT JOIN

Table: Customers

customer_id	first_name
1	John
2	Robert
<u>3</u>	David
4	John
<u>5</u>	Betty

Table: Orders

order_id	amount	customer
1	200	10
2	500	<u>3</u>
3	300	6
4	800	<u>5</u>
5	150	8

customer_id	first_name	amount
3	David	500
5	Betty	800
		200
		300
		150



Combinaciones de datos / unión exterior



SQL FULL OUTER JOIN

Table: Customers

customer_id	first_name
1	John
2	Robert
<u>3</u>	David
4	John
<u>5</u>	Betty

Table: Orders

order_id	amount	customer
1	200	10
2	500	<u>3</u>
3	300	6
4	800	<u>5</u>
5	150	8

customer_id	first_name	amount
3	David	200
5	Betty	500
		300
2	Robert	800
4	John	150

Joins



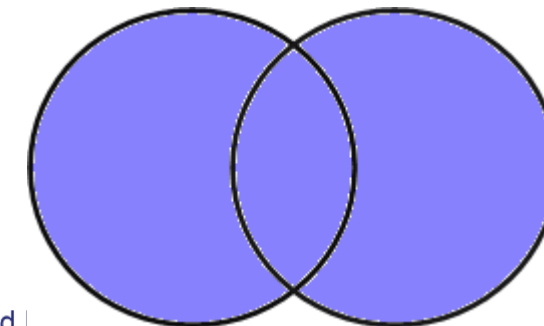
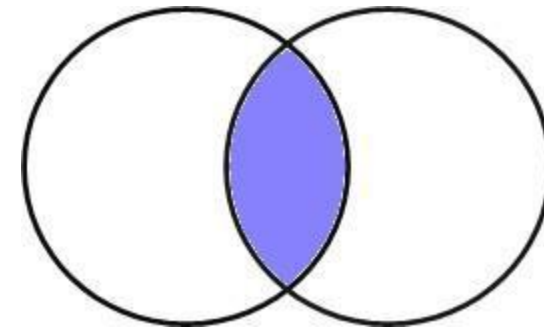
Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"

(INNER) JOIN: devuelve registros que tienen valores coincidentes en ambas tablas

FULL (OUTER) JOIN: devuelve todos los registros cuando hay una coincidencia en la tabla izquierda o derecha

```
SELECT column_name(s)
FROM table1
INNER JOIN table2 ON
table1.column_name=table2.column_name
```

```
SELECT column_name(s)
FROM table1
FULL JOIN table2 ON
table1.column_name=table2.column_name
```



Joins



(INNER) JOIN: devuelve registros que tienen valores coincidentes en ambas tablas

FULL (OUTER) JOIN: devuelve todos los registros cuando hay una coincidencia en la tabla izquierda o derecha

```
SELECT e.last_name, d.department_id, d.department_name  
FROM   employees e FULL OUTER JOIN departments d  
ON     (e.department_id = d.department_id) ;
```

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Whalen	10	Administration
2	Hartstein	20	Marketing
3	Fay	20	Marketing
4	Higgins	110	Accounting

...

17	Zlotkey	80	Sales
18	Abel	80	Sales
19	Taylor	80	Sales
20	Grant	(null)	(null)
21	(null)	190	Contracting

Joins



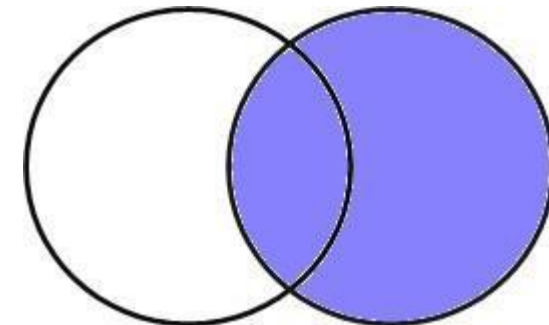
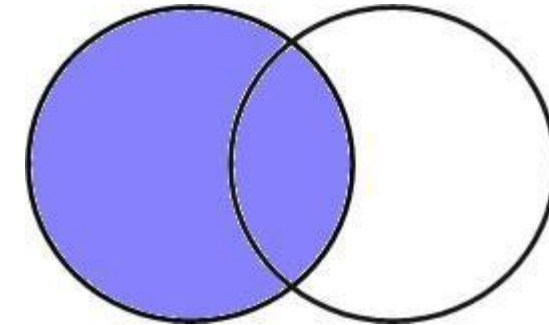
Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"

LEFT JOIN: devuelve todos los registros de la tabla izquierda y los registros coincidentes de la tabla derecha

RIGHT JOIN: devuelve todos los registros de la tabla derecha y los registros coincidentes de la tabla izquierda

```
SELECT column(s)
FROM table1
LEFT JOIN table2 ON table1.column=table2.column
```

```
SELECT column(s)
FROM table1
RIGHT JOIN table2 ON table1.column=table2.column
```



Joins



Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"

LEFT JOIN

```
SELECT e.last_name, e.department_id, d.department_name
FROM   employees e LEFT OUTER JOIN departments d
ON     (e.department_id = d.department_id) ;
```

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Whalen	10	Administration
2	Fay	20	Marketing
3	Hartstein	20	Marketing
4	Vargas	50	Shipping
5	Matos	50	Shipping

...

16	Kochhar	90	Executive
17	King	90	Executive
18	Gietz	110	Accounting
19	Higgins	110	Accounting
20	Grant	(null)	(null)

RIGHT JOIN

```
SELECT e.last_name, d.department_id, d.department_name
FROM   employees e RIGHT OUTER JOIN departments d
ON     (e.department_id = d.department_id) ;
```

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Whalen	10	Administration
2	Hartstein	20	Marketing
3	Fay	20	Marketing
4	Davies	50	Shipping
5	Vargas	50	Shipping
6	Rajs	50	Shipping
7	Mourgos	50	Shipping
8	Matos	50	Shipping

...

18	Higgins	110	Accounting
19	Gietz	110	Accounting
20	(null)	190	Contracting

Joins



```
SELECT Customers.CompanyName, Orders.OrderID
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
where Orders.OrderID is null
```

CompanyName	OrderID
FISSA Fabrica Inter. Salchichas S.A.	NULL
Paris spécialités	NULL

```
SELECT Customers.CompanyName, Orders.OrderID
FROM Customers
RIGHT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
where orders.OrderID is null
```

CompanyName	OrderID

Joins



INNER vs OUTER JOIN

- SQL 1999 el join entre dos tablas retorna solo las filas que hacen match (INNER)
- Un join entre dos tablas que retornan con filas que no hicieron match (LEFT o RIGHT) es llamado OUTER JOIN
- Un Join entre dos tablas que retorna un INNER JOIN y también los resultados LEFT y RIGHT JOIN se llama FULL OUTER JOIN

DEPARTMENTS

	DEPARTMENT_NAME	DEPARTMENT_ID
1	Administration	10
2	Marketing	20
3	Shipping	50
4	IT	60
5	Sales	80
6	Executive	90
7	Accounting	110
8	Contracting	190

There are no employees in department 190.

Employee "Grant" has not been assigned a department ID.

