Entity- Relationship Model

Group Q&A

Yan Yan Du Shimeng Guo Fangran
Zheng Ruxu Lin Yuxin

Content

Section	Topic
1	Relational model
2	ERM theory
3	ERD process
4	Object orientation
5	Hybrid modeling
6	Derived design concept

Relational model: Review

- Definition: a subset of D1×D2×…×Dn

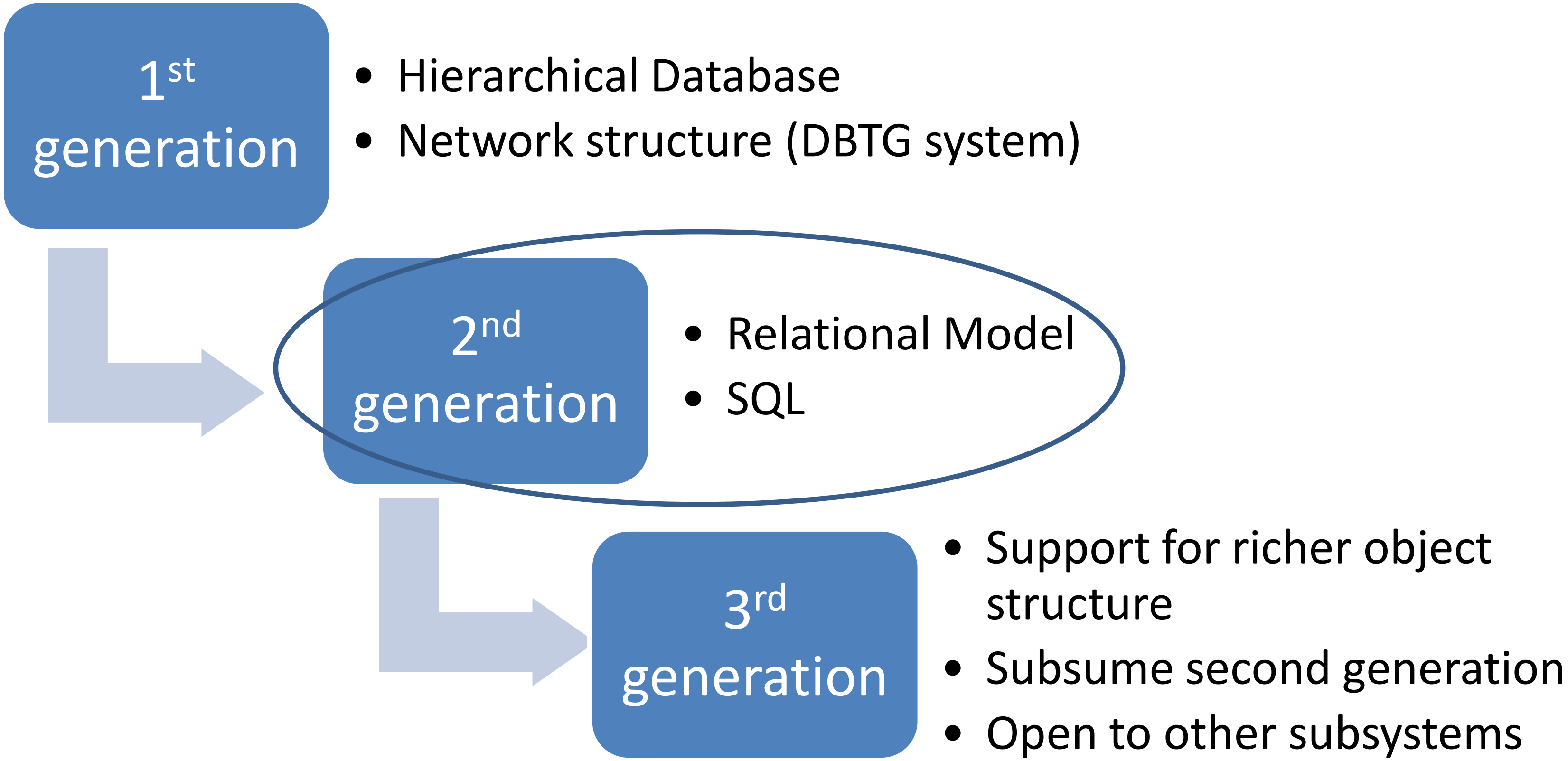
- Terminology: attribute, tuple, etc.

- Operation Set: union, selection, etc.

Content

Section	Topic
1	Relational model
2	ERM theory
3	ERD Process
4	Object orientation
5	Hybrid modeling
6	Derived design concept

ERM theory: Development of DBMS



1st generation

- Hierarchical Database
- Network structure (DBTG system)

2nd generation

- Relational Model
- SQL

3rd generation

- Support for richer object structure
- Subsume second generation
- Open to other subsystems

ERM theory: Comparison

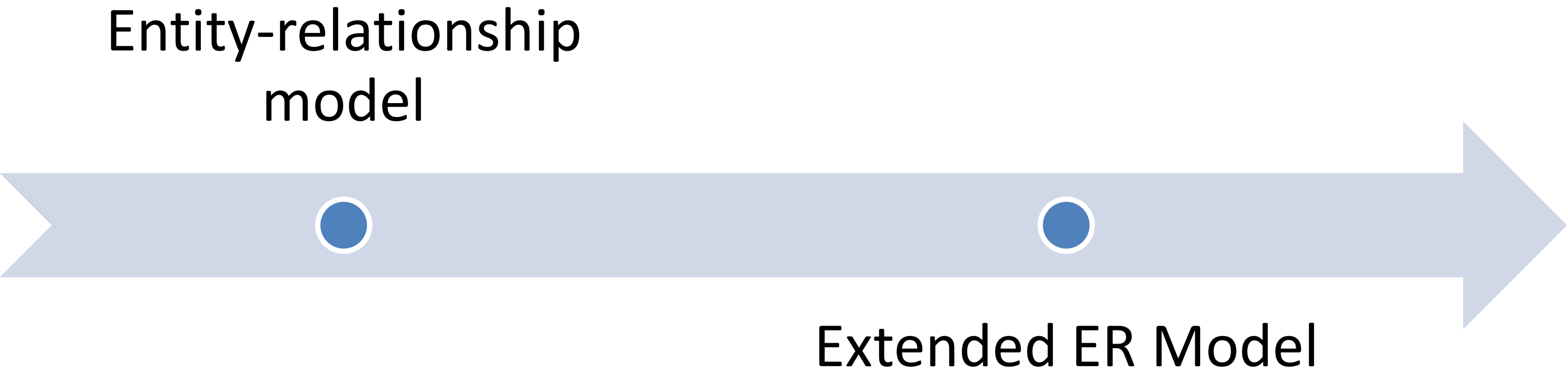
E.F.Codd

Peter Chen

Slide position_3:5

ERM theory: History

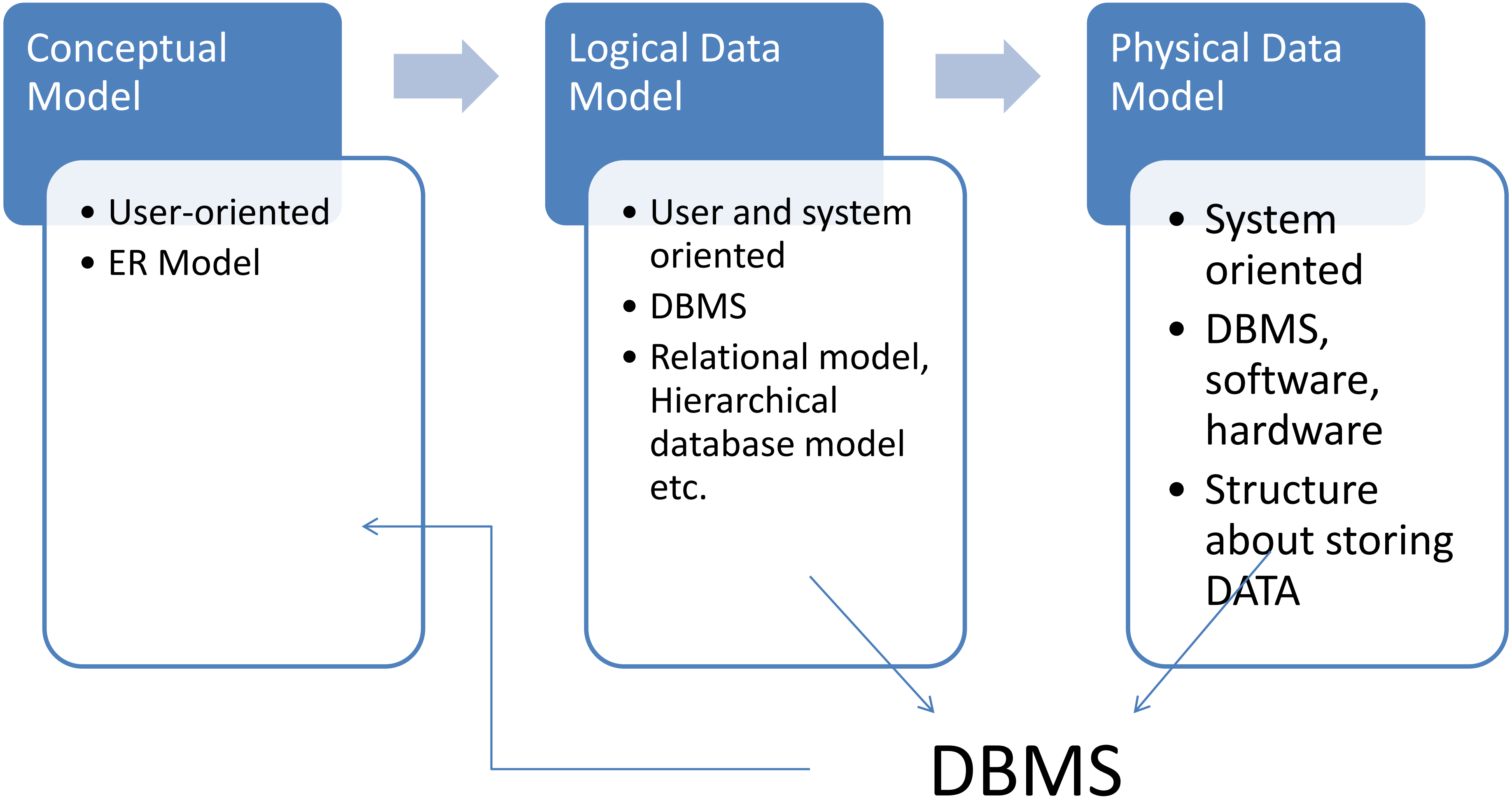
- Peter Chen (1976)
- Later improvement —Extended ER Model



Entity-relationship model

Extended ER Model

ERM theory: Conceptual model



Conceptual Model

- User-oriented
- ER Model

Logical Data Model

- User and system oriented
- DBMS
- Relational model, Hierarchical database model etc.

Physical Data Model

- System oriented
- DBMS, software, hardware
- Structure about storing DATA

DBMS

ERM theory: Introduction

- Entity and entity set
- Relationship and relationship set
- Attribute and keys
- Mapping cardinality
- Relation, entity and relationship
- Model legend

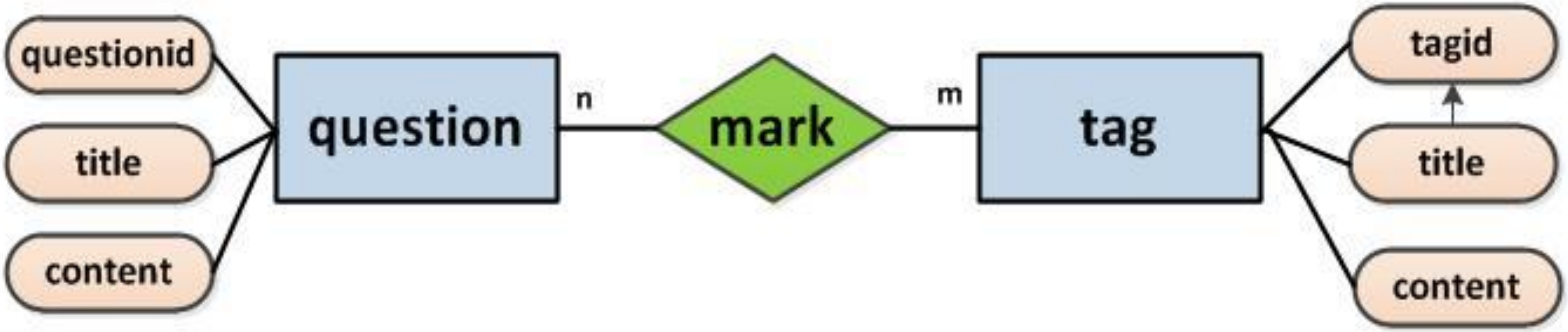
Content

Section	Topic
1	Relational model
2	ERM theory
3	ERD process
4	Object orientation
5	Hybrid modeling
6	Derived design concept

