101 - Crear tabla a partir de otra (select - into)

Primer problema:

Un supermercado almacena los datos de sus empleados en una tabla denominada "empleados" y en una
tabla llamada "sucursales" los códigos y ciudades de las diferentes sucursales.

1- Elimine las tablas "empleados" y "sucursales" si existen:

```
 if object_id('empleados')is not null
  drop table empleados;
 if object_id('sucursales')is not null
  drop table sucursales;
```

2- Cree la tabla "sucursales":

```
 create table sucursales(
  codigo int identity,
  ciudad varchar(30) not null,
  primary key(codigo)
 );
```

3- Cree la tabla "empleados":

```
 create table empleados(
  documento char(8) not null,
  nombre varchar(30) not null,
  domicilio varchar(30),
  seccion varchar(20),
  sueldo decimal(6,2),
  codigosucursal int,
  primary key(documento),
  constraint FK_empleados_sucursal
   foreign key (codigosucursal)
   references sucursales(codigo)
   on update cascade
 );
```

4- Ingrese algunos registros para ambas tablas:

```
 insert into sucursales values('Cordoba');
 insert into sucursales values('Villa Maria');
 insert into sucursales values('Carlos Paz');
 insert into sucursales values('Cruz del Eje');

 insert into empleados values('22222222','Ana Acosta','Avellaneda
111','Secretaria',500,1);
 insert into empleados values('23333333','Carlos Caseros','Colon
222','Sistemas',800,1);
 insert into empleados values('24444444','Diana Dominguez','Dinamarca
333','Secretaria',550,2);
 insert into empleados values('25555555','Fabiola Fuentes','Francia
444','Sistemas',750,2);
 insert into empleados values('26666666','Gabriela Gonzalez','Guemes
555','Secretaria',580,3);
 insert into empleados values('27777777','Juan Juarez','Jujuy 777','Secretaria',500,4);
 insert into empleados values('28888888','Luis Lopez','Lules 888','Sistemas',780,4);
```

```
 insert into empleados values('29999999','Maria Morales','Marina
999','Contaduria',670,4);
```

5- Realice un join para mostrar todos los datos de "empleados" incluyendo la ciudad de la sucursal:

```
 select documento,nombre,domicilio,seccion,sueldo,ciudad
  from empleados
  join sucursales on codigosucursal=codigo;
```

6-Cree una tabla llamada "secciones" que contenga las secciones de la empresa (primero elimínela, si existe):

```
 if object_id('secciones') is not null
  drop table secciones;

 select distinct seccion as nombre
  into secciones
  from empleados;
```

7- Recupere la información de "secciones":

```
 select *from secciones;
```

3 registros.

8- Se necesita una nueva tabla llamada "sueldosxseccion" que contenga la suma de los sueldos de los empleados por sección. Primero elimine la tabla, si existe:

```
 if object_id('sueldosxseccion') is not null
  drop table sueldosxseccion;

 select seccion, sum(sueldo) as total
  into sueldosxseccion
  from empleados
  group by seccion;
```

9- Recupere los registros de la nueva tabla:

```
 select *from sueldosxseccion;
```

10- Se necesita una tabla llamada "maximossueldos" que contenga los mismos campos que "empleados" y guarde los 3 empleados con sueldos más altos. Primero eliminamos, si existe, la tabla "maximossueldos":

```
 if object_id('maximossueldos') is not null
  drop table maximossueldos;

  select top 3 *
  into maximossueldos
  from empleados
  order by sueldo;
```

11- Vea los registros de la nueva tabla:

```
 select *from maximossueldos;
```

12- Se necesita una nueva tabla llamada "sucursalCordoba" que contenga los nombres y sección de los
empleados de la ciudad de Córdoba. En primer lugar, eliminamos la tabla, si existe. Luego, consulte
las tablas "empleados" y "sucursales" y guarde el resultado en la nueva tabla:

```
 if object_id('sucursalCordoba') is not null
  drop table sucursalCordoba;

 select nombre,ciudad
  into sucursalCordoba
  from empleados
  join sucursales
  on codigosucursal=codigo
  where ciudad='Cordoba';
```

13- Consulte la nueva tabla:

```
 select *from sucursalCordoba;
```

103 - Vistas

Primer problema:

Un club dicta cursos de distintos deportes. Almacena la información en varias tablas. El director no quiere que los empleados de administración conozcan la estructura de las tablas ni
algunos datos de los profesores y socios, por ello se crean vistas a las cuales tendrán acceso.

1- Elimine las tablas y créelas nuevamente:

```
 if object_id('inscriptos') is not null
  drop table inscriptos;
 if object_id('socios') is not null
  drop table socios;
 if object_id('profesores') is not null
  drop table profesores;
 if object_id('cursos') is not null
  drop table cursos;

 create table socios(
  documento char(8) not null,
  nombre varchar(40),
  domicilio varchar(30),
  constraint PK_socios_documento
   primary key (documento)
 );

 create table profesores(
  documento char(8) not null,
  nombre varchar(40),
  domicilio varchar(30),
  constraint PK_profesores_documento
   primary key (documento)
 );
```

```
 create table cursos(
  numero tinyint identity,
  deporte varchar(20),
  dia varchar(15),
   constraint CK_inscriptos_dia check (dia
in('lunes','martes','miercoles','jueves','viernes','sabado')),
  documentoprofesor char(8),
  constraint PK_cursos_numero
   primary key (numero),
 );

 create table inscriptos(
  documentosocio char(8) not null,
  numero tinyint not null,
  matricula char(1),
  constraint CK_inscriptos_matricula check (matricula in('s','n')),
  constraint PK_inscriptos_documento_numero
   primary key (documentosocio,numero)
 );
```

