

Lab 20

Emiliano Valdivia Lara

A01276258

Consulta de un tabla completa

Algebra relacional.

materiales

SQL

select * from materiales

1	SELECT *
2	FROM materiales

materiales (45r × 4c)				
#	clave	descripcion	precio	impuesto
1	1,000	Varilla 3/16	100	10
2	1,010	Varilla 4/32	115	11.5
3	1,020	Varilla 3/17	130	13
4	1,030	Varilla 4/33	145	14.5
5	1,040	Varilla 3/18	160	16
6	1,050	Varilla 4/34	175	17.5
7	1,060	Varilla 3/19	190	19
8	1,070	Varilla 4/35	205	20.5
9	1,080	Ladrillos rojos	50	5
10	1,090	Ladrillos grises	35	3.5
11	1,100	Block	30	3
12	1,110	Megablock	40	4
13	1,120	Sillar rosa	100	10
14	1,130	Sillar gris	110	11
15	1,140	Cantera blanca	200	20
16	1,150	Cantera gris	1,210	121
17	1,160	Cantera rosa	1,420	142
18	1,170	Cantera amarilla	230	23
19	1,180	Recubrimiento P1001	200	20
20	1,190	Recubrimiento P1010	220	22
21	1,200	Recubrimiento P1019	240	24

Selección

Algebra relacional.

$SL\{clave=1000\}(materiales)$

SQL

```
select * from materiales
```

```
where clave=1000
```

1	select	*	from	materiales
2	where	clave=1000		

materiales (1r × 4c)				
#	clave	descripcion	precio	impuesto
1	1,000	Varilla 3/16	100	10

Proyección

Algebra relacional.

$PR\{clave, rfc, fecha\}(entregan)$

SQL

```
select clave, rfc, fecha from entregan
```

```
1 select clave,rfc,fecha from entregan
```

entregan (87r x 3c)			
#	clave	rfc	fecha
1	1,000	AAAA800101	2001-12-13
2	1,200	EEEE800101	2003-03-15
3	1,400	AAAA800101	1999-04-07
4	1,010	BBBB800101	1998-07-28
5	1,210	FFFF800101	2000-05-21
6	1,410	BBBB800101	2000-05-18
7	1,020	CCCC800101	2003-12-16
8	1,220	GGGG800101	2005-07-03
9	1,420	CCCC800101	2001-09-09
10	1,030	DDDD800101	1998-01-12
11	1,230	HHHH800101	1998-09-12
12	1,430	DDDD800101	2005-04-30
13	1,040	EEEE800101	1999-12-18
14	1,150	HHHH800101	2002-11-14
15	1,240	AAAA800101	0000-00-00
16	1,050	FFFF800101	2004-02-07
17	1,140	GGGG800101	2000-06-30
18	1,250	BBBB800101	2003-10-12
19	1,300	CCCC800101	2004-02-28

Reunión Natural

Algebra relacional.
entregan JN materiales

SQL
select * from materiales,entregan
where materiales.clave = entregan.clave

Si algún material no ha se ha entregado ¿Aparecería en el resultado de esta consulta?

```

1 select * from materiales,entregan
2 where materiales.clave = entregan.clave

```

Resultado #1 (87r x 9c)

#	clave	descripcion	precio	impuesto	clave	rfc	numero	fecha	cantidad
1	1,000	Varilla 3/16	100	10	1,000	AAAA800101	5,000	2001-12-13	165
2	1,000	Varilla 3/16	100	10	1,000	AAAA800101	5,019	1999-07-13	254
3	1,010	Varilla 4/32	115	11.5	1,010	BBBB800101	5,001	1998-07-28	528
4	1,010	Varilla 4/32	115	11.5	1,010	BBBB800101	5,018	1997-02-09	523
5	1,020	Varilla 3/17	130	13	1,020	CCCC800101	5,002	2003-12-16	582
6	1,020	Varilla 3/17	130	13	1,020	CCCC800101	5,017	2000-03-29	8
7	1,030	Varilla 4/33	145	14.5	1,030	DDDD800101	5,003	1998-01-12	202
8	1,030	Varilla 4/33	145	14.5	1,030	DDDD800101	5,016	2005-06-07	295
9	1,040	Varilla 3/18	160	16	1,040	EEEE800101	5,004	1999-12-18	263
10	1,040	Varilla 3/18	160	16	1,040	EEEE800101	5,015	1999-05-29	540
11	1,050	Varilla 4/34	175	17.5	1,050	FFFF800101	5,005	2004-02-07	503
12	1,050	Varilla 4/34	175	17.5	1,050	FFFF800101	5,014	2000-04-18	623
13	1,060	Varilla 3/19	190	19	1,060	GGGG800101	5,006	2004-09-29	324
14	1,060	Varilla 3/19	190	19	1,060	GGGG800101	5,013	2002-07-23	692
15	1,070	Varilla 4/35	205	20.5	1,070	HHHH800101	5,007	2006-11-21	2
16	1,070	Varilla 4/35	205	20.5	1,070	HHHH800101	5,012	2004-11-27	503
17	1,080	Ladrillos rojos	50	5	1,080	AAAA800101	5,008	2005-04-03	86
18	1,080	Ladrillos rojos	50	5	1,080	AAAA800101	5,011	2003-08-28	699

