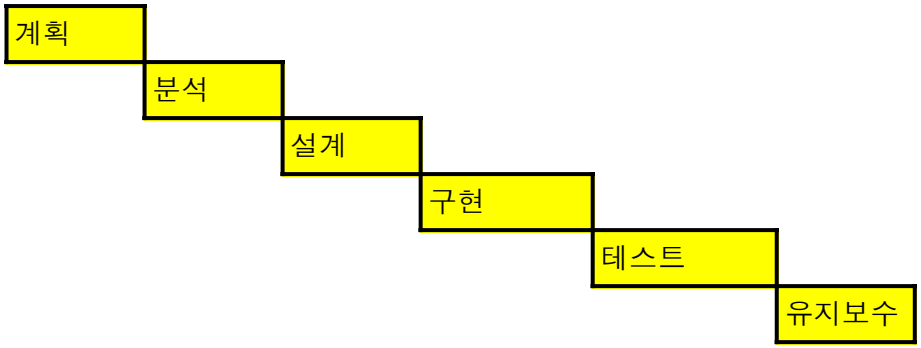
프로젝트 개발 방법론

1. 폭포수 방법론(Waterfall)



<필요성>

1) 주먹구구식 개발을 정확한 절차에 의거해 개발하는 필요성

2) 항공모함, NASA 우주선 개발과 같은 큰 비용, 미지의 세계

철저한 사전준비, 사전계획, 사전분석, 사전설계

<단점>

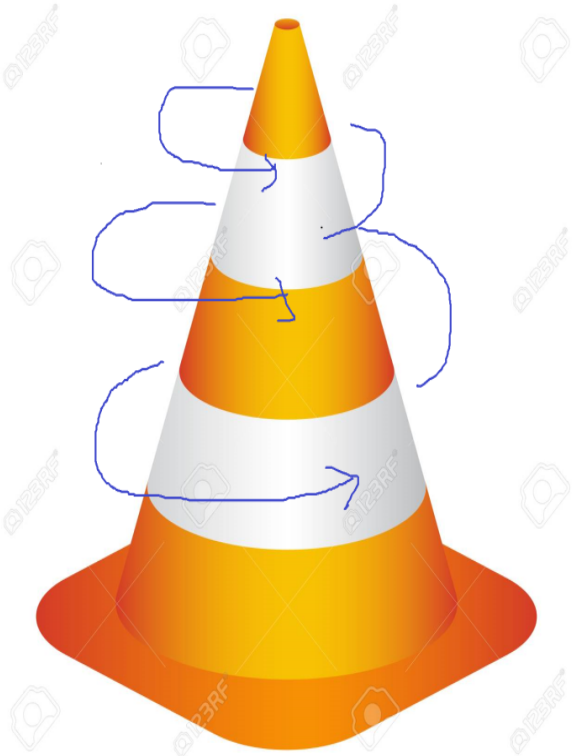
1) 현실의 변화하는 상황을 반영하지 못하는 유연하지 못한 개발

2) 미지의 개발시 국가와 같은 대량 자본, 수많은 엘리트 투입가능경우는 양호

자본이나 인재가 넉넉하지 않은 주체는 많이 힘들다

But, 모든 개발방법론의 근간을 이루는 방법론

2. 나선형 방법론(Spiral)



