https://www.halvorsen.blog

SQL Server and Structured Query Language (SQL)

Hans-Petter Halvorsen

Contents

- **SQL Server** and **SQL Server Management Studio**
- Tables
  - Create Tables using **erwin Data Modeler**
  - Create/Update Tables using built-in Designer in SQL Server Management studio
- **Structured Query Language (SQL)**
  - Insert, Retrieve, Update, Delete Data
- **Stored Procedures** and **Views**

https://www.halvorsen.blog

SQL Server

and SQL Server Management Studio

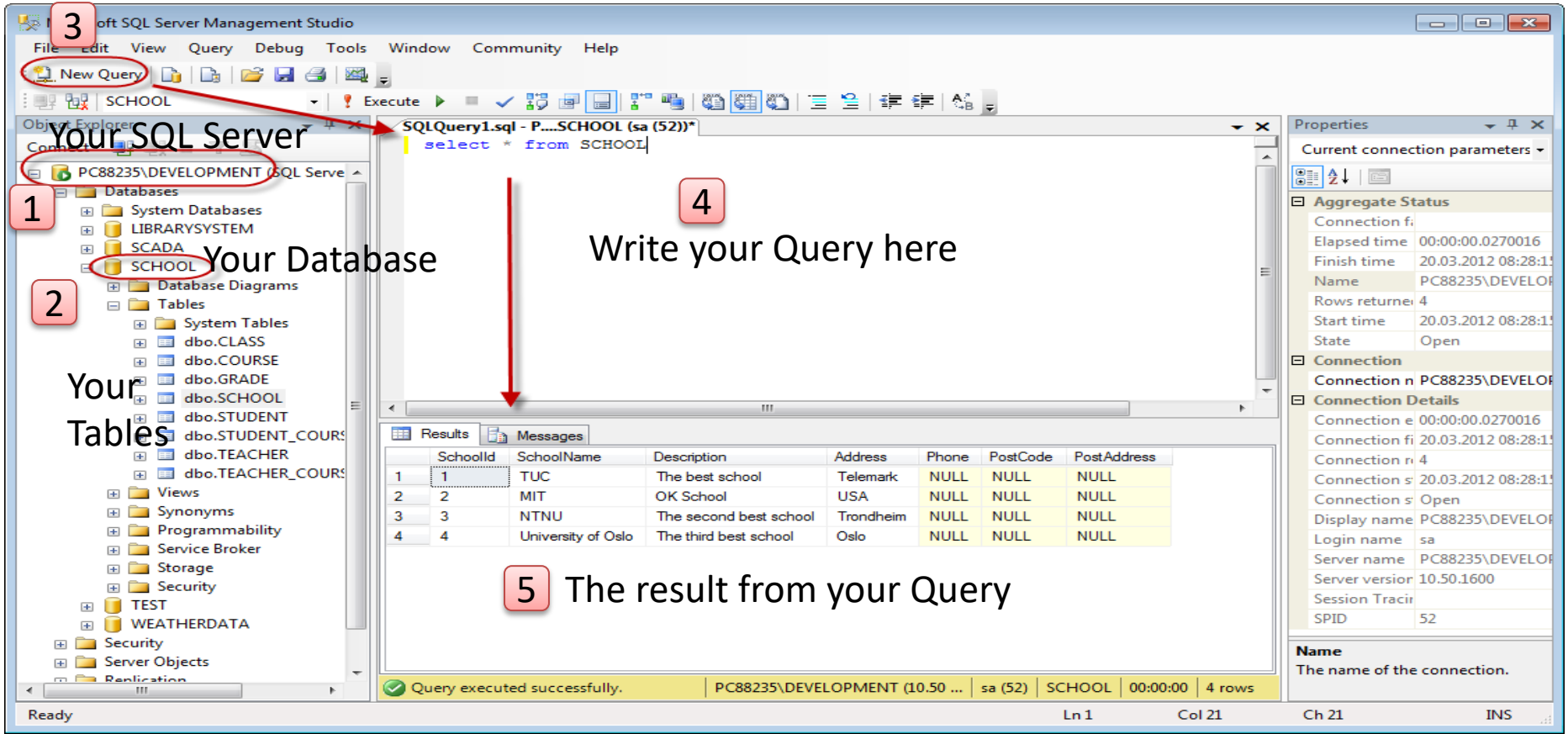
