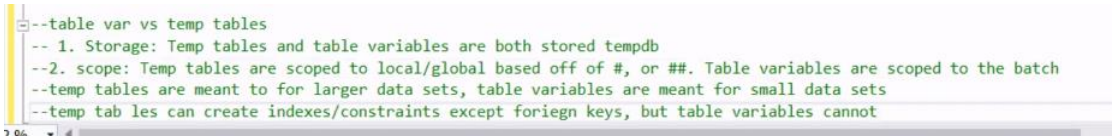
Day 4

temp tables

在 SQL Server 中，"@variable" 和 "#variable" 都是用于定义变量的语法。但是它们之间有一些重要的区别：

作用范围不同：
"@variable" 是一种局部变量，只能在定义它的批处理或存储过程中使用。
"#variable" 是一种临时表，在创建它的会话中的所有批处理和存储过程都可以访问它。
存在时间不同：
"@variable" 在定义它的批处理或存储过程的执行结束后就会被销毁。
"#variable" 在创建它的会话结束或者显式删除之前都会一直存在。

##variable" 是一种用于定义全局临时表的语法



Define functions

main purpose is for calculation

```
Go
Create Function GetTotalRevenue( @price money, @discount real, @quantity smallint)
Returns Money
as
Begin
    Declare @Revenue money
    set @Revenue = @price * @quantity * (1-@discount)
    return @revenue
End
Go
Select *
from [Order Details]

select UnitPrice, Discount, Quantity, dbo.GetTotalRevenue(UnitPrice, Discount, Quantity) "Total Revenue"
From [Order Details]
Go
Create Function ExpensiveProduct(@threshold money)
Returns Table
as
return Select * From Products where UnitPrice > @threshold
go
Select * From dbo.ExpensiveProduct(100)
```

Define procedure

```
--return vs out
--Both are used to receive a value out
--return can only give back 1 integer while out can be used for multiple values of different data types
```

```
CREATE PROCEDURE myProcedure
AS
BEGIN
  SELECT employeeId, FirstName, LastName
  FROM Employees
  WHERE LastName LIKE 'A%';
END;

EXEC myProcedure;
```

```
--SP and Functions differences:
--Usage: sp is for DML statements while functions are mostly used for calculations
--calling: sp uses execute/exec, while functions require a query as well as input parameters
--output: SP may or may not require any return or output, but functions must return something
--"Returns and return"
--SP can call functions but functions cant call SP.

--Pagination: divides a large dataset into smaller discrete pages
--
```

Properties of Transactions

```
--Properties of Transactions
--ACID:
--Atomicty:
--Consistancy:
--Isolation:
--Durability:
```

```
Begin Transaction


Select * from School
Drop Table School


Rollback
```

isolation level and problem

Isolation Level	Dirty Read	Non-Repeatable Read	Phantom Read
Read Uncommitted	Yes	Yes	Yes
Read Committed	No	Yes	Yes
Repeatable Read	No	No	Yes
Serializable	No	No	No

performance tuning

```
--performance tuning
--1. look at the execution plan
--2. choose index wisely
--3. avoid unnecessary joins
--4. avoid SELECT *
--5. JOIN to replace subquery
--6. derived table to avoid a lot of grouping by
```