المحاضرة الرابعة

القيود والتكامل المرجعي

والفهارس

DB Constraints & Referential Integration

& Indexes

الهدف من الجلسة:

سوف نتعرف في هذه الجلسة على:

قيد المفتاح الأولي Primary Key Constraint.

القيود الفريدة Unique Constraints.

قيود التحقق Check Constraints.

القيود الإفتراضية Default Constraints.

قيد المفتاح الثانوي Foreign Key Constraint.

مقدمة

تمثل القيود منطق عمل Business Logic أو القواعد التي يقوم مخدم قواعد البيانات بضمان تحقيقه على حقول (أعمدة) جداول قاعدة البيانات. وتشمل هذه القيود مجال القيم التي يمكن إدخالها في عمود إضافة إلى التكامل المرجعي للبيانات.

عند إضافة أو حذف أو تعديل أي بيانات في جدول ما من جداول قاعدة بيانات، فإن مخدم قواعد البيانات يراعي جميع القيود المعرفة على هذه الجدول قبل تنفيذ العملية ولا ينفذ هذه العملية إذا كانت تؤدي إلى أي خلل في القيود المفروضة.

مثال: إذا كان لدينا مفتاح فريد Unique على جدول ما، فلا يمكنك إضافة أو تعديل سطر ما بحيث يصبح لديك تكرارات في قيمة المفتاح الفريد.

مثال: إذا كان لديك قيد على مجال القيم لعمود ما فإن إضافة أو تعديل قيمة هذا العمود لسطر ما خارج هذا المجال سوف تؤدي إلى توليد خطأ من قبل مخدم قواعد البيانات.

تراعي مخدمات قواعد البيانات التكاملات المرجعية المعرفة بين الجداول، فلا يمكنك مثلا إضافة سطر في الجدول الممثل للطرف كثير إن لم يكن هناك سطر موافق في الجدول الممثل للطرف واحد.

وبالمثل: فإنه لا يمكنك حذف سطر من الطرف واحد ما لم تكن جميع الأسطر الموافقة له في الطرف كثير قد حذفت أو أنه لا يوجد أي سطر موافق في الطرف كثير. (يوجد استثناءات لهذه القاعدة؟)

تمهيد:

ننشئ قاعدة المعطيات SVU:

```
Create database svu
```

ليكن لدينا الجدول Department:

```
CREATE TABLE Department
(
  deptNo int,
  deptName varchar (50),
  mangerSN int,
  managerStartDate datetime
)
```

نضيف الأسطر إلى الجدول:
سنستخدم هذه الأسطر لتجربة مفعول القيود لاحقاً.

```
Insert into Department (deptNo, deptName, mangerSN, managerStartDate)
        Values (1, 'Delivery',1, '1/1/2006')

Insert into Department (deptNo, deptName, mangerSN, managerStartDate)
        Values (1, 'Development',2, '1/1/2004')

Insert into Department (deptNo,deptName, mangerSN, managerStartDate)
        Values (NULL,'EXAMS',8, '1/1/2003')
```

```
Insert into Department (deptNo, deptName, mangerSN, managerStartDate)
        Values (2, 'Development',3, '1/1/2005')

Insert into Department (deptNo, mangerSN, managerStartDate)
        Values (3,2, '1/1/2004')
```

نلاحظ أن عملية الاضافة تتم مع أنها مخالفة لمنطق العمل،
الآن لنتعرف كيف نطبق القيود لنجعل منطق العمل يتماشي مع عمليات الاضافة والتعديل

نلاحظ أن:
عدم وجود قيد مفتاح أولي يسمح بتكرار القيم في الحقل أو اعطاء قيمة معدومة NULL 1-2-3
عدم وجود قيد وحدانية يسمح بتكرار القيم في الحقل - الأسطر 2- 4
عدم وجود قيد NOT NULL يسمح بترك قيم الحقل فارغة – السطر 5

لنحذف الجدول بتعليمة `Drop Table Department`
ثم نقوم باضافة القيود على التتالي.

قيد المفتاح الأولي Primary Key Constraint.