No, porque la información de las tablas se despliega cuando la clave de material se encuentra en también en la de entregan.

Reunión con criterio específico

Algebra relacional.

entregan JN{entregan.numero <= proyectos.numero} proyectos

SQL

```
select * from entregan,proyectos
```

```
where entregan.numero < = proyectos.numero
```

```

1 select * from entregan,proyectos
2 where entregan.numero < = proyectos.numero

```

Unnamed: Error



Error de SQL (1064): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near '= proyectos.numero' at line 2

Aceptar



Encontrar ayuda acerca de este error (=> [ecosia.org](https://www.ecosia.org))

Unión (se ilustra junto con selección)

Algebra relacional.

$SL\{clave=1450\}(entregan) \cup SL\{clave=1300\}(entregan)$

SQL

```
(select * from entregan where clave=1450)
```

union

```
(select * from entregan where clave=1300)
```

¿Cuál sería una consulta que obtuviera el mismo resultado sin usar el operador Unión? Compruébalo.

```
1 (select * from entregan where clave=1450)
2 union
3 (select * from entregan where clave=1300)
```

entregan (2r x 5c)					
#	clave	rfc	numero	fecha	cantidad
1	1,300	GGGG800101	5,005	2004-02-28	521
2	1,300	GGGG800101	5,010	2001-02-10	119

Una consulta que arroja un resultado idéntico sin usar unión es la siguiente: `(select * from entregan where clave=1300)`, esto se debe a que la clave 1450 no existe.

```
1
2 (select * from entregan where clave=1300)
```

entregan (2r x 5c)					
#	clave	rfc	numero	fecha	cantidad
1	1,300	GGGG800101	5,005	2004-02-28	521
2	1,300	GGGG800101	5,010	2001-02-10	119

Intersección (se ilustra junto con selección y proyección)

Algebra relacional.

$PR\{clave\}(SL\{numero=5001\}(entregan)) \cap PR\{clave\}(SL\{numero=5018\}(entregan))$

SQL

Nota: Debido a que en SQL server no tiene definida alguna palabra reservada que nos permita hacer esto de una manera entendible, veremos esta sección en el siguiente laboratorio con el uso de Subconsultas. Un ejemplo de un DBMS que si tiene la implementación de una palabra reservada para esta función es Oracle, en él si se podría generar la consulta con una sintaxis como la siguiente:

```
(select clave from entregan where numero=5001)
```

intersect

```
(select clave from entregan where numero=5018)
```

```

1 (select clave from entregan where numero=5001)
2 intersect
3 (select clave from entregan where numero=5018)

```

entregan (1r × 1c)

#	clave
1	1,010

Diferencia (se ilustra con selección)

Algebra relacional.

entregan - SL{clave=1000}(entregan)

SQL

(select * from entregan)

minus

(select * from entregan where clave=1000)

Nuevamente, "minus" es una palabra reservada que no está definida en SQL Server, define una consulta que regrese el mismo resultado.

```

1 (select * from entregan)
2 except
3 (select * from entregan where clave=1000)

```

entregan (85r × 5c)

