★ رقم التمرين : الأول

★ اسم التمرين : المناظير VIEWS

★ المناظير VIEWS

ابسط طريقة لفهم المناظير هي اعتبارها تعابير Select محفوظة تسمح للمستخدم بالعمل مع النتائج التي تعيدها وهي غالباً ما يستخدم كطريقة لتغليف الاستعلامات أو عمليات ربط المعقدة بين الجداول ولتطبيق العمليات التجميعية المعقدة وصياغة البيانات لعادة مرتبي أكثر. كما انها تعد من اهم ادوات الأمان لانها تساعد على منع المستخدمين ذوي الصلاحيات المحدودة من الوصول الى الجداول الأصلية في قواعد البيانات ومنحهم صلاحيات قراءة فقط او منحهم صلاحيات لرؤية العرض الحقول من الجداول فقط كما تعد اهم المناظير في اخفاء اسماء الحقول الحقيقية في الجداول الأساسية.

صيغة إنشاء منظار

CREATE VIEW view_name AS Select_Statement

نلاحظ اننا يجب ان نعطي للمنظار اسما أحدد الواصفة View_name ومن ثم نكتب الاستعلام المغلف بهذا المنظار بعد الكلمة المفتاحية as

مثلاً لتغليف الاستعلام الذي يعيد اسم وبلد كل زبون في الجدول Customers من القاعدة Northwind فاننا نكتب الأمر التالي.

```
USE northwind
Go
GREATE VIEW VW Customer Country
AS
Select Company Name . Country FROM Customers
```

نستطيع الآن الاستعلام عن الصلات الناتجة عن المنظار كما في حالة الجداول باستخدام (Select) مثلاً لتحديد قائمة الزبائن الذين يحوي اسم بلدهم على حرف u نستخدم الاستعلام التالي: ←

Select * From V W Customer Country

Where country like '% u %'

صيغة تعديل منظار

نستخدم التعليمة التالية

Alter View View_name AS

NEW_select_statement

مثلاً لتعديل منظار سابق ليشمل اسم مدينة الزبون نكتب التعليمة التالية

USE northwind

Go

Alter View V W Customer Country

AS

Select Company Name, County, City fRom Customer

صيغة حذف منظار

لحذف منظار نستخدم التعليمة التالية

DROP View view_name

وهنا سيحذف المنظار view_name من القاعدة الحالية

مثلاً لحذف المنظار VW Customer Country من قاعدة Northwind ستضع الأوامر التالية

USE northwind

Go

drop View VW Customer Country

الفائدة من استخدام المناظر

1. يمكن للأشخاص الذين يعملون على تطبيقات التعامل مع استعلامات بسيطة على المنظار الذي يعامل كجدول دون الاضطرار الى كتابة استعلامات المعقدة والمزيجة وهذا مفيد في حال الحاجة الى استعلامات بشكل متكرر

2. تفصيل بيانات حسب حاجة التطبيقات دون الحاجة الى خلق بيانات مكررة

3. تحديد قيود على الحقول المرئية.

★ رقم التمرين : الثاني - 2 -

★ اسم التمرين : الفهارس indexes

★ الغاية من التمرين : التعامل مع الفهارس

الفهارس indexes

الغرض من الفهارس : الغرض من الفهارس هو رفع أداء قاعدة البيانات أثناء عمليات الاستعلام والبحث

المبدأ : إنشاء آلية لتسريع عملية تحديد كل من مجموعة سجلات يعتمد هذه الآلية على محرك البيانات

وهي إنشاء جداول مهرسة لربط قيمة الحقل الذي تمت إضافة الفهرس إليه مع موقع السجل الخاص بهذه القيمة

ترفع الفهارس الأداء في حالة الاستعلام والبحث وليس التعديل لذلك يجب مراعاة مهرسة الحقول التي تتم عليها الكثير من عمليات البحث

ملاحظة

عادة ما يقوم محرك البيانات بإنشاء فهارس تلقائية للمفاتيح الرئيسية

1- الفهارس البسيطة

```
Create Index index_name
oN table_name
(
   Column_naamel
)
```

ويمكن فيها أن تتكرر القيم في عمود (أو أعمدة) الفهرس

2- الفهارس الفريدة

```
Creat unique index index_name
oN table_name
(
   Column_namel
)
```

