النسخ الاحتياطي والاستعادة في SQL Server

الرابط التالي مطلوب بالاضافة للمحاضرة

https://www.systoolsgroup.com/updates/backup-and-restore-database-in-sql-server/?srsltid=AfmBOopb2Ap3-IiFK0LknnvwFiMeCcC4Lk3eQwi8UTyPFLHNaOz8v8Y-

مسائل أخرى في SQL Server

هذه بعض النقاط المهمة التي لم تتعرض لها السلسلة السابقة.

ربط وفصل (Attach/Detach) ملف قاعدة البيانات SQL Server:

تتجسد قاعدة البيانات SQL Server في نوعين من الملفات أساسا:

- **Primary Data File:** هي ملفات بلاحقة MDF، وهي الملفات التي تخزِن البيانات.
- **Transaction Log File:** هو ملف بلاحقة LDF، ويخزن فيه كل الاستعلامات والتعليمات المنفذة على قاعدة البيانات.
- **Secondary Data Files:** عند إنشاء قاعدة البيانات يمكن توزيعها على عدة أقسام ملفات (Filegroups)، وتخزن في القرص بلاحقة **NDF**.

للمزيد حول ملفات أنماط الملفات وطريقة تخزين البيانات انظر الرابط:

http://msdn.microsoft.com/en-us/library/aa174545(SQL.80).aspx

ربط (Attach) ملف قاعدة البيانات:

عند إنشاء قاعدة البيانات باستخدام Microsoft SQL Server Management Studio يقوم هذا الأخير (تلقائيا) بربط ملف قاعدة البيانات (MDF File) إلى الخادم، وبالتالي يمكنه التعرف إلى قاعدة البيانات وتنفيذ الاستعلامات عنها إضافة إلى عمليات الحذف، التعديل، وغير ذلك...

إذا قمت بنسخ ملف قاعدة البيانات (MDF File) إلى جهاز آخر، فإن الخادم في هذا الجهاز لن يتعرف على قاعدة البيانات، وسيرسل رسالة خطأ إذا تلقى أي استعلام عنها.

ليتعرف الخادم على ملف قاعدة البيانات يجب ربطها (Attach) بإرسال استعلام مثل الآتي:

```
CREATE DATABASE [MyDataBase] ON
( FILENAME = N'C:\path\MyDataBase.mdf' ),
( FILENAME = N'C:\path\MyDataBase_log.ldf' )
```

```
FOR ATTACH
```

في البداية عبارة **CREATE DATABASE** يتبعها الاسم المستعار الذي سيستعمل لتسجيل (ربط) قاعدة البيانات إلى الخادم، ويمكن لهذا الاسم أن يختلف عن اسم الملف.

بعد **FILENAME** يأتي المسار الكامل لاسم ملف قاعدة البيانات (MDF File)

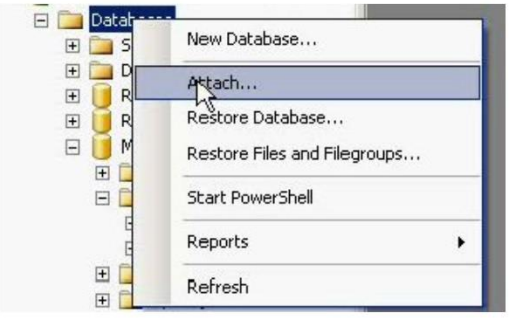
ثم **FILENAME** الثانية (اختيارية) لربط ملف أي ملف آخر لقاعدة البيانات (في المثال ملف LDF وهو ملف اختياري).

FOR ATTACH تعني إنشاء قاعدة البيانات على الخادم من خلال ربط ملف قاعدة بيانات موجود على القرص الصلب.

بطريقة أخرى يمكن استخدام الإجراء المخزن sp_attach_db كما يلي:

```
sp_attach_db    'MyDataBase',
                'C:\path\MyDataBase.mdf',
                'C:\path\MyDataBase.ldf'
```

للقيام بذلك باستخدام MS SQL Server Management Studio حدد البند Databases ثم انقر Attach، قم بإضافة ملفات قواعد البيانات التي تريد ربطها بالزر ...Add، ثم OK.



ملاحظة:

- قبل إرسال مثل هذه الاستعلام يجب التأكد أن قاعدة البيانات الحالية تختلف عن التي سنقوم بربطها أو فصلها، كذلك الأمر فيما يخص عمليات الحذف والتعديل.

لنتأكد أن قاعدة البيانات الحالية غير التي سنجري عمليات عليها، نقوم بتغيير قاعدة البيانات الحالية إلى أي قاعدة بيانات أخرى، مثل master قاعدة بيانات النظام في SQL Server.

لأجل ذلك نضيف قبل الاستعلامات السابقة عبارة:

```
USE [master]
GO
```

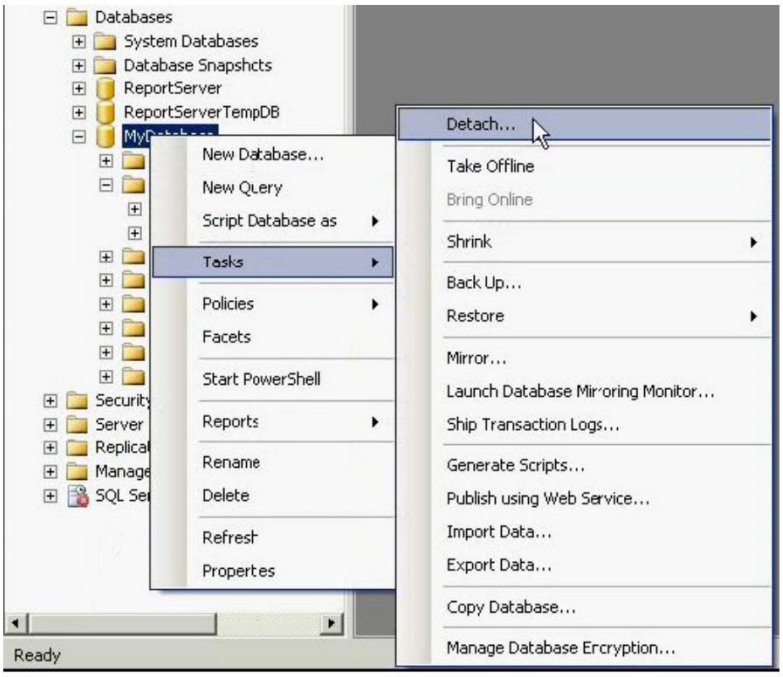
وبذلك نكون متأكدين أن قاعدة البيانات الحالية بعد هذين السطرين هي master.