#	clave	rfc	numero	fecha	cantidad
1	1,010	BBBB800101	5,001	1998-07-28	528
2	1,010	BBBB800101	5,018	1997-02-09	523
3	1,020	CCCC800101	5,002	2003-12-16	582
4	1,020	CCCC800101	5,017	2000-03-29	8
5	1,030	DDDD800101	5,003	1998-01-12	202
6	1,030	DDDD800101	5,016	2005-06-07	295
7	1,040	EEEE800101	5,004	1999-12-18	263
8	1,040	EEEE800101	5,015	1999-05-29	540
9	1,050	FFFF800101	5,005	2004-02-07	503
10	1,050	FFFF800101	5,014	2000-04-18	623
11	1,060	GGGG800101	5,006	2004-09-29	324
12	1,060	GGGG800101	5,013	2002-07-23	692
13	1,070	HHHH800101	5,007	2006-11-21	2
14	1,070	HHHH800101	5,012	2004-11-27	503
15	1,080	AAAA800101	5,008	2005-04-03	86
16	1,080	AAAA800101	5,011	2003-08-28	699
17	1,090	BBBB800101	5,009	1997-03-13	73
18	1,090	BBBB800101	5,010	1998-11-17	421
19	1,100	CCCC800101	5,000	2000-12-07	456

Producto cartesiano

Algebra relacional.

entregan X materiales

SQL

select * from entregan,materiales

¿Cómo está definido el número de tuplas de este resultado en términos del número de tuplas de entregan y de materiales?

```
1 select * from entregan,materiales
```

Resultado #1 (3,915r × 9c)

#	clave	rfc	numero	fecha	cantidad	clave	descripcion	precio	impuesto
1	1,000	AAAA800101	5,000	2001-12-13	165	1,000	Vanilla 3/16	100	10
2	1,000	AAAA800101	5,000	2001-12-13	165	1,010	Vanilla 4/32	115	11.5
3	1,000	AAAA800101	5,000	2001-12-13	165	1,020	Vanilla 3/17	130	13
4	1,000	AAAA800101	5,000	2001-12-13	165	1,030	Vanilla 4/33	145	14.5
5	1,000	AAAA800101	5,000	2001-12-13	165	1,040	Vanilla 3/18	160	16
6	1,000	AAAA800101	5,000	2001-12-13	165	1,050	Vanilla 4/34	175	17.5
7	1,000	AAAA800101	5,000	2001-12-13	165	1,060	Vanilla 3/19	190	19
8	1,000	AAAA800101	5,000	2001-12-13	165	1,070	Vanilla 4/35	205	20.5
9	1,000	AAAA800101	5,000	2001-12-13	165	1,080	Ladrillos rojos	50	5
10	1,000	AAAA800101	5,000	2001-12-13	165	1,090	Ladrillos grises	35	3.5
11	1,000	AAAA800101	5,000	2001-12-13	165	1,100	Block	30	3
12	1,000	AAAA800101	5,000	2001-12-13	165	1,110	Megablock	40	4
13	1,000	AAAA800101	5,000	2001-12-13	165	1,120	Sillar rosa	100	10
14	1,000	AAAA800101	5,000	2001-12-13	165	1,130	Sillar gris	110	11
15	1,000	AAAA800101	5,000	2001-12-13	165	1,140	Cantera blanca	200	20
16	1,000	AAAA800101	5,000	2001-12-13	165	1,150	Cantera gris	1,210	121
17	1,000	AAAA800101	5,000	2001-12-13	165	1,160	Cantera rosa	1,420	142
18	1,000	AAAA800101	5,000	2001-12-13	165	1,170	Cantera amarilla	230	23

Se entrega el producto de cruzar cada registro de ambas tablas, por lo tanto, el total de registros es una multiplicación de la cantidad de tuplas de cada tabla.

Construcción de consultas a partir de una especificación

Plantea ahora una consulta para obtener las descripciones de los materiales entregados en el año 2000.

Recuerda que la fecha puede indicarse como '01-JAN-2000' o '01/01/00'.

Importante: Recuerda que cuando vayas a trabajar con fechas, antes de que realices tus consultas debes ejecutar la instrucción "set dateformat dmy". Basta con que la ejecutes una sola vez para que el manejador sepa que vas a trabajar con ese formato de fechas.