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

لحذف فهرس نستخدم التعليمة التالية

```
Drop index table_name index name
```

نريد الآن انشاء فهرس يشمل اسم و كنية كل موظف في جدول الموظفين نستخدم لتعليمة التالية.

use nothwnind

Go

Create index indx-employee - full name

on Employees (First name , last name)

وذلك اثناء هذا الفهرس إلى تسريع عمليات الاستعلام التي تشمل شروطاً على اسم الموظف فقط أو على اسم الموظف وكنيته (هذا الفهرس غير مفيد للاستعلامات التي تشمل فقط كنية الموظف)

مثلاً قد يكون لفهرس السابق مفيداً لتسريع الاستعلام التالي:

selept* from Employees where

First - name like "a%"

and

last name lik "d%"

الذي يعيد قائمة بيانات الموظفين الذي يبدأ اسمهم في a و كنيتهم بالحرف d وهنا قد يقدم المحرك الفهرس لتسريع عملية تنفيذ هذا الاستعلام لكون جميع حقول الفترة موجودة ضمن هذا الفهرس كما يمكن للمحرك ان يستخدم هذا الفهرس لو كان هناك فترة على الاسم فقط وون الكنية

لحذف الفهرس السابق نستخدم التعليمة التالية

use northwind

Go

Drop index Employees idx - employee - Full name

رقم التمرين: التالي (3)

اسم التمرين: التوابع Sql Function

الغاية من التمرين: التعامل مع التوابع

التوابع في (Function in Sql)

التابع: هو عبارة عن تعبير رياضي يأخذ مجموعة من قيم الدخل التي نرمز لها بمعاملات وتعيد قيمة خرج وحيدة نرمز لها بقيمة التابع تعتمد قيمة التابع (أي الخرج) بمعاملات الدخل (أي الدخل)

كمال التابع الذي يحسب مجموع قيم عددية

تقسم توابع Sql حسب معيار Sql -99 إلى نوعين

- التوابع التجميعية: وهي التوابع التي تأخذ كمعاملات مجموعة من القيم وتعيد قيمة وحيدة مثل التابع الذي يحسب مجموع أعداد حقيقية مثال $f(x,y,z) = x+y+z$
- التوابع القياسية: وهي التوابع التي تأخذ معاملاً وحيداً وتعيد قيمة وحيدة مثل تابع القيمة المطلقة لعدد حقيقي مثال $f(x) = |x|$

أولا: التوابع التجميعية Aggregate Functins

تعريف التابع التجميعي

هو تابع يولد قيم مختصرة Summary. يقوم التابع التجميعي بمعالجة جميع القيم المختارة في عمود ما لتوليد ناتج وحيد.

تطبق التوابع التجميعية على الأسطر المختارة في عملية الانتقاء

المخطط العام للتوابع التجميعية AGG-FUNC([ALL DITINCT] expression)

ALL: تطبيق التابع التجميعي على جميع القيم

Ditinct: تطبيق التابع التجميعي على القيم المختلفة فقط (تجاهل التكرار)

expression: هو قيمة ثابتة أو اسم عمود أو تعبير حسابي أو محرفي ما

استخدام التوابع التجميعية في تعليمة الانتقاء (Select)

تابع المجموع Sum

يقوم بحساب مجموع القيم في تعبير ما يتضمن عموداً واحداً أو أكثر

```
Select Sum ([ALL | Ditinct] column-name)
from table-name;
```

ستخدم لخيار All عندما تريد الحصول على مجموعة قيم حقل معين بما فيها القيم المكررة.
يعتبر هذا الخيار هو الخيار التلقائي في حال عدم تحديد أي من الخيارين DISTINCT أو All.

ستخدم لخيار DISTINCT عندما تريد الحصول على مجموعة قيم حقل معين مع استبعاد أي تكرار للقيم.

```
Select (distinct order price) From orders.
```

O-Id	Order Data	Order price	Customer
1	2008/11/12	1000	ياسر
2	2008/10/23	1600	إحسان
3	2008/09/02	700	ياسر
4	2008/09/03	300	ياسر
5	2008/08/30	2000	محمد
6	2008/10/04	1000	احسان

تابع الوسطي AVG

يقوم بحساب القيمة الوسطية لتعبير حسابي يتضمن عمودا واحدا أو أكثر.

```
Select AVG ([All | DISTINCT] column-name)
From Table_name;
```

