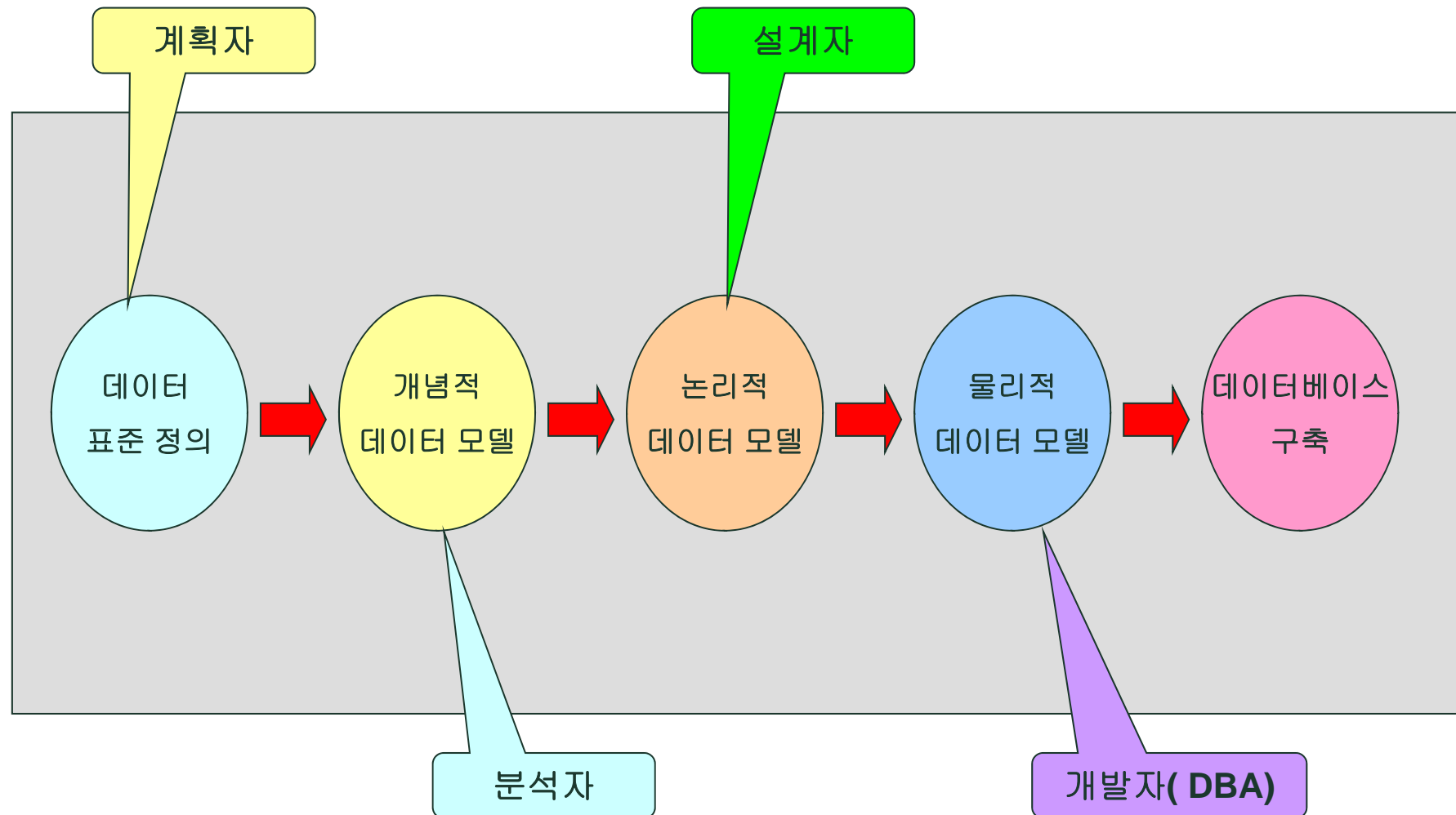


# Data Arcitecture 란 ?

- . **데이터베이스 구축**을 위한 계획, 분석, 설계, 구현과 관련된 전 단계의 절차를 체계화하여 기업의 목적을 달성하기 **위한 인프라를 구축하는 것**을 의미합니다.



# 방법론이란 ?

- . 정보 시스템의 계획, 분석, 구축, 운영 등과 관련된 방법들의 체계적인 집합을 의미

## 방법론(Methodology)

구조적 방법론

정보공학 방법론

객체지향 방법론

## 방법 (Method)

DFD(Data Flow Diagram)

ERD(Entity Relationship Diagram)

Class Diagram

# 방법론의 필요성

- . 효과적인 정보 시스템을 구축하고 운영하는 것은 기업 경영의 결정적인 요인이다. 또한, 정보 시스템의 효과성과 효율성은 적용하는 방법론에 따라 성패가 좌우된다.

1) 다수 개발자 간의 의사소통의 수단

2) 정보 시스템의 품질보증 수단

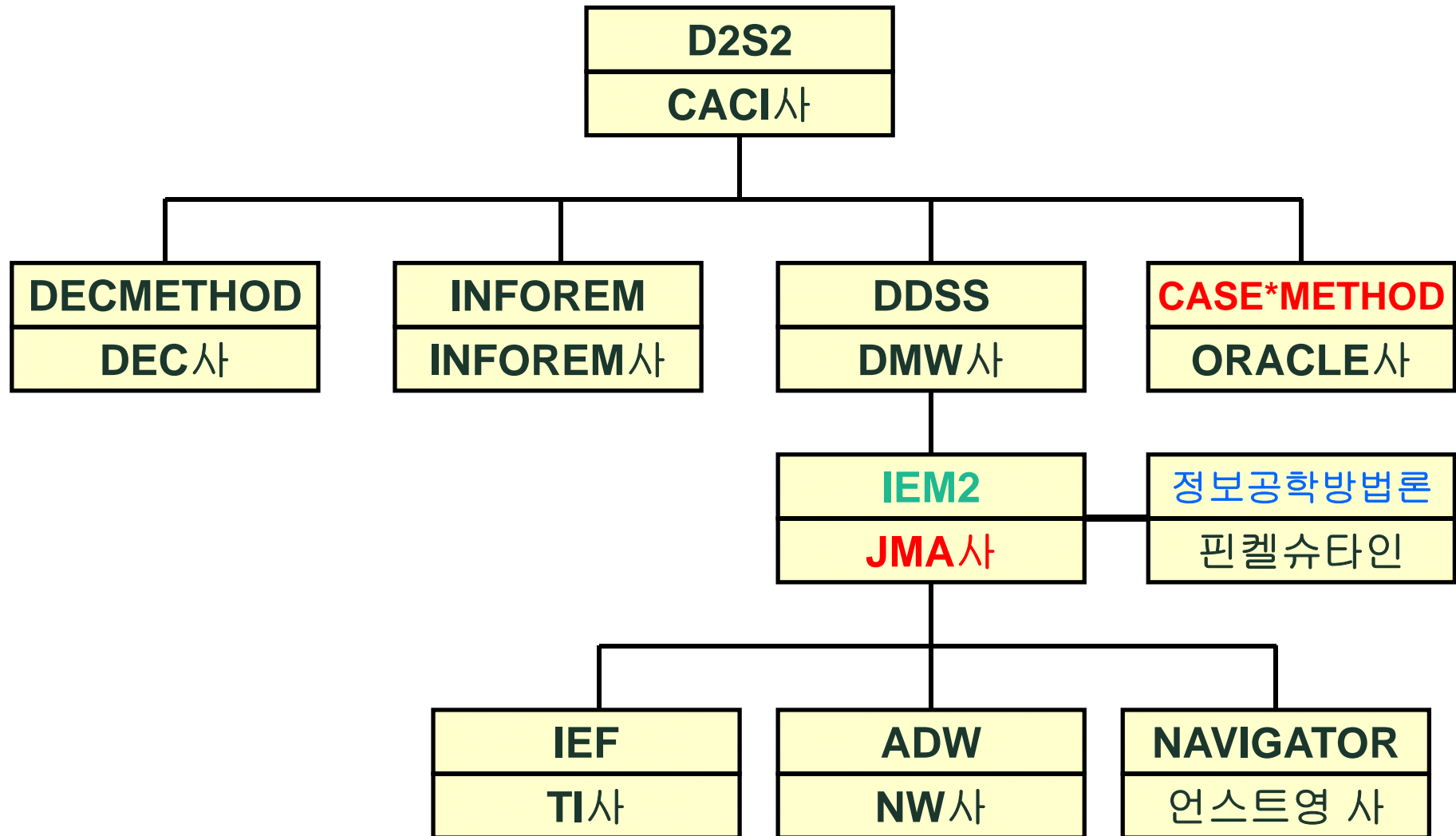
3) 생산성의 향상

4) 재사용의 가능성

5) 효과적인 프로젝트의 관리

6) 기술수준의 향상

# 정보공학 방법론의 발전과정



# 개발단계

사용자의 요구

계획

정보에 대한 요구  
(DATA)

프로그램에 대한 요구  
(PROCESS)

분석

개념적 데이터 모델링  
(Entity Relational Diagram)

기능 모델링  
(Function Hierarchy Diagram)

설계

논리적 데이터 모델링  
(TABLE, INDEX, VIEW 등)

프로그램 설계  
(Highrical Diagram)

구축

물리적 데이터 모델링  
(테이블과 인덱스 크기 설계 등)

프로그램 구축

정보 시스템

## 2

### 요구사항 수집 및 장표 분석

# 요구사항

저는 스포츠용품 소매 상점들의 주문을 받아 스포츠 용품을 전 세계적으로 판매하는 회사의 Manager 입니다. 각 상점들이 우리의 고객이며 지금 현재 전 세계적으로 15 고객사와 거래를 하고 있으며 점점 확장하려고 합니다.

우리의 가장 큰 고객은 미국 캘리포니아 주의 샌프란시스코에 있는 Big John's Sports Emporium과 워싱턴 주의 워싱턴에 있는 Woman Sport입니다. 우리는 각 고객에게 고객번호와 이름을 부여하고 그것을 관리합니다. 거기엔 전화번호, 주소, 도시명, 주명, 국가명, 우편번호, 담당 영업사원, 신용등급(EXCELLENT, GOOD, POOR), 지역, 비고를 함께 기재합니다. 저희 회사의 주력상품은 스포츠용품이며 약 150 종류의 제품에 대해 제품별 고유번호가 있고 제품명을 함께 사용 합니다.

그리고, 각 지역마다 창고를 배치하여 제품에 대한 재고를 창고별로 저장하고 고객의 주문을 받습니다. 각각의 주문에 대해서는 주문번호가 있고 고객번호, 고객명, 주문날짜, 선적날짜, 담당 영업사원, 주문금액 그리고 가능한 경우에 한해 지불방법(Credit, Cash), 선적여부(Y 또는 N)들을 함께 기재합니다.

또한, 주문에 대한 주문항목은 항목번호, 제품번호, 제품명, 단가, 주문수량, 선적수량 등으로 구성됩니다. 최근에는 월 평균 10,000건의 주문접수를 받고 있습니다.

현재, 우리의 시장은 세계 5개 지역으로 나눠 놓았는데, 북미, 남미, 아프리카 및 중동, 아시아, 그리고 유럽입니다. 이 곳들은 각각의 창고로 미리 구분되어 할당을 해 놓습니다.

각각의 창고는 창고 고유번호가 있어서 그 번호로 창고명, 지역, 전화번호, 주소, 도시명, 주명, 국가명, 우편번호, 담당 관리사원 등을 알 수 있습니다. 그리고, 모든 창고는 담당 영업 사원에 의해 관리됩니다. 현재는 각 지역에 하나의 창고 만 있지만 앞으로 곧 늘어 날 계획입니다.

저는 영업부에서 스포츠용품의 주문을 접수하고 관리하는 일을 담당합니다. 고객이 원할 때마다 주문을 받고 그 주문을 관리하는 책임을 가지고 있습니다. 또한, 관리부에서는 각 부서별 고유번호와 부서명, 지역명을 관리하고 있습니다. 가끔 고객이 바쁘지 않을 때에는 우편으로 주문을 하기도 하지만, 대개는 팩스나 전화로 주문을 합니다. 저희는 고객들이 주문한 후 즉각적으로 재고검색을 통해 주문접수에 대한 정보를 제공함으로써 고객의 신뢰를 확보할 계획입니다. 고객이 주문한 제품이 창고 어딘가에 있을 경우에는 반드시 바로 다음 날까지 배송을 하려고 합니다. 그리고 가능하다면, 모든 제품에 대한 재고량, 재주문 시점, 최대 재고량, 재고가 바닥난 이유, 그리고 제품이 다시 재고로 쌓이는 날짜, 창고명 등에 대한 정보도 관리합니다.

그리고, 제품이 운송이 될 때에, 우리의 자동 운송 시스템을 통해서 고객들에게 운송상태를 팩스로 보냅니다. 이것은 저희 부서의 업무는 아닙니다. 단지 저희 부서에서는 고객들에게 정확한 대금 정보를 제공하고 고객들의 대금결재의 신용을 확인하는 관리업무를 담당합니다. 그리고 고객들에 대한 일반적인 의견을 기록해 두기도 합니다.

우리는 고객들이 요청한 제품들이 재고에 있는지 확인한 후 만약 재고가 있다면, 고객에게 주문번호와 주문총량을 알려주어 주문을 진행 합니다. 만약, 제품의 재고가 없다면, 고객이 모두 다 주문할 것인지, 아니면 부분적으로 주문 할 것인지를 고객이 선택할 수 있게 합니다.

회계부서에서는 고객에 대한 정보 관리, 특히 새로운 고객에게 고유 번호를 지정하는 업무를 담당 합니다. 또한, 고객에 대한 정보 변경은 고객이 제품을 주문하고 대금 지불을 하는 경우 발송할 주소가 바뀌었을 때에 만 변경 가능합니다. 또한, 고객의 제품대금에 대한 모든 관리도 저희가 하는 것은 아닙니다. 그것은 모두 은행계좌에 의해 온라인으로 처리되기 때문에 회계부서에서 확인을 하게 됩니다.

다만, 회계부서에서는 판매에 대해 종업원들에게 일정한 커미션을 주기 위해서 종업원들의 커미션 비율을 관리하고 있습니다. 그래서, 회계부서에는 모든 사원들의 고유 번호나 last name을 알고 있어야 합니다. 어떤 경우에는 First Name 과 시스템 User ID, 입사일, 직급, 그리고 월 급여를 알 필요도 있습니다. 또한 사원들의 커미션 비율(10,12.5,15,17.5,20%)과 개인들에 관한 의견들을 관리하기도 합니다.

주문을 받는 담당 영업 사원들은 우리의 생산라인을 잘 숙지하고 있습니다. 마케팅팀과 활발한 교류를 통하여 새로운 제품에 대해 정보를 보유하도록 합니다. 무엇보다도, 고객들에게 보다 많은 답변을 할 수 있기 때문에 고객 만족을 향상시킬 수 있습니다. 이것은 몇몇의 선택적인 고객을 상대하고 특별 생산라인을 유지함으로써 가능합니다. 가끔은 제품의 요약된 설명, 제안된 가격, 판매 단위를 알아야만 할 때도 있습니다. 또한, 우리는 앞으로 필요하다면 우리 제품에 대한 긴 설명과 사진을 관리하고 제공하려고 합니다.

궁극적으로, 우리의 마지막 목표는 고객이 인터넷을 통해 우리 제품을 볼 수 있도록 해주고, 주문까지 할 수 있도록 하려고 합니다.