<필요성>

1) 폭포수는 이전단계로 돌아갈 수 없는 고정된 방식의 문제점을 개선

현실은 항상 변화한다

2) 개발하는 과정에서 다시 문제점/보완점은 계획->구현/테스트

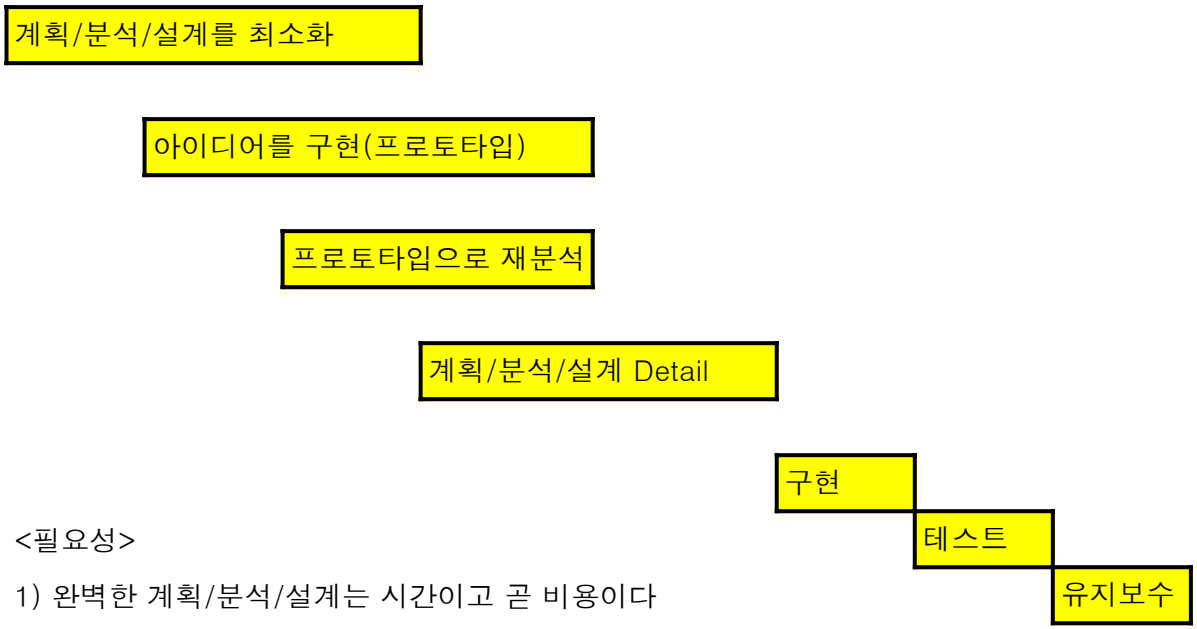
3) 폭포수시대보다 SW 규모가 매우 커짐

<단점>

1) 오래 걸린다

2) 고객이 중간에 계속 바꿔도 좋다고 생각한다

3. 발전적 프로토타입 개발 방법



<필요성>

1) 완벽한 계획/분석/설계는 시간이고 곧 비용이다

2) 어차피 구체적인 모습을 보기 전에는 허상일 수도 있다

3) 빨리 만들어서 구체적인 모습으로 다시 계획/분석/설계

4) 소규모 주체, 비용/인재가 많지 않은 미지의 개발

5) 실 HW와 결합된 경우, 효과적인 경우가 많다

<단점>

1) 프로토타입은 프로토타입일 뿐,

프로토타입을 버리고 다시 시작해야 하는데

비용때문에 프로토타입을 연장으로 가져가는 경우가 매우 많다

2) 계획/분석/설계를 건너뛰려고 하는 경우가 종종 발생한다

4. 애자일 방법론

1) 지나친 문서화를 경계

사문서가 되는 경우가 매우 많다

App은 유지보수에 의해 변하는데 문서는 No Update

2) 실제 SW개발현실은 개발자의 자질에 의해 좌우

시스템 절대주의는 위험하다

3) 절차에 의한 느린 대응/본질이 아닌 것을 배격

현장의 빠른 대응/개발

4) 지속적 통합 : 고객/현장의 요구사항을 빠르게 개발에 적용

일일빌드: 빌드를 시작하면 통합/테스트/완제품 서비스

자동화를 매우 중시

<방법>

1) 문서보다 코드를 잘 이해할 수 있게 짠다

코드표기법 사전정의

2) 코드를 설명하듯이, 쉽게, 코드 자체가 문서

3) 지나친 회의는 No

아침에 10분 스탠드 업 – 오늘 할일 얘기

4) 툴 많이 사용

소스관리, 테스트, 프로젝트 관리, 버그추적

5) 짝 프로그래밍(Pair Programming)

한 사람이 코딩, 옆의 사람이 관찰, 의견교류

고객과 함께 프로그래밍, 의견을 적극 수용

시스템 < 사람

<단점>

1) 문서를 배격하다보니 아예 안만드는 사람도 있다

근거가 없다

2) 맘에 맞는 사람이랑만 일하려는 증세가 생긴다

3) 조금이라도 형식이 있으면 허례허식이라고 공격

시스템 무시 경향

4) 무형시스템은 강하고 유형시스템은 약해서

자질이 좀 부족하면 적응이나 교육이 힘들다

업무 분석

문서(서류, 장표, 보고서, 양식)

데이터베이스 (권한)

