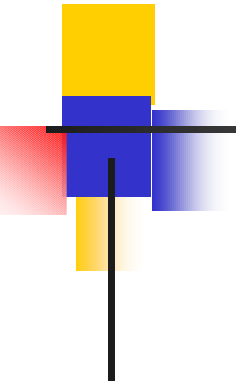
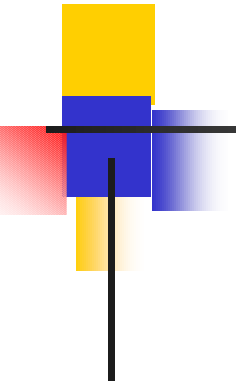


데이터베이스 모델링 및 설계구축 실무과정

목 차

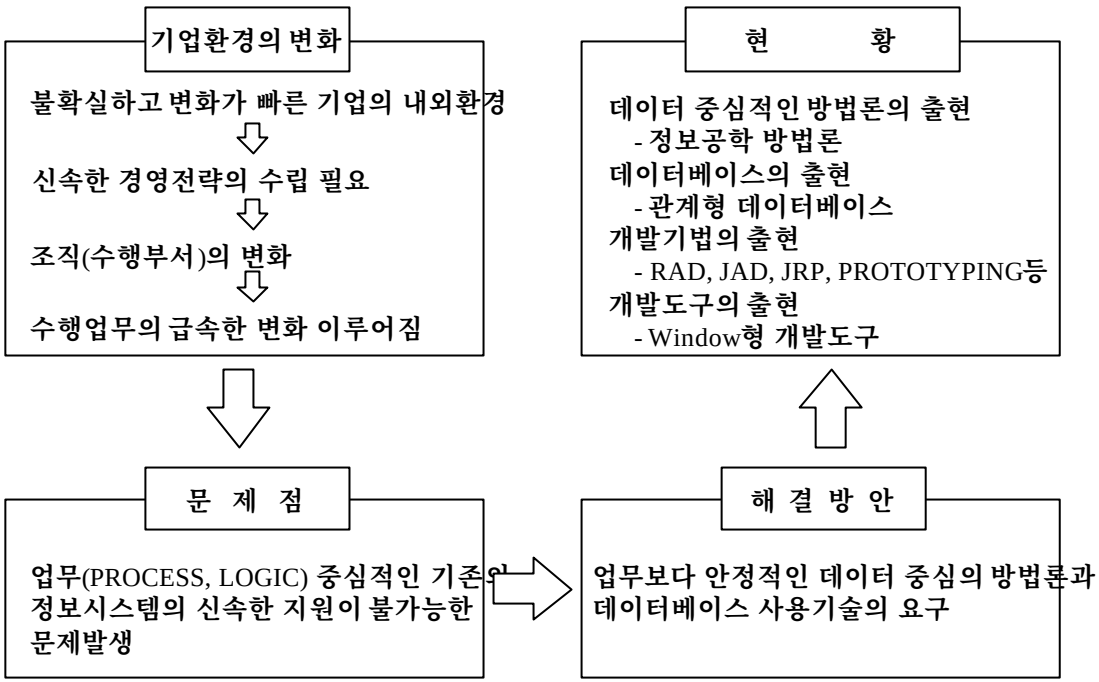


1부. 데이터베이스 모델링 및 설계

목 차

1. 관계형 이론 및 관계형 데이터베이스 개념

1.1 관계형 이론의 탄생배경



1.2 관계형 모델의 정의 및 특성

● 관계형 모델의 정의

관계형 모델은 업무에서 사용되는 데이터를 인식하여 손쉬운 방법으로 그들을 구성하되 체계적으로 정리하여 그들을 관리하는데 있어서 적용되는 정연하고도 직관적인 방법이다. (테이블형태로 기술 및 조작, 처리) 즉 관계형 모델은 데이터가 사용자에게 어떻게 비쳐지며, 사용자는 데이터에 어떠한 조작을 하며, 데이터가 처리할 때 데이터가 어떻게 유지되어야 하는 것에 대한 토대이다.

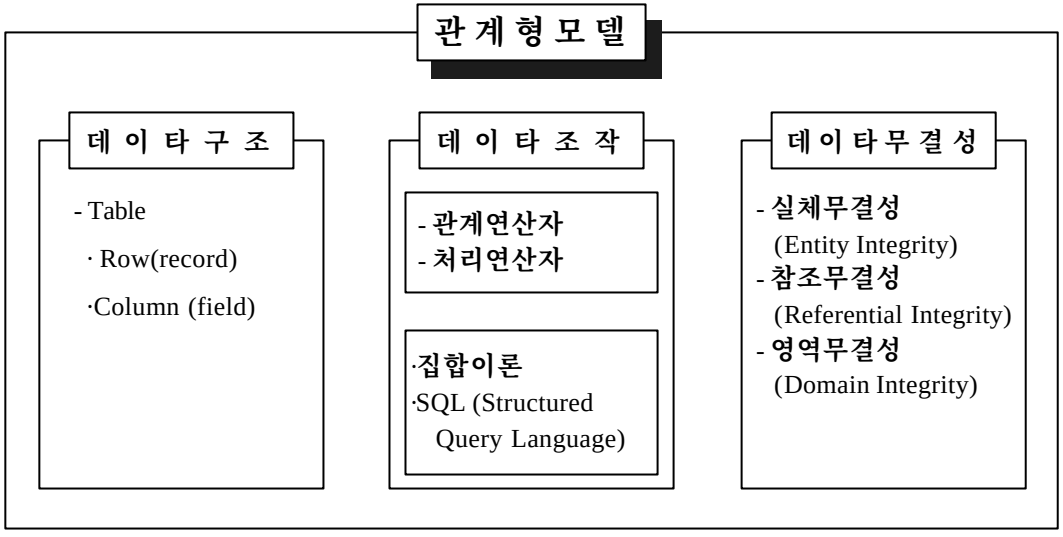
● 관계형 모델의 특성

Relational Theory (E.F.Codd, C.J.Date)
: 사용자의 데이터방식에 근거하여 집합개념으로 데이터를 이해하고 처리
: 관계형 데이터 모델은 엔티티(Entity, 객체;실체;테이블;파일등)와 그들 사이의 관계 (Relationship)로 구성되고 표현되어지며, 엔티티 사이의 데이터조작과 데이터조회는 테이블상의 작업으로써 쉽게 나타내진다.

● 주요 성공 요소

- 데이터 중심분석의 정확한 개념 이해
- 데이터 실체 및 단순속성으로의 구분방법과 각각의 특성별 분류방법 이해

1.3 관계형 모델의 구성3요소



●주요 성공 요소

- 데이터구조에서 실체(Entity)와 테이블의 연결 및 관계설정 방법의 이해
- 데이터조작에서 SQL사용방법의 숙지
- 데이터무결성에서 실무의 업무규칙과 무결성과의 관계를 정확히 이해

가. 데이터 구조

1) 관계(Relation)

- 행(Row)과 열(Column)로 이루어진 2차원 테이블

2) 관계형 테이블의 특성

- 각 열(Column)은 오직 하나의 값을 가짐(No repeating group)
- 동일 테이블의 동일 열(Column)은 동종의 값을 가짐(Same domain)
- 각 행은 특정 열의 값에 의해 유일하게 식별됨(Unique identifier)
- 각 열의 순서는 무의미(No hidden meaning by the order)
- 각 행의 순서는 무의미(Ability to retrieve rows in any sequence)
- 각 열은 유일한 이름을 가짐(Referenced by name not by position)

3) 관계의 두 형태

- Base Relation : 관계형 테이블
- Derived Relation : 뷰 (View, 가상테이블)

나. 데이터 조작

1) 특성

- SET **처리** (not one record at a time)
- **물리적 저장방법과 무관**

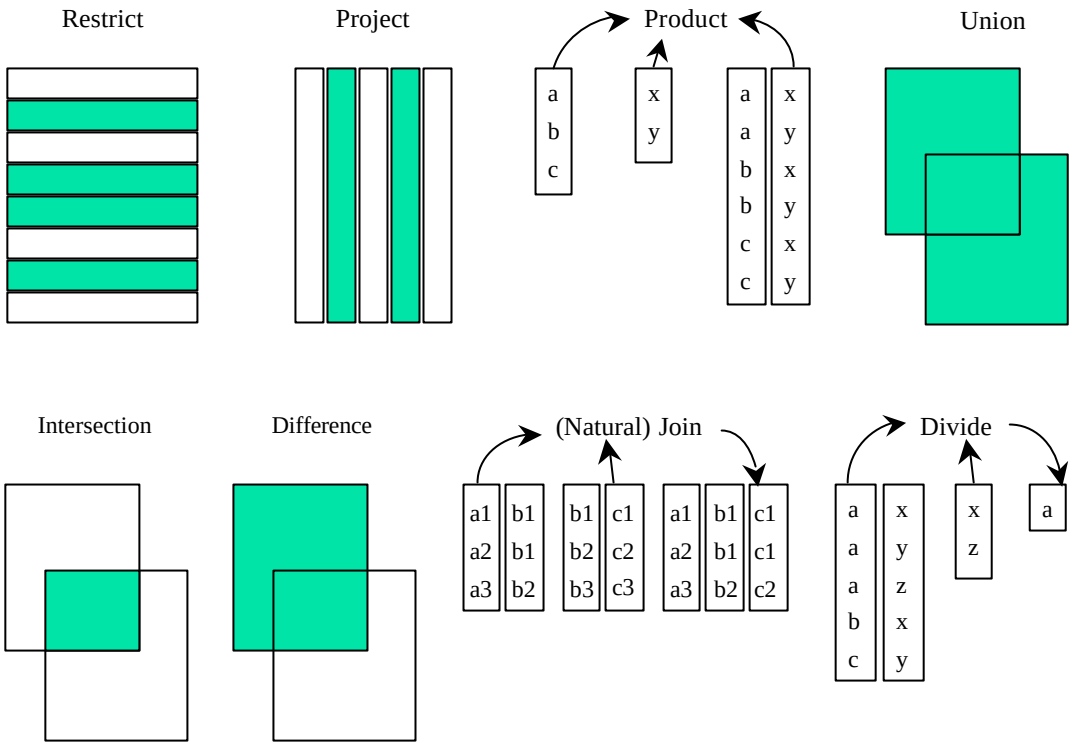
2) 기본 연산자 : 조회

- Select(or Restrict) : **열에 의거하여 행의** Subset**을 조회**
- Project : **열의** Subset**을 조회**
- Product : **두 관계형 테이블간 각 행의 조합을 묶음**(concatenation)
  전체 행의 수는 한 테이블의 행의 수와 다른 테이블의 행의 수를 곱한 것과 같다.
- Join : **특정열의 선택기준에 의거하여 각 행의 조합을 수평적으로 묶음**
- Union : **중복을 없이하여 각 행을 수직적으로 묶음** (Stack)
- Intersection : **관계형 테이블간에 공통된 행**
- Difference : **한 관계형 테이블에만 있는 행**
- Division : **어느 한 관계형 테이블**(A)**에서 다른 테이블**(B)**의 모든 행에 대응하는 열 값을 가진** B**테이블 열을 제외한** A**테이블의 열**

3) 처리 연산자 : 관계형 테이블의 내용에 변화

- Insert : **행의 입력**
- Update : **열의 수정**
- Delete : **행의 삭제**

4) 연산자(OVERVIEW)



다. 데이터 무결성

1) 무결성 법칙

- 데이터에 의미를 부여
- 각 데이터에 허용 가능한 값을 제한

2) 종류

① 엔티티 무결성 법칙(Entity Integrity Rule)
- 기본 키는 Null값을 포함하지 않는다.
- 기본 키는 해당 테이블의 행을 유일하게 식별 가능하게 한다.

② 참조 무결성 법칙(Referential Integrity Rule)
- 외부 키(Foreign Key)가 있는 테이블의 경우, 외부 키의 값은 Null이거나 관련 테이블에 이와 대응하는 기본 키가 있어야 한다.
  (Insert, Update, Delete시 반영)

③ 영역 무결성 법칙(Domain Integrity Rule)
- 테이블 내의 모든 행에 관한 무결성 법칙
- 데이터 형태, 범위 검사, 기본값, 유일성등에 관한 제한

④ 속성무결성 법칙(Triggering Operation ; 연쇄작용)
- 테이블, 속성값과 관련된 데이터의 무결성을 유지하기 위한 Operation

1.4 파일시스템, 계층형/망형 DB 시스템, 관계형 DB시스템의 차이

● 핵심내용

파일 시스템	계층형/망형 DB시스템	관계형 DB시스템
- 주요 관점의 대상은 어플리케이션 (응용업무프로그램)이고 관련 데이터는 파일에 저장 - 어플리케이션에 특화된 형태로 데이터가 저장되고, 만약 조금이라도 다른 형태의 데이터를 필요로 하는 경우 중복된 데이터라도 다른 형식으로 다시 저장해야 한다.	- 한 레코드는 다른 레코드에 대해 한 개 이상의 Pointer를 가짐 - DB는 레코드 타입을 포함한다. - DB Search는 첫번째 Record를 찾은 후 다음 레코드를 Pointer로써 읽음 - 데이터사이의 동적(Dynamic)인 상호연관사항은 존재치 않음 - Data 독립성(Independence)의 보장이 안 된다, 즉 Record Format이 변하면 어플리케이션도 모두 바뀐다.	- SQL 지원 - 이전에 어플리케이션에서 처리해야 했던 많은 기능들을 DBMS가 지원 - 데이터모델링이 간편해짐 - DB는 Table로 구성됨 - 레코드는 다른 레코드에 대하여 어떤 Pointer라도 갖지 못한다. - Data 사이의 상호관계는 JOIN을 통하여 일어난다. - DB Schema에 대한 동적인 변화가 가능하다.

● 주요성공요소

- RDB는 Pointer가 아닌 Table들 끼리의 JOIN등으로 DB Search
- SQL의 이해 및 데이터모델 별 사용방법 등의 숙련도 요구

1.5 데이터베이스 개념 및 설계

가. 데이터베이스의 정의 및 특성

1) 데이터베이스(Database) 정의

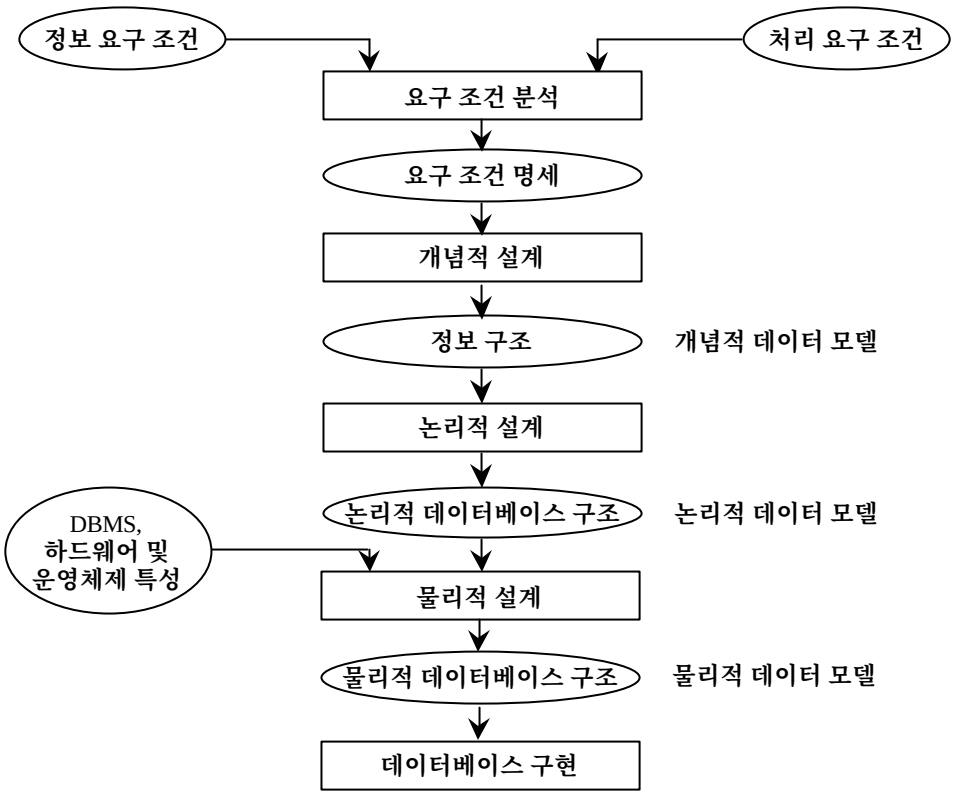
어느 한 조직의 다수 응용 시스템들이 사용하기 위해 통합, 저장된 서로 관련된 데이터의 집합

- 통합된 데이터(Integrated data) : 중복 배제
- 저장된 데이터(stored data) : 저장 장치에 수록
- 운영 데이터(operational data) : 유용성 있는 데이터 집합
- 공용 데이터(shared data) : 상이한 다수 사용자들이 서로 다른 목적으로 이용

2) 특성

- 실시간 접근성 (real-time accessibility)
- 계속적인 변화 (continuous evolution)
- 동시 공유 (concurrent sharing)

나. 데이터베이스 설계


정보 요구 조건
처리 요구 조건
요구 조건 분석
요구 조건 명세
개념적 설계
정보 구조
개념적 데이터 모델
논리적 설계
논리적 데이터베이스 구조
논리적 데이터 모델
DBMS,
하드웨어 및
운영체제 특성
물리적 설계
물리적 데이터베이스 구조
물리적 데이터 모델
데이터베이스 구현