مثال احسب الوسطي لسعر بحيث عندما نريد زيادة سعر كل كتاب بمقدار 10$

```
Select AVG (price+10) as 'AVG price' From TITLES
```

تابع العد Count

يستخدم تابع Count عند لبيانات موجودة مثل جدول من أجل حقل معين أو يقوم بحساب عدد الأسطر الموجودة

```
Select Count ([*|ALL| Distinct] column-name)
From table-name;
```

ستخدم لخيار * عندما تريد الحصول على عدد لبيانات الموجودة مثل جدول بالنسبة لحقل معين سواء لبيانات ذات القيمة Null ولكن بدون كتابة اسم الحقل

مثال احسب عدد المدن التي يوجد فيها كتاب (مدن فقط)

```
select Count (Distinct city) as 'city Count' from AuTHors
```

ستخدم لخيار All عندما نريد الحصول على مجموعة قيم حقل معين بما فيها القيم المكررة.
يعتبر هذا الخيار هو الخيار التلقائي في حال عدم تحديد أي من الخيارين All أو DISTINCT

ستخدم لخيار DISTINCT عندما نريد الحصول على مجموعة قيم حقل معين مع استبعاد أي تكرار في القيم.

```
Select (distinct order price) From orders.
```

O_Id	Order Data	Order price	Customer
1	2008/11/12	1000	ياسر
2	2008/10/23	1600	إحسان
3	2008/09/02	700	ياسر
4	2008/09/03	300	ياسر
5	2008/08/30	2000	محمد
6	2008/10/04	1000	احسان

تابع الوسطي AVG

يقوم بحساب القيمة الوسطية لتعبير حسابي يتضمن عموداً واحداً أو أكثر

```
Select AVG ([All | DISTINCT] column-name)
From Table_name;
```

مثال احسب وسطي اسعار الكتب بعدما نريد زيادة سعر كل كتاب بمقدار 10$

```
Select AVG (PRICE+10) as 'AVG PRICE' From TITLES
```

تابع العد Count

يستخدم تابع Count عد البيانات الموجودة في الجدول من أجل حقل معين أو يقوم بعد كل الأسطر الموجودة

```
Select Count ([*|ALL| Distinct] column_name)
From table-name;
```

ستخدم لخيار * عندما نريد الحصول على عدد البيانات الموجودة في الجدول بالنسبة لحقل معين بما فيها البيانات ذات القيمة Null ولكن بدون كتابة اسم الحقل

مثال احسب عدد المدن التي يوجد فيها كتاب (مدن لبنان)

```
select Count (Distinct city) as 'city Count' from AUTHORS
```

مثال أوجد عدد الكتب الموجودة

Select count(*) As 'titles count' from titles

* لا يمكن استخدام اكثر من تابع تجميعي من نفس العملية لواحدة

تابع القيمة العظمى MAX : يعيد قيمة تعظمى من تعبير وذلك بعد اهمال القيم Null

Select max(column_name) from table_name

تجدر الاشارة الى عدم وجود أي تأثير للخيارين All و Distinct على توابع MAX و MIN

رغم انه بالإمكان استخدامها مع القيمة العظمى أو القيمة الصغرى لكل حقل بقى نفسها

حتى ولو كان هناك تكرار من قيم الحقل وحتى ولو كان هناك قيم غير محددة (أي خالي null)

مثال أوجد أعلى قيمة مبيعات للكتب

Select Max(ytd_sales) 'Best Sales' FROm titles

تابع القيمة الدنيا min : يعيد ادنى قيمة من تعبير وذلك بعد اهمال القيم Null

Select min(column_name) from table_name;

مثال Select min(ytd_sales) 'Lowest Sales' from titles

★ رقم التمرين: الرابع -٦-

★ اسم التمرين: Sql Function

★ الغاية من التمرين:

ثانيا: التوابع الداخلية Scalar Functions

1. التوابع الرياضية Mathematical Function

a. توابع سلاسل الحارف String Function

b. توابع التاريخ والزمن Date and Time Function

c. توابع الحماية Security Function

d. توابع النظام System Function

التوابع الرياضية Mathematical Function

وهي توابع تقوم بإجراء عمليات على متحولات وقيم عددية.

CeiLiNG: يعيد العدد الصحيح الأكبر مباشرة أو يساوي قيمة تعبير وفق عالي

نمط قيمة الإعادة من التابع مطابق لنمط التعبير العالي.