담당자 인터뷰

분석가의 관찰력, 상식, 논리, 경험

데이터베이스 모델링

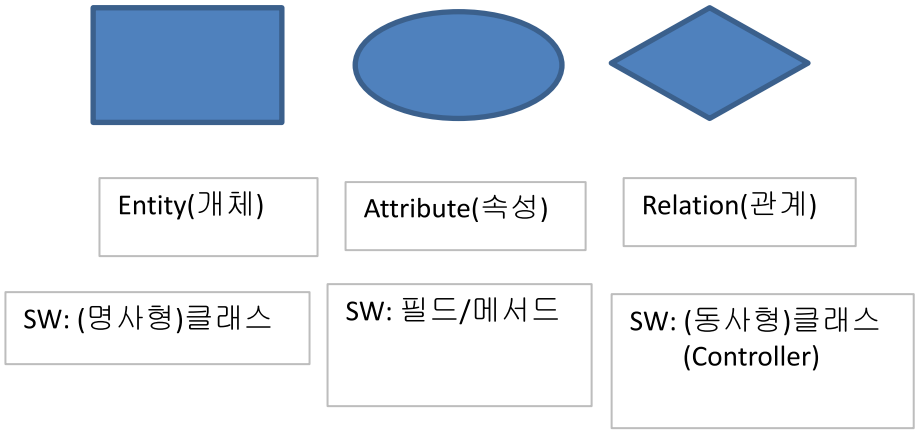
건물의 바닥을 다지는 공사

SW 설계/구현 보다도 데이터베이스 모델링이 선행작업

1. 개념적 데이터베이스 모델링이란?

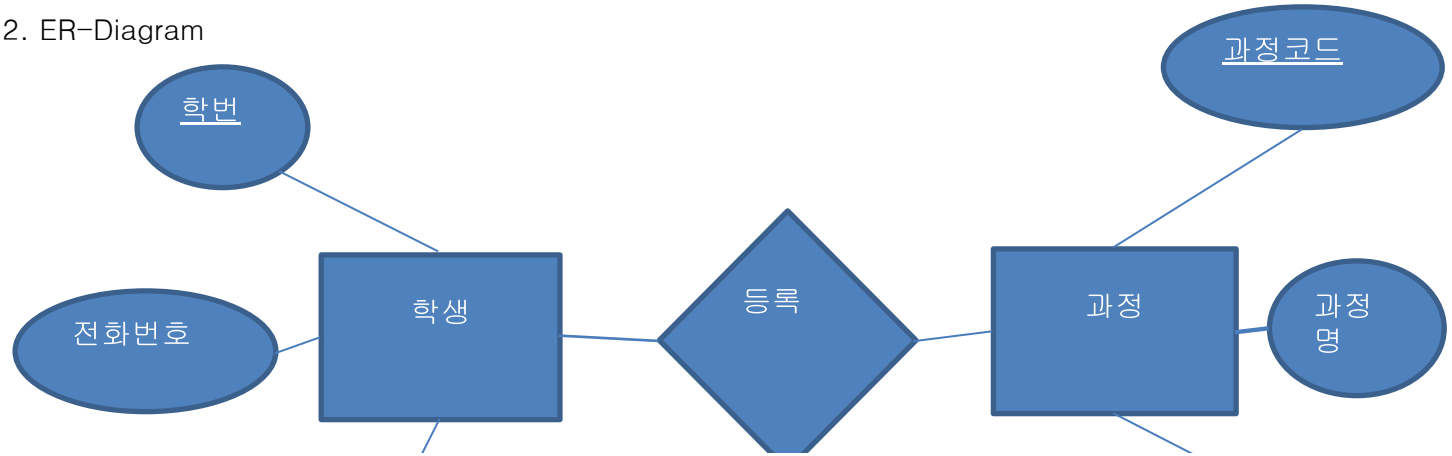
업무 분석 단계에서 얻어진 내용을 토대로
우선 엔티티(Entity)를 추출하고 엔티티내의 어트리뷰트(Attribute
를 구성하며, 엔티티간의 관계를 정리해서
ER-Diagram으로 정의하는 단계이다.

1. E-R Model(Entity - Relation Model, 개체-관계 모델)



1976. P.Chen이 제안한 모델

2. ER-Diagram




학생이름
학점

1. Entity (실체) 정의

1) 명사-동사 분석법

명사중에 큰 개념 -> Entity

명사중에 작은 개념 -> Attribute

2) 개별단어들의 공통 집합

자료구조, 데이터베이스, C, Java : 과목

컴공, 경영, 국문, 영문, 행정, 한의: 학과

3) 관련 지식 공부

해당 조직 특성 공부

회계, 업무 관련 서적

4) 서류, DB

5) "학원에서 학생들의 출결상태와 성적등을 과목별로 관리하기를 원하고 있다"

명사 : 학원, 학생, 출결상태, 성적, 과목

학원이 여러 개인지 아니면 그냥 그 학원에서만 관리

앞에 해당하면 학원도 Entity, 뒤에 해당하면 No Entity

6) 모델링에서 상상력을 동원하는 것은 매우 중요하다

그러나 모델링 작업시 실제 업무에 있는 실체인지 아닌지 검증해야 한다.

