



Tecnológico de Monterrey

Tecnológico de Monterrey

Campus Querétaro

Construcción de software y toma de decisiones (Gpo 501)

Ejercicio:

“Laboratorio 20: Consultas en SQL”

Angel Mauricio Ramírez Herrera - A01710158

Profesores:

Enrique Alfonso Calderón Balderas

Denisse L. Maldonado Flores

Alejandro Fernández Vilchis 22 de Abril de 2024

Consulta de un tabla completa

Algebra relacional.
materiales

SQL
select * from materiales

	Clave	Descripcion	Precio
►	1000	Varilla 3/16	100
	1010	Varilla 4/32	115
	1020	Varilla 3/17	130
	1030	Varilla 4/33	145
	1040	Varilla 3/18	160
	1050	Varilla 4/34	175
	1060	Varilla 3/19	190
	1070	Varilla 4/35	205
	1080	Ladrillos rojos	50
	1090	Ladrillos grises	35
	1100	Block	30
	1110	Megablock	40
	1120	Sillar rosa	100
	1130	Sillar gris	110
	1140	Cantera blanca	200

Selección

Algebra relacional.
SL{clave=1000}(materiales)

SQL
select * from materiales
where clave=1000

	Clave	Descripcion	Precio
►	1000	Varilla 3/16	100
•	NULL	NULL	NULL

Proyección

Algebra relacional.

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

SQL

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

	clave	rfc	fecha
▶	1270	DDDD800101	1997-09-03 00:00:00
	1090	BBBB800101	1998-01-03 00:00:00
	1290	FFFF800101	1998-01-08 00:00:00
	1430	DDDD800101	1998-01-09 00:00:00
	1170	BBBB800101	1998-02-04 00:00:00
	1140	GGGG800101	1998-02-07 00:00:00
	1030	DDDD800101	1998-02-21 00:00:00
	1070	HHHH800101	1998-02-23 00:00:00
	1120	EEEE800101	1998-03-12 00:00:00
	1420	CCCC800101	1998-04-07 00:00:00
	1030	DDDD800101	1998-04-09 00:00:00
	1250	BBBB800101	1998-05-08 00:00:00
	1400	AAAA800101	1998-06-05 00:00:00
	1090	BBBB800101	1998-06-06 00:00:00
	1330	BBBB800101	1998-06-12 00:00:00

Reunión Natural

Algebra relacional.

entregan JN materiales

SQL

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

	Clave	Descripcion	Precio	Fecha	Cantidad	Clave	Numero	RFC
►	1000	Varilla 3/16	100	1998-07-08 00:00:00	165	1000	5000	AAAA800101
	1000	Varilla 3/16	100	1999-08-08 00:00:00	254	1000	5019	AAAA800101
	1000	Varilla 3/16	100	2000-04-06 00:00:00	7	1000	5019	AAAA800101
	1010	Varilla 4/32	115	2000-05-03 00:00:00	528	1010	5001	BBBB800101
	1010	Varilla 4/32	115	2000-11-10 00:00:00	667	1010	5018	BBBB800101
	1010	Varilla 4/32	115	2002-03-29 00:00:00	523	1010	5018	BBBB800101
	1020	Varilla 3/17	130	1999-02-04 00:00:00	8	1020	5017	CCCC800101
	1020	Varilla 3/17	130	2001-05-04 00:00:00	478	1020	5017	CCCC800101
	1020	Varilla 3/17	130	2001-07-29 00:00:00	582	1020	5002	CCCC800101
	1030	Varilla 4/33	145	1998-02-21 00:00:00	202	1030	5003	DDDD800101
	1030	Varilla 4/33	145	1998-04-09 00:00:00	139	1030	5016	DDDD800101
	1030	Varilla 4/33	145	2000-11-05 00:00:00	295	1030	5016	DDDD800101
	1040	Varilla 3/18	160	1999-12-11 00:00:00	263	1040	5004	EEEE800101
	1040	Varilla 3/18	160	2000-06-10 00:00:00	546	1040	5015	EEEE800101
	1040	Varilla 3/18	160	2002-07-12 00:00:00	540	1040	5015	EEEE800101

Si algún material no ha se ha entregado ¿Aparecería en el resultado de esta consulta?
No aparecería el resultado, ya que su Id no estaría relacionado como FK con entregan

Reunión con criterio específico

Algebra relacional.