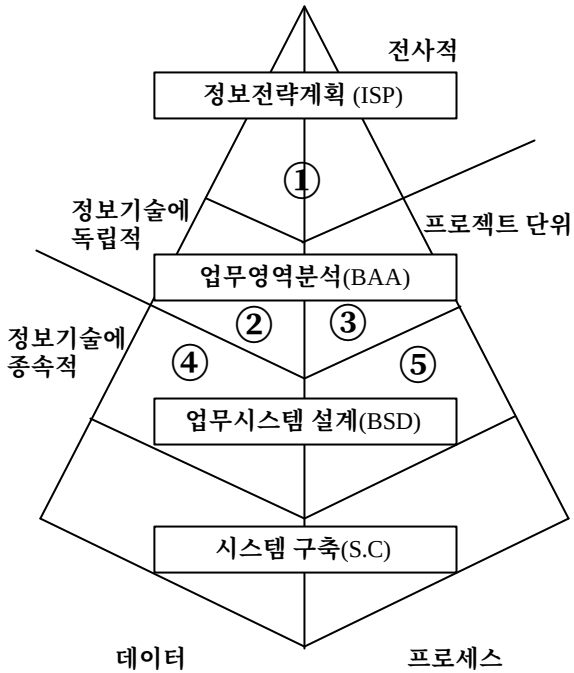
2. 정보공학과 개발단계

2.1 정보공학의 정의 및 기본 원리

- 정보공학의 정의
- 기업의 필요지식을 관리
- 공학적 접근
  - 하향식 접근
  - 분할과 정복
  - 전단계의 업무활동 및 산출물과 다음단계의 업무활동 및 산출물의 연계
- 엔터프라이즈모델에 기반
  - Top - Down 분석 및 경영전략의 체계적인 정보시스템화
- 정보시스템의 지속적 개선
  - 통합 DB를 가능케 해 주는 Modeling Document SPEC(사양)에 의한 확장성 보장
- 현업참여 환경 조성
  - 현업 사용자들과 함께하는 분석설계 기술과 기법 및 각종 자동화 도구의 지원
  - Graphic User Interface 환경(Window 등)
  - 도형에 의한 의견교환
- 다양한 개발경로

2.2 정보공학의 개발단계

가. 정보공학 피라미드



① Enterprise Modeling

② Logical Data Modeling

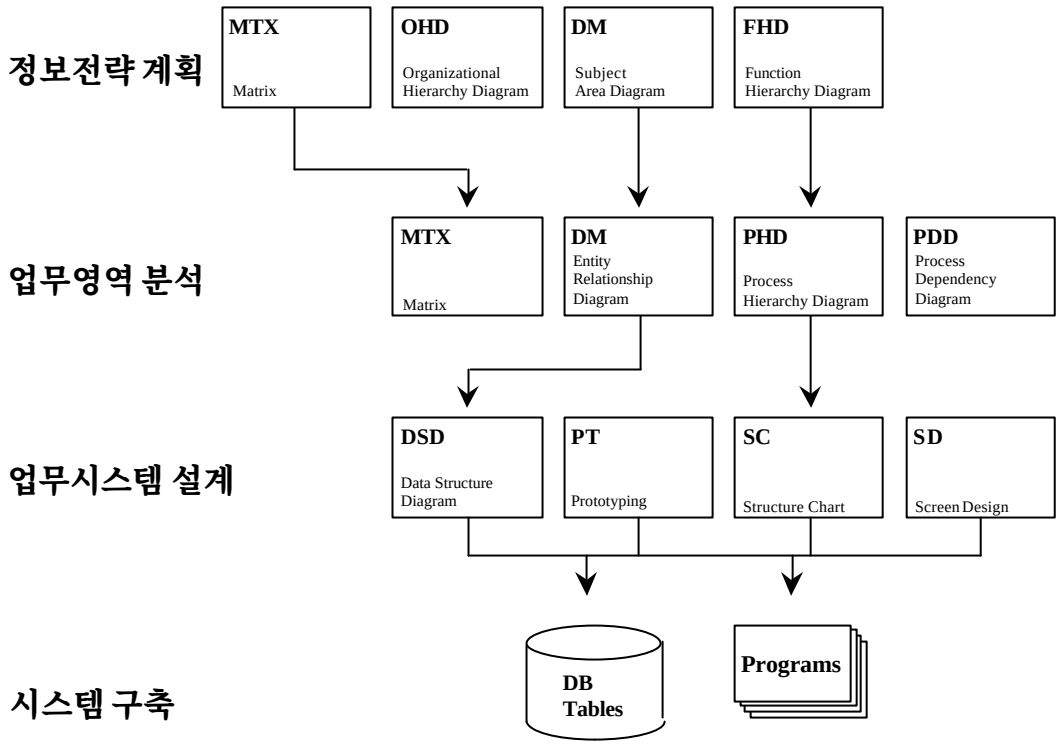
③ Process Modeling

④ Transition Design (Database Design)

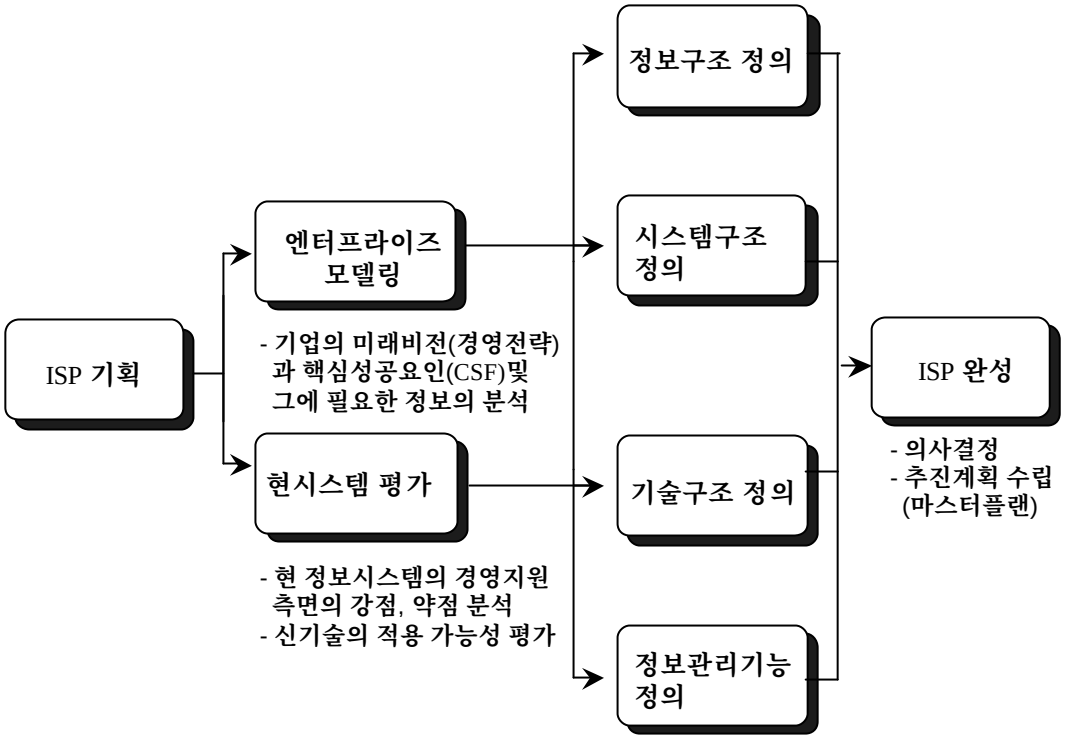
⑤ Information Technology Design

(Application Program Design)

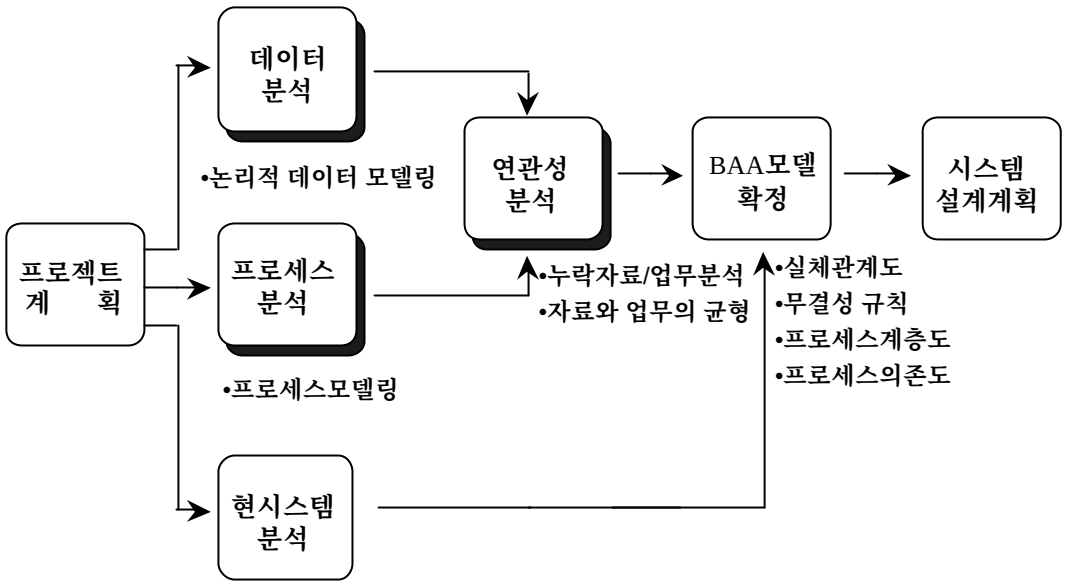
나. 단계별 산출물


정보전략 계획
MTX
Matrix
OHD
Organizational Hierarchy Diagram
DM
Subject Area Diagram
FHD
Function Hierarchy Diagram
업무영역 분석
MTX
Matrix
DM
Entity Relationship Diagram
PHD
Process Hierarchy Diagram
PDD
Process Dependency Diagram
업무시스템 설계
DSD
Data Structure Diagram
PT
Prototyping
SC
Structure Chart
SD
Screen Design
시스템 구축
DB Tables
Programs

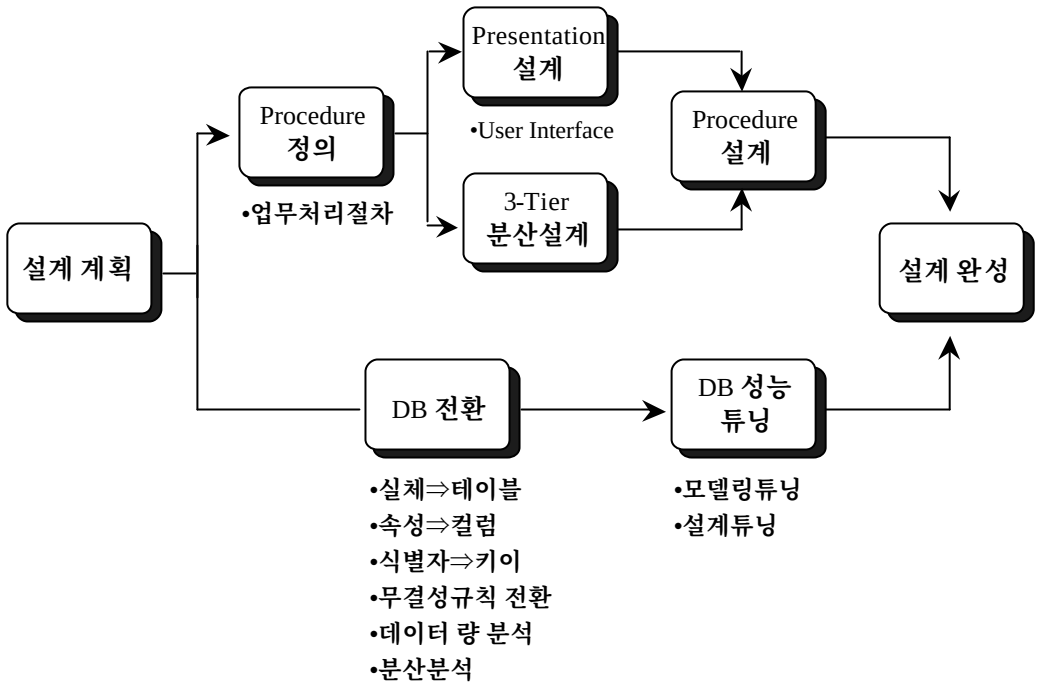
다. 정보전략계획 프로세스



라. 업무영역분석 프로세스



마. 설계 프로세스


설계 계획
Procedure 정의
•업무처리절차
Presentation 설계
•User Interface
3-Tier 분산설계
Procedure 설계
설계 완성
DB 전환
•실체⇒테이블
•속성⇒컬럼
•식별자⇒키이
•무결성규칙 전환
•데이터 량 분석
•분산분석
DB 성능 튜닝
•모델링튜닝
•설계튜닝

2.3 RDB 구축 시 정보공학 개발단계의 이해

● 핵심내용

- 개발 생명주기별 각 단계별 산출물들의 연계

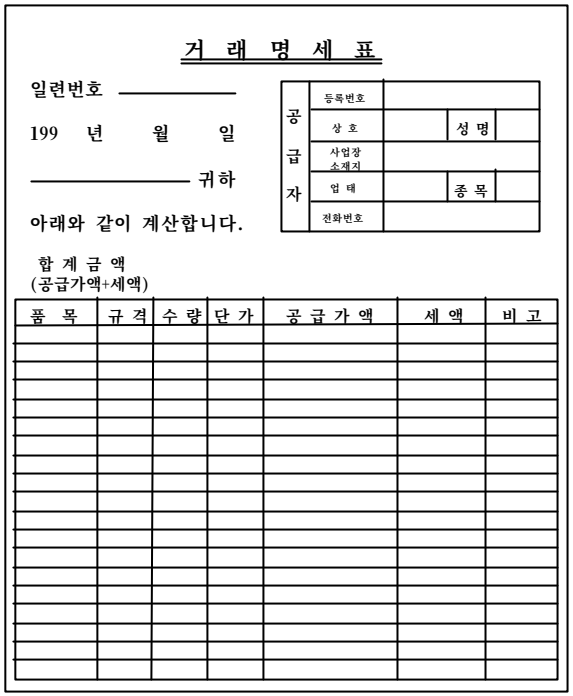
예) 관계형 모델을 활용하여 업무시스템을 개발하는 개발생명주기의 예

1) 거래명세표상의 Data특성 (속성유형기준)
2) 거래명세표상의 Data특성 (실체자격기준)
3) 거래명세표의 관계형 데이터모델 도출 : 업무영역분석(BAA)단계
   : 실체관계도 (Entity-Relationship Diagram)
4) 거래명세표의 논리적 관계형 데이터 모델(Logical ERD)로부터 물리적 관계형 데이터 모델(Physical ERD)전환 : 업무시스템 설계(BSD) 단계
5) 거래명세표의 물리적 관계형 데이터 모델링(Physical ERD)로 부터의 참조무결성규칙 도출
   : 업무시스템 설계(BRD) 단계
6) 거래명세표의 물리적 관계형 데이터 모델링(Physical ERD)로 부터의 Table Create
   : 업무시스템설계(BSD) 단계
7) 거래명세표의 물리적 관계형 데이터 모델(Physical ERD)로 부터 데이터조작과 조회
   : 구축(SC) 단계

● 주요성공요소

- 데이터 중심분석의 개념파악
- 데이터의 실체(Entity)및 단순속성(Attribute)분류기준이해
- 실체(Entity)의 종류 분석방법이해 및 숙지
- 속성(Attribute)의 종류 분석방법이해 및 숙지
- 각 단계별 산출물들끼리의 Mapping방법 이해

가. 거래명세표상의 데이터의 특성(속성유형 기준)



거 래 명 세 표

일련번호 _____

199 년 월 일

_____ 귀하

아래와 같이 계산합니다.

공급자	등록번호			
	상 호		성 명	
	사업장 소재지			
	업 태		종 목	
	전화번호			

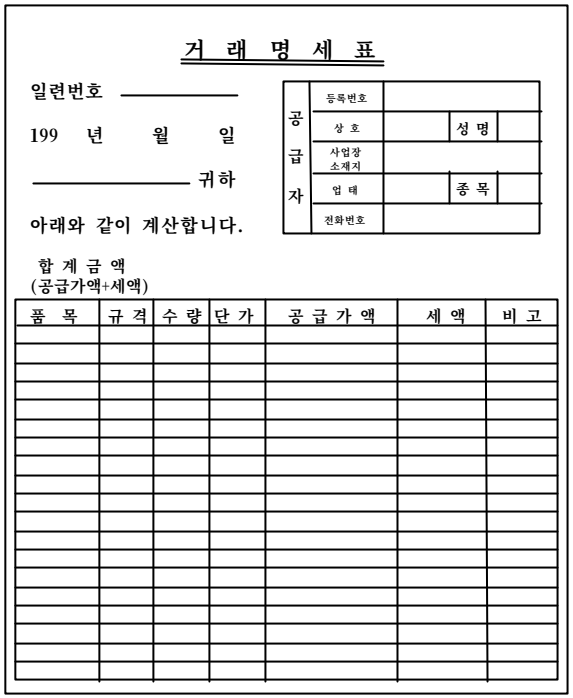
합 계 금 액
(공급가액+세액)

품 목	규 격	수 량	단 가	공 급 가 액	세 액	비 고

데이터의 특성별 분류(속성유형기준)

- **기초속성** : 실제적(자연적)으로 존재하는 데이터
  거래명세표) 거래명세, 년월일, 공급받는자, 공급자, 상호, 성명, 사업장소재지, 업태, 품목, 규격, 수량, 단가, 비고
- **설계속성** : 원래 존재하지 않지만 필요에 따라 설계자가 만든 데이터
  거래명세표) 일련번호, 등록번호, 전화번호
- **추출속성** : 다른 자료로 부터 계산 등의 가공처리를 통해 만들어진 데이터
  거래명세표) 공급가액, 세액, 합계금액

나. 거래명세표상의 데이터의 특성 (실체자격기준)



거 래 명 세 표

일련번호 _____

199 년 월 일

_____ 귀하

아래와 같이 계산합니다.