Hans-Petter Halvorsen

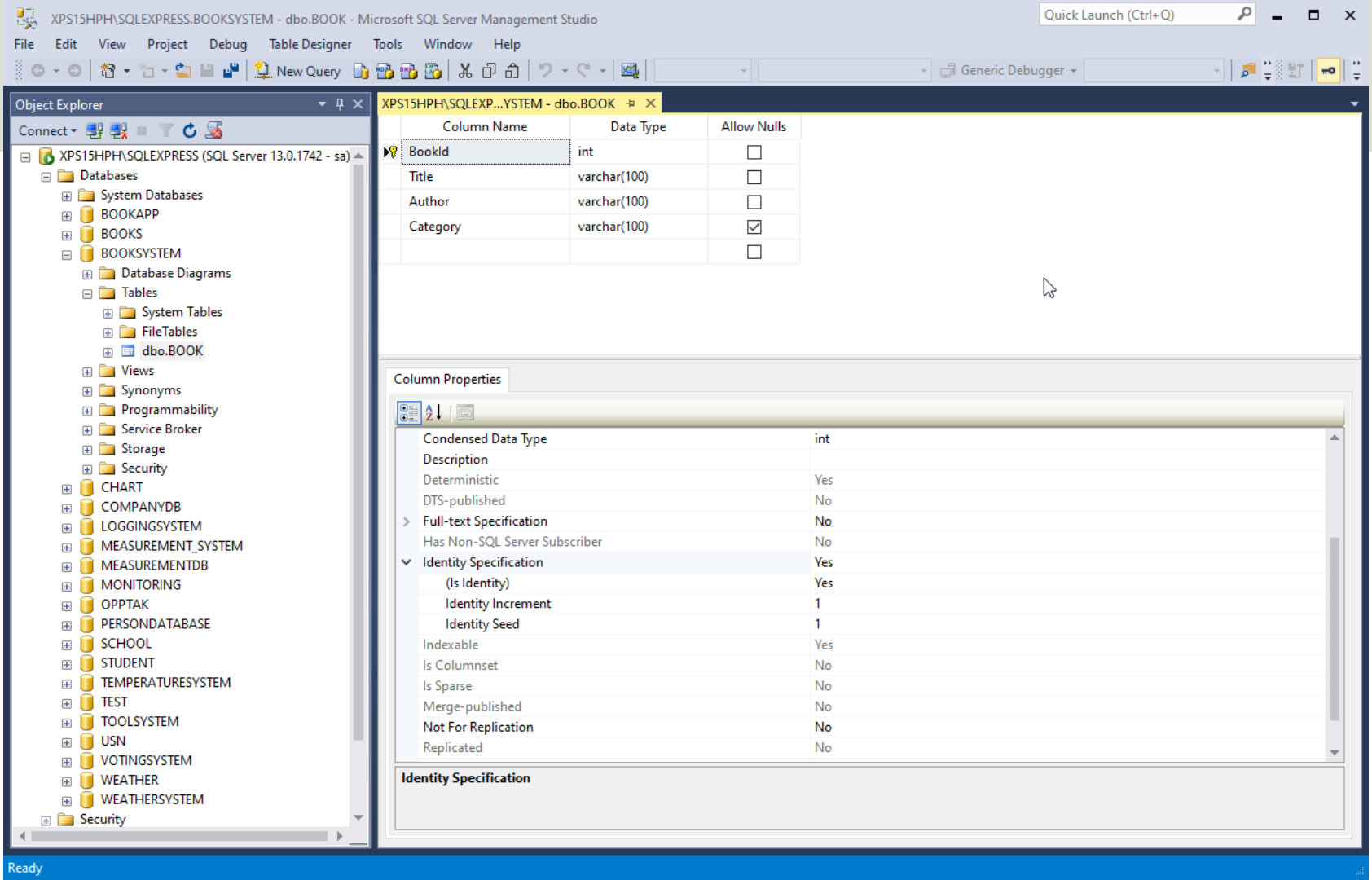
Table of Contents

SQL Server

- SQL Server Express
  - Free version of SQL Server that has all we need for the exercises in this Tutorial
- SQL Server Express consist of 2 parts (separate installation packages):
  - SQL Server Express
  - SQL Server Management Studio (SSMS) – This software can be used to create Databases, create Tables, Insert/Retrieve or Modify Data, etc.
- SQL Server Express Installation: https://youtu.be/hhhggAlUYo8