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

SQL

select * from entregan,proyectos

where entregan.numero < = proyectos.numero

Fecha	Cantidad	Clave	Numero	RFC	Numero	Denominacion
1998-07-08 00:00:00	165	1000	5000	AAAA800101	5000	Vamos Mexico
2000-03-05 00:00:00	177	1200	5000	EEEE800101	5000	Vamos Mexico
2002-03-12 00:00:00	382	1400	5000	AAAA800101	5000	Vamos Mexico
1998-07-08 00:00:00	165	1000	5000	AAAA800101	5001	Aztecón
2000-03-05 00:00:00	177	1200	5000	EEEE800101	5001	Aztecón
2002-03-12 00:00:00	382	1400	5000	AAAA800101	5001	Aztecón
1999-11-05 00:00:00	43	1210	5001	FFFF800101	5001	Aztecón
2000-02-05 00:00:00	601	1410	5001	BBBB800101	5001	Aztecón
2000-05-03 00:00:00	528	1010	5001	BBBB800101	5001	Aztecón
1998-07-08 00:00:00	165	1000	5000	AAAA800101	5002	CIT Campeche
2000-03-05 00:00:00	177	1200	5000	EEEE800101	5002	CIT Campeche
2002-03-12 00:00:00	382	1400	5000	AAAA800101	5002	CIT Campeche
1999-11-05 00:00:00	43	1210	5001	FFFF800101	5002	CIT Campeche
2000-02-05 00:00:00	601	1410	5001	BBBB800101	5002	CIT Campeche
2000-05-03 00:00:00	528	1010	5001	BBBB800101	5002	CIT Campeche

Unión (se ilustra junto con selección)

Algebra relacional.

SL{clave=1450}(entregan) UN SL{clave=1300}(entregan)

SQL

(select * from entregan where clave=1450)

union

(select * from entregan where clave=1300)

	Fecha	Cantidad	Clave	Numero	RFC
►	2002-06-10 00:00:00	521	1300	5005	GGGG800101
	2003-01-08 00:00:00	119	1300	5010	GGGG800101
	2003-02-02 00:00:00	457	1300	5005	GGGG800101

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

SELECT *

FROM entregan

WHERE clave = 1450 OR clave = 1300;

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)
```

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.

Producto cartesiano

Algebra relacional.

$entregan \times materiales$

SQL

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

	Fecha	Cantidad	Clave	Numero	RFC	Clave	Descripcion	Precio
►	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1430	Pintura B1022	125
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1420	Pintura C1012	125
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1410	Pintura B1021	125
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1400	Pintura C1011	125
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1390	Pintura B1021	125
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1380	Pintura C1011	725
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1370	Pintura B1020	125
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1360	Pintura C1010	125
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1350	Tubería 3.8	260
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1340	Tubería 4.5	250
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1330	Tubería 3.7	240
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1320	Tubería 4.4	230
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1310	Tubería 3.6	220
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1300	Tubería 4.3	210
	1997-09-03 00:00:00	546	1270	5008	DDDD800101	1290	Tubería 3.5	200

Result 8 x

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

El número de tuplas en este resultado, está definido por el producto cartesiano por dos tablas, si la tabla entregan tiene n tuplas y la tabla materiales tiene m tuplas, entonces el resultado es de $n * m$ tuplas.

Construcción de consultas a partir de una especificación

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

```
select m.descripcion from materiales as m
join entregan as e
on m.clave = e.clave
where year(e.fecha) = 2000;
```

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

	descripcion
►	Varilla 3/16
	Varilla 4/32
	Varilla 4/32
	Varilla 4/33
	Varilla 3/18
	Varilla 4/34
	Varilla 3/19
	Varilla 3/19
	Varilla 3/19
	Varilla 4/35
	Ladrillos g...
	Block
	Megablock
	Cantera r...
	Recubrimi...

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?

Porque se entregaron materiales con la misma descripción en 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?

descripcion
Varilla 3/16
Varilla 4/32
Varilla 4/33
Varilla 3/18
Varilla 4/34
Varilla 3/19
Varilla 4/35
Ladrillos grises
Block
Megablock
Cantera rosa
Recubrimient...
Recubrimient...
Arena

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.