فصل (Dettach) ملف قاعدة البيانات:

إذا أردت نقل ملف قاعدة بيانات من جهاز لآخر، عليك أولا فصل ملف قاعدة البيانات عن الخادم، ويتم ذلك باستخدام الإجراء المخزن sp_detach_db كما يلي:

```
USE [master]
GO
EXEC sp_detach_db @dbname = N'MyDataBase'
GO
```

باستخدام MS SQL Server Management Studio حدد قاعدة البيانات من متصفح الكائنات ثم Tasks -> Detach، ثم انقر OK للموافقة.



ملاحظة:

- بعض العمليات في SQL Server يمكن تنفيذها بعدة طرق باستخدام SQL، حيث تضيف Microsoft في الإصدارات الحديثة أوامر جديدة، وتبقي على الإجراءات التي كانت تستخدم في الإصدارات السابقة (لأغراض التوافقية Backward Compatibility)، تمهيدا لحذف هذه الإجراءات لاحقا.

مثال:

لتغيير اسم قاعدة البيانات في SQL Server حسب الإصدارات السابقة يستخدم الإجراء المخزن sp_renamedb كما يلي:

```
USE [master]
GO
EXEC sp_renamedb 'OldDBName', 'NewDBName'
GO
```

بقي ذلك ساري المفعول لأغراض التوافق مع الإصدارات الجديدة. والآن في الإصدارات الحديثة يستحسن استعمال ما يلي:

```
USE [master]
GO
/* MyDBOldName تغيير اسم قاعدة البيانات */
ALTER DATABASE MyDBOldName MODIFY NAME = MyDBNewName
GO
```

للمزيد حول ذلك طالع ملفات المساعدة المرفقة مع SQL Server.

حفظ قاعدة البيانات Backing up SQL Server:

يوفر SQL Server عدة خيارات لحفظ واسترجاع قاعدة البيانات. يمكن حفظ نسخة من قاعدة البيانات بثلاث طرق:

- **Full Backup:** حفظ نسخة من قاعدة البيانات بكاملها دون ملف الأحداث (Log file)
- **Differential Backup:** لحفظ نسخة من التغيرات الحاصلة منذ عملية الحفظ الأخيرة
- **Transaction Log:** لحفظ نسخة من ملف الأحداث (Log file)

يوفر SQL Server ثلاث أوضاع مختلفة لحفظ نسخة قاعدة البيانات:

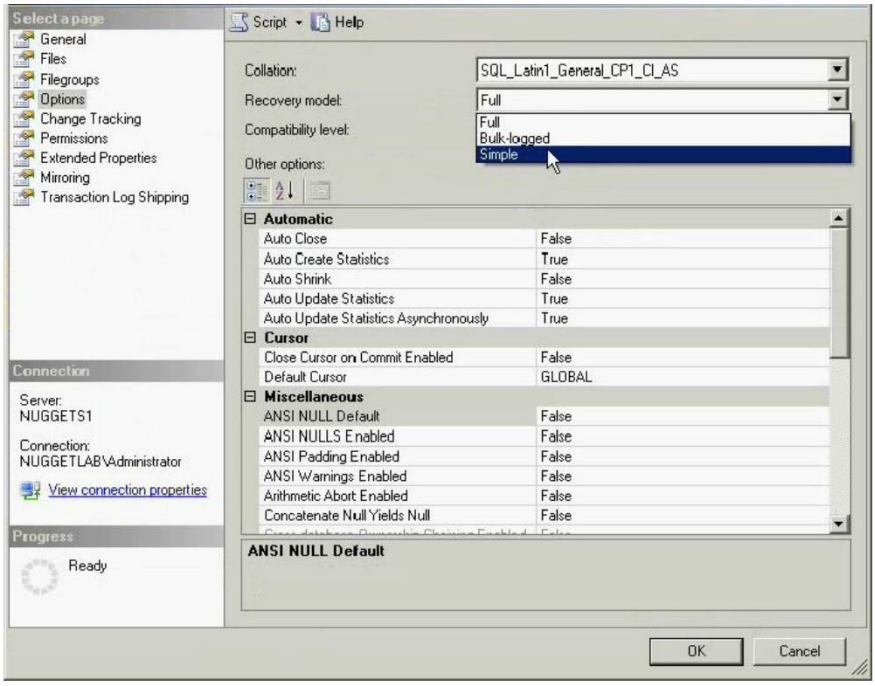
- **Full:** سيستخدم هذا الوضع ملف الأحداث (Transaction Log) لاستعادة قاعدة البيانات باسترجاع الاستعلامات المنفذة، وبذلك يمكنك العودة إلى أي وضع سابق.
- **Bulk-logged:** سيتم في هذا الوضع استخدام ملف الأحداث لاسترجاع التغيرات الحاصلة على قاعدة البيانات من خلال استرجاع أهم الاستعلامات المنفذة.
- **Simple:** في هذا الوضع لا حاجة لملف الأحداث، ويتم استرجاع نسخة من قاعدة البيانات بكاملها، وأي تغيير يحصل بعد آخر عملية حفظ لن يتم استرجاعه.

للمزيد من التوضيحات وتفاصيل أخرى حول حفظ نسخة احتياطية، راجع الرابط:

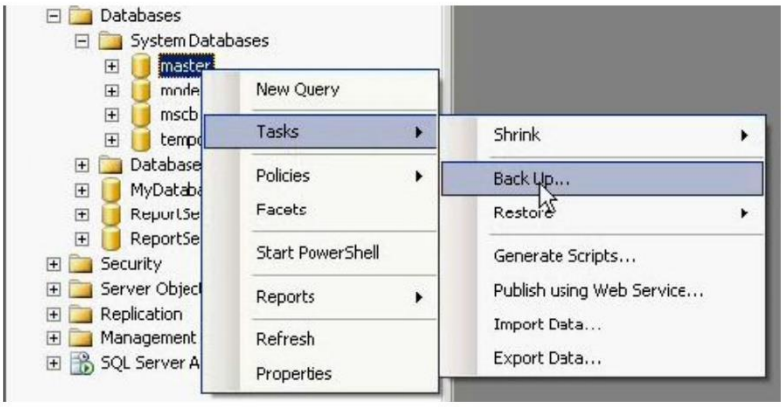
http://msdn.microsoft.com/en-us/library/ms187510.aspx

نأتي إلى SQL Server Management Studio لاختيار وضع الاسترجاع، انقر بالزر الأيمن على قاعدة البيانات واختر Properties، حدد التبويب Options لتجد طريقة الاسترجاع من قائمة الاختيارات Recovery Model (Simple افتراضيا):

