https://www.halvorsen.blog

Week Assignment

Database Design & Implementation

Hans-Petter Halvorsen

Week Assignment

1. **Create Database Design** (Modelling with erwin Data Modeler)
2. **Database Implementation** (SQL Server)
   - Install Tables inside your local SQL Server
   - Create basic Database API - Views and Stored Procedures
   - Your .sql scripts (Tables, Views, Stored Procedures) should be in the Azure DevOps repository
3. Coding and Implementation
   - **Make sure you can Communicate with the Database from C#.**
   - Make a simple ASP.NET Application that get and save data to/from your Database.

Note! The Database diagram(s) with descriptions should be part of the Software Requirements and Design document: SRS/SDD -> SRD

https://www.halvorsen.blog

Database Design & Implementation

Hans-Petter Halvorsen

Table of Contents

Database Systems

- A Database is a structured way to store lots of information. The information is stored in different tables.
- - “Everything” today is stored in databases!

- **Examples:**
- Bank/Account systems
- Information in Web pages such as Facebook, Wikipedia, YouTube, etc.
- … lots of other examples!

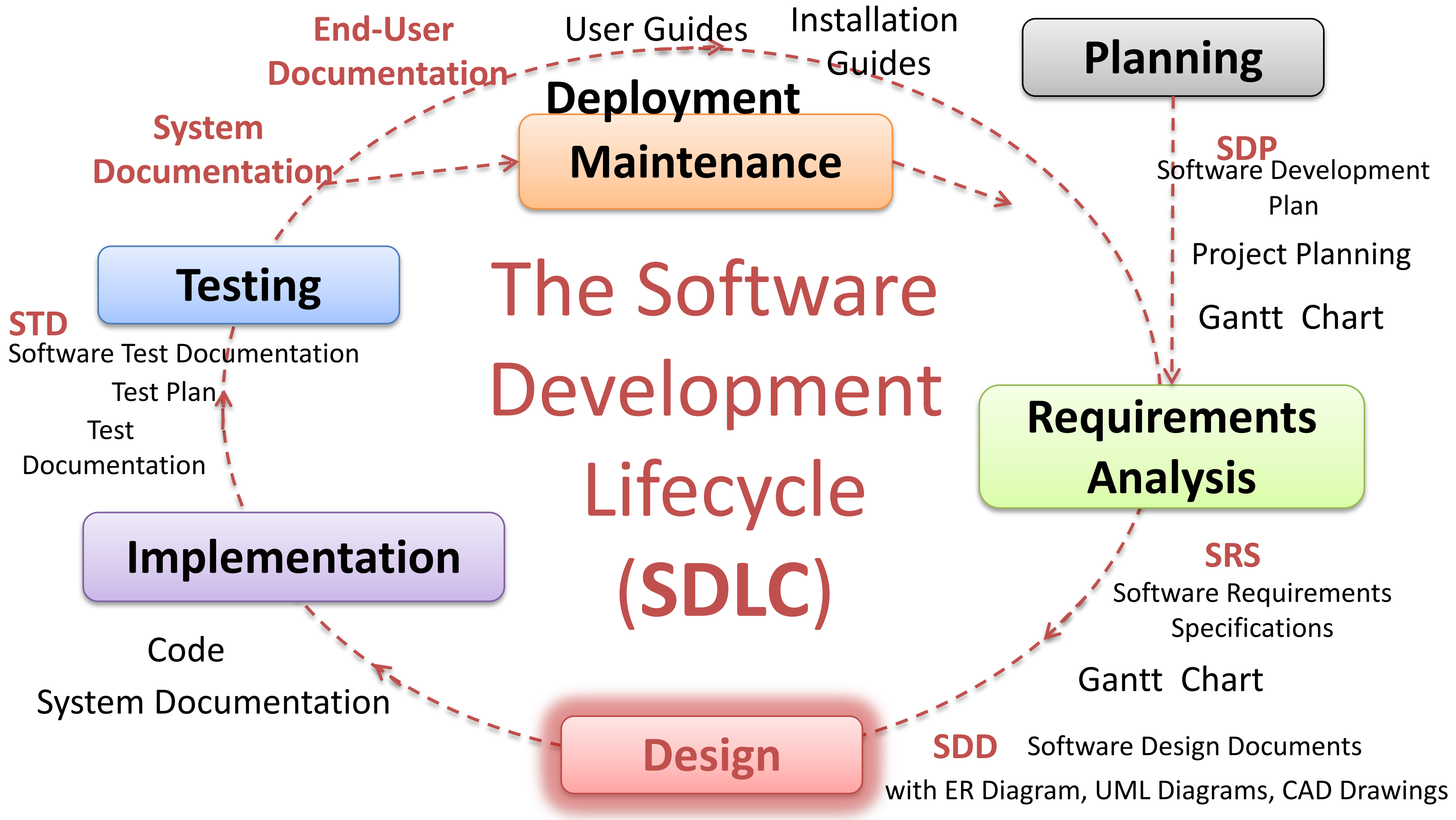
Old fashion Database (Data-storage) Systems

Not too long ago, this was the only data-storage device most companies needed. Those days are over.

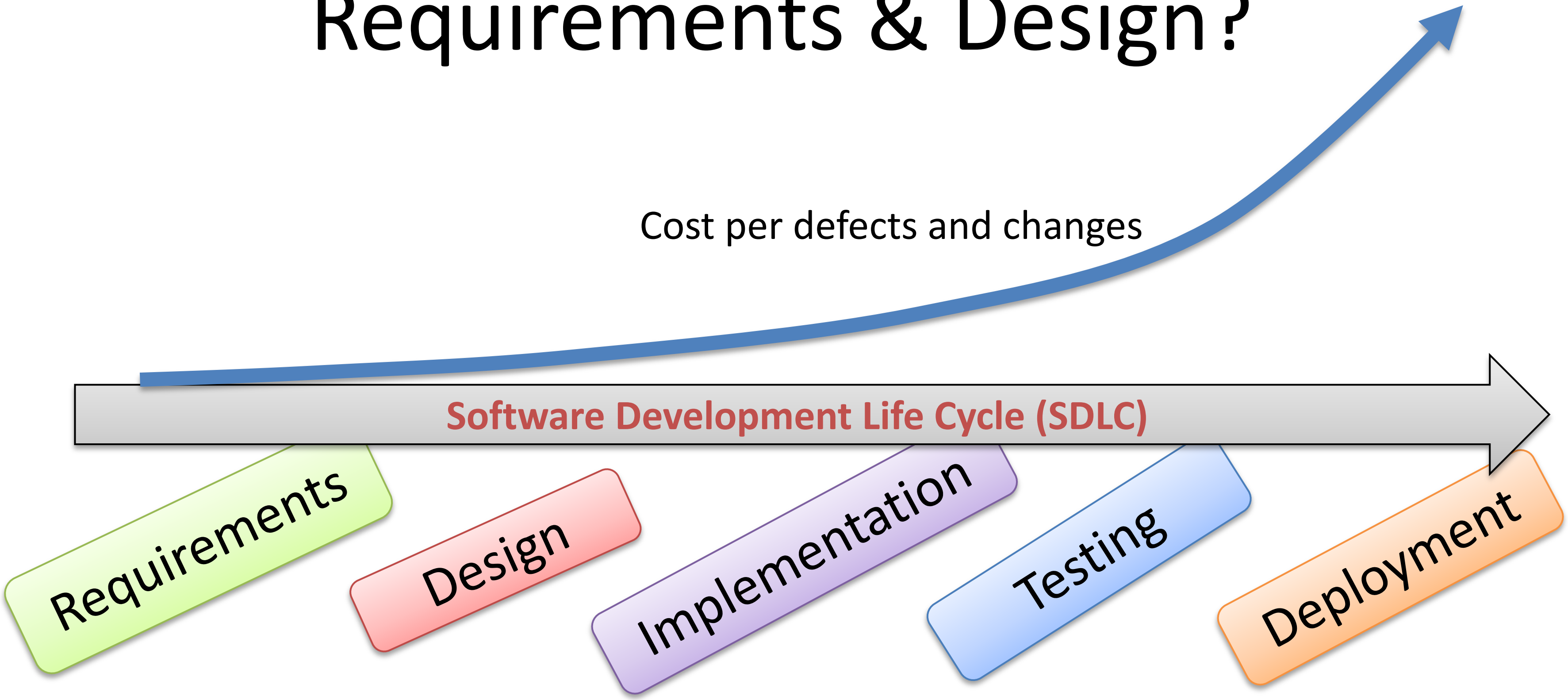
Database Management Systems (DBMS)