Entity명을 부여할 때 실무에서 자주쓰는 용어, 명사를 사용하여 가독성을 높이는 것이 좋다

1. Attribute(속성)이란?

Entity(실체)를 이루는 구체적인 항목

성질, 분류, 수량, 상태, 특성, 특징…

엔티티당 대략 10개 내외로 하는 게 좋다.

그 이상으로 많이 나오면 엔티티의 분리를 고려

2. 예시

학생 : 학번, 이름, 주민번호, 전화번호, 주소…

사원 : 사번, 이름, 주민번호, 전화번호, 주소, 입사일, 급여…

1. 속성의 유형

1) 기초 속성 : 원래 갖고 있는 속성, 현업에서 기본적으로 사용되는 속성

2) 추출 속성 : 기초 속성으로부터 가공처리(계산)에 의해서 얻어질 수 있는 속성. 이는 자료의 중복성 및 무결성 확보를 위해 최소화시키는 것이 바람직하다

3) 설계 속성 : 실제로 존재하지는 않으나 시스템의 효율성을 도모하기 위해 설계자가 임의로 부여하는 속성

2. 예시

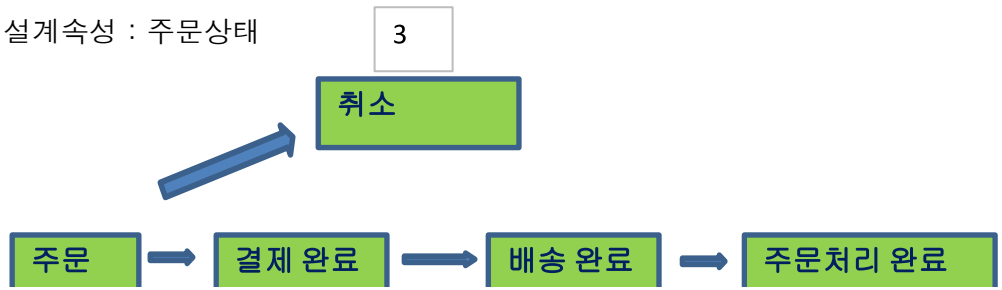
주문

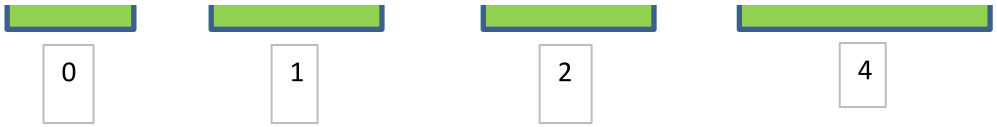
주문번호	고객명	주문상품	주문일자	단가	수량	주문총금액	주문상태
1	홍길동	H302	20200105	10,000	2	20,000	0
2	임꺽정	L202	20200107	15,000	3	45,000	3
3	장길산	J101	20200210	20,000	1	20,000	2

기초속성 : 주문번호, 고객명, 주문상품, 주문일자, 단가, 수량

추출속성 : 주문총금액

설계속성 : 주문상태





주문 프로세스를 관찰하여 설계속성을 추가한다

이 설계 속성은 정해진 것이 아니라 설계자의 관찰에 따르기 때문에

어느 정도 주관적인 요소를 가지게 된다

또 주문상태를 처리할 때 테이블에서는 "문자열"을 사용하지 않고 "숫자"를

사용하게 된다. 숫자 처리가 더 빠르다

테이블에 "주문총금액"과 같은 추출 속성을 포함시키게 되면

미리 계산되어진 데이터를 읽기 때문에 조회속도가 향상되게 된다.

자주 조회되는 컬럼은 이렇게 추출 속성을 만들어 놓을 것을 고려하게 된다.

But, 지나치게 많은 추출속성을 사용하게 되면 무결성을 훼손할 위험도 있다.

단가	수량	주문총금액
10,000	2	25,000

1. 식별자란?

한 Entity내에서 각각의 인스턴스를 유일(Unique)하게
구분할 수 있는 단일 속성 또는 속성 그룹을 말한다

RDBMS에서는 모든 Entity는 1개 이상의 식별자를 보유하는
것을 원칙으로 한다

2. 식별자의 유형

1) 후보키(Candidate Key)

Entity내의 각각의 인스턴스를 유일하게 구분하는 속성
기본키(Primary Key)가 될 수 있는 후보 속성

Entity : 사원

사원번호	**주민번호**	이름	주소	소속부서
1	1111-1111	홍길동	지리산	총무부
2	2222-2222	임꺽정	구월산	전산부
3	3333-3333	장길산	황해도	인사부

여기서 후보키는 '사원번호'와 '주민번호'이다.