401

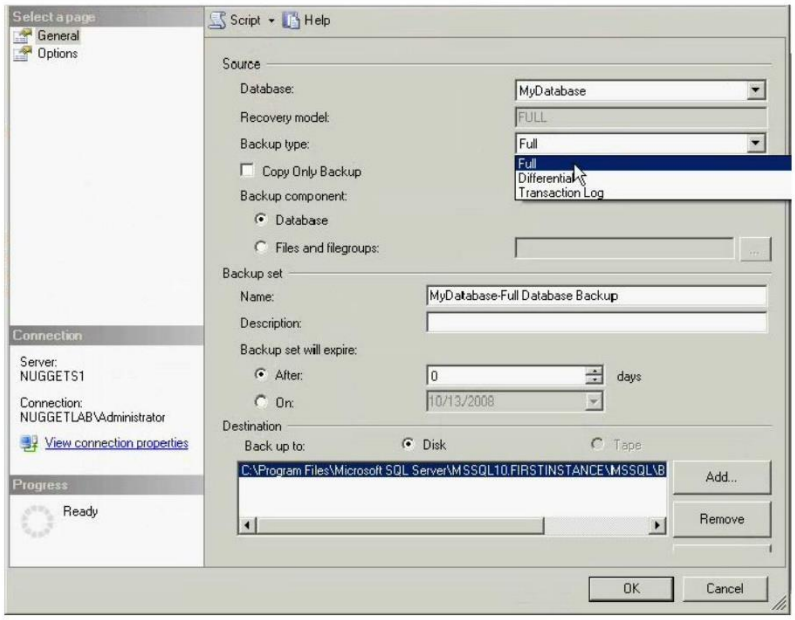


لحفظ نسخة من قاعدة البيانات انقر بالزر الأيمن على قاعدة بيانات أخرى[1] (ولتكن قاعدة البيانات النظامية master) ثم اختر Tasks -> Back Up...



حدد من قائمة الاختيارات Database قاعدة البيانات التي تود حفظ نسخة منها، ومن Backup type اختر نوع الحفظ (Full كخيار افتراضي):

[1] لا يمكنك حفظ نسخ من قاعدة البيانات المستعملة حاليا لأنها قيد الاستعمال، لذلك قم بالاتصال بقاعدة بيانات أخرى.



هناك أيضا خيارات إضافية أخرى، منها حفظ أحد الملفات أو قسم من الملفات (Filegroups)، أيضا تحديد تاريخ انقضاء النسخة المحفوظة لمنع استرجاع نسخ قديمة، وغير ذلك من الخيارات المتقدمة في التبويب Options.

انقر ...Add لإضافة مجلد للحفظ مع اختيار اسم الملف بأي لاحقة شئت (*.bak افتراضيا)، ثم انقر OK للموافقة.

باستخدام SQL يمكنك الحصول على نفس النتيجة بإرسال استعلام كالآتي:

```
BACKUP DATABASE [MyDatabase]
TO  DISK = N' C:\Program Files\Microsoft SQL
Server\MSSQL10.SQLEXPRESS\MSSQL\Backup\BackupMyDatabase.bak', SKIP,
NOREWIND, NOUNLOAD, COMPRESSION,  STATS = 10
GO
```

استرجاع قاعدة البيانات Restoring SQL Server:

تتم عملية استرجاع البيانات في SQL Server من ثلاثة طرق:

- **Database Restore:** وهو الخيار الافتراضي الذي يتم فيه استرجاع نسخة قاعدة البيانات بأكملها.
- **File Restore:** بهذه الطريقة يمكنك استرجاع أحد ملفات قاعدة البيانات دون الملفات الأخرى.
- **Page Restore:** في هذه الطريقة يتوجب اختيار إحدى طريقتين الحفظ Bulk أو Full فقط، ومن خلالها يمكنك استعادة جزء محدد من قاعدة البيانات.

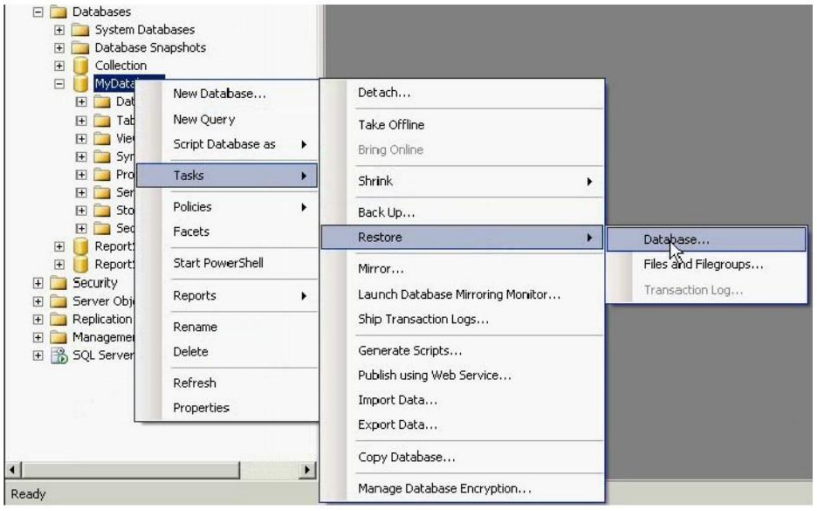
لاسترجاع نسخة من قاعدة البيانات يجب قطع الاتصال بها، والاتصال بقاعدة بيانات أخرى (master مثلا) لتنفيذ عملية الاسترجاع[1].

يمكن عند حفظ نسخة من قاعدة البيانات استخدام كل طرق الحفظ السابقة (Full، Differential، Log) وتخزين الناتج في ملف وحيد، وحينها يمكن أن تتم عملية استرجاع نسخة من قاعدة بيانات SQL Server وفق الترتيب الآتي:

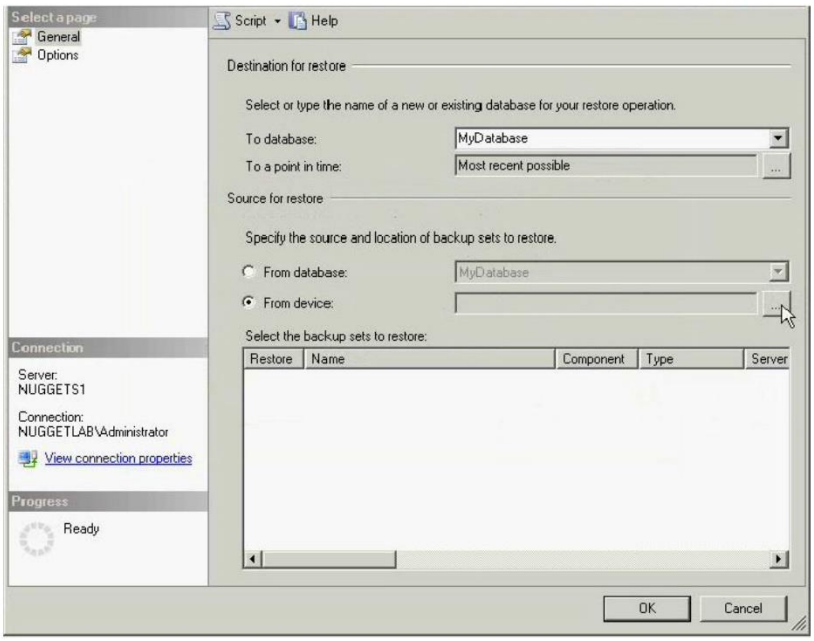
- **Full Backup:** استرجاع نسخة من قاعدة البيانات بكاملها، وهنا يمكن تحديد خيار NORECOVERY لنخبر SQL Server ألا يقوم بالاسترجاع فورا، لأننا نود استرجاع بيانات أخرى من:
- **Differential Backup:** الذي يحتوي على بيانات (أو عمليات) تم حفظها بعد حفظ نسخة كاملة من قاعدة البيانات، ويمكن هنا أيضا استعمال الخيار NORECOVERY لاسترجاع:
- **Log Backup:** الذي يخزن الأحداث التي جرت على قاعدة البيانات، وبعده تبدأ فعليا عملية الاسترجاع.

