الجداول والفهارس

- أنشئ قاعدة بيانات لشركة . تحوي جدول للموظفين حيث لكل موظف (رقم، اسم ، عنوان، تاريخ ميلاد، عمر، عدد الأولاد)

```
CREATE TABLE Employees
(
EmployeeID int IDENTITY NOT NULL,
FirstName varchar(25) NOT NULL,
Addres varchar(25) NOT NULL,
BirthDate date NOT NULL,
age AS DateDIff(yy,BirthDate,GetDate()),
children_number int Default 0 NOT NULL ,
)
```

حاول ادخال سجل لموظف جديد ماذا تلاحظ؟؟؟

- عدل على الجدول السابق بحيث تعدل خصائص عمود العمر.
نلاحظ أنه لا يمكن تعديل هذا العمود لأنه محسوب..

```
ALTER TABLE Employees
Alter column age varchar(35) NOT NULL;
```

```
Messages
Msg 4928, Level 16, State 1, Line 1
Cannot alter column 'age' because it is 'COMPUTED'.
```

- عدل على الجدول السابق بحيث يصبح نوع العمود العنوان عبارة عن رقم وليس سلسلة.

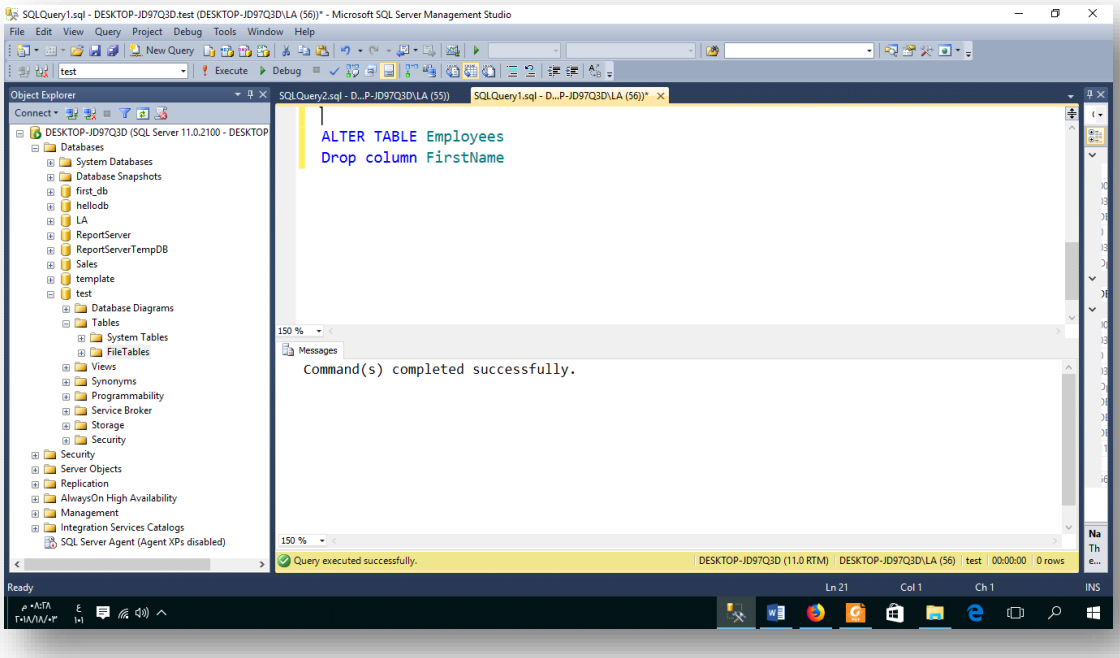
```
ALTER TABLE Employees
Alter column addres int NOT NULL;
```

```
Messages
Command(s) completed successfully.
```

- عدل الجدول السابق حيث تضيف عمود يمثل الشهادة الحاصل عليها الموظف.

```
alter table Employees
ADD
studying varchar(13) NULL
```

- من الجدول السابق احذف عمود اسم الشخص.



- احذف الجدول السابق.

```
Drop table Employees;
```

الفهرس

لفَهْرَسة هي ببساطة عبارة عن مؤشر يحتوي على نسخة من جزء من البيانات في الجدول، بحيث تقوم هذه النسخة من البيانات بمهمة "الدليل" أو "المُؤَشّر "الذي يسرع الوصول إلى البيانات الأصلية الكاملة الموجودة في الجدول، بحيث لا تحتاج المرور الكامل على كل الجدول (No Full Table Scan) عند البحث عن البيانات.