¿Por qué aparecen varias veces algunas descripciones de material?

```
1 select descripcion
2 FROM materiales NATURAL JOIN entregan
3 WHERE fecha LIKE '2000%'
```

materiales (12r × 1c)

#	descripcion
1	Vanilla 3/17
2	Vanilla 4/34
3	Block
4	Sillar gris
5	Sillar gris
6	Cantera blanca
7	Recubrimiento P1028
8	Tubería 3.6
9	Pintura C1010
10	Pintura B1021
11	Pintura B1021
12	Pintura B1022

Algunas descripciones aparecen más de una vez porque se entregaron más de una vez durante el año 2000.

Uso del calificador distinct

En el resultado anterior, observamos que una misma descripción de material aparece varias veces.

Agrega la palabra **distinct** inmediatamente después de la palabra **select** a la consulta que planteaste antes.

¿Qué resultado obtienes en esta ocasión?

```

1 select DISTINCT descripcion
2 FROM materiales NATURAL JOIN entregan
3 WHERE fecha LIKE '2000%'

```

materiales (10r x 1c)	
#	descripcion
1	Varilla 3/17
2	Varilla 4/34
3	Block
4	Sillar gris
5	Cantera blanca
6	Recubrimiento P1028
7	Tubería 3.6
8	Pintura C1010
9	Pintura B1021
10	Pintura B1022

Ya no se repiten los registros que antes se repetían.

Ordenamientos.

Si al final de una sentencia select se agrega la cláusula

order by campo [desc] [,campo [desc] ...]

donde las partes encerradas entre corchetes son opcionales (los corchetes no forman parte de la sintaxis), los puntos suspensivos indican que pueden incluirse varios campos y la palabra desc se refiere a descendente. Esta cláusula permite presentar los resultados en un orden específico.

Obtén los números y denominaciones de los proyectos con las fechas y cantidades de sus entregas, ordenadas por número de proyecto, presentando las fechas de la más reciente a la más antigua.

```

1 SELECT numero, denominacion, fecha, cantidad
2 FROM proyectos NATURAL JOIN entregan
3 ORDER BY numero, fecha DESC

```

Resultado #1 (87r x 4c)					
#	numero	denominacion	fecha	cantidad	
1	5,000	Vamos Mexico	2003-03-15	177	
2	5,000	Vamos Mexico	2001-12-13	165	
3	5,000	Vamos Mexico	1999-04-07	382	
4	5,001	Aztecon	2000-05-21	43	
5	5,001	Aztecon	2000-05-18	601	
6	5,001	Aztecon	1998-07-28	528	
7	5,002	CIT Campeche	2005-07-03	24	
8	5,002	CIT Campeche	2003-12-16	582	
9	5,002	CIT Campeche	2001-09-09	603	
10	5,003	Mexico sin ti no estamos completos	2005-04-30	576	
11	5,003	Mexico sin ti no estamos completos	1998-09-12	530	
12	5,003	Mexico sin ti no estamos completos	1998-01-12	202	
13	5,004	Educando en Coahuila	2002-11-14	453	
14	5,004	Educando en Coahuila	1999-12-18	263	
15	5,004	Educando en Coahuila	0000-00-00	152	
16	5,005	Infonavit Durango	2004-02-28	521	
17	5,005	Infonavit Durango	2004-02-07	503	
18	5,005	Infonavit Durango	2003-10-12	71	
19	5,005	Infonavit Durango	2000-06-30	583	
20	5,006	Reconstrucción del templo de Guadalupe	2004-09-29	324	

Uso de expresiones.

En álgebra relacional los argumentos de una proyección deben ser columnas. Sin embargo en una sentencia SELECT es posible incluir expresiones aritméticas o funciones que usen como argumentos de las columnas de las tablas involucradas o bien constantes. Los operadores son:

+ Suma
 - Resta
 * Producto
 / División

Las columnas con expresiones pueden renombrarse escribiendo después de la expresión un alias que puede ser un nombre arbitrario; si el alias contiene caracteres que no sean números o letras (espacios, puntos etc.) debe encerrarse entre comillas dobles (" nuevo nombre"). Para SQL Server también pueden utilizarse comillas simples.

Operadores de cadena

El operador LIKE se aplica a datos de tipo cadena y se usa para buscar registros, es capaz de hallar coincidencias dentro de una cadena bajo un patrón dado.

También contamos con el operador comodín (%), que coincide con cualquier cadena que tenga cero o más caracteres. Este puede usarse tanto de prefijo como sufijo.

```
SELECT * FROM productos where Descripcion LIKE 'Si%'
```

¿Qué resultado obtienes?

