UNIVERSIDAD PRIVADA DE TACNA

FACULTAD DE INGENIERIA

Escuela Profesional de Ingeniería de Sistemas

PRACTICA DE LABORATORIO: UTILIZANDO EXPRESIONES DE TABLA

Curso: Base de Datos II

Docente: Mag. Ing. Patrick Cuadros Quiroga

Integrantes:

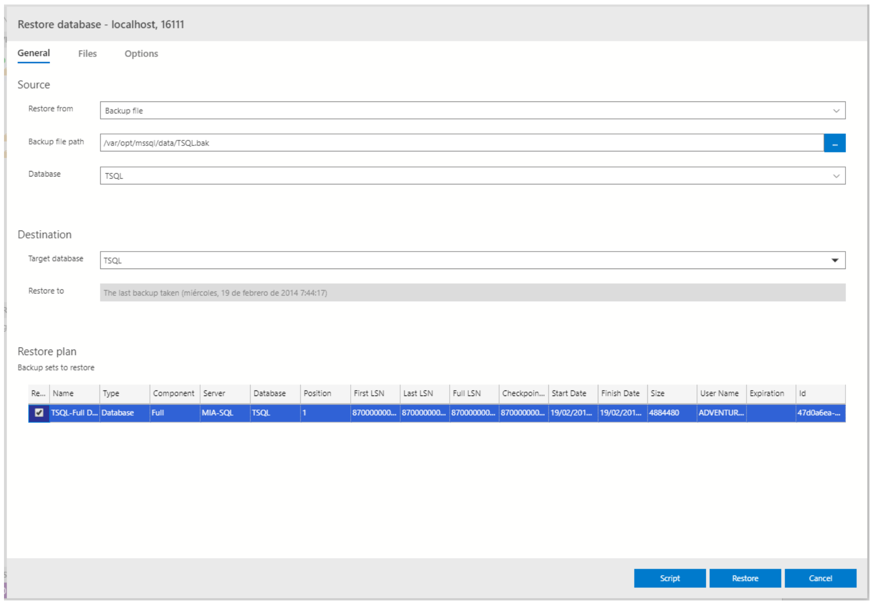
Puma Villa, Brayan **(2015052787)**

Tacna – Perú
2022

UTILIZANDO EXPRESIONES DE TABLA

RESTAURAR BASE DE DATOS

1. Una vez desplegado e iniciado nuestro contenedor, vamos a restaurar la base de datos TSQL.



Source

Restore from: Backup file

Backup file path: /var/opt/mssql/data/TSQL.bak

Database: TSQL

Destination

Target database: TSQL

Restore to: The last backup taken (miércoles, 19 de febrero de 2014 7:44:17)

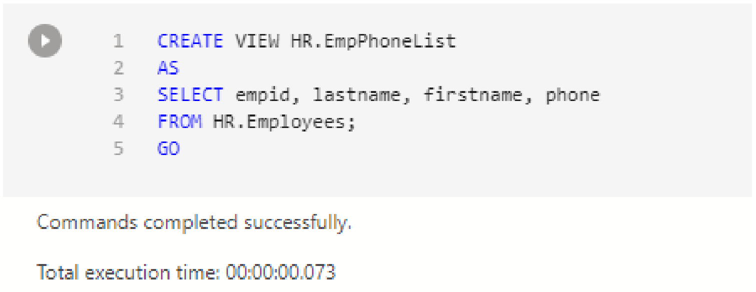
Restore plan

Backup sets to restore

Re...	Name	Type	Component	Server	Database	Position	First LSN	Last LSN	Full LSN	Checkpoin...	Start Date	Finish Date	Size	User Name	Expiration	Id
✓	TSQL-Full D...	Database	Full	MIA-SQL	TSQL	1	870000000...	870000000...	870000000...	870000000...	19/02/201...	19/02/201...	4884480	ADVENTUR...		47d0a6ea-...