كيف تُعرَّف الفهارس؟

يُعرَّف الفهرس بطريقتيْن :

- تعريفه ضمنيًّا: تُبنَى الفهارس ضمنيا على الأعمدة التي يُطَبَّق عليها القيد الفريد وقيد المفتاح الرئيسي، فعند تعريف أحد القيود السابقة، يُبنى فهرس تلقائيًّا على العمود أو الأعمدة المُقيَّدة .
- تعريفه صراحةً: يُبنَى الفهرس بطريقة مباشرة على العمود أو الأعمدة الذي نرغب وذلك باستخدام جملة .Create Index

متى نستخدم الفهارس؟

يفضل أن يتم بناء الفهارس على الأعمدة التي :

- يُبحث عنها في جملة .Where
- تُكتَب في جملة الترتيب .Order By
- تُكتَب في جملة التجميع .Group By
- تُستخدَم في جمل الربط .Joins
- تُستخدَم في الدوال الإحصائية مثل minو maxو .median

متى نتجنب استخدام الفهارس؟

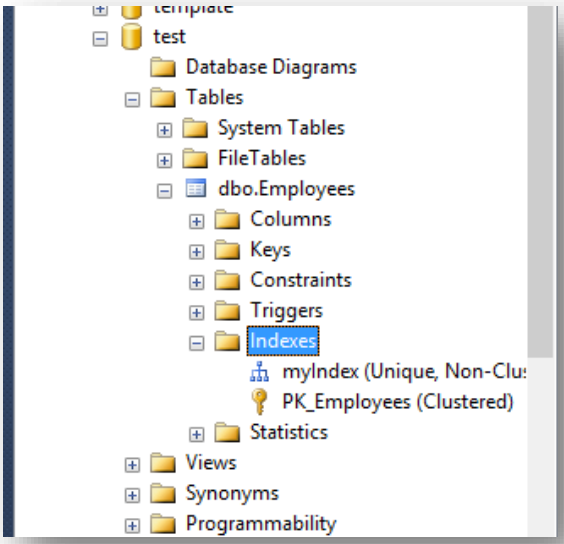
لا تعدّ الفهارس مناسبة على الأعمدة التي:
-تحتوي على قيم فريدة قليلة مثل عمود الجنس (قيمتان فقط)، أو الحالة الاجتماعية.
-نادرة الاستخدام في جمل الاستعلام .SELECT
-التي تكون جزءًا من جدول ذي سجلات قليلة

ملاحظة:
يُسمى الفهرس الذي يُعرَّف على عمود واحد "فهرسا بسيطا"، والفهرس الذي يُعرَّف على أكثر من عمود يسمى "فهرسا مركبا."
في جدول الموظفين الذي تم تعريفه سابقاً في هذه المحاضرة أنشئ فهرس فريد على اسم الموظف.

```
CREATE UNIQUE INDEX myIndex ON Employees (name);
```

نلاحظ أنه أصبح لدينا فهرسين في جدول الموظفين:

١. فهرس المفتاح الرئيسي.
٢. الفهرس الفريد الذي تم إنشاؤه على الاسم.



لحذف الفهرس الذي تم إنشاؤه على العمود حقل الاسم.

```
Drop index Employees.myIndex;
```

- ليكن لدينا جدول الموظفين والاقسام كما موضح بالشكل التالي

	dept_id	dept_name	dept_date
1	1	it	1990-03-17
2	2	resource	NULL
3	3	market	2000-05-22

	e_id	e_name	age	address	dept_id
1	1	lama	27	hama	1
2	2	hala	55	homs	1
3	3	wael	44	homs	2
4	4	tala	23	hama	3
5	5	ola	33	hama	1

إنشاء منظار

- أنشئ منظار يحوي أسماء الأشخاص الذين يتجاوز عمرهم ٣٥ سنة.

```
create view view1(employee_name,age)
AS

select  e_name , age
FROM    dbo.Employee
WHERE     (age > 35)
```

ولعرض محتويات المنظار السابق

```
select * from view1
```

	employee_name	age
1	hala	55
2	wael	44

- أنشئ منظار يعيد عدد الموظفين في كل قسم

```
create view view2(dept_id,numberOfEmployee)
AS
select  dbo.Departement.dept_id ,
COUNT(Employee.e_id)
FROM    dbo.Employee ,dbo.Departement
WHERE
dbo.Employee.dept_id=dbo.Departement.dept_id
group by Departement.dept_id
```

وتكون النتيجة كالتالي

```
select * from view2
```

	dept_id	numberOfEmployee
1	1	3
2	2	1
3	3	1

التعديل على بنية المنظار

- عدل على بنية المنظور view1 حيث يتم إضافة عمود عنوان الشخص إضافة لتعديل شرط العمر

```
alter view view1
as
select
Employee.e_name,Employee.age,Employee.address
from Employee
where age>30
```

وعند عرض هذا المنظار نجد

	e_name	age	address
1	hala	55	homs
2	wael	44	homs
3	ola	33	hama

التعديل على البيانات من خلال المنظار

من خلال استخدام المنظار يمكن ادراج وتعديل و حذف سجلات من الجدول الأصلي ولكن يجب أن يتحقق بالمنظار مجموعة من الشروط وهي:

- The SELECT clause may not contain the keyword DISTINCT.
- The SELECT clause may not contain summary functions.
- The SELECT clause may not contain set functions.
- The SELECT clause may not contain set operators.
- The SELECT clause may not contain an ORDER BY clause.
- The FROM clause may not contain multiple tables.
- The WHERE clause may not contain subqueries.
- The query may not contain GROUP BY or HAVING.