- Microsoft SQL Server
- Oracle
- MySQL
- Sybase
- Microsoft Access
- ... (we have hundreds different DBMS)

Article about MySQL: http://www.digi.no/itnorge/2015/11/12/mysql-blir-stadig-mer-norsk


End-User Documentation
User Guides
Installation Guides
Planning
Deployment
Maintenance
System Documentation
SDP
Software Development Plan
Project Planning
Gantt Chart
Testing
The Software Development Lifecycle (SDLC)
STD
Software Test Documentation
Test Plan
Test Documentation
Requirements Analysis
Implementation
SRS
Software Requirements Specifications
Gantt Chart
Code
System Documentation
Design
SDD
Software Design Documents
with ER Diagram, UML Diagrams, CAD Drawings

Why spend time on Requirements & Design?


Cost per defects and changes
Software Development Life Cycle (SDLC)
Requirements
Design
Implementation
Testing
Deployment

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**

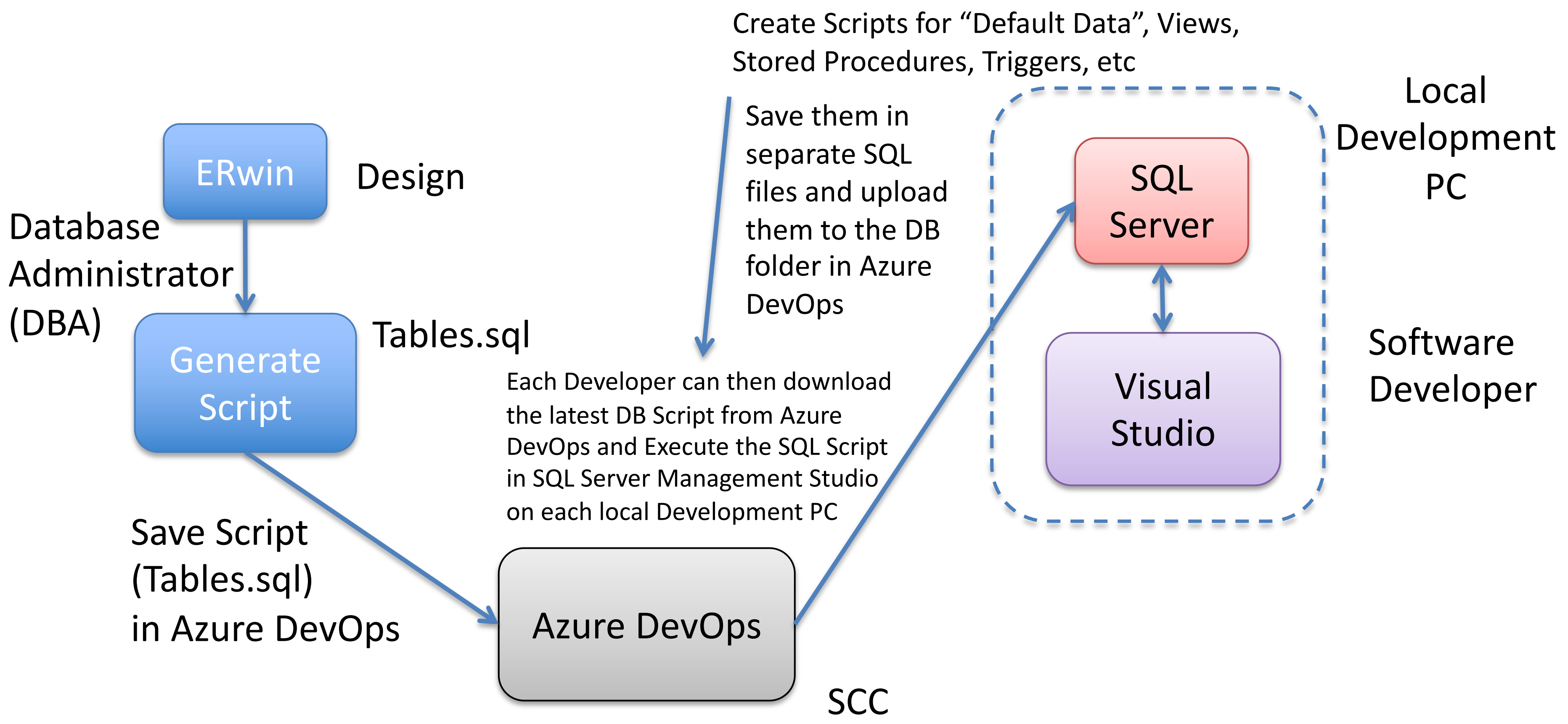
Establish Connection between SQL Server Database and C#