```
select p.*, e.fecha, e.cantidad from proyectos as p
join entregan as e
on p.numero = e.numero
order by p.numero asc, e.fecha desc;
```

	Numero	Denominacion	fecha	cantidad
►	5000	Vamos Mexico	2002-03-12 00:00:00	382
	5000	Vamos Mexico	2000-03-05 00:00:00	177
	5000	Vamos Mexico	1998-07-08 00:00:00	165
	5001	Aztecón	2000-05-03 00:00:00	528
	5001	Aztecón	2000-02-05 00:00:00	601
	5001	Aztecón	1999-11-05 00:00:00	43
	5002	CIT Campeche	2003-02-01 00:00:00	24
	5002	CIT Campeche	2001-07-29 00:00:00	582
	5002	CIT Campeche	1998-04-07 00:00:00	603
	5003	Mexico sin ti no estamos completos	2003-01-06 00:00:00	530
	5003	Mexico sin ti no estamos completos	1999-09-02 00:00:00	576
	5003	Mexico sin ti no estamos completos	1998-02-21 00:00:00	202
	5004	Educando en Coahuila	2003-09-01 00:00:00	270
	5004	Educando en Coahuila	2003-01-12 00:00:00	152
	5004	Educando en Coahuila	2001-08-10 00:00:00	453

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?

Se obtienen todos los productos con una descripción que inicie con Si

Explica que hace el símbolo '%'.

El símbolo % significa que puede haber 0 o más caracteres antes o después de la expresión buscada.

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

Sólo devuelve las descripciones que solo almacenen Si

¿Qué resultado obtienes?

Se obtienen las descripciones que coinciden con Si y que solo almacenen eso

Explica a qué se debe este comportamiento.

Este comportamiento es debido a que no se tienen comodines, por lo que no se especifica si se cuentan con caracteres antes o después de Si.

Otro operador de cadenas es el de concatenación, (+, +=) este operador concatena dos o más cadenas de caracteres.

Su sintaxis es : Expresión + Expresión.

Un ejemplo de su uso, puede ser: Un ejemplo de su uso, puede ser:

```
SELECT (Apellido + ', ' + Nombre) as Nombre FROM Personas;
```

```
DECLARE @foo varchar(40);
```

```
DECLARE @bar varchar(40);
```

```
SET @foo = '¿Que resultado';
```

```
SET @bar = ' ¿¿¿?? ';
```

```
SET @foo += ' obtienes?';
```

```
PRINT @foo + @bar;
```

¿Qué resultado obtienes de ejecutar el siguiente código?

Se debería de obtener una frase.

¿Para qué sirve DECLARE?

Sirve para definir una variable y asignarle un tipo de dato

¿Cuál es la función de @foo?

No es una función, es el nombre de la variable

¿Que realiza el operador SET?

Sirve para asignar valores a variables

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';
```

El primero sirve para obtener los RFC que comienzan con una letra que está entre la A y la D, seguido de otras letras.

El segundo sirve para obtener un RFC que no comiencen con la letra A.

El tercero es para obtener un número con tres números antes del 6.

Operadores compuestos.

Los operadores compuestos ejecutan una operación y establecen un valor.

+ = (Suma igual)

- = (Restar igual)

* = (Multiplicar igual)

/ = (Dividir igual)

% = (Módulo igual)

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?

Haciendo lo mismo con BETWEEN, pero especificando si se quieren rangos en años, meses o días o con la fecha completa.