- Calculated columns may not be updated.

- من خلال المنظار view1 اجعل عنوان سكن الأشخاص الذين يبلغون من العمر ٥٥ سنة في Damascus.
  - `update view1 set address='damascus' where age=55;`

	e_id	e_name	age	address	dept_id
1	1	lama	27	hama	1
2	2	hala	55	damascus	1
3	3	wael	44	homs	2
4	4	tala	23	hama	3
5	5	ola	33	hama	1

- من خلال المنظار view1 احذف الشخص الذي عمره ٣٣ عاما.
  - `delete from view1 where age=33;`

	e_id	e_name	age	address	dept_id
1	1	lama	27	hama	1
2	2	hala	55	damascus	1
3	3	wael	44	homs	2
4	4	tala	23	hama	3

ملاحظة:
عند اجراء أي محاولة تعديل او حذف من خلال المنظار view2 نحصل على خطأ. لأن هذا الجدول يتكون من دمج جداول.

- احذف المنظار view2

```
drop view view2;
```

- ليكن لدينا جدول الموظفين والاقسام كما موضح بالشكل التالي

	dept_id	dept_name	dept_date
1	1	it	1990-03-17
2	2	resource	NULL
3	3	market	2000-05-22

e_id	e_name	age	address	dept_id	salary
1	lama	27	hama	1	10000
4	lana	36	damascus	1	20000
3	tala	23	hama	3	150
5	ahmed	37	hama	2	13566
6	esraa	22	hama	3	75894
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

المطلوب:

١. أنشئ إجرائية تعيد أسماء الموظفين في القسم رقم ١

```
CREATE PROCEDURE viewEmployeeData1
AS
SELECT e_name FROM Employee
where dept_id=1
GO
```

ولتنفيذ هذه الإجرائية

```
execute viewEmployeeData1;
```

150 %

Results | Messages

	e_name
1	lama
2	lana

٢. عدل على الإجرائية السابقة لتعيد أسماء الموظفين لأي قسم يحدده المستخدم.

```
alter PROCEDURE viewEmployeeData1(@id int)
AS
SELECT e_name FROM Employee
where dept_id=@id
GO
```

وبالتنفيذ

```
execute viewEmployeeData1 3;
```

150 %

Results | Messages

	e_name
1	esraa
2	tala

٣. أنشئ إجرائية لادراج قسم جديد.

```
create PROCEDURE viewEmployeeData2(@id int,@name
varchar(23),@date Date='null')
AS
insert into Departement values(@id,@name,@date)
GO
```

وبالتنفيذ

```
execute viewEmployeeData2 4,'med','2014-12-12';
```

٤. انشئ إجرائية تعيد عدد الأشخاص الذين يعملون في قسم ما.

```
create PROCEDURE viewEmployeeData3(@id int,@count
int output)
AS
set @count=(select Count(e_id)
from Employee
where dept_id=@id)
GO
```

بالتنفيذ

```
DECLARE @result int;
EXECUTE viewEmployeeData3 1 , @result OUTPUT;
PRINT @result;
```

فيكون الناتج ٢.

٥. أنشئ إجرائية تحسب مجموع رواتب الموظفين حسب المدينة التي يقطنون بها والقسم الذي يعملون به.

```
create PROCEDURE viewEmployeeData4(@city varchar(20),@id int,@count
int output)
AS
set @count=(select sum(salary)
from Employee,Departement
where Departement.dept_id=@id and
Departement.dept_id=Employee.dept_id and Employee.address=@city)
GO
```

وبالتنفيذ

```
DECLARE @result int;
EXECUTE viewEmployeeData4 'hama',3, @result OUTPUT;
PRINT @result;
```

يكون الخرج هو ٧٦٠٤٤.

٦. أنشئ إجرائية لطباعة أسماء الموظفين في قسم it

```
create PROCEDURE viewEmployeeData5
as
select e_name
from Employee,Departement
where  Departement.dept_id=Employee.dept_id and
Departement.dept_name='it'
GO
```

٧. أنشئ إجرائية تعيد ناتج مجموع وطرح عددين .

```
create procedure myprocedure ( @x int,@y int,@sum int
output,@sub int output)
as
set @sum=@x+@y;
set @sub=@x-@y;
```

وبالتنفيذ

```
declare @sum int,@sub int
execute myprocedure 1,5,@sum out,@sub out
print @sum;
print @sub;
```

ويكون الناتج ٦و-٤

٨. احذف الإجرائية السابقة

```
drop procedure myprocedure
```

- ليكن لدينا جدول الموظفين والاقسام كما موضح بالشكل التالي

	dept_id	dept_name	dept_date
1	1	it	1990-03-17
2	2	resource	NULL
3	3	market	2000-05-22

e_id	e_name	age	address	dept_id	salary
1	lama	27	hama	1	10000
4	lana	36	damascus	1	20000
3	tala	23	hama	3	150
5	ahmed	37	hama	2	13566
6	esraa	22	hama	3	75894
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