ERD process: Reason

- **Why E-R diagram?**

Displays the real world in similar way with human beings

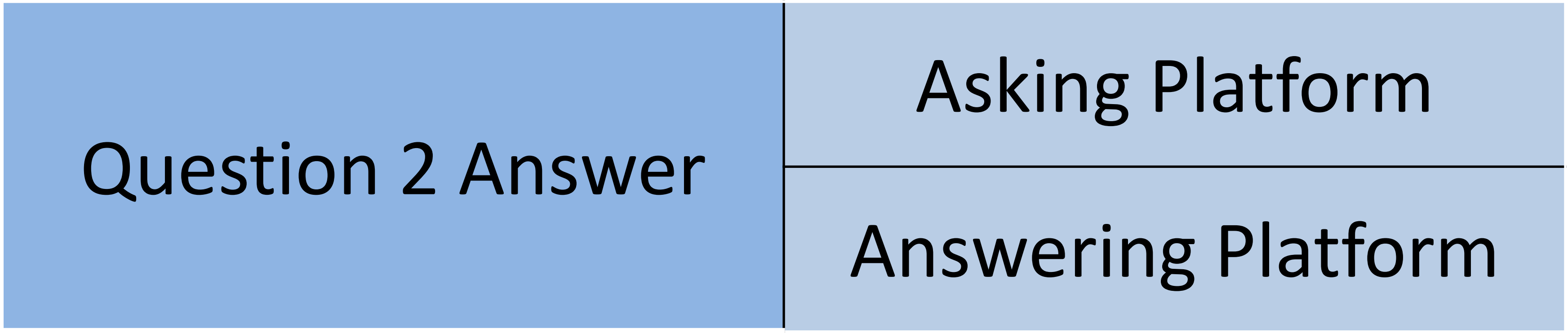


ERD process: Design philosophy

- Relative principle
- Consistency principle
- Simple principle

ERD process: Instance

Determine range of local structure



Question 2 Answer

Asking Platform

Answering Platform

ERD process: Instance

Define Entities

Asking Platform	Requester, Question, Tag
Answering Platform	Responder, Answer, Question

ERD process: Instance

Allocate Attributes

Asking Platform	requester	requester_id, requester_name, email, avatar
	question	question_id, question_title, question_content
	tag	tag_id, tag_title, tag_content
Answering Platform	responder	responder_id, responder_name, email, avatar
	answer	answer_id, answer_title, answer_content
	question	question_id, question_title, question_content

ERD process: Instance

Define Relationship

Asking Platform	requester to question	1 : n
	tag to question	n : m
	requester to tag	1 : n
Answering Platform	responder to answer	1 : n
	answer to question	n : 1

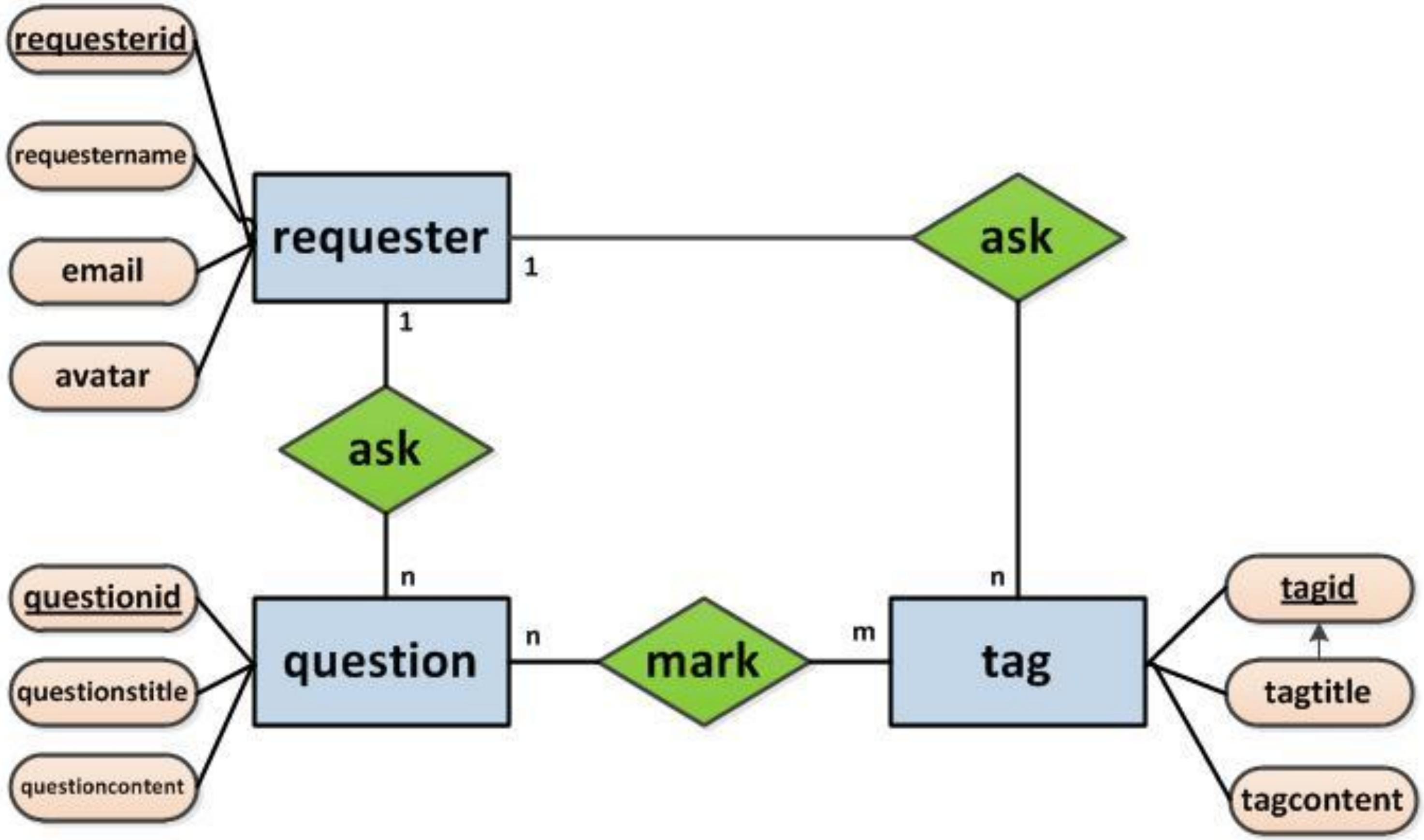
ERD process: Instance

Define Relationship

Asking Platform	requester to question	1 : n
	tag to question	n : m
	requester to tag	1 : n
Answering Platform	responder to answer	1 : n
	answer to question	n : 1

