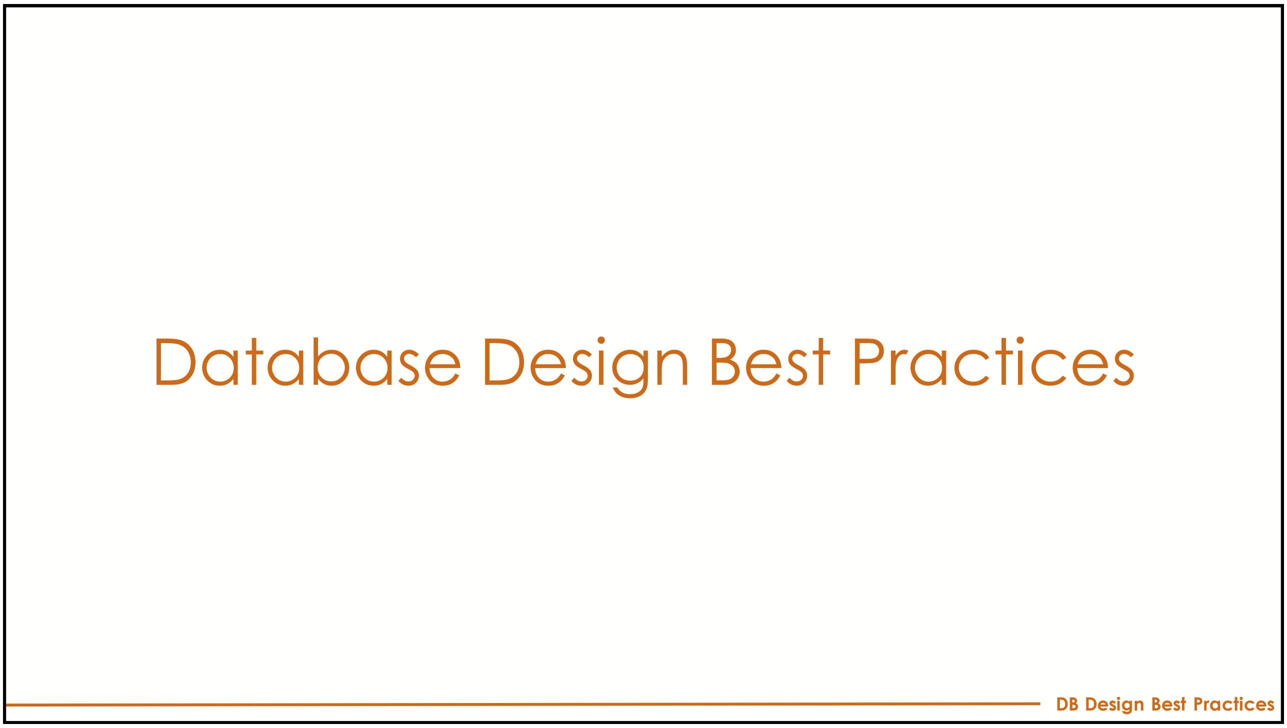
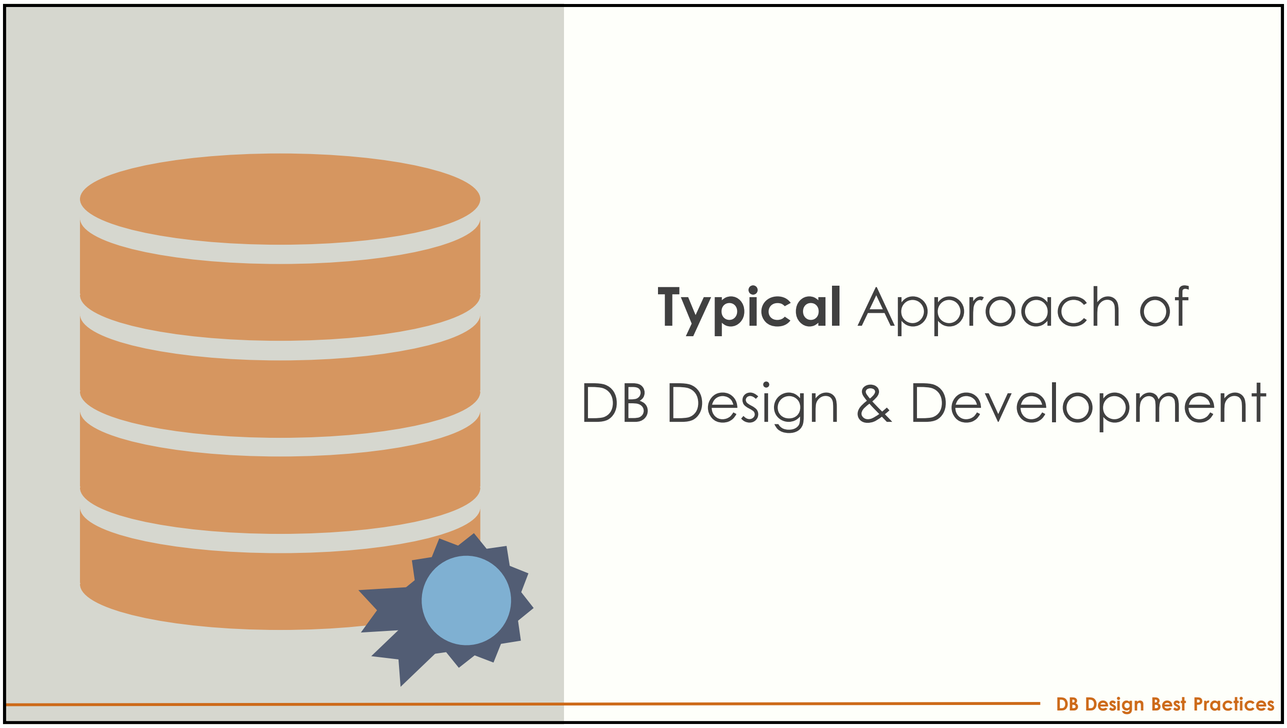


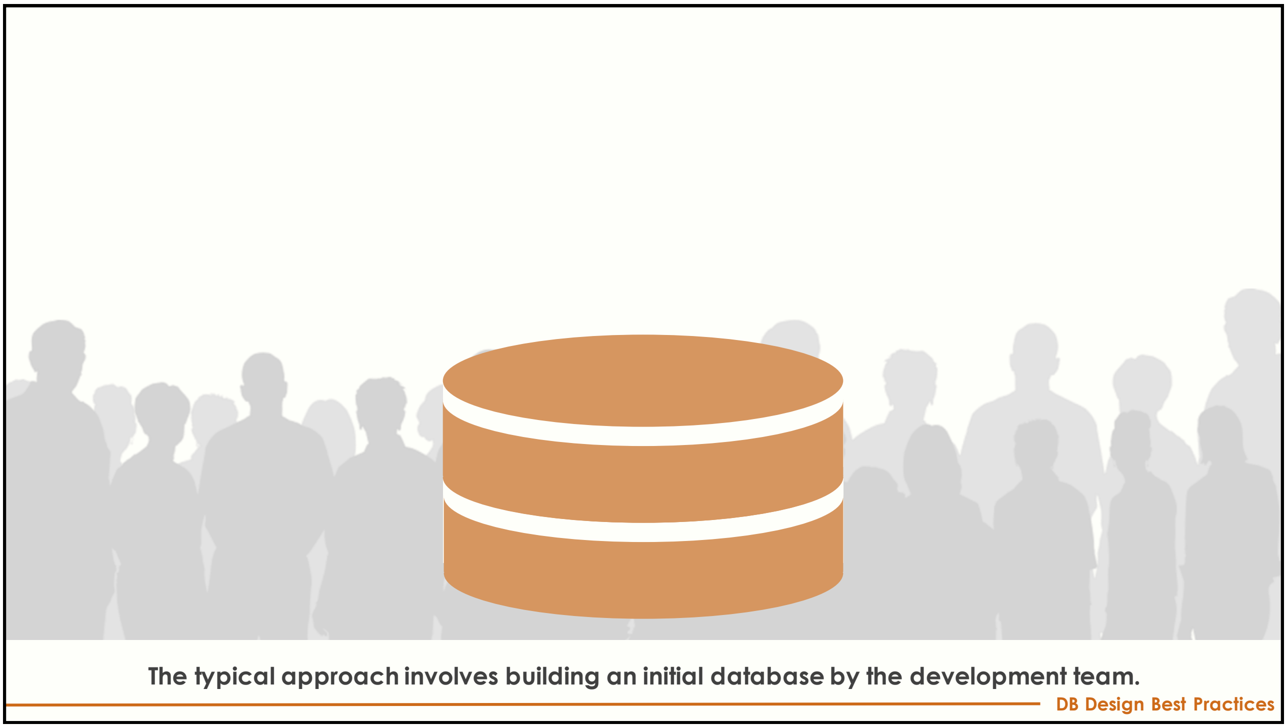
Database Design Best Practices

DB Design Best Practices



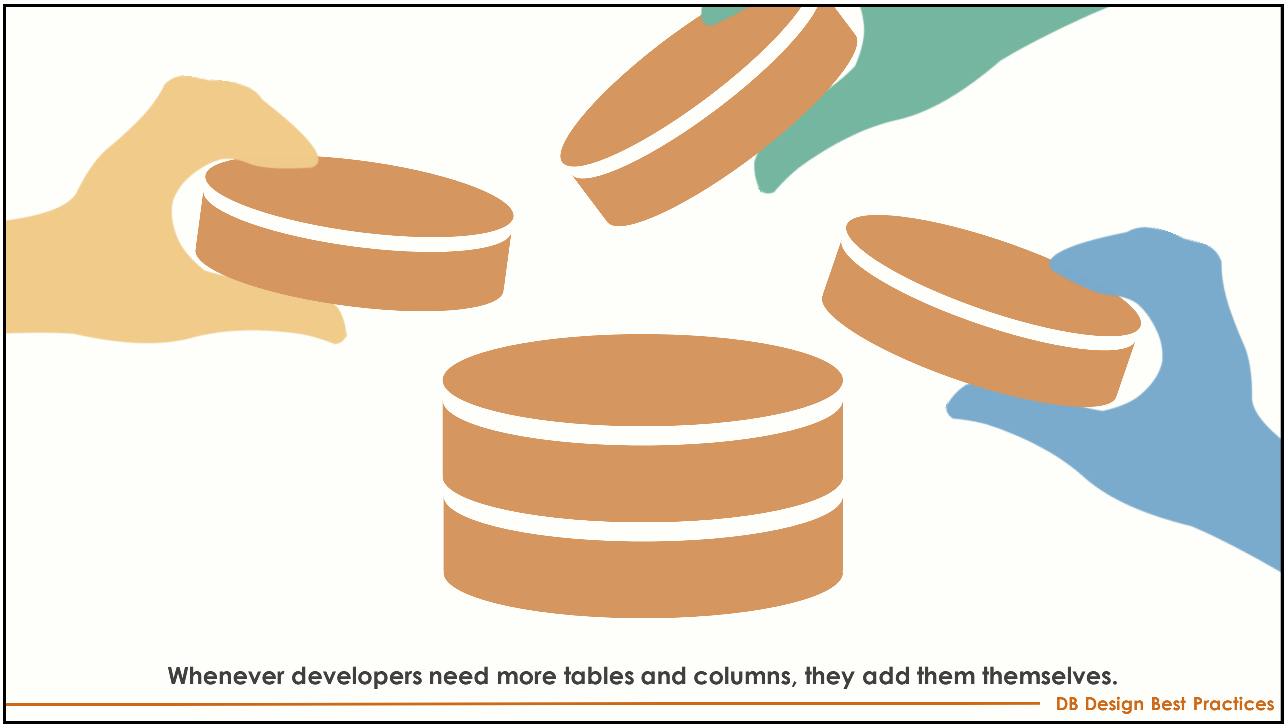
Typical Approach of DB Design & Development