# 장표 수집

# 장표 #1

## 사원정보 현황

부서명 : 전체

사원번호	사원명/성	사용자 계정	입사일자	상관번호	직무	부서명	급여	커미션
1	Velasquez Ca	cvelasqu	03/03/90		President	Admin.	2,500	
2	Ngao LaDoris	Ingao	08/03/90	1	VP	Operat.	1,450	
3	Nagayama M.	mnagayam	17/06/91	1	VP	Sales	1,400	20
4	Quick-To-Se	mquickto	07/04/90	1	VP	Finance	1,450	

---

전체	4 명						6,800	
----	-----	--	--	--	--	--	-------	--

---

SUMMIT2 (주)

## #2

### 부서정보 현황

부서번호	부 서 명	지 역 명
10	Finance	North America
31	Sales	South America
41	Operations	Africa / Middle East
50	Administration	Asia

# #3-1

## 재고 현황

창고명 : 전체

제품번호	제품명	창고지역	재고 수량	재 주문 시점	최대재고 보유량	재고 없는 사유	재입고 될 날짜
10011	Bunny Boot	North America	650	625	1,100		
10012	Ace Ski Boot	South America	600	560	1,000		
10013	Pro Ski Boot	Asia	400	400	700		
10021	Bunny Ski Pole	Europe	500	425	740		

---

SUMMIT2 (주)

# #4

## 주 문 전 표

주문번호	2006-09-012345	담당사원	Magee		
고객명	Womansport (주)				
주문날짜	1992-08-31	선적날짜	1992-09-10	선적여부	Y
주문 총금액	601,100	지불방법	CREDIT		

항목번호	제 품 명	단 가	주문수량	선적수량	금 액
1	Bunny Boot	135	500	500	67,500
2	Pro Ski Boot	380	400	400	152,000
3	Bunny Ski Pole	14	500	500	7,000
4	Pro Ski Pole	36	400	400	14,400
5	Himalaya Bicycle	582	600	600	349,200
6	New Air Pump	20	450	450	9,000
7	Prostar 10Pd.Weight	8	250	250	2,000

SUMMIT2 (주)

# #6

## 거래명세서

주문 날짜	2006/09/10	사업자번호	212-16-93350
주문 번호	<b>2006-09-012345</b>	사업장 주소	서울시 강남구 삼성동 <b>13</b> 번지
고객명	Womansport (주)	업태	제조
주문 총금액	601,100	업종	스포츠용품 소매판매

항목번호	제품명	단가	주문수량	공급금액	세액
<b>1</b>	Bunny Boot	<b>135</b>	<b>500</b>	<b>67,500</b>	
<b>2</b>	Pro Ski Boot	<b>380</b>	<b>400</b>	<b>152,000</b>	
<b>3</b>	Bunny Ski Pole	<b>14</b>	<b>500</b>	<b>7,000</b>	
<b>4</b>	Pro Ski Pole	<b>36</b>	<b>400</b>	<b>14,400</b>	
<b>5</b>	Himalaya Bicycle	<b>582</b>	<b>600</b>	<b>349,200</b>	
<b>6</b>	New Air Pump	<b>20</b>	<b>450</b>	<b>9,000</b>	
<b>7</b>	Prostar 10Pd.Weight	<b>8</b>	<b>250</b>	<b>2,000</b>	

# 3

## 기본 개념적 데이터 모델링

# 모델과 모델링

## 모델(Model)

회화, 조각, 사진 등의 대상이 되는 인물 < 미술 >

작품의 소재가 되는 실제 인물 < 문학 >

성능, 디자인에 따라 구별되는 제품의 종류 < 산업 >

## 모델링(Modeling)

모델링(한글사전)

실체를 나타내는 일

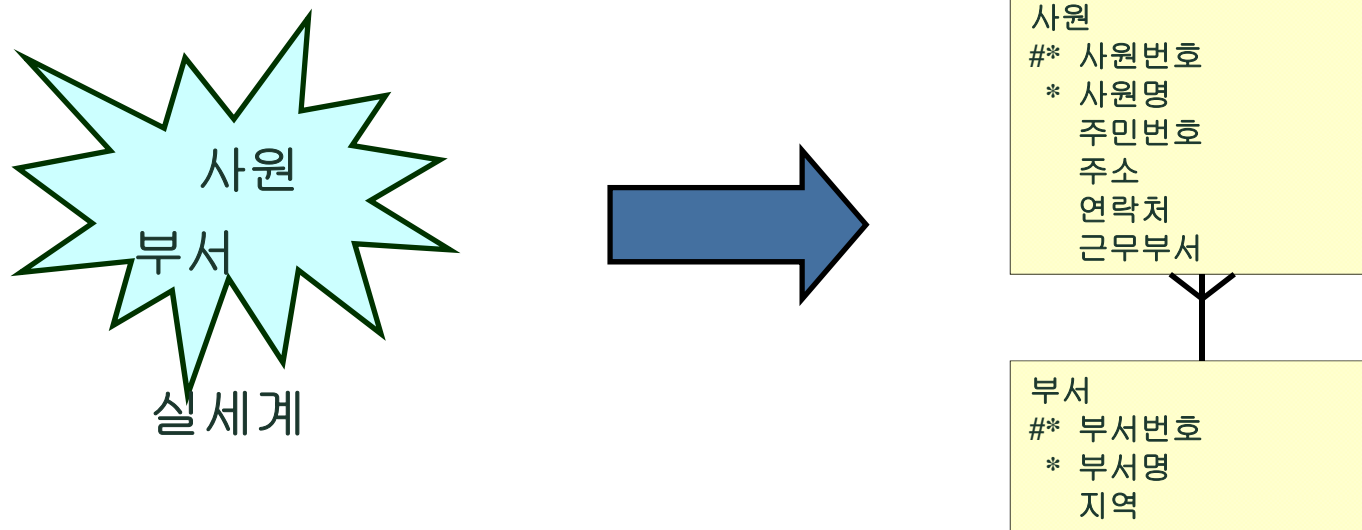
Modeling(영한사전)

모형화(模型化)

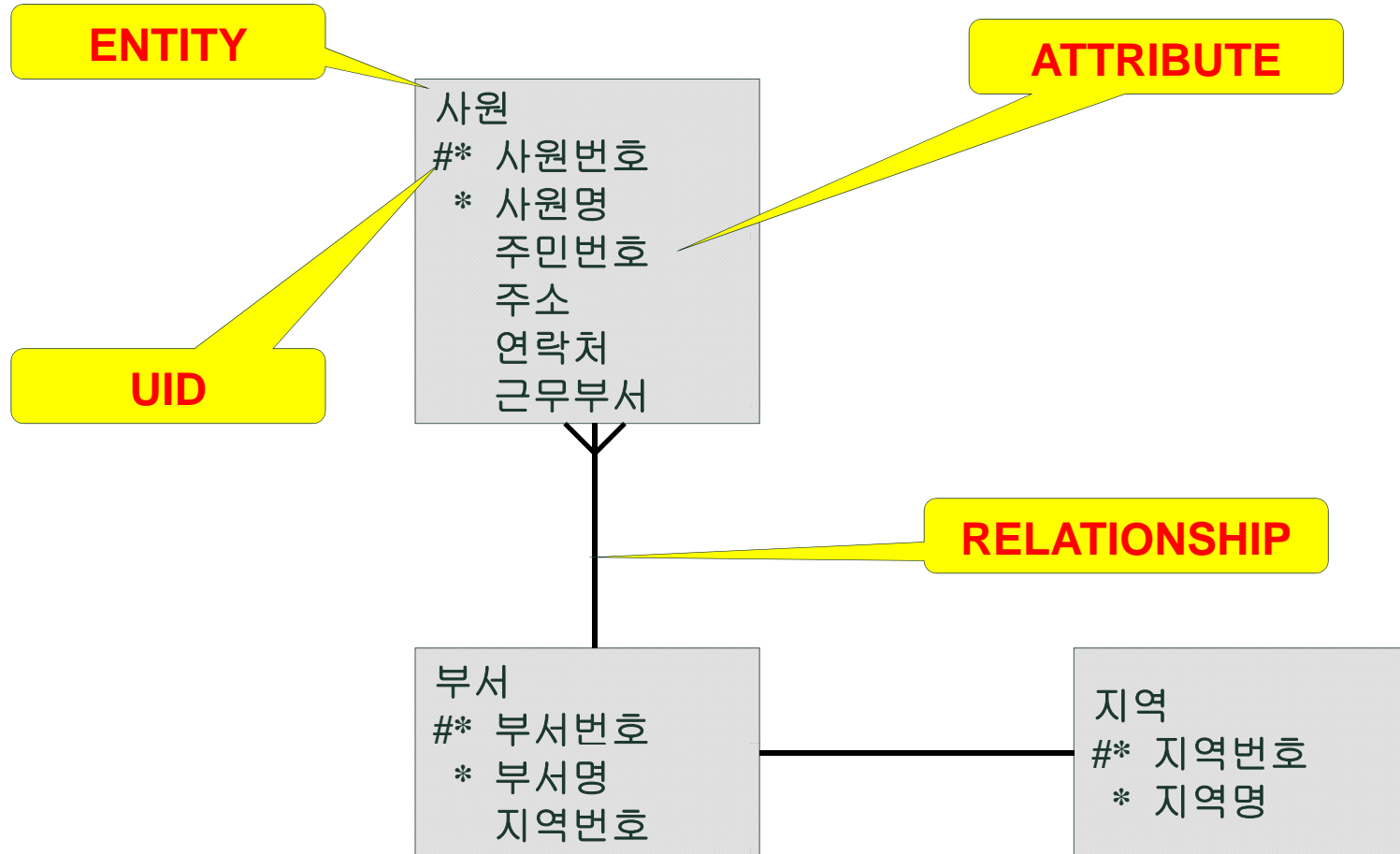


# 개념적 데이터 모델링

- . 개념적 데이터 모델링이란 **ISP(Information Strategy Facility)** 단계에서 이루어지며 **실세계의 현상을 알기 쉽고 체계적으로 모형화** 해 놓은 것이다.
- . 시스템적인 측면(**H/W, DBMS, O/S**)이 아닌 **실세계 그대로를 표현**해야 한다.
- . **프로세스와는 독립적**으로 사용자의 관점에서 인식하고 분석해야 한다.
- . 구체적인 설계 단계가 아니라 **개괄적인 의미를 표현하는 단계이다.**



# 용어 이해

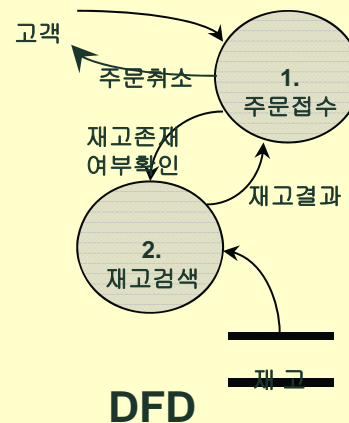


# Entity(실체)

- 1) 기업의 실세계에서 지속적인 관심을 가지고 **정보화를 해야 하는 대상을 Entity**라고 하며 물리적 모델링 단계에서 **테이블이 될 수 있는 후보이다.**  
(예) 사람, 사물, 위치, 장소, 개념, 활동, 사건 등
- 2) 영속적이며 식별 가능한 **데이터 저장 단위**이고 수집된 **장표들이 Entity**가 될 수 있다.
- 3) DFD 작성할 때 **Data-Store들이 Entity**가 될 수 있다.
- 4) 현행 업무 중 **마스터 파일, 기타 파일 등이 Entity**가 될 수 있다.

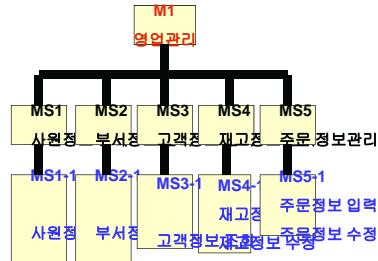
주문접수					
주문번호	2009-05-01/12345		주문사명	Maxim	
고객명	Wonsangpark				
주문일자	2009-05-12	주회번호	2009-05-12	주회금액	7
주문금액	400,100	주문금액		잔액	
주문번호	구분명	수량	단위	금액	총액
1	Bunny Book	100	권	100	10,000
2	Pro 3D Book	100	권	100	10,000
3	Bunny 3D Book	14	권	350	4,900
4	Pro 3D Book	35	권	400	14,000
5	Heavy 3D Book	100	권	400	40,000
6	Heavy Air Pump	20	개	450	9,000
7	Pro 3D Weight	8	개	350	2,800

장표



매출\_2010\_01.xls

# Entity 후보 수집대상



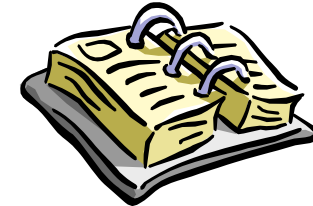
DFD/FHD



사용자 요구사항



시스템 Manual



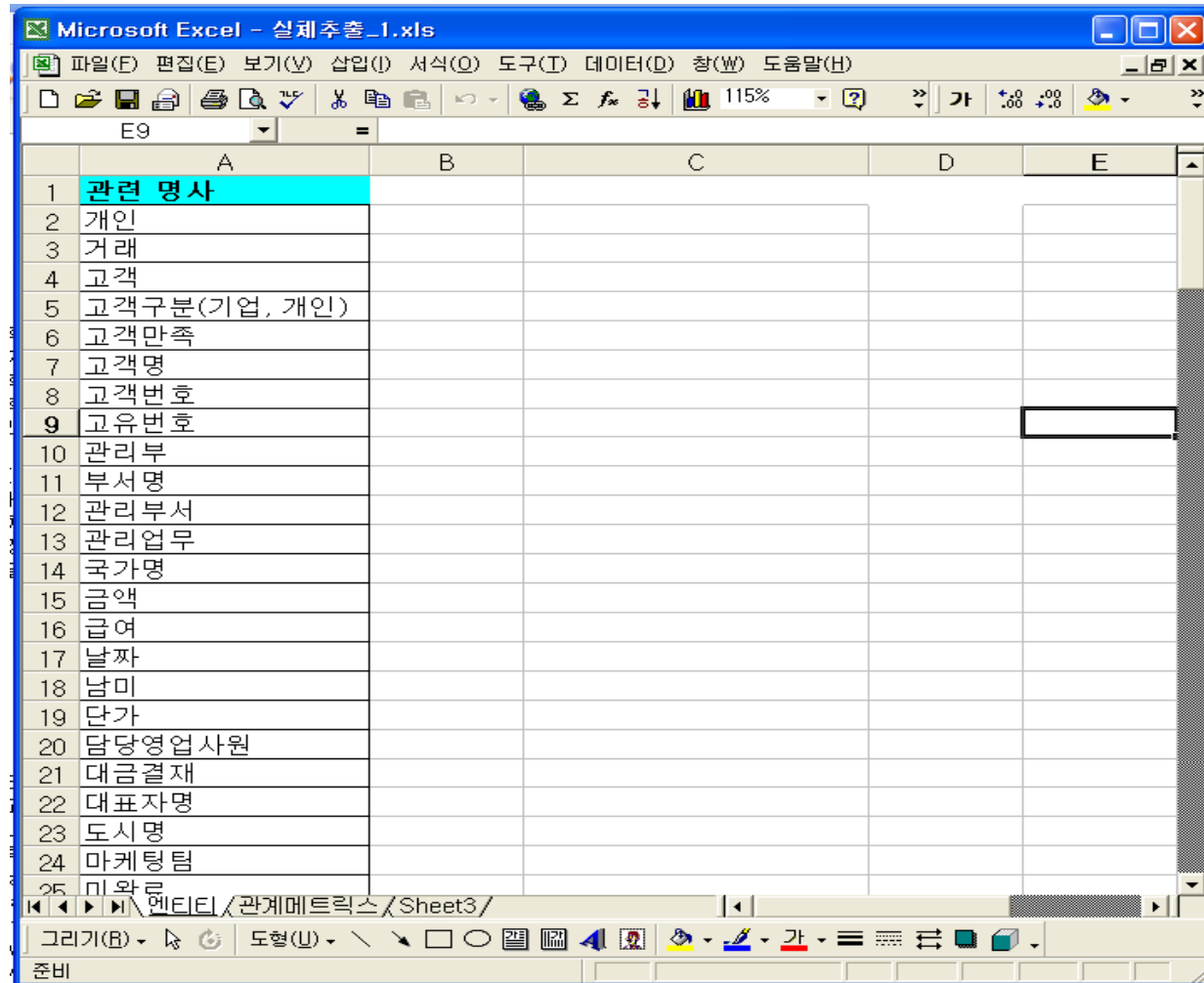
장표/보고서

주문

타시스템 Entity

# Entity의 추출

Step-1. Entity 후보 수집 대상으로부터 명사로 된 단어를 찾아라.



**Step-2.** 애매 모호하거나 확신이 없더라도 쉽게 버려서는 안되며  
명사적 단어에 대한 해석이 불 명확한 경우에는 데이터 항목 일  
람표를 참조하고 가장 적합한 엔티티에 배치해야 한다.

**Step-3.** 분석하고자 하는 비즈니스 범위 내에서 해석해야 한다.  
(개발 범위 밖의 명사라라도 개발 범위에서 자주 참조해야 한다  
면 일단 추출하라.)

**Step-4** 명사로 된 단어 중에 같은 의미로 사용되면서 다르게 표현되는 단어  
는 버려라. (이음동의어, 동음이의어)

부서	사원(종업원)	지역
고객	주문	주문항목
제품	재고	창고
<del>시원</del>	<del>생산라인</del>	<del>종업원</del>

**Step-5. Entity는 하나 이상의 Attribute로 구성 되어야 한다.**  
속성으로 표현되는 단어를 분류하고 어떤 Entity에 포함되는지 파악하라.  
**(Entity, Attribute, Attribute의 부가설명을 분류하라.)**

Microsoft Excel - 실체추출\_2.xls

파일(F) 편집(E) 보기(V) 삽입(I) 서식(O) 도구(T) 데이터(D) 창(W) 도움말(H)

C17

	A	B	C	D	E
1	관련 명사			Entity	
2	개인	제거	고객구분 Att의 부가설명	고객	고객명
3	거래	제거			고객구분(기업)
4	고객				고객번호
5	고객구분				담당영업사원
6	고객만족	제거			신용등급
7	고객명				전화번호
8	고객번호				E-MAIL주소
9	고유번호	애매모함			FAX번호
10	관리부	제거	부서명 Att의 부가설명		도시명
11	부서명				주명
12	관리부서	제거	부서명 Att의 부가설명		은행계좌
13	관리업무				사업자등록번호
14	국가명				대표자명
15	금액				업종
16	급여				업태
17	날짜				주민번호
18	남미				지역(참고)
19	단가				
20	담당영업사원				
21	대금결제				
22	대표자명			사원	사원번호
23	도시명				사원명
24	마케팅팀				사원성
25	미완료				User id

Entity / 관계메트릭스 / Sheet3 /

그리기(B) 도형(U)