يضمن قيد المفتاح الأولي عدم تكرار القيم في مجموعة الأعمدة المكونة له كما يمنع إدخال القيمة NULL في أي من أعمدته. وهو يستخدم لضمان الوحدانية. فمثلا في الجدول Authors من قاعدة البيانات Pubs فإن العمود au_id هو مفتاح أولي يضمن وحدانية رقم كل مؤلف من جهة ومن جهة أخرى يضمن عدم إسناد قيمة NULL لهذا العمود من أجل أي مؤلف.

أمثلة:

تحديد المفتاح الأولي عند إنشاء الجدول: لإنشاء الجدول Department، مع ضمان وحدانية وعدم انعدام قيمة رقم القسم ، نكتب التعليمة التالية:

```
CREATE TABLE Department
(
  deptNo int  Primary Key,
  deptName varchar (50)  NOT NULL ,
  mangerSN int NOT NULL,
  managerStartDate datetime NULL
)
```

نضيف الأسطر 1 -2 -6 إلى الجدول للتجربة.

```
Insert into Department (deptNo, deptName, mangerSN, managerStartDate)
        Values (1, 'Delivery',1, '1/1/2006')

Insert into Department (deptNo, deptName, mangerSN, managerStartDate)
        Values (1, 'Development',2, '1/1/2004')

Insert into Department (deptNo,deptName, mangerSN, managerStartDate)
        Values (NULL,'EXAMS',8, '1/1/2003')
```

تحديد المفتاح الأولي بعد إنشاء الجدول: في المثال السابق تم إنشاء الجدول وتعريف المفتاح الأولي بشكل مباشر. يمكن إنشاء الجدول ومن ثم تعريف المفتاح الأولي على هذا الجدول (إذا احتوى الجدول على بيانات فإن توليد المفتاح الأولي سينجح فقط إذا لم تحتوِ أعمدة المفتاح الأولي على قيم مكررة وإذا لم يكن الحقل يسمح بقيم معدومة Null).

ملاحظة: يمكن أن يحوي الجدول على مفتاح أولي واحد على الأكثر.

```
DROP TABLE Department
GO
CREATE TABLE Department
(
  deptNo int NOT NULL,
  deptName varchar (50)  NOT NULL ,
  mangerSN int NOT NULL,
  managerStartDate datetime NULL
)
```

لإضافة القيد بعد إنشاء الجدول نعدل تعريف الجدول كما يلي:

```
ALTER TABLE Department
ADD CONSTRAINT PK_Department PRIMARY KEY
( deptNo )
```

وفي هذه الحالة يمكن إلغاء المفتاح الأولي في أي لحظة دون أي ضياع في البيانات . (وذلك لأننا قمنا بتحديد اسم المفتاح بشكل صريح).

لحذف قيد معرف على جدول نستخدم التعليمة التالية:

```
ALTER TABLE Department
DROP CONSTRAINT PK_Department
```

ملاحظة:

- لا نستطيع اضافة قيد مفتاح أولي (أساسي) على حقل ليس عليه قيد NOT NULL
- لا نستطيع اضافة قيد مفتاح أولي (أساسي) على حقل تم ادخال قيم مكررة فيه سابقا

القيود الفريدة Unique Constraints

يضمن القيد الفريد عدم تكرار القيم في الأعمدة المكونة له بين الأسطر المختلفة للجدول. لكنه يسمح بإدخال قيمة NULL في أي من أعمدته.

ملاحظة : يمكن للجدول أن يحتوي على أكثر من قيد فريد، كما يمكن لنفس العمود (الحقل) أن يشارك بعدة مفاتيح فريدة.

في الجدول السابق نلاحظ أن اسم القسم يجب أن لا يتكرر وبالتالي نعدل الجدول السابق بحيث نضيف قيد فريد على اسم القسم.