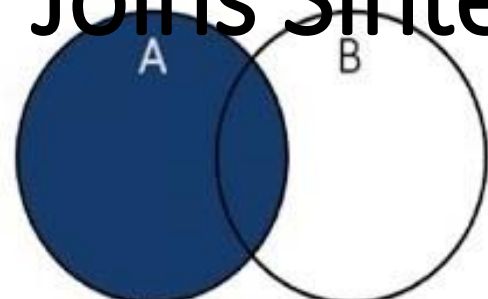
Equijoin with EMPLOYEES

	DEPARTMENT_ID	LAST_NAME
1	10	Whalen
2	20	Hartstein
3	20	Fay
4	110	Higgins
5	110	Gietz
6	90	King
7	90	Kochhar
8	90	De Haan
9	60	Hunold
10	60	Ernst

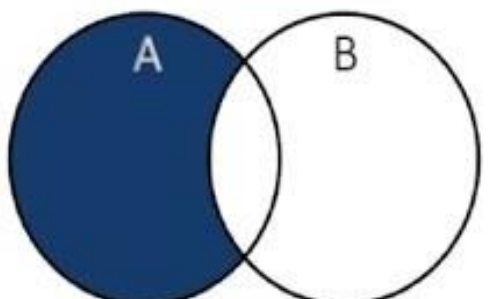
...

18	80	Abel
19	80	Taylor

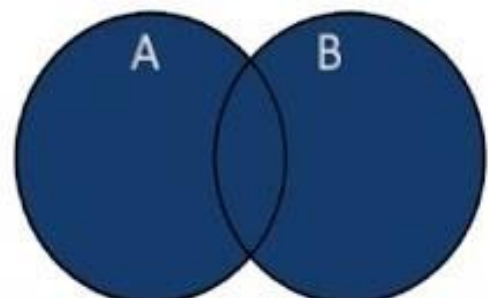
Joins Síntesis



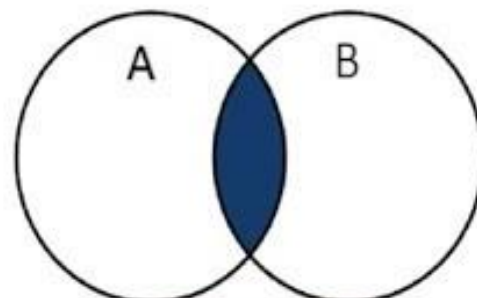
LEFT INCLUSIVE



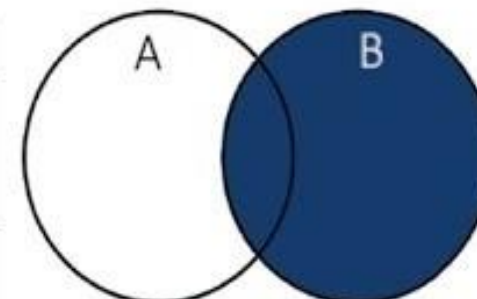
LEFT EXCLUSIVE



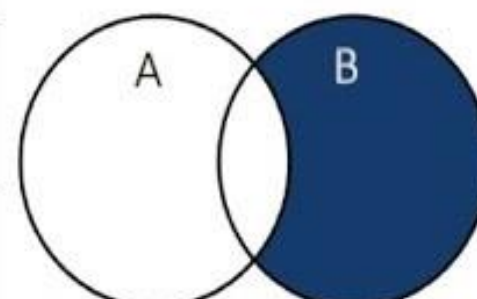
FULL OUTER INCLUSIVE



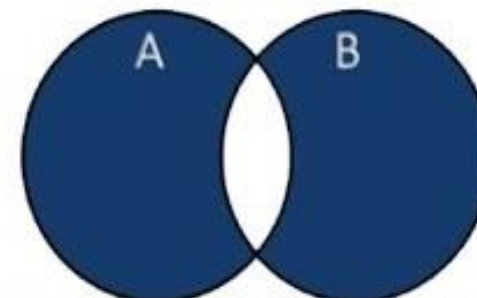
INNER JOIN



RIGHT INCLUSIVE



RIGHT EXCLUSIVE



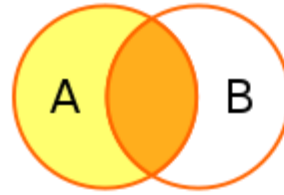
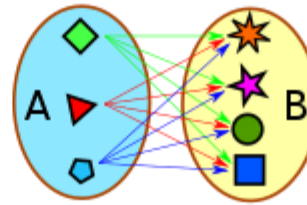
FULL OUTER EXCLUSIVE

SQL JOINS	
LEFT INCLUSIVE SELECT [Select List] FROM TableA A LEFT OUTER JOIN TableB B ON A.Key = B.Key	RIGHT INCLUSIVE SELECT [Select List] FROM TableA A RIGHT OUTER JOIN TableB B ON A.Key = B.Key
LEFT EXCLUSIVE SELECT [Select List] FROM TableA A LEFT OUTER JOIN TableB B ON A.Key = B.Key WHERE B.Key IS NULL	RIGHT EXCLUSIVE SELECT [Select List] FROM TableA A LEFT OUTER JOIN TableB B ON A.Key = B.Key WHERE A.Key IS NULL
FULL OUTER INCLUSIVE SELECT [Select List] FROM TableA A FULL OUTER JOIN TableB B ON A.Key = B.Key	FULL OUTER EXCLUSIVE SELECT [Select List] FROM TableA A FULL OUTER JOIN TableB B ON A.Key = B.Key WHERE A.Key IS NULL OR B.Key IS NULL
INNER JOIN SELECT [Select List] FROM TableA A INNER JOIN TableB B ON A.Key = B.Key	

Select (campos)
From A Inner Join B
On A.Clave = B.Clave



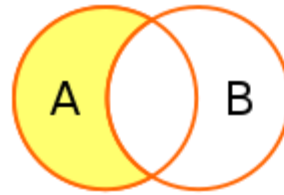
Select (campos)
From A Cross Join B



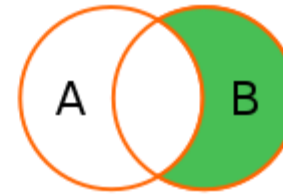
Select (campos)
From A Left Join B
On A.Clave = B.Clave



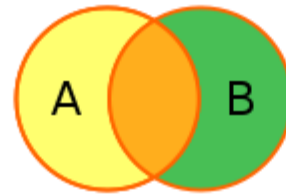
Select (campos)
From A Right Join B
On A.Clave = B.Clave



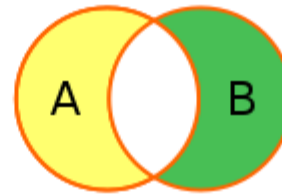
Select (campos)
From A Left Join B
On A.Clave = B.Clave
Where B.Clave is Null



Select (campos)
From A Right Join B
On A.Clave = B.Clave
Where A.Clave is Null



Select (campos)
From A Full Outer Join B
On A.Clave = B.Clave



Select (campos)
From A Full Outer Join B
On A.Clave = B.Clave
Where (A.Clave is Null) Or (B.Clave is Null)

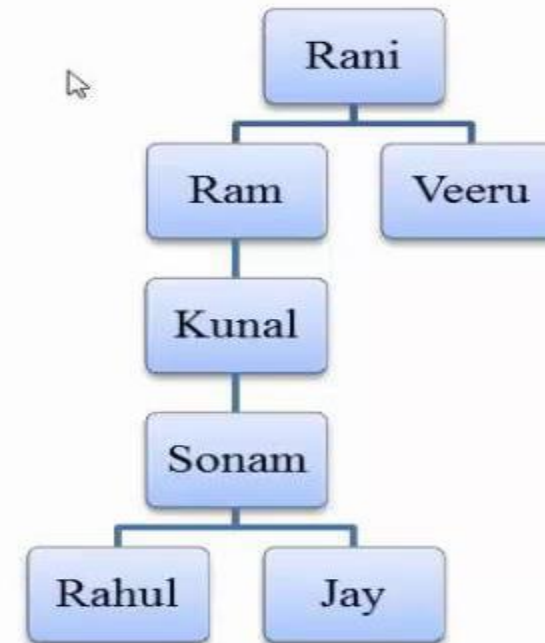
Joins del SQL

Self Join o tablas que se relacionan consigo misma



Self join is a table joined to itself.