2- Ingrese algunos registros para todas las tablas:

```
 insert into socios values('30000000','Fabian Fuentes','Caseros 987');
 insert into socios values('31111111','Gaston Garcia','Guemes 65');
 insert into socios values('32222222','Hector Huerta','Sucre 534');
 insert into socios values('33333333','Ines Irala','Bulnes 345');

 insert into profesores values('22222222','Ana Acosta','Avellaneda 231');
 insert into profesores values('23333333','Carlos Caseres','Colon 245');
 insert into profesores values('24444444','Daniel Duarte','Sarmiento 987');
 insert into profesores values('25555555','Esteban Lopez','Sucre 1204');

 insert into cursos values('tenis','lunes','22222222');
 insert into cursos values('tenis','martes','22222222');
 insert into cursos values('natacion','miercoles','22222222');
 insert into cursos values('natacion','jueves','23333333');
 insert into cursos values('natacion','viernes','23333333');
 insert into cursos values('futbol','sabado','24444444');
 insert into cursos values('futbol','lunes','24444444');
 insert into cursos values('basquet','martes','24444444');

 insert into inscriptos values('30000000',1,'s');
 insert into inscriptos values('30000000',3,'n');
 insert into inscriptos values('30000000',6,null);
 insert into inscriptos values('31111111',1,'s');
 insert into inscriptos values('31111111',4,'s');
 insert into inscriptos values('32222222',8,'s');
```

3- Elimine la vista "vista_club" si existe:

```
 if object_id('vista_club') is not null drop view vista_club;
```

4- Cree una vista en la que aparezca el nombre y documento del socio, el deporte, el día y el nombre

del profesor.

5- Muestre la información contenida en la vista.

6- Realice una consulta a la vista donde muestre la cantidad de socios inscriptos en cada deporte
ordenados por cantidad.

7- Muestre (consultando la vista) los cursos (deporte y día) para los cuales no hay inscriptos.

8- Muestre los nombres de los socios que no se han inscripto en ningún curso (consultando la vista)

9- Muestre (consultando la vista) los profesores que no tienen asignado ningún deporte aún.

10- Muestre (consultando la vista) el nombre y documento de los socios que deben matrículas.

11- Consulte la vista y muestre los nombres de los profesores y los días en que asisten al club para
dictar sus clases.

12- Muestre la misma información anterior pero ordenada por día.

13- Muestre todos los socios que son compañeros en tenis los lunes.

14- Elimine la vista "vista_inscriptos" si existe y créela para que muestre la cantidad de
inscriptos por curso, incluyendo el número del curso, el nombre del deporte y el día.

15- Consulte la vista:
```
 select *from vista_inscriptos;
```

107 - Vistas (with check option)

Primer problema:

Una empresa almacena la información de sus clientes en dos tablas llamadas "clientes" y "ciudades".

1- Elimine las tablas, si existen:
```
 if object_id('clientes') is not null
  drop table clientes;
 if object_id('ciudades') is not null
  drop table ciudades;
```

2- Cree las tablas:
```
 create table ciudades(
  codigo tinyint identity,
  nombre varchar(20),
  constraint PK_ciudades
```

```
   primary key (codigo)
 );

 create table clientes(
  nombre varchar(20),
  apellido varchar(20),
  documento char(8),
  domicilio varchar(30),
  codigociudad tinyint
   constraint FK_clientes_ciudad
    foreign key (codigociudad)
   references ciudades(codigo)
   on update cascade
 );

3- Ingrese algunos registros:
 insert into ciudades values('Cordoba');
 insert into ciudades values('Carlos Paz');
 insert into ciudades values('Cruz del Eje');
 insert into ciudades values('La Falda');

 insert into clientes values('Juan','Perez','22222222','Colon 1123',1);
 insert into clientes values('Karina','Lopez','23333333','San Martin 254',2);
 insert into clientes values('Luis','Garcia','24444444','Caseros 345',1);
 insert into clientes values('Marcos','Gonzalez','25555555','Sucre 458',3);
 insert into clientes values('Nora','Torres','26666666','Bulnes 567',1);
 insert into clientes values('Oscar','Luque','27777777','San Martin 786',4);

4- Elimine la vista "vista_clientes" si existe:
 if object_id('vista_clientes') is not null
  drop view vista_clientes;

5- Cree la vista "vista_clientes" para que recupere el nombre, apellido, documento,
domicilio, el
código y nombre de la ciudad a la cual pertenece, de la ciudad de "Cordoba"
empleando "with check
option".

6- Consulte la vista:
 select * from vista_clientes;

7- Actualice el apellido de un cliente a través de la vista.

8- Verifique que la modificación se realizó en la tabla:
 select * from clientes;

9- Intente cambiar la ciudad de algún registro.
Mensaje de error.
```

108 - Vistas (modificar datos de una tabla a través de vistas)

Primer problema:

Un club dicta cursos de distintos deportes. Almacena la información en varias tablas.

1- Elimine las tabla "inscriptos", "socios" y "cursos", si existen:

```
 if object_id('inscriptos') is not null
  drop table inscriptos;
 if object_id('socios') is not null
  drop table socios;
 if object_id('cursos') is not null
  drop table cursos;
```

2- Cree las tablas:

```
 create table socios(
  documento char(8) not null,
  nombre varchar(40),
  domicilio varchar(30),
  constraint PK_socios_documento
   primary key (documento)
 );

 create table cursos(
  numero tinyint identity,
  deporte varchar(20),
  dia varchar(15),
   constraint CK_inscriptos_dia check (dia
in('lunes','martes','miercoles','jueves','viernes','sabado')),
  profesor varchar(20),
  constraint PK_cursos_numero
   primary key (numero),
 );

 create table inscriptos(
  documentosocio char(8) not null,
  numero tinyint not null,
  matricula char(1),
  constraint PK_inscriptos_documento_numero
   primary key (documentosocio,numero),
  constraint FK_inscriptos_documento
   foreign key (documentosocio)
   references socios(documento)
   on update cascade,
  constraint FK_inscriptos_numero
   foreign key (numero)
   references cursos(numero)
   on update cascade
  );
```