공급자	등록번호			
	상 호		성 명	
	사업장 소재지			
	업 태		종 목	
	전화번호			

합 계 금 액
(공급가액+세액)

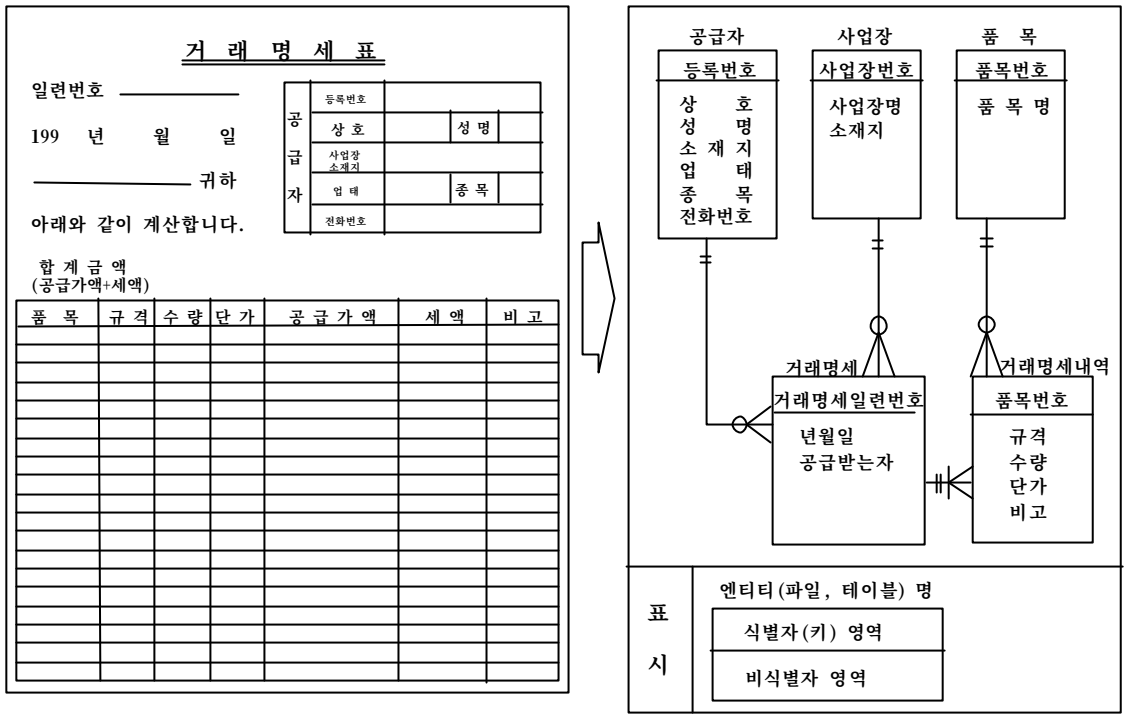
품 목	규 격	수 량	단 가	공 급 가 액	세 액	비 고

데이터의 특성별 분류(실체자격기준)

- 실체의 자격 - 대표값을 가질것 (Uniqueness), 대표값외의 속성을 가질것. 사용자가 DB에 저장 관리하려고 할것. 다른 실체와 관계가 있을 것.
- 실체 - 거래명세, 공급자, 사업장, 품목, 거래명세내역
- 실체별 속성
  - 거래명세) 거래명세,일련번호,년월일, 공급받는자
  - 공급자) 등록번호,상호,성명,사업장소재지,업태,종목,전화번호
  - 사업장) 사업장번호,사업장명,소재지
  - 품목) 품목번호,품목명
  - 거래명세내역) 품목번호,규격,수량, 단가,비고

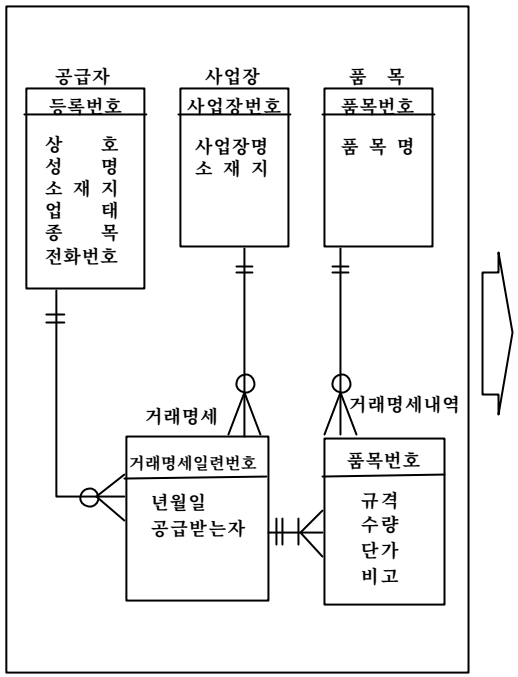
다. 거래명세표의 관계형 데이터 모델(ERD) 도출

< 거래명세표의 논리적 ERD >

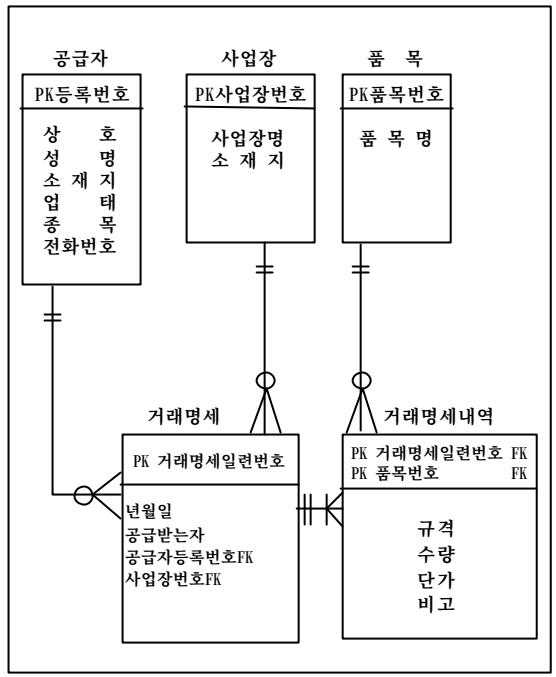


라. 거래명세표의 논리적ERD로부터 물리적ERD 도출

< 거래명세표의 논리적 ERD >

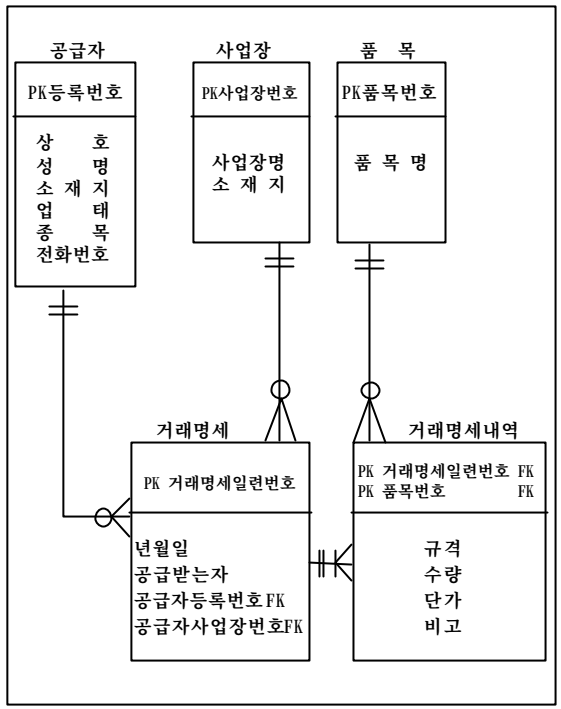


< 거래명세표의 물리적 ERD >



마. 거래명세표의 물리적 ERD로 부터 참조무결성 규칙 도출

< 거래명세표의 물리적 ERD>

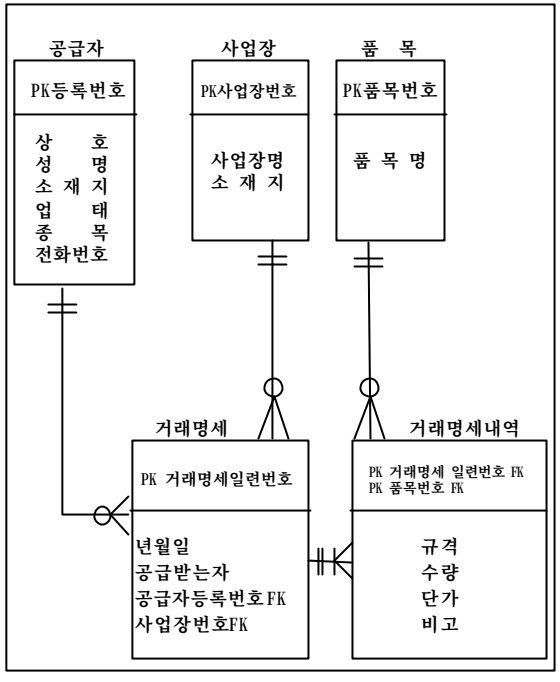


부모실체	관 계	자식실체	참조무결성규칙	
			입 력	삭 제
공급자		거래명세	Dependent	Restrict
거래명세		거래명세 내역	Dependent	Cascade

설명)
- 입력규칙(자식실체기준)
  Dependent: 대응되는 부모실체가 있는 경우에만 자식실체의 입력/수정을 허용
- 삭제규칙(부모실체기준)
  Restrict: 대응되는 자식실체가 없는 경우에만 부모실체의 삭제를 허용
  Cascade: 부모실체의 삭제를 항상 허용하고 동시에 대응되는 자식실체의 모든 건을 자동 삭제

바. 거래명세표의 물리적 ERD상의 거래명세 테이블의 Create

< 거래명세표의 물리적 ERD>



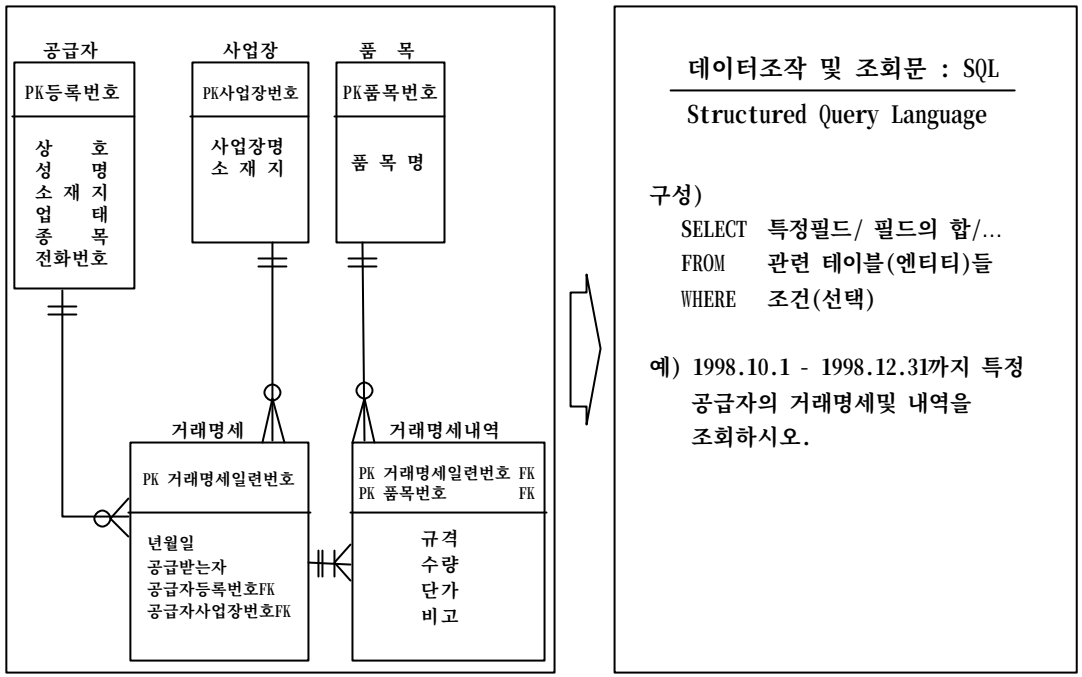
< 물리적 ERD의 거래명세 Entity의 Table로의 전환설계 및 Table Create>

```
CREATE TABLE TRANS
transsrno   NUMBER  CONTRAINT pk_ transPRIMARY KEY
transdate   DATE        DEFAULT SYSDATE;
customer    VARCHAR(10)   CONSTRAINT nn_name NOT NULL
                          CHECK(customer=UPPER(customer)),
cusregno      NUMBER      CONSTRAINT fk_cusregno
                          REFERENCES client(cusregno)
                          ON DELETE Restrict
cuscompno   NUMBER        CONSTRAINT fk_cuscompno
                          REFERENCES complace(cuscompno)
                          ON DELETE Default

    PCTFREE 5 PCTUSED 75
    TABLESPACE human_resource
    STORAGE (INITIAL 8K NEXT 4K
    MINEXTENTS 1 MAXEXTENTS 10
    PCTINCREASE 20) ;
```

사. 거래명세표의 물리적 ERD로 부터의 데이터조작과 조회

< 거래명세표의 물리적 ERD>



데이터조작 및 조회문 : SQL

Structured Query Language

구성)

SELECT 특정필드/ 필드의 합/...
FROM 관련 테이블(엔티티)들
WHERE 조건(선택)

예) 1998.10.1 - 1998.12.31까지 특정 공급자의 거래명세및 내역을 조회하시오.

3. 논리적 데이터 모델링

3.1 데이터 모델

● 정의

데이터의 집합을 기술하는데 사용되는 개념과 데이터를 조작할 수 있는 연산들의 모임

● 종류

·개념적 모델(Conceptual Model)

- 데이터베이스 설계시 사용
- 주어진 실세계로부터 개념의 집합을 명세
- 높은 수준의 추상화에서 실세계를 표현하는 도구
- 실세계를 이해하기 쉽고 해석하기 쉽도록 실세계를 명세

※ 개념적 모델의 평가 기준

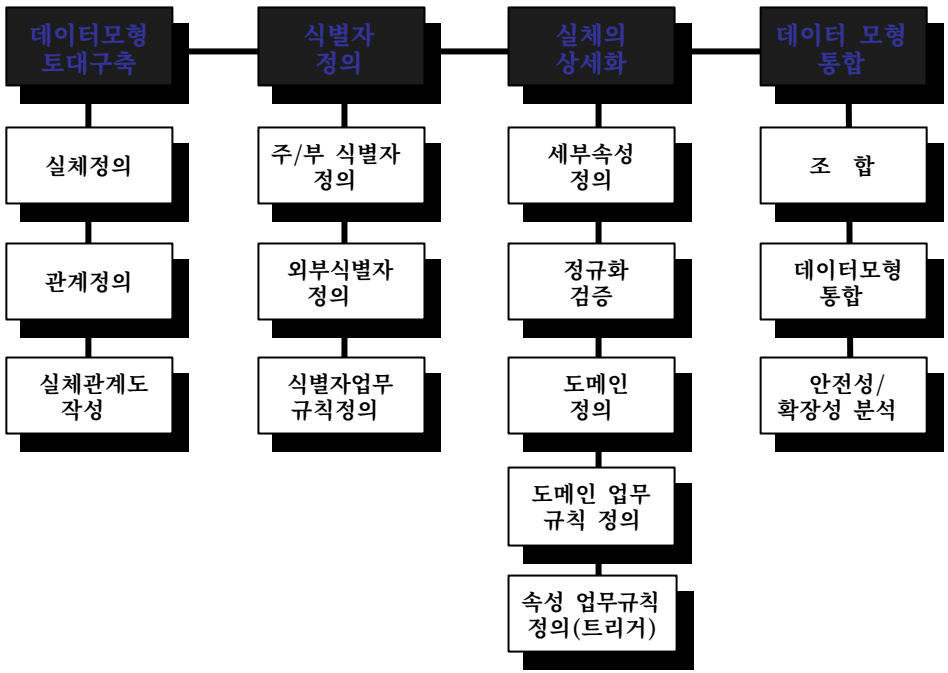
- 표현성 (Expressiveness)
- 단순성 (Simpilicity)
- 최소성 (Minimality)
- 정형성 (Formality)

·논리적 모델(Logical Model)

- 특정 DBMS에 의해 지원됨
- 컴퓨터에 의해 처리될 수 있는 데이터 명세를 지원
- 종류 : 계층형 모델, CODASYL 모델, 관계형 모델

3.2 논리적 데이터 모델링

가. 논리적 데이터 모델링 절차



나. 데이터 모형 토대 구축

1) 실체(Entity) 정의

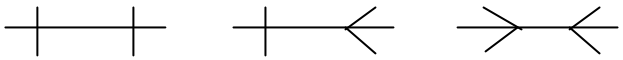
- 실체의 종류
- 실체의 자격조건
- 실체의 예

① 사람(사원(직원, 행원, 공원,...), 계약자(가입자, 회원,...), 이용자(학생, 환자,...))
② 물건(재료(부품, 원자재, 연료, ...), 상품(제품,...), 시설(건물, 창고, 운송센터,...), 지점(영업소, 소매점,...))
③ 사건(계약(수주,발주,...), 작업(공정, 보관, 선전, 광고,...), 사고(재해, 고장,...))
④ 장소(구획(창고, 선반, 진열케이스, 생산라인, ...), 지역(판매구역, 관할구, 선거구,...), 하천, 항만(부두, 선창,...))
⑤ 개념(목표, 계획(지침, 방침, 지표, 판매목표, 생산계획, 판매계획, 인원계획,...), 시간(월, 일, 년, 시각, 시각분할,...), 평가(기준, 지표))
⑥ 금전(예입금(구좌,...), 예산(년간예산, 수정예산, 실행예산,...), 차입(단기, 장기,...), 융자(단기, 장기,...))