Empid	Name	ManagerID
1	Rahul	3
2	Jay	3
3	Sonam	4
4	Kunal	5
5	Ram	6
6	Rani	NULL
7	Veeru	6



Ejemplos de uso de Self Join



Universidad
Nacional de
Cajamarca
"Norte de la Universidad Peruana"

Para el ejemplo tenemos la tabla empleados con una columna ID_EMPLEADO que almacena el código del empleado y ID_GERENTE que almacena el ID del gerente al que el empleado reporta, es decir su jefe. Para ver esto listamos y filtramos a los que reportan al empleado con ID_GERENTE = 114 o el empleado con ID_EMPLEADO = 114:

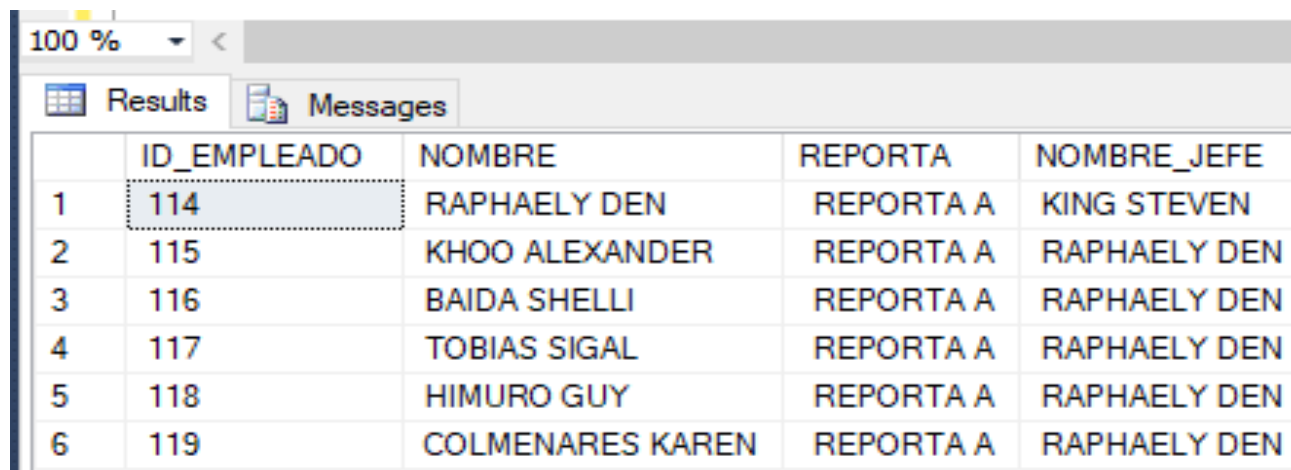
```
SELECT E.ID_EMPLEADO, CONCAT(E.APELLIDOS, ' ', E.NOMBRES) AS NOMBRE, E.ID_GERENTE  
FROM dbo.CL_EMPLEADOS E  
WHERE E.ID_GERENTE = 114 OR E.ID_EMPLEADO = 114;
```

	ID_EMPLEADO	NOMBRE	ID_GERENTE
1	114	RAPHAELY DEN	100
2	115	KHOO ALEXANDER	114
3	116	BAIDA SHELLI	114
4	117	TOBIAS SIGAL	114
5	118	HIMURO GUY	114
6	119	COLMENARES KAREN	114

Ejemplos de uso de Self Join

Hagamos un query que permita listar ID_EMPLEADO, LOS NOMBRES DEL EMPLEADO, la palabra "Reporta a", y el NOMBRE de su Jefe aplicando SELF JOIN y para relaciona inner join

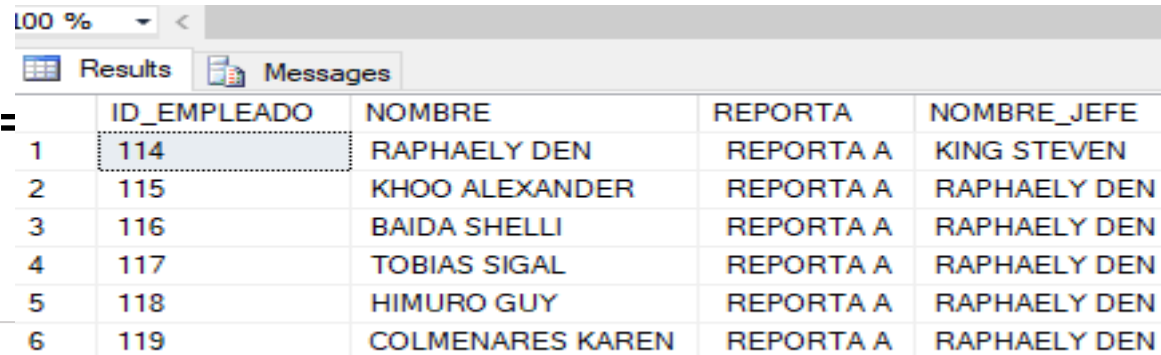
```
SELECT G.ID_EMPLEADO, CONCAT(G.APELLIDOS, ' ', G.NOMBRES) AS NOMBRE, ' REPORTA A' AS REPORTA,  
CONCAT(E.APELLIDOS, ' ', E.NOMBRES) AS NOMBRE_JEFE FROM dbo.CL_EMPLEADOS E  
INNER JOIN dbo.CL_EMPLEADOS G ON E.ID_EMPLEADO = G.ID_GERENTE  
WHERE G.ID_GERENTE = 114 OR G.ID_EMPLEADO = 114;
```



	ID_EMPLEADO	NOMBRE	REPORTA	NOMBRE_JEFE
1	114	RAPHAELY DEN	REPORTA A	KING STEVEN
2	115	KHOO ALEXANDER	REPORTA A	RAPHAELY DEN
3	116	BAIDA SHELLI	REPORTA A	RAPHAELY DEN
4	117	TOBIAS SIGAL	REPORTA A	RAPHAELY DEN
5	118	HIMURO GUY	REPORTA A	RAPHAELY DEN
6	119	COLMENARES KAREN	REPORTA A	RAPHAELY DEN

Ejemplos de uso de Self Join

- Hagamos un query que permita listar ID_EMPLEADO, LOS NOMBRES DEL EMPLEADO, la palabra "Reporta a", y el NOMBRE de su Jefe aplicando SELF JOIN y para relacionar usemos
- **SELECT G.ID_EMPLEADO, CONCAT(G.APELLIDOS, ' ', G.NOMBRES) AS NOMBRE,**
- **' REPORTA A' AS REPORTA, CONCAT(E.APELLIDOS, ' ', E.NOMBRES) AS NOMBRE_JEFE**
- **FROM dbo.CL_EMPLEADOS E, dbo.CL_EMPLEADOS G WHERE (E.ID_EMPLEADO =**
- **G.ID_GERENTE**
- **AND G.ID_GERENTE = G.ID_EMPLEADO = 114);**



	ID_EMPLEADO	NOMBRE	REPORTA	NOMBRE_JEFE
1	114	RAPHAELY DEN	REPORTA A	KING STEVEN
2	115	KHOO ALEXANDER	REPORTA A	RAPHAELY DEN
3	116	BAIDA SHELLI	REPORTA A	RAPHAELY DEN
4	117	TOBIAS SIGAL	REPORTA A	RAPHAELY DEN
5	118	HIMURO GUY	REPORTA A	RAPHAELY DEN
6	119	COLMENARES KAREN	REPORTA A	RAPHAELY DEN

GERENTE AND

De Tipo Sub consulta

Quienes tienen el salario mas alto que Joel?