SELECT, INSERT UPDATE between your GUI and SQL Server

Update and Improve

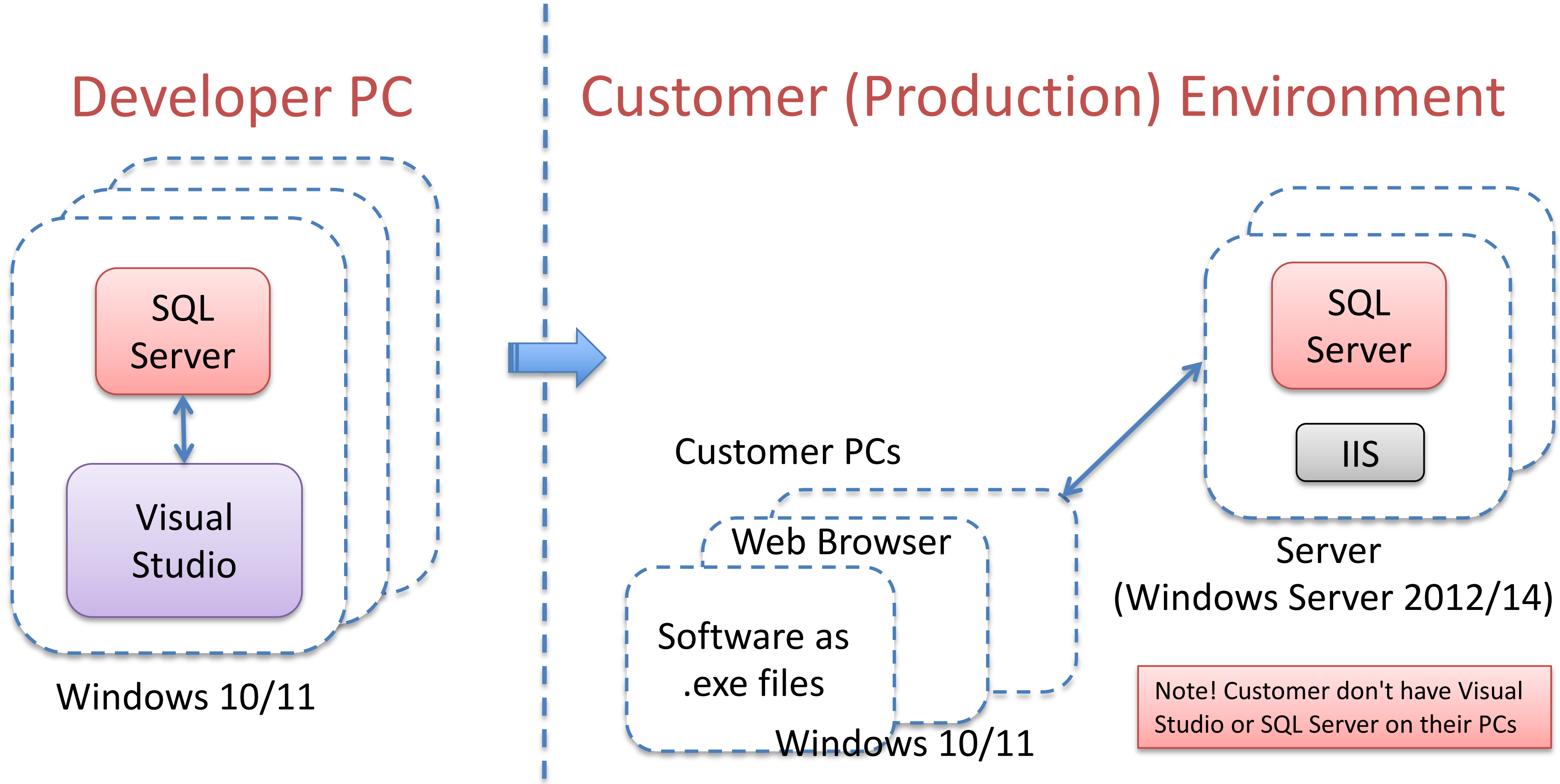
Update and Improve

Create a Table Script


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



Note! Customer don't have Visual Studio or SQL Server on their PCs

Coding Conventions

- Coding Styles and Guidelines (both SQL and C#, etc.)
- Pascal, Camel-case, lower-case or upper-case?
- Table and Columns formatting? etc,.
- It is important that all developer follow the same conventions
- How do you collaborate and store database information (tables, Stored Procedures, Scripts, …)?
- How and Where do you store this in Azure DevOps, etc.
- Etc.

All this information should be part of SDP – Make sure to update your SDP with this information

Database Design & Implementation

Recommended Steps:

1. Database Modelling/Design using erwin Data Modeler
2. Generate SQL Table Script using erwin Data Modeler (you might need to adjust/improve it in order to make it more robust)
3. Generate Tables in SQL Server using the SQL Script generated by erwin Data Modeler
4. Create Stored Procedures, View, Triggers, etc. inside SQL Server if needed (one File for each View, Stored Procedure, etc.). These .sql files should be stored in Azure DevOps
5. Create some simple ASP.NET Applications for saving and retrieving data to/from SQL Server

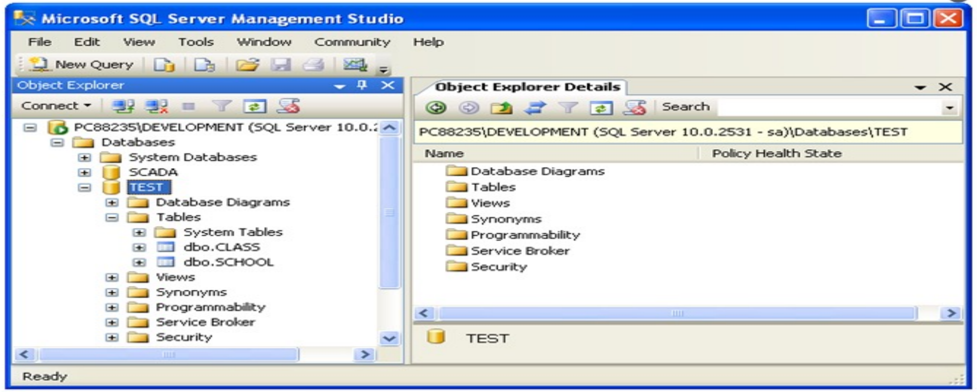
Microsoft SQL Server

SQL Server consists of a **Database Engine** and a **Management Studio**. The **Database Engine** has no graphical interface - it is just a service running in the background of your computer (preferable on the server). The **Management Studio** is graphical tool for configuring and viewing the information in the database. It can be installed on the server or on the client (or both).

Database Engine

A Service running on the computer in the background

Management Studio



A Graphical User Interface to the database used for configuration and management of the database

https://www.halvorsen.blog

Database Design

Hans-Petter Halvorsen

Table of Contents

Database Design

- Everybody needs to install erwin Data Modeler. It is required both in the project and in the final exam.
- Design your Database (you may start with a simple sketch using pen and paper).
- Create an ER diagram (Entity Relationship) using erwin Data Modeler
- Use “Academic Edition” (free of charge)
- The Database will be used by all the Modules in your Software, so the Team should do the Database Design session together!

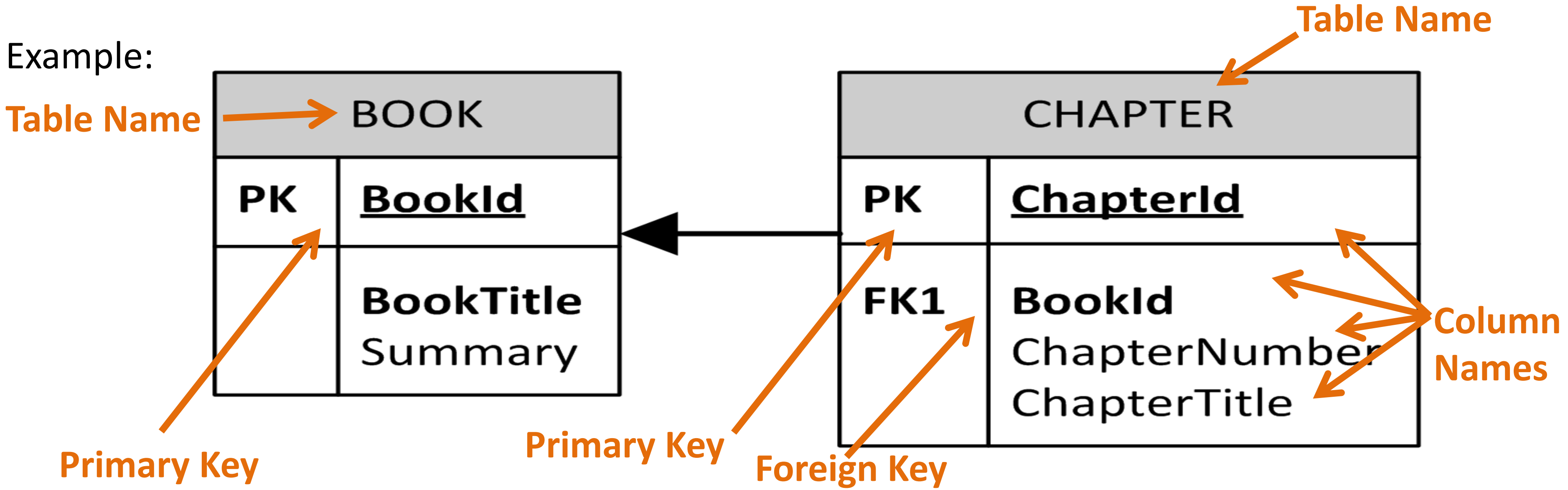
See Next Slides for more details...