SQL Server Management Studio



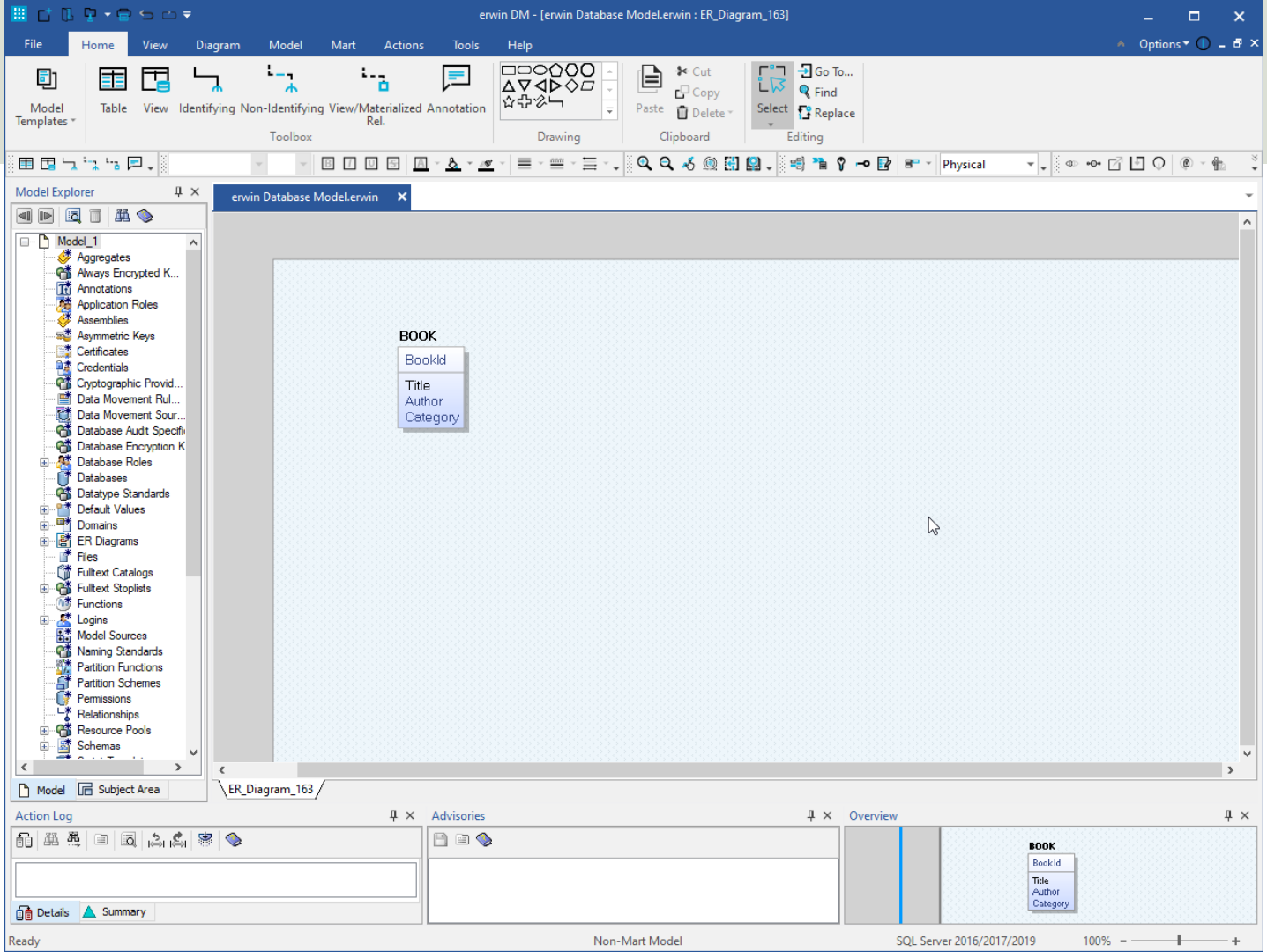

XPS15HPH\SQLEXPRESS.BOOKSYSTEM - dbo.BOOK - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Project Debug Table Designer Tools Window Help
New Query
Generic Debugger
Object Explorer
Connect
XPS15HPH\SQLEXPRESS (SQL Server 13.0.1742 - sa)
Databases
System Databases
BOOKAPP
BOOKS
BOOKSYSTEM
Database Diagrams
Tables
System Tables
FileTables
dbo.BOOK
Views
Synonyms
Programmability
Service Broker
Storage
Security
CHART
COMPANYDB
LOGGINGSYSTEM
MEASUREMENT_SYSTEM
MEASUREMENTDB
MONITORING
OPPTAK
PERSONDATABASE
SCHOOL
STUDENT
TEMPERATURESYSTEM
TEST
TOOLSYSTEM
USN
VOTINGSYSTEM
WEATHER
WEATHERSYSTEM
Security
XPS15HPH\SQLEXP...YSTEM - dbo.BOOK
Column Name
Data Type
Allow Nulls
BookId
int
Title
varchar(100)
Author
varchar(100)
Category
varchar(100)
Column Properties
Condensed Data Type
int
Description
Deterministic
Yes
DTS-published
No
Full-text Specification
No
Has Non-SQL Server Subscriber
No
Identity Specification
Yes
(Is Identity)
Yes
Identity Increment
1
Identity Seed
1
Indexable
Yes
Is Columnset
No
Is Sparse
No
Merge-published
No
Not For Replication
No
Replicated
No
Identity Specification
Ready

https://www.halvorsen.blog

erwin Data Modeler

Hans-Petter Halvorsen

Table of Contents


erwin DM - [erwin Database Model.erwin : ER_Diagram_163]
File
Home
View
Diagram
Model
Mart
Actions
Tools
Help
Options
Model Templates
Table
View
Identifying
Non-Identifying
View/Materialized Rel.
Annotation
Toolbox
Paste
Cut
Copy
Delete
Drawing
Clipboard
Select
Go To...
Find
Replace
Editing
Physical
Model Explorer
Model_1
Aggregates
Always Encrypted K...
Annotations
Application Roles
Assemblies
Asymmetric Keys
Certificates
Credentials
