جامعة المنارة
كلية الهندسة
قسم الهندسة المعلوماتية

مقرر قواعد بيانات1

لغة تعريف البيانات DDL

د.م.كندة أبو قاسم

م.جمال محمود – م.جهاد عيسى

جلسة الأسبوع السادس

(الفصل الثاني 2023 | 2024)

الغاية من الجلسة

- ✓ التعريف بلغة التعامل مع البيانات DDL.
- ✓ إنشاء قواعد البيانات وتعديلها وحذفها.
- ✓ إنشاء الجداول وتعديلها وحذفها.
- ✓ التعريف بالقيود وأساليب إدارتها.

لغة تعريف البيانات DDL

DDL أو Data Definition Language هي مجموعة جزئية من أوامر لغة SQL تستخدم لإنشاء وتعديل وحذف أغراض البيانات في قاعدة البيانات.

تتضمن هذه المجموعة، التعليمات التالية:

CREATE: وهي مخصصة لإنشاء أغراض جديدة في قاعدة البيانات.

DROP: وهي مخصصة لحذف غرض من قاعدة البيانات.

ALTER: وهي مخصصة لتعديل غرض في قاعدة البيانات.

تحوي كل قاعدة البيانات على العديد من الأغراض Database Objects ومن أهم هذه الأغراض:

- المستخدمون USERS
- الجداول TABLES
- القيود CONSTRAINTS
- الفهارس INDEXES
- المناظير VIEWS
- الاجرائيات المخزنة STORED PROCEDURES.
- التوابع المعرفة من قبل المستخدم USER-DEFINED FUNCTIONS.
- القادحات TRIGGERS.

البنية الفيزيائية لقاعدة البيانات

تتكون كل قاعدة بيانات في محرك مايكروسوفت من الملفات التالية:

1. **الملف الأساسي Primary File:**

يجب أن تحتوي أي قاعدة بيانات على ملف أساسي Primary واحد وواحد فقط (ملف لاحقته المقترحة من مايكروسوفت mdf) ويحوي هذا الملف البيانات الأساسية لقاعدة البيانات أو ما نطلق عليه اسم البيانات السامية لقاعدة البيانات Metadata التي تشتمل على تعاريف مختلف الأغراض المخزنة في قاعدة البيانات. كما يمكن للملف الأساسي أن يحوي أي عدد من أغراض قاعدة البيانات أي يمكن بناء الجداول و الأغراض الأخرى الخاصة بالتطبيقات ضمنه

2. **الملفات الثانوية Secondary Files**

يمكن لأي قاعدة بيانات أن تحوي ملف ثانوي واحد أو أكثر وهي ملفات مخصصة لتخزين بيانات أغراض قاعدة البيانات (بيانات الجداول والفهارس على سبيل المثال) ولكنها لا تحوي على بيانات سامية. لاحقة هذه الملفات ndf

3. **ملفات المناقلات Transaction Log Files**

وهي ملفات لتخزين تسلسل العمليات المنفذة على القاعدة ونسخ البيانات قبل وبعد أي عملية. تتميز هذه الملفات بصغر حجمها عموما الأمر الذي يسرع من عمليات القراءة والكتابة فيها لاحقة هذا النوع من الملفات ldf.

يتم إنشاء قاعدة بيانات بالتعليمة Create Database وحذفها بالتعليمة Drop Database وتعديلها بالتعليمة Alter Database. وسنعرض فيما يلي أمثلة عن الإنشاء بأشكاله المختلفة إضافة إلى مثال عن الحذف والتعديل.

إنشاء قاعدة بيانات

إنشاء قاعدة بيانات بالإعدادات الافتراضية

أبسط طريقة لإنشاء قاعدة بيانات هي باستخدام الإعدادات الافتراضية للمخدم وفيها لا نحدد أي ملفات ليقوم عندها المخدم تلقائيا بإنشاء الملفات الأساسية الكافية لعمل هذه القاعدة.

لنفترض أننا نريد إنشاء قاعدة اسمها HelloDB بالإعدادات الافتراضية. التعليمة الموافقة:

```
CREATE DATABASE HelloDB;
```

يقوم المخدم تلقائيا بإنشاء ملفين موافقين للقاعدة الجديدة بحيث يشتق اسم هذين الملفين من اسم القاعدة وهذين الملفين هما:

- الملف الرئيسي HelloDB.mdf بحجم أولي مقداره 8 ميغابايت وحجم نهائي غير محدود ومعدل نمو مقداره 1 ميغابايت.
- ملف المناقلات HelloDB_Log.ldf بحجم أولي مقداره 8 ميغابايت وحجم نهائي غير محدود ومعدل نمو مقداره 10% من الحجم الحالي لهذا الملف.

