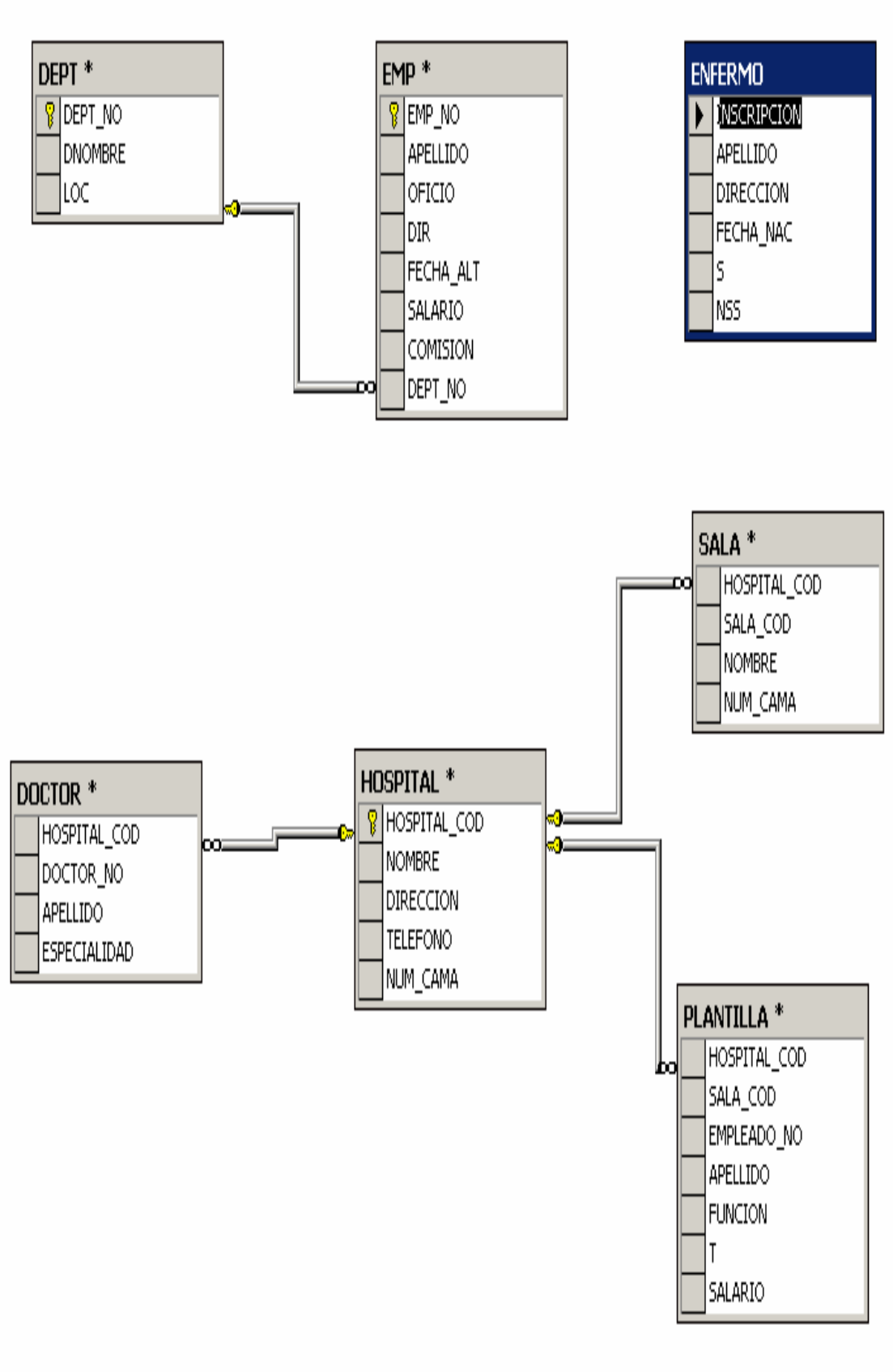
ESTRUCTURA BASE DE DATOS HOSPITAL

RELACIONES ENTRE TABLAS



TIPOS DE DATOS DE LAS COLUMNAS

EMP

	Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
▶	EMP_NO	int	4	✓
	APELLIDO	nvarchar	50	✓
	OFICIO	nvarchar	50	✓
	DIR	int	4	✓
	FECHA_ALT	smalldatetime	4	✓
	SALARIO	int	4	✓
	COMISION	int	4	✓
	DEPT_NO	int	4	✓

DEPT

	Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
	DEPT_NO	int	4	✓
	DNOMBRE	nvarchar	50	✓
	LOC	nvarchar	50	✓

HOSPITAL

	Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
	HOSPITAL_COD	int	4	✓
	NOMBRE	nvarchar	50	✓
	DIRECCION	nvarchar	50	✓
	TELEFONO	nvarchar	50	✓
	NUM_CAMA	int	4	✓

DOCTOR

Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
HOSPITAL_COD	int	4	✓
DOCTOR_NO	int	4	✓
APELLIDO	nvarchar	50	✓
ESPECIALIDAD	nvarchar	50	✓

PLANTILLA

Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
HOSPITAL_COD	int	4	✓
SALA_COD	int	4	✓
EMPLEADO_NO	int	4	✓
APELLIDO	nvarchar	50	✓
FUNCION	nvarchar	50	✓
T	nvarchar	50	✓
SALARIO	int	4	✓

SALA

Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
HOSPITAL_COD	int	4	✓
SALA_COD	int	4	✓
NOMBRE	nvarchar	50	✓
NUM_CAMA	int	4	✓

ENFERMO

Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
INSCRIPCION	int	4	✓
APELLIDO	nvarchar	50	✓
DIRECCION	nvarchar	50	✓
FECHA_NAC	smalldatetime	4	✓
S	nvarchar	50	✓
NSS	int	4	✓

DATOS DE LA TABLA EMP

	EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
1	7369	SANCHEZ	EMPLEADO	7902	1980-12-17 00:00:00	104000	0	20
2	7499	ARROYO	VENDEDOR	7698	1981-02-22 00:00:00	208000	39000	30
3	7521	SALA	VENDEDOR	689	1981-02-22 00:00:00	162500	65000	30
4	7566	JIMENEZ	DIRECTOR	7839	1981-04-02 00:00:00	386750	0	20
5	7654	MARTIN	VENDEDOR	7698	1981-09-28 00:00:00	182000	182000	30
6	7698	NEGRO	DIRECTOR	7839	1981-05-01 00:00:00	370500	0	30
7	7782	CEREZO	DIRECTOR	7839	1981-06-09 00:00:00	318500	0	10
8	7788	NINO	ANALISTA	7566	1987-03-30 00:00:00	390000	0	20
9	7839	REY	PRESIDENTE	0	1981-11-17 00:00:00	650000	0	10
10	7844	TOVAR	VENDEDOR	7698	1981-09-08 00:00:00	195000	0	30
11	7876	ALONSO	EMPLEADO	7788	1987-05-03 00:00:00	143000	0	20
12	7900	JIMENO	EMPLEADO	7698	1981-12-03 00:00:00	123500	0	30
13	7902	FERNANDEZ	ANALISTA	7566	1981-12-03 00:00:00	390000	0	20
14	7934	MUÑOZ	EMPLEADO	7782	1982-06-23 00:00:00	169000	0	10
15	7119	SERRA	DIRECTOR	7839	1983-11-19 00:00:00	225000	39000	20
16	7322	GARCIA	EMPLEADO	7119	1982-10-12 00:00:00	129000	0	20

DATOS DE LA TABLA DEPT

DEPT_NO	DNOMBRE	LOC
10	CONTABILIDAD	ELCHE
20	INVESTIGACION	MADRID
30	VENTAS	BARCELONA
40	PRODUCCION	SALAMANCA

DATOS DE LA TABLA HOSPITAL

HOSPITAL_COD	NOMBRE	DIRECCION	TELEFONO	NUM_CAMA
19	Provincial	O' Donell 50	964-4256	502
18	General	Atocha s/n	595-3111	987
22	La Paz	Castellana 1000	923-5411	412
45	San Carlos	Ciudad Universitaria	597-1500	845

DATOS DE LA TABLA DOCTOR

HOSPITAL_COD	DOCTOR_NO	APELLIDO	ESPECIALIDAD
22	386	Cabeza D.	Psiquiatría
22	398	Best D.	Urología
19	435	López A.	Cardiología
22	453	Galo D.	Pediatría
45	522	Adams C.	Neurología
18	585	Miller G.	Ginecología
45	607	Chuki P.	Pediatría
18	982	Cajal R.	Cardiología

DATOS DE LA TABLA PLANTILLA

HOSPITAL_COD	SALA_COD	EMPLEADO_NO	APELLIDO	FUNCION	T	SALARIO
22	6	1009	Higueras D.	Enfermera	T	200500
45	4	1280	Amigo R.	Interino	N	221000
19	6	3106	Hernández	Enfermero	T	275000
19	6	3754	Díaz B.	Enfermera	T	226200
22	1	6065	Rivera G.	Enfermera	N	162600
18	4	6357	Karplus W.	Interino	T	337900
22	1	7379	Carlos R.	Enfermera	T	211900
22	6	8422	Bocina G.	Enfermero	M	183800
45	1	8526	Frank H.	Enfermera	T	252200
22	2	9901	Núñez C.	Interino	M	221000

DATOS DE LA TABLA ENFERMO

INSCRIPCION	APELLIDO	DIRECCION	FECHA_NAC	S	NSS
10995	Laguía M.	Goya 20	16-may-56	M	280862422
14024	Fernández M.	Recoletos 50	21-may-60	F	284991452
18004	Serrano V.	Alcalá 12	23-jun-67	F	321790059
36658	Domin S.	Mayor 71	01-ene-42	M	160654471
38702	Neal R.	Orense 11	18-jun-40	F	380010217
39217	Cervantes M.	Perón 38	29-feb-52	M	440294390
59076	Miller B.	López de Hoyos 2	16-sep-45	F	311969044
63827	Ruiz P.	Ezquerdo 103	26-dic-80	M	100973253
64823	Fraiser A.	Soto 3	10-jul-80	F	285201776
74835	Benítez E.	Argentina	05-oct-57	M	154811767

DATOS DE LA TABLA SALA

HOSPITAL_COD	SALA_COD	NOMBRE	NUM_CAMA
22	1	Recuperación	10
45	1	Recuperación	15
22	2	Maternidad	34
45	2	Maternidad	24
19	3	Cuidados Intensivos	21
18	3	Cuidados Intensivos	10
18	4	Cardiología	53
45	4	Cardiología	55
19	6	Psiquiátricos	67
22	6	Psiquiátricos	118

PRÁCTICA Nº__: CONSULTAS DE SELECCIÓN

NOMBRE:

CURSO:

EDICIÓN:

---

1. Mostrar todos los datos de los empleados de nuestra tabla emp.

```
select * from emp
```

2. Mostrar el apellido, oficio, salario anual, con las dos extras para aquellos empleados con comisión mayor de 100000.

```
SELECT APELLIDO, OFICIO,
SALARIO, SALARIO * 14 AS "SALARIO ANUAL" FROM EMP
```

3. Idem del anterior , pero para aquellos empleados que su salario anual con extras supere los 2.200.000 ptas.

```
SELECT APELLIDO, OFICIO,
SALARIO, SALARIO * 14 AS "SALARIO ANUAL" FROM EMP
WHERE SALARIO * 14 > 2200000
```

4. Idem del anterior, pero para aquellos empleados que sumen entre salario anual con extras y comisión los 3.000.000 millones.

```
SELECT APELLIDO, OFICIO,
SALARIO, SALARIO * 14 AS "SALARIO ANUAL" FROM EMP
WHERE SALARIO * 14 + comision > 3000000
```

5. Mostrar todos los datos de empleados ordenados por departamento y dentro de este por oficio para tener una visión jerárquica.

```
select * from emp order by dept_no, oficio
```

6. Mostrar todas las salas para el hospital 45.

```
select * from sala where hospital_cod = 45
```

7. Mostrar todos los enfermos nacidos antes de 1970.

```
select * from enfermo where fecha_nac < '01/01/1970'
```

8. Igual que el anterior, para los nacidos antes de 1970 ordenados por número de inscripción descendente

```
select * from enfermo where fecha_nac < '01/01/1970'
order by inscripcion desc
```

9. Listar todos los datos de la plantilla del hospital del turno de mañana

```
select * from plantilla where T ='M'
```

10. Idem del turno de noche.

```
select * from plantilla where t='N'
```

11. Visualizar los empleados de la plantilla del turno de mañana que tengan un salario entre 200.000 y 225.000 ptas.

```
select * from plantilla where salario between 200000 and 225000
```

12. Visualizar los empleados de la tabla emp que no se dieron de alta entre el 01/01/80 y el 12/12/82.

```
select * from emp where fecha_alt not between '01/01/1980' and
'31/12/1982'
```

13. Mostrar los nombres de los departamentos situados en Madrid o en Barcelona.

```
select DNOMBRE from dept where LOC in ('MADRID','BARCELONA')
```

PRÁCTICA Nº__: CONSULTAS DE SELECCIÓN II

NOMBRE:

CURSO:

EDICIÓN:

---

1. Mostrar aquellos empleados con fecha de alta posterior al 1 de Julio de 1985.

```
select * from emp
where fecha_alt > '01-01-1985'
```

2. Lo mismo que en el ejercicio 1 pero con salario entre 150000 y 400000.

```
select * from emp
where fecha_alt > '01-01-1985'
and salario between 150000 and 4000000
```

3. Igual que en el ejercicio 2, pero también incluimos aquellos que no siendo analista pertenecen al departamento 20.

```
select * from emp
where fecha_alt > '01-01-1985'
and salario between 150000  and 4000000
or (oficio <> 'ANALISTA' and  dept_no = 20)
```

4. Mostrar aquellos empleados cuyo apellido termine en 'Z' ordenados por departamento, y dentro de este por antigüedad.

```
select * from emp
where apellido like '%z'
order by dept_no, fecha_alt asc
```

5. De los empleados del ejercicio 5 quitar aquellos que superen las 200000 ptas mensuales.

```
select * from emp
where apellido like '%z'
and salario > 200000
order by dept_no, fecha_alt asc
```

6. Mostrar todos los empleados cuyo oficio no sea analista.

```
select * from emp
where oficio <> 'ANALISTA'
```

7. Igual que el 6, pero mostrandolos de forma que se aprecien las diferencias de salario dentro de cada oficio.

```
select * from emp
where oficio <> 'ANALISTA'
order by oficio, salario desc
```

8. De los del 7, nos quedamos solo con aquellos cuyo número de empleado no este entre 7600 y 7900.

```
select * from emp
where oficio <> 'ANALISTA'
and emp_no not between 7600 and 7900
order by oficio, salario desc
```

PRÁCTICA Nº__: CONSULTAS DE SELECCIÓN III

NOMBRE:

CURSO:

EDICIÓN:

---

9. Mostrar los distintos oficios de los empleados.

```
select distinct oficio from emp
```

10. Mostrar los distintos nombres de sala.

```
select distinct nombre from sala
```

11. Mostrar que personal "No Interino" existe en cada sala de cada hospital, ordenado por hospital y sala.

```
select hospital_cod, sala_cod, apellido, funcion
from plantilla
where funcion not in ('interino')
order by hospital_cod, sala_cod
```

12. Justificar el resultado de la siguiente consulta SELECT APELLIDO DISTINCT DEPT_NO FROM EMP;. Indicar que ocurre y modificarla para que todo vaya bien.

```
select distinct apellido,dept_no from emp
```

13. Seleccionar los distintos valores del sexo que tienen los enfermos.

```
select distinct s as "SEXO"
from enfermo
```

14. Indicar los distintos turnos de la plantilla del hospital, ordenados por turno y por apellido.

```
select distinct t as "TURNO", apellido
from plantilla
order by turno, apellido
```

15. Seleccionar las distintas especialidades que ejercen los médicos, ordenados por especialidad y apellido.

```
select distinct especialidad, apellido
from doctor
order by especialidad, apellido
```

MANUAL TRANSACT SQL

Existen 3 tipos de instrucciones para el lenguaje en SQL.

- **Lenguaje de control de datos (DDL) :** Creación y eliminación de tipos de datos y objetos.

  . CREATE Crear Objeto
  . ALTER Modificar los datos creados
  . DROP Eliminar el Objeto

- **Lenguaje de control de datos (DCL) :** Se basa en los derechos que tiene el usuario sobre la base da datos (Permisos).

  . GRANT Dar permisos a un usuario para efectuar determinadas instrucciones
  . DENY Eliminar el permiso que se ha concedido con el GRANT
  . REVOKE Eliminar todos los permisos

- **Lenguaje de manipulación de datos (DML) :** Desarrollo de la programación de la base de datos.

  . SELECT
  . INSERT
  . UPDATE
  . DELETE

Elementos de sintaxis:

Directivas de procesos por lotes

- **GO:** Envia lotes de intrucciones de TRANSACT SQL a las herramientas y utilidades (Sirve para separar bloques de instrucciones)
- **EXEC O EXECUTE:** Ejecuta funciones definidas por el usuario, procedimientos de sistema y procedimientos almacenados.

Comentarios en SQL:

- En línea: **--**
- En Bloque: **/*** *comentario* ***/**

Tablas en SQL:

Tabla master: Es la tabla que contiene como generar una base de datos y sobre ella, se crean todas las bases de datos.
Tabla model: Es la tabla modelo, las bases de datos creadas se basan en esta tabla como modelo.
Tabla Northwind y Pubs: Son tablas de ejemplos que vienen con SQL y todo usuario puede trabajar con ellas.

Identificadores para los objetos:

Los nombres que se le dan a las tablas, lo primero es que no pueden empezar por un número, deben empezar por un signo alfabético, pueden incluir el guion bajo (_), la arroba @ y la almohadilla #.
Generalmente para las variables locales se usan @ + el nombre.
EJEMPLO: @Contador.
Para las variables totales se usan dos arrobas + el nombre @@Contador
EJEMPLO: @@Error
#Nombre: indica una tabla o procedimiento temporal (Local)
##Nombre: Igual que el anterior pero global.