تحديد قيد فريد أثناء إنشاء الجدول:

```
DROP TABLE Department
GO
CREATE TABLE Department
(
  deptNo int NOT NULL Primary Key,
  deptName varchar (50) NOT NULL UNIQUE,
  mangerSN int NOT NULL,
  managerStartDate datetime NULL
)
```

نضيف الأسطر إلى الجدول للتجربة.

```
Insert into Department (deptNo, deptName, mangerSN, managerStartDate)
                Values (1, 'Development',2, '1/1/2004')
Insert into Department (deptNo, deptName, mangerSN, managerStartDate)
                Values (2, 'Development',7, '1/1/2005')

Insert into Department (deptNo, deptName, mangerSN, managerStartDate)
Values (3, NULL,7, '1/1/2005')
```

```
Insert into Department (deptNo, deptName, mangerSN, managerStartDate)
Values (4, NULL , 9 , '1/1/2005')
```

تحديد قيد فريد بعد إنشاء الجدول:

يمكننا حذف القيود الفريدة في أي لحظة شريطة معرفة اسم هذا القيد كما تبين التعليمة التالية:

```
DROP TABLE Department
GO
CREATE TABLE Department
(
  deptNo int NOT NULL Primary Key,
  deptName varchar (50)  NULL ,
  mangerSN int NOT NULL,
  managerStartDate datetime NULL
)
GO
ALTER TABLE Department
ADD CONSTRAINT UK_deptName UNIQUE
(deptName)
```

لحذف القيد السابق:

```
ALTER TABLE Department
DROP CONSTRAINT UK_deptName
```

قيود التحقق Check Constraints.

تفرض قيود التحقق شروطا على القيمة value التي يمكن أن توضع في عمود أو مجموعة أعمدة أو تفرض قيودا على تنسيق format القيمة التي توضع في عمود أو مجموعة أعمدة.

مثلا في جدول الموظفين: لكل موظف تاريخ ميلاد birthDate وتاريخ توظيف hireDate. يقبل مخدم قواعد البيانات أي تاريخ للعمودين دون أي قيود. لفرض قيد كون تاريخ الميلاد أكبر من 1950 و كون تاريخ التوظيف أكبر من تاريخ الميلاد + 18 سنة. نعرف القيدين التاليين:

```
ALTER TABLE Employee
      DROP CONSTRAINT CK_Employee_birthDate
GO
ALTER TABLE Employee
      DROP CONSTRAINT CK_Employee_hireDate

GO
ALTER TABLE Employee
      ADD  CONSTRAINT CK_Employee_birthDate
                              CHECK  ((datepart(year,birthDate) > 1950 )
GO
ALTER TABLE Employee
      ADD  CONSTRAINT CK_Employee_hireDate
   CHECK  ((datepart(year,hireDate)>=(datepart(year,birthDate)+(18))))
```

يجب وضع القيد بعد تعريف الجدول إذا كان يشمل على اكثر من عمود (لا يمكن وضعه مع عمود ما)

تجدر الإشارة إلى أن جميع الأعمدة المشاركة في قيد تحقق يجب أن تكون من نفس الجدول (سنرى لاحقا كيفية فرض قيود تحقق تشمل حقول من عدة جداول في جلسات لاحقة).

القيود الإفتراضية: Default Constraints

```
ALTER TABLE Department

   ADD CONSTRAINT def_Department DEFAULT 5 for mangerSN
```

ندخل السطر التالي:

```
   Insert into  Department (deptNo, deptName,  managerStartDate)

                  Values  (7, 'SA', '1/1/2004')

   Select * from Department
```

نلاحظ قيمة الحقل mangerSN

لنفترض أننا ندخل بيانات عدد كبير من الموظفين في اليوم الواحد. سيكون إدخال تاريخ التوظيف موحدا لجميع هؤلاء الموظفين وبالتالي سيكون عملا تكراريا. يمكن افتراض أن جميع الموظفين المضافين في هذا