DB Design Best Practices



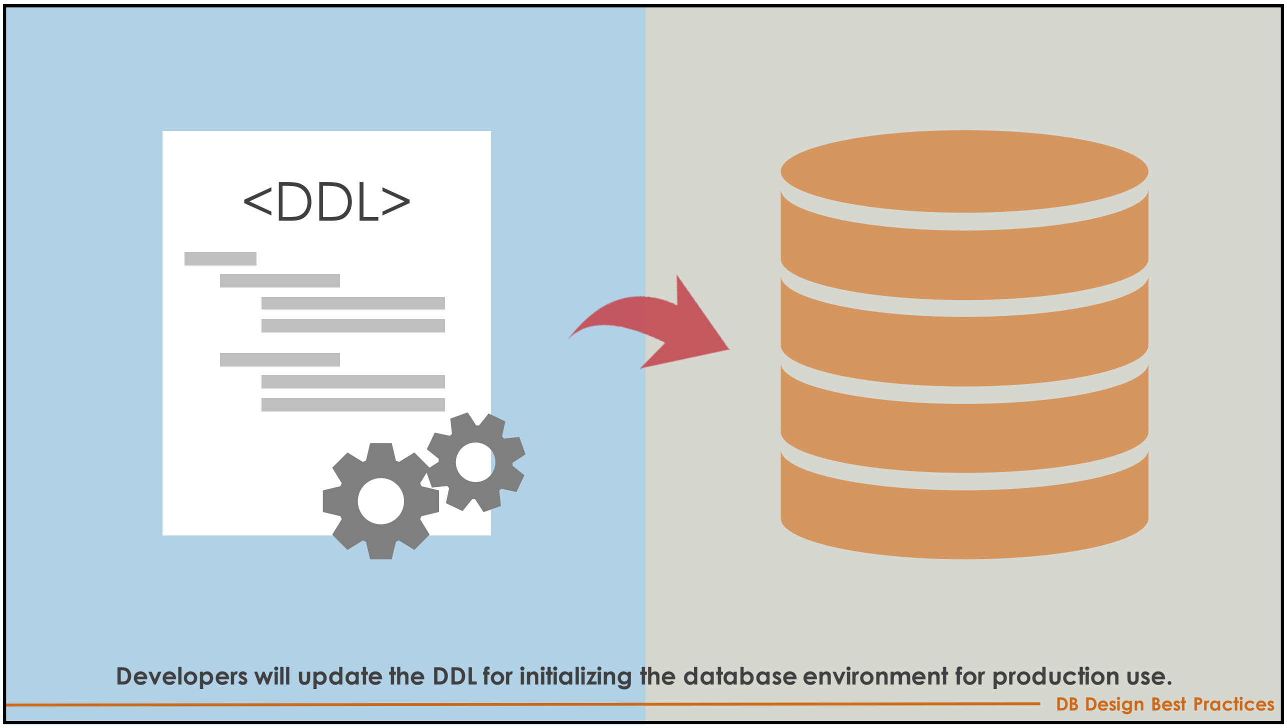
The typical approach involves building an initial database by the development team.

DB Design Best Practices



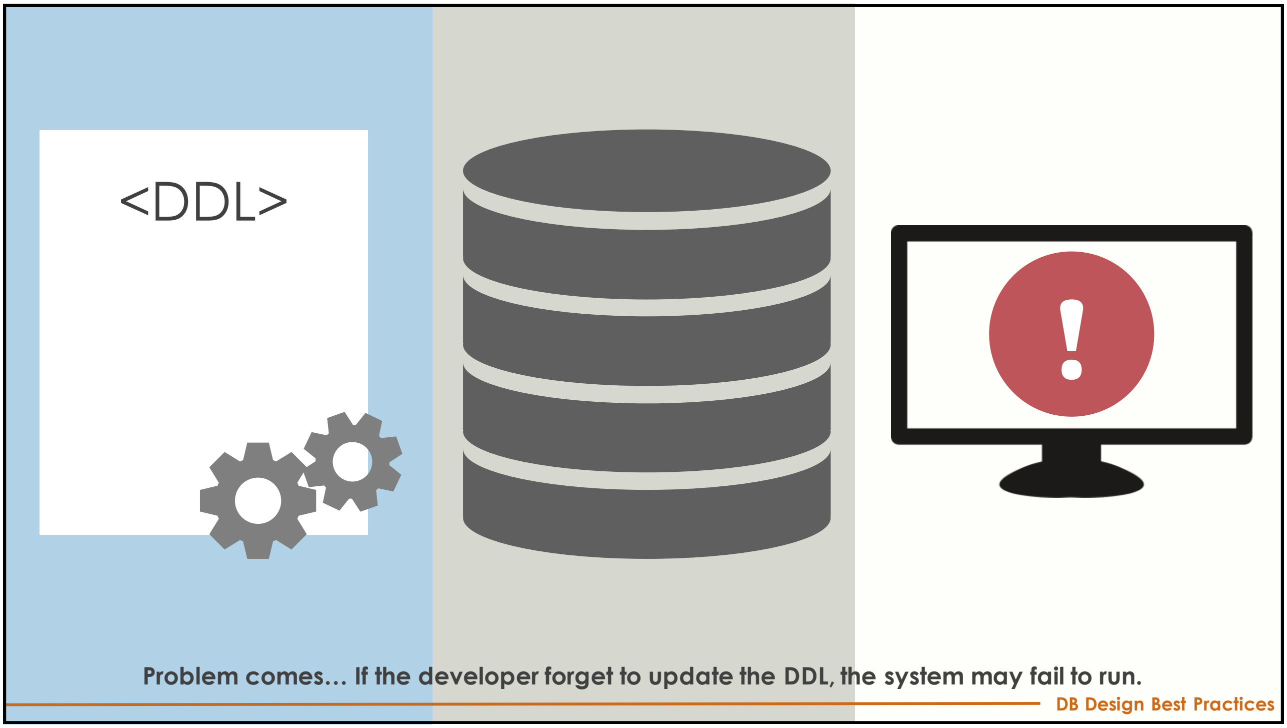
Whenever developers need more tables and columns, they add them themselves.

DB Design Best Practices



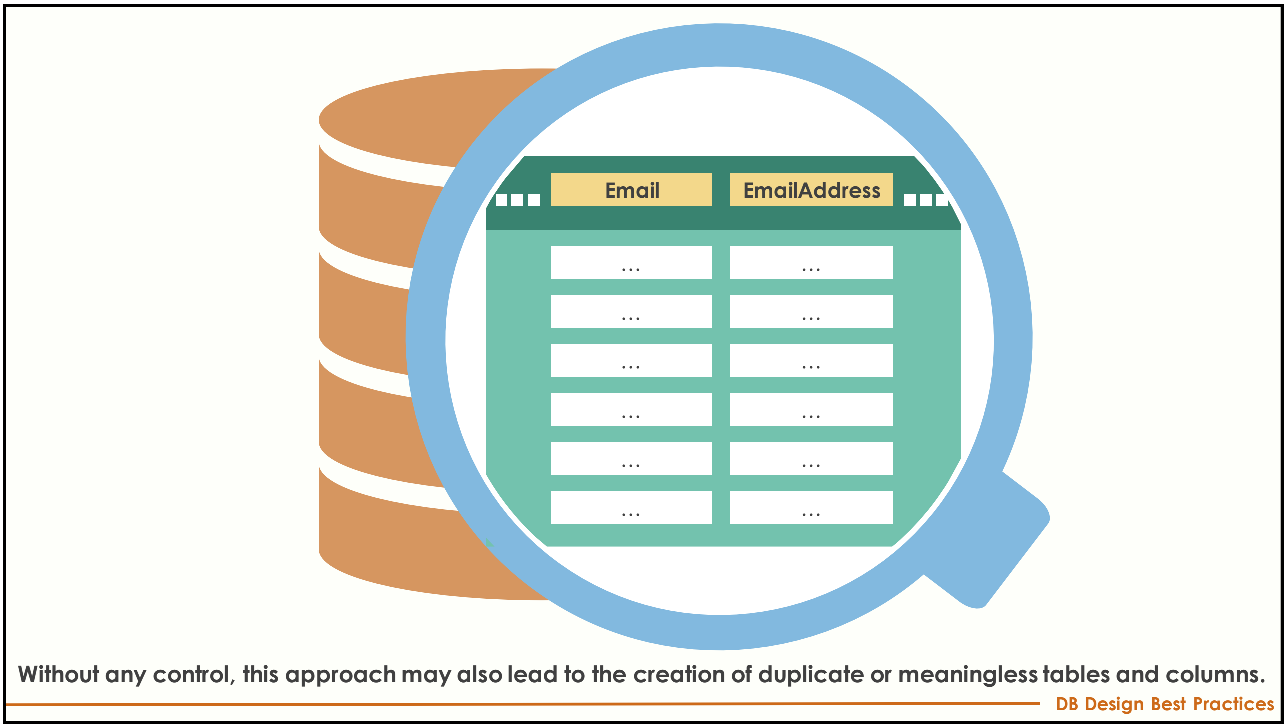
Developers will update the DDL for initializing the database environment for production use.