2) 관계(Relationship) 정의

● 기수성(Cardinality)

- 1:1, 1:M, M:N 관계
- 해당실체 1건에 대한 상대실체의 기수성을 상대실체쪽에 표기



● 선택성(Optionality)

- 집합의미 (포함, 불포함)
- 1:0 (Optional), 1:1 (Mandatory)
- 해당실체 1건에 대한 상대실체의 기수성을 상대실체쪽에 표기

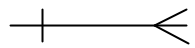
● 관계의 완성 : 기수성과 선택성의 통합 [James Martin]

- 해당실체를 기준으로 기수성의 경우의 수와 선택성의 경우의 수를 합하여 최소값과 최대값의 경우의 수를 구한후 해당실체쪽에 최대값을 바깥쪽에 최소값을 표기한다.
- 상대실체도 유사한 방법으로 표기한다.

3) 실체관계도 작성

● 작도방법

- 사각형의 도형안에 실체명을 기록
- 업무흐름의 진행순서와 관련된 실체는 진행순서를 고려하여 좌에서 우 또는 상에서 하로 중심부에 배열 (“주문”→ “출고”)
- 중심에 배열된 실체와 관계를 가진 연관실체(종속실체)를 가까운 쪽으로 배열 (“주문” : “주문품목”, “출고” : “출고품목”)
- 배열된 실체와 관계를 갖는 핵심실체(Kernal Entity)를 외곽으로 전개 (“주문”, “고객”, “영업담당자”, “창고”, “품목”, “제품”)
- 해당실체의 한 건에 대한 상대실체의 기수성(Cardinality)을 상대로 실체쪽에 표시함으로서 관계의 기수성을 표기 :



- 해당실체의 한 건에 대한 상대실체의 선택성(Optionality)을 상대실체 쪽에 표기함으로서 관계의 선택성을 표기 : ,

● 주요성공요소

- 실체를 식별하고, 관계를 도출한 후 ERD작도법에 맞추어 ERD를 작성
- 업무흐름 및 업무규칙의 ERD작도시 활용

다. 식별자 정의

1) 주식별자 (Primary Identifier) 정의

- 각 실체별로 하나의 주식별자 선택
- 후보 식별자 중 가장 중요한 하나를 주식별자로, 나머지를 대체키로 지정
- Subtype실체의 주식별자는 Supertype실체의 주식별자와 동일하게 선택
- 데이터 이름에 대한 표준약어목록의 이용

2) 외부 식별자(Foreign Identifier)정의

- 두 실체간의 관계를 결정하여 주는 속성으로 관계에 의한 자식실체에 위치하며 부모실
  체의 주식별자가 같은 값을 갖는다.
- 논리적 데이터 모델내의 모든 관계에 관련된 외부 키를 규명한다.

3) 식별자 업무규칙 정의

● 주식별자 업무규칙

- 주식별자는 Unique하며, Not Null이다.
- 주식별자는 한번 설정되면 수정될 수 없다.
- DDL(데이터정의어)에 'Primary Key' 옵션을 주어 표기

● 외부식별자 업무규칙 (참조무결성 규칙)

- 입력규칙(자식실체의 1건을 입력시 관련된 부모실체와의 영향비교)

·Dependent : 대응되는 외부키를 가진 부모실체가 있는 경우에만 자식실체의 입력허용

·Automatic : 자식실체의 입력을 항상 허용하되, 대응되는 외부키를 가진 부모실체가 없을 경우에 새로이 생성시킴

·Nullify : 자식실체의 입력을 항상 허용하되, 대응되는 외부키를 가진 부모실체가 없는 경우 자식실체의 외부키 값을 Null로 만듬

·Default : 자식실체의 입력을 항상 허용하되, 대응되는 외부키를 가진 부모실체가 없는 경우에는 자식실체의 외부키 값을 Default Value로 Set.

·Customized : 외부키에 대응하는 부모실체가 특정한 검증조건이 만족되는 경우에만 자식실체의 입력을 허용

·No Effect : 외부키와 대응되는 부모실체에 관계없이 자식실체의 입력을 언제나 허용

- 삭제/수정규칙 (부모실체의 1건을 삭제/수정시 관련된 자식실체와의 영향비교)

·Restrict : 부모실체의 기본키와 대응되는 외부키를 가진 자식실체가 없는 경우에만 부모실체의 삭제/수정 허용

·Cascade : 부모실체의 기본키와 대응되는 외부키를 가진 자식실체를 모두 삭제/수정

·Nullify : 부모실체의 삭제/수정을 언제나 허용하고 부모실체의 기본키와 대응되는 외부키를 가진 자식실체의 외부키를 'Null'로 Set

·Default : 부모실체의 삭제/수정을 언제나 허용하고 부모실체의 기본키와 대응되는 외부키를 가진 자식실체의 외부키를 'Default Value'로 Set

·Customized : 자식실체의 외부키에 대하여 특정한 검증조건이 만족되는 경우에만 부모실체의 삭제/수정을 허용

·No Effect : 부모실체의 삭제를 언제나 허용

라. 실체의 상세화

1) 속성정의

"속성이란 엔티티를 구성하는 더 이상 분리될 수 없는 정보단위로 식별자류(기본, 대체, 외부 키)와 비식별자(nonkey)로 구분"

- 정보 분석단계에서 수집된 각종자료 참조
- 엔티티, 관계 정의시 파악
- 기존 정보시스템 분석 - 관련 DB나 file의 field
- 속성의 이름을 부여 - 표준화 규칙 사용, 자료사전에 기록
- 표기 예

실체	속성	속성유형	식별자 구분	비고
제품	제품코드 제품명 기대수요 재주문요구량	설계 기초 기초 기초	PK	

2) 정규화

- 목적 : 데이터베이스 성능향상, 자료저장공간 간소화, DB불일치 위험최소화, DB구조 안정성

- 1차정규화
  - 비정규형에서 1차정규형을 만드는 과정
  - 반복 또는 중복되는 속성값을 갖는 속성을 제거
  - 관계형 데이터베이스의 Table Join을 위하여 반드시 실행되어야 함
    (반복 또는 중복되는 속성들의 JOIN이 불가함)

- 2차 정규화
  - 1차정규형에서 2차정규형을 만드는 과정
  - 기본키에 종속되지 않는 속성을 제거
  - 기본키의 일부에 종속되는 속성을 제거
  - 함수적인 종속성(Functional Dependency) : 특정 data가 특정 data와 관련 지어지는 특성
    예) 주문번호, 주문일자, 주문금액 data는 주문 data에 함수적 종속성이 있음

● 3차 정규화
- 2차정규형에서 3차정규형을 만드는 과정
- 기본키가 아닌 속성에 종속적인 속성의 제거

● 4차 정규화
- 3차정규형에서 4차정규형을 만드는 과정
- 특정 속성값에 따라 선택적인 속성의 제거

정규화(Normalization) 예

※ 비 정규형

제품 주문품목

제품코드	제품명	재고수량	주문번호	수출내수 구분	고객코드	사업자 등록번호	고객우선 순위	주문수량	제품코드
110	TV	12,505	94002	X	A031		0	5,000	110
			94073	N	C010	1234567890	3	20	110
165	VTR	16,606	94028	N	C238		2	55	165
			94101	X	A031	0987654321	0	7,700	165
850	Aircon	795	94007	X	Y211		1	500	165
			94036	X	Y211		1	350	850

반복그룹

※ 1차 정규형

제품

제품코드	제품명	재고수량
110	TV	12,505
165	VTR	16,606
850	Aircon	795

주문품목

제품코드	주문번호	수출내수 구분	고객코드	사업자 등록번호	고객우선 순위	주문수량
110	94002	X	A031		0	5,000
110	94073	N	C010	1234567890	3	20
165	94028	N	C238		2	55
165	94101	X	A031	0987654321	0	7,700
165	94007	X	Y211		1	500
850	94036	X	Y211		1	350

※ 2차 정규형

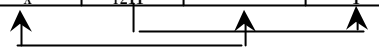
제품

제품코드	제품명	재고수량
110	TV	12,505
165	VTR	16,606
850	Aircon	795

주문품목

제품코드	주문번호	주문수량
110	94002	5,000
110	94073	20
165	94028	55
165	94101	7,700
165	94007	500
850	94036	350

주문번호	수출내수 구분	고객코드	사업자 등록번호	고객우선 순위
94002	X	A031		0
94073	N	C010	1234567890	3
94028	N	C238		2
94101	X	A031	0987654321	0
94007	X	Y211		1
94036	X	Y211		1



※ 3차 정규형

제품

제품코드	제품명	재고수량
110	TV	12,505
165	VTR	16,606
850	Aircon	795

주문

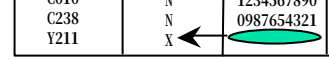
주문번호	고객코드
94002	A031
94073	C010
94028	C238
94101	A031
94007	Y211
94036	Y211

주문품목

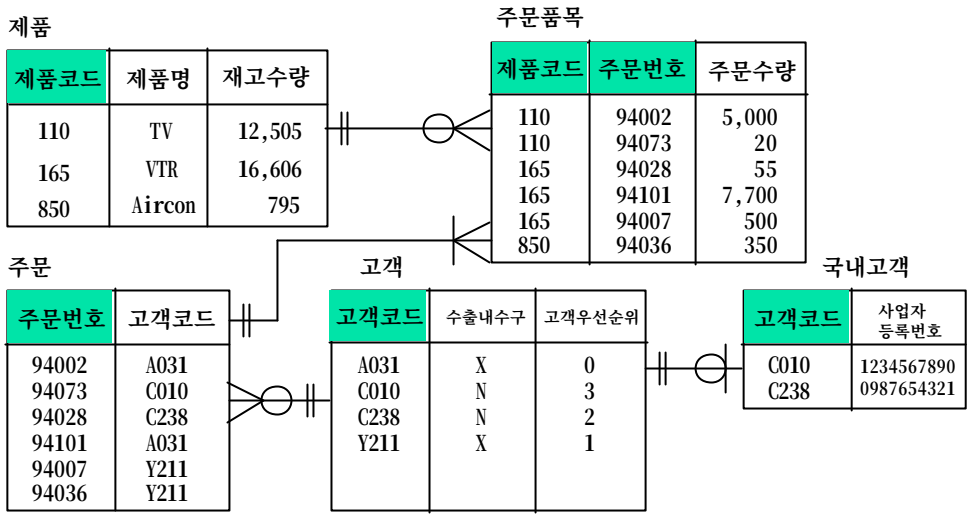
제품코드	주문번호	주문수량
110	94002	5,000
110	94073	20
165	94028	55
165	94101	7,700
165	94007	500
850	94036	350

고객

고객코드	수출내수 구분	사업자 등록번호	고객우선 순위
A031	X		0
C010	N	1234567890	3
C238	N	0987654321	2
Y211	X		0



※ 4차 정규형


제품
제품코드 제품명 재고수량
110 TV 12,505
165 VTR 16,606
850 Aircon 795
주문품목
제품코드 주문번호 주문수량
110 94002 5,000
110 94073 20
165 94028 55
165 94101 7,700
165 94007 500
850 94036 350
주문
주문번호 고객코드
94002 A031
94073 C010
94028 C238
94101 A031
94007 Y211
94036 Y211
고객
고객코드 수출내수구 고객우선순위
A031 X 0
C010 N 3
C238 N 2
Y211 X 1
국내고객
고객코드 사업자 등록번호
C010 1234567890
C238 0987654321

3) 도메인 정의

"도메인이란 각 속성이 가질 수 있는 값에 대한 업무적 제약조건으로 부터 파악된 일련의 특성이다."

● 도메인 특성 :

데이터 형태(Data Type), 길이(Length), 형식(Format/Mask), 허용되는 값의 제약조건(Allowable Value Constraints), 의미(Meaning), 유일성(Uniqueness), 널의 허용(Null Support), 초기값(Default Value)

● 절차

- 각 속성에 대해 도메인 특성을 정의
- 기본키에 대한 도메인 정의
- 외부키에 대한 도메인 정의
- 추출속성에 대한 도메인 정의
- Subtype 기본키의 도메인은 관련된 Supertype 기본키
  ·도메인의 Subset 형태로 정의

※ 도메인정의 표기 예

실체 명 : 구매주문

속성명	타입	길이(n,m)	포맷	범위	유일성	널여부	초기값
주문번호	NUM	7	YYMM999		Unique	Not Null	
공급자번호	CHAR	5	A9999			Not Null	
주문일자	DATE	8	YY.MM.DD				Today
납기일	DATE	8	YY.MM.DD	>=Today		Not Null	
상태	CHAR	1	A	IN (‘C’, ‘E’, ‘D’)		Not Null	‘C’
최근접촉일	DATE	8	YY.MM.DD				
총금액	NUM						

제약조건 : 총금액 = SUM(주문품목.수량.주문품목.단가*(100-주문품목.할인율)/100
WHERE 주문품목.주문번호 = 주문.주문번호

4) 도메인 업무규칙 정의

- **도메인이 갖는 제약조건을 DDL(데이터정의어)상에 표기함으로써 무결성 규칙의 적용을 받는다.**

  예) Create Table Order

  Order-date date not null, default sysdate

  Order-num number(3) not null,

  Customer char(5) not null,

  check customer Between ‘AAAAA’ and ‘CZZZZ’

5) 속성업무규칙정의(Trigger : 연쇄작용)

● 정의

한 실체의 어떤 속성에 대해 입력, 수정, 삭제 발생시 다른 실체나, 같은 실체내의 다른 속성에 미치는 효과를 관리하는 업무규칙을 말하며, 또한 이것은 한 속성에 대해 일어난 사건(Event)에 대하여 연쇄적으로 발생하는 작용이므로 연쇄작용(Triggering Operation)이라고 한다.

● 작성절차

* 속성값에 대한 무결성, 일치성을 유지하는 모든 업무규칙의 연쇄작용을 정의
  - 연쇄작용을 발생시키는 작업(입력,수정,삭제,조회)
  - 작업의 대상(실체명,속성명)
  - 연쇄작용이 일어나게 되는 조건
  - 연쇄작용의 결과로 수행되어질 작업
* 추출속성에 관계된 원시속성(source attribute)의 event에 대한 추출속성의 연쇄작용을 정의
* Sub-type Occurence의 삭제시, 해당 Super-type에 대한 연쇄작용(삭제)을 정의
  (외부키 업무 규칙과는 다른 개념임)
* system time(현재일자 등) 변수의 변화로 유발되는 복수개의 실체에 대한 속성을 포함
  또는 동일 실체에서 복수의 속성을 포함, 연쇄작용을 정의

● Trigger 예

* Trigger 유형
  - creating, debugging, altering, dropping, enabling, and disabling triggers
  - user triggers, All triggers, DBA triggers

※ 속성업무규칙 작성 예

사용자규칙	사건	실체명	속성명	조건	연쇄작용
주문 완료건에 대하여 1년 경과시 매월초에 삭제함.	매월초	구매주문, 주문품목	N/A	상태='E' AND 납기일자<Today-1년>	삭제
납기일이 지나도 일부 품목이 납품되지 않으면 후속조치	매일	주문	납기일	납기일<Today AND 상태 = 'C'>	구매주문, 상태 = 'D'
구매주문, 총금액 추출 알고리즘	INSERT	주문품목	N/A		구매주문,총금액 += 품목금액(수량*단가*(100-할인율)/100)
	UPDATE		수량,단가 할인율		총금액+-=변동된 품목금액
	DELETE		N/A	마지막 품목이 아님	총금액-=품목금액
주문품목, 단가, 할인율 추출	INSERT	주문품목	단가,할인율	공급조건의 단가, 할인율과 불일치	확인 MESSAGE
송장품목, 단가, 할인율 추출	INSERT	송장품목	단가,할인율	주문품목의 단가, 할인율과 불일치	확인 MESSAGE
해당 납품건에 대하여 대금이 청구되었음을 체크한다.	INSERT	송장		해당 납품건이 존재하지 않음	입력취소
구매주문과 주문품목간의 관계	DELETE	주문품목	N/A	마지막 품목	해당 구매주문 삭제
납품과 납품품목간의 관계	DELETE	납품품목	N/A	마지막 품목	해당 납품 삭제
송장과 송장품목간의 관계	DELETE	송장품목	N/A	마지막 품목	해당 송장 삭제

마. 데이터 모형 통합

1) 데이터 모형 조합통합

- 정의
  각 특정 업무 영역별 관점에 의해 설정된 논리적 데이터 모델들을 통합하여 궁극적으로 기업전체의 시각을 유지하는 전사적 데이터 모델로 발전

- 목적
  - 복합적인 논리적 데이터 모형내의 실체유형들을 정확하게 표현
  - 중복성 제거
  - 실체 및 관계 유형간의 불일치성 해결
  - 실체유형간의 새로운 관계나 업무규칙을 추가

- 진행단계
  - 실체유형과 관련 업무규칙의 통합
  - 관계유형과 관련 업무규칙의 통합
  - 속성유형과 관련 업무규칙의 통합

- 실체의 통합
  - 동일한 주식별자와 주식별자 도메인을 갖는 실체를 통합

- subtype의 통합
  - 주식별자의 도메인에 의하여 sub-type이나 super-type에 실체를 통합

2) 데이터 모형 검증

- **검증단계**
  - Group Check
  - User 확인
  - 규칙
  - 실체품질 검증
  - 속성품질 검증
  - 관계품질 검증
  - 완전성 검증
- Group Check
  - Business rule에 의한 완전한 이해와 E-R Modeling에 대한 완전한 이해를 가진 숙련된 분석가가 최선의 답이다.
  - Project 팀 내의 동료끼리 상호 모델을 Check하고 오류를 찾아 본다.
- **사용자 확인**
  - 정기적으로 사용자에게 모델을 제시하면서 확인한다.
  - 사용자를 참여시켜 Error와 누락된 것을 check한다.
- **완전성 검증**
  - 사용자 INTERVIEW, 서류양식, 장표, 보고서 등과 비교점검
  - 향후 입력, 출력보고서가 모두 적용될 수 있는지 점검

4. 관계형 데이터베이스 전환설계

4.1 특성 및 필수성공요소

- 데이터 중심 - 논리적 데이터 모형에 업무규칙을 통합
- 처리지향(Process-Oriented) - 특정기능과 성능목표에의 요구를 감안
- 관계형 DBMS 상품에 의존적
- 사전에 명명규칙(Naming Convension)과 사용서식에 관한 표준안을 확정한 후 단계적 절차로 접근
- 참여인력의 경험과 임무를 신중히 평가하고 적용방법론에 대한 이해를 공유
- 일반적인 설계기법에 제품특성에 따른 특정 DBMS의 설계기법을 추가
- 자료사전을 작성하여 단계별 검토와 논리적 데이터모형과의 상호비교에 활용
- 논리적 데이터모형을 기초로 하여 관계형 데이터베이스 설계작업 수행
- 관계형 데이터베이스의 구현에 구조적 관점과 무결성 관점을 포함
- 관계형 데이터베이스 설계의 조감을 위해 도형화기법 사용

4.2 데이터 사용량 분석

● ERD의 물리적 확장(데이터 구조도)

- 실체건수 / 보관주기
- Row의 Length (최대 길이), 평균길이
- 실체 유형
  - M - 마스터
  - T - 트랜잭션
  - C - 코드
  - S - 집계
  - X - 중간 처리
- 보관 기간
- 기수성 (기수율) : 1대다인 경우 기수율 기록
- 무결성 규칙 (외부키 업무규칙) : 입력/삭제 규칙을 약어로 기록
- 참조시작점 : 다른 실체를 경유하지 않고 최초로 엑세스 되는 실체인 경우, 엑세스 경로가 되는 속성을 기록.