المطلوب:

١. أنشئ تابع يعيد المتوسط الحسابي لرواتب الموظفين.

```
create function MyAvg () returns int
as
begin
declare @sum int,@count int;
set @sum=(select sum(salary) from Employee);
set @count=(select count(Employee.e_id) from
Employee);
return @sum/@count;
end

select dbo.MyAvg() as average;
```

٢. أنشئ تابع يتحقق من وجود موظف باسم معين.

```
create function Found (@name varchar(12)) returns int
as
begin
declare @found int,@count int;
set @count=(select count(Employee.e_id) from Employee where
Employee.e_name=@name);
if @count>=1 set @found=1;
else
set @found=0;
return @found;
end

select dbo.Found('lana') as found;
```

٣. لحذف تابع

```
drop function Found;
```

- ليكن لدينا جدول الموظفين والاقسام كما موضح بالشكل التالي

	dept_id	dept_name	dept_date
1	1	it	1990-03-17
2	2	resource	NULL
3	3	market	2000-05-22

e_id	e_name	age	address	dept_id	salary
1	lama	27	hama	1	10000
4	lana	36	damascus	1	20000
3	tala	23	hama	3	150
5	ahmed	37	hama	2	13566
6	esraa	22	hama	3	75894
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

المطلوب:

٩. أنشئ إجرائية تقوم بإدراج قسم جديد حيث يقوم المستخدم بإدخال رقم القسم واسمه وتقوم الإجرائية بإدخال سجل جديد في حال عدم وجود قسم سابق بنفس الاسم.

```
create procedure InsertDepartement(@id int,@name
varchar(13),@result varchar(14) output)
as
declare @n varchar(15);
set  @n=(select dept_name from Departement where
dept_name=@name)
if @n!= ''
begin
set @result='Error';
end
else
begin
insert into Departement(dept_id,dept_name)
values(@id,@name);
set @result='done';
end
```

لاستدعاء الإجرائية السابقة

```
DECLARE    @result  varchar(14)
execute InsertDepartement 5,'econom',@result output
print @result
```

١٠. أنشئ إجرائية تحسب مجموع رواتب الموظفين الذين يتبعون لقسم معين وراتبهم أعلى من حد معين.

```
create procedure calculate (@id int,@level int,@result int
output)
as
 if @id<=0
 set @id=1
set @result=( select sum(salary)
from Employee
where   dept_id=@id and
salary >=@level)
go
```

```
declare @re int;
execute calculate 0,1000,@re output
print @re
```

١١. أنشئ إجرائية تحذف سجل موظف معين في حال وجوده

```
create procedure deleteEmp (@name varchar(16),@r int output)
as

set @r=(select count(e_name) from Employee where e_name=@name
)
if @r=0
begin
print 'No Employee Found'
end
else
delete from Employee where e_name=@name
go

declare @w int
execute deleteEmp 'lamalala',@w output
print @w
```

١٢. أنشئ إجرائية تطبع اسم الموظف وراتبه واسم القسم الذي يعمل به اعتماداً على رقم القسم الذي يعمل به.

```
create procedure viewData
as
SELECT   e_name, salary, "dept_Name" =
      CASE
         WHEN dept_id =1 THEN 'it'
         WHEN dept_id =2 THEN 'resourse'
         WHEN dept_id =3 then 'market'
         ELSE 'other'
      END
       from Employee

       go

       execute viewData
```

١٣. أنشئ إجرائية تقوم بمضاعفة رواتب الموظفين طالما المتوسط الحسابي لهذه الرواتب أقل من ٥٠٠٠ أوفي حال وصل اعلى راتب لقيمة ٩٠٠٠٠

```
create procedure UpdateProcedure
as
declare @r int
WHILE ( SELECT AVG(salary) FROM dbo.Employee) < 5000
BEGIN

  set @r= (SELECT AVG(salary) FROM dbo.Employee)
  print @r
    UPDATE dbo.Employee
        SET salary = salary * 2;

    IF ( SELECT MAX (salary) FROM dbo.Employee) >
90000
        BREAK;
END

execute UpdateProcedure
```

- أنشئ قاعدة بيانات لشركة . تحوي جدول للموظفين حيث لكل موظف (رقم، اسم ، عنوان، تاريخ ميلاد، عمر، عدد الأولاد)

```
CREATE TABLE Employees
(
EmployeeID int IDENTITY NOT NULL,
FirstName varchar(25) NOT NULL,
Addres varchar(25) NOT NULL,
BirthDate date NOT NULL,
age AS DateDIff(yy,BirthDate,GetDate()),
children_number int Default 0 NOT NULL ,
)
```

حاول ادخال سجل لموظف جديد ماذا تلاحظ؟؟؟

- عدل على الجدول السابق بحيث تعدل خصائص عمود العمر.
نلاحظ أنه لا يمكن تعديل هذا العمود لأنه محسوب..

```
ALTER TABLE Employees
Alter column age varchar(35) NOT NULL;
```

```
Messages
Msg 4928, Level 16, State 1, Line 1
Cannot alter column 'age' because it is 'COMPUTED'.
```

- عدل على الجدول السابق بحيث يصبح نوع العمود العنوان عبارة عن رقم وليس سلسلة.