يتم إنشاء الملفات السابقة ضمن المجلد MSSQL/DATA الواقع ضمن مسار تثبيت محرك قواعد البيانات.

إنشاء قاعدة بيانات ضمن مجلد خاص مع تسمية الملفات المكونة لها:

لنفترض أننا نريد أن ننشئ قاعدة بيانات منطقية اسمها Sales بحيث تكون ملفاتها مخزنة على المسار الفيزيائي D:\sql_server. نريد أن تحوي هذه القاعدة على ملف أساسي اسمه MySales و ملف ثانوي اسمه data_sales وملف مناقلات وحيد اسمه MySalesLog.

```
CREATE DATABASE Sales
ON  (
        NAME = Sales_dat1,
        FILENAME = 'D:\sql_server\MySales.mdf',
        SIZE = 10,
        MAXSIZE = UNLIMITED,
        FILEGROWTH = 5
       ),
       (
         NAME= Sales_dat2,
         FILENAME = 'D:\sql_server\data_sales.ndf',
         SIZE = 5MB,
         MAXSIZE = 15,
         FILEGROWTH = 3
       )
LOG ON (
              NAME = 'Sales_log',
              FILENAME = 'D:\sql_server\MySalesLog.ldf',
               SIZE = 5MB,
              MAXSIZE = 25MB,
               FILEGROWTH = 10%
            )
```

نلاحظ أن لكل ملف العديد من المواصفات أهمها:

1. الاسم المنطقي Logical Name الذي تحدده الواصفة NAME وهو اسم للاستخدام المنطقي للملف ضمن المحرك بصرف النظر عن اسمه الفيزيائي ومكان تخزينه
2. الاسم الفيزيائي للملف File Name وهو ما تحدده الواصفة FileName وتحوي المسار الكامل للملف.
3. الحجم البدائي للملف Size ويمثل الحجم الأولي المحجوز للملف مقدرة افتراضيا بالميغابايت.

4. الحجم الأعظمي Max Size وهو ما تحدده الواصفة MaxSize مقدرة بالميغابايت، وإذا أردنا أن يكون حجم هذا الملف غير محدودا نستخدم القيمة الخاصة Unlimited لنسمح للملف بالنمو تلقائيا بما تحدده الواصفة FileGrowth .
5. معدل النمو File Growth وهو ما تحدده الواصفة FileGrowth والتي يمكن أن تكون قيمتها إما مطلقة أو نسبية. تتم زيادة حجم الملف بعد امتلاء الحجم الحالي كاملا بالبيانات ويتم حساب الحجم الجديد وفق إحدى القاعدتين:
   i. إذا كان النمو مطلقا: الحجم الجديد = الحجم الحالي + معدل النمو
   ii. إذا كان النمو نسبيا: الحجم الجديد = الحجم الحالي * (1 + معدل النمو)

تنمو جميع ملفات البيانات نموا متجانسا بحيث يتم التخزين بشكل يتناسب مع حجوم كل من هذه الملفات.

تعديل قاعدة بيانات

لتعديل أي قاعدة بيانات نستخدم التعليمة ALTER DATABASE.

لنفترض أننا نريد تعديل قاعدة البيانات HelloDB بحيث يصبح اسمها TestDB فإننا نكتب:

```
ALTER DATABASE HelloDB
MODIFY Name=TestDB;
```

حذف قاعدة بيانات

لحذف أي قاعدة بيانات نستخدم التعليمة DROP DATABASE.

لنفترض أننا نريد حذف قاعدة البيانات TestDB فإننا نكتب:

```
DROP DATABASE TestDB;
```

ملاحظة:

يمكن استخدام التابع sp_helpdb للاستعلام عن قاعدة بيانات ما وفق الصيغة: **Sp_helpdb database_name;** **مثال:** **Sp_helpdb Pubs;** **حيث يتم عرض معلومات عن قاعدة البيانات وملفاتها ومواصفاتها.**

الجداول TABLES

قبل البدء ببناء الجداول لنتعرف على أتواع البيانات في sql server

Exact numeric	Bigint, numeric, bit, smallint, decimal ,smallmoney, int, tinyint ,money
Approximate numerics	Float, real
Date and time	Date,datetimeoffset,datetime2,smalldatetime,datetime,time
Character strings	Char,varchar,text
Unicode character strings	Nchar,nvarchar,ntext
Binary strings	Binary,varbinary,image
Other data types	Cursor,rowversion,hierarchyid,uniqueidentifier ,sql_variant,xml,Spatial Geometry Types Spatial Geography Types,table