Tipos de datos:

- **Numéricos:**
  - Enteros int, tinyint, smallint, bigint
  - Decimales numeric, decimal, money, smallmoney
  - Coma Flotante float, real

- **Fechas:**
  - datetime 0,333 s
  - smalldatetime 1 minuto

- **Caracteres:**
  - Ancho fijo: char, nchar
  - Ancho Variable: varchar, nvarchar

- **Texto e Imagen:**
  - Text
  - Ntext
  - Rowversion

- **Binario:**
  - Binary, varbinary Valores tipo byte
  - Bit Un solo bit (1 o ninguno)

- **Identificadores Unicos:**

  Representa un identificador global único (GUID)
  Si queremos que no se repita el dato en la base de datos, usamos este identificador

  - Uniqueidentifier

OPERADOR LIKE

% Cualquier número de caracteres
_ Para un carácter individual
[ ] Para un conjunto de caracteres que esté dentro del corchete
[^] Que el carácter individual que no esté dentro del corchete

EJEMPLO: **LIKE '%een'** Muestra todos los caracteres que acaben con een
EJEMPLO: **LIKE '%een%'** Muestra todos los caracteres que contengan een en ese orden
EJEMPLO: **LIKE '_en'** Muestra todos los caracteres que contenga tres letras y acaben en en
EJEMPLO: **LIKE '[CK%]'** Muestra todos los caracteres que empiecen por C o K
EJEMPLO: **LIKE '[S-V]ing'** Nombre de 4 letras cuya primera letra estuviera entre S o V y acabe en ing
EJEMPLO: **LIKE 'M[^c]%'** Todos los que empiecen por M y segunda letra no sea una c. No hay limite de caracteres.

CONSULTAS CON LIKE

1. Seleccionar todos los empleados cuyo apellido comience por M

```
select * from emp where apellido like 'M%'
```

	EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
1	7654	MARTIN	VENDEDOR	7698	1981-09-28 00:00:00	182000	182000	30
2	7934	MUÑOZ	EMPLEADO	7782	1982-06-23 00:00:00	169000	0	10

2. Seleccionar todos los empleados cuyo apellido termine con la letra Z

```
select * from emp where apellido like '%z'
```

	EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
1	7369	SANCHEZ	EMPLEADO	7902	1980-12-17 00:00:00	104000	1	20
2	7566	JIMENEZ	DIRECTOR	7839	1981-04-02 00:00:00	386750	0	20
3	7902	FERNANDEZ	ANALISTA	7566	1981-12-03 00:00:00	390000	0	20
4	7934	MUÑOZ	EMPLEADO	7782	1982-06-23 00:00:00	169000	0	10

3. Seleccionar todos los empleados que contengan en su apellido ER.

```
select * from emp where apellido like '%er%'
```

	EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
1	7782	CEREZO	DIRECTOR	7839	1981-06-09 00:00:00	318500	0	10
2	7902	FERNANDEZ	ANALISTA	7566	1981-12-03 00:00:00	390000	0	20

4. Mostrar todos los empleados cuyo nombre sea de 4 letras y su apellido termine con la letra a

```
select * from emp where apellido like '___a'
```

	EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
1	7521	SALA	VENDEDOR	698	1981-02-22 00:00:00	162500	65000	30

5. Mostrar todos los empleados cuyo apellido comience entre las letras E y F.

```
select * from emp where apellido like '[E-F]%'
```

	EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
1	7902	FERNANDEZ	ANALISTA	7566	1981-12-03 00:00:00	390000	0	20

6. Mostrar todos los empleados cuyo apellido comience por la letra A, contenga dentro de su apellido de la letra A a la M y que terminen en O.

```
select * from emp where apellido like 'A%[a-m]%o'
```

	EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
1	8888	Angulo	NULL	NULL	1999-12-12 00:00:00	NULL	NULL	20
2	8888	Angulo	NULL	NULL	1999-12-12 00:00:00	NULL	NULL	20

7. Mostrar todos los empleados cuyo apellido comience por la letra M y la segunda letra no sea una A.

```
select * from emp where apellido like 'M[^A]%'
```

	EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
1	7934	MUÑOZ	EMPLEADO	7782	1982-06-23 00:00:00	169000	0	10

8. Mostrar todos los empleados cuyo apellido sea de 5 letras y su tercera letra sea entra la A y la S terminando en Z.

```
select * from emp where apellido like '__[a-ñ]_z'
```

EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
7934	MUÑOZ	EMPLEADO	7782	1982-06-23 00:00:00	169000	0	10

9. Mostrar todos los empleados cuyo apellido sea de 6 letras y no comience entre la A y la D.

```
select * from emp where apellido like '[^a-d]_____'
```

EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
7654	MARTIN	VENDEDOR	7698	1981-09-28 00:00:00	182000	182000	30
7900	JIMENO	EMPLEADO	7698	1981-12-03 00:00:00	123500	0	30

10. Mostrar todos los que empiecen por la A y cuya cuarta letra no esté comprendida entre A – G

```
select * from emp where apellido like 'A__[^a-g]%'
```

EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
7499	ARROYO	VENDEDOR	7698	1981-02-22 00:00:00	208000	39000	30
8888	Angulo	NULL	NULL	1999-12-12 00:00:00	NULL	NULL	20
8888	Angulo	NULL	NULL	1999-12-12 00:00:00	NULL	NULL	20

Cómo se procesan las consultas no almacenadas en caché (ad hoc)

Analizar = Resolver = Optimizar = Compilar = Ejecutar

Consultas almacenadas en caché
Primera ejecución

Analizar = Resolver = Optimizar = Compilar = Ejecutar
||
Ejecución Posterior ||
Procedimiento en Caché = Ejecutar

Cache es una parte de la memoria en la que se almacenan las partes de ejecución de las Consultas.

La mejor forma de almacenar las consultas en caché es con bloques ad hoc:
Lotes ad hoc (Se limitan a coincidencias exactas) :

```
SELECT * FROM Productos WHERE Precio = 100
SELECT * FROM Productos WHERE Precio = 50
SELECT * FROM Productos WHERE Precio = 100
```

Otra forma de almacenar consultas en cache Parametrización automatica:
Guarda las constantes que se le pasan con unos parámetros, para la próxima vez que haya una consulta parecida, ya la tiene pero con otros parámetros.

```
SELECT * FROM Productos WHERE Precio = 100
SELECT * FROM Productos WHERE Precio = 50
SELECT * FROM Productos WHERE Precio = 120
```

*Esto se hace automáticamente.

Agrupar y resumir datos

Sacar los *n* primeros valores

SELECT TOP n Nombre, Apellido... (Saca los n primeros valores)
SELECT TOP n PERCENT Nombre, Apellido... (Saca el n por ciento)
SELECT TOP n WITH TIES Nombre, Apellido.. ORDER BY (Saca los n primeros ordenados por lo que sea y con coincidencias en el valor n.)

Select top 5 emp_no, apellido from emp Devuelve los 5 primeros empleados

Select top 50 percent apellido from emp Devuelve el 50% de los empleados

```
select top 3 with ties oficio, apellido from emp
where oficio = 'EMPLEADO'
order by oficio    Devuelve los 3 primeros registros con oficio
empleado, y si existe algún empleado más, también lo devuelve, porque
coincide con lo que buscamos.
```

Funciones de agregado:

Son funciones que se utilizan para calcular valores en las tablas. Si queremos usarlas combinándolas junto con otros campos debemos utilizar Group by y agrupar los datos que no son funciones.
Con la sentencia group by no se utiliza la clausula where, se utilizara una clausula propia de la expresión: HAVING. Equivalente a where

- COUNT: Cuenta los registros que hay en la consulta.
  Si pongo un valor dentro de la expresión devolverá la cuenta de todos los registros no nulos.
  Si pongo un asterisco contará todos los registros aunque tengan valores nulos.

```
select count(*) from emp      Valores con Nulos
select count(oficio) from emp      Valores sin nulos
```

- AVG: Realiza la media sobre la expresión dada, debe ser un tipo de dato Int.

```
select avg(salario) from emp
```

- MAX: Saca el valor máximo de una consulta.

```
select max(fecha_alt) from emp
```

- MIN: Devuelve el valor mínimo de una consulta.

```
select min(fecha_alt) from emp
```

- SUM: Devuelve la suma de los salarios

```
select  sum(salario) from emp
```

Operadores de SQL:

- Lógicos:

  AND, OR , NOT

- De Comparación:

=	Igual
<	Menor
>	Mayor
<>	Diferente
>=	Mayor o igual
<=	Menor o igual

FUNCIONES DE AGREGADO

1. Encontrar el salario medio de los analistas, mostrando el número de los empleados con oficio analista.

```
select count(*) as [Numero de Empleados], oficio,
avg(salario) as [Salario Medio] from
emp group by oficio having oficio ='ANALISTA'
```

	Numero de Empleados	oficio	Salario Medio
1	3	ANALISTA	335000

2. Encontrar el salario mas alto, mas bajo y la diferencia entre ambos de todos los empleados con oficio EMPLEADO.

```
select  oficio, max(salario) as [Salario mas alto]
, min(salario) as [Salario mas Bajo]
, max(salario) - min(salario) as [Diferencia entre Ambos]
from emp group by oficio having oficio = 'EMPLEADO'
```

	oficio	Salario mas alto	Salario mas Bajo	Diferencia entre Ambos
1	EMPLEADO	169000	100000	69000

3. Visualizar los salarios mayores para cada oficio.

```
select oficio, max(salario) as [Salario Máximo] from emp group by oficio
```

	oficio	Salario Máximo
1	ANALISTA	390000
2	DIRECTOR	390000
3	EMPLEADO	169000
4	PRESIDENTE	650000
5	VENDEDOR	208000

4. Visualizar el número de personas que realizan cada oficio en cada departamento.

```
select dept_no as [Nº de departamento],
count(*) as [Nº de personas], oficio
from emp group by dept_no, oficio
order by 1
```

	N° de departamento	N° de personas	oficio
1	10	1	DIRECTOR
2	10	1	EMPLEADO
3	10	1	PRESIDENTE
4	20	1	VENDEDOR
5	20	2	DIRECTOR
6	20	3	ANALISTA
7	20	3	EMPLEADO
8	30	1	EMPLEADO
9	30	1	DIRECTOR
10	30	4	VENDEDOR
11	50	1	EMPLEADO

5. Buscar aquellos departamentos con cuatro o mas personas trabajando.

```
select dept_no as [Nº de departamento]
, count(*) as [Nº de personas] from emp
group by dept_no having count(*) > 3
```

	N° de departamento	N° de personas
1	20	9
2	30	6

6. Mostrar el número de directores que existen por departamento.

```
select count(*) as [Numero empleados], dept_no from emp
where oficio = 'director'
group by dept_no
```

	Numero empleados	dept_no
1	1	10
2	2	20
3	1	30

7. Visualizar el número de enfermeros, enfermeras e interinos que hay en la plantilla, ordenados por la función.

```
select count(*) as [Nº de personas], FUNCION from plantilla
group by funcion
having funcion in ('ENFERMERO','ENFERMERA','INTERINO')
order by funcion
```

	N° de personas	FUNCION
1	5	Enfermera
2	2	Enfermero
3	3	Interino

8. Visualizar departamentos, oficios y número de personas, para aquellos departamentosque tengan dos o más personas trabajando en el mismo oficio.

```
select dept_no as [Nº de Departamento],count(*) as [Nº de personas],
oficio from emp group by dept_no,oficio  having count(*) > 1
```

	N° de Departamento	N° de personas	oficio
1	20	3	ANALISTA
2	20	2	DIRECTOR
3	20	3	EMPLEADO
4	30	4	VENDEDOR

9. Calcular el salario medio, Diferencia, Máximo y Mínimo de cada oficio. Indicando el oficio y el número de empleados de cada oficio.

```
select oficio, count(*) as [nº de empleados], min(salario) as [Salario mínimo]
, max(salario) as [Salario máximo], max(salario) - min(salario) as [Diferencia]
, avg(salario) as [Media] from emp group by  oficio
```

	oficio	nº de empleados	Salario mínimo	Salario máximo	Diferencia	Media
1	ANALISTA	3	225000	390000	165000	335000
2	DIRECTOR	4	318500	390000	71500	366437
3	EMPLEADO	6	100000	169000	69000	124166
4	PRESIDENTE	1	650000	650000	0	650000
5	VENDEDOR	5	129000	208000	79000	175300

10. Calcular el valor medio de las camas que existen para cada nombre de sala. Indicar el nombre de cada sala y el número de cada una de ellas.

```
select sala_cod as [Sala], Nombre
, avg(num_cama) as [Media de Camas]
from sala group by nombre, sala_cod
```

	Sala	Nombre	Media de Camas
1	1	Recuperación	12
2	2	Maternidad	29
3	3	Cuidados Intensivos	15
4	4	Cardiología	54
5	6	Psiquiátricos	92

11. Calcular el salario medio de la plantilla de la sala 6, según la función que realizan. Indicar la función y el número de empleados.

```
select count(*) as [Nº de empleados], funcion, avg(salario) as [Salario Medio]
from plantilla group by funcion, sala_cod having sala_cod = 6
```

	Nº de empleados	funcion	Salario Medio
1	2	Enfermera	213350
2	2	Enfermero	229400

12. Averiguar los últimos empleados que se dieron de alta en la empresa en cada uno de los oficios, ordenados por la fecha.

```
select max(fecha_alt) as [Fecha], Oficio from emp
group by oficio
order by 1
```

	Fecha	Oficio
1	1981-10-10 00:00:00	DIRECTOR
2	1981-11-17 00:00:00	PRESIDENTE
3	1982-12-11 00:00:00	VENDEDOR
4	1987-03-30 00:00:00	ANALISTA
5	1987-05-03 00:00:00	EMPLEADO

13. Mostrar el número de hombres y el número de mujeres que hay entre los enfermos.

```
select count(*) as [Número],  s as [Sexo] from enfermo group by s
```

	Número	Sexo
1	5	F
2	5	M

14. Mostrar la suma total del salario que cobran los empleados de la plantilla para cada función y turno.

```
select funcion, t as [Turno], sum(salario) as [Suma de Salarios]
from plantilla group by funcion, t
```

15. Calcular el número de salas que existen en cada hospital.

```
select count(*) as [Nº Salas], hospital_cod from sala
group by hospital_cod
```

16. Mostrar el número de enfermeras que existan por cada sala.

```
select count(*) as [Nº Personas], sala_cod, funcion from plantilla
where funcion='enfermera'
group by sala_cod, Funcion
order by 1
```

CONSULTAS DE COMBINACIÓN

JOIN

Se usa para combinar resultados entre varias tablas. Microsoft recomienda usar Join ya que consume menos recursos.
Para ver como manejamos este tipo de consultas.

Consultas Internas

Combina las tablas comparando los valores comunes de los campos indicados mediante combinaciones cruzadas.
Sintaxis:
Select *TablaPrincipal.Campo, Tablaconlaquecombinar.Campo*
From *TablaPrincipal*
Inner Join / Full Join *Tablaconlaquecombinar*
On
Condición para combinar los campos

- ***Inner Join***: Indica que combine los campos con resultados comunes
- ***Full Join***: Indica que combine todos los campos aunque los resultados sean diferentes.

```
select apellido,oficio,dnombre
from emp
inner join dept
on emp.dept_no=dept.dept_no
order by dept.dnombre
```

Devuelve todos los Empleados
que tengan asociado un departamento.

apellido	oficio	dnombre
CEREZO	DIRECTOR	CONTABILIDAD
REY	PRESIDENTE	CONTABILIDAD
COALESCE	EMPLEADO	CONTABILIDAD
MUÑOZ	EMPLEADO	CONTABILIDAD
REY	PRESIDENTE	CONTABILIDAD
SANCHEZ	EMPLEADO	INVESTIGACION
JIMENEZ	DIRECTOR	INVESTIGACION
GIL	ANALISTA	INVESTIGACION
FERNANDEZ	ANALISTA	INVESTIGACION
COAL	EMPLEADO	INVESTIGACION
Angulo	NULL	INVESTIGACION

apellido	oficio	dnombre
SERRA	DIRECTOR	NULL
CEREZO	DIRECTOR	CONTABILIDAD
REY	PRESIDENTE	CONTABILIDAD
COALESCE	EMPLEADO	CONTABILIDAD
MUÑOZ	EMPLEADO	CONTABILIDAD
REY	PRESIDENTE	CONTABILIDAD
NULL	NULL	EDICION
Angulo	NULL	INVESTIGACION
Angulo	NULL	INVESTIGACION
COAL	EMPLEADO	INVESTIGACION
FERNANDEZ	ANALISTA	INVESTIGACION

```
select apellido,oficio,dnombre
from emp
full join dept
on emp.dept_no=dept.dept_no
order by dept.dnombre
```

La combinación **Full Join** muestra las coincidencias de la tabla Dept con Emp, más los valores que no coincidan, como el Empleado SERRA que no tiene departamento y el departamento EDICIÓN, que no tiene empleados.

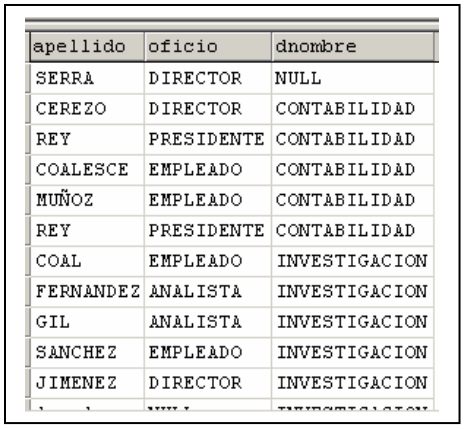
Se podría decir que es como la suma de utilizar left join y right join.