2) 기본키(Primary Key)

후보키중에 대표성을 가지기에 가장 적합한 키
1) 해당 Entity를 대표할 것

2) 업무적 활용도가 높을 것

3) 길이가 짧을 것(검색 속도)

기본키를 정의한 컬럼은 기본적으로 Not Null과

No Duplicate 속성이 적용되며

Unique한 Clustered Index가 만들어지게 된다

(해당 속성으로 검색이 빨라짐)

여기서는 '사원번호'가 기본키로 선정된다

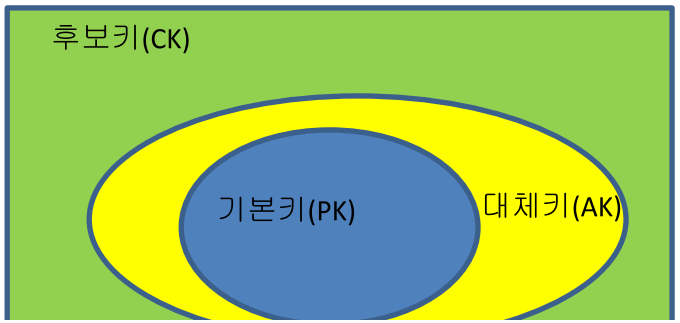
3) 대체키(Alternate Key)

후보키 중에 기본키로 선정되지 않는 후보키

여기서는 '주민번호'가 된다.

나중에 검색에 자주 사용될 수 있고

Index를 생성할 때 고려 대상이 된다



4) 복합키(Composite Key)

하나의 속성으로는 기본키가 될 수 없는 경우에
둘 이상의 속성을 묶어서 식별자로 정의하는데
이를 복합키라 부른다.

이 때 고려해야 할 사항은 어느 복합키를 먼저
둘 것이냐 하는 것이다.
먼저 두는 속성을 기준으로 Index가 생성되므로
복합키 속성중에 주로 조회의 조건으로 사용되는
컬럼을 먼저 정의하는 것이 성능에 좋다.

Entity: 급여내역

급여지급연월	사번	지급일자	급여액
202001	1	20200110	4,500,000
202001	2	20200110	5,000,000
202002	1	20200210	4,850,000
202002	2	20200210	5,200,000

기본키 : 사번 : 매월 사번이 쌓이므로 해당이 안된다
기본키 : 급여지급연월 : 같은 달에 여러 사원 지급

기본키 : 급여지급연월+사번

5) 대리키(Surrogate Key)

식별자가 너무 길거나 여러 개의 속성으로 구성된 경우
인위적으로 추가한 식별키로 '인공키'라고도 한다.

Entity에서 식별자가 여러 속성으로 구성된 경우는
데이터를 수정하거나 검색을 하는데 수행 속도가
떨어질 수가 있다.

대리키는 성능향상을 고려하는 역정규화의 한가지 방법이다

Entity: 급여내역

급여일련번호	급여지급연월	사번	지급일자	급여액
1	202001	1	20200110	4,500,000
2	202001	2	20200110	5,000,000
3	202002	1	20200210	4,850,000
4	202002	2	20200210	5,200,000

자식 테이블과 Join하는 경우 복합키를 가지는 것보다
일련번호'를 추가해서 단순한 식별키를 외부키(FK)로 전이시키면
Join 속도와 검색속도를 향상시킬 수 있다.
이를 '대리키'또는 '인공키'라고 한다.

Oracle에서는 Sequence로 일련번호를 생성한다

1. 관계(Relation)란?

Entity간의 업무적인 연관성이다.

2. 예시

<비디오 대여점 관리 서비스>

회원은 비디오를 대여한다.

비디오는 회원에게 대여되어진다.

