Database Design, Implementation & Maintenance

using ERwin, SQL Server, Visual Studio and Visual Studio Team Services

Hans-Petter Halvorsen, M.Sc.

Necessary Steps

1 Database Modelling

ERwin

- Create ER Diagram
- Generate SQL Table Script

2 Database Implementation

Microsoft® SQL Server®

SQL

- Execute Table Script
- Create Views, Stored Procedures and Triggers

3 Database Communication

Visual Studio

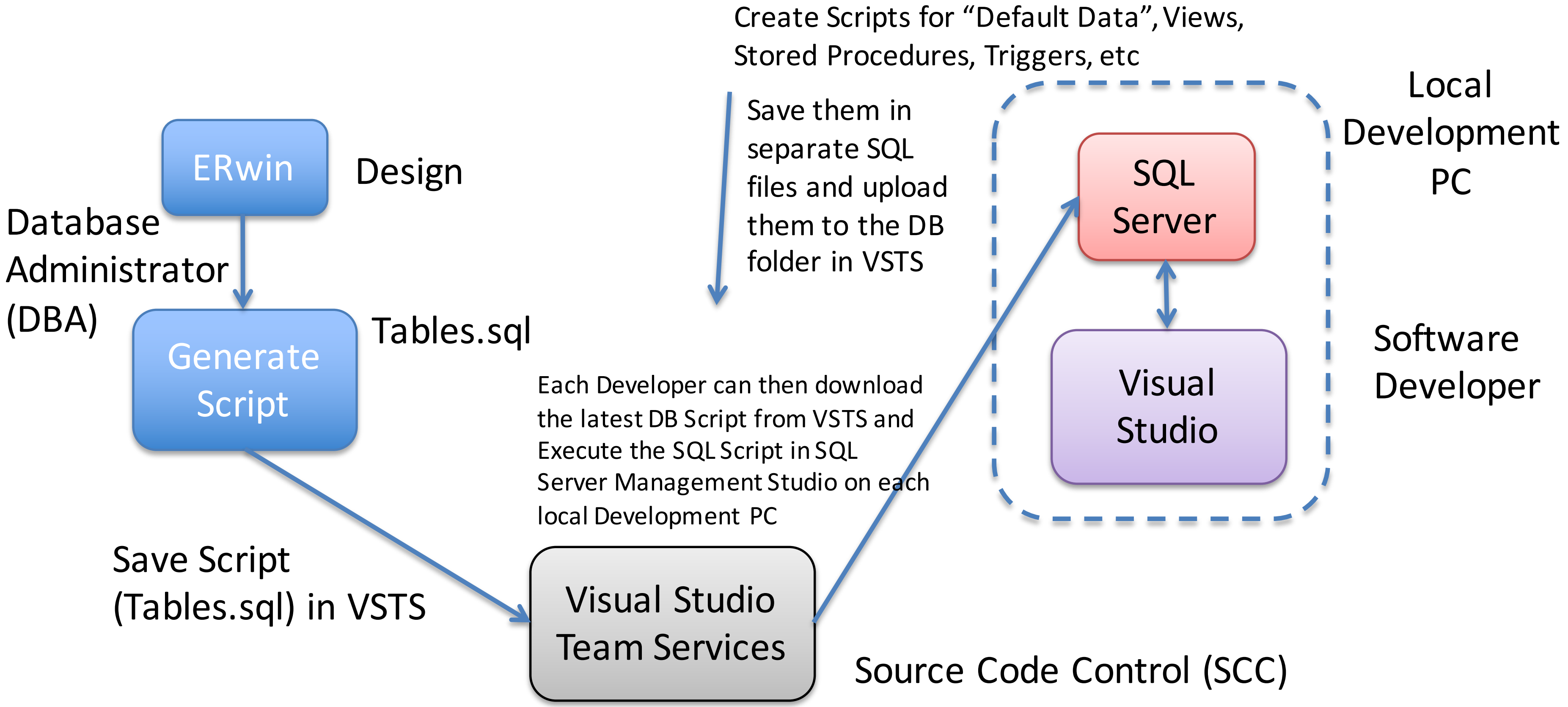
ADO.NET **SQL**

- Establish Connection between SQL Server Database and C#
- SELECT, INSERT UPDATE between your GUI and SQL Server