<데이터 구조도 표현방법>

● Data 사용량 분석 결과의 활용

* 물리적 데이터베이스 설계 기초자료
  - 데이터베이스 객체(테이블,인덱스) 초기저장 중간 확보
  - 디스크상의 배치 결정
  - 데이터베이스 할당
* 데이터 분산 설계의 기초 자료
* 성능 향상 튜닝의 기초 자료

※ 실체 분석표 사용

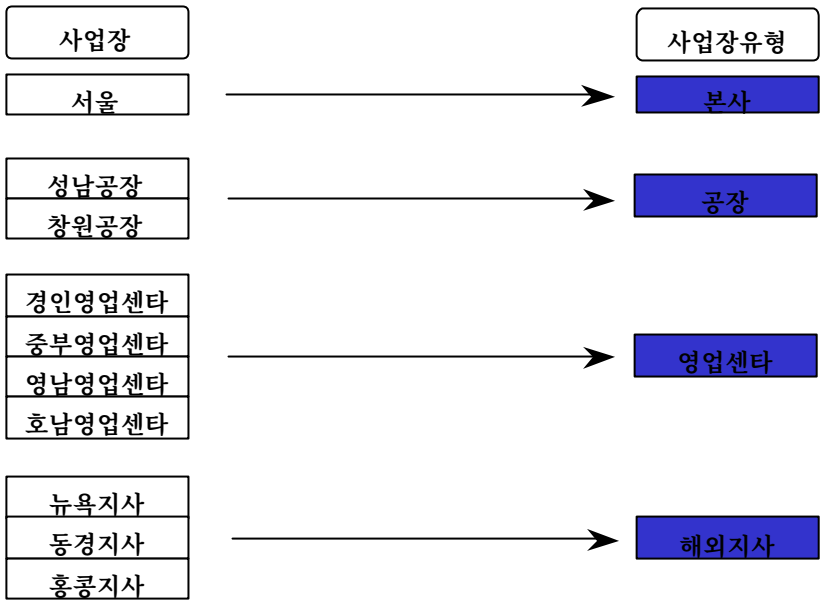
실체명	실체유형	건 수	주 기	최대길이	평균길이	보관주기	비 고
주 문	T	1,000	월	50	50	1년	
생산지시	T	120	월	40	40	1년	
자 재	M	15,000	장기	250	200	장기	
거래처	M	1,200	장기	350	290	장기	
자재재고	S	15,000	1년	550	550	10년	
제조구분	C	20	장기	50	30	10년	

유형 - M : 마스터, T : 트랜잭션, C : 코드, S : 집계, X : 중간처리

4.3 데이터 분산분석

가. 사업장유형

- 목적 : 분산/집중처리 분석의 기준
- 유사한 프로세스와 정보요구의 집합으로 이루어진 논리적 묶음



나. 분산분석표 - 업무활동/사업장 대비표

사업장 / 업무활동	본 사	공 장1	공 장2
주문접수	○		
생산지시	○		
제품생산		○	○
출하지시	○		
제품출하		○	○
대금청구	○		
입금처리	○		

다. 데이터 저장 분산 요건

● C = Central Data

지역 Copy 이전에 갱신되는 중앙 Master

CR = 중앙 Copy

● P = Partitioned Data

동일한 데이터구조이나 지역별로 다른 값

동일구조의 데이터를 해당지역 분만 저장관리

● S = Subset Data

Master의 추출(Extract)

Host 전체데이터의 일부분씩을 각 지역에서 저장관리

● R = Replicated (Duplicated) Data

완전한 복제, 동일한 데이터를 여러 지역에서 저장관리

● O = reOrganized Data

대부분 동일 데이터이나 다른 데이터 구조

원시데이터를 가공, 집계한 형태로 별도 저장관리

● V = Variant (Saparated-Schema) Data

조금 다른 데이터구조이고 다른 값

각 지역별 다른 구조의 원사 데이터를 전체적으로 통합 DB 구축

데이터저장 \ 사업장유형	본사	지역본부	창고
고 객	C		
제 품	CR	R	
인 력	C	V	
부 품	O		P
총계정원장	C		

라. 시스템분산 요건

LEGEND
3 = General (One System)
2 = General (Local System)
1 = Unique Local system

SYSTEM \ 사업장유형	본사	영업센타	영업점	해외지사			
구매	2	2	2				
제품관리	3	3	3	3			
시장분석	3	3	3				
재고관리	3	3	3				
주문처리		1	2				
제품배송		1	2				
시설장비 관리	2	2	2				
판매관리	2	1	2				
매출관리	2	1	2				
고객고충처리	3	3	3				

마. 데이터 분산과 성능

1) 개별적(응용프로그램별) 관점

- 트랜잭션 처리
  - 초당 트랜잭션 수
  - 응답시간
  - 트랜잭션 비용
- 일괄처리
  - 경과시간
  - 프로세서, CPU시간
- 질의 처리
  - 분당 질의수
  - 응답시간

2) 전체적(시스템) 관점

- 불충분한(미약한) 시스템 자원에 부담 최소화
- 기존 하드웨어와 소프트웨어 자원의 최대 활용
- 처리량 극대화
- 연결하고 있거나 동시에 사용중인 사용자 수

" 경쟁우위의 새로운 시스템을 개발하려면, 이러한 2가지 관점간의 균형을 잘 유지하는 것이 중요하다. "

바.데이터 분산 체크리스트

1) 논리적 데이터베이스 디자인 수행
   - 테이블 디자인
   - 테이블간의 관계 결정
2) 테이블의 OWNER 결정
   - 지역 결정을 위한 첫 시도
3) 환경적 제약 요건을 평가
   - 데이터 보안성, 가용성 요구사항 평가
   - 물리적 데이터베이스 크기 고려
4) 트랜잭션 용도 분석 수행
   - 각 테이블에 대한 지역별 접근횟수 결정
   - 각 테이블이 자주 사용되는 지역을 결정
   - 원거리 데이터 엑세스 최소화
   - WAN보다 LAN을 사용
5) 여러 지역에서 공동으로 사용할 테이블의 위치 결정
   - 테이블의 수평적 분할 고려
   - 다중 계층 네트웍에서는 상위 계층에 테이블을 위치
6) 테이블 중복 필요성 평가
   - 데이터 가용 요구사항 고려
   - 코드 테이블과 참조 테이블의 용도 평가
   - 정보의 휘발성 (소멸성, 추가삭제 빈도수), Refresh 빈도 고려
   - 정보처리 요구량 고려
7) 데이터 무결성 평가
   - 테이블간의 무결성 규칙 유지 방안
   - 다중지역 UPDATE 수행 방안
   - 단일 트랜잭션 내에서의 다중지역 UPDATE를 최소화

4.4 논리적 데이터 구조 전환

가. 1단계 : 단위 ENTITY를 TABLE로 전환

- 복수가 아닌 단수형으로 한다.
- 명칭은 30자리 이하의 영문을 한다.
- (대부분 DBMS가 한글을 지원하고 있으나, 이 경우 에플리케이션 성능 저하의 원인이 됨.
- _, $, # 을 제외한 특수문자는 사용할 수 없다.
- 의미있는 명칭을 부여하거나 업무별로 표준화된 명칭을 부여한다.
- 하나의 테이블을 254 column 이하로 구성

나. 2단계 : 속성을 컬럼으로 전환

- ENTITY에 있는 속성들을 사례표의 컬럼명 란에 기록
- 컬럼의 명칭은 속성의 명칭과 반드시 일치할 필요는 없으나 프로그래머와 사용자의 혼동을 피하기 위해 가능한 표준화된 약어를 사용한다.
- SQL의 예약어 (Reserved Word)의 사용을 피한다.
- 가능한 컬럼 명칭은 짧은 것이 좋으며 짧은 명칭은 SQL 해독 시간을 감소시킨다.
- 필수입력 속성은 Nulls/Unique란에 NN을 표시한다.
- 견본 데이터를 입력시킨다.

다. 3단계 : UID를 기본 키로 전환

1) 1단계 : 기본

- 키형태 란에 ENTITY의 UID에 속하는 모든 속성들에 PK를 표시
- PK로 표시된 모든 컬럼들은 Nulls/Unique란에 반드시 NN,U로 표시되어야 함.
- 여러 개의 컬럼으로 UID가 구성되어 있는 경우는 각각의 컬럼에 NN,U1을 표시
- 또 다른 Unique Key가 있다면 NN,U2로 표시

2) 2단계 : PK & FK의 경우

- 테이블에 외부키 컬럼을 포함시킨다.
- PK의 일부분으로 표시
  - ·Nulls/Unique란에 각각 NN,U1을 표시
  - ·키형태란에 PK,FK를 표시
  - ·여러 FK가 있는 경우는 PK,FK1,PK,FK2...
  - ·여러 컬럼으로 구성된 경우 PK,FK1을 각각에 표시
- 추가된 FK 컬럼에 표본 데이터를 추가

* UID : Unique Identifier (유일 식별자)

라. 4단계 : 관계를 외부 키로 전환

1) 1 : 1 관계(Mandatory)

- Mandatory 반대쪽에 있는 테이블의 PK를 Mandatory쪽 테이블의 FK로
- NN 표시

2) 1 : 1 관계(Optional)

- 어느 한쪽의 PK를 다른 쪽의 FK로
- 보다 빈번하게 사용되는 테이블이 FK를 가지는 것이 유리하다.
- FK에 U 표시

3) 1 : 1 순환관계

4) M : 1 관 계

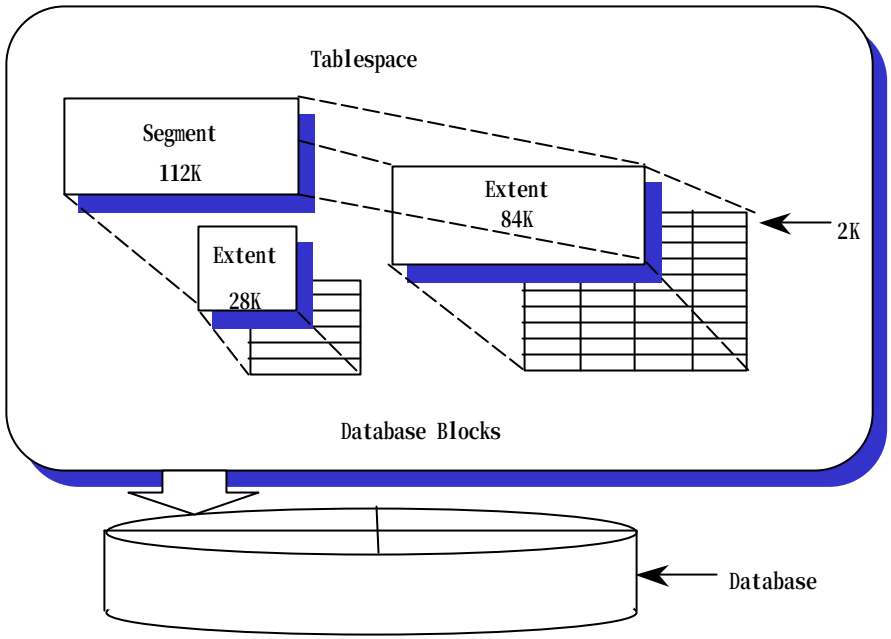
- 1(ONE)에 있는 PK를 M(many)의 FK로
  - ·FK의 명칭을 결정
  - ·키 형태란에 FK 표시
  - ·Nulls/Unique 란에 NN 표시 (must be 관계시)
- 표본 데이터 추가

5) M : 1 순환관계

- 해당 테이블 내에 UNIQUE한 FK 컬럼 추가
- 이 FK는 같은 테이블 내의 다른 로우의 PK 컬럼을 참조하게 됨.
- FK컬럼명칭은 가능한 관계명칭을 반영
- 이 FK는 결코 NN이 될 수 없음.

마. 저장공간 사용량 계획 수립

1) 저장공간 구조



2) Table Space

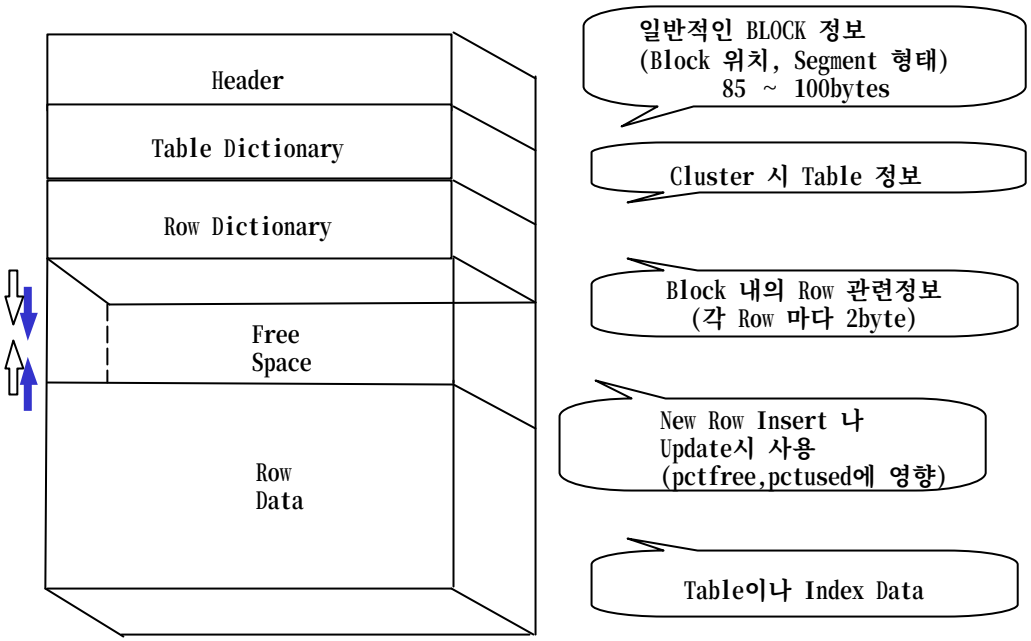
● **Table Space의 용도**

- DB Data에 대한 Disk 할당 통제, DB유저에 대한 쿼터제한, Tablespace에 대한 Online/Offline관계, 부분DB백업 & 복구, 성능향상을 위한 DISK간 배치

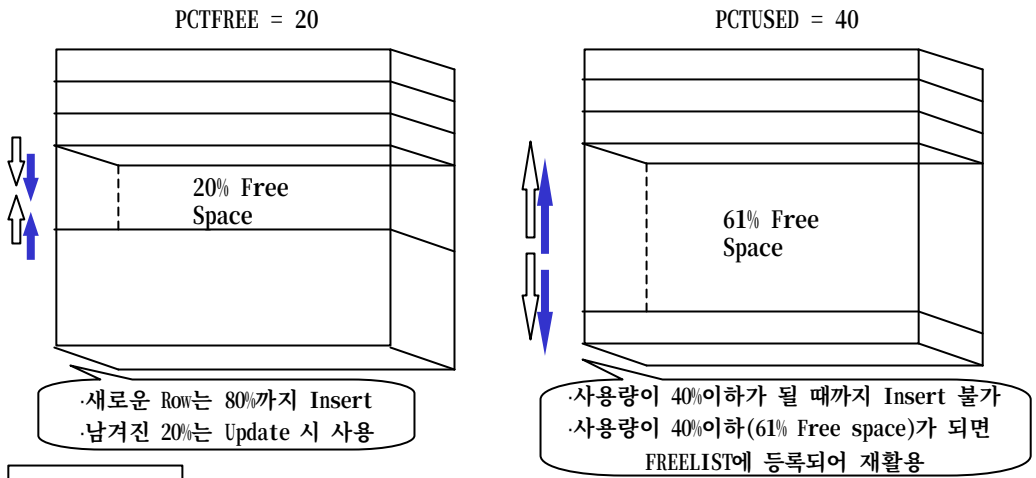
● **Table Space의 종류**

- System, Data, Data_2(정적테이블), Indexes, Indexes_2(정적테이블 사용), RBS, RBS_2(데이터로드사용전문 RBS), Temp, User(특수사용자), Tools

