# Introduction About T-SQL

* Transact-SQL (T-SQL)
* What is T-SQL?

An extension of SQL developed by Microsoft

* Which of the following best describes T-SQL?

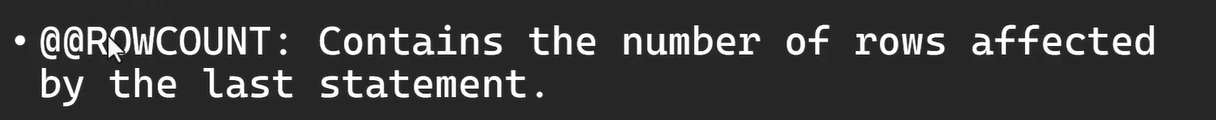
A programming language for managing relational databases

* Transact-SQL (T-SQL) provides a robust programming language with features that let you temporarily store values in variables, apply conditional execution of commands, pass parameters to stored procedures, and control the flow of your programs.
* Which feature is provided by T-SQL for handling errors?

TRY...CATCH blocks

# Variables

Special Variables @@ROWCOUNT



# Error Handling with IF

* بشكل عام يتنفيذ التعليمات بشكل متسلسل، وعند تنفيذ كل تعليمة برمجية يقوم ال TSQL بإرجاع رقم 0 ضمن ال special variable @@ERROR اذا كانت العملية ناجحة، وإلا برجع رقم ال error.
* لكن بدك ننتبه لإستعمالها بشكل صحيح، يعني اذا عطت error والجملة اللي بعدها كانت صح رح ترجع 0 وتتجاهل رقم ال error

# Case Statement

* ماعندي switch -case ضمن ال T-SQL متل لغات البرمجة الأخرى، لكن فيني استخدمها ضمن عدة سياقات كالتالي:

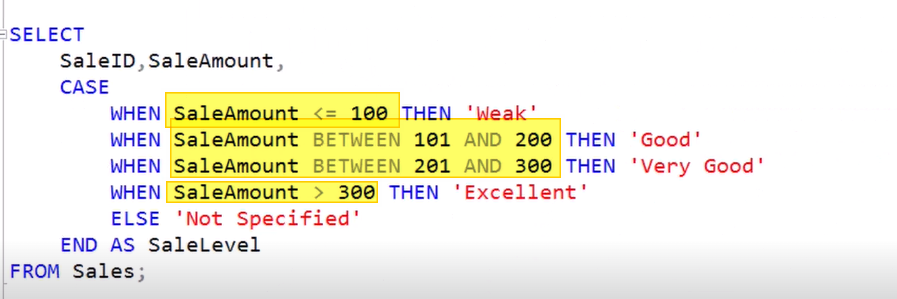
1. Simple CASE as SWITCH

* ضمن ال select statement



1. Searched CASE (More Flexible)

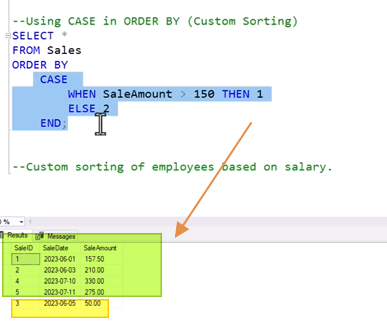
* ليش سموها searched ؟ **لاني بحط ضمنها شرط وليس قيمة ثابتة**، كما موضح بالشكل التالي:



1. Using CASE in ORDER BY (Custom Sorting)

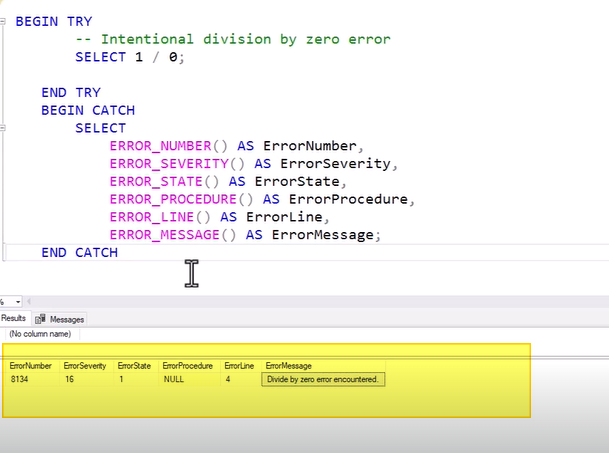
* **انتبه هون الفكرة كتير مهمة:**

بالمثال التالي رح يقوم بإنشاء **عمود وهمي** ويعطيه قيمة 1 اذا كان ال saleAmount > 150 وكل شي تحت ال 150 رح يعطيه القيمة 2، وبعدين رح يرتب الداتا ويعرض ال 1 بالاول وبعدين ال 2 (يعني هون عطيت Priorety Or Rank بالترتيب – لاحظ انه عرض ال 50 بالأخير)



# Error Handling

Error Functions



# Transactions

* اذا عندي أكثر من Transaction ورا بعض وبدي اتأكد انه الكل تنفذو وإلا بلغي العملية كلها.
* هون لازم افهم مصطلح جديد اسمه ال **ACID Properties** سوف نشرحه بالتفصيل كمايلي:

1. Atomicity:

يعني يجب تنفيذ كامل ال Transactions حتى يتم الحفظ، وإلا يتم الغائهم جميها

1. Consistency:

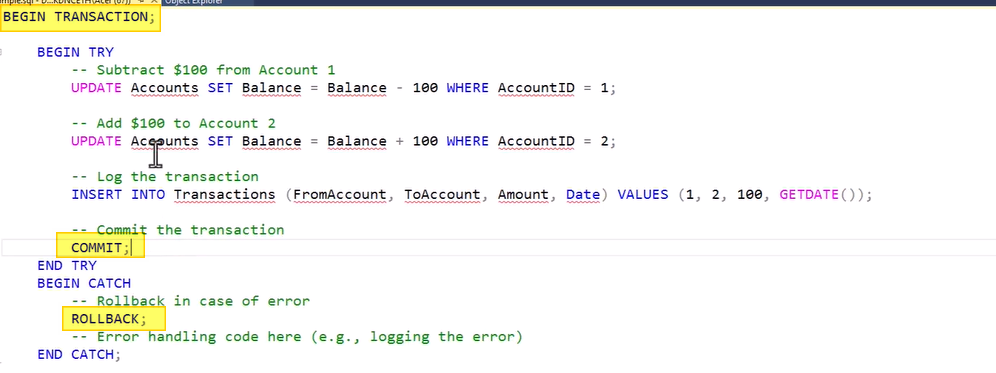
بتحترم قواعد ترابط الجداول ضمن الداتابيز

1. Isolation:

يتم فصل كل Transaction لحالها، لكن بتم تنفيذ جميع ال Transactions ضمن block واحد (**مشان حقق المفهوم الأول Atomicity)**

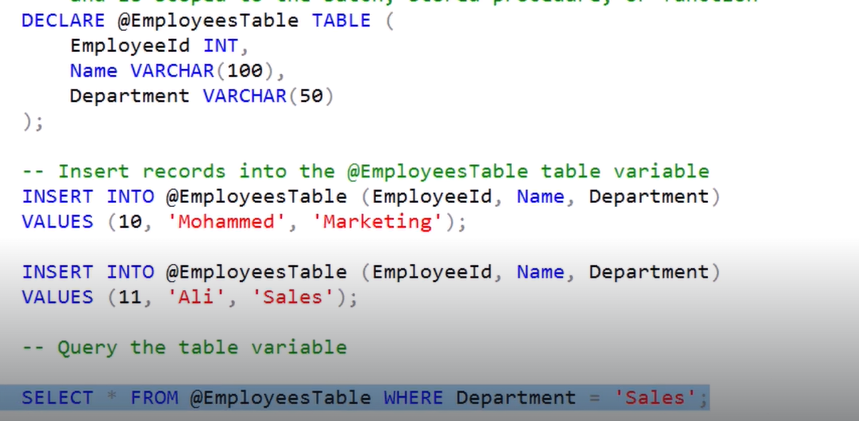
1. Durability:

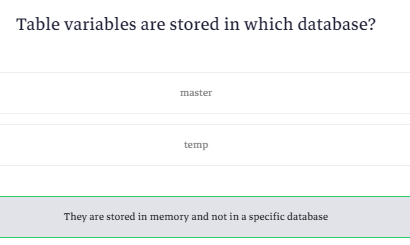
اخر شي لما يتم تنفيذ جميع ال Transactions بشكل صحيح وهو وعم يعمل commit وانقطعت الكهربا مثلا (طبعا في حال قلتله **Begin Transaction** هو بخزن كل التعليمات ضمن ال Log File) بيرجع بيقرأ ال log file وبنفذ كل شي تعليمات معلقة

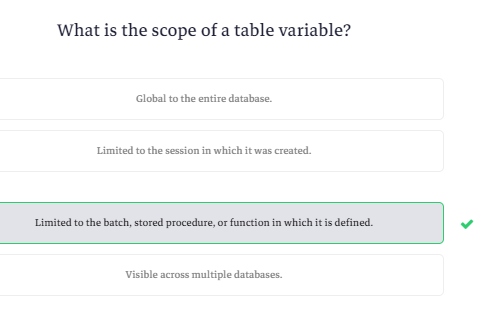


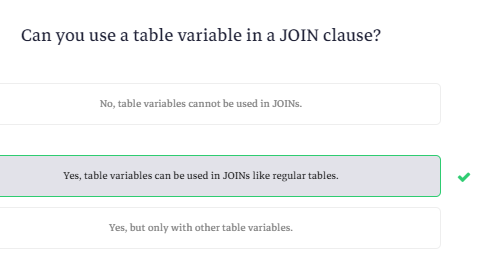
# Variable Table

* فينا نعمل variable من نوع Table ونعمل عليه كامل عمليات ال CRUD ماعدا ال Drop (بيخلص ضمن ال scope بشكل تلقائي) وكمان joins وغيرها...



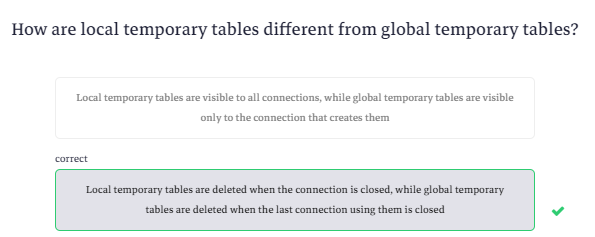


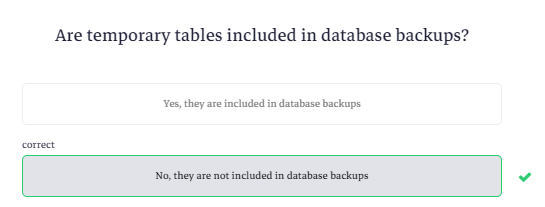




# Temporary Tables

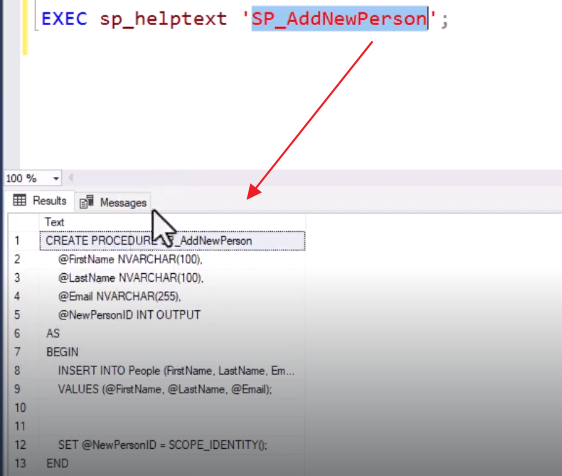
* بينخلق ضمن ال Temp Database وفيك تعمل عليه كامل عمليات ال CRUD ويفضل انك تعمله Delete بس تخلص منه (مع العلم أنه يتم حذفه تلقائيا بعد فترة)
* اله نوعين Local انت فقط بتشوفه وبكون مسبوق ب # ، والنوع التاني global الكل بشوفه وبكون مسبوق ب ##





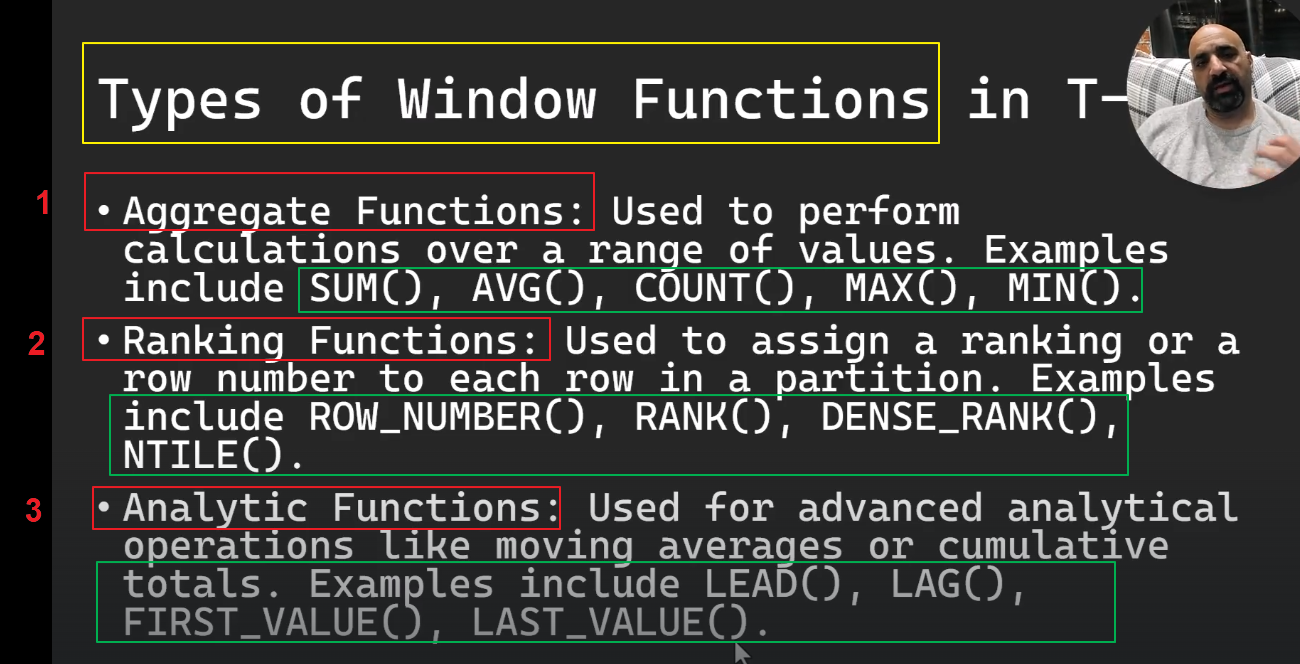
# Stored Procedures: SP\_HELPTEXT command

* بتعرضلك كود البروسيجر كما موضح بالشكل التالي:



# What are Window Functions?

* هنن نفسهن ال Analytical Functions
* الهن عدة أنواع كما موضح بالشكل التالي:

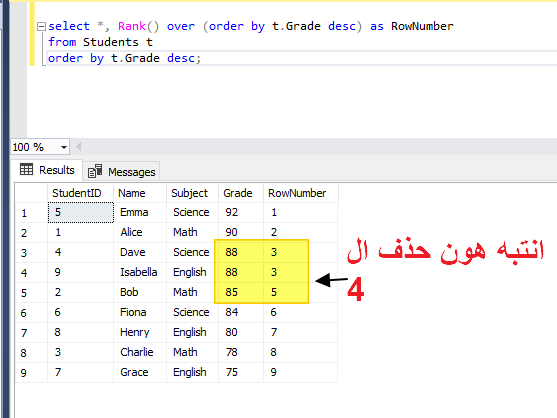


Aggregation Functions

Ranking Functions

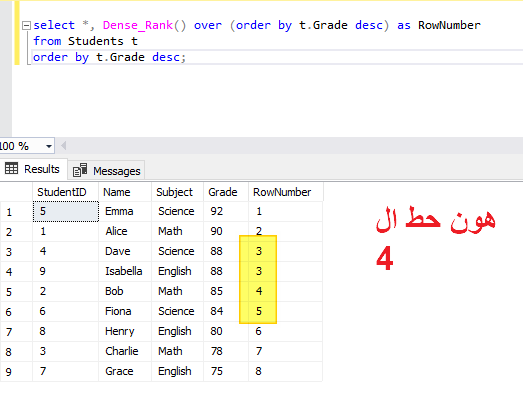
Rank ()

* بيعطي أرقام حسب ال Rank (**المتشابهين بيعطيهن نفس ال Rank لكن انتبه بيحذف الرقم التالي عند عملية التكرار**) كما موضح بالشكل التالي:



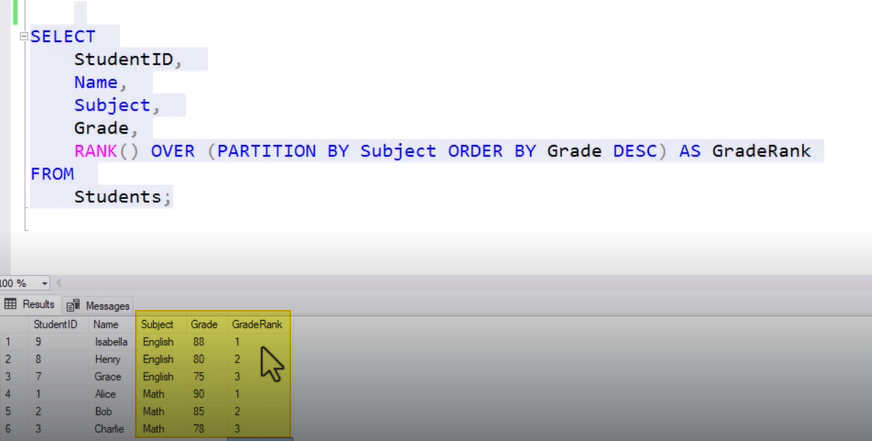
Dense\_Rank

بيحل مشكلة حذف الرقم بال Rank() كما موضح بالشكل التالي:



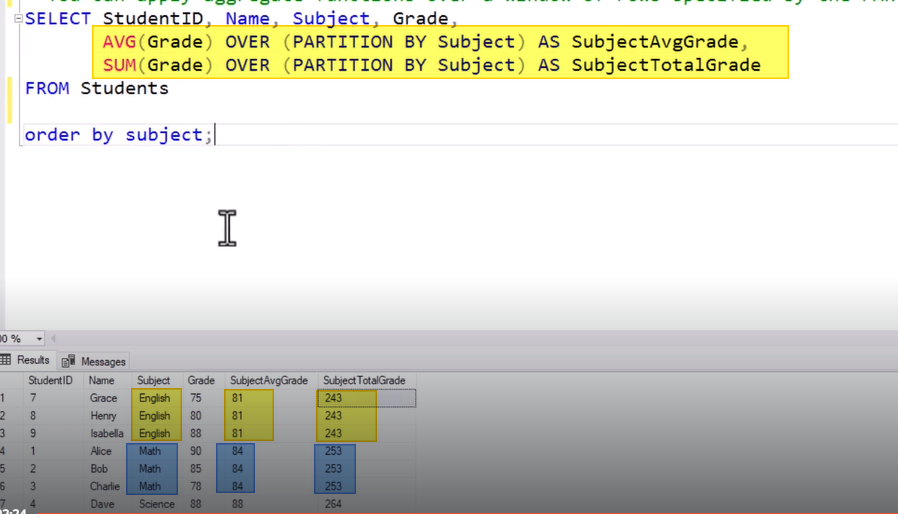
Using the RANK () Function with PARTITION BY

لاحظ هون انه قسمهن حسب المواد بعدين عملهن Rank حسب كل مادة (**يعني ال Rank اجى ع ال subject وليس ال grade**) كما موضح بالشكل التالي:



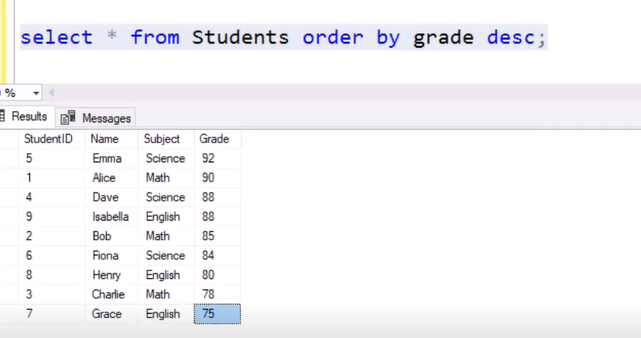
Aggregate Functions with Partition

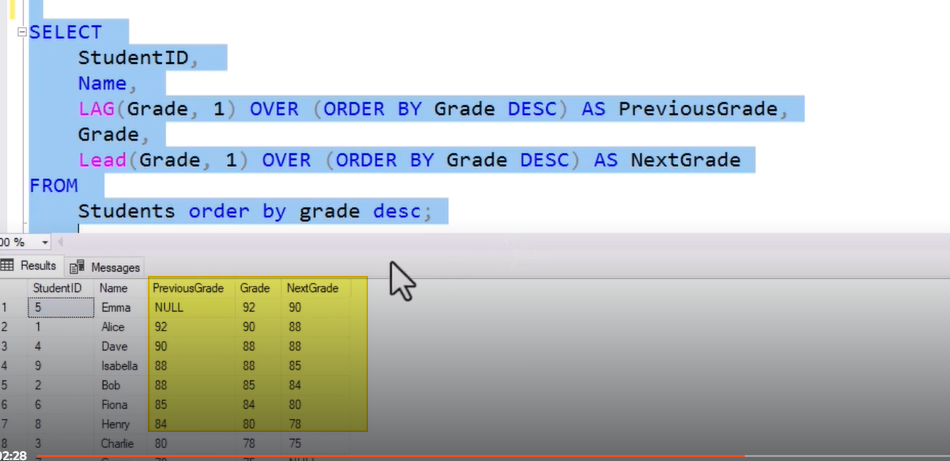
هون بدي جيب اسم الطالب وعلامته وحط جنبهن ال avg and sum كما موضح بالشكل التالي:



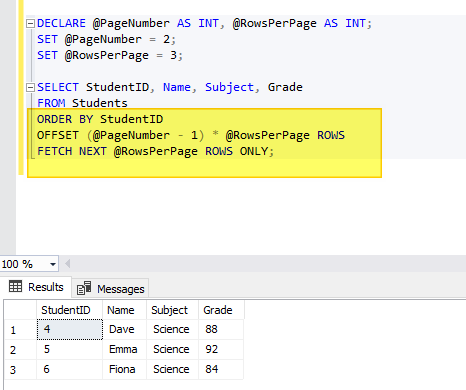
Exploring LAG and LEAD Functions Using a Single SQL Query

اذا عندي هي الداتا وبقلك بدي العلامة للسطر اللي قبل أو بعد السطر رقم 2 (العلامة بالسطر 2 هي 90 والعلامة اللي بالسطر اللي قبلها هي 92 واللي بعدها هي 88 ) **عن طريق ال LAG السطر اللي قبل وال LEADالسطر اللي بعد** وهكذا...