[1] لا حاجة لذلك عند استخدام بعض نسخ SQL Server مثل Entreprise Edition.

لاسترجاع نسخة من قاعدة البيانات باستخدام SQL Server Management Studio قم بتحديدها من متصفح الكائنات ثم اختر Tasks -> Resotre -> Database...

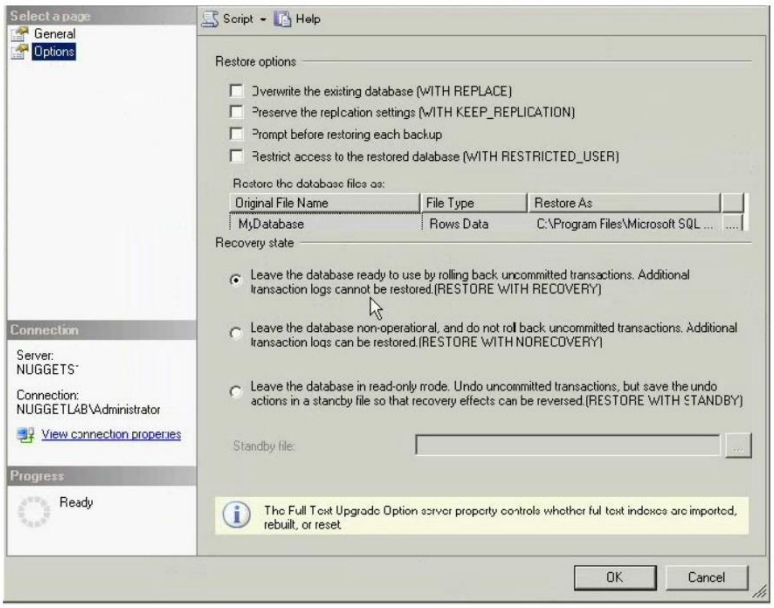


حدد قاعدة البيانات ثم من Form device وزر التفاصيل (...) اختر ملف الاسترجاع:



بعد اختيار ملف الاسترجاع، قم بتحديد (Check) العناصر التي ترغب باسترجاعها، ثم OK للموافقة على ذلك.

هناك خيارات إضافية عديدة يوفرها SQL Server Management Studio، مثل الرجوع إلى بيانات وقت معين، استرجاع البيانات مع تعويض البيانات الموجودة، استرجاع أجزاء من البيانات تدريجيا وغير ذلك...



طالع المزيد حول استرجاع البيانات في SQL Server من خلال الرابط التالي:

http://msdn.microsoft.com/en-us/library/ms177429.aspx

باستخدام SQL يمكن حفظ نسخة من البيانات بمثل الاستعلام الآتي:

```
RESTORE DATABASE [MyDatabase]
FROM  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL10.SQLEXPRESS\MSSQL\Backup\BackupMyDatabase.bak' WITH
FILE = 1,  NOUNLOAD,  STATS = 10
GO
```

ملاحظة:

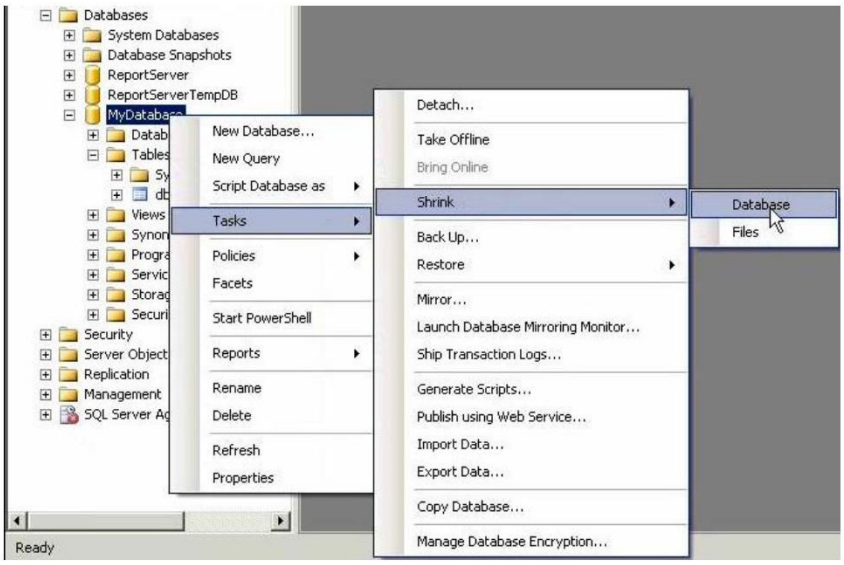
تأكد عند استرجاع نسخة من قاعدة البيانات أنك (أو النظام) لا تستخدمها حاليا، يمكنك لأجل ذلك الاتصال بقاعدة البيانات أخرى.

تقليص (Shrink) حجم ملف قاعدة البيانات:

عند إنشاء قاعدة بيانات جديدة يقوم SQL Server بإنشاء ملفات (أو ملف) قاعدة البيانات، ويحجز لذلك مساحة من القرص (افتراضيا 3 Mb للملف MDF، و1 Mb للملف LDF).

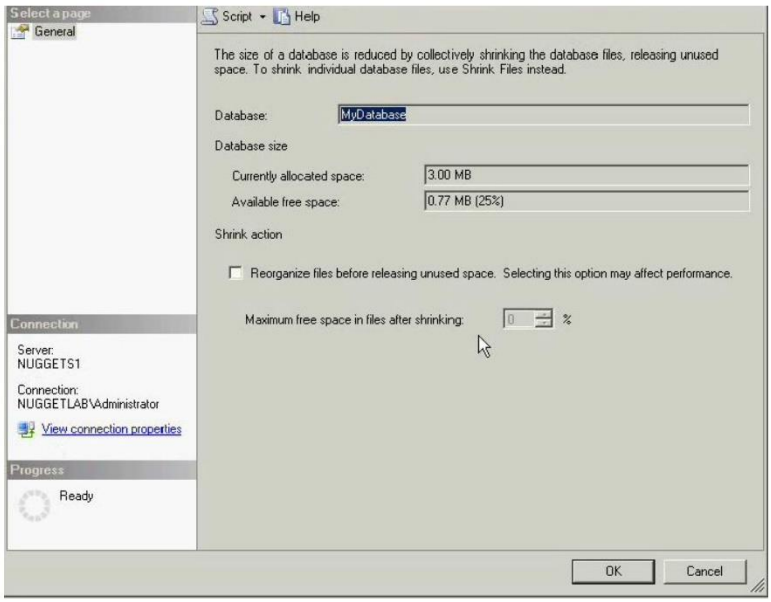
يمكنك التحكم في حجم ملفات قاعدة البيانات عند إنشائها، كما يمكنك تقليص أحجام الملفات لحجز مساحة أقل.