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] )
```

	RFC	Cantidad	Fecha	Numero
▶	AAAA800101	116	1998-06-05 00:00:00	5010
	AAAA800101	165	1998-07-08 00:00:00	5000
	AAAA800101	86	1999-01-12 00:00:00	5008
	AAAA800101	382	2002-03-12 00:00:00	5000
	AAAA800101	152	2003-01-12 00:00:00	5004
	CCCC800101	603	1998-04-07 00:00:00	5002
	CCCC800101	460	1999-05-10 00:00:00	5006
	CCCC800101	631	1999-08-09 00:00:00	5009
	CCCC800101	278	2000-08-02 00:00:00	5008
	CCCC800101	466	2000-08-06 00:00:00	5009
	CCCC800101	444	2001-02-12 00:00:00	5008
	CCCC800101	582	2001-07-29 00:00:00	5002
	CCCC800101	699	2001-09-10 00:00:00	5010
	CCCC800101	523	2002-05-07 00:00:00	5009
	CCCC800101	2	2003-02-10 00:00:00	5009

¿Qué hace la consulta?

Obtiene una tabla con el RFC, Cantidad, Fecha y Numero de la tabla Entregan donde el Numero se encuentra entre 5000 y 5010 y existen Proveedores con una razón social que su descripción empiece con La.

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

Se utiliza para realizar una subconsulta.

IN Especifica si un valor dado tiene coincidencias con algún valor de una subconsulta.

NOTA: Se utiliza dentro del WHERE pero debe contener un parametro. Ejemplo: Where proyecto.id IN Lista_de_Proyectos_Subquery

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

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

	RFC	Cantidad	Fecha	Numero
►	AAAA800101	116	1998-06-05 00:00:00	5010
	AAAA800101	165	1998-07-08 00:00:00	5000
	AAAA800101	86	1999-01-12 00:00:00	5008
	AAAA800101	382	2002-03-12 00:00:00	5000
	AAAA800101	152	2003-01-12 00:00:00	5004
	CCCC800101	603	1998-04-07 00:00:00	5002
	CCCC800101	460	1999-05-10 00:00:00	5006
	CCCC800101	631	1999-08-09 00:00:00	5009
	CCCC800101	278	2000-08-02 00:00:00	5008
	CCCC800101	466	2000-08-06 00:00:00	5009
	CCCC800101	444	2001-02-12 00:00:00	5008
	CCCC800101	582	2001-07-29 00:00:00	5002
	CCCC800101	699	2001-09-10 00:00:00	5010
	CCCC800101	523	2002-05-07 00:00:00	5009

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.

```

SELECT RFC,Cantidad, Fecha,Numero
FROM Entregan
WHERE Numero Between 5000 and 5010
AND NOT EXISTS (
SELECT RFC
FROM Proveedores
WHERE RazonSocial LIKE 'La%' and Entregan.RFC = Proveedores.RFC );

```

	RFC	Cantidad	Fecha	Numero
▶	DDDD800101	546	1997-09-03 00:00:00	5008
	BBBB800101	421	1998-01-03 00:00:00	5010
	FFFF800101	132	1998-01-08 00:00:00	5009
	DDDD800101	13	1998-01-09 00:00:00	5007
	GGGG800101	651	1998-02-07 00:00:00	5005
	DDDD800101	202	1998-02-21 00:00:00	5003
	HHHH800101	2	1998-02-23 00:00:00	5007
	EEEE800101	167	1998-03-12 00:00:00	5007
	BBBB800101	690	1998-05-08 00:00:00	5010
	BBBB800101	612	1998-06-06 00:00:00	5010
	FFFF800101	336	1999-01-07 00:00:00	5006
	DDDD800101	506	1999-03-10 00:00:00	5007
	FFFF800101	673	1999-05-06 00:00:00	5006
	DDDD800101	576	1999-09-02 00:00:00	5003
	FFFF800101	43	1999-11-05 00:00:00	5001