Update and Improve

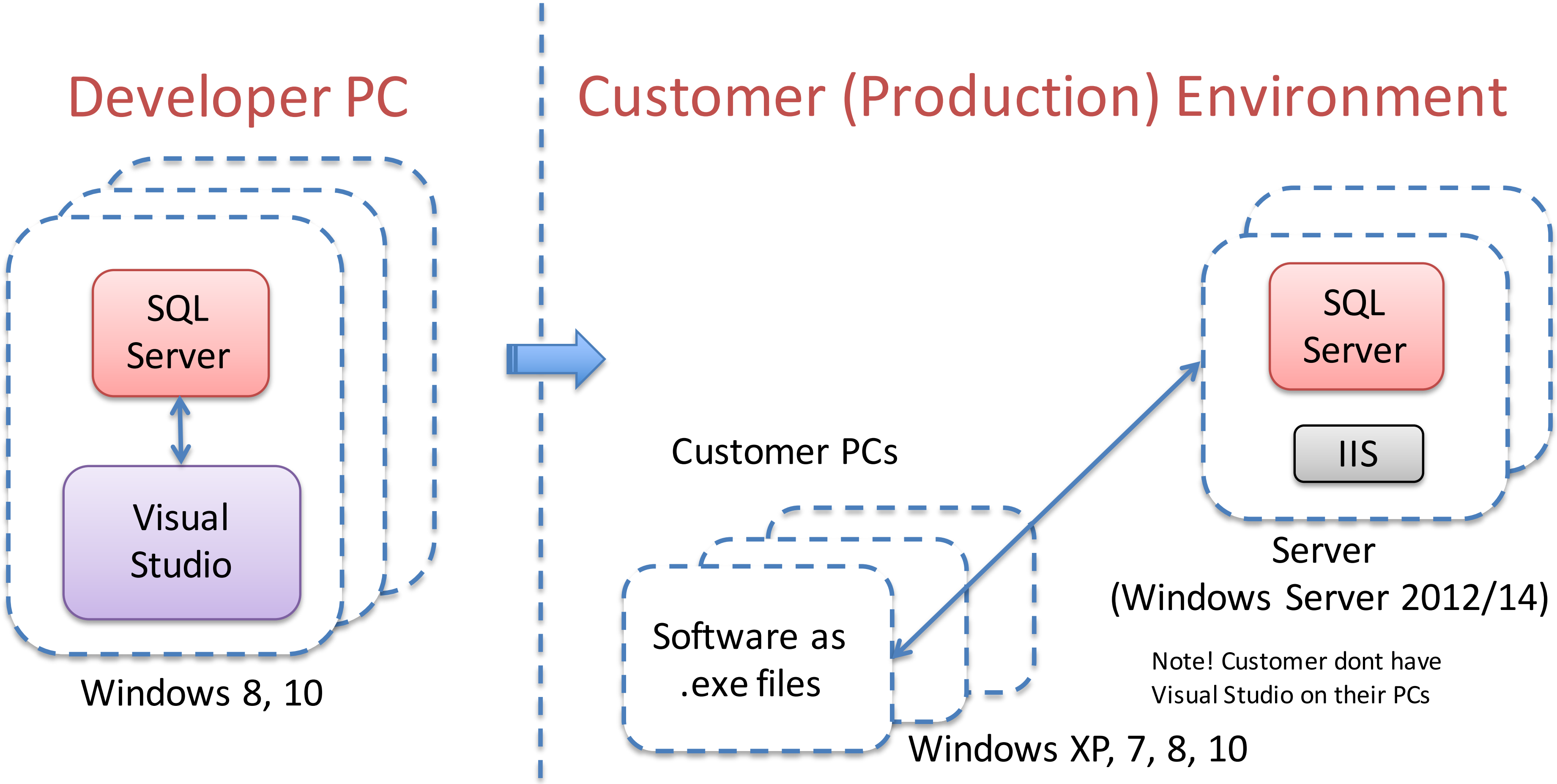
Update and Improve

Create a Table Script


ERwin
Design
Database Administrator (DBA)
Generate Script
Tables.sql
Save Script (Tables.sql) in VSTS
Visual Studio Team Services
Source Code Control (SCC)
Create Scripts for “Default Data”, Views, Stored Procedures, Triggers, etc
Save them in separate SQL files and upload them to the DB folder in VSTS
Each Developer can then download the latest DB Script from VSTS and Execute the SQL Script in SQL Server Management Studio on each local Development PC
SQL Server
Visual Studio
Local Development PC
Software Developer

The DBA is in charge of maintening the DB Script that can be used on the Developer PCs and later deployed in the Customer Environment

Developer Environment vs. Production Environment


Developer PC
SQL Server
Visual Studio
Windows 8, 10
Customer (Production) Environment
Customer PCs
Software as .exe files
Windows XP, 7, 8, 10
SQL Server
IIS
Server (Windows Server 2012/14)
Note! Customer dont have Visual Studio on their PCs

ERwin

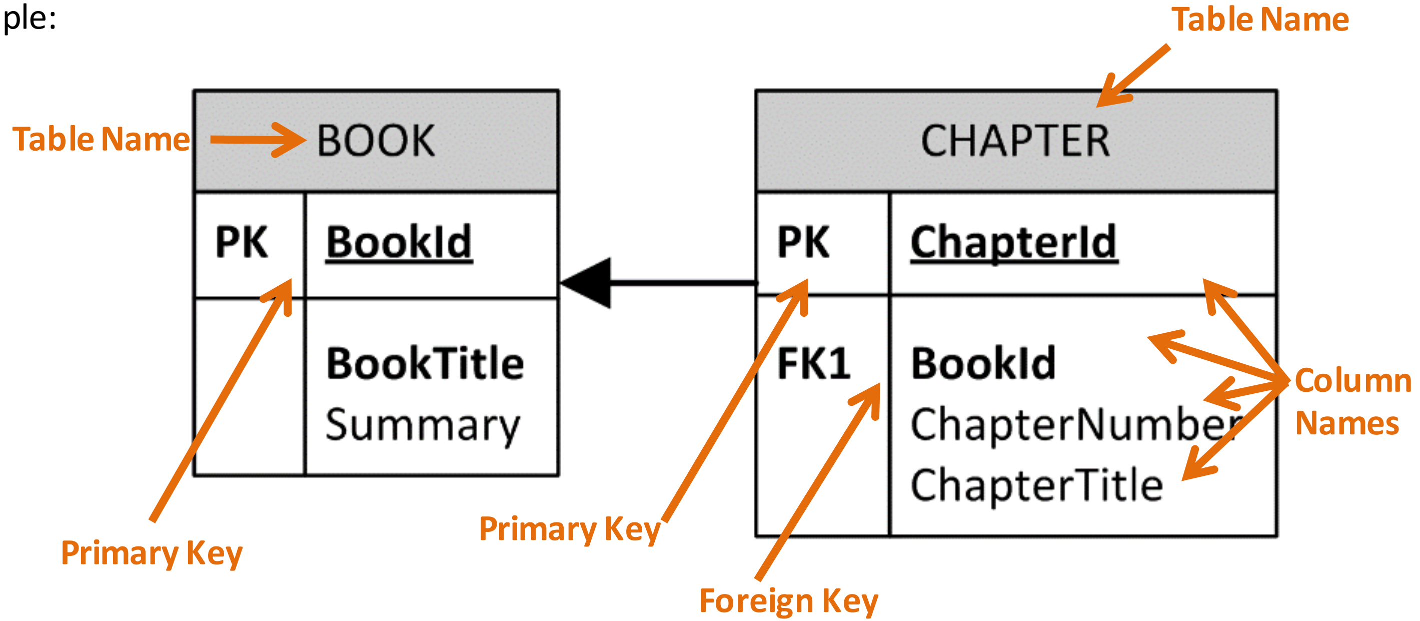
Hans-Petter Halvorsen, M.Sc.

Database Design – ER Diagram

ER Diagram (Entity-Relationship Diagram)

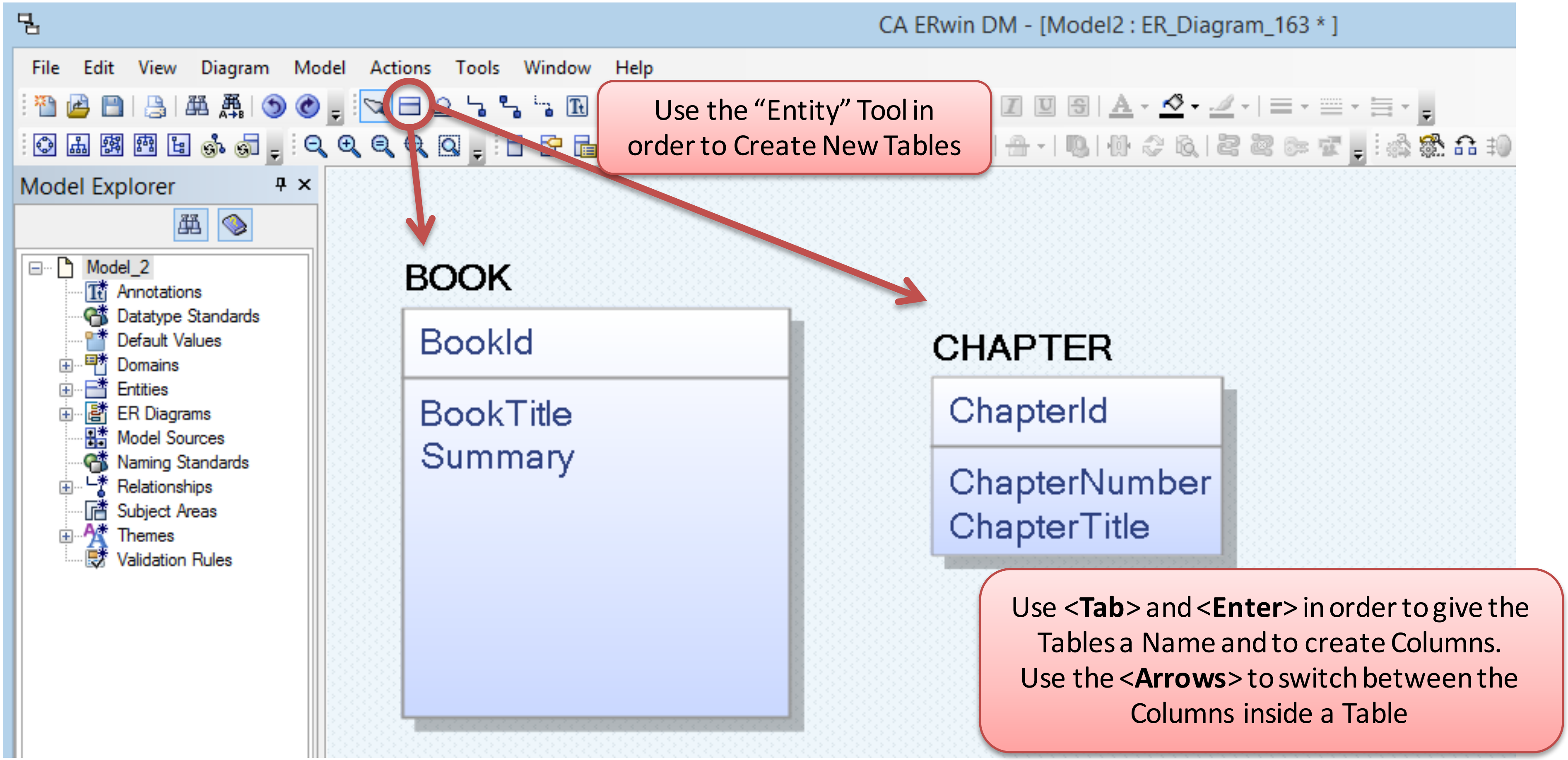
- Used for Design and Modeling of Databases.
- Specify Tables and **relationship** between them (**Primary Keys** and **Foreign Keys**)