Select CeilinG(2) Select Ceiling (1.01)

Select Ceiling(1.03) Select ceiling(1.06)

FlooR: يعيد العدد الصحيح الأصغر أو يساوي قيمة التعبير الداخل.

Select FlooR(1) Select Floor(1.01)

Select FlooR(1.5) Select Floor(1.06)

Exp: يعيد قيمة التابع لتبرير e لعاقفة العدد الحقيقي

```
Declare @Var Flat
SET @Var = 4
Select 'The Exp is :' + Convert ( Varchar , Exp(@Var))
```

Log: يعيد قيمة التابع اللغاريتمي لتبرير التعبير لعاقفة

```
Declare @ Var Float
SET @ var = 5.175643
Select 'the Log is :' + Convert (Varchar , Log (@ Var))
```

POWER : يعيد ناتج رفع المتغول الأول الى القوة التي يمثلها المتغول الثاني

```
Declare @ VAlue int, @ Counter int
    set@ Value = 2
    Set @ Counter = 0
    while@ Counter<5
  Begin
Select power(@ value, @ Counter
Set @ Counter = @ Counter +1
  End
```

RAND : يعيد عدد عشوائي بين 0 و 1

```
Declare @ Counter smallint
 set @ Counter = 1
while @ Counter<5
  BEGIN
Select RAND (@counter) RANDOM_number
--- Select RAND (@ Counter * 100000) * 100 as RANDOM_NumB
 Set @ Counter = @ Counter +1
  End
```

SQUARE : يعيد القيمة التربيعية لتعبير ما

SQRT : يعيد قيمة الجذر التربيعي لتعبير ما

* توابع سلاسل المحارف String Functions

وهي توابع تطبق على سلاسل المحارف وتعيد اما سلاسل محارف أو قيم عددية

التابع SUBSTRING

يعيد التابع SUBSTR جزء من سلسلة محارف ابتداءً من موقع محدد من تلك السلسلة وطول عدد محدد من المحارف علماً اننا كتابة الاستعلام الذي يعيد سلسلة محارف طول 5 محارف اعتباراً من قيم الحقل Title متبدئاً من المحارف رقم 1 نكتب الصيغة التالية

مصر المحارف

```
Select SUBSTRING (First_name, 1, 5) from Employees
```

أكمل

LEN - DATALENGTH : يعيد عدد الحارف في سلسلة حارف مايدون الفراغات في نهاية السلسلة

DATALENGTH : يعطي الطول كاملاً مع الفراغات.

مثال

```
DECLAER @STR VARChar(50)
Set @ STR = 'Hello, Sql' ;
Select @ STR, LEN (@ STR) as "LeN"
Select LEN (name) from sysdb
Select DATALENGTE ('ahmad'), LEN ('ahmad')
```

LOWER : يعيد صيغة سلسلة حارف بعد تحويل الحارف الكبيرة الى صغيرة

مثال

```
DECLAER @STR VARChar (20)
Set @ STR = "Hello, Sql"
Select @ STR, Lower (@ STR) as "LEN"
```

Date and Time Functions توابع الزمن والتاريخ

التابع GETDATE : الذي يعيد التاريخ الحالي متضمن السنة والشهر واليوم والساعة والدقيقة والثانية فالاظهار التاريخ الحالي يمكننا استخدام الصيغة التالية GETDATE();

```
Select GetdAte()
```

يعطي الزمن والتاريخ بدقة 3 اجزاء من الثانية

```
SySDATETiME
DateAdd (date part, number, date)
```

يضيف عدد معين من اي تاريخ محدد

Example:

```
Select DateAdd (day, 6, '2006-07-31');
Select DateAdd (year, 6, '2006-07-31');
```

★ رقم التمرين : الخامس

★ اسم التمرين : الإجراءات المخزنة Stored procedures

★ الإجراءات :

تمتلك الإجراءات Sql - Server متغيرات وقيمة مرجعة ويمكن ان تمرر لمتغيرات اما بالقيمة أو بالمرجع والقيمة المرجعة هي . احتلالاً . التي تشير الى حالة الخطأ الذي ينتج عند تنفيذ الإجراءات . فتعطي عندما تكون القيمة المرجعية صفراً فهذا يعني ان الاجرائية نفذت بلا اخطاء . وتشير القيمة المغايرة للصفر الى جزء الاجرائية الذي فشل عند تنفيذ الإجرائية .