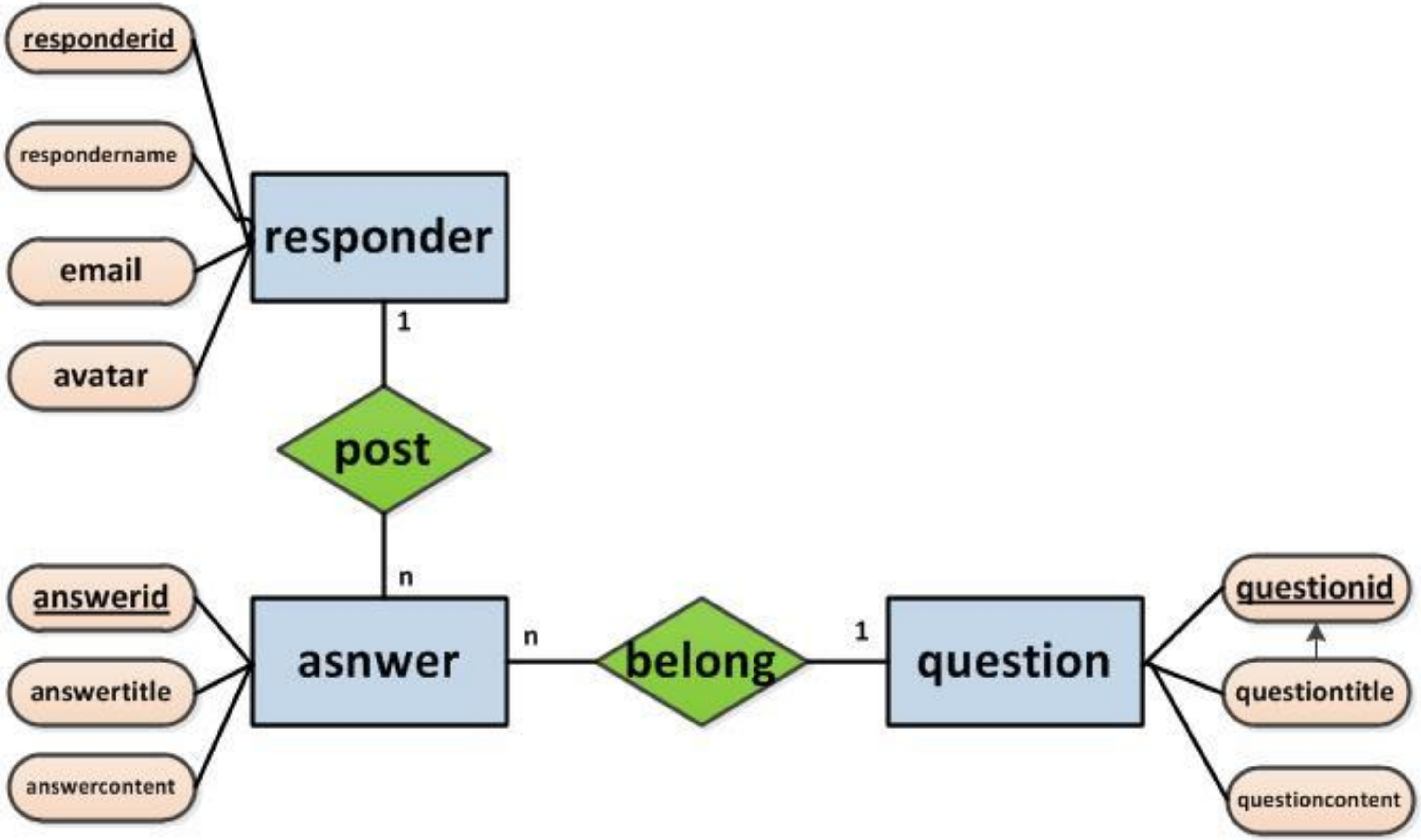
ERD process: Instance

Local ERM Diagram – Asking Platform



ERD process: Instance

Local ERM Diagram – Answering Platform

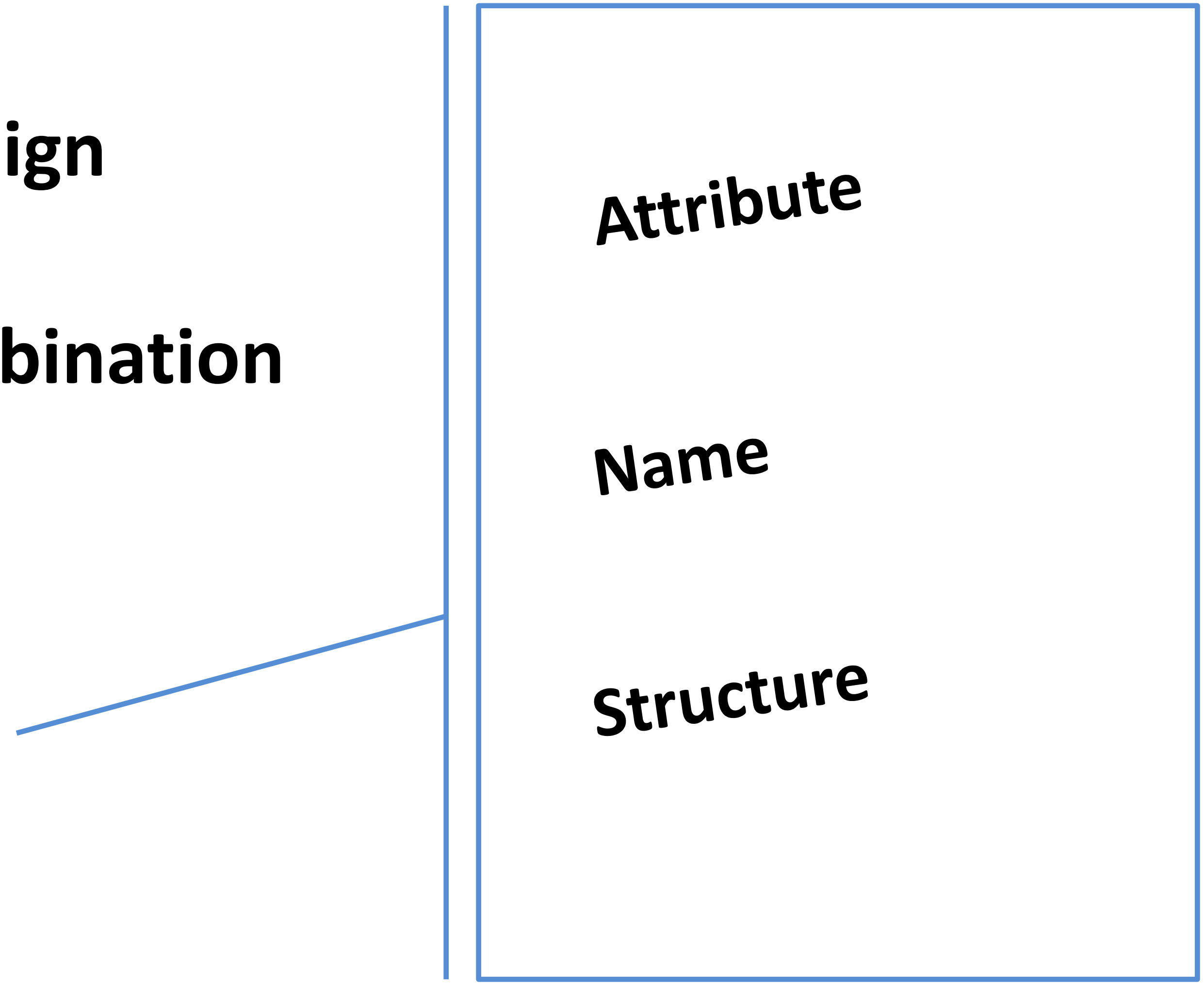


ERD process: Instance

Global ERM Design

= Local Combination

! Conflicts



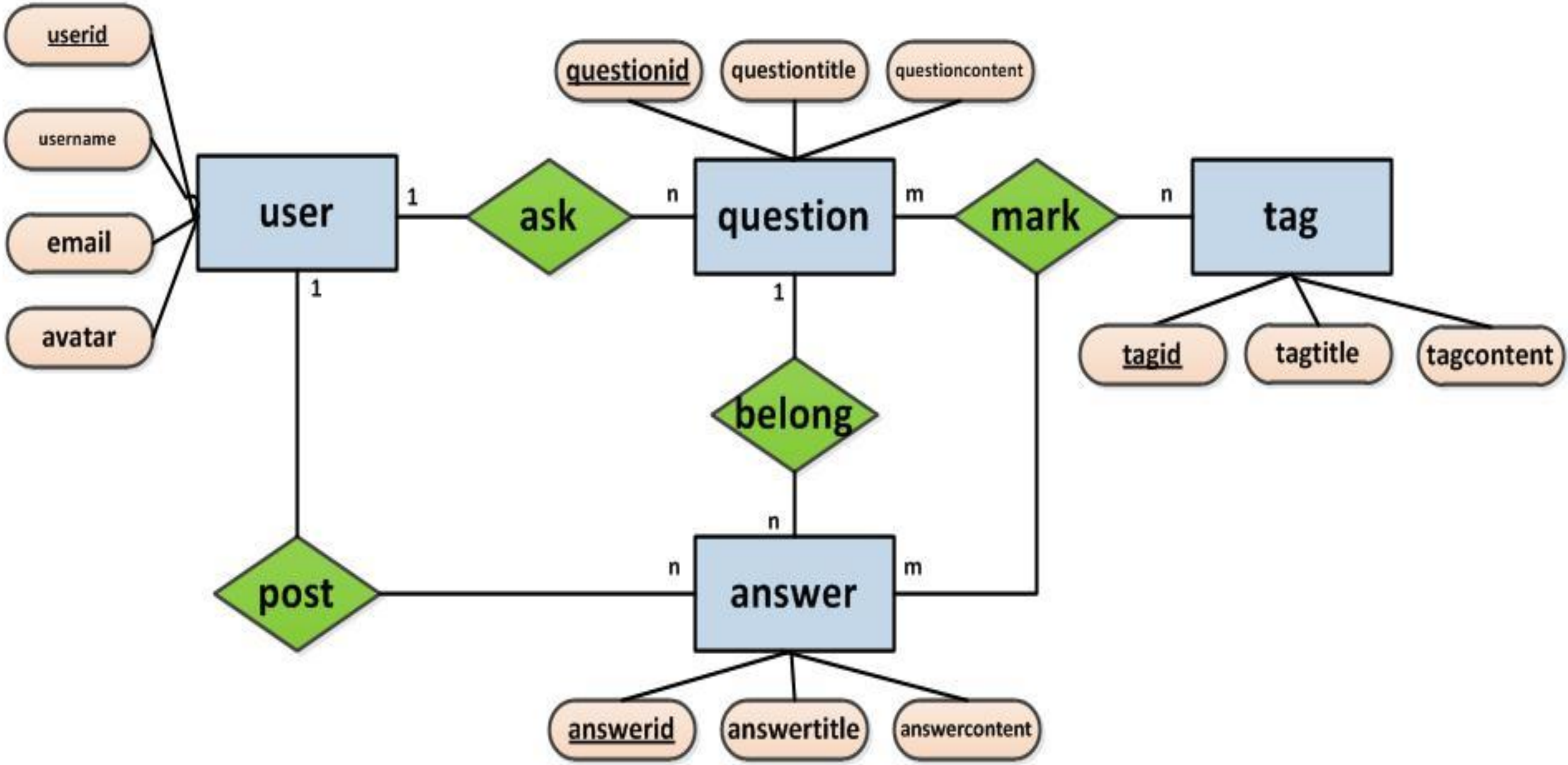
Attribute

Name

Structure

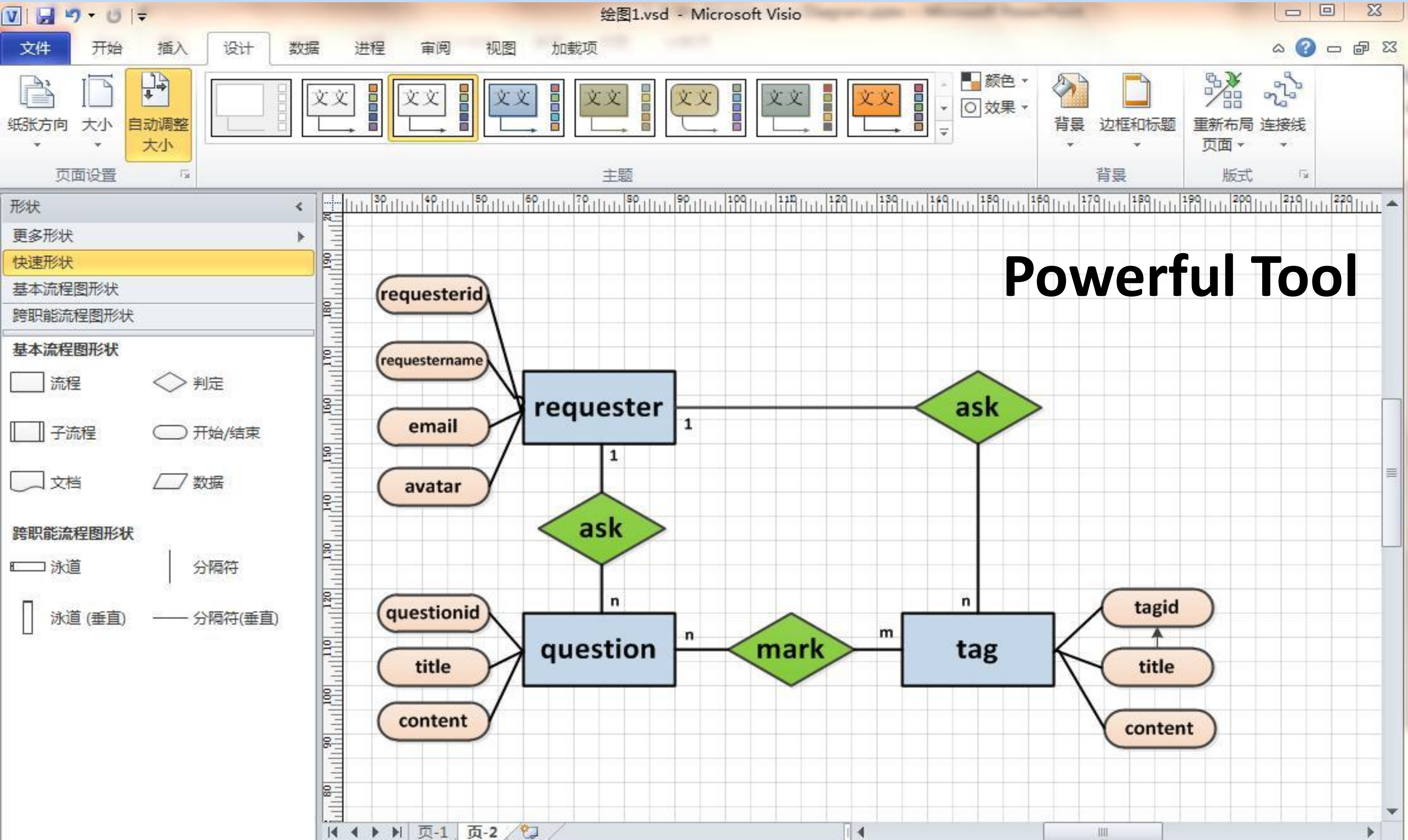
ERD process: Instance

Global ERM Design



Slide position_10:11

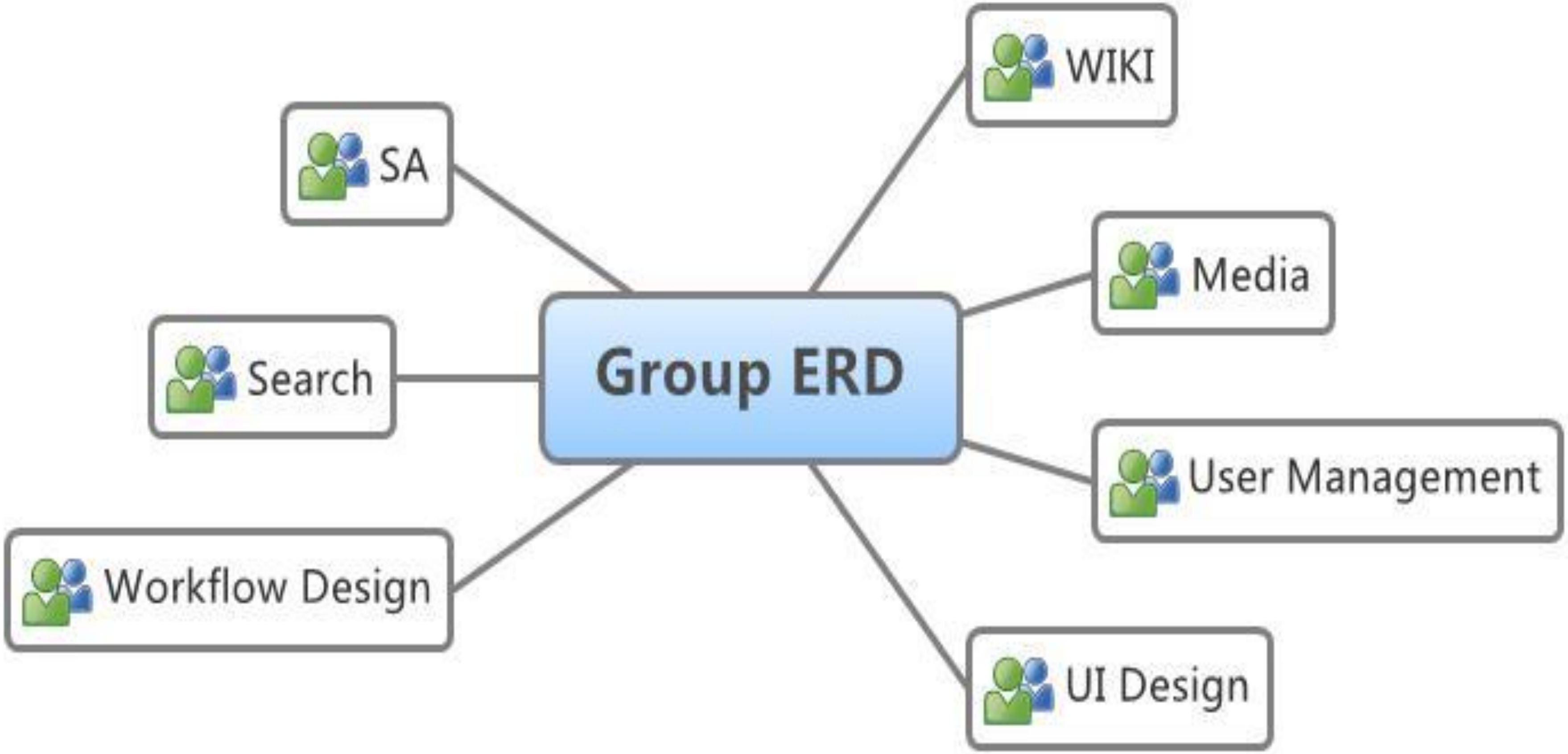
ERD process: Instance



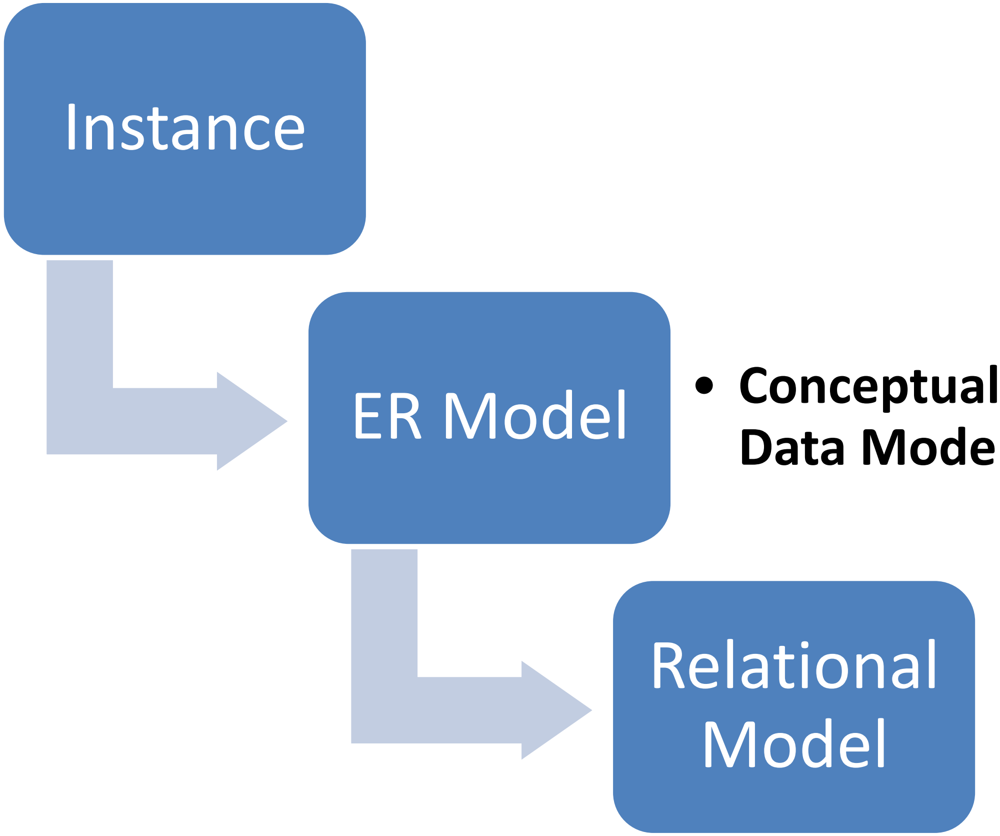
Powerful Tool

Slide position_11:11

