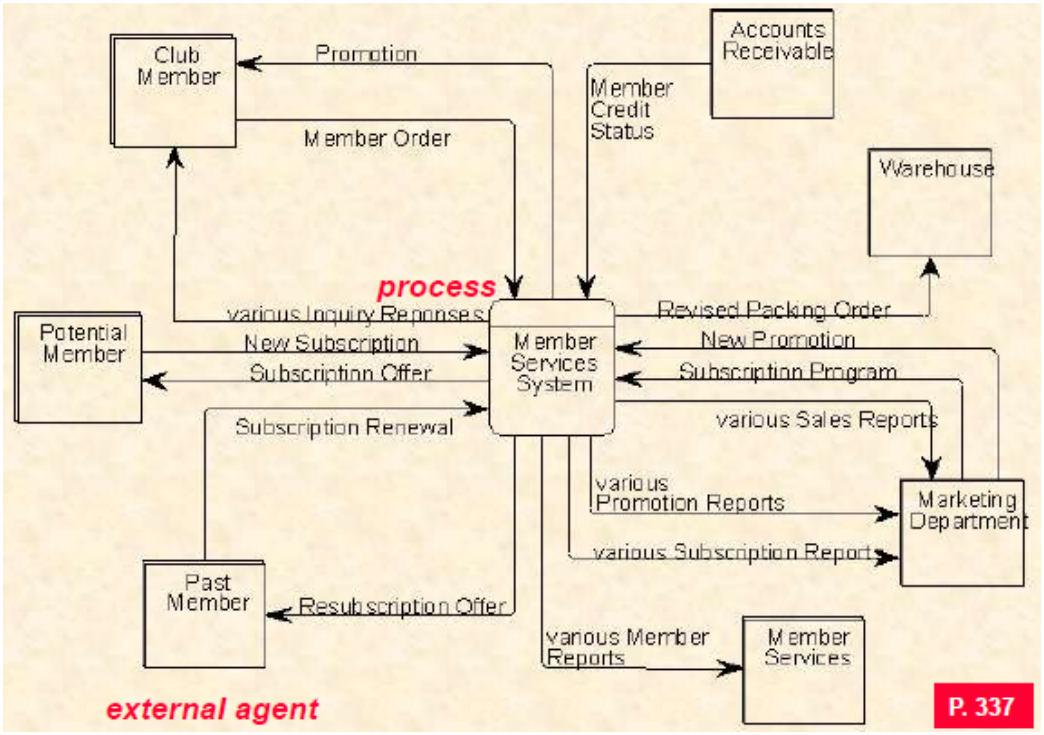
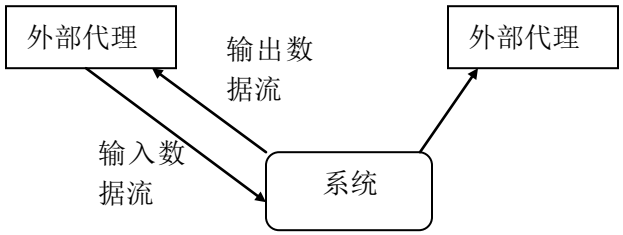
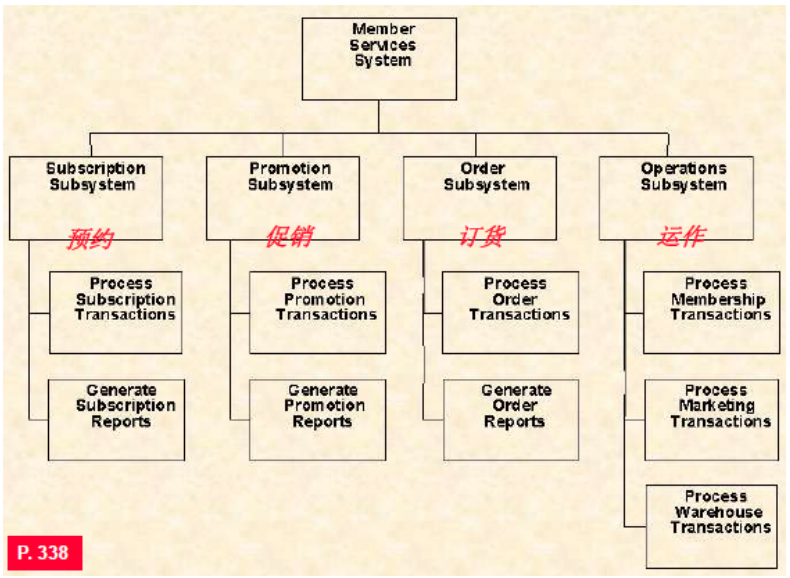
SAD 60 分超大题之女王大人简要总结版