Consultas Externas

Al igual que las consultas de combinación internas, combina los valores comunes de los campos indicados y además de la tabla que queramos, devuelve también el resto de valores aunque no coincidan. Para ello usaremos las siguientes opciones combinadas con join:
Sintaxis:
Select tablaprincipal.campo, tablaacombinar.campo
From tablaprincipal
left join / right join / cross join tabla
on condición

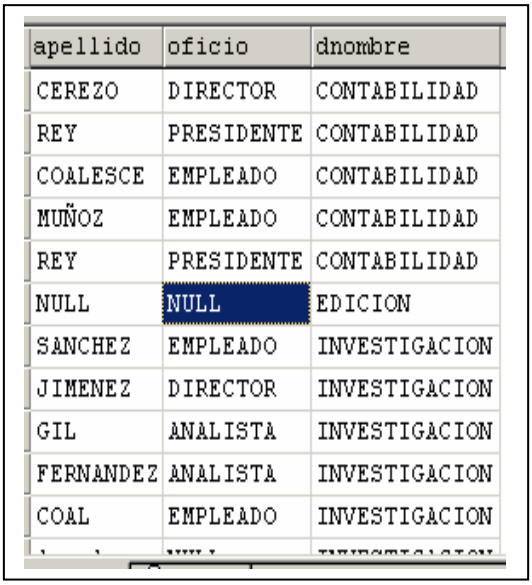
- ***left Join:*** Indica que muestre todos los resultados de la columna de la izquierda
- ***Right Join:*** Indica que muestre todos los resultados de la columna de la derecha
- ***Cross Join:*** Muestra un producto cartesiano combinando todos los resultados de las dos tablas.

```
select apellido,oficio,dnombre
from emp
left outer join dept
on emp.dept_no=dept.dept_no
order by dept.dnombre
```



apellido	oficio	dnombre
SERRA	DIRECTOR	NULL
CEREZO	DIRECTOR	CONTABILIDAD
REY	PRESIDENTE	CONTABILIDAD
COALESCE	EMPLEADO	CONTABILIDAD
MUÑOZ	EMPLEADO	CONTABILIDAD
REY	PRESIDENTE	CONTABILIDAD
COAL	EMPLEADO	INVESTIGACION
FERNANDEZ	ANALISTA	INVESTIGACION
GIL	ANALISTA	INVESTIGACION
SANCHEZ	EMPLEADO	INVESTIGACION
JIMENEZ	DIRECTOR	INVESTIGACION

El empleado Serra tiene el nombre del departamento con el valor null porque no tiene ningún departamento asociado y nosotros en la consulta le estamos diciendo que seleccione los empleados aunque no tengan departamento asociado, ponemos como principal la tabla de la izquierda (EMP).



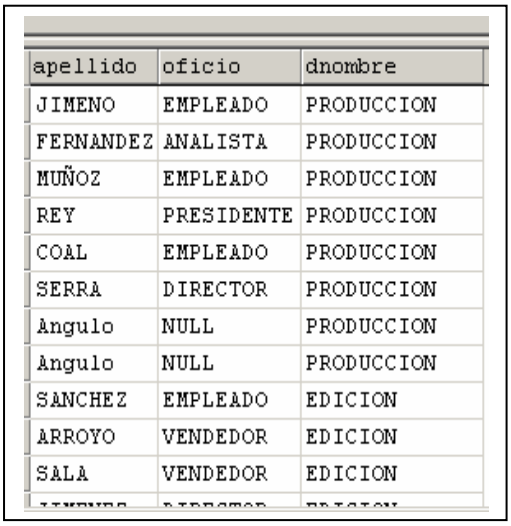
apellido	oficio	dnombre
CEREZO	DIRECTOR	CONTABILIDAD
REY	PRESIDENTE	CONTABILIDAD
COALESCE	EMPLEADO	CONTABILIDAD
MUÑOZ	EMPLEADO	CONTABILIDAD
REY	PRESIDENTE	CONTABILIDAD
NULL	NULL	EDICION
SANCHEZ	EMPLEADO	INVESTIGACION
JIMENEZ	DIRECTOR	INVESTIGACION
GIL	ANALISTA	INVESTIGACION
FERNANDEZ	ANALISTA	INVESTIGACION
COAL	EMPLEADO	INVESTIGACION

```
select apellido,oficio,dnombre
from emp
right outer join dept
on emp.dept_no=dept.dept_no
order by dept.dnombre
```

En esta consulta el departamento de edición tiene valores null porque le hemos dicho que seleccione la tabla de la derecha como principal (dept), con lo cual selecciona todos los campos de la tabla departamentos con coincidencias con emp o sin ellas.

```
select apellido,oficio,dnombre
from emp
cross join dept
```

Realiza un producto cartesiano combinando todos los empleados con todos los departamentos.



apellido	oficio	dnombre
JIMENO	EMPLEADO	PRODUCCION
FERNANDEZ	ANALISTA	PRODUCCION
MUÑOZ	EMPLEADO	PRODUCCION
REY	PRESIDENTE	PRODUCCION
COAL	EMPLEADO	PRODUCCION
SERRA	DIRECTOR	PRODUCCION
Angulo	NULL	PRODUCCION
Angulo	NULL	PRODUCCION
SANCHEZ	EMPLEADO	EDICION
ARROYO	VENDEDOR	EDICION
SALA	VENDEDOR	EDICION

Combinaciones con mas de dos tablas

Ya hemos visto como combinar 2 tablas con inner join, el siguiente ejemplo muestra como combinar las 3 tablas que tenemos en la base de datos. Podremos combinar tantas tablas como queramos usando inner join o full join.

Apellido y Nombre	Sala	Hospital	Nº de camas
Hernández J.	Psiquiátricos	Provincial	67
Díaz B.	Psiquiátricos	Provincial	67
Karplus W.	Cardiología	General	53
Rivera G.	Recuperación	La Paz	10
Carlos R.	Recuperación	La Paz	10
Higueras D.	Psiquiátricos	La Paz	118
Bocina G.	Psiquiátricos	La Paz	118
Núñez C.	Maternidad	La Paz	34
Amigo R.	Cardiología	San Carlos	55
Frank H.	Recuperación	San Carlos	13

```
select p.apellido as [Apellido]
,s.nombre as [Sala]
,h.nombre as [Hospital],
s.num_cama as [Nº de camas]
from plantilla p inner join sala as s
on p.hospital_cod = s.hospital_cod
and p.sala_cod = s.sala_cod
inner join hospital_cod as h
on h.hospital_cod = p.hospital_cod
```

Podremos usar tantos inner join como queramos en nuestras consultas, pero habrá que tener cuidado a la hora de realizar las combinaciones para que no salgan productos cartesianos en la consulta.
Esta consulta devuelve el nombre del empleado, el nombre de la sala donde trabaja, el nombre del hospital y el número de camas.

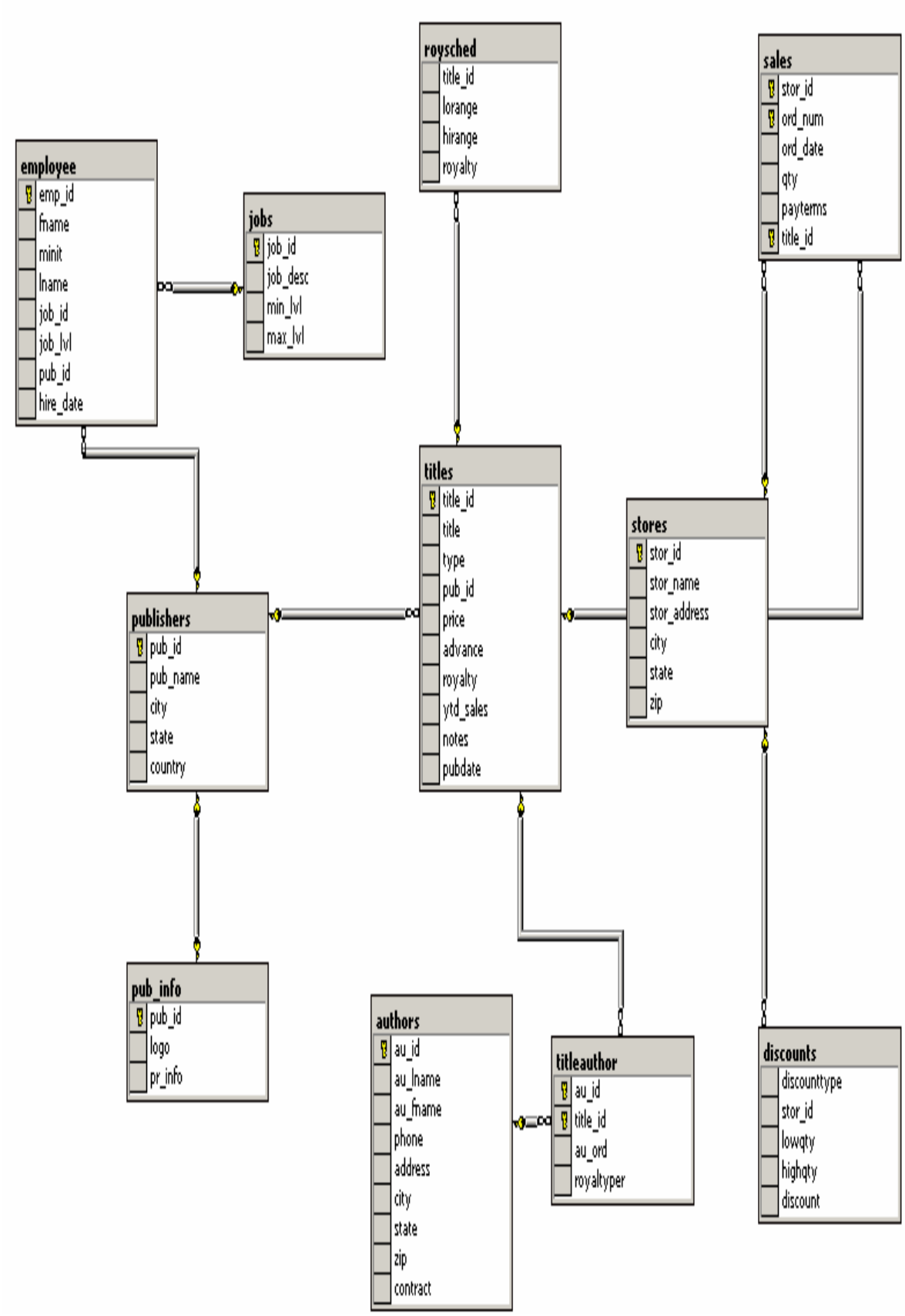
Combinar los valores de una tabla sobre sí misma

Para ello crearemos dos copias de la misma tabla poniéndole un alías, para posteriormente combinar los resultados de ambas copias.

```
select a.emp_no as [Primer Empleado]
,a.apellido,a.dept_no
,b.emp_no as [Segundo Empleado],
b.apellido
from emp as a inner join emp as b
on a.emp_no = b.emp_no
order by b.emp_no
```

Primer Empleado	apellido	dept_no	Segundo Empleado	apellido
7369	SANCHEZ	20	7369	SANCHEZ
7499	ARROYO	30	7499	ARROYO
7521	SALA	30	7521	SALA
7566	JIMENEZ	20	7566	JIMENEZ
7654	MARTIN	30	7654	MARTIN
7698	NEGRO	30	7698	NEGRO
7782	CEREZO	10	7782	CEREZO
7788	GIL	20	7788	GIL
7839	REY	10	7839	REY
7839	REY	10	7839	REY

DIAGRAMA DE LA TABLA PUBS



EJERCICIOS CON COMBINACIONES INTERNAS Y EXTERNAS

1) Realizar una consulta que muestre los nombres de los autores y editores ubicados en la misma ciudad

```
NOMBRE          APELLIDO                       EDITOR
--------------- -------------------------------- ----------------------------------------
Cheryl          Carson                         Algodata Infosystems
Abraham         Bennet                           Algodata Infosystems

(2 filas afectadas)
```

SOLUCION:

```
use pubs
select a.au_fname as [NOMBRE]
,a.au_lname as [APELLIDO]
,p.pub_name as [EDITOR]
from authors as a
inner join publishers as p
on
a.city = p.city
```

2) Obtener todos los nombres y editores de todos los libros cuyos anticipos pagados son superiores a 7500

```
TITULO                                                                           EDITOR                         ANTICIPO
-------------------------------------------------------------------------------- ------------------------------ ---------------------
You Can Combat Computer Stress!                                                  New Moon Books                  10125.0000
The Gourmet Microwave                                                            Binnet & Hardley                15000.0000
Secrets of Silicon Valley                                                        Algodata Infosystems             8000.0000
Sushi, Anyone?                                                                   Binnet & Hardley                 8000.0000

(4 filas afectadas)
```

SOLUCION:

```
use pubs
select t.title as [TITULO]
,p.pub_name as [EDITOR]
,t.advance as [ANTICIPO]
from titles as t
inner join publishers as p
on
t.pub_id=p.pub_id
where
t.advance>7500
```

3) Seleccionar todos los titulos, nombre y apellidos del autor de todos los libros de cocina tradicional.

```
NOMBRE                                   APELLIDO           TITULO
---------------------------------------- ------------------ ------------------------------------------------------------
Panteley                                 Sylvia     Onions, Leeks, and Garlic: Cooking Secrets of the Mediterranean
Blotchet-Halls                           Reginald           Fifty Years in Buckingham Palace Kitchens
O'Leary                                  Michael            Sushi, Anyone?
Gringlesby                               Burt               Sushi, Anyone?
Yokomoto                                 Akiko              Sushi, Anyone?

(5 filas afectadas)
```

SOLUCION:

```
select a.au_lname as [NOMBRE]
,a.au_fname as [APELLIDO]
,t.title as [TITULO]
from authors as a
inner join titleauthor as ta
on
a.au_id=ta.au_id
inner join titles as t
on
ta.title_id=t.title_id
where t.type= 'trad_cook'
```

4) Seleccione nombre, apellido de los autores y el nombre de la editorial de todos aquellos escritores cuya ciudad sea la misma que la de la editorial. Pero en la consulta también se incluirán los demás autores de la tabla authors

NOMBRE	APELLIDO	EDITOR
White	Johnson	NULL
Green	Marjorie	NULL
Carson	Cheryl	Algodata Infosystems
O'Leary	Michael	NULL
Straight	Dean	NULL
Smith	Meander	NULL
Bennet	Abraham	Algodata Infosystems
Dull	Ann	NULL
Gringlesby	Burt	NULL
Locksley	Charlene	NULL
Greene	Morningstar	NULL
Blotchet-Halls	Reginald	NULL
Yokomoto	Akiko	NULL
del Castillo	Innes	NULL
DeFrance	Michel	NULL
Stringer	Dirk	NULL
MacFeather	Stearns	NULL
Karsen	Livia	NULL
Panteley	Sylvia	NULL
Hunter	Sheryl	NULL
McBadden	Heather	NULL
Ringer	Anne	NULL
Ringer	Albert	NULL

(23 filas afectadas)

SOLUCION:

```
select a.au_lname as [NOMBRE]
,a.au_fname as [APELLIDO]
,p.pub_name as [EDITOR]
from authors as a
left join publishers as p
on
a.city=p.city
```

5) Recuperar los títulos y el índice del almacén de todos los libros que vendieron más de 25 unidades.

```
ALMACEN ID TITULO
---------- --------------------------------------------------------------------------------
7066       Secrets of Silicon Valley
7066       Is Anger the Enemy?
7067       Onions, Leeks, and Garlic: Cooking Secrets of the Mediterranean
7896       You Can Combat Computer Stress!
8042       But Is It User Friendly?

(5 filas afectadas)
```

SOLUCION:

```
select s.stor_id as [ALMACEN ID]
,t.title as [TITULO]
from titles as t
inner join sales as s
on
t.title_id = s.title_id
where
s.qty>25
```

6) Modificación al ejercicio anterior: incluir también los títulos de aquellos libros que no superaron las 25 unidades en sus ventas

```
ALMACEN ID TITULO
---------- --------------------------------------------------------------------------------
8042       But Is It User Friendly?
NULL         Computer Phobic AND Non-Phobic Individuals: Behavior Variations
NULL         Cooking with Computers: Surreptitious Balance Sheets
NULL         Emotional Security: A New Algorithm
NULL         Fifty Years in Buckingham Palace Kitchens
7066       Is Anger the Enemy?
NULL         Life Without Fear
NULL         Net Etiquette
7067       Onions, Leeks, and Garlic: Cooking Secrets of the Mediterranean
NULL         Prolonged Data Deprivation: Four Case Studies
7066       Secrets of Silicon Valley
NULL         Silicon Valley Gastronomic Treats
NULL         Straight Talk About Computers
NULL         Sushi, Anyone?
NULL         The Busy Executive's Database Guide
NULL         The Gourmet Microwave
NULL         The Psychology of Computer Cooking
7896       You Can Combat Computer Stress!

(18 filas afectadas)
```

SOLUCION:

```
select s.stor_id as [ALMACEN ID]
,t.title as [TITULO]
from titles as t
left join sales   as s
on
t.title_id = s.title_id and s.qty>25
```

7) Realizar una consulta que devuelva el titulo, editorial y autor de cada libro.