PARTE 1: UTILIZANDO VISTAS

1. Seleccionamos y ejecutamos el código para crear una vista.

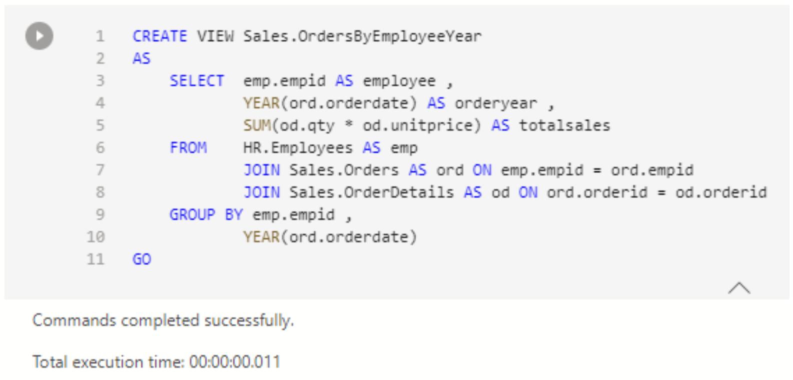


```
CREATE VIEW HR.EmpPhoneList
AS
SELECT empid, lastname, firstname, phone
FROM HR.Employees;
GO
```

```
Commands completed successfully.

Total execution time: 00:00:00.073
```

2. Creamos una vista utilizando un JOIN multi-tabla.

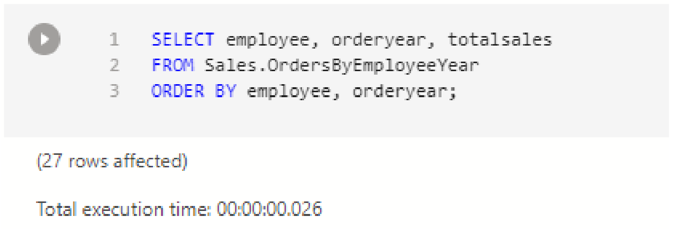


```
CREATE VIEW Sales.OrdersByEmployeeYear
AS
    SELECT  emp.empid AS employee ,
            YEAR(ord.orderdate) AS orderyear ,
            SUM(od.qty * od.unitprice) AS totalsales
    FROM    HR.Employees AS emp
            JOIN Sales.Orders AS ord ON emp.empid = ord.empid
            JOIN Sales.OrderDetails AS od ON ord.orderid = od.orderid
    GROUP BY emp.empid ,
            YEAR(ord.orderdate)
GO
```

```
Commands completed successfully.

Total execution time: 00:00:00.011
```

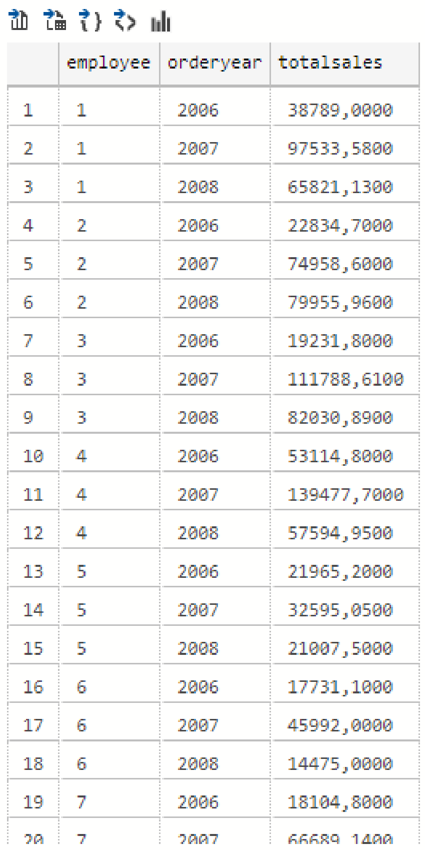
3. Seleccionamos la vista.



```
SELECT employee, orderyear, totalsales
FROM Sales.OrdersByEmployeeYear
ORDER BY employee, orderyear;
```

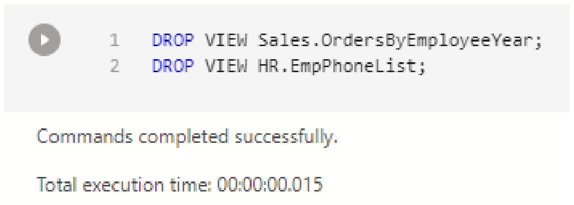
```
(27 rows affected)

Total execution time: 00:00:00.026
```



	employee	orderyear	totalsales
1	1	2006	38789,0000
2	1	2007	97533,5800
3	1	2008	65821,1300
4	2	2006	22834,7000
5	2	2007	74958,6000
6	2	2008	79955,9600
7	3	2006	19231,8000
8	3	2007	111788,6100
9	3	2008	82030,8900
10	4	2006	53114,8000
11	4	2007	139477,7000
12	4	2008	57594,9500
13	5	2006	21965,2000
14	5	2007	32595,0500
15	5	2008	21007,5000
16	6	2006	17731,1000
17	6	2007	45992,0000
18	6	2008	14475,0000
19	7	2006	18104,8000
20	7	2007	66689,1400