Query principal

Cuales son los empleados que tienen el salario mayor al de Joel?

SubQuery

Cual es el salario de Joel?

```
SELECT last_name, salary
FROM employees
WHERE salary > 11000
      (SELECT salary
       FROM employees
       WHERE last_name = 'Abel');
```

	LAST_NAME	SALARY
1	Hartstein	13000
2	Higgins	12000
3	King	24000
4	Kochhar	17000
5	De Haan	17000

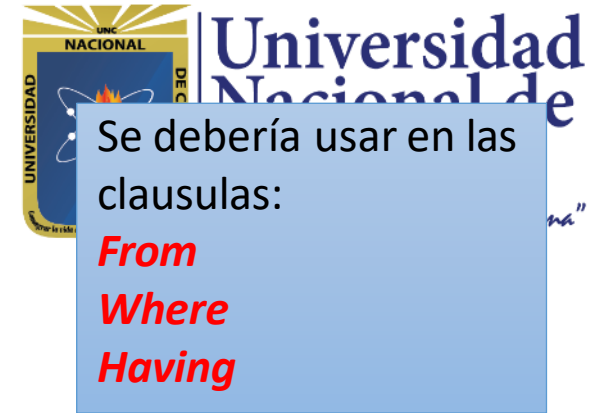
Para el uso de SubQueries:

- Incluir subconsultas entre paréntesis
- Coloque las subconsultas en el lado derecho de la condición de comparación para facilitar la lectura. (Sin embargo, las subconsultas pueden aparecer a ambos lados del operador de comparación)
- Utilice operadores de una sola fila con subconsultas de una sola fila y operadores de varias filas con subconsultas de varias filas.

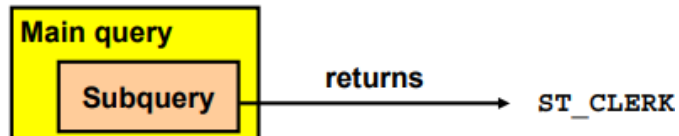
De Tipo Sub consulta

Un **SubQuery** es una sentencia SELECT anidada en una clausula de otra sentencia SELECT

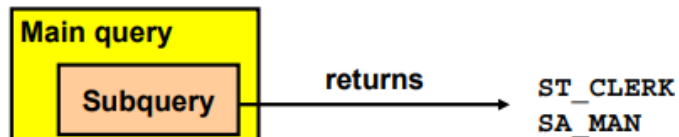
```
SELECT select_list
FROM table
WHERE expr operator ( SELECT select_list
                        FROM table )
```



Single-row subquery



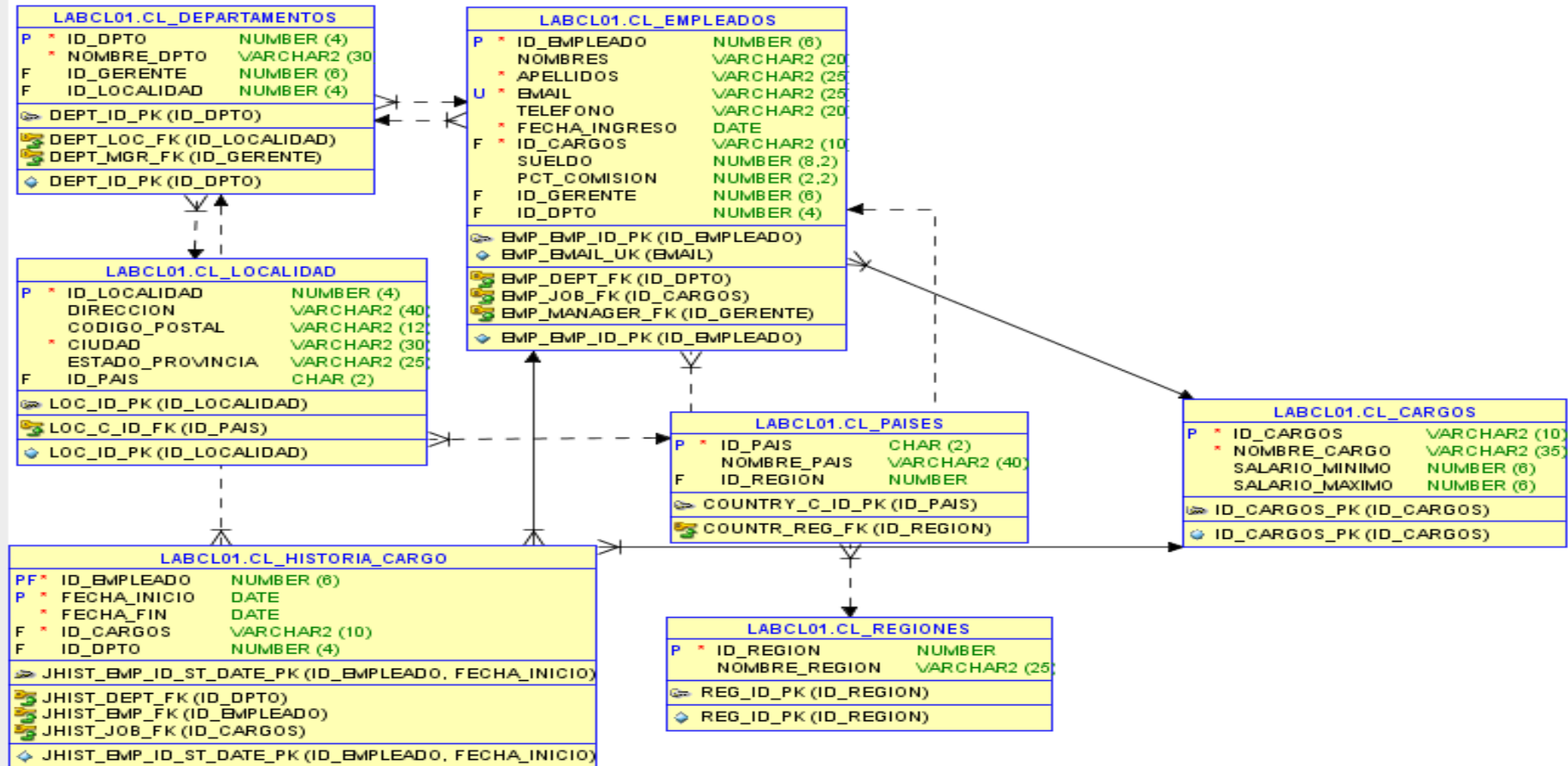
Multiple-row subquery



Single Row	Multiple Row
Retorna solo una fila	Retorna mas de una fila
Usa el operador de comparación de una sola fila = > < >= <= <>	Usa los operadores de comparación para múltiples filas IN, ANY, ALL