3- Ingrese algunos registros para todas las tablas:

```
 insert into socios values('30000000','Fabian Fuentes','Caseros 987');
 insert into socios values('31111111','Gaston Garcia','Guemes 65');
 insert into socios values('32222222','Hector Huerta','Sucre 534');
 insert into socios values('33333333','Ines Irala','Bulnes 345');

 insert into cursos values('tenis','lunes','Ana Acosta');
```

```
 insert into cursos values('tenis','martes','Ana Acosta');
 insert into cursos values('natacion','miercoles','Ana Acosta');
 insert into cursos values('natacion','jueves','Carlos Caseres');
 insert into cursos values('futbol','sabado','Pedro Perez');
 insert into cursos values('futbol','lunes','Pedro Perez');
 insert into cursos values('basquet','viernes','Pedro Perez');

 insert into inscriptos values('30000000',1,'s');
 insert into inscriptos values('30000000',3,'n');
 insert into inscriptos values('30000000',6,null);
 insert into inscriptos values('31111111',1,'s');
 insert into inscriptos values('31111111',4,'s');
 insert into inscriptos values('32222222',1,'s');
 insert into inscriptos values('32222222',7,'s');
```

4- Realice un join para mostrar todos los datos de todas las tablas, sin repetirlos:

```
 select documento,nombre,domicilio,c.numero,deporte,dia, profesor,matricula
  from socios as s
  join inscriptos as i
  on s.documento=documentosocio
  join cursos as c
  on c.numero=i.numero;
```

5- Elimine, si existe, la vista "vista_cursos":

```
 if object_id('vista_cursos') is not null
  drop view vista_cursos;
```

6- Cree la vista "vista_cursos" que muestre el número, deporte y día de todos los cursos.

7- Consulte la vista ordenada por deporte.

8- Ingrese un registro en la vista "vista_cursos" y vea si afectó a "cursos".
Puede realizarse el ingreso porque solamente afecta a una tabla base.

9- Actualice un registro sobre la vista y vea si afectó a la tabla "cursos".
Puede realizarse la actualización porque solamente afecta a una tabla base.

10- Elimine un registro de la vista para el cual no haya inscriptos y vea si afectó a "cursos".
Puede realizarse la eliminación porque solamente afecta a una tabla base.

11- Intente eliminar un registro de la vista para el cual haya inscriptos.
No lo permite por la restricción "foreign key".

12- Elimine la vista "vista_inscriptos" si existe y créela para que muestre el documento y nombre
del socio, el numero de curso, el deporte y día de los cursos en los cuales está inscripto.

13- Intente ingresar un registro en la vista.
No lo permite porque la modificación afecta a más de una tabla base.

14- Actualice un registro de la vista.
Lo permite porque la modificación afecta a una sola tabla base.

15- Vea si afectó a la tabla "socios":
```
 select * from socios;
```

16- Intente actualizar el documento de un socio.
No lo permite por la restricción.

17- Intente eliminar un registro de la vista.
No lo permite porque la vista incluye varias tablas.

109 - Vistas modificar (alter view)

Primer problema:
Un club dicta cursos de distintos deportes. Almacena la información en varias tablas.
1- Elimine las tabla "inscriptos", "socios" y "cursos", si existen:
```
 if object_id('inscriptos') is not null
  drop table inscriptos;
 if object_id('socios') is not null
  drop table socios;
 if object_id('cursos') is not null
  drop table cursos;
```

2- Cree las tablas:
```
 create table socios(
  documento char(8) not null,
  nombre varchar(40),
  domicilio varchar(30),
  constraint PK_socios_documento
   primary key (documento)
 );

 create table cursos(
  numero tinyint identity,
  deporte varchar(20),
  dia varchar(15),
   constraint CK_inscriptos_dia check (dia
in('lunes','martes','miercoles','jueves','viernes','sabado')),
  profesor varchar(20),
  constraint PK_cursos_numero
   primary key (numero),
 );

 create table inscriptos(
  documentosocio char(8) not null,
  numero tinyint not null,
  matricula char(1),
  constraint PK_inscriptos_documento_numero
   primary key (documentosocio,numero),
  constraint FK_inscriptos_documento
```

```
   foreign key (documentosocio)
   references socios(documento)
   on update cascade,
  constraint FK_inscriptos_numero
   foreign key (numero)
   references cursos(numero)
   on update cascade
  );
```

3- Ingrese algunos registros para todas las tablas:

```
 insert into socios values('30000000','Fabian Fuentes','Caseros 987');
 insert into socios values('31111111','Gaston Garcia','Guemes 65');
 insert into socios values('32222222','Hector Huerta','Sucre 534');
 insert into socios values('33333333','Ines Irala','Bulnes 345');

 insert into cursos values('tenis','lunes','Ana Acosta');
 insert into cursos values('tenis','martes','Ana Acosta');
 insert into cursos values('natacion','miercoles','Ana Acosta');
 insert into cursos values('natacion','jueves','Carlos Caseres');
 insert into cursos values('futbol','sabado','Pedro Perez');
 insert into cursos values('futbol','lunes','Pedro Perez');
 insert into cursos values('basquet','viernes','Pedro Perez');

 insert into inscriptos values('30000000',1,'s');
 insert into inscriptos values('30000000',3,'s');
 insert into inscriptos values('30000000',6,null);
 insert into inscriptos values('31111111',1,'n');
 insert into inscriptos values('31111111',4,'s');
 insert into inscriptos values('32222222',1,'n');
 insert into inscriptos values('32222222',7,'n');
```

4- Elimine la vista "vista_deudores" si existe:

```
 if object_id('vista_deudores') is not null
  drop view vista_deudores;
```

5- Cree la vista "vista_deudores" que muestre el documento y nombre del socio, el deporte, el día y
la matrícula, de todas las inscripciones no pagas colocando "with check option".

6- Consulte la vista:

```
 select * from vista_deudores;
```

7- Veamos el texto de la vista.

8- Intente actualizar a "s" la matrícula de una inscripción desde la vista.
No lo permite por la opción "with check option".

9- Modifique el documento de un socio mediante la vista.

10- Vea si se alteraron las tablas referenciadas en la vista:

```
 select * from socios;
 select * from inscriptos;
```

11- Modifique la vista para que muestre el domicilio, coloque la opción de encriptación y omita
"with check option".