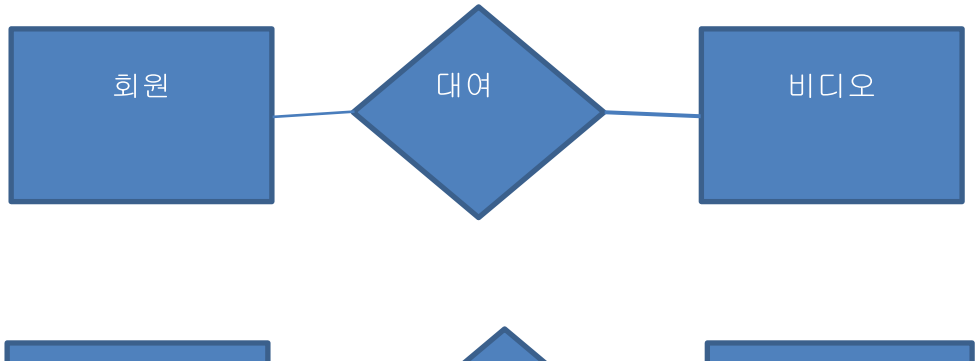
<학사관리 데이터베이스>

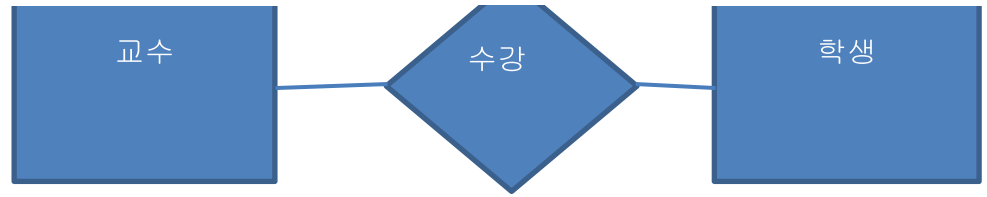
교수는 학생을 가르친다.

학생은 교수로부터 배운다.

두 Entity간에 동사를 중심으로 관계를 정의한다

3. 관계 부여




교수
수강
학생


회원
구매/판매/주문
상품

1. 차수성이란?

한 Entity의 하나의 인스턴스가 다른 Entity의 몇 개의 인스턴스와 관련될 수 있는가?를 정의하는 것

2. 종류

2-1) 1 : 1

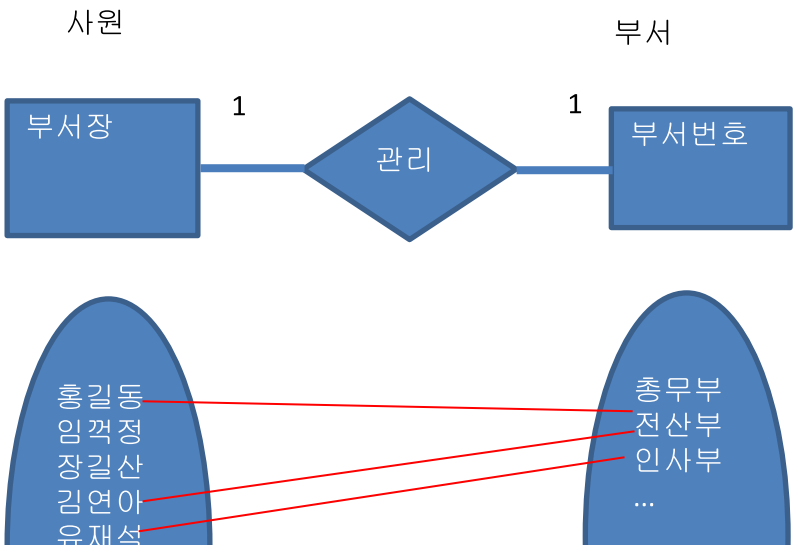
2-2) 1 : 다

2-3) 다 : 다

3. 1 : 1

ex) 사원 Entity의 부서장 사번(사원번호)