DB Design Best Practices



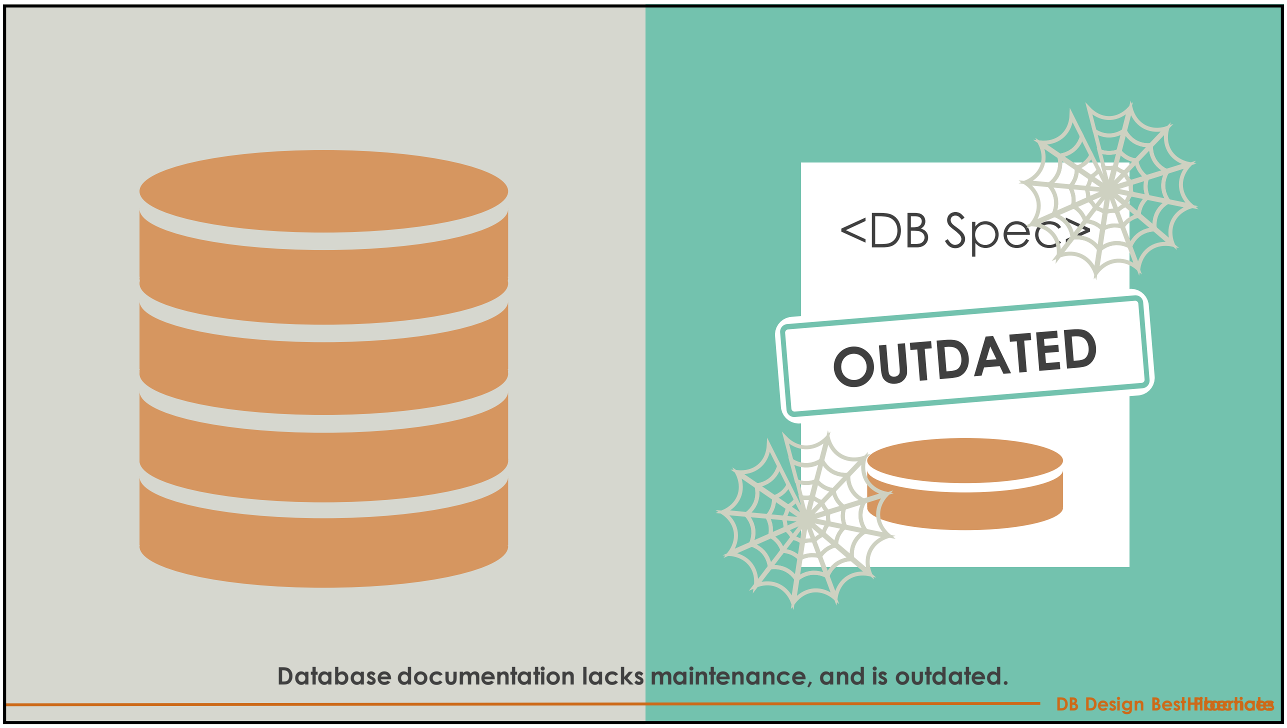
Problem comes… If the developer forget to update the DDL, the system may fail to run.

DB Design Best Practices



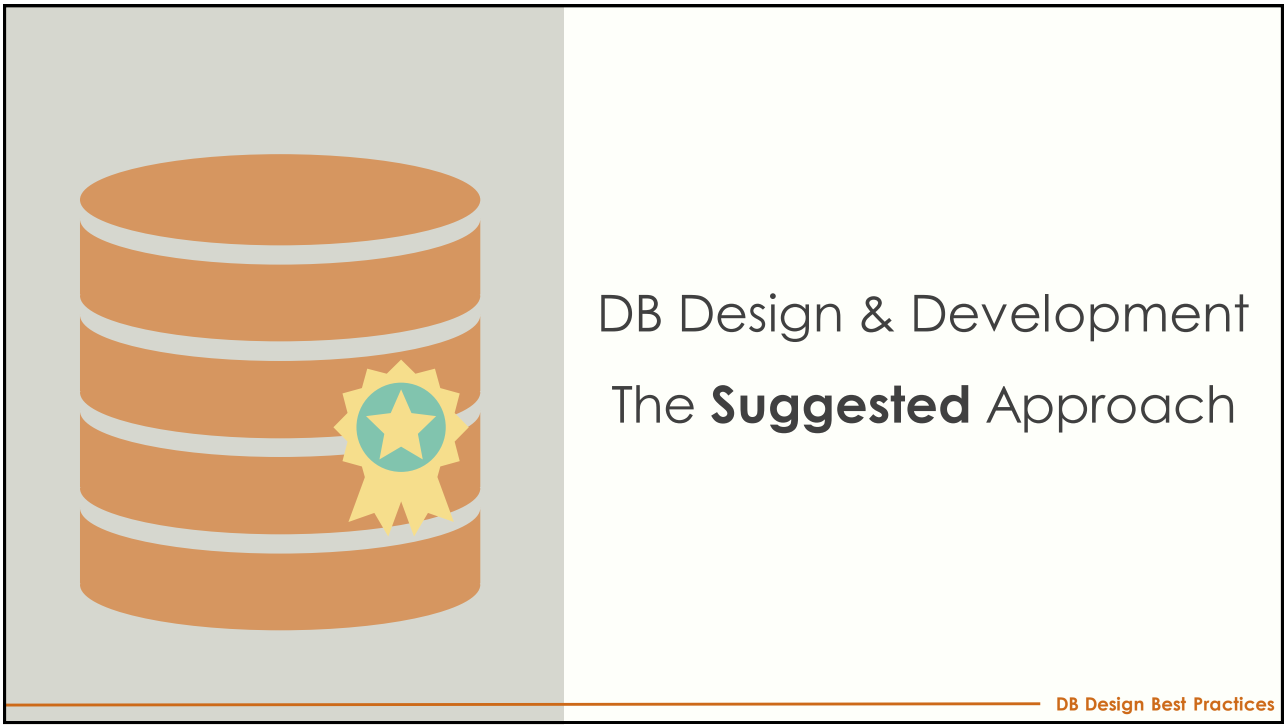
Without any control, this approach may also lead to the creation of duplicate or meaningless tables and columns.

DB Design Best Practices



Database documentation lacks maintenance, and is outdated.