TITULO	AUTOR	EDITOR
The Busy Executive's Database Guide	Green	Algodata Infosystems
The Busy Executive's Database Guide	Bennet	Algodata Infosystems
Cooking with Computers: Surreptitious Balance Sheets	O'Leary	Algodata Infosystems
Cooking with Computers: Surreptitious Balance Sheets	MacFeather	Algodata Infosystems
You Can Combat Computer Stress!	Green	New Moon Books
Straight Talk About Computers	Straight	Algodata Infosystems
Silicon Valley Gastronomic Treats	del Castillo	Binnet & Hardley
The Gourmet Microwave	DeFrance	Binnet & Hardley
The Gourmet Microwave	Ringer	Binnet & Hardley
But Is It User Friendly?	Carson	Algodata Infosystems
Secrets of Silicon Valley	Dull	Algodata Infosystems
Secrets of Silicon Valley	Hunter	Algodata Infosystems
Net Etiquette	Locksley	Algodata Infosystems
Computer Phobic AND Non-Phobic Individuals: Behavior Variations	MacFeather	Binnet & Hardley
Computer Phobic AND Non-Phobic Individuals: Behavior Variations	Karsen	Binnet & Hardley
Is Anger the Enemy?	Ringer	New Moon Books
Is Anger the Enemy?	Ringer	New Moon Books
Life Without Fear	Ringer	New Moon Books
Prolonged Data Deprivation: Four Case Studies	White	New Moon Books
Emotional Security: A New Algorithm	Locksley	New Moon Books
Onions, Leeks, and Garlic: Cooking Secrets of the Mediterranean	Panteley	Binnet & Hardley
Fifty Years in Buckingham Palace Kitchens	Blotchet-Halls	Binnet & Hardley
Sushi, Anyone?	O'Leary	Binnet & Hardley
Sushi, Anyone?	Gringlesby	Binnet & Hardley
Sushi, Anyone?	Yokomoto	Binnet & Hardley

(25 filas afectadas)

SOLUCION:

```
select t.title as [TITULO]
,a.au_lname as [AUTOR]
,p.pub_name as [EDITOR]
from publishers as p
inner join titles as t
on
t.pub_id = p.pub_id
inner join titleauthor as ta
on
ta.title_id = t.title_id
inner join authors as a
on
ta.au_id = a.au_id
```

CONSULTAS DE UNION INTERNAS

1. **Seleccionar el apellido, oficio, salario, numero de departamento y su nombre de todos los empleados cuyo salario sea mayor de 300000**

```
select e.apellido
,e.oficio
,e.salario
,d.dept_no as [N° DEPT]
,dnombre as [DEPARTAMENTO]
from emp as e
inner join dept as d on
e.dept_no = d.dept_no
where e.salario > 300000
```

	apellido	oficio	salario	N° DEPT	DEPARTAMENTO
1	CEREZO	DIRECTOR	318500	10	CONTABILIDAD
2	NEGRO	DIRECTOR	370500	30	VENTAS
3	JIMENEZ	DIRECTOR	386750	20	INVESTIGACION
4	GIL	ANALISTA	390000	20	INVESTIGACION
5	FERNANDEZ	ANALISTA	390000	20	INVESTIGACION
6	SERRA	DIRECTOR	390000	20	INVESTIGACION
7	REY	PRESIDENTE	650000	10	CONTABILIDAD

2. **Mostrar todos los nombres de Hospital con sus nombres de salas correspondientes.**

```
select s.nombre as [Nombre Sala]
,h.nombre as [Nombre Hospital]
from sala s inner join hospital h on
s.hospital_cod = h.hospital_cod
```

	Nombre Sala	Nombre Hospital
1	Cuidados Intensivos	Provincial
2	Psiquiátricos	Provincial
3	Cuidados Intensivos	General
4	Cardiología	General
5	Recuperación	La Paz
6	Psiquiátricos	La Paz
7	Maternidad	La Paz
8	Cardiología	San Carlos
9	Recuperación	San Carlos
10	Maternidad	San Carlos

3. Calcular cuantos trabajadores de la empresa hay en cada ciudad.

```
select count(e.emp_no) as [N° de trabajadores]
,d.loc as [Ciudad]
from emp as e
right outer join dept as d
on d.dept_no = e.dept_no
group by d.loc
```

	N° de trabajadores	Ciudad
1	6	BARCELONA
2	3	ELCHE
3	9	MADRID
4	0	SALAMANCA

4. Visualizar cuantas personas realizan cada oficio en cada departamento mostrando el nombre del departamento.

```
select d.dnombre as [N° Departamento]
,count(*) as [N° de personas]
,e.oficio
from emp e
right outer join dept d
on e.dept_no = d.dept_no
group by e.dept_no, e.oficio,d.dnombre
```

	N° Departamento	N° de personas	oficio
1	PRODUCCION	0	NULL
2	CONTABILIDAD	1	DIRECTOR
3	CONTABILIDAD	1	EMPLEADO
4	CONTABILIDAD	1	PRESIDENTE
5	INVESTIGACION	3	ANALISTA
6	INVESTIGACION	2	DIRECTOR
7	INVESTIGACION	3	EMPLEADO
8	INVESTIGACION	1	VENDEDOR
9	VENTAS	1	DIRECTOR
10	VENTAS	1	EMPLEADO
11	VENTAS	4	VENDEDOR

5. Contar cuantas salas hay en cada hospital, mostrando el nombre de las salas y el nombre del hospital.

```
select count(s.nombre) as [Numero de salas]
,s.nombre as [Sala]
,h.nombre as [Hospital]
from sala as s
inner join hospital as h
on h.hospital_cod = s.hospital_cod
group by s.nombre,h.nombre
```

	Numero de salas	Sala	Hospital
1	1	Cardiología	General
2	1	Cuidados Intensivos	General
3	1	Maternidad	La Paz
4	1	Psiquiátricos	La Paz
5	1	Recuperación	La Paz
6	1	Cuidados Intensivos	Provincial
7	1	Psiquiátricos	Provincial
8	1	Cardiología	San Carlos
9	1	Maternidad	San Carlos
10	1	Recuperación	San Carlos

6. **Calcular cuantos trabajadores hay en cada departamento(nombre de departamento)**

```
select count(e.emp_no) as [N° de trabajadores]
,d.dnombre as [Departamento]
from emp e
right outer join dept d
on d.dept_no = e.dept_no
group by d.dnombre
```

	N° de trabajadores	Departamento
1	3	CONTABILIDAD
2	9	INVESTIGACION
3	0	PRODUCCION
4	6	VENTAS

7. **Buscar aquellos departamentos con cuatro o mas personas trabajando.**

```
select d.dnombre as [Departamento]
,count(*) as [N° de personas]
from emp e
inner join dept d
on e.dept_no = d.dept_no
group by d.dept_no,d.dnombre
having count(*) >= 4
```

	Departamento	N° de personas
1	INVESTIGACION	9
2	VENTAS	6

8. **Calcular el valor medio de las camas que existen para cada nombre de sala. Indicar el nombre de cada sala y el codigo de cada una de ellas.**

```
select avg(num_cama) as [MEDIA CAMAS]
,nombre
,sala_cod
from sala
group by nombre,sala_cod
```

	MEDIA CAMAS	nombre	sala_cod
1	11	Recuperación	1
2	29	Maternidad	2
3	18	Cuidados Intensivos	3
4	54	Cardiología	4
5	62	Psiquiátricos	6

9. Calcular la media salarial por ciudad.

```
select d.loc as [Ciudad]
,avg(salario) as [Media Salarial]
from emp e inner join dept d
on d.dept_no = e.dept_no
group by d.loc
```

	Ciudad	Media Salarial
1	BARCELONA	206916
2	ELCHE	379166
3	MADRID	250861

10. Mostrar los doctores junto con el nombre de hospital en el que ejercen, la dirección y el teléfono del mismo.

```
select d.apellido
,h.nombre
,h.direccion
,h.telefono
from doctor d
inner join hospital_cod h
on h.hospital_cod = d.hospital_cod
```

	apellido	nombre	direccion	telefono
1	Cabeza D.	La Paz	Castellana 1000	923-5411
2	Best D.	La Paz	Castellana 1000	923-5411
3	López A.	Provincial	O' Donell 50	964-4256
4	Galo D.	La Paz	Castellana 1000	923-5411
5	Adams C.	San Carlos	Ciudad Univeritaria	597-1500
6	Miller G.	General	Atocha s/n	595-3111
7	Nino P.	San Carlos	Ciudad Univeritaria	597-1500
8	Cajal R.	General	Atocha s/n	595-3111

11. Mostrar los nombres de los hospitales junto con el mejor salario de los empleados de cada hospital.

```
select h.nombre as [Hospital]
,max(p.salario) as [SALARIO MAXIMO]
from plantilla p
inner join hospital_cod h
on p.hospital_cod = h.hospital_cod
group by h.nombre
```

	Hospital	SALARIO MAXIMO
1	General	337900
2	La Paz	221000
3	Nino	4343
4	Provincial	275000
5	Prueba	32432
6	San Carlos	252200

12. Visualizar el nombre de los empleados de la plantilla junto con el nombre de la sala, el nombre del hospital y el número de camas libres de cada una de ellas.

```
select p.apellido as [Apellido y Nombre]
,s.nombre as [Sala]
,h.nombre as [Hospital],
s.num_cama as [N° de camas]
from plantilla p inner join sala as s
on p.hospital_cod = s.hospital_cod
and p.sala_cod = s.sala_cod
inner join hospital as h
on h.hospital_cod = p.hospital_cod
```

	Apellido y Nombre	Sala	Hospital	N° de camas
1	Hernández J.	Psiquiátricos	Provincial	67
2	Díaz B.	Psiquiátricos	Provincial	67
3	Karplus W.	Cardiología	General	53
4	Rivera G.	Recuperación	La Paz	10
5	Carlos R.	Recuperación	La Paz	10
6	Higueras D.	Psiquiátricos	La Paz	118
7	Bocina G.	Psiquiátricos	La Paz	118
8	Núñez C.	Maternidad	La Paz	34
9	Amigo R.	Cardiología	San Carlos	55
10	Frank H.	Recuperación	San Carlos	15

13. Visualizar el máximo salario, mínimo salario de los empleados dependiendo de la ciudad en la que trabajen. Indicando el número total de trabajadores por ciudad.

```
select count(e.emp_no) as [N° de trabajadores]
,d.loc as [Ciudad]
,max(e.salario) as [Salario Máximo]
,min(e.salario) as [Salario Mínimo]
from emp e
inner join dept d
on e.dept_no = d.dept_no
group by d.loc
```

	N° de trabajadores	Ciudad	Salario Máximo	Salario Mínimo
1	6	BARCELONA	370500	123500
2	3	ELCHE	650000	169000
3	9	MADRID	390000	100000

14. Averiguar la combinación de que salas podría haber por cada uno de los hospitales.

```
select s.nombre as [Nombre sala]
,h.nombre as [Nombre Hospital]
from sala as s
cross join hospital as h
```

15. Mostrar el Numero de empleado, apellido, oficio y Nombre del departamento de los empleados, junto al Número de empleado, apellido, oficio y Nombre del departamento de sus subordinados respectivamente, para obtener una visión jerarquica de la empresa.

```
select a.emp_no as [Nº de Empleado]
,a.apellido as [Jefe], a.Oficio
,d.dnombre as [Departamento]
,b.emp_no as [Nº Empleado]
,b.apellido as [Subordinado]
,b.Oficio
,d.dnombre as [Departamento]
from emp as a
inner join emp as b
on b.dir = a.emp_no
inner join dept as d
on a.dept_no = d.dept_no
and b.dept_no = d.dept_no
order by b.dir
```

OPERADOR UNION

Es un operador que combina un conjunto de resultados, por ejemplo, una sentencia SELECT con OTRA

Saca las filas de los empleados y después saca la de Plantilla

APELLIDO	OFICIO/FUNCION	SALARIO
Amigo R.	Interino	221000
Angulo	NULL	NULL
ARROYO	VENDEDOR	208000
Bocina G.	Enfermero	183800
Carlos R.	Enfermera	211900
CEREZO	DIRECTOR	318500
COAL	EMPLEADO	NULL
COALESCE	EMPLEADO	NULL
Díaz B.	Enfermera	226200
FERNANDEZ	ANALISTA	390000
Frank H.	Enfermera	252200

```
Select Apellido,
Oficio as
'OFICIO/FUNCION'
,salario from emp
UNION
Select Apellido,
Funcion, Salario from
Plantilla
```

Recomendaciones a la hora de usar las combinaciones:

- Combinar tablas en funcion de claves principales y externas
- Limite el número de tablas de las combinaciones.

SUBCONSULTAS

Es una SELECT anidada en una instrucción INSERT, DELETE, SELECT o UPDATE.

- **Como una tabla derivada**

```
Select e.emp_no as Numero
,e.Apellido
from (select emp_no, apellido from emp) as e
```

Numero	Apellido
7369	SANCHEZ
7499	ARROYO
7521	SALA
7566	JIMENEZ
7654	MARTIN
7698	NEGRO
7782	CEREZO
7788	GIL
7839	REY
7844	TOVAR
7861	COALESCE

- **Como una expresión**

```
Select emp_no as [Numero]
,Apellido
,Salario
,(select avg(Salario) from emp)
as Diferencia
from emp
where oficio = 'Empleado'
```

Numero	Apellido	Salario	Diferencia
7369	SANCHEZ	104000	287473
7900	JIMENO	123500	287473
7934	MUÑOZ	169000	287473

- **Para correlacionar datos:**

- Expresión dinámica que cambia en cada fila de una consulta externa
- Es una combinación entre la subconsulta y la fila de la consulta externa.
- Dividen consultas complejas con dos o más consultas simples relacionadas

Subconsulta correlacionada

```
Select apellido, oficio,dept_no
from emp as e
where 20 <
(select dept_no from dept as d
where  e.dept_no  =  d.dept_no
and d.dnombre = 'Ventas')
```

apellido	oficio	dept_no
ARROYO	VENDEDOR	30
SALA	VENDEDOR	30
MARTIN	VENDEDOR	30
NEGRO	DIRECTOR	30
TOVAR	VENDEDOR	30
JIMENO	EMPLEADO	30

Simulacion de una clausula JOIN

Vamos a mostrar los oficios que están en más de un departamento.

```
Select distinct e1.oficio from emp as e1
where e1.oficio in(Select e2.oficio from emp as e2
where e1.dept_no <> e2.dept_no)
```

Se debe utilizar antes una combinación que una subconsulta, la combinación sería asi, dando los mismos resultados:

```
Select distinct e1.Oficio from emp as e1
inner join emp as e2
on e1.oficio = e2.oficio
where e1.dept_no <> e2.dept_no
```

Estos son los dos oficios que están en más de un departamento.

oficio
DIRECTOR
EMPLEADO

Subconsulta para simular una clausula HAVING

```
select e1.apellido,e1.oficio, e1.salario
from emp as e1
where e1.salario >
(select avg(e2.salario) from emp as e2
where e1.oficio = e2.oficio)
```

apellido	oficio	salario
CEREZO	DIRECTOR	318500
NEGRO	DIRECTOR	370500
JIMENEZ	DIRECTOR	386750
MUÑOZ	EMPLEADO	169000
TOVAR	VENDEDOR	195000
ARROYO	VENDEDOR	208000

Esta es la consulta utilizando el HAVING, que es la que deberíamos utilizar antes que una subconsulta de simulación HAVING:

```
SELECT e1.apellido,e1.oficio, e1.salario
FROM emp AS e1
INNER JOIN emp AS e2
ON e1.oficio = e2.oficio
GROUP BY e1.oficio, e1.salario,e1.apellido
HAVING e1.salario > AVG (e2.salario)
```

Clausulas EXISTS Y NOT EXISTS

Comprueba si el dato que buscamos existe en la consulta.

Mostrar los empleados que no tienen departamento:

```
select apellido,fecha_alt,salario
from emp as e
where not exists(select * from dept as o
where e.dept_no = o.dept_no)
```

apellido	fecha_alt	salario
SERRA	1999-12-11 00:00:00	12345

Recomendaciones:

- Utilizar subconsultas para dividir una consulta compleja.
- Utilizar Alias para tablas en subconsultas correlacionadas y combinaciones.
- Utilizar EXISTS en vez de IN

SUBCONSULTAS

1. Mostrar el numero de empleado, el apellido y la fecha de alta del empleado mas antiguo de la empresa

```
SELECT emp_no,apellido,
fecha_alt from emp where fecha_alt = (select min(fecha_alt) from emp)
```

	emp_no	apellido	fecha_alt
1	7369	SANCHEZ	1980-12-17 00:00:00

2. Mostrar el numero de empleado, el apellido y la fecha de alta del empleado mas modernos de la empresa.

```
SELECT emp_no,apellido,
fecha_alt from emp where fecha_alt = (select max(fecha_alt) from emp)
```

	emp_no	apellido	fecha_alt
1	7876	ALONSO	1987-05-03 00:00:00

3. Visualizar el apellido y el oficio de los empleados con el mismo oficio que Jiménez.

```
select apellido, oficio from emp where
oficio = (select oficio from emp where apellido = 'JIMENEZ')
```

	apellido	oficio
1	JIMENEZ	DIRECTOR
2	NEGRO	DIRECTOR
3	CEREZO	DIRECTOR
4	SERRA	DIRECTOR

4. Queremos saber el apellido, oficio, salario y número de departamento de los empleados con salario mayor que el mejor salario del departamento 30.

```
Select apellido, oficio, salario, dept_no from emp where salario > (select
max (salario) from emp where dept_no = 30)
```

	apellido	oficio	salario	dept_no
1	JIMENEZ	DIRECTOR	386750	20
2	GIL	ANALISTA	390000	20
3	REY	PRESIDENTE	650000	10
4	FERNANDEZ	ANALISTA	390000	20
5	SERRA	DIRECTOR	390000	20

5. Mostrar el apellido, la función, sala o departamento de todos los empleados que trabajen en la empresa.

```
select e.Apellido, e.Oficio as [Cargo en Empresa], d.dnombre as [Oficina]
from emp as e
inner join dept as d
on e.dept_no = d.dept_no
union
select p.Apellido, p.funcion, s.nombre
from plantilla as p
inner join sala as s
on p.hospital_cod = s.hospital_cod
and p.sala_cod = s.sala_cod
union
select d.Apellido, d.especialidad, h.nombre
from doctor as d
inner join hospital as h
on d.hospital_cod = h.hospital_cod
order by 3
```

6. Averiguar el salario de todos los empleados de la empresa, de forma que se aprecien las diferencias entre ellos.

```
select Apellido, Salario from emp
union
select Apellido, Salario from plantilla
order by 2 desc
```

7. Mostrar apellidos y oficio de los empleados del departamento 20 cuyo trabajo sea el mismo que el de cualquier empleado de ventas.

```
Select apellido, oficio from emp where dept_no = 20 and oficio in
 (select oficio from emp where dept_no =
(select dept_no from dept where dnombre = 'ventas'))
order by 2
```

	apellido	oficio
1	JIMENEZ	DIRECTOR
2	SERRA	DIRECTOR
3	ALONSO	EMPLEADO
4	SANCHEZ	EMPLEADO
5	TRICO	EMPLEADO
6	GARCIA	VENDEDOR

8. Mostrar los empleados que tienen mejor salario que la media de los directores, no incluyendo al presidente.

```
Select * from emp where salario >
(select avg (salario) from emp where oficio = 'DIRECTOR')
and oficio <> 'PRESIDENTE'
```

	EMP_NO	APELLIDO	OFICIO	DIR	FECHA_ALT	SALARIO	COMISION	DEPT_NO
1	7566	JIMENEZ	DIRECTOR	7839	1981-04-02 00:00:00	386750	0	20
2	7698	NEGRO	DIRECTOR	7839	1981-05-01 00:00:00	370500	0	30
3	7788	GIL	ANALISTA	7566	1987-03-30 00:00:00	390000	0	20
4	7902	FERNANDEZ	ANALISTA	7566	1981-12-03 00:00:00	390000	0	20
5	7919	SERRA	DIRECTOR	7839	1981-10-10 00:00:00	390000	21000	20

9. Mostrar el apellido, función, salario y código de hospital de los empleados de la plantilla que siendo enfermeros o enfermeras pertenecen al hospital SAN CARLOS.

```
Select apellido, funcion, salario, hospital_cod from plantilla where (funcion
= 'ENFERMERO' or funcion = 'ENFERMERA') and hospital_cod = (select
hospital_cod from hospital where nombre = 'SAN CARLOS')
```

```
Select apellido, funcion, salario, hospital_cod from plantilla where
funcion in ('ENFERMERO','ENFERMERA')
and hospital_cod = (select hospital_cod from hospital
where nombre = 'SAN CARLOS')
```

	apellido	funcion	salario	hospital_cod
1	Frank H.	Enfermera	252200	45

	apellido	funcion	salario	hospital_cod
1	Frank H.	Enfermera	252200	45

10. Visualizar los datos de los hospitales que tienen personal (Doctores) de cardiología.

```
Select * from hospital where hospital_cod in (select hospital_cod from
doctor where especialidad = 'Cardiología')
```

	HOSPITAL_COD	NOMBRE	DIRECCION	TELEFONO	NUM_CAMA
1	19	Provincial	O' Donell 50	964-4256	502
2	18	General	Atocha s/n	595-3111	987

11. Visualizar el salario anual de los empleados de la plantilla del Hospital Provincial y General.

```
Select apellido, funcion, salario * 12 as [SALARIO ANUAL] from plantilla
where hospital_cod in (select hospital_cod from hospital where nombre =
'PROVINCIAL' or nombre= 'GENERAL')
```

	apellido	funcion	SALARIO ANUAL
1	Hernández J.	Enfermero	3300000
2	Díaz B.	Enfermera	2714400
3	Karplus W.	Interino	4054800

12. Mostrar el apellido de los enfermos que nacieron antes que el Señor Miller.

```
Select apellido from enfermo where Fecha_nac < (select fecha_nac from
enfermo where apellido = 'MILLER B.')
```

	apellido
1	Domin S.
2	Neal R.