Database Design – ER Diagram

ER Diagram (Entity-Relationship Diagram)

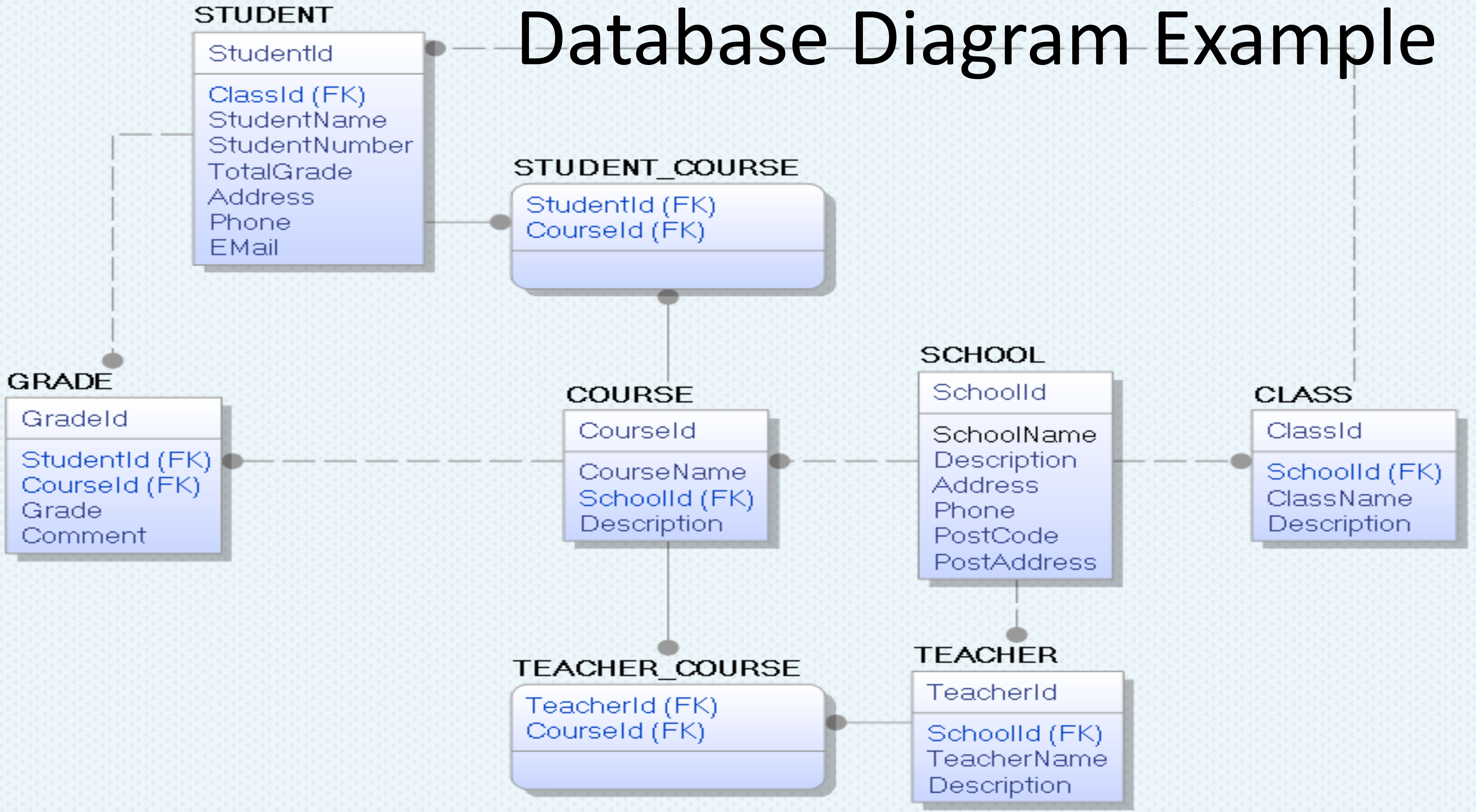
- Used for Design and Modeling of Databases.
- Specify Tables and **relationship** between them (**Primary Keys** and **Foreign Keys**)

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.

