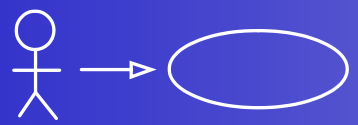
为把控AI而做的领域建模

分析模型

更新时间：2026. 08

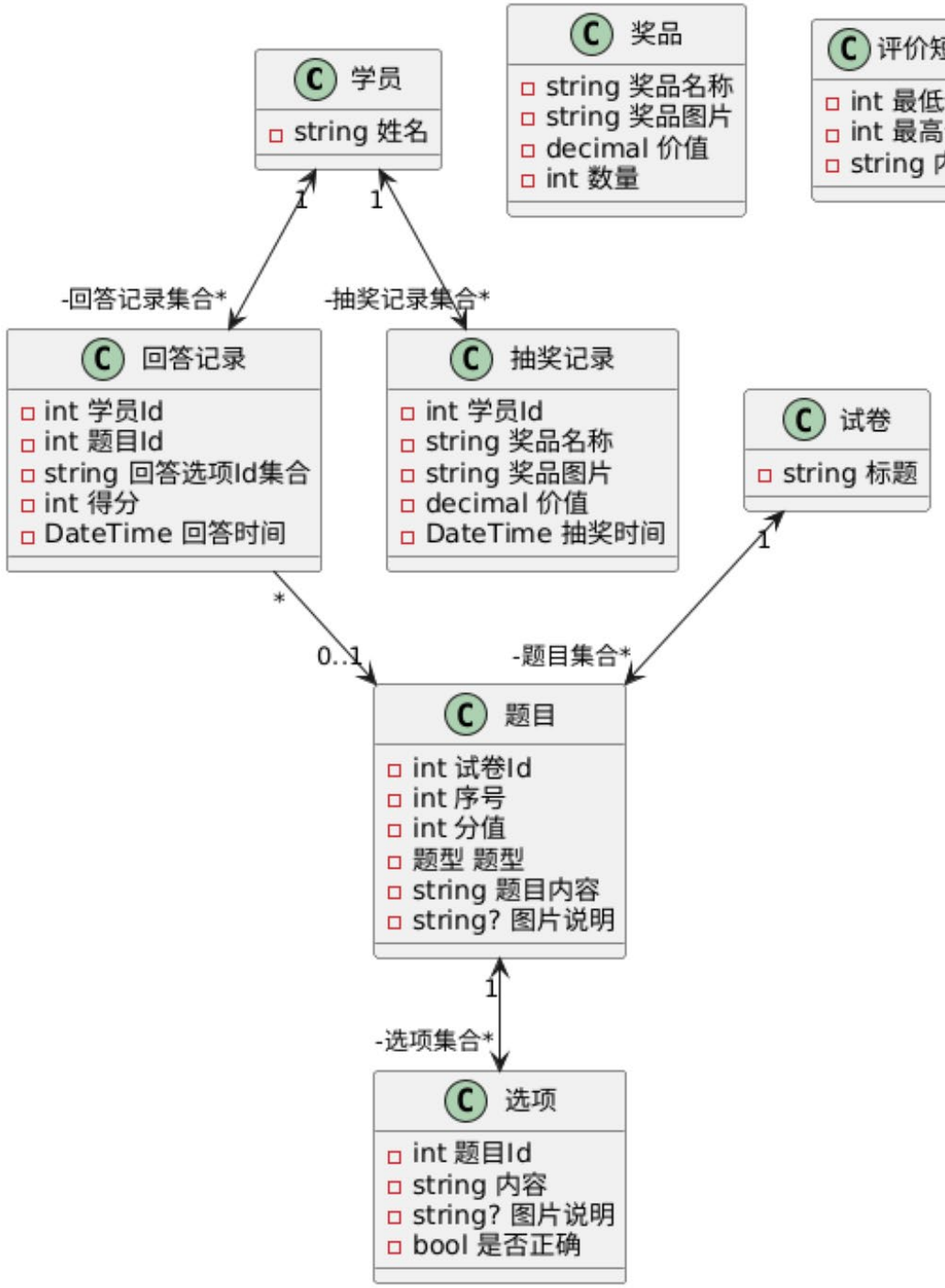
潘加宇



AI的结果

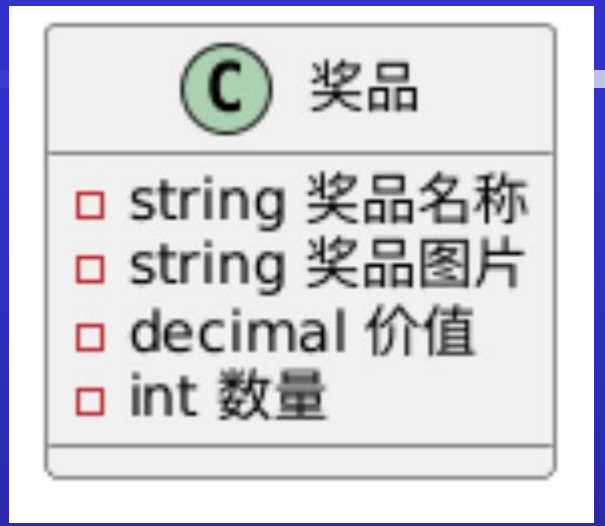
从生成的代码逆向

得到的分析类图



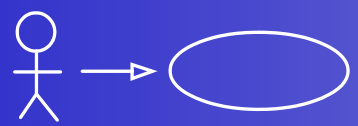
AI的结果

- 删除异常
  - 全部抽掉→奖品数据从系统中消失
- 插入异常
  - 不在奖池里的奖品怎么办
- 命名冗余（小问题）
  - 属性名前带类名



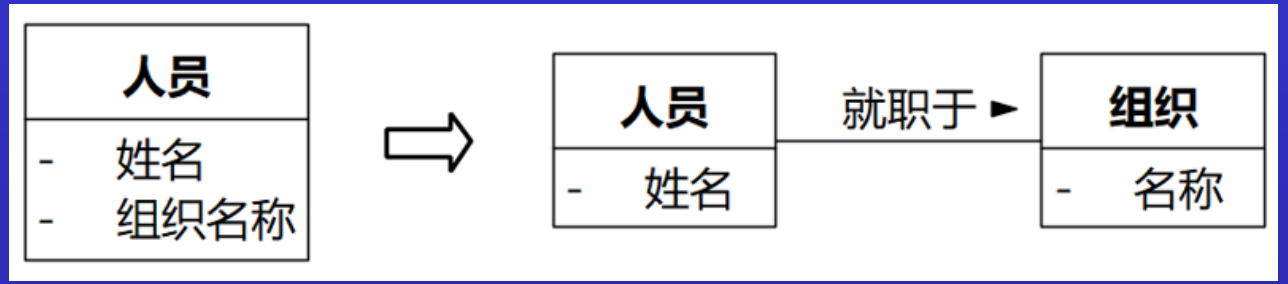
“数量”的存在
意味着 其实是规格
这是一个对象

ID	名称	图片	价值	数量
1	百事可乐罐装330ml	……	3.5	2
2	红牛250ml	……	6	1
3	现金5.12元	……	5.12	1
4	乐事原味桶装薯片104g	……	8	1
5	脆香米巧克力24g	……	4.5	3

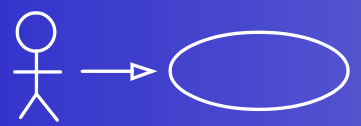


类和属性

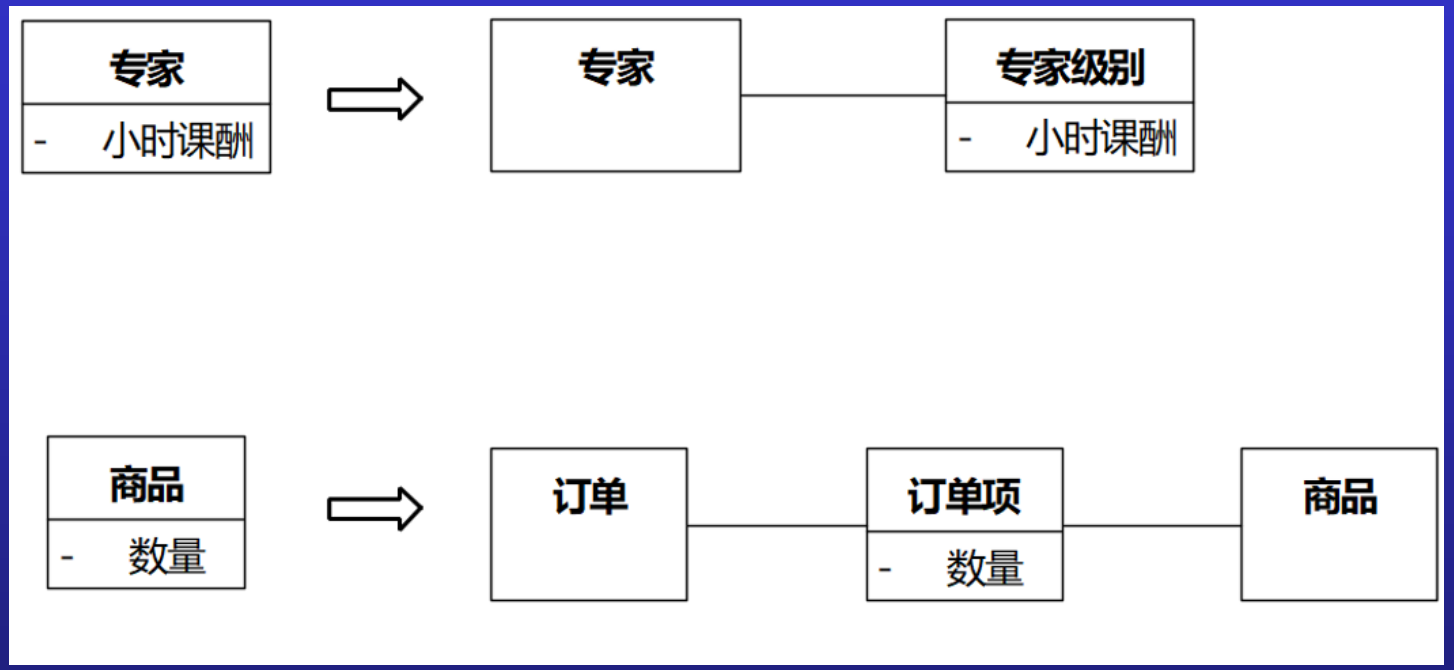
- 什么的什么
- x→y，且不存在x→a→y 。
- 思考：
  - “类”能决定“属性”吗？
  - 如果能，是怎么决定的？
  - 如果不能，还需要什么？
  - （a, b, c）→y，谁能把（a, b, c）圈进来？



审查：属性是否直接描述类



类和属性



- 指定专家为潘**，小时课酬能定吗？
  - 能，2000元。
- 怎么决定的？
  - 潘**的级别是一级专家，一级专家课酬是2000元。存在传递依赖：专家→级别→课酬
- 指定某个商品，购买数量能定吗？
  - 不能。
- 还需要什么？
  - 还需要知道是哪个订单
  - 添加“订单项”

审查：属性是否直接描述类

类和属性

复习GJ-002

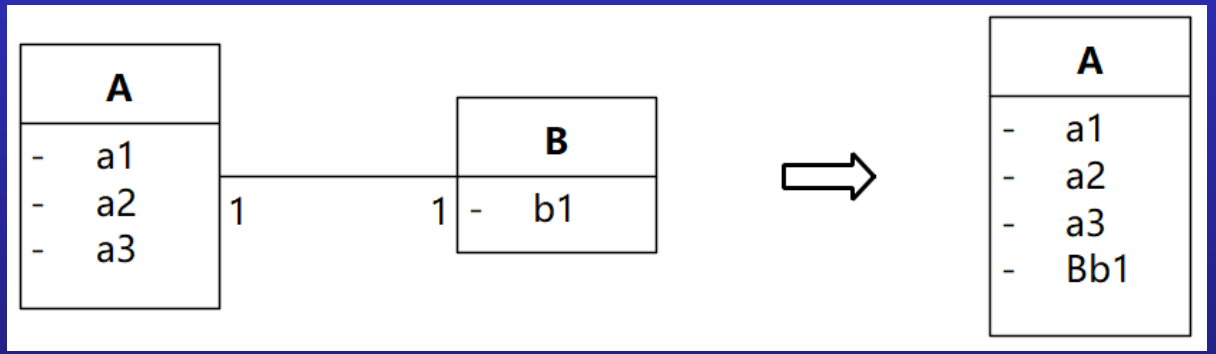
- 违反的后果
  - 大量不同对象的某些属性值相同

345677: 人员
姓名 = 张大龙 组织名称 = 腾讯科技（深圳）有限公司

345679: 人员
姓名 = 王力红 组织名称 = 北京字节跳动科技有限公司

345679: 人员
姓名 = 罗大凤 组织名称 = 腾讯科技（深圳）有限公司

- 可以合并的情况
  - A和B存在1对1的关联（多对1合适吗）
  - 且B只有一个属性

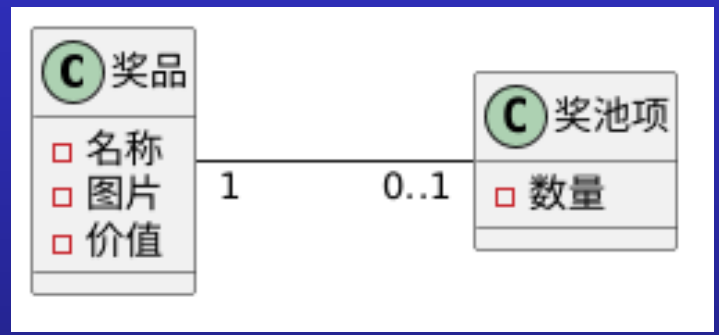


审查：属性是否直接描述类

AI的结果

- 奖品 → 数量 ×
- （奖品，奖池） → 数量 ✓
- 添加“奖池项”
- 奖池只需一个
  - 删除“奖池”
  - 多重性变为0..1

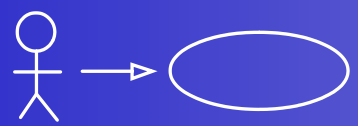
```
@startuml
left to right direction
class 奖品 {
 - 名称
 - 图片
 - 价值
}

class 奖池项 {
 - 数量
}

class 奖池 {
}

奖品 "1" -- "*" 奖池项
奖池项 "*" -- "1" 奖池
@enduml
```



奖池项

ID	奖品ID	数量
100	1	2
101	2	1
102	3	1

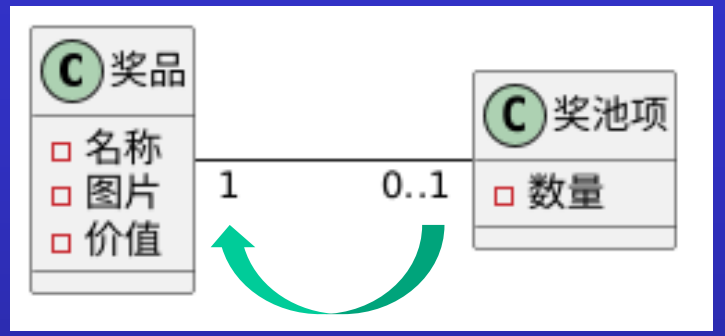
修正

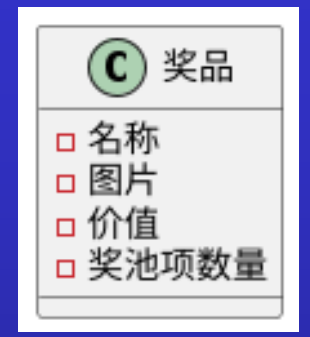


AI的结果

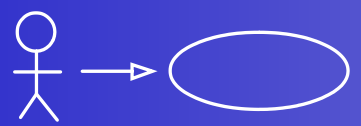
- 把“奖池项”合并到“奖品”
- AI的结果“灵活变通”岂不是也能用？
- 问题出在0..1
- 大量的数量为0
  - 含义模糊：为什么是0
  - 类似情况：空字符串、NULL

“灵活变通”？





ID	名称	图片	价值	数量
1	百事可乐罐装330ml	……	3.5	2
2	红牛250ml	……	6	1
3	现金5.12元	……	5.12	1
4	乐事原味桶装薯片104g	……	8	0
5	脆香米巧克力24g	……	4.5	0
6	没抽中	……	0	4
7	人月神话	……	48	0
8	现金2.56元	……	2.56	0
9	企业应用架构模式	……	98	0
10	莫斯利安酸奶	……	5	0
…	…	……	….	0

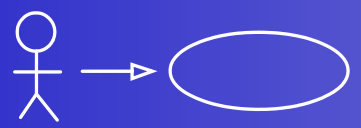


复习GJJ-001

集合论和关系代数的知识片段

≈34分钟

（ 178分钟完整视频随本课程录像附赠）



幂集（Powerset）

- $P(S) = \{x \mid x \subseteq S\}$
  - S的所有可能子集组成的集合
- $(A \in P(B)) \Leftrightarrow (A \subseteq B)$
- $(\#A = n) \Leftrightarrow (\# P(A) = 2^n)$

$$(1+1)^n = C_n^0 + \ldots\ldots + C_n^m + \ldots\ldots + C_n^n = 2^n$$

V := {1,2,3}

W := {1,{2,3},4}

P(V) = { ∅, {1},{2},{3}, {1,2},{1,3},{2,3}, {1,2,3}}

P(W) = { ∅, {1},{{2,3}},{4}, {1,{2,3}},{1,4},{{2,3},4}, {1,{2,3},4}}

{2,3} ∈ P(V)

{2,3} ∈ W

{1,{2,3}} ∈ P(W)

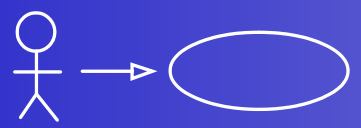
P (∅) = {∅}

有序对（Ordered Pair）

- 集合：{a, b}={b, a}
- 一对元素，顺序有意义：（a, b）≠ （b, a）

<a,b>

- $(a \neq b) \Leftrightarrow ((a,b) \neq (b,a))$
- $((a,b) = (c,d)) \Leftrightarrow ((a = c) \wedge (b = d))$
- $((a,b) \neq (c,d)) \Leftrightarrow ((a \neq c) \vee (b \neq d))$
- $\pi_1(a,b)=a \quad \pi_2(a,b)=b$



笛卡尔积（Cartesian Product）

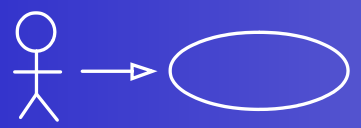
- $A \times B = \{ (a,b) \mid a \in A \wedge b \in B \}$.
- 从集合A中选择第一个坐标和从集合B中选择第二个坐标来构造的所有有序对的集合
- $(A \times B = B \times A) \Leftrightarrow (A = B)$
- $\#(A \times B) = \#A * \#B$
- $A \times \varnothing = \varnothing \times A = \varnothing$

A := {1,2,3}

B := {3,4}

A×B = { (1,3), (1,4), (2,3), (2,4), (3,3), (3,4) }

B×A = { (3,1), (3,2), (3,3), (4,1), (4,2), (4,3) }



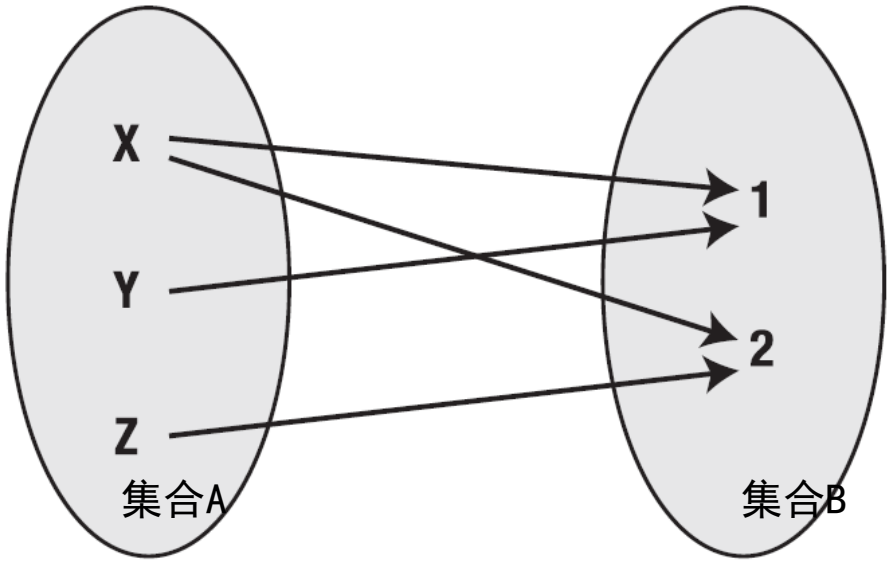
关系

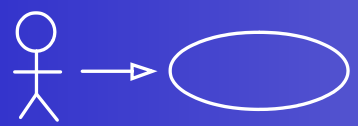
- 和以下不是一个东西
  - UML里的类的关系（泛化、关联、依赖）
  - E/R（实体/关系）图的R（关系）
- 定义：笛卡尔积A×B 的子集
  - (∀p∈R1: p∈A×B)
- R是从集合A到集合B的二元关系 ⇔ R∈P(A×B)