اليوم لهم تاريخ توظيف موافق لتاريخ اليوم الحالي وبالتالي نضع قيد قيمة افتراضية على العمود .hireDate

```
DROP TABLE Employee
GO
CREATE TABLE Employee
(
     empSN int CONSTRAINT  Ttestt  primary key,
     fName varchar(50)  NULL,
     lName varchar(50) CONSTRAINT test_notNull NOT NULL,
     birthDate datetime NULL CHECK
                ((datepart(year,birthDate)>(1950))),
     hireDate datetime NOT NULL CONSTRAINT  def_testttt DEFAULT
                                                (getdate()) ,
     address varchar(50)  NULL,
     sex bit NOT NULL,
     salary money NOT NULL,
     managerSN int NULL,
     deptNo int NULL,
)
```

قمنا بوضع القيد القادم بعد تعريف الجدول لانه مرتبط باكثر من عمود وبالتالي لا يمكن وضعه مع عمود ما أثناء بناء الجدول. أو يمكن وضعه أثناء بناء الجدول ولكن بعد الانتهاء من تعريف جميع الأعمدة.

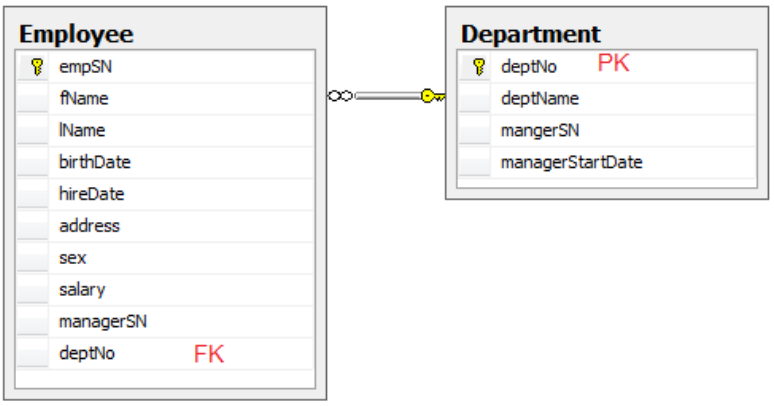
```
GO
ALTER TABLE Employee
     ADD  CONSTRAINT CK_Employee_hireDate
   CHECK ((datepart(year,hireDate)>=(datepart(year,birthDate)+(18))))
```

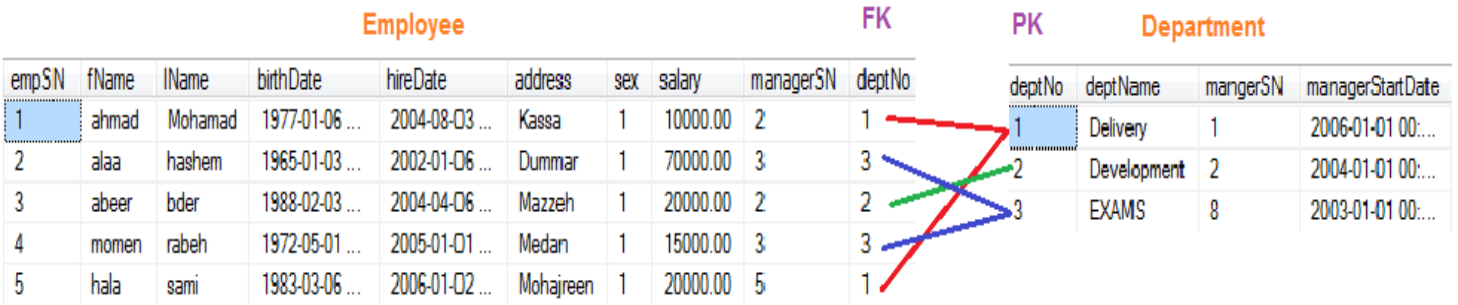
قيد المفتاح الثانوي Foreign Key Constraint.

يعمل المفتاح الثانوي بالتزامن مع مفتاح أولي أو مع مفتاح فريد Unique. يضمن إنشاء مفتاح ثانوي في جدول ما كون القيم الممكن إدخالها في أعمدة هذا المفتاح مطابقة لبيانات المفتاح الأولي أو الفريد الموافق. تسمح هذه التقنية بالحد من تكرار البيانات في قاعدة المعطيات، فمثلا نكتفي بإدخال بيانات الأقسام لمرة واحدة ثم نربط جدول الموظفين بجدول الأقسام لتحديد القسم الذي يتبع له كل موظف عن طريق تحديد رقم

القسم مع كل موظف. كما أن المفتاح الثانوي يضمن تكامل المعطيات حيث يتحقق مخدم البيانات من كون الرقم المدخل في حقل رقم القسم موجود فعلا في جدول الأقسام.