Example:



Relational Database. In a relational database all the tables have one or more relation with each other using Primary Keys (PK) and Foreign Keys (FK). Note! You can only have one PK in a table, but you may have several FK's.

Create Tables, Columns & Data Types

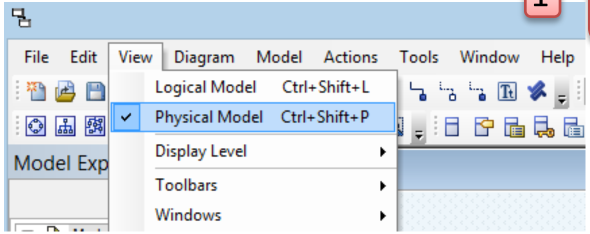


Use the “Entity” Tool in order to Create New Tables

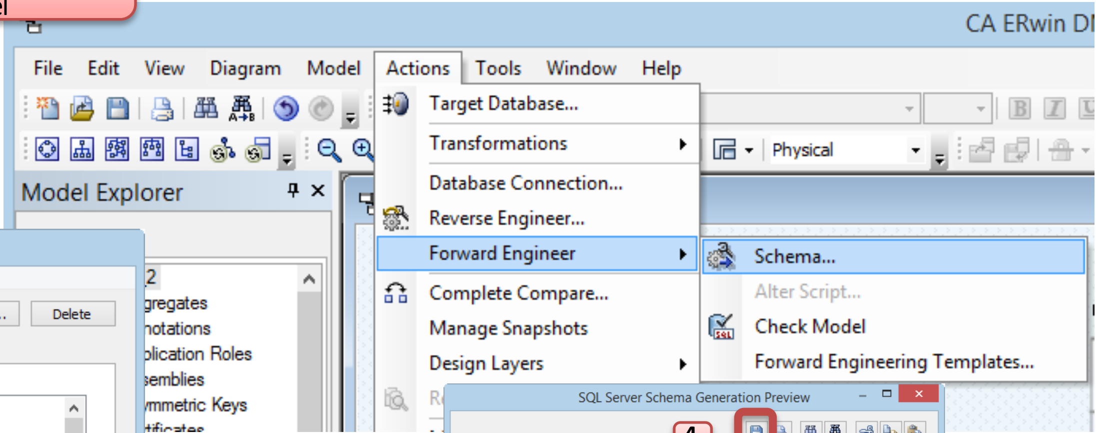
Use <**Tab**> and <**Enter**> in order to give the Tables a Name and to create Columns. Use the <**Arrows**> to switch between the Columns inside a Table

Create SQL Script

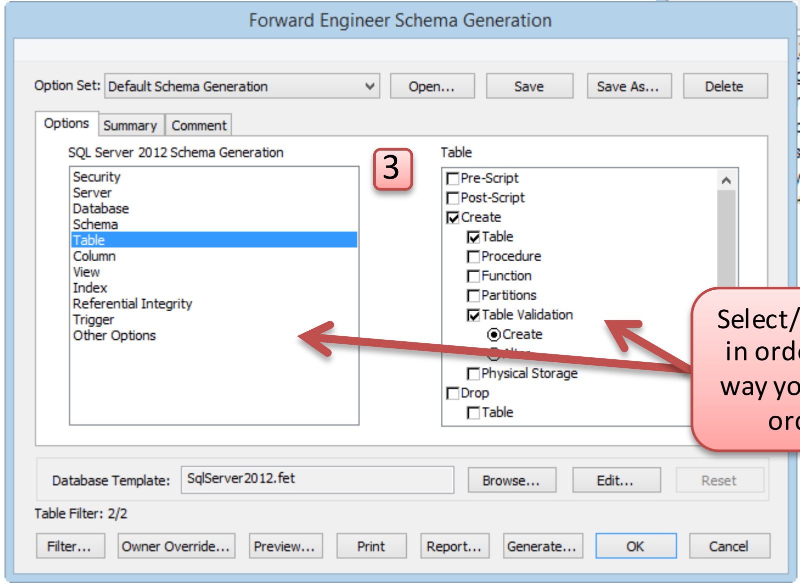
1 Make sure you are using the Physical Model



2 Select "Forward Engineering" and "Schema…"



3



Option Set: Default Schema Generation

SQL Server 2012 Schema Generation

- Security
- Server
- Database
- Schema
- Table
- Column
- View
- Index
- Referential Integrity
- Trigger
- Other Options

Table

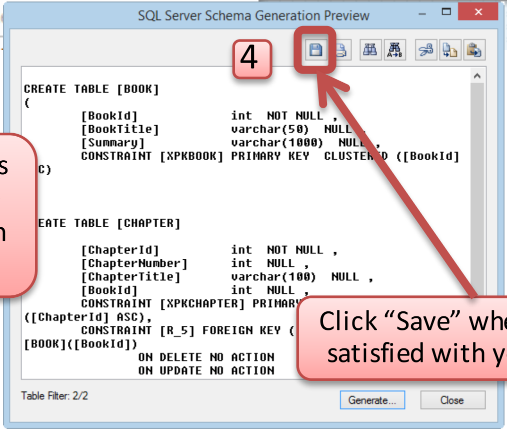
- ☐ Pre-Script
- ☐ Post-Script
- ☑ Create
  - ☑ Table
  - ☐ Procedure
  - ☐ Function
  - ☐ Partitions
  - ☑ Table Validation
    - ● Create
  - ☐ Physical Storage
- ☐ Drop
  - ☐ Table

Database Template: SqlServer2012.fet

Table Filter: 2/2

Select/Deselect different Options in order to make your script the way you want. Click "Preview" in order to see the results.