A := {X,Y,Z}

B := {1,2}

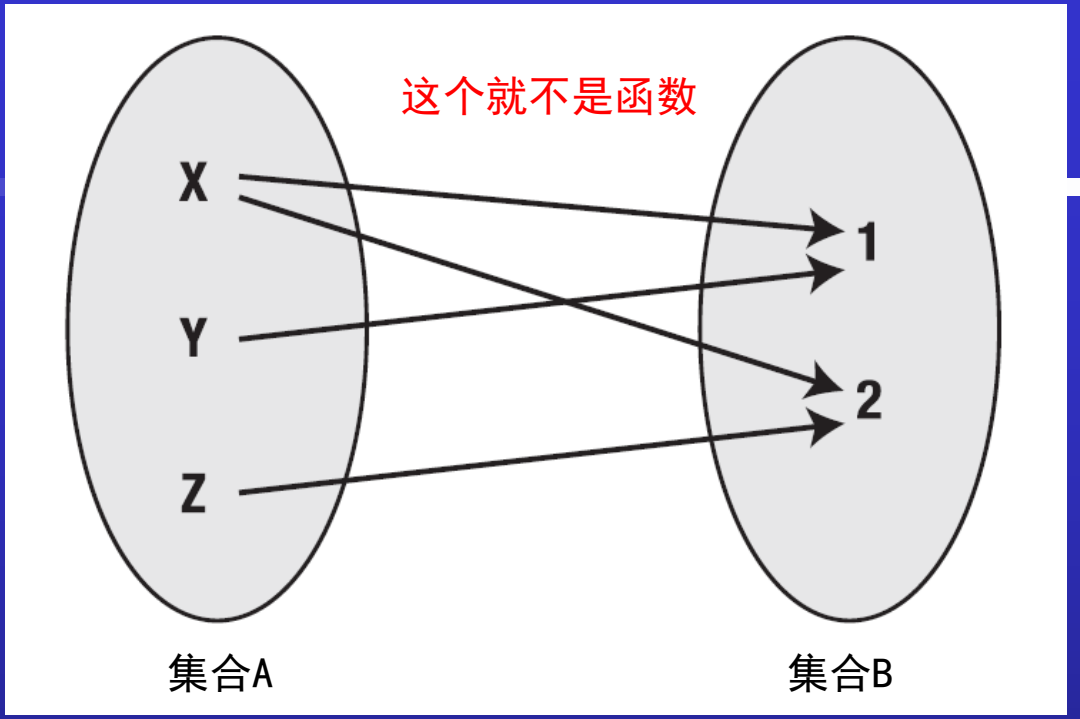
R1 := { (X,1), (X,2), (Y,1), (Z,2) }





函数

复习GJJ-001 1:12:20-1:19:00



- 没有两个元素有相同第一坐标的二元关系
- 集合中的每个元素都恰好有一条出发的箭头，

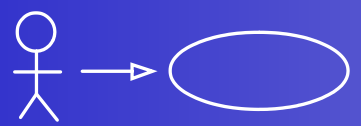
该二元关系是在该集合上的函数

- F是A上的函数⇔

$$F \in P(A \times B) \wedge (\forall p1,p2 \in F: p1 \neq p2 \Rightarrow \pi_1(p1) \neq \pi_2(p2)) \wedge \{\pi_1(p) \mid p \in F\} = A$$

对于属于函数F的有序对p1和p2，如果p1和p2是不同的对，则它们应该有不同的第一坐标

A的每个元素都必须作为第一坐标出现



表（Table）

- 共享相同定义域的函数集合
- T是H上的表⇔(t∈T: t是H上的函数)
- H：标题；函数：元组
- 元组的顺序任意
- 元组内部有序对顺序任意

T1 :=

{ { (零件编号,1), (名称,'螺栓'), (库存,22), (价格,10) }

,{ (零件编号,2), (名称,'螺钉'), (库存,19), (价格,5) }

,{ (零件编号,3), (名称,'斧'), (库存,0), (价格,30) }

,{ (零件编号,4), (名称,‘锯'), (库存,4), (价格,15) }

,{ (零件编号,5), (名称,'扳手'), (库存,7), (价格,20) }

,{ (零件编号,6), (名称,‘剪'), (库存,32), (价格,5) } }

T2 := { { (X,2), (Y,1) }, { (Y,8), (X,0) }, { (Y,10), (X,5) } }

是表

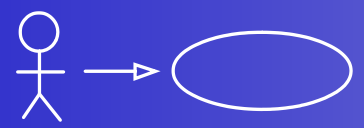
T3 := { { (partno,3), (name,’螺栓') }, { (pno,4), (pname,'nail') } }

不是表

T4 := { { (empno,105), (ename,'Mrs. Sparks'), (born,'03-apr-1970') }

,{ (empno,202), (ename,'Mr. Tedesco') } }

不是表

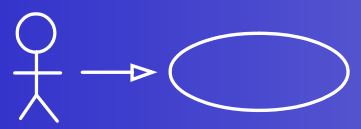


范式

- 符合范式，所有数据都可以通过基本运算获取
- 先懂得规范，才有资格谈不规范
- 很多教科书例子……稍幼稚

- 第一范式（1NF）：原子属性
- 第二范式（2NF）：消除非主属性的部分依赖
- 第三范式（3NF）：消除传递依赖
- 巴斯-科德范式（BCNF）：消除主属性的部分依赖
- 第四范式（4NF）：……&@￥%%
- 第五范式（5NF）：@#￥&*%……

5NF⊂4NF⊂BCNF⊂3NF⊂2NF⊂1NF



3NF

复习GJJ-001 2:28:40-2:35:40

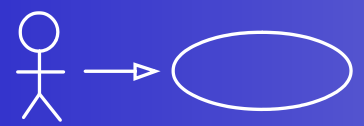
员工ID→部门ID→部门名称

- 3NF：不允许存在传递的函数依赖
  - A→B→C

- 问题
  - 插入:新增一个暂时没有员工的部门
  - 删除:删除ID为1、2、5的员工，部门信息消失
  - 修改:财务部改名资金管理部，员工1、2、5相应信息都需要修改

员工ID	员工姓名	部门ID	部门名称
1	罗永浩	1	财务部
2	罗志祥	1	财务部
3	罗大佑	2	行政部
4	罗玉凤	2	行政部
5	罗振宇	1	财务部

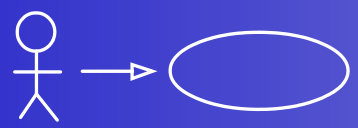
员工ID	员工工号	员工姓名	部门ID
1	2007101	罗永浩	1
2	2008201	罗志祥	1
3	2008203	罗大佑	2
4	2009301	罗玉凤	2
5	2010101	罗振宇	1



应用

复习GJJ-001 2:45:00-2:49:30

- 建模思路
  - 领域知识
    - 根据对领域知识的理解分解类和属性
    - 遵守《软件方法》里的要点
  - 观察和思考
    - 观察对象的属性值
    - 逐个思考各个概念之间的依赖，按照规则分解



应用

复习GJJ-001 2:45:00-2:49:30

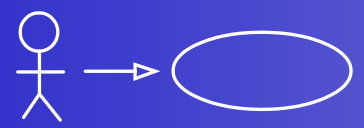
- 去掉领域概念的加成，从证据来判断，这个类的属性合理吗（对象的标识隐含）？

A
- A1 - A2 - A3 - A4

A1	A2	A3	A4
x1	x2	x3	x4

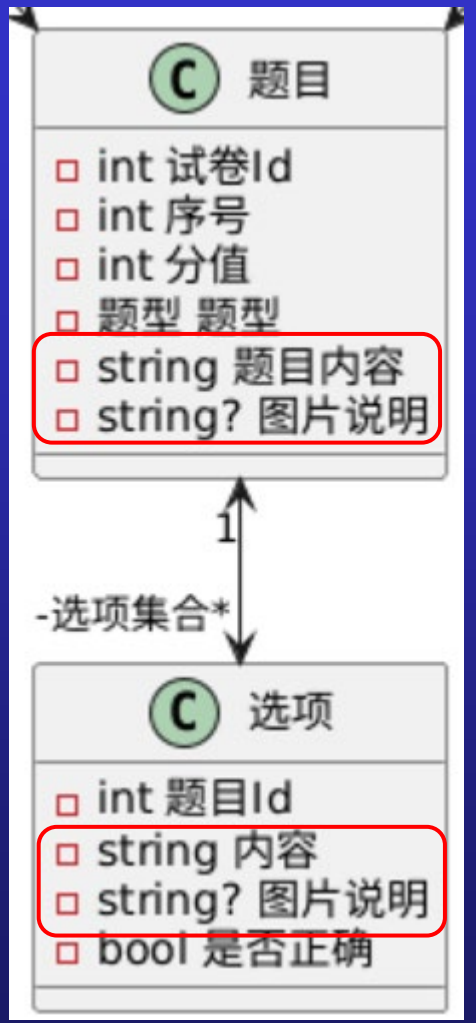
A1	A2	A3	A4
x1	x2	x3	x4
x5	x6	x7	x8

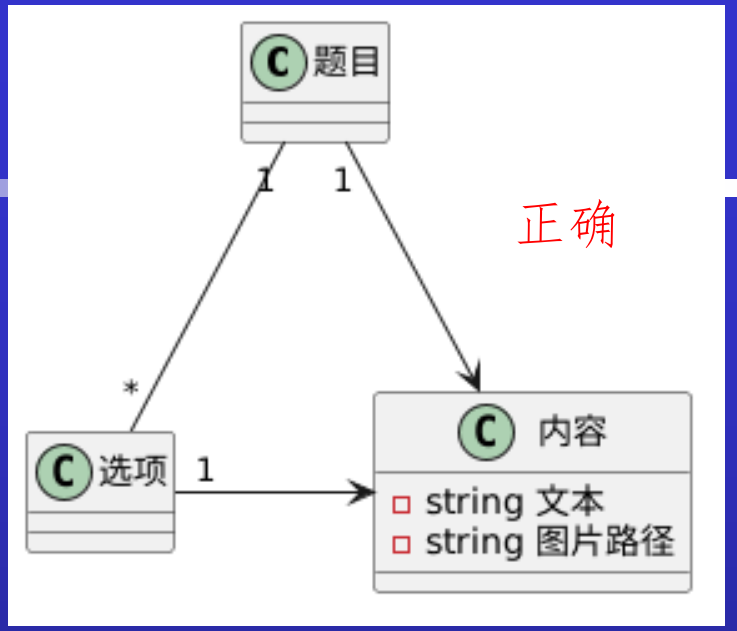
A1	A2	A3	A4
x1	x2	x3	x4
x5	x6	x7	x8
x5	x9	x10	x8
x5	x11	x10	x8

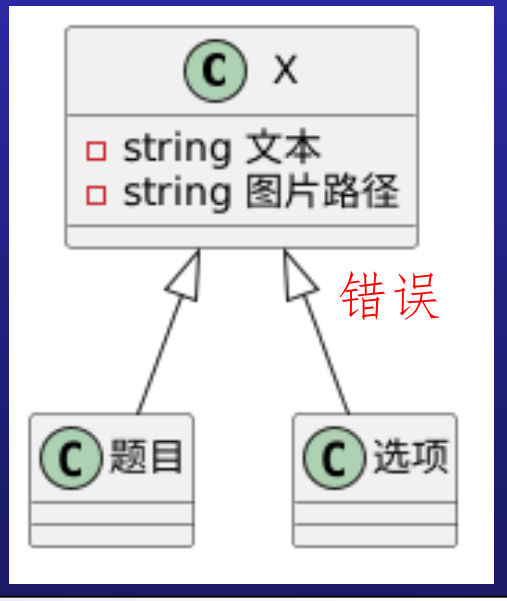


AI的结果

- 题目和选项
  - 看似小问题，很有迷惑性
  - 重复属性
  - 泛化还是关联
    - 给X造个词，还是需要“内容”
    - X直接叫“内容”呢？

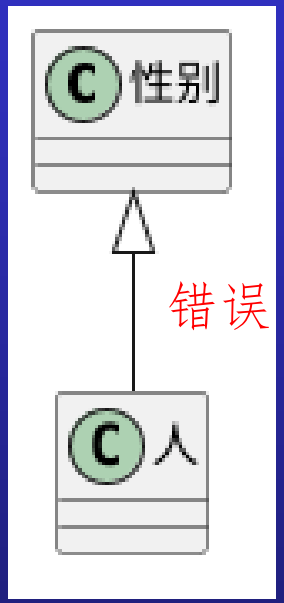






AI的结果

- 泛化误用
  - 集合的划分
    - 自行复习GJJ-001视频58:50
  - 人是性别的子类吗
    - 无意识做了投影
    - 人是整数的子类吗
  - A可能会有多个B吗
    - 能消除绝大部分



{男，女}

一个人的性别

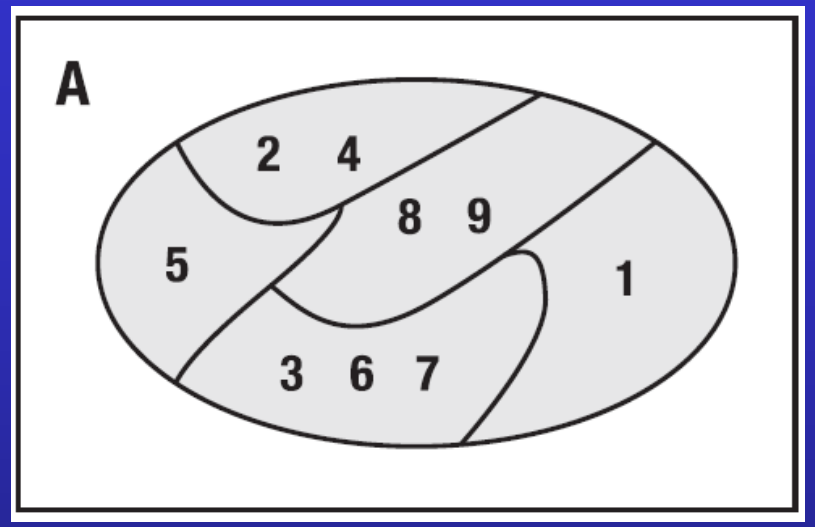
要么男，要么女

必然是{男，女}的子集

划分（Partition）

S := {1,2,3,4,5,6,7,8,9}
P := {{1},{2,4},{3,6,7},{5},{8,9}}

- 如果S是一个非空集合，那么P是S的一个划分，当且仅当P是由互不相交的S非空子集构成的集合，并且它们的并集等于S。
- (P是S的划分) ⇔ (P ⊆ P(S)–{∅}) ∧ (如果A,B∈P，有 (A≠B ⇒ (A∩B)=∅)) ∧ (∪P = S)
- S的每个元素出现且仅出现在P的一个元素中



{T := {1,2,3}

P1 := {{1},{2},{3}}
P2 := {{1,2},{3}}
P3 := {{1,3},{2}}
P4 := {{2,3},{1}}
P5 := {{1,2,3}}

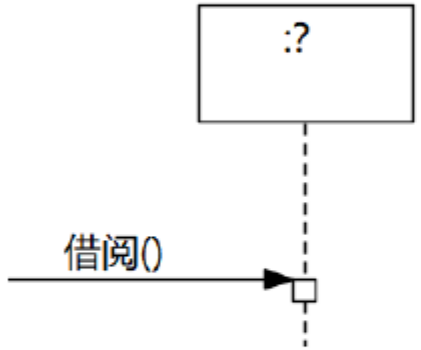
复习GJJ-001 58:50

AI的结果

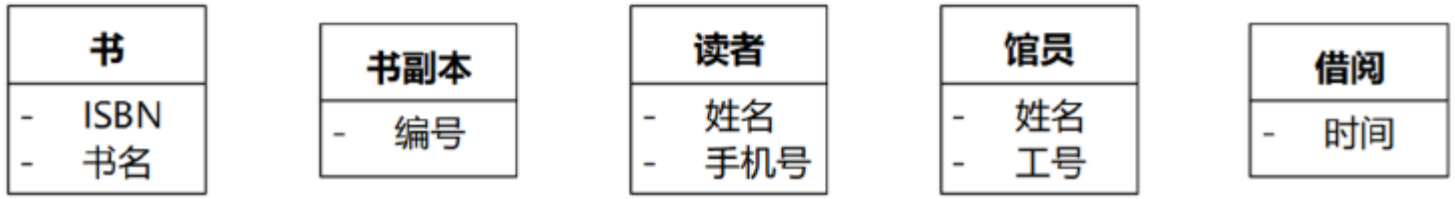
- 无法满足
  - 多个文字图片交替

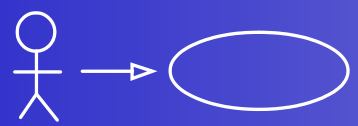
图书馆里，读者拿着书到出纳台找馆员办理借阅手续，馆员使用信息系统“图书馆管理系统”为读者办理借阅（还比较落后，没有办法做到自助借阅）。

如果这个“图书馆管理系统”是用面向对象的方法学分析和设计的，在分配责任时，可能会遇到这样的情况：



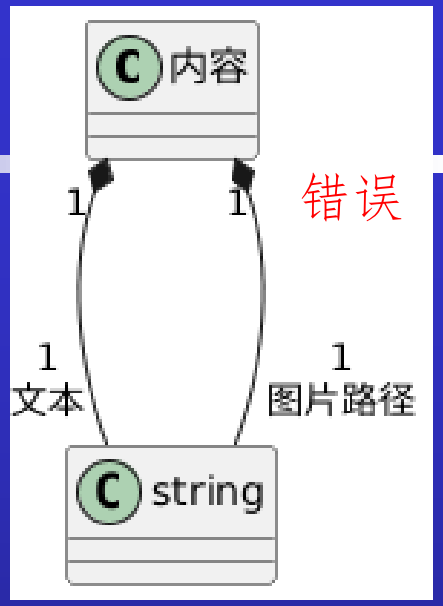
按照我们对常见的图书馆业务的理解，图中？的地方不会是以下哪些类（以下类中已去掉关联，只保留原生类型的属性）？