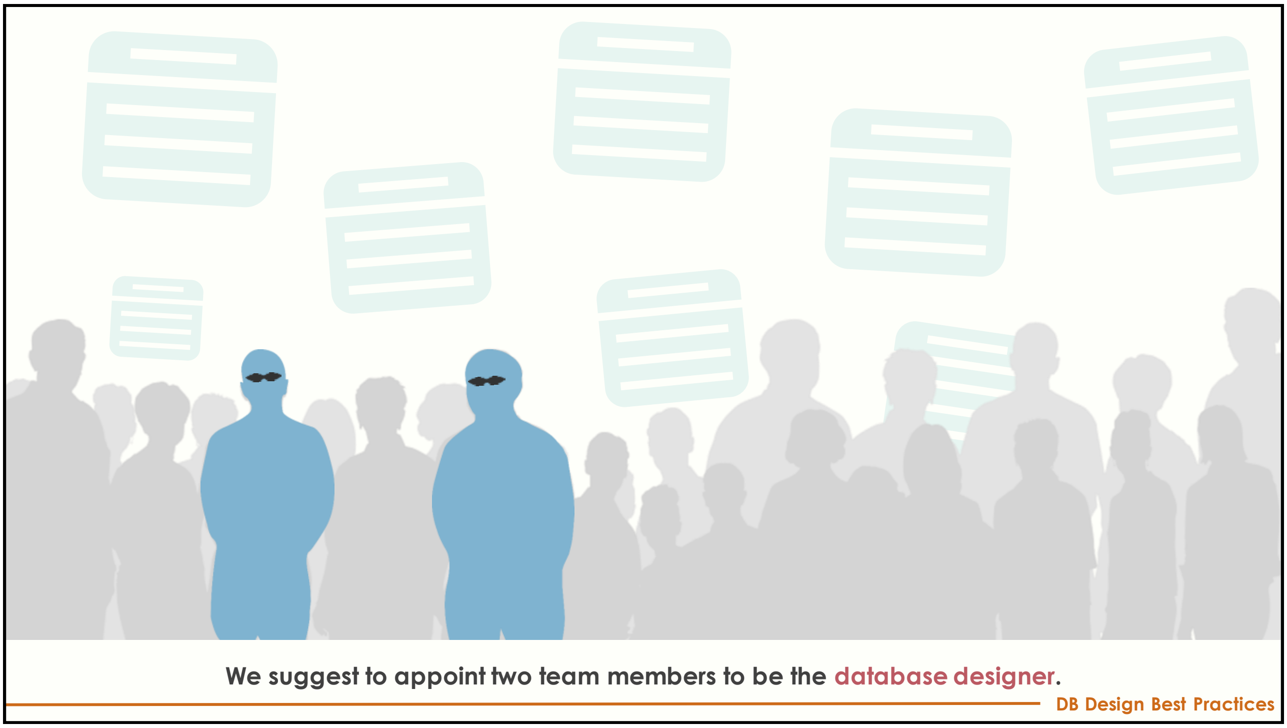
DB Design Best Practices



DB Design & Development

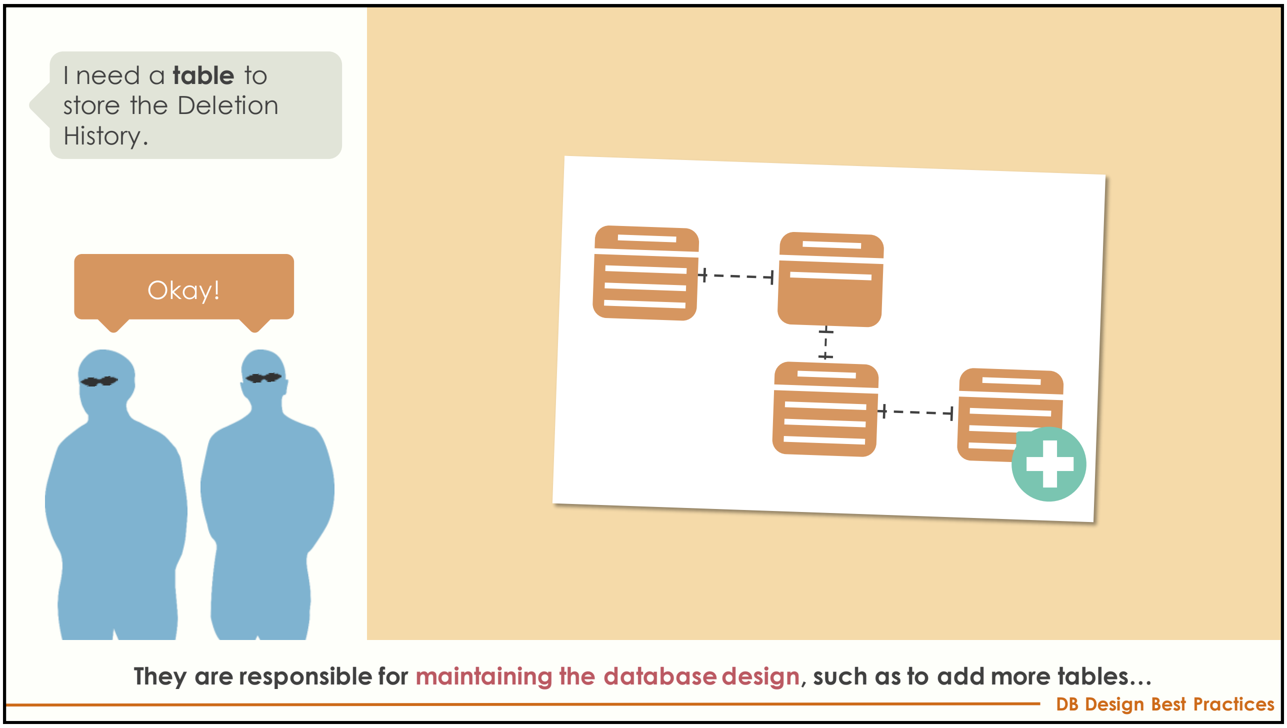
The **Suggested** Approach

DB Design Best Practices



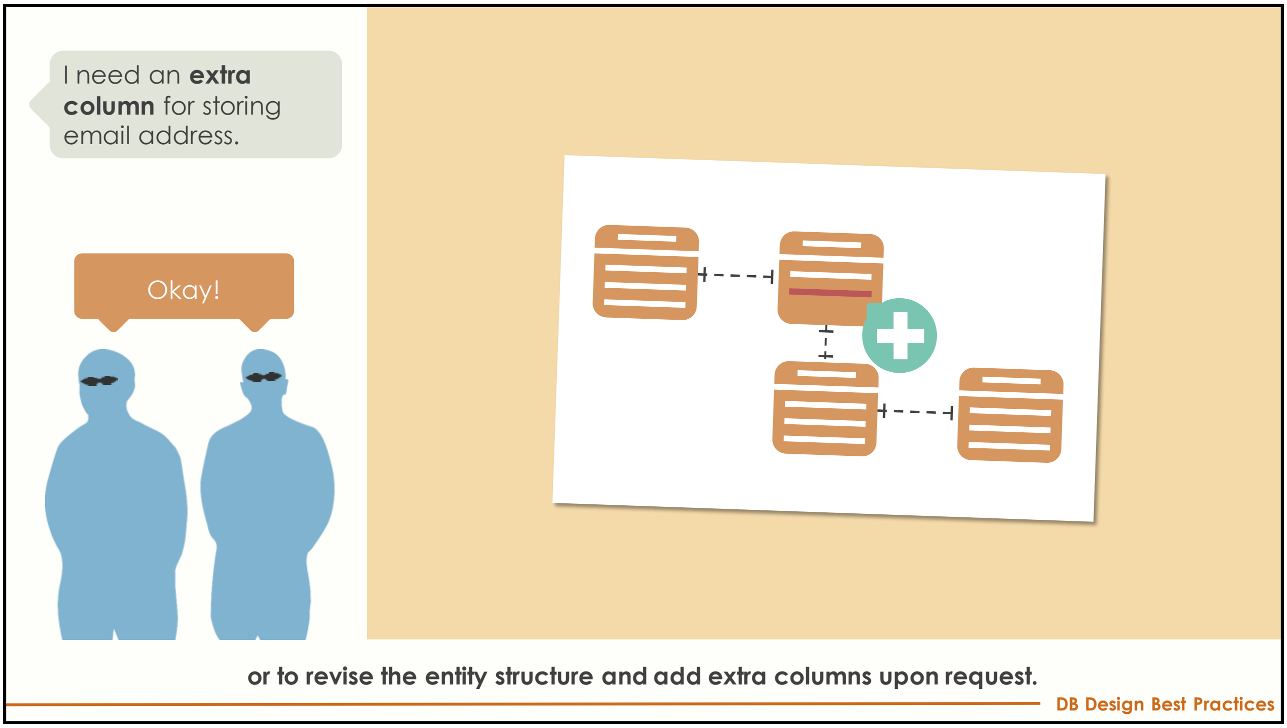
We suggest to appoint two team members to be the database designer.

DB Design Best Practices



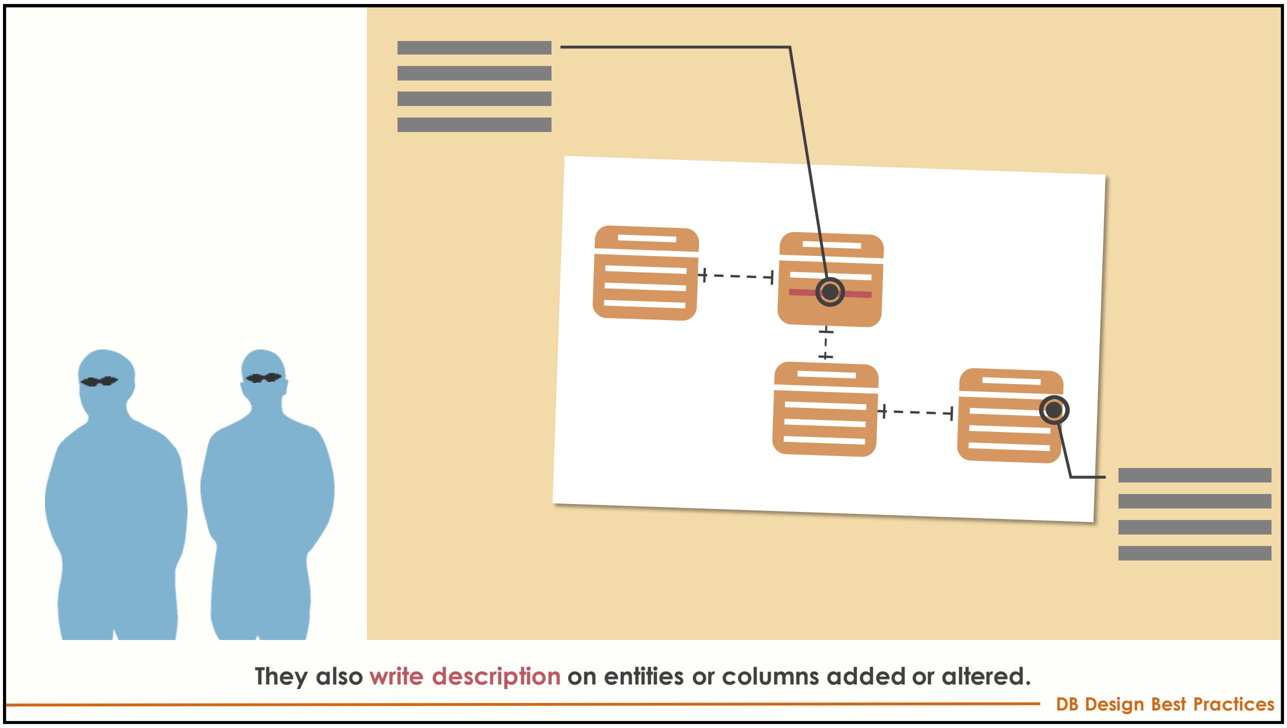
They are responsible for maintaining the database design, such as to add more tables…

DB Design Best Practices



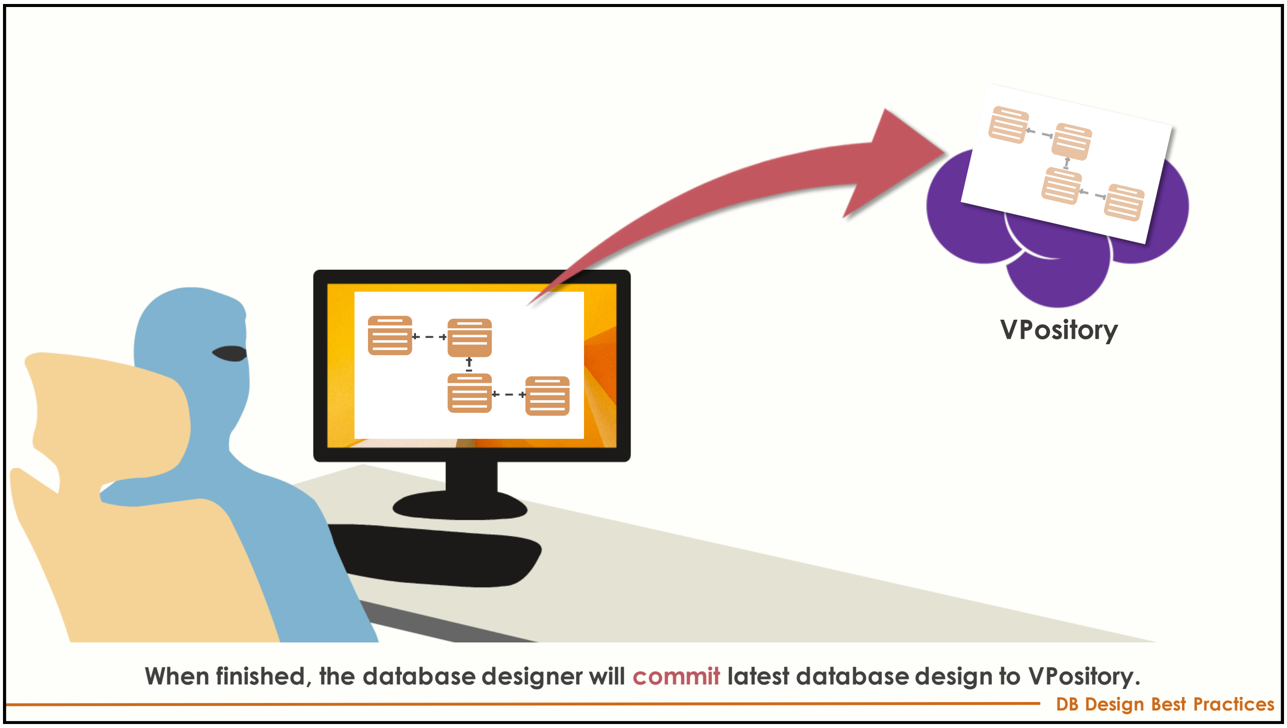
or to revise the entity structure and add extra columns upon request.

DB Design Best Practices



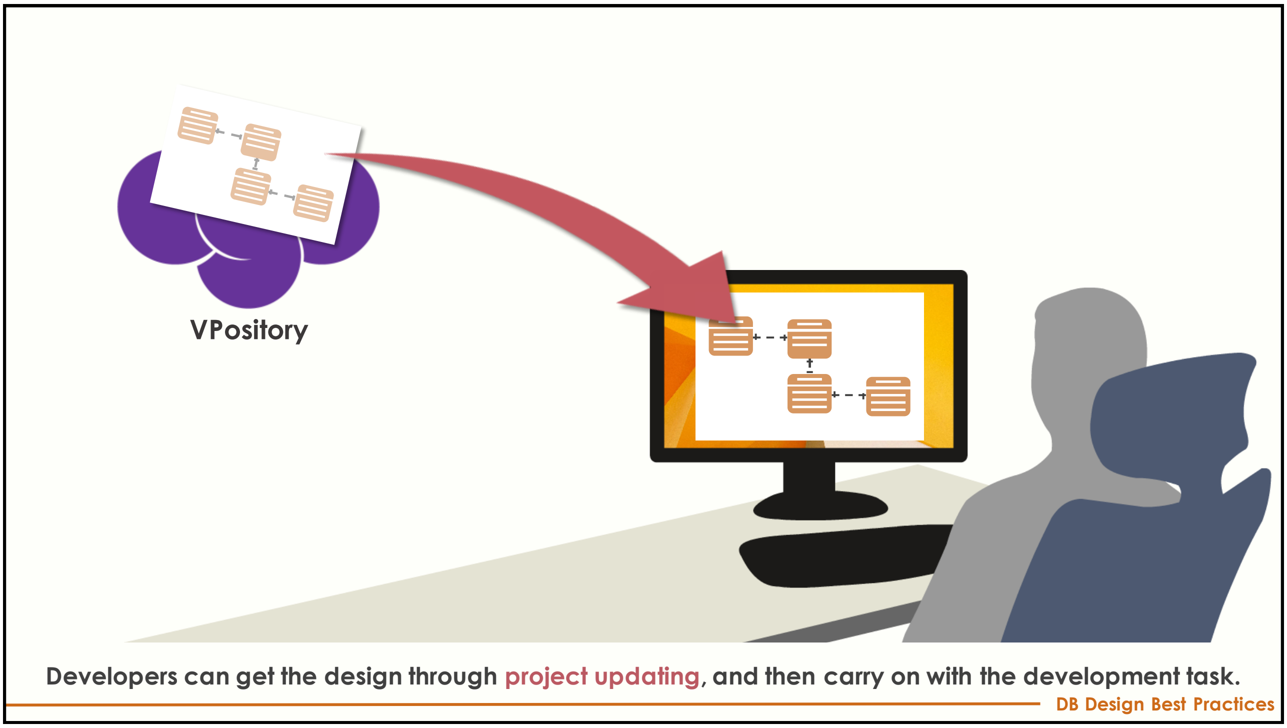
They also write description on entities or columns added or altered.