Cryptographic Provid...
Data Movement Rul...
Data Movement Sour...
Database Audit Specifi
Database Encryption K
Database Roles
Databases
Datatype Standards
Default Values
Domains
ER Diagrams
Files
Fulltext Catalogs
Fulltext Stoplists
Functions
Logins
Model Sources
Naming Standards
Partition Functions
Partition Schemes
Permissions
Relationships
Resource Pools
Schemas
Model
Subject Area
erwin Database Model.erwin
BOOK
BookId
Title
Author
Category
ER_Diagram_163
Action Log
Details
Summary
Advisories
Overview
BOOK
BookId
Title
Author
Category
Ready
Non-Mart Model
SQL Server 2016/2017/2019
100%

Database Script

```
CREATE TABLE [BOOK]
(
	[BookId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
	[Title] [varchar](100) NOT NULL UNIQUE,
	[Author] [varchar](100) NOT NULL,
	[Category] [varchar](100) NOT NULL
)
GO
```

https://www.halvorsen.blog

SQL

Structured Query Language

Hans-Petter Halvorsen

Table of Contents

Structured Query Language

- Structured Query Language (SQL) is used to write, read and update data from the Database System
- You can use SQL inside the “SQL Server Management Studio” or inside your Python script.
- SQL Example: select * from SCHOOL

Database CRUD

All Database Systems supports CRUD

C – Create or Insert Data

R – Retrieve Data

U – Update Data

D – Delete Data

Let's go through some examples

SQL Examples

Query Examples:

```
• insert into STUDENT (Name , Number, SchoolId)
  values ('John Smith', '100005', 1)

• select SchoolId, Name from SCHOOL

• select * from SCHOOL where SchoolId > 100

• update STUDENT set Name='John Wayne' where StudentId=2

• delete from STUDENT where SchoolId=3
```

We have 4 different Query Types: **INSERT**, **SELECT**, **UPDATE** and **DELETE**