للقيام بذلك باستخدام SQL Server Management Studio حدد قاعدة البيانات من Object Explorer ثم Tasks -> Shrink -> Database



يمكنك تقليص حجم جميع ملفات قاعدة البيانات باختيار Database، أو تقليص حجم أحد الملفات (MDF أو LDF) باختيار Files، مع بعض الخيارات الإضافية.

تقليص حجم الملفات يعني حذف الجزء غير المستغل من قبل قاعدة البيانات، أيضا يمكن تفعيل خاصية التقليص التلقائي (Auto Shrink) من خصائص قاعدة البيانات.



في الصورة Currently allocated space يمثل الحجم الإجمالي لجميع الملفات. الحجم Available free space بنسبة 25% تمثل المساحة غير المستغلة من الملفات.

كيفية الحصول على الاستعلامات:

الآن، كيف نقوم بتقليص حجم الملفات باستخدام SQL؟

لاحظ، في جميع نوافذ الخيارات SQL Server Management Studio هناك زر Script وبه قائمة منسدلة تحمل بعض الخيارات الإضافية.

من خلال هذا الزر يمكنك الحصول على الاستعلام الذي يؤدي إلى تنفيذ الخيارات المحددة في نافذة الخيارات.

مثلا لو نقرنا في النافذة السابقة على الزر Script لحصلنا على استعلام تقليص حجم ملفات قاعدة البيانات:

مثال تدريبي محلول للتقوية في المادة

تتمحور هذه المسألة حول أتمتة مبسطة لعملية الدفع الالكتروني في الجامعة.

بحيث يقوم الطالب بتنفيذ عملية الدفع (من اجل التسجيل على مواد جديدة_ أو الاعتراض على علامة_ أو تقديم طلب كشف علامات) (في حال كانت العملية تسجيل على مواد يجب أن تكون المواد التي يرغب بالتسجيل عليها مسرودة في جدول التفاصيل بحيث بمجرد تنفيذ الدفعة يتم ادراج اسماء المواد التي دفع عنها أيضا في جدول studies profile.

آلية العمل: يوجد للطالب عدة دفعات موجودة أصلا معلوماتها في جدول payment يقوم الطالب بتنفيذ عملية الدفع عن طريق بطاقة ما. وعند نجاح عملية الدفع يتم ما يلي (تحويل حالة الدفعة ispaid إلى القيمة 1 ، في حال كانت نوع الدفعة هي تسجيل على مواد فتتم قراءتها من جدول payment_course_details ومن ثم إدراجها في جدول studies_profile مع مراعاة الفصل الذي تمت به عملية الدفع ، يتم تخزين معلومات عملية الدفع في جدول الـ Payment_log)

شرح الجداول المستخدمة

- جدول course يحتوي على معلومات عامة المواد:

Field name	Description
Course_id	رقم مميز للمادة
Course_code	رمز المادة
Course_name	اسمها
Course_program_code	رمز البرنامج الموجودة فيه المادة

- جدول term يحتوي على معلومات عامة عن الفصول:

Field name	Description
Term_id	رقم مميز للفصل
Term_name	رقم الفصل

- جدول card_info يحتوى على معلومات تتعلق بالبطاقة المصرفية الخاصة بالدفع:

Field name	Description

Card_id	رقم مميز للبطاقة
Card_number	رقم البطاقة المصرفي
Card_pass	كلمة السر
Card_expire	تاريخ انتهاء البطاقة
Card_balance	رصيد البطاقة

- بجدول student يحوي معلومات الطالب:

Field name	Description
Std_id	رقم الطالب
Std_fname	الاسم الأول
Std_lname	الكنية
Std_mail	ايميل الطالب
Std_birthdate	تاريخ تولد الطالب

- جدول payment_log يحوي أرشفة لعمليات الدفع:

Field name	Description
Pay_log_id	رقم مميز للعملية
Pay_id	توصيف الحالة
Date_time	تاريخ عملية الدفع وادراجها في هذا الجدول
Pay_type	نوع عملية الدفع

- جدول payment يحوي بيانات عملية الدفع:

Field name	Description
Pay_id	رقم عملية الدفع
Pay_balance	قيمة الفاتورة الواجب دفعها
Pay_std_id	رقم الطالب
Pay_term_id	رقم الفصل
Pay_card_id	رقم البطاقة التي تمت منها عملية الدفع (تكون فارغة في قبل تنفيذ عملية الدفع)
Pay_isPaid	حالة الدفعة
Pay_type	نوع عملية الدفع(تسجيل مواد، كشف علامات،اعتراض على علامة)