DB Design Best Practices



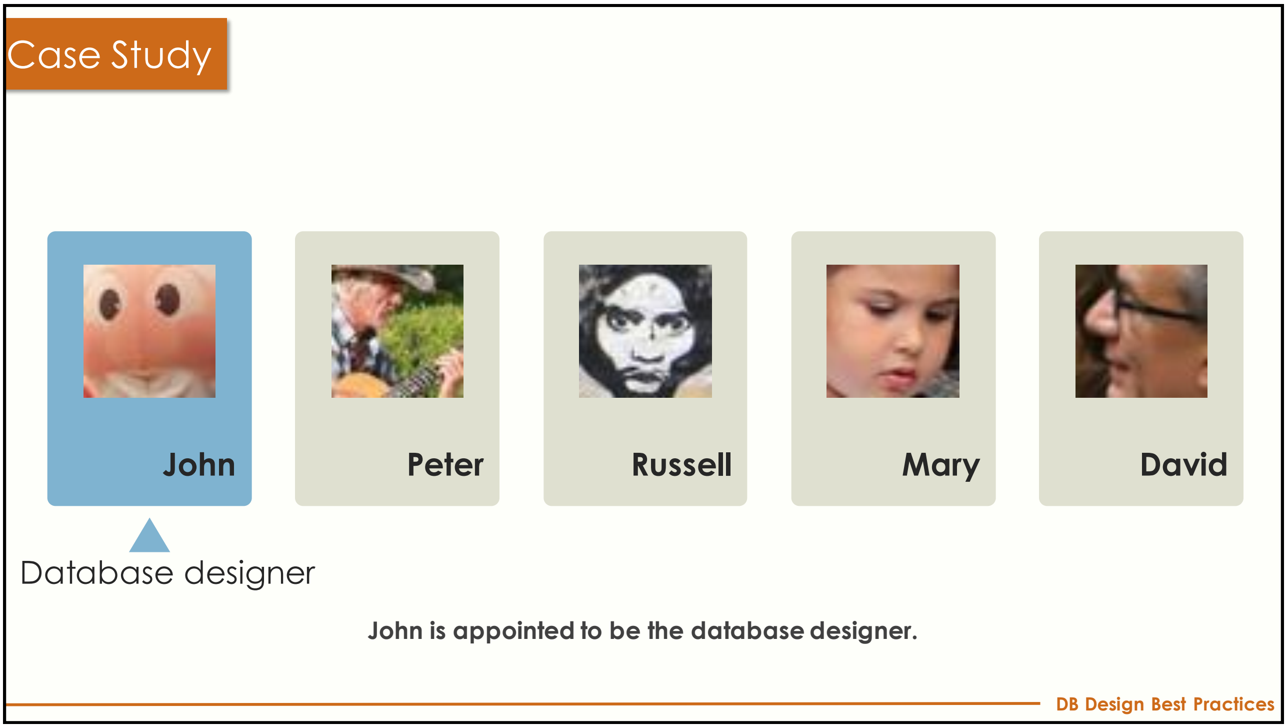
When finished, the database designer will commit latest database design to VPository.

DB Design Best Practices



Developers can get the design through project updating, and then carry on with the development task.

DB Design Best Practices



Case Study

Database designer

John is appointed to be the database designer.

DB Design Best Practices

Case Study

Peter, a developer needs an extra column for storing email addresses.

DB Design Best Practices

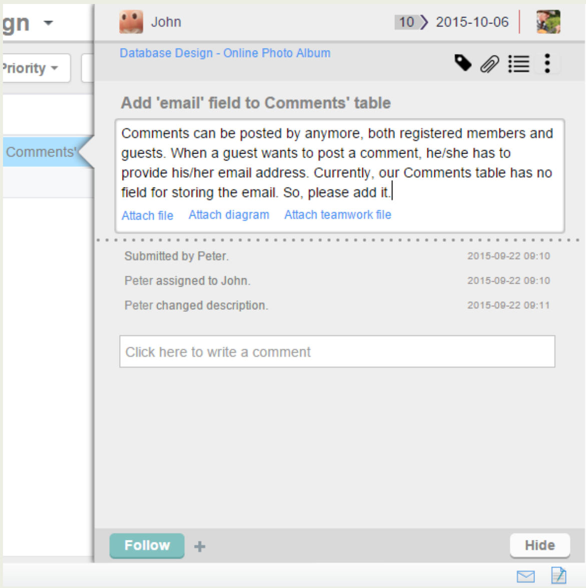
Case Study

Let’s see how John will work with Peter to update the database design

DB Design Best Practices

DB Design & Dev – The Steps

1. **Peter** creates a task to John in Tasifier to request for the change.



Add 'email' field to Comments' table

Comments can be posted by anymore, both registered members and guests. When a guest wants to post a comment, he/she has to provide his/her email address. Currently, our Comments table has no field for storing the email. So, please add it.

Attach file   Attach diagram   Attach teamwork file

Submitted by Peter. 2015-09-22 09:10

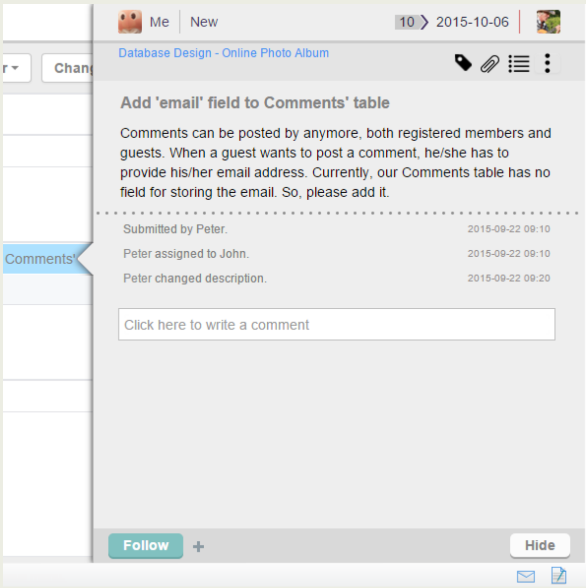
Peter assigned to John. 2015-09-22 09:10

Peter changed description. 2015-09-22 09:11

Click here to write a comment

DB Design & Dev – The Steps

1. **Peter** creates a task to John in Tasifier to request for the change.
2. **John** opens the task to study Peter's request.



Add 'email' field to Comments' table

Comments can be posted by anymore, both registered members and guests. When a guest wants to post a comment, he/she has to provide his/her email address. Currently, our Comments table has no field for storing the email. So, please add it.

Submitted by Peter. 2015-09-22 09:10

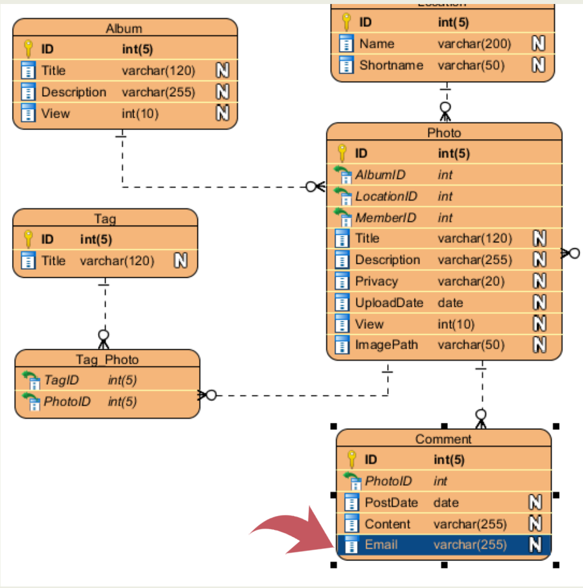
Peter assigned to John. 2015-09-22 09:10

Peter changed description. 2015-09-22 09:20

Click here to write a comment

DB Design & Dev – The Steps

1. **Peter** creates a task to John in Tasifier to request for the change.
2. **John** opens the task to study Peter's request.
3. **John** modifies the ERD by adding the column Peter needed.



Album		
ID	int(5)	
Title	varchar(120)	N
Description	varchar(255)	N
View	int(10)	N

Location		
ID	int(5)	
Name	varchar(200)	N
Shortname	varchar(50)	N

Tag		
ID	int(5)	
Title	varchar(120)	N

Tag_Photo	
TagID	*int(5)*
PhotoID	*int(5)*

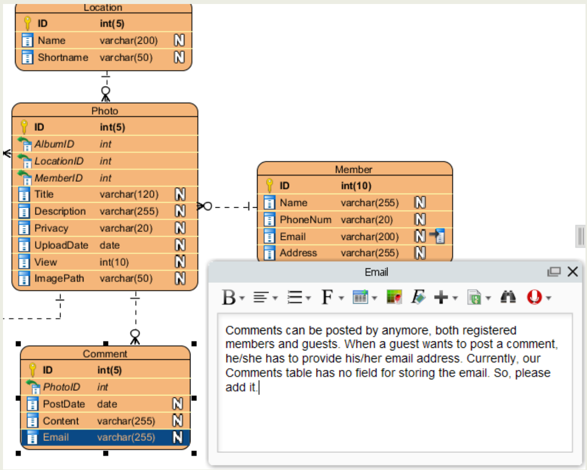
Photo		
ID	int(5)	
AlbumID	*int*	
LocationID	*int*	
MemberID	*int*	
Title	varchar(120)	N
Description	varchar(255)	N
Privacy	varchar(20)	N
UploadDate	date	N
View	int(10)	N
ImagePath	varchar(50)	N

Comment		
ID	int(5)	
PhotoID	*int*	
PostDate	date	N
Content	varchar(255)	N
Email	varchar(255)	N

DB Design Best Practices

DB Design & Dev – The Steps

1. **Peter** creates a task to John in Tasifier to request for the change.
2. **John** opens the task to study Peter's request.
3. **John** modifies the ERD by adding the column Peter needed.
4. **John** describes the change.