ERD process: Instance



ERD process: Conversion

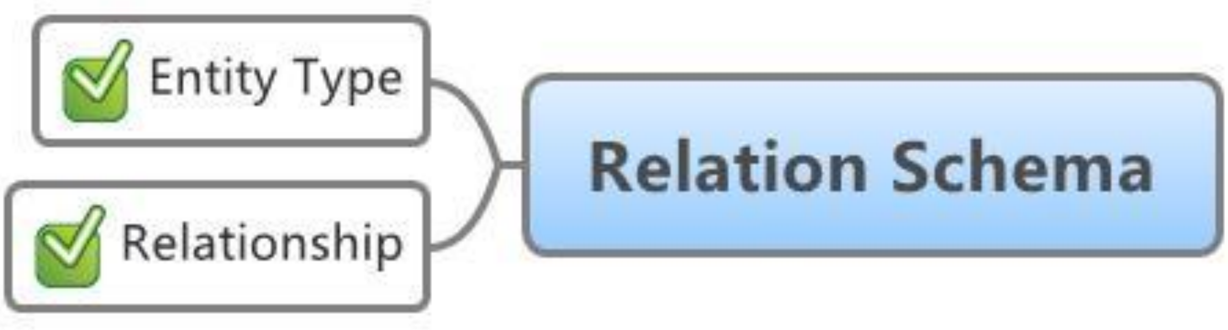


- **Conceptual Data Model**

- **Logical Data Model**

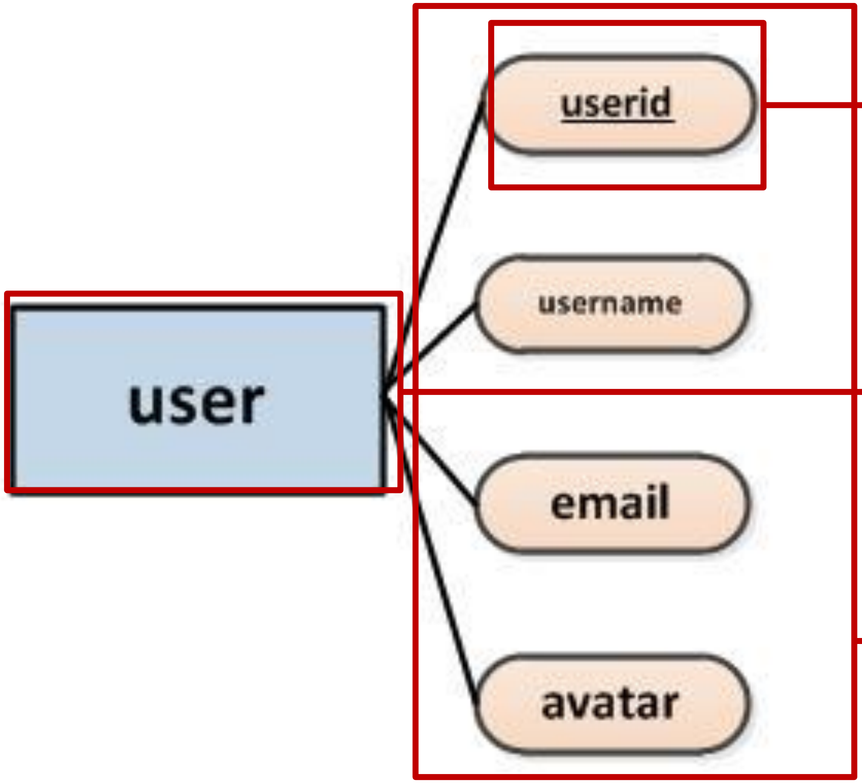
Slide position_2:10

ERD process: Conversion



ERD process: Conversion

Entity type to Relation schema



Key of relation

Name of relation

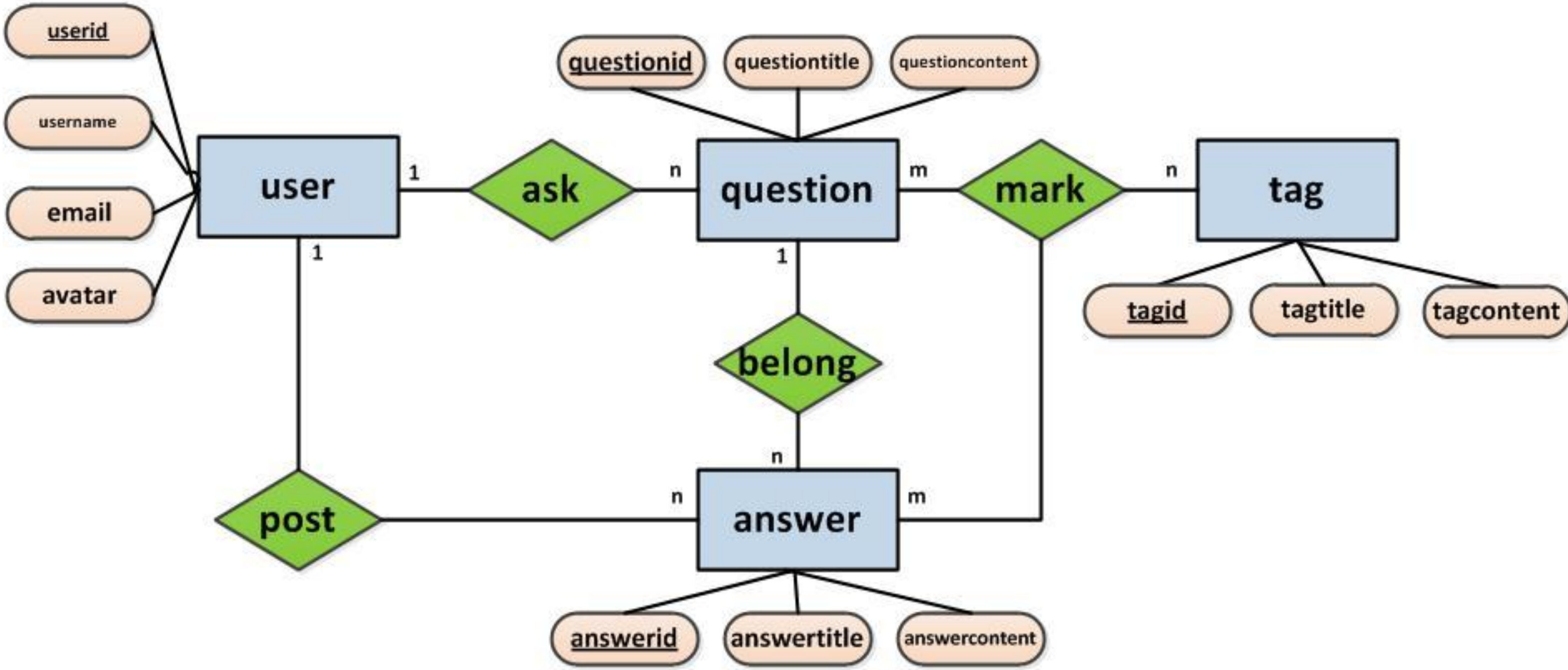
Attribute of relation

user (<u>userid</u>, username, email, avatar)

Slide position_4:10

ERD process: Conversion

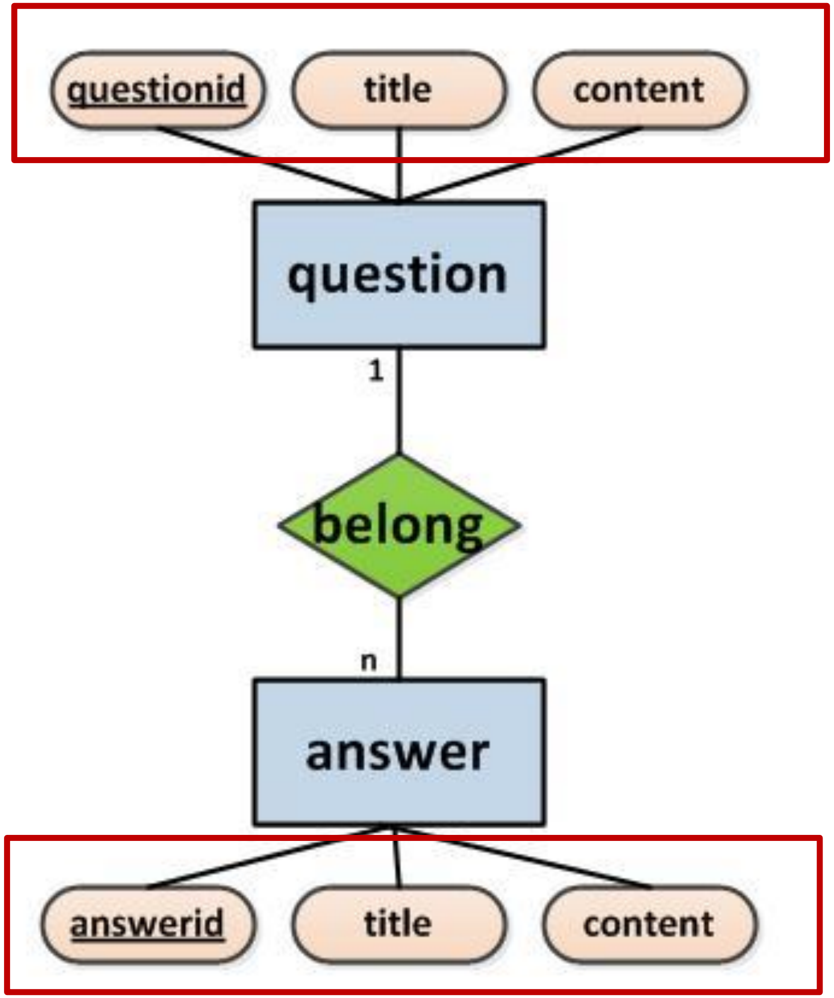
Entity type to Relation schema



tag (<u>tagid</u>, tagtitle, tagcontent)