CRUD: **C** – Create or Insert Data, **R** – Retrieve (Select) Data, **U** – Update Data, **D** – Delete Data

Insert Data

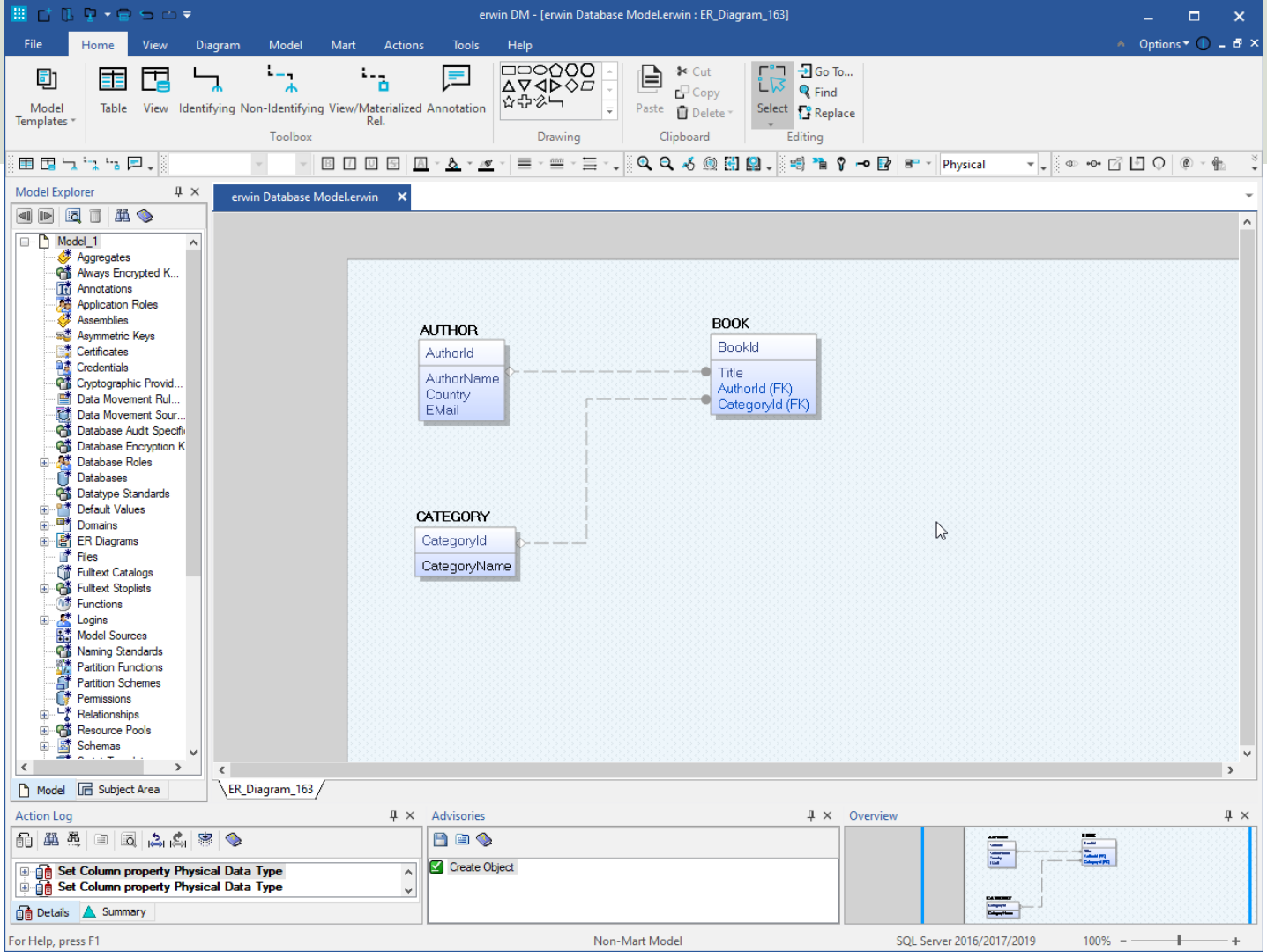
```
INSERT INTO BOOK (Title, Author, Category)
VALUES ('Python Program','Knut Hamsun', 'Data’)
GO

INSERT INTO BOOK (Title, Author, Category)
VALUES ('Music History','Elvis Presley', 'Music’)
GO

INSERT INTO BOOK (Title, Author, Category)
VALUES ('Et Dukkehjem','Henrik Ibsen', 'Novel')
GO
```