준비

# Entity의 결정

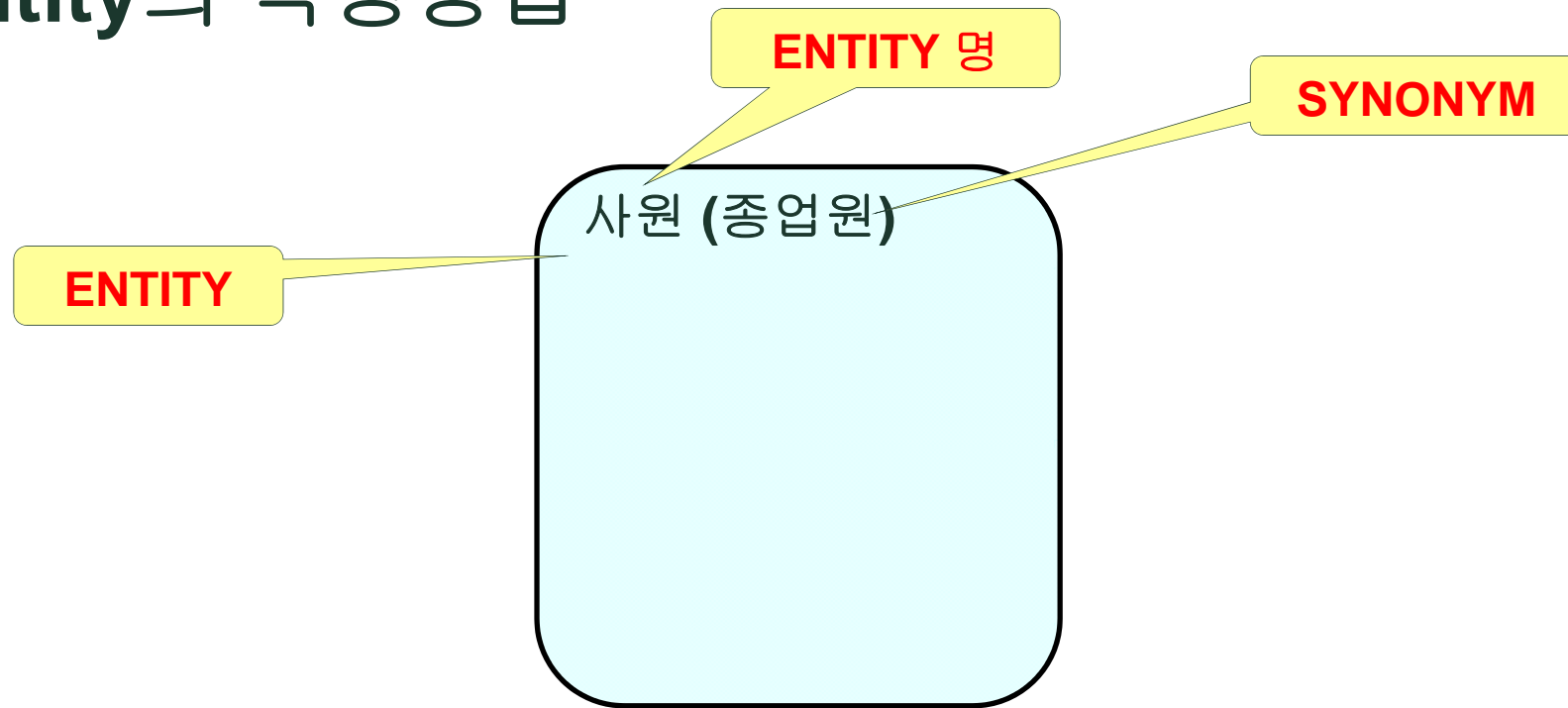
기본 개념적 모델링 단계는 **조감도를 만드는 단계이다.**

처음부터 **Entity**를 **세분화하여 분리하지 마라.** 지금 추출된 **Entity**들은 상세 개념적 모델링 단계에서 제거될 수도 있고 통합될 수도 있기 때문이다. **(조감도는 설계도가 아니다.)**





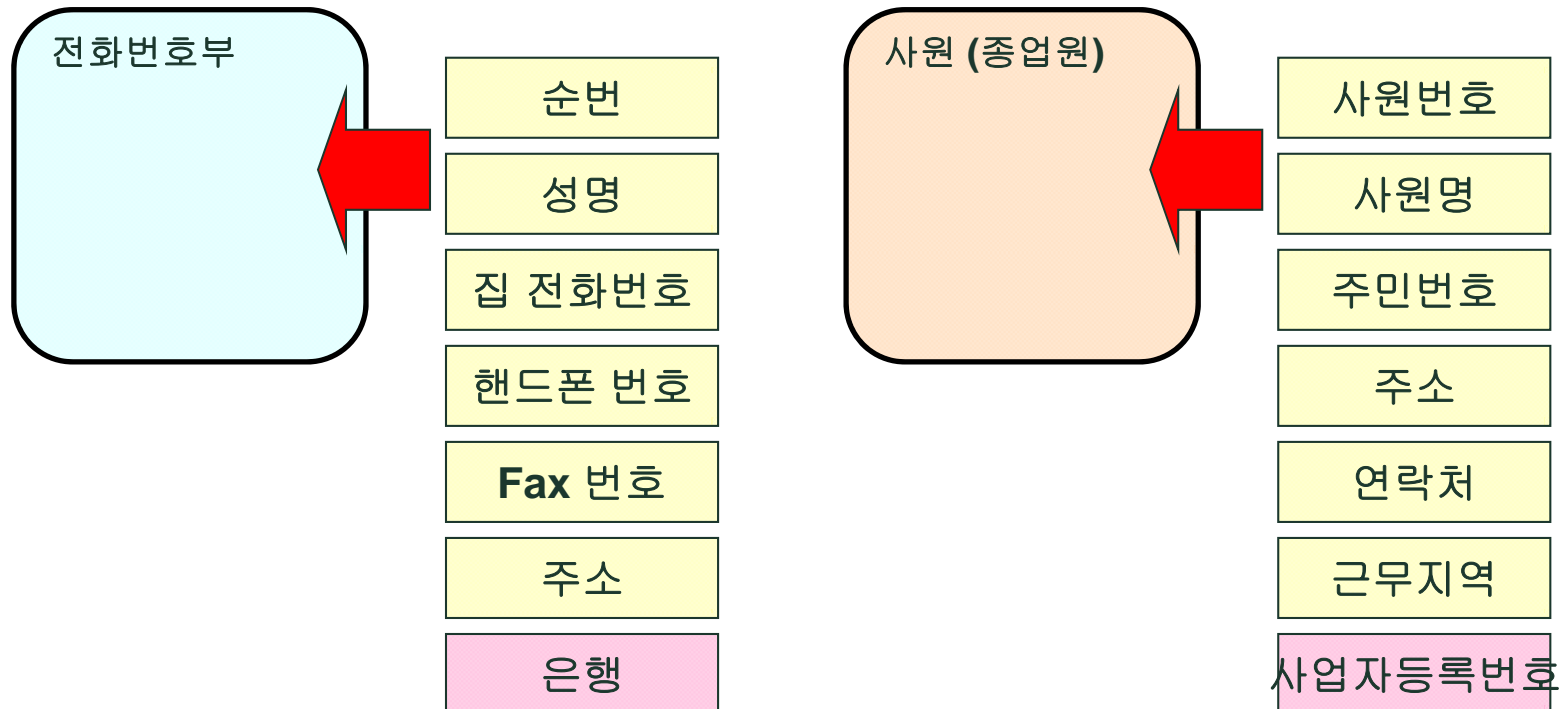
# Entity의 작성방법



- 1) Entity는 모서리가 둥근 직사각형으로 표현합니다.
- 2) Entity 이름은 단수형, 유일한 이름, 대문자로 표시합니다.
- 3) 동의어는 괄호로 표시한 후 기술합니다.
- 4) 반드시, 하나 이상의 Attribute를 가져야 합니다.

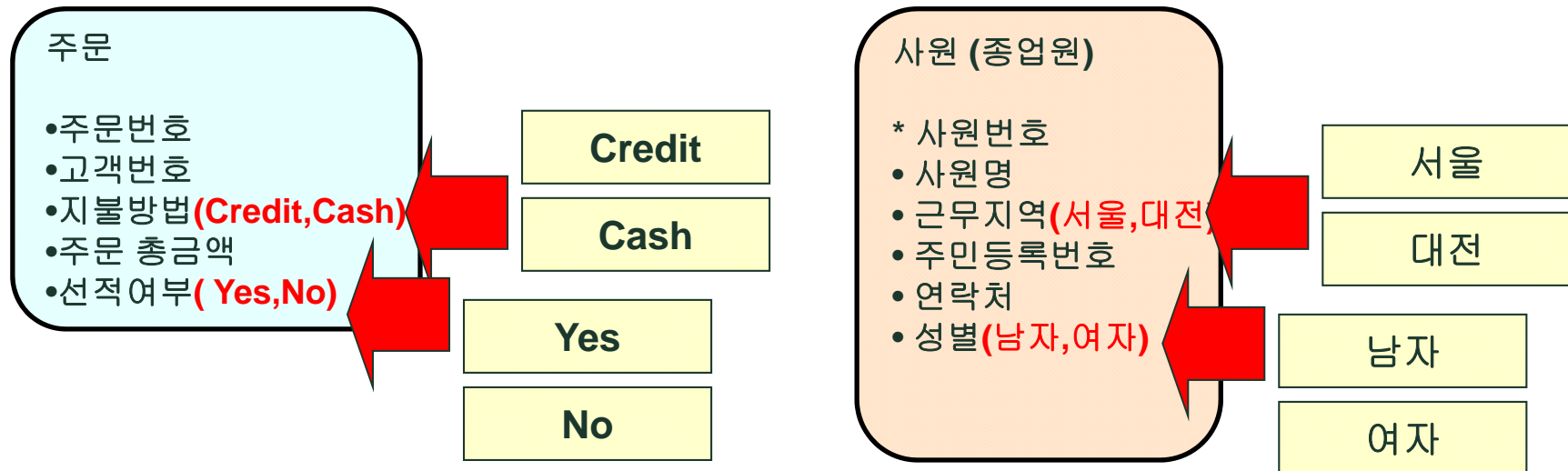
# Attribute(속성)

- 1) 하나의 **Entity**는 **하나 이상의 Attribute**로 구성되며  
하나의 **Attribute**는 더 이상의 구성 요소를 갖지 않는 명사이다.
- 2) **Entity**가 누락되어 배치할 수 없는 **Attribute**도 함부로 버려서는 안되며  
일단 가장 적합하다고 판단되는 **Entity**에 배치해야 한다.



# Attribute의 부가설명

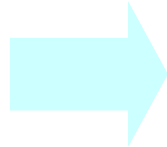
- 1) 추출된 명사적 단어들 중에 **Entity**도 아니고, **Attribute**도 아닌 단어는 **Attribute**를 구체적으로 표현하는 **데이터 값**으로 **Attribute**의 부가설명이라고 한다.
- 2) **Attribute**의 부가 설명은 **괄호로 표현**한다,



# Attribute의 결정

- 1) 데이터 항목의 **최소 단위**로써 업무적 성격에 따라 **독자적인 성질**을 가질 수 있어야 한다.

성 명
주 종 면
홍 길 동

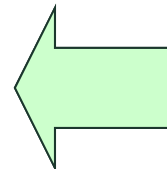


성	명
주	종 면
홍	길 동

Last\_Name = '주'

Substr>Last\_Name1,2) Like '주%'

매출일자
2010-05-01
2010-05-20



매출년	월	일
2010	05	01
2010	05	20

YYYYMMDD >= '201005%' and

YYYYMMDD <= '201006%'

YYYY = '2010'

MM between 06 and 12

## 2) 추출(Derived) 값은 속성이 될 수 없으므로 버려라.

- 건수, 합계, 최대값, 최소값, 평균, 계산된 값 등
- 추출 값은 중복되고 일관성 없는 값을 유도할 수 있다.

주문

항목	제품번호	수 량	단 가	금 액
01	1234	20	125	2,500
02	4567	35	230	8,050
총금액				10,550

사원

사원번호	주민번호	성별
1101	650107-1...	남
1102	680217-2	여



# Attribute의 작성방법

\* : Mandatory  
o : Optional

**ATTRIBUTE**

사원 (종업원)

- \* 사원번호
- \* 사원명
- o 주민번호
- o 주소
- o 연락처
- o 핸드폰 번호
- o 근무지역(서울,대전)

**ATTRIBUTE의 부가설명**

1) Entity 이름 다음에 **순차적**으로 작성

2) 반드시 **필요한 속성은 '\*'**로 표시, **존재하지 않을 수도 있는 경우에는 'o'**로 표시한다.

# Unique Identifier(식별자)

- 1) 하나의 **ENTITY**는 반드시 하나의 **UID**를 가져야 한다.  
각 인스턴스는 다른 인스턴스와 구별되어야 한다.
- 2) 해당 속성은 반드시 **Mandatory**이어야 하며 **유일(Unique)**해야 한다.
- 3) 하나의 **ENTITY**를 대표하는 속성으로 여러 개의 속성이 하나의 **UID**가 될 수 있다.
- 4) **UID**의 선정은 최소한의 속성으로 결정해야 한다.

# Key의 유형

Candidate Key

Unique Identifier

Alternate Key

Artificial Key

UID  
(Candidate – Key)

Alternate – Key  
(Candidate – Key)

사원 (종업원)

# \* 사원번호  
\* 사원명  
○ 주민번호  
○ 주소  
○ 연락처

지역

# \* 지역번호  
\* 지역명

Artificial– Key



# UID의 결정

주 문 전 표

주문번호	2006-09-012345	담당사원	Magee		
고객명	Womansport (주)				
주문날짜	2006-09-12	선적날짜	2006-09-12	선적여부	Y
주문 총금액	601,100	지불방법	CREDIT		

항목번호	제 품 명	단 가	주문수량	선적수량	금 액
1	Bunny Boot	135	500	500	67,500
2	Pro Ski Boot	380	400	400	152,000
3	Bunny Ski Pole	14	500	500	7,000
4	Pro Ski Pole	36	400	400	14,400
5	Himalaya Bicycle	582	600	600	349,200
6	New Air Pump	20	450	450	9,000
7	Prostar 10Pd.Weight	8	250	250	2,000

## 주문 Entity

SUMMIT2 (주)

주문번호	고객명	담당사원	...	선적유무	항번	제품번호	제품명	단가	주문수량
2006-09-012345	Womansport (주)	Magee	...	Y	1	1111	BOOT	135	500
2006-09-012345	Womansport (주)	Magee	...	Y	2	1112	BAT	380	400
2006-09-012345	Womansport (주)	Magee	...	Y	3	1113	GLOVE	14	500
2006-09-012345	Womansport (주)	Magee	...	Y	4	1114	Man's	36	400
2007-12-000001	Womansport (주)	Magee	...	Y	1	1111	BOOT	135	500
2007-12-000001	Womansport (주)	Magee	...	Y	2	1112	BAT	120	10
2007-12-000001	Womansport (주)	Magee	...	Y	3	1113	GLOVE	75	300

# 인조 **UID**

1) 추출된 **UID**의 길이가 너무 길어 이해하기 힘든 경우

편리성, 단순성을 확보하기 위해 인조 **UID**를 사용할 수 있다.

2) 유일한 식별자를 만들기 위해 인조 **UID**를 사용하며

인조 **UID**의 자릿수마다 나름대로의 의미를 부여해도 좋다.

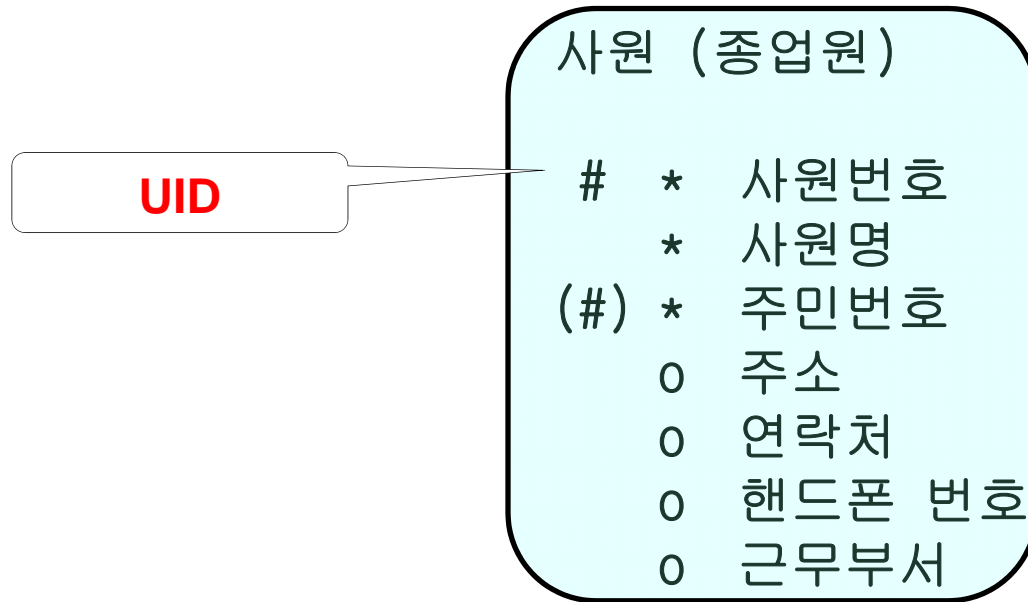
(예) 사원번호의 첫 번째 자리는 지역, 두 번째 자리는 직무,  
세 번째 자리는 성별 등으로 구분할 수 있음.)

3) 누구나 인정할 수 있고 객관적 단어를 인조 **Key**로 활용  
하는 것이 좋다.

(예) 사업자등록번호, 주민번호, 운전면허번호

# UID의 작성방법

# : UID (Primary-Key)  
(#) : 보조 UID (Alternate-Key)

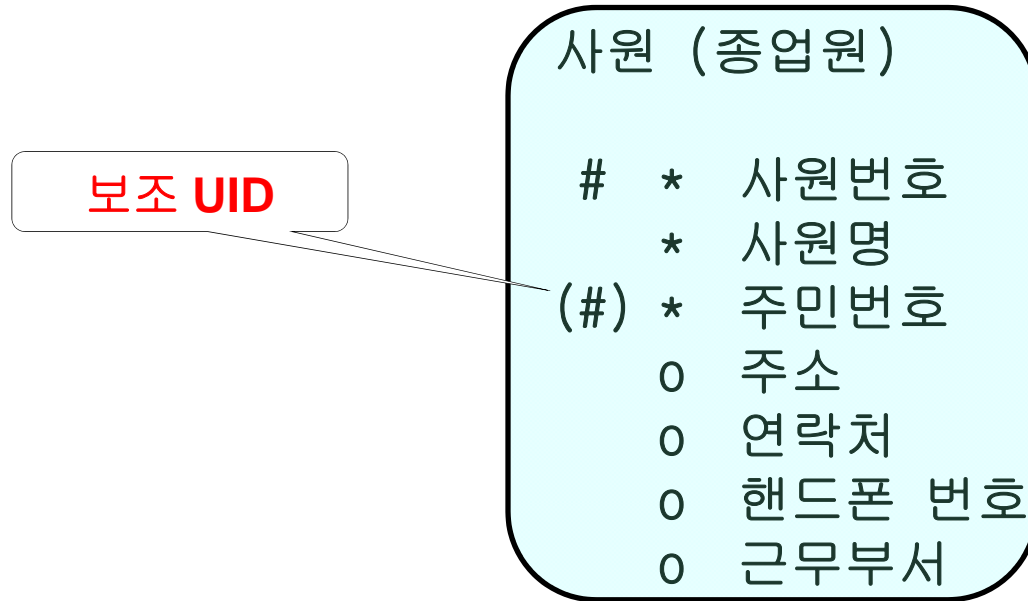


1) Entity 를 대표하는 속성 (**Mandatory, Unique**)

2) UID는 하나의 속성 또는 여러 개의 속성으로 구성된다.