Explica que hace el símbolo '%'.

¿Qué sucede si la consulta fuera : LIKE 'Si' ?

¿Qué resultado obtienes?

Explica a qué se debe este comportamiento.

```
1 SELECT * FROM materiales where Descripcion LIKE 'Si%'
```

#	clave	descripcion	precio	impuesto
1	1,120	Sillar rosa	100	10
2	1,130	Sillar gris	110	11

El símbolo % indica que este puede ser sustituido por cualquier cadena de caracteres de cualquier longitud, incluyendo caracteres nulos.

Si la consulta fuera LIKE 'Si', solo se buscarían esos 2 caracteres específicamente y no como parte de un patrón, por lo que no recibiríamos un resultado.

Sin embargo, tenemos otros operadores como [], ^ y _.

[] - Busca coincidencia dentro de un intervalo o conjunto dado. Estos caracteres se pueden utilizar para buscar coincidencias de patrones como sucede con LIKE.

^ - En contra parte, este operador coincide con cualquier caracter que no se encuentre dentro del intervalo o del conjunto especificado.

_ - El operador _ o guion bajo, se utiliza para coincidir con un caracter de una comparación de cadenas.

Ahora explica el comportamiento, función y resultado de cada una de las siguientes consultas:

```
SELECT RFC FROM Entregan WHERE RFC LIKE '[A-D]';
SELECT RFC FROM Entregan WHERE RFC LIKE '[^A]';
SELECT Numero FROM Entregan WHERE Numero LIKE '___6';
```

Operadores Lógicos.

Los operadores lógicos comprueban la verdad de una condición, al igual que los operadores de comparación, devuelven un tipo de dato booleano (True, false o unknown).

ALL Es un operador que compara un valor numérico con un conjunto de valores representados por un subquery. La condición es verdadera cuando todo el conjunto cumple la condición.

ANY o SOME Es un operador que compara un valor numérico con un conjunto de valores. La condición es verdadera cuando al menos un dato del conjunto cumple la condición.

La sintaxis para ambos es: valor_numerico {operador de comparación} subquery

BETWEEN Es un operador para especificar intervalos. Una aplicación muy común de dicho operador son intervalos de fechas.

```
SELECT Clave,RFC,Numero,Fecha,Cantidad
FROM Entregan
WHERE Numero Between 5000 and 5010;
```

¿Cómo filtrarías rangos de fechas?

```

1 SELECT Clave,RFC,Numero,Fecha,Cantidad
2 FROM Entregan
3 WHERE Numero Between 5000 and 5010;

```

#	Clave	RFC	Numero	Fecha	Cantidad
1	1,000	AAAA800101	5,000	2001-12-13	165
2	1,010	BBBB800101	5,001	1998-07-28	528
3	1,020	CCCC800101	5,002	2003-12-16	582
4	1,030	DDDD800101	5,003	1998-01-12	202
5	1,040	EEEE800101	5,004	1999-12-18	263
6	1,050	FFFF800101	5,005	2004-02-07	503
7	1,060	GGGG800101	5,006	2004-09-29	324
8	1,070	HHHH800101	5,007	2006-11-21	2
9	1,080	AAAA800101	5,008	2005-04-03	86
10	1,090	BBBB800101	5,009	1997-03-13	73
11	1,090	BBBB800101	5,010	1998-11-17	421
12	1,100	CCCC800101	5,009	2000-12-07	466
13	1,100	CCCC800101	5,010	2001-11-19	699
14	1,110	DDDD800101	5,008	2005-06-03	337
15	1,120	EEEE800101	5,007	2003-11-21	692
16	1,130	FFFF800101	5,006	2000-04-13	562
17	1,140	GGGG800101	5,005	2000-06-30	583
18	1,150	HHHH800101	5,004	2002-11-14	453
19	1,200	EEEE800101	5,000	2003-03-15	177
20	1,210	FFFF800101	5,001	2000-05-21	43

Filtraría rangos de fecha usando las instrucciones YEAR(), MONTH() y DAY() junto con BETWEEN.