Improved Database Design

We update our simple Database with more Tables

Hans-Petter Halvorsen

Table of Contents


erwin DM - [erwin Database Model.erwin : ER_Diagram_163]
File
Home
View
Diagram
Model
Mart
Actions
Tools
Help
Options
Model Templates
Table
View
Identifying
Non-Identifying
View/Materialized Rel.
Annotation
Toolbox
Paste
Cut
Copy
Delete
Clipboard
Select
Go To...
Find
Replace
Editing
Drawing
Physical
Model Explorer
Model_1
Aggregates
Always Encrypted K...
Annotations
Application Roles
Assemblies
Asymmetric Keys
Certificates
Credentials
Cryptographic Provid...
Data Movement Rul...
Data Movement Sour...
Database Audit Specifi
Database Encryption K
Database Roles
Databases
Datatype Standards
Default Values
Domains
ER Diagrams
Files
Fulltext Catalogs
Fulltext Stoplists
Functions
Logins
Model Sources
Naming Standards
Partition Functions
Partition Schemes
Permissions
Relationships
Resource Pools
Schemas
Model
Subject Area
erwin Database Model.erwin
AUTHOR
AuthorId
AuthorName
Country
EMail
BOOK
BookId
Title
AuthorId (FK)
CategoryId (FK)
CATEGORY
CategoryId
CategoryName
ER_Diagram_163
Action Log
Set Column property Physical Data Type
Set Column property Physical Data Type
Details
Summary
Advisories
Create Object
Overview
For Help, press F1
Non-Mart Model
SQL Server 2016/2017/2019
100%

Forward Engineer Schema Generation Wizard

Schema Generation Preview

This page provides a preview of the Forward Engineer Schema Generation.

- Overview
- Option Selection
- Summary
- Owner Override
- Table Filter
- **Preview**

```
CREATE TABLE [AUTHOR]
(
        [AuthorId]          int  IDENTITY ( 1,1 )  NOT NULL ,
        [AuthorName]        varchar(100)  NOT NULL ,
        [EMail]             varchar(100)  NULL ,
        [Country]           varchar(100)  NULL ,
        PRIMARY KEY  CLUSTERED ([AuthorId] ASC)
)
go

CREATE TABLE [CATEGORY]
(
        [CategoryId]        int  IDENTITY ( 1,1 )  NOT NULL ,
        [CategoryName]      varchar(100)  NOT NULL ,
        PRIMARY KEY  CLUSTERED ([CategoryId] ASC),
        UNIQUE ([CategoryName]  ASC)
)
go

CREATE TABLE [BOOK]
(
        [BookId]            int  IDENTITY ( 1,1 )  NOT NULL ,
        [Title]             varchar(100)  NOT NULL ,
        [AuthorId]          int  NOT NULL ,
        [CategoryId]        int  NULL ,
        PRIMARY KEY  CLUSTERED ([BookId] ASC),
         FOREIGN KEY ([AuthorId]) REFERENCES [AUTHOR]([AuthorId]),
         FOREIGN KEY ([CategoryId]) REFERENCES [CATEGORY]([CategoryId])
)
go
```

< Back | Next > | Generate | OK | Cancel | Help

```
CREATE TABLE [AUTHOR]
(
            [AuthorId]          int  IDENTITY ( 1,1 )  NOT NULL ,
            [AuthorName]        varchar(100)  NOT NULL ,
            [EMail]             varchar(100)  NULL ,
            [Country]           varchar(100)  NULL ,
            PRIMARY KEY  CLUSTERED ([AuthorId] ASC)
)
go

CREATE TABLE [CATEGORY]
(
            [CategoryId]        int  IDENTITY ( 1,1 )  NOT NULL ,
            [CategoryName]      varchar(100)  NOT NULL ,
            PRIMARY KEY  CLUSTERED ([CategoryId] ASC),
            UNIQUE ([CategoryName]  ASC)
)
go

CREATE TABLE [BOOK]
(
            [BookId]            int  IDENTITY ( 1,1 )  NOT NULL ,
            [Title]             varchar(100)  NOT NULL ,
            [AuthorId]          int  NOT NULL ,
            [CategoryId]        int  NULL ,
            PRIMARY KEY  CLUSTERED ([BookId] ASC),
            FOREIGN KEY ([AuthorId]) REFERENCES [AUTHOR]([AuthorId]),
            FOREIGN KEY ([CategoryId]) REFERENCES [CATEGORY]([CategoryId])
)
go
```

