51 - Restricción unique

Primer problema:
Una empresa de remises tiene registrada la información de sus vehículos en una tabla llamada
"remis".
1- Elimine la tabla si existe:

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

2- Cree la tabla con la siguiente estructura:

```
 create table remis(
  numero tinyint identity,
  patente char(6),
  marca varchar(15),
  modelo char(4)
 );
```

3- Ingrese algunos registros, 2 de ellos con patente repetida y alguno con patente nula:

```
 insert into remis values('ABC123','Renault clio','1990');
 insert into remis values('DEF456','Peugeot 504','1995');
 insert into remis values('DEF456','Fiat Duna','1998');
 insert into remis values('GHI789','Fiat Duna','1995');
 insert into remis values(null,'Fiat Duna','1995');
```

4- Intente agregar una restricción "unique" para asegurarse que la patente del remis no tomará
valores repetidos.
No se puede porque hay valores duplicados.

5- Elimine el registro con patente duplicada y establezca la restricción.
Note que hay 1 registro con valor nulo en "patente".

6- Intente ingresar un registro con patente repetida (no lo permite)

7- Intente ingresar un registro con valor nulo para el campo "patente".
No lo permite porque la clave estaría duplicada.

8- Muestre la información de las restricciones:

```
 exec sp_helpconstraint remis;
```

53 - Eliminar restricciones (alter table - drop)

Primer problema:
Una playa de estacionamiento almacena cada día los datos de los vehículos que ingresan en la tabla
llamada "vehiculos".
1- Elimine la tabla, si existe:

```
 if object_id('vehiculos') is not null
```

```
 drop table vehiculos;
```

2- Cree la tabla:

```
create table vehiculos(
 patente char(6) not null,
 tipo char(1),--'a'=auto, 'm'=moto
 horallegada datetime not null,
 horasalida datetime
);
```

3- Establezca una restricción "check" que admita solamente los valores "a" y "m" para el campo "tipo":

```
alter table vehiculos
add constraint CK_vehiculos_tipo
check (tipo in ('a','m'));
```

4- Establezca una restricción "default" para el campo "tipo" que almacene el valor "a" en caso de no ingresarse valor para dicho campo:

```
alter table vehiculos
 add constraint DF_vehiculos_tipo
 default 'a'
 for tipo;
```

5- Establezca una restricción "check" para el campo "patente" para que acepte 3 letras seguidas de 3 dígitos:

```
alter table vehiculos
add constraint CK_vehiculos_patente_patron
check (patente like '[A-Z][A-Z][A-Z][0-9][0-9][0-9]');
```

6- Agregue una restricción "primary key" que incluya los campos "patente" y "horallegada":

```
alter table vehiculos
add constraint PK_vehiculos_patentellegada
primary key(patente,horallegada);
```

7- Ingrese un vehículo:

```
insert into vehiculos values('SDR456','a','2005/10/10 10:10',null);
```

8- Intente ingresar un registro repitiendo la clave primaria:

```
insert into vehiculos values('SDR456','m','2005/10/10 10:10',null);
```

No se permite.

9- Ingrese un registro repitiendo la patente pero no la hora de llegada:

```
insert into vehiculos values('SDR456','m','2005/10/10 12:10',null);
```

10- Ingrese un registro repitiendo la hora de llegada pero no la patente:

```
insert into vehiculos values('SDR111','m','2005/10/10 10:10',null);
```

11- Vea todas las restricciones para la tabla "vehiculos":

```
exec sp_helpconstraint vehiculos;
```

aparecen 4 filas, 2 correspondientes a restricciones "check", 1 a "default" y 1 a "primary key".

12- Elimine la restricción "default" del campo "tipo".

13- Vea si se ha eliminado:
```
exec sp_helpconstraint vehiculos;
```

14- Elimine la restricción "primary key" y "check".

15- Vea si se han eliminado:
```
exec sp_helpconstraint vehiculos;
```

54 - Crear y asociar reglas (create rule - sp_bindrule)

Primer problema:

Una playa de estacionamiento almacena cada día los datos de los vehículos que ingresan en la tabla
llamada "vehiculos".

1- Elimine la tabla, si existe:
```
if object_id('vehiculos') is not null
  drop table vehiculos;
```

2- Elimine las siguientes reglas:
```
if object_id ('RG_patente_patron') is not null
   drop rule RG_patente_patron;
if object_id ('RG_horallegada') is not null
   drop rule RG_horallegada;
if object_id ('RG_vehiculos_tipo') is not null
   drop rule RG_vehiculos_tipo;
if object_id ('RG_vehiculos_tipo2') is not null
   drop rule RG_vehiculos_tipo2;
if object_id ('RG_menor_fechaactual') is not null
   drop rule RG_menor_fechaactual;
```

3- Cree la tabla:
```
create table vehiculos(
  patente char(6) not null,
  tipo char(1),--'a'=auto, 'm'=moto
  horallegada datetime not null,
  horasalida datetime
);
```

4- Ingrese algunos registros:
```
insert into vehiculos values ('AAA111','a','1990-02-01 08:10',null);
insert into vehiculos values ('BCD222','m','1990-02-01 08:10','1990-02-01 10:10');
insert into vehiculos values ('BCD222','m','1990-02-01 12:00',null);
insert into vehiculos values ('CC1234','a','1990-02-01 12:00',null);
```

5- Cree una regla para restringir los valores que se pueden ingresar en un campo "patente" (3 letras

seguidas de 3 dígitos):
```
create rule RG_patente_patron
as @patente like '[A-Z][A-Z][A-Z][0-9][0-9][0-9]'
```

6- Ejecute el procedimiento almacenado del sistema "sp_help" para ver que la regla creada
anteriormente existe:
```
exec sp_help;
```

7- Ejecute el procedimiento almacenado del sistema "sp_helpconstraint" para ver que la regla creada
anteriormente no está asociada aún a ningún campo de la tabla "vehiculos".