12- Consulte la vista para ver si se modificó:
 select * from vista_deudores;
Aparece el nuevo campo.

13- Vea el texto de la vista.
No lo permite porque está encriptada.

14- Actualice la matrícula de un inscripto.
Si se permite porque la opción "with check option" se quitó de la vista.

15- Consulte la vista:
 select * from vista_empleados;
Note que el registro modificado ya no aparece porque la matrícula está paga.

16- Elimine la vista "vista_socios" si existe:
 if object_id('vista_socios') is not null
  drop view vista_socios;

17- Cree la vista "vista_socios" que muestre todos los campos de la tabla "socios".

18- Consulte la vista.

19- Agregue un campo a la tabla "socios".

20- Consulte la vista "vista_socios".
El nuevo campo agregado a "socios" no aparece, pese a que la vista indica que muestre todos los
campos de dicha tabla.

21- Altere la vista para que aparezcan todos los campos.

22- Consulte la vista.

110 - Lenguaje de control de flujo (case)

Primer problema:
Una empresa registra los datos de sus empleados en una tabla llamada "empleados".
1- Elimine la tabla "empleados" si existe:
 if object_id('empleados') is not null
  drop table empleados;

2- Cree la tabla:
 create table empleados(
  documento char(8) not null,
  nombre varchar(30) not null,
  sexo char(1),
  fechanacimiento datetime,

```
  fechaingreso datetime,
  cantidadhijos tinyint,
  sueldo decimal(5,2),
  primary key(documento)
);
```

3- Ingrese algunos registros:

```
 insert into empleados values ('22333111','Juan Perez','m','1970-05-10','1987-04-05',2,550);
 insert into empleados values ('25444444','Susana Morales','f','1975-11-06','1990-04-06',0,650);
 insert into empleados values ('20111222','Hector Pereyra','m','1965-03-25','1997-04-12',3,510);
 insert into empleados values ('30000222','Luis LUque','m','1980-03-29','1999-11-06',1,700);
 insert into empleados values ('20555444','Laura Torres','f','1965-12-22','2003-11-06',3,400);
 insert into empleados values ('30000234','Alberto Soto','m','1989-10-10','1999-11-06',2,420);
 insert into empleados values ('20125478','Ana Gomez','f','1976-09-21','1998-11-06',3,350);
 insert into empleados values ('24154269','Ofelia Garcia','f','1974-05-12','1990-11-06',0,390);
 insert into empleados values ('30415426','Oscar Torres','m','1978-05-02','1997-11-06',1,400);
```

4- Es política de la empresa festejar cada fin de mes, los cumpleaños de todos los empleados que
cumplen ese mes. Si los empleados son de sexo femenino, se les regala un ramo de rosas, si son de
sexo masculino, una corbata. La secretaria de la Gerencia necesita saber cuántos ramos de rosas y
cuántas corbatas debe comprar para el mes de mayo.

5- Además, si el empleado cumple 10,20,30,40... años de servicio, se le regala una placa
recordatoria. La secretaria de Gerencia necesita saber la cantidad de años de servicio que cumplen
los empleados que ingresaron en el mes de abril para encargar dichas placas.

6- La empresa paga un sueldo adicional por hijos a cargos. Para un sueldo menor o igual a $500 el
salario familiar por hijo es de $200, para un sueldo superior, el monto es de $100 por hijo. Muestre
el nombre del empleado, el sueldo básico, la cantidad de hijos a cargo, el valor del salario por
hijo, el valor total del salario familiar y el sueldo final con el salario familiar incluido de todos los empleados.

111 - Lenguaje de control de flujo (if)

Primer problema:

Una empresa registra los datos de sus empleados en una tabla llamada "empleados".

1- Elimine la tabla "empleados" si existe:

```
 if object_id('empleados') is not null
  drop table empleados;
```

2- Cree la tabla:

```
 create table empleados(
  documento char(8) not null,
  nombre varchar(30) not null,
  sexo char(1),
  fechanacimiento datetime,
  sueldo decimal(5,2),
  primary key(documento)
);
```

3- Ingrese algunos registros:

```
 insert into empleados values ('22333111','Juan Perez','m','1970-05-10',550);
 insert into empleados values ('25444444','Susana Morales','f','1975-11-06',650);
 insert into empleados values ('20111222','Hector Pereyra','m','1965-03-25',510);
 insert into empleados values ('30000222','Luis LUque','m','1980-03-29',700);
 insert into empleados values ('20555444','Laura Torres','f','1965-12-22',400);
 insert into empleados values ('30000234','Alberto Soto','m','1989-10-10',420);
 insert into empleados values ('20125478','Ana Gomez','f','1976-09-21',350);
 insert into empleados values ('24154269','Ofelia Garcia','f','1974-05-12',390);
 insert into empleados values ('30415426','Oscar Torres','m','1978-05-02',400);
```

4- Es política de la empresa festejar cada fin de mes, los cumpleaños de todos los empleados que
cumplen ese mes. Si los empleados son de sexo femenino, se les regala un ramo de rosas, si son de
sexo masculino, una corbata. La secretaria de la Gerencia necesita saber cuántos ramos de rosas y
cuántas corbatas debe comprar para el mes de mayo.

Segundo problema:

Un teatro con varias salas guarda la información de las entradas vendidas en una tabla llamada
"entradas".

1- Elimine la tabla, si existe:

```
 if object_id('entradas') is not null
  drop table entradas;
```

2- Cree la tabla:

```
 create table entradas(
  sala tinyint,
  fechahora datetime,
  capacidad smallint,
  entradasvendidas smallint,
  primary key(sala,fechahora)
 );
```

3- Ingrese algunos registros:

```
 insert into entradas values(1,'2006-05-10 20:00',300,50);
 insert into entradas values(1,'2006-05-10 23:00',300,250);
 insert into entradas values(2,'2006-05-10 20:00',400,350);
 insert into entradas values(2,'2006-05-11 20:00',400,380);
 insert into entradas values(2,'2006-05-11 23:00',400,400);
 insert into entradas values(3,'2006-05-12 20:00',350,350);
 insert into entradas values(3,'2006-05-12 22:30',350,100);
 insert into entradas values(4,'2006-05-12 20:00',250,0);
```

4- Muestre, si existen, todas las funciones para la cuales hay entradas disponibles, sino un mensaje
que indique que están agotadas.