EXISTS Se utiliza para especificar dentro de una subconsulta la existencia de ciertas filas.

```

SELECT RFC,Cantidad, Fecha,Numero
FROM [Entregan]
WHERE [Numero] Between 5000 and 5010 AND
Exists ( SELECT [RFC]
FROM [Proveedores]
WHERE RazonSocial LIKE 'La%' and [Entregan].[RFC] = [Proveedores].[RFC] )

```

¿Qué hace la consulta?

¿Qué función tiene el paréntesis () después de EXISTS?

```

1 SELECT RFC,Cantidad, Fecha,Numero
2 FROM Entregan
3 WHERE Numero Between 5000 and 5010 AND
4 Exists ( SELECT RFC
5 FROM Proveedores
6 WHERE RazonSocial LIKE 'La%' and Entregan.RFC = Proveedores.RFC )

```

#	RFC	Cantidad	Fecha	Numero
1	AAAA800101	165	2001-12-13	5,000
2	CCCC800101	582	2003-12-16	5,002
3	AAAA800101	86	2005-04-03	5,008
4	CCCC800101	466	2000-12-07	5,009
5	CCCC800101	699	2001-11-19	5,010
6	AAAA800101	152	0000-00-00	5,004
7	CCCC800101	460	2001-04-09	5,006
8	CCCC800101	631	2001-07-28	5,009
9	AAAA800101	382	1999-04-07	5,000
10	AAAA800101	116	2005-04-21	5,010
11	CCCC800101	603	2001-09-09	5,002
12	CCCC800101	278	1999-05-05	5,008

La consulta recupera el RFC, cantidad, fecha y numero de la tabla entregan cuando el numero está entre 5000 y 5010 y el RFC sea de una razón social que comience con La y continúe con cualquier secuencia de caracteres.

El paréntesis después de exists encapsula a la subconsulta sobre la que se ejecutará la función para determinar lo que se mostrará en la consulta original.

Tomando de base la consulta anterior del EXISTS, realiza el query que devuelva el mismo resultado, pero usando el operador IN

NOT Simplemente niega la entrada de un valor booleano.

Tomando de base la consulta anterior del EXISTS, realiza el query que devuelva el mismo resultado, pero usando el operador NOT IN Realiza un ejemplo donde apliques algún operador : ALL, SOME o ANY.

```

1 SELECT RFC,Cantidad, Fecha,Numero
2 FROM Entregan
3 WHERE Numero Between 5000 and 5010 AND
4 RFC IN ( SELECT RFC
5 FROM Proveedores
6 WHERE RazonSocial LIKE 'La%' and Entregan.RFC = Proveedores.RFC )

```

#	RFC	Cantidad	Fecha	Numero
1	AAAA800101	165	2001-12-13	5,000
2	CCCC800101	582	2003-12-16	5,002
3	AAAA800101	86	2005-04-03	5,008
4	CCCC800101	466	2000-12-07	5,009
5	CCCC800101	699	2001-11-19	5,010
6	AAAA800101	152	0000-00-00	5,004
7	CCCC800101	460	2001-04-09	5,006
8	CCCC800101	631	2001-07-28	5,009
9	AAAA800101	382	1999-04-07	5,000
10	AAAA800101	116	2005-04-21	5,010
11	CCCC800101	603	2001-09-09	5,002
12	CCCC800101	278	1999-05-05	5,008

```

1 SELECT RFC, cantidad, fecha, numero
2 FROM entregan
3 WHERE numero BETWEEN 5000 AND 5010 AND
4 RFC NOT IN (SELECT rfc
5 FROM proveedores
6 WHERE RazonSocial NOT LIKE 'La%' AND entregan.RFC =

```

#	RFC	cantidad	fecha	numero
1	AAAA800101	165	2001-12-13	5,000
2	CCCC800101	582	2003-12-16	5,002
3	AAAA800101	86	2005-04-03	5,008
4	CCCC800101	466	2000-12-07	5,009
5	CCCC800101	699	2001-11-19	5,010
6	AAAA800101	152	0000-00-00	5,004
7	CCCC800101	460	2001-04-09	5,006
8	CCCC800101	631	2001-07-28	5,009
9	AAAA800101	382	1999-04-07	5,000
10	AAAA800101	116	2005-04-21	5,010
11	CCCC800101	603	2001-09-09	5,002
12	CCCC800101	278	1999-05-05	5,008

El Operador TOP, es un operador que recorre la entrada, un query, y sólo devuelve el primer número o porcentaje específico de filas basado en un criterio de ordenación si es posible.