8- Asocie la regla al campo "patente":
Note que hay una patente que no cumple la regla, SQL Server NO controla los datos existentes, pero
si controla las inserciones y actualizaciones:
```
select * from empleados;
```

9- Intente ingresar un registro con valor para el campo "patente" que no cumpla con la regla.
aparece un mensaje de error indicando que hay conflicto con la regla y la inserción no se realiza.

10- Cree otra regla que controle los valores para el campo "tipo" para que solamente puedan
ingresarse los caracteres "a" y "m".

11- Asocie la regla al campo "tipo".

12- Intente actualizar un registro cambiando el valor de "tipo" a un valor que no cumpla con la
regla anterior.
No lo permite.

13- Cree otra regla llamada "RG_vehiculos_tipo2" que controle los valores para el campo "tipo" para
que solamente puedan ingresarse los caracteres "a", "c" y "m".

14- Si la asociamos a un campo que ya tiene asociada otra regla, la nueva regla reeemplaza la
asociación anterior. Asocie la regla creada en el punto anterior al campo "tipo".

15- Actualice el registro que no pudo actualizar en el punto 12:
```
update vehiculos set tipo='c' where patente='AAA111';
```

16- Cree una regla que permita fechas menores o iguales a la actual.

17- Asocie la regla anterior a los campos "horallegada" y "horasalida":
```
exec sp_bindrule RG_menor_fechaactual, 'vehiculos.horallegada';
exec sp_bindrule RG_menor_fechaactual, 'vehiculos.horasalida';
```

18- Ingrese un registro en el cual la hora de entrada sea posterior a la hora de salida:

```
 insert into vehiculos values ('NOP555','a','1990-02-01 10:10','1990-02-01 08:30');
```

19- Intente establecer una restricción "check" que asegure que la fecha y hora de llegada a la playa
no sea posterior a la fecha y hora de salida:

```
 alter table vehiculos
 add constraint CK_vehiculos_llegada_salida
 check(horallegada<=horasalida);
```

No lo permite porque hay un registro que no cumple la restricción.

20- Elimine dicho registro:

```
 delete from vehiculos where patente='NOP555';
```

21- Establezca la restricción "check" que no pudo establecer en el punto 19:

```
 alter table vehiculos
 add constraint CK_vehiculos_llegada_salida
 check(horallegada<=horasalida);
```

22- Cree una restricción "default" que almacene el valor "b" en el campo "tipo":

```
 alter table vehiculos
 add constraint DF_vehiculos_tipo
 default 'b'
 for tipo;
```

Note que esta restricción va contra la regla asociada al campo "tipo" que solamente permite los
valores "a", "c" y "m". SQL Server no informa el conflicto hasta que no intenta ingresar el valor
por defecto.

23- Intente ingresar un registro con el valor por defecto para el campo "tipo":

```
 insert into vehiculos values ('STU456',default,'1990-02-01 10:10','1990-02-01 15:30');
```

No lo permite porque va contra la regla asociada al campo "tipo".

24- Vea las reglas asociadas a "empleados" y las restricciones aplicadas a la misma tabla ejecutando
"sp_helpconstraint".
Muestra 1 restricción "check", 1 restricción "default" y 4 reglas asociadas.

55 - Eliminar y dasasociar reglas (sp_unbindrule - drop rule)

Primer problema:
Una playa de estacionamiento almacena cada día los datos de los vehículos que ingresan en la tabla
llamada "vehiculos".
1- Elimine la tabla, si existe:

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

2- Elimine las siguientes reglas, si existen:

```
 if object_id ('RG_patente_patron') is not null
```

```
   drop rule RG_patente_patron;
 if object_id ('RG_vehiculos_tipo') is not null
   drop rule RG_vehiculos_tipo;
 if object_id ('RG_vehiculos_tipo2') is not null
   drop rule RG_vehiculos_tipo2;
```

3- Cree la tabla:

```
 create table vehiculos(
  patente char(6) not null,
  tipo char(1),--'a'=auto, 'm'=moto
  horallegada datetime not null,
  horasalida datetime
 );
```

4- Cree una regla para restringir los valores que se pueden ingresar en un campo "patente" (3 letras
seguidas de 3 dígitos):

```
 create rule RG_patente_patron
 as @patente like '[A-Z][A-Z][A-Z][0-9][0-9][0-9]';
```

5-Asocie la regla al campo "patente":

```
 exec sp_bindrule RG_patente_patron,'vehiculos.patente';
```

6- Intente ingresar un registro con valor para el campo "patente" que no cumpla con la regla:

```
 insert into vehiculos values ('FGHIJK','a','1990-02-01 18:00',null);
```

aparece un mensaje de error indicando que hay conflicto con la regla y la inserción no se realiza.

7- Cree otra regla que controle los valores para el campo "tipo" para que solamente puedan
ingresarse los caracteres "a" y "m":

```
 create rule RG_vehiculos_tipo
 as @tipo in ('a','m')
```

8- Asocie la regla al campo "tipo":

```
 exec sp_bindrule RG_vehiculos_tipo, 'vehiculos.tipo';
```

9- Intente ingresar un registro con el valor 'c' para "tipo":

```
 insert into vehiculos values('AAA111','c','2001-10-10 10:10',NULL);
```

No lo permite.

10- Cree otra regla llamada "RG_vehiculos_tipo2" que controle los valores para el campo "tipo" para
que solamente puedan ingresarse los caracteres "a", "c" y "m":

```
 create rule RG_vehiculos_tipo2
 as @tipo in ('a','c','m');
```

11- Si la asociamos a un campo que ya tiene asociada otra regla, la nueva regla reeemplaza la
asociación anterior. Asocie la regla creada en el punto anterior al campo "tipo".