```

SELECT RFC,Cantidad, Fecha,Numero
FROM Entregan
WHERE Numero Between 5000 and 5010
AND RFC NOT IN (
SELECT RFC
FROM Proveedores
WHERE RazonSocial LIKE 'La%' and Entregan.RFC = Proveedores.RFC );

```


	RFC	Cantidad	Fecha	Numero
▶	DDDD800101	546	1997-09-03 00:00:00	5008
	BBBB800101	421	1998-01-03 00:00:00	5010
	FFFF800101	132	1998-01-08 00:00:00	5009
	DDDD800101	13	1998-01-09 00:00:00	5007
	GGGG800101	651	1998-02-07 00:00:00	5005
	DDDD800101	202	1998-02-21 00:00:00	5003
	HHHH800101	2	1998-02-23 00:00:00	5007
	EEEE800101	167	1998-03-12 00:00:00	5007
	BBBB800101	690	1998-05-08 00:00:00	5010
	BBBB800101	612	1998-06-06 00:00:00	5010
	FFFF800101	336	1999-01-07 00:00:00	5006
	DDDD800101	506	1999-03-10 00:00:00	5007
	FFFF800101	673	1999-05-06 00:00:00	5006

```

SELECT *
FROM entregan
WHERE cantidad < ALL (
    SELECT precio
    FROM materiales
);

```

	Fecha	Cantidad	Clave	Numero	RFC
▶	1998-01-09 00:00:00	13	1430	5007	DDDD800101
	1998-02-23 00:00:00	2	1070	5007	HHHH800101
	1999-02-04 00:00:00	8	1020	5017	CCCC800101
	1999-02-10 00:00:00	11	1340	5016	CCCC800101
	2000-04-01 00:00:00	1000	5019	AAAA800101	
	2003-02-01 00:00:00	24	1220	5002	GGGG800101
	2003-02-10 00:00:00	2	1260	5009	CCCC800101
●	NULL	NULL	NULL	NULL	NULL

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

```

Selecciona las primeras dos filas de la consulta, ya que se especifica que tome las primeras dos consultas con TOP 2.

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

```
SELECT TOP Numero FROM Proyectos
```

Selecciona el primer número del proyecto ya que solo se consulta la primera fila con TOP.

Modificando la estructura de un tabla existente.

Agrega a la tabla materiales la columna PorcentajImpuesto con la instrucción:

```
ALTER TABLE materiales ADD PorcentajImpuesto NUMERIC(6,2);
```

A fin de que los materiales tengan un impuesto, les asignaremos impuestos ficticios basados en sus claves con la instrucción:

```
UPDATE materiales SET PorcentajImpuesto = 2*clave/1000;
```

esto es, a cada material se le asignará un impuesto igual al doble de su clave dividida entre diez.

Revisa la tabla de materiales para que compruebes lo que hicimos anteriormente.

¿Qué consulta usarías para obtener el importe de las entregas es decir, el total en dinero de lo entregado, basado en la cantidad de la entrega y el precio del material y el impuesto asignado?

```
select e.cantidad, m.precio, m.porcentajeimpuesto, e.cantidad * (m.precio +  
(m.porcentajeimpuesto / 100 * m.precio)) AS total from entregan as e  
join materiales as m  
on e.clave = m.clave;
```

	cantidad	precio	porcentajeimpuesto	total
▶	165	100	2.00	16830.000000
	254	100	2.00	25908.000000
	7	100	2.00	714.000000
	528	115	2.02	61946.544000
	667	115	2.02	78254.441000
	523	115	2.02	61359.929000
	8	130	2.04	1061.216000
	170	130	2.04	62407.656000

Result 23 x

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.

La parte (nombrecolumna1,nombrecolumna2,..de la sentencia create view puede ser omitida si no hay ambigüedad en los nombres de las columnas de la sentencia select asociada.

Importante: Las vistas no pueden incluir la cláusula order by.

```
create view getTotal as
```

```
select e.cantidad, m.precio, m.porcentajeimpuesto, e.cantidad * (m.precio +  
m.porcentajeimpuesto) AS total from entregan as e  
join materiales as m  
on e.clave = m.clave;
```

```
select * from getTotal;
```

```
create view getProjectInfo as  
select p.*, e.fecha, e.cantidad from proyectos as p  
join entregan as e  
on p.numero = e.numero;
```

```
select * from getProjectInfo;
```

```
create view getDistinctDesc as  
select distinct m.descripcion from materiales as m  
join entregan as e  
on m.clave = e.clave  
where year(e.fecha) = 2000;
```

```
select * from getDistinctDesc;
```

```
create view getRFC as  
SELECT RFC,Cantidad, Fecha,Numero  
FROM Entregan  
WHERE Numero Between 5000 and 5010  
AND RFC IN (  
SELECT RFC  
FROM Proveedores  
WHERE RazonSocial LIKE 'La%' and Entregan.RFC = Proveedores.RFC );
```

```
select * from getRFC;
```

```
create view getAllMats as  
select * from materiales;
```