Variables

- Definidas en una instrucción DECLARE (Con una sola @)
- Valores asignados con instrucción SET o SELECT (Con una sola @)
- Ámbito global o local

Sintaxis: DECLARE @VariableLocal tipodatos,...
SET @VariableLocal = expresión

SELECT @VariableLocal = Nombre from Empleados WHERE Id = 5

BUCLES

Nivel de instrucción:

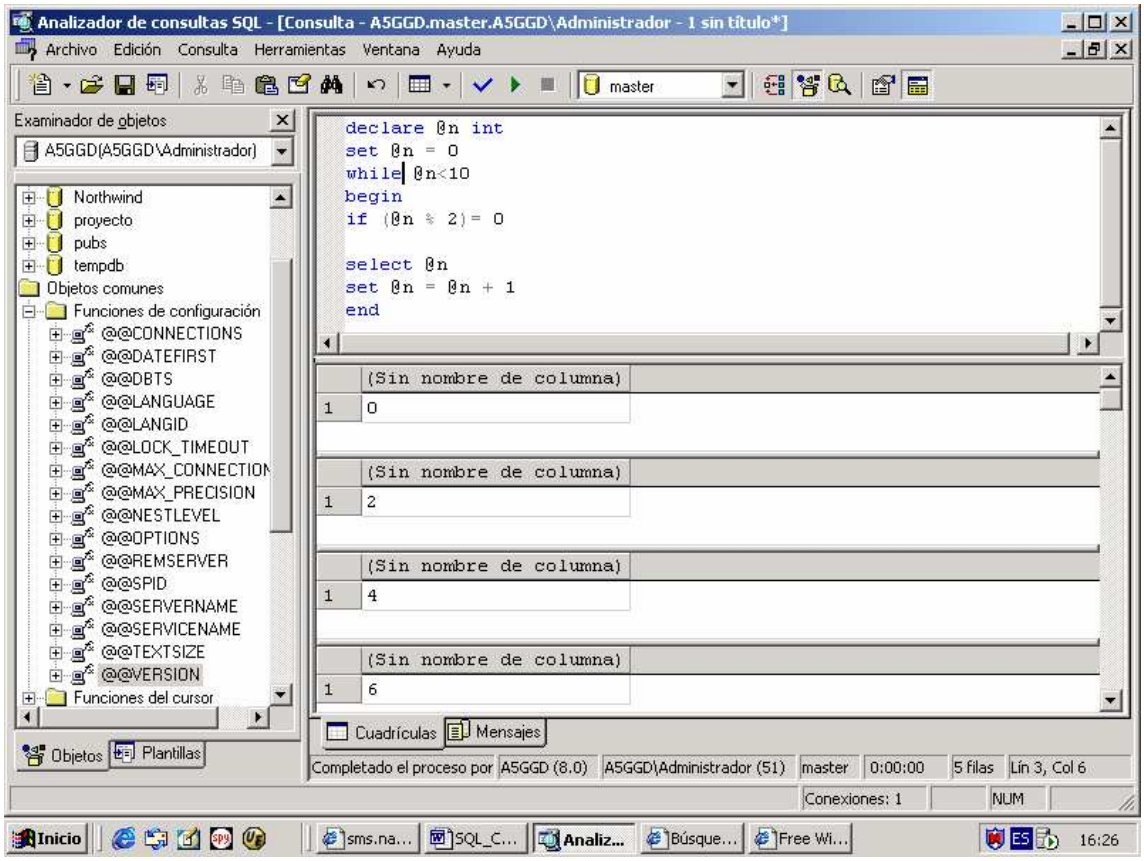
- Bloques BEGIN.....END
- Bloques IF.....ELSE
- Construcciones WHILE

Sacar Números pares con IF

```
--IF
declare @n int
set @n = 5
if (@n % 2)= 0
print 'PAR'
else
print 'IMPAR'

--BUCLES
Declare @n int
set @n = 0
while @n < 10
begin
if (@n % 2) = 0
select @n as Numero
set @n = @n + 1
end
```

EJEMPLO: Sacar los numeros pares hasta 10 dentro de un bucle en SQL



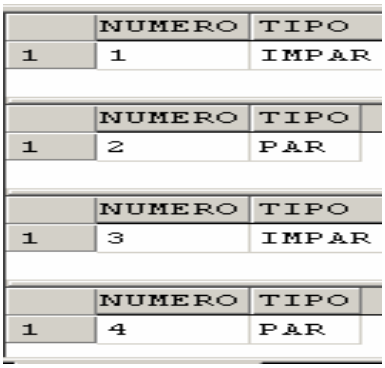
Nivel De fila

Las expresiones a nivel de fila evaluan un resultado devuelto por la consulta y dependiendo de los valores que utilicemos lo sustituyen para mejorar la presentación de los datos.

```
CASE expresion
        WHEN valor1 THEN resultado1
        ELSE resultadoN
END

CASE
        WHEN verdadero THEN resultado1
        ELSE resultado2
END
```

```
--CASE
DECLARE @N INT
SET @N = 1
      WHILE (@N<100)
      BEGIN
            SELECT @N AS 'NUMERO',
            CASE WHEN (@N % 2) = 1
            THEN
            'IMPAR'
            ELSE
            'PAR'
            END AS 'TIPO'
           SET @N = @N +
           1
END
```



	NUMERO	TIPO
1	1	IMPAR

	NUMERO	TIPO
1	2	PAR

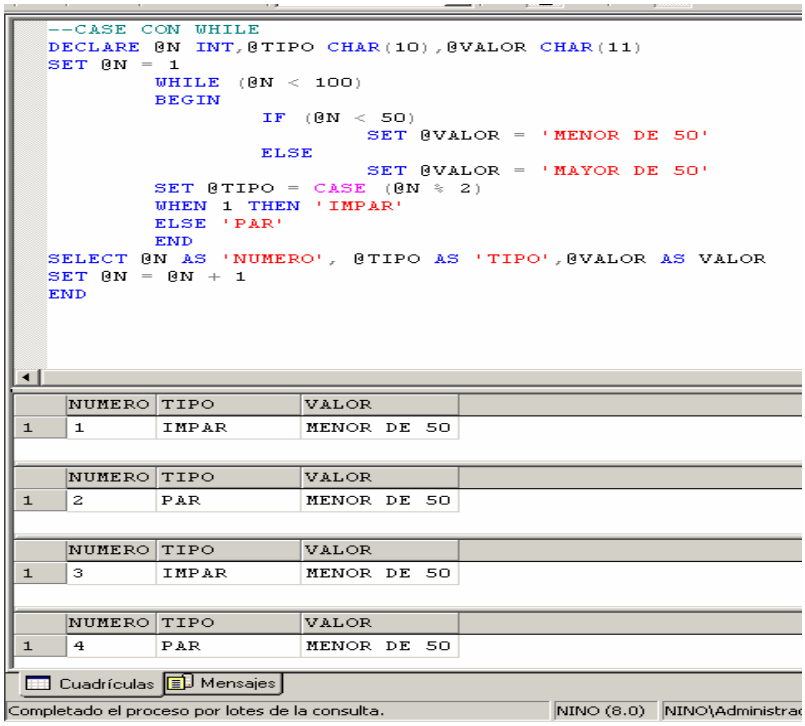
	NUMERO	TIPO
1	3	IMPAR

	NUMERO	TIPO
1	4	PAR

EJEMPLO:

Esto se puede utilizar también para inicializar variables. Es el mismo caso pero utilizando el CASE para inicializar una variable.

```
DECLARE @N INT,@TIPO CHAR(10),@VALOR CHAR(11)
SET @N = 1
      WHILE (@N < 100)
      BEGIN
            IF (@N < 50)
                  SET @VALOR = 'MENOR DE 50'
            ELSE
                  SET @VALOR = 'MAYOR DE 50'
      SET @TIPO = CASE (@N % 2)
      WHEN 1 THEN 'IMPAR'
      ELSE 'PAR'
      END
SELECT @N AS 'NUMERO', @TIPO AS 'TIPO',@VALOR AS VALOR
SET @N = @N + 1
END
```



CONVERT se usa para convertir el numero a una cadena de dos caracteres y lo concatenamos con @tipo.
CAST realiza lo mismo que CONVERT

```
DECLARE @N INT,@TIPO CHAR(10)
SET @N = 5
IF (@N BETWEEN 4 AND 6)
BEGIN
      WHILE (@N > 0)
      BEGIN
      SET @TIPO = CASE (@N % 2)
      WHEN 1 THEN 'IMPAR'
      ELSE 'PAR'
      END
SELECT @N AS 'NUMERO', @TIPO AS 'TIPO'
PRINT CONVERT (CHAR(2),@N) + @TIPO
PRINT CAST (@N AS CHAR(2)) + @TIPO
SET @N = @N - 1
END
END
```

Ponemos Cast y Convert para poder concatenar un valor de tipo Int con un valor de tipo Char, sino intentaría sumarlos y daría error.

EJERCICIOS CON SENTENCIAS

1. Queremos saber a qué empleados eliminariamos si quitasemos los departamentos 10 y 30 y cúales se mantendrían. Mostrar un informe con el apellido, salario, oficio y fechas de alta en la empresa.

```
select apellido,salario,dept_no,oficio,'Accion' = case
when dept_no <> 10 then 'Empleado de baja'
else
'Se Mantiene'
end
from emp where oficio = 'empleado'
```

2. Debemos hacer recortes de salario en la empresa, para ello debemos saber a que personas recortaremos el sueldo, cuales se mantendrán y cuales subiremos el puesto. Utilizar todos los empleados de la empresa(Plantilla y Empleados) Cuando el salario sea menor de 100000, Subiremos sueldo, cuando esté entre 100000 y 250000 lo mantendremos y cuando sea superior, lo bajaremos.

```
select Apellido, Salario, 'Accion' = case
when salario < 100000 then 'Subir sueldo'
when salario between 150000 and 250000 then 'Mantener sueldo'
else
 'Bajar sueldo'
end
from emp
union
select apellido,salario, 'Accion' = case
when salario < 100000 then 'Subir sueldo'
when salario between 150000 and 250000 then 'Mantener sueldo'
else
 'Bajar sueldo'
end
from plantilla
```

3. Queremos saber que empleados de la plantilla trabajan en turno de tarde, noche o en otros, para ello mostraremos ‘Tarde’ o ‘Noche’ dependiendo de sus valores.

```
select 'Número empleado' = empleado_no, 'Apellido' = apellido,
        'Turno' = case t
        when 'T' then 'Tarde'
        when 'm' then 'Mañana'
        else 'Otros'
        end
from plantilla
```

4. Queremos cambiar de localidad en Barcelona, para ello tenemos que saber qué empleados cambiarian de localidad y cúales no. Combinar tablas y mostrar el nombre del departamento junto a los datos del empleado.

```
select d.dnombre as [Departamento]
,e.apellido,'Cambiar de Localidad' = case
when d.loc = 'SEVILLA' THEN 'CAMBIA DE LOCALIDAD'
else
'NO CAMBIA DE LOCALIDAD'
end
from emp as e inner join dept as d
on e.dept_no = d.dept_no
```

5. Queremos saber el número de trabajadores que cambiarían de localidad si cambiasemos a Barcelona y que número de trabajadores no cambiarian de localidad.

```
select count(e.emp_no) as [Nº de trabajadores],
d.loc as [Ciudad],'Cambiar' = case
when d.loc = 'BARCELONA' THEN 'CAMBIA DE LOCALIDAD'
else 'no cambia'
end
from emp as e inner join dept as d
on d.dept_no = e.dept_no
group by d.loc
```

6. Mostrar el apellido, la dirección, la fecha de nacimiento mostrando la década en la que está cada persona y el sexo mostrando si es masculino o femenino de la tabla enfermo.

```
SELECT APELLIDO, DIRECCION,
[FECHA NACIMIENTO] = CASE
WHEN FECHA_NAC BETWEEN '01/01/1940' AND '01/01/1950' THEN
'DECADA DE LOS 40'
WHEN FECHA_NAC BETWEEN '01/01/1950' AND '01/01/1960' THEN
'DECADA DE LOS 50'
WHEN FECHA_NAC BETWEEN '01/01/1960' AND '01/01/1970' THEN
'DECADA DE LOS 60'
WHEN FECHA_NAC BETWEEN '01/01/1970' AND '01/01/1980' THEN
'DECADA DE LOS 70'
WHEN FECHA_NAC BETWEEN '01/01/1980' AND '01/01/1990' THEN
'DECADA DE LOS 80'
ELSE
'DEMASIADO VIEJO O DEMASIADO JOVEN'
END
,SEXO = CASE S
WHEN 'F' THEN 'MUJER'
```

```
ELSE
'HOMBRE'
END
FROM ENFERMO
ORDER BY FECHA_NAC,S
```

7. Mostrar el apellido, el salario, el oficio y el nombre del departamento de todos los empleados aunque no tengan departamento. Si no tienen departamento mostraré que no tienen departamento. Mostraré además si tienen comisión o si no tienen comisión.

```
select e.apellido,e.oficio,e.salario
,DEPARTAMENTO =
ISNULL(d.dnombre,'SIN DEPARTAMENTO')
,Comision = case Comision
when 0 then 'SIN COMISION'
else
'CON COMISION'
end
from emp e
LEFT JOIN DEPT d
ON E.DEPT_NO = D.DEPT_NO
order by d.dnombre
```

8. Mostrar todas las camas que existen para cada hospital y cada sala. Mostraré el nombre del hospital, las salas y su número de camas. Si no hubiese camas para algún hospital las dejaré a 0. También mostraré que son muchas camas cuando sean más de 90, buen número cuando sean mayores de 40 y pocas camas para las demás.

```
select h.nombre as [HOSPITAL]
,isnull(s.num_cama,0) as [Nº DE CAMAS]
,s.nombre as [SALAS]
,CAMAS = CASE
when s.num_cama > 90 then 'DEMASIADAS CAMAS'
when s.num_cama between 40 and 89 then 'BUEN NUMERO'
else
'POCAS CAMAS'
end
from sala as s
full join hospital as h
on h.hospital_cod = s.hospital_cod
group by h.nombre,s.num_cama,s.nombre
order by h.nombre,s.num_cama
```

9. Seleccionar qué empleados están dentro de la media y cuales están por debajo de la media, mostrando el apellido, oficio, salario, comisión y el nombre de los departamentos. No dejar ningún campo a NULL.

```
declare @media int
select @media = avg(salario) from emp
select e.apellido,e.oficio,e.salario,e.comision
,MEDIA = case
when salario > @media then 'DENTRO DE LA MEDIA'
else
'POR DEBAJO DE LA MEDIA'
end
,isnull(d.dnombre,'SIN DEPARTAMENTO')
from emp as e
left join dept d
on e.dept_no = d.dept_no
group by e.apellido,e.oficio,e.salario,e.comision,d.dnombre
order by Media
```

INSERCIÓN, ELIMINACIÓN Y MODIFICACIÓN DE DATOS

- **Inserción de una fila mediante valores:**

  INSERT INTO {NombreTabla | NombreVista} [Valor de la Columna]
  VALUES Valores

* Cuando hay llaves es porque se debe elegir entre uno de los dos, esta barra | indica que se debe poner uno de los dos valores.

- **Uso INSERT...SELECT:**

  INSERT NombreTabla SELECT ListaColumnas FROM ListaTablas
  WHERE CondicionBusqueda

  Se introducen en la tabla las columnas y filas que devuelva con sus respectivos datos. La consulta SELECT debe devolver los datos adecuados para la tabla donde vamos a introducir los valores.

- **Creación de una tabla mediante SELECT INTO:** Creación de una tabla que a la vez se le introducen valores.