12- Ingrese el registro que no pudo ingresar en el punto 9.

13- Intente eliminar la regla "RG_vehiculos_tipo2".
No es posible porque está asociada a un campo de "vehiculos".

14- Elimine la regla "RG_vehiculos_tipo".
Es posible porque no está asociada a ningún campo.

15- Intente eliminar la regla "RG_patente_patron".
No es posible porque está asociada.

16- Quite la asociación de la regla con el campo "patente" de "vehiculos".

17- Vea si la regla "RG_patente_patron" está asociada a algún campo de "vehiculos".
No lo está.

18- Verifique que la regla aún existe en la base de datos activa:
```
 exec sp_help;
```
aparece la regla.

19- Elimine la regla que no pudo eliminar en el punto 15.

20- Verifique que la regla ya no existe en la base de datos activa.
No aparece la regla "RG_patente_patron".

57 - Valores predeterminados (create default)

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

2- Recuerde que si elimina una tabla, las asociaciones de reglas y valores predeterminados de sus
campos desaparecen, pero las reglas y valores predeterminados siguen existiendo. Si intenta crear
una regla o un valor predeterminado con igual nombre que uno existente, aparecerá un mensaje
indicándolo, por ello, debe eliminarlos (si existen) para poder crearlos nuevamente:
```
 if object_id ('VP_legajo_patron') is not null
   drop default VP_legajo_patron;
 if object_id ('RG_legajo_patron') is not null
   drop rule RG_legajo_patron;
 if object_id ('RG_legajo') is not null
   drop rule RG_legajo;
 if object_id ('VP_datodesconocido') is not null
   drop default VP_datodesconocido;
 if object_id ('VP_fechaactual') is not null
   drop default VP_fechaactual;
```

3- Cree la tabla:

```
create table clientes(
  legajo char(4),
  nombre varchar(30),
  domicilio varchar(30),
  ciudad varchar(15),
  provincia varchar(20) default 'Cordoba',
  fechaingreso datetime
 );
```

4- Cree una regla para establecer un patrón para los valores que se ingresen en el campo "legajo" (2
letras seguido de 2 cifras) llamada "RG_legajo_patron":

5- Asocie la regla al campo "legajo".

6- Cree un valor predeterminado para el campo "legajo" ('AA00') llamado "VP_legajo_patron".

7- Asócielo al campo "legajo".
Recuerde que un campo puede tener un valor predeterminado y reglas asociados.

8- Cree un valor predeterminado con la cadena "??" llamado "VP_datodesconocido".

9- Asócielo al campo "domicilio".

10- Asócielo al campo "ciudad".
Recuerde que un valor predeterminado puede asociarse a varios campos.

11- Ingrese un registro con valores por defecto para los campos "domicilio" y "ciudad" y vea qué
almacenaron.

12- Intente asociar el valor predeterminado "VP_datodesconocido" al campo "provincia".
No se puede porque dicho campo tiene una restricción "default".

13- Cree un valor predeterminado con la fecha actual llamado "VP_fechaactual".

14- Asócielo al campo "fechaingreso".

15- Ingrese algunos registros para ver cómo se almacenan los valores para los cuales no se insertan
datos.

16- Asocie el valor predeterminado "VP_datodesconocido" al campo "fechaingreso".
Note que se asoció un valor predeterminado de tipo caracter a un campo de tipo "datetime"; SQL
Server lo permite, pero al intentar ingresar el valor aparece un mensaje de error.

17- Ingrese un registro con valores por defecto.
No lo permite porque son de distintos tipos.

18- Cree una regla que entre en conflicto con el valor predeterminado "VP_legajo_patron".

19- Asocie la regla al campo "legajo".
Note que la regla especifica que el campo "legajo" debe comenzar con la letra "B", pero el valor
predeterminado tiene el valor "AA00"; SQL Server realiza la asociación, pero al intentar ingresar el
valor predeterminado, no puede hacerlo y muestra un mensaje de error.

20- Intente ingresar un registro con el valor "default" para el campo "legajo".
No lo permite porque al intentar ingresar el valor por defecto establecido con el valor predeterminado entra en conflicto con la regla "RG_legajo".

58 - Desasociar y eliminar valores predeterminados

Primer problema:
Una librería almacena los datos de sus libros en una tabla llamada "libros".
1- Elimine la tabla si existe:

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

2- Recuerde que si elimina una tabla, las asociaciones de reglas y valores predeterminados de sus
campos desaparecen, pero las reglas y valores predeterminados siguen existiendo. Si intenta crear
una regla o un valor predeterminado con igual nombre que uno existente, aparecerá un mensaje
indicándolo, por ello, debe eliminarlos (si existen) para poder crearlos nuevamente:

```
 if object_id ('VP_cero') is not null
   drop default VP_cero;
 if object_id ('VP_desconocido') is not null
   drop default VP_desconocido;
 if object_id ('RG_positivo') is not null
   drop rule RG_positivo;
```

3- Cree la tabla:

```
 create table libros(
  codigo int identity,
  titulo varchar(40) not null,
  autor varchar(30),
  editorial varchar(20),
  precio decimal(5,2),
  cantidad smallint
 );
```

4- Cree una regla para impedir que se ingresen valores negativos, llamada "RG_positivo".

5- Asocie la regla al campo "precio".

6- Asocie la regla al campo "cantidad".

7- Cree un valor predeterminado para que almacene el valor cero, llamado "VP_cero".

8- Asócielo al campo "precio".

9- Asócielo al campo "cantidad".