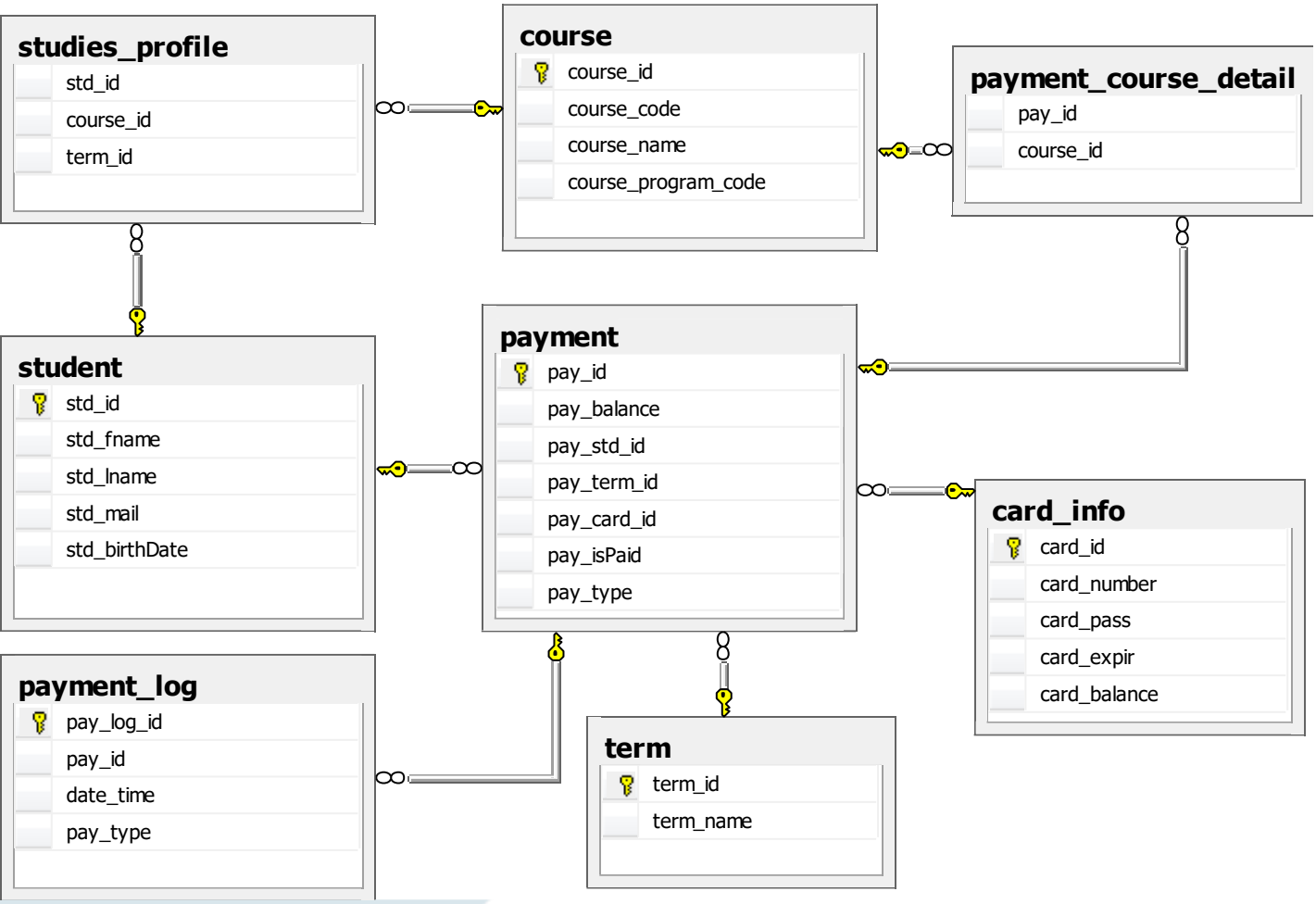
- جدول studies_profile يحوي بيانات عن مواد الطالب في كل فصل:

Field name	Description
Std_id	رقم الطالب
Course_id	رقم المادة
Term_id	رقم الفصل

- جدول payment_course_details يحوي على أسماء المواد المرتبطة مع الدفعة في حال كان نوع الدفعة هو تسجيل على مواد:

Field name	Description
Pay_id	رقم مميز للحجز
Course_id	رقم الزبون

يبيّن الشكل التالي مخطط الكيانات – العلاقات ERD الخاص بهذه القاعدة:



المطلوب:

1. إنشاء قاعدة البيانات والجداول المذكورة سابقاً (كتابة التعليمات اللازمة لإنشاء القاعدة والقيود والفهارس المرتبطة بالجداول) (**يجب كتابة التعليمات بشكل يدوي وكل كود يقوم الطالب بتوليده تلقائيا يخسر الطالب علامته بشكل كامل**).

ثم القيام بـ:

- اقتراح وإنشاء القيود مع كتابة التعليل وسبب اقتراحها: ووضعها جميعا في جدول مثل:

نوع القيد	اسمه	الجداول المشاركة	الحقول المشاركة	التعليل

- اقتراح وإنشاء الفهارس مع كتابة التعليل وسبب اقتراحها: ووضعها جميعا في جدول مثل:

نوع الفهرس	اسمه	الجدول	الحقول المشاركة	التعليل

2. إدخال البيانات:

كتابة تعليمات ادخال بيانات على جميع الجداول المذكورة سابقاً (عدة أسطر على الأقل لكل جدول)

تنفيذ الاستعلامات:

1) عرض معلومات كل طالب مع الكورسات التي سجل عليها مرتبة حسب الفصول.
2) عرض معلومات كل طالب (الاسم، الكنية، الايميل) مع قيمة مجموع دفعاته من اجل كل نوع.
3) قائمة بالطلاب الخمسة الأكثر دفعا في تاريخ الجامعة (من كل نوع الدفعات)
4) قائمة بأسماء الطلاب الذين لم يقوموا بأي عملية دفع خلال فصل F15

3. كتابة إجرائية تقوم بدفع فاتورة معينة لطالب بحيث: (متحولات الدخل: هي رقم البطاقة، رقم الطالب، رقم الفاتورة المطلوب دفعها)، متحول خرج يطبع رسالة تدل على نجاح أو فشل العملية.

يجب أن تقوم الإجرائية بالتحقق من كفاية الرصيد، وأن الدفعة ليست مدفوعة سابقا، ثم تقوم بإدراج المواد في جدول مواد الطالب في حال كانت الفاتورة "تسجيل على مواد"

تقوم بتغيير حالة الفاتورة إلى مدفوعة

قم باستدعاء الاجرائية وتنفيذ الاجرائية.

4. كتابة قادح يقوم بـ

إدراج سطر جديد في جدول Payment_log في كل مرة نقوم بعملية دفع.

قم بعملية تتطلب استدعاء القادح وتنفيذه

إنشاء قاعدة البيانات والجداول المذكورة سابقاً(كتابة التعليمات اللازمة لإنشاء القاعدة والقيود والفهارس المرتبطة بالجداول)

إن المفاتيح الرئيسية هي حقول مفهرسة بالأصل لأنها تعرف كائنات الكيان و تسمح لمحرك قاعدة البيانات بتنفيذ الاستعلامات عليها بسهولة و نوع الفهرسة افتراضيا هي فهرسة عنقودية مع ملاحظة إن البرنامج لا يسمح لنا سوى بفهرس عنقودي واحد على كل جدول في قاعدة البيانات و في حال رغبنا في تحديد فهرس آخر على انه عنقودي يمكننا ذلك من خلال إزالة إشارة التحقق عن نوع المفتاح ..

بناء على عمليات البحث التي تم ذكرها في نص الوظيفة و طبيعة العمل فقد قمنا بوضع الفهارس لتكون خصوصا على الأسماء.

نستخدم الفهارس غير العنقودية البحث عن قيمة معينة و ليس عن مجال من القيم لان المعطيات ضمن الجدول لا ترتب فيزيائيا وفق الفهرس الغير عنقودي.

نلاحظ إن وجود الفهارس بكثرة في قاعدة البيانات قد يسرع بعض عمليات البحث في بعض الأحيان إلا إن عملية تنظيم و ادارة الفهارس في قاعدة البيانات تعتبر عملية مرهقة لمحرك قاعدة البيانات.