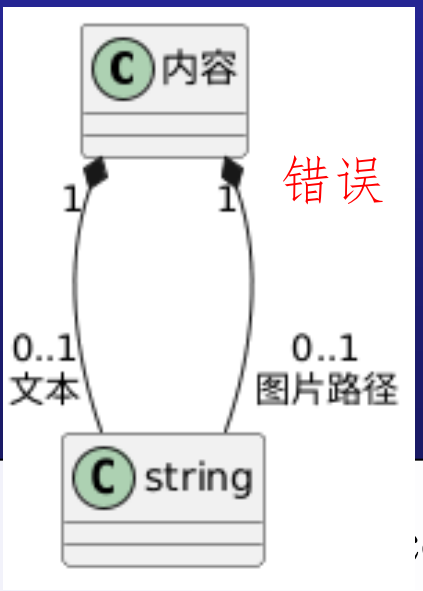


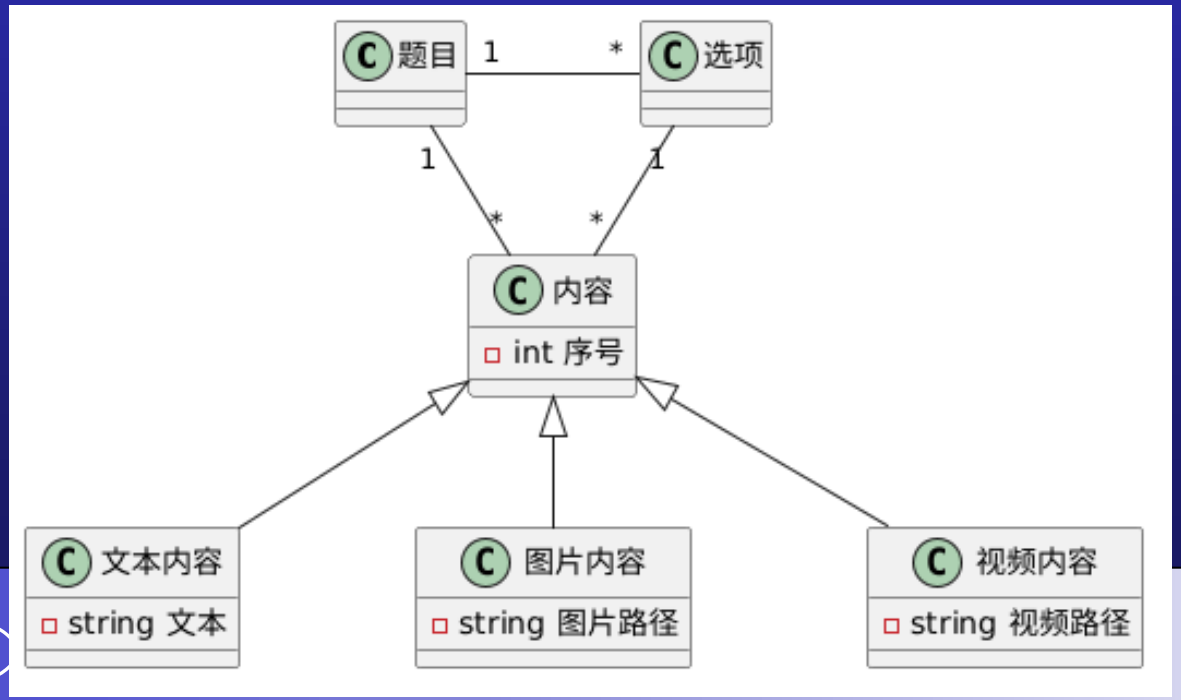


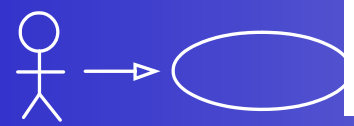
AI的结果

- 关联误用
  - 有的无图片，有的无文字
  - 更多类型的内容（视频……）



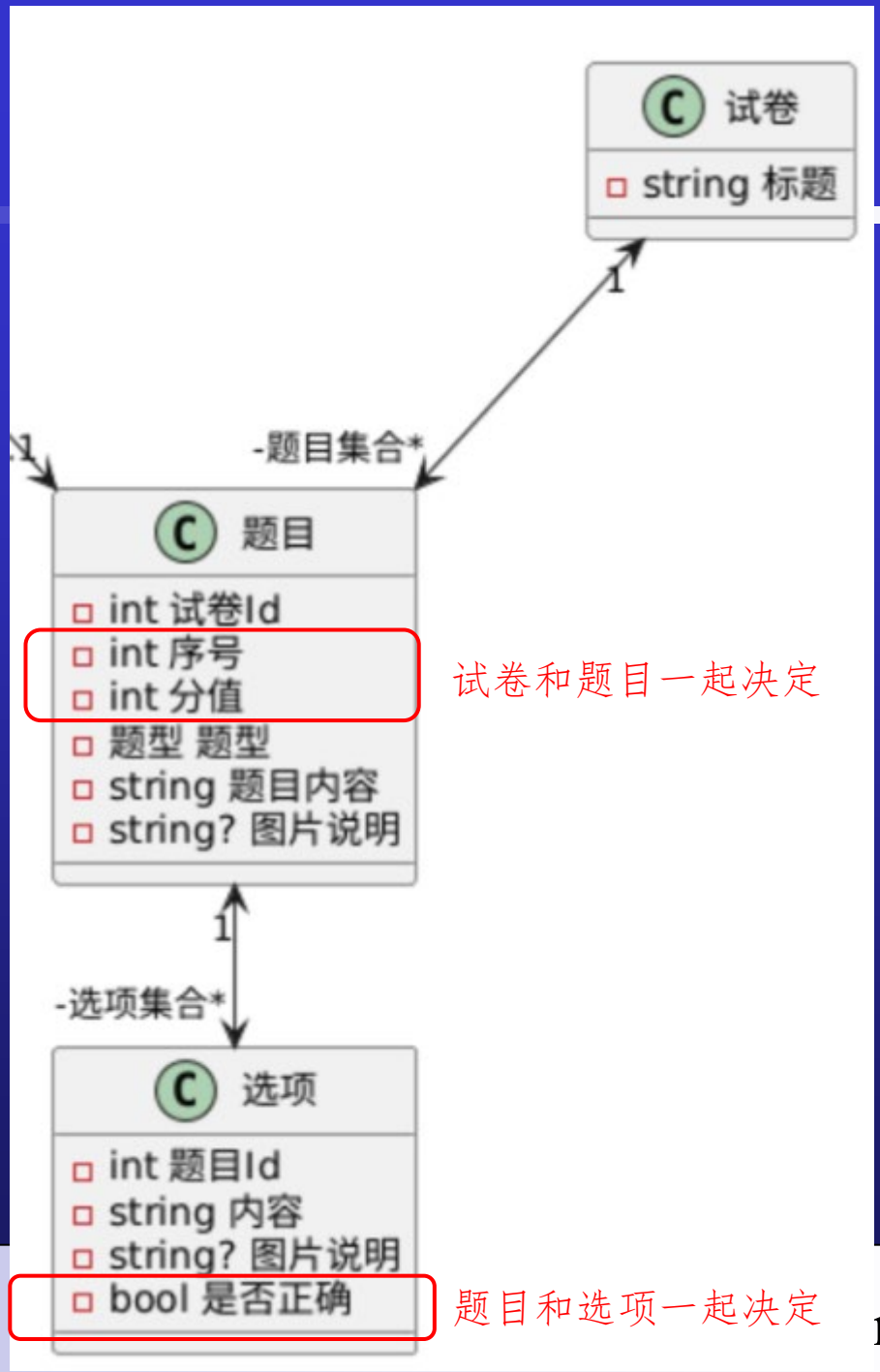


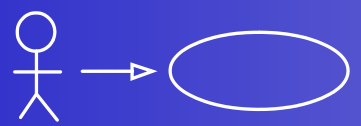




AI的结果

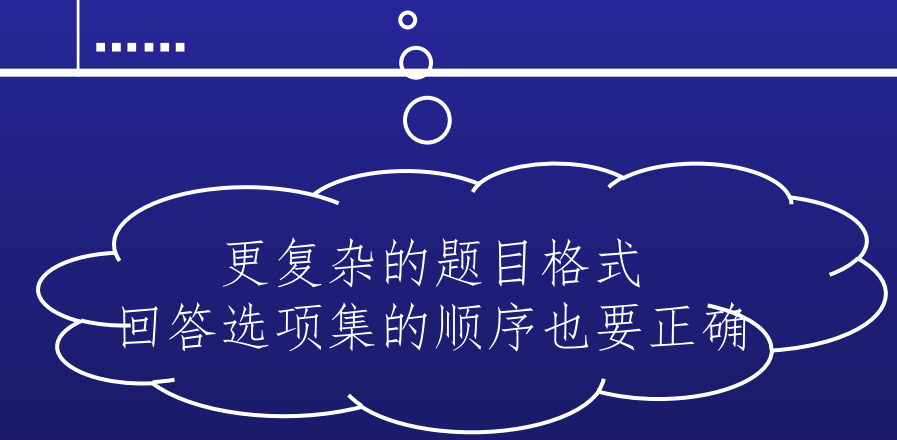
- 1对多还是多对多
  - 选项离开题目是否有意义
  - 选项经常需要被题目共享
  - 题目离开试卷是否有意义
  - 题目经常需要被试卷共享

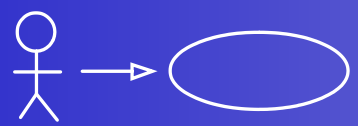




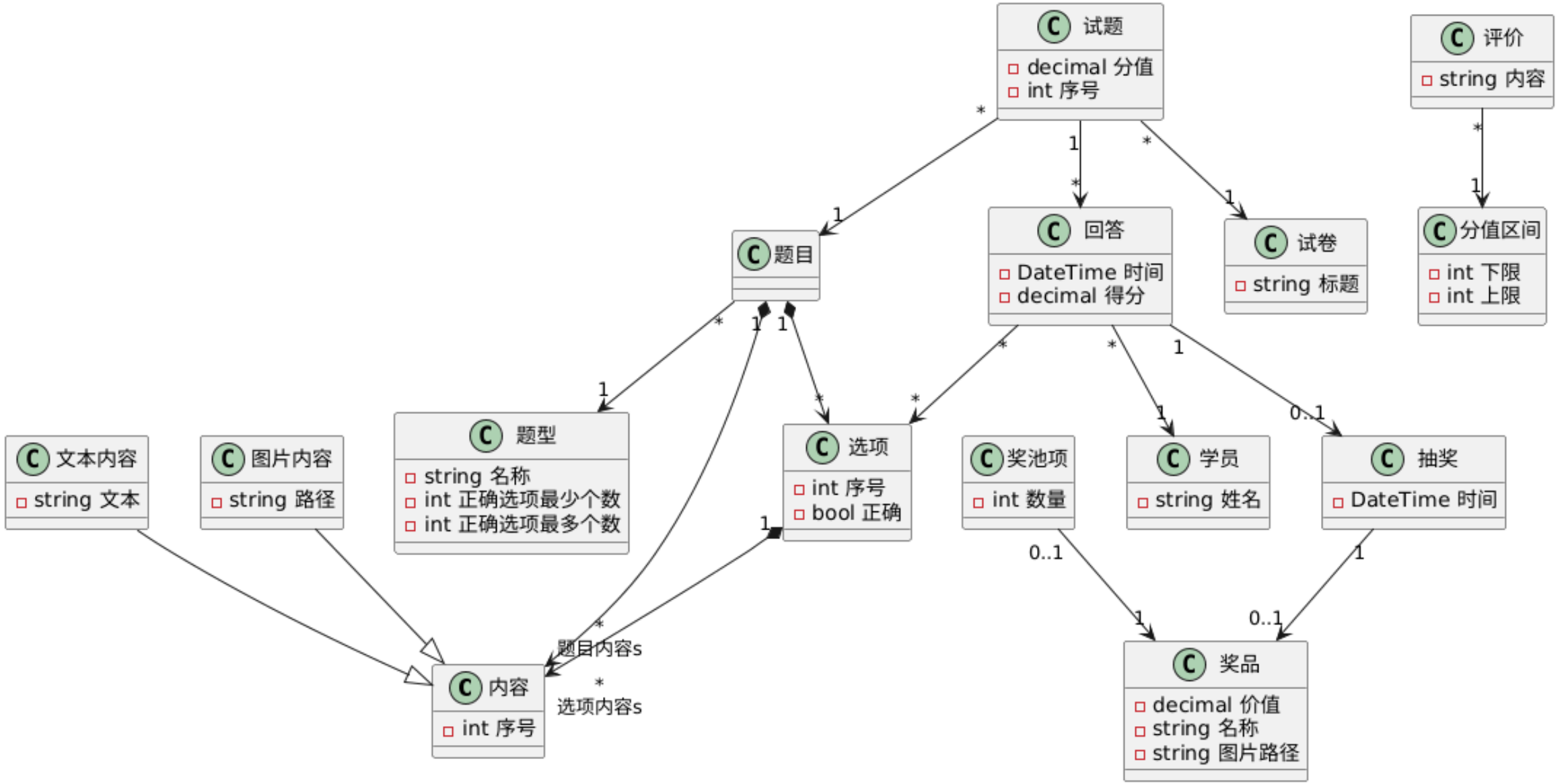
AI的结果

以医院信息系统为研究对象，合理选项是_____	以医院为研究对象，合理选项是_____	以医院信息系统为研究对象，合理选项是_____，以医院为研究对象，合理选项是_____
A “患者→看病”是用例 B “患者→挂号”是用例 ……	A “患者→看病”是用例 B “患者→挂号”是用例 ……	A “患者→看病”是用例 B “患者→挂号”是用例 ……



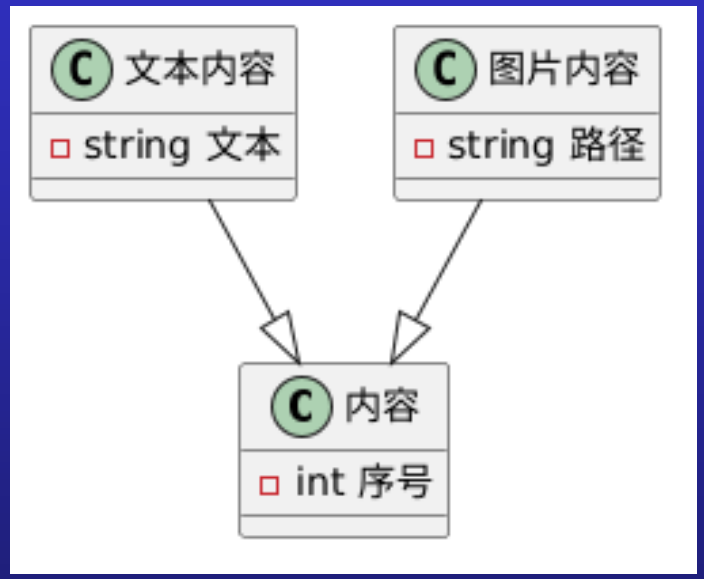


类图修正

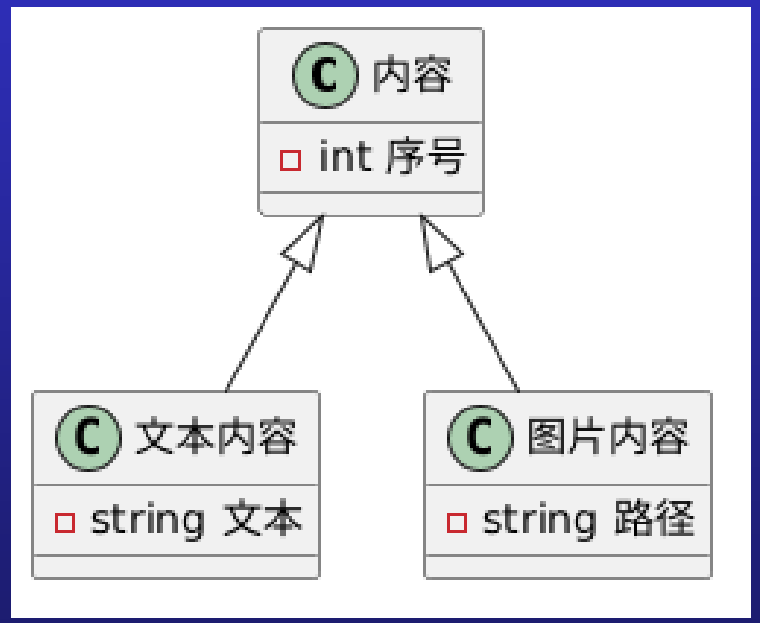

试题
decimal 分值
int 序号
评价
string 内容
题目
回答
DateTime 时间
decimal 得分
试卷
string 标题
分值区间
int 下限
int 上限
题型
string 名称
int 正确选项最少个数
int 正确选项最多个数
选项
int 序号
bool 正确
文本内容
string 文本
图片内容
string 路径
奖池项
int 数量
学员
string 姓名
抽奖
DateTime 时间
内容
int 序号
题目内容s
选项内容s
奖品
decimal 价值
string 名称
string 图片路径

类图语法

```
class 内容  类
{ 私有
- int 序号      属性
}
class 文本内容
{
- string 文本
}
class 图片内容
{
- string 路径
}
            泛化
文本内容 --|>内容
图片内容 --|>内容
```




```
文本内容 -up-|>内容
图片内容 -up-|>内容
```



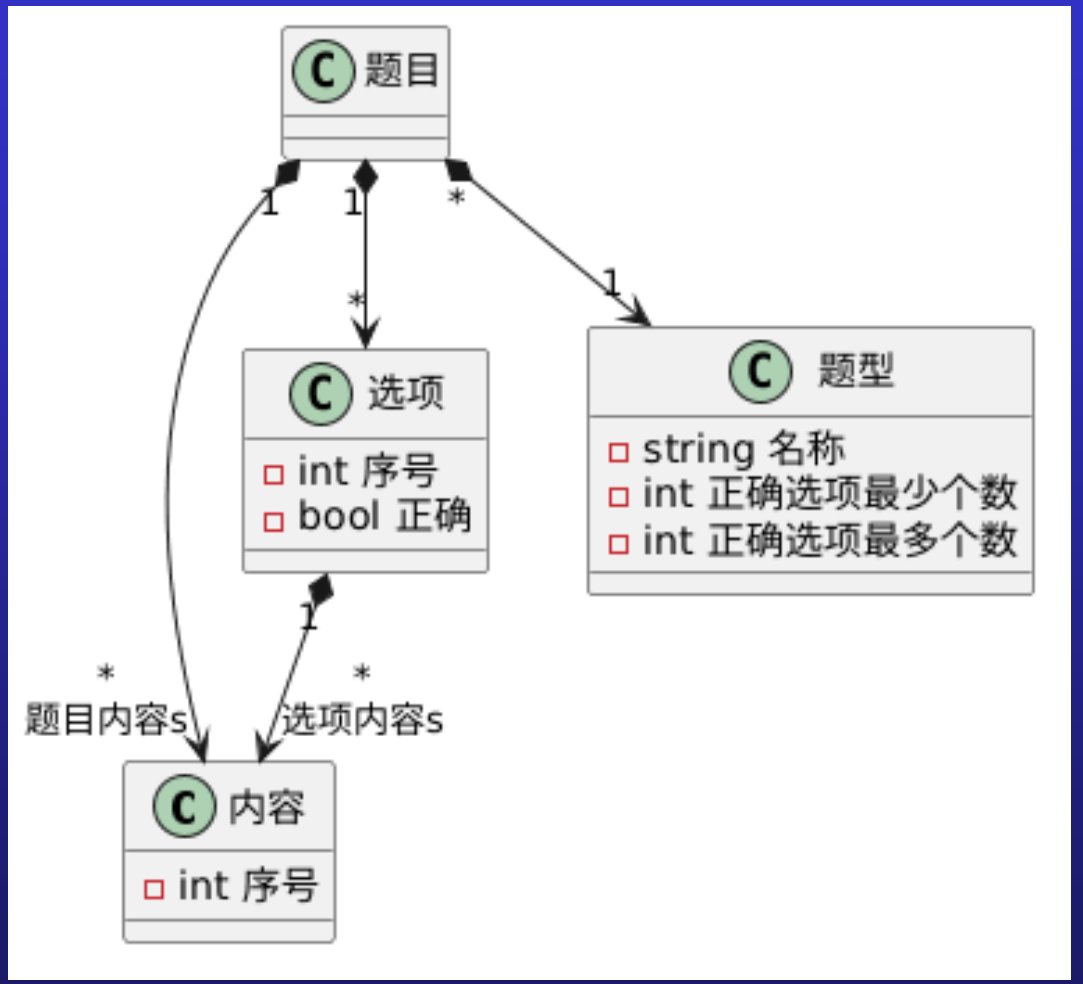
类图语法

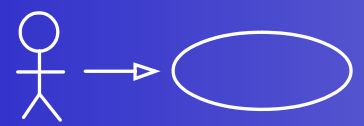
```
class 题目

class 选项
{
- int 序号
- bool 正确
}
class 题型
{
- string 名称
- int 正确选项最少个数
- int 正确选项最多个数
}

        组合
题目 "*" *--> "1" 题型
题目 "1" *--> "*" 选项
题目 "1" *--> "*\n题目内容s" 内容
选项 "1" *--> "*\n选项内容s" 内容
```