10- Cree un valor predeterminado con la cadena "Desconocido" llamado "VP_desconocido".

11- Asócielo al campo "autor".

12- Asócielo al campo "editorial".

13- Vea las reglas y valores predeterminados con "sp_help":
exec sp_help;

14- Vea las reglas y valores predeterminados asociados a "libros".
Aparecen 6 filas, 2 corresponden a la regla "RG_positivo" asociadas a los campos "precio" y
"cantidad"; 2 al valor predeterminado "VP_cero" asociados a los campos "precio" y "cantidad" y 2 al
valor predeterminado "VP_desconocido" asociados a los campos "editorial" y "autor".

15- Ingrese un registro con valores por defecto para todos los campos, excepto "titulo" y vea qué se
almacenó.

15- Quite la asociación del valor predeterminado "VP_cero" al campo "precio".

16- Ingrese otro registro con valor predeterminado para el campo "precio" y vea cómo se almacenó.

17- Vea las reglas y valores predeterminados asociados a "libros".
5 filas; el valor predeterminado "VP_cero" ya no está asociado al campo "precio".

18- Verifique que el valor predeterminado "VP_cero" existe aún en la base de datos.

19- Intente eliminar el valor predeterminado "VP_cero".
No se puede porque está asociado al campo "cantidad".

20- Quite la asociación del valor predeterminado "VP_cero" al campo "cantidad".

21- Verifique que ya no existe asociación de este valor predeterminado con la tabla "libros".
4 filas.

22- Verifique que el valor predeterminado "VP_cero" aun existe en la base de datos.

23- Elimine el valor predeterminado "VP_cero".

24- Verifique que ya no existe en la base de datos.

62 - Creación de índices

Primer problema:

Un profesor guarda algunos datos de sus alumnos en una tabla llamada "alumnos".

1- Elimine la tabla si existe y créela con la siguiente estructura:

```
 if object_id('alumnos') is not null
  drop table alumnos;
 create table alumnos(
  legajo char(5) not null,
  documento char(8) not null,
  apellido varchar(30),
  nombre varchar(30),
  notafinal decimal(4,2)
 );
```

2- Ingresamos algunos registros:

```
 insert into alumnos values ('A123','22222222','Perez','Patricia',5.50);
 insert into alumnos values ('A234','23333333','Lopez','Ana',9);
 insert into alumnos values ('A345','24444444','Garcia','Carlos',8.5);
 insert into alumnos values ('A348','25555555','Perez','Daniela',7.85);
 insert into alumnos values ('A457','26666666','Perez','Fabian',3.2);
 insert into alumnos values ('A589','27777777','Gomez','Gaston',6.90);
```

3- Intente crear un índice agrupado único para el campo "apellido".
No lo permite porque hay valores duplicados.

4- Cree un índice agrupado, no único, para el campo "apellido".

5- Intente establecer una restricción "primary key" al campo "legajo" especificando que cree un
índice agrupado.
No lo permite porque ya existe un índice agrupado y solamente puede haber uno por tabla.

6- Establezca la restricción "primary key" al campo "legajo" especificando que cree un índice NO
agrupado.

7- Vea los índices de "alumnos":

```
 exec sp_helpindex alumnos;
```

2 índices: uno "I_alumnos_apellido", agrupado, con "apellido" y otro "PK_alumnos_legajo", no
agrupado, unique, con "legajo" que se creó automáticamente al crear la restricción "primary key".

8- Analice la información que muestra "sp_helpconstraint":

```
 exec sp_helpconstraint libros;
```

En la columna "constraint_type" aparece "PRIMARY KEY" y entre paréntesis, el tipo de índice creado.

9- Cree un índice unique no agrupado para el campo "documento".

10- Intente ingresar un alumno con documento duplicado.
No lo permite.

11- Veamos los indices de "alumnos".
Aparecen 3 filas, uno por cada índice.

12- Cree un índice compuesto para el campo "apellido" y "nombre".
Se creará uno no agrupado porque no especificamos el tipo, además, ya existe uno agrupado y
solamente puede haber uno por tabla.

13- Consulte la tabla "sysindexes", para ver los nombres de todos los índices creados para
"alumnos":
```
 select name from sysindexes
  where name like '%alumnos%';
```
4 índices.

14- Cree una restricción unique para el campo "documento".

15- Vea la información de "sp_helpconstraint".

16- Vea los índices de "alumnos".
Aparecen 5 filas, uno por cada índice.

17- Consulte la tabla "sysindexes", para ver los nombres de todos los índices creados para
"alumnos":
```
 select name from sysindexes
  where name like '%alumnos%';
```
5 índices.

18- Consulte la tabla "sysindexes", para ver los nombres de todos los índices creados por usted:
```
 select name from sysindexes
  where name like 'I_%';
```
3 índices. Recuerde que los índices que crea SQL Server automáticamente al agregarse una restricción
"primary" o "unique" no comienzan con "I_".

63 - Regenerar índices

Primer problema:
Un profesor guarda algunos datos de sus alumnos en una tabla llamada "alumnos".
1- Elimine la tabla si existe y créela con la siguiente estructura:
```
 if object_id('alumnos') is not null
  drop table alumnos;
 create table alumnos(
  legajo char(5) not null,
  documento char(8) not null,
  apellido varchar(30),
```

```
  nombre varchar(30),
  notafinal decimal(4,2)
 );
```

2- Cree un índice no agrupado para el campo "apellido".

3- Vea la información de los índices de "alumnos".

4- Modifíquelo agregando el campo "nombre".

5- Verifique que se modificó:

```
 exec sp_helpindex alumnos;
```

6- Establezca una restricción "unique" para el campo "documento".