# 보조 UID의 작성방법

# : UID  
(#1): 보조 UID-1  
(#2): 보조 UID-2

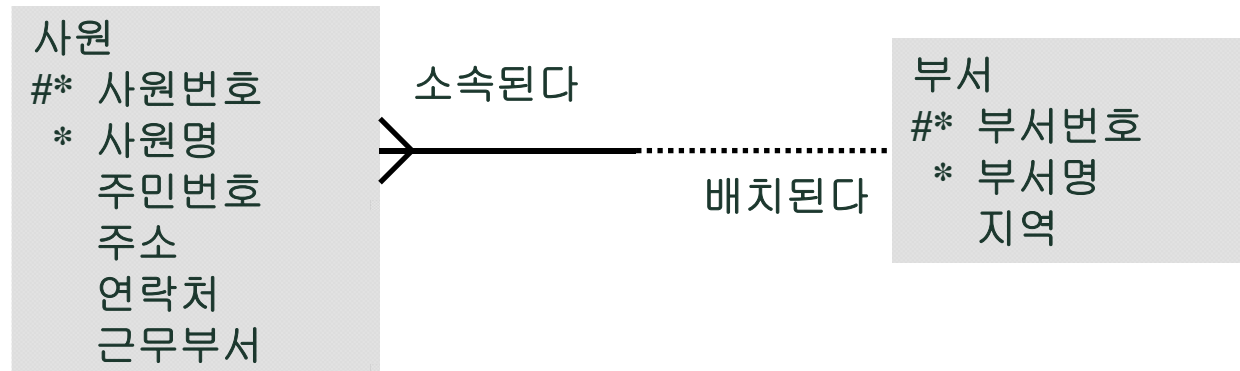


1) UID를 대신하는 보조 식별자(Unique)

2) 반드시 필요한 것은 아니며, UID보다 보조 UID가 유리할 수 있는 경우에 만 사용할 수 있다.

# Relationship(관계)

1) 각 **ENTITY** 간의 **업무적 상관관계**를 **도식화** 한 것.



2) 관계를 표현할 때 **관계의 명칭, 선택사양, 관계형태**를 표시한 것.

(Entity 간의 관계형태)

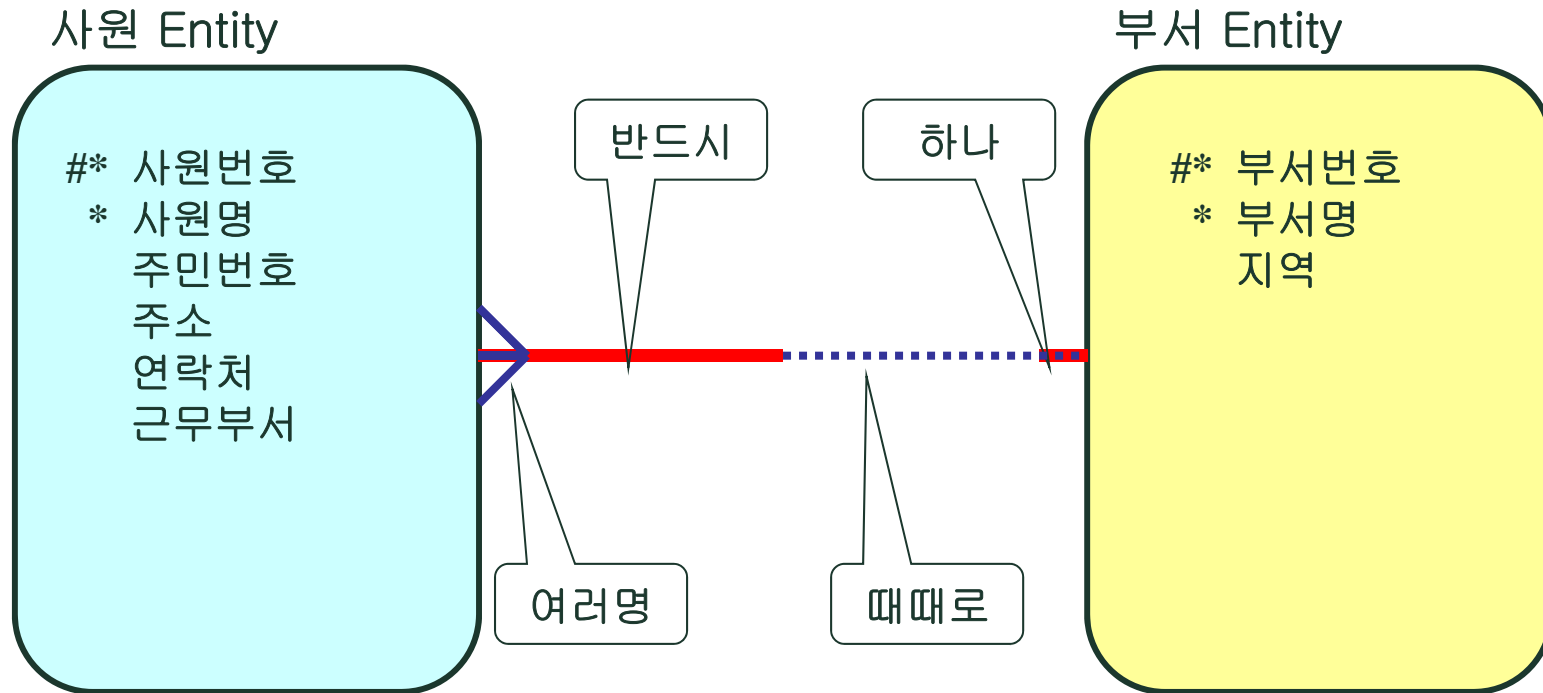
- 1 : N 관계
- N : M 관계
- 1 : 1 관계

(관계조건)

- 필수사양(Mandatory) —————
- 선택사양(Optional) .....

# Relationship

사원과 부서 관계  한명의 사원은 **하나**의 부서에 **반드시** 근무한다.



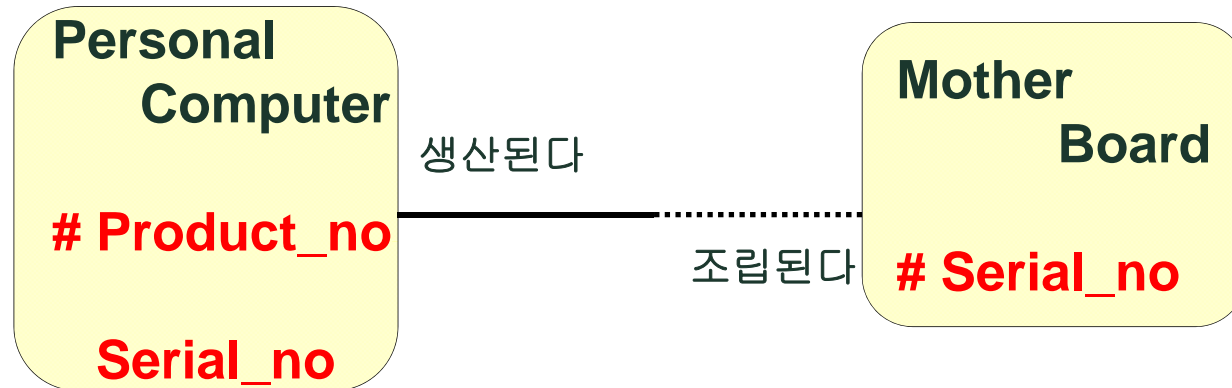
하나의 부서에는 **여러명**의 사원들이 **때때로** 근무한다.  부서와 사원 관계

# Relationship의 조건

- 1) 실체와 실체 간에 **중복되는 속성**을 가졌으면 **RELATION**이 될 수 있다.
- 2) 실세계에서 사용되는 **동사적 단어**들이 **RELATION**이 될 수 있다.
- 3) **프로세스에 대한 명확한 이해**를 해야 만 **Relationship**을 설정할 수 있다.

Relation	Relation
사원은 부서에 근무한다.	주문에 대한 주문항목은 구성된다.
창고는 담당영업사원에 의해 관리된다.	제품에 대한 재고를 저장한다.
고객은 주문을 한다.	지역마다 창고를 배치한다.
담당 영업사원은 고객을 관리한다.	재고는 창고별로 저장된다.

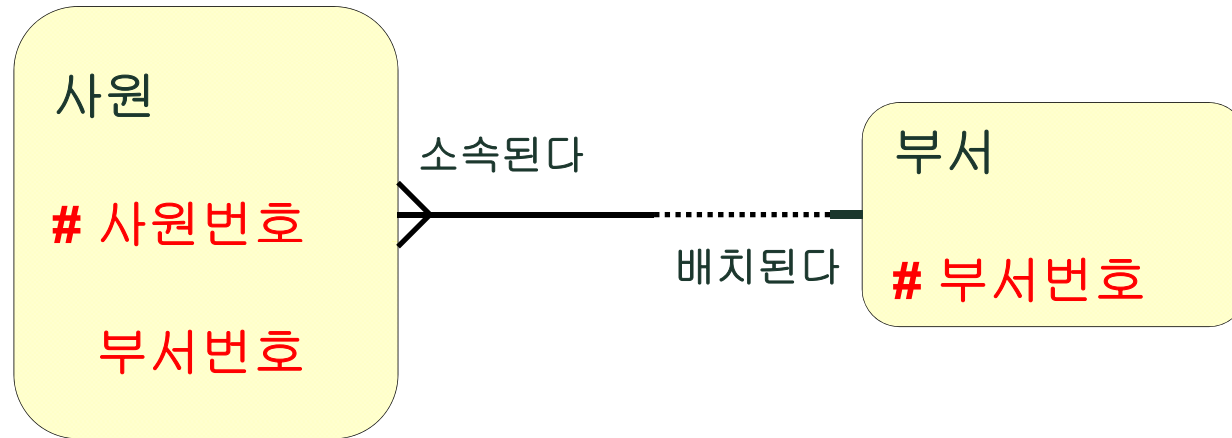
# Degree의 유형-1




- 1) 1:1의 관계는 실선(——) 또는 점선 (.....)으로 표시하고 상대 **ENTITY** 하나의 실체에 대해 해당 **ENTITY**에도 하나의 실체가 존재하는 경우 표시합니다.
- 2) 실세계에서는 드물게 나타나는 형태입니다.

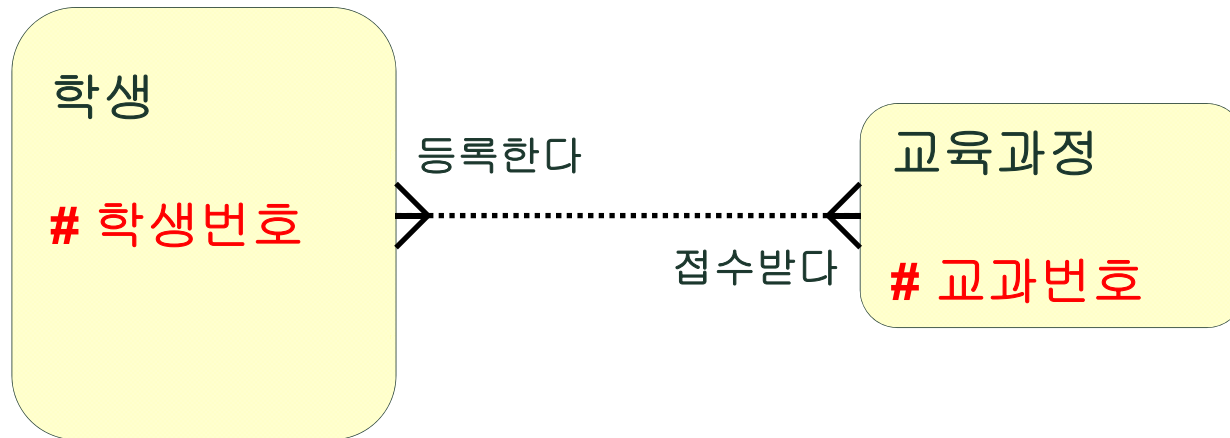


## Degree의 유형-2



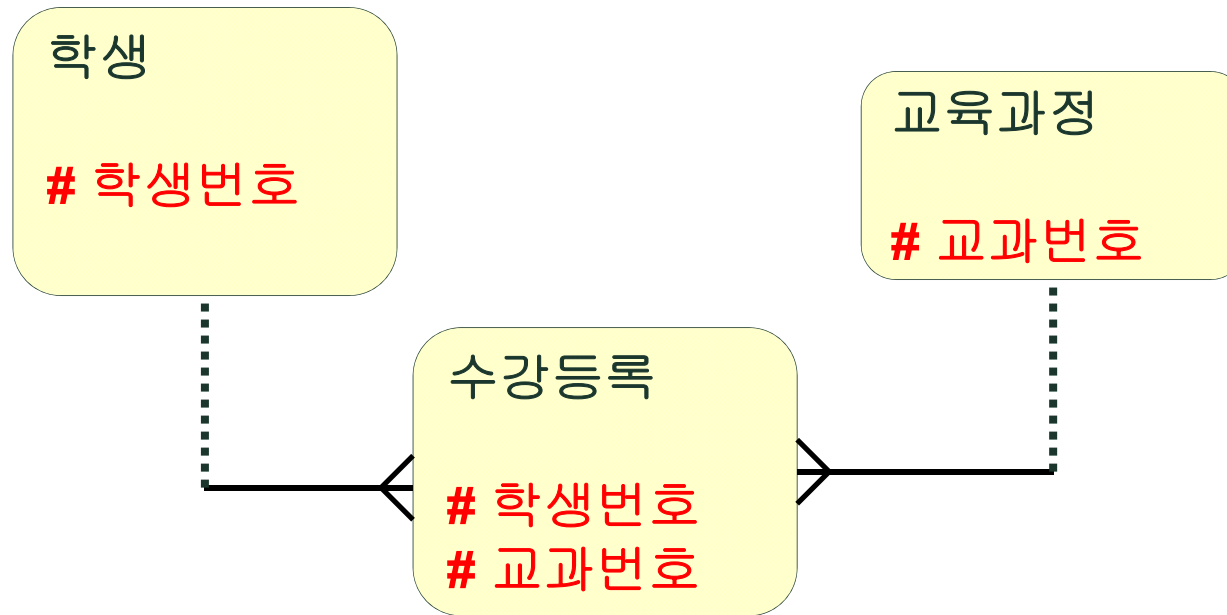
- 1) N:1의 관계는  으로 표시하고 상대 **ENTITY** 하나의 실체에 대해 해당 **ENTITY**에는 여러 개의 실체가 존재하는 경우 표시합니다.
- 2) 실세계에서 가장 흔하게 나타나는 형태입니다.

## Degree의 유형-3



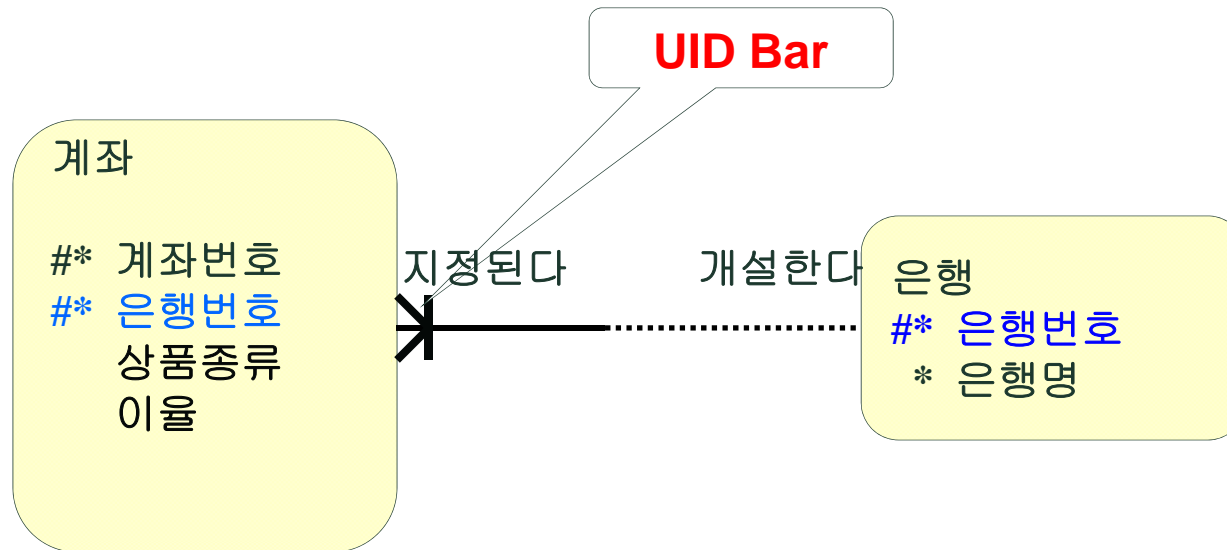
- 1) **N:M**의 관계는  $\text{>---<}$  으로 표시하고 상대 **ENTITY** 여러 개의 실체에 대해 해당 **ENTITY**에도 여러 개의 실체가 존재하는 경우 표시합니다.(**Conditional**은 반드시 점선으로 나타낼 수 밖에 없다)
- 2) 개념적 모델링 단계에서는 나타날 수 있는 관계형태이지만 논리적 모델링 단계에서는 존재할 수 없는 형태입니다.
- 3) 비즈니스-룰의 잘못된 해석 또는 **Entity**의 누락으로 인한 잘못된 관계 해석이 원인인 경우가 대부분이다.

# N:M 관계의 해소



- 1) 관계형 데이터베이스에서는 **N:M** 관계를 위한 데이터 구조를 제공하지 않기 때문에 반드시 해소 시켜야 합니다.
- 2) 비즈니스-룰의 잘못된 해석 또는 **Entity**의 누락으로 인한 잘못된 관계이지만 다음 단계인 상세 개념적 데이터 모델링 단계에서 재 해석 및 엔티티의 추가 분석을 통해 자연스럽게 해소될 수도 있습니다.