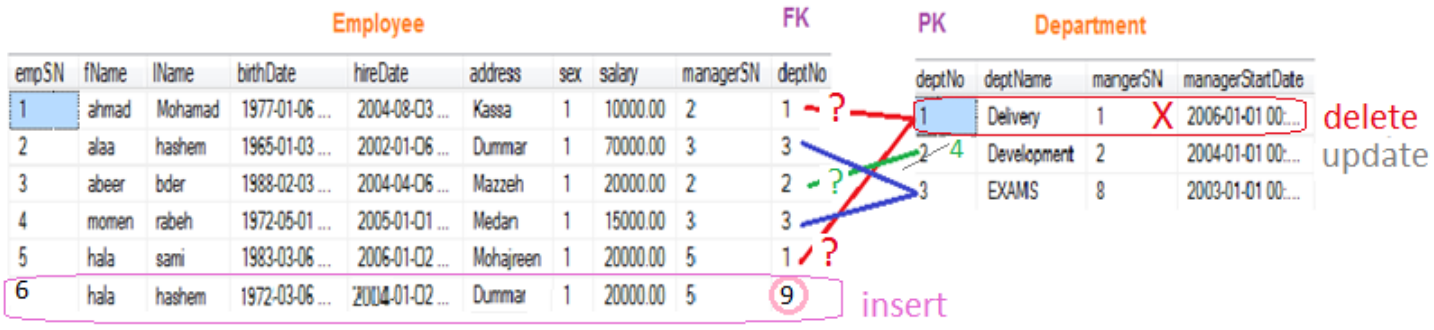
تجدر الإشارة إلى أنه لا يمكننا بعد تعريف المفتاح الخارجي أن نقوم بحذف المفتاح الرئيسي (أو الفريد) الموافق له ما لم نقم أولا بحذف المفتاح الخارجي.





empSN	fName	lName	birthDate	hireDate	address	sex	salary	managerSN	deptNo
1	ahmad	Mohamad	1977-01-06 ...	2004-08-03 ...	Kassa	1	10000.00	2	1
2	alaa	hashem	1965-01-03 ...	2002-01-06 ...	Dummar	1	70000.00	3	3
3	abeer	bder	1988-02-03 ...	2004-04-06 ...	Mazzeh	1	20000.00	2	2
4	momen	rabeh	1972-05-01 ...	2005-01-01 ...	Medan	1	15000.00	3	3
5	hala	sami	1983-03-06 ...	2006-01-02 ...	Mohajreen	1	20000.00	5	1

deptNo	deptName	mangerSN	managerStartDate
1	Delivery	1	2006-01-01 00:...
2	Development	2	2004-01-01 00:...
3	EXAMS	8	2003-01-01 00:...



empSN	fName	lName	birthDate	hireDate	address	sex	salary	managerSN	deptNo
1	ahmad	Mohamad	1977-01-06 ...	2004-08-03 ...	Kassa	1	10000.00	2	1
2	alaa	hashem	1965-01-03 ...	2002-01-06 ...	Dummar	1	70000.00	3	3
3	abeer	bder	1988-02-03 ...	2004-04-06 ...	Mazzeh	1	20000.00	2	2
4	momen	rabeh	1972-05-01 ...	2005-01-01 ...	Medan	1	15000.00	3	3
5	hala	sami	1983-03-06 ...	2006-01-02 ...	Mohajreen	1	20000.00	5	1
6	hala	hashem	1972-03-06 ...	2004-01-02 ...	Dummar	1	20000.00	5	9

deptNo	deptName	mangerSN	managerStartDate
1	Delivery	1	2006-01-01 00:...
2	Development	2	2004-01-01 00:...
3	EXAMS	8	2003-01-01 00:...