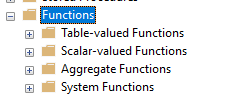


Paging in SQL using OFFSET and FETCH NEXT



# T-SQL Function Types

* فكرة مهمة: **كل ال functions هم read only** (يعني مافيك تستخدم ضمنهن تعليماتinsert, update, delete, merge فقط تعليمات select)
* يوجد العديد من أنواع ال functions كما موضح بالصورة التالية:



Scalar Functions

* فنكشن بيعمل لوجيك معين (ممكن انت تعمله او بكون جاهز) **برجع قيمة واحدة حصرا**

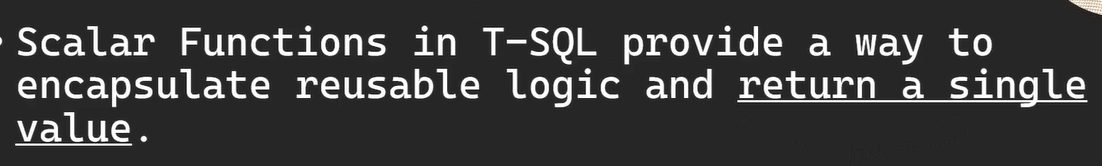
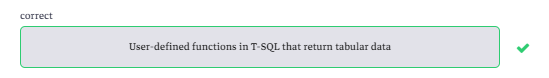
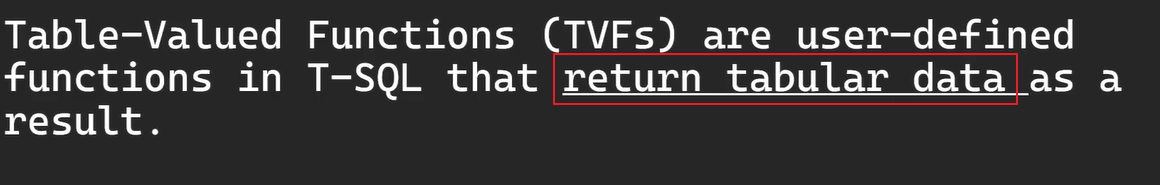


Table-Valued Functions - TVFs

* برجع Table، وبتعمل فيها كل العمليات على الجداول (بتتعامل معهن وكأنه جدول حقيقي ضمن الداتابيز)





له نوعان:

1. Inline Table-Valued Function – ITVFs:

* بيسمحلك استخدام جملة **select statement واحدة فقط ضمنه** (مافيك تحط اكتر من وحدة ضمنه)
* ال return تبعه بكون جدول Table

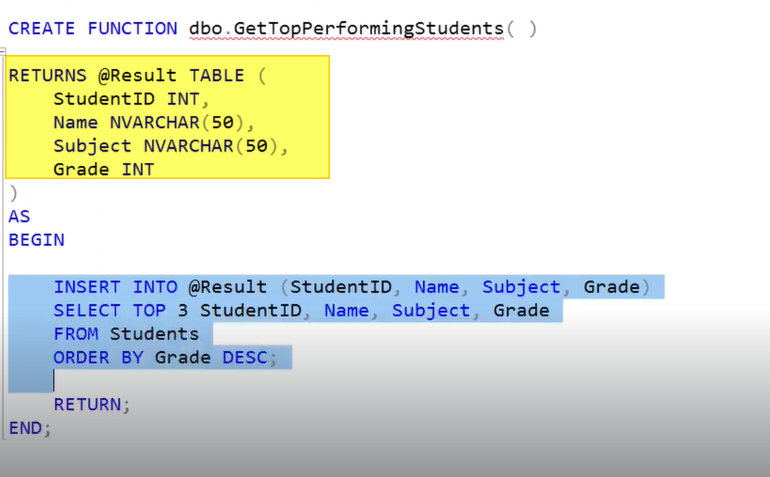




1. Multi-Statement Table-Valued Functions – MTVFs:

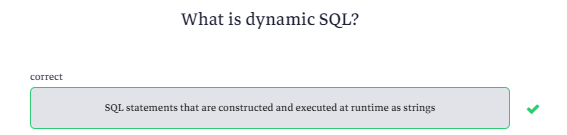
* بيسمحلك استخدام اكثر من جملة select statement
* هاد بعرفلك variable table (**يعني فيك تعمل عليه كامل العمليات على الجدول insert, update, delete وليس select فقط، طبعا عليه وليس على الجدول الأساسي**)



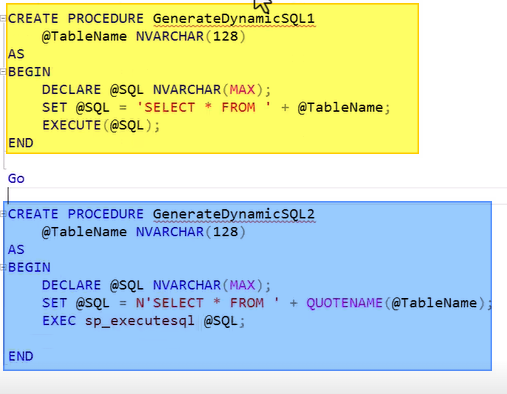


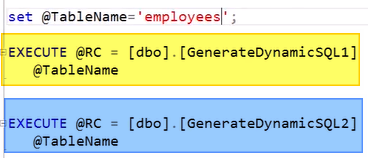
# Dynamic SQL

* الفكرة منها انه تشكل جملة ال sql تبعك عن طريق string بشكل dynamic



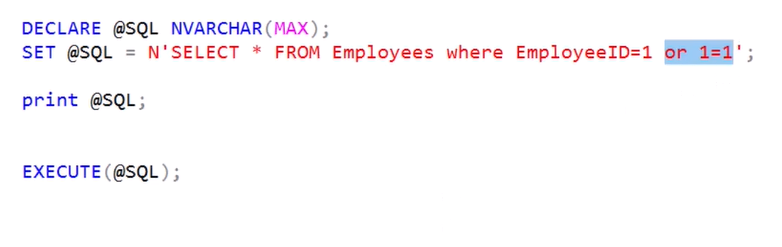
* لايفضل استخدامها لآنها أبطأ





# SQL injection attack

* ثغرات أمنية بسبب استخدام ال dynamic sql
* في المثال التالي اذا بحقن الشرط or 1=1 برجع كامل الداتا (**تخيل انها جملة update بتخرب الداتا كلها**)



* **الحل استخدام select statement عادية مع بارامترات لنرفع مستوى ال security**



# Triggers

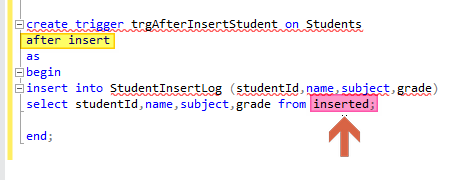
* ببساطة هو نفسه ال Events (لما يصير شي ناديلي هي الشغلة) وفيني طبقه على ال tables and views
* هو نوع خاص من ال stored procedures
* اله نوعين، الأول After Triggers (insert – update - delete) والثاني Instead of triggers مثلا لما بدك تعمل delete من جدول معين مابدك ال sql server هو اللي يعمل ال delete فيك تعمل override لل delete الأساسية وتخليه ينادي ال stored procedure تبعك اللي فيها ال business تبعك
* ال Trigger مكون من 3 أمور رئيسية:

1. Trigger Event
2. Trigger Condition
3. Trigger Action

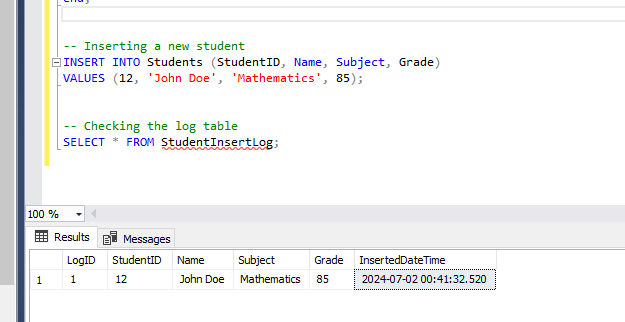
After Insert Triggers

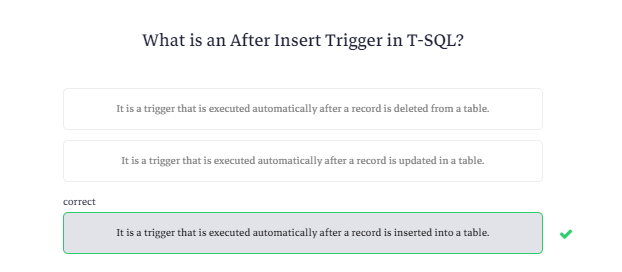
1. After Insert Triggers:

* يتم مناداته لما حدا يعمل insert على الجدول، في المثال التالي بدي اعمل log لكل عملية insert على جدول ال students كما موضح بالشكل التالي:



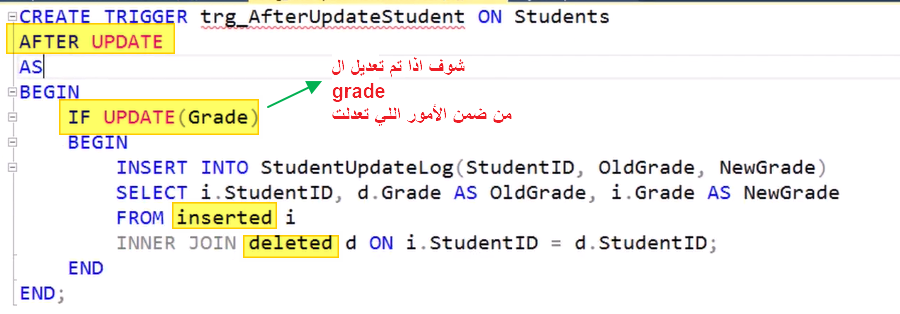
* انتبه على ال **inserted** : ال trigger يقوم بخلق جدولين مشان track الداتا، الأول اسمه inserted بحط فيه الداتا اللي عم يتم ادخالها (ويتم استخدامه مع ال update أيضا)، والثاني اسمه deleted بينحط فيه ال deleted records



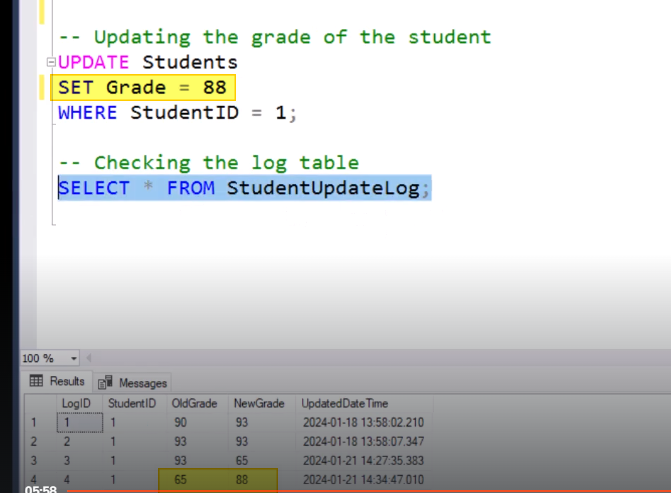


After Update Trigger

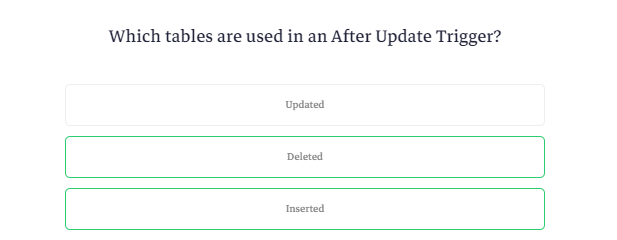
* يتم مناداته لما حدا يعمل update على الجدول
* في المثال التالي بدي خزن علامة الطالب القديمة في حال تم تعديلها، كما موضح بالشكل التالي:



* **انتبه: لما تعمل update يتم انشاء الجدولين inserted and deleted بشكل temporary (inserted بحطلك القيم الجديدة الماضفة او المعدلة، وال deleted بحطلك فيها القيم القديمة المعدلة او المحذوفة)**

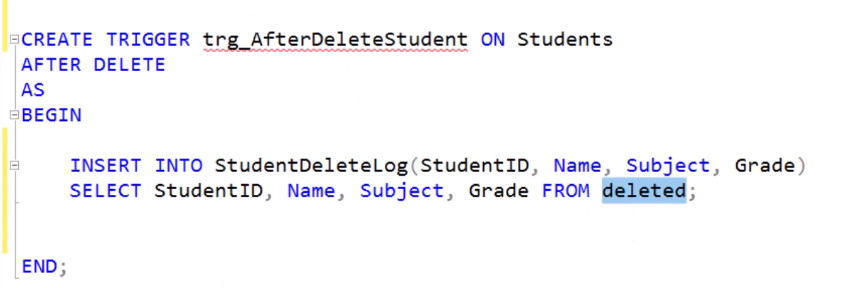


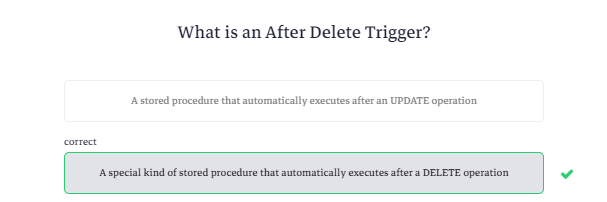




After Delete Trigger

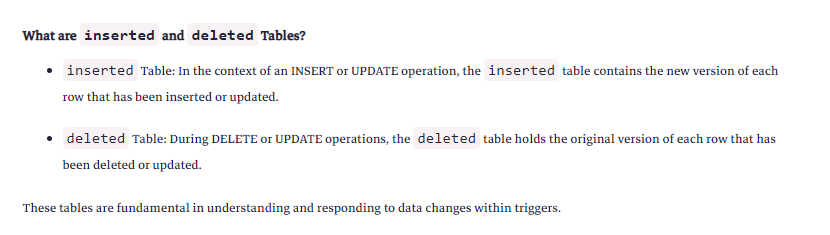
* يتم استدعائه بعد عملية ال delete على الجدول
* بدي اعمل log لل students اللي انعملهن delete، كما موضح بالشكل التالي:



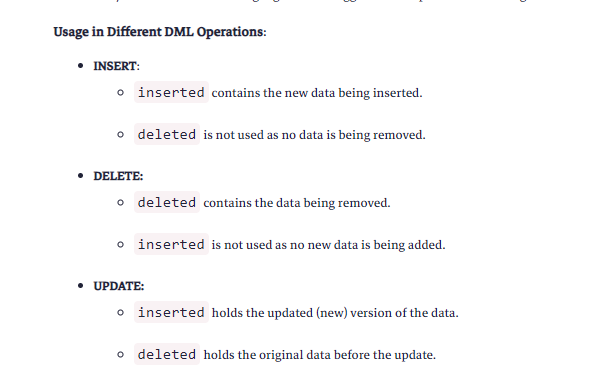


## Inserted and Deleted Tables in T-SQL Triggers







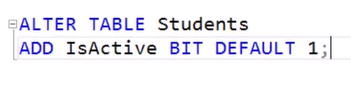


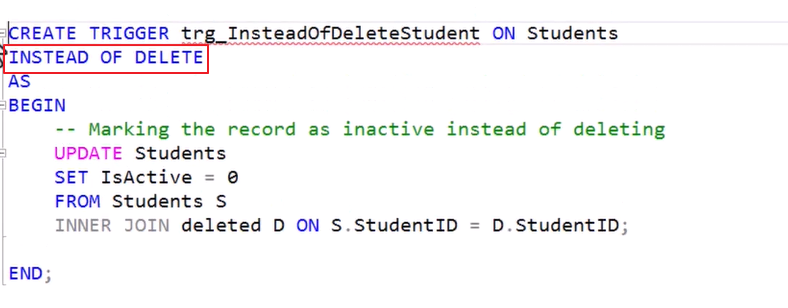
# Instead of Triggers

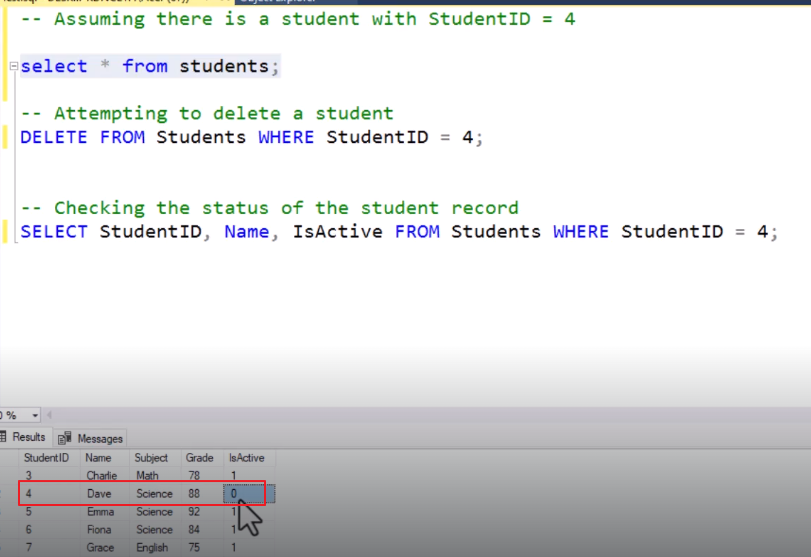
* فيك تعمل instead of insert – instead of update – instead of delete **لكن بدك تكون حذر جدا عند استخدامهن، لأنك انت هون عم تعمل override عليهن على مستوى الجدول نفسه**

Instead Of Delete Trigger

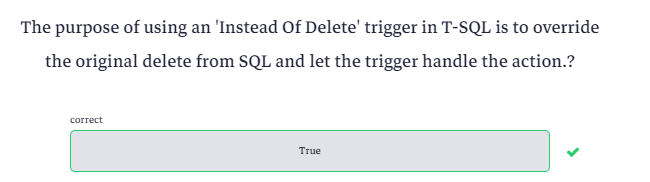
* انا قررت اني امنع عملية ال delete بال database عندي، وبدالها بدي خليه IsActive = 0 (بدي استبدل ال hard delete بال soft delete ) كما موضح بالشكل التالي:





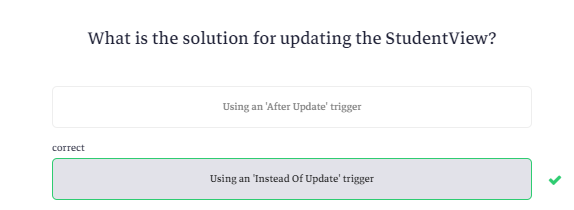


* **ملاحظة مهمة: instead of delete هي أقوى من ال after delete (يعني اذا كان مطبق عندي على نفس الجدول النوعين سوف يتم تنفيذ اللوجيك تبع ال instead of delete) وطبعا نفس الشي لل insert and update**



Instead Of Update Trigger

* **هون عمل update على view**

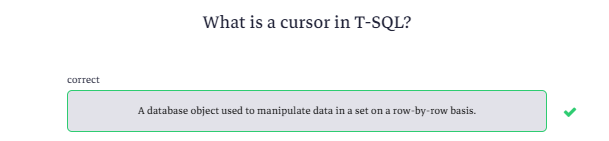


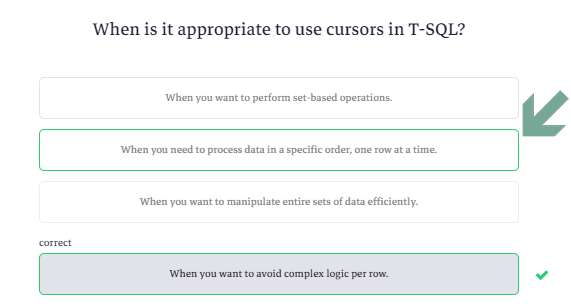
Instead Of Insert Trigger

* نفس الشي طبقنا ال insert على view تحتوي على جدولين، ومنها تفرعت عملية ال insert على الجدولين

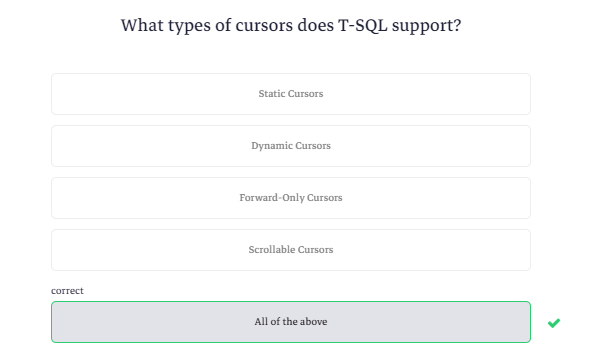
# Cursors in T-SQL

* **أول شي قلنا ياه الأستاذ لا تستخدم ال cursor إلا اذا كنت متطر عليه، لأنه يعتبر شي بطيء بالdatabase، ولانستخدمه الا بالحالات النادرة**
* ال cursor: هو object بال database بخليك تعمل iteration على ال data، يعني اذا بدي اعمل select \* from employees وبدي امشي عليه ب for loop (row by row)