# UID Bar

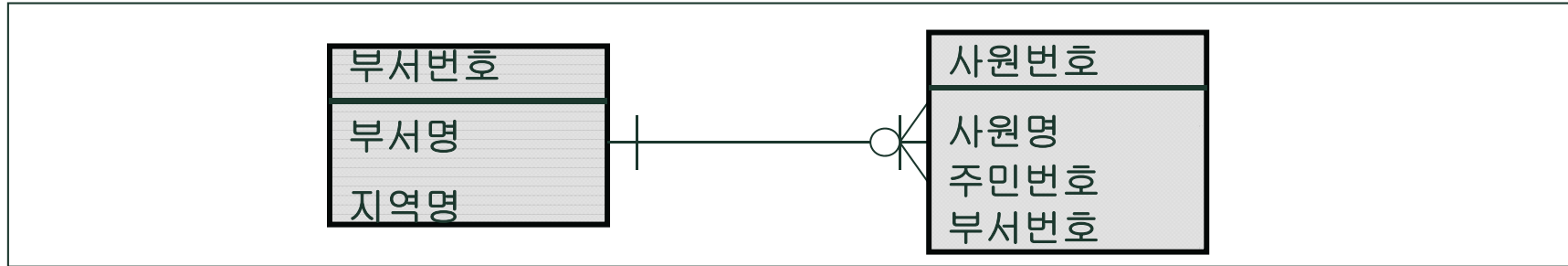


. **UID Bar**는 **UID**에 **Relationship**이 포함되어 있는 것을 의미합니다.

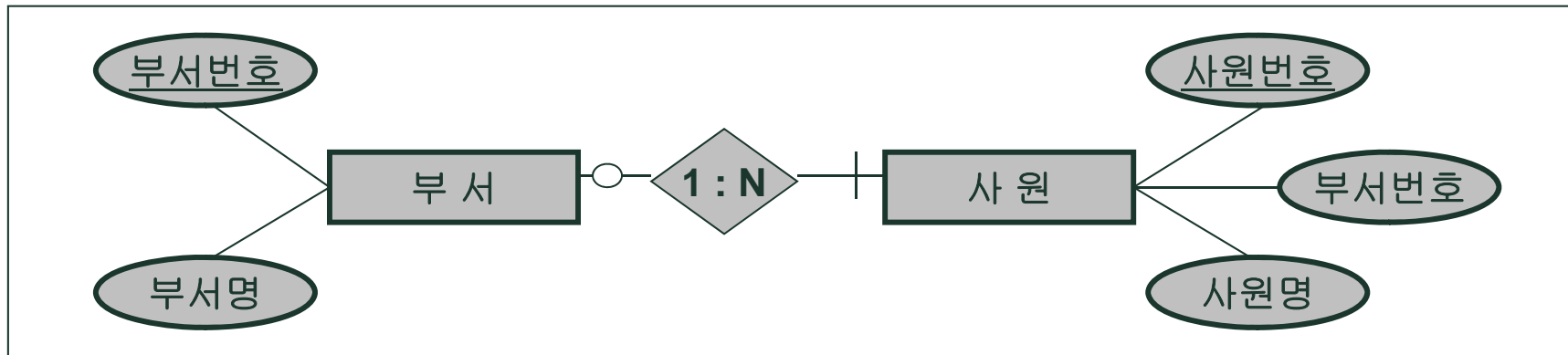
- 은행 **Entity**의 식별자를 계좌 **Entity**의 식별자로 참조하는 상속 관계
- 여러 개의 속성으로 조합된 **UID**로 반드시 **Mandatory**이어야 합니다.

# 기타 Notation

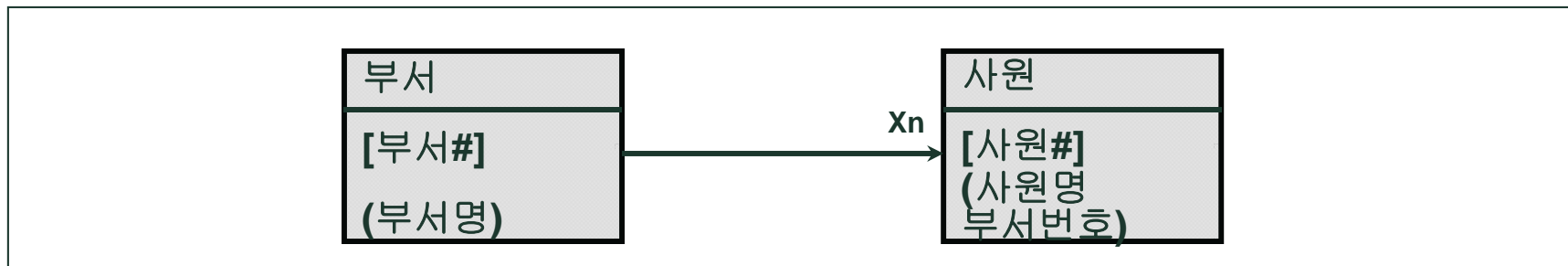
## James Martin의 표기법



## Chen의 표기법



## 프바키의 표기법



# Barker와 James Martin 표기법의 구별

표기 방법	Richard Barker	James Martin	설 명
실 체 속 성 식별자	<div>주문</div> <div>#*주문번호 고객번호 선적유무 선적날짜</div> <div>주문상세</div> <div>#*주문번호 #*항목번호 제품번호 주문수량</div>	<div>주문</div> <div>주문번호(PK) 고객번호 선적유무 선적날짜</div> <div>주문상세</div> <div>주문번호(PK,FK) 항목번호 제품번호 주문수량</div>	<p>1) Barker의 표기법에서는 소프트 Box를 실체라고 표현하지만 James Martin은 직사각형의 Box를 실체라고 한다. 하지만, Main 실체와의 관계에 따라 소프트 Box로 표현되기도 한다.</p> <p>2) Barker는 # 기호를 통해 식별자를 구분하는 반면 James Martin은 직사각형의 도형 상단에 바(Bar)로 식별자를 구분한다.</p>

# 관계 표기법의 구별

관계 표기 방법	Richard Barker	James Martin	설 명
Identifying (식별 관계)			0,1, 그 이상의 개체를 허용
			1, 그 이상의 개체를 허용
			0, 또는 1개체를 허용
			1 개체만 허용
Non-Identifying (비식별 관계)			0,1, 그 이상의 개체를 허용
			1, 그 이상의 개체를 허용
			0, 또는 1개체를 허용
			1 개체만 허용

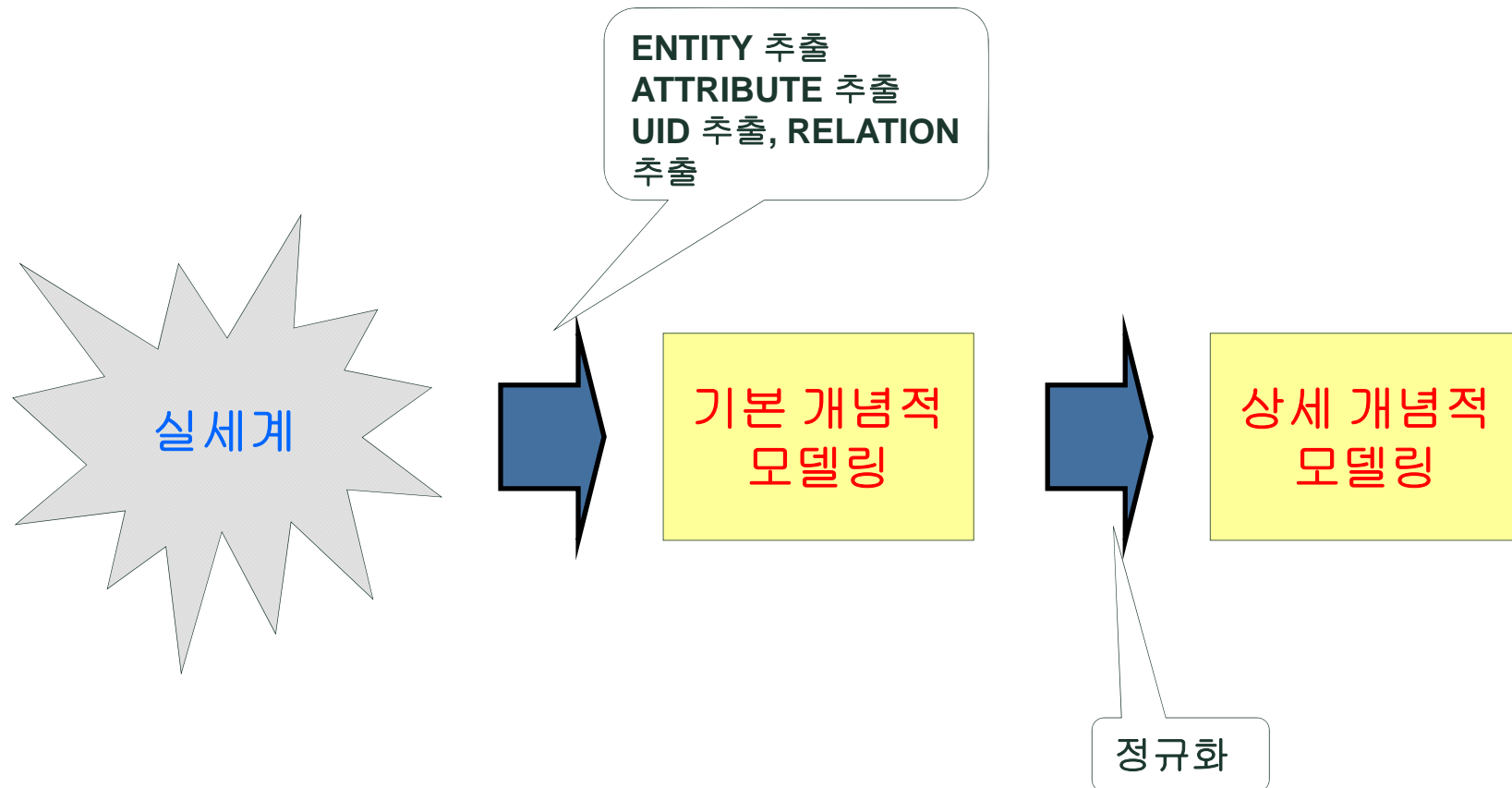
# 4

## 상세 개념적 데이터 모델링



# 상세 개념적 모델링

- 기본 개념적 모델링은 실 세계의 현상을 그대로 표현한 것이라면 상세 개념적 모델링은 보다 명확하게 **ENTITY**를 표현하고 **ATTRIBUTE**를 제거 또는 추가하는 과정입니다.



# 정규화를 하는 이유

1) 데이터의 중복성을 제거할 수 있다.

2) 데이터 모형을 단순화 시킬 수 있다.

3) **ATTRIBUTE**의 배열 상태를 검증해 볼 수 있다.

4) **ENTITY, ATTRIBUTE**의 누락여부를 검증해 볼 수 있다.

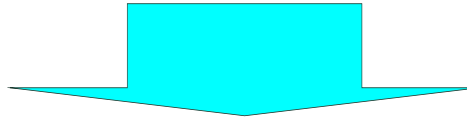
5) 데이터 모형의 안정성을 유지시킬 수 있다.

# 정규화

- . 정규화(**Nomalization**)란 개념적 모델링 과정에서 정의된 각 **ENTITY**에서 발생하는 데이터의 중복성을 제거하고 새로운 **ENTITY**를 창출해내는 과정을 의미한다.

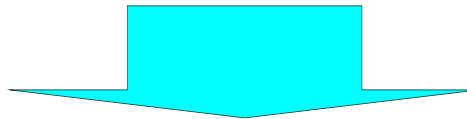
## 제 1 정규화

**Entity** 내의 모든 속성은 반드시 하나의 값을 가져야 한다.



## 제 2 정규화

**Entity** 내의 모든 속성은 반드시 **UID**에 종속되어야 한다.



## 제 3 정규화

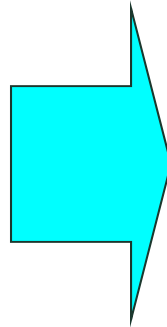
**Entity** 내의 식별자를 제외한 모든 속성은 종속관계를 가질 수 없다.

# 제 1 정규화

**Entity** 내의 모든 속성은 반드시 하나의 값을 가져야 한다. (중복의 제거)

주문

#\* 주문번호  
#\* 항목번호  
\* 고객명  
담당사원명  
주문일자  
선적일자  
주문합계금액  
지불방법  
선적유무  
제품번호  
\* 제품명  
주문단가  
주문수량  
선적수량



주문

#\* 주문번호  
#\* 항목번호  
제품번호  
\* 제품명  
주문단가  
주문수량  
선적수량

주문 공통

#\* 주문번호  
\* 고객명  
주문일자  
담당사원명  
선적일자  
주문합계금액  
지불방법  
선적유무

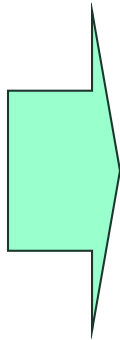
주문번호	항번	고객명	담당사원	...	선적유무	제품번호	제품명	단가	주문수량
2006-09-012345	1	Womansport (주)	Magee	...	Y	1111	BOOT	135	500
2006-09-012345	2	Womansport (주)	Magee	...	Y	1112	BAT	250	150
2006-09-012345	3	Womansport (주)	Magee	...	Y	1113	GLOVE	75	3400
2006-09-012345	4	Womansport (주)	Magee	...	Y	1114	Man's	120	40
2007-12-000001	1	Womansport (주)	Magee	...	Y	1111	BOOT	135	500
2007-12-000001	2	Womansport (주)	Magee	...	Y	1112	BAT	120	10
2007-12-000001	3	Womansport (주)	Magee	...	Y	1113	GLOVE	75	300

## 제 2 정규화

**Entity** 내의 모든 속성은 반드시 **UID**에 종속 되어야 한다.  
(부분 함수적 종속관계의 제거)

### 재고

#\* 창고번호  
#\* 제품번호  
창고명  
재고수량  
재주문시점수량  
최대보유수량  
재고가바닥난이유  
재입고날짜



### 재고

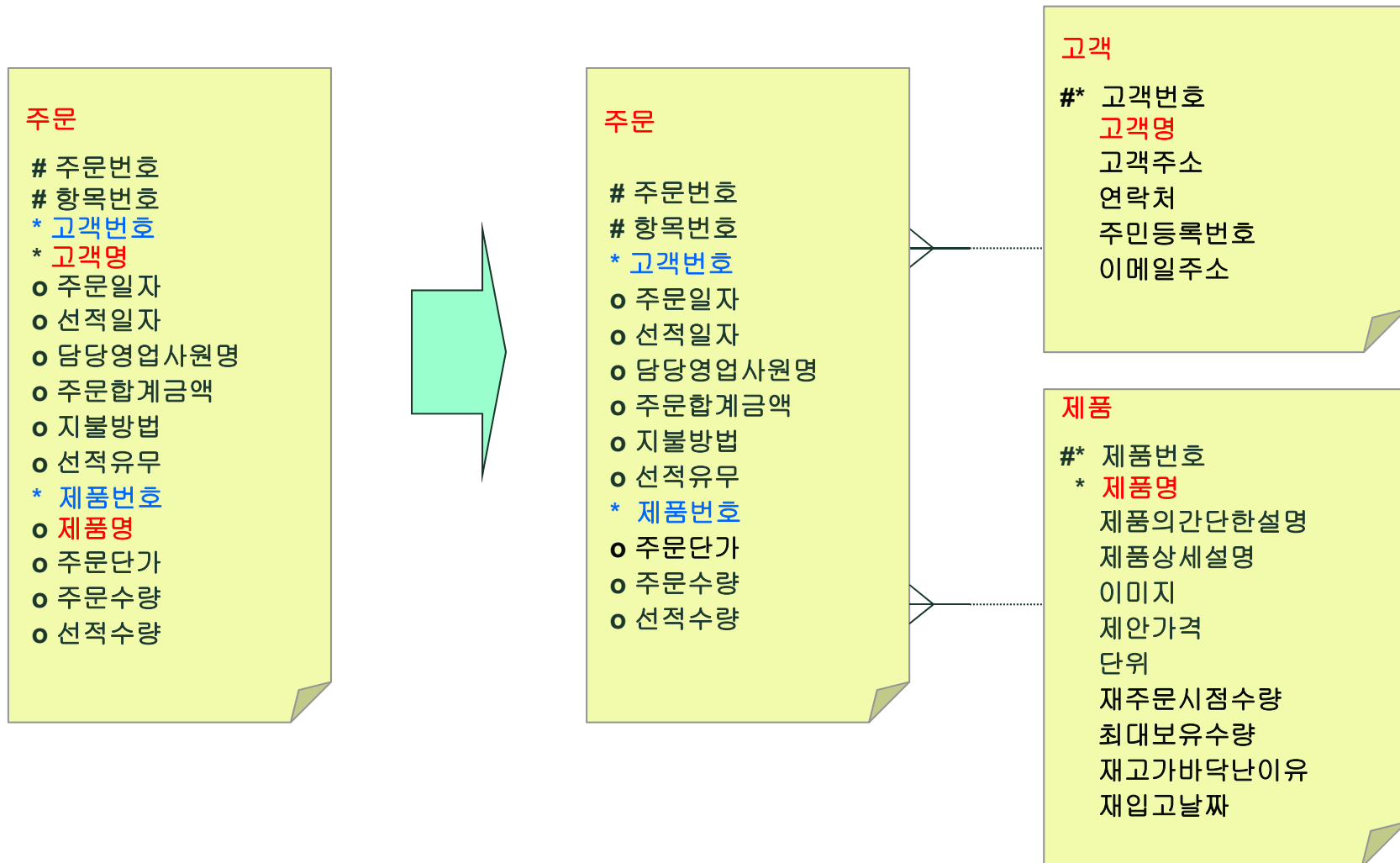
#\* 창고번호  
#\* 제품번호  
재고수량  
재주문시점수량  
최대보유수량  
재고가바닥난이유  
재입고날짜

### 창고

#\* 창고번호  
창고명  
지역번호  
주소  
도시명  
주명  
국가명  
우편번호  
전화번호  
관리자사원번호

# 제 3 정규화

**Entity** 내의 식별자를 제외한 모든 속성은 종속관계를 가질 수 없다.  
(이행 함수적 종속관계의 제거)



# 정규화의 장점과 단점

## 정규화 정도를 높은 경우

- 1) 유연한 데이터를 구축할 수 있다.
- 2) 데이터의 정확성이 높아진다.
- 3) 물리적 접근이 복잡해 진다.
- 4) 길이가 짧은 데이터가 생긴다.