إنشاء قاعدة البيانات:

```
CREATE DATABASE TEACHING ON  PRIMARY
( NAME = 'TEACHING_DB', FILENAME = 'C:\itd310\TEACHINGDB.mdf' )
```

LOG ON
(NAME = 'TEACHINGLOG', FILENAME = 'C:\itd310\TEACHINGLOG.ldf')

تعليمات إنشاء الجداول:

create table card_info(card_id int not null primary key,card_number varchar(50),card_pass varchar(50),card_expire date,card_balance int)

نوع القيد	اسمه	الجداول المشاركة	الحقول المشاركة	التعليل
مفتاح أساسي		*card_info*	*card_id*	رقم البطاقة رقم فريد

create table course(course_id int not null primary key,course_name varchar(50),course_code varchar(50),course_program_code int)
create index index_course on course(course_name)

نوع القيد	اسمه	الجداول المشاركة	الحقول المشاركة	التعليل
مفتاح أساسي		*course*	*course_id*	رقم المقرر رقم فريد

نوع الفهرس	اسمه	الجدول	الحقول المشاركة	التعليل
index	*index_course*	*course*	*course_name*	اي عملية لها علاقة بالمقررات تتضمن عرض اسم المقرر وبالتالي

				الوصول بسرعة الى الاسم

create table payment(pay_id int not null primary key,pay_balance int,pay_std_id int,pay_term_id int,pay_card_id int,pay_ispaid int,pay_type varchar(50))
alter table payment add constraint paymentcard_infofk foreign key(pay_card_id) references card_info(card_id)
alter table payment add constraint paymentstudentfk foreign key(pay_std_id) references student(std_id)
alter table payment add constraint paymenttermfk foreign key(pay_term_id) references term(term_id)

نوع القيد	اسمه	الجداول المشاركة	الحقول المشاركة	التعليل
مفتاح أساسي		*payment*	*pay_id*	رقم الدفعة رقم تسلسلي لايتكرر
مفتاح خارجية	*paymentcard_infofk*	*card_info*	*card_id*	الربط بين الدفعة والبطاقة التي تمت منها الدفعة
مفتاح خارجية	*paymentstudentfk*	*student*	*std_id*	الربط بين الدفعة والطالب الذي قام بدفع الدفعة
مفتاح خارجية	*paymenttermfk*	*term*	*term_id*	الربط بين الدفعة والفصل الذي تمت فيه الدفعة

create table payment_course_details(pay_id int not null,course_id int not null,constraint pk_payment_course_details primary key (pay_id,course_id))
alter table payment_course_details add constraint payment_course_detailscoursefk foreign key(course_id) references course(course_id)
alter table payment_course_details add constraint payment_course_detailspaymentfk foreign key(pay_id)
references payment(pay_id)

نوع القيد	اسمه	الجداول المشاركة	الحقول المشاركة	التعليل
مفتاح أساسي	pk_payment_course_details	payment_course_details	pay_id,course_id	الدفعة تتم مرة واحدة برقم واحد على المقرر
مفتاح خارجية	payment_course_detailscoursefk	Course	course_id	الربط بين الدفعة والمقرر
مفتاح خارجية	payment_course_detailspaymentfk	payment	pay_id	الربط بين الجدول والدفعات

create table payment_log(pay_log_id int not null primary key,pay_id int,date_time date,pay_type varchar(50))
alter table payment_log add constraint payment_logpaymentfk foreign key(pay_id) references payment(pay_id)

نوع القيد	اسمه	الجداول المشاركة	الحقول المشاركة	التعليل
مفتاح أساسي		payment_log	pay_log_id	رقم السطر لايتكرر

	payment_logpaymentfk	payment	pay_id	الربط بين سجل الدفعات والدفعات

create table student(std_id int not null primary key,std_fname varchar(50),std_lname varchar(50),std_mail varchar(50),std_birthdate date)
create index index_student on student(std_fname,std_lname)

نوع القيد	اسمه	الجداول المشاركة	الحقول المشاركة	التعليل
مفتاح أساسي		student	std_id	رقم الطالب لايتكرر

نوع الفهرس	اسمه	الجدول	الحقول المشاركة	التعليل
	index_student	student	std_fname,std_lname	أي عملية عرض للحركات تتضمن عرض اسم الطالب وبالتالي استدعاء حقلين الاسم الأول والاسم الأخير

create table student_profile(std_id int not null,course_id int not null ,term_id int not null,constraint pk_student_profile primary key(std_id,course_id,term_id))
alter table student_profile add constraint student_profilecoursefk foreign key(course_id) references course(course_id)
alter table student_profile add constraint student_profilestudentfk foreign key(std_id) references student(std_id)
alter table student_profile add constraint student_profiletermfk foreign key(term_id) references term(term_id)

نوع القيد	اسمه	الجداول المشاركة	الحقول المشاركة	التعليل
مفتاح أساسي	*pk_student_profile*	*student_profile*	*std_id,course_id,term_id*	الطالب يسجل على المادة مرة واحدة في فصل معين
مفتاح خارجية	*student_profilecoursefk*	*course*	*course_id*	الربط بين التسجيل والمقرر

مفتاح خارجية	student_profilestudentfk	student	std_id	الربط بين التسجيل والطالب
	student_profiletermfk	term	term_id	الربط بين التسجيل والفصل

create table term(term_id int not null primary key,term_name varchar(50))

نوع القيد	اسمه	الجداول المشاركة	الحقول المشاركة	التعليل
مفتاح أساسي	pk_student_profile	term	term_id	رقم الفصل فريد لايتكرر

إدخال البيانات:

```
insert into card_info (card_id,card_number,card_pass,card_expire,card_balance) values
(1,'1111','1111',null,null)

insert into card_info (card_id,card_number,card_pass,card_expire,card_balance) values
(2,'2222','2222',null,null)

insert into student (std_id,std_fname,std_lname,std_mail,std_birthdate) values
(1,'rama',null,null,null)

insert into student (std_id,std_fname,std_lname,std_mail,std_birthdate) values
(2,'student',null,null,null)

insert into term (term_id,term_name) values (1,'f15')

insert into term (term_id,term_name) values (2,'s15')
```

```
insert into course (course_id,course_name,course_code,course_program_code) values (1,'db','db1',1)

insert into course (course_id,course_name,course_code,course_program_code) values (2,'db','db1',1)

insert into student_profile (std_id,course_id,term_id) values (1,1,1)

insert into student_profile (std_id,course_id,term_id) values (1,2,1)
```

1) عرض الفصول مجموع الدفعات التي تم دفعها في كل فصلمرتبة ابتداء من أكبر دفعة.

```
select sum(pay_balance),term_name from payment  inner join term
on payment.pay_term_id =term.term_id
group by term_name
order by sum(pay_balance) desc
```

2) عرض معلومات كل طالب (الاسم، الكنية، الايميل)(لجميع الطلاب سواء كان له دفعة أم لاء) مع قيمة مجموع دفعاته إن وجدت من اجل كل نوع.