4. Limpiamos los cambios.



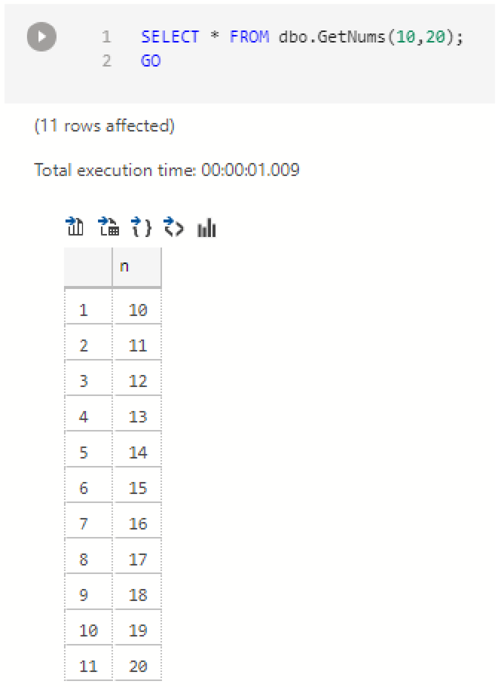
```
DROP VIEW Sales.OrdersByEmployeeYear;
DROP VIEW HR.EmpPhoneList;
```

```
Commands completed successfully.

Total execution time: 00:00:00.015
```

PARTE 2: UTILIZANDO FUNCIONES DE TABLA EN LINEA

1. Ejecutarnos y consultamos la siguiente función de ejemplo dbo.GetNums() toma como parámetros: @low (bigint) y @high (bigint).



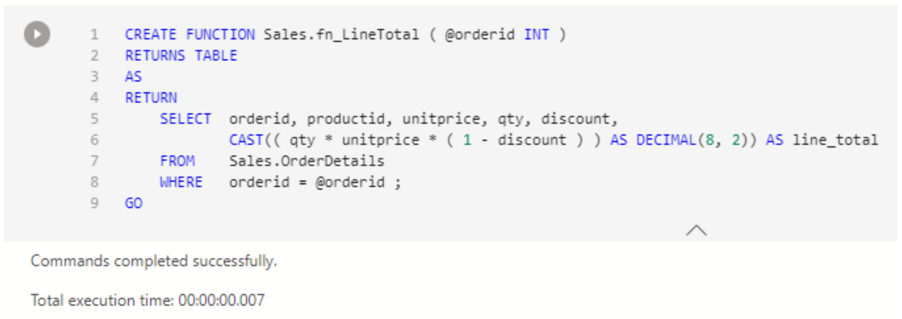
```
SELECT * FROM dbo.GetNums(10,20);
GO
```

```
(11 rows affected)

Total execution time: 00:00:01.009
```

	n
1	10
2	11
3	12
4	13
5	14
6	15
7	16
8	17
9	18
10	19
11	20

2. Creamos una función para calcular para extensión de líneas para órdenes de compra.

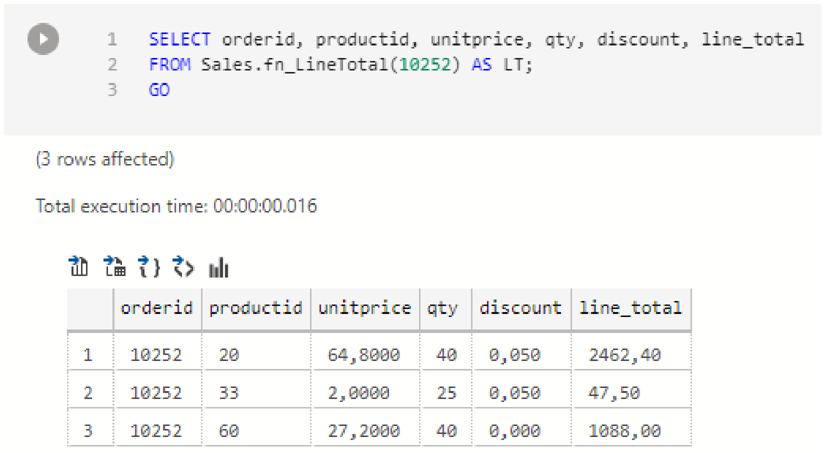


```
CREATE FUNCTION Sales.fn_LineTotal ( @orderid INT )
RETURNS TABLE
AS
RETURN
    SELECT  orderid, productid, unitprice, qty, discount,
            CAST(( qty * unitprice * ( 1 - discount ) ) AS DECIMAL(8, 2)) AS line_total
    FROM    Sales.OrderDetails
    WHERE   orderid = @orderid ;
GO
```

```
Commands completed successfully.

Total execution time: 00:00:00.007
```