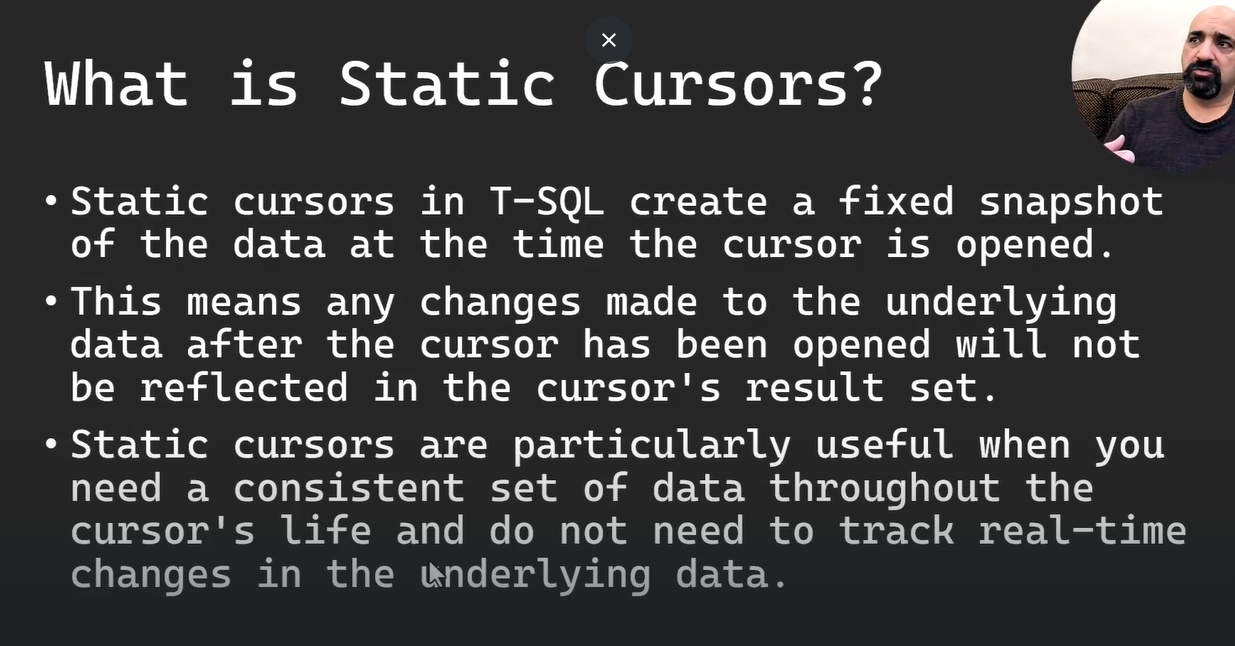


* يوجد أربعة أنواع من ال cursors، وكل نوع بقدملك ميزات مابقدملك ياها النوع التاني:

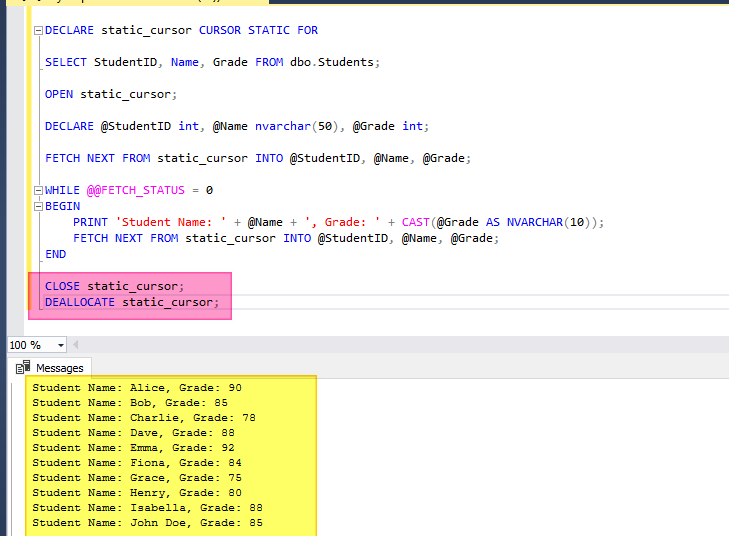


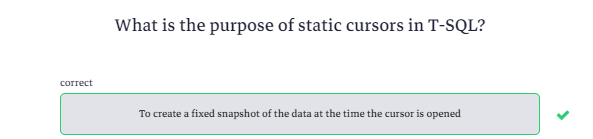
Static Cursors

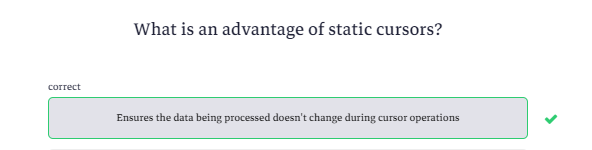
* بياخد snapshot (copy من ال data الأساسية وبيشتغل عليها offline) وبيفصلها عن الداتا الأساسية وبخرنها في ال memory، وأي تعديل على الداتا الأساسية مابشوفو (يعني اذا انضاف student على الجدول الأساسي وانت ماشي عليه loop مابشوفو)