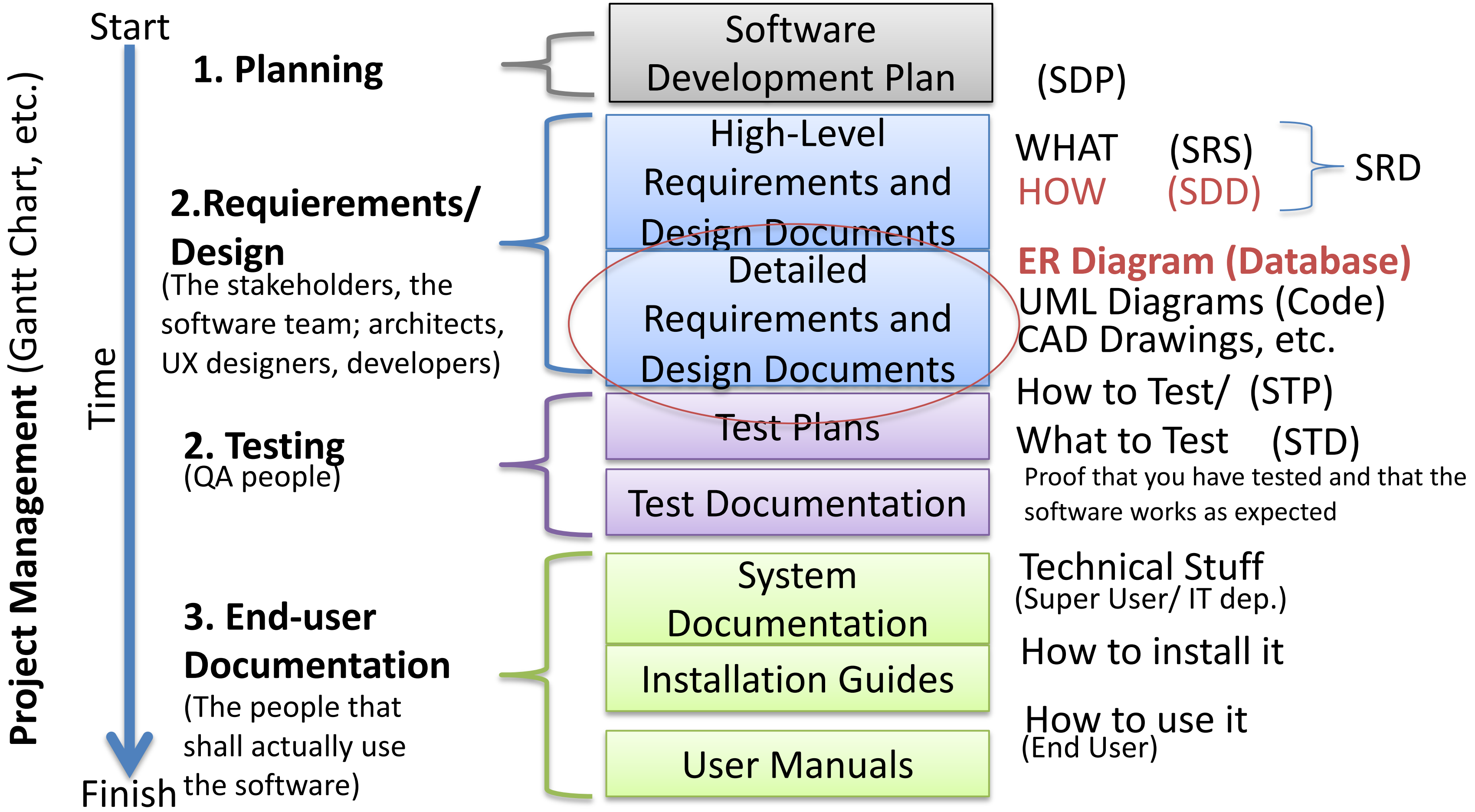
Database Diagram Example



Database - “Best Practice”

- **Tables**: Use upper case and singular form in table names – not plural, e.g., “STUDENT” (not students)
- **Columns**: Use Pascal notation, e.g., “StudentId”
- **Primary Key**:
  - If the table name is “COURSE”, name the Primary Key column “CourseId”, etc.
  - “Always” use Integer and Identity(1,1) for Primary Keys. Use UNIQUE constraint for other columns that needs to be unique, e.g. RoomNumber
- Specify **Required** Columns (NOT NULL) – i.e., which columns that need to have data or not
- Standardize on few/these **Data Types**: *int*, *float*, *varchar(x), datetime*, *bit*
- Use English for table and column names
- Avoid abbreviations! (Use RoomNumber – not RoomNo, RoomNr, ...)

Typical Software Documentation


Project Management (Gantt Chart, etc.)
Start
Time
Finish
1. Planning
2.Requierements/ Design
(The stakeholders, the software team; architects, UX designers, developers)
2. Testing
(QA people)
3. End-user Documentation
(The people that shall actually use the software)
Software Development Plan
High-Level Requirements and Design Documents
Detailed Requirements and Design Documents
Test Plans
Test Documentation
System Documentation
Installation Guides
User Manuals
(SDP)
WHAT (SRS)
HOW (SDD)
SRD
ER Diagram (Database)
UML Diagrams (Code)
CAD Drawings, etc.
How to Test/ (STP)
What to Test (STD)
Proof that you have tested and that the software works as expected
Technical Stuff
(Super User/ IT dep.)
How to install it
How to use it
(End User)

Database Implementation

Hans-Petter Halvorsen

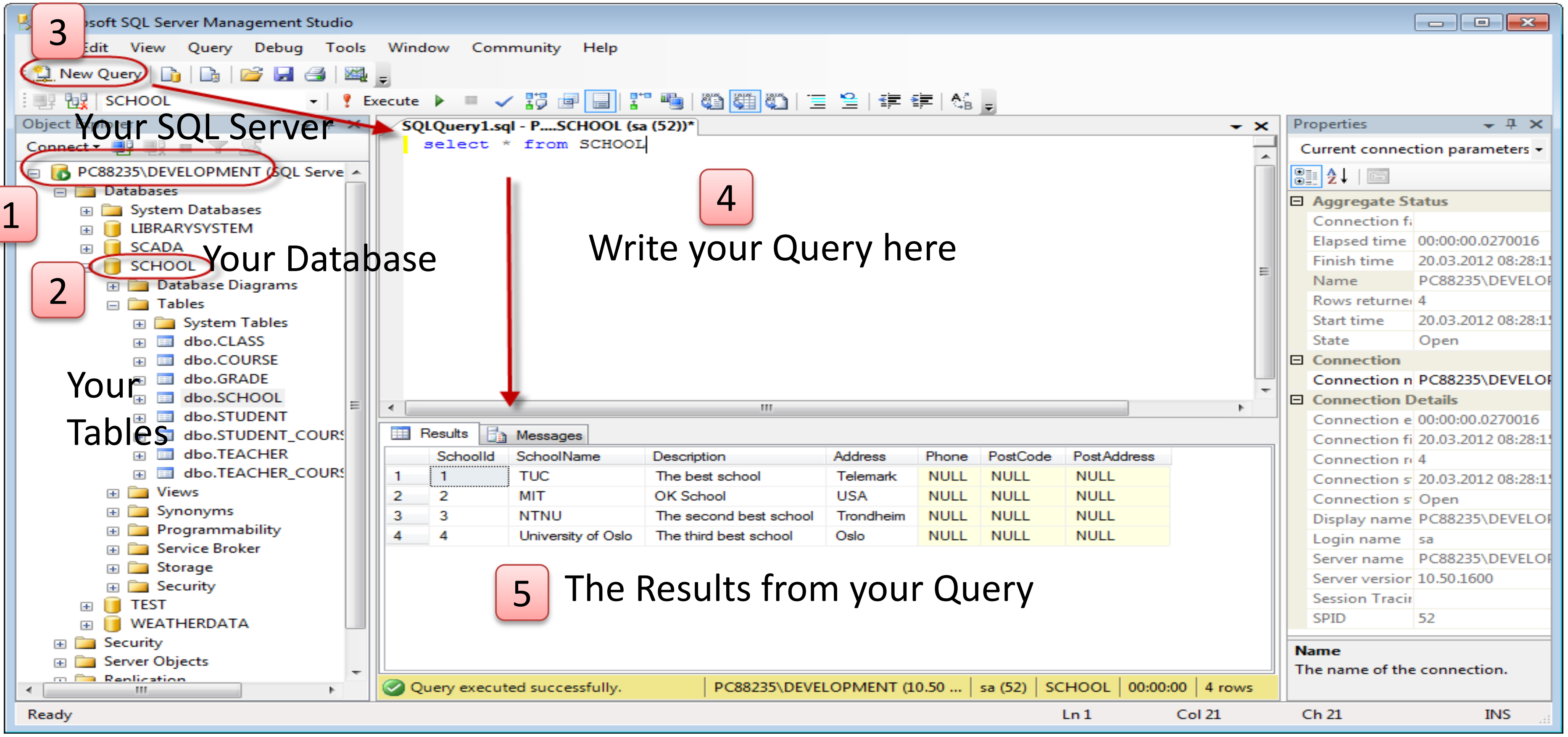
Table of Contents

Database Implementation

- Create **SQL Scripts** and Implement the database tables in **SQL Server**. You may use erwin Data Modeler in order to generate such a SQL Script
- The **SQL scripts** for your tables should be uploaded to Azure DevOps.
- Create the tables using a Script makes it easy to create the necessary tables on another computer (Development Environment, Test Environment, Production Environment) -> Deployment
- Start Create a Database API

See Next Slides for more details...