Location		
ID	int(5)	
Name	varchar(200)	N
Shortname	varchar(50)	N

Photo		
ID	int(5)	
AlbumID	*int*	
LocationID	*int*	
MemberID	*int*	
Title	varchar(120)	N
Description	varchar(255)	N
Privacy	varchar(20)	N
UploadDate	date	N
View	int(10)	N
ImagePath	varchar(50)	N

Member		
ID	int(10)	
Name	varchar(255)	N
PhoneNum	varchar(20)	N
Email	varchar(200)	N
Address	varchar(255)	N

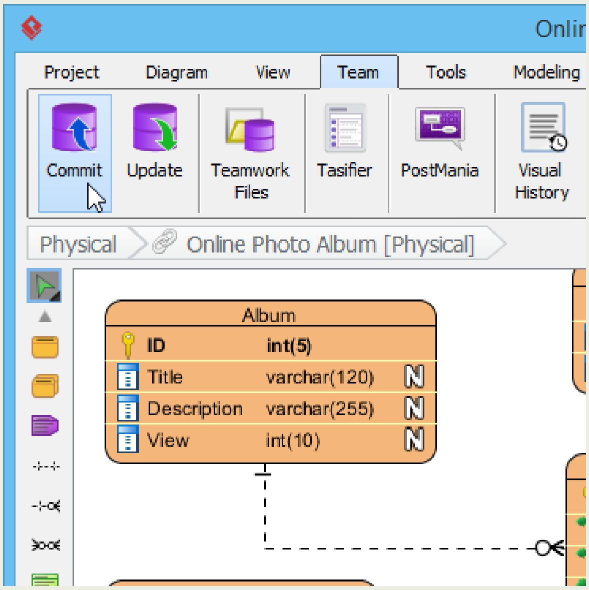
Comment		
ID	int(5)	
PhotoID	*int*	
PostDate	date	N
Content	varchar(255)	N
Email	varchar(255)	N

Comments can be posted by anymore, both registered members and guests. When a guest wants to post a comment, he/she has to provide his/her email address. Currently, our Comments table has no field for storing the email. So, please add it.

DB Design Best Practices

DB Design & Dev – The Steps

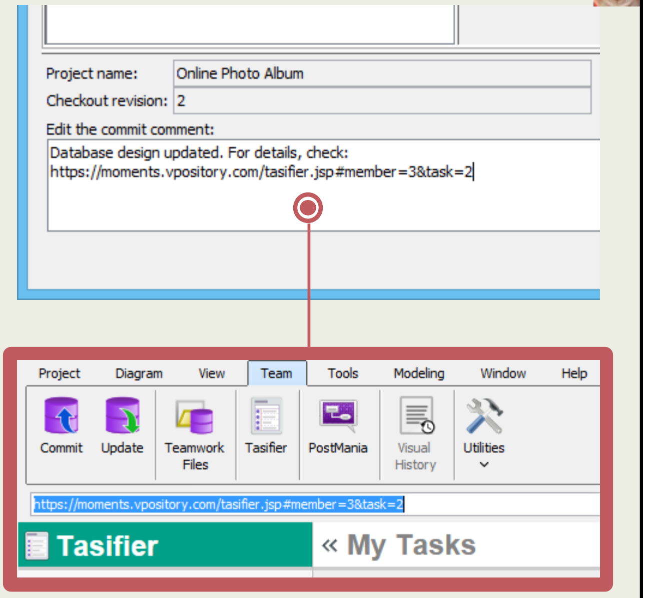
1. **Peter** creates a task to John in Tasifier to request for the change.
2. **John** opens the task to study Peter's request.
3. **John** modifies the ERD by adding the column Peter needed.
4. **John** describes the change.
5. **John** commits the change to server.



Album		
ID	int(5)	
Title	varchar(120)	N
Description	varchar(255)	N
View	int(10)	N

DB Design & Dev – The Steps

1. **Peter** creates a task to John in Tasifier to request for the change.
2. **John** opens the task to study Peter's request.
3. **John** modifies the ERD by adding the column Peter needed.
4. **John** describes the change.
5. **John** commits the change to server.
6. **John** pastes the URL of the task as commit comment.


Project name: Online Photo Album

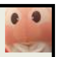
Checkout revision: 2

Edit the commit comment:

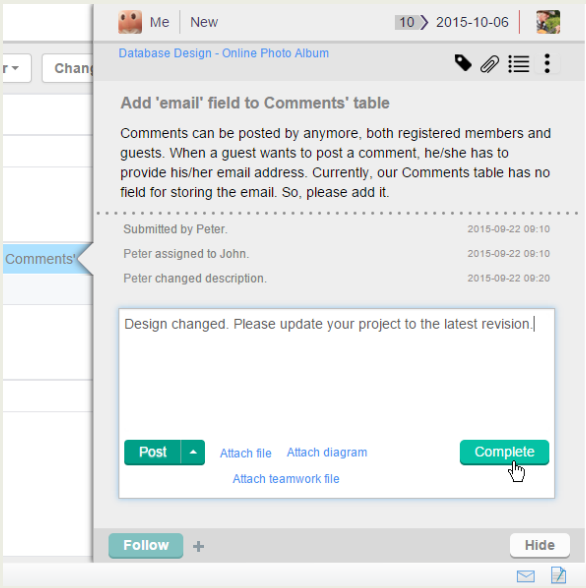
Database design updated. For details, check:
https://moments.vpository.com/tasifier.jsp#member=3&task=2

https://moments.vpository.com/tasifier.jsp#member=3&task=2

DB Design & Dev – The Steps



1. Peter creates a task to John in Tasifier to request for the change.
2. John opens the task to study Peter's request.
3. John modifies the ERD by adding the column Peter needed.
4. John describes the change.
5. John commits the change to server.
6. John pastes the URL of the task as commit comment.
7. John marks the task complete.



Database Design - Online Photo Album

Add 'email' field to Comments' table

Comments can be posted by anymore, both registered members and guests. When a guest wants to post a comment, he/she has to provide his/her email address. Currently, our Comments table has no field for storing the email. So, please add it.

Submitted by Peter. 2015-09-22 09:10

Peter assigned to John. 2015-09-22 09:10

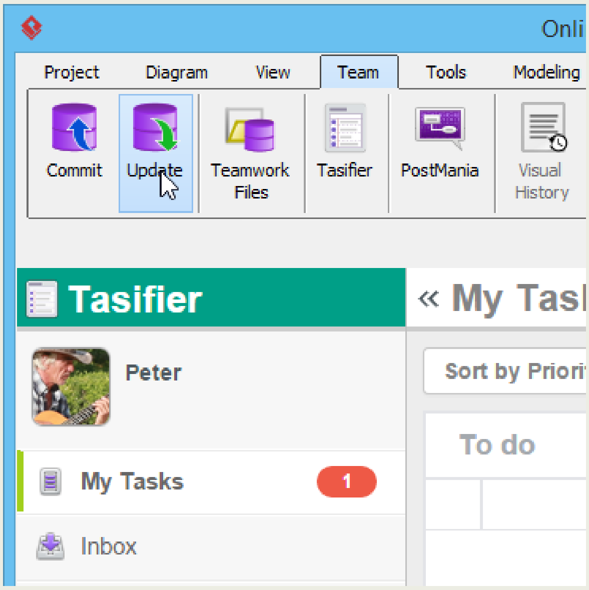
Peter changed description. 2015-09-22 09:20

Design changed. Please update your project to the latest revision.

DB Design Best Practices

DB Design & Dev – The Steps

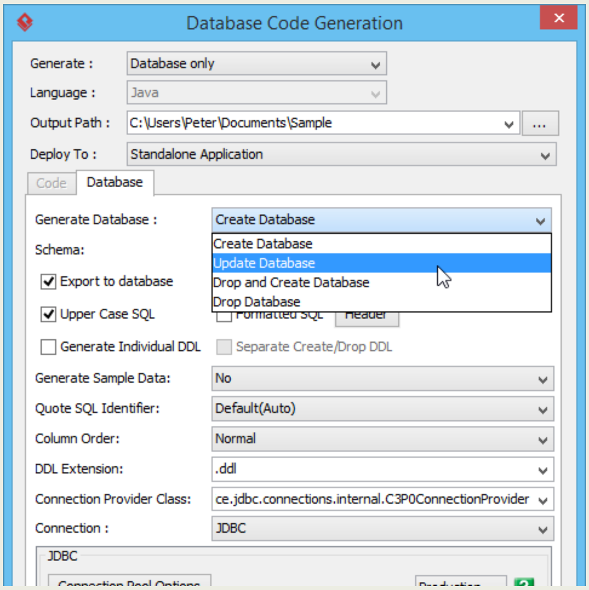
1. Peter creates a task to John in Tasifier to request for the change.
2. John opens the task to study Peter's request.
3. John modifies the ERD by adding the column Peter needed.
4. John describes the change.
5. John commits the change to server.
6. John pastes the URL of the task as commit comment.
7. John marks the task complete.
8. Peter updates the project from server.



DB Design Best Practices

DB Design & Dev – The Steps

1. **Peter** creates a task to John in Tasifier to request for the change.
2. **John** opens the task to study Peter's request.
3. **John** modifies the ERD by adding the column Peter needed.
4. **John** describes the change.
5. **John** commits the change to server.
6. **John** pastes the URL of the task as commit comment.
7. **John** marks the task complete.
8. **Peter** updates the project from server.
9. **Peter** patch the changes to his development database.