方向

角色名和多重性放一起

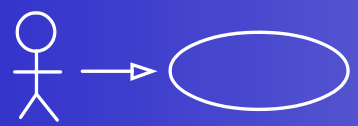
换行



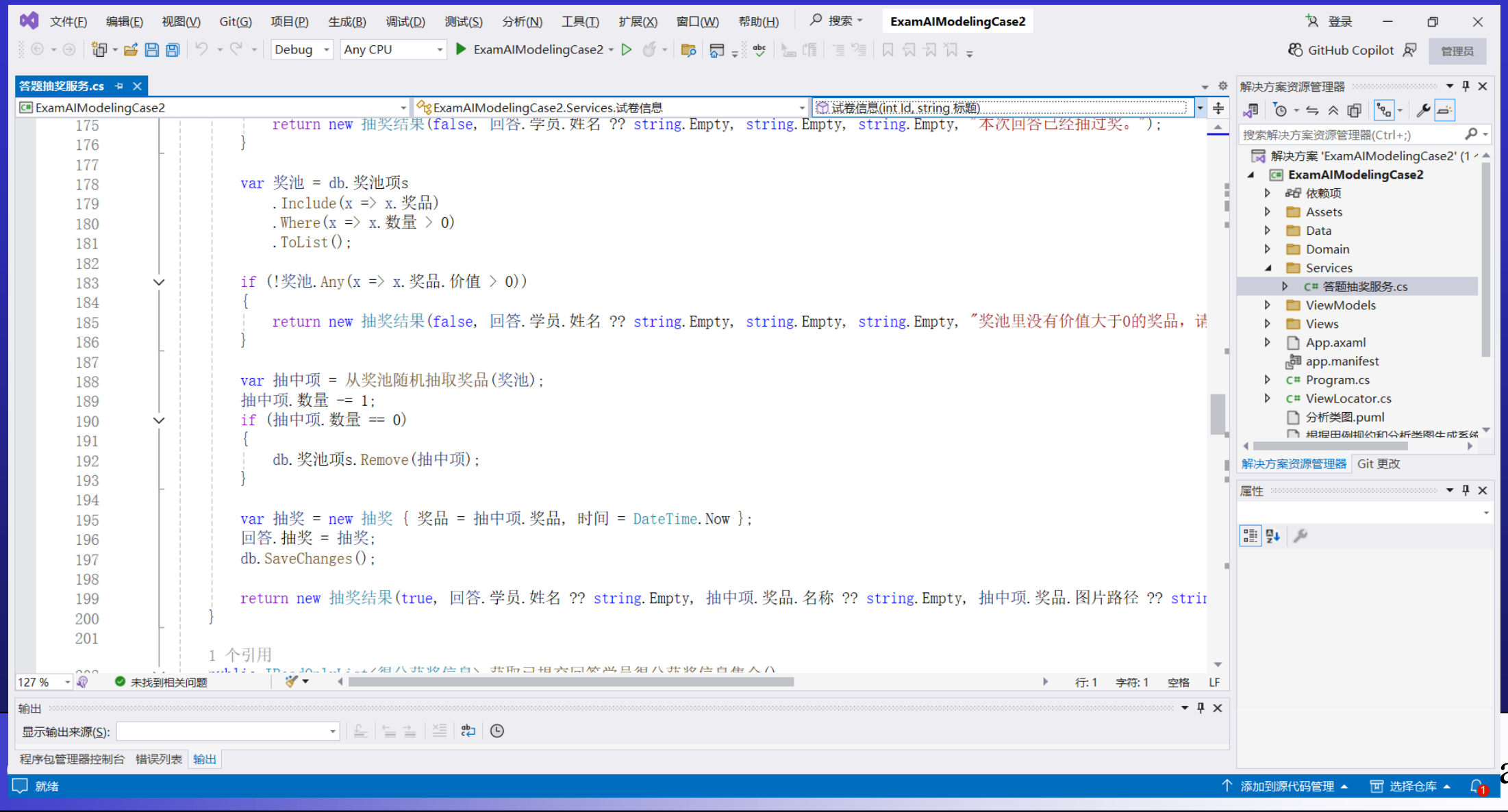


用例规约+分析类图

再给AI看看效果



AI

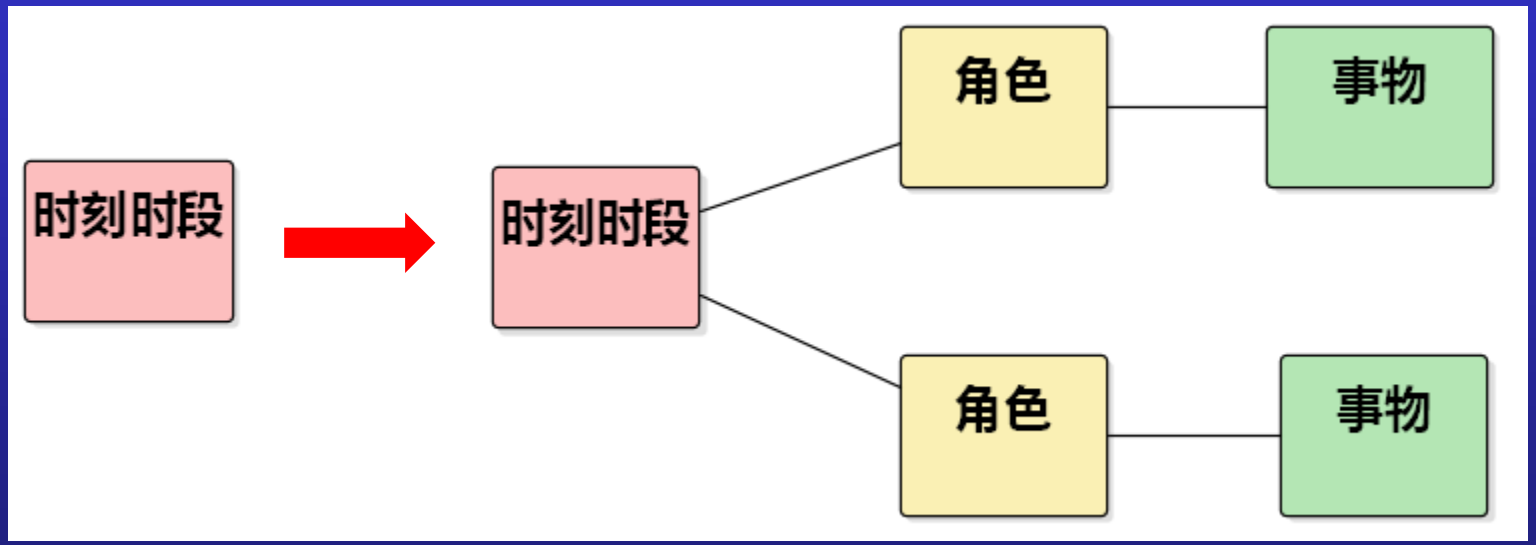

文件(F) 编辑(E) 视图(V) Git(G) 项目(P) 生成(B) 调试(D) 测试(S) 分析(N) 工具(T) 扩展(X) 窗口(W) 帮助(H)
搜索
ExamAIModelingCase2
登录
Debug
Any CPU
ExamAIModelingCase2
GitHub Copilot
管理员
答题抽奖服务.cs
ExamAIModelingCase2
ExamAIModelingCase2.Services.试卷信息
试卷信息(int Id, string 标题)
return new 抽奖结果(false, 回答.学员.姓名 ?? string.Empty, string.Empty, string.Empty, "本次回答已经抽过奖。");
}
var 奖池 = db.奖池项s
.Include(x => x.奖品)
.Where(x => x.数量 > 0)
.ToList();
if (!奖池.Any(x => x.奖品.价值 > 0))
{
return new 抽奖结果(false, 回答.学员.姓名 ?? string.Empty, string.Empty, string.Empty, "奖池里没有价值大于0的奖品，请
}
var 抽中项 = 从奖池随机抽取奖品(奖池);
抽中项.数量 -= 1;
if (抽中项.数量 == 0)
{
db.奖池项s.Remove(抽中项);
}
var 抽奖 = new 抽奖 { 奖品 = 抽中项.奖品, 时间 = DateTime.Now };
回答.抽奖 = 抽奖;
db.SaveChanges();
return new 抽奖结果(true, 回答.学员.姓名 ?? string.Empty, 抽中项.奖品.名称 ?? string.Empty, 抽中项.奖品.图片路径 ?? strir
}
1 个引用
127 %
未找到相关问题
行: 1 字符: 1 空格 LF
解决方案资源管理器
搜索解决方案资源管理器(Ctrl+;)
解决方案 'ExamAIModelingCase2' (1
ExamAIModelingCase2
依赖项
Assets
Data
Domain
Services
答题抽奖服务.cs
ViewModels
Views
App.axaml
app.manifest
Program.cs
ViewLocator.cs
分析类图.puml
解决方案资源管理器 Git 更改
属性
输出
显示输出来源(S):
程序包管理器控制台 错误列表 输出
就绪
添加到源代码管理
选择仓库

类建模

- 建模思路
  - 复习GJ-002
- 警惕搅浑水
  - 把（自己掌握的）非核心域引入核心域
  - 搞出各种“技巧”

值得一提的

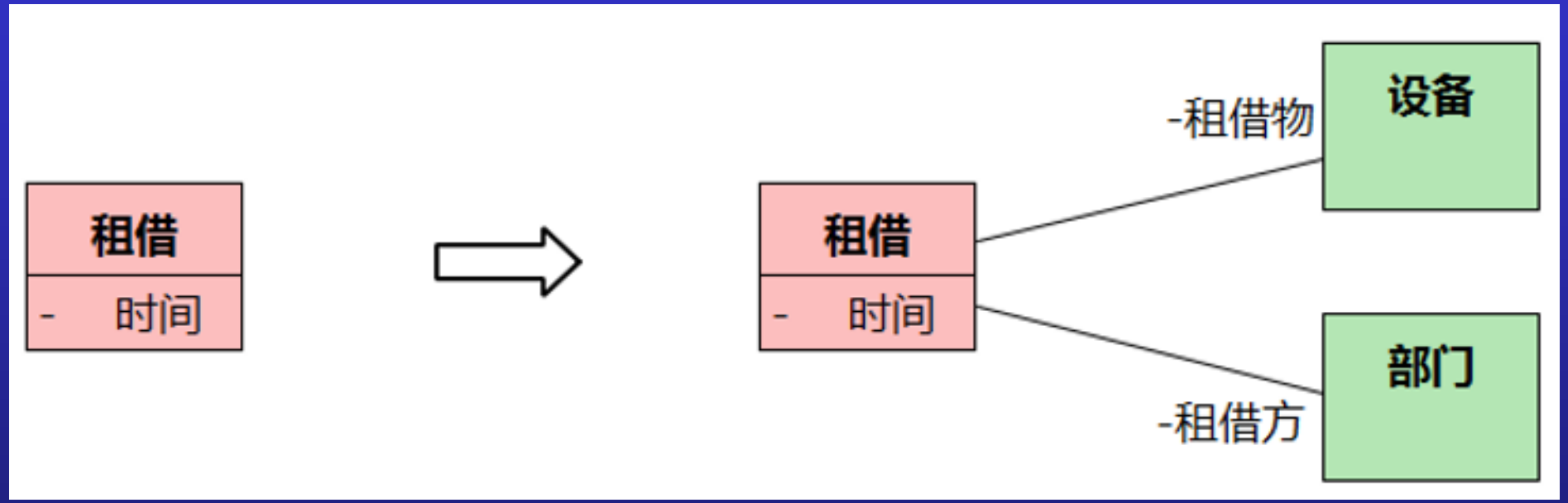
架构型-建模思路



从时刻时段开始

架构型-建模思路

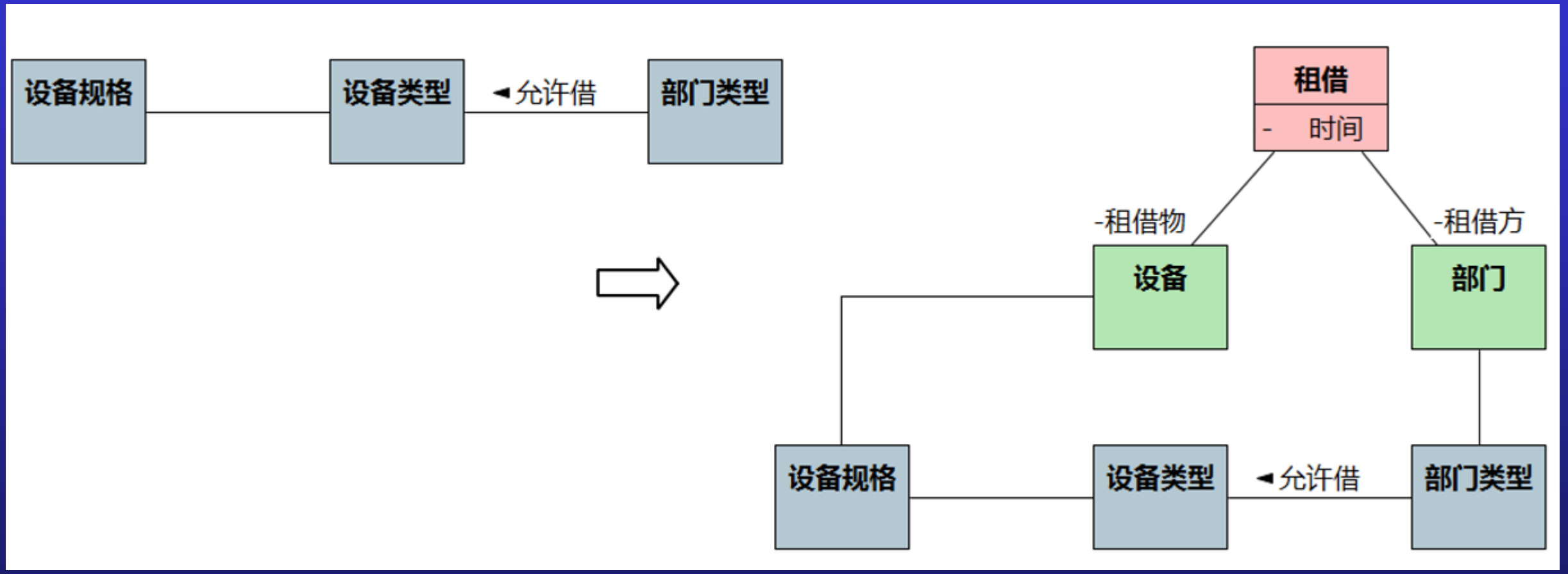
复习GJ-002



从时刻时段开始

架构型-建模思路

复习GJ-002



从描述开始

搅混水

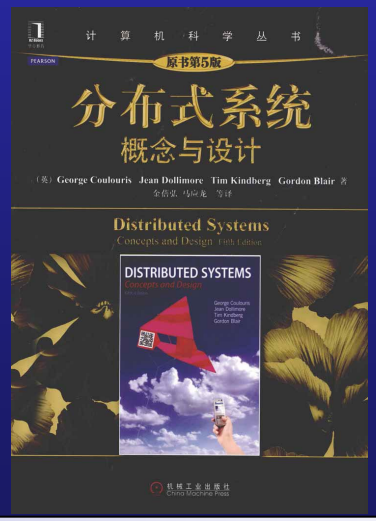
- 借口
  - 性能
  - 分布式
  - 程序员沟通
  - ……
- 回应
  - 你说的这个问题，是这个领域特有的吗

第 18 章

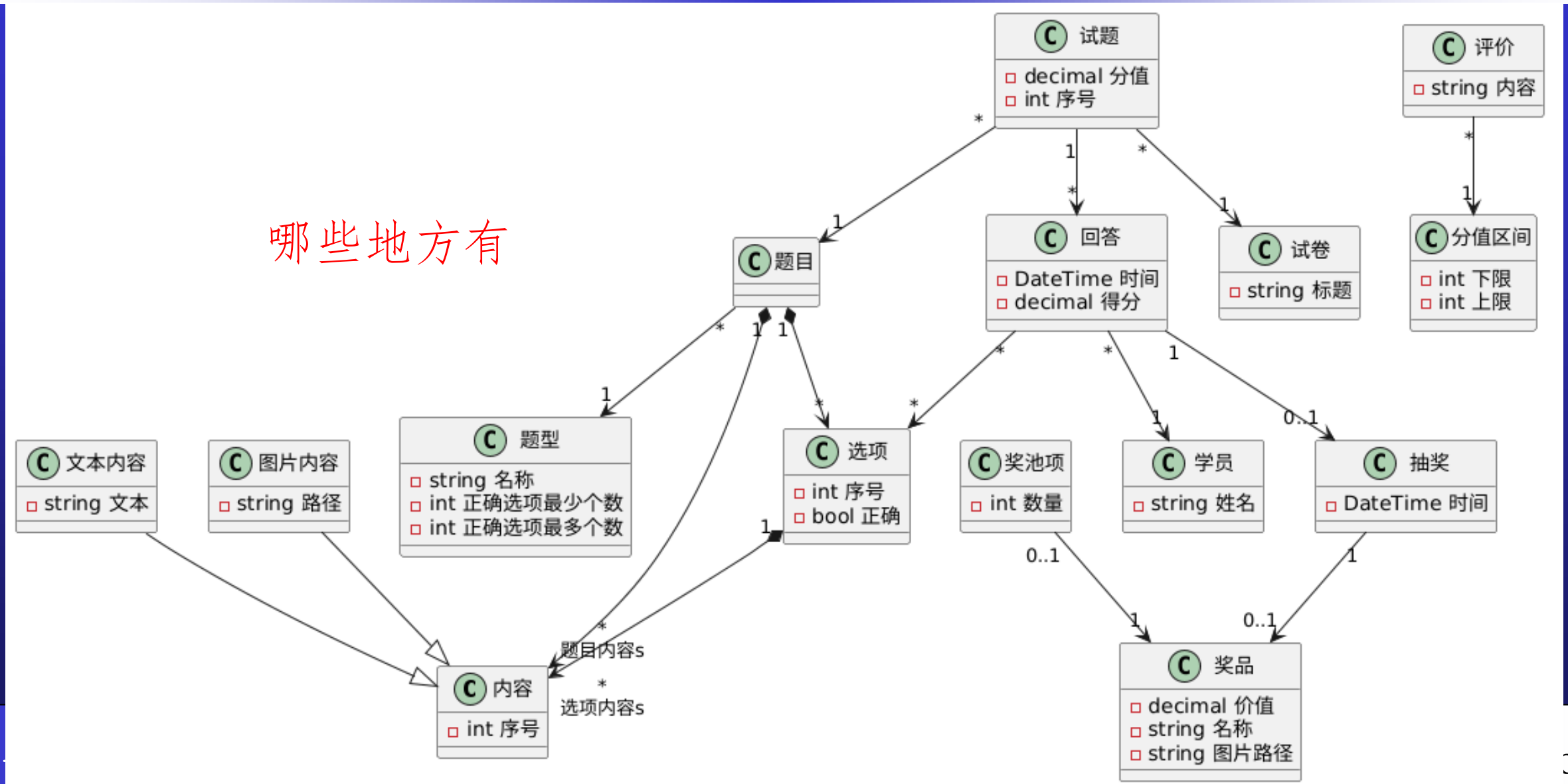
Distributed Systems: Concepts and Design, Fifth Edition

复　　制

在分布式系统中，复制是提供高可用性和容错的关键技术。随着受移动计算的发展和与此相关的断链操作频繁发生，高可用性日益引起人们广泛的兴趣。容错在安全是关键要素的系统和其他重要系统中通常作为一项必备的服务被提供。

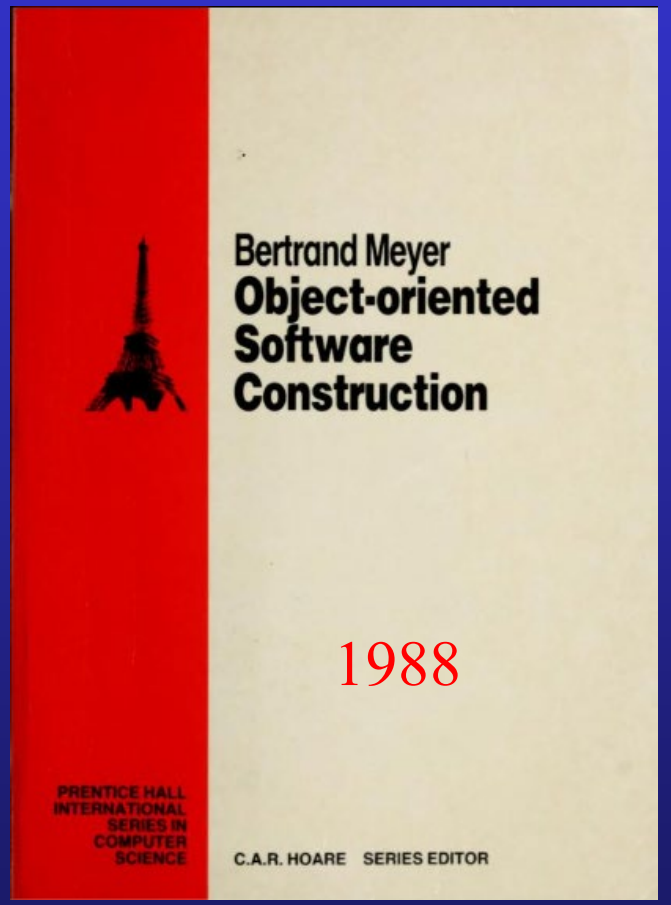


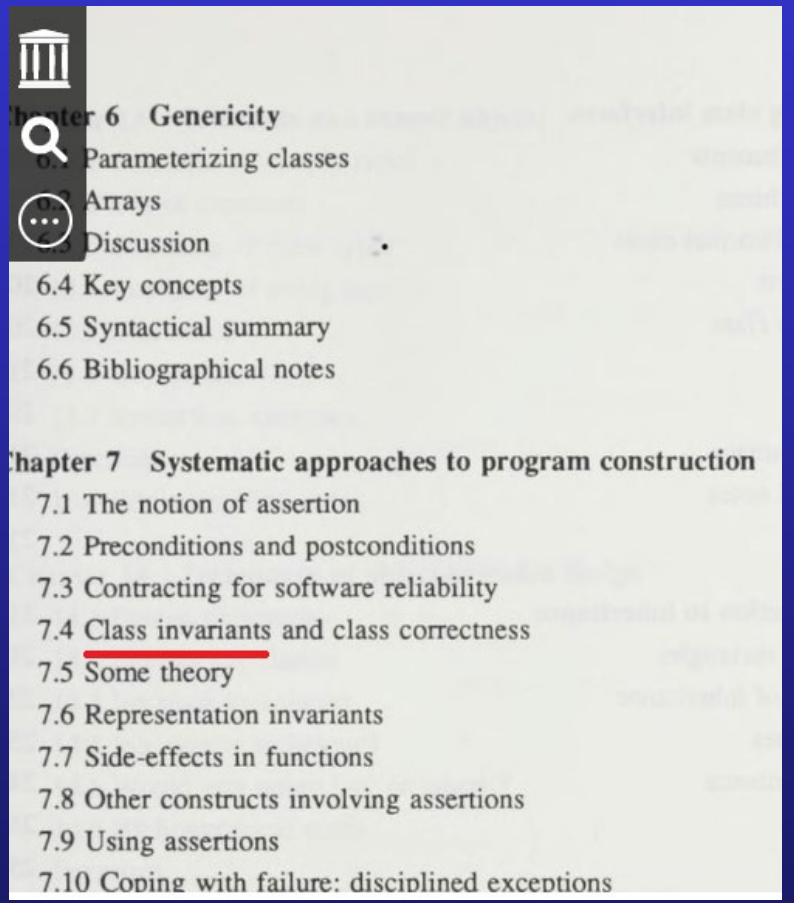
约束


试题
decimal 分值
int 序号
评价
string 内容
哪些地方有
题目
回答
DateTime 时间
decimal 得分
试卷
string 标题
分值区间
int 下限
int 上限
文本内容
string 文本
图片内容
string 路径
题型
string 名称
int 正确选项最少个数
int 正确选项最多个数
选项
int 序号
bool 正确
奖池项
int 数量
学员
string 姓名
抽奖
DateTime 时间
内容
int 序号
题目内容s
选项内容s
奖品
decimal 价值
string 名称
string 图片路径

约束

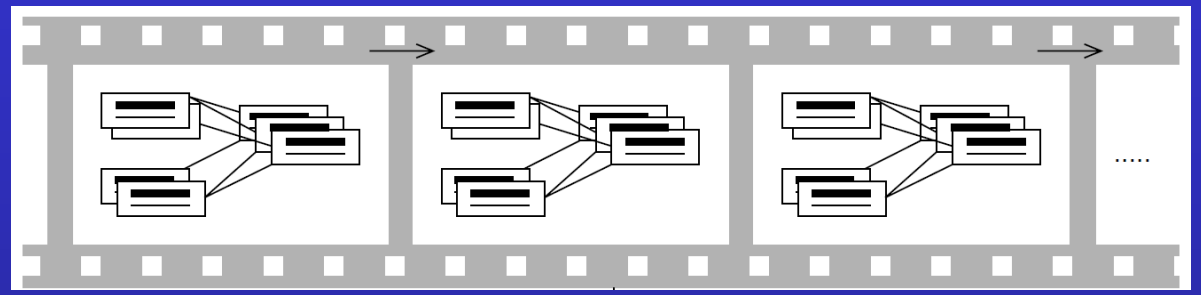
- 关于属性的谓词
  - （类的）不变式
  - （操作的）前置后置条件
- 表达“不得不”的逻辑
  - AI时代最重要的技能
- 状态机图（或活动图）
  - 什么情况下该做什么