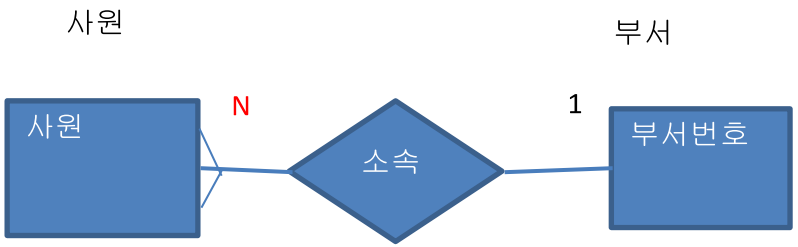
부서 Entity의 부서번호

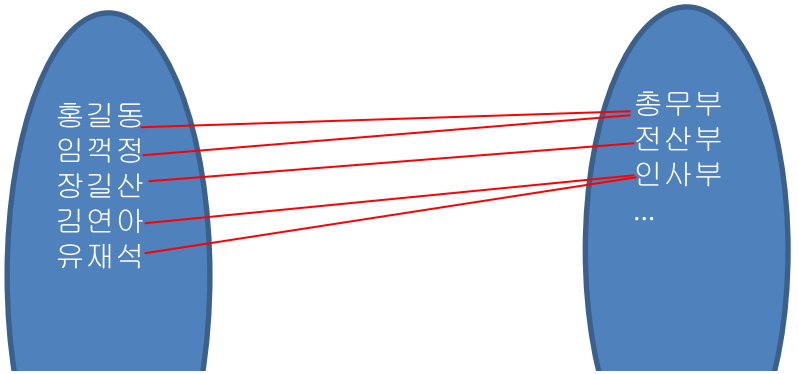


4. 1:다

ex) 사원 Entity의 인스턴스 : 다

부서 Entity의 인스턴스 : 1

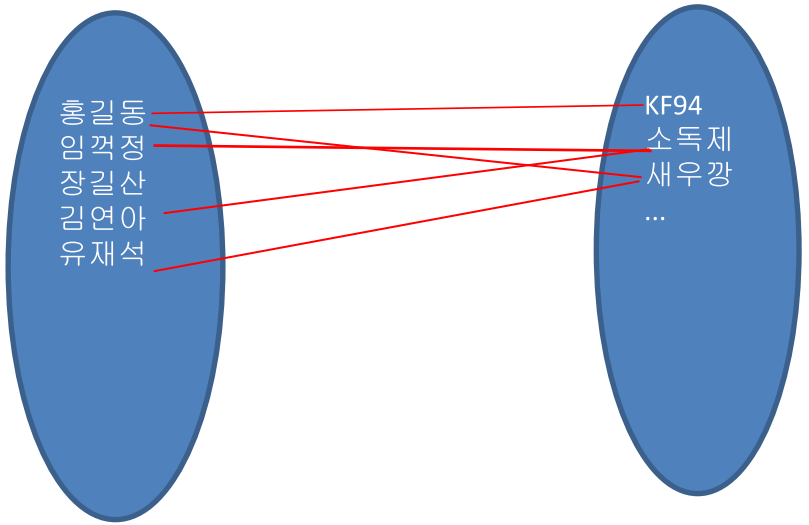




5. 다 : 다

ex) 고객 Entity : 다

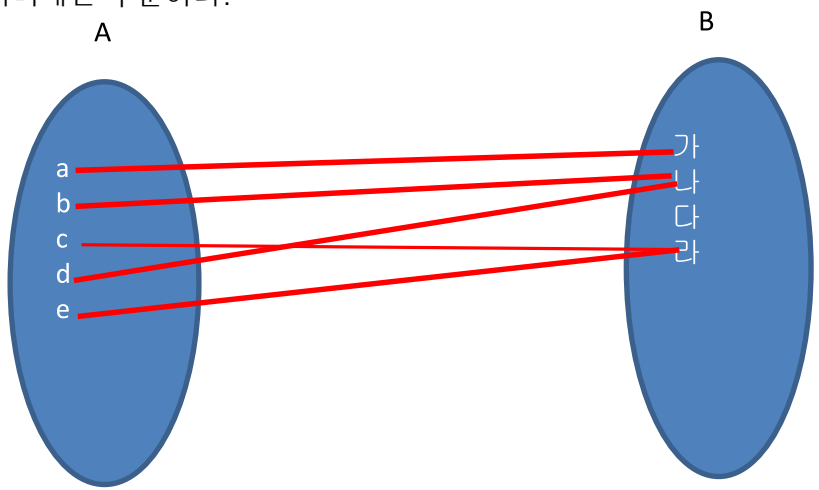
상품 Entity : 다



N:M관계는 모델링 과정을 통해 1:N관계로 변환되어진다

1. 선택성

두 Entity(실체)간에 관계가 설정되었을 때, 항상 두 실체의 모든 인스턴스간에 관계가 존재해야 하는지, 아니면 모든 인스턴스에 대해 존재할 필요가 없는지를 나타내는 부분이다.