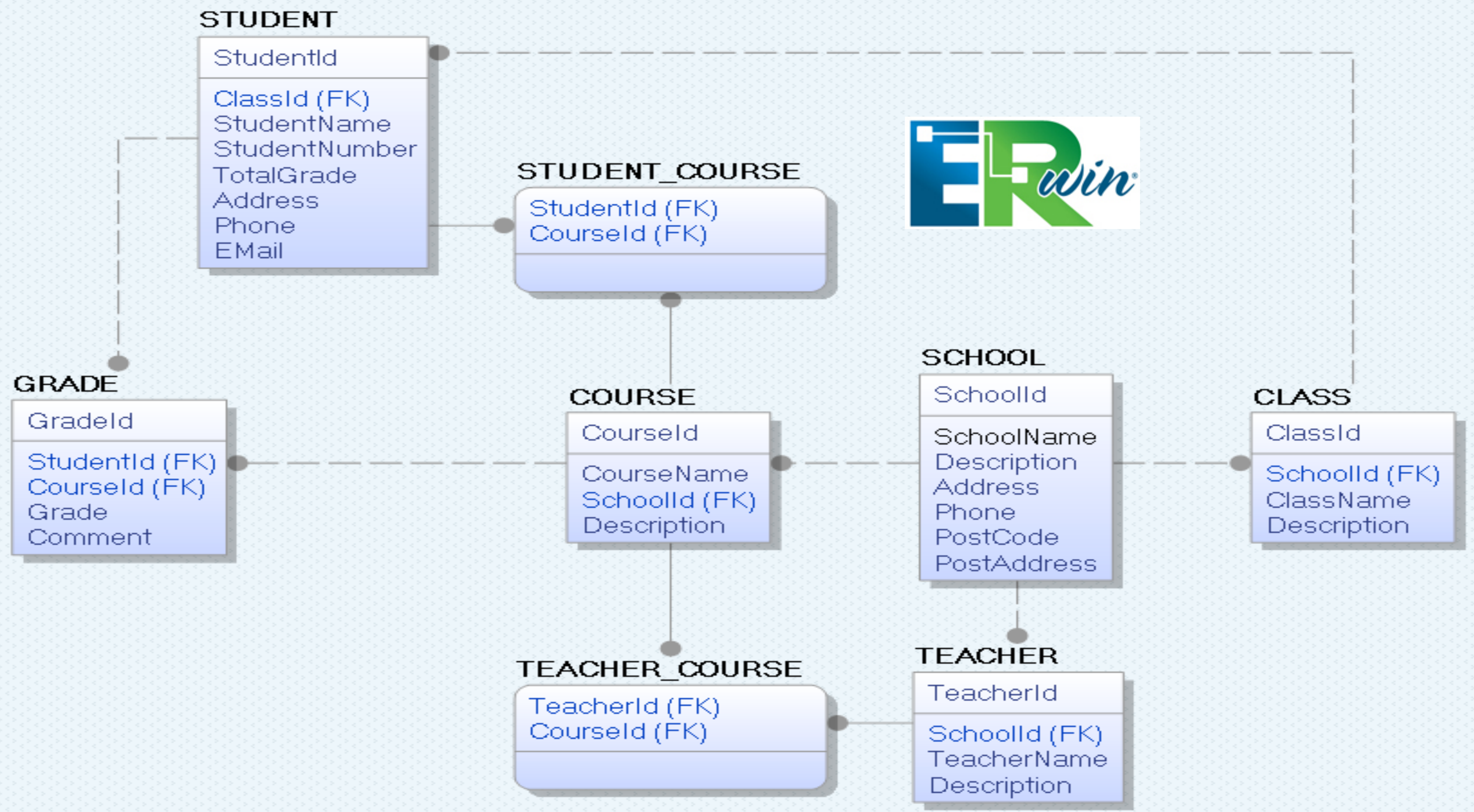
4



```
CREATE TABLE [BOOK]
(
        [BookId]             int  NOT NULL ,
        [BookTitle]          varchar(50)  NULL
        [Summary]            varchar(1000)  NUL  ,
        CONSTRAINT [XPKBOOK] PRIMARY KEY  CLUSTER D ([BookId]
 C)



 EATE TABLE [CHAPTER]

        [ChapterId]          int  NOT NULL ,
        [ChapterNumber]      int  NULL ,
        [ChapterTitle]       varchar(100)  NULL ,
        [BookId]             int  NULL ,
        CONSTRAINT [XPKCHAPTER] PRIMARY
([ChapterId] ASC),
        CONSTRAINT [R_5] FOREIGN KEY (
[BOOK]([BookId])
                ON DELETE NO ACTION
                ON UPDATE NO ACTION
```

Table Filter: 2/2

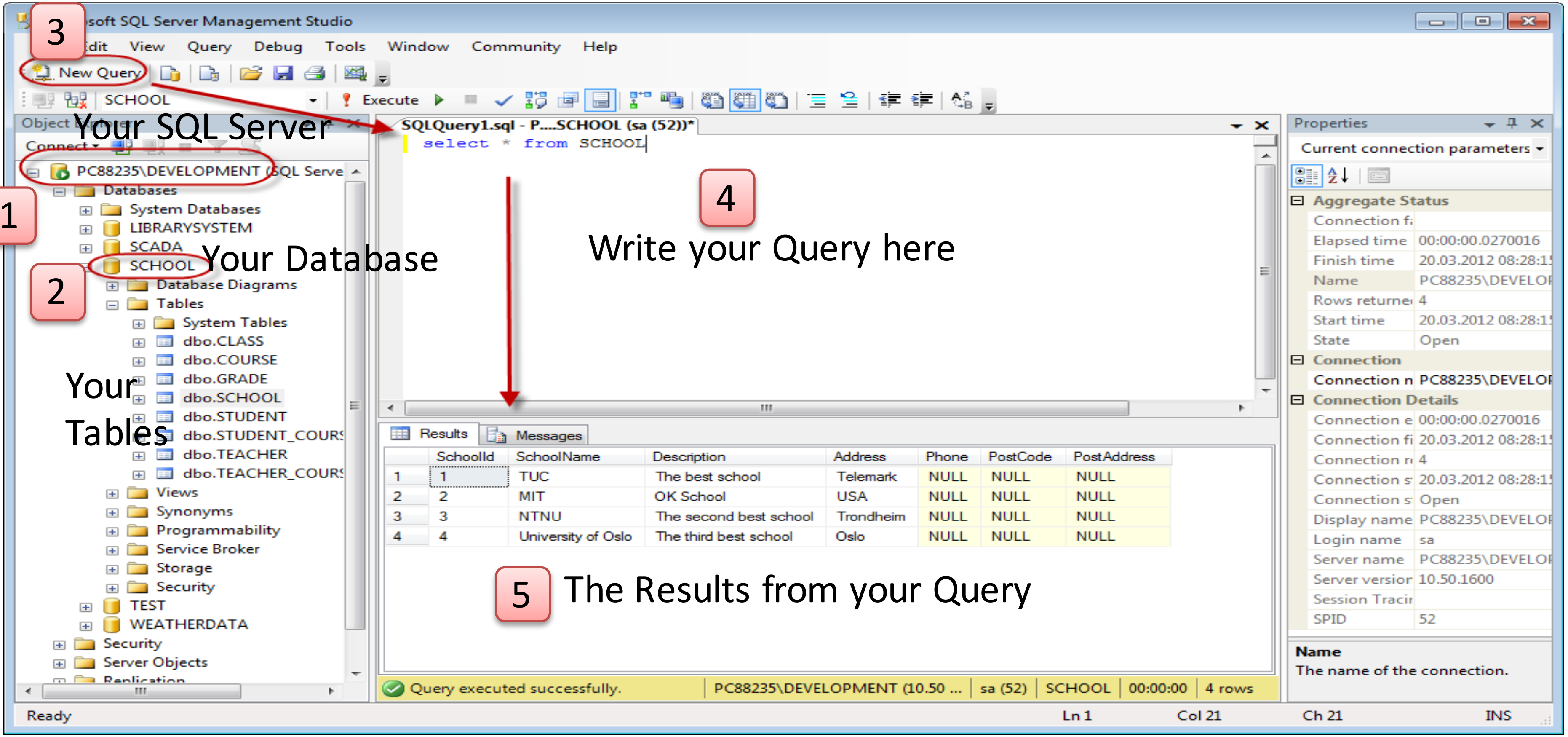
Click "Save" when you are satisfied with your Script


STUDENT
StudentId
ClassId (FK)
StudentName
StudentNumber
TotalGrade
Address
Phone
EMail
STUDENT_COURSE
StudentId (FK)
CourseId (FK)
GRADE
GradeId
StudentId (FK)
CourseId (FK)
Grade
Comment
COURSE
CourseId
CourseName
SchoolId (FK)
Description
SCHOOL
SchoolId
SchoolName
Description
Address
Phone
PostCode
PostAddress
CLASS
ClassId
SchoolId (FK)
ClassName
Description
TEACHER_COURSE
TeacherId (FK)
CourseId (FK)
TEACHER
TeacherId
SchoolId (FK)
TeacherName
Description

SQL Server

Hans-Petter Halvorsen, M.Sc.

Microsoft SQL Server



```
select * from SCHOOL
```

	SchoolId	SchoolName	Description	Address	Phone	PostCode	PostAddress
1	1	TUC	The best school	Telemark	NULL	NULL	NULL
2	2	MIT	OK School	USA	NULL	NULL	NULL
3	3	NTNU	The second best school	Trondheim	NULL	NULL	NULL
4	4	University of Oslo	The third best school	Oslo	NULL	NULL	NULL

Query executed successfully. | PC88235\DEVELOPMENT (10.50 ... | sa (52) | SCHOOL | 00:00:00 | 4 rows

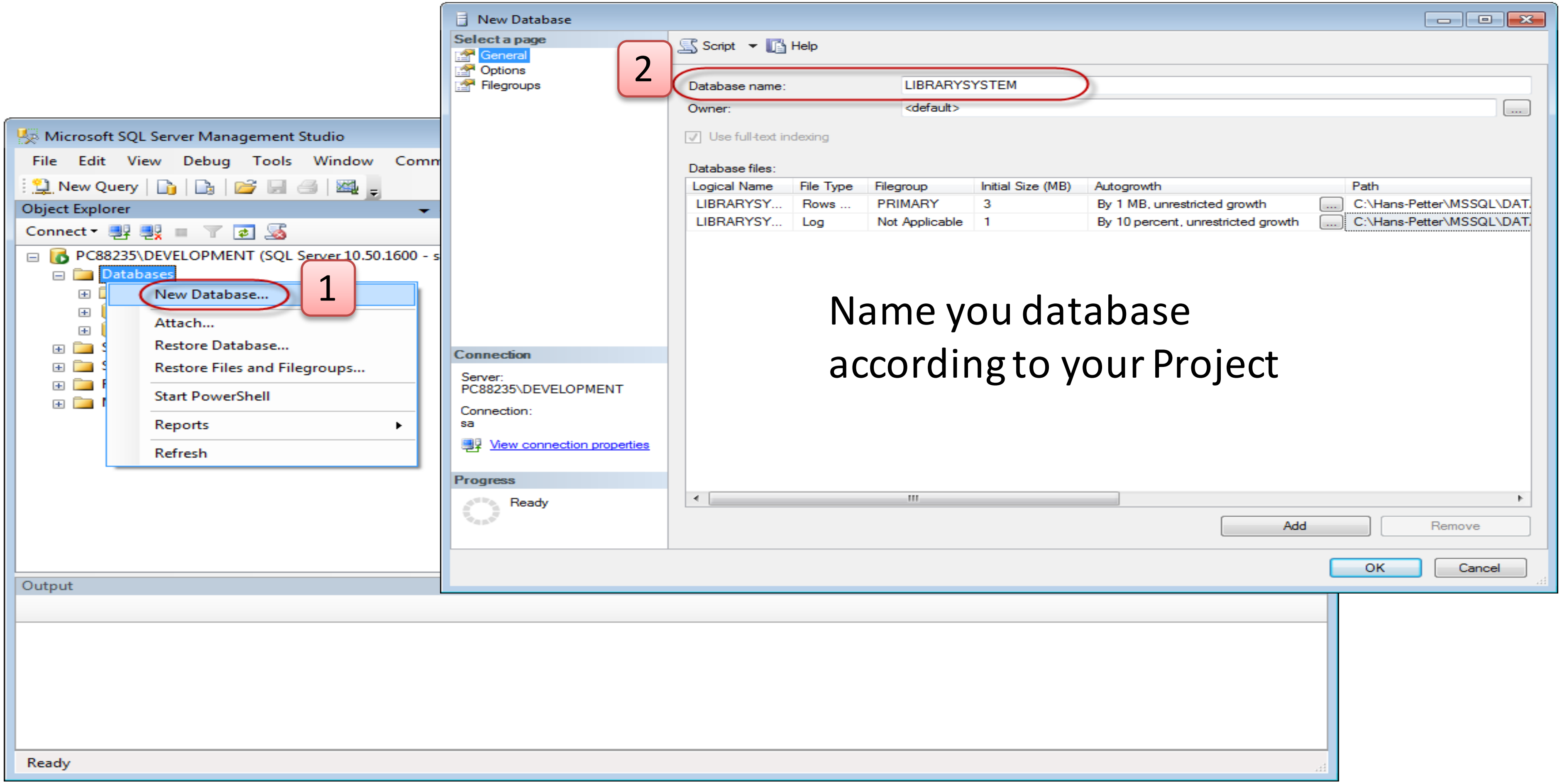
Properties — Current connection parameters

Aggregate Status	
Connection fi	
Elapsed time	00:00:00.0270016
Finish time	20.03.2012 08:28:1
Name	PC88235\DEVELOI
Rows returne	4
Start time	20.03.2012 08:28:1
State	Open
Connection	
Connection n	PC88235\DEVELOI
Connection Details	
Connection e	00:00:00.0270016
Connection fi	20.03.2012 08:28:1
Connection r	4
Connection s	20.03.2012 08:28:1
Connection s	Open
Display name	PC88235\DEVELOI
Login name	sa
Server name	PC88235\DEVELOI
Server versior	10.50.1600
Session Tracir	
SPID	52

Name
The name of the connection.

Ready   Ln 1   Col 21   Ch 21   INS

Microsoft SQL Server – Create a New Database



Database name: LIBRARYSYSTEM

Owner: <default>

Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth	Path
LIBRARYSY...	Rows ...	PRIMARY	3	By 1 MB, unrestricted growth	C:\Hans-Petter\MSSQL\DAT
LIBRARYSY...	Log	Not Applicable	1	By 10 percent, unrestricted growth	C:\Hans-Petter\MSSQL\DAT

Server:
PC88235\DEVELOPMENT

Connection:
sa

View connection properties

Name you database according to your Project

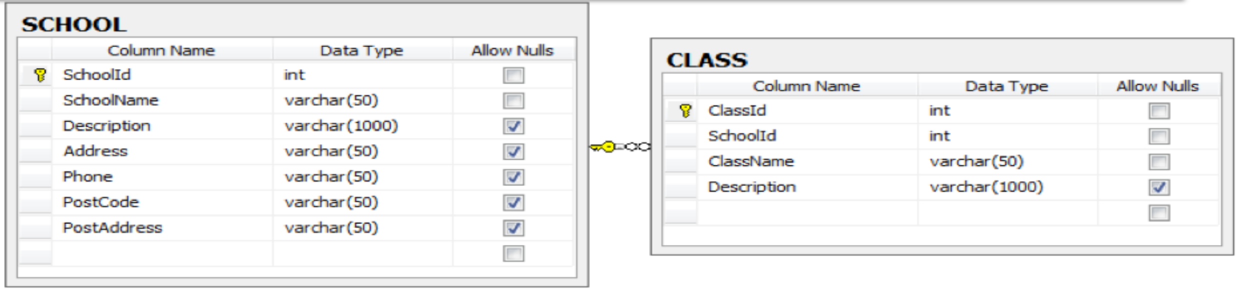
SQL Script Example

Create Tables using SQL

```
if not exists (select * from dbo.sysobjects where id = object_id(N'[SCHOOL]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
CREATE TABLE [SCHOOL]
(
            [SchoolId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
            [SchoolName] [varchar](50) NOT NULL UNIQUE,
            [Description] [varchar](1000) NULL,
            [Address] [varchar](50) NULL,
            [Phone] [varchar](50) NULL,
            [PostCode] [varchar](50) NULL,
            [PostAddress] [varchar](50) NULL,
)
GO
```

```
if not exists (select * from dbo.sysobjects where id = object_id(N'[CLASS]') and OBJECTPROPERTY(id, N'IsUserTab
CREATE TABLE [CLASS]
(
            [ClassId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
            [SchoolId] [int] NOT NULL FOREIGN KEY REFERENCES [SCHOOL] ([SchoolId]),
            [ClassName] [varchar](50) NOT NULL,
            [Description] [varchar](1000) NULL,
)
GO
```

Create them using the Query Editor in SQL Server (based on the Script generated from ERwin)



SCHOOL

	Column Name	Data Type	Allow Nulls
🔑	SchoolId	int	☐
	SchoolName	varchar(50)	☐
	Description	varchar(1000)	✓
	Address	varchar(50)	✓
	Phone	varchar(50)	✓
	PostCode	varchar(50)	✓
	PostAddress	varchar(50)	✓
			☐

CLASS

	Column Name	Data Type	Allow Nulls
🔑	ClassId	int	☐
	SchoolId	int	☐
	ClassName	varchar(50)	☐
	Description	varchar(1000)	✓
			☐

```
if not exists (select * from dbo.sysobjects where id = object_id(N'[CUSTOMER]') and OBJECTPROPERTY(id,
N'IsUserTable') = 1)
CREATE TABLE CUSTOMER
(
      CustomerId int PRIMARY KEY,
      CustomerNumber int NOT NULL UNIQUE,
      LastName varchar(50) NOT NULL,
      FirstName varchar(50) NOT NULL,
      AreaCode int NULL,
      Address varchar(50) NULL,
      Phone varchar(50) NULL,
)
GO

if exists(select * from dbo.syscolumns where id = object_id(N'[CUSTOMER]') and OBJECTPROPERTY(id,
N'IsUserTable') = 1 and name = 'CustomerId')
ALTER TABLE CUSTOMER ALTER COLUMN CustomerId int
Else
ALTER TABLE CUSTOMER ADD CustomerId int
GO

if exists(select * from dbo.syscolumns where id = object_id(N'[CUSTOMER]') and OBJECTPROPERTY(id,
N'IsUserTable') = 1 and name = 'CustomerNumber')
ALTER TABLE CUSTOMER ALTER COLUMN CustomerNumber int
Else
ALTER TABLE CUSTOMER ADD CustomerNumber int
GO
...
```

SQL Script Example that has been generated with ERwin but has been modified in SQL Server Management Studio for more robustness. The Script handles that tables may already exist, etc.

Creating Views, Stored Procedures and Data Scripts

Hans-Petter Halvorsen, M.Sc.

Views

Hans-Petter Halvorsen, M.Sc.

Creating Views using SQL code

1

Create View:

```
IF EXISTS (SELECT name
        FROM   sysobjects
        WHERE  name = 'CourseData'
        AND      type = 'V')
     DROP VIEW CourseData
GO

CREATE VIEW CourseData
AS

SELECT
SCHOOL.SchoolId,
SCHOOL.SchoolName,
COURSE.CourseId,
COURSE.CourseName,
COURSE.Description

FROM
SCHOOL
INNER JOIN COURSE ON SCHOOL.SchoolId = COURSE.SchoolId
GO
```

A View is a "virtual" table that can contain data from multiple tables

This part is not necessary – but if you make any changes, you need to delete the old version before you can update it

The Name of the View

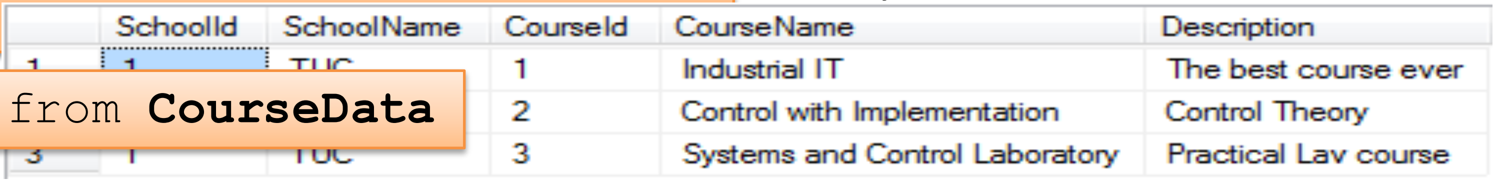
Inside the View you join the different tables together using the **JOIN** operator

You can Use the View as an ordinary table in Queries:

Using the View:

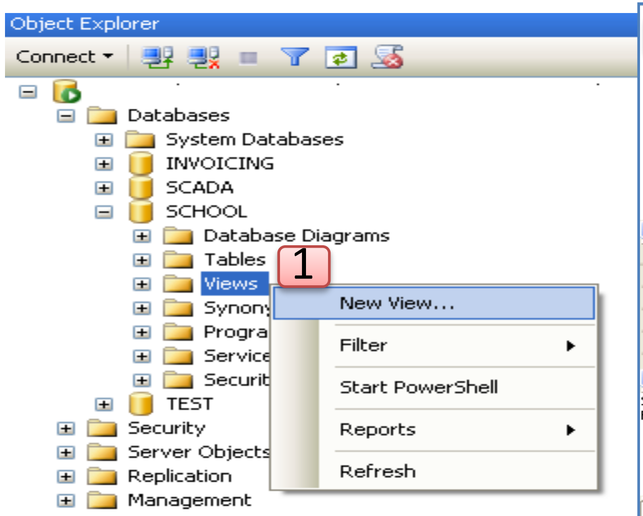
2

```
select * from CourseData
```



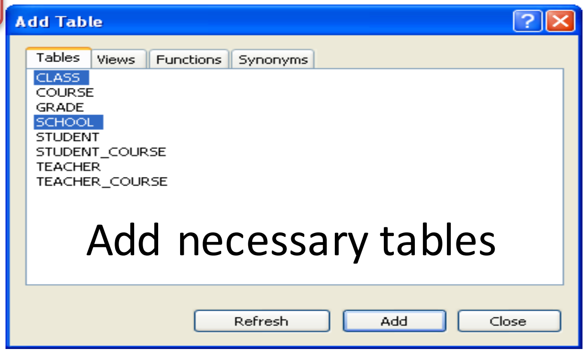
	SchoolId	SchoolName	CourseId	CourseName	Description
1	1	TUC	1	Industrial IT	The best course ever
			2	Control with Implementation	Control Theory
3	1	TUC	3	Systems and Control Laboratory	Practical Lav course

Creating Views using the Editor



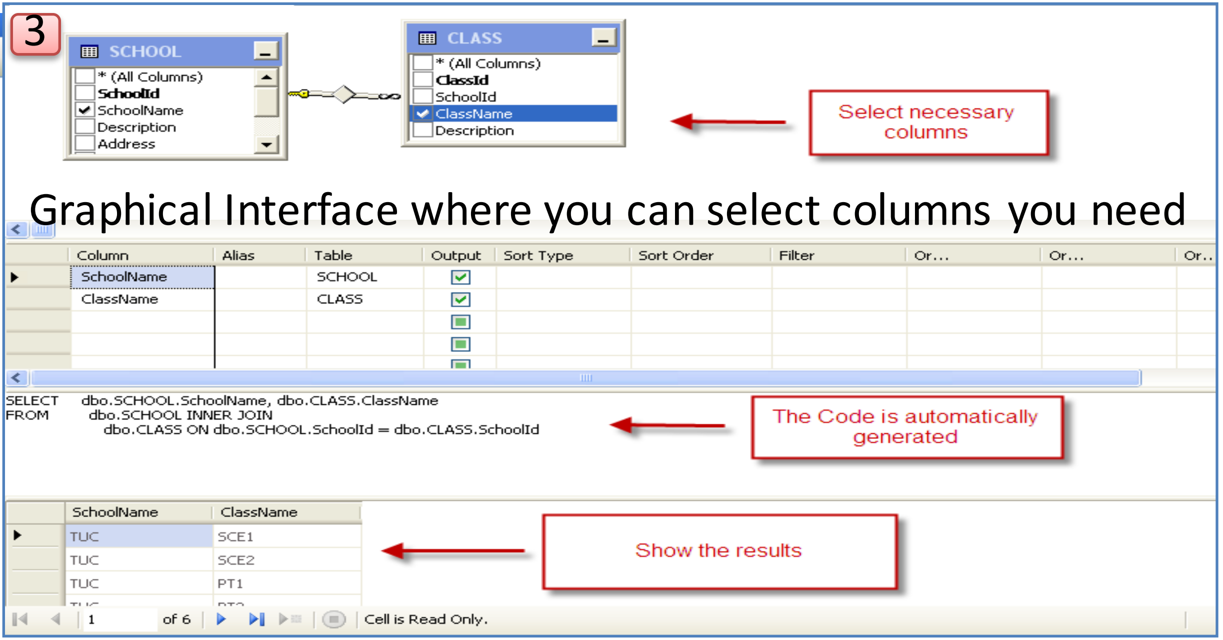
1

2



Add necessary tables

3



Select necessary columns

Graphical Interface where you can select columns you need

Column	Alias	Table	Output	Sort Type	Sort Order	Filter	Or...	Or...	Or...
SchoolName		SCHOOL	✓						
ClassName		CLASS	✓						

```
SELECT    dbo.SCHOOL.SchoolName, dbo.CLASS.ClassName
FROM       dbo.SCHOOL INNER JOIN
              dbo.CLASS ON dbo.SCHOOL.SchoolId = dbo.CLASS.SchoolId
```

The Code is automatically generated

SchoolName	ClassName
TUC	SCE1
TUC	SCE2
TUC	PT1

1 of 6 Cell is Read Only.

Show the results

4

Copy the SQL Code and Create a New Script in the Management Studio

Stored Procedures

Hans-Petter Halvorsen, M.Sc.

Stored Procedure

1 Create Stored Procedure:

```
IF EXISTS (SELECT name
           FROM   sysobjects
           WHERE  name = 'StudentGrade'
           AND    type = 'P')
         DROP PROCEDURE StudentGrade
OG

CREATE PROCEDURE StudentGrade
@Student varchar(50),
@Course varchar(10),
@Grade varchar(1)

AS

DECLARE
@StudentId int,
@CourseId int

select @StudentId = StudentId from STUDENT where StudentName = @Student

select @CourseId = CourseId from COURSE where CourseName = @Course

insert into GRADE (StudentId, CourseId, Grade)
values (@StudentId, @CourseId, @Grade)
GO
```

A Stored Procedure is like a Method in C# - it is a piece of code with SQL commands that do a specific task – and you reuse it

This part is not necessary – but if you make any changes, you need to delete the old version before you can update it

Procedure Name

Input Arguments

Internal/Local Variables
Note! Each variable starts with @

SQL Code (the "body" of the Stored Procedure)

2 Using the Stored Procedure:

```
execute StudentGrade 'John Wayne', 'SCE2006', 'B'
```

Data Scripts

Hans-Petter Halvorsen, M.Sc.

Data Script

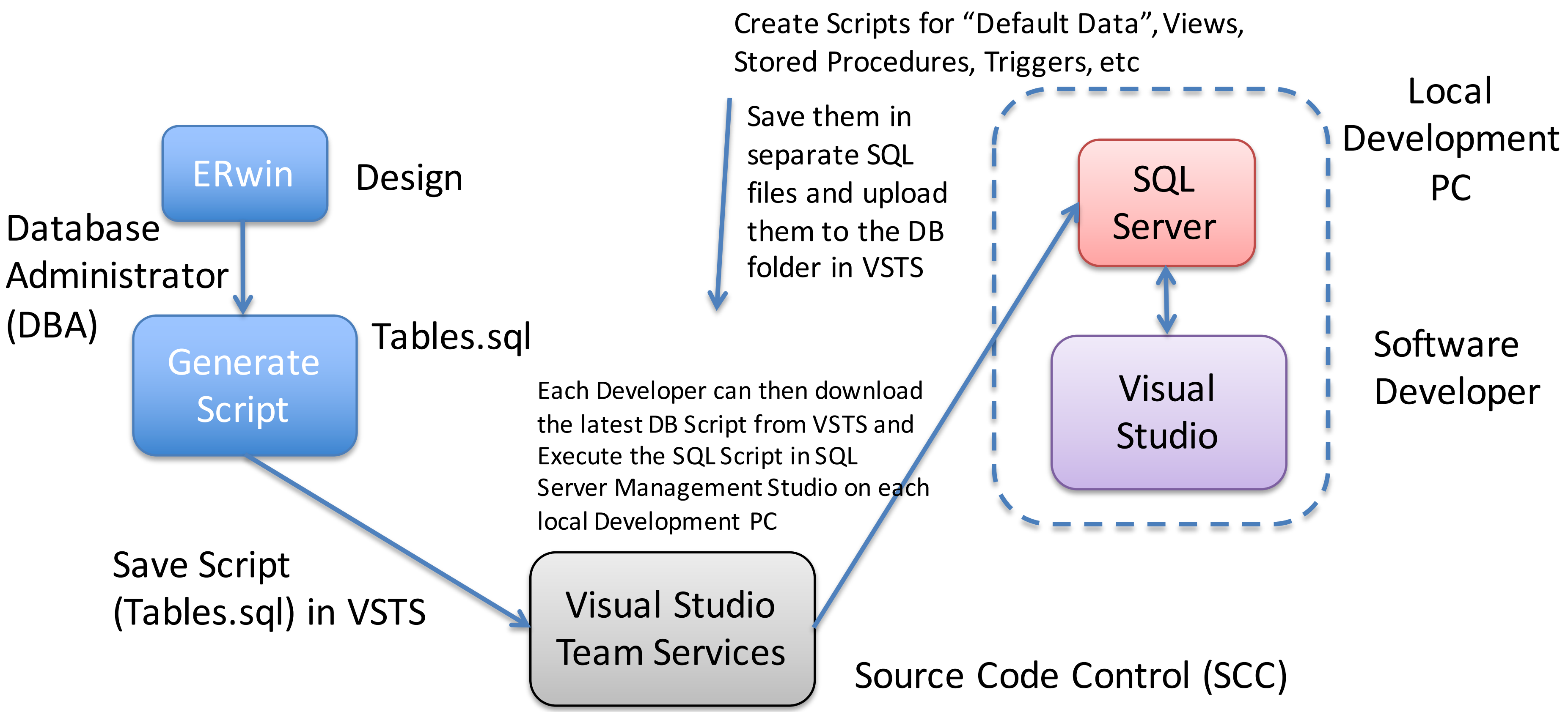
- Typically we need to have some data in the Database, typically some information needed by our Software Program
- It could be e.g., some “Schools”, etc. that our Software System need to run properly
- All these “Default Data” can be entered in a Script

Visual Studio

Visual Studio Team Services

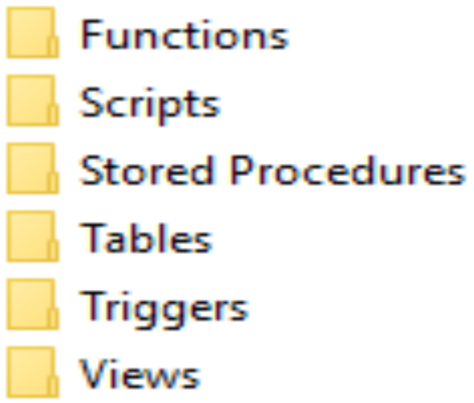
Hans-Petter Halvorsen, M.Sc.

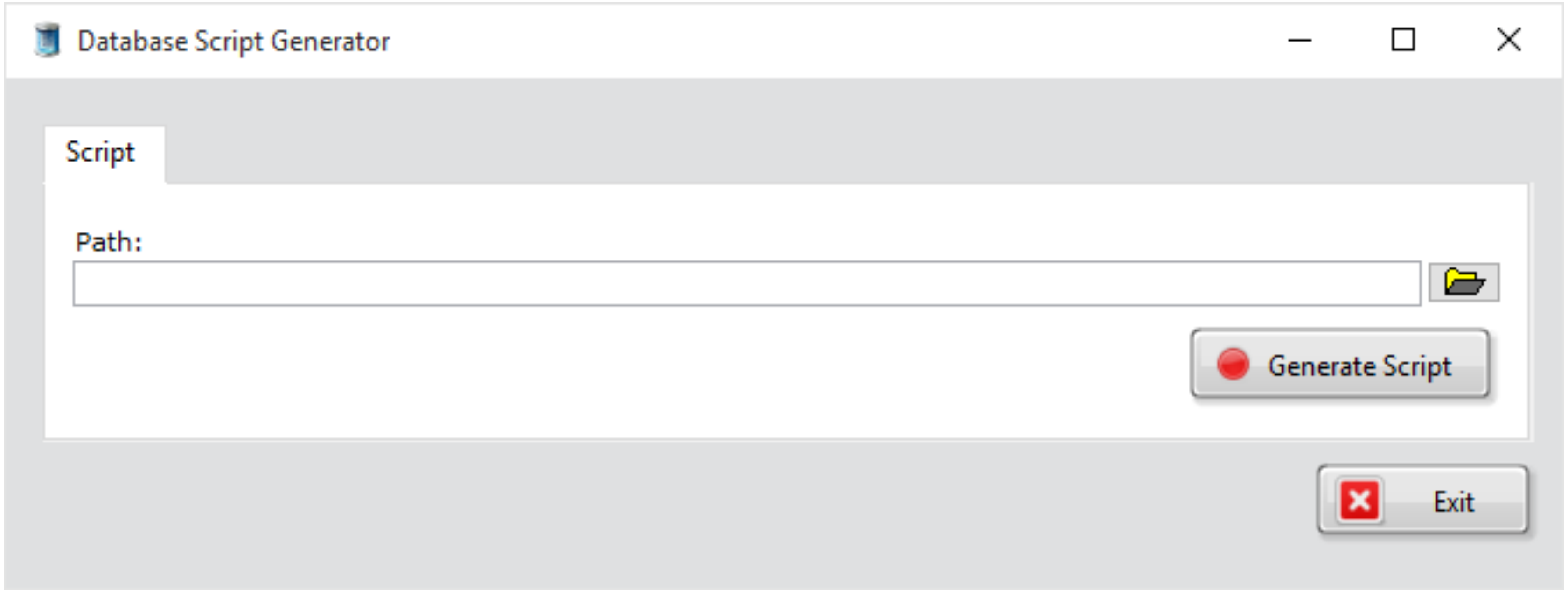
Save/Update Scripts to VSTS


ERwin
Design
Database Administrator (DBA)
Generate Script
Tables.sql
Save Script (Tables.sql) in VSTS
Visual Studio Team Services
Source Code Control (SCC)
Create Scripts for "Default Data", Views, Stored Procedures, Triggers, etc
Save them in separate SQL files and upload them to the DB folder in VSTS
Each Developer can then download the latest DB Script from VSTS and Execute the SQL Script in SQL Server Management Studio on each local Development PC
SQL Server
Visual Studio
Local Development PC
Software Developer

The DBA is in charge of maintening the DB Script that can be used on the Developer PCs and later deployed in the Customer Environment

Database Script Generator





Hans-Petter Halvorsen, M.Sc.

University College of Southeast Norway

www.usn.no

E-mail: hans.p.halvorsen@hit.no

Blog: http://home.hit.no/~hansha/