3) Database Block



4) PCTFREE와 PCTUSED

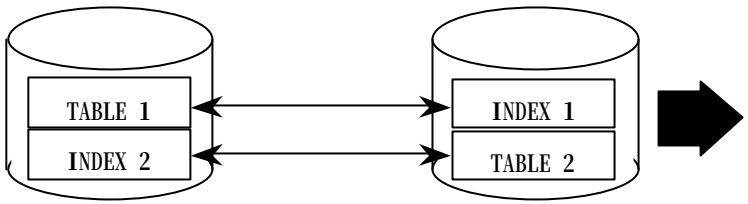


적용사례

상 황	PCTFREE	PCTUSED
Null 컬럼의 입력이나 길이가 증가되는 Update가 수시로 발생되며 한번에 여러 Row가 처리	20	40
대부분의 Update는 길이를 증가시키지 않으며, Insert, Delete는 빈번하나 Update는 드물게 발생	20	50
테이블이 매우 커서 정장공간이 중요하며 대부분의 처리는 Read-Only로 발생	20	50

바. 데이터 객체 구축

● 데이터 객체(테이블, 인덱스)의 디스크 할당



- Disk I/O Contention 감소
- 빠른 Access time

● 테이블별 초기기억공간(Primarly Space), 확장기억공간(Extent), 자유기억공간(Free Space) 지정

- 보관주기내의 데이터전체가 저장될 수 있는 충분한 초기기억공간 할당
- 예상되는 입력/수정에 적합하도록 자유공간 할당

→ - Disk의 효율적 사용
- 과다 Extents 억제

4.5 데이터 무결성 전환

가. 실체무결성 규칙 전환

● 실체의 주식별자에 대한 논리적 성질(Uniqueness, Minimality Disallowance of Null)을 해당 DBMS 구조로 전환

방법1)
```
Create Table Order
        (order_date date not null primary key,
         order_num number(3) not null,
         customer char(5) not null,
                              .
                              .                          )
```

방법2)
```
Create Table Order
        (order_date date not null,
         order_num number(3) not null,
         customer char(5) not null,
                              .
                              .
         primary key(order_date, order_num))
```

방법3)
```
Create Table Order
        (order_date date not null,
          order_num number(3) not null)
      Create unique index on order
         (order_date, order_num)
```

나. 참조무결성 규칙 전환

● 실체간의 참조무결성규칙을 해당 DBMS 구조로 전환
- 종실체 입력규칙
- 주실체 삭제/수정규칙

방법1) DDL에 의한 정의

```
Create Table Order
(order_date date not null,
 order_num number (3) not null,
 primary key(order_date, order_num)
foreign key(customer reference customer on update cascade)
                .
                .                          )
```

방법2) 연쇄작용(Triggering Operation) 구현
: DB Trigger

다. 도메인 무결성 규칙 전환

● 논리적 데이터 모형에서 정의한 무결성을 관리하는 업무규칙을 해당 DBMS에 적용
- 도메인(Domain)
  : Data type, Length, Default value, constraints,
  Uniqueness, null support, Format Picture, fixed or variable

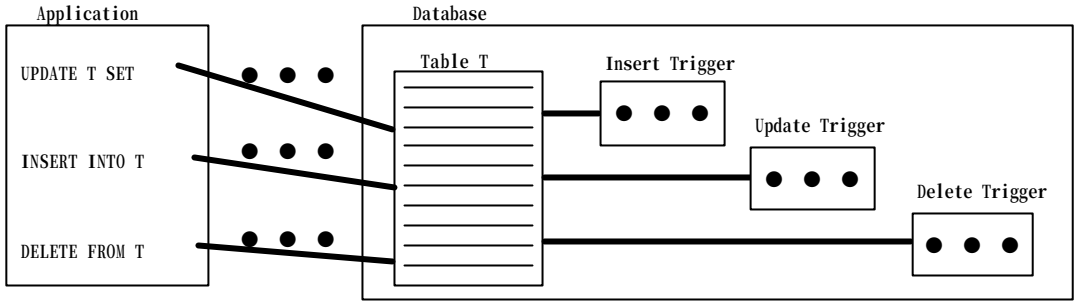
방법1) DDL에 의한 정의

```
Create Table Order
order_date date not null, default sysdate
order_num number(3) not null,
customer char(5) not null,
        check customer between 'AAAAA' and 'CZZZZ'
```

라. 속성 무결성(Trigger) 전환

1) Database Trigger

지정한 Table에 대해 Insert, Update, Delete가 발생할 때 자동으로 실행되는 Stored Procedure



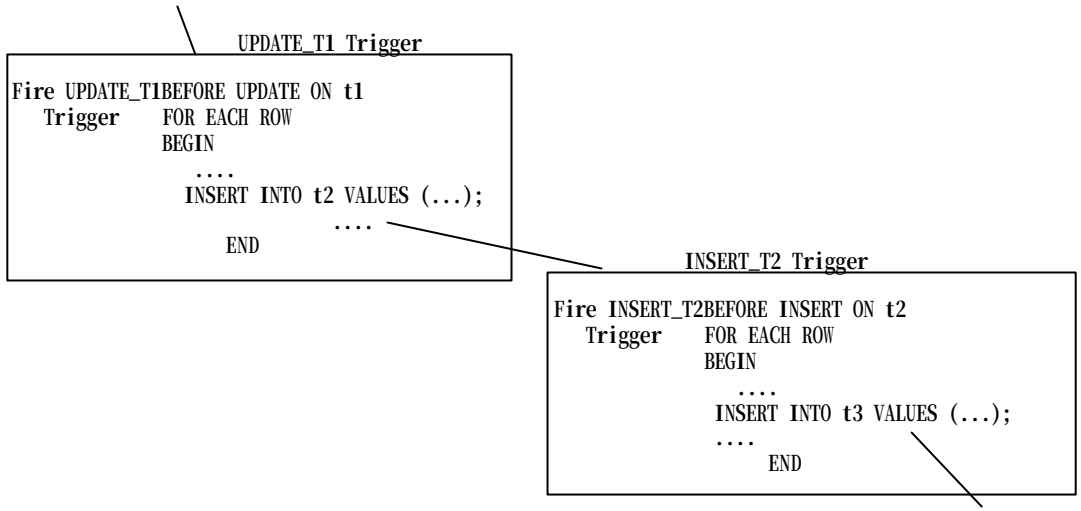
● Trigger의 용도
- 사용자의 업무에 맞춰진 DBMS(Highly Customized DBMS)를 구축하는 방법
- 일반적인 용도

- 계산 컬럼 값을 자동 계산	- 유효하지 않은 트랜잭션 방지
- 보안 관리의 강제화	- 분산 DB상의 참조 무결성 유지
- 복잡한 업무규칙 구현	- 복잡한 감사 방법 구현
- 동시 Table 복사	- Table 접근에 대한 통계 자료 작성

● Trigger의 사용자 규칙 정의
- System Time(현재일자등) 변수의 변화로 인해 유발되는 연쇄작용을 정의
- Subtype Occurence의 삭제시, 해당 Supertype에 대한 연쇄작용(삭제)을 정의
- 추출속성에 관계된 모든 원시속성(Source Attribute)의 수정시 일어나게 될 연쇄작용(추출속성의 수정)을 정의
- 속성의 값에 대한 무결성, 일치성을 유지하는 모든 연쇄작용의 업무규칙을 정의

2) Trigger의 연쇄적 실행

- SQL Statement
- UPDATE t1 SET....;



- Trigger는 꼭 필요한 때에만 사용함.
- 과도한 Trigger의 사용은 복잡한 상호연관으로 인해 대형 Application시 관리의 곤란 발생함

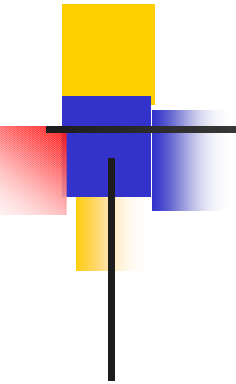
3) Trigger의 종류

- ROW Trigger :
  Trigger 시켜주는 문장에서 받는 값이나 변경된 컬럼의 값을 Trigger Code가 사용할 때 사용함
  예) UPDATE문이 여러 개의 Row에 영향을 미칠때 각 Row의 UPDATE시마다 한번씩 Action을 실행함.
- Statement Trigger
  영향받는 Table Row의 수에 관계없이 Trigger가 한번만 실행됨.
  예) DELETE 명령에서 여러 Row를 삭제해도 관련 Trigger는 한번만 수행할 경우

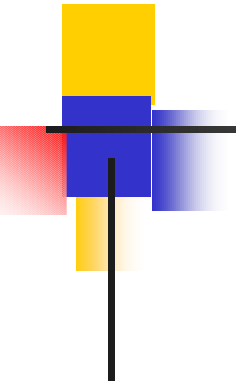
4) Trigger의 구성

- 기본 구성
  Triggering Event / Statement, Trigger 제한, Trigger Action
  예) 재발주 (REORDER) Trigger

```
AFTER UPDATE OF 현재고수량 ON 재고목록
WHEN (new, 현재고수량 < new, 재발주수량)
FOR EACH ROW
DECLARE
        NUMBER x ;
BEGIN
                SELECT COUNT(*) INTO x
                FROM 대기주문
                WHERE 재고번호 = :new.재고번호 ;
IF x = 0
      THEN
          INSERT INTO 대가주문
          VALUES(new,재고번호,new,재발주수량, sysdate);
          END IF ;
END ;
```

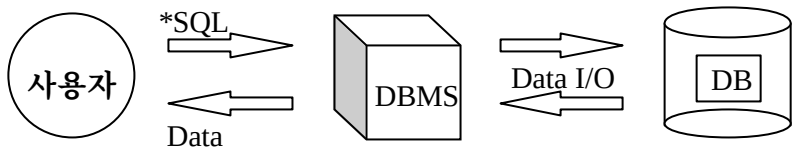


2부. SQL (Structured Query Language)



목 차

1. SQL의 특징



SQL의 가장 두드러진 특징은 다음과 같다.

● **표준화**(All Data Base)
- **모든 관계형** DBMS**에서 동일하게 사용될 수 있다**.
- ISO/ANSI **표준**

● **관계형 모델에 근거**
- **자료 추출에 대한 물리적 구조로부터 독립**
- **수학의 집합론을 바탕으로 개발**

● **비 절차형 언어**(Non-Procedural)

● **고급 언어이고 영어와 비슷한 구조**

● **모든 처리를** SQL**로 수행**(All Users)
- DBA, **일반 사용자**, **프로그래머**
- System **관리**, **보안관리**, DB Access, Dbupdate ...

2. SQL 사용법

SQL을 사용하기 위해서는 SQL과 관련된 함수(Function), 예약어(Reserved Word), 수식과 조건식, 연산자(Operator) 및 SQL명령어(SQL Command) 등의 문법 등에 대하여 자세히 알 필요가 있다.

2.1 함수

2.1.1 단일 행에 대한 수치함수(계속)

함수	리 턴 값
ABS (n)	n의 절대값
CEIL (n)	n보다 크거나 같은 최소의 정수
COS (n)	n의 cosine값
COSH (n)	n의 hyperbolic cosine값
EXP (n)	e(2.71818283 ..)의 n 지수승
FLOOR (n)	n보다 작거나 같은 최대의 정수
LN (n)	n (>0)의 natural logarithm
LOG (m,n)	m의 n logarithm
MOD (m,n)	m을 n으로 나눈 나머지
POWER (m,n)	m의 n 지수승

함수	리 턴 값
ROUND (n,[m])	소수점 m자리에서 반올림 : m이 생략된 경우 디폴트치는 0
SIGN (n)	n<0이면 -1, n=0 이면 0, n>0 이면1
SIN (n)	n의 sine값
SINH (n)	n의 hyperbolic sine값
SQRT (n)	n의 루트근 : n<0인 경우 Null
TAN (n)	n의 tangent
TANH (n)	n의 hyperbolic tangent
TRUNC (n,[m])	소수점 m자리에서 절삭 : m이 생략된 경우 디폴트치는 0

2.1.2 단일 행에 대한 문자함수(계속)

함수	리 턴 값
CHR (n)	숫자 n에 대응하는 문자
CONCAT (char1, char2)	char1과 char2의 연결 값
INITCAP (char)	각 단어의 첫 문자를 대문자로 변환
LOWER (char)	소문자로 변환
LPAD (char1, n, [char2])	char1의 왼쪽에 char2를 덧붙여서 전체 n문자로 되도록 한다. char2가 지정되지 않으면 디폴트로 공백(blank)이 적용된다.
LTRIM (char[,set])	char는 첫 문자부터 확인해서 set에 지정된 문자가 나타나는 동안 해당문자를 제거한다. set의 디폴트치는 공백이다.
NLS_INITCAP (n[,nls_sort])	각국어에 대한 INITCAP함수를 수행한다.
NLS_LOWER (n[,nls_sort])	각국어에 대한 LOWER함수를 수행한다.
NLS_UPPER (n[,nls_sort])	각국어에 대한 UPPER함수를 수행한다.
REPLACE (char, search_str [,replace_str])	search_str의 각 문자를 대응하는 replace_str으로 치환, replace_str이 지정되지 않으면 search_str이 제거된다.
RPAD(char1,n[,char2])	char1의 오른쪽에 char2를 덧붙여서 전체 n문자가 되도록 한다. char2가 지정되지 않으면 디폴트로 공백이 지정된다.
RTRIM (char[,set])	char의 마지막 문자부터 확인해서 set에 지정된 문자가 나타나는 동안 해당문자를 제거한다. set의 디폴트치는 공백이다.
SOUNDEX (char)	영어단어의 음성기호를 나타낸다.

함수	리 턴 값
SUBSTR (char,m[,n])	char의 m번째 문자부터 n개의 문자를 취한다. n의 디폴트치는 마지막 문자까지 이다.
SUBSTRB (char,m[,n])	char의 m번째 byte부터 n개 byte를 취한다. n의 디폴트치는 마지막 문자까지 이다.
TRANSLATE (char, from,to)	char내에서 from문자열을 to문자열로 치환한다.
UPPER (char)	대문자로 변환
ASCII (char)	char의 첫 문제에 대한 십진수를 나타낸다. multi-byte문자열의 경우 첫 byte에 대한 값을 나타낸다.
INSTR (char1, char2[,n[,m])	char1내에서 n번째 문자부터 찾기시작하여 char2가 m번째 나타나는 위치를 찾는다. n,m의 디폴트치는 1이다.
INSTRB (char1, char2[,n [,m])	char1내에서 n번째 byte부터 찾기시작하여 char2가 m번째 나타나는 위치를 찾는다.
LENGTH (char)	char의 문자수
LENGTHB (char)	char의 byte수
NLSSORT (char[,nls_sort])	char의 정렬시 nls의 대응값

2.1.3 그룹함수

함수	리 턴 값
AVG ([DISTINCT|ALL] n)	n의 평균값 : null값의 occurrence는 제외됨
COUNT ([ALL] *)	query / subquery로부터 추출된 행수
COUNT ([DISTINCT|ALL] expr)	expr이 null이 아닌 행의 수
MAX ([DISTINCT|ALL] expr)	expr의 최대값
MIN ([DISTINCT|ALL] expr)	expr의 최소값
STDDEV ([DISTINCT|ALL] n)	n의 표준편차 : null값의 occurrence는 제외됨
SUM ([DISTINCT|ALL] n)	n의 합계
VARIANCE ([DISTINCT|ALL] n)	n의 분산 : null값의 occurrence는 제외됨

2.1.4 날짜함수와 리턴 값

함수	리 턴 값
ADD_MONTHS (d,n)	날짜 d에 n개월을 더한 날짜
LAST_DAY (d)	날짜 d에 해당월의 마지막 날짜
MONTHS_BETWEEN (d,e)	d와 e날짜간의 월수
NEW_TIME (d,a,b)	a표준시에 d일자를 b표준시의 일자로 나타냄
NEXT_DAY (d,char)	날짜 d를 포함해서 그 이후 나타나는 첫 char 요일
SYSDATE	현재 날짜와 시각

2.1.5 날짜의 절삭 / 반올림함수

함 수	리 턴 값
ROUND (d[,fmt])	저장된 fmt 단위(포맷모델)로 날짜 d를 반올림
TRUNC (d[,fmt])	저장된 fmt 단위(포맷모델)로 날짜 d를 절삭

2.1.6 변환함수와 리턴 값(계속)

함수	리 턴 값
CHARTOROWID (char)	char에 대한 CHAR값을 ROWID로 변환
CONVERT (char, dest_char_set [,source_char_set])	source_char_set 표현을 dest_char_set 표현으로 변환
HEXTORAW (char)	16진수 char를 이진 RAW 값으로 변환
ROWTOHEX (row)	RAW타입의 이진 raw 값을 CHAR 타입의 16진수로 변환
RAWIDTOCHAR (rowid)	ROWID 형태의 rowid 값을 char 값으로 변환
TO_CHAR (expr[,fmt[, 'nls_num_fmt]])	NUMBER나 DATA값 expr을 fmt 형식의 CHAR로 변환 : fmt가 생략되면 DATA는 디폴트 형식으로 NUMBER는 유효자리를 모두 나타낼 수 있도록 변환된다.
TO_DATE(char[,fmt[, 'nls_lang']])	fmt 형식의 char를 DATA값으로 변환 : fmt를 생략하는 경우 char는 디폴트 날짜 형식이어야 한다.