تنشأ اجرائية Sql - Server باستخدام الأمر Create Procedure ويعرفها متغيرات الدخل ومتغيرات الخرج وتقوم باعادة القيمة المعادة عند تنفيذ هذه الإجرائية .

مثال

```
use northwind
go
Create procedure passChar (@C nvarchar (100))
as
begin
return len (@C)
and
```

لنشرح هذه الإجرائية سطراً سطراً طبعاً أولاً طريقة تعريف قاعدة البيانات التي سيتم تخزين الإجرائية فيها طبعاً هنا لدينا الكلمة المفتاحية Create procedure لتعريف اجرائية بعد ذلك اسم الاجرائية ما بين القوسين هو متغير الدخل دائماً ومتغيرات الدخل دائماً نضع اشارة @ قبلها في حالتنا اسم متغير الدخل هو C وتم تعريف نوعه Sql Server طبعاً جميع الأنواع المعرفة يجب ان تكون من انماط بيانات nVarchar (100) . as في السطر الرابع ستتقدم بعد نهاية تعريف متغيرات الدخل .

Begin : بداية كتلة تعليمات T-Sql ويمكننا استخدام عدة كتل متداخلة داخل بعضها . وكل begin يجب ان تكون لها End .

return : تعليمة اعادة القيمة التي يعيدها التابع .

هذه الإجرائية موجودة في قاعدة البيانات northwind يتم تخزينها ضمن فئة programmability ثم ضمن فئة stored procedure بالشكل

orabico(Sql-Server) ← data base ← northwind ←

programmability ← passchar dbo.

لاستدعاء هذه الإجرائية مباشرة يمكننا استخدام التعليمات التالية

```
1. declare @i tinyint
2. exec @i = passchar 'hello'
3. print @i
```

طبعاً في السطر الأول تم التعريف متحول اسمه i من نوع tinyint

في السطر الثاني تم استدعاء الإجرائية بتعليمة exec وتم إسناد القيمة العائدة الى المتحول i

في السطر الثالث أمر طباعة المتحول i على الخرج messages كما بالشكل

حيث يعطي الرقم صفر

طبعاً كان من الأفضل ان تعالج الإجرائية حالة null حتى تعمل بشكل أفضل

طبعاً هناك مجموعة من الأمور يجب مراعاتها عند انشاء procedure وهي:

1. يمكن ان تحتوي مجموعة التعليمات Batch أوامر أخرى غير تعليمة Create procedure مثلاً مجموعة التعليمات التالية لا تنفذ

```
use northwind
create procedure Temp
```

2. يجب ان يملك صلاحية مالك قاعدة بيانات dbowner كل من يريد ان ينشأ إجرائية مخزنة

3. يجب تحديد اسم الإجرائية المخزنة. الإجرائية المخزنة هي عبارة عن قاعدة بيانات التي يتبع اسمها قواعد تسمية المتحولات

4. يمكن انشاء إجرائية مخزنة في قاعدة بيانات حالية.

عند انشاء اجرائية يجب تحديد النقاط التالية.

1. اسم الاجرائية procedure name
2. معاملات الدخل input parameters
3. معاملات الخرج output parameters
4. القيمة المعادة الى الاجرائية المستدعية للاجرائية الحالية والتي تدل على فشل او نجاح الاجرائية

كما يجب تحديد حالة الخطأ المراقبة للقيمة المعادة

تتخاطب الاجرائية مع البرامج التي تستدعيها عبر مجموعة وسائط parameters حيث يكون البرنامج مهمتها من تمرير قيم الى الاجرائية input parameters كما تمكنه من الحصول على قيم من الاجرائية output parameters

- يجب ان تكون وسائط الاجرائية وحيدة (أي انه لا يمكن ان تتشارك اجرائية تحتوي على وسيطين بنفس الاسم) يجب ان تبدأ كل وسائط الاجرائية بالرمز @ (كل وسائط الاجرائية تتبع هذه التسمية هذه تسمية اصطلاحية ويتبع الحرف تحتوي على احرف أو ارقام أو ...).

★ اسم التمرين: ربط الجداول Joining Tables

★ رقم التمرين: 1 - 2 - 3 - 4 - 5 - 6 -

★ لدينا قاعدة بيانات مكونة من الجداول الآتية.