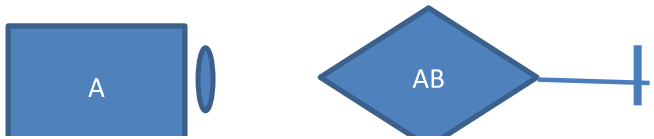


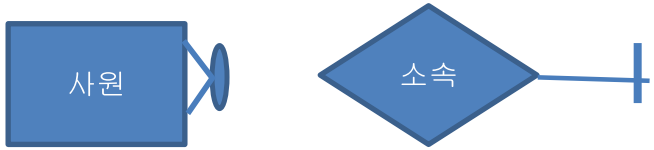
관계에 모든 인스턴스가 참여하는 것을 Mandatory라고 한다

관계에 일부 인스턴스가 참여하는 것을 Optional이라고 한다

A의 인스턴스는 반드시 AB관계에 참여해야만 한다(Must be)

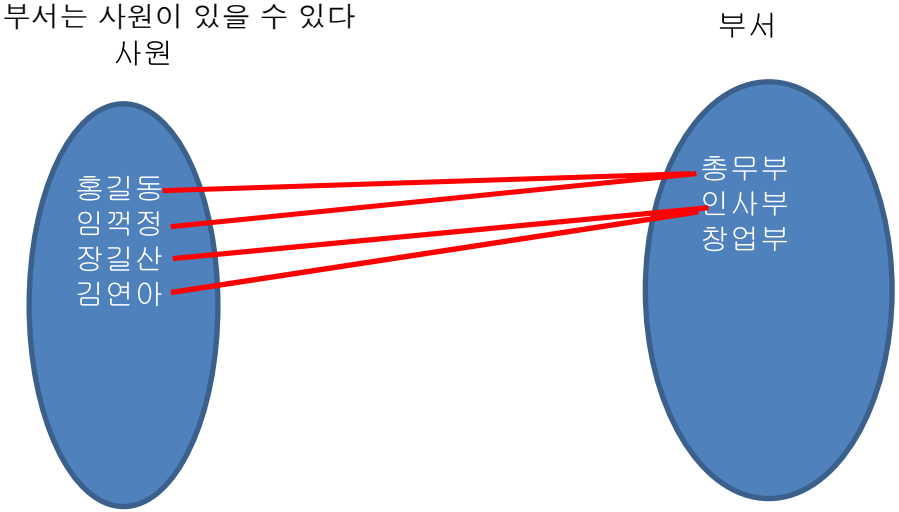
B의 인스턴스는 AB관계에 참여할지도 모른다(Maybe)





사원은 반드시 부서에 소속되어야 한다

부서는 사원이 있을 수 있다



<고객과 상품간의 관계>


고객
주문

1단계

업무 분석 : 고객의 요구사항 기반

부서와 사원을 관리한다

모든 사원은 부서에 소속되어야 한다

부서는 사원이 있을 수 있다

2단계

Entity(실체) 도출

부서, 사원

3단계

Attibute 도출

부서 - 부서번호, 부서명, 위치

사원 - 사번, 이름, 주소, 성별, 입사일

4단계

Attibute중에 식별자(identifier) 도출

부서 - 부서번호

사원 - 사번

5단계

관계 설정

부서에는 사원이 배치되어진다.

사원은 부서에 소속되어진다.

6단계

관계 차수 설정

각 부서에는 하나 이상의 사원이 배치되어질지도 모른다

각 사원은 단 하나의 부서에 소속되어진다

.=> 1:N 관계 성립

7단계 관계 선택성 도출

각 부서에는 하나 이상의 사원이배치되어질지도 모른다

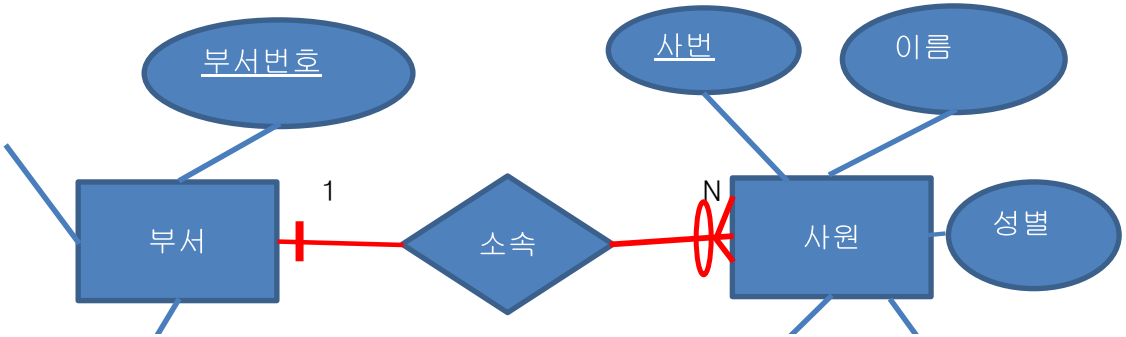
각 사원은 단 하나의 부서에 소속되어진다

부서 .-> 사원 : Optional(Maybe)

사원 .-> 부서 : Mandatory(Must be)

8단계

개념적 데이터베이스 모델링 ER-Diagram




위치
입사일
주소