ERD process: Conversion

Relationship to Relation schema



Attribute of relationship

=

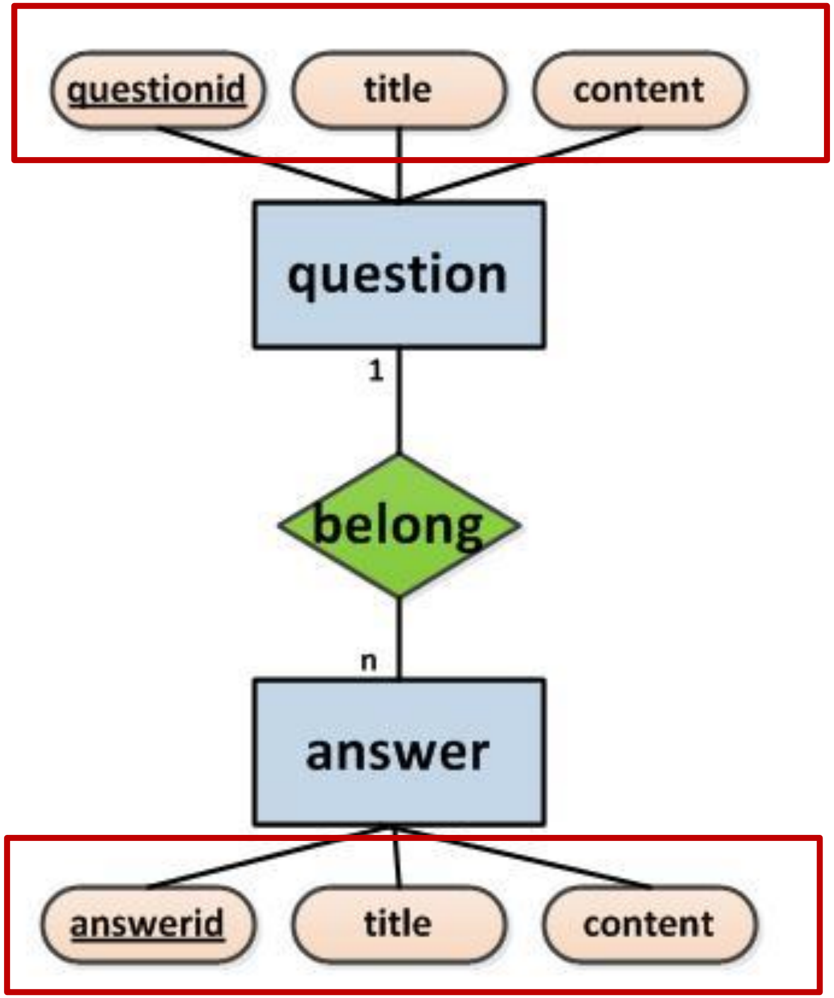
Attribute of itself

+

Key attribute of relevant entity

ERD process: Conversion

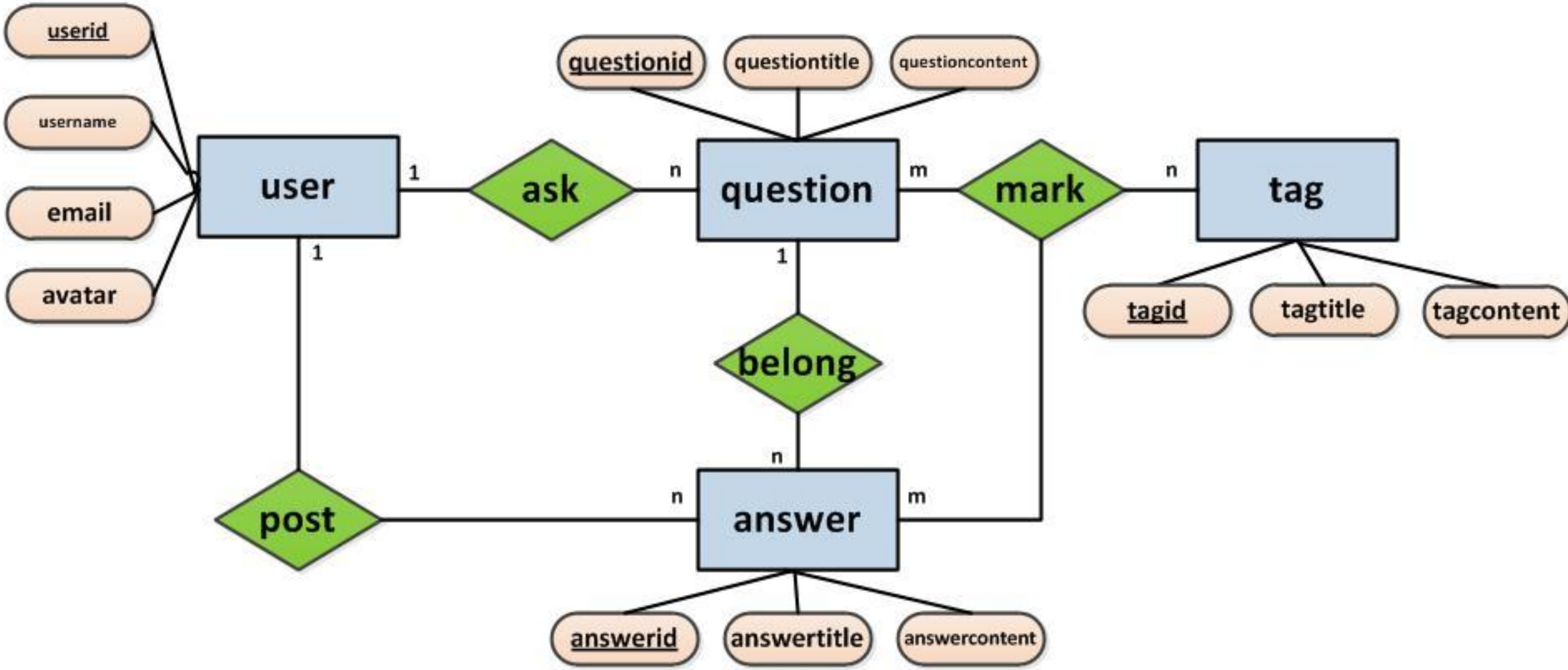
Relationship to Relation schema



R	Key
1:1	either of relevant entities
1:n	n-side entity
n:m	union of keys of relevant entities

ERD process: Conversion

Relationship to Relation schema



mark (?)

ERD process: Conversion

Number of entities + relationship

= Number of relation schema

?

ERD process: Conversion

Combine relation schema with same keys

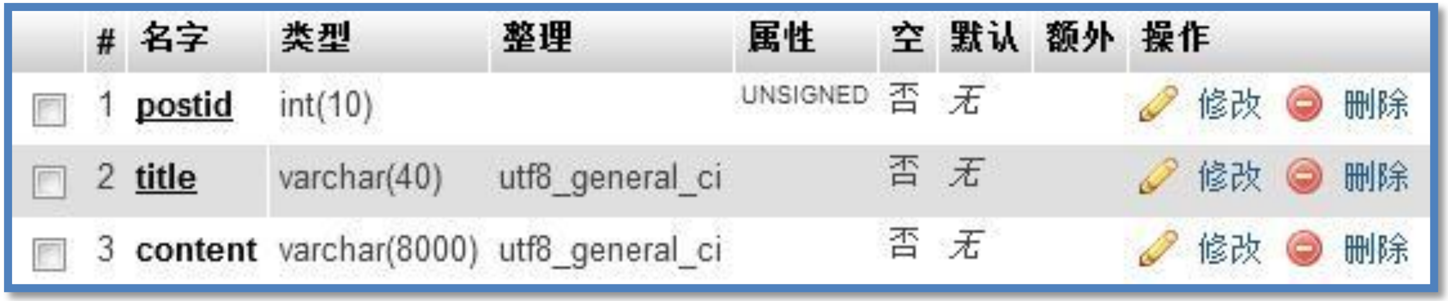
belong	(answerid, questionid)
Post	(answerid, userid)
answer	(answerid, answertitle, answercontent)

↓

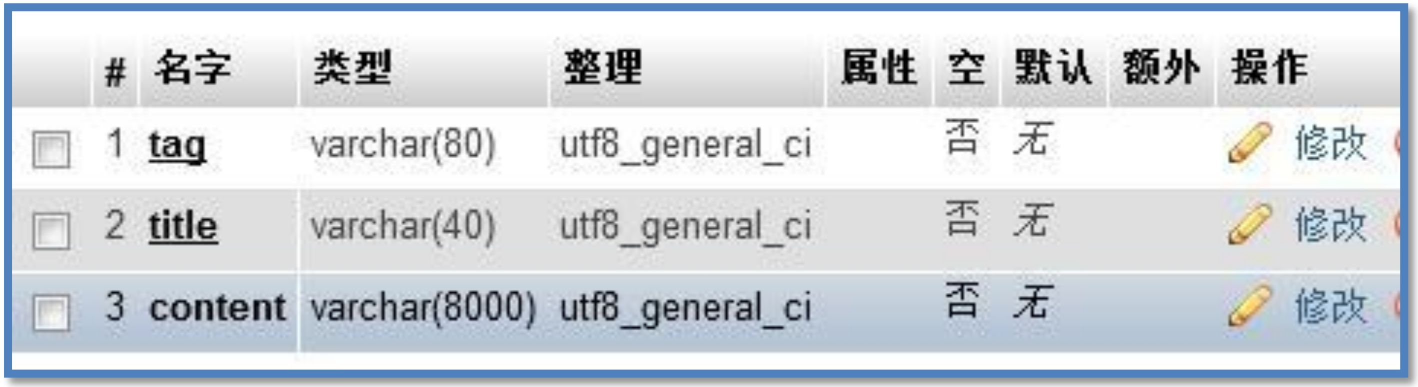
answer (answerid, userid, questionid, answertitle, answercontent)

ERD process: Conversion

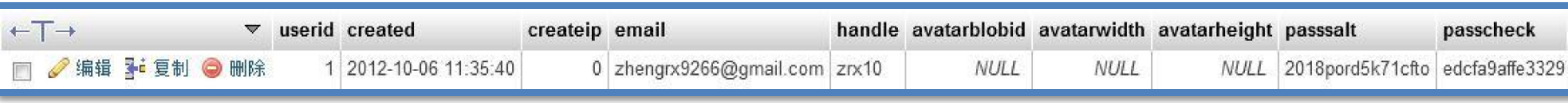
Relational Model in Q2A



	#	名字	类型	整理	属性	空	默认	额外	操作
□	1	**postid**	int(10)		UNSIGNED	否	无		修改 删除
□	2	**title**	varchar(40)	utf8_general_ci		否	无		修改 删除
□	3	**content**	varchar(8000)	utf8_general_ci		否	无		修改 删除



	#	名字	类型	整理	属性	空	默认	额外	操作
□	1	**tag**	varchar(80)	utf8_general_ci		否	无		修改
□	2	**title**	varchar(40)	utf8_general_ci		否	无		修改
□	3	**content**	varchar(8000)	utf8_general_ci		否	无		修改



←T→	userid	created	createip	email	handle	avatarblobid	avatarwidth	avatarheight	passsalt	passcheck
□ 编辑 复制 删除	1	2012-10-06 11:35:40	0	zhengrx9266@gmail.com	zrx10	*NULL*	*NULL*	*NULL*	2018pord5k71cfto	edcfa9affe3329

Content

Section	Topic
1	Relational model
2	ERM theory
3	ERD process
4	Object orientation
5	Hybrid modeling
6	Derived design concept

Object orientation

- What
- Why
- How
- Object-Oriented Model

Object orientation: What

- A point of view
- Basic point
- Concepts
- Features

Object orientation: Why

- Characteristics of OO
- 唯一性
- 分类性/抽象性
- 继承性
- 多态性

Object orientation: How

- 几种面向对象的开发方法（略讲）
- Booch
- Coad
- OMT
- OOSE&UML

Object orientation: OOM

- What is OOM
- 3 levels of OOM
- 对象模型
- 动态模型
- 功能模型

Content

Section	Topic
1	Relational model
2	ERM theory
3	ERD process
4	Object orientation
5	Hybrid modeling
6	Derived design concept