Database Code Generation

Generate : Database only

Language : Java

Output Path : C:\Users\Peter\Documents\Sample

Deploy To : Standalone Application

Generate Database : Create Database

Schema:

☑ Export to database

☑ Upper Case SQL

□ Formatted SQL

□ Generate Individual DDL

□ Separate Create/Drop DDL

Generate Sample Data: No

Quote SQL Identifier: Default(Auto)

Column Order: Normal

DDL Extension: .ddl

Connection Provider Class: ce.jdbc.connections.internal.C3P0ConnectionProvider

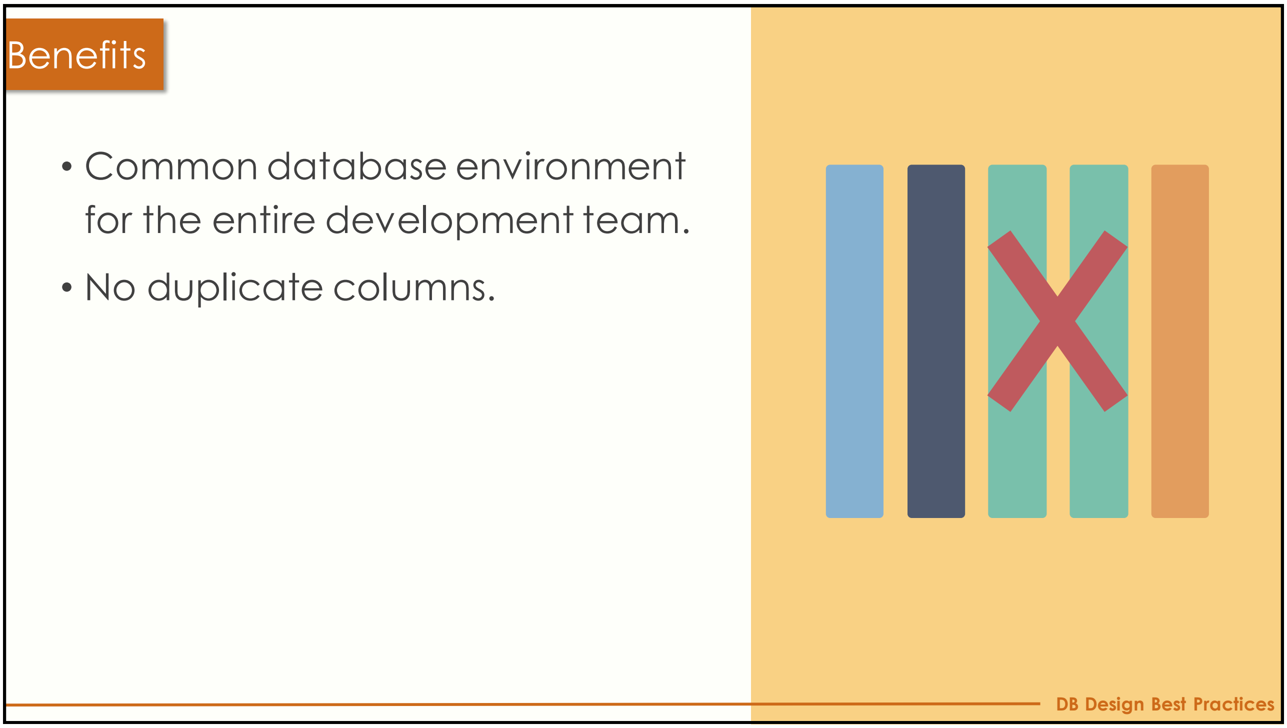
Connection : JDBC

DB Design Best Practices

Benefits

- Common database environment for the entire development team.

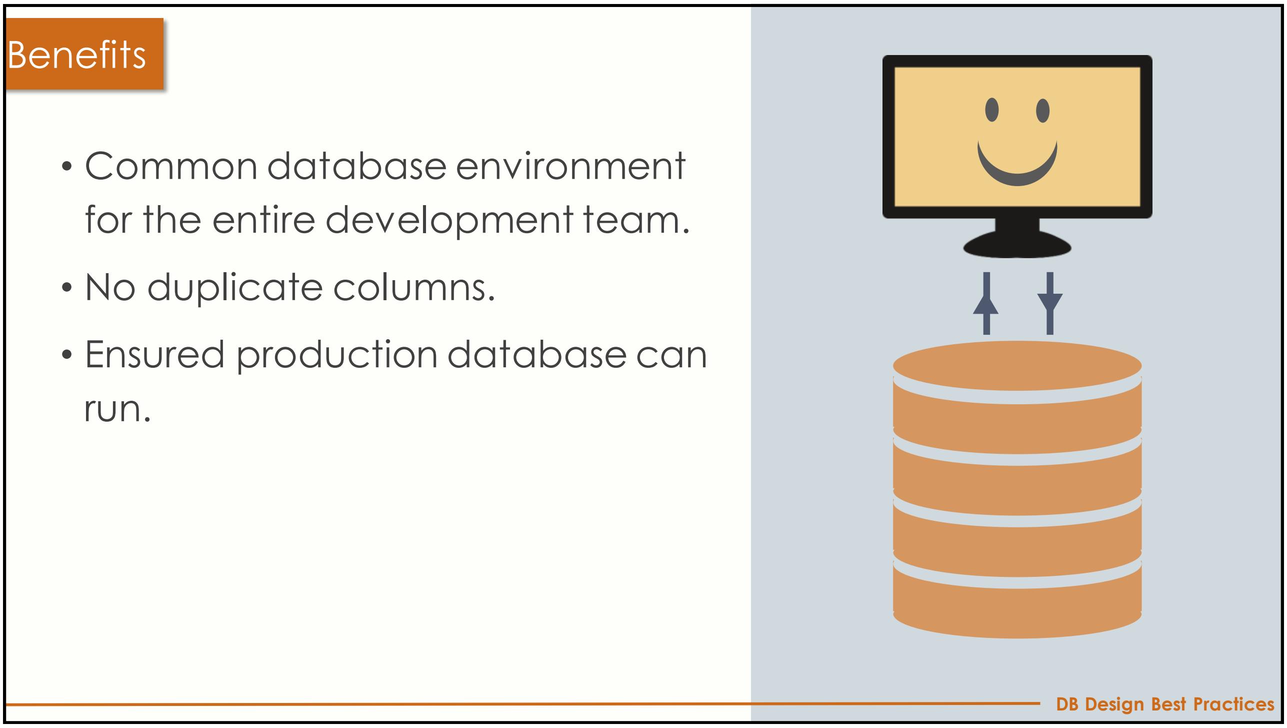
DB Design Best Practices



Benefits

- Common database environment for the entire development team.
- No duplicate columns.

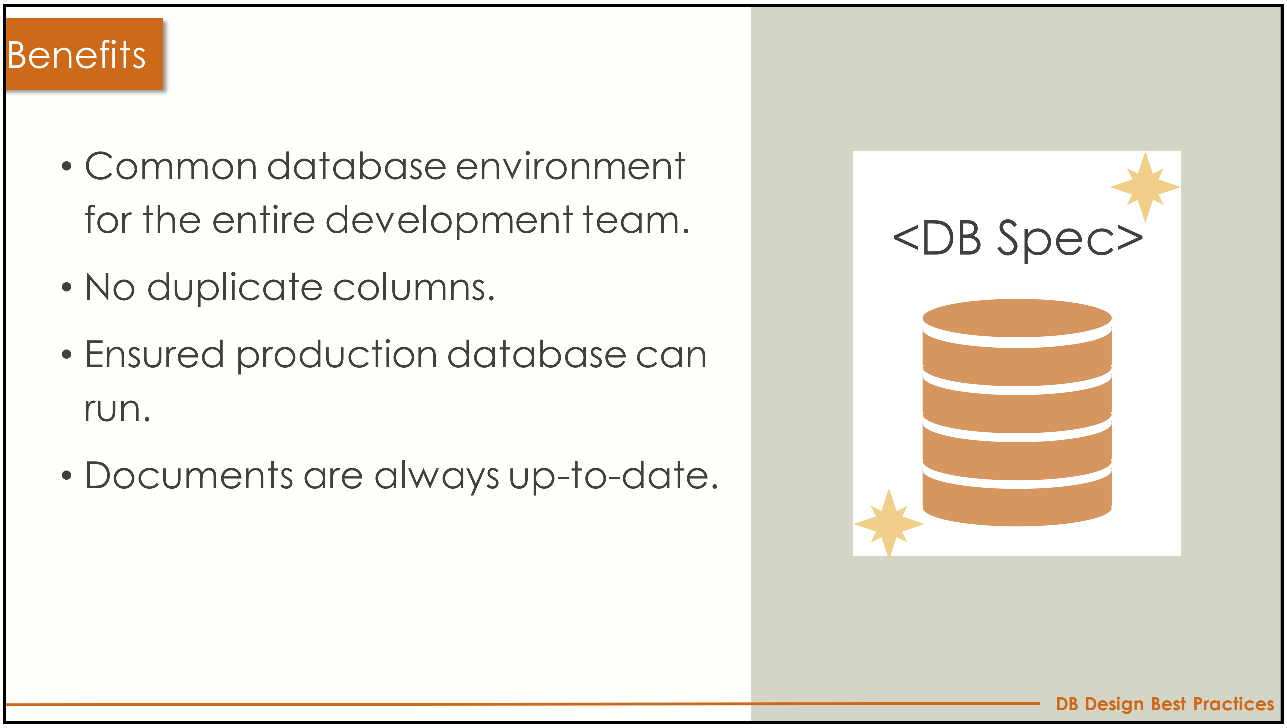
DB Design Best Practices



Benefits

- Common database environment for the entire development team.
- No duplicate columns.
- Ensured production database can run.

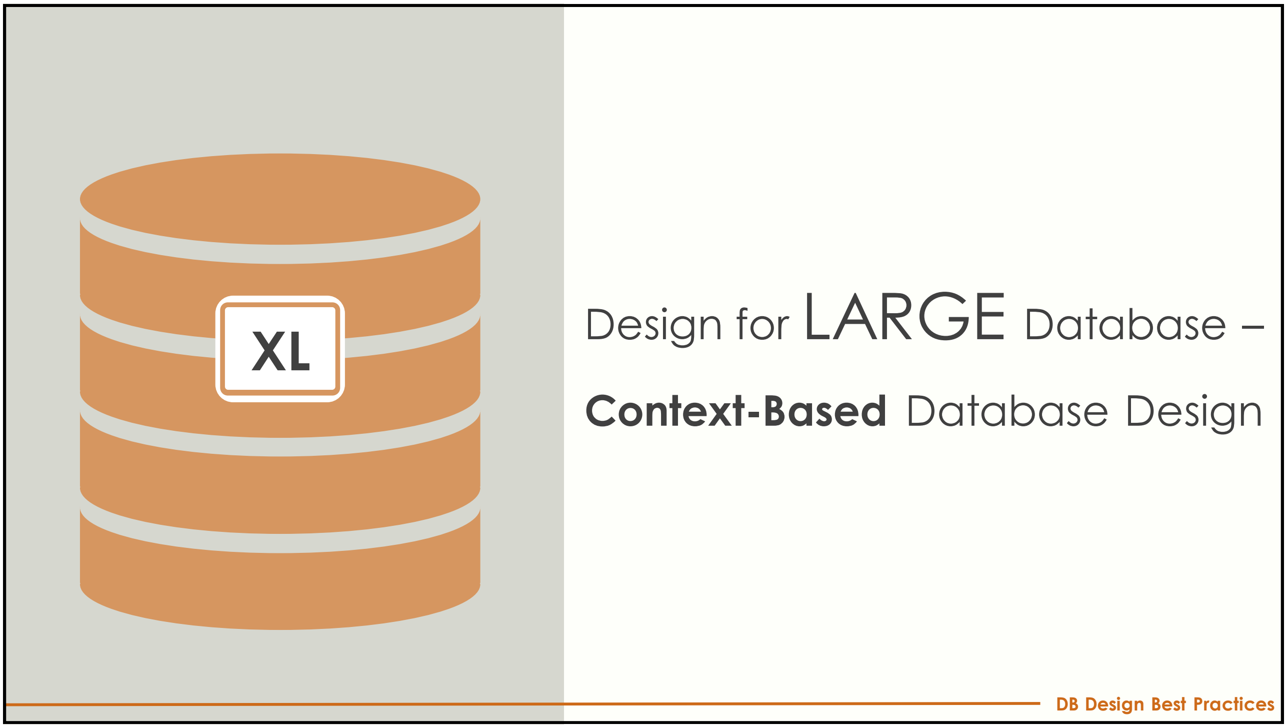
DB Design Best Practices



Benefits

- Common database environment for the entire development team.
- No duplicate columns.
- Ensured production database can run.
- Documents are always up-to-date.