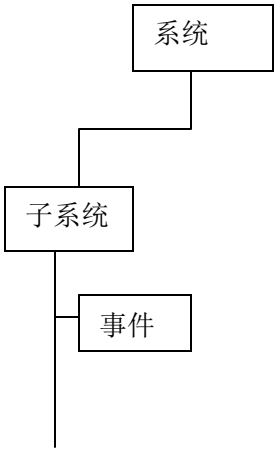
1、上下文数据流图



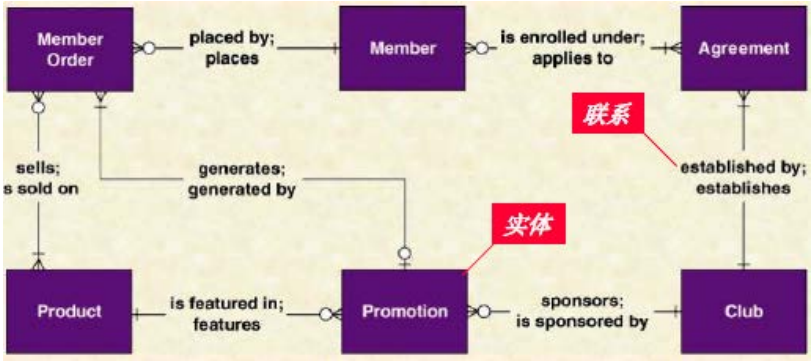


2、功能分解图





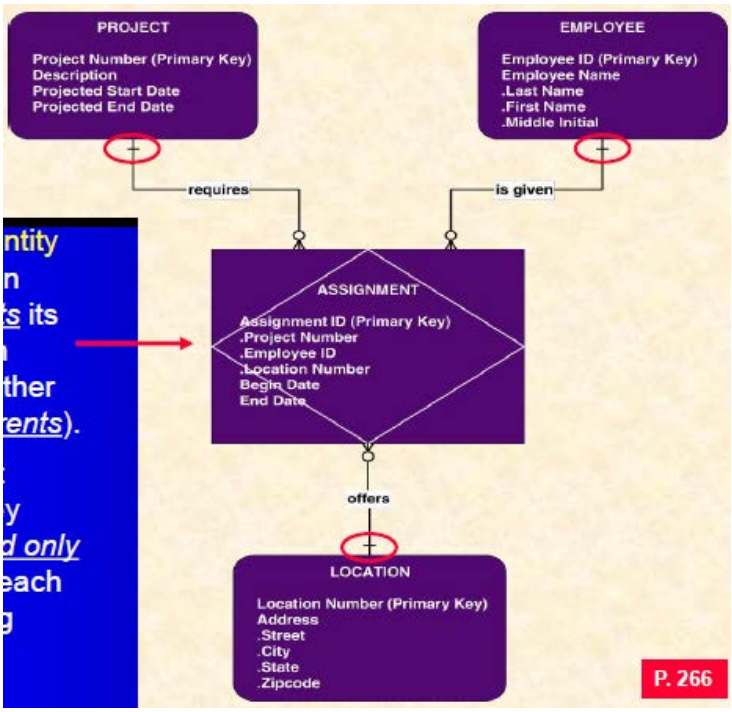
3、ER 图



实体：圆角矩形

基数含义	最小实例数	最大实例数	图形化符号
正好一个(一个且仅一个)	1	1	或
零个或一个	0	1	
一个或多个	1	多个(>1)	
零个、一个或多个	0	多个(>1)	
大于一个	>1	>1	