112 - Variables de usuario

Primer problema:

Un profesor almacena el documento y nombre de sus alumnos en una tabla llamada "alumnos" y en otra
tabla llamada "notas" almacena las notas de los mismos.

1- Elimine las tablas, si existen:

```
 if object_id('alumnos') is not null
  drop table alumnos;
 if object_id('notas') is not null
  drop table notas;
```

2- Créelas con los campos necesarios. Agregue una restricción "primary key" para el campo
"documento" y una restricción "foreign key" para que en la tabla "notas" el documento del alumno
haga referencia al documento de la tabla "alumnos":

```
 create table alumnos(
  documento char(8) not null
   constraint CK_alumnos_documento check (documento like '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),
  nombre varchar(30),
  constraint PK_alumnos
  primary key(documento)
 );

 create table notas(
  documento char(8) not null,
  nota decimal(4,2)
   constraint CK_notas_nota check (nota between 0 and 10),
  constraint FK_notas_documento
  foreign key(documento)
  references alumnos(documento)
  on update cascade
 );
```

3- Ingrese algunos registros:

```
 insert into alumnos values('30111111','Ana Algarbe');
 insert into alumnos values('30222222','Bernardo Bustamante');
 insert into alumnos values('30333333','Carolina Conte');
 insert into alumnos values('30444444','Diana Dominguez');
 insert into alumnos values('30555555','Fabian Fuentes');
 insert into alumnos values('30666666','Gaston Gonzalez');

 insert into notas values('30111111',5.1);
 insert into notas values('30222222',7.8);
 insert into notas values('30333333',4);
 insert into notas values('30444444',2.5);
 insert into notas values('30666666',9.9);
 insert into notas values('30111111',7.3);
 insert into notas values('30222222',8.9);
 insert into notas values('30444444',6);
 insert into notas values('30666666',8);
```

4- Declare una variable llamada "@documento" de tipo "char(8)" y vea su contenido:

```
 declare @documento char(8)
 select @documento;
```

5- Intente usar la variable "@documento" para almacenar el documento del alumno con la nota más alta:

```
 select @documento= documento from notas
  where nota=(select max(nota) from notas);
```

No se puede porque la variable fue declarada en otro lote de sentencias y no es reconocida.

6- Declare una variable llamada "@documento" de tipo "char(8)" y almacene en ella el documento del
alumno con la nota más alta, luego recupere el nombre del alumno:

```
 declare @documento char(8)
 select @documento= documento from notas
  where nota=(select max(nota) from notas)
 select nombre from alumnos where documento=@documento;
```

113 - Tipos de datos text, ntext y image (remplazados por varchar(max), nvarchar(max) y varbinary(max))

Primer problema:

En una página web se guardan los siguientes datos de las visitas: número de visita, nombre, mail,
pais, fecha.

1- Elimine la tabla "visitas", si existe:

```
 if object_id('visitas') is not null
  drop table visitas;
```

2- Créela con la siguiente estructura:

```
 create table visitas (
  numero int identity,
  nombre varchar(30),
```

```
  mail varchar(50),
  pais varchar (20),
  fecha datetime
  constraint DF_visitas_fecha default getdate(),
  comentarios text
  constraint DF_visitas_comentarios default 'Ninguno',
  constraint PK_visitas
  primary key(numero)
);
```

3- Ingrese algunos registros:

```
 insert into visitas values ('Ana Maria
Lopez','AnaMaria@hotmail.com','Argentina','2006-10-10 10:10',null);
 insert into visitas values ('Gustavo
Gonzalez','GustavoGGonzalez@hotmail.com','Chile','2006-10-10 21:30',default);
 insert into visitas values ('Fabiola
Martinez','MartinezFabiola@hotmail.com','Mexico',default,'Excelente página');
 insert into visitas values ('Mariano Perez','PerezM@hotmail.com','Argentina','2006-11-
11 14:30','Muy buena y divertida');
```

4- Recupere todos los registros:

```
 select * from visitas;
```

5- Cuente la cantidad de visitas que tienen comentarios.
Retorna 3.

6- Intente agregar una restricción "check" al campo de tipo "text":

```
 alter table visitas
  add constraint CK_comentarios
  check (comentarios not like '[0-9]%');
```

No lo permite.

7- Intente alterar el campo de tipo "text" para que no acepte valores nulos:

```
 alter table visitas
  alter column comentarios text not null;
```

No lo permite.

8- Elimine la regla llamada "RG_texto" (si existe):

```
 if object_id('RG_texto') is not null
  drop rule RG_texto;
```

9- Cree la regla "RG_texto" que no permita que el primer caracter sea un dígito:

```
 create rule RG_texto
 as @valor not like '[0-9]%';
```

10- Asóciela al campo "nombre":

```
 exec sp_bindrule RG_texto, 'visitas.nombre';
```

11- Intente asociarla al campo "comentarios":

```
 exec sp_bindrule RG_texto, 'visitas.comentarios';
```

No lo permite.