3. Utilizamos la función.



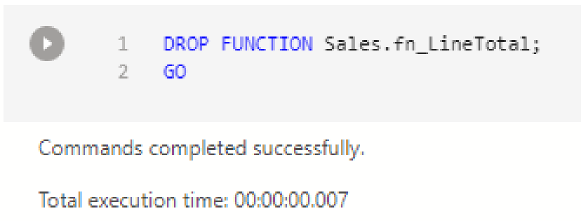
```
SELECT orderid, productid, unitprice, qty, discount, line_total
FROM Sales.fn_LineTotal(10252) AS LT;
GO
```

```
(3 rows affected)

Total execution time: 00:00:00.016
```

	orderid	productid	unitprice	qty	discount	line_total
1	10252	20	64,8000	40	0,050	2462,40
2	10252	33	2,0000	25	0,050	47,50
3	10252	60	27,2000	40	0,000	1088,00

4. Limpiamos los cambios realizados.



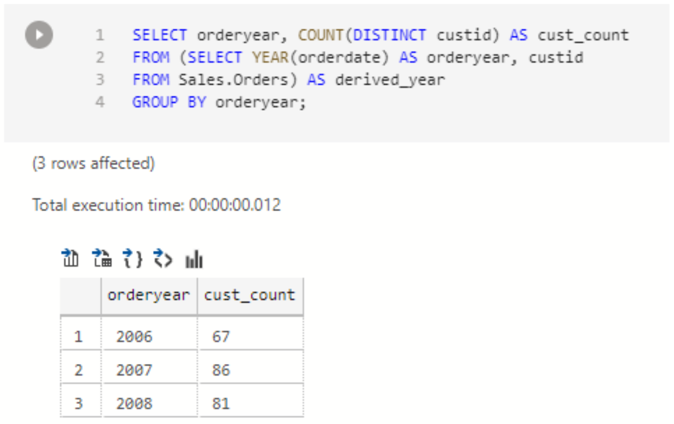
```
DROP FUNCTION Sales.fn_LineTotal;
GO
```

```
Commands completed successfully.

Total execution time: 00:00:00.007
```

PARTE 3: UTILIZANDO TABLAS DERIVADAS

1. Visualizando alias internos de columnas.



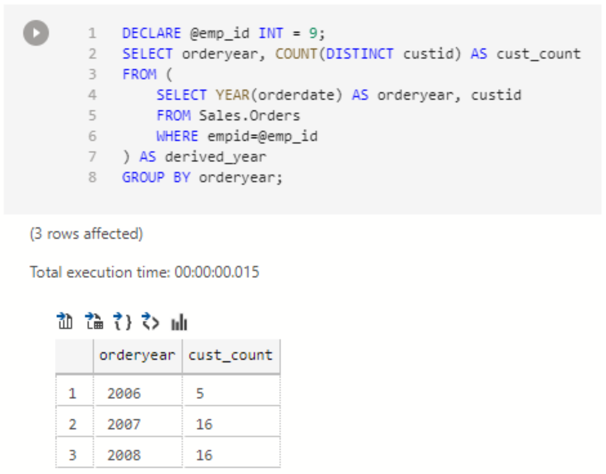
```sql
SELECT orderyear, COUNT(DISTINCT custid) AS cust_count
FROM (SELECT YEAR(orderdate) AS orderyear, custid
FROM Sales.Orders) AS derived_year
GROUP BY orderyear;
```

```
(3 rows affected)

Total execution time: 00:00:00.012
```