图 8-6 基数的符号



关联实体：从一个或多个其他实体继承主键的实体，只能指向每个连接实体的一个实例。

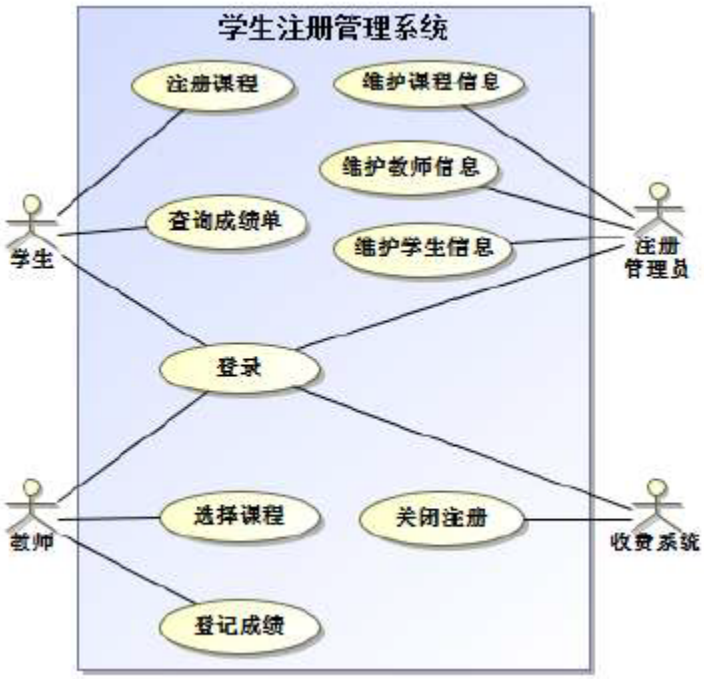
4、 normalized table

类似于数据库中的 关系模式

表名（主键#，外键#，属性）

- **Employee(employee number#, name, current address, telephone number, date of birth， gender)**
- **Landlord(social security number#, name, address, telephone number）**
- **Property(identifying number#, social security number#,address, type of property, maximum number of tenants, amount of the rent)**
- **Viewing(viewing id, date, social security number)**
- **ViewingEmployees(viewing id, employee number)**

5、用例图

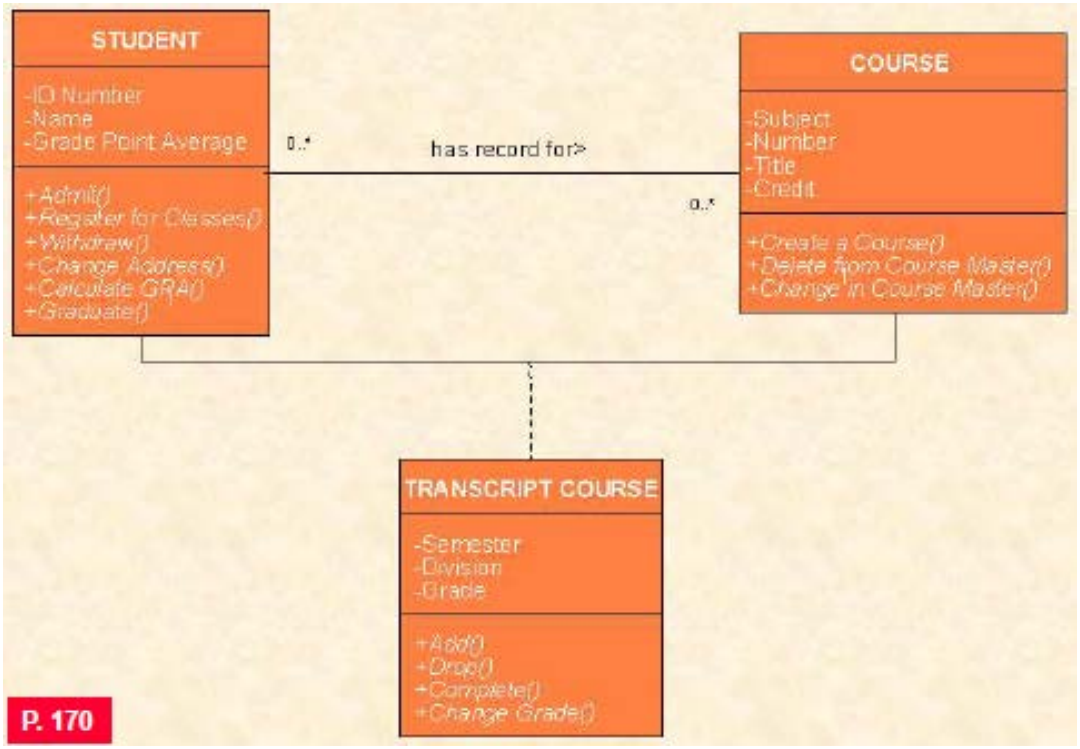


若有条件：当到了某个确定的日期或时间自动执行（自动触发）的，这个时序时间的参与者是**时间**

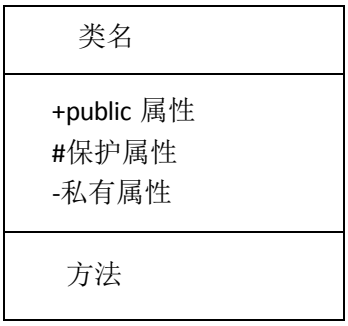
6、A primary use case 表格形式用例

用例名称		
参与者		
描述		
前置条件	用例执行前，系统状态约束条件	
事件流	参与者动作	系统响应
可替代事件过程	出现异常或变化时的用例行为	
结论	描述用例何时成功	
后置条件	用例执行后，系统的约束条件	

