جامعة المنارة

كلية: الهندسة

قسم: المعلوماتية

اسم المقرر: قواعد بيانات ٣

رقم الجلسة (٤)

عنوان الجلسة

المعاملات و الأقفال في sql server

Transactions & Locks

د. ايهاب ديباجة

م.جمال محمود – م.جهاد عيسى

الفصل الدراسي : الثاني

العام الدراسي :٢٠٢٣-٢٠٢٤

جدول المحتويات

Contents

الغاية من الجلسة:

- ✓ شرح مفهوم المعاملات في sql server مع أمثلة عملية
- ✓ شرح أنواع الأقفال قي sql server مع الأمثلة عليها

المعاملات Transactions في sql server

المعاملة هي تعليمة أو مجموعة تعليمات نوع DML يتم تنفيذها كوحدة عمل واحدة. إذا نجحت المعاملة، فسيتم الالتزام بجميع تعديلات البيانات التي تم إجراؤها أثناء المعاملة وتصبح جزءًا دائمًا من قاعدة البيانات. إذا واجهت معاملة ما أخطاء ويجب إلغاؤها أو التراجع عنها، فسيتم مسح جميع تعديلات البيانات.

تملك المعاملات في sql server الخصائص التالية :

- ✓ التثبيت التلقائي للمعاملات
- ✓ كل تعليمة منفردة تعتبر معاملة

المعاملات الصريحة

تبدأ كل معاملة بشكل صريح بعبارة BEGIN TRANSACTION وتنتهي بشكل صريح بعبارة COMMIT أو ROLLBACK.

المعاملات الضمنية

تبدأ معاملة جديدة ضمنيًا عند اكتمال المعاملة السابقة

فيما يلي الأوامر المستخدمة للتحكم في المعاملات:

- ✓ BEGIN TRANSACTION: هو الأمر الذي يشير إلى بداية كل معاملة.
- ✓ COMMIT: هو أمر يستخدم لحفظ التغييرات بشكل دائم في قاعدة البيانات.
- ✓ ROLLBACK: هو أمر يستخدم لإلغاء جميع التعديلات والعودة إلى حالتها السابقة.
- ✓ SAVEPOINT: يقوم هذا الأمر بإنشاء نقاط ضمن مجموعات من المعاملات التي تسمح لنا باستعادة جزء فقط من المعاملة بدلاً من المعاملة بأكملها.
- ✓ RELEASE SAVEPOINT: يتم استخدامه لإزالة نقطة حفظ موجودة بالفعل.
- ✓ SET TRANSACTION: يمنح هذا الأمر اسمًا للمعاملة، والذي يمكن استخدامه لجعلها للقراءة فقط أو للقراءة/الكتابة أو تعيينها لمقطع تراجع محدد

أمثلة :

```
create database test1

use test1

create table emp_test(id1 int,name1 varchar(50));
```

لنقم الآن بتنفيذ المعاملة التالية :

```
Begin Tran t1

Insert Into emp_test Values ( 1,'kareem')
```

```
  Insert Into emp_test Values ( 2,'yara')
  Insert Into emp_test Values ( 3,'sami')
Rollback Tran t1
select * from emp_test
```

نلاحظ أن المعاملة نفذت rollback و بالتالي لا يتم ادخال أي بيانات على الجدول

و الأن لنقم بإضافة سطر واحد save t1 قبل ال rollback كالتالي

```
Begin Tran t1
Insert Into emp_test Values ( 1,'kareem')
  Insert Into emp_test Values ( 2,'yara')
  Insert Into emp_test Values ( 3,'sami')
  save tran t1
Rollback Tran t1
select * from emp_test
```

و هنا تم تثبيت المعاملة قبل تعليمة التراجع و بالتالي لم يعد لتعليمة التراجع أي معنى

الآن لنقم بإضافة savepoint عدد اثنين إلى المعاملة الأول بعد ادخال البيانات و الثاني بعد التعديل و لننفذ المعاملة التالية :

```
Begin Tran T1
delete from emp_test
Insert Into emp_test Values ( 1,'kareem')
  Insert Into emp_test Values ( 2,'yara')
  Insert Into emp_test Values ( 3,'sami')
 Save Tran SP1;
  Update emp_test  Set name1 ='Sari' Where id1=1
  Update emp_test  Set name1 ='sara' Where id1=2
  Update emp_test  Set name1 ='mary' Where id1=3
 Save Tran SP2
  delete from  emp_test
```

```
Rollback Tran SP2
  Delete From emp_test Where id1 =1
   Commit Tran T1
   Select * From  emp_test
```

نلاحظ هنا الخرج يعطينا الادخال و التعديل sp2 و لكن يتم التراجع عن الحذف لكل سجلات الجدول

```
Delete from emp_test
```

و لكن يتم حذف السجل ١ لأن التعليمة بعد rollback و بعدها commit

و يمكن تغيير rollback sp2 و استبدالها ب rollback sp1 اشرح النتيجة

Locks الأقفال

القفل عبارة عن آلية مرتبطة بجدول لتقييد الوصول غير المصرح به إلى البيانات. يتم استخدامه بشكل أساسي لحل مشكلة التزامن في المعاملات. يمكننا تطبيق قفل على مستوى الصف ومستوى الجدول ومستوى الصفحة ومستوى قاعدة البيانات.