Microsoft SQL Server



Microsoft SQL Server Management Studio

```
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

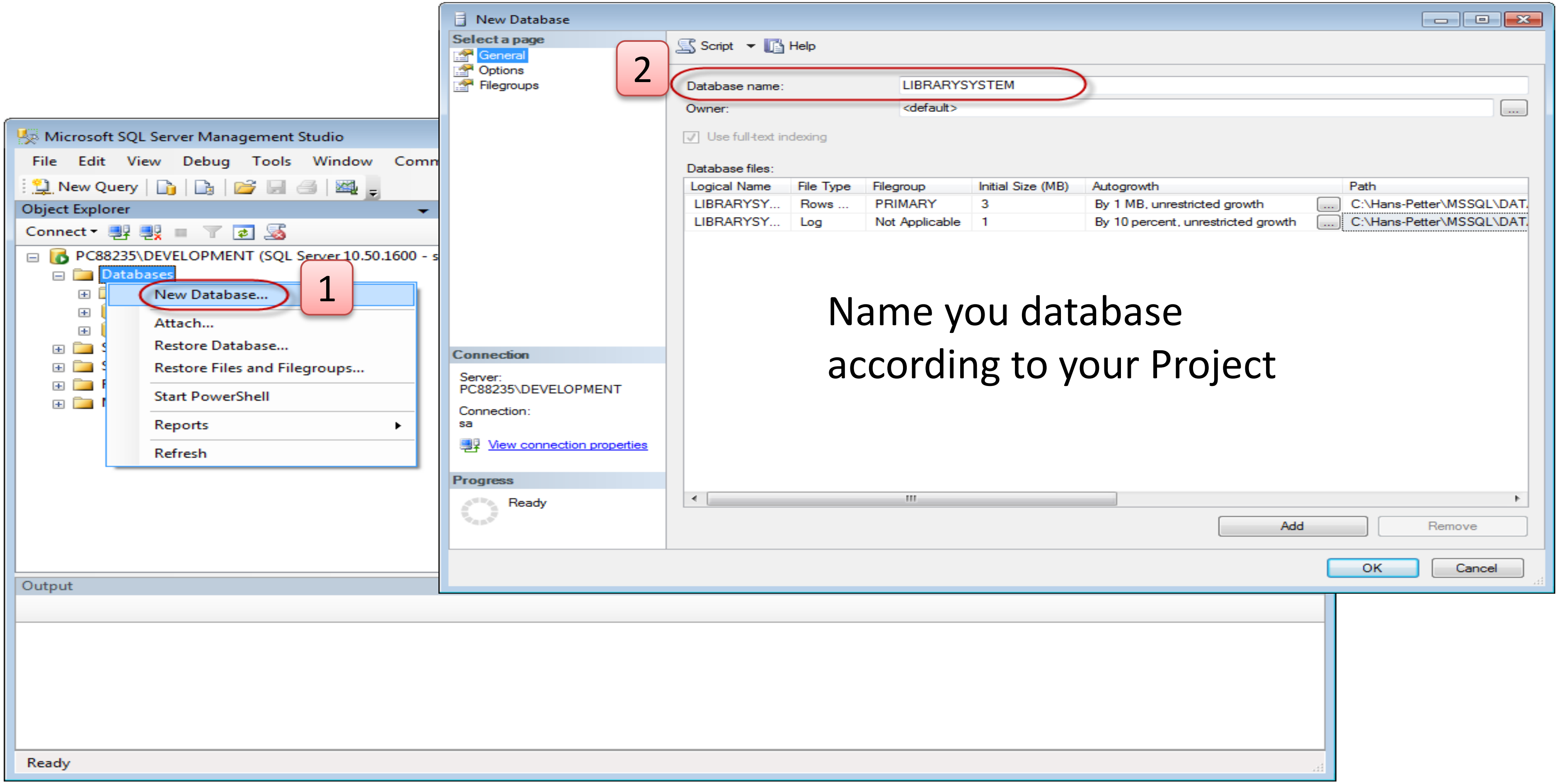
Query executed successfully. | PC88235\DEVELOPMENT (10.50 ... | sa (52) | SCHOOL | 00:00:00 | 4 rows

Properties: Current connection parameters

Aggregate Status	
Connection fa	
Elapsed time	00:00:00.0270016
Finish time	20.03.2012 08:28:1
Name	PC88235\DEVELOP
Rows returne	4
Start time	20.03.2012 08:28:1
State	Open
Connection	
Connection n	PC88235\DEVELOP
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\DEVELOP
Login name	sa
Server name	PC88235\DEVELOP
Server versior	10.50.1600
Session Tracir	
SPID	52

Name
The name of the connection.

Microsoft SQL Server – Create a New Database



Database name: LIBRARYSYSTEM

Owner: <default>

Use full-text indexing

Server: PC88235\DEVELOPMENT

Connection: sa

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.

Name you database according to your Project

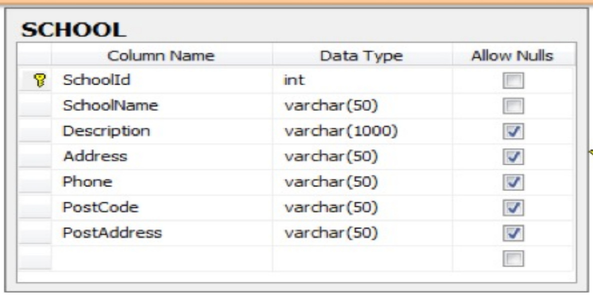
SQL Script Example

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
```