	orderyear	cust_count
1	2006	67
2	2007	86
3	2008	81

2. Utilizamos una variable como parámetro de una tabla derivada.



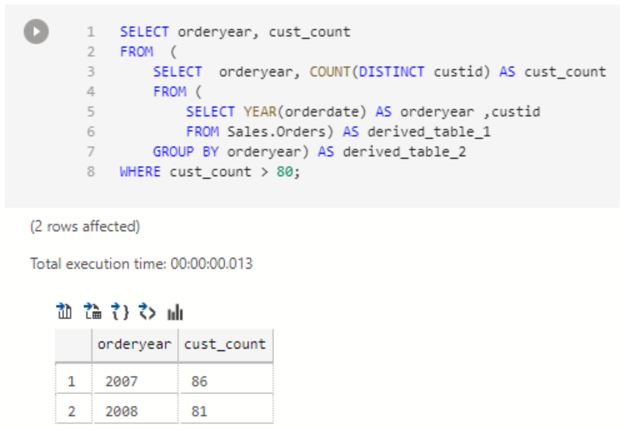
```sql
DECLARE @emp_id INT = 9;
SELECT orderyear, COUNT(DISTINCT custid) AS cust_count
FROM (
    SELECT YEAR(orderdate) AS orderyear, custid
    FROM Sales.Orders
    WHERE empid=@emp_id
) AS derived_year
GROUP BY orderyear;
```

```
(3 rows affected)

Total execution time: 00:00:00.015
```

	orderyear	cust_count
1	2006	5
2	2007	16
3	2008	16

3. Enlazamos tablas derivadas.



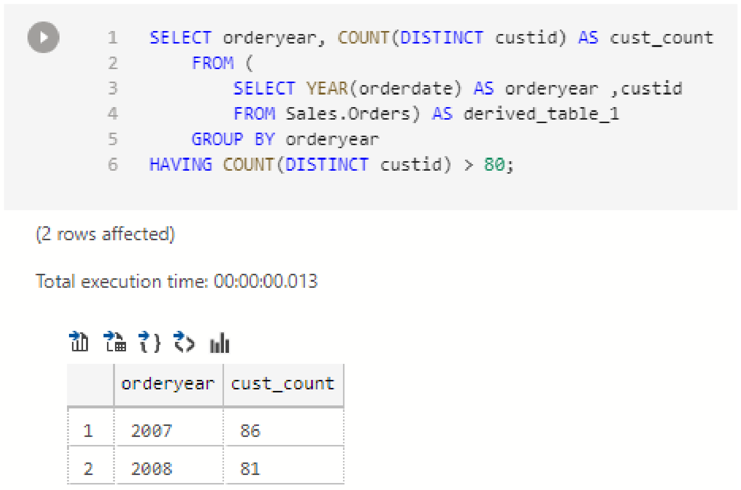
```sql
SELECT orderyear, cust_count
FROM  (
    SELECT  orderyear, COUNT(DISTINCT custid) AS cust_count
    FROM (
        SELECT YEAR(orderdate) AS orderyear ,custid
        FROM Sales.Orders) AS derived_table_1
    GROUP BY orderyear) AS derived_table_2
WHERE cust_count > 80;
```

```
(2 rows affected)

Total execution time: 00:00:00.013
```

	orderyear	cust_count
1	2007	86
2	2008	81

4. Una alternativa al ejemplo anterior sería.



```sql
SELECT orderyear, COUNT(DISTINCT custid) AS cust_count
    FROM (
        SELECT YEAR(orderdate) AS orderyear ,custid
        FROM Sales.Orders) AS derived_table_1
    GROUP BY orderyear
HAVING COUNT(DISTINCT custid) > 80;
```

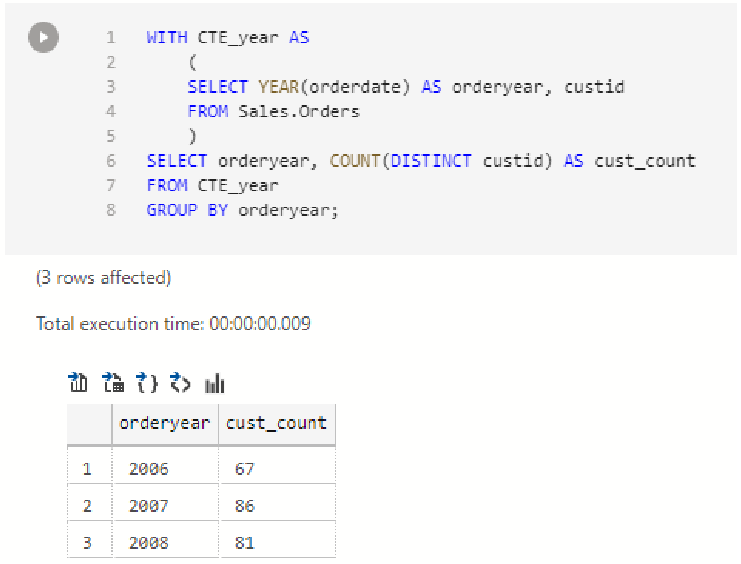
```
(2 rows affected)

Total execution time: 00:00:00.013
```