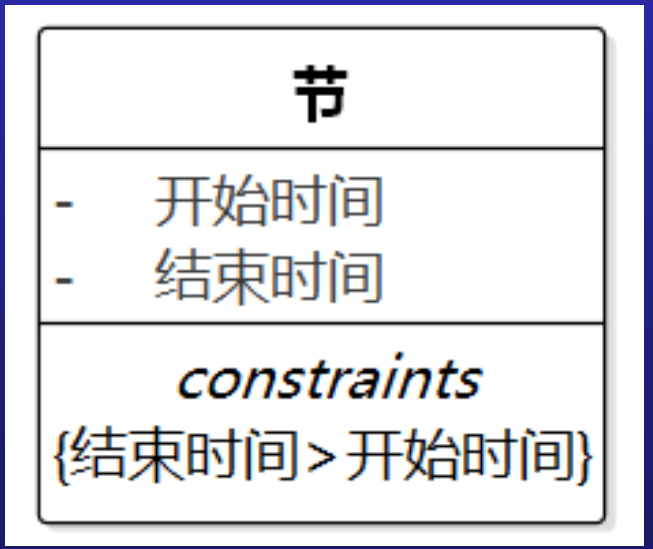




不变式

- 稳定状态下对象的属性值的约束
  - 如果可以随意，复杂性↓↓
- 不得不添加
  - 类图无冗余
  - 类图没有表达且无法推导





不变式

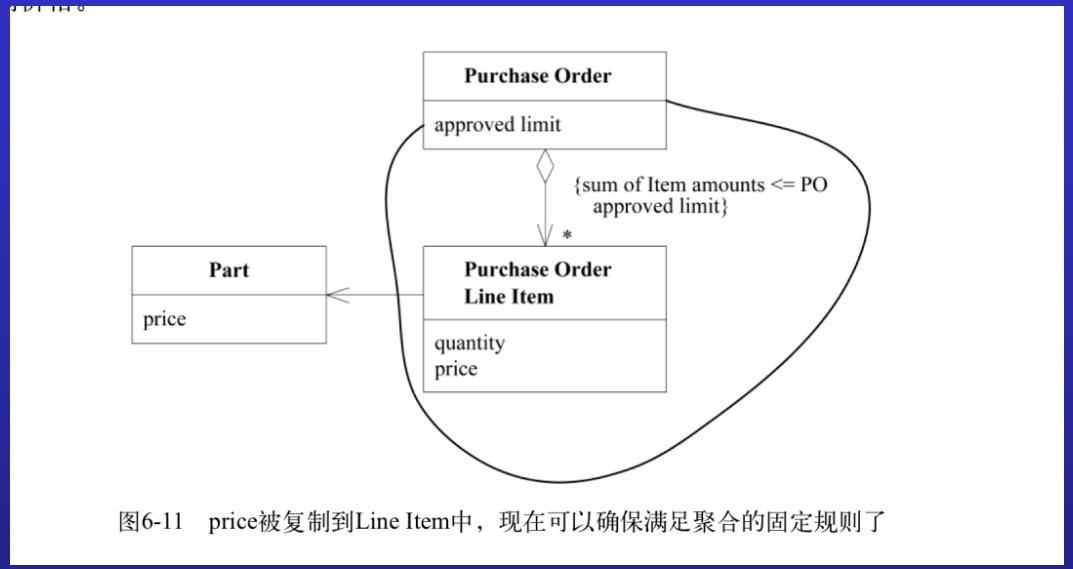


图6-11　price被复制到Line Item中，现在可以确保满足聚合的固定规则了

领域驱动设计

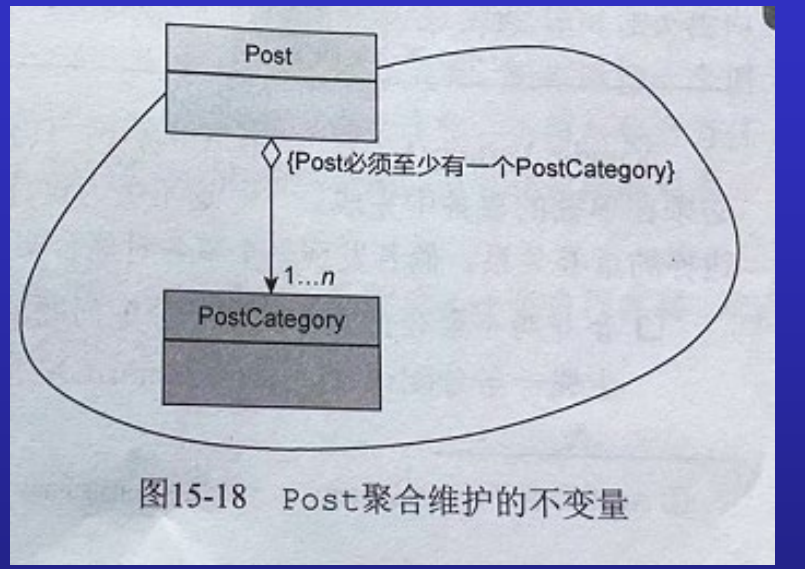
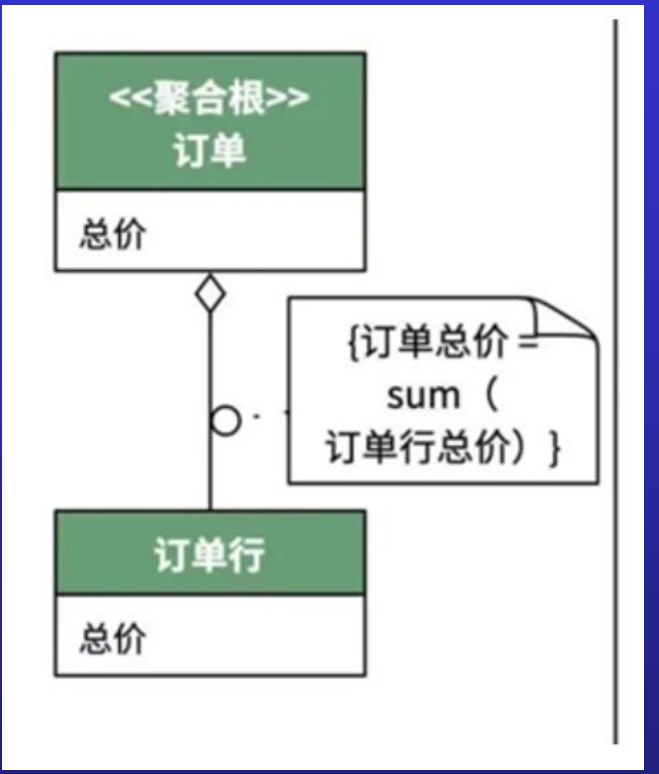


图15-18　Post聚合维护的不变量

国内模仿

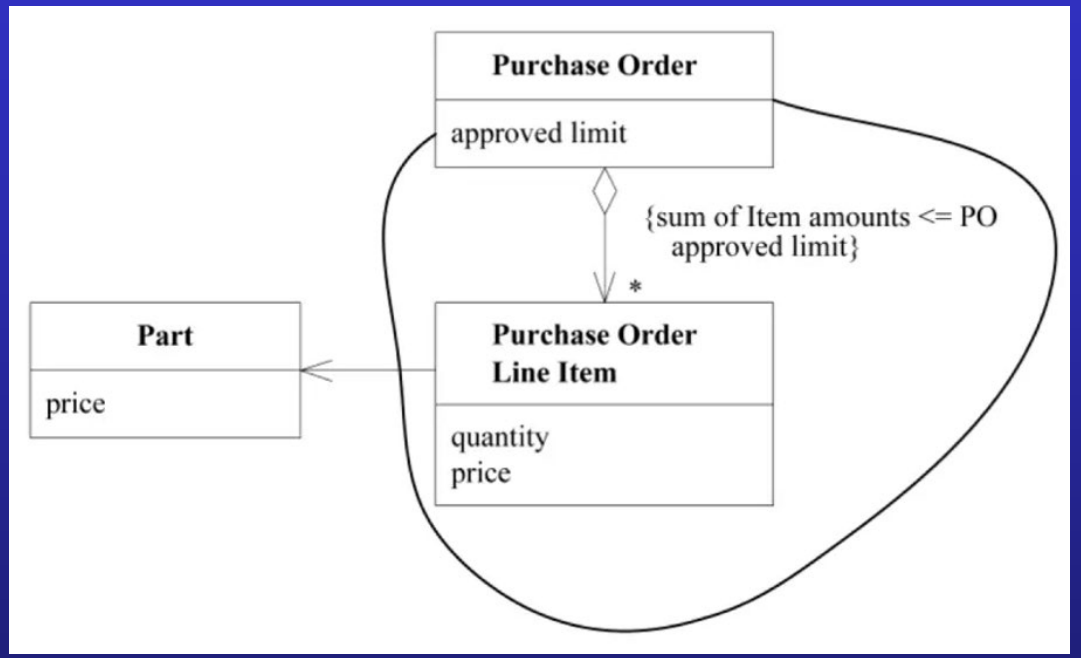


DDD圈子的“不变式”

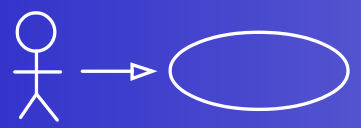
练习

[单选]

以下是《领域驱动设计》中的一个图，图中花括号圈住的内容“sum of Item amounts ******”的最大问题是：



- ☐ A）此内容在图中的其他地方已有表达，属于冗余内容。
- ☐ B）此内容表达为加在两个类之间的关联之上的约束。
- ☐ C）此内容没有采用形式化语言如OCL表达。
- ☐ D）此内容中的amounts和其他地方的内容对不上。



练习

［单选］

右图摘自《领域驱动设计》，被圈住的“不变式”体现了领域驱动设计投资少、见效快、产量高、门槛低、仪式感十足的特点。

请问，选项中的哪一个“不变式”没有体现出这样的特点？

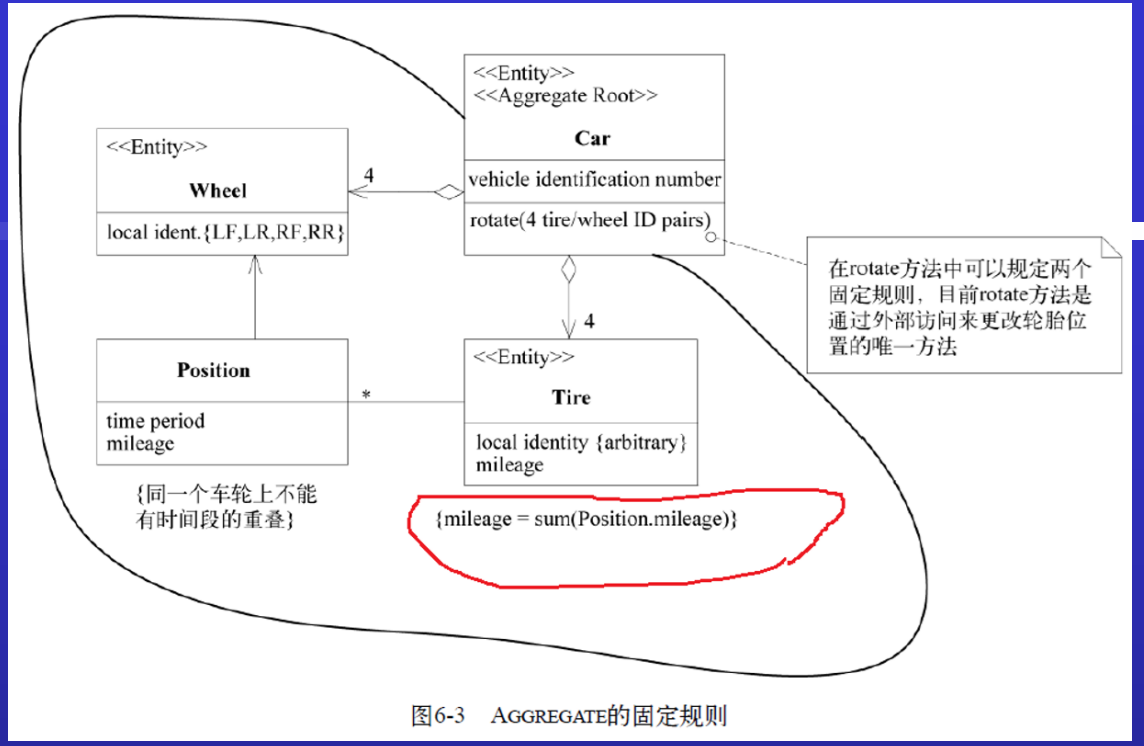
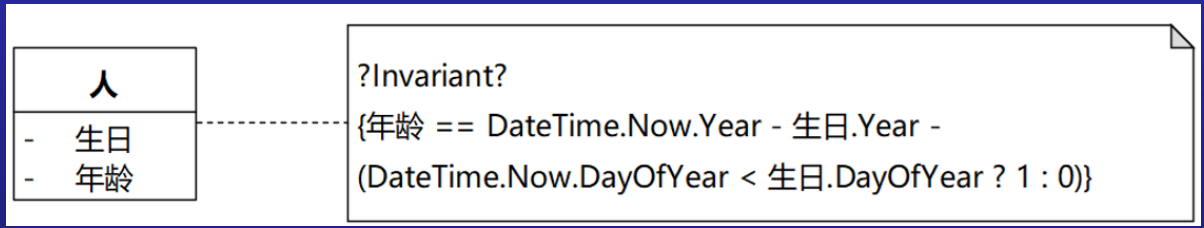


图6-3 AGGREGATE的固定规则

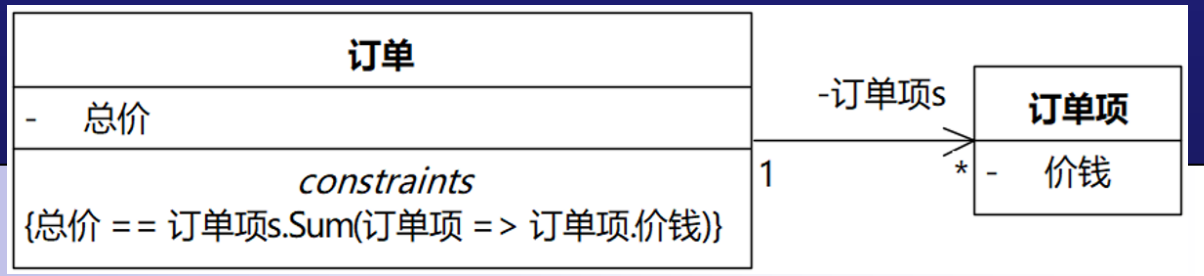
☐ A)

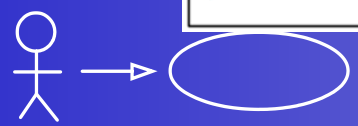


☐ B)

☐ C)

☐ D)

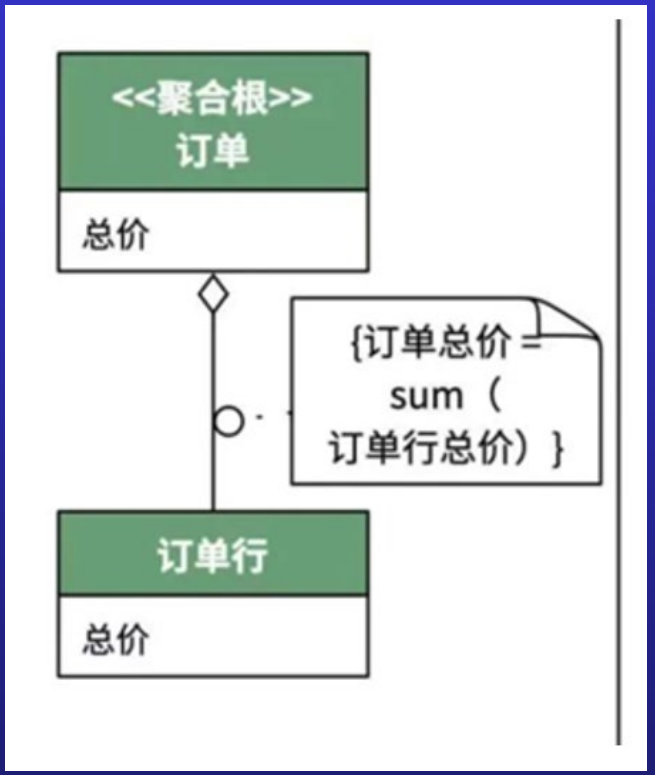




练习

［单选］DDD圈子曾经写文章展示了一个等式的“不变式”（该文用词为“固定规则），受到了批评。

抛开图中所谓“不变式”的位置错误、表示错误等不谈，请问，什么情况下这样内容的“不变式”是合理的？

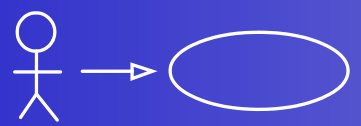


☐ A）拉一群年轻的资深敏捷专家和DDD专家，边喝酒边头脑风暴足足三小时，投票决定：这样内容的“不变式”是合理的。

☐ B）使用领域驱动设计的革命性创造，令在某个上下文中，这样的不变式是合理的。

☐ C）系统有一个从总价出发做假账的功能。

☐ D）系统需要从订单项的属性值来计算订单的属性值。



不变式

- 关系闭包（Closure）
  - X+：可以由X函数确定的属性集

单选题 1分

关系模式R（U,F）中，属性集U={A,B,C,D,E},函数依赖集F=（A→BC,C→D,BD→A,AD→E,BD→E}。则（CE）F+=（）。

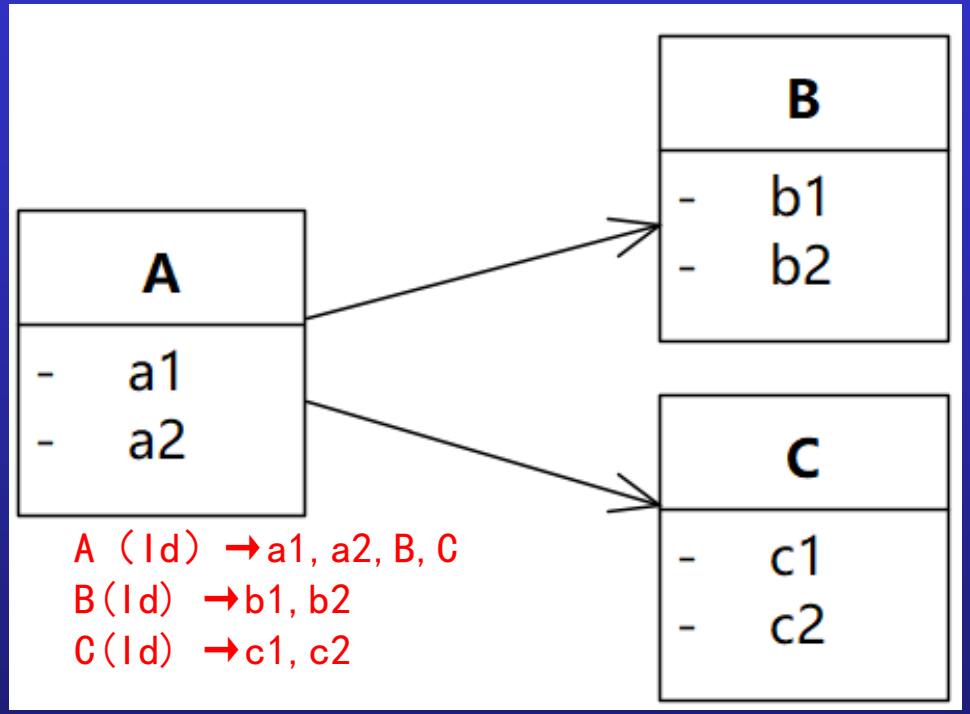
（【中级】数据库系统工程师/2021年上半年（上午）《数据库系统工程师》真题）

A CE

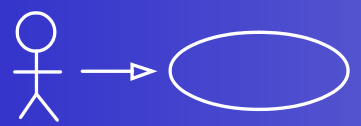
B BCE

C CED

D BCED



直接和间接属性



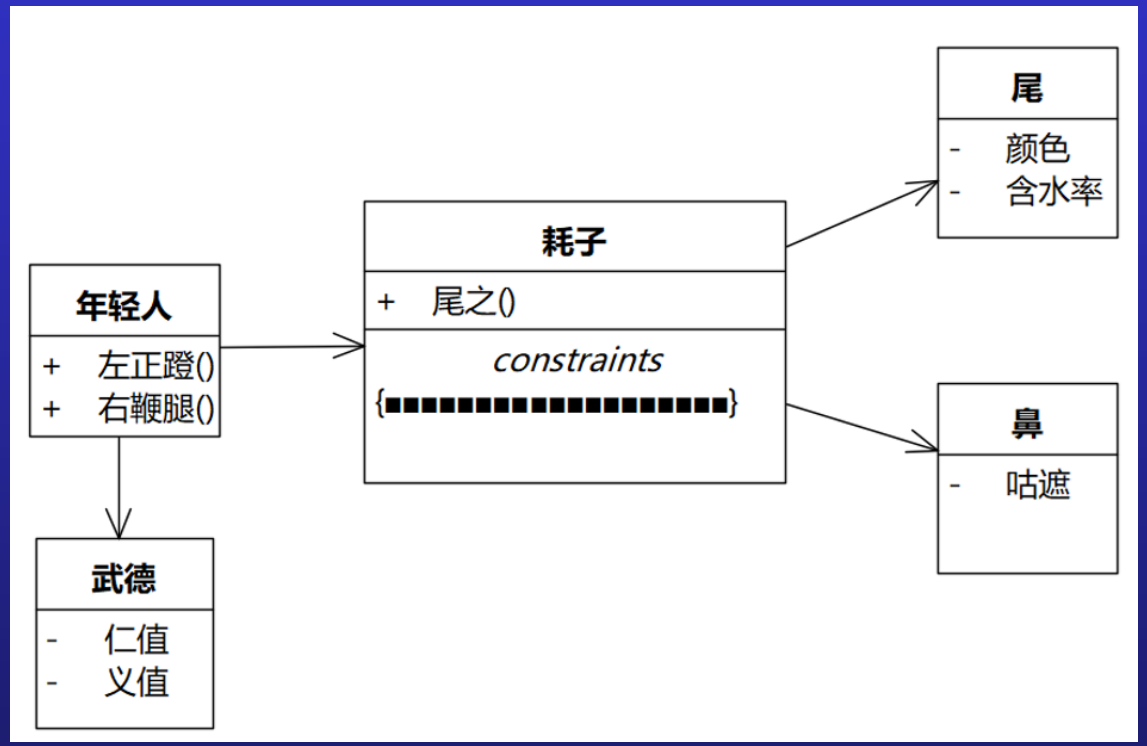
练习

[单选] 浑元形意太极掌门马老师有三大爱好：抽烟、喝酒、烫头。

这天，马老师一边抽烟，一边喝酒，一边烫头，还一遍捧着祖传的武功秘笈细细欣赏。

马老师一不小心，烟头把武功秘笈烧了一小块，像下面这样（咦？怎么是UML类图？是的，当年仓颉造字也顺便造出了UML，马老师的祖传秘笈就是用UML表示的）。

马老师慌了，武功秘笈传了十八代，怎么到我这里毁了呢？于是，赶紧悬赏找人修复。有四位应征的工匠分别给出了长条黑块的部分修复结果，请问哪一位工匠最靠谱？

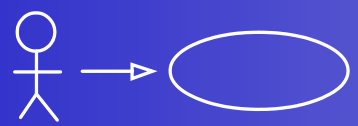


☐ A) ■■■■咕■■■■水■■

☐ B) ■■■■之■■■■子■■

☐ C) ■■■■值■■■■遮■■

☐ D) ■■■■蹬■■■■义■■

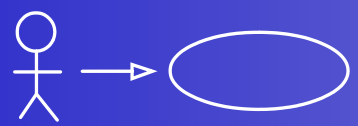


不变式语言

- 建模
  - OCL，对象约束语言
  - 和UML、SysML配合建模
- 实现
  - LINQ，语言集成查询，.NET生态
  - 其他生态对应物
  - 函数式语言
- AI已熟练掌握