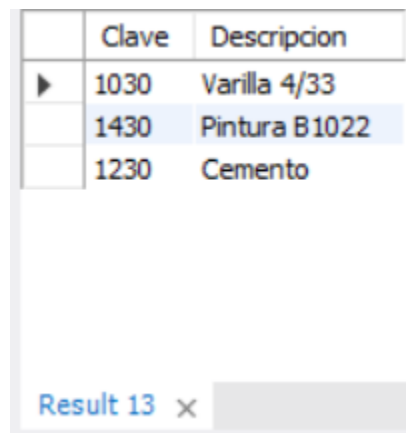
```
select * from getAllMats;
```

A continuación se te dan muchos enunciados de los cuales deberás generar su correspondiente consulta.

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".

```
SELECT m.Clave, m.Descripcion
FROM Materiales AS m
JOIN Entregan AS e ON m.Clave = e.Clave
JOIN Proyectos AS p ON e.Numero = p.Numero
WHERE p.Denominacion = 'Mexico sin ti no estamos completos';
```



	Clave	Descripcion
▶	1030	Varilla 4/33
	1430	Pintura B1022
	1230	Cemento

Result 13 x

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

```
SELECT m.Clave, m.Descripcion
FROM Materiales AS m
JOIN Entregan AS e ON m.Clave = e.Clave
JOIN Proveedores AS p ON e.RFC = p.RFC
WHERE p.RazonSocial = 'Acme tools';
```

Clave	Descripcion
Result 14 x	

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

```
SELECT e.RFC
FROM Entregan AS e
WHERE YEAR(e.Fecha) = 2000
GROUP BY e.RFC
HAVING AVG(e.Cantidad) >= 300;
```

Result Grid	
	RFC
▶	BBBB800101
	CCCC800101
	DDDD800101
	EEEE800101
	FFFF800101
	GGGG800101
	HHHH800101
Entregan 15 x	

El Total entregado por cada material en el año 2000.

```
SELECT e.Clave, SUM(e.Cantidad) AS Total_Entregado
FROM Entregan AS e
WHERE YEAR(e.Fecha) = 2000
GROUP BY e.Clave;
```

Result Grid			Filter Rows:
	Clave	Total_Entregado	
▶	1000	7	
	1010	1195	
	1030	295	
	1040	546	
	1050	503	
	1060	1063	
	1070	516	
	1000	72	
Result 16			×

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

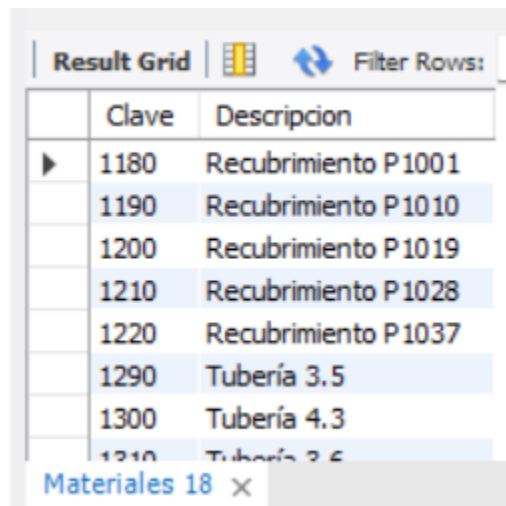
```
CREATE VIEW Total_Entregado_2001 AS
SELECT e.Clave, SUM(e.Cantidad) AS Total_Entregado
FROM Entregan AS e
WHERE YEAR(e.Fecha) = 2001
GROUP BY e.Clave;
```

```
SELECT Clave
FROM Total_Entregado_2001
WHERE Total_Entregado = (SELECT MAX(Total_Entregado) FROM Total_Entregado_2001);
```