12- Quite la restricción "default" del campo "comentarios":

```
 alter table visitas
  drop DF_visitas_comentarios;

13- Ingrese un registro con valores por defecto y recupere todos los registros:
 insert into visitas default values;
 select * from visitas;

14- Elimine el valor predeterminado llamado "VP_SinComentarios":
 if object_id('VP_Sincomentarios') is not null
  drop default VP_Sincomentarios;

15- Cree un valor por defecto que almacene el valor "Sin comentarios":
 create default VP_Sincomentarios
  as 'Sin comentarios';

16- Asócielo al campo "comentarios":
 exec sp_bindefault VP_Sincomentarios, 'visitas.comentarios';

17- Ingrese un registro con valores por defecto y recupere todos los registros:
 insert into visitas default values;
 select * from visitas;

18- Borre la tabla y creela nuevamente con el campo comentarios con tipo
varchar(max)
```

115 - Tipo de dato text - ntext e image (leer)

Primer problema:

```
En una página web se guardan los siguientes datos de las visitas: número de visita,
nombre, mail,
pais, fecha.
1- Elimine la tabla "visitas", si existe:
 if object_id('visitas') is not null
  drop table visitas;

2- Créela con la siguiente estructura:
 create table visitas (
  numero int identity,
  nombre varchar(30),
  mail varchar(50),
  pais varchar (20),
  fecha datetime
  constraint DF_visitas_fecha default getdate(),
  comentarios text,
  constraint PK_visitas
  primary key(numero)
);

3- Ingrese algunos registros:
 insert into visitas values ('Ana Maria
Lopez','AnaMaria@hotmail.com','Argentina','2006-10-10 10:10',null);
```

```
 insert into visitas values ('Gustavo
Gonzalez','GustavoGGonzalez@hotmail.com','Chile','2006-10-10 21:30',default);
 insert into visitas values ('Fabiola
Martinez','MartinezFabiola@hotmail.com','Mexico',default,'Excelente página');
 insert into visitas values ('Mariano Perez','PerezM@hotmail.com','Argentina','2006-11-
11 14:30','Muy buena y divertida');
```

4- Leemos la información almacenada en el campo "comentarios" de "visitas" del registro número 3,
desde la posición 0, 10 caracteres.

5- Intente leer el campo "comentarios" del registro número 1.
Error, porque el puntero es inválido.

6- Recupere el campo "comentarios" de la visita número 1 (desde el comienzo al final), controlando
que el puntero sea válido.

116 - Tipo de dato text - ntext e image (escribir)

Primer problema:

En una página web se guardan los siguientes datos de las visitas: número de visita, nombre, mail,
pais, fecha.

1- Elimine la tabla "visitas", si existe:

```
 if object_id('visitas') is not null
  drop table visitas;
```

2- Créela con la siguiente estructura:

```
 create table visitas (
  numero int identity,
  nombre varchar(30),
  mail varchar(50),
  pais varchar (20),
  fecha datetime
  constraint DF_visitas_fecha default getdate(),
  comentarios text,
  constraint PK_visitas
  primary key(numero)
);
```

3- Ingrese algunos registros:

```
 insert into visitas values ('Ana Maria
Lopez','AnaMaria@hotmail.com','Argentina','2006-10-10 10:10',null);
 insert into visitas values ('Gustavo
Gonzalez','GustavoGGonzalez@hotmail.com','Chile','2006-10-10 21:30',default);
 insert into visitas values ('Fabiola
Martinez','MartinezFabiola@hotmail.com','Mexico',default,'Excelente página');
 insert into visitas values ('Mariano Perez','PerezM@hotmail.com','Argentina','2006-11-
11 14:30','Muy buena y divertida');
```

4- Recupere todos los registros:

```
 select *from visitas;
```

5- Reemplace el texto del campo "comentarios" del registro con número 3.

6- Lea el campo "comentarios" de la visita número 3 para ver si se actualizó.

7- Intente actualizar el campo "text" de la visita número 1.
Error, puntero inválido.

8- Vuelva a intentar la actualización del punto anterior pero controlando que el puntero sea válido.

9- Ingrese un nuevo registro con cadena vacía para el campo "comentarios":

```
 insert into visitas values ('Salvador
Quiroga','salvador@hotmail.com','Argentina','2006-09-09 18:25','');
```

10- Actualice el campo "comentarios" del registro ingresado anteriormente.

11- Verifique que se actualizó.

117 - Tipo de dato text - ntext e image (actualizar)

Primer problema:
Un maestro almacena los datos de sus alumnos en una tabla denominada "alumnos", incluye el
documento, el nombre, la nota y un comentario acerca del comportamiento de cada uno de ellos.
1- Elimine la tabla si existe:

```
 if object_id('alumnos') is not null
  drop table alumnos;
```

2- Créela con la siguiente estructura:

```
 create table alumnos (
  documento char(8),
  nombre varchar(30),
  nota decimal(4,2),
  concepto text,
  constraint PK_alumnos
  primary key(documento)
 );
```

3- Ingrese algunos registros:

```
 insert into alumnos values ('22222222','Ana Acosta',3,'Participativo. Generoso');
 insert into alumnos values ('23333333','Carlos Caseres',7,'Poco participativo');
 insert into alumnos values ('24444444','Diego Duarte',8,'Buen compañero');
 insert into alumnos values ('25555555','Fabiola Fuentes',2,null);
```

4- Recupere todos los registros:

```
 select *from alumnos;
```

5- Inserte en el concepto del alumno con documento "23333333" el texto "comunicativo", en la
posición 5, borrando todos los caracteres siguientes. Verifique que el puntero sea válido, en caso
de no serlo, muestre un mensaje de error.

6- Lea el campo "concepto" actualizado anteriormente para verificar que se actualizó.

7- Intente actualizar el concepto del alumno con documento "25555555" el texto "Muy comunicativo".
Verifique que el puntero sea válido, en caso de no serlo, muestre un mensaje de error.
Puntero inválido.

8- Intente agregar texto al campo "concepto" del alumno con documento "24444444" en la posición 20.
Mensaje de error porque el texto tiene una longitud menor.

9- Inserte en el concepto del alumno con documento "24444444" el texto "alumno y", en la posición 5,
sin borrar ningún caracter. Verifique que el puntero sea válido antes de pasar el puntero a la
función "updatetext".

10- Lea el campo "concepto" actualizado anteriormente para verificar que se actualizó.