Insert Authors

```
insert into AUTHOR (AuthorName, Country, EMail)
values ('Knut Hamsun', 'Norway', 'knut.hamsun@gmail.com')
go

insert into AUTHOR (AuthorName, Country, EMail)
values ('Henrik Ibsen', 'Norway', 'henrik.ibsen@gmail.com')
go

insert into AUTHOR (AuthorName, Country, EMail)
values ('Jo Nesbø', 'Norway', 'jo.nesbo@gmail.com')
go
```

Insert Categories

```
insert into CATEGORY (CategoryName)
values ('Programming')
go

insert into CATEGORY (CategoryName)
values ('Science')
go

insert into CATEGORY (CategoryName)
values ('Science fiction')
go
```

Insert Books

```
INSERT INTO BOOK (Title, AuthorId, CategoryId)
VALUES ('Python Programming', 1, 1)
GO

INSERT INTO BOOK (Title, AuthorId, CategoryId)
VALUES ('Music History', 2, 2)
GO

INSERT INTO BOOK (Title, AuthorId, CategoryId)
VALUES ('Et Dukkehjem', 3, 3)
GO
```

Stored Procedures

Hans-Petter Halvorsen

Table of Contents

Stored Procedure

```
CREATE PROCEDURE CreateBook
@Title varchar(100),
@AuthorName varchar(100),
@CategoryName varchar(100)
AS

DECLARE
@AuthorId int,
@CategoryId int

select @AuthorId=AuthorId from AUTHOR where AuthorName=@AuthorName

select @CategoryId=CategoryId from CATEGORY where CategoryName=@CategoryName

INSERT INTO BOOK (Title, AuthorId, CategoryId)
VALUES (@Title, @AuthorId, @CategoryId)

GO
```

Using Stored Procedure

```
CreateBook 'SQL Programming', 'Knut Hamsun’, Programming'
```