7- Vea la información que muestra "sp_helpindex":

```
  exec sp_helpindex alumnos;
```

8- Intente modificar con "drop_existing" alguna característica del índice que se creó automáticamente al agregar la restricción "unique":

```
 create clustered index UQ_alumnos_documento
  on alumnos(documento)
  with drop_existing;
```

No se puede emplear "drop_existing" con índices creados a partir de una restricción.

9- Cree un índice no agrupado para el campo "legajo".

10- Muestre todos los índices:

```
 exec sp_helpindex alumnos;
```

11- Convierta el índice creado en el punto 9 a agrupado conservando las demás características.

12- Verifique que se modificó:

```
 exec sp_helpindex alumnos;
```

13- Intente convertir el índice "I_alumnos_legajo" a no agrupado:

```
 create nonclustered index I_alumnos_legajo
  on alumnos(legajo)
  with drop_existing;
```

No se puede convertir un índice agrupado en no agrupado.

14- Modifique el índice "I_alumnos_apellido" quitándole el campo "nombre".

15- Intente convertir el índice "I_alumnos_apellido" en agrupado:

```
 create clustered index I_alumnos_apellido
  on alumnos(apellido)
  with drop_existing;
```

No lo permite porque ya existe un índice agrupado.

16- Modifique el índice "I_alumnos_legajo" para que sea único y conserve todas las otras
características.

17- Verifique la modificación:
 exec sp_helpindex alumnos;

18- Modifique nuevamente el índice "I_alumnos_legajo" para que no sea único y conserve las demás
características.

19- Verifique la modificación:
 exec sp_helpindex alumnos;

64 - Eliminar índices

Primer problema:
Un profesor guarda algunos datos de sus alumnos en una tabla llamada "alumnos".
1- Elimine la tabla si existe y créela con la siguiente estructura:

```
 if object_id('alumnos') is not null
  drop table alumnos;
 create table alumnos(
  legajo char(5) not null,
  documento char(8) not null,
  apellido varchar(30),
  nombre varchar(30),
  notafinal decimal(4,2)
 );
```

2- Cree un índice no agrupado para el campo "apellido".

3- Establezca una restricción "primary" para el campo "legajo" y especifique que cree un índice
"agrupado".

4- Vea la información que muestra "sp_helpindex":

```
  exec sp_helpindex alumnos;
```

5- Intente eliminar el índice "PK_alumnos_legajo" con "drop index":

```
 drop index PK_alumnos_legajo;
```

No se puede.

6- Intente eliminar el índice "I_alumnos_apellido" sin especificar el nombre de la tabla:

```
 drop index I_alumnos_apellido;
```

Mensaje de error.

7- Elimine el índice "I_alumnos_apellido" especificando el nombre de la tabla.

8- Verifique que se eliminó:

```
 exec sp_helpindex alumnos;
```

9- Solicite que se elimine el índice "I_alumnos_apellido" si existe:

```
 if exists (select name from sysindexes
  where name = 'I_alumnos_apellido')
```

```
   drop index alumnos.I_alumnos_apellido;
```

10- Elimine el índice "PK_alumnos_legajo" (quite la restricción).

11- Verifique que el índice "PK_alumnos_legajo" ya no existe:

```
 exec sp_helpindex alumnos;
```

66 - Combinación interna (inner join)

Primer problema:

Una empresa tiene registrados sus clientes en una tabla llamada "clientes", también tiene una tabla
"provincias" donde registra los nombres de las provincias.

1- Elimine las tablas "clientes" y "provincias", si existen:

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

2- Créelas con las siguientes estructuras:

```
 create table clientes (
  codigo int identity,
  nombre varchar(30),
  domicilio varchar(30),
  ciudad varchar(20),
  codigoprovincia tinyint not null,
  primary key(codigo)
 );

 create table provincias(
  codigo tinyint identity,
  nombre varchar(20),
  primary key (codigo)
 );
```

3- Ingrese algunos registros para ambas tablas:

```
 insert into provincias (nombre) values('Cordoba');
 insert into provincias (nombre) values('Santa Fe');
 insert into provincias (nombre) values('Corrientes');

 insert into clientes values ('Lopez Marcos','Colon 111','Córdoba',1);
 insert into clientes values ('Perez Ana','San Martin 222','Cruz del Eje',1);
 insert into clientes values ('Garcia Juan','Rivadavia 333','Villa Maria',1);
 insert into clientes values ('Perez Luis','Sarmiento 444','Rosario',2);
 insert into clientes values ('Pereyra Lucas','San Martin 555','Cruz del Eje',1);
 insert into clientes values ('Gomez Ines','San Martin 666','Santa Fe',2);
 insert into clientes values ('Torres Fabiola','Alem 777','Ibera',3);
```

4- Obtenga los datos de ambas tablas, usando alias:

```
 select c.nombre,domicilio,ciudad,p.nombre
  from clientes as c
```

```
  join provincias as p
  on c.codigoprovincia=p.codigo;
```

5- Obtenga la misma información anterior pero ordenada por nombre de provincia.

6- Recupere los clientes de la provincia "Santa Fe" (2 registros devueltos)

Segundo problema:

Un club dicta clases de distintos deportes. Almacena la información en una tabla llamada
"inscriptos" que incluye el documento, el nombre, el deporte y si la matricula esta paga o no y una
tabla llamada "inasistencias" que incluye el documento, el deporte y la fecha de la inasistencia.