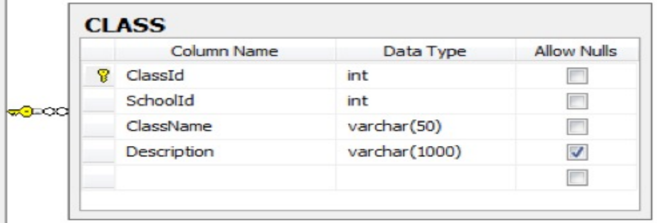
```
if not exists (select * from dbo.sysobjects where id = object_id(N'[CLASS]') and OBJECTPROPERTY(id, N'IsUserTab
CREATE TABLE [CLASS]
(
          [ClassId] [int] IDENTITY(1, 1) NOT NULL PRIMARY KEY,
          [SchoolId] [int] NOT NULL FOREIGN KEY REFERENCES [SCHOOL] ([SchoolId]),
          [ClassName]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

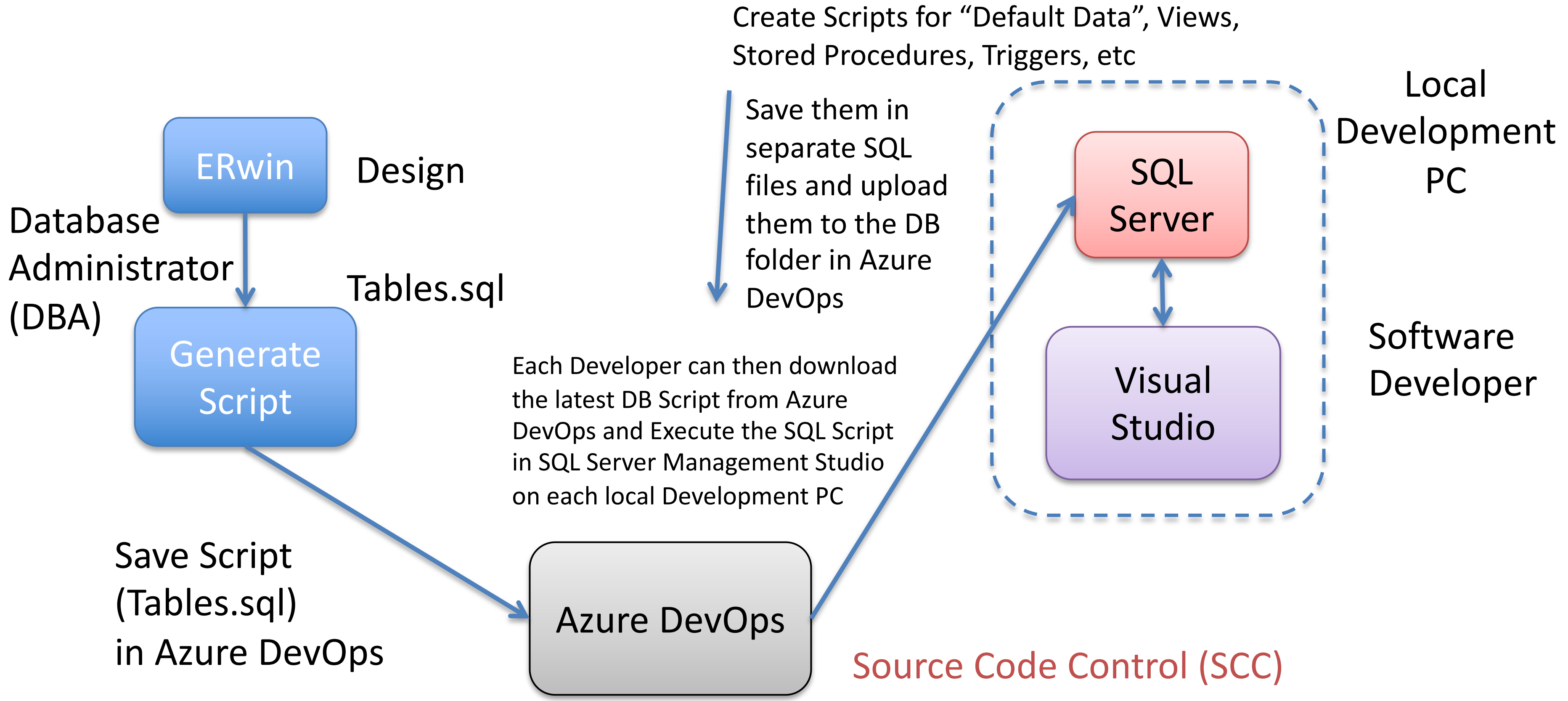
```
if not exists (select * from dbo.sysobjects where id = object_id(N'[CUSTOMER]') and OBJECTPROPERTY(id, 
N'IsUserTable') = 1)
CREATE TABLE CUSTOMER
(
      CustomerId int PRIMARY KEY,
      CustomerNumber int NOT NULL UNIQUE,
      LastName varchar(50) NOT NULL,
      FirstName varchar(50) NOT NULL,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

Save/Update Scripts to Azure DevOps

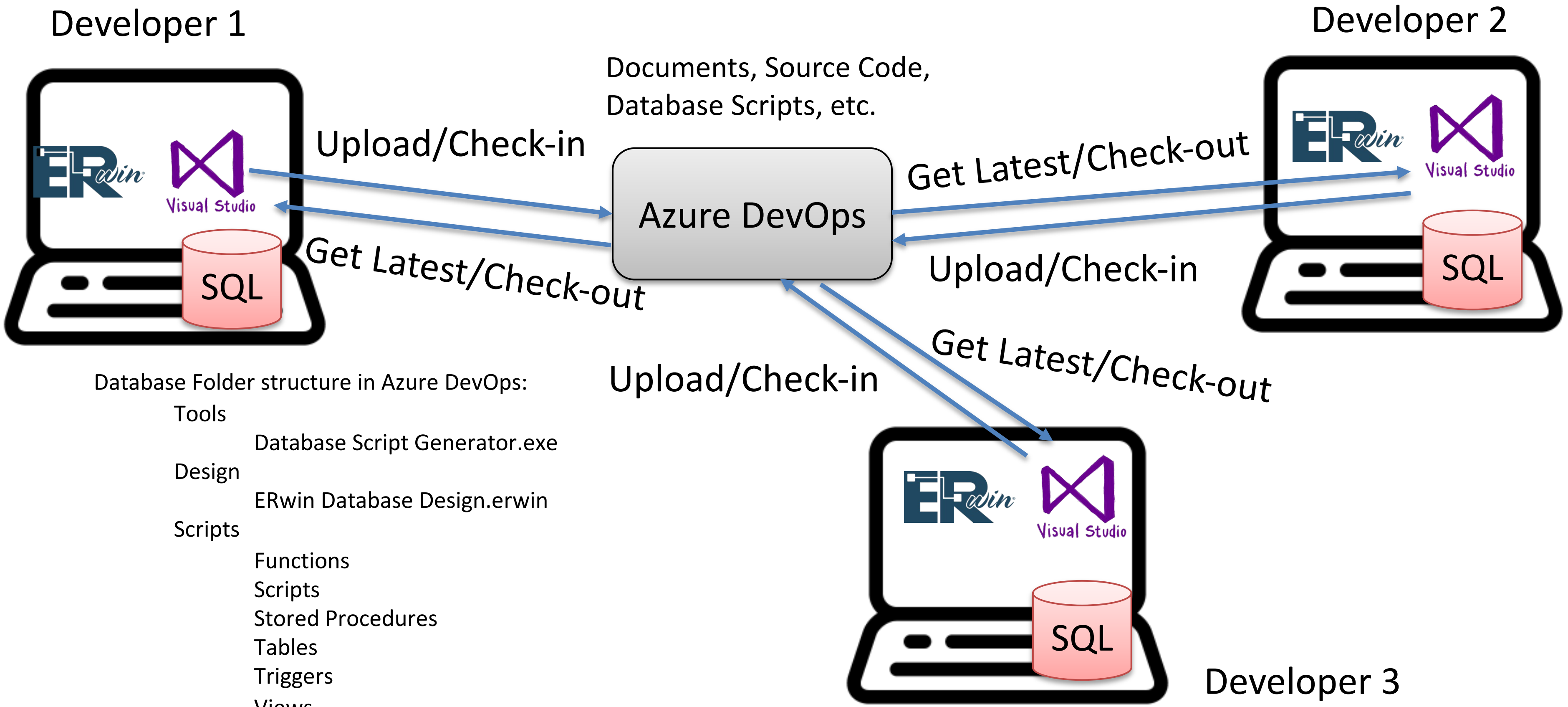


Create Scripts for "Default Data", Views, Stored Procedures, Triggers, etc
Save them in separate SQL files and upload them to the DB folder in Azure DevOps
ERwin
Design
Database Administrator (DBA)
Tables.sql
Generate Script
Each Developer can then download the latest DB Script from Azure DevOps and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

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

Database Development

Use Visual Studio/Azure DevOps to Synchronize (Get Latest, Check-in/Check-out) Documents, Code files, Database files, etc.