★

Department

DeptID	Dept Name
Actg	Accountiong
Admn	Administration
Fin	Finauce
MKTg	Marketing

Employee

Emp Id	First Name	last name	Sex	Dept Id
1	عمر	السهلي	Male	Actg
2	طارق	عواد	Male	MKTg
3	سامية	سعاد	Female	MKTg
4	صناء	آنا	Female	NULL

Children

Child Id	Child name	Sex	Birth data	Emp ID
1	سحر	Female	11/02/1990	1
2	طوني	Male	19/02/1990	2
3	رهام	Female	30/08/1992	1
4	لبيس	Female	19/09/1980	3
5	وسيم	Male	04/03/1999	4
6	حسن	Male	02/07/2000	1
7	محمد	Male	29/05/1991	3
8	عبير	Famale	09/01/2001	2

had S A B B A G H

/ /

أ. المفتاح الرئيسي primary key :

حقل أو أكثر يعرف فيه قيمة أو قيم فريدة لكل سجل في الجدول بالإضافة سيتم إنشاء حقل EmpId مفتاحاً رئيسياً لجدول Employeee وحقل DeptId مفتاحاً رئيسياً لجدول Department وحقل ChildId مفتاحاً رئيسياً لجدول Children.

ب. المفتاح الخارجي Foreign Key

حقل أو أكثر من حقول الجداول يشير إلى حقل أو حقول المفتاح الرئيسي في جدول آخر. في الجداول السابقة سيتم إنشاء الحقل DeptId الموجود في الجدول Employeee مفتاحاً خارجياً للجدول Department.

ج. الربط الداخلي Inner Join

يعرف الربط الداخلي على انه الربط البسيط بين سجلات جداول أول وسجلات جدول ثاني اعتماداً على مساواة بين قيمة حقل في كل من الجدول الأول (عادة المفتاح الخارجي) لصيغة

Select Table1 - Field , Table2 - Field
From Table1 Inner Join Table2 ON Table1.field = Table2.field

امثلة

1- اسماء الموظفين الذين لديهم ولد اسمه وسيم

Select First name , Last name
From Employeee Inner Join Children on Employeee.EmpID = Children.EmpI
WHERE Child name = 'وسيم'

2- اسماء الأبناء الذكور للموظفين يعاملون في قسم التسويق.

Select Child name
From Department Inner Join Employee ON Department.DeptId
= Employeee.DeptID
Inner Join Children ON Employeee.EmpId = Children.EmpId
where Children.Sex = 'Male' And Deptname = 'Marketing'

ذكر

3. أسماء وأعداد أولاد الموظفين في جميع الأقسام

```
Select First name, Last name, count (Child Id) as
                                    Children Count
From Employee Inner Join Children on Employee.EmpId = Children.EmpId
    GRoup By First name, last name.
```

★ اسم لتمرين : قواعد التحقق من القيم

★ رقم لتمرين : السابع

★ مقدمة

تمثل القيود منطق عمل Business logic الذي يقوم بخدم قواعد البيانات بضمان التحقق على حقول واعمدة جداول قواعد البيانات وتشمل هذه القيود مجال القيم التي يمكن ادخالها في عمود اضافة الى الكامل الاعمدة في البيانات

عند اضافة أو حذف أو تعديل أي بيانات في جدول ما من جداول قاعدة بيانات فان نخدم قواعد البيانات يراقي جميع القيود للتعرف على هذه الجداول قبل تنفيذ العملية ولا ينفذ هذه العملية اذا كانت تؤدي الى أي خلل في القيود المرفوضة

مثال

اذا كان مفتاح فريد Unique على جدول ما فلا يمكنك اضافة أو تعديل سطر ما بحيث يصبح لديك تكرارات في قيمة المفتاح الفريد

مثال

اذا كان لديك قيد على مجال القيم لعمود ما فان اضافة أو تعديل قيمة هذا العمود لسطر ما خارج هذا المجال سوف تؤدي الى توليد خطأ من قبل مخدم قواعد البيانات .