  SELECT ListaColumnas INTO NuevaTabla FROM TablaOrigen
  WHERE CondicionBusqueda

  ```
  select apellido,salario,dept_no into #Temporal
  from emp
  where dept_no = 60
  ```

  Se utiliza mucho para crear tablas temporales

- **Inserción de datos parciales**: No introducir todos los datos, solo meter datos en un determinado campo o en varios, pero no en toda la tabla.

- **Inserción de datos mediante valores de columna predeterminados: Se usa para no dejar a las tablas con el valor null y así no da error.**

  Se utilizan dos clausulas:

  - DEFAULT: Especificar que cogiera en la lista de valores el valor por defecto de esa columna
  - DEFAULT VALUES: Crea una nueva fila con los valores por defecto de todas las columnas

  ```
  USE Hospital
  INSERT INTO emp  (Apellido,Salario)
  VALUES ('SERRA', DEFAULT)
  ```

Con esta sentencia se pone el valor predeterminado que tenga la tabla, si no tiene valor por defecto, pondrá null, lo que equivale a no poner el dato. Los valores por defecto se verán más adelante.

ELIMINACIÓN DE DATOS

- DELETE: Elimina una o varias filas. Hay un control de las modificaciones (Borrado) que se estan haciendo.

```
DELETE [FROM (Opcional) ] {NombreTabla | NombreVista }
WHERE CondicionBusqueda
Delete from emp where apellido = 'SERRA'
```

- TRUNCATE TABLE: Elimina todas las filas de la tabla (La tabla con su estructura no se elimina, sólo los datos de la tabla). No crea filas en el registro de transacciones, con lo cual es el método más rápido de borrar.

```
TRUNCATE TABLE NombreTabla
Truncate Table emp
```

- Eliminación de filas basada en otras tablas

```
DELETE [ FROM ] {NombreTabla | NombreVista}
[ FROM, OrigenTabla,... ]  [ WHERE CondicionBusqueda ]
```

Borra los campos de emp donde tienen relacion con informática

```
delete from emp
from emp as e
inner join departamento as d
on e.dept_no = d.dept_no
where d.dnombre = 'INFORMATICA'
```

ACTUALIZACIONES

- Actualización de filas basadas en datos de la propia tabla

```
UPDATE {NombreTabla | NombreVista }
SET NombreColumna = expresión { DEFAULT | NULL, ... }
```

```
USE Northwind
UPDATE products
SET unitprice = (unitprice * 1.1 )
```

- Actualización de filas basadas en otras tablas

```
UPDATE {NombreTabla | NombreVista }
SET NombreColumna = expresión { DEFAULT | NULL, ... }
FROM OrigenTabla
WHERE CondicionBusqueda
```

Cambiar el salario de los empleados del dept 30 donde el departamento sea 60.

```
update emp set salario = 130000 from emp
inner join dept
on emp.dept_no = dept.dept_no
where dept.dept_no = 60
```

CONSULTAS DE ACCION

1. Dar de alta con fecha actual al empleado Jose Escriche Barrera como programador perteneciente al departamento de informatica.Tendra un salario base de 70000 pts/mes y no cobrara comision, ¿qué dificultad plantea el alta de este empleado? ¿Cómo podria solucionarse ?

   ```
   Insert into dept(dept_no,dnombre,loc)
   values(60,'INFORMATICA','MADRID')

   Insert into emp(apellido,oficio,fecha_alt,salario,comision,dept_no)
   values('Escriche','PROGRAMADOR','07/02/02',70000,0,60)
   ```

2. Se quiere dar de alta un departamento de informática situado en Fuenlabrada (Madrid).

   ```
   insert into dept(dept_no,dnombre,loc)
   values(70,'INFORMATICA','FUENLABRADA')
   ```

3. El departamento de ventas por motivos de peseteros se traslada a Lerida , realizar dicha modificación.

   ```
   update dept set loc='LERIDA' where DNOMBRE='VENTAS'
   ```

4. En el departamento anterior se dan de alta dos empleados: Julián Romeral y Luis Alonso. Su salario base es de 80000 pts y cobrarán una comisión del 15% de su salario.

   ```
   insert into emp(dept_no,apellido,salario,comision,emp_no)
   values(30, 'Romeral',80000,80000*0.15,7500)

   insert into emp(dept_no,apellido,salario,comision,emp_no)
   values(30, 'Alonso',80000,80000*0.15,7600)
   ```

5. Modificar la comisión de los empleados de la empresa, de forma que todos tengan un incremento del 10% del salario.

   ```
   update emp set salario=salario*1.1
   ```

6. Incrementar un 5% el salario de los interinos de la plantilla que trabajen en el turno de noche.

   ```
   update plantilla set salario = salario*1.05 where funcion='Interino'
   and  T='N'
   ```

7. Incrementar en 5000 pts el salario de los empleados del departamento de ventas y del presidente, tomando en cuenta los que se dieron de alta antes que el presidente de la empresa.

```
update emp set salario = salario + 5000
from emp as e
inner join dept as d
on e.dept_no = d.dept_no
WHERE FECHA_ALT <  (select fecha_alt from emp where oficio =
'PRESIDENTE')
AND e.OFICIO = 'PRESIDENTE'
OR d.dnombre = 'VENTAS'
```

8. Se tienen que desplazar cien camas del Hospital SAN CARLOS para un Hospital de Venezuela. Actualizar el número de camas del Hospital SAN CARLOS.

```
update Hospital  set num_cama = num_cama-100 where nombre='San
Carlos'
```

9. Crear una tabla llamada Mujeres e insertar los enfermos con este sexo.

```
insert into Mujeres select * from enfermo where S = 'F'
```

10. Crear una tabla llamada Empleados e introducir todos los datos de la tabla EMP en ella.

```
INSERT INTO empleados select * from emp
```

11. Utilizar la tabla anterior. Subir el salario y la comisión en un millón de pesetas y doscientas veinticinco mil pesetas respectivamente a los empleados que se dieron de alta en este año.

```
update empleados set salario = salario + 1000000/12,
comision = comision + 225000/12 where fecha_alt > '01/01/02'
```

12. Borrar de la tabla mujer al enfermo con número de inscripción igual a 64823.

```
delete from Mujeres where inscripcion= '64823'
```

13. Borrar todos los registros de la tabla Mujeres de la forma más rápida.

```
truncate table mujeres
```

14. Utilizar la tabla Empleados. Borrar todos los empleados dados de alta entre las fechas 01/01/80 y 31/12/82.

```
delete from empleados where fecha_alt between '01/01/80' and '31/12/82'
```

15. Modificar el salario de los empleados trabajen en la paz y esten destinados a Psiquiatría. Subirles el sueldo 20000 ptas más que al señor Amigo R.

```
update plantilla set salario =
(select salario + 20000 from plantilla
where apellido = 'Amigo R.')
from plantilla as p
inner join hospital as h
on h.hospital_cod = p.hospital_cod
inner join sala as s
on s.sala_cod = p.sala_cod
where h.nombre = 'La Paz'
and s.nombre = 'Psiquiátricos'
```

16. Borrar los empleados cuyo nombre de departamento sea producción.

```
delete from emp
from emp as e
inner join departamento as d
on e.dept_no = d.dept_no
where d.dnombre = 'PRODUCCION'
```

MÓDULO 4: PROCEDIMIENTOS PARA AGRUPAR Y RESUMIR DATOS

En el examen de certificación las bases de datos que se suelen usar son las que vienen de ejemplo en SQL, es decir Northwind y Pubs
Use Base de datos Indica que la siguiente sentencia usará la base de datos indicada.
Ejemplo: Ponemos en el analizador de consultas lo siguiente:

```
Use Hospital
Select * from Hospital
```

[Order Details]

Esta tabla perteneciente a la Northwind, se encarga de manejar los pedidos

Order Id : Nº de Pedido.
Producto Id : Nº de Producto.

Ambos campos son Primary Key, con lo que no puede haber una combinación de ambos campos que sea igual.

OrderId	ProductId
1	A
1	B
1	C
2	A
3	C

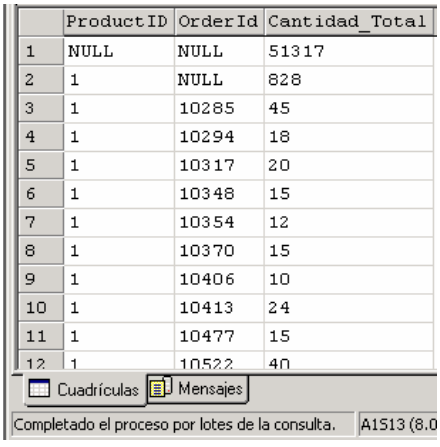
Es decir en este caso no podría existir una nueva combinación “1 A” o “2 A”.
Quantity: Es la cantidad del producto del pedido.

ROLLUP

Se usa para presentar resúmenes de datos. A de usarse junto con la clausula group by, lo que hace es realizar un resumen de los campos incluidos en el rollup.

```
Select ProductID, OrderId,
Sum(Quantity) As Cantidad_Total
From [Order Details]
Group by ProductID, OrderID
with Rollup
Order by ProductID, OrderID
```

Este ejemplo suma todas las cantidades, y mediante rollup, muestra una fila con la suma de todas las cantidades de cada producto, y además, otra fila con la suma de todas las cantidades de todos los productos. El resultado de este ejemplo, sería el que muestra la imagen:

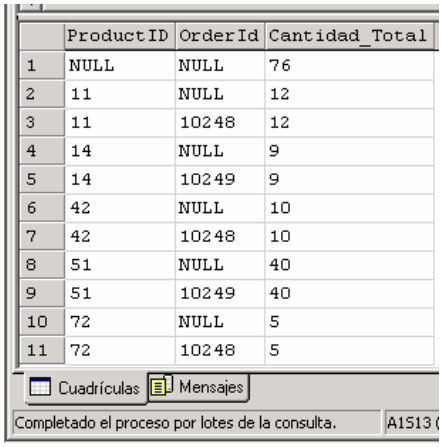


	ProductID	OrderId	Cantidad_Total
1	NULL	NULL	51317
2	1	NULL	828
3	1	10285	45
4	1	10294	18
5	1	10317	20
6	1	10348	15
7	1	10354	12
8	1	10370	15
9	1	10406	10
10	1	10413	24
11	1	10477	15
12	1	10522	40

Cuadrículas | Mensajes

Completado el proceso por lotes de la consulta. A1513 (8.0

```
Select ProductID, OrderId,
Sum(Quantity) As Cantidad_Total
From [Order Details]
where orderid < 10250
Group by ProductID, OrderID
with Rollup
Order by ProductID, OrderID
```



	ProductID	OrderId	Cantidad_Total
1	NULL	NULL	76
2	11	NULL	12
3	11	10248	12
4	14	NULL	9
5	14	10249	9
6	42	NULL	10
7	42	10248	10
8	51	NULL	40
9	51	10249	40
10	72	NULL	5
11	72	10248	5

Cuadrículas Mensajes

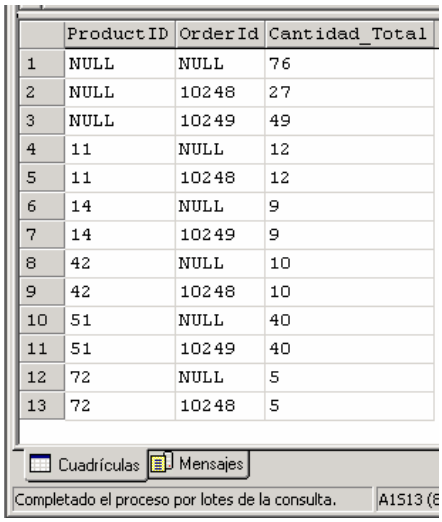
Completado el proceso por lotes de la consulta. A1513 (

CUBE

Al igual que Rollup realiza resúmenes de campos agrupados. Pero en este caso muestra un resumen con cada combinación posible de los campos agrupados.

```
Select ProductID, OrderId,
Sum(Quantity) As Cantidad_Total
From [Order Details]
where orderid < 10250
Group by ProductID, OrderID
with Cube
Order by ProductID, OrderID
```

En este caso como vemos en la imagen, hace un resumen con la suma de la cantidad de cada combinación posible entre el productid y el orderid



	ProductID	OrderId	Cantidad_Total
1	NULL	NULL	76
2	NULL	10248	27
3	NULL	10249	49
4	11	NULL	12
5	11	10248	12
6	14	NULL	9
7	14	10249	9
8	42	NULL	10
9	42	10248	10
10	51	NULL	40
11	51	10249	40
12	72	NULL	5
13	72	10248	5

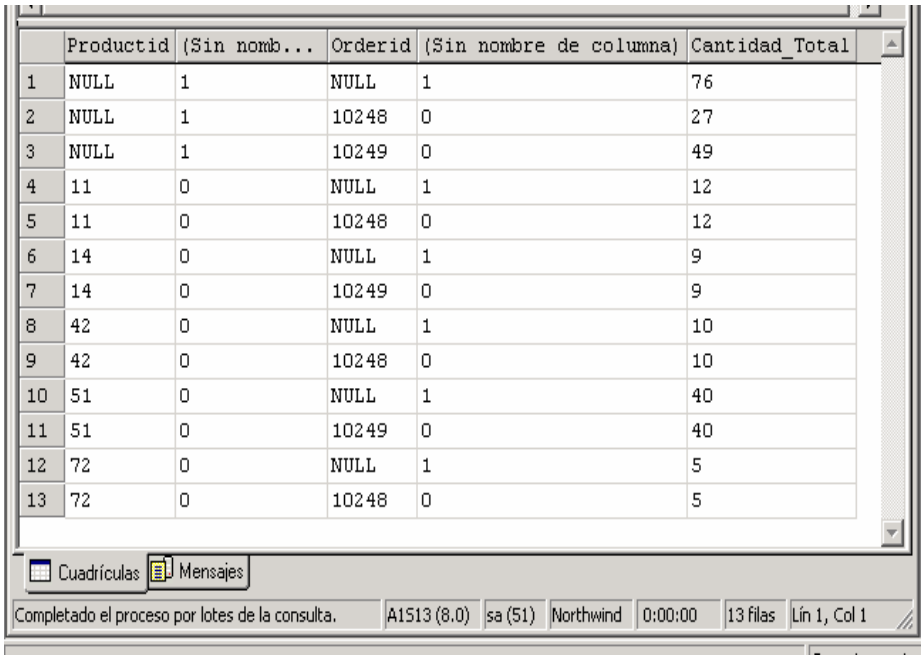
Cuadrículas Mensajes

Completado el proceso por lotes de la consulta. A1513 (8

GROUPING

Indica si el resultado de un campo es el que hay en la propia tabla o se ha introducido mediante una cláusula de resumen, es decir, para saber por ejemplo si un “Null” de una celda es de la propia tabla o ha es debido a una clausula Cube o Rollup.

```
Select Productid, grouping(ProductID), Orderid, grouping(OrderId),
Sum(Quantity) As Cantidad_Total
From [Order Details]
where orderid < 10250
Group by ProductID, OrderID
with Cube
Order by ProductID, OrderID
```



	Productid	(Sin nomb...	Orderid	(Sin nombre de columna)	Cantidad_Total
1	NULL	1	NULL	1	76
2	NULL	1	10248	0	27
3	NULL	1	10249	0	49
4	11	0	NULL	1	12
5	11	0	10248	0	12
6	14	0	NULL	1	9
7	14	0	10249	0	9
8	42	0	NULL	1	10
9	42	0	10248	0	10
10	51	0	NULL	1	40
11	51	0	10249	0	40
12	72	0	NULL	1	5
13	72	0	10248	0	5

Cuadrículas Mensajes

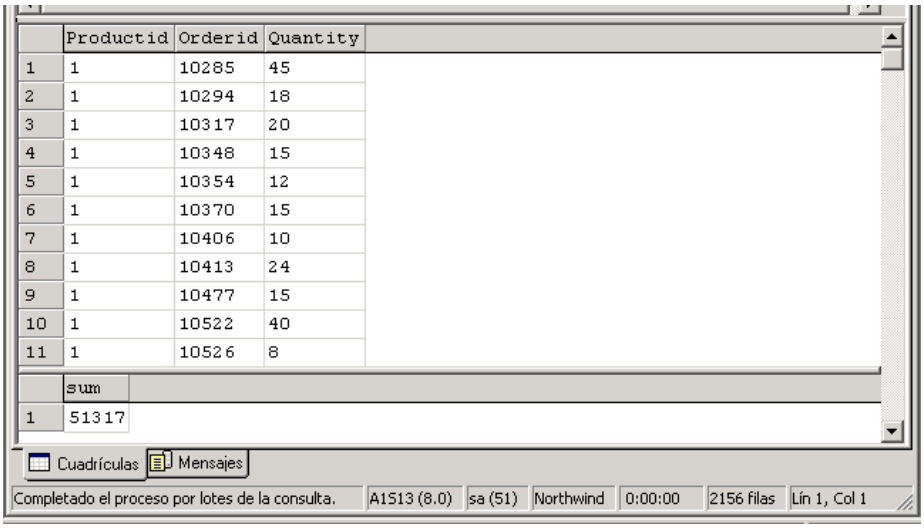
Completado el proceso por lotes de la consulta. | A1513 (8.0) | sa (51) | Northwind | 0:00:00 | 13 filas | Lín 1, Col 1

Vemos que por cada grouping que hemos puesto, sale una columna, 1 indica que el Null es creado por la clausula Cube y 0 indica que es de la tabla. Poniéndole un alias al grouping saldría el nombre de columna que le indiquemos.