في قواعد البيانات يمكن لعدة مستخدمين الوصول إلى موارد قواعد البيانات في نفس الوقت. ونتيجة لذلك، يعد القفل ضروريًا لنجاح المعاملة (Transaction) ويحمي البيانات من التلف أو الإبطال عندما يحاول العديد من المستخدمين قراءة قاعدة البيانات أو كتابتها أو تحديثها. عادةً ما يكون القفل عبارة عن بنية في الذاكرة تحتوي على المالكين والأنواع وتجزئة المورد الذي من المفترض أن يقوموا بحمايته.

يجب أن نفهم أن "القفل مصمم لضمان سلامة البيانات واتساقها مع تمكين الوصول المتزامن للبيانات، لأنه يفرض على كل معاملة اجتياز اختبار ACID. عندما يصل عدة مستخدمين إلى قاعدة بيانات لتغيير بياناتها في نفس الوقت، فإنه ينفذ التحكم في التزامن."

يحتوي اختبار ACID على المتطلبات التالية لإجراء المعاملة

- ✓ وحدة العمليات Atomicity : تضمن نجاح جميع البيانات أو العمليات المضمنة في المعاملة. وبخلاف ذلك، سيتم إرجاع العمليات إلى حالتها السابقة.
- ✓ الاتساق Consistency : يضمن أن قاعدة البيانات يجب أن تقوم فقط ببناء حالة صالحة عند تنفيذ المعاملة بنجاح.
- ✓ العزل Isolation : يضمن أن جميع المعاملات مستقلة عن المعاملات الأخرى. كما يضمن أيضًا أن تكون البيانات شفافة بالنسبة لبعضها البعض.
- ✓ المتانة Durability : تضمن بقاء نتيجة المعاملات الملتزم بها في قاعدة البيانات بشكل دائم، حتى في حالة تعطل النظام أو فشله.

أين يتم وضع الأقفال في قاعدة البيانات؟

الآن، سنعرف أين توجد الأقفال فعليًا في قاعدة البيانات، أي المورد الذي يتم قفله أو لا يتم قفله عليه.
يوضح الجدول أدناه الموارد التي يمكن لـ SQL Server وضع الأقفال عليها:

أنواع الأقفال

يتم استخدام وضع القفل لمنع الأشخاص الآخرين من قراءة المورد المقفل أو تغييره. و يوجد ستة أنواع أساسية للقفل سنتعرف على ما يلي منها و التي يتم تطبيقها على مستوى السجل

- ✓ القفل الحصري (X)
- ✓ القفل المشترك (S)
- ✓ قفل التحديث (U)

القفل الحصري Exclusive Lock (X)

تعتبر الأقفال الحصرية مفيدة في عمليات DML مثل عبارات INSERT أو UPDATE أو DELETE. هذا القفل، عند فرضه على معاملة ما، يمنع الأشخاص الآخرين من الوصول إلى الموارد المقفلة. وهذا يعني أن القفل الحصري يمكنه الاحتفاظ بمعاملة واحدة فقط على المورد في نفس الوقت. يُعرف مستخدم هذا القفل بالكاتب. يتم فرض هذا القفل عندما تريد المعاملة تعديل بيانات الصفحة أو الصف. لا يمكن الاحتفاظ به إلا من خلال الصفحة أو الصف عندما لا يكون هناك تعليق تأمين مشترك أو حصري آخر على الهدف.

الأقفال المشتركة Shared Locks

بمجرد تطبيق التأمين المشترك على الصفحة أو الصف، سيتم حجزه لأغراض القراءة فقط. وهذا يعني أنه لا يمكن لأي معاملة أخرى تغيير المورد المقفل طالما أن القفل نشط. كما يوحي الاسم، يمكن لعدة معاملات الاحتفاظ بهذا القفل على نفس المورد في وقت واحد. يُعرف مستخدم هذا القفل بالقارئ. بالإضافة إلى ذلك، سيسمح هذا القفل أيضًا بعمليات الكتابة، ولكن لن يُسمح بإجراء تغييرات على DDL.