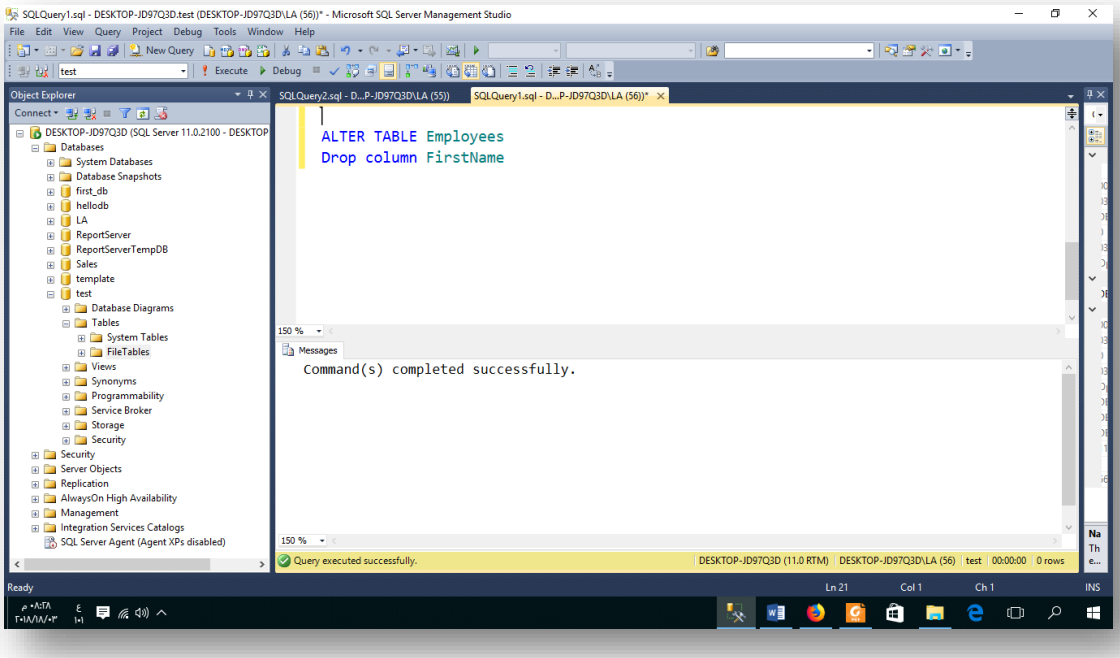
```
ALTER TABLE Employees
Alter column addres int NOT NULL;
```

```
Messages
Command(s) completed successfully.
```

- عدل الجدول السابق حيث تضيف عمود يمثل الشهادة الحاصل عليها الموظف.

```
alter table Employees
ADD
studying varchar(13) NULL
```

- من الجدول السابق احذف عمود اسم الشخص.



- احذف الجدول السابق.

```
Drop table Employees;
```

- ليكن لدينا جدول الموظفين والاقسام كما موضح بالشكل التالي

	dept_id	dept_name	dept_date
1	1	it	1990-03-17
2	2	resource	NULL
3	3	market	2000-05-22

e_id	e_name	age	address	dept_id	salary
1	lama	27	hama	1	10000
4	lana	36	damascus	1	20000
3	tala	23	hama	3	150
5	ahmed	37	hama	2	13566
6	esraa	22	hama	3	75894
NULL	NULL	NULL	NULL	NULL	NULL

المطلوب:

١. أنشئ قادح عند ادراج قسم جديد يقوم بطباعة البيانات المدرجة.

```
CREATE TRIGGER trgAfterInsert ON Departement
FOR INSERT
AS
  declare @id int;
  declare @name varchar(100);
  declare @date date;

  select @id=i.dept_id from inserted i;
  select @name=i.dept_name from inserted i;
  select @date=i.dept_date from inserted i;



  PRINT 'AFTER INSERT trigger fired for '+ ','+@name;

GO

insert into Departement values(5,'qqq','2010-11-11');
```

٢. أنشئ قادح عند حذف قسم يحذف جميع الموظفين الذين يعملون فيه.

```
CREATE TRIGGER trgAfterDelete ON Departement
FOR Delete
AS
      declare @id int;


      select @id=d.dept_id from Deleted d;
            Delete from Employee where dept_id=@id;

      PRINT 'after delete trigger was fired';

GO
```

```
delete from Departement where dept_id=3;
```

٣. أنشئ قادح يطبع مجموع رواتب الموظفين عند محاولة تعديل راتب موظف ما.

```
CREATE TRIGGER trgAfterUpdate ON Employee
FOR UPDATE
AS

	declare @empsal int;
	declare @sum int;


	select @empsal=i.salary from inserted i;


	if update(salary)
			set @sum=(select sum(salary) from Employee);
	PRINT @sum;
GO

update Employee set salary=400000 where e_id=3

update Employee set e_name='lolo' where e_id=3
```

٤. اجعل القادح السابق غير مفعل .

```
alter table Employee disable TRIGGER trgAfterUpdate
```

نلاحظ الآن عند محاولة تعديل راتب موظف لا يتم استدعاء القادح.