COMPUTE

Realiza un resumen en una columna aparte con el resultado de la función de agregado indicada. Su formato sería Compute función(campo). No se puede utilizar en aplicaciones cliente / servidor, es una clausula meramente informativa. Tampoco se le pueden poner alias a los resúmenes.

```
Select Productid, Orderid, Quantity
From [Order Details]
order by ProductID, Orderid
compute Sum(quantity)
```



	Productid	Orderid	Quantity
1	1	10285	45
2	1	10294	18
3	1	10317	20
4	1	10348	15
5	1	10354	12
6	1	10370	15
7	1	10406	10
8	1	10413	24
9	1	10477	15
10	1	10522	40
11	1	10526	8

	sum
1	51317

Cuadrículas Mensajes

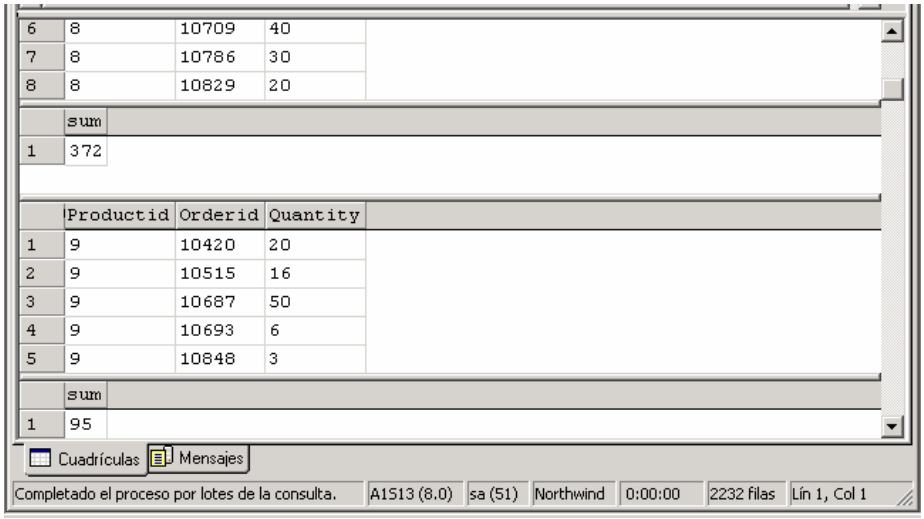
Completado el proceso por lotes de la consulta. | A1513 (8.0) | sa (51) | Northwind | 0:00:00 | 2156 filas | Lín 1, Col 1

Vemos que muestra la columna con la suma total de todas las cantidades.

COMPUTE BY

Hace un resumen similiar al realizado mediante Cube o Rollup. Realiza grupos del campo indicado, y muestra el resultado de la función indicada en una columna aparte por cada grupo que haya. Su formato es Compute Función(campo) By Campo
NOTA: Los campos por los que van después de la clausula **By** deben ir incluidos en la clausula ORDER BY y además en el mismo orden en el que aparecen.

```
Select Productid, Orderid, Quantity
From [Order Details]
order by ProductID, Orderid
compute Sum(quantity) by productId
```



	Productid	Orderid	Quantity
6	8	10709	40
7	8	10786	30
8	8	10829	20

	sum
1	372

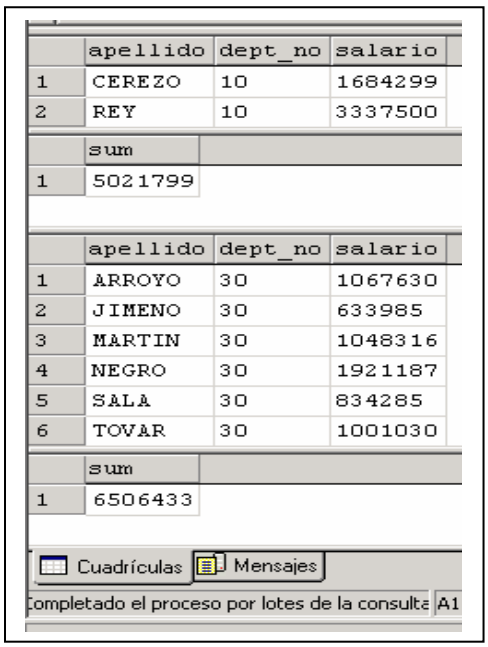
	Productid	Orderid	Quantity
1	9	10420	20
2	9	10515	16
3	9	10687	50
4	9	10693	6
5	9	10848	3

	sum
1	95

Cuadrículas | Mensajes

Completado el proceso por lotes de la consulta. | A1513 (8.0) | sa (51) | Northwind | 0:00:00 | 2232 filas | Lín 1, Col 1

```
select apellido, dept_no, salario
from emp
order by dept_no, apellido
compute sum(salario) by dept_no
```



	apellido	dept_no	salario
1	CEREZO	10	1684299
2	REY	10	3337500

	sum
1	5021799

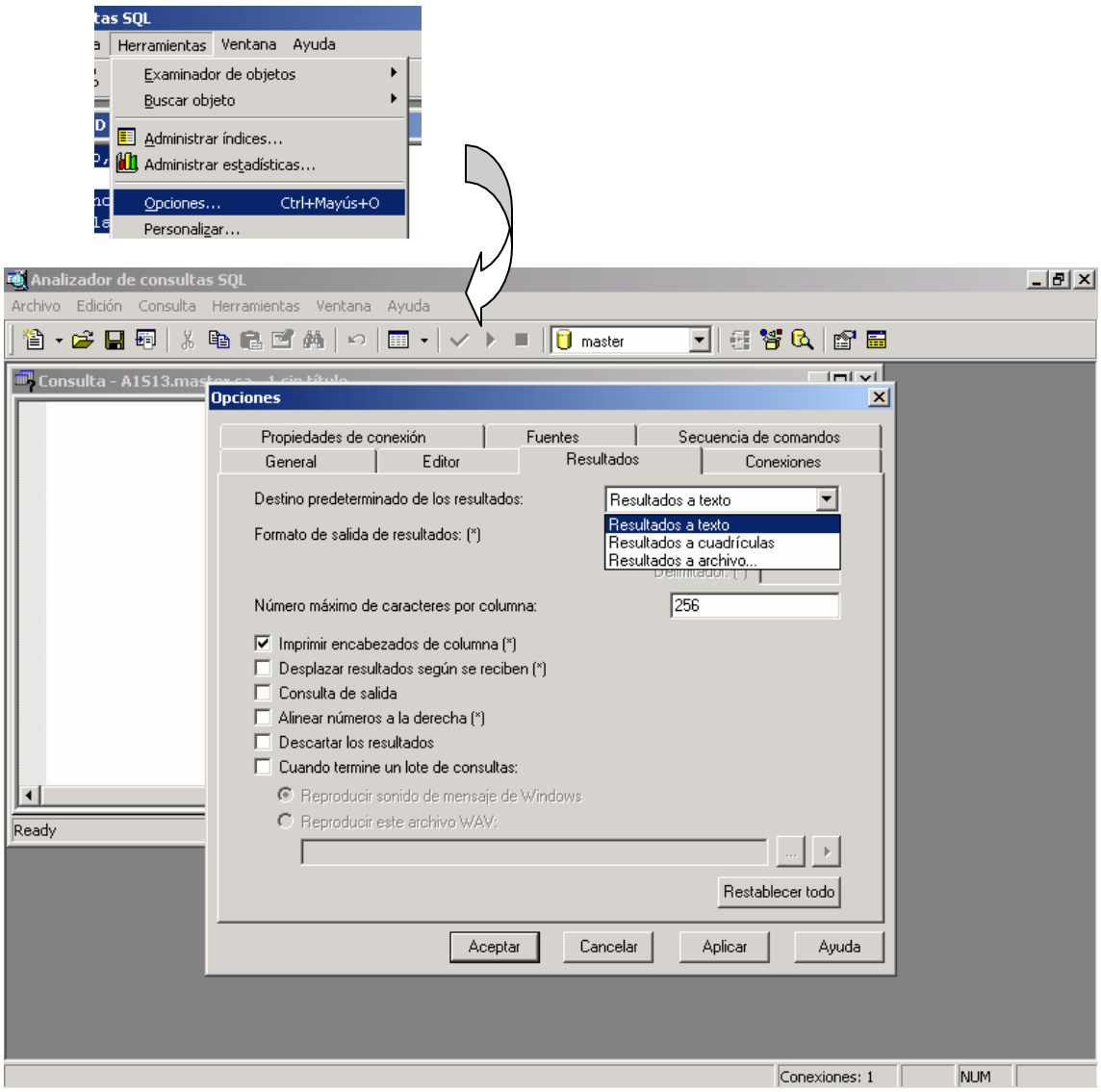
	apellido	dept_no	salario
1	ARROYO	30	1067630
2	JIMENO	30	633985
3	MARTIN	30	1048316
4	NEGRO	30	1921187
5	SALA	30	834285
6	TOVAR	30	1001030

	sum
1	6506433

Cuadrículas | Mensajes

Completado el proceso por lotes de la consulta | A1

El resultado de este procedimiento se puede mostrar en lugar de en cuadrículas, mediante texto, para ello, realizaremos los siguientes pasos:



EJERCICIOS CON COMPUTE:

1) Generar un resumen de subtotales en una consulta que devuelva el número de pedido, la cantidad pedida para todos los orderid 11070

```
Select Orderid as [Nº PEDIDO], Quantity AS [CANTIDAD]
From [Order Details]
where orderid >= 11070
order by Orderid
compute Sum(quantity) by orderid
compute sum(quantity)
```

```
Nº PEDIDO   CANTIDAD
----------- --------
11070       40
11070       20
11070       30
11070       20

          sum
          ===========
          110


          sum
          ===========
          110

(6 filas afectadas)
```

2) Generar un resumen con calculo de subtotales similar al anterior pero para los orderid 11075 y 11076

```
Select Orderid as [Nº PEDIDO], Quantity as [CANTIDAD]
From [Order Details]
where orderid = 11075 or orderid = 11076
order by Orderid
compute Sum(quantity) by ordered
```

```
Nº PEDIDO   CANTIDAD
----------- --------
11075       10
11075       30
11075       2

         sum
         ===========
         42


Nº PEDIDO   CANTIDAD
----------- --------
11076       20
11076       20
11076       10

         sum
         ===========
         50

(8 filas afectadas)
```

3) Modificación al ejercicio anterior: Agregar la cantidad total y la cantidad promedio al final del informe

```
Select Orderid as [Nº PEDIDO], Quantity as [CANTIDAD]
From [Order Details]
where orderid = 11075 or orderid = 11076
order by Orderid
compute Sum(quantity) by orderid
compute sum(quantity)
compute avg(quantity)
```

```
Nº PEDIDO   CANTIDAD
----------- --------
11075       10
11075       30
11075       2

        sum
        ===========
        42


Nº PEDIDO   CANTIDAD
----------- --------
11076       20
11076       20
11076       10

        sum
        ===========
        50


        sum
        ===========
        92


        avg
        ===========
        15

(10 filas afectadas)
```

4) Seleccionar de forma agrupada por tipo, todos los tipos, la suma de los precios y la suma del anticipo de la tabla títulos

```
Select type as [TIPO], price as [PRECIO], advance as [ANTICIPO]
From titles
order by type
compute sum(price) by type
compute avg (advance) by type
```

```
TIPO        PRECIO              ANTICIPO
------------ --------------------- ---------------------
business     19.9900             5000.0000
business     11.9500             5000.0000
business     2.9900              10125.0000
business     19.9900             5000.0000
```

```
          sum
          ======================
          54.9200


                        avg
                        ======================
                        6281.2500


TIPO        PRECIO              ANTICIPO
----------- ------------------- --------------------
mod_cook     19.9900             .0000
mod_cook     2.9900             15000.0000

          sum
          ======================
          22.9800


          avg
          ======================
          7500.0000


TIPO        PRECIO              ANTICIPO
----------- ------------------- --------------------
popular_comp 22.9500             7000.0000
popular_comp 20.0000             8000.0000
popular_comp NULL                NULL

          sum
          ======================
          42.9500


          avg
          ======================
          7500.0000


TIPO        PRECIO              ANTICIPO
----------- ------------------- --------------------
psychology   21.5900             7000.0000
psychology   10.9500             2275.0000
psychology   7.0000              6000.0000
psychology   19.9900             2000.0000
psychology   7.9900              4000.0000

          sum
          ======================
          67.5200


           avg
           ======================
           4255.0000

(30 filas afectadas)
```

5) Obtener todos los tipos, precios y anticipos de la tabla titulos individualmente ordenados por el tipo en un informe que, además, muestre la suma total de los precios y anticipo por cada tipo

```
Select type as [TIPO], price as [PRECIO], advance as [ANTICIPO]
From titles
order by type
compute sum(price) by type
compute sum(advance) by type
```

```
TIPO       PRECIO              ANTICIPO
---------- ------------------- ---------------------
business     19.9900             5000.0000
business     11.9500             5000.0000
business     2.9900             10125.0000
business     19.9900             5000.0000

          sum
          ======================
          54.9200


          sum
          ======================
          25125.0000


TIPO       PRECIO              ANTICIPO
---------- ------------------- ---------------------
mod_cook     19.9900             .0000
mod_cook     2.9900            15000.0000

          sum
          ======================
          22.9800


          sum
          ======================
          15000.0000


TIPO       PRECIO              ANTICIPO
---------- ------------------- ---------------------
popular_comp 22.9500             7000.0000
popular_comp 20.0000             8000.0000
popular_comp NULL               NULL

          sum
          ======================
          42.9500


          sum
          ======================
          15000.0000


(30 filas afectadas)
```

6) Generar un informe que muestre la suma de los precios de los libros de psicología de cada editor

```
Select type AS [TIPO]
,pub_id as [CODIGO EDITOR]
,price as [PRECIO]
From titles
where type = 'psychology'
order by pub_id
compute sum(price) by pub_id
```

```
TIPO       CODIGO EDITOR PRECIO
---------- ------------- ---------------------
psychology  0736         10.9500
psychology  0736         7.0000
psychology  0736         19.9900
psychology  0736         7.9900

                    sum
                    ======================
                    45.9300


TIPO       CODIGO EDITOR PRECIO
---------- ------------- ---------------------
psychology  0877         21.5900

                    sum
                    ======================
                    21.5900


(7 filas afectadas)
```

7) Generar un informe que obtenga la suma de los precios de todos los libros de psicología, así como la suma de los precios de los libros de psicología por editor

```
Select type as [TIPO]
,pub_id as [CODIGO EDITOR]
,price as [PRECIO]
From titles
where type = 'psychology'
order by pub_id
compute sum(price) by pub_id
compute sum(price)
```

```
TIPO       CODIGO EDITOR PRECIO
---------- ------------- ---------------------
psychology  0736         10.9500
psychology  0736         7.0000
psychology  0736         19.9900
psychology  0736         7.9900

                    sum
                    ======================
                    45.9300
```

```
TIPO        CODIGO EDITOR PRECIO
------------ ------------ ---------------------
psychology   0877         21.5900

                    sum
                    ======================
                    21.5900


                    sum
                    ======================
                    67.5200

(8 filas afectadas)
```

8) Generar un informe que obtenga la suma de los precios y los anticipos para cada tipo de libro de cocina

```
Select type as [TIPO]
,pub_id [CODIGO EDITOR]
,price as [PRECIO]
,advance as [ANTICIPO]
From titles
where type like '%cook'
order by type, advance
compute sum(price) by type
compute sum(advance) by type
```

```
TIPO        CODIGO EDITOR PRECIO              ANTICIPO
------------ ------------ --------------------- ---------------------
mod_cook     0877         19.9900               .0000
mod_cook     0877         2.9900                15000.0000

                    sum
                    ======================
                    22.9800


                                sum
                                ======================
                                15000.0000

TIPO        CODIGO EDITOR PRECIO              ANTICIPO
------------ ------------ --------------------- ---------------------
trad_cook    0877         11.9500               4000.0000
trad_cook    0877         20.9500               7000.0000
trad_cook    0877         14.9900               8000.0000

                    sum
                    ======================
                    47.8900


                                sum
                                ======================
                                19000.0000

(9 filas afectadas)
```

9) Generar un informe que muestre todos los títulos, precio y anticipo de aquellos títulos que tengan un precio superior a 20 dólares. Ofrecerá también la suma total del precio y del anticipo.

```
Select title as [TITULO]
,price as [PRECIO]
,advance as [ANTICIPO]
From titles
where price > 20
order by title
compute sum(price)
compute sum(advance)
```

```
TITULO                                                                          PRECIO              ANTICIPO
------------------------------------------------------------------------------- ------------------- ---------------
-------
But Is It User Friendly?                                                         22.9500             7000.0000
Computer Phobic AND Non-Phobic Individuals: Behavior Variations                  21.5900
7000.0000
Onions, Leeks, and Garlic: Cooking Secrets of the Mediterranean                  20.9500
7000.0000

sum
=====================
65.4900


sum
=====================
21000.0000


(5 filas afectadas)
```

10) Generar un informe que muestre el precio, el tipo y el número de editor de la tabla titles de todos aquellas obras que pertenezcan la categoría de "business". Además aparecerá la suma total de precios y el máximo pub_id de esta selección

```
Select type as [TIPO]
,price as [PRECIO]
,pub_id as [CODIGO EDITOR]
From titles
where type = 'business'
order by pub_id
compute sum(price)
compute max(pub_id)
```

```
TIPO         PRECIO               CODIGO EDITOR
------------ -------------------- -------------
business     2.9900               0736
business     19.9900               1389
business     19.9900               1389
business     11.9500               1389

          sum
          =====================
          54.9200

          max
          ====
          1389
 (6 filas afectadas)
```

11) Realizar una consulta que recupere el tipo, precio y anticipo de la tabla titles para los libros de cocina. Mostrará la suma de los precios y los anticipos por tipo y luego, calculará el total general de los precios y los anticipos para todos los registros seleccionados.

```
Select type as [TIPO]
,price as [PRECIO]
,advance as [ANTICIPO]
From titles
where type like '%cook'
order by type
compute sum(price) by type
compute sum(advance) by type
compute sum(price)
compute sum(advance)
```

```
TIPO       PRECIO             ANTICIPO
---------- ------------------ ---------------------
mod_cook     19.9900            .0000
mod_cook     2.9900           15000.0000

          sum
          ======================
          22.9800


          sum
          ======================
          15000.0000


TIPO       PRECIO             ANTICIPO
---------- ------------------ ---------------------
trad_cook    20.9500            7000.0000
trad_cook    11.9500            4000.0000
trad_cook    14.9900            8000.0000

          sum
          ======================
          47.8900


          sum
          ======================
          19000.0000


          sum
          ======================
          70.8700


           sum
           ======================
           34000.0000

(11 filas afectadas)
```

ROLLUP

1) Realizar una consulta que resuma la cantidad de artículos pedidos por cada índice de producto y número de pedido. Realizando un cálculo acumulativo.

```
select productid as [PRODUCTO]
,orderid as [Nº PEDIDO]
,sum(quantity) as [SUMA TOTAL]
from [Order Details]
group by productid, orderid
with rollup
order by productid, orderid
```

```
PRODUCTO    Nº PEDIDO   SUMA TOTAL
----------- ----------- -----------
NULL        NULL        51317
1           NULL        828
1           10285       45
1           10294       18
1           10317       20
1           10348       15
1           10354       12
1           10370       15
1           10406       10
1           10413       24
1           10477       15
1           10522       40
1           10526       8
1           10576       10
1           10590       20
…
```

2) Modificación al ejercicio anterior: Realizar el mismo resumen pero únicamente para el producto cuyo id es 50

```
select productid as [PRODUCTO]
,orderid as [Nº PEDIDO]
,sum(quantity) as [SUMA TOTAL]
from [Order Details]
where productid = 50
group by productid, orderid
with rollup
order by productid, ordered
```

```
PRODUCTO    Nº PEDIDO   SUMA TOTAL
----------- ----------- -----------
NULL        NULL        235
50          NULL        235
50          10350       15
50          10383       15
50          10429       40
50          10465       25
50          10637       25
50          10729       40
50          10751       20
50          10920       24
50          10948       9
50          11072       22

(12 filas afectadas)
```

3) Realizar un resumen por medio de CUBE y de GROUPING sobre la modificación del ejercicio anterior de tal manera que sea posible obtener un resumen por producto y por pedido.

```
select Productid as [PRODUCTO]
,grouping(productid) as [GRUPO PRODUCTO]
,orderid as [Nº PEDIDO]
,grouping(orderid) as [GRUPO PEDIDO]
,sum(quantity) as [SUMA TOTAL]
from [Order Details]
where productid = 50
group by Productid,orderid
with cube
order by productid,orderid
```

```
PRODUCTO    GRUPO PRODUCTO Nº PEDIDO   GRUPO PEDIDO SUMA TOTAL
----------- -------------- ----------- ------------ -----------
NULL        1              NULL        1            235
NULL        1              10350       0            15
NULL        1              10383       0            15
NULL        1              10429       0            40
NULL        1              10465       0            25
NULL        1              10637       0            25
NULL        1              10729       0            40
NULL        1              10751       0            20
NULL        1              10920       0            24
NULL        1              10948       0            9
NULL        1              11072       0            22
50          0              NULL        1            235
50          0              10350       0            15
50          0              10383       0            15
50          0              10429       0            40
50          0              10465       0            25
50          0              10637       0            25
50          0              10729       0            40
50          0              10751       0            20
50          0              10920       0            24
50          0              10948       0            9
50          0              11072       0            22

(22 filas afectadas)
```

¿Qué filas son de resumen? ¿Cuáles el resumen por producto y cuáles por pedido?