Hybrid modeling: Reasons

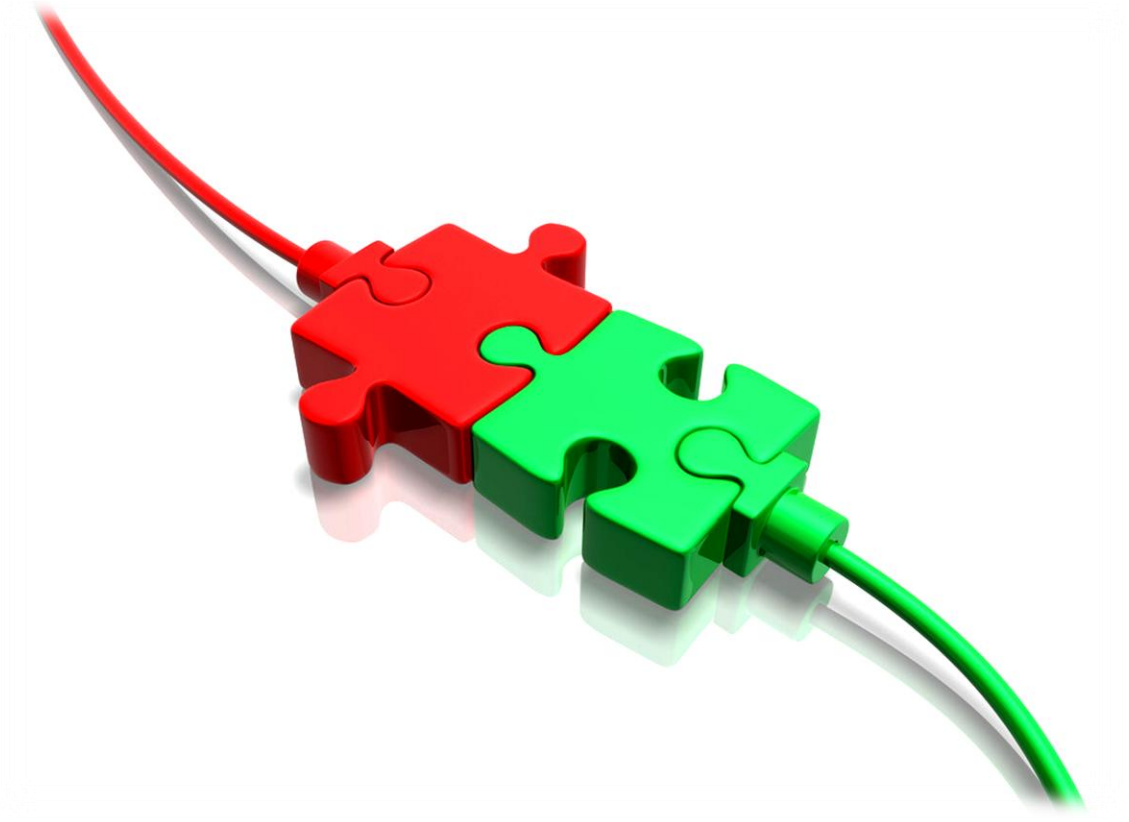
- **Multimedia Storage**

Hybrid modeling: Reasons

- **Code Compression**

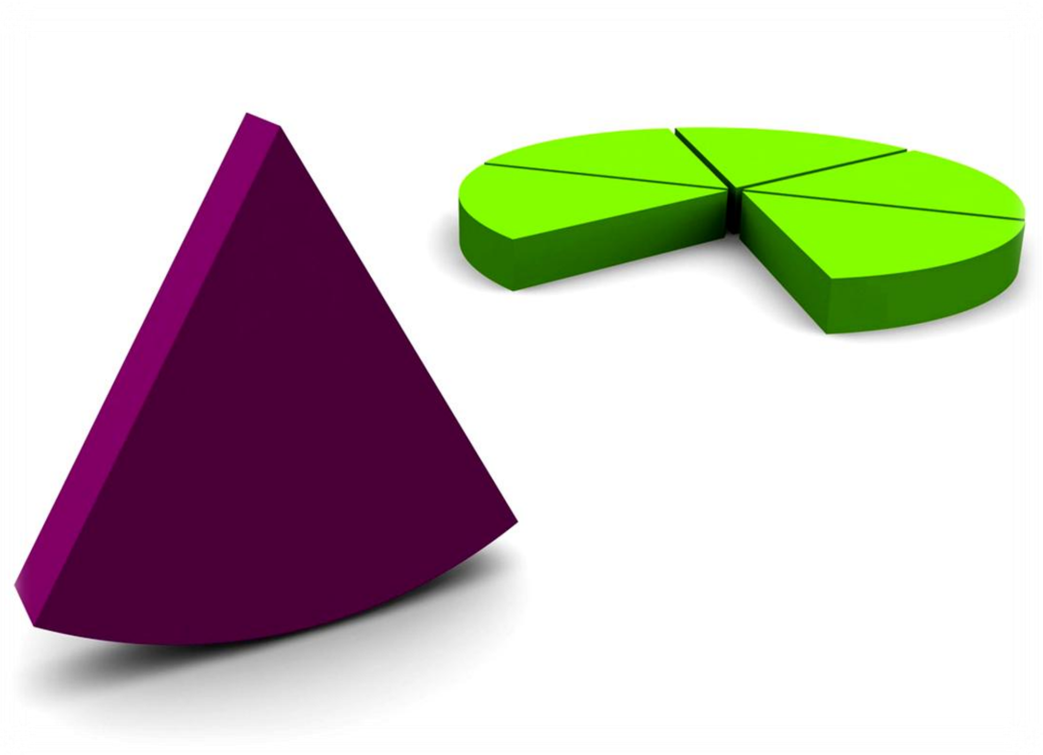
Hybrid modeling: Reasons

- **Data Relationship**



Hybrid modeling: Reasons

- **Semantic fault**



Hybrid modeling: Reasons

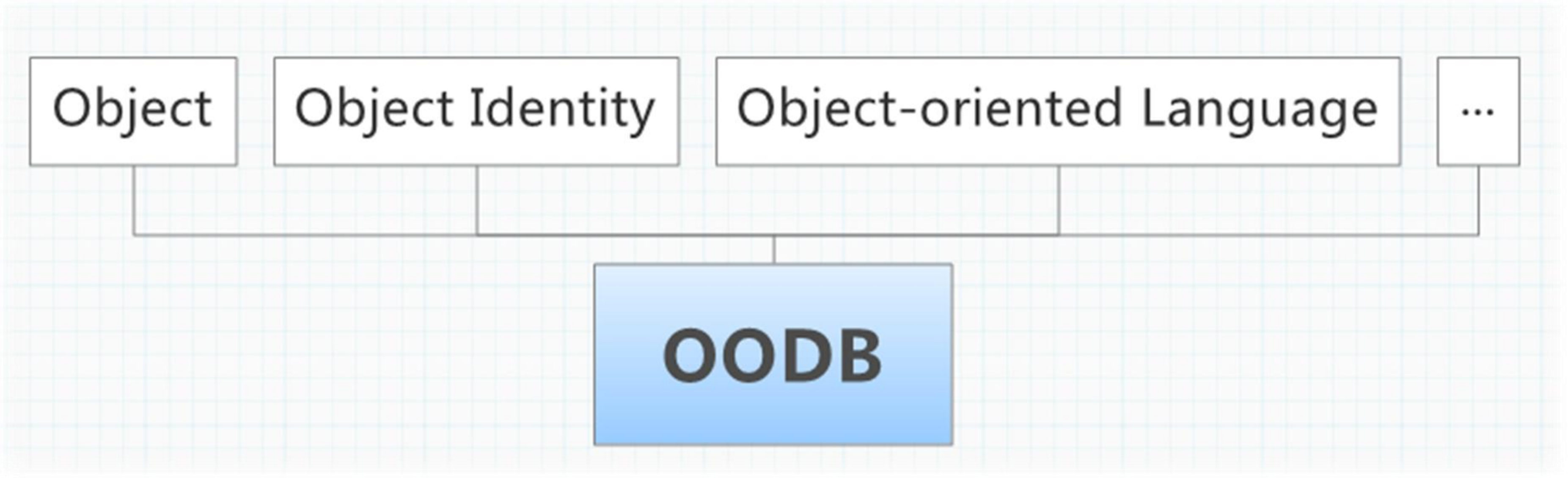
- **Automatic Detection**

Slide position_1:8

Hybrid modeling: Methodology

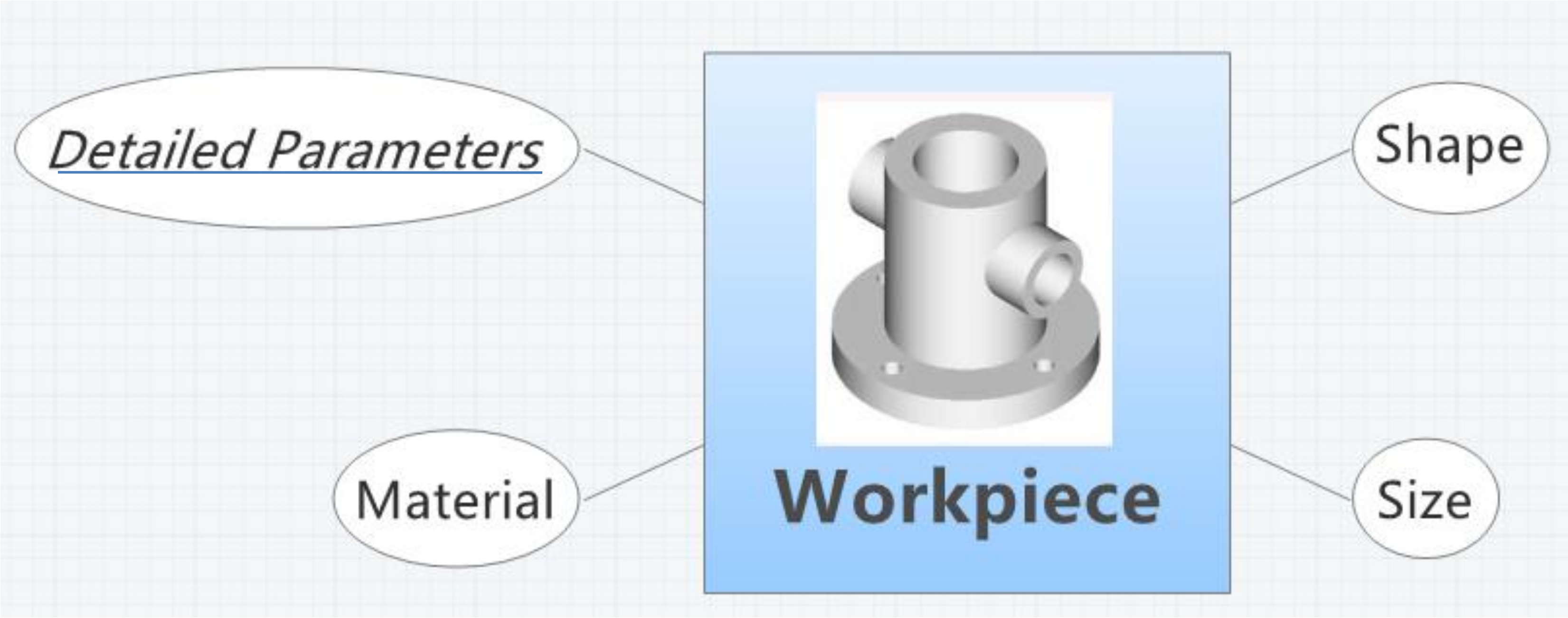
- OODB
- ORDB
- ORM

Hybrid modeling: Methodology



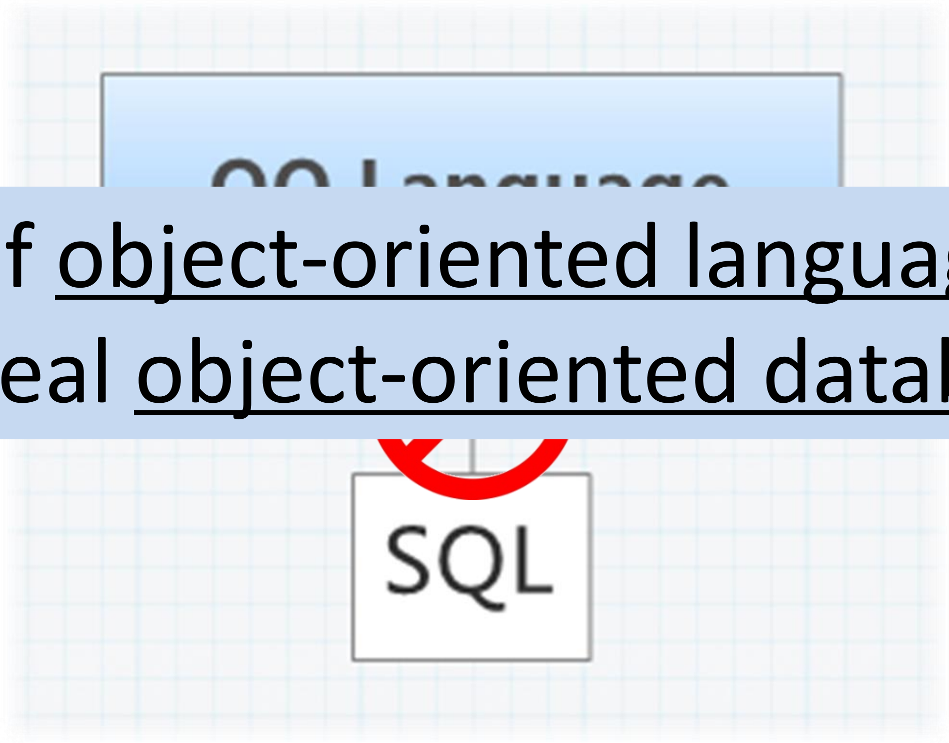
Hybrid modeling: Methodology

- Engineering Professions



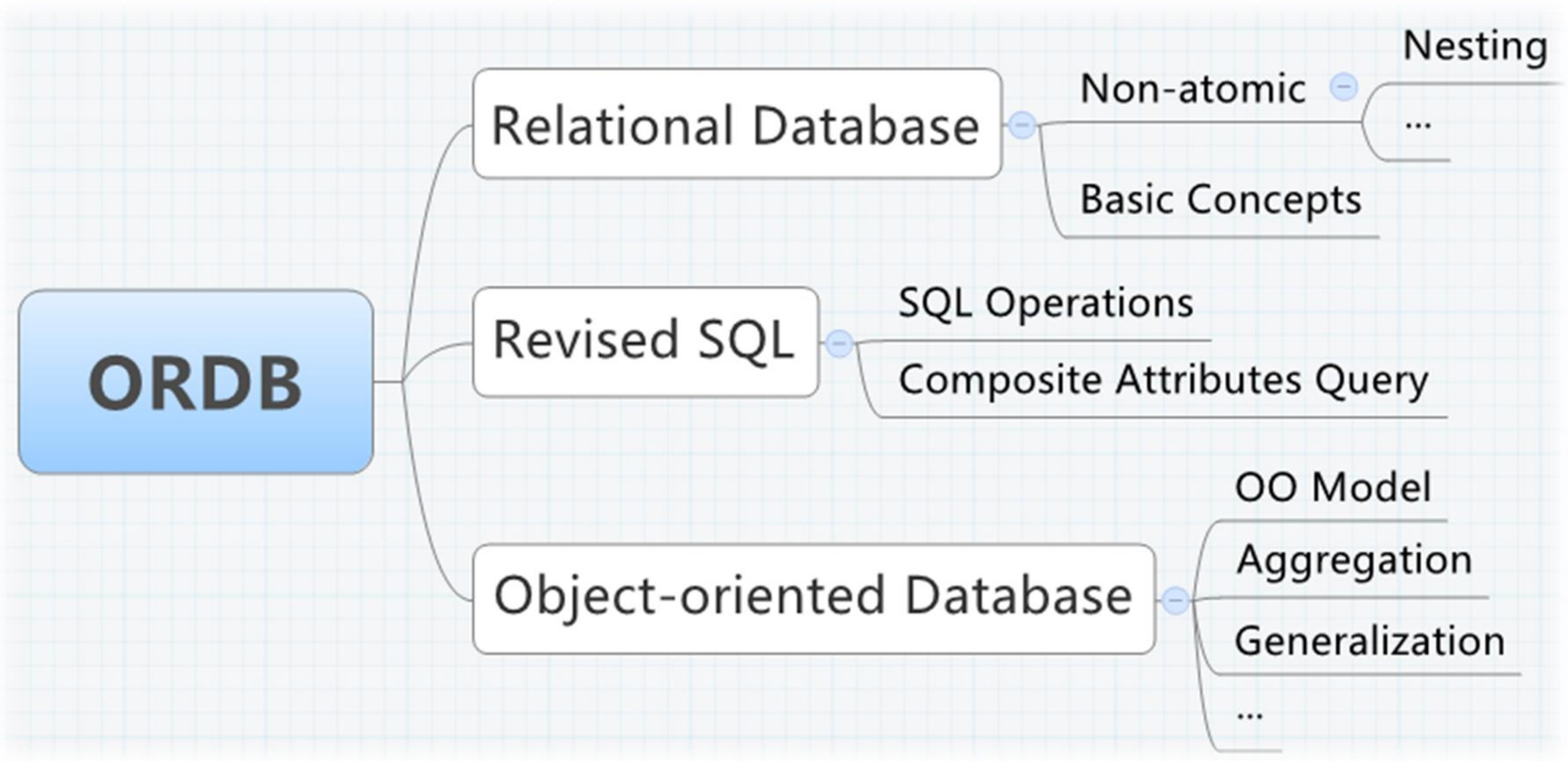
Hybrid modeling: Methodology

- Limited Modeling Language



Union of object-oriented languages rather than a real object-oriented database.

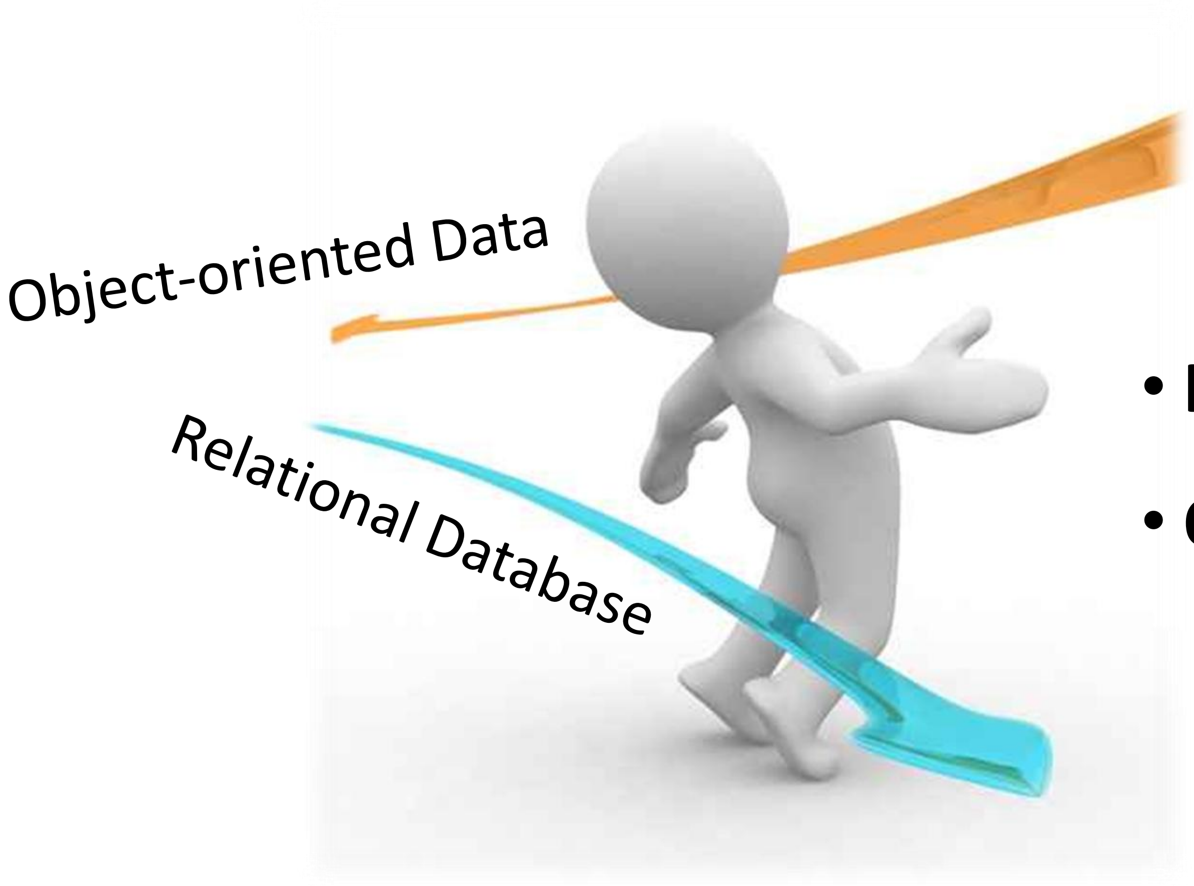
Hybrid modeling: Methodology



Hybrid modeling: Methodology

```
class qa_event_notify {
function process_event($event, $userid, $handle, $cookieid,
    $params)
        {
                switch ($event) {
                        case 'q_post':...
                        case 'a_post':...
                        case 'c_post':...
                                ...
                                ...
                                ...
```

Hybrid modeling: Methodology



ORM

- **Impedance Mismatch**