٥. احذف القادح السابق.

```
drop trigger trgAfterUpdate;
```

القيود و التكامل المرجعي

أنشئ جدول الأقسام حيث لكل قسم رقم ، اسم، رقم المدير.

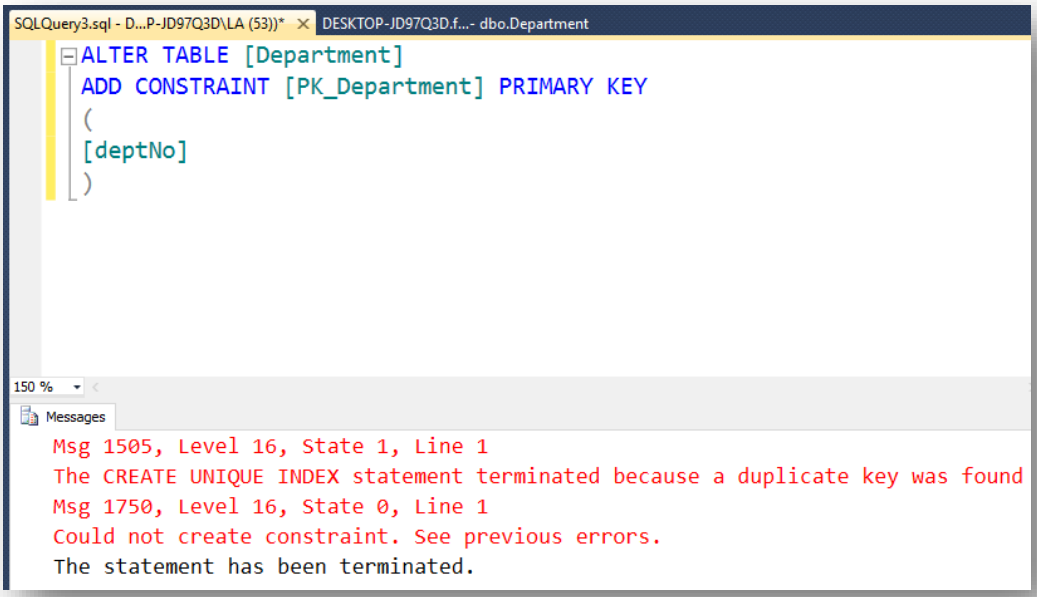
```
CREATE TABLE [Department]
(
[deptNo] [int] NOT NULL ,
[deptName] [varchar] (50) NOT NULL ,
[mangerSN] [int] NOT NULL
);
```

نلاحظ عند إضافة سجلين بنفس رقم القسم لا ينتج أي خطأ نظراً لعدم وجود أي قيد على الأعمدة.

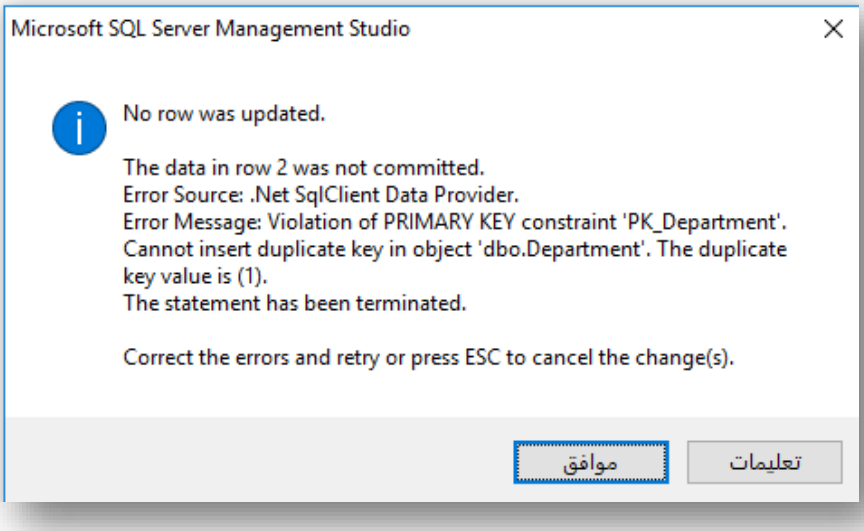
DESKTOP-JD97Q3D.f...- dbo.Department

	deptNo	deptName	mangerSN
	1	IT	9
	2	Sale	5
	1	humanResources	2
▶*	NULL	NULL	NULL

١. **قيد المفتاح الأولي** :عدل على الجدول السابق بحيث لا يسمح بإنشاء قسمين بنفس الرقم ولا يمكن أن يكون رقم القسم في أحد السجلات قيمته null لذلك نجعل رقم القسم مفتاح أساسي.
عند محاولة جعل رقم القسم في الجدول السابق مفتاح أساسي نلاحظ أنه ينتج خطأ وذلك لأننا قمنا سابقاً بإضافة قسمين بالرقم ١ ...