11- Elimine la tabla "reprobados" si existe:

```
 if object_id('reprobados') is not null
  drop table reprobados;
```

12- Cree la tabla "reprobados" que contenga 2 campos: documento y concepto:

```
 create table reprobados(
  documento char(8) not null,
  concepto text
 );
```

13- Ingrese los siguientes registros en "reprobados" (en el campo "concepto" ingresamos cadenas
vacías para que se creen punteros válidos):

```
 insert into reprobados values('22222222','');
 insert into reprobados values('25555555','');
```

14- Actualice el "concepto" del alumno "22222222" de la tabla "reprobados" con el concepto de dicho
alumno de la tabla "alumnos". Verifique que los punteros sean válidos.

15- Verifique la actualización.

16- Intente actualizar el "concepto" del alumno "25555555" de la tabla "reprobados" con el concepto
de dicho alumno de la tabla "alumnos". Verifique que los punteros sean válidos.
Mensaje de error porque hay un puntero inválido, el de la tabla "alumnos", porque el registro
consultado contiene "null" en "concepto".

17- Intente actualizar el "concepto" del alumno "23333333" de la tabla "reprobados" con el concepto
de dicho alumno de la tabla "alumnos". Verifique que los punteros sean válidos.
Mensaje de error porque hay un puntero inválido, el de "reprobados", no existe el registro
consultado.

120 - Procedimientos almacenados (crear - ejecutar)

Primer problema:
Una empresa almacena los datos de sus empleados en una tabla llamada "empleados".
1- Eliminamos la tabla, si existe y la creamos:

```
 if object_id('empleados') is not null
  drop table empleados;

 create table empleados(
  documento char(8),
  nombre varchar(20),
  apellido varchar(20),
  sueldo decimal(6,2),
  cantidadhijos tinyint,
  seccion varchar(20),
  primary key(documento)
 );
```

2- Ingrese algunos registros:

```
 insert into empleados values('22222222','Juan','Perez',300,2,'Contaduria');
 insert into empleados values('22333333','Luis','Lopez',300,0,'Contaduria');
 insert into empleados values ('22444444','Marta','Perez',500,1,'Sistemas');
 insert into empleados values('22555555','Susana','Garcia',400,2,'Secretaria');
 insert into empleados values('22666666','Jose Maria','Morales',400,3,'Secretaria');
```

3- Elimine el procedimiento llamado "pa_empleados_sueldo" si existe:

```
 if object_id('pa_empleados_sueldo') is not null
  drop procedure pa_empleados_sueldo;
```

4- Cree un procedimiento almacenado llamado "pa_empleados_sueldo" que seleccione los nombres,
apellidos y sueldos de los empleados.

5- Ejecute el procedimiento creado anteriormente.

6- Elimine el procedimiento llamado "pa_empleados_hijos" si existe:

```
 if object_id('pa_empleados_hijos') is not null
  drop procedure pa_empleados_hijos;
```

7- Cree un procedimiento almacenado llamado "pa_empleados_hijos" que seleccione los nombres,
apellidos y cantidad de hijos de los empleados con hijos.

8- Ejecute el procedimiento creado anteriormente.

9- Actualice la cantidad de hijos de algún empleado sin hijos y vuelva a ejecutar el procedimiento
para verificar que ahora si aparece en la lista.

122 - Procedimientos almacenados (parámetros de entrada)

Primer problema:

Una empresa almacena los datos de sus empleados en una tabla llamada "empleados".

1- Eliminamos la tabla, si existe y la creamos:

```
 if object_id('empleados') is not null
  drop table empleados;

 create table empleados(
  documento char(8),
  nombre varchar(20),
  apellido varchar(20),
  sueldo decimal(6,2),
  cantidadhijos tinyint,
  seccion varchar(20),
  primary key(documento)
 );
```

2- Ingrese algunos registros:

```
 insert into empleados values('22222222','Juan','Perez',300,2,'Contaduria');
 insert into empleados values('22333333','Luis','Lopez',300,0,'Contaduria');
 insert into empleados values ('22444444','Marta','Perez',500,1,'Sistemas');
 insert into empleados values('22555555','Susana','Garcia',400,2,'Secretaria');
 insert into empleados values('22666666','Jose Maria','Morales',400,3,'Secretaria');
```

3- Elimine el procedimiento llamado "pa_empleados_sueldo" si existe:

```
 if object_id('pa_empleados_sueldo') is not null
  drop procedure pa_empleados_sueldo;
```

4- Cree un procedimiento almacenado llamado "pa_empleados_sueldo" que seleccione los nombres,
apellidos y sueldos de los empleados que tengan un sueldo superior o igual al enviado como
parámetro.

5- Ejecute el procedimiento creado anteriormente con distintos valores:

```
 exec pa_empleados_sueldo 400;
 exec pa_empleados_sueldo 500;
```

6- Ejecute el procedimiento almacenado "pa_empleados_sueldo" sin parámetros.
Mensaje de error.

7- Elimine el procedimiento almacenado "pa_empleados_actualizar_sueldo" si existe:

```
 if object_id('pa_empleados_actualizar_sueldo') is not null
  drop procedure pa_empleados_actualizar_sueldo;
```

8- Cree un procedimiento almacenado llamado "pa_empleados_actualizar_sueldo" que actualice los
sueldos iguales al enviado como primer parámetro con el valor enviado como segundo parámetro.

9- Ejecute el procedimiento creado anteriormente y verifique si se ha ejecutado correctamente:
```
 exec pa_empleados_actualizar_sueldo 300,350;
 select * from empleados;
```

10- Ejecute el procedimiento "pa_empleados_actualizar_sueldo" enviando un solo parámetro.
Error.

11- Ejecute el procedimiento almacenado "pa_empleados_actualizar_sueldo" enviando en primer lugar el
parámetro @sueldonuevo y en segundo lugar @sueldoanterior (parámetros por nombre).

12- Verifique el cambio:
```
 select * from empleados;
```

13- Elimine el procedimiento almacenado "pa_sueldototal", si existe:
```
 if object_id('pa_sueldototal') is not null
  drop procedure pa_sueldototal;
```

14- Cree un procedimiento llamado "pa_sueldototal" que reciba el documento de un empleado y muestre
su nombre, apellido y el sueldo total (resultado de la suma del sueldo y salario por hijo, que es de
$200 si el sueldo es menor a $500 y $100, si el sueldo es mayor o igual a $500). Coloque como valor
por defecto para el parámetro el patrón "%".