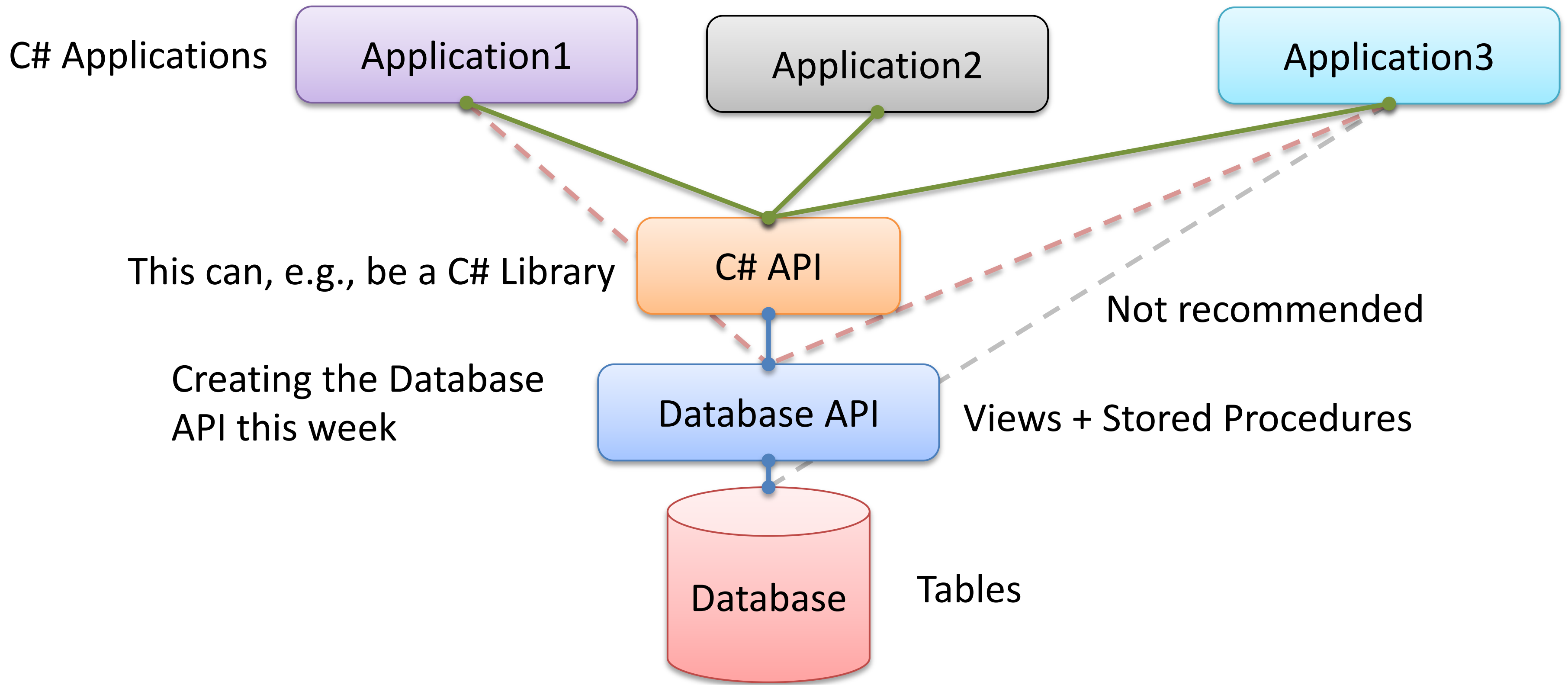


Database Folder structure in Azure DevOps:

- Tools
  - Database Script Generator.exe
- Design
  - ERwin Database Design.erwin
- Scripts
  - Functions
  - Scripts
  - Stored Procedures
  - Tables
  - Triggers
  - Views

Database API

It is recommended that you start creating a simple Database API, this means creating some common Views and Stored Procedures that can be shared and used by the C# code.



Coding and Implementation

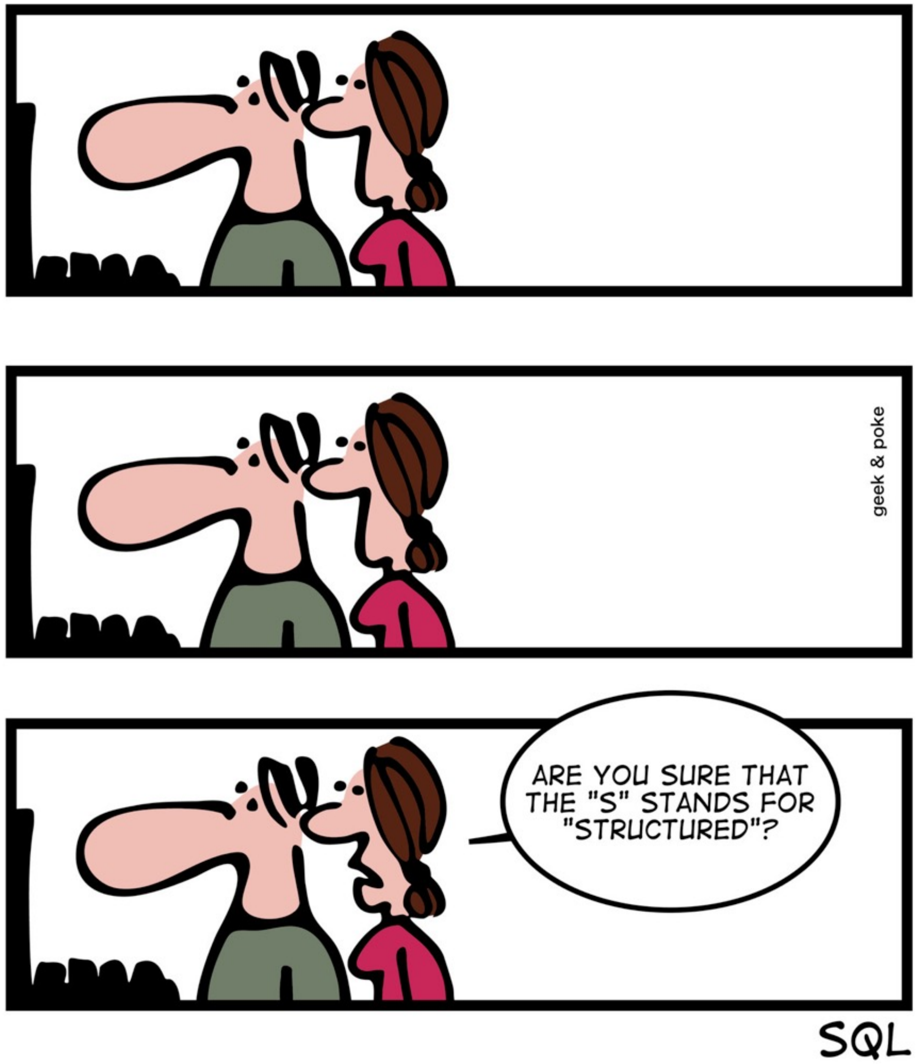
Hans-Petter Halvorsen

Table of Contents

Coding and Implementation

- Install necessary Software
- Get an overview of the software platforms, programming languages you shall use, etc. (update SDP and SRD documents if necessary)
- **Create a good structure for your code.**
- **Start planning and Implementing the code structure of your application.**
- **Start creating the main shell for your application (both code and GUI).** Preferably make an ASP.NET Application
- **Test that you are able to communicate with the Database**
- Make sure to upload/check-in code to Azure DevOps
- It is important that we have a working software at all times (so it can be reviewed, tested, etc. during the whole project)!

See Next Slides for more details...



http://geek-and-poke.com

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

Web: https://www.halvorsen.blog