## 정규화 정도를 낮은 경우

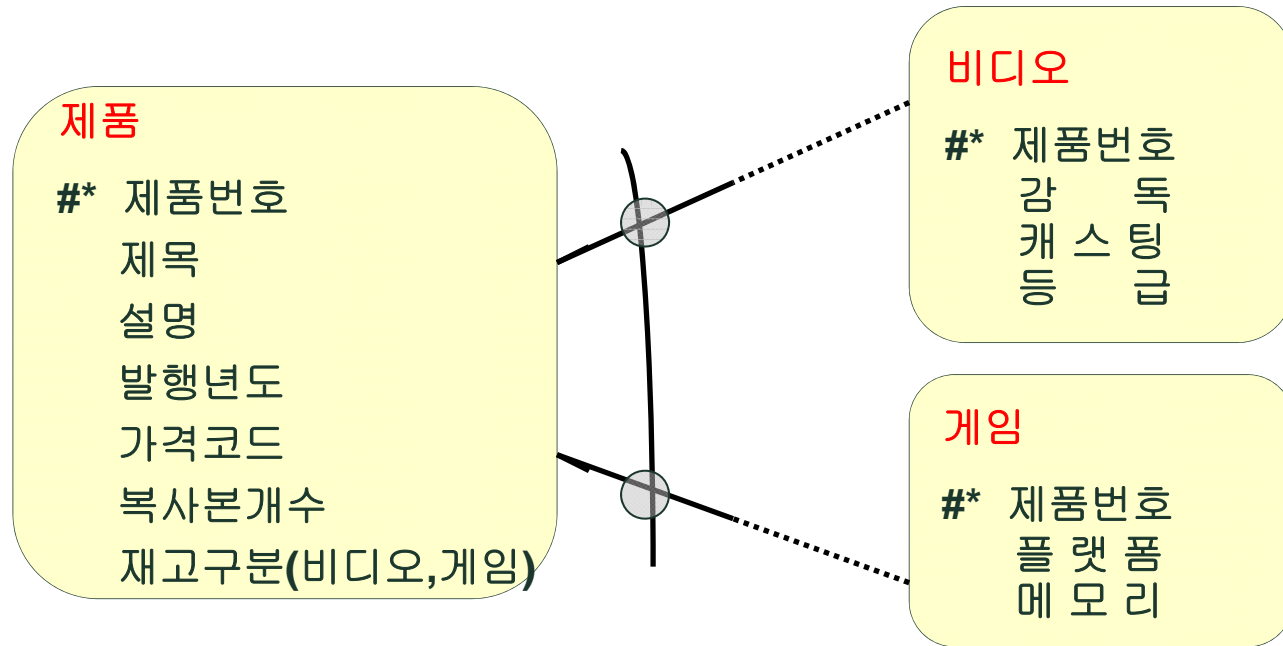
- 1) 데이터의 결합처리가 감소된다.
- 2) 물리적 접근이 단순하다.
- 3) 데이터에 많은 **Lock**이 발생한다.
- 4) 길이가 긴 데이터가 생길 수 있다

# 과도한 정규화를 피해야 하는 경우

- 1) 대상의 **Entity**가 검색 만으로 사용되는 경우
- 2) 사용되는 빈도가 적은 경우
- 3) 소규모의 데이터를 저장하는 **Entity**이거나, 드물게 변경되는 경우
- 4) 중복 데이터의 변경이 거의 발생하지 않고 저장공간이 비교적 적은 경우

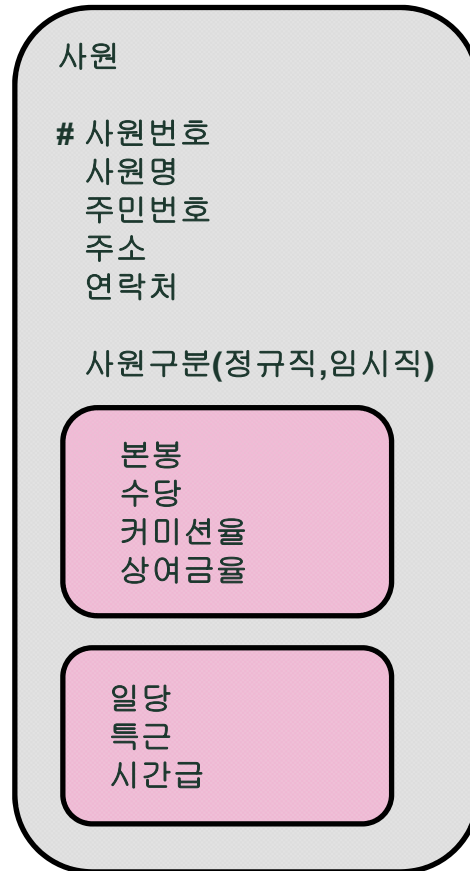


# Pattern : Basket

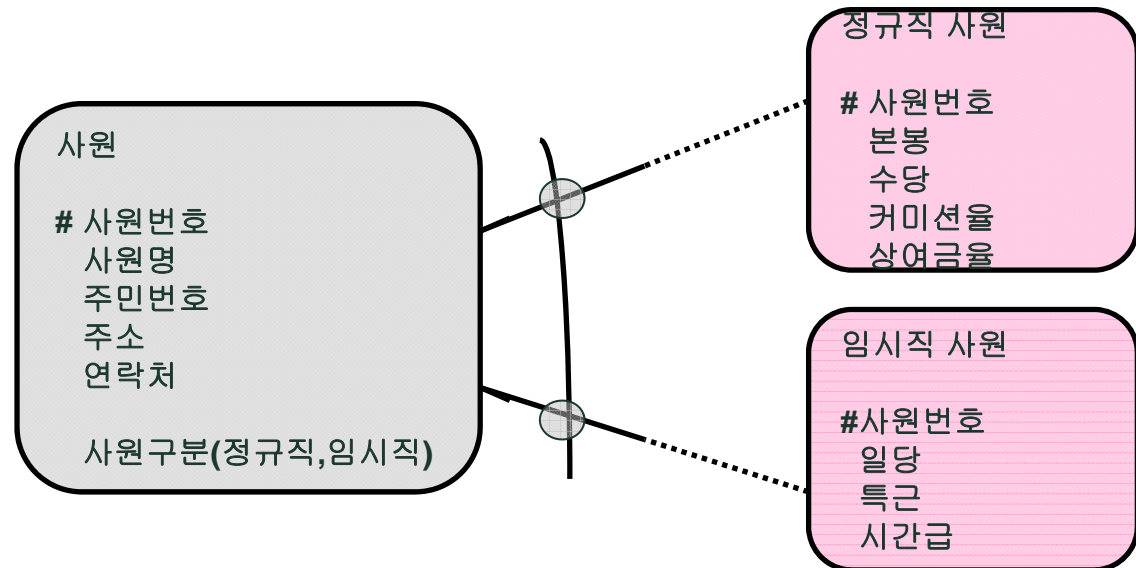


- 1) 다양한 항목에 대한 저장소 역할을 수행한다.
- 2) 구분자는 경우에 따라 다른 데이터 값을 가질 수 있다.
- 3) 배타적 관계 **Data Model**이라고 한다.

# Arc와 SubType의 비교



구현 단계에서 컬럼의 개수가 많은 하나의 테이블로 생성되어지며 구분자에 의해 사용되어지는 컬럼의 개수가 적은 경우에 사용될 수 있다.



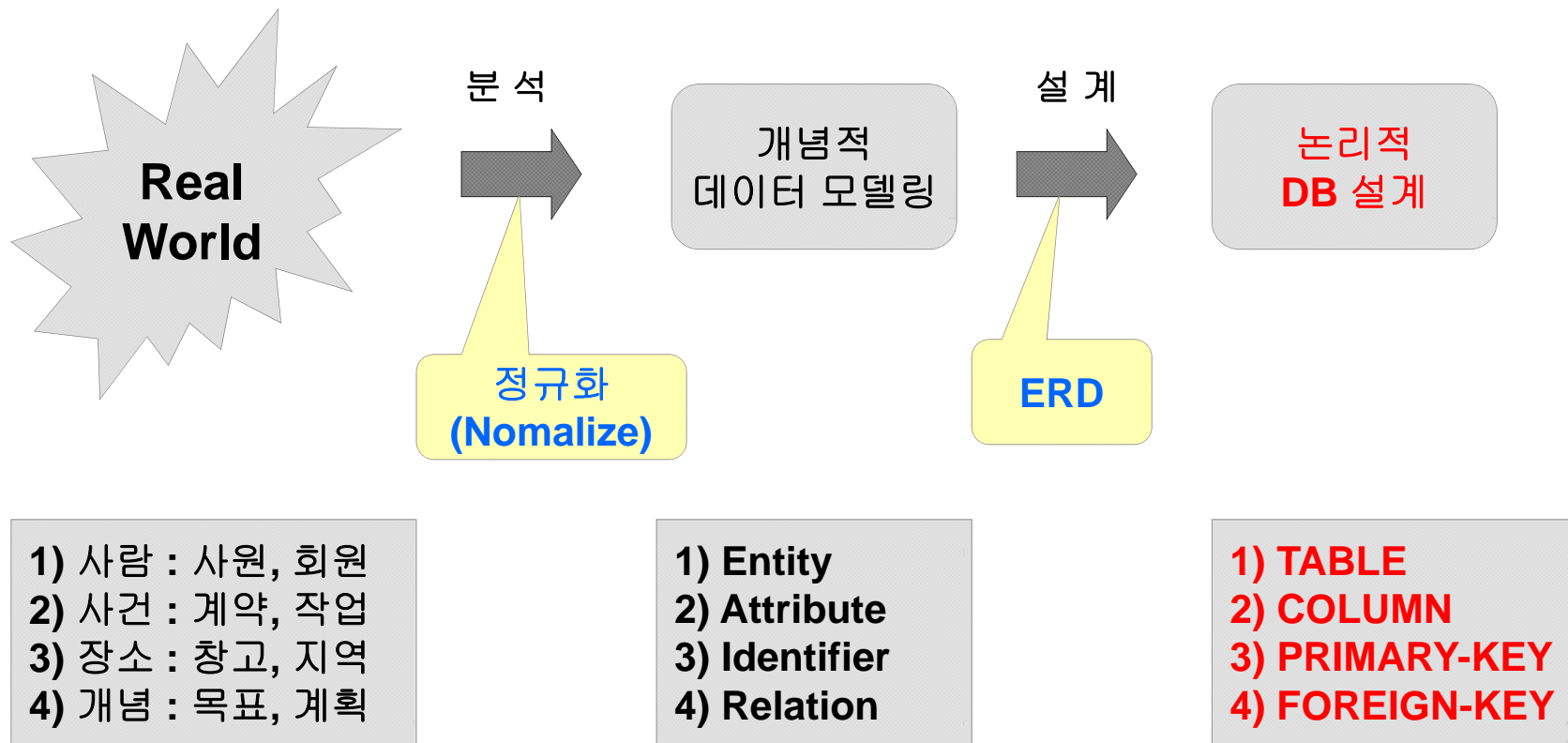
어떤 하나의 구분자에 의해 사용되는 컬럼의 개수가 상대적으로 많고 각 구분자에 의한 독립적인 시스템 구축이 요구되는 경우에 사용될 수 있다.

# 5

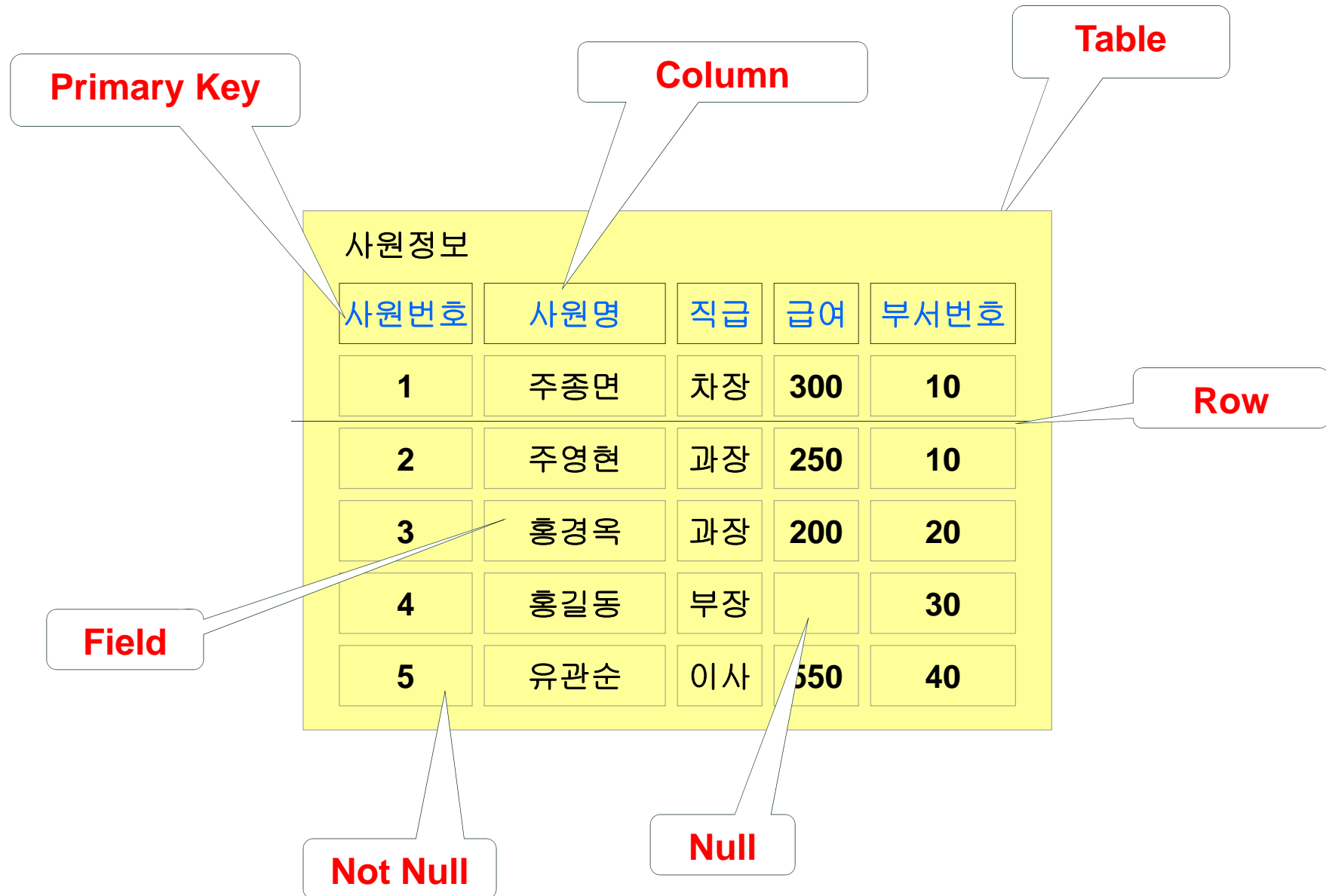
## 논리적 데이터 모델링

# 논리적 데이터베이스 설계

- 개념적 모델링 과정을 통해 만들어진 개념적 구조(**ERD**)들을 **DBMS**가 처리할 수 있는 객체로 생성하는 과정입니다.



# 관계형 데이터베이스의 개념



사원정보				
사원번호	사원명	직급	급여	부서번호
1	주종면	차장	300	10
2	주영현	과장	250	10
3	홍경옥	과장	200	20
4	홍길동	부장	350	
5	유관순	이사	550	40

Foreign Key

Primary Key

부서정보		
부서번호	부서명	지역코드
10	전산과	1
20	경영지원과	1
30	자재과	2
40	회계과	1

Primary Key

Foreign Key

지역정보	
부서코드	지역명
1	서울
2	부산

Primary Key

# Mapping 규칙

개념적 모델링

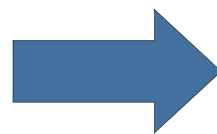
ENTITY

ATTRIBUTE

UID

RELATIONSHIP

Att.의 mandatory  
optional



논리적 DB 모델링

TABLE

COLUMN

PRIMARY-KEY

FOREIGN-KEY  
(전이관계)

NOT NULL  
NULL

# Entity를 Table로 작성

1) ERD에서 정의된 **Entity**는 논리적 **DB** 설계 단계에서 **Table**로 작성된다.

2) ERD의 **Entity** 명을 **Table** 명으로 사용하는 것이 좋다.  
(한글 **Entity**명을 영문 **Table**명으로 변경할 때는 **표준화**  
**기본 원칙과 표준 지침을 근거로 변경하라.**)

3) **Table** 명은 되도록 영문으로 작성하고 표준지침에 의해  
**너무 길지 않게 작성한다.(8문자 ~12문자 범위)**

< 표준 단어 사전 >

번 호	한글명	설 명	영문명	영문 약어명	단어 유형
1	고객	고객이름	CLNT	CT	
2	주소	고객 주소	ADR	AD	



# Att.를 Column으로 작성

- 1) **ERD**에서 정의된 **Attribute**는 논리적 **DB** 설계 단계에서 **Column**으로 작성된다.
- 2) 개발자의 혼돈을 피하기 위해 가능한 표준화된 약어를 사용한다.(표준 단어 사전을 활용하라.)
- 3) **SQL** 언어의 예약어를 컬럼 명으로 사용하지 마라.
- 4) 가능한 컬럼명은 표준지침에 의해 작성하고 너무 길지 않게 작성한다.(8문자 ~12문자 범위)

# UID를 Primary-Key로 작성

- 1) ERD에서 정의된 **UID**는 논리적 DB 설계 단계에서 **PRIMARY-KEY**로 작성되고 보조 **UID**는 **UNIQUE**로 작성된다.
- 2) **PRIMARY-KEY**와 **UNIQUE-KEY**는 구현 단계에서 인덱스로 생성된다.
- 3) 여러 개의 속성으로 구성된 **PRIMARY-KEY**와 **UNIQUE-KEY**의 컬럼 순서는 데이터의 빠른 검색을 결정하기 때문에 **컬럼 간의 우선 순위를 결정**해야 한다.