إنشاء مفتاح ثانوي عند إنشاء الجدول: لإنشاء جدول الموظفين بحيث يشير الحقل **deptNo** إلى الحقل الذي يحمل نفس الاسم في الجدول **Department** ننفذ التعليمة التالية:

```
DROP TABLE Department
```

```
GO
CREATE TABLE Department
(
  deptNo int NOT NULL,
  deptName varchar (50)  NULL ,
  mangerSN int NOT NULL,
  managerStartDate datetime NULL
)
GO
ALTER TABLE Department
ADD CONSTRAINT PK_Department PRIMARY KEY (deptNo)

GO
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
                FOREIGN KEY REFERENCES Department(deptNO)
            On delete cascade
            On update set null
    )
```

لاضافة سجلات في الجدولين:

```
        -- Insert into Department
Insert into Department Values (1, 'Delivery',1, '1/1/2006')
Insert into Department Values (2, 'Development',2, '1/1/2004')
Insert into Department Values (3,'EXAMS',8, '1/1/2003')
        -- Insert into Employee
```

```
Insert into Employee Values (1, 'ahmad', 'mohamad', '6/1/1977', '3/8/2004',
'Kasa',1,10000,2,1)
Insert into Employee Values (2, 'alaa', 'hashem', '3/1/1965', '6/1/2002',
'Dummar',1,70000,3,3)
Insert into Employee Values (3, 'abeer', 'bder', '3/2/1988', '6/4/2004',
'Mazzeh',1,20000,2,2)
Insert into Employee Values (4, 'momen', 'rabeh', '1/5/1972', '1/1/2005',
'Medan',1,15000,3,3)
Insert into Employee Values (5, 'hala', 'sami', '6/3/1988', '2/1/2006',
'Mohajreen',1,20000,5,1)
Insert into Employee Values (6, 'hala', 'hashem', '6/3/1972', '2/1/2004',
'Dummar',1,20000,5,9)
```

ننفذ التعليمات السابقة نجد خطأ عند اضافة الموظف رقم 6 (لماذا؟)

```
Delete from Department where deptNo=1
```

لماذا تعطي خطأ عند التنفيذ؟

```
Update Department set deptNo=4 where deptNo=2
```

إنشاء مفتاح ثانوي بعد إنشاء الجدول: يمكن بنفس الشكل بناء مفتاح ثانوي بعد إنشاء الجدول وفي هذه الحالة يجب أن تكون البيانات الموجودة حاليا في حقول المفتاح الثانوي تكافي بيانات موجودة في حقل (أو حقول) المفتاح الأولي أو الفريد الموافق وإلا فإن عملية الإنشاء تكون فاشلة.

```
DROP TABLE Employee
DROP TABLE Department
GO
CREATE TABLE Department
(
  deptNo int NOT NULL,
  deptName varchar (50)  NULL ,
  mangerSN int NOT NULL,
  managerStartDate datetime NULL
)
 go
ALTER TABLE Department
ADD CONSTRAINT PK_Department PRIMARY KEY (deptNo)

GO
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     )
GO
ALTER TABLE Employee [with check | nocheck]
    ADD CONSTRAINT FK_Employee_Department
    FOREIGN KEY (deptNo)       REFERENCES Department(deptNo)
```

On update cascade

On delete set null

لحذف مفتاح ثانوي:

```
ALTER TABLE Employee
                DROP CONSTRAINT FK_Employee_Departmet
```

يمكن حذف المفتاح الثانوي في أي وقت بدون أي اعتبارات إضافية.

1. **التحديث التلقائي لبيانات المفتاح الثانوي:**

ماذا يحدث إذا حذفنا قسما وكان هناك موظفين في هذا القسم؟

وماذا يحدث لو عدلنا رقم أحد الأقسام الذي يحوي موظفين؟

مبدئيا لا يمكن حذف قسم يحوي موظفين، كما لايمكن تعديل رقم قسم يحوي موظفين. إذا للقيام بأي من هاتين العمليتين لابد من حذف جميع الموظفين أولا قبل تنفيذ أي عملية على الأسطر الموافقة من جدول الأقسام.