7、类图



Multiplicity	UML Multiplicity Notation	Association with Multiplicity	Association Meaning
Exactly 1	1 or *leave blank*	Employee — Works for — 1 — Department Employee — Works for — Department	An employee works for one and only one department.
Zero or one	0..1	Employee — Has — 0..1 — Spouse	An employee has either one or no spouse配偶
Zero or more	0..* or *	Customer — Makes — 0..* — Payment Customer — Makes — * — Payment	A customer can make no payment up to many payments.
One or more	1..*	University — Offers — 1..* — Course	A university offers at least 1 course up to many courses.



类图关系：

1. 泛化 (Generalization)

【泛化关系】：是一种继承关系，表示一般与特殊的关系，它指定了子类如何特化父类的所有特征和行为。例如：老虎是动物的一种，即有老虎的特性也有动物的共性。

【箭头指向】：带三角箭头的实线，箭头指向父类

2. 实现 (Realization)

【实现关系】：是一种类与接口的关系，表示类是接口所有特征和行为的实现.

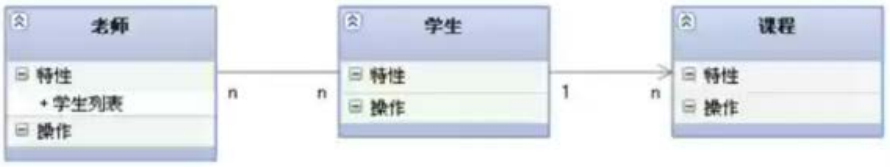
【箭头指向】：带三角箭头的虚线，箭头指向接口

3. 关联 (Association)

【关联关系】：是一种拥有的关系，它使一个类知道另一个类的属性和方法；如：老师与学生，丈夫与妻子关联可以是双向的，也可以是单向的。双向的关联可以有两个箭头或者没有箭头，单向的关联有一个箭头。

【代码体现】：成员变量

【箭头及指向】：带普通箭头的实心线，指向被拥有者



上图中，老师与学生是双向关联，老师有多名学生，学生也可能有多名老师，但学生与某课程间的关系为单向关联，一名学生可能要上多门课程，课程是个抽象的东西他不拥有学生。

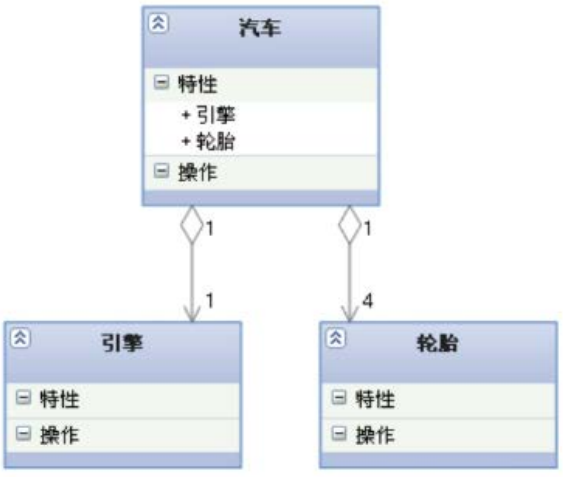
4. 聚合 (Aggregation)

【聚合关系】：是整体与部分的关系，且部分可以离开整体而单独存在。如车和轮胎是整体和部分的关系，轮胎离开车仍然可以存在。

聚合关系是关联关系的一种，是强的关联关系；关联和聚合在语法上无法区分，必须考察具体的逻辑关系。

【代码体现】：成员变量

【箭头及指向】：带空心菱形的实心线，菱形指向整体



5. 组合(Composition)

【组合关系】：是整体与部分的关系，但部分不能离开整体而单独存在。如公司和部门是整体和部分的关系，没有公司就不存在部门。

组合关系是关联关系的一种，是比聚合关系还要强的关系，它要求普通的聚合关系中代表整体的对象负责代表部分的对象的生命周期。

【代码体现】：成员变量

【箭头及指向】：带实心菱形的实线，菱形指向整体

6. 依赖(Dependency)

【依赖关系】：是一种使用的关系，即一个类的实现需要另一个类的协助，所以要尽量不使用双向的互相依赖。

【代码表现】：局部变量、方法的参数或者对静态方法的调用

【箭头及指向】：带箭头的虚线，指向被使用者

各种关系的强弱顺序：

泛化 = 实现 > 组合 > 聚合 > 关联 > 依赖

下面这张UML图，比较形象地展示了各种类图关系：