재고

#\* 제품번호  
# 선반위치  
#\* 창고번호  
재고수량  
재주문시점수량  
최대보유수량  
재고가바닥난이유  
재입고날짜

Case-1

재고

#\* 창고번호  
#\* 제품번호  
# 선반위치  
재고수량  
재주문시점수량  
최대보유수량  
재고가바닥난이유  
재입고날짜

Case-2

재고

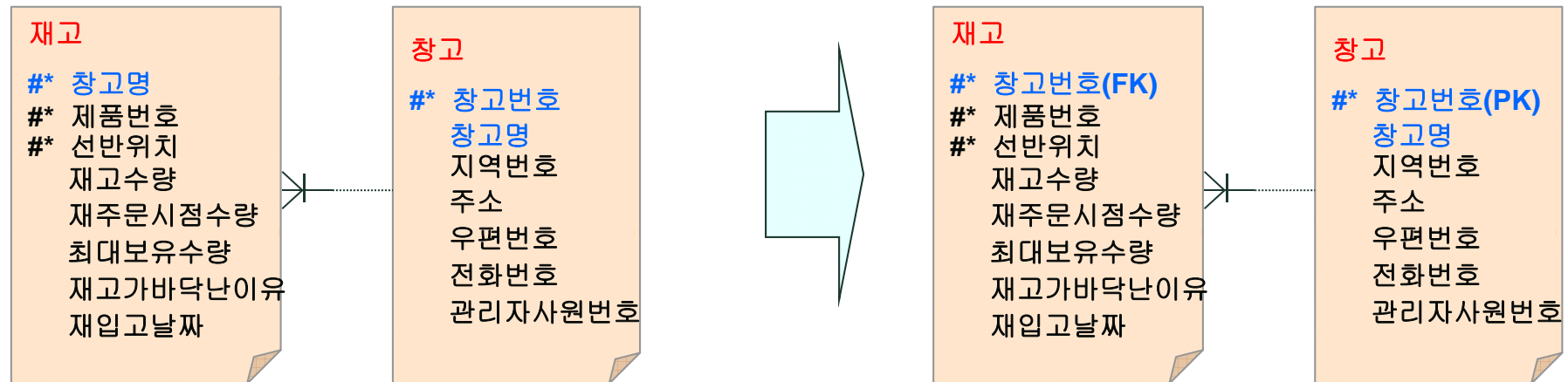
# 선반위치  
#\* 창고번호  
#\* 제품번호  
재고수량  
재주문시점수량  
최대보유수량  
재고가바닥난이유  
재입고날짜

Case-3

# Relation을 Foreign-Key로 작성

1) 개념적 모델링 단계에서 설정된 **Relationship**은 논리적 모델링 단계에서는 **Primary-Key**와 **Foreign-Key**로 설정된다. (전이 관계)

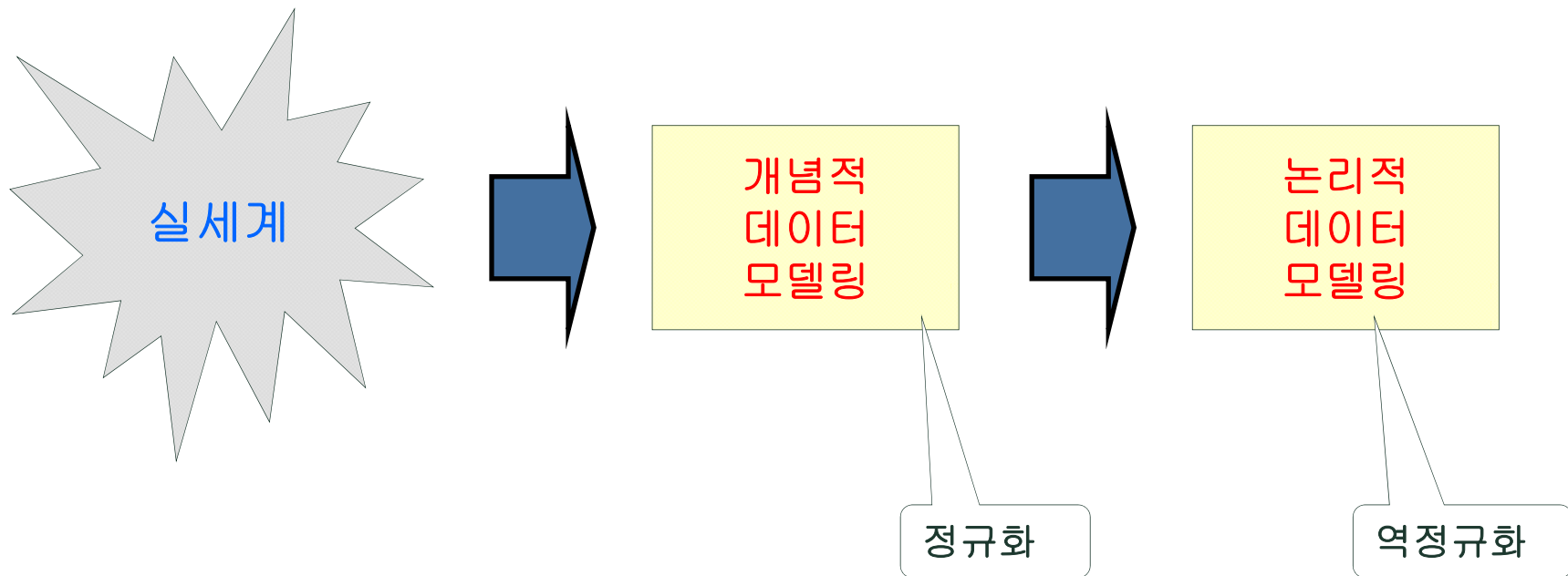
2) 순환 관계형 데이터 모델에서 자신의 **Primary-Key**를 **Foreign-Key**로 정의된다.



역 정규화

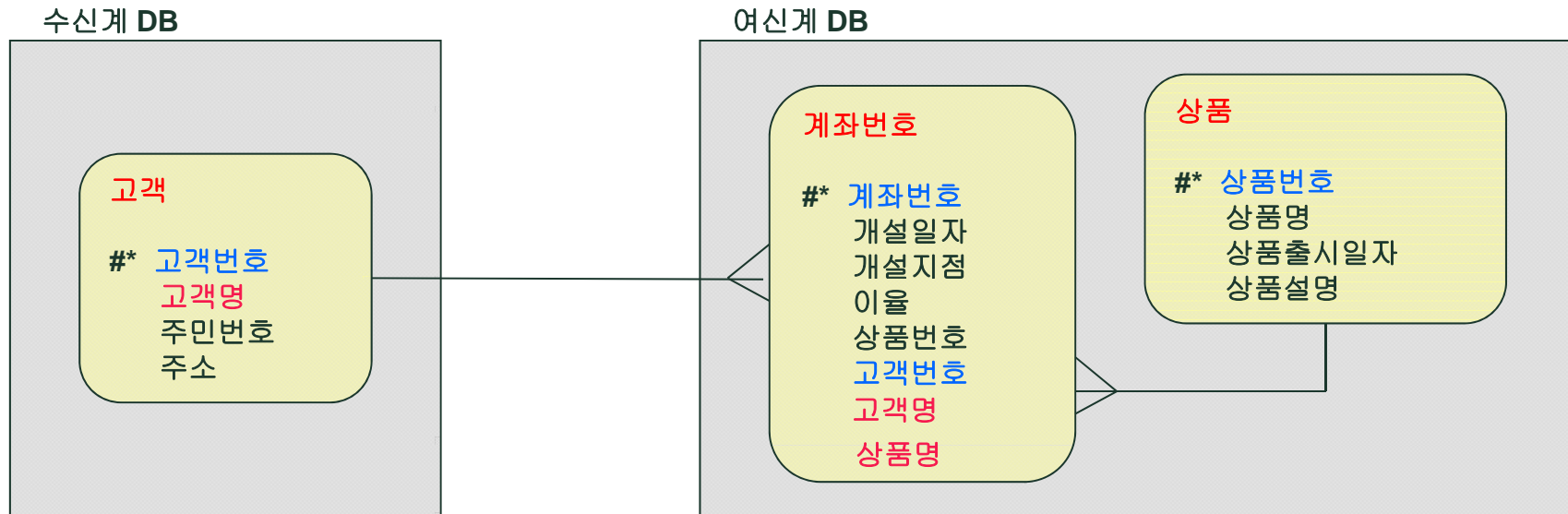
# 정규화와 역정규화

- . 데이터베이스의 성능 향상 및 효과적인 관리를 위해  
정규화에 위반되는 설계 행위를 역정규화(DeNomalization)  
라고 합니다.



# 역정규화

- 1) 데이터의 중복을 피하기 위한 과도한 정규화는 **불필요한 논리적, 물리적 결합을 유발**시킬 수 있다.



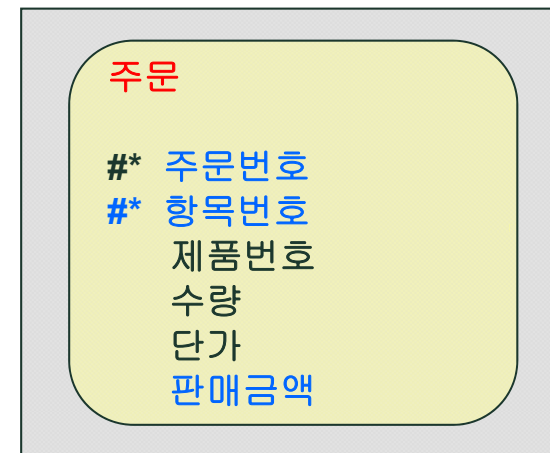
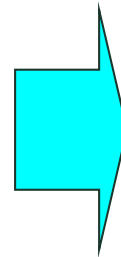
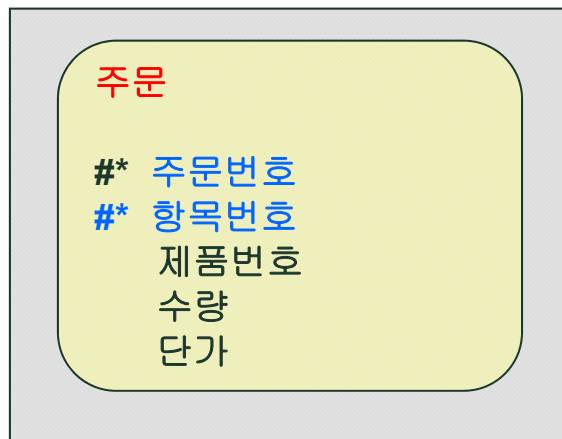
```
SELECT 계좌번호, 개설일자, 상품명, 고객명
FROM   계좌번호@여신 a, 상품@여신 c, 고객@수신 b
WHERE  a.상품번호 = b.상품번호 and
       a.고객번호 = c.고객번호 and
       ( a.계좌번호 >= '839-024101-02-001' and
         a.계좌번호 <= '839-024101-02-999' );
```

```
SELECT 계좌번호, 개설일자, 상품명, 고객명
FROM   계좌번호
WHERE  계좌번호 >= '839-024101-02-001' and
       계좌번호 <= '839-024101-02-999' ;
```

## 2) 개념적 데이터 모델링의 **Attribute**를 추출하는 단계에서

**Deriving(추출) 컬럼**은 다른 **Att.**의 계산 및 조합에 의해 추출될 수 있기 때문에 제거되었다.

하지만, 구현 단계에서 과도한 계산 및 공식이 적용되는 경우라면 **필요에 따라 Deriving 컬럼을 추가**할 수 있다.



```
SELECT 주문번호, 항목번호, 수량, 단가,  
       decode(신용도, 'A', (수량*단가) * 0.010,  
              'B', (수량*단가) * 0.005,  
              'C', (수량*단가)) 판매금액  
FROM 주문  
WHERE 주문번호 >= '2010-01-00001' and  
       주문번호 <= '2010-01-00001';
```

```
SELECT 주문번호, 항목번호, 수량, 단가, 판매금액  
FROM 주문  
WHERE 주문번호 >= '2010-01-00001' and  
       주문번호 <= '2010-01-00001';
```

3) 하나의 테이블에 정의할 수 있는 컬럼의 개수가 너무 많으면 불필요한 I/O가 발행하고 메모리의 효율성을 저하시킬 뿐 만 아니라 CPU의 과부하를 유발시켜 성능을 저하시키는 원인이 될 수 있으므로 수직 분할을 통해 여러 개의 테이블로 쪼개는 것이 유리할 수 있다.

제품 이미지 컬럼 포함

주문 테이블

주문번호	항목번호	제품번호	단가	수량	선택수량	제품이미지
20000101-01	01	1101	500	1200	1200	
20001231-11	10	1193	25	30	30	
20010101-01	01	1110	350	75	75	
20011231-99	08	1109	456	1300	1300	
20020101-01	01	1110	350	75	75	
20021231-22	34	3450	1300	23	23	

```
SELECT 주문번호, 항목번호, 제품번호, 수량, 단가
FROM 주문
WHERE 주문번호 >= '2010-01-00001' and
주문번호 <= '2010-01-00001';
```

제품 이미지 컬럼 포함 안됨

주문 테이블

주문번호	항목번호	제품번호	단가	수량	선택수량
20000101-01	01	1101	500	1200	1200
20001231-11	10	1193	25	30	30
20010101-01	01	1110	350	75	75
20011231-99	08	1109	456	1300	1300
20020101-01	01	1110	350	75	75
20021231-22	34	3450	1300	23	23

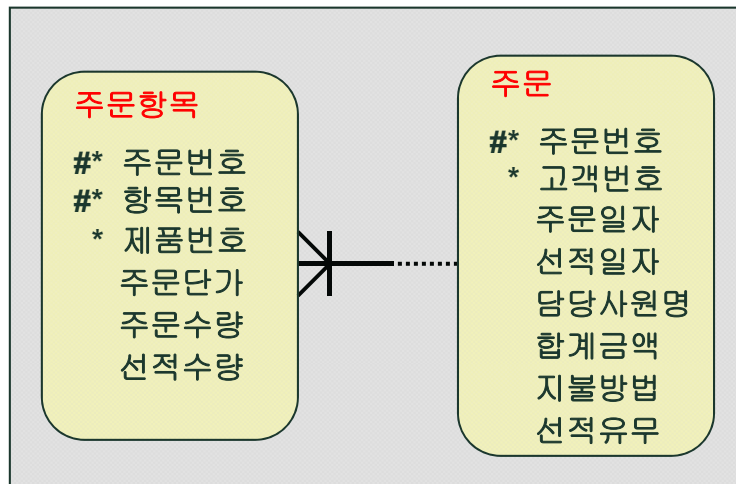
이미지 테이블

제품번호	제품이미지
1101	
1193	
1110	
1109	
1110	
3450	

```
SELECT 주문번호, 항목번호, 제품번호, 수량, 단가
FROM 주문
WHERE 주문번호 >= '2010-01-00001' and
주문번호 <= '2010-01-00001';
```