¿Qué hace la siguiente sentencia? Explica por qué.

```
SELECT TOP 2 * FROM Proyectos
```

```
1 SELECT *
2 FROM proyectos LIMIT 2
3
```

proyectos (2r x 2c)		
#	numero	denominacion
1	5,000	Vamos Mexico
2	5,001	Aztecnn

¿Qué sucede con la siguiente consulta? Explica por qué.

```
SELECT TOP Numero FROM Proyectos
```

```
1 SELECT numero
2 FROM proyectos LIMIT 1
3
```

proyectos (1r x 1c)	
#	numero
1	5,000

Creación de vistas

La sentencia:

```
Create view nombrevista (nombrecolumna1 , nombrecolumna2 ,..., nombrecolumna3 )
as select...
```

Permite definir una vista. Una vista puede pensarse como una consulta etiquetada con un nombre, ya que en realidad al referirnos a una vista el DBMS realmente ejecuta la consulta asociada a ella, pero por la cerradura del álgebra relacional, una consulta puede ser vista como una nueva relación o tabla, por lo que es perfectamente válido emitir la sentencia:

```
select * from nombrevista
```

¡Como si nombrevista fuera una tabla!

Comprueba lo anterior, creando vistas para cinco de las consultas que planteaste anteriormente en la práctica . Posteriormente revisa cada vista creada para comprobar que devuelve el mismo resultado.

```

1 CREATE VIEW vista1(clave,descripcion)
2 AS SELECT clave, descripcion
3 FROM materiales NATURAL JOIN entregan
4 WHERE fecha LIKE '2000%';
5
6 SELECT *
7 FROM vista1

```

vista1 (12r x 2c)		
#	clave	descripcion
1	1,020	Varilla 3/17
2	1,050	Varilla 4/34
3	1,100	Block
4	1,130	Sillar gris
5	1,130	Sillar gris
6	1,140	Cantera blanca
7	1,210	Recubrimiento P1028
8	1,310	Tubería 3.6
9	1,360	Pintura C1010
10	1,390	Pintura B1021
11	1,410	Pintura B1021
12	1,430	Pintura B1022

En el reporte incluye la sentencia, una muestra de la salida (dos o tres renglones) y el número de renglones que SQL Server reporta al final de la consulta.

Los materiales (clave y descripción) entregados al proyecto "México sin ti no estamos completos".

Los materiales (clave y descripción) que han sido proporcionados por el proveedor "Acme tools".

El RFC de los proveedores que durante el 2000 entregaron en promedio cuando menos 300 materiales.

El Total entregado por cada material en el año 2000.

La Clave del material más vendido durante el 2001. (se recomienda usar una vista intermedia para su solución)

Productos que contienen el patrón 'ub' en su nombre.

Denominación y suma del total a pagar para todos los proyectos.

Denominación, RFC y RazonSocial de los proveedores que se suministran materiales al proyecto Televisa en acción que no se encuentran apoyando al proyecto Educando en Coahuila (Solo usando vistas).

Denominación, RFC y RazonSocial de los proveedores que se suministran materiales al proyecto Televisa en acción que no se encuentran apoyando al proyecto Educando en Coahuila (Sin usar vistas, utiliza not in, in o exists).

Costo de los materiales y los Materiales que son entregados al proyecto Televisa en acción cuyos proveedores también suministran materiales al proyecto Educando en Coahuila.

Reto: Usa solo el operador NOT IN en la consulta anterior (No es parte de la entrega).

Nombre del material, cantidad de veces entregados y total del costo de dichas entregas por material de todos los proyectos.

```

1 SELECT clave, descripcion
2 FROM materiales NATURAL JOIN entregan NATURAL JOIN proyectos
3 WHERE denominacion = "México sin ti no estamos completos"

```

materiales (3r x 2c)			
#	clave		descripcion
1	1,030		Varilla 4/33
2	1,230		Cemento
3	1,430		Pintura B1022

```

1 SELECT clave, descripcion
2 FROM materiales NATURAL JOIN entregan NATURAL JOIN proveedores
3 WHERE razonsocial = "Acme tools"

```

materiales (0r × 2c)