La subconsulta se ejecuta **antes** de la consulta principal. El resultado de la subconsulta es utilizado por la consulta principal

Modelo Relacional RRHH



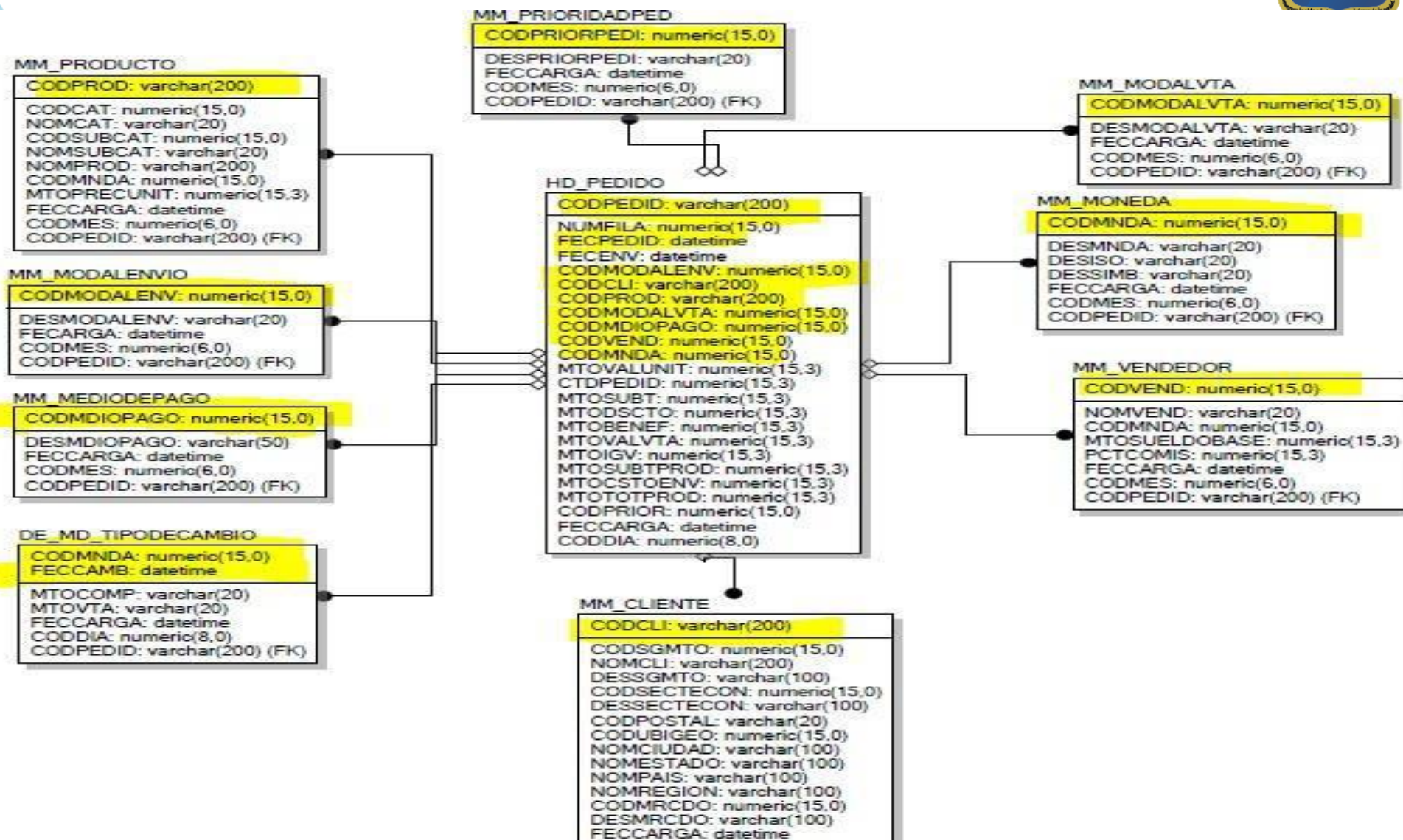
- Para poder desarrollar el laboratorio tener presente el modelo entidad, específicamente como se relacionan las tablas del modelo y las columnas que poseen las mismas.
- En este modelo de RRHH podríamos decir que la tabla principal es la tabla **CL_EMPLEADOS** y esta se relaciona con **CL_DEPARTAMENTOS** por la columna **ID_DPTO**.
- La tabla **CL_DEPARTAMENTOS** se relaciona a su vez con la tabla **CL_LOCALIDAD** por la columna **ID_LOCALIDAD**.
- La tabla **CL_LOCALIDAD** a su vez se relaciona con la tabla **CL_PAISES** por la columna **ID_PAIS**.
- La tabla **CL_PAIS** se relaciona con la tabla **CL_REGIONES** por la columna **ID_REGION**.
- La tabla **CL_EMPLEADOS** se relaciona con la tabla **CL_CARGOS** a través de la columna **ID_CARGOS**.

Modelo Relacional Ordenes y pedidos



Universidad
Nacional de
Cajamarca

"Norte de la Universidad Peruana"



Modelo Relacional Ordenes y pedidos

En este modelo observar cual es la tabla principal y como se relaciona con el resto de las tablas (modelo estrella)