Result Grid		Filter Rows:
	Clave	
▶	1020	
Total_Entregado_2001 17		×

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

```
SELECT Clave, Descripcion
FROM Materiales
WHERE Descripcion LIKE '%ub%';
```

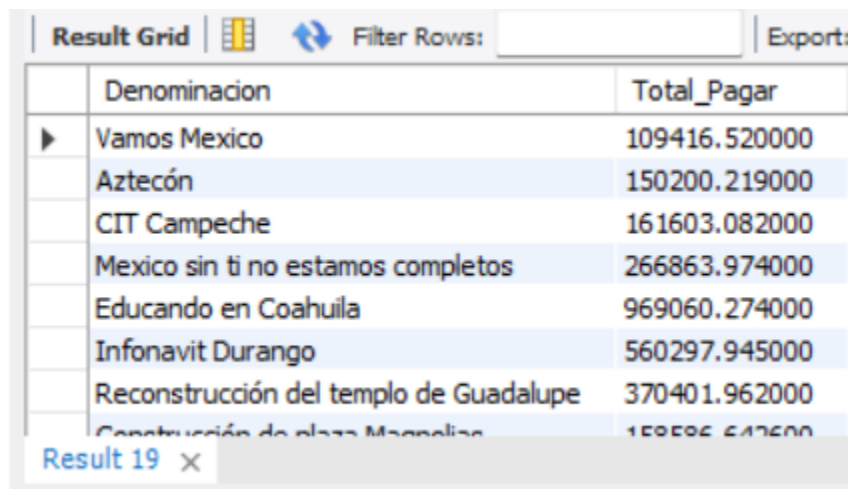


The screenshot shows a 'Result Grid' window with a table containing two columns: 'Clave' and 'Descripcion'. The table lists several materials, all of which have 'ub' in their description. The rows are numbered 1 through 10. The last row is highlighted in blue. Below the table, there is a tab labeled 'Materiales 18' with a close button 'x'.

	Clave	Descripcion
▶	1180	Recubrimiento P1001
	1190	Recubrimiento P1010
	1200	Recubrimiento P1019
	1210	Recubrimiento P1028
	1220	Recubrimiento P1037
	1290	Tubería 3.5
	1300	Tubería 4.3
	1310	Tubería 3.5

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

```
SELECT pr.Denominacion, regan AS e
JOIN Proyectos AS pr ON e.Numero = pr.Numero
JOIN Materiales AS m ON e.Clave = m.Clave
GROUP BY pr.Denominacion;
```



The screenshot shows a 'Result Grid' window with a table containing two columns: 'Denominacion' and 'Total_Pagar'. The table lists the total payment for various projects. The rows are numbered 1 through 10. The last row is highlighted in blue. Below the table, there is a tab labeled 'Result 19' with a close button 'x'.

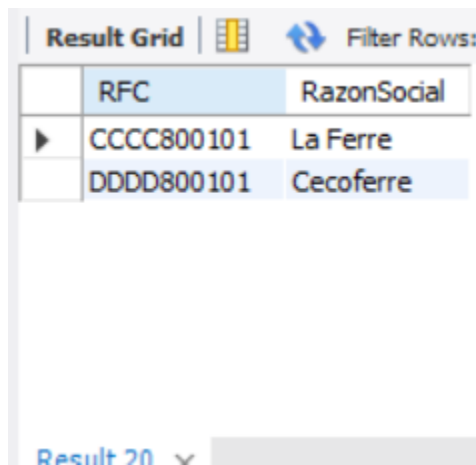
	Denominacion	Total_Pagar
▶	Vamos Mexico	109416.520000
	Aztecón	150200.219000
	CIT Campeche	161603.082000
	Mexico sin ti no estamos completos	266863.974000
	Educando en Coahuila	969060.274000
	Infonavit Durango	560297.945000
	Reconstrucción del templo de Guadalupe	370401.962000
	Construcción de plaza Magdalena	158596.617600