لذلك نحذف السجلات التي تم ادراجها في الخطوة السابقة ، ثم نضيف قيد المفتاح الأولي على رقم القسم.
والآن لنحاول إضافة سجلين بنفس رقم القسم.



٢. القيود الفريدة

عند إضافة قيد فريد لأحد الاعمدة هذا يدل:

- لا يمكن أن تتكرر قيمة هذا الحقل
- يمكن أن تكون قيمة هذا الحقل null

أضف قيد فريد على اسم القسم في الجدول السابق

```
ALTER TABLE [Department]
ADD CONSTRAINT [UK_deptName] UNIQUE
(
[deptName]
)
```

لحذف القيد الفريد

```
ALTER TABLE [Department]
DROP CONSTRAINT [UK_deptName]
```

٣. قيد المفتاح الثانوي

المفتاح الثانوي في جدول ما هو مفتاح أساسي في جدول آخر.
مثال: في كل قسم يوجد مجموعة موظفين، لذلك نضيف لكل موظف رقمه ، اسمه، الشهادة، رقم القسم الذي يعمل به.

```
CREATE TABLE [Employee] (
[empSN] [int] NOT NULL ,
[fName] [varchar] (50) ,
[lName] [varchar] (50) NOT NULL ,
[birthDate] [datetime] NULL ,
[hireDate] [datetime] NOT NULL ,
[address] [varchar] (50) NULL ,
[sex] [bit] NOT NULL ,
[salary] [money] NOT NULL ,
[managerSN] [int] NULL ,
[deptNo] [int] NULL
FOREIGN KEY REFERENCES Department(deptNO)
)
```

Foreign key 'FK__Employee__67112487fccc7284deb7' references invalid table 'Department'.

لنحاول حذف المفتاح الأساسي في جدول الأقسام

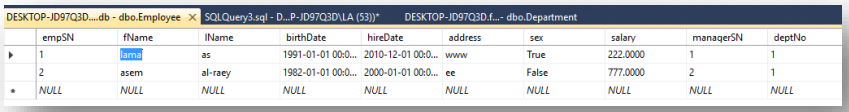
```
alter table Department
DROP CONSTRAINT PK_Department
```

```
Messages
Msg 3725, Level 16, State 0, Line 1
The constraint 'PK_Department' is being referenced by table 'Employee', foreign
Msg 3727, Level 16, State 0, Line 1
Could not drop constraint. See previous errors.
```

نلاحظ أنه لا يمكن حذف المفتاح الأساسي لأنه مرتبط بمفتاح ثانوي في جدول الموظفين.

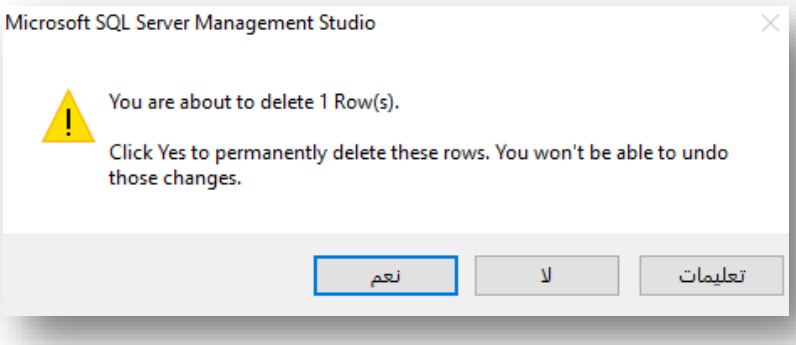
```
ALTER TABLE Employee
 DROP CONSTRAINT [FK Employee Departmet]
```

نقوم بادراج موظفين في جدول الموظفين وكلاهما يعمل في القسم رقم ١:



empSN	fName	lName	birthDate	hireDate	address	sex	salary	managerSN	deptNo
1	lama	as	1991-01-01 00:0...	2010-12-01 00:0...	www	True	222.0000	1	1
2	asem	al-raey	1982-01-01 00:0...	2000-01-01 00:0...	ee	False	777.0000	2	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

الآن عند محاولة حذف القسم رقم ١ أو تعديله من جدول الأقسام نحصل على رسالة الخطأ الموضحة بالشكل



يمكن استخدام الخاصيتان (On update cascade, on delete cascade) لتطبيق التغييرات التي تحدث في جدول الأقسام على جدول الموظفين بشكل تلقائي

```
ALTER TABLE [Employee]
ADD CONSTRAINT [FK_Employee_Department]
FOREIGN KEY([deptNo])
REFERENCES [Department] ([deptNo])
ON UPDATE CASCADE
ON DELETE CASCADE
```

والآن عند حذف القسم رقم ١ يقوم تلقائيا بحذف الموظفين التابعين لهذا القسم.