Views

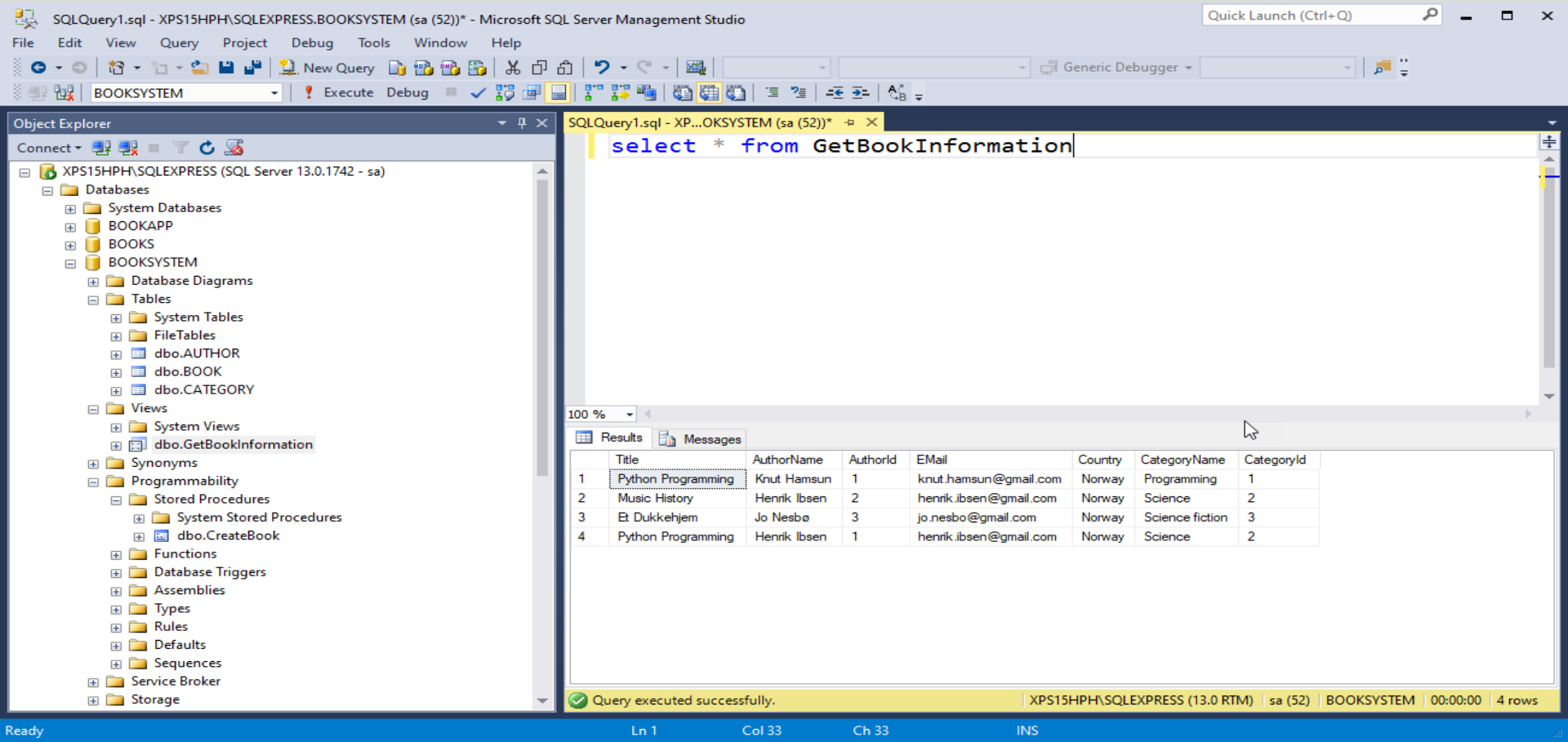
Hans-Petter Halvorsen

Table of Contents

View

```
CREATE VIEW GetBookInformation
AS
SELECT
BOOK.Title,
AUTHOR.AuthorName,
BOOK.AuthorId,
AUTHOR.EMail,
AUTHOR.Country,
CATEGORY.CategoryName,
BOOK.CategoryId
FROM  BOOK
INNER JOIN AUTHOR ON BOOK.CategoryId = AUTHOR.AuthorId
INNER JOIN CATEGORY ON BOOK.CategoryId = CATEGORY.CategoryId
GO
```

Using View


SQLQuery1.sql - XPS15HPH\SQLEXPRESS.BOOKSYSTEM (sa (52))* - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Debug Tools Window Help
New Query
Generic Debugger
BOOKSYSTEM
Execute Debug
Object Explorer
Connect
XPS15HPH\SQLEXPRESS (SQL Server 13.0.1742 - sa)
Databases
System Databases
BOOKAPP
BOOKS
BOOKSYSTEM
Database Diagrams
Tables
System Tables
FileTables
dbo.AUTHOR
dbo.BOOK
dbo.CATEGORY
Views
System Views
dbo.GetBookInformation
Synonyms
Programmability
Stored Procedures
System Stored Procedures
dbo.CreateBook
Functions
Database Triggers
Assemblies
Types
Rules
Defaults
Sequences
Service Broker
Storage
SQLQuery1.sql - XP...OKSYSTEM (sa (52))*
select * from GetBookInformation
100 %
Results
Messages
Title AuthorName AuthorId EMail Country CategoryName CategoryId
1 Python Programming Knut Hamsun 1 knut.hamsun@gmail.com Norway Programming 1
2 Music History Henrik Ibsen 2 henrik.ibsen@gmail.com Norway Science 2
3 Et Dukkehjem Jo Nesbø 3 jo.nesbo@gmail.com Norway Science fiction 3
4 Python Programming Henrik Ibsen 1 henrik.ibsen@gmail.com Norway Science 2
Query executed successfully.
XPS15HPH\SQLEXPRESS (13.0 RTM) sa (52) BOOKSYSTEM 00:00:00 4 rows
Ready
Ln 1 Col 33 Ch 33 INS

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

Web: https://www.halvorsen.blog