함수	리 턴 값
TO_MULTI_BYTE (char)	지정된 single-byte char를 대응되는 multi-byte문자로 변환
TO_NUMBER (char,[,fmt [,'nls_lang']])	숫자를 나타내는 char 문자를 숫자로 변환
TO_SINGLE_BYTE(char)	지정된 multi-byte char를 대응되는 single-byte문자로 변환

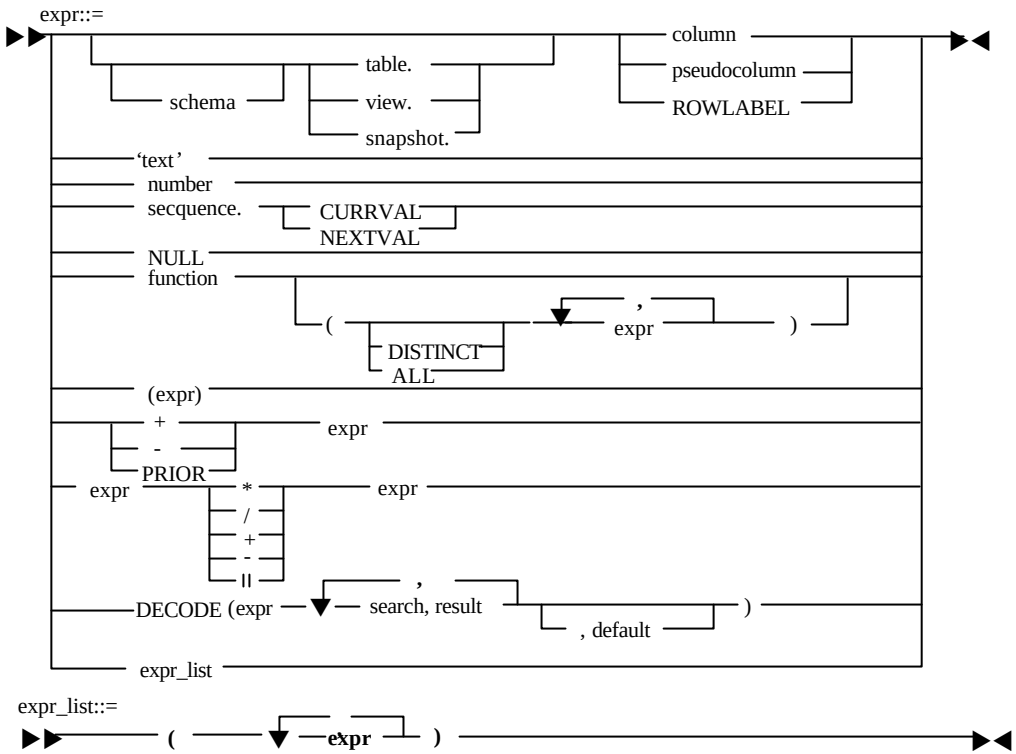
2.1.7 기타함수와 리턴 값

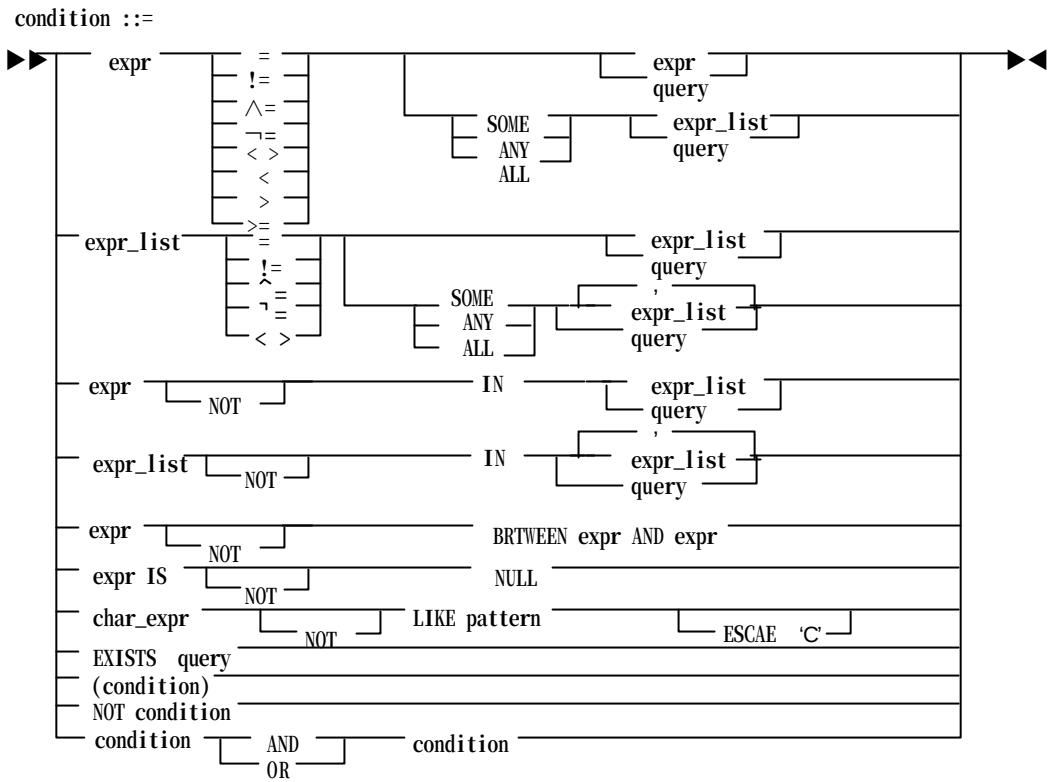
함수	리 턴 값
DECODE (expr,search1,return1 [,search2,return2]...[default])	expr이 search n과 같으면 return n을 나타내고, 같지 않으면 default를 나타낸다.
DUMP (expr[,diplay_format [,start_position[,length]]])	expr의 DBMS 내부 형식을 나타낸다.
GREATEST (expr[,expr]...)	expr중 최대값을 나타낸다.
LEAST (expr[,expr]...)	expr중 최소값을 나타낸다.
NVL (expr1,expr2)	expr1이 null이면 expr2를 나타낸다.
UID	현재 user에 대한 유일한 번호를 나타낸다.
USER	현재 user명을 나타낸다.
USERENV ('option')	현재 세션에 대한 option 정보를 나타낸다. option은 ENTRYID, SESSIOND, TERMINAL, LANGUAGE, LABEL이 지정될 수 있다.
VSIZE (expr)	expr의 DBMS 내부표현시 byte수를 나타낸다.
SYSDATE	현재 날짜 / 시간

2.2 SQL 예약어

ACCESS*	DEFAULT*	INTEGER	OPTION*	START*
ADD*	DELETE*	INTERSECT*	OR*	SUCCESSFUL
ALL*	DESC*	INTO*	ORDER*	SYNONYM
ALTER*	DISTINCT*	IS*	PCTFREE*	SYSDATA
AND*	DROP*	LEVEL*	PRIOR*	TABLE
ANY*	ELSE*	LIKE*	PRIVILEGES	THEN*
AS	EXCLUSIVE	LOCK	PUBLIC*	TO*
ASC*	EXISTS*	LONG	RAW	TRIGGER
AUDIT	FILE	MAXEXSTENTS	RENAME*	UID
BETWEEN*	FLOAT	MINUS*	RESOURCE*	UNION*
BY*	FOR*	MODE	REVOKE	UINQUE*
CHAR*	FROM*	MODIFY	ROW	UPDATE*
CHECK*	GRANT*	MOAUDIT	ROWID	USER
CLUSTER*	GROUP*	NOCOMPRESS*	ROWLABLE	VALIDATE
COLUMN	HAVING*	NOT*	ROWNUM*	VALUES
COMMENT	IDENTIFIED*	NOWAIT	ROWS	VARCHAR*
COMPRESS*	IMMEDIATE	NULL*	SELECT*	VARCHAR2*
CONNECT*	IN*	NUMBER*	SESSION	VIEW*
CREATE*	INCREMENT	OF*	SET*	WHENEVER
CURRENT*	INDEX*	OFFLINE	SHARE	WHERE*
DATA*	INITIAL	ON*	SIZE*	WITH*
DECIMAL*	INSERT*	ONLINE	SMALLINT	

2.3 SQL의 수식과 조건식 규칙




condition ::=
expr
=
!=
^=
¬=
< >
<
>
>=
expr
query
SOME
ANY
ALL
expr_list
query
expr_list
=
!=
^=
¬=
< >
expr_list
query
SOME
ANY
ALL
,
expr_list
query
expr
NOT
IN
expr_list
query
expr_list
NOT
IN
,
expr_list
query
expr
NOT
BRTWEEN expr AND expr
expr IS
NOT
NULL
char_expr
NOT
LIKE pattern
ESCAE 'C'
EXISTS query
(condition)
NOT condition
condition
AND
OR
condition

2.4 연산자

2.4.1 수치연산자

연산자	기 능
+ - (부호연산자)	수식의 양, 음을 지정
* /	곱셈/나눗셈
+ -	덧셈/뺄셈

2.4.2 논리연산자

연산자	기 능
‖	문자값의 연결 연산
NOT	논리식의 결과를 반대로 결과한다.
AND	둘 이상의 논리식의 연결연산. 각 논리식이 모두 참이면 참을 결과하고, 그렇지 않으면 거짓을 결과한다.
OR	둘 이상의 논리식의 연결연산. 하나의 논리식이라도 참이면 참을 결과하고, 그렇지 않으면 거짓을 결과한다.

2.4.3 비교연산자

연산자	기 능
=	같음을 연산
!= ^= ¬= <>	같지 않음을 연산
> >= < <=	큼, 크거나 같음, 작거나 같음을 연산
IN	지정된 집합이나 subquery의 member중의 하나와 같음을 비교. '= ANY '와 같은 연산
NOT IN	지정된 집합이나 subquery의 어느 member와도 같지 않음을 비교. '!= ALL '과 같은 연산
ANY,SOME	열거된 식이나 subquery에서 추출된 하나 혹은 하나 이상의 값에 대해 만족하면 참
ALL	열거된 식이나 subquery의 모든값에 대해 만족하는지 연산
[NOT]BETWEEN x AND y	x보다 크거나 같고(작고), y보다 작거나 같은지(큰지)비교
[NOTE] EXISTS	subquery에서 최소 한행을 추출하면(한행도 추출하지 않으면)참
[NOT]LIKE p [ESCAPE'C']	지정된 문자열(p) 패턴을 만족하는지 비교 패턴 '% '는 0혹은 하나 이상의 임의의 문자와, '_ '는 반드시 하나의 문자와 비교된다.
IS[NOT]NULL	NULL인지(아닌지) 비교

2.4.4 SELECT 문장들간의 집행연산자

연산자	기 능
UNION	두 쿼리의 결과를 결합 : 각 쿼리의 서로 다른 행들을 결과
UNION ALL	두쿼리의 결과를 집합 : 각 쿼리의 모든 행들을 결과
INTERSECT	두쿼리의 결과를 결합 : 각 쿼리의 공통 행들을 결과
MINUS	처음 쿼리의 결과행에서 다음 쿼리의 결과행을 제외한 나머지행의 결과

2.4.5 기타 SELECT 문 연산자

연산자	기 능
(+)	outer join을 수행할 열명뒤에 지정
*	테이블이나 뷰로부터 추출한 열명대신 지정되어 모든열의 의미
PRIOR	select문을 통해 상하구조(tree structured)의 정보를 추출시 arent-child관계(logical한 관계가 아니라 실제 추출순서의 우선을 의미한다.)
ALL	쿼리결과행의 중복을 허용한다(default)
DISTINCT	쿼리결과행의 중복을 제거한다.

2.5 SQL 명령어

2.5.1 명령어 종류

- Data Definition Language
  - CREATE, ALTER, DROP, RENAME
- Data Manipulation Language
  - (SELECT), INSERT, UPDATE, DELETE
  - COMMIT : **논리적으로 서로 연관된 일련의 데이터 처리문을 영구히 기록한다.**
  - ROLLBACK : **전체 트랜젝션 또는** savepoint **이후의 트랜젝션을 취소하여 원래의 상태대로 복귀시킨다.**
- Data Control Language
  - GRANT, REVOKE, AUDIT, LOCK, ROLE, SESSION**等**
- PL/SQL : Oracle
- Data Definition Language

2.5.2 Table 생성문

테이블의 최고 컬럼 수 : 250

(1) create table sawon

(sawon_id char(7) not null,

sawon_name hname,

...

sawon_sex char(1) not null)

(2) create table jiyok as select * [ALL/DISTINCT]

from jijom where samusil_nbr > 3

- 1. **기존의 테이블로부터 새로운 테이블의 구조를 복사한다.**
  2. **데이터가 존재한다면, 자동으로 데이터도 복사한다.**
- **빈 테이블의 생성은** false **조건절을 사용한다.**
- **디폴트, 트리거, 인덱스 등은 새로운 테이블에 복사되지 않는다.**

2.5.3 Insert, Update, Delete, Truncate

- insert sawon(sawon_id, sawon_kid)
  values('4312', '5703121243213')
- insert sawon select * from sawon_master
- update sawon set sawon_name = '김성수'
  where sawon_kid = '5703121243213'
- update book set price = price * 1.4
  from book b, jijom j
  where b.id = j.id and j.jiyok = '인천'
- delete book where book.id = '134132'
- truncate table 이름 : table owner만 가능

2.5.4 Order By

● **조회의 결과를 순차적(디폴트) 또는 역순으로 나타냄**

● Order By

- **하나의 열**
- **다수의 열**
- **표현식**
- **실제 조회 결과로 나타나지 않는 열**

SELECT id, name, kid FROM sawon

ORDER BY id (ASC), kid DESC

SELECT id, name, kid FROM sawon

ORDER BY jikchek DESC

2.5.5 Group By, Having

● select 부서 buseo, min(sal), max(sal) from sawon

where jichek = '부장' group by buseo

부서	min(sal)	max(sal)
판매	3000	7000
개발	4200	5500
관리	3800	5500

● select 부서 buseo, min(sal), max(sal) from sawon

where jikchek = '부장' group by buseo

having min(sal) < 4000

부서	min(sal)	max(sal)
판매	3000	7000
관리	3800	5500

2.5.6 Like Wildcard, String / Data 함수

● like : pattern 또는 wild card를 포함한다.

- select * from... where jiyok_name like "경상%"
- % : 0, 1,..., _ :1, [abc], [a-c], [^a]

例 : SELECT * FROM sawon WHERE name LIKE '_인수'
SELECT * FROM sawon WHERE name LIKE '김%'

● string

- Ora : substr('abcde', 2, 2) Syb : substring('abcde' 2, 2)
right('abcde', 3), ltrim, ···
- lower, upper, ascii, ···

● date

- Ora : add_months(d, n), last_day(d), months_between(d1, d2),
new_time(d, z1, z2), next_day(d, char), round(d[, fmt],
sysdate, trunc(d[, fmt])
- Syb : getdate(), datename(datepart, date),
datepart(datepart, date), datediff(datepart, date, date)
dateadd(datepart, number, date)

3. SQL 실습-I

♣예제 테이블

① 고객 테이블 (테이블 명 : Customer)

cust_no	cust_name	cust_address
C005	서울상회	서울 종로구 수송동
D010	부산상회	부산 중구 남포동
G001	광주상회	광주 중구 금남로
R001	마산상회	마산시 마산동

② 수주 테이블 (테이블 명 : Order)

cust_no	approve_no	order_date
C005	1	940807
C005	2	940901
D010	1	940728
G001	1	940910
K005	1	940830

③ **수주 명세표 (테이블 명** : Order_detail)

cust_no	approve_no	ser_no	goods_code	amount
C005	1	1	PR1	20
C005	1	2	PX0	15
C005	2	1	QP1	10
C005	2	2	SO0	5
D010	1	1	PX0	30
D010	1	2	SO0	6
G001	1	1	PX0	25

④ **상품표 (테이블 명** : Goods)

goods_code	goods_name	unit_price
PR1	**프린터 1형**	300
PX0	**프린터 3형**	250
QP1	**디스크 1형**	910
SO0	**시스템 1형**	4500

♣문제

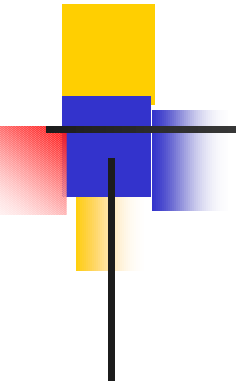
◎ 다음의 SQL문에 대한 답을 작성하시오.