٤. **قيود التحقق :** يمكن فرض قيد للتحقق من صحة قيم الحقول عند إدخالها.
مثال : عند ادخال موظف جديد يجب التحقق من أن تاريخ تعيينه اكبر من تاريخ ميلاده وان راتبه لا يقل عن ١٠٠٠٠ ولا يتجاوز ٢٠٠٠٠.

```
alter table Employee
add constraint ck_salary check (salary between 10000 and 20000)
go
alter table Employee
ADD CONSTRAINT [CK_Employee_hireDate]
CHECK
((datepart(year,[hireDate])>=(datepart(year,[birthDate])+(
18))))
```

٥. **القيود الافتراضية**
لاعطاء قيمة افتراضية لاحد الحقول يمكن أن نستخدم القيد default.
مثال: لادراج موظف جديد يجب أن يكون راتبه ٢٠٠٠٠ عند عدم ادخال أي قيمة أخرى.

```
ALTER TABLE [Employee]
ADD CONSTRAINT [Employee] DEFAULT 10000 for [salary]
```

٦. **قيد NotNull** يستخدم هذا القيد لمنع ادخال حقول تملك القيمة null.
مثال اسم الموظف لا يجب أن يقوم null وعند محاولة ادخال موظف بدون اسم يتم إعطاء رسالة خطأ.

٧. **قيد Identity**
يمكن أن نجعل قيم حقل ما تبدأ بقيمة معينة ثم تتزايد بمقدار ما.
مثال:

أنشئ جدول الطلاب بحيث تبدأ ارقام الطلاب من ١٠٠ وتزداد بمقدار ١ عند ادراج أي طالب جديد.

```
CREATE TABLE Students (Name varchar(50) ,
ID INT IDENTITY (100,1) PRIMARY KEY NOT NULL);
```

- ليكن لدينا جدول الموظفين والاقسام كما موضح بالشكل التالي

	dept_id	dept_name	dept_date
1	1	it	1990-03-17
2	2	resource	NULL
3	3	market	2000-05-22

e_id	e_name	age	address	dept_id	salary
1	lama	27	hama	1	10000
4	lana	36	damascus	1	20000
3	tala	23	hama	3	150
5	ahmed	37	hama	2	13566
6	esraa	22	hama	3	75894
NULL	NULL	NULL	NULL	NULL	NULL

المطلوب:

١. أنشئ مؤشر يعيد اسم وعنوان كل موظف.

```
DECLARE @myaddress varchar(15);
 DECLARE @myName varchar(50);
  declare myCursor CURSOR FOR Select e_name, address from
Employee;
  OPEN myCursor;
  FETCH next from myCursor INTO @myName,@myaddress;
    WHILE @@FETCH_STATUS = 0
  begin
  PRINT @myName+' '+ @myaddress;
  FETCH NEXT from myCursor INTO  @myName,@myaddress ;
  end
```

٢. صرح عن مؤشر يعيد مجموع رواتب الموظفين بالقسم الأول.

```
 declare @sum int;
  declare sum_cursor CURSOR FOR select sum(salary) from
Employee where dept_id=1
  OPEN sum_cursor;
  FETCH next from sum_cursor into @sum;

  print @sum;
 close sum_cursor;
```

٣. عدل على أرقام الموظفين في جدول الموظفين بحيث يبدأ العد من ١.باستخدام المؤشرات

```
  declare @id int;
  set @id=1;
  declare myCursor5 CURSOR FOR select * from Employee
   OPEN myCursor5;
   FETCH next from myCursor5;

   while @@FETCH_STATUS=0
   begin
   update Employee set e_id=@id
   WHERE  CURRENT OF myCursor5;

    set @id = @id + 1;

    FETCH next from myCursor5;

   end
      close myCursor5;
```

٤. احذف الموظفين الذين يعملون بالقسم الثالث باستخدام المؤشرات.

```
 declare myCursor6 CURSOR FOR select * from Employee where
dept_id=3
  OPEN myCursor6;
  FETCH next from myCursor6;
```

```
    while @@FETCH_STATUS=0
    begin
    delete from Employee where current of myCursor6

    FETCH next from myCursor6;
    end
   close myCursor6;
```

٥. اطبع اسم كل موظف واسم القسم الذي يعمل به باستخدام المؤشرات

```
declare @Name varchar(15);
 declare @id int;
 declare @deptName varchar(14);
 declare emp_cur1 CURSOR FOR select e_name,dept_id
       from Employee ;
  OPEN emp_cur1;

  FETCH next from emp_cur1 into @Name,@id;
  while @@FETCH_STATUS=0
 begin
 declare  dept_cur1 CURSOR FOR select dept_name from
       Departement where dept_id=@id;
 open dept_cur1;
  FETCH next from dept_cur1 into @deptName;
  print @Name+' '+@deptName;

  FETCH next from emp_cur1 into @Name,@id;
 close dept_cur1;
 DEALLOCATE dept_cur1;
 end

       close emp_cur1;
       DEALLOCATE emp_cur1;
```