1- Elimine las tablas si existen y cree las tablas:

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

 create table inscriptos(
  nombre varchar(30),
  documento char(8),
  deporte varchar(15),
  matricula char(1), --'s'=paga 'n'=impaga
  primary key(documento,deporte)
 );

 create table inasistencias(
  documento char(8),
  deporte varchar(15),
  fecha datetime
 );
```

2- Ingrese algunos registros para ambas tablas:

```
 insert into inscriptos values('Juan Perez','22222222','tenis','s');
 insert into inscriptos values('Maria Lopez','23333333','tenis','s');
 insert into inscriptos values('Agustin Juarez','24444444','tenis','n');
 insert into inscriptos values('Marta Garcia','25555555','natacion','s');
 insert into inscriptos values('Juan Perez','22222222','natacion','s');
 insert into inscriptos values('Maria Lopez','23333333','natacion','n');

 insert into inasistencias values('22222222','tenis','2006-12-01');
 insert into inasistencias values('22222222','tenis','2006-12-08');
 insert into inasistencias values('23333333','tenis','2006-12-01');
 insert into inasistencias values('24444444','tenis','2006-12-08');
 insert into inasistencias values('22222222','natacion','2006-12-02');
 insert into inasistencias values('23333333','natacion','2006-12-02');
```

3- Muestre el nombre, el deporte y las fechas de inasistencias, ordenado por nombre y deporte.
Note que la condición es compuesta porque para identificar los registros de la tabla "inasistencias"

necesitamos ambos campos.

4- Obtenga el nombre, deporte y las fechas de inasistencias de un determinado inscripto en un
determinado deporte (3 registros)

5- Obtenga el nombre, deporte y las fechas de inasistencias de todos los inscriptos que pagaron la
matrícula(4 registros)

67 - Combinación externa izquierda (left join)

Primer problema:
Una empresa tiene registrados sus clientes en una tabla llamada "clientes", también tiene una tabla
"provincias" donde registra los nombres de las provincias.
1- Elimine las tablas "clientes" y "provincias", si existen y cree las tablas:

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

create table clientes (
 codigo int identity,
 nombre varchar(30),
 domicilio varchar(30),
 ciudad varchar(20),
 codigoprovincia tinyint not null,
 primary key(codigo)
);

create table provincias(
 codigo tinyint identity,
 nombre varchar(20),
 primary key (codigo)
);
```

2- Ingrese algunos registros para ambas tablas:

```
insert into provincias (nombre) values('Cordoba');
insert into provincias (nombre) values('Santa Fe');
insert into provincias (nombre) values('Corrientes');

insert into clientes values ('Lopez Marcos','Colon 111','Córdoba',1);
insert into clientes values ('Perez Ana','San Martin 222','Cruz del Eje',1);
insert into clientes values ('Garcia Juan','Rivadavia 333','Villa Maria',1);
insert into clientes values ('Perez Luis','Sarmiento 444','Rosario',2);
insert into clientes values ('Gomez Ines','San Martin 666','Santa Fe',2);
insert into clientes values ('Torres Fabiola','Alem 777','La Plata',4);
insert into clientes values ('Garcia Luis','Sucre 475','Santa Rosa',5);
```

3- Muestre todos los datos de los clientes, incluido el nombre de la provincia:

```
 select c.nombre,domicilio,ciudad, p.nombre
  from clientes as c
  left join provincias as p
  on codigoprovincia = p.codigo;
```

4- Realice la misma consulta anterior pero alterando el orden de las tablas:

```
 select c.nombre,domicilio,ciudad, p.nombre
  from provincias as p
  left join clientes as c
  on codigoprovincia = p.codigo;
```

5- Muestre solamente los clientes de las provincias que existen en "provincias" (5 registros):

```
 select c.nombre,domicilio,ciudad, p.nombre
  from clientes as c
  left join provincias as p
  on codigoprovincia = p.codigo
  where p.codigo is not null;
```

6- Muestre todos los clientes cuyo código de provincia NO existe en "provincias" ordenados por
nombre del cliente (2 registros):

```
 select c.nombre,domicilio,ciudad, p.nombre
  from clientes as c
  left join provincias as p
  on codigoprovincia = p.codigo
  where p.codigo is null
  order by c.nombre;
```

7- Obtenga todos los datos de los clientes de "Cordoba" (3 registros):

```
 select c.nombre,domicilio,ciudad, p.nombre
  from clientes as c
  left join provincias as p
  on codigoprovincia = p.codigo
  where p.nombre='Cordoba';
```

68 - Combinación externa derecha (right join)

Primer problema:

Una empresa tiene registrados sus clientes en una tabla llamada "clientes", también tiene una
tabla "provincias" donde registra los nombres de las provincias.

1- Elimine las tablas "clientes" y "provincias", si existen y cree las tablas:

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

 create table clientes (
  codigo int identity,
  nombre varchar(30),
```

```
 domicilio varchar(30),
 ciudad varchar(20),
 codigoprovincia tinyint not null,
 primary key(codigo)
);

create table provincias(
 codigo tinyint identity,
 nombre varchar(20),
 primary key (codigo)
);
```

2- Ingrese algunos registros para ambas tablas:

```
insert into provincias (nombre) values('Cordoba');
insert into provincias (nombre) values('Santa Fe');
insert into provincias (nombre) values('Corrientes');

insert into clientes values ('Lopez Marcos','Colon 111','Córdoba',1);
insert into clientes values ('Perez Ana','San Martin 222','Cruz del Eje',1);
insert into clientes values ('Garcia Juan','Rivadavia 333','Villa Maria',1);
insert into clientes values ('Perez Luis','Sarmiento 444','Rosario',2);
insert into clientes values ('Gomez Ines','San Martin 666','Santa Fe',2);
insert into clientes values ('Torres Fabiola','Alem 777','La Plata',4);
insert into clientes values ('Garcia Luis','Sucre 475','Santa Rosa',5);
```

3- Muestre todos los datos de los clientes, incluido el nombre de la provincia empleando un "right
join".

4- Obtenga la misma salida que la consulta anterior pero empleando un "left join".

5- Empleando un "right join", muestre solamente los clientes de las provincias que existen en
"provincias" (5 registros)

6- Muestre todos los clientes cuyo código de provincia NO existe en "provincias" ordenados por
ciudad (2 registros)

69 - Combinación externa completa (full join)

Primer problema:

Un club dicta clases de distintos deportes. Almacena la información en una tabla llamada "deportes"
en la cual incluye el nombre del deporte y el nombre del profesor y en otra tabla llamada
"inscriptos" que incluye el documento del socio que se inscribe, el deporte y si la matricula está
paga o no.

1- Elimine las tablas si existen y cree las tablas:

```
 if (object_id('deportes')) is not null
```

```
 drop table deportes;
if (object_id('inscriptos')) is not null
 drop table inscriptos;
create table deportes(
 codigo tinyint identity,
 nombre varchar(30),
 profesor varchar(30),
 primary key (codigo)
);
create table inscriptos(
 documento char(8),
 codigodeporte tinyint not null,
 matricula char(1) --'s'=paga 'n'=impaga
);
```

2- Ingrese algunos registros para ambas tablas:

```
insert into deportes values('tenis','Marcelo Roca');
insert into deportes values('natacion','Marta Torres');
insert into deportes values('basquet','Luis Garcia');
insert into deportes values('futbol','Marcelo Roca');

insert into inscriptos values('22222222',3,'s');
insert into inscriptos values('23333333',3,'s');
insert into inscriptos values('24444444',3,'n');
insert into inscriptos values('22222222',2,'s');
insert into inscriptos values('23333333',2,'s');
insert into inscriptos values('22222222',4,'n');
insert into inscriptos values('22222222',5,'n');
```

3- Muestre todos la información de la tabla "inscriptos", y consulte la tabla "deportes" para
obtener el nombre de cada deporte (6 registros)

4- Empleando un "left join" con "deportes" obtenga todos los datos de los inscriptos (7 registros)

5- Obtenga la misma salida anterior empleando un "rigth join".

6- Muestre los deportes para los cuales no hay inscriptos, empleando un "left join" (1 registro)

7- Muestre los documentos de los inscriptos a deportes que no existen en la tabla "deportes" (1
registro)

8- Emplee un "full join" para obtener todos los datos de ambas tablas, incluyendo las inscripciones
a deportes inexistentes en "deportes" y los deportes que no tienen inscriptos (8 registros)

70 - Combinaciones cruzadas (cross join)

Primer problema:
Una agencia matrimonial almacena la información de sus clientes de sexo femenino en una tabla
llamada "mujeres" y en otra la de sus clientes de sexo masculino llamada "varones".
1- Elimine las tablas si existen y créelas:

```
 if object_id('mujeres') is not null
  drop table mujeres;
 if object_id('varones') is not null
  drop table varones;
 create table mujeres(
  nombre varchar(30),
  domicilio varchar(30),
  edad int
 );
 create table varones(
  nombre varchar(30),
  domicilio varchar(30),
  edad int
 );
```

2- Ingrese los siguientes registros:

```
 insert into mujeres values('Maria Lopez','Colon 123',45);
 insert into mujeres values('Liliana Garcia','Sucre 456',35);
 insert into mujeres values('Susana Lopez','Avellaneda 98',41);

 insert into varones values('Juan Torres','Sarmiento 755',44);
 insert into varones values('Marcelo Oliva','San Martin 874',56);
 insert into varones values('Federico Pereyra','Colon 234',38);
 insert into varones values('Juan Garcia','Peru 333',50);
```

3- La agencia necesita la combinación de todas las personas de sexo femenino con las de sexo
masculino. Use un "cross join" (12 registros)

4- Realice la misma combinación pero considerando solamente las personas mayores de 40 años (6 registros)

5- Forme las parejas pero teniendo en cuenta que no tengan una diferencia superior a 10 años (8 registros)

71 - Autocombinación

Primer problema:
Una agencia matrimonial almacena la información de sus clientes en una tabla llamada "clientes".
1- Elimine la tabla si existe y créela:

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

 create table clientes(
```

```
  nombre varchar(30),
  sexo char(1),--'f'=femenino, 'm'=masculino
  edad int,
  domicilio varchar(30)
 );
```

2- Ingrese los siguientes registros:

```
 insert into clientes values('Maria Lopez','f',45,'Colon 123');
 insert into clientes values('Liliana Garcia','f',35,'Sucre 456');
 insert into clientes values('Susana Lopez','f',41,'Avellaneda 98');
 insert into clientes values('Juan Torres','m',44,'Sarmiento 755');
 insert into clientes values('Marcelo Oliva','m',56,'San Martin 874');
 insert into clientes values('Federico Pereyra','m',38,'Colon 234');
 insert into clientes values('Juan Garcia','m',50,'Peru 333');
```

3- La agencia necesita la combinación de todas las personas de sexo femenino con las de sexo masculino. Use un "cross join" (12 registros)

4- Obtenga la misma salida enterior pero realizando un "join".

5- Realice la misma autocombinación que el punto 3 pero agregue la condición que las parejas no tengan una diferencia superior a 5 años (5 registros)

Segundo problema:

Varios clubes de barrio se organizaron para realizar campeonatos entre ellos. La tabla llamada "equipos" guarda la informacion de los distintos equipos que jugarán.