- تراقي مخدمات قواعد البيانات التكاملات المرجعية للتعرف بين الجداول فلا يمكنك

مثلاً اضافة سطر في الجدول المثل للطرف كثيراً لم يكن هناك سطر موافق في الجدول المثل للطرف واحد وبالمثل فانه لا يمكنك حذف سطر من الطرف واحد عالم تكن جميع الأسطر الموافقة له في الطرف كثير قد حذفت أو انه لا يوجد أي سطر موافق في الطرف كثير (يوجد استثناءات لهذه القاعدة؟)

1- تحديد المفتاح الأولي عند انشاء الجدول :

لانشاء الجدول Department نكتب التعليمة التالية

تعريف المفتاح الأولي : يضمن قيد المفتاح الأولي عدم تكرار القيم في مجموعة الأعمدة بحيث لا يمكن لها يمنع ادخال القيمة Null في أي من اعمدته وهو يستخدم لضمان الوحدانية .

```
Create table [Department]
[dept No] [int] Not null primary key [dept name]
[Varchar](30) not null
```

* لاضافة قيد بعد انشاء الجدول نعدل تعريف الجدول كما يلي

```
Alter Table [Department]
Add Constraint [pk_Department] primary key
(
[deptNo]
)
```

القيود لتعريف unique constraints

يضمن القيد لتعريف عدم تكرار القيم في الأعمدة المكونة له بين الصفوف 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 في عدة مفاتيح فريدة

مثال

من 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 انشاء الجدول

```
Drop table (Department)
go
Cerate table (Department)
[deptNo] [int] notnull
[deptname] [varchar] (50) not null UNIQUE
[mangerSN] [int] not null,
[mangerstartDate] [datetime] null
```

تحديد قيد فريد بعد انشاء الجدول

```
Alter table [Department]
Add constraint [uk_deptname] unique
(
  (deptname)
)
```

Foreign Key Constraint يقيد المفتاح الثانوي

يعمل المفتاح الثانوي بالتزامن مع مفتاح أولي أو مفتاح فريد unique يمنعك إنشاء مفتاح ثانوي في جدول ما أذا القيم المكن إدخالها في أعمدة هذا المفتاح مطابقة لبيانات المفتاح الأولي أو الفريد الموافق.

ستمنع هذه المقيدة بالحد من تكرار البيانات في قاعدة البيانات فمثلاً نكتفي بإدخال بيانات الأقسام مرة واحدة ثم نربط جدول الموظفين بجدول الأقسام لتحديد القسم الذي يتبع له كل موظف عن طريق تحديد رقم القسم مع كل موظف. كما ان المفتاح الثانوي يضمن تكامل المعطيات حيث تتحقق محتوى البيانات من كون الرقم المدخل في حقل رقم القسم موجود فعلاً في جدول الأقسام تجدر الإشارة إلى انه لا يمكننا بعد تعريف المفتاح الخارجي ان نقوم بحذف المفتاح الرئيسي (أو الفريد) الموافق له مالم نقم أولاً بحذف المفتاح الخارجي.

مثال

1- إنشاء مفتاح ثانوي بعد انشاء الجدول: لإنشاء جدول الموظفين بحيث سيربط الحقل deptNo إلى الحقل الذي يحمل نفس الاسم من الجدول Department نتبع التعليمة التالية:

```
drop tabel [Department]
  go
Create table [Department]
(
[dept No] (int) not null
[dept name] (nvarchar) (50) not null
[mangerS N] [int] not null,
[mangerstart Date] (datetime) null
)
  go
Alter TABle [Department]
Add Constaraint (pk_Department) primary key
(
  deptno
)
  go
Create table [Employee] ( [emp SN] [int] not null,
[fname] [varchar] (50), [Lname] [varchar] (50) not null,
[birth Date] [datetime] null, [hire Date] [datetime] not null
```

[address] [~~datetime~~] not null ،

[Sex] [bit] not null ، [Salay] [money] not null ،

[mangerSN] [int] null ،

[deptNo] [int] null

FreiGN ~~Fra~~Key Reeerences Department (dept no)

قيد لتحقق Check Constraints :

يعرض قيد التحقق شروطاً على القيمة Value التي يمكن ان توضع في عمود أو مجموعة اعمدة

أو يفرض قيداً على تنسيق Format القيمة التي توضع في عمود أو مجموعة اعمدة ؛

مثلاً في جدول الموظفين

لكل موظف تاريخ ميلاد birthDate وتاريخ توظيف hireDate يفيد بخدم قواعد البيانات

أي تاريخ للعودين ودون أي قيود . لفرض قيد كون تاريخ الميلاد

تعرف القيد التالي

```
Alter Table [Employee]
  Add constaint [ck_Employee_birthdate]
Check ((date part (year, [birthdate) > (1970)))
```