DB Design Best Practices



Design for LARGE Database –

Context-Based Database Design

DB Design Best Practices

Without the help of any design method, the design of a large database is always large and complicated.

DB Design Best Practices

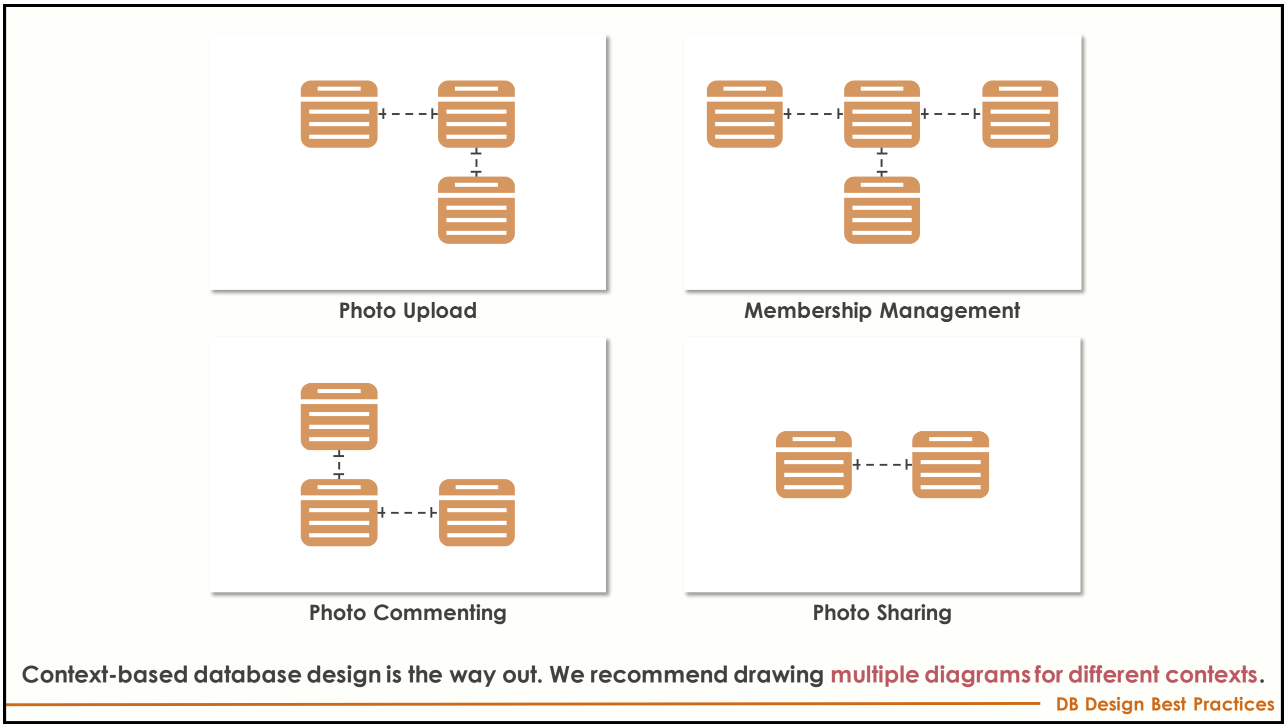


Photo Upload

Membership Management

Photo Commenting

Photo Sharing

Context-based database design is the way out. We recommend drawing multiple diagrams for different contexts.

DB Design Best Practices

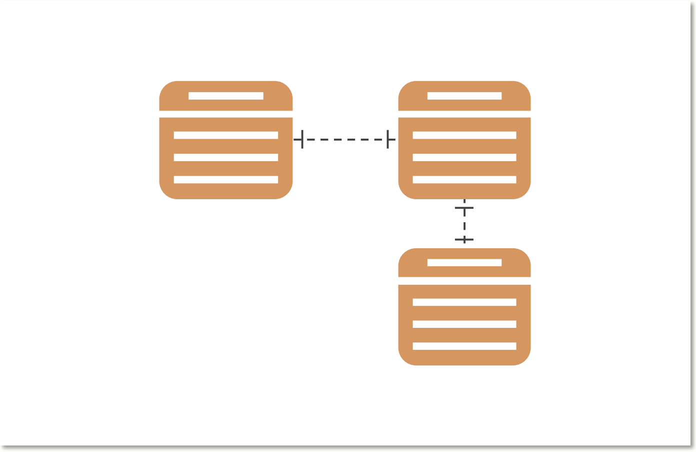
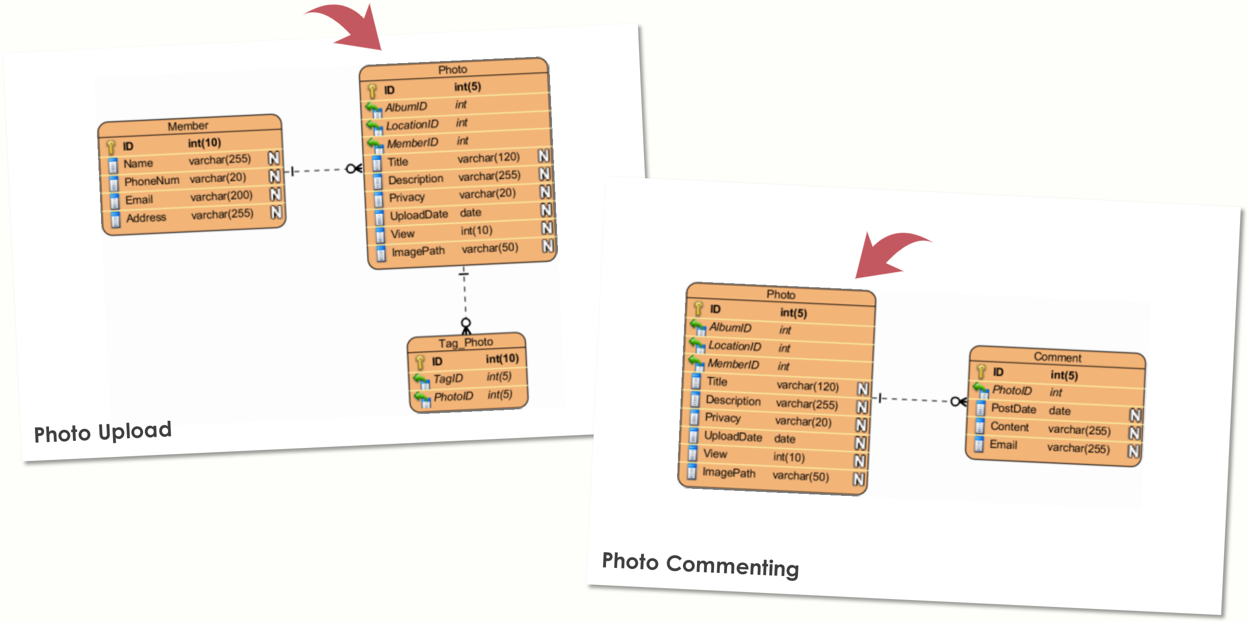


Photo Upload

Team members can focus on a small set of relevant entities both when designing and development, which makes understanding and communication much easier.

DB Design Best Practices



Member	
ID	int(10)
Name	varchar(255)
PhoneNum	varchar(20)
Email	varchar(200)
Address	varchar(255)

Photo	
ID	int(5)
AlbumID	*int*
LocationID	*int*
MemberID	*int*
Title	varchar(120)
Description	varchar(255)
Privacy	varchar(20)
UploadDate	date
View	int(10)
ImagePath	varchar(50)

Tag_Photo	
ID	int(10)
TagID	*int(5)*
PhotoID	*int(5)*

Photo Upload

Photo	
ID	int(5)
AlbumID	*int*
LocationID	*int*
MemberID	*int*
Title	varchar(120)
Description	varchar(255)
Privacy	varchar(20)
UploadDate	date
View	int(10)
ImagePath	varchar(50)

Comment	
ID	int(5)
PhotoID	*int*
PostDate	date
Content	varchar(255)
Email	varchar(255)

Photo Commenting

With the context-based design approach, very often you will re-use an entity in another contexts (diagrams).

DB Design Best Practices

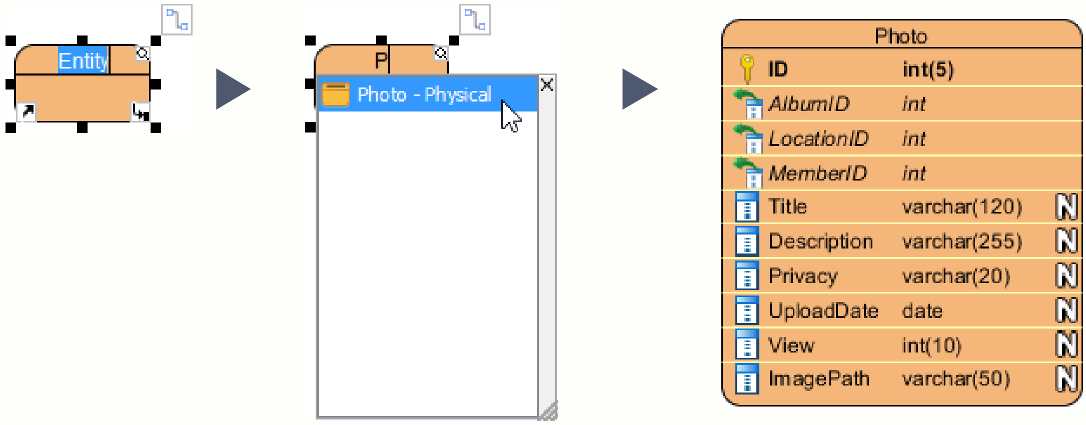


Photo		
ID	**int(5)**	
AlbumID	*int*	
LocationID	*int*	
MemberID	*int*	
Title	varchar(120)	N
Description	varchar(255)	N
Privacy	varchar(20)	N
UploadDate	date	N
View	int(10)	N
ImagePath	varchar(50)	N

To re-use an entity is simple. When you create an entity, simply type in the name of that entity, or select it from the completion list (press Ctrl-Space to toggle it)

DB Design Best Practices

Summary

- Typical approach of database design and development
- The suggested approach of database design and development
- Design large database by drawing context-based ERDs

DB Design Best Practices