- **Object CRUD**

Hybrid modeling: Methodology

```
//Create a new post in the database and return its ID
function qa_db_post_create($type, $parentid, $userid,
    $cookieid, $ip, $title, $content, $format, $tagstring, $notify,
    $categoryid=null)
```

```
function qa_db_post_acount_update($questionid)
function qa_db_category_path_qcount_update($path)
function qa_db_ifcategory_qcount_update($categoryid)
```

Content

Section	Topic
1	Relational model
2	ERM theory
3	ERD process
4	Object orientation
5	Hybrid modeling
6	Derived design concept

Slide position_1:10

Derived design concept: Goal

Code compression

Slide position_2:10

Derived design concept: Fundamental

- Object, class
- Encapsulation
- Inheritance
- Polymorphism

Derived design concept: Ideas

OOD addresses a bigger picture.

Ideas:

- Object oriented
- Re-usable
- Variable with minimal effort
- Extendable without change

Slide position_4:10

Derived design concept: Why

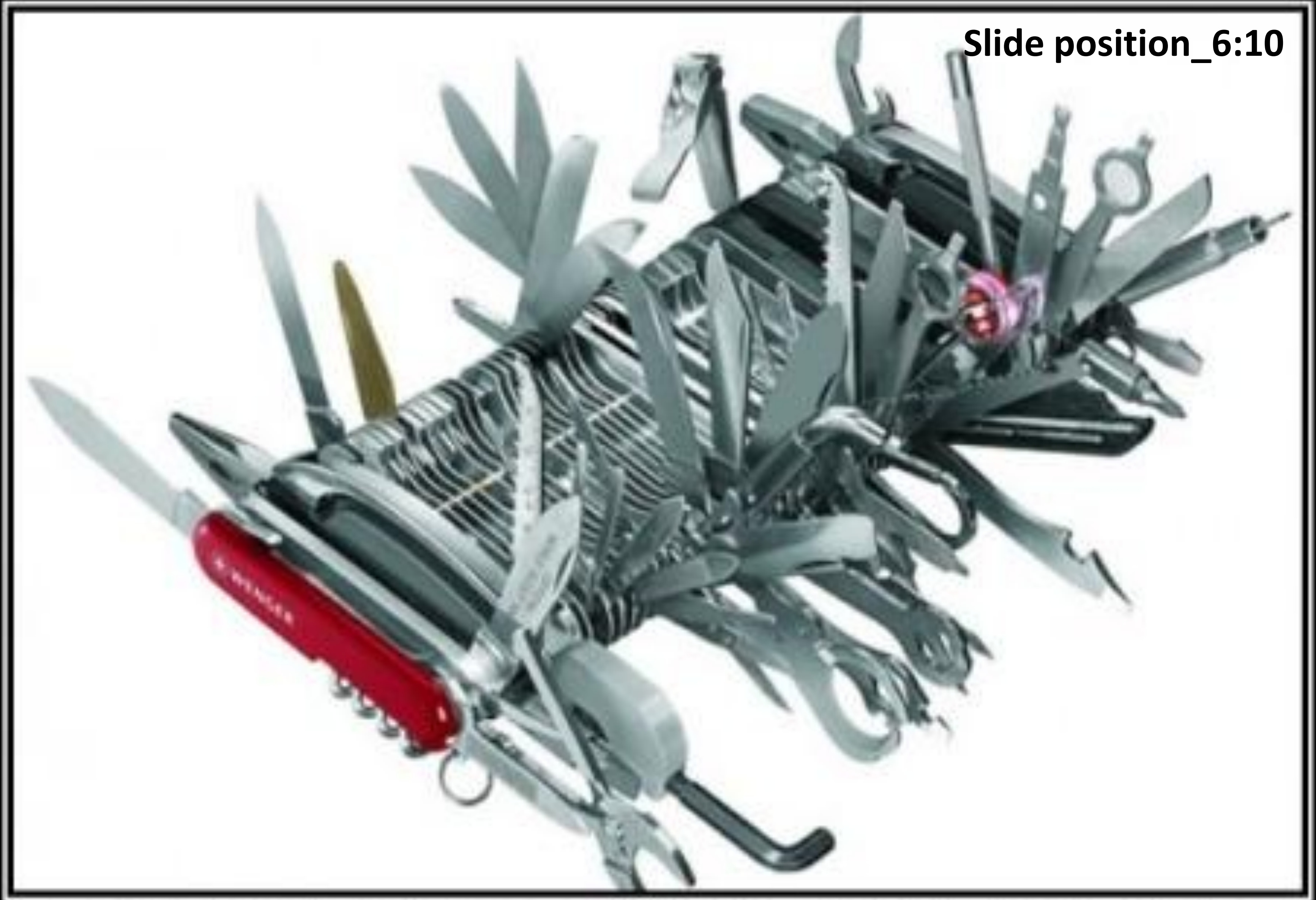
- "Walking on water and developing software from a specification are easy if both are frozen.“

- *Edward V. Berard*

Derived design concept: Principles

- **S** = Single Responsibility Principle
- **O** = Opened Closed Principle
- **L** = Liskov’s Substitution Principle
- **I** = Interface Segregation Principle
- **D** = Dependency Inversion Principle