	orderyear	cust_count
1	2007	86
2	2008	81

PARTE 4: UTILIZANDO EXPRESIONES COMUNES DE TABLA

1. Ejecutaremos la siguiente expresión.



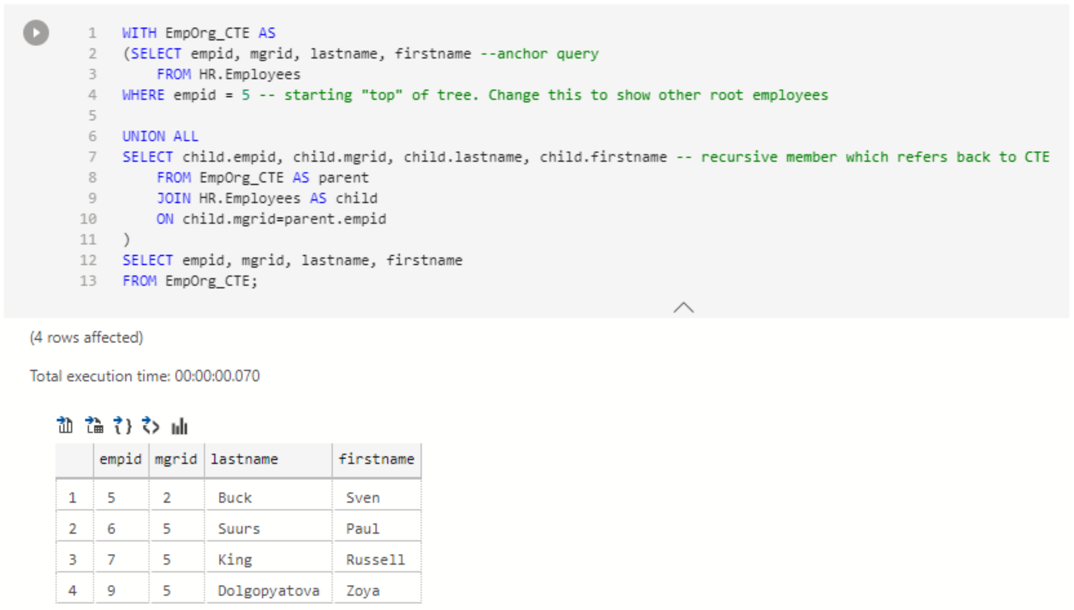
```sql
WITH CTE_year AS
    (
    SELECT YEAR(orderdate) AS orderyear, custid
    FROM Sales.Orders
    )
SELECT orderyear, COUNT(DISTINCT custid) AS cust_count
FROM CTE_year
GROUP BY orderyear;
```

```
(3 rows affected)

Total execution time: 00:00:00.009
```

	orderyear	cust_count
1	2006	67
2	2007	86
3	2008	81

2. También se puede utilizar de modo recursivo.



```
WITH EmpOrg_CTE AS
(SELECT empid, mgrid, lastname, firstname --anchor query
    FROM HR.Employees
WHERE empid = 5 -- starting "top" of tree. Change this to show other root employees

UNION ALL
SELECT child.empid, child.mgrid, child.lastname, child.firstname -- recursive member which refers back to CTE
    FROM EmpOrg_CTE AS parent
    JOIN HR.Employees AS child
    ON child.mgrid=parent.empid
)
SELECT empid, mgrid, lastname, firstname
FROM EmpOrg_CTE;
```

```
(4 rows affected)

Total execution time: 00:00:00.070
```

	empid	mgrid	lastname	firstname
1	5	2	Buck	Sven
2	6	5	Suurs	Paul
3	7	5	King	Russell
4	9	5	Dolgopyatova	Zoya