#	clave	descripcion
---	-------	-------------

```

1 SELECT rfc
2 FROM entregan NATURAL JOIN proveedores
3 WHERE YEAR(fecha) = 2000
4 GROUP BY rfc
5 HAVING AVG(cantidad) >= 300

```

entregan (3r × 1c)

#	rfc
1	BBBB800101
2	FFFF800101
3	GGGG800101

```

1 SELECT descripcion, SUM(cantidad)
2 FROM entregan NATURAL JOIN materiales
3 WHERE YEAR(fecha) = 2000
4 GROUP BY clave
5

```

materiales (11r × 2c)

#	descripcion	SUM(cantidad)
1	Varilla 3/17	8
2	Varilla 4/34	623
3	Block	466
4	Sillar gris	625
5	Cantera blanca	583
6	Recubrimiento P1028	43
7	Tubería 3.6	72
8	Pintura C1010	265
9	Pintura B1021	107
10	Pintura B1021	601
11	Pintura B1022	13

```

1 SELECT clave
2 FROM entregan NATURAL JOIN materiales
3 WHERE YEAR(fecha) = 2001
4 GROUP BY clave
5 HAVING SUM(cantidad)
6 ORDER BY cantidad DESC
7 LIMIT 1
8

```

entregan (1r × 1c)	
#	clave
1	1,100

```

1 SELECT clave
2 FROM entregan NATURAL JOIN materiales
3 WHERE YEAR(fecha) = 2001
4 GROUP BY clave
5 HAVING SUM(cantidad)
6 ORDER BY cantidad DESC
7 LIMIT 1
8

```

entregan (1r × 1c)	
#	clave
1	1,100

```

1 SELECT denominacion, SUM(precio*cantidad)
2 FROM materiales NATURAL JOIN entregan NATURAL JOIN proyectos
3 GROUP BY numero
4

```

proyectos (20r × 2c)		
#	denominacion	SUM(precio*cantidad)
1	Vamos Mexico	106,730
2	Aztecon	146,595
3	CIT Campeche	157,755
4	Mexico sin ti no estamos completos	260,290
5	Educando en Coahuila	620,610
6	Infonavit Durango	321,135
7	Reconstrucción del templo de Guadalupe	220,580
8	Construcción de plaza Magnolias	122,969
9	Televisa en acción	99,848
10	Disco Atlantic	158,100
11	Construcción de Hospital Infantil	144,295
12	Remodelación de aulas del IPP	225,835
13	Restauración de instalaciones del CEA	846,380
14	Reparación de la plaza Sonora	527,485
15	Remodelación de Soriana	284,450

Resultado #1 (3r x 3c)

```
1 SELECT SUM(precio*cantidad), descripcion
2 FROM proveedores NATURAL JOIN entregan NATURAL JOIN proyectos NATURAL JOIN materiales
3 WHERE denominacion = "Televisa en acción" AND rfc IN (SELECT rfc
4 FROM proveedores NATURAL JOIN entregan NATURAL JOIN proyectos
5 WHERE denominacion = "Educando en Coahuila")
```

```
1 SELECT denominacion, descripcion, COUNT(cantidad), precio*cantidad
2 FROM entregan NATURAL JOIN proyectos NATURAL JOIN materiales
3 GROUP BY denominacion, descripcion
4
```

#	denominacion	descripcion	COUNT(cantidad)	precio*cantidad
1	Ampliación de la carretera a la huasteca	Cantera rosa	1	230,040
2	Ampliación de la carretera a la huasteca	Pintura C1010	1	45,500
3	Ampliación de la carretera a la huasteca	Recubrimiento P1010	1	78,320
4	Ampliación de la carretera a la huasteca	Tubería 4.5	1	168,500
5	Ampliación de la carretera a la huasteca	Varilla 4/33	1	42,775
6	Aztecon	Pintura B1021	1	75,125
7	Aztecon	Recubrimiento P1028	1	10,750
8	Aztecon	Varilla 4/32	1	60,720
9	CIT Campeche	Pintura C1012	1	75,375
10	CIT Campeche	Recubrimiento P1037	1	6,720
11	CIT Campeche	Varilla 3/17	1	75,660
12	CIT Yucatan	Cantera gris	1	554,180
13	CIT Yucatan	Recubrimiento P1019	1	140,400
14	CIT Yucatan	Tubería 3.8	1	70,720
15	CIT Yucatan	Varilla 3/18	1	86,400