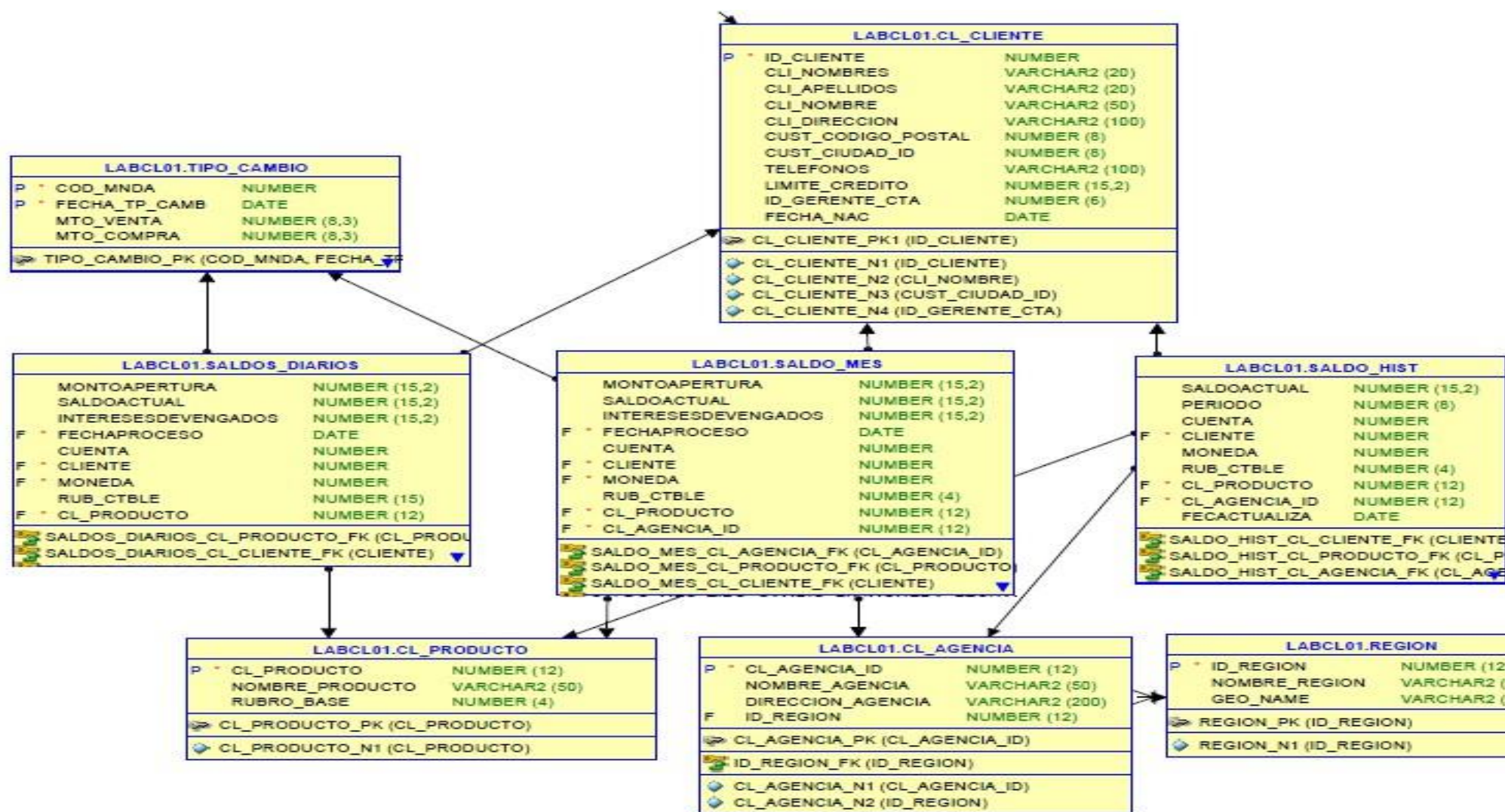
- Por ejemplo La tabla **HD_PEDIDO** contiene el detalle de los pedidos de un sistema de ventas en línea.
- A su vez la tabla **HD_PEDIDO** se relaciona con la tabla **MM_CLIENTE** por la Columna **CODCLIENTE**.
- A su vez la tabla **HD_PEDIDO** se relaciona con la tabla **MM_PRODUCTO** por la Columna **CODPROD**.
- A su vez la tabla **HD_PEDIDO** se relaciona con la tabla **MM_VENDEDOR** por la Columna **CODVEND**.
- A su vez la tabla **HD_PEDIDO** se relaciona con la tabla **MM_MODALENVIO** por la Columna **CODMODALENV**.
- A su vez la tabla **HD_PEDIDO** se relaciona con la tabla **MM_MEDIOPAGO** por la Columna **CODMEDIOPAGO**.
- A su vez la tabla **HD_PEDIDO** se relaciona con la tabla **MM_PRIORIDADPED** por la Columna **CODPRIORPED**.
- A su vez la tabla **HD_PEDIDO** se relaciona con la tabla **MM_MODALENVIO** por la Columna **CODMODALVTA**.

Modelo Relacional Saldos Bancarios



**Universidad
Nacional de
Cajamarca**

"Norte de la Universidad Peruana"



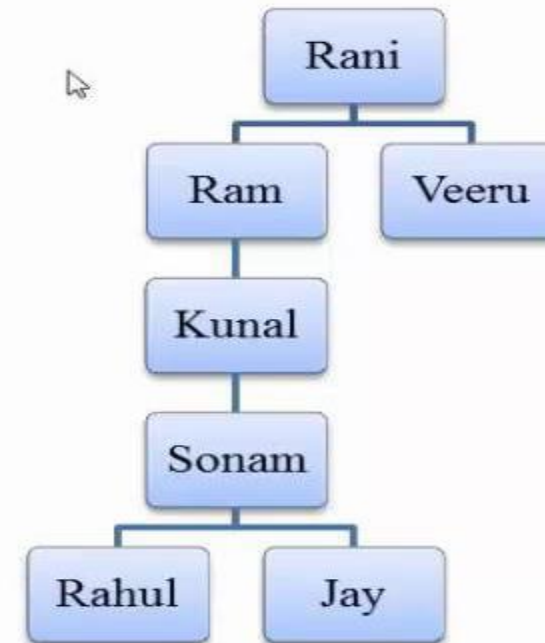
- A continuación, mostramos el modelo entidad relación del modelo Banca. Donde observamos 3 tablas principales o tablas de hechos que se relacionan con la otras tablas o dimensiones.
- Para este caso **SALDOS_DIARIOS**, **SALDO_MES** Y **SALDO_HIST** que contienen información de 3 meses de historia de movimientos. Cabe señalar que, a diferencia de los modelos previos, aquí se muestra una relación de 2 columnas entre las tablas mencionadas arriba con la tabla **TIPO_CAMBIO** ya que los saldos están en soles y dólares y para obtener el tipo de cambio de una fecha determinada hay que relacionar en el caso de **SALDOS_DIARIOS** las columnas **MONEDA y FECHAPROCESO** como una llave compuesta, es decir de más de una columna.

Self Join o tablas que se relacionan consigo misma



Self join is a table joined to itself.

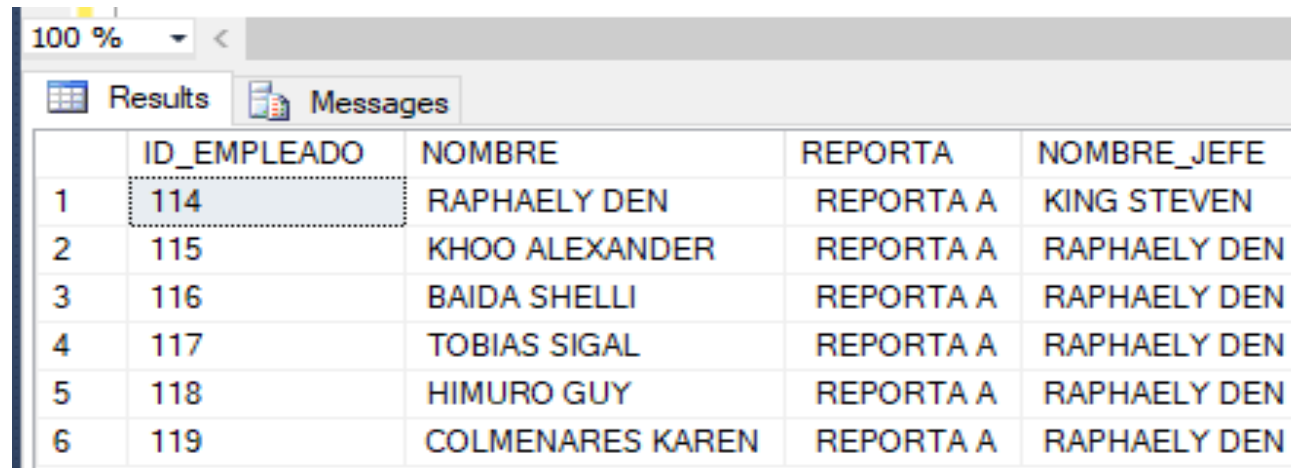
Empid	Name	ManagerID
1	Rahul	3
2	Jay	3
3	Sonam	4
4	Kunal	5
5	Ram	6
6	Rani	NULL
7	Veeru	6