لحل المشكلتين السابقتين نعرف واصفتين لعملية تعريف المفتاح الثانوي:

ON UPDATE CASCADE : تعديل بيانات القسم ينتقل بشكل آلي إلى جميع الموظفين المرتبطين بهذا القسم.

ON DELETE CASCADE: حذف أي قسم يؤدي إلى حذف جميع الموظفين في هذا القسم.

هنك واصفات أخرى :

ON UPDATE {SET DEFAULT| NO ACTION|SET NULL|CASCADE}

ON DELETE {SET DEFAULT| NO ACTION|SET NULL|CASCADE}

مثال:

```
ALTER TABLE Employee
      DROP CONSTRAINT FK_Employee_Department
GO
```

لمعرفة جميع القيود في قاعدة بيانات:

```
SELECT * FROM sys.objects
WHERE type_desc like '%CONSTRAINT%'
```

الفهارس Indexes

الغرض من الفهارس هو رفع أداء قاعدة البيانات أثناء عمليات الاستعلام و البحث.

المبدأ: إنشاء آلية لتسريع عملية تحديد لسجل من مجموعة سجلات. تعتمد هذه الآلية على محرك البيانات وهي إنشاء جداول فهرسة لربط قيمة الحقل الذي تمت إضافة الفهرس إليه مع موقع السجل الخاص بهذه القيمة.

ترفع الفهارس الأداء في حالة الاستعلام و البحث و ليس التعديل لذلك يجب مراعاة فهرسة الحقول التي تتم عليها الكثير من عمليات البحث.

ملاحظة: عادة ما يقوم محرك البيانات بإنشاء فهارس تلقائية للمفاتيح الرئيسية.

انشاء الفهرس:

تستعمل تعليمة Create index لانشاء الفهرس وللفهارس بصورة أساسية نوعين:

الفهارس البسيطة

```
CREATE INDEX index_name
ON table_name
(       column_name1    )
```

ويمكن فيها أن تتكرر القيم في عمود (أو أعمدة) الفهرسة.

الفهارس الفريدة

```
CREATE UNIQUE INDEX index_name
ON table_name
(    column_name1
)
```

وتختلف عن الفهارس البسيطة في أنها لا تسمح بتكرار القيم في عمود (أو أعمدة) الفهرسة

نريد الآن أن ننشئ فهرسا يشمل اسم وكنية كل موظف في جدول الموظفين. نستخدم التعليمة التالية:

```
Use northwind
Go
CREATE Index idx_employee_FullName
On Employees   ( FirstName , LastName )
```

يؤدي إنشاء هذا الفهرس إلى تسريع عمليات الاستعلام التي تشمل شروطا على اسم الموظف فقط أو على اسم الموظف وكنيته (هذا الفهرس غير مفيد للاستعلامات التي تشمل فقط كنية الموظف).

مثلا قد يكون الفهرس السابق مفيدا لتسريع الاستعلام التالي:

```
select * from Employees
where      FirstName like 'a%' and  LastName like 'd%'
```

الذي يعيد قائمة ببيانات الموظفين الذي يبدأ اسمهم بالحرف a وكنيتهم بالحرف d وهنا قد يستخدم المحرك الفهرس idx_employee_FullName لتسريع عملية تنفيذ هذا الاستعلام لكون جميع حقول الفلترة موجودة ضمن هذا الفهرس. كما يمكن للمحرك أن يستخدم هذا الفهرس لو كان هناك فلترة على الاسم فقط دون الكنية.

تعديل الفهرس:

نستخدم تعليمة Alter index لتعديل فهرس.

لتعديل الفهرس السابق وتعطيل عمله:

```
ALTER INDEX idx_employee_FullName ON Employees
DISABLE
```

حذف الفهرس:

نستخدم التعليمة drop index

لحذف الفهرس السابق ننفذ التعليمة التالية:

```
Use northwind
Go
DROP INDEX Employees.idx_employee_FullName
```

انتهت المحاضرة