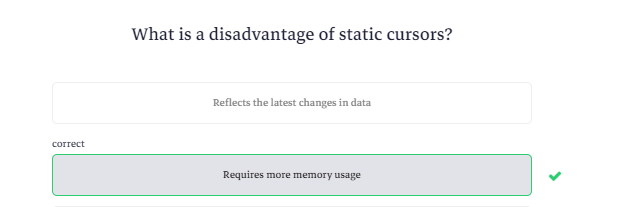


* هاد النوع بيستهلك ميموري بس مابيعملك locking للجدول



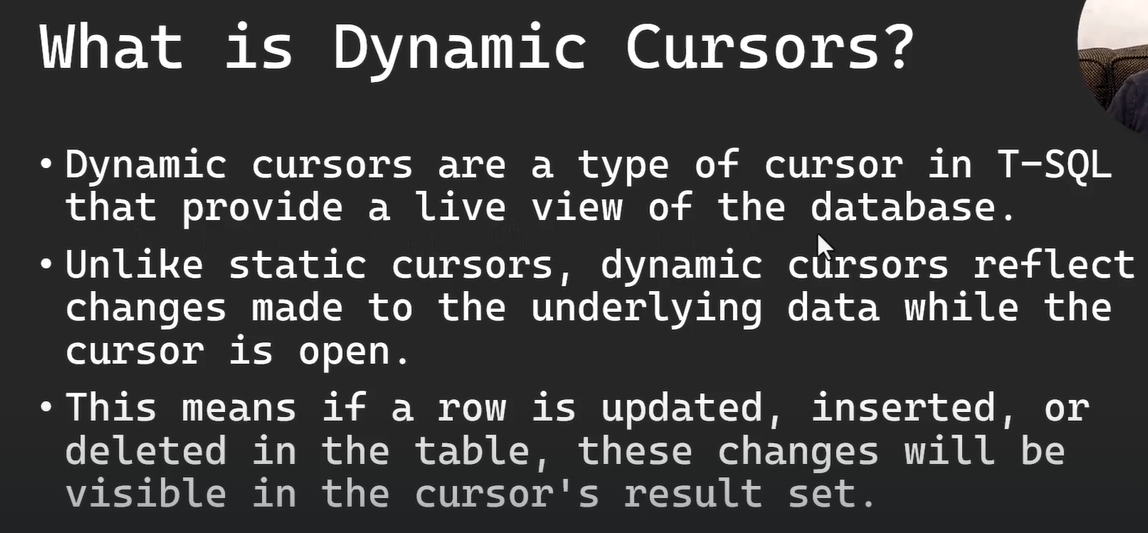


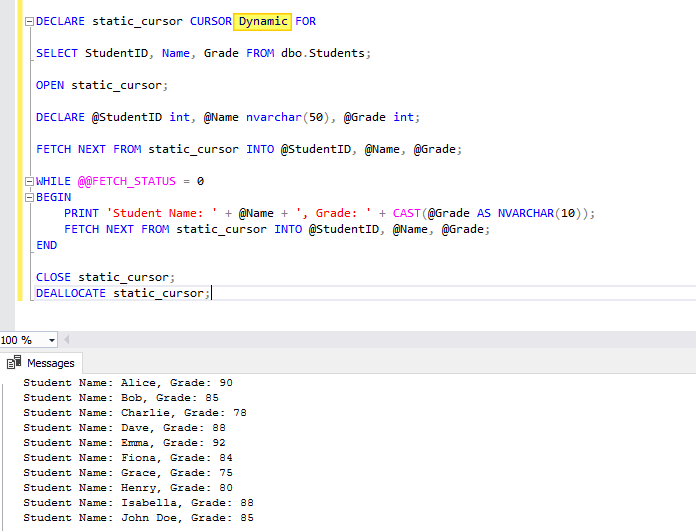


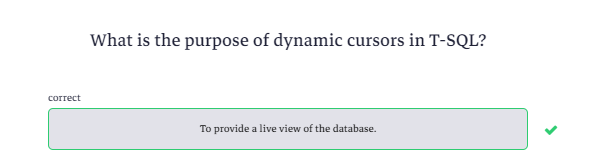


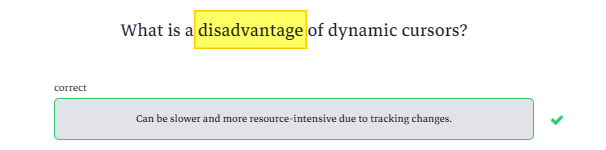
Dynamic Cursors

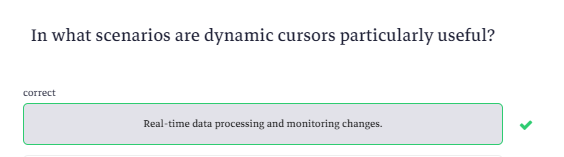
* نفس ال static cursor بالفكرة والتعريف (بتستخدم كلمة dynamic) **لكنه بضل فاتح ال student table مثلا وأي record يتم اضافته أو تعديله بينضاف على ال cursor (Real-Time)**
* ال performance أبطأ لأنه بضل فاتح connection مع ال data resource مشان تعمل Real-Life monitoring، وبيعمل locking لل data
* Static cursor بيعمل snapshot view بينما هاد بيعمل real life view

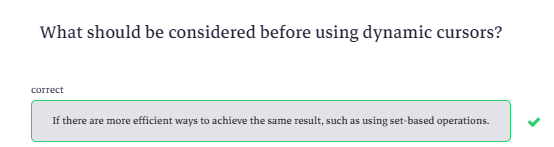






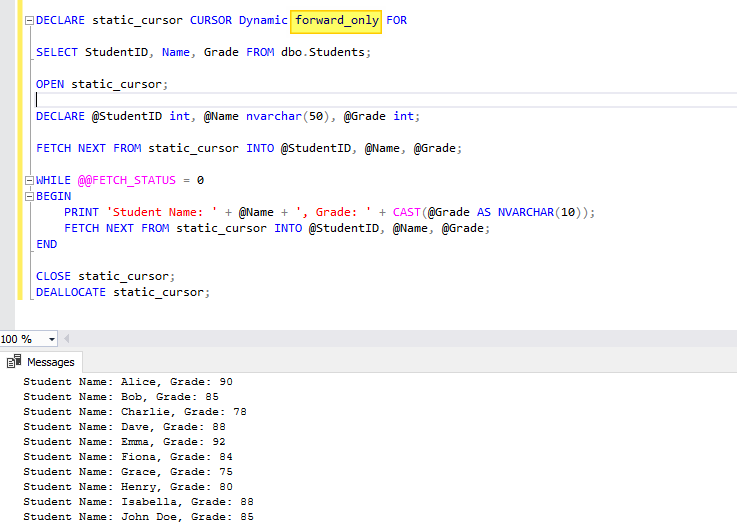


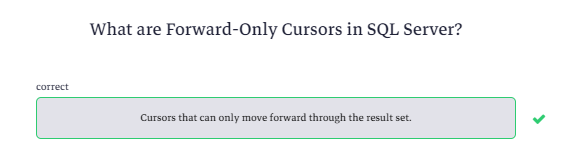




Forward-Only Cursors

* فيني افتح ال static or dynamic cursors as forward only، **يعني بخليك تمشي بالداتا record by record ومابتحسن ترجع فيه بشكل عكسي backward**
* يعتبر سريع لأنه بخليك تمشي بالداتا one-way
* يتم تعريفه (مع ال static and dynamic cursors ) كما موضح بالشكل التالي:

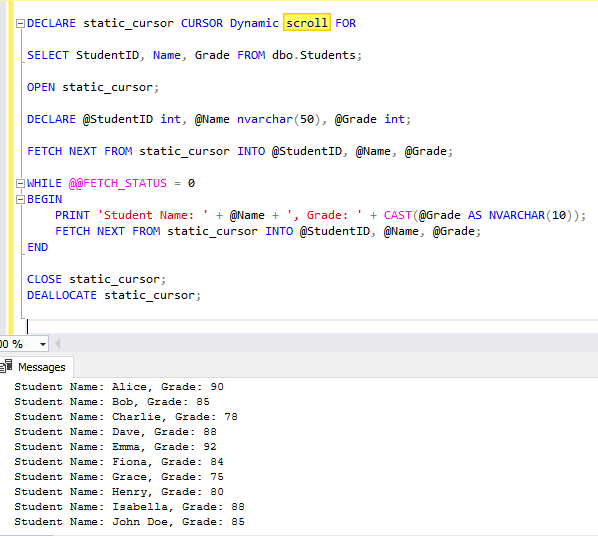


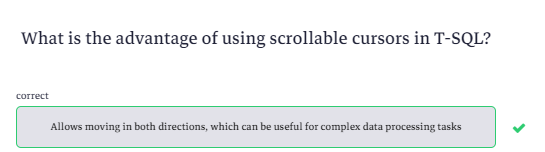


Scrollable Cursors

* هو forward and backward cursor، فبالتي هو أبطأ
* هون عندك fetch-next and fetch-previous(**fetch prior**)



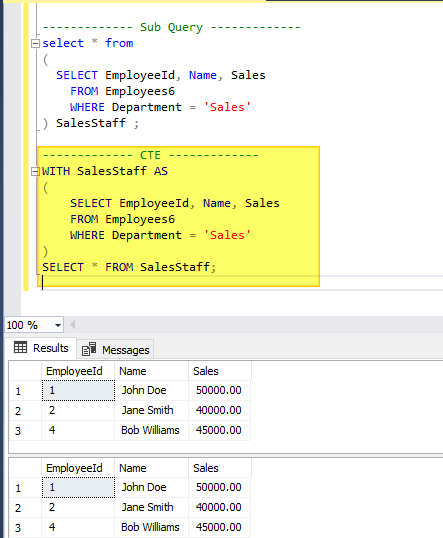




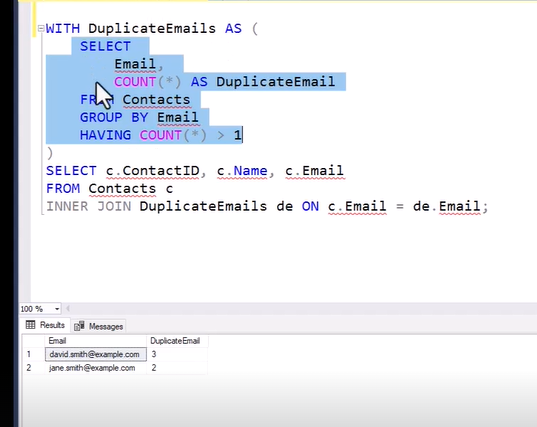
ملخص ال Cursors لاتستخدمها 😊 إلا بالحالات النادرة

# Common Table Expressions (CTEs)

* بتخلي ال queries تبعك مقروءة أكثر more readably، كما موضح بالشكل التالي:



* هاد مثال حلو عنها، أنا هون بدي طالع الإيميلات المكررة duplicate emails، وهاد الشي رح يتم على مرحلتين:



اول شي بجيب المكرر، بعدين بدمجه مع جدول ال contacrts، كمل موضح بالشكل التالي:

