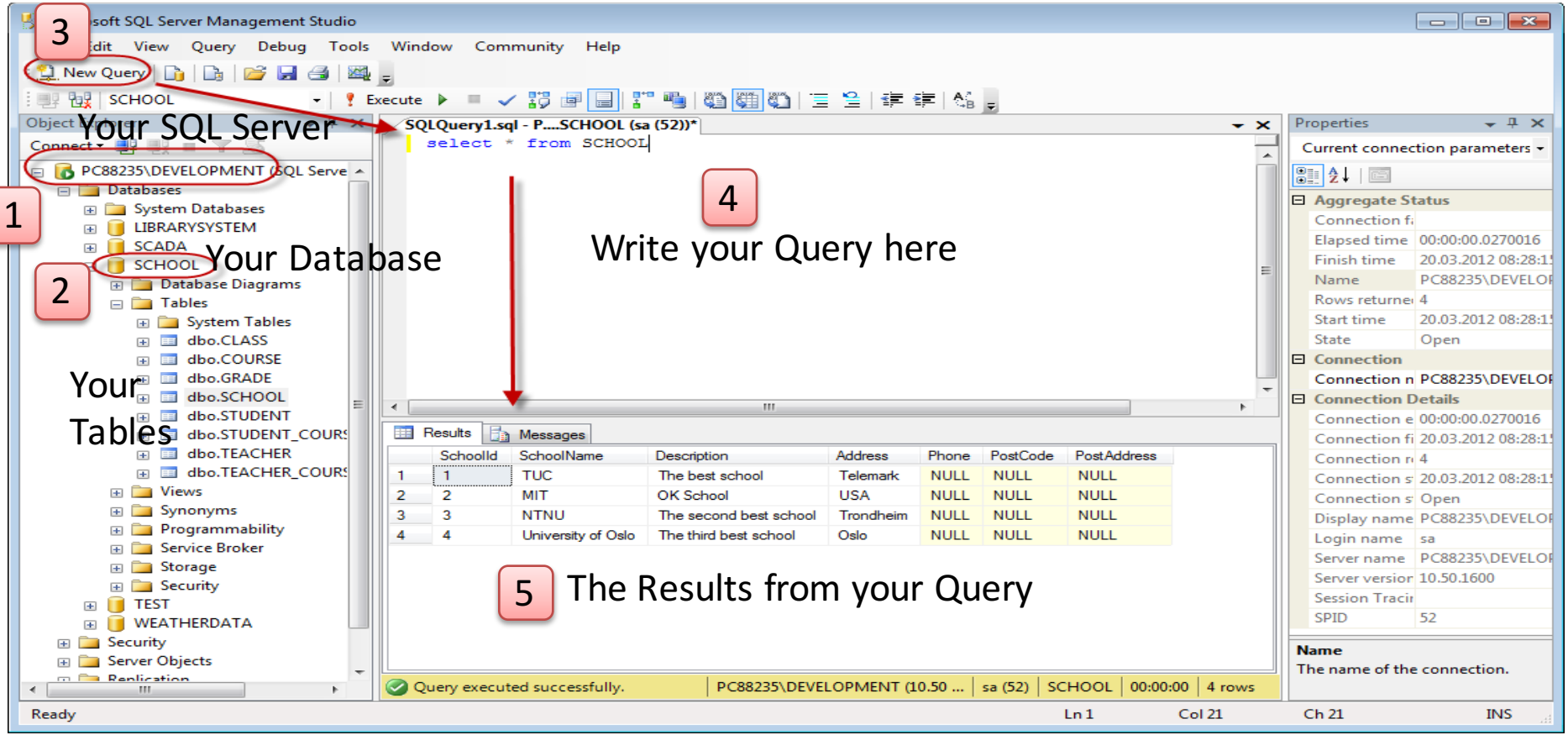
Database Views & Stored Procedures

Hans-Petter Halvorsen, M.Sc.

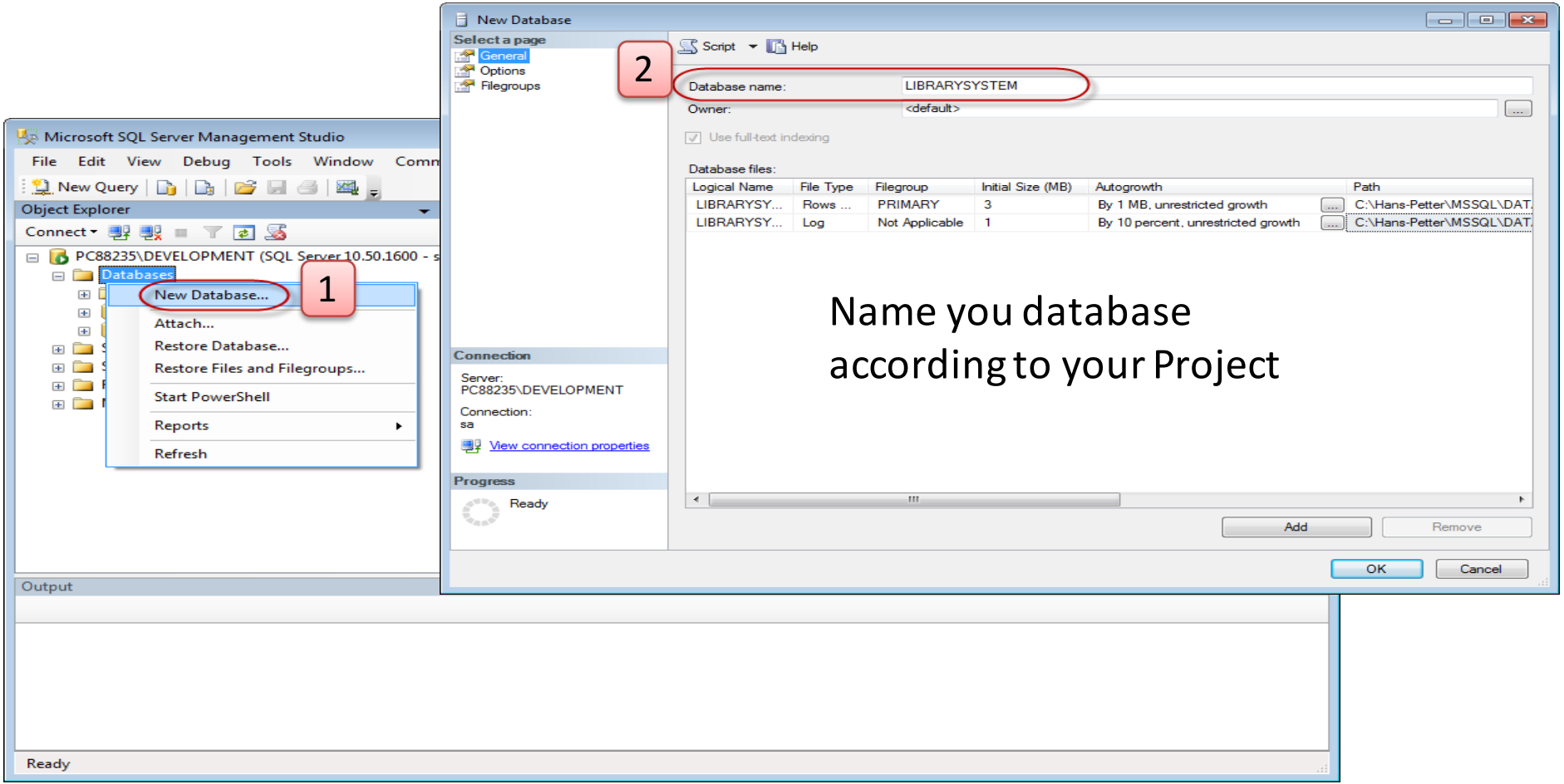
SQL Server

Hans-Petter Halvorsen, M.Sc.

Microsoft SQL Server



Microsoft SQL Server – Create a New Database



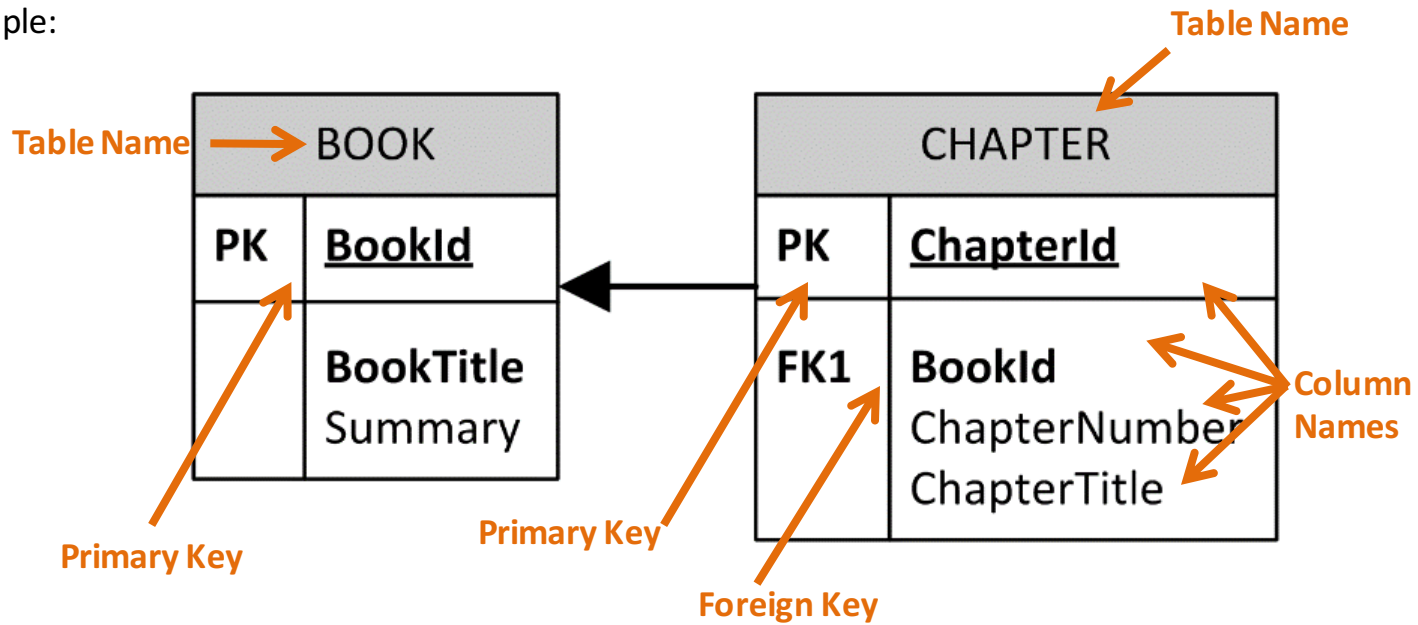
- **Views**: Views are virtual tables for easier access to data stored in multiple tables.
- **Stored Procedures**: A Stored Procedure is a precompiled collection of SQL statements. In a stored procedure you can use if sentence, declare variables, etc. (like a Method in C#)
- **Triggers**: A database trigger is code that is automatically executed in response to certain events on a particular table in a database.
- **Functions**: With SQL and SQL Server you can use lots of built-in functions or you may create your own functions

Database Design – ER Diagram

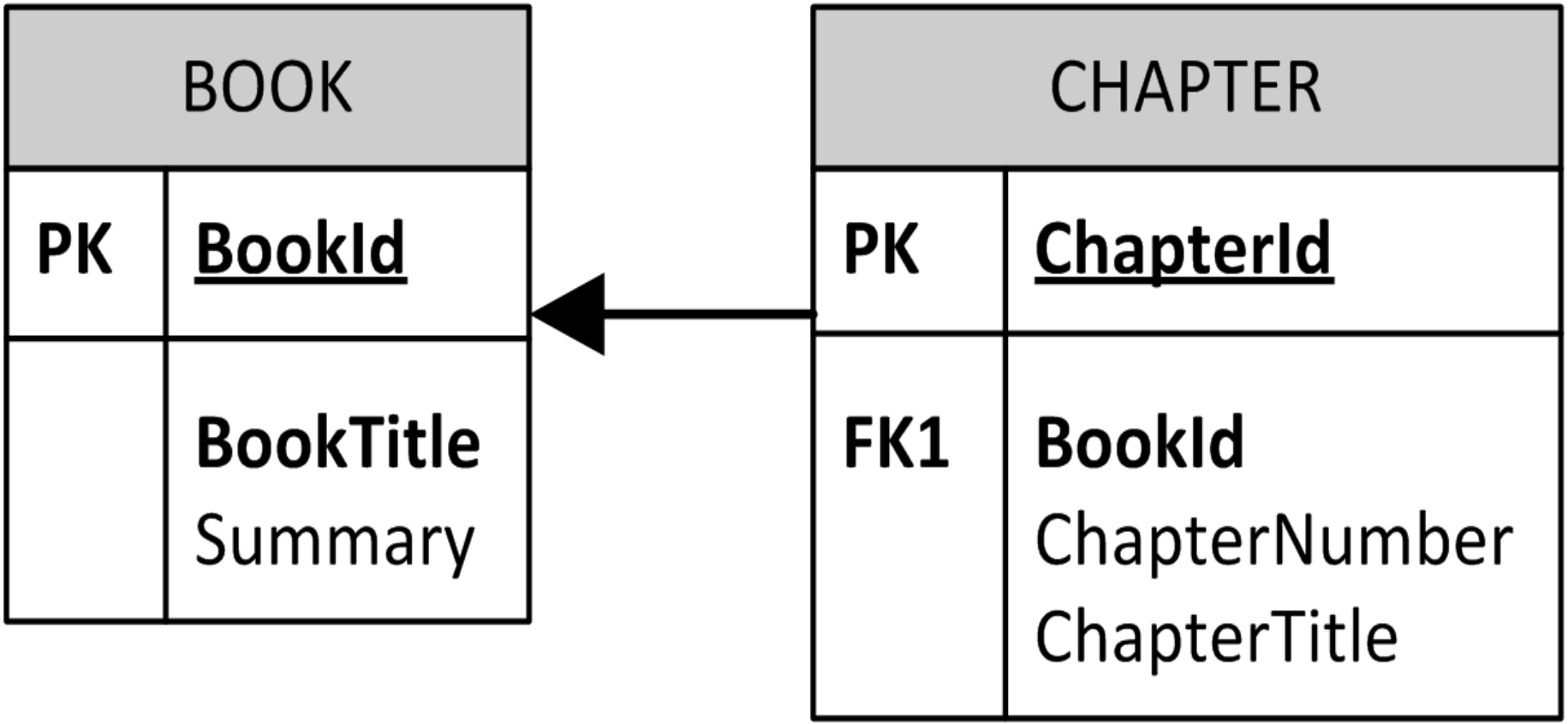
ER Diagram (Entity-Relationship Diagram)

- 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.


BOOK
PK
BookId
BookTitle
Summary
CHAPTER
PK
ChapterId
FK1
BookId
ChapterNumber
ChapterTitle

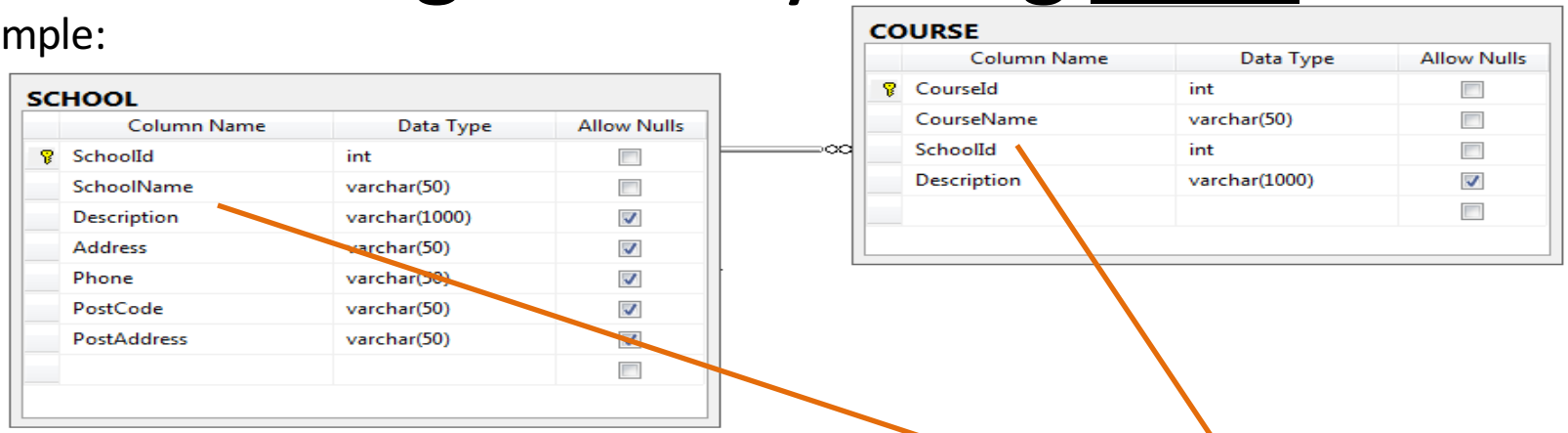
Lets Create these Tables using SQL Server

Views

Hans-Petter Halvorsen, M.Sc.

Get Data from multiple tables in a single Query using Joins

Example:




```
select
SchoolName,
CourseName
from
SCHOOL
inner join COURSE on SCHOOL.SchoolId = COURSE.SchoolId
```

	SchoolName	CourseName
1	TUC	Industrial IT
2	TUC	Control with Implementation
3	TUC	Systems and Control Laboratory

You link Primary Keys and Foreign Keys together

1

Creating Views using SQL code

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:

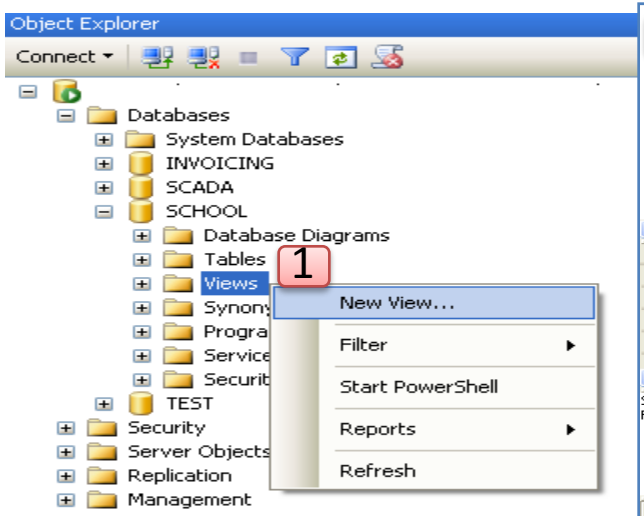
Using the View:

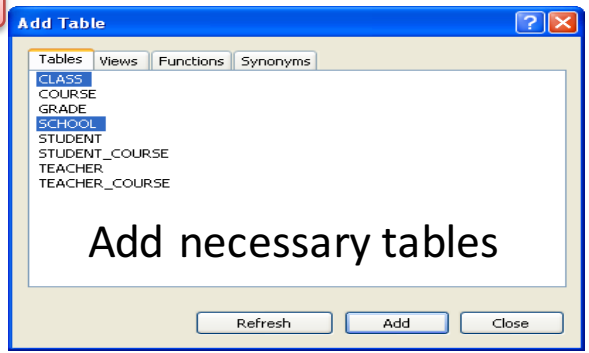
2

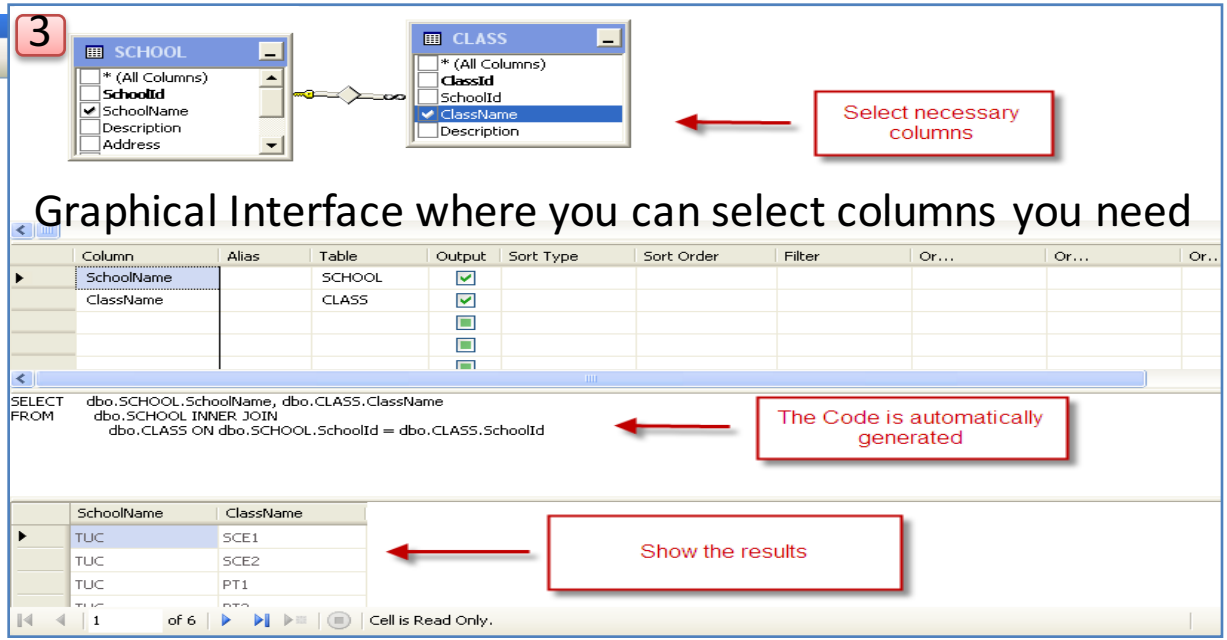
```
select * from CourseData
```

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

Creating Views using the Editor







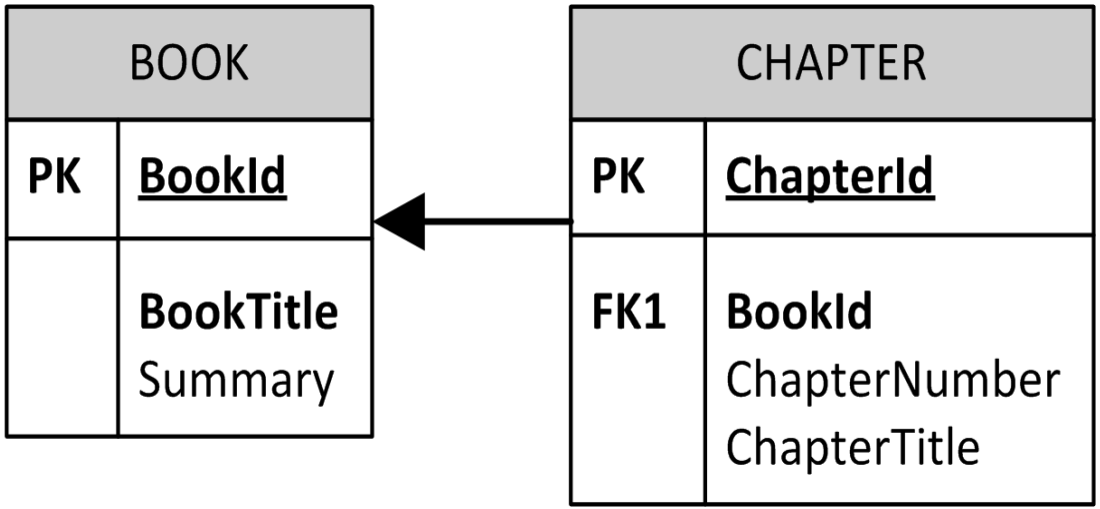
4

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

View Template

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

CREATE VIEW <ViewName>
AS

SELECT
<TableName>.<ColumnName>,
<TableName>.<ColumnName>,
<TableName>.<ColumnName>,
<TableName>.<ColumnName>,
<TableName>.<ColumnName>

FROM
<TableName1>
INNER JOIN <TableName2> ON <TableName1>.<PrimKeyColumnName1> = <TableName2>.<PrimKeyColumnName2>
GO
```

Creating Views - Exercise



GetBookChapters

Create the View **GetBookChapters**

“GetBookChapters” View

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

CREATE VIEW GetBookChapters
AS

SELECT
BOOK.BookId,
BOOK.BookTitle,
BOOK.Summary,
CHAPTER.ChapterNumber,
CHAPTER.ChapterTitle

FROM BOOK
INNER JOIN CHAPTER ON BOOK.BookId = CHAPTER.BookId

GO
```

Stored Procedures

Hans-Petter Halvorsen, M.Sc.

1 Create Stored Procedure:

Stored Procedure

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

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)

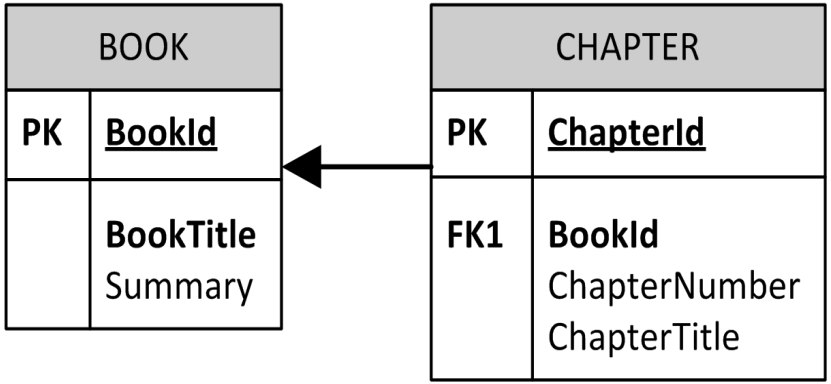
2 Using the Stored Procedure:

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

Stored Procedure Template

```
IF EXISTS (SELECT name
            FROM   sysobjects
            WHERE  name = '<StoredProcedureName>'
            AND    type = 'P')
          DROP PROCEDURE <StoredProcedureName>
GO

CREATE PROCEDURE <StoredProcedureName>
@<InputVariable1> <DataType>,
@<InputVariable2> <DataType>
 AS

DECLARE
@<InternalVariable1> <DataType>,
@<InternalVariable2> <DataType>

select @<InternalVariable1> = <ColumnName> from <TableName> where <ColumnName> = @<InputVariable1>

insert into <TableName> (<ColumnName1>, <ColumnName2>, ...) values (@<InternalVariable1>, @<Inputvariable1>, ...)
GO
```

Creating Stored Procedures - Exercise



CreateBook(BookName, Summary)

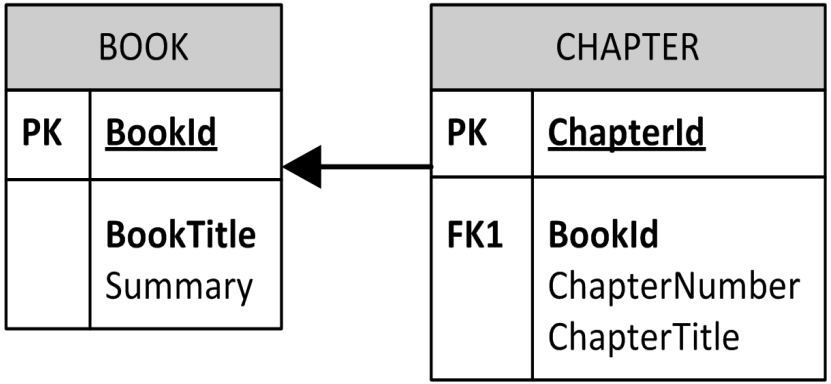
CreateChapter(BookName, ChapterNumber, ChapterTitle)

Create the Stored Procedure
CreateBook(BookName, Summary)

“CreateBook” Stored Procedure

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

CREATE PROCEDURE CreateBook
@BookTitle varchar(50),
@Summary varchar(255)
AS

insert into BOOK (BookTitle, Summary) values (@BookTitle, @Summary)

GO
```

Creating Stored Procedures - Exercise



CreateBook(BookName, Summary)

CreateChapter(BookName, ChapterNumber, ChapterTitle)

Create the Stored Procedure
CreateChapter(BookName, ChapterNumber, ChapterTitle)

“CreateChapter” Stored Procedure

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

CREATE PROCEDURE CreateChapter
@BookTitle varchar(50),
@ChapterNumber int,
@ChapterTitle varchar(50)
AS

DECLARE
@BookId int

select @BookId = BookId from BOOK where BookTitle = @BookTitle

insert into CHAPTER(BookId, ChapterNumber, ChapterTitle) values (@BookId, @ChapterNumber, @ChapterTitle)
GO
```

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/