الحصول على دفعات الطالب حسب النوع من خلال استعلام فرعي يقوم بتجميع الدفعات وفق رقم الطالب ثم الربط مع جدول الطلاب من اجل الحصول على معلومات الطلاب:

قمنا باستعمال الربط الخارجي من اجل الحصول على معلومات كامل الطلاب مع الربط مع استعلام مجموع الدفعات لكل طالب وحسب نوع الدفعة

```
SELECT        student.std_fname, student.std_lname, std_mail, tbl.sum_pay, tbl.pay_type
FROM            student FULL OUTER JOIN
                        (SELECT        SUM(pay_balance) AS sum_pay, pay_std_id, pay_type
                          FROM            payment
                          GROUP BY pay_std_id, pay_type) AS tbl ON student.std_id = tbl.pay_std_id
```

3) عرض اسماء المواد جميعها مع ذكر أرقام وأسماء الطلاب المسجلين إن وجدوا.

قمنا باستخدام الجداء الخارجي من اجل عرض جميع المواد حتى لو كان الطلاب غير مسجل بالمادة

```
SELECT        course.course_name, student.std_fname, student.std_lname
FROM            payment_course_details FULL OUTER JOIN
                  course ON payment_course_details.course_id = course.course_id INNER JOIN
                  payment ON payment_course_details.pay_id = payment.pay_id INNER JOIN
                  student ON payment.pay_std_id = student.std_id
```

4) قائمة بجميع المواد مع ذكر مجموع ما تم دفعه من تسجيلات على هذه المادة.

```
SELECT        course.course_name, SUM(payment.pay_balance) AS Expr1

FROM            course INNER JOIN
```

```
                        payment_course_details ON course.course_id =
payment_course_details.course_id INNER JOIN
                        payment ON payment_course_details.pay_id = payment.pay_id
GROUP BY course.course_name
```

5) قائمة بأسماء الطلاب الذين لم يقوموا بأي عملية دفع خلال فصل F15

الحصول على الطلاب الذين ليس لهم دفعات في عام *F15* من خلال استعلام فرعي يقوم بالحصول على الطلاب الذين دفعوا في الفصل المطلوب ثم الحصول على معلومات الطلاب الذين لم ترد ارقامهم في الاستعلام السابق.

```
select * from student where std_id not in(
select pay_std_id from payment where Pay_term_id =(select term_id from term where term_name='F15'))
```

كتابة إجرائية تقوم بدفع فاتورة معينة لطالب بحيث: (متحولات الدخل: هي رقم البطاقة، رقم الطالب، رقم الفاتورة المطلوب دفعها)، متحول خرج يطبع رسالة تدل على نجاح أو فشل العملية.

يجب أن تقوم الإجرائية بالتحقق من كفاية الرصيد، وأن الدفعة ليست مدفوعة سابقا، ثم تقوم بإدراج المواد في جدول مواد الطالب في حال كانت الفاتورة "تسجيل على مواد"

تقوم بتغيير حالة الفاتورة إلى مدفوعة

قم باستدعاء الاجرائية وتنفيذ الإجرائية.

الإجراء المخزن المطلوب يقوم بالحصول على المبلغ الموجود في البطاقة و المبلغ المطلوب في الدفعة والمقارنة وبنفس الشرط يقوم بالتحقق من إن الفاتورة لم يتم دفعها من قبل وفي حال كانت مدفوعة يتم طباعة رسالة خطأ وإلا يتم تعديل حالة الفاتورة وإضافة المادة إلى جدول بروفايل الطالب:

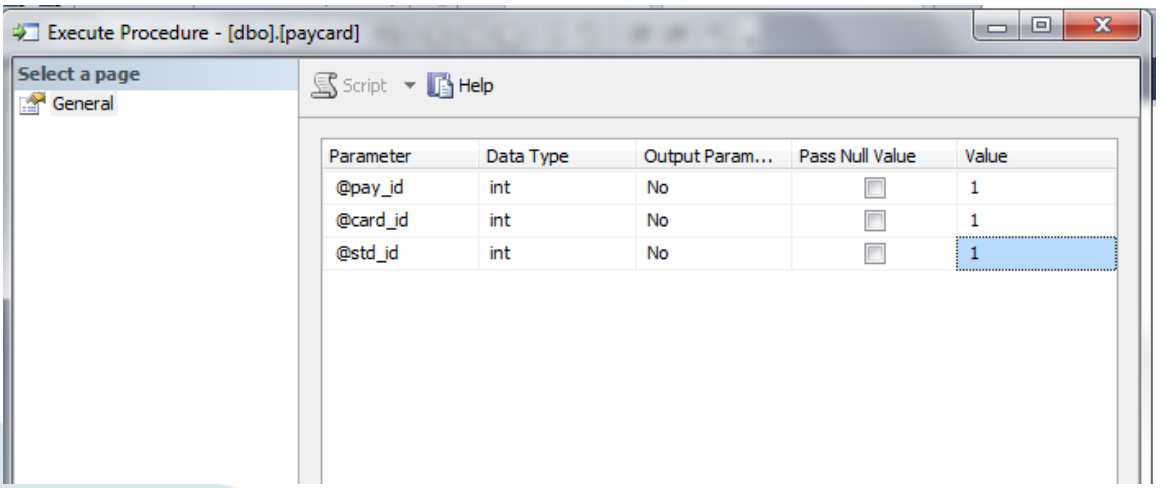
```
create procedure paycard(@pay_id int ,@card_id int ,@std_id int )
as
declare @bal int

declare @ispaid int
declare @pay_type varchar(10)
declare @termid int
select @bal =pay_balance,@ispaid =Pay_isPaid,@pay_type =Pay_type,@termid=pay_term_id   from payment
where pay_id=@pay_id
declare @cardbal int

select @cardbal=card_balance from card_info where card_id=@card_id
if (@bal>@cardbal ) or (@ispaid =1)
 print ('error')
```

```
 else
 begin
 update payment set Pay_isPaid =1 where Pay_id =@pay_id
if (@pay_type='registeration')
begin
insert into student_profile select @std_id,course_id,@termid from payment_course_details where Pay_id
=@pay_id
end
END
```

تنفيذ الاجراء المخزن:



كتابة قادح يقوم بإدراج سطر جديد في جدول **Payment_log** في كل مرة نسجل فيها دفعة جديدة

القادح المطلوب يقوم بالحصول على معلومات الفاتورة التي تم تعديلها ومن ثم يقوم بإضافة القيم المطلوبة إلى سجل الدفعات

```
create trigger log_trig on payment for insert
as
declare @payid int
declare  @pay_type varchar(10)
select @payid =pay_id,@pay_type=pay_type from inserted
insert into payment_log(pay_id,date_time,pay_type) values(@payid ,GETDATE(),@pay_type)
```