قواعد التسمية في SQL SERVER

- ✓ يجب تجنب استخدام كلمات SQL الأساسية والكلمات المحجوزة.
- ✓ ينبغي إعطاء الجداول والأعمدة أسماء واضحة وسهلة الفهم ومحددة بشكل فريد

- ✓ أعمدة SQL وأسماء الجداول يجب أن تبدأ بحرف، ولا تنتهي بشرطة سفلية، ويجب أن تحتوي على أحرف أبجدية رقمية فقط.
- ✓ الأسماء المكونة من مقطعين يجب وضعها ضمن أقواس متوسطة [ ]

إنشاء الجدول

نستخدم الأمر CREATE TABLE لإنشاء الجدول وفق الصيغة العامة التالية:

```
CREATE TABLE table_name(
        Column1 datatype,
        Column2 datatype,
        Column3 datatype,
        ........
);
```

مثال:

```
CREATE DATABASE Manara
USE Manara
Go
CREATE TABLE Department
(
 deptNo  int ,
 deptName    varchar (50) ,
 [manger SN] [int],
 [managerStartDate] [datetime]
);
```

تعديل الجدول

نستخدم الأمر ALTER TABLE لتعديل الجدول (تعديل حقل – إضافة حقل – حذف حقل).

يأخذ الأمر الصيغة العامة التالية:

```
ALTER TABLE table_name
    ADD|ALTER|DROP column_name datatype;
```

مثال1- لإضافة حقول إلى الجدول Department:

```
ALTER TABLE Department
    ADD  Min_Salary  money, [employee Nb]  int ;
```

مثال 2- لتعديل حقل في الجدول Department

```
ALTER TABLE table_name
    ALTER COLUMN deptName varchar(100);
```

مثال3 – حذف حقل في الجدول Department:

```
ALTER TABLE Department
    DROP COLUMN  [Min_Salary], [employee Nb] ;
```

حذف الجدول

نستخدم الأمر DROP TABLE لحذف الجدول

يأخذ الأمر الصيغة العامة التالية:

```
DROP TABLE table_name
```

مثال1- لحذف الجدول Department

```
DROP TABLE [Department]
```

أنواع القيود التي يمكن تطبيقها على أعمدة الجدول كما يلي

- ✓ قيد المفتاح الأولي Primary Key Constraint.
- ✓ القيود الفريدة Unique Constraints.
- ✓ قيود التحقق Check Constraints.
- ✓ القيود الإفتراضية Default Constraints.
- ✓ قيد المفتاح الثانوي Foreign Key Constraint.

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.

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؟)

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 و يستخدم كمعرف للسجل.

مثال1- 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
)
```

لنحاول إضافة الأسطر التالية 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')'
```

نلاحظ أنه تم قبول ادخال البيانات من التعليمة الأولى و رفض التعليمتين التاليتين لماذا ؟؟

مثال2- 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.

لإضافة القيد بعد إنشاء الجدول نعدل تعريف الجدول كما يلي:

```
ALTER TABLE Department
 deptNo int  Primary Key,
 ADD CONSTRAINT PK_Department PRIMARY KEY ( deptNo )
```

مثال3- لحذف قيد معرف على الجدول نستخدم الأمر التالي:

```
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.

مثال1- 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
)
```

لنحاول إضافة الأسطر التالية 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');
```

مثال2- تحديد قيد فريد بعد إنشاء الجدول

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

```
ALTER TABLE Department
ADD CONSTRAINT unq_dep unique (deptName)
```

مثال 3- لحذف القيد السابق:

```
ALTER TABLE Department
DROP CONSTRAINT unq_dep;
```

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 1970 نعرف القيد التالي:

```
CREATE TABLE Employee
(
        empSN int CONSTRAINT  Ttestt  primary key,
        fName varchar(50)  NULL,
        lName varchar(50) CONSTRAINT test_notNull NOT NULL,
        birthDate datetime,
        hireDate datetime NOT NULL,
        address varchar(50)  NULL,
        sex bit NOT NULL,
        salary money NOT NULL,
        managerSN int NULL,
        deptNo int NULL,
)
ALTER TABLE Employee
        ADD  CONSTRAINT CK_Emp_bDate CHECK  (birthDate  > '01/01/1970')
INSERT INTO employee (empsn,lname,birthdate,hiredate,salary,sex)
VALUES(2,'naser','11/11/1969','01/02/2001',30000,1)