Ejemplos de uso de Self Join

Hagamos un query que permita listar ID_EMPLEADO, LOS NOMBRES DEL EMPLEADO, la palabra "Reporta a", y el NOMBRE de su Jefe aplicando SELF JOIN y para relaciona inner join

```
SELECT G.ID_EMPLEADO, CONCAT(G.APELLIDOS, ' ', G.NOMBRES) AS NOMBRE, ' REPORTA A' AS REPORTA,  
CONCAT(E.APELLIDOS, ' ', E.NOMBRES) AS NOMBRE_JEFE FROM dbo.CL_EMPLEADOS E  
INNER JOIN dbo.CL_EMPLEADOS G ON E.ID_EMPLEADO = G.ID_GERENTE  
WHERE G.ID_GERENTE = 114 OR G.ID_EMPLEADO = 114;
```



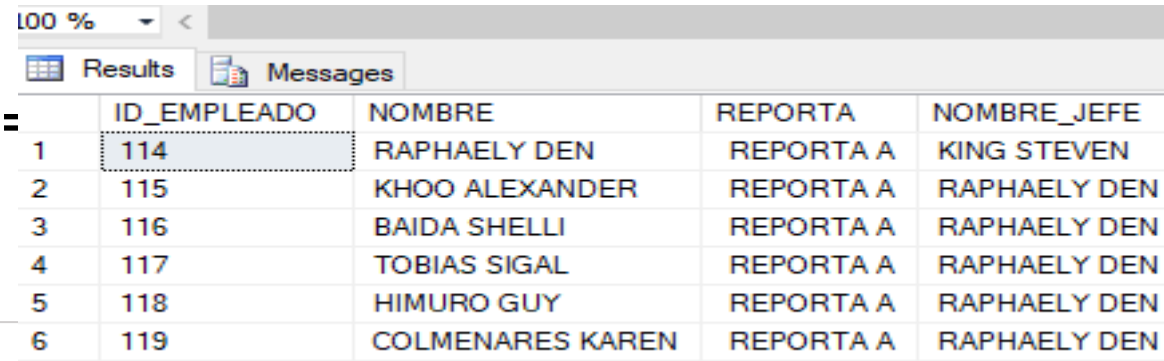
100 %

Results Messages

	ID_EMPLEADO	NOMBRE	REPORTA	NOMBRE_JEFE
1	114	RAPHAELY DEN	REPORTA A	KING STEVEN
2	115	KHOO ALEXANDER	REPORTA A	RAPHAELY DEN
3	116	BAIDA SHELLI	REPORTA A	RAPHAELY DEN
4	117	TOBIAS SIGAL	REPORTA A	RAPHAELY DEN
5	118	HIMURO GUY	REPORTA A	RAPHAELY DEN
6	119	COLMENARES KAREN	REPORTA A	RAPHAELY DEN

Ejemplos de uso de Self Join

- Hagamos un query que permita listar ID_EMPLEADO, LOS NOMBRES DEL EMPLEADO, la palabra "Reporta a", y el NOMBRE de su Jefe aplicando SELF JOIN y para relacionar usemos
- **SELECT G.ID_EMPLEADO, CONCAT(G.APELLIDOS, ' ', G.NOMBRES) AS NOMBRE,**
- **' REPORTA A' AS REPORTA, CONCAT(E.APELLIDOS, ' ', E.NOMBRES) AS NOMBRE_JEFE**
- **FROM dbo.CL_EMPLEADOS E, dbo.CL_EMPLEADOS G WHERE (E.ID_EMPLEADO =**
- **G.ID_GERENTE**
- **AND G.ID_GERENTE = G.ID_EMPLEADO = 114);**



100 %

Results Messages

	ID_EMPLEADO	NOMBRE	REPORTA	NOMBRE_JEFE
1	114	RAPHAELY DEN	REPORTA A	KING STEVEN
2	115	KHOO ALEXANDER	REPORTA A	RAPHAELY DEN
3	116	BAIDA SHELLI	REPORTA A	RAPHAELY DEN
4	117	TOBIAS SIGAL	REPORTA A	RAPHAELY DEN
5	118	HIMURO GUY	REPORTA A	RAPHAELY DEN
6	119	COLMENARES KAREN	REPORTA A	RAPHAELY DEN

GERENTE AND



Técnicas de Agregación (Incorrecta)

```
SELECT  
SUM(SALARY) AS SUELDO,  
DEP.DEPARTMENT_NAME  
FROM EMPLOYEES EMP  
LEFT JOIN DEPARTMENTS DEP ON EMP.DEPARTMENT_ID = DEP.DEPARTMENT_ID  
WHERE EMP.DEPARTMENT_ID NOT IN (90,60,30)  
GROUP BY DEP.DEPARTMENT_NAME;
```

Output pane

Data Output

Explain

Messages

History

	sueldo numeric	department_name character varying(30)
1	10000.00	PUBLIC RELATIONS
2	20300.00	ACCOUNTING
3	6500.00	HUMAN RESOURCES
4	51600.00	FINANCE
5	4400.00	ADMINISTRATION
6	304500.00	SALES
7	19000.00	MARKETING



Técnicas de Agregación (correcta)

```
SELECT
EMP.SUELDO,
EMP.DEPARTMENT_ID,
DEP.DEPARTMENT_NAME
FROM
  (SELECT
    SUM(EMP.SALARY) AS SUELDO,
    EMP.DEPARTMENT_ID
    FROM EMPLOYEES EMP
    GROUP BY EMP.DEPARTMENT_ID) EMP
LEFT JOIN DEPARTMENTS DEP ON EMP.DEPARTMENT_ID = DEP.DEPARTMENT_ID
```

Output pane

Data Output Explain Messages History

	sueldo numeric	department_id numeric(4,0)	department_name character varying(30)
1	4400.00	10	ADMINISTRATION
2	19000.00	20	MARKETING
3	24900.00	30	PURCHASING
4	6500.00	40	HUMAN RESOURCES
5	156400.00	50	SHIPPING
6	28800.00	60	IT
7	10000.00	70	PUBLIC RELATIONS



Subquerys expresions (Not in)

El lado derecho es una subconsulta entre paréntesis, que debe devolver exactamente una columna. La expresión de la mano izquierda se evalúa y se compara con cada fila del resultado de la subconsulta. El resultado de NOT IN es "verdadero" si solo se encuentran filas de subconsulta desiguales (incluido el caso especial donde la subconsulta no devuelve filas). El resultado es "falso" si se encuentra una fila igual.

Tenga en cuenta que si la expresión de la mano izquierda produce un valor nulo, o si no hay valores iguales de la mano derecha y al menos una línea derecha produce un valor nulo, el resultado de la construcción NOT IN será nulo, no verdadero. Esto está de acuerdo con las reglas normales de SQL para combinaciones booleanas de valores nulos.

Al igual que con EXISTS, es imprudente suponer que la subconsulta se evaluará



Uso de Consultas anidadas con With

Una cláusula WITH es una cláusula opcional que precede a la lista SELECT en una consulta. La cláusula WITH define una o más expresiones de tabla. Cada expresión de tabla común (CTE) define una tabla temporal, que es similar a la definición de una vista. Puede hacer referencia a estas tablas temporales en la cláusula FROM. Se usan solo mientras se ejecuta la consulta a la que pertenecen. Cada CTE de la cláusula WITH especifica un nombre de tabla, una lista opcional de nombres de columnas y una expresión de consulta que se evalúa como una tabla (una instrucción SELECT). Cuando hace referencia al nombre de la tabla temporal en la cláusula FROM de la misma expresión de consulta que lo define, el CTE es recursivo.