15- Ejecute el procedimiento anterior enviando diferentes valores:
```
 exec pa_sueldototal '22333333';
 exec pa_sueldototal '22444444';
 exec pa_sueldototal '22666666';
```

16- Ejecute el procedimiento sin enviar parámetro para que tome el valor por defecto. Muestra los 5 registros.

123 - Procedimientos almacenados (parámetros de salida)

Primer problema:
Una empresa almacena los datos de sus empleados en una tabla llamada "empleados".
1- Eliminamos la tabla, si existe y la creamos:
```
 if object_id('empleados') is not null
  drop table empleados;
```

```
create table empleados(
 documento char(8),
 nombre varchar(20),
 apellido varchar(20),
 sueldo decimal(6,2),
 cantidadhijos tinyint,
 seccion varchar(20),
 primary key(documento)
);
```

2- Ingrese algunos registros:

```
insert into empleados values('22222222','Juan','Perez',300,2,'Contaduria');
insert into empleados values('22333333','Luis','Lopez',350,0,'Contaduria');
insert into empleados values ('22444444','Marta','Perez',500,1,'Sistemas');
insert into empleados values('22555555','Susana','Garcia',null,2,'Secretaria');
insert into empleados values('22666666','Jose Maria','Morales',460,3,'Secretaria');
insert into empleados values('22777777','Andres','Perez',580,3,'Sistemas');
insert into empleados values('22888888','Laura','Garcia',400,3,'Secretaria');
```

3- Elimine el procedimiento llamado "pa_seccion" si existe:

```
if object_id('pa_seccion') is not null
 drop procedure pa_seccion;
```

4- Cree un procedimiento almacenado llamado "pa_seccion" al cual le enviamos el nombre de una
sección y que nos retorne el promedio de sueldos de todos los empleados de esa sección y el valor
mayor de sueldo (de esa sección)

5- Ejecute el procedimiento creado anteriormente con distintos valores.

6- Ejecute el procedimiento "pa_seccion" sin pasar valor para el parámetro "sección". Luego muestre
los valores devueltos por el procedimiento.
Calcula sobre todos los registros porque toma el valor por defecto.

7- Elimine el procedimiento almacenado "pa_sueldototal", si existe y cree un procedimiento con ese
nombre que reciba el documento de un empleado y retorne el sueldo total, resultado de la suma del
sueldo y salario por hijo, que es $200 si el sueldo es menor a $500 y $100 si es mayor o igual.

8- Ejecute el procedimiento anterior enviando un documento existente.

9- Ejecute el procedimiento anterior enviando un documento inexistente.
Retorna "null".

10- Ejecute el procedimiento anterior enviando el documento de un empleado en cuyo campo "sueldo"
contenga "null".
Retorna "null".

11- Ejecute el procedimiento anterior sin enviar valor para el parámetro "documento". Retorna el valor calculado del último registro.

124 - Procedimientos almacenados (return)

Primer problema:

Una empresa almacena los datos de sus empleados en una tabla llamada "empleados".

1- Eliminamos la tabla, si existe y la creamos:

```
 if object_id('empleados') is not null
  drop table empleados;

 create table empleados(
  documento char(8),
  nombre varchar(20),
  apellido varchar(20),
  cantidadhijos tinyint,
  seccion varchar(20),
  primary key(documento)
 );
```

2- Ingrese algunos registros:

```
 insert into empleados values('22222222','Juan','Perez',2,'Contaduria');
 insert into empleados values('22333333','Luis','Lopez',0,'Contaduria');
 insert into empleados values ('22444444','Marta','Perez',NULL,'Sistemas');
 insert into empleados values('22555555','Susana','Garcia',2,'Secretaria');
 insert into empleados values('22666666','Jose Maria','Morales',1,'Secretaria');
 insert into empleados values('22777777','Andres','Perez',3,'Sistemas');
 insert into empleados values('22888888','Laura','Garcia',3,'Secretaria');
```

3- Elimine el procedimiento llamado "pa_empleados_seccion", si existe:

```
 if object_id('pa_empleados_seccion') is not null
  drop procedure pa_empleados_seccion;
```

4- Cree un procedimiento que muestre todos los empleados de una sección determinada que se ingresa
como parámetro. Si no se ingresa un valor, o se ingresa "null", se muestra un mensaje y se sale del
procedimiento.

5- Ejecute el procedimiento enviándole un valor para el parámetro.

6- Ejecute el procedimiento sin parámetro.

7- Elimine el procedimiento "pa_actualizarhijos", si existe:

```
 if object_id('pa_actualizarhijos') is not null
  drop procedure pa_actualizarhijos;
```

8- Cree un procedimiento almacenado que permita modificar la cantidad de hijos ingresando el
documento de un empleado y la cantidad de hijos nueva. Ambos parámetros DEBEN ingresarse con un

valor distinto de "null". El procedimiento retorna "1" si la actualización se realiza (si se ingresan valores para ambos parámetros) y "0", en caso que uno o ambos parámetros no se ingresen o
sean nulos.

9- Declare una variable en la cual se almacenará el valor devuelto por el procedimiento, ejecute el
procedimiento enviando los dos parámetros y vea el contenido de la variable.
El procedimiento retorna "1", con lo cual indica que fue actualizado.

10- Verifique la actualización consultando la tabla:
select *from empleados;

11- Ejecute los mismos pasos, pero esta vez envíe solamente un valor para el parámetro "documento".
Retorna "0", lo que indica que el registro no fue actualizado.

12- Verifique que el registro no se actualizó consultando la tabla:
select *from empleados;

13- Emplee un "if" para controlar el valor de la variable de retorno. Enviando al procedimiento valores para los parámetros.
Retorna 1.

14- Verifique la actualización consultando la tabla:
select *from empleados;

15- Emplee nuevamente un "if" y envíe solamente valor para el parámetro "hijos".
Retorna 0.