- 声明式风格
- 基于集合的操作
- 无副作用
- 面向对象的导航

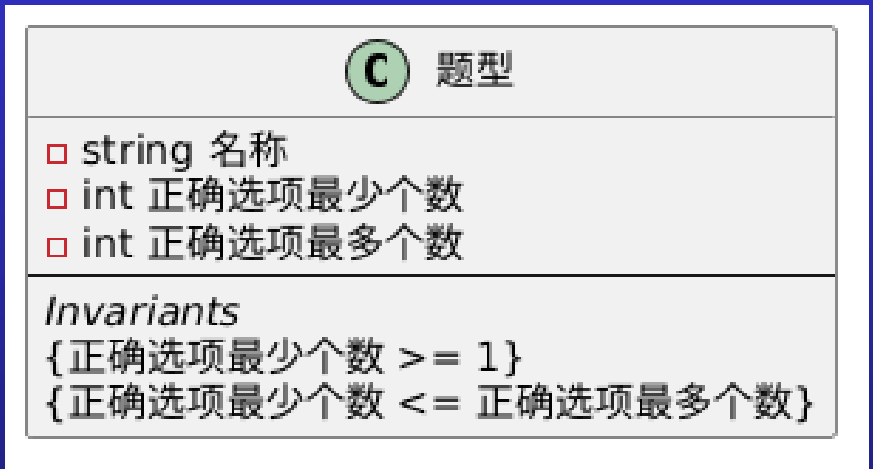
OCL	LINQ	含义
c->select(x | P(x))	c.Where(x => P(x))	筛选
c->collect(x | f(x))	c.Select(x => f(x))	投影/映射
c->exists(x | P(x))	c.Any(x => P(x))	存在/∃
c->forAll(x | P(x))	c.All(x => P(x))	全称/∀
c->includes(x)	c.Contains(x)	包含
……	……	……

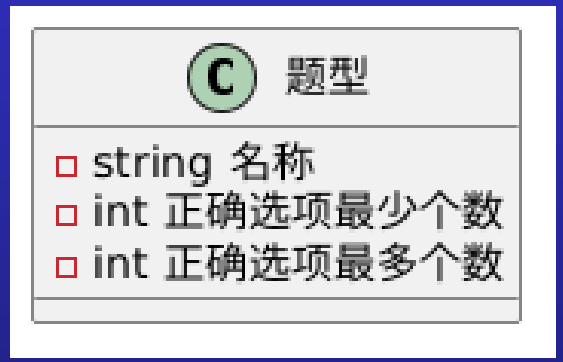


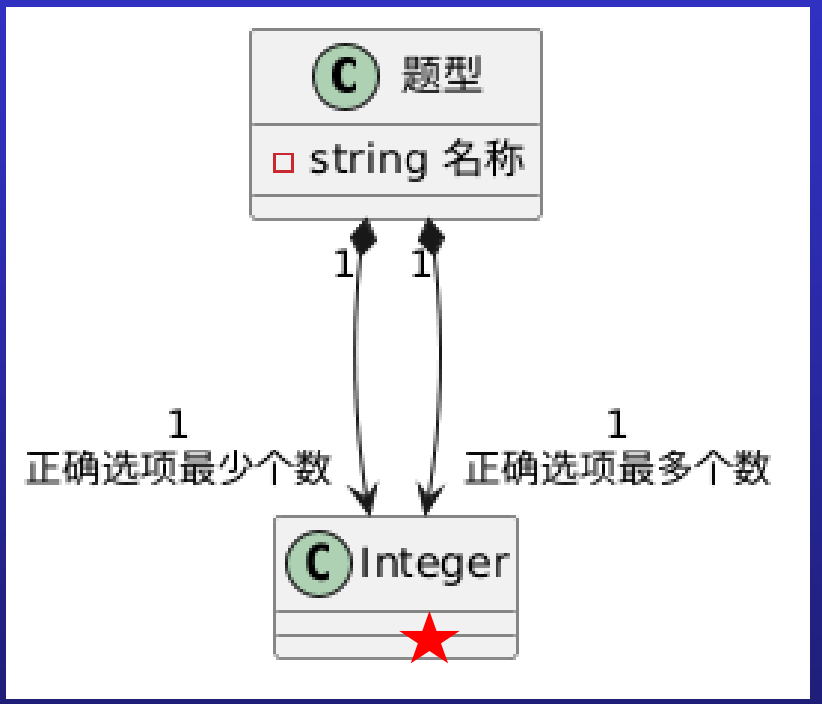
不变式

路径1：→正确选项最少个数→int

路径2：→正确选项最多个数→int



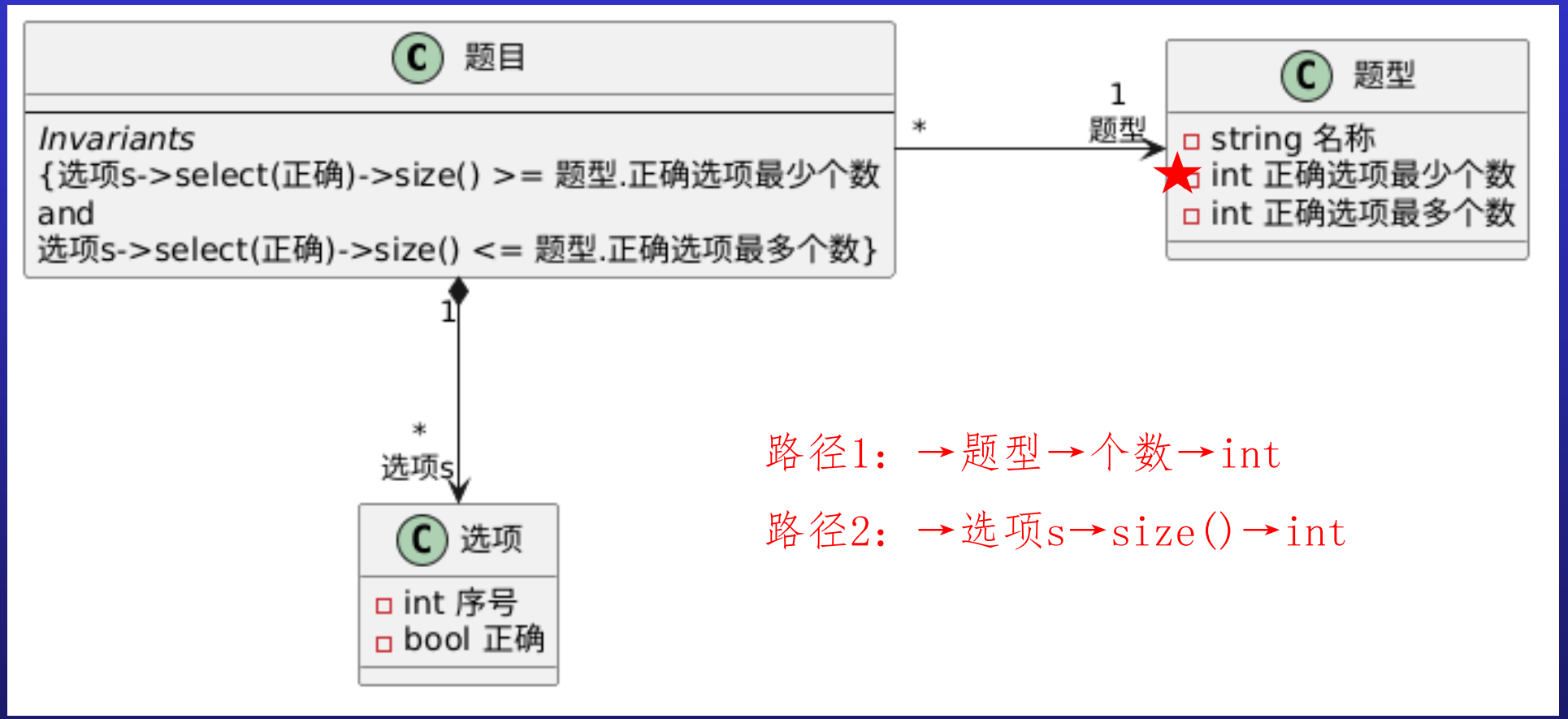




沿不同路径导航到同一个类（类型）

不变式

更长的路径

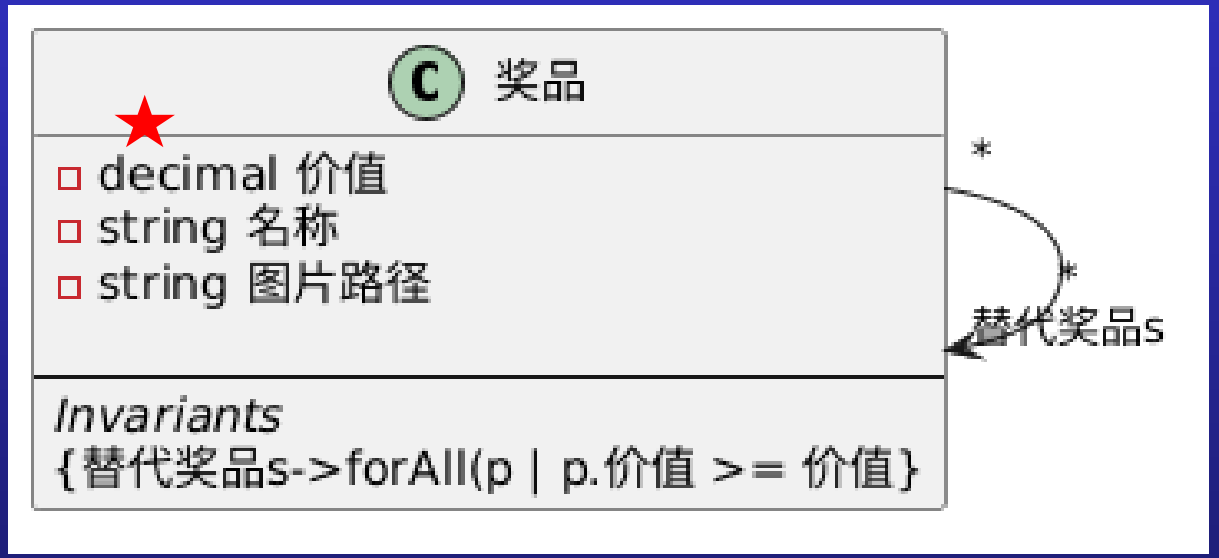

题目
Invariants
{选项s->select(正确)->size() >= 题型.正确选项最少个数
and
选项s->select(正确)->size() <= 题型.正确选项最多个数}
*
1
题型
题型
string 名称
int 正确选项最少个数
int 正确选项最多个数
1
*
选项s
选项
int 序号
bool 正确
路径1：→题型→个数→int
路径2：→选项s→size()→int

不变式

路径1：→价值→decimal

路径2：→替代奖品→价值→decimal

自反关联也行

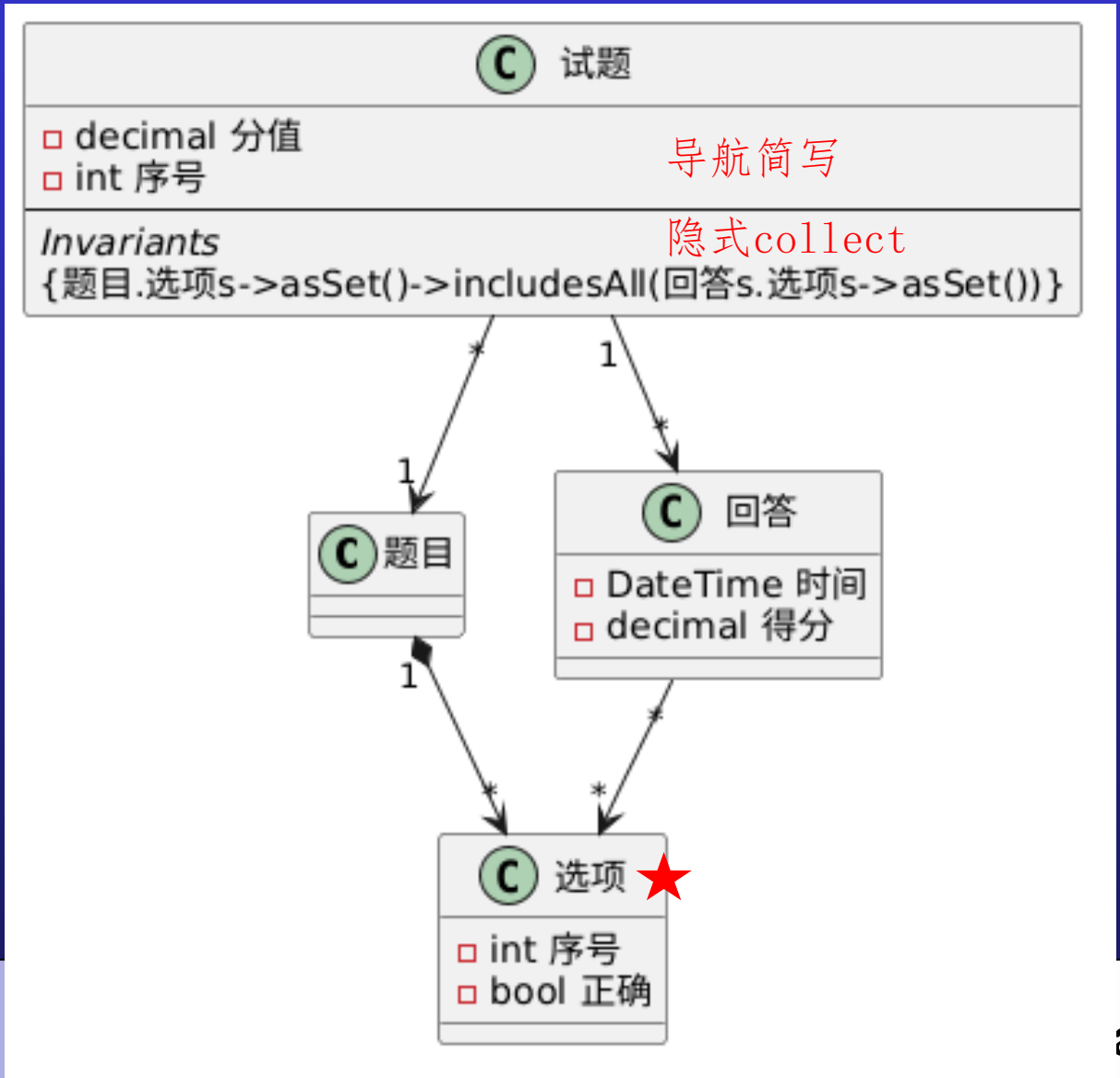


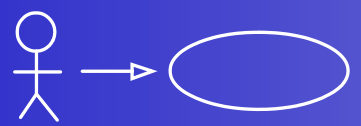
不变式

不是基本类型

路径1：→题目→选项s

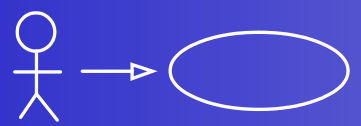
路径2：→回答s→选项s





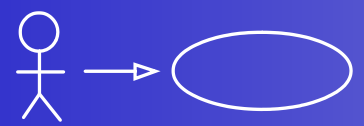
不变式

复习一些逻辑运算知识



不变式

原表达	等价表达	OCL
((P ∨ Q) ∧ R)	((P ∧ R) ∨ (Q ∧ R))	
((P ∧ Q) ∨ R)	((P ∨ R) ∧ (Q ∨ R))	
¬ (P ∨ Q)	(¬P ∧ ¬Q)	
¬ (P ∧ Q)	(¬P ∨ ¬Q)	
(P⇒Q)	(¬P)∨Q	not P or Q
if P then Q else R endif (Q、R是布尔表达式)	(P⇒Q) ∧ (¬ P⇒R) ⇔ ((¬P)∨Q) ∧ (P∨R)	(not P or Q) and (P or R)
P ⊕R	(P∨Q) ∧ ¬ (P∧Q) ⇔ (P∨Q) ∧ (¬P∨¬Q)	(P or Q) and (not P or not Q)
X->exists(Y)	not X->forAll(not Y)	
X->select(Y)->forAll(Z)	X-> forAll(Y ⇒ Z)	



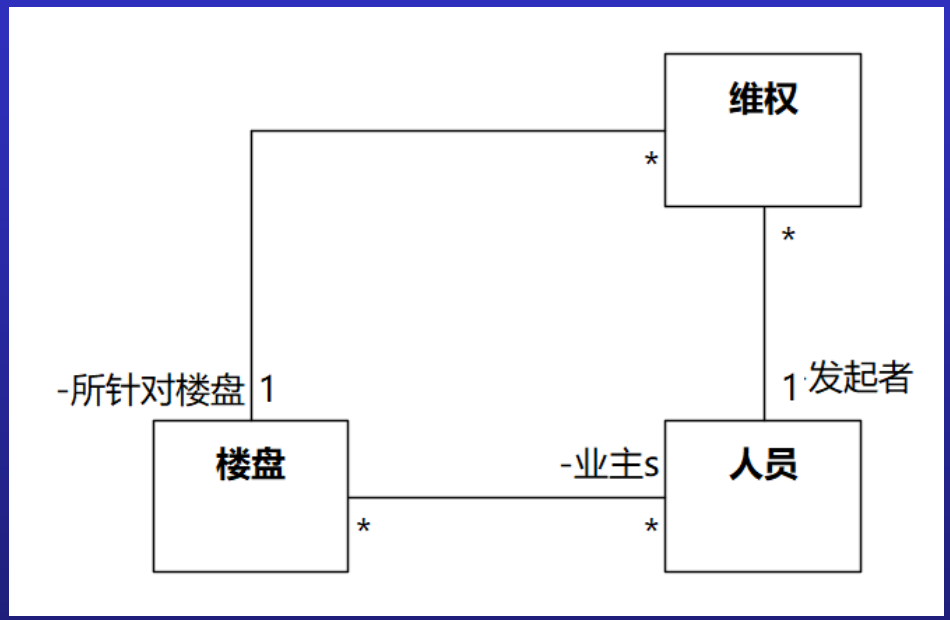
练习

[单选]

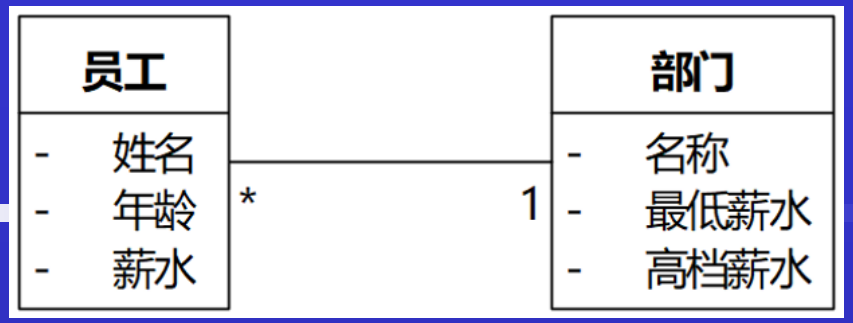
大家好，她是中原御府花园业主。2025年，抖音昵称为“冬奇与雾凇”的业主维权引起关注。如果有右侧类图（严格来说，业主是针对房屋，本图简化）

想要用OCL（对象约束语言）表达“维权的发起者必须是所针对楼盘的业主”，如果针对“维权”加约束，以下选项中最合适的是：

- ☐ A)所针对楼盘.业主s-> includes（人员）
- ☐ B)发起者.业主s->includes(所针对楼盘)
- ☐ C)所针对楼盘.人员->includes(发起者)
- ☐ D)所针对楼盘.业主s->includes(发起者)