1- Elimine la tabla, si existe y créela nuevamente:

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

 create table equipos(
  nombre varchar(30),
  barrio varchar(20),
  domicilio varchar(30),
  entrenador varchar(30)
 );
```

2- Ingrese los siguientes registros:

```
 insert into equipos values('Los tigres','Gral. Paz','Sarmiento 234','Juan Lopez');
 insert into equipos values('Los leones','Centro','Colon 123','Gustavo Fuentes');
 insert into equipos values('Campeones','Pueyrredon','Guemes 346','Carlos Moreno');
 insert into equipos values('Cebollitas','Alberdi','Colon 1234','Luis Duarte');
```

4- Cada equipo jugará con todos los demás 2 veces, una vez en cada sede. Realice un "cross join" para combinar los equipos teniendo en cuenta que un equipo no juega consigo mismo (12 registros)

5- Obtenga el mismo resultado empleando un "join".

6- Realice un "cross join" para combinar los equipos para que cada equipo juegue con cada uno de los
otros una sola vez (6 registros)

72 - Combinaciones y funciones de agrupamiento

Primer problema:

Un comercio que tiene un stand en una feria registra en una tabla llamada "visitantes" algunos datos
de las personas que visitan o compran en su stand para luego enviarle publicidad de sus productos y
en otra tabla llamada "ciudades" los nombres de las ciudades.

1- Elimine las tablas si existen:

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

2- Cree las tablas:

```
 create table visitantes(
  nombre varchar(30),
  edad tinyint,
  sexo char(1) default 'f',
  domicilio varchar(30),
  codigociudad tinyint not null,
  mail varchar(30),
  montocompra decimal (6,2)
 );

 create table ciudades(
  codigo tinyint identity,
  nombre varchar(20)
 );
```

3- Ingrese algunos registros:

```
 insert into ciudades values('Cordoba');
 insert into ciudades values('Carlos Paz');
 insert into ciudades values('La Falda');
 insert into ciudades values('Cruz del Eje');

 insert into visitantes values
   ('Susana Molina', 35,'f','Colon 123', 1, null,59.80);
 insert into visitantes values
   ('Marcos Torres', 29,'m','Sucre 56', 1, 'marcostorres@hotmail.com',150.50);
 insert into visitantes values
   ('Mariana Juarez', 45,'f','San Martin 111',2,null,23.90);
 insert into visitantes values
   ('Fabian Perez',36,'m','Avellaneda 213',3,'fabianperez@xaxamail.com',0);
 insert into visitantes values
   ('Alejandra Garcia',28,'f',null,2,null,280.50);
```

```
 insert into visitantes values
   ('Gaston Perez',29,'m',null,5,'gastonperez1@gmail.com',95.40);
 insert into visitantes values
   ('Mariana Juarez',33,'f',null,2,null,90);
```

4- Cuente la cantidad de visitas por ciudad mostrando el nombre de la ciudad (3 filas)

5- Muestre el promedio de gastos de las visitas agrupados por ciudad y sexo (4 filas)

6- Muestre la cantidad de visitantes con mail, agrupados por ciudad (3 filas)

7- Obtenga el monto de compra más alto de cada ciudad (3 filas)

73 - Combinación de más de dos tablas

Primer problema:

Un club dicta clases de distintos deportes. En una tabla llamada "socios" guarda los datos de los
socios, en una tabla llamada "deportes" la información referente a los diferentes deportes que se
dictan y en una tabla denominada "inscriptos", las inscripciones de los socios a los distintos
deportes.
Un socio puede inscribirse en varios deportes el mismo año. Un socio no puede inscribirse en el
mismo deporte el mismo año. Distintos socios se inscriben en un mismo deporte en el mismo año.

1- Elimine las tablas si existen:

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

2- Cree las tablas con las siguientes estructuras:

```
 create table socios(
  documento char(8) not null,
  nombre varchar(30),
  domicilio varchar(30),
  primary key(documento)
 );
 create table deportes(
  codigo tinyint identity,
  nombre varchar(20),
  profesor varchar(15),
  primary key(codigo)
 );
 create table inscriptos(
  documento char(8) not null,
  codigodeporte tinyint not null,
```

```
  anio char(4),
  matricula char(1),--'s'=paga, 'n'=impaga
  primary key(documento,codigodeporte,anio)
 );

3- Ingrese algunos registros en "socios":
 insert into socios values('22222222','Ana Acosta','Avellaneda 111');
 insert into socios values('23333333','Betina Bustos','Bulnes 222');
 insert into socios values('24444444','Carlos Castro','Caseros 333');
 insert into socios values('25555555','Daniel Duarte','Dinamarca 44');
4- Ingrese algunos registros en "deportes":
 insert into deportes values('basquet','Juan Juarez');
 insert into deportes values('futbol','Pedro Perez');
 insert into deportes values('natacion','Marina Morales');
 insert into deportes values('tenis','Marina Morales');

5- Inscriba a varios socios en el mismo deporte en el mismo año:
 insert into inscriptos values ('22222222',3,'2006','s');
 insert into inscriptos values ('23333333',3,'2006','s');
 insert into inscriptos values ('24444444',3,'2006','n');

6- Inscriba a un mismo socio en el mismo deporte en distintos años:
 insert into inscriptos values ('22222222',3,'2005','s');
 insert into inscriptos values ('22222222',3,'2007','n');

7- Inscriba a un mismo socio en distintos deportes el mismo año:
 insert into inscriptos values ('24444444',1,'2006','s');
 insert into inscriptos values ('24444444',2,'2006','s');

8- Ingrese una inscripción con un código de deporte inexistente y un documento de
socio que no
exista en "socios":
 insert into inscriptos values ('26666666',0,'2006','s');

9- Muestre el nombre del socio, el nombre del deporte en que se inscribió y el año
empleando
diferentes tipos de join.

10- Muestre todos los datos de las inscripciones (excepto los códigos) incluyendo
aquellas
inscripciones cuyo código de deporte no existe en "deportes" y cuyo documento de
socio no se
encuentra en "socios".

11- Muestre todas las inscripciones del socio con documento "22222222".
```