قفل تحديث البيانات Update Locks

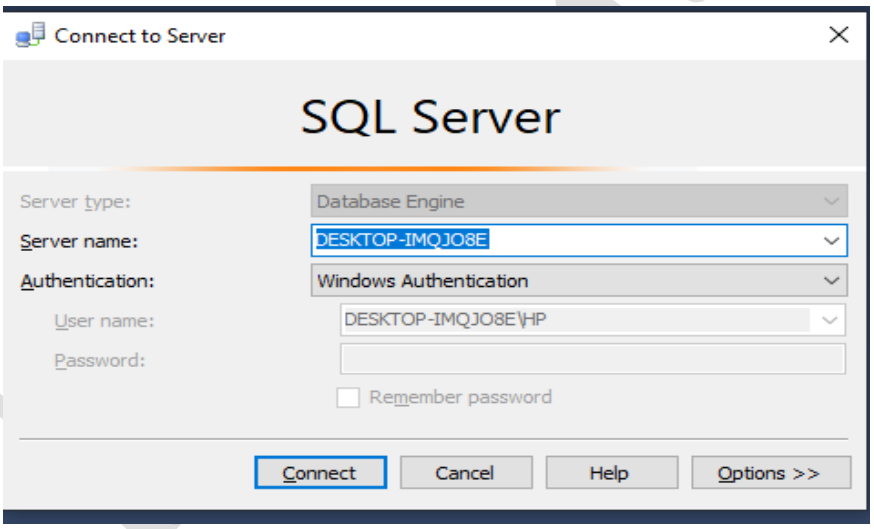
قفل التحديث هو نفس القفل الحصري، ولكنه مصمم ليكون أكثر قابلية للتكيف. يمكن منح المعاملة التي تحتوي بالفعل على قفل مشترك قفل تحديث. في مثل هذه الحالات، يمكن أن يحتفظ قفل التحديث بقفل مشترك آخر على الصفحة أو الصف المستهدف. يمكن تغيير هذا القفل إلى قفل خاص عندما تقوم المعاملة التي تحتوي على قفل التحديث بتغيير البيانات. يتم استخدامه بشكل عام عندما يقوم الخادم بتصفية السجلات لإجراء التحديث.

الجدول التالي يوضح التكامل بين أنواع الأقفال الثلاثة المشروحة سابقاً

Modes	Exclusive (X)	Shared (S)	Update (U)
Exclusive (X)	✗	✗	✗
Shared (S)	✗	✓	✓
Update (U)	✗	✓	✗

مثال :

لنفرض الدخول بحساب لديه سماحية الدخول من نظام التشغيل و لديه سماحية مالك قاعدة البيانات



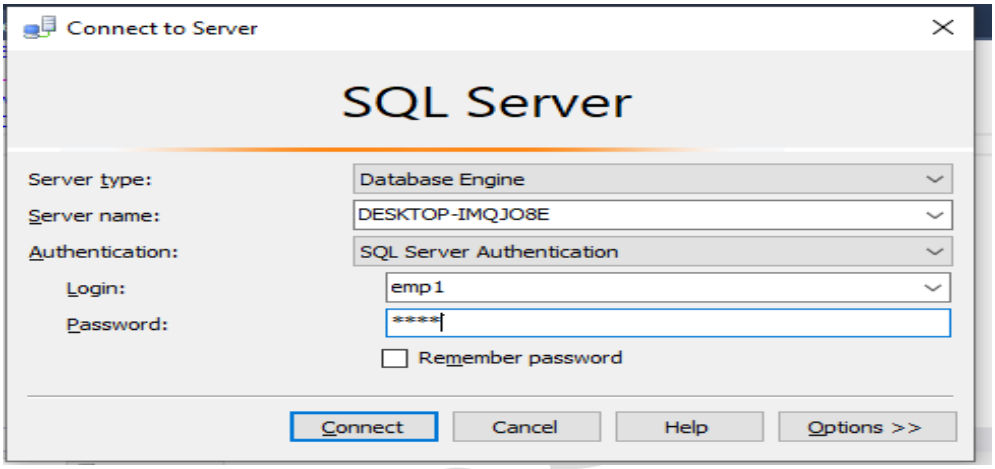

```
CREATE TABLE  lock_t
(Id INT ,
Name VARCHAR(100))
INSERT INTO lock_t
values(1, 'Flutter'), (2, 'MySQL');
```

في الخطوة التالية سنقوم بتنفيذ معاملة لتحديث البيانات و من ثم تحليل الأقفال على الموارد

```
BEGIN TRAN
UPDATE lock_t   SET   Name='SQL Server' where Id=2
WAITFOR DELAY '00:03:00'
```

Commit tran

الأن عند الدخول بحساب آخر لديه الصلاحية لتعديل السجل نفسه فإن تنفيذ العملية لن يتم حتى يتم انتهاء المعاملة الأولى



و عند الدخول إلى قاعدة البيانات test1 و تنفيذ التعليمة

```
UPDATE lock_t   SET   Name='SQL Server' where Id=2
```

فلن يتم تنفيذها حتى انتهاء المعاملة من الحساب الآخر

و يمكن الاستعلام عن الاقفال باستخدام الاستعلام التالي :

```
SELECT * FROM sys.dm_tran_locks  WHERE request_session_id=
 (SELECT @@SPID AS session_id  )
SELECT * FROM sys.dm_tran_locks;
```