练习

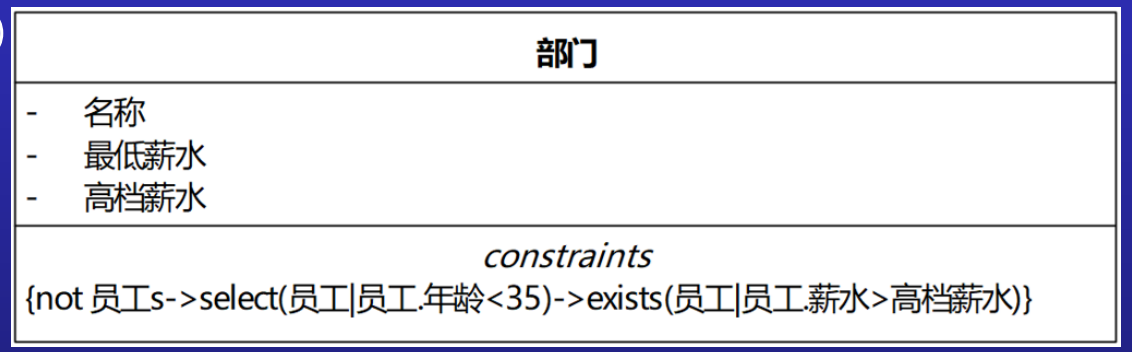


[多选]

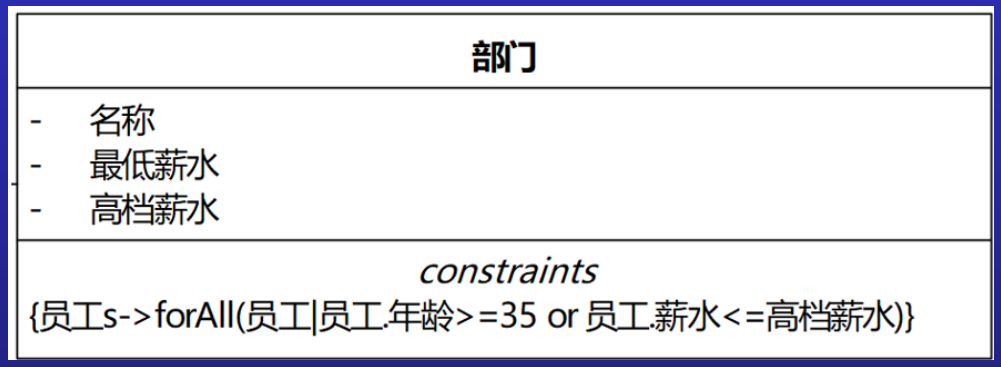
如果针对以上类图有约束：员工年龄未达35岁，薪水不得超过其所属部门的高档薪水。

以下选项中，正确表达此约束的有：

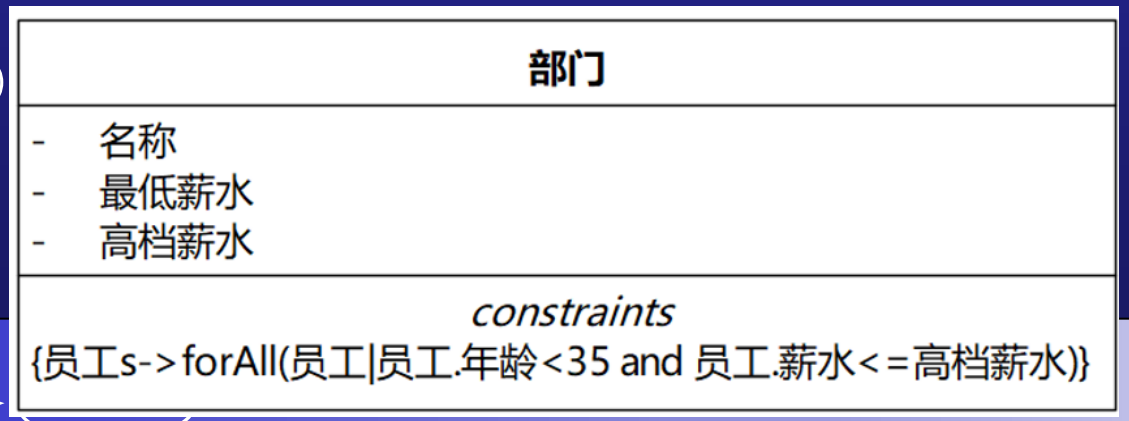
☐ A)



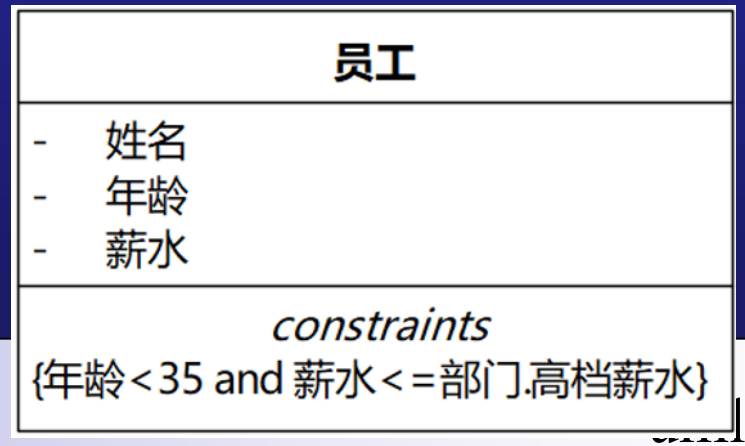
☐ B)

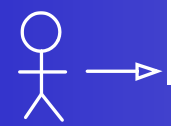


☐ C)



☐ D)





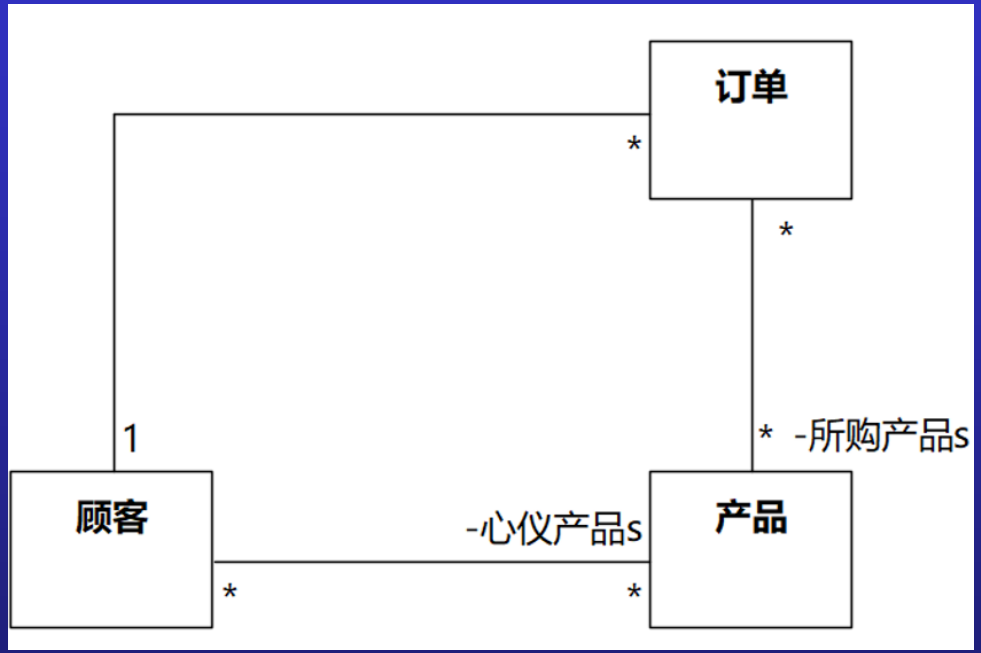
练习

[单选]如果有右侧的类图：

如果系统要贯彻企业的一项奇葩口号——"我们的销售信念是，确保顾客每次消费都能得到他们真正想要的东西。因此，每份订单购买的产品必须和顾客的心仪产品有交集。"

那么，“订单”上的约束用OCL表示，以下最合适的是

- ☐ A) 所购产品s->exists(p | 顾客.心仪产品s->includes(p))
- ☐ B) 所购产品s->includes(顾客.心仪产品s)
- ☐ C) 所购产品s->forAll(p | 顾客.心仪产品s->includes(p))
- ☐ D) 拉一群领域驱动设计专家讨论，得到“订单业务领域实体”、“顾客业务领域实体”、“产品业务领域实体”“订单业务领域服务”、“订单购买业务领域事件”、“订单已购买业务领域状态”、“购买产品业务领域服务”、“产品已购买业务领域状态”、“顾客购买业务领域事件”、“顾客已购买业务领域状态”、“顾客上下文”、“订单上下文”、“产品上下文”以及通用语言"核心业务领域逻辑算法规则001-每份订单购买的产品必须和顾客的心仪产品有交集，以便确保顾客每次消费都能得到他们真正想要的东西。"



练习

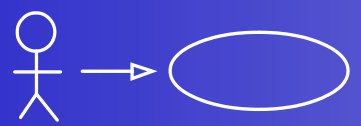
[单选]

说唱歌曲，揽佬SKAI ISYOURGOD的《大展鸿图》中有一句歌词：世上君子不贪杯。

我们把“贪杯”定义为“24小时内摄入酒精量超过20克”。当然，24小时内摄入酒精量需要计算得到，本题跳过这个问题。

以下选项中，给“人”加约束 时，最恰当表达“世上君子不贪杯”的是：

- ☐ A) oclIsKindOf(君子) and 24小时内摄入酒精克数>20
- ☐ B)not oclIsKindOf(君子) or 24小时内摄入酒精克数<=20
- ☐ C) 24小时内摄入酒精克数>20 or oclIsKindOf(君子)
- ☐ D) oclIsKindOf(君子) or 24小时内摄入酒精克数<=20



练习

[多选]

说唱歌曲，揽佬SKAI ISYOURGOD的《大展鸿图》中有一句歌词：世上君子不贪杯。

我们把“贪杯”定义为“24小时内摄入酒精量超过20克”。当然，24小时内摄入酒精量需要计算得到，本题跳过这个问题。

以下选项中，给“人”用OCL加约束 时，最恰当表达“要么君子，要么贪杯”的有：

- ☐ A) (24小时内摄入酒精克数>20 or oclIsKindOf(君子)) and (not oclIsKindOf(君子) or 24小时内摄入酒精克数<=20)
- ☐ B)24小时内摄入酒精克数>20 or oclIsKindOf(君子)
- ☐ C)24小时内摄入酒精克数>20 xor oclIsKindOf(君子)
- ☐ D)not oclIsKindOf(君子) or 24小时内摄入酒精克数<=20

练习

1. 红色高亮部分，《离婚法》实际不存在，为编造。
2. 黄色高亮部分，我没有找到相应数据，文中也没有对应的引用。

在中国，根据中国社会科学院的数据，近 30%的家庭成员遭受过不同程度的家庭暴力，其中 90%的施暴者为男性。从时间来看，根据中国妇女社会地位调查：1990 年的调查数据显示我国妇女经历家庭暴力占比为 30%，2000 年的调查数据显示我国妇女经历家庭暴力占比为 22.5%，而在 2001 年随着《离婚法》的出台与宣传，2010 年的调查数据显示我国妇女遭受家庭暴力的占比降至 8.8%，2015 年时我国又通过了《反家庭暴力法》，随着该法在 2016 年的开始实行，2020 年

[单选] 杨*媛在论文中探讨了家庭暴力。 假设我们对任一项家庭暴力案件有以下要求：

（1）一个人不能同时是案件的施暴者和受害者。（2）施暴者中，一定有人和受害者中的人是家人关系

那么，如果给以下UML类图的“家庭暴力案件”类用OCL加约束，最合适的是：

补注：

（1）题目假设存在群殴以及外人介入的可能，例如离异带两娃的父（或母）及其女（或男）友一起对两个娃施暴。

（2）为简化问题，把关系简化为“家人”，并忽略关系存续时间以及关系的方向（上下级，夫妻，父子）等。

☐ A)施暴者s->intersection(受害者s)->isEmpty() and 施暴者s->exists(p | p.关系s->exists(r | r.人员关系类型 = 人员关系类型::家人)) and 受害者s->exists(v | v.关系s->exists(r | r.人员关系类型 = 人员关系类型::家人))

☐ B)施暴者s->intersection(受害者s)->isEmpty() and 施暴者s->forAll(p1 | 受害者s->forAll(p2 | p1.关系s->exists(r | r.人员关系类型 = 人员关系类型::家人 and r.人员s->includes(p2))))

☐ C)施暴者s->intersection(受害者s)->isEmpty() and 施暴者s->exists(p1 | 受害者s->exists(p2 | p1.关系s = p2.关系s))

☐ D)施暴者s->intersection(受害者s)->isEmpty() and 施暴者s->exists(p1 | 受害者s->exists(p2 | p1.关系s->exists(r | r.人员关系类型 = 人员关系类型::家人 and r.人员s->includes(p2))))

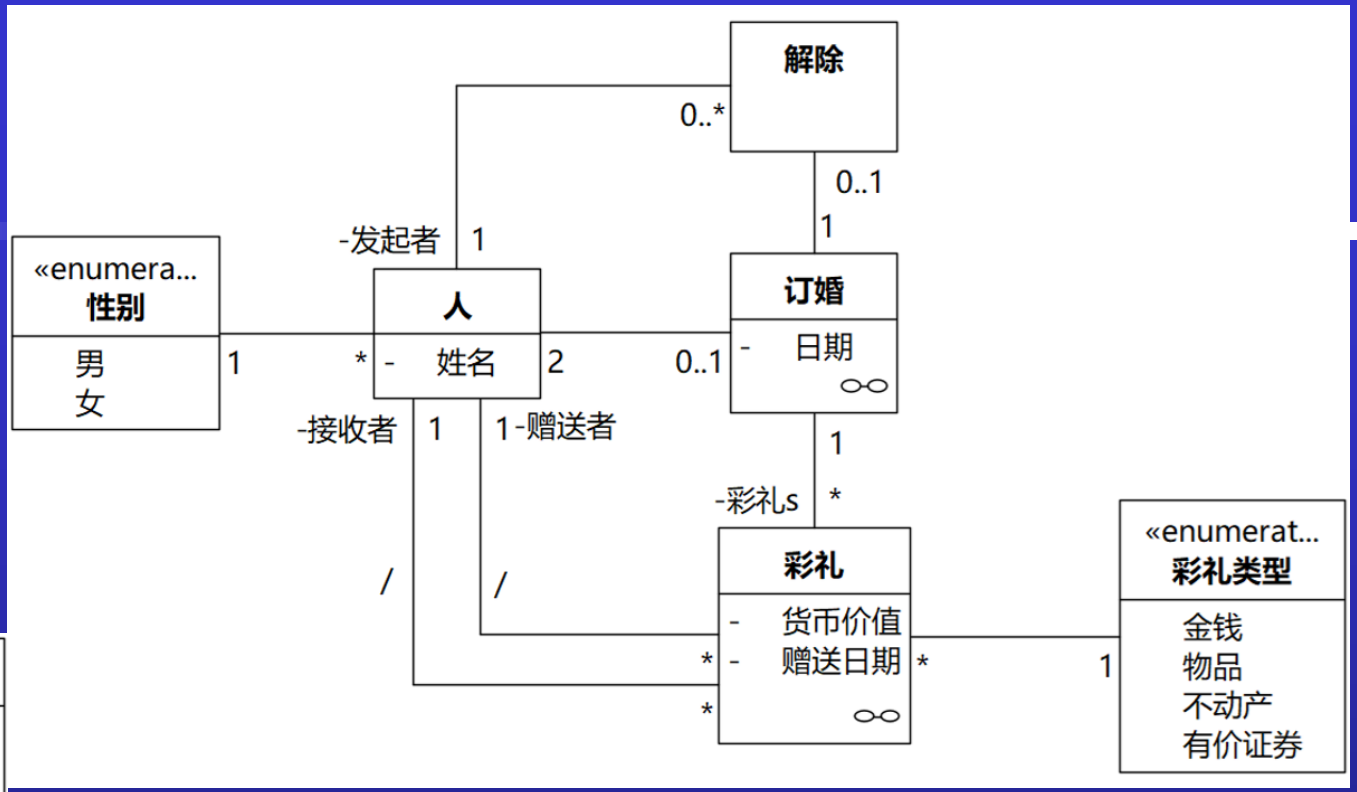
练习

［单选］

右侧是订婚、彩礼相关的类图。如果要表达以下规则：

订婚后如果因女方反悔主张解除，则所有彩礼可退回。

以下选项中的约束合适的是



☐ A)

解除

constraints

{发起者.性别 = 性别::女 implies 订婚.彩礼s->exists(b | b.状态 = 应退回)}

☐ B)

人

- 姓名

constraints

{性别 = 性别::女 and 发起者->notEmpty() implies 发起者.订婚.彩礼s->forAll(b | b.状态 = 应退回)}

☐ C)

订婚

- 日期

constraints

{解除->IsEmpty() or 解除.发起者.性别 <> 性别::女 or 彩礼s->forAll(b | b.状态 = 应退回)}

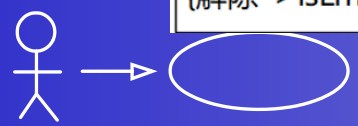
☐ D)

彩礼

- 货币价值
- 赠送日期

constraints

{订婚.解除->IsEmpty() or 状态 = 应退回}

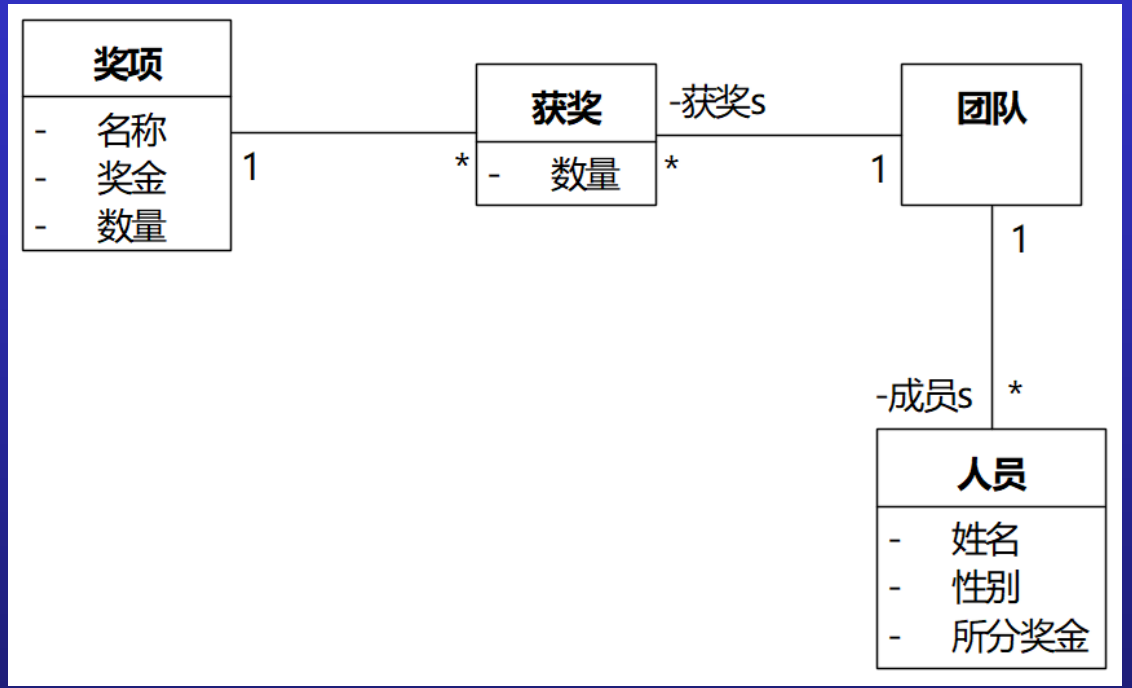


练习

[单选]最近，上海交大樊同学奖金事件引起关注。

用UML类图表示樊同学事件相关概念如下，假设同一团队可以同时获得多个奖项，包括同一等级，例如，F同学和K同学组成的团队可以获得两个二等奖，总奖金2*5000=10000元。

如果给“团队”加一个约束：团队所得奖金总和等于团队人员所分奖金的总和，用OCL如何表达？

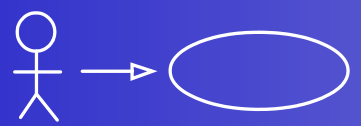


☐ A) 获奖s->collect(x | x.数量 * x.奖项.奖金)->sum() = 成员s.所分奖金->sum()

☐ B) 获奖s.奖项.奖金->sum() = 成员s.所分奖金->sum()

☐ C) 获奖s.数量->sum() * self.获奖s.奖项.奖金->sum() = 成员s.所分奖金->sum()

☐ D) 获奖s->collect(x | x.数量 * x.奖项.奖金)->sum() = 成员s.所分奖金->asSet()->sum()



练习

[单选]

