▶ 관계 (Relationship)

1) Association

- 구조적 관계 표현
- 실선으로 표기, 필요에따라 Role Name , Multiplicity , Navigable 표시
- 객체별 연관관계 표시
- 두 클래스가 Association 관계에 있다면 한쪽에서 다른한쪽을 참조 할수 있음을 의미
- Notation

- Code

```
class A{
      B b;
 }

class B{
}
```

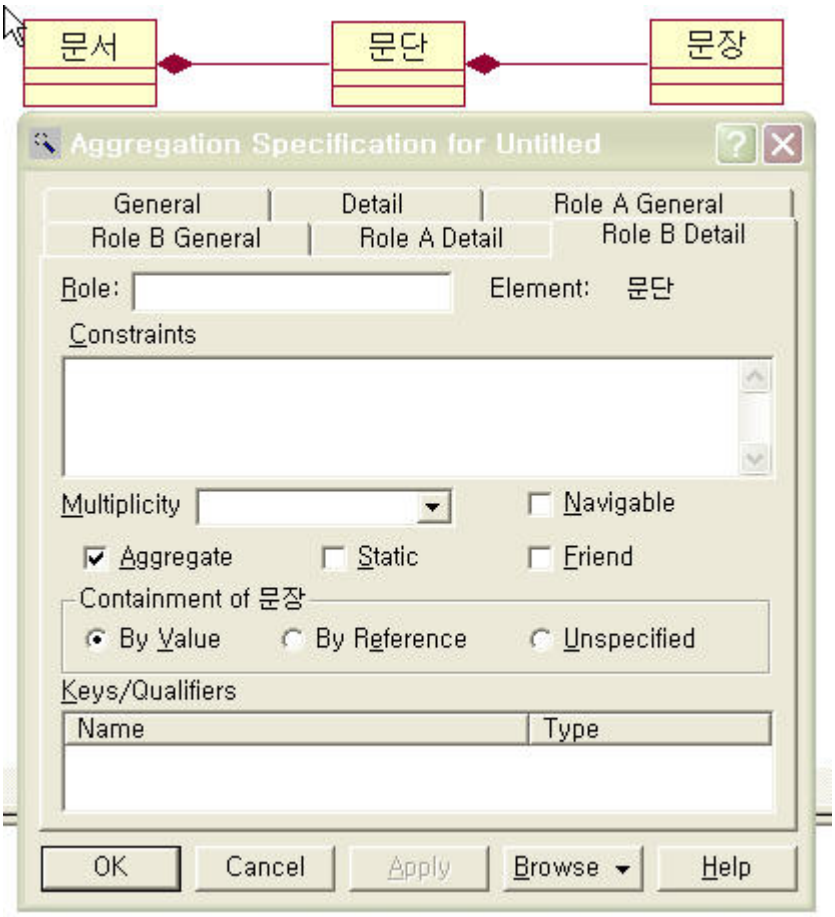
2) Aggregation : 집합

- 두 클래스가 Association 관계 이면서 전체와 부분의 관계 일경우
- Notation

- Rose에서 Association 표기 선택 후 마우스 오른쪽 클릭 Aggregate

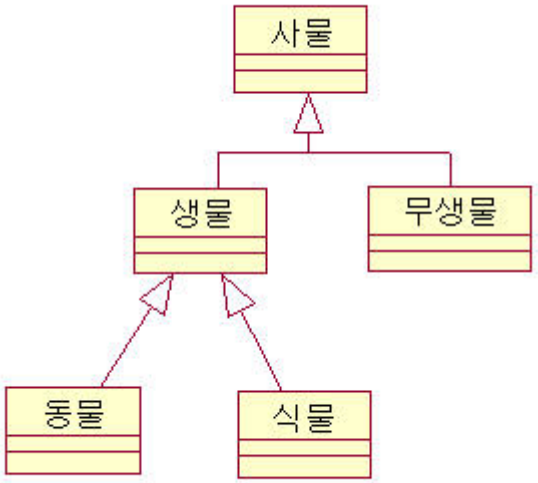
3) Composition : Aggregation의 특수한 경우

- Aggregation의 특수 경우 이며 전체 소멸시 부분도 함께 소멸
- 강한 소유의 표시
- Notation



* Aggregation 선택후 Rose 에서 마우스 오른쪽 > Open Spec > By Value 체크

4) Generalization : 일반화
- 일반화된 개념적 사물과 구체화된 특수 사물의 관계 표현
- 부모 자식 간의 상속 개념
- 자식은 부모의 속성과 행동을 공유
- Notation



5) Realization : 실체화
- 정의 와 구현 관계 표현
- Use Case에 정의된 기능을 구현하는 Collaboration에 연결시 사용
- 정의된 interface와 이를 구현하는 Class 연결시 사용
- Notation

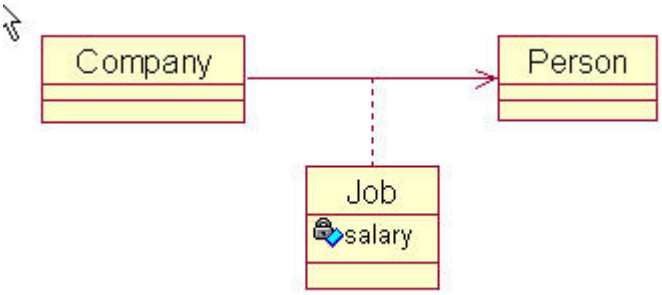
6) Dependency
- 의존형 관계 표시
- Class 간 의존은 필요할때 만들어 사용하여 버린다는 의미
- Association 은 지속적이며 강한 관계, Dependency 는 일시적인 약한 관계
- Notation

- Code

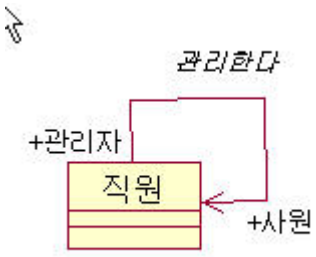
```
class A
{
      void play(B b)
     {
     }
}

class B
{
   int num;
}
```

7) Association Class
- Association자체가 속성을 가질때 클래스로 모델링 한다.
- Notation



8) Recursie Association
- 동일한 클래스 내에서의 Association
- Notation


관리한다
+관리자
직원
+사원