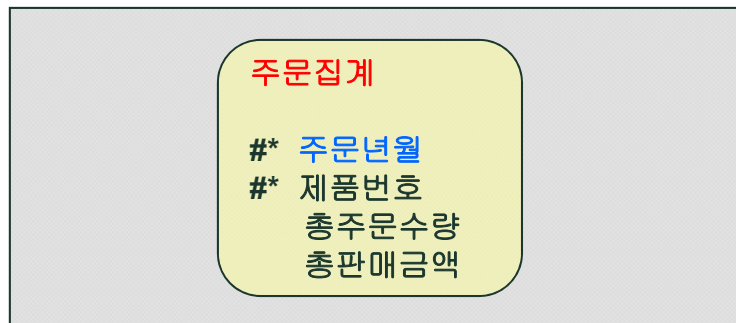


4) 데이터베이스를 운영하다 보면 월말마다 **집계 데이터**를 분석해야 하는 경우도 생기며, 기존 데이터에 대한 **로그 정보**를 남겨야 하는 경우도 발생하게 된다. 비록, 개념적 데이터 모델링 단계를 통해 분석된 **Entity**는 아니지만 사용자의 필요에 따라 **기능성 테이블**이 생성될 수도 있다.



```

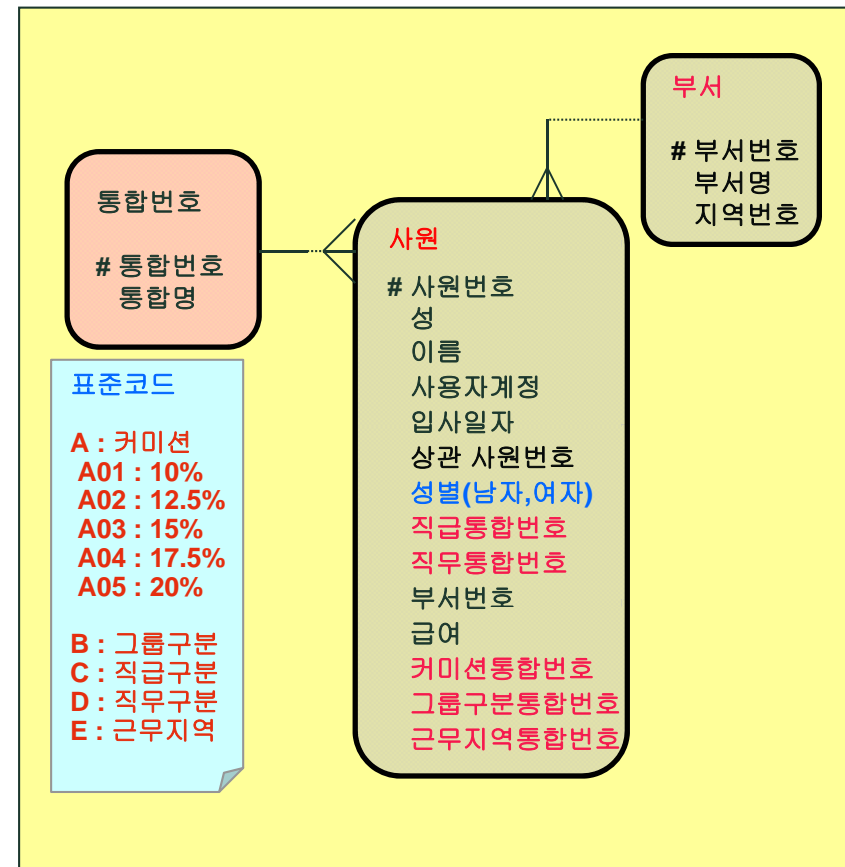
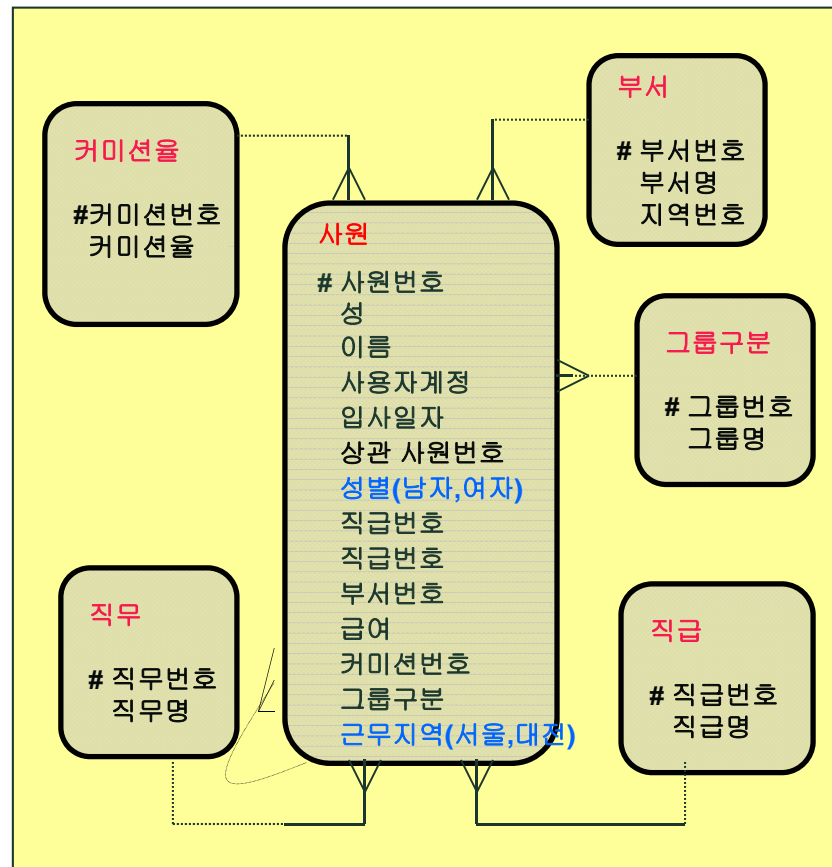
SELECT  COUNT(CASE WHEN TO_CHAR(주문날짜,'MM') = '01' THEN COUNT(*) END) AS Jan,
        COUNT(CASE WHEN TO_CHAR(주문날짜,'MM') = '02' THEN COUNT(*) END) AS Feb,
        COUNT(CASE WHEN TO_CHAR(주문날짜,'MM') = '03' THEN COUNT(*) END) AS Mar,
        COUNT(CASE WHEN TO_CHAR(주문날짜,'MM') = '04' THEN COUNT(*) END) AS Apr,
        COUNT(CASE WHEN TO_CHAR(주문날짜,'MM') = '05' THEN COUNT(*) END) AS May,
        COUNT(CASE WHEN TO_CHAR(주문날짜,'MM') = '06' THEN COUNT(*) END) AS Jun,
        COUNT(*) Total
FROM 주문, 주문항목, 제품
WHERE 주문.주문번호 = 주문항목.주문번호 AND 주문항목.제품번호 = 제품.제품번호
GROUP BY 주문날짜
Having TO_CHAR(주문날짜,'YYYYMM') >= 2010'01' AND TO_CHAR(주문날짜,'YYYYMM') <= 2010'06' ;
  
```



```

SELECT 주문년월, 제품번호, 총주문수량, 총판매금액
FROM 주문집계
WHERE to_char(주문년월, 'YYYYMM') >= '201001' AND
      to_char(주문년월, 'YYYYMM') <= '201006';
  
```

- 5) 정규화를 과도하게 수행하게 되면 행 길이가 짧은 테이블 개수가 늘어나서 관리 비용이 증가하게 되며 SQL문을 작성하게 되면 과도한 조인이 발생하고 개발 생산성이 떨어지며 때에 따라 성능이 저하되는 문제점을 유발시키게 된다.  
(유사한 데이터 구조를 가진 Entity는 통합코드 테이블 생성)



# 표준 코드 정의

- 현 코드를 기준으로 통합에 대한 **요구사항과 통합의 필요성에 따라** 통합해야 할 대상 **Entity**를 파악한 후 **표준코드를 정의**해야 한다.

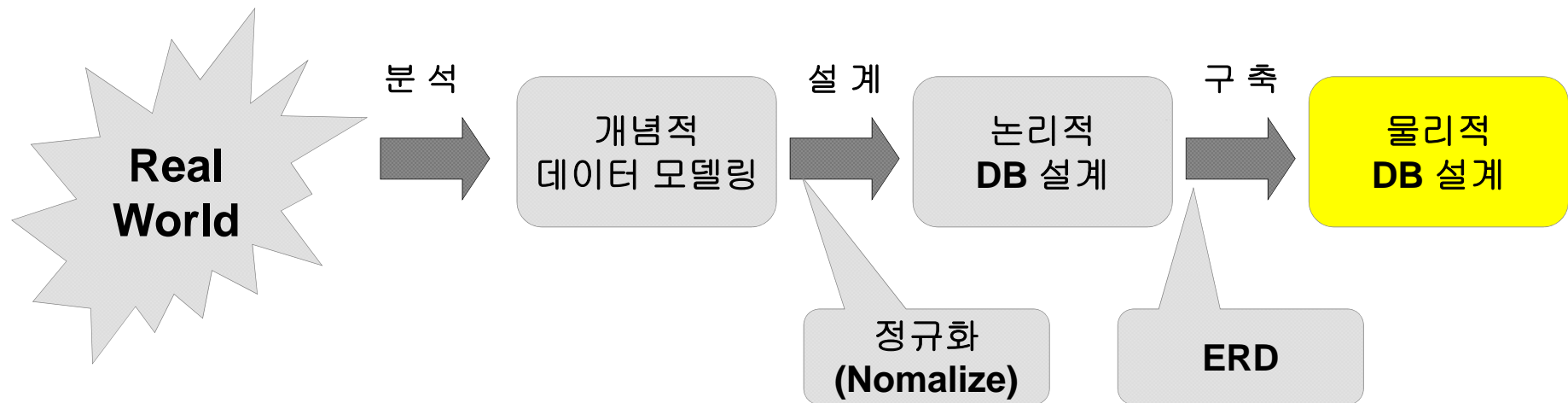
단독 코드 테이블	통합 코드 테이블	제약 조건
<p>1) 개념적 데이터 모델링 단계에서 과도한 정규화에 의해 발생한다.</p> <p>2) 주로 1~3개 컬럼으로 구성되며 소량의 데이터가 저장된다.</p> <p>(예) 직무 테이블 지역 테이블 커미션 테이블</p>	<p>1) 과도한 정규화 작업을 통해 생성된 단독코드 테이블 중에서 유사한 데이터 구조를 가진 테이블을 하나의 테이블로 통합 생성할 수 있다.</p> <p>2) 통합코드 테이블은 데이터 구조를 단순화시키며 <b>SQL</b>문을 효율적으로 작성할 수 있게 해준다.</p>	<p>1) 개념적 데이터 모델링 단계에 추출된 <b>Att.</b>의 부가 설명 중에서 데이터의 지속적인 생성, 수정, 삭제가 요구되는 항목은 단독코드 테이블 또는 통합코드 테이블로 구현되며 그렇지 못한 데이터 항목은 테이블 생성 시 <b>CHECK</b> 제약조건으로 구현된다.</p> <p>(예) 성별 (남자 : 1, 여자 : 2) 신용도(최우수 : 1, 우수 : 2) 선적여부(Y, N)</p>

# 6

## 물리적 데이터 모델링

# 물리적 데이터베이스 설계

- . 개념적 모델링 과정과 논리적 **DB** 설계과정을 통해 산출된 **ERD**를 근거로 **DBMS**를 선택한 후 설치하는 단계를 말합니다.
- . 최종 설계된 **ERD** 모델에 대해 테이블의 물리적 저장구조에 대한 설계를 한 후 스키마를 생성합니다.



# DB 설계

논리적 DB 설계

물리적 DB 설계

```
CREATE TABLE s_emp
(id          NUMBER(7)
last_name    VARCHAR2(25)
first_name   VARCHAR2(25),
userid       VARCHAR2(8)
start_date   DATE,
comments     VARCHAR2(255),
manager_id   NUMBER(7),
title        VARCHAR2(25),
dept_id      NUMBER(7)

salary       NUMBER(11, 2),
commission_pct NUMBER(4, 2),
```

```
CONSTRAINT s_emp_id_nn      NOT NULL,
CONSTRAINT s_emp_last_name_nn NOT NULL,

CONSTRAINT s_emp_userid     NOT NULL,
```

```
CONSTRAINT s_emp_dept_id_fk
References s_dept(id),
```

```
CONSTRAINT s_emp_id_pk      PRIMARY KEY (id),
CONSTRAINT s_emp_userid_uk  UNIQUE (userid),
CONSTRAINT s_emp_commission_pct_ck
CHECK (commission_pct IN (10, 12.5, 15, 17.5, 20)))
```

**TABLESPACE SALES**

```
STORAGE (INITIAL      10K      NEXT      10K
          MINEXTENTS   1        MAXEXTENTS  UNLIMITED
          PCTINCREASE  50)
PCTFREE  10    PCTUSED  40    FREELIST  1
PARALLEL          CACHE;
```

# 제 약 조 건

# DB의 무결성 보장방법

사원관리

입력 수정 삭제 종료

사원번호 2001

성명 주종면

급여 10000

메시지 사원번호는 5자리입니다 !!!!

```
Create trigger chk_emp
Before update or delete or insert on emp
Begin
  If Len(:new.no) <> 5 Then
    Dbms_output.put_line('사원번호는 5자리!!');
  end if;
End;
```

```
Create table emp
( no          number(4)      Primary Key,
  name        varchar2(10)   Not Null,
  Loc         varchar2(15)   CHECK (Loc in ('서울' , '부산' )),
  jumin_no    char(15)       Unique,
  deptno      number(2)      References dept(deptno));
```



# CONSTRAINTS

EMP				
no	ename	loc	jumin	deptno
1	주종면	서울	65...	10
2	주영현	서울	78...	10
3	홍경옥	부산	67...	20

DEPT		
deptno	dname	loc
10	전산과	1
20	경영지원과	1
30	자재과	2

```

Create table emp
(no      number(4)   Primary Key,
name    varchar2(10) Not Null,
Loc     varchar2(15)
CHECK (Loc in ('서울' , '부산' )),
jumin   char(15)    Unique,
deptno  number(2)
References dept(deptno)
On Delete Cascade);
    
```

```

Create table dept
(deptno  number(2) Primary Key,
dname   varchar2(15) default ' ',
Loc     char(1)
CHECK(LOC IN ('1','2'));
    
```

# 테이블 설계서

## 지역

##\* 지역번호 number(7)  
\* 지역명 varchar2(50)

## 부서

##\* 부서코드 number(7)  
\* 부서명 varchar2(25)  
지역코드 number(7)

테이블 명(영문) (한글)	s_region		관 련 업 무		영업관리	
	지역		DB 사용자계정		sales	
영문 컬럼(한글)	제약조건	Null 여부	F.K 테이블	F.K 컬럼	Data 타입	길이
id (지역코드)	PK	NN			NUMBER	7
name (지역명)	UQ	NN			VARCHAR2	50

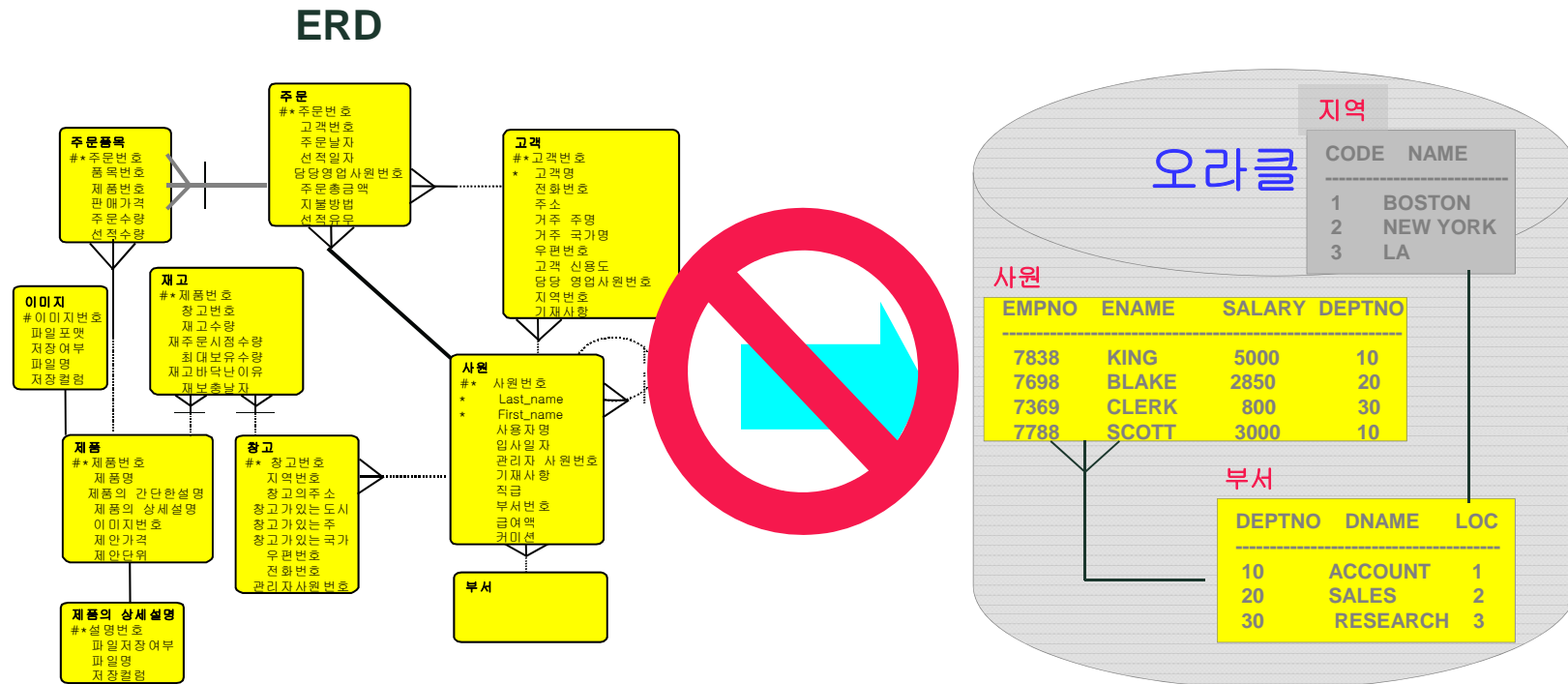
테이블 명(영문) (한글)	s_dept 3		관 련 업 무		영업관리 1	
	부서 4 7		DB 사용자계정		sales 2	
영문 컬럼(한글)	제약조건	Null 여부	F.K 테이블	F.K 컬럼	Data 타입	길이
id (부서코드)	PK	NN			NUMBER	7
name (부서명)		NN			VARCHAR2	25
region_id(지역코			s_region	id	NUMBER	7

# 제약 조건 명

1) **DBMS** 내의 정보와 논리적 설계의 결과가 다른 경우를 위해 반드시 제약조건 명을 부여해야 한다. ([테이블명]\_[컬럼명]\_[제약조건명])

2) 테이블 명과 컬럼 명에 대해 주석을 부여하여 효과적인 검색이 가능하도록

**COMMENT** 명령어를 사용한다. (COMMENT ON TABLE ~ IS ~ ;  
COMMENT ON COLUMN ~ IS ~;)



# 제약 조건 유형 및 제약 조건 명

## 1) 컬럼 레벨

```
CREATE TABLE s_region  
(id    NUMBER(7)    PRIMARY KEY (id),  
 name  VARCHAR2(50) UNIQUE));
```

## 2) 테이블 레벨

```
CREATE TABLE s_region  
(id    NUMERIC(7)    CONSTRAINT nn_s_region_id  NOT NULL,  
 name  VARCHAR(50)   CONSTRAINT nn_s_region_name NOT NULL,  
                                CONSTRAINT pk_s_region_id  PRIMARY KEY (id),  
                                CONSTRAINT uk_s_region_name UNIQUE (name));
```

# 테이블 생성 스크립트

summit2.sql

지역

**#\*** 지역번호 number(7)  
**\*** 지역명 varchar2(50)

```
CREATE TABLE s_region
(id      NUMBER(7)    CONSTRAINT s_region_id_nn  NOT NULL,
 name   VARCHAR2(50) CONSTRAINT s_region_name_nn NOT NULL,
                                CONSTRAINT s_region_id_pk  PRIMARY KEY (id),
                                CONSTRAINT s_region_name_uk UNIQUE (name))
;
```

부서

**#\*** 부서코드 number(7)  
**\*** 부서명 varchar2(25)  
지역코드 number(7)

```
CREATE TABLE s_dept
(id          NUMBER(7)    CONSTRAINT s_dept_id_nn  NOT NULL,
 name       VARCHAR2(25) CONSTRAINT s_dept_name_nn NOT NULL,
 region_id  NUMBER(7),
                                CONSTRAINT s_dept_id_pk  PRIMARY KEY (id))
;
```