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).

```
CREATE VIEW Proveedores_Televisa AS
SELECT DISTINCT e.RFC
FROM Entregan AS e
JOIN Proyectos AS p ON e.Numero = p.Numero
WHERE p.Denominacion = 'Televisa en acción';
```

```
CREATE VIEW Proveedores_Educando_Coahuila AS
SELECT DISTINCT e.RFC
FROM Entregan AS e
JOIN Proyectos AS p ON e.Numero = p.Numero
WHERE p.Denominacion = 'Educando en Coahuila';
```

```
SELECT p.RFC, p.RazonSocial
FROM Proveedores AS p
JOIN Proveedores_Televisa AS pt ON p.RFC = pt.RFC
LEFT JOIN Proveedores_Educando_Coahuila AS pec ON p.RFC = pec.RFC
WHERE pec.RFC IS NULL;
```



The screenshot shows a 'Result Grid' window with a 'Filter Rows' button. The grid contains two columns: 'RFC' and 'RazonSocial'. There are two data rows: one with RFC 'CCCC800101' and RazonSocial 'La Ferre', and another with RFC 'DDDD800101' and RazonSocial 'Cecoferre'. At the bottom, it says 'Result 20' with a dropdown arrow.

	RFC	RazonSocial
▶	CCCC800101	La Ferre
	DDDD800101	Cecoferre

Result 20 ▼

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).

```
SELECT p.RFC, p.RazonSocial
FROM Proveedores AS p
```

```

JOIN Entregan AS e ON p.RFC = e.RFC
JOIN Proyectos AS pr ON e.Numero = pr.Numero
WHERE pr.Denominacion = 'Televisa en acción'
AND p.RFC NOT IN (
    SELECT p2.RFC
    FROM Proveedores AS p2
    JOIN Entregan AS e2 ON p2.RFC = e2.RFC
    JOIN Proyectos AS pr2 ON e2.Numero = pr2.Numero
    WHERE pr2.Denominacion = 'Educando en Coahuila'
);

```

Result Grid			Filter Rows:
	RFC	RazonSocial	
▶	CCCC800101	La Ferre	
	CCCC800101	La Ferre	
	DDDD800101	Cecoferre	
	DDDD800101	Cecoferre	

Result 21 x

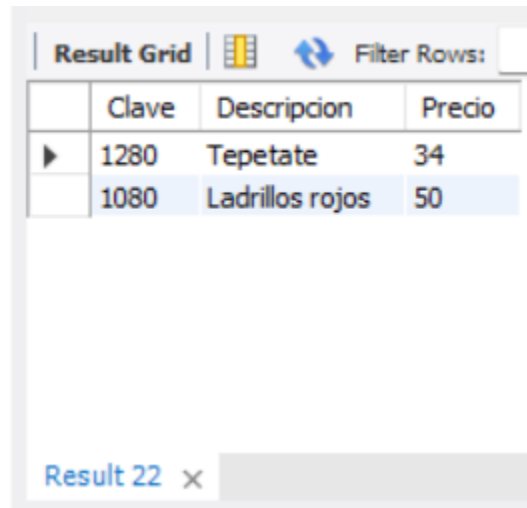
Costo de los materiales y los Materiales que son entregados al proyecto Televisa en acción cuyos proveedores también suministran materiales al proyecto Educando en Coahuila.

```

SELECT m.Clave, m.Descripcion, m.Precio
FROM Materiales AS m
JOIN Entregan AS e ON m.Clave = e.Clave
JOIN Proyectos AS pr ON e.Numero = pr.Numero
JOIN Proveedores AS p ON e.RFC = p.RFC
WHERE pr.Denominacion = 'Televisa en acción'
AND p.RFC IN (
    SELECT p2.RFC
    FROM Proveedores AS p2
    JOIN Entregan AS e2 ON p2.RFC = e2.RFC
    JOIN Proyectos AS pr2 ON e2.Numero = pr2.Numero
    WHERE pr2.Denominacion = 'Educando en Coahuila'
);

```

);



	Clave	Descripcion	Precio
▶	1280	Tepetate	34
	1080	Ladrillos rojos	50

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.

Muchas de estas consultas requieren la utilización de funciones agregadas...

Se recomienda que revises nuevamente la lectura.