TRANSACCIONES

Una transacción es un conjunto de instrucciones de manipulación de datos que se ejecutan en una misma unidad de trabajo. El ejemplo más claro son las transacciones bancarias.

- Inicio de una transacción:
  BEGIN TRAN[SACTION][NombreTransaccion]
- Validación de transacción:
  COMMIT TRAN[SACTION] [NombreTransaccion]
- Decalración de punto de control:
  SAVE TRAN[SACTION] [NombrePuntoControl]
- Anulación de Transacción:

ROLLBACK TRAN[SACTION] [NombreTransaccion | NombrePuntoControl]

```
BEGIN TRAN Modificación
UPDATE Tabla SET ....
UPDATE Tabla1 SET ....
SAVE TRAN a
UPDATE Tabla2 SET ....
UPDATE Tabla3 SET ....
ROLLBACK TRAN a
ROLLBACK TRAN Modificacion
COMMIT TRAN Modificación
```

El primer Rollback guardaria las que estan en el Begin y el punto de salvamento, deshaciendo las demás ordenes.

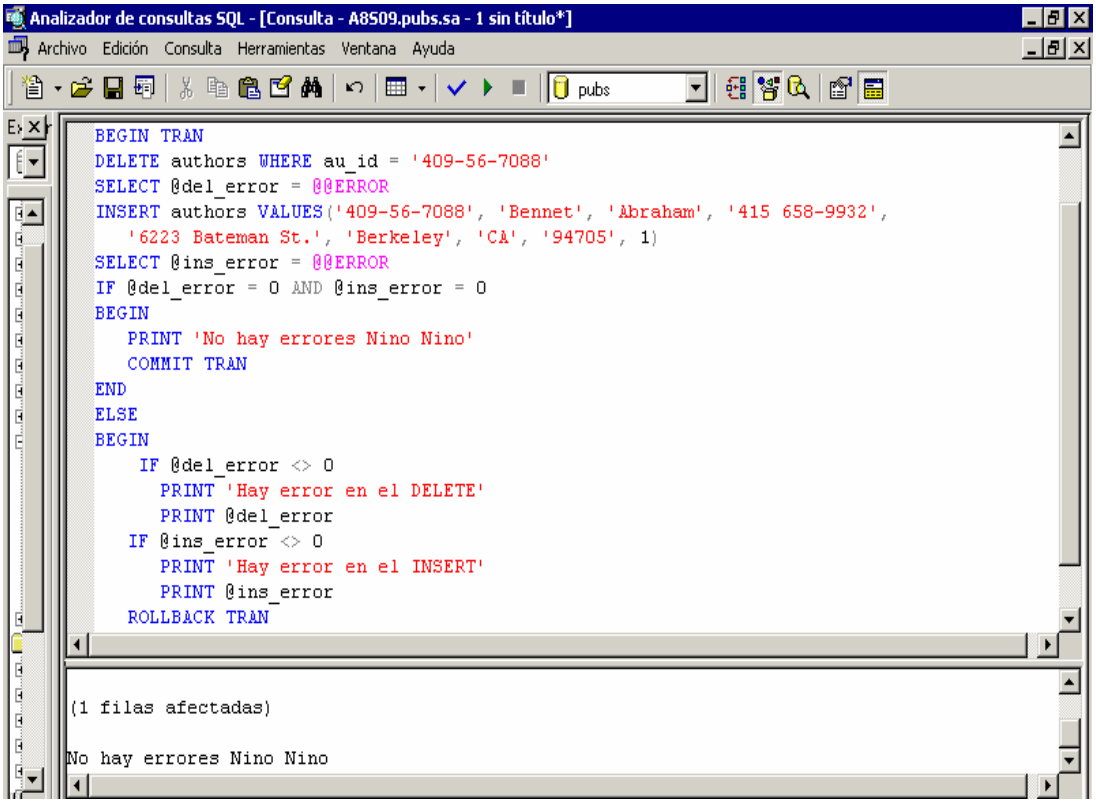
Ejemplo de transacciones usando @@ERROR, con esto se almacena el número del error que se ha producido en última instancia. Se actualiza en cada instrucción, por eso después de cada sentencia se debe comprobar el valor de la variable para ver si tiene error. Lo que se hace es crear variables en cada instrucción almacenando el valor de @@ERROR en cada una de ellas. El valor de las variables debe ser entero.

```
USE Pubs
GO
DECLARE @del_error int, @ins_error int
BEGIN TRAN
DELETE authors WHERE au_id = '409-56-7088'
SELECT @del_error = @@ERROR

INSERT authors VALUES ('409-56-7008', 'Bennet', 'Abraham', '415 658-9932',
  '6223 Bateman St.', 'Berkeley', 'CA', '94705', 1)
SELECT @ins_error = @@ERROR

IF @del_error = 0 AND @ins_error = 0
BEGIN
  PRINT 'No hay errores Nino Nino'
  COMMIT TRAN
END
ELSE
BEGIN

  IF @del_error <> 0
    PRINT 'Hay error en el DELETE'
    PRINT @ins_error
  IF @ins_error <> 0
    PRINT 'Hay error en el INSERT'
    PRINT @ins_error
  ROLLBACK TRAN
END
GO
```



FUNCIONES DE FECHA

Parte de la fecha	Abreviaturas
Year	yy, yyyy
Quarter	qq, q
Month	mm, m
dayofyear	dy, y
Day	dd, d
Week	wk, ww
Hour	Hh
Minute	mi, n
Second	ss, s
Millisecond	Ms

GetDate

```
select getdate()   Funcion que recupera la fecha actual del sistema.
```

Convert, Cast

Convierten explícitamente una expresión de un tipo de datos en otro. CAST y CONVERT proporcionan funciones similares.
Convert(Tipodedatosdestino, Origen, Estilo)

- *Tiposdedatosdestino:* Siempre ha de ser tipo carácter
- *Origen:* Puede ser tipo fecha, numérico o moneda.
- *Estilo:* Opcional. Es un código que indica el formato en el que devuelve la cadena de caracteres.

Sintaxis:

Convert(TipoDato,Dato)
Cast (Dato as TipoDato)

```
Ejemplo:

declare @n int,@palabra nvarchar(10)
set @n = 1
set @palabra ='Número'
print convert(nvarchar(2),@n) + ' ' + @palabra
print cast(@n as nvarchar(2)) + ' ' + @palabra

while (@n<11)
      begin
            print convert(nvarchar(2),@n) + ' ' + @palabra
            print cast(@n as nvarchar(2)) + ' ' + @palabra
      set @n = @n + 1
      end
```

DateName

Devuelve una cadena de caracteres que representa la parte de la fecha especificada de la fecha especificada
Los calculos para las horas no son exactos cuando se trata de SmallDateTime, por lo que devuelve 0.

```
select datename(month,fecha_alt) as 'Nombre mes' from emp where emp_no = 7867
   diciembre
select datename(m,fecha_alt) as 'Nombre mes' from emp where emp_no =
7867   diciembre
select datename(week,fecha_alt) as 'Numero Semana' from emp where
emp_no = 7867   51
select datename(week,fecha_alt) as 'Numero Semana' from emp where
emp_no = 7867    51
select datename( weekday,fecha_alt) as 'Dia Semana' from emp where
emp_no=7867    miercoles
select datename(dw,fecha_alt) as 'Dia Semana' from emp where
emp_no=7867    miercoles
```

Horas:

```
select datename(mi,fecha_alt) as 'Minutos' from emp where emp_no =
7867
select datename( minute,getdate()) as 'minutos'
select datename( mi,getdate()) as 'minutos'
select datename( hh,fecha_alt) as 'hora' from emp where emp_no = 7867
select datename( hour,getdate()) as 'hora'-->17
select datename( hh,getdate()) as 'hora'-->17
```

DatePart

Devuelve la parte de la fecha u hora indicada. Sintaxis:
Datepart(Valoradevolver, fecha)

```
select datepart(mm,fecha_alt) as 'Mes',apellido from emp-->11
select datepart(hh,getdate()) as 'Hora'-->17
select datepart(mi,getdate()) as 'Minutos'-->54
```

Nombres de Fechas

```
Day(fecha)
Devuelve un INT, equivale a datepart

select day(getdate()) as dia-->12
select datepart(dd,getdate())-->12
select month(getdate())-->7
select datepart(mm,getdate())-->7
select year(getdate())-->2002
select year(fecha_alt) from emp where emp_no = 7867
select datediff(yyyy,fecha_alt,getdate()) as 'Diferencia' from emp
where emp_no = 7867
```

DateAdd

```
DateAdd( datepart , number, date )
Añade un número a la fecha puesta
DatePart es el formato de lo que queremos añadir.
Number es el número que queremos incrementar la fecha expuesta.
```

```
select convert(datetime,'1-1-02')
select dateadd(dd,7,'1-1-02')
```

DateDiff

Devuelve la diferencia entre dos fechas en el intervalo que le indiquemos. Sintaxis:

DateDiff (DatoqueDevuelve, Fecha1, Fecha2)

- *Datoquedevuelve*: Indicamos como queremos que haga la comparación y el tipo de dato que nos devolverá, años, días, minutos etc.

```
select datediff(yyyy,fecha_alt,getdate()) as 'Diferencia' from emp
where emp_no = 7867
```

FUNCIONES MATEMATICAS

ABS

Es el valor Absoluto

```
Select ABS(-4) as 'VALOR ABSOLUTO'-->4
```

CEILING

Devuelve el entero más pequeño mayor o igual que la expresión numérica dada.

```
Select CEILING(5.4) as 'CEILING'--6
```

FLOOR

Devuelve el entero más grande menor o igual que la expresión numérica dada.

```
Select FLOOR(5) as 'FLOOR'-->5
```

POWER

Devuelve el valor de la expresión indicada elevada a la potencia especificada.

```
Select POWER(3,2) as '3 ELEVADO A 2'-->9
```

RAND

Devuelve un valor float aleatorio de 0 a 1.
Las llamadas repetitivas de RAND() en una única consulta producirán el mismo valor.

```
Select RAND(6) as 'ALEATORIO'--0.71368515806921451
Select RAND(6) as 'ALEATORIO'--0.71368515806921451
Select RAND(4) as 'ALEATORIO'--0.7136478921266981
```

Rand sobre los milisegundos actuales

```
Select RAND(DATEPART(ms,GETDATE())) as 'ALEATORIO'--
>0.71443047691954253
Select RAND(999999999)-->0.68504257551273573
```

ROUND

Devuelve una expresión numérica, redondeada a la longitud o precisión especificada.
Round(Numero, Redondeo del Número)
ROUND siempre devuelve un valor. Si length es un valor negativo y mayor que el número de dígitos anteriores al separador decimal, ROUND devuelve 0.

```
Select ROUND(123.4567,2)-->123.4600
Select ROUND(123.4567,-2)-->100.0000
Select ROUND(123.4567,0)-->123.0000
Select ROUND(123.4567,-3)--->0
```

SIGN

Devuelve el signo positivo (+1), cero (0) o negativo (-1) de la expresión especificada.
Dice el valor negativo, positivo o neutro (0) del valor especificado

```
Select SIGN(-3)-->-1
Select SIGN(3)-->1
Select SIGN(0)-->0
```

SQUARE

Devuelve el cuadrado de la expresión especificada.

```
Select SQUARE(4) as 'Cuadrado'-->16.0
```

SQRT

Devuelve la raíz cuadrada de la expresión especificada.

```
Select SQRT(4) as [RAIZ CUADRADA]-->2.0
```

FUNCIONES DE CADENA

ASCII

Devuelve el código ASCII del carácter más a la izquierda de una expresión de caracteres.

```
Select ASCII('A')-->65
Select ASCII('a')-->97
Select ascii('aula')-->97
```

CHAR

Una función de cadena que convierte un código ASCII int en un carácter.

```
select char(65)-->A
select char(97)-->a
```

CHARINDEX

Devuelve la posición inicial de la expresión especificada en una cadena de caracteres.

```
CHARINDEX ( expression1 , expression2 [ , start_location ] )
```

Argumentos
expression1
Es una expresión que contiene la secuencia de caracteres que se desea buscar.
Expression1 es una expresión del tipo de cadenas cortas de caracteres.

Expression2
Es una expresión, normalmente una columna,
en la que se busca la cadena especificada.
Expression2 es de la categoría del tipo de datos cadena de caracteres.

start_location
Es la posición del carácter de expression2 en el que se empieza la búsqueda de expression1.
Si no se especifica start_location, es un número negativo o es cero, la búsqueda empieza al principio de la cadena expression2.

Si expression1 no se encuentra en expression2, CHARINDEX devuelve 0.
Si alguno de los dos es null, devuelve null

```
select charindex('cie','murcielago')-->4
select charindex('cie','murcielago',2)-->4
select charindex('cie','murcielago',5)-->0
select charindex('cie','murcielago',-6)-->4
```

LEFT

Devuelve la parte de una cadena de caracteres que comienza en un número de caracteres especificado a partir de la izquierda

```
select left('murcielago',5)-->murci
```

RIGHT

Devuelve la parte de una cadena de caracteres que comienza en el número de caracteres especificado en integer_expression a partir de la derecha.

```
select right('hola que tal',5)-->e tal
```

LEN

Cuenta el número de caracteres que se incluyen en la cadena.

```
select len('murcielago')-->10
```

LOWER

Convierte a Minúsculas la cadena especificada

```
select lower('MurcIELaGO') as [minusculas]-->murcielago
```

UPPER

Convierte a Mayúsculas la cadena especificada

```
select upper('murcielago') as [MAYUSCULAS]-->MURCIELAGO
```

RTRIM y LTRIM

Elimina los espacios que existen a la izquierda y a la derecha respectivamente.

```
select Rtrim ('   murcielago       ') AS [SIN ESPACIOS]-->murcielago
select Ltrim ('   murcielago       ') AS [SIN ESPACIOS]-->murcielago
select ltrim(rtrim('   hola   '))+'.'
```

REPLACE

Reemplaza por una tercera expresión todas las apariciones de la segunda expresión de cadena proporcionada en la primera expresión de cadena

```
select replace('hola que tal estas','a','A')-->holA que tAl estAs
select replace('buenos dias, que tal estas','ue','ñññ')-->bñññnos
dias,qñññ tal estas
```

SPACE

Coloca el número de espacios que se le indiquen para entre una cadena de caracteres.

```
select 'hola'+space(5)+'que tal'-->hola      que tal
```

SUBSTRING

```
      Devuelve parte de una expresión de caracteres, binaria, de texto
      o de imagen.
Sintaxis:

SUBSTRING (Expresión , Comienzo , Duración )

Argumentos
expression
Es una cadena de caracteres, cadena binaria, texto, imagen,
columna o expresión que incluye una columna.
No deben usarse expresiones que incluyan funciones de agregado.

start
Es un entero que especifica el punto en que comienza la subcadena.

length
Es un entero que especifica la longitud de la subcadena
(el número de caracteres o bytes que se devuelven).

select substring('murcielago',3,5)-->rciel
select substring('murcielago',3,len('murcielago'))-->rciel
```

REVERSE

```
      Devuelve invertida una expresión de carácter.

select reverse('hola')
```

REPLICATE

```
      Repite una expresión de caracteres un número especificado de
veces.
      select replicate('murcielago',5)

replicate, replicate, replicate, replicate, replicate
```

STUFF

```
      Elimina el número de caracteres especificado e inserta
      otro conjunto de caracteres en un punto de inicio indicado.

Select STUFF('Murcielago', 2, 3, 'ijklmn')   Mijklmnielago
```