Las subconsultas de la cláusula WITH son una manera eficiente de definir tablas que pueden utilizarse durante la ejecución de una consulta. En todos los casos, se pueden obtener los mismos resultados al utilizar subconsultas en el cuerpo principal de la instrucción SELECT, pero las subconsultas de la cláusula WITH pueden ser más simples de escribir y leer. Donde sea posible, las subconsultas de la cláusula WITH a las que se hace referencia varias veces se optimizan como subexpresiones comunes; es decir, puede ser posible evaluar una subconsulta WITH una vez y reutilizar sus resultados. (Tenga en cuenta que las subexpresiones más frecuentes no se limitan a aquellas que se definen en la cláusula WITH).



Uso de Consultas anidadas con With

Ejemplos

```
SELECT TS.PERIODO, PR.NOMPROD, CL.NOMCLI, TS.VENTAS
FROM
(SELECT
FORMAT(P.FECPEDID, 'yyyyMM') as PERIODO,
P.CODPROD,
P.CODCLI,
SUM(P.MTOVALVTA) AS VENTAS
FROM DBO.HD_PEDIDO P
WHERE P.FECPEDID >= '20140101'
AND P.FECPEDID <= '20140131'
GROUP BY FORMAT(P.FECPEDID, 'yyyyMM'), P.CODPROD, P.CODCLI) TS
INNER JOIN DBO.MM_PRODUCTO PR
ON TS.CODPROD = PR.CODPROD
INNER JOIN DBO.MM_CLIENTE CL ON TS.CODCLI = CL.CODCLI;
```

	PERIODO	NOMPROD	NOMCLI	VENTAS
1	201401	Cardinal Binding Machine, Clear	APARICIO TORRES, CONSUELO	866.400
2	201401	Elite Scissors, Serrated	APARICIO TORRES, CONSUELO	312.240
3	201401	Panasonic Printer, Durable	FERNANDEZ BACA CALDERON VDA DE VALDEZ, HILDA GRA...	2567.430
4	201401	Advantus Photo Frame, Black	BECERRA MENESES, JAIME	-47.800
5	201401	Tenex Photo Frame, Duo Pack	MIDEROS MALDONADO, JOSE LUIS	871.800
6	201401	Xerox 1994	PEÑA CAPITAN, JOSE SALVADOR	42.768
7	201401	Binney & Smith Canvas, Blue	BLAS MURGA, MARIA ISABEL	3365.040
8	201401	Advantus Door Stop, Ergonomic	PIAGGIO SIMPSON, GLORIA BEATRIZ	44.521
9	201401	Fellowes File Cart, Blue	OLANO FLORES, HUMBERTO	2986.372
10	201401	Cisco Headset, Cordless	NEIRA PEREA, JUAN CARLOS	6017.760
11	201401	Epson Calculator, Wireless	NEIRA PEREA, JUAN CARLOS	231.120
12	201401	Rogers Shelving, Single Width	BASTARRACHEA GARCIA, GUSTAVO ALFONSO	449.348
13	201401	Lesro Round Table, Fully Assembled	BASTARRACHEA GARCIA, GUSTAVO ALFONSO	1832.059
14	201401	Electrix Fluorescent Magnifier Lamps & Weighted ...	CORTEZ CASTILLO, ALBERTO FAUSTO	2052.544
15	201401	Xerox 1882	CORTEZ CASTILLO, ALBERTO FAUSTO	175.777
16	201401	Green Bar Computer Printout Paper, Recycled	LUNA TIRADO, COSME	135.240
17	201401	SanDisk Cards & Envelopes, Recycled	LUNA TIRADO, COSME	447.930
18	201401	Eldon Shelving, Blue	LUNA TIRADO, COSME	865.680
19	201401	Rogers File Cart, Single Width	LUNA TIRADO, COSME	2318.640
20	201401	Advantus Staples, Metal	LUNA TIRADO, COSME	46.500

Uso de Consultas anidadas con With

Ejemplos

```
WITH TWTS AS
(
  SELECT
    FORMAT(P.FECPEDID, 'yyyyMM') AS PERIODO,
    P.CODPROD,
    P.CODCLI,
    SUM(P.MTOVALVTA) AS VENTAS
  FROM DBO.HD_PEDIDO P
  WHERE P.FECPEDID >= '20140101'
  AND P.FECPEDID <= '20140131'
  GROUP BY FORMAT(P.FECPEDID, 'yyyyMM'), P.CODPROD,
    P.CODCLI )

SELECT TS.PERIODO, PR.NOMPROD, CL.NOMCLI,
TS.VENTAS
FROM TWTS TS
INNER JOIN dbo.MM_PRODUCTO PR
ON TS.CODPROD = PR.CODPROD
INNER JOIN DBO.MM_CLIENTE CL ON TS.CODCLI =
CL.CODCLI;
```

	PERIODO	NOMPROD	NOMCLI	VENTAS
1	201401	Cardinal Binding Machine, Clear	APARICIO TORRES, CONSUELO	866.400
2	201401	Elite Scissors, Serrated	APARICIO TORRES, CONSUELO	312.240
3	201401	Panasonic Printer, Durable	FERNANDEZ BACA CALDERON VDA DE VALDEZ, HILDA GRA...	2567.430
4	201401	Advantus Photo Frame, Black	BECERRA MENESES, JAIME	-47.800
5	201401	Tenex Photo Frame, Duo Pack	MIDEROS MALDONADO, JOSE LUIS	871.800
6	201401	Xerox 1994	PEÑA CAPITAN, JOSE SALVADOR	42.768
7	201401	Binney & Smith Canvas, Blue	BLAS MURGA, MARIA ISABEL	3365.040
8	201401	Advantus Door Stop, Ergonomic	PIAGGIO SIMPSON, GLORIA BEATRIZ	44.521
9	201401	Fellowes File Cart, Blue	OLANO FLORES, HUMBERTO	2986.372
10	201401	Cisco Headset, Cordless	NEIRA PEREA, JUAN CARLOS	6017.760
11	201401	Epson Calculator, Wireless	NEIRA PEREA, JUAN CARLOS	231.120
12	201401	Rogers Shelving, Single Width	BASTARRACHEA GARCIA, GUSTAVO ALFONSO	449.348
13	201401	Lesro Round Table, Fully Assembled	BASTARRACHEA GARCIA, GUSTAVO ALFONSO	1832.059
14	201401	Electrix Fluorescent Magnifier Lamps & Weighted ...	CORTEZ CASTILLO, ALBERTO FAUSTO	2052.544
15	201401	Xerox 1882	CORTEZ CASTILLO, ALBERTO FAUSTO	175.777
16	201401	Green Bar Computer Printout Paper, Recycled	LUNA TIRADO, COSME	135.240
17	201401	SanDisk Cards & Envelopes, Recycled	LUNA TIRADO, COSME	447.930
18	201401	Eldon Shelving, Blue	LUNA TIRADO, COSME	865.680
19	201401	Rogers File Cart, Single Width	LUNA TIRADO, COSME	2318.640
20	201401	Advantus Staples, Metal	LUNA TIRADO, COSME	46.500

REFERENCIAS

Joins

<https://docs.microsoft.com/es-es/sql/relational-databases/performance/joins?view=sql-server-ver15>

Selects

<https://docs.microsoft.com/es-es/sql/t-sql/queries/select-transact-sql?view=sql-server-ver15>

Sub Consultas

<https://docs.microsoft.com/es-es/sql/relational-databases/performance/subqueries?view=sql-server-ver15>



Herramientas de Trabajo

Online



**SQL Server
Management Studio**

v. 19.1



Fin de la sesión