Slide position_6:10



SINGLE RESPONSIBILITY PRINCIPLE

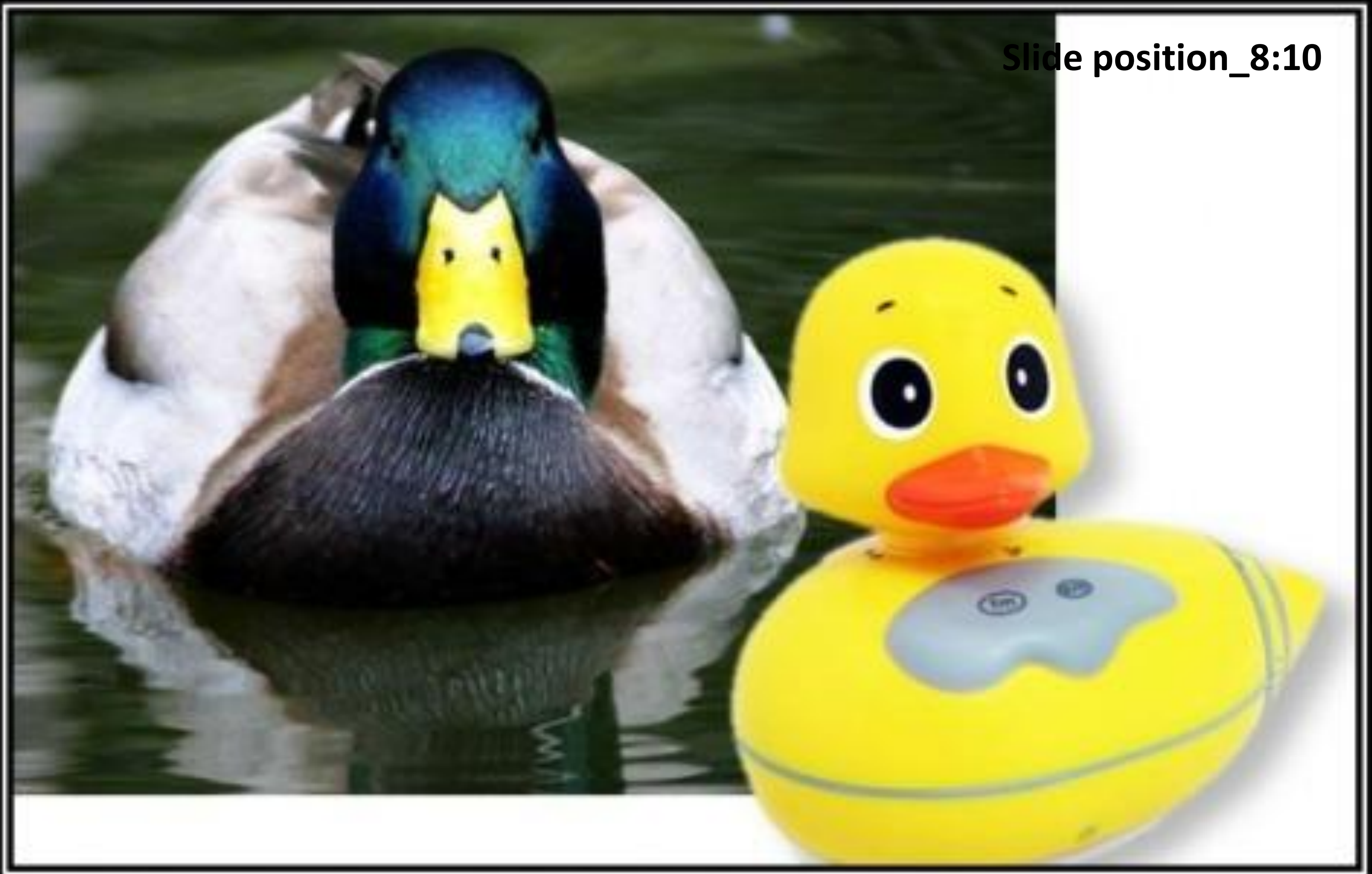
Just Because You Can, Doesn't Mean You Should

Slide position_7:10

OPEN CLOSED PRINCIPLE

Open Chest Surgery Is Not Needed When Putting On A Coat

Slide position_8:10



LISKOV SUBSTITUTION PRINCIPLE

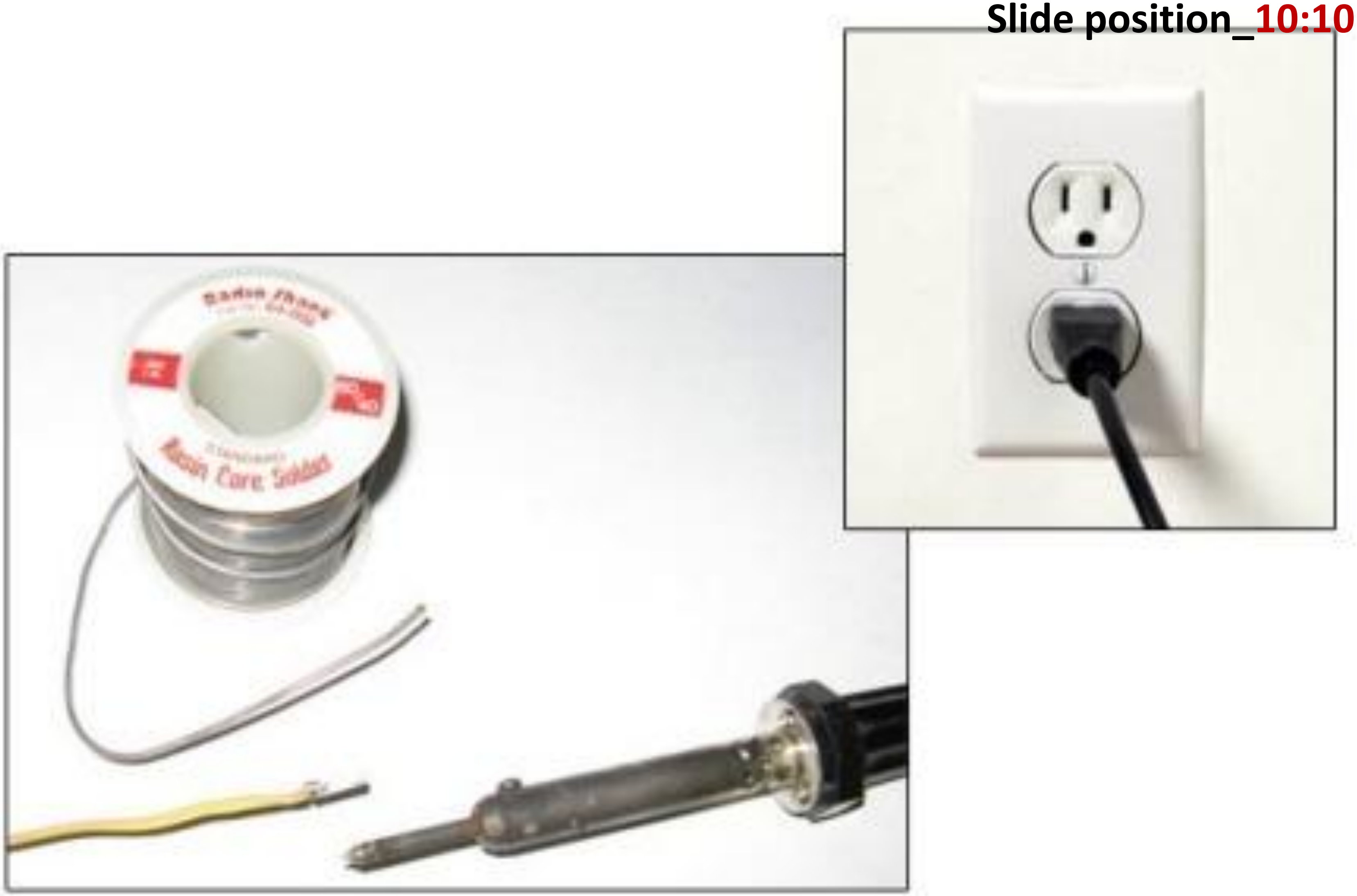
If It Looks Like A Duck, Quacks Like A Duck, But Needs Batteries - You Probably Have The Wrong Abstraction

Slide position_9:10

INTERFACE SEGREGATION PRINCIPLE

You Want Me To Plug This In, Where?

Slide position_10:10



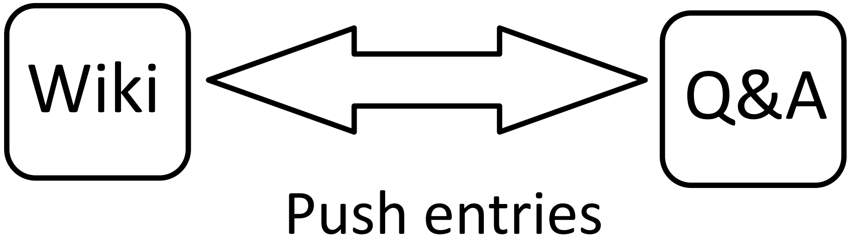
DEPENDENCY INVERSION PRINCIPLE

Would You Solder A Lamp Directly To The Electrical Wiring In A Wall?

Derived design concept: DbC

Design by Contract

- Pre-condition
- Post-condition
- class invariant

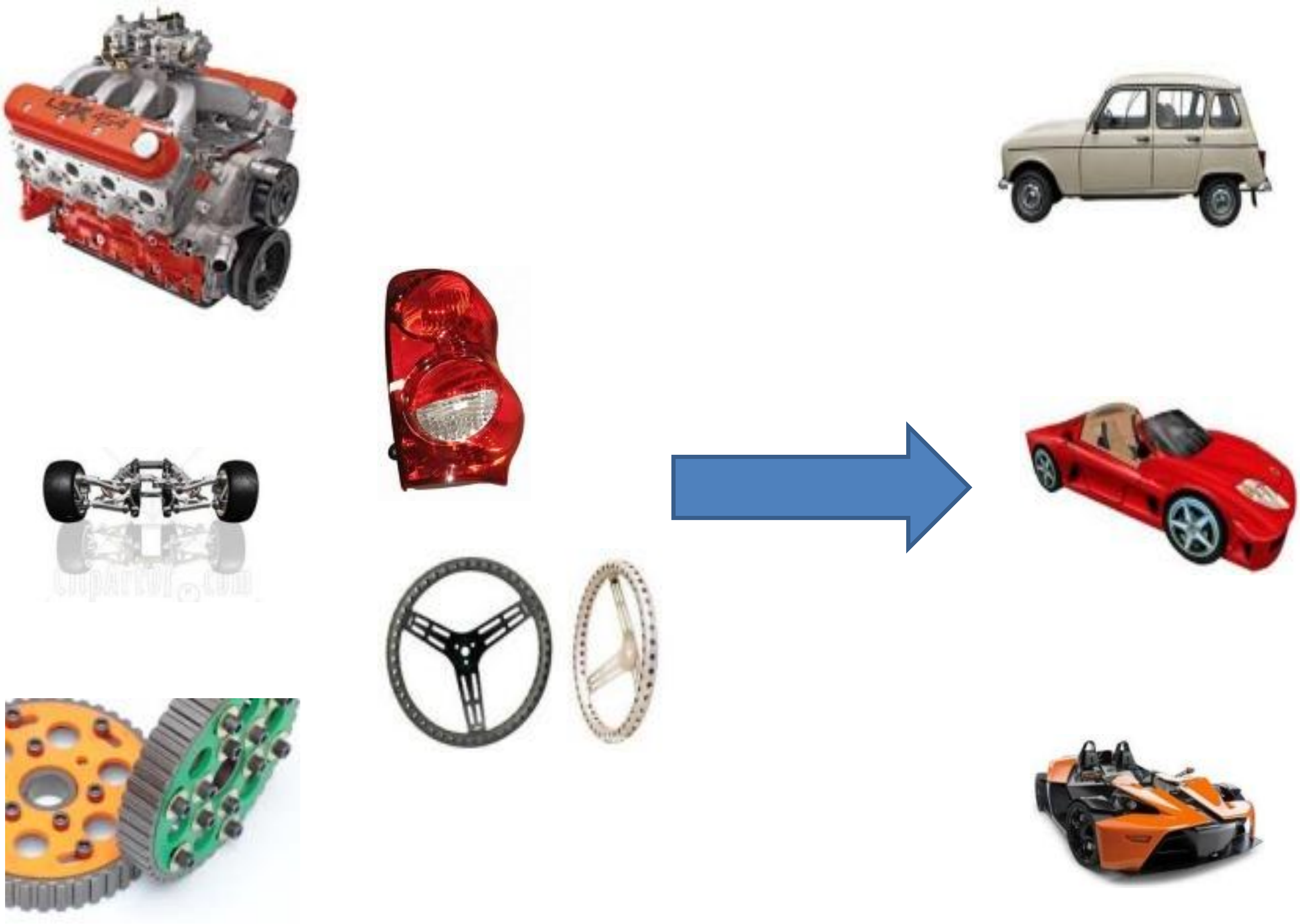


Slide position_1:4

Derived design concept: Design patterns

- Standardized design
- OOD principles

Slide position_2:4



Derived design concept: Design patterns

Basic elements

- Pattern name
- Problem
- Solution
- Consequences

Slide position_4:4