最近，“中国人不需要蛋白质”的视频引起关注。

如果有人要做一个系统贯彻这个理念，系统的类图如下：

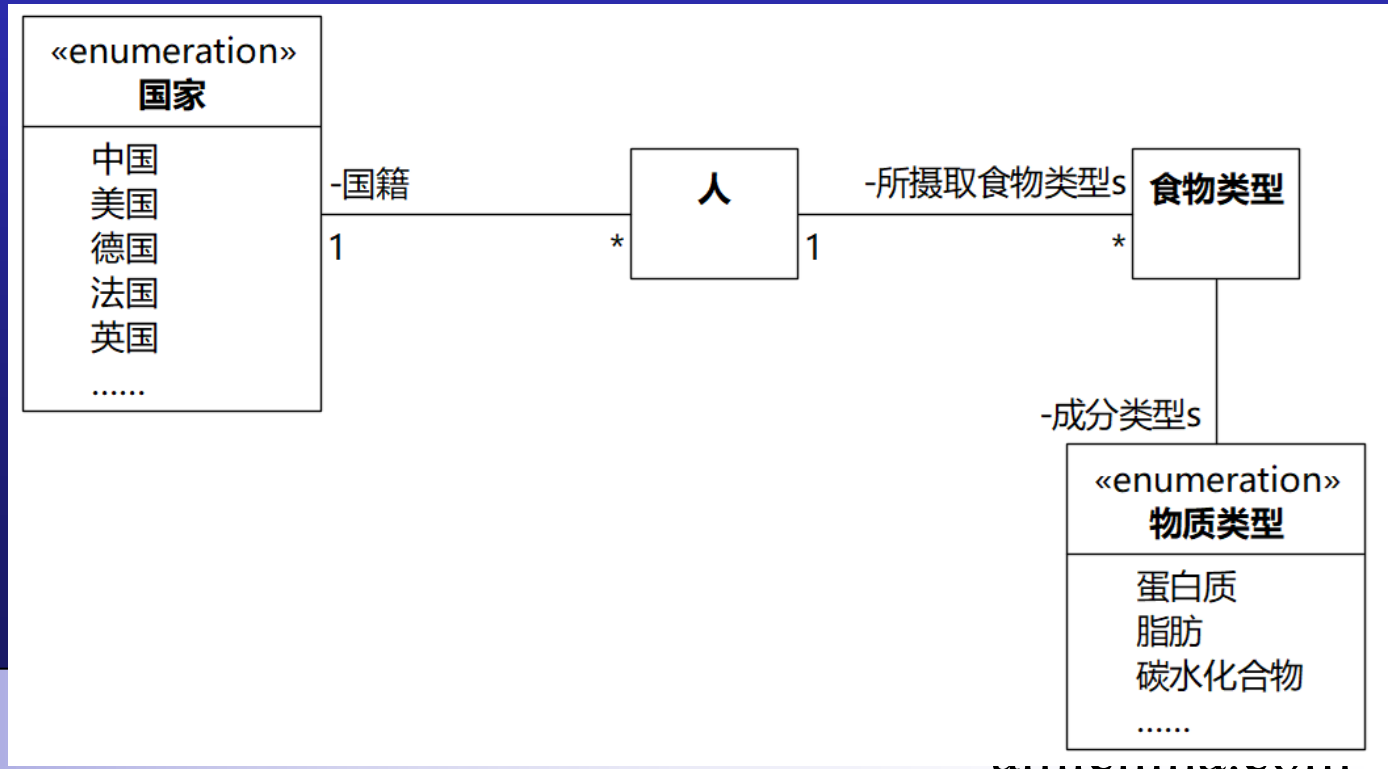
如果给“人”加一个约束：“中国人不需要蛋白质”，以下合适的是：

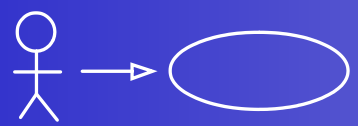
☐ A) 国籍 = 国家::中国 or所摄取食物类型s.成分类型s->excludes(物质类型::蛋白质)

☐ B) 国籍 = 国家::中国 and所摄取食物类型s.成分类型s->excludes(物质类型::蛋白质)

☐ C) 所摄取食物类型s.成分类型s->includes(物质类型::蛋白质) or国籍 = 国家::中国

☐ D) 国籍 <> 国家::中国 or所摄取食物类型s.成分类型s->excludes(物质类型::蛋白质)





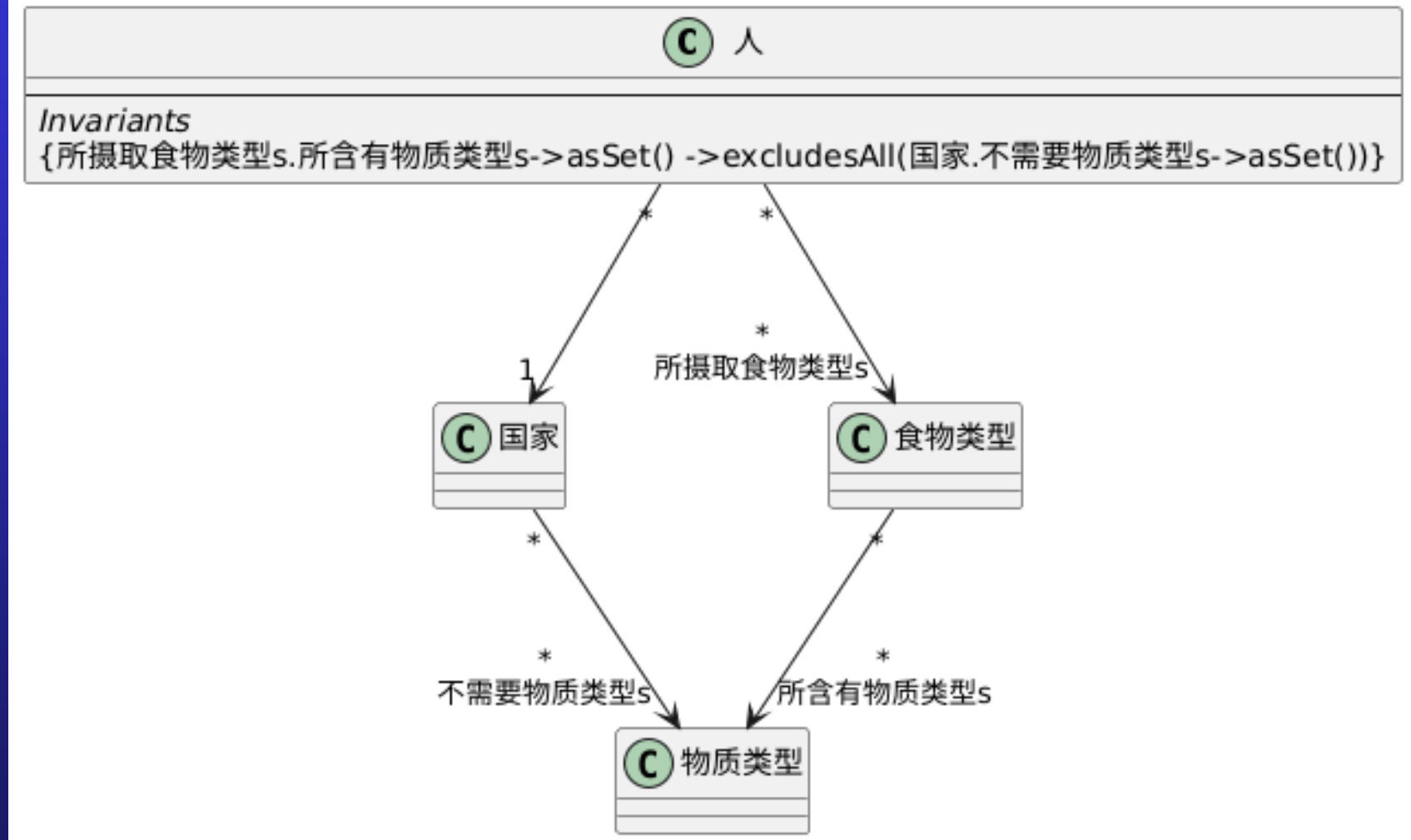
不变式

路径1：→食物类型→物质类型

路径2：→国家→物质类型

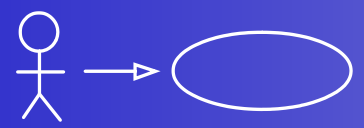
中国人不需要蛋白质

某个国家不需要某个物质类型

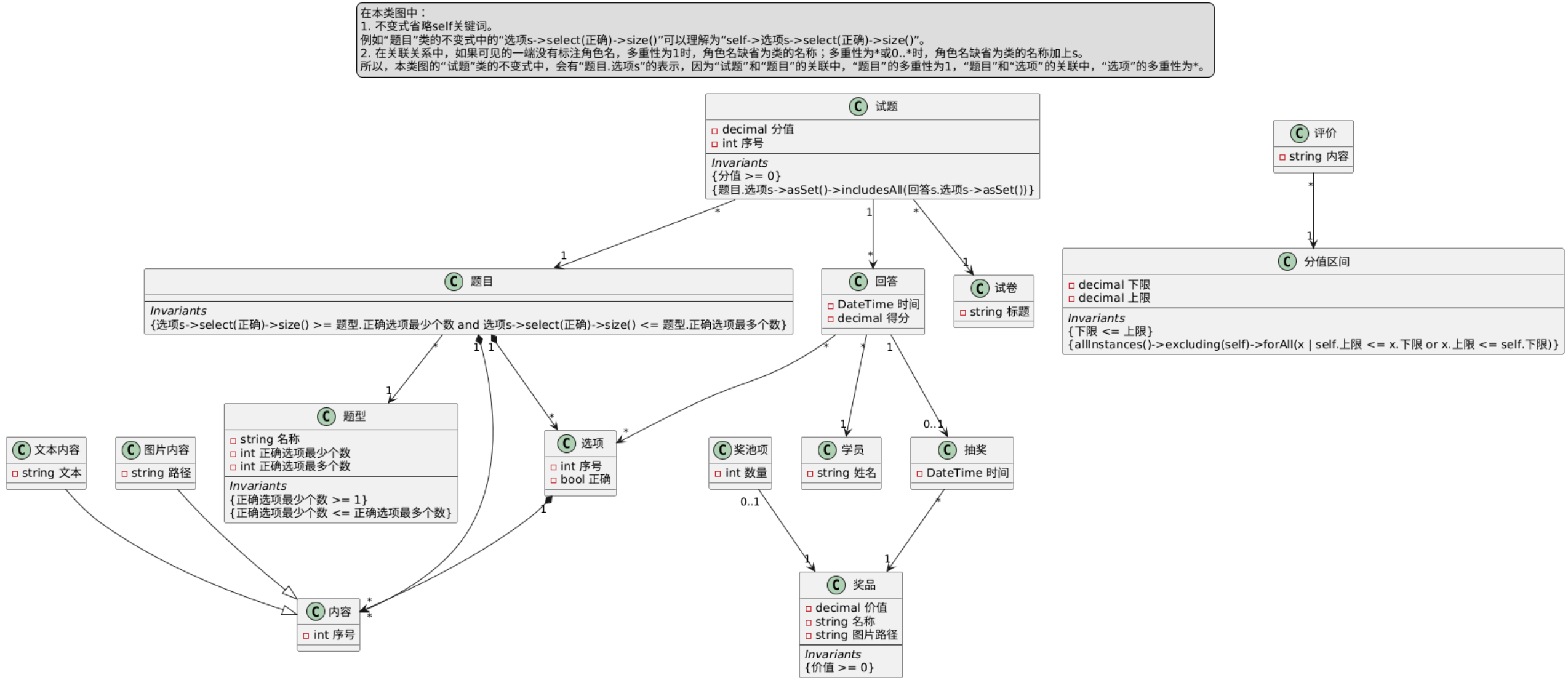


不变式实现

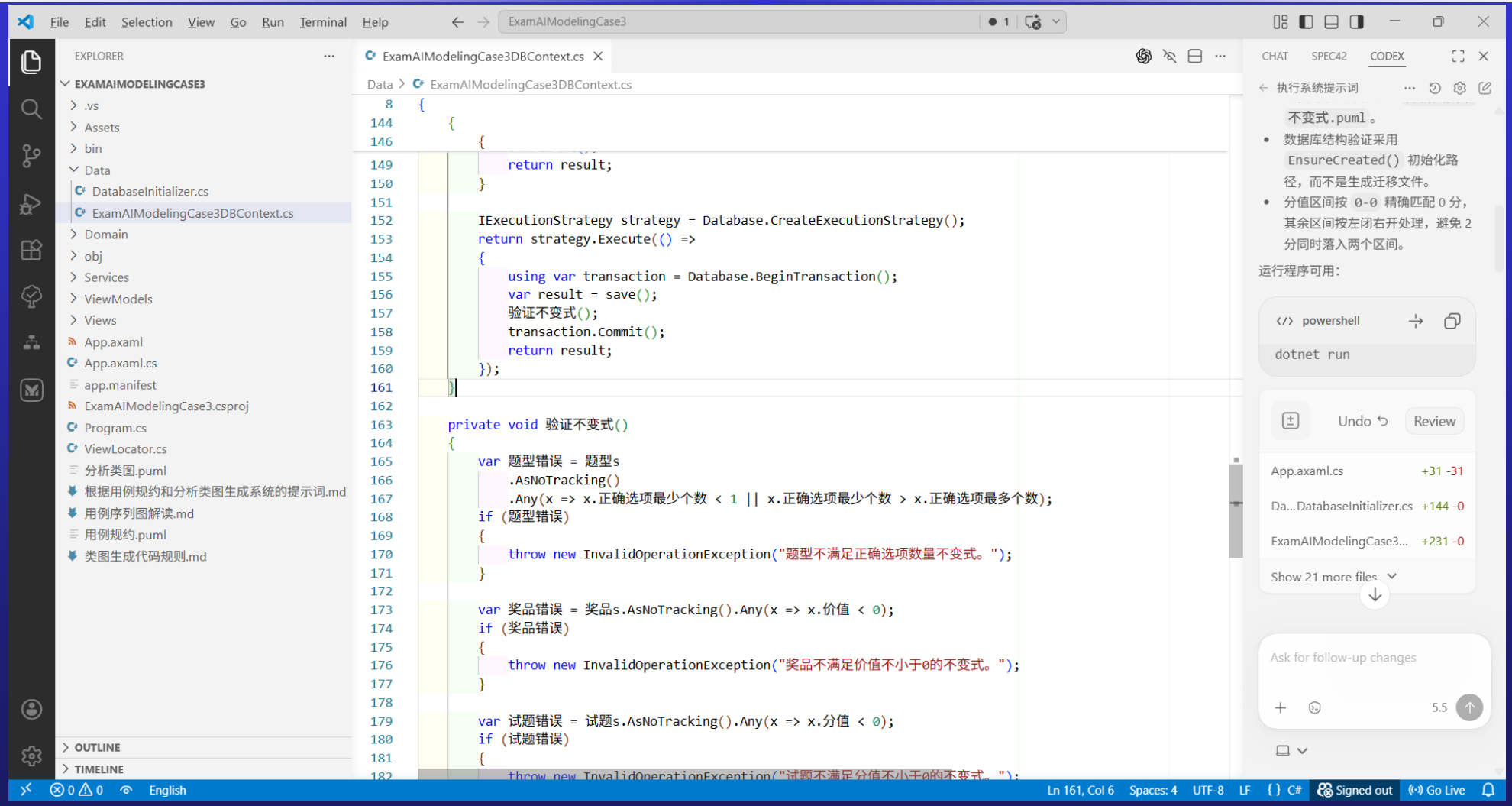
- setter和构造器中检查
- 类的ValidateInvariants 方法
- 独立Validator 类
- EF的SaveChanges验证
- 数据库的CHECK/FK/Trigger
- Contract实现框架
  - Spec# Metalama
- 内在支持的编程语言
  - Eiffel D



类图进一步修正

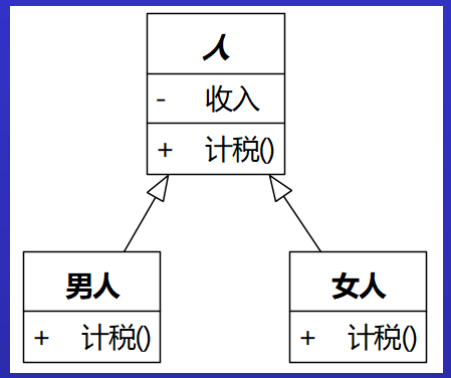

在本类图中：
1. 不变式省略self关键词。
例如“题目”类的不变式中的“选项s->select(正确)->size()”可以理解为“self->选项s->select(正确)->size()”。
2. 在关联关系中，如果可见的一端没有标注角色名，多重性为1时，角色名缺省为类的名称；多重性为*或0..*时，角色名缺省为类的名称加上s。
所以，本类图的“试题”类的不变式中，会有“题目.选项s”的表示，因为“试题”和“题目”的关联中，“题目”的多重性为1，“题目”和“选项”的关联中，“选项”的多重性为*。
试题
decimal 分值
int 序号
Invariants
{分值 >= 0}
{题目.选项s->asSet()->includesAll(回答s.选项s->asSet())}
评价
string 内容
题目
Invariants
{选项s->select(正确)->size() >= 题型.正确选项最少个数 and 选项s->select(正确)->size() <= 题型.正确选项最多个数}
回答
DateTime 时间
decimal 得分
试卷
string 标题
分值区间
decimal 下限
decimal 上限
Invariants
{下限 <= 上限}
{allInstances()->excluding(self)->forAll(x | self.上限 <= x.下限 or x.上限 <= self.下限)}
题型
string 名称
int 正确选项最少个数
int 正确选项最多个数
Invariants
{正确选项最少个数 >= 1}
{正确选项最少个数 <= 正确选项最多个数}
选项
int 序号
bool 正确
文本内容
string 文本
图片内容
string 路径
奖池项
int 数量
学员
string 姓名
抽奖
DateTime 时间
内容
int 序号
奖品
decimal 价值
string 名称
string 图片路径
Invariants
{价值 >= 0}

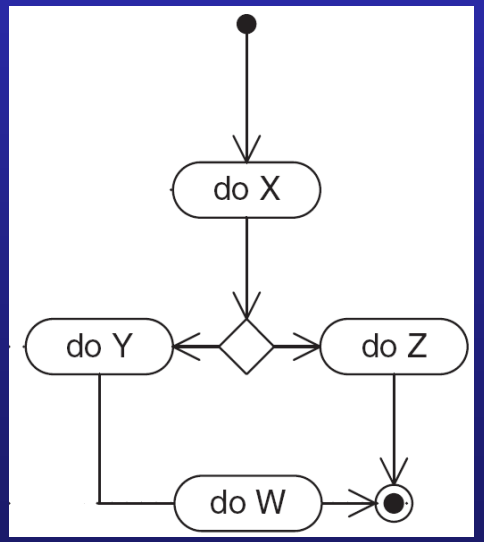
不变式实现

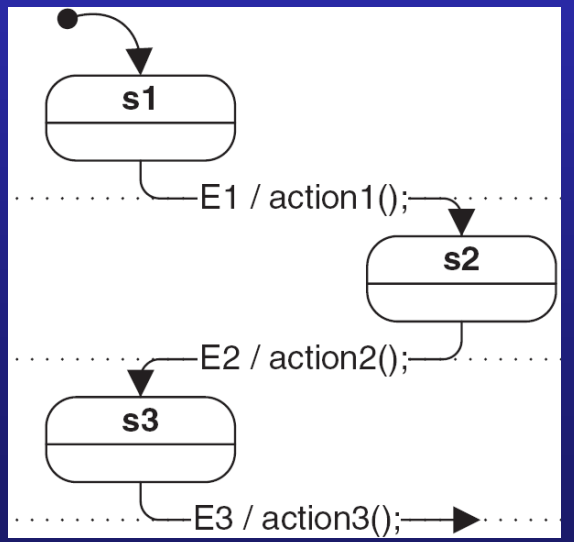

File Edit Selection View Go Run Terminal Help
ExamAIModelingCase3
EXPLORER
EXAMAIMODELINGCASE3
.vs
Assets
bin
Data
DatabaseInitializer.cs
ExamAIModelingCase3DBContext.cs
Domain
obj
Services
ViewModels
Views
App.axaml
App.axaml.cs
app.manifest
ExamAIModelingCase3.csproj
Program.cs
ViewLocator.cs
分析类图.puml
根据用例规约和分析类图生成系统的提示词.md
用例序列图解读.md
用例规约.puml
类图生成代码规则.md
OUTLINE
TIMELINE
ExamAIModelingCase3DBContext.cs
Data > ExamAIModelingCase3DBContext.cs
return result;
}
IExecutionStrategy strategy = Database.CreateExecutionStrategy();
return strategy.Execute(() =>
{
using var transaction = Database.BeginTransaction();
var result = save();
验证不变式();
transaction.Commit();
return result;
});
}
private void 验证不变式()
{
var 题型错误 = 题型s
.AsNoTracking()
.Any(x => x.正确选项最少个数 < 1 || x.正确选项最少个数 > x.正确选项最多个数);
if (题型错误)
{
throw new InvalidOperationException("题型不满足正确选项数量不变式。");
}
var 奖品错误 = 奖品s.AsNoTracking().Any(x => x.价值 < 0);
if (奖品错误)
{
throw new InvalidOperationException("奖品不满足价值不小于0的不变式。");
}
var 试题错误 = 试题s.AsNoTracking().Any(x => x.分值 < 0);
if (试题错误)
{
CHAT SPEC42 CODEX
执行系统提示词
不变式.puml。
数据库结构验证采用 EnsureCreated() 初始化路径，而不是生成迁移文件。
分值区间按 0-0 精确匹配 0 分，其余区间按左闭右开处理，避免 2 分同时落入两个区间。
运行程序可用：
powershell
dotnet run
Undo
Review
App.axaml.cs +31 -31
Da...DatabaseInitializer.cs +144 -0
ExamAIModelingCase3... +231 -0
Show 21 more files
Ask for follow-up changes
5.5
0 0
English
Ln 161, Col 6 Spaces: 4 UTF-8 LF C# Signed out Go Live

状态机

- 集合运算优先
  - 尽量变成属性值的差别
  - 通过关联显式建模
- 实在没办法，再变成行为差别
  - 活动图
  - 状态机图
- 实现风格。。。
  - 条件分支
  - 泛化







现在已经不重要

状态机

- 依据必须和领域特定相关
  - 状态：存在 已删除 ×
  - 行为差别：添加行为 成功 失败 ×
- 几个思考
  - 从类思考形容词
  - 对领域行为的不同响应
  - 生命周期跨越多次交互

不变式更合适：

题目.选项s->asSet()->includesAll(回答s.选项s->asSet())

值得画状态机的实体类