ALTER TABLE Employee
ADD  CONSTRAINT CK_Emp_salary CHECK  (salary  < 50000)
INSERT INTO employee (empsn,lname,birthdate,hiredate,salary,sex)
VALUES (3,'salam','11/12/1999','01/02/2022',60000,1)
```

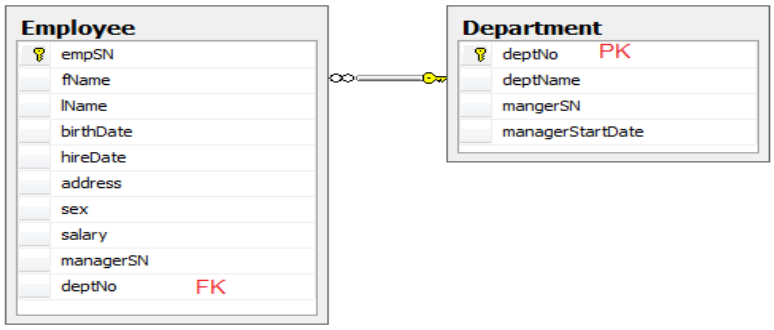
القيود الإفتراضية :Default Constraints :

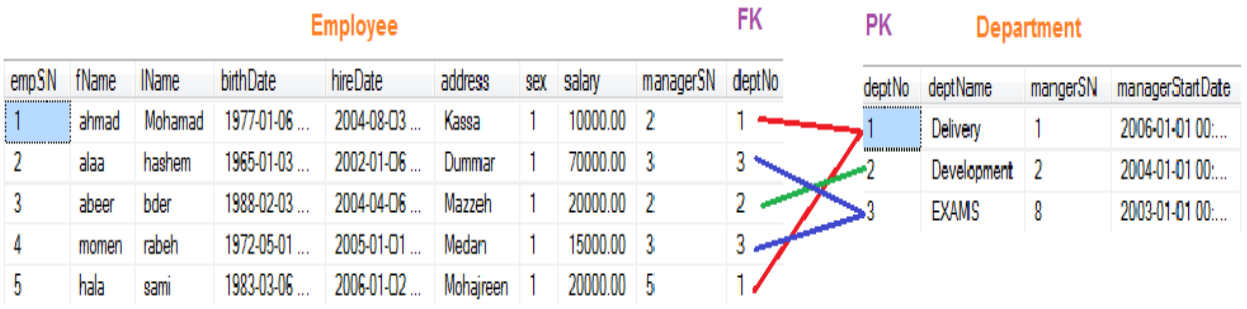
```
ALTER TABLE Department
        ADD  CONSTRAINT def_Department DEFAULT 5 for mangerSN
INSERT INTO Department (deptNo, deptName,  managerStartDate)
VALUES (7, 'SA', '1/1/2004')
```

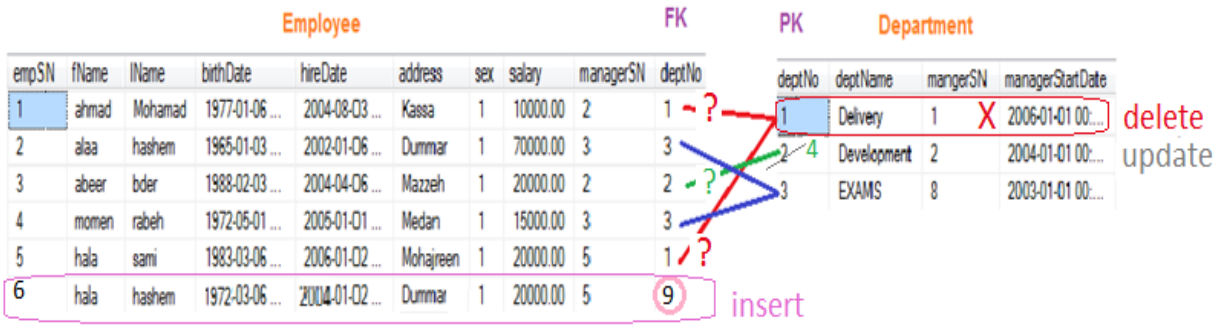
قيد المفتاح الثانوي Foreign Key Constraint

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.







مثال - لإنشاء 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 employee
ALTER TABLE Department
ADD CONSTRAINT pk_dep primary key (deptNo)
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)
    )
```

كما يمكن اضافة القيد بعد بناء الجدول

```
ALTER TABLE Employee
ADD CONSTRAINT FK_EMP_DEP FOREIGN KEY REFERENCES Department(deptNO)
    )
```

لإضافة سجلات في الجدولين:

```
ALTER TABLE Employee
ADD CONSTRAINT FK_EMP_DEP FOREIGN KEY REFERENCES Department(deptNO)
    )
--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

لحذف مفتاح ثانوي:

```
ALTER TABLE Employee
DROP CONSTRAINT FK_EMP_DEP
```

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

لمعرفة جميع القيود في قاعدة بيانات:

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

نهاية الجلسة