1) SELCT cust_no, order_date FROM Order
   WHERE order_date >= '940801'
      AND order_date <= '940831'

2) SELECT goods_name, unit_price, unit_price * 0.9
     FROM Goods
   WHERE goods_name LIKE '디스크%'

3) SELECT cust_no, MAX(order_date)
     FROM Order
  GROUP BY cust_no
  HAVING COUNT(*) >= 2

4) SELECT goods_code, COUNT(*), AVG(amount),
          SUM(amount)
     FROM Order_detail
   WHERE goods_code LIKE 'P%'
  GROUP BY goods_code



```
5) SELECT   *  FROM  goods
    WHERE   unit_price <=1000
   ORDER BY unit_price DESC

6) SELECT cust_no, approve_no, COUNT(*)
        FROM Order_detail
   GROUP BY cust_no, approve_no
   ORDER BY cust_no, approve_no

7) SELECT  customer.cust_no, cust_name, approve_no,
                order_date
        FROM Cstomer, Order
    WHERE  Customer.cust_no = Order.cust_no

8) SELECT  approve_no, Order_detail.cust_no,
                amount * unit_price
        FROM  Order_detail, Goods
    WHERE  Order_detail.goods_code =
               Goods.goods_code
          AND  cust_no = 'D010'
```

9) SELECT goods_name, amount
FROM Customer x, Order_detail y, Goods z
WHERE cust_nmae = '부산상회'
AND x.cust_no = y.cust_no
AND y.goods _code = z.goods_code
ORDER BY 2

10) SELECT goods_name
FROM Goods
WHERE goods_code IN (
SELECT x.goods_code
FROM Order_detail x, Customer y
WHERE cust_name ='부산상회'
AND x.cust_no = y.cust_no)

11) SELECT goods _name, unit_price
FROM Goods
WHERE unit_price > ALL(
SELECT unit_price
FROM Goods
WHERE goods_name LIKE '프린터%')

12) SELECT cust_no FROM Order_detail
WHERE goods_code = 'PX0'
AND amount > SOME(
SELECT amont FROM Order_detail
WHERE cust_no = 'C005'
AND goods_code = 'PX0')

13) SELECT cust _name FROM Customer x

WHERE 'PRI' IN (

SELECT goods_code

FROM Order_detail y

WHERE y.cust_no = x.cust_no

AND goods_code = 'SO0')

14) SELECT goods_code

FROM Goods

WHERE unit_price >= 1000

UNION

SELECT goods_code

FROM Order_detail

WHERE amount >= 20

15) SELECT x.cust _no, x.approve_no, x.order_date,
ser_no, goods_code, amount
FROM Order x, Order_detail y
WHERE x.cust_no = y.cust_no
AND x.approve_no = y.approve_no
AND x.cust_no = ‘D010’

16) CREATE VIEW Order_Inspect
(cust_no, approve_no, order_date) As
SELECT cust_no, approve_no, order_date
FROM Order
WHERE order_date < ‘940901’

17) (SELECT * FROM Goods) INTERSECT
(SELECT goods_code FROM Order)

18) (SELECT goods_code FROM Goods) EXCEPT
(SELECT goods_code FROM Order)

19) SELECT goods_code FROM Goods x
WHERE EXISTS (
SELECT * FROM Order y
WHERE x.goods_code = y.goods_code)

4. SQL 실습-II

♣예제 테이블

① **테이블명** : SAWON(사원)

② **컬럼명 및 데이터**

SABEON	NAME	BUSEO_CODE	MANAGER	SALARY	BONUS
1001	이영신	00	0	3,000,000	3,000,000
1002	진상효	03	2	1,500,000	1,500,000
1003	박래영	02	2	1,300,000	1,300,000
1004	문지영	05	1	1,300,000	1,300,000
1005	양명성	02	3	1,100,000	1,100,000
1006	조연주	02	3	1,100,000	1,100,000
1007	김수봉	03	3	1,000,000	1,000,000
1008	이종인	03	3	1,000,000	1,000,000
1009	이정현	04	1	1,000,000	1,000,000
1010	최선묵	01	2	900,000	900,000
Char(4)	Char(10)	Char(2)	Char(4)	int	int

③ **테이블명** : BUSEO(부서)

④ **컬럼명 및 데이터**

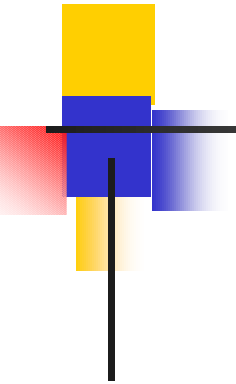
BUSEO_CODE	BUSEO_NAME	LOCATE
00	SI본부	본사
01	교육연구부	연구소
02	컨설팅부	본사
03	S/W 개발부	공장
04	관리부	본사
05	영업부	영업소
06	기획관리부	본사
Char(4)	Char(16)	Char(16)

♣문제

◎ 다음 문제에 대하여 SQL문을 작성하시오.

1) 사원테이블의 내용을 모두 출력
2) 사원테이블의 이름, 월급, 상여금을 출력
3) 사원의 평균급여계산 (사원별)
4) 사원번호가 '1003'번인 사원을 출력
5) 월급이 '1,100,000' 이상인 사원만을 출력
6) 부서코드가 '02'이고 급여가 '1,100,000' 이상인 사원만을 출력
7) 부서코드가 '02'이거나 급여가 '1,100,000' 보다 크고
   부서코드가 '03'도 아니고 급여가 '1,300,000' 이상도 아닌 사원 출력
8) '김'씨 성을 가진 사람만 출력 (메타문자(%_)와 LIKE 사용)
9) 부서코드가 '02'보다 크고 '05'보다 작은 부서의 사원 출력
   (BETWEEN AND사용)
10) 부서별 급여합계계산 (SUM함수 사용)
11) 부서별 월급의 최대 수령자와 최소 수령자 계산 (MAX함수 사용)

12) 월급종류별 인원수 계산 (COUNT(*) 사용)
13) 부서인원이 3명 이상인 부서의 급여합계 (COUNT(*), SUM함수 사용)
14) 급여가 1,100,000 이상인 사원이 2명 이상인 부서의 급여합계
15) 사원 성명순으로 출력
16) 부서코드순으로, 급여 작은 순으로 출력
17) 급여가 1,100,000 보다 작은 사원과, 1,500,000보다 큰 사원의 출력
(UNION ; 합집합)
18) 부서별로 가장 월급이 많은 사원의 출력 (Subquery 및 MAX함수 사용)
19) 각사원에 해당하는 부서명의 출력 (JOIN)
20) 급여가 1,100,000 보다 큰 사원의 부서명, 이름, 월급 출력
21) 새로운 사원에 대한 정보의 추가 (1011, 서희석, 500,000원)
22) 전체 사원의 보너스를 10,000,000으로 수정
23) 영업부사원의 보너스를 100,000,000으로 수정
24) 사원번호가 1001인 사원을 삭제
25) 부서명이 '리부'로 끝나는 사원 중 급여가 1,000,000 보다 큰 사원의 삭제



3부. 최신 데이터베이스 이슈

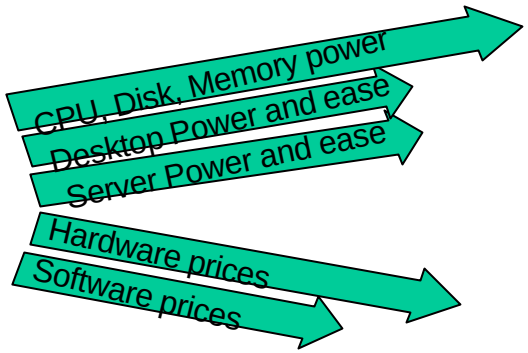
목 차

1. **Data Warehouse**
2. **Data Mart**
3. **CRM (Customer Relationship Management)**

1. Data Warehouse

1.1 출현 배경

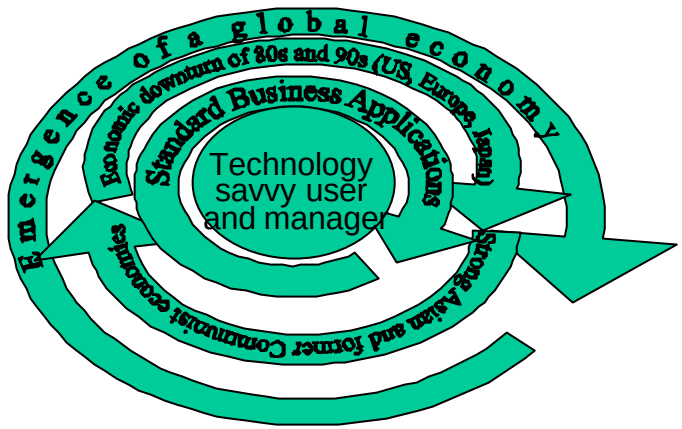
기술환경의 변화



- 프로세서의 강력한 처리속도
- 값싼 디스크, 수백 기가 충분히 저장
- 강력해진 데스크 탑 -> 분석 툴로서의 자리 굳힘
- 서버 S/W -> 비용 Down, 더욱 강력해지고 관리용이
- H/W, S/W 가격의 급격한 Down

기업환경의 변화

- "지구촌 경제" 대두
- 표준 기업 어플리케이션 등장
- 기술적으로도 잘 아는 End-User
- 정보기술(IT) : 기업의 핵심 전략자산으로 인식

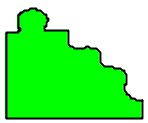


1.2 특징

주제 중심(Subject-Oriented)

Operational

공동구매 

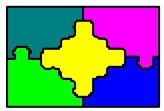
일반경매 

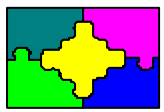
일반거래 

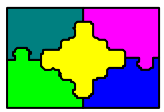
즉시구매 

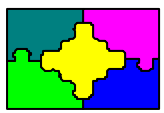
기능/프로세스 중심

Data Warehouse

 회원

 페이지뷰

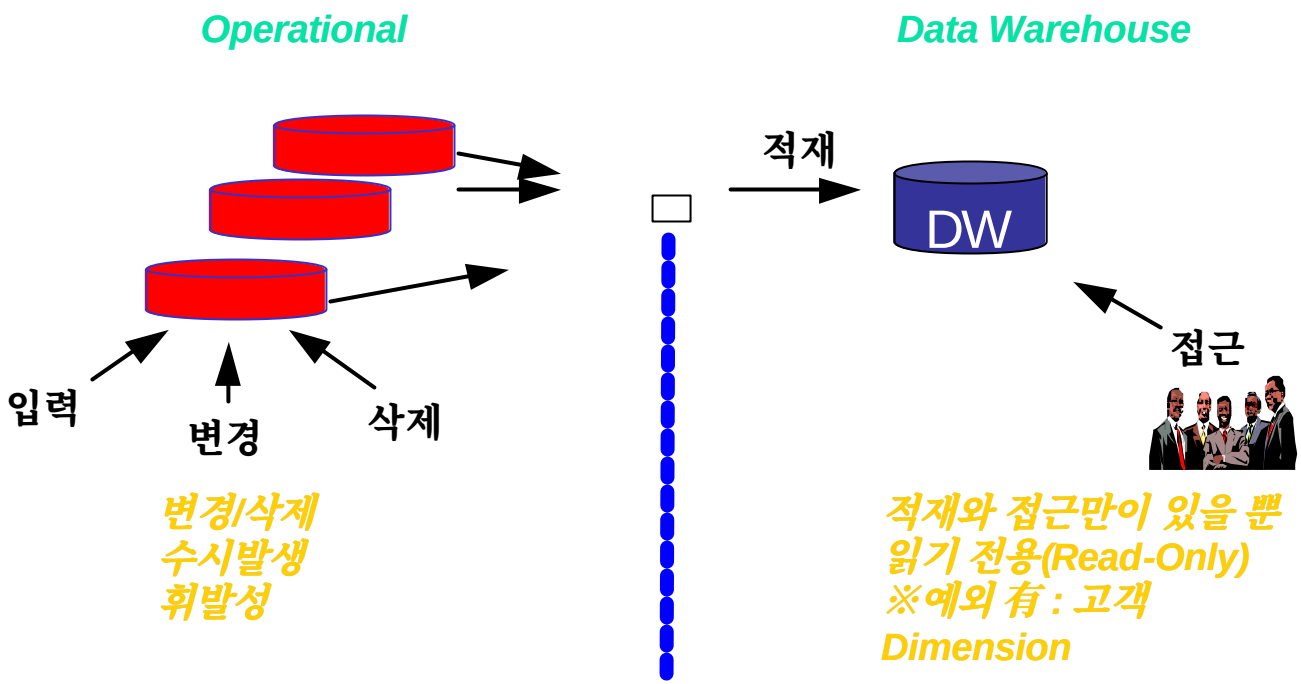
 주문

 클레임

주제영역(=관심영역) 중심

1.2 특징

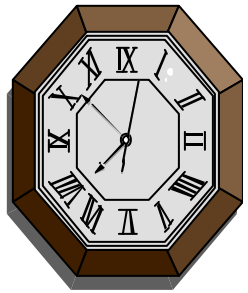
비 휘발성 (Non-Voratile)



1.2 특징

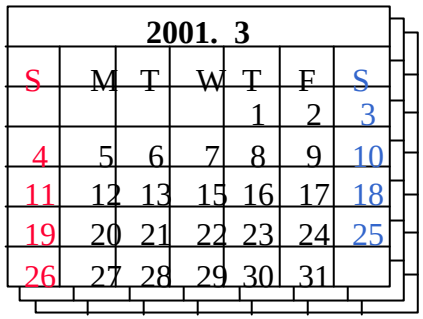
시계열성 (Time_Variant)

Operational



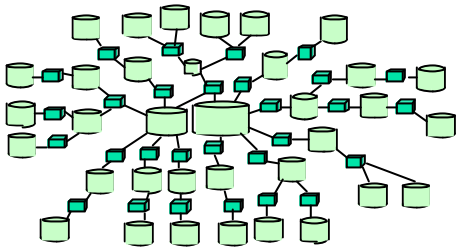
갱신
키에 시간요소 특별히 고려하지 않음

Data Warehouse



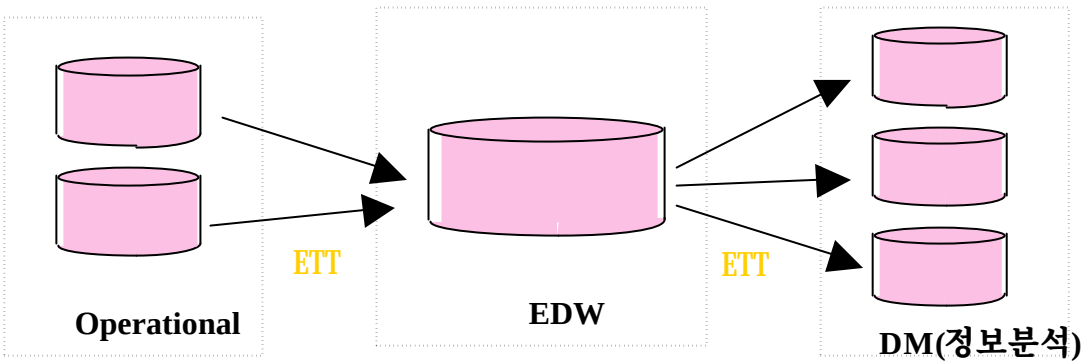
스냅샷 : 일정시간의 데이터
대변
-> 갱신 무의미
키에 시간요소 포함

1.3 (E)DW/DM/ODS - EDW (Enterprise Data Warehouse)

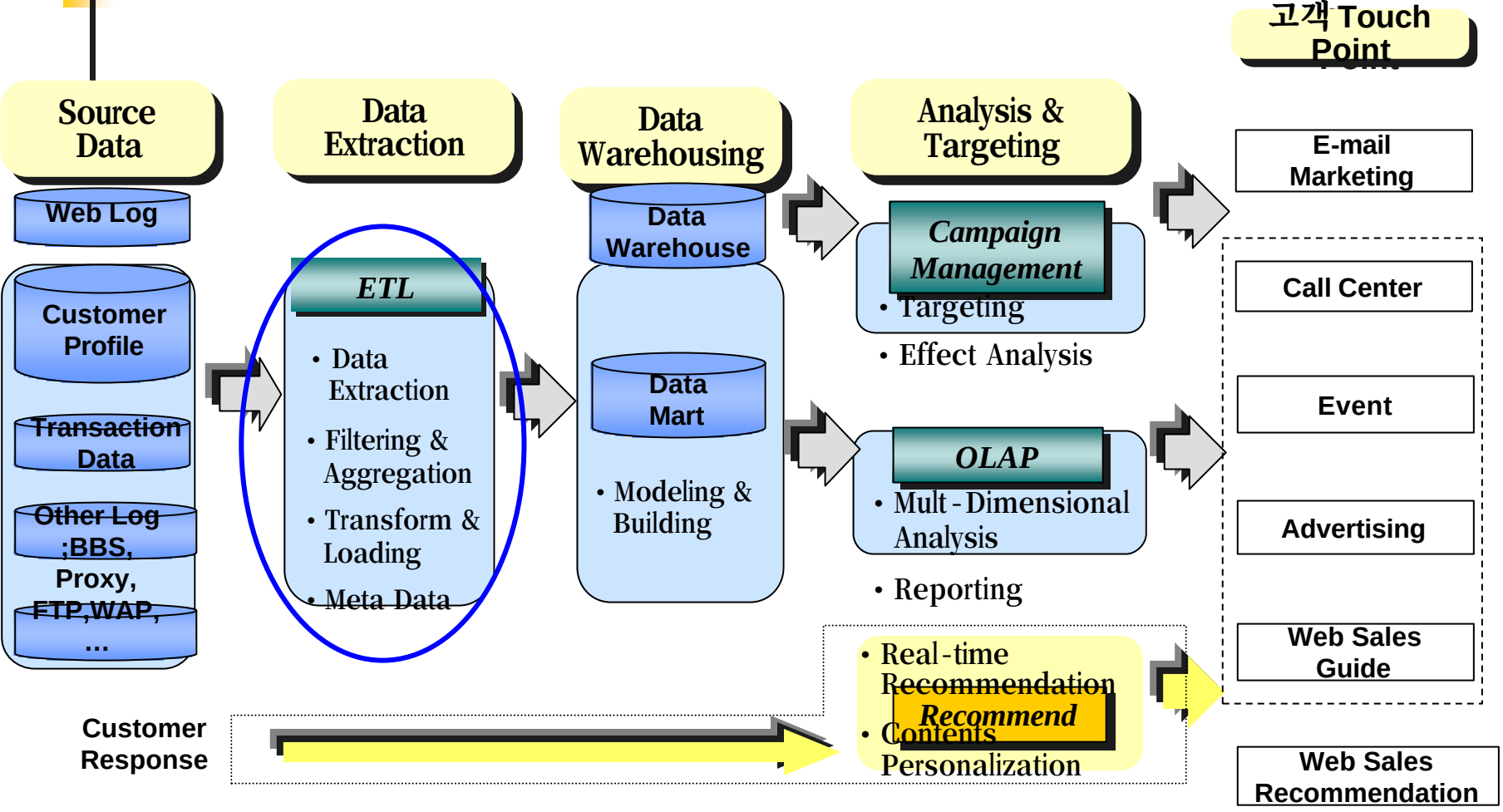


- **정보처리 프로세싱을 지원하는 단일 프로세스**
  - 전사처원의 정보분석계 요건을 충족하는 통합데이터 구조
  - 데이터의 일관성 유지
  - 운영계로부터 테이터획득 및 DM에 데이터 공급
  - DM이 다수인 경우 발생하는 Spider Web현상 해소

- **EDW특징**
  - 통합구조(Integrated) : 데이터 정합성과 물리적 통일성을 갖는 데이터 구조
  - Denormalized ER Model
  - Atomic Data
  - Historical Data 관리



1.4 요소기술 및 솔루션


고객 Touch Point
Source Data
Data Extraction
Data Warehousing
Analysis & Targeting
Web Log
Customer Profile
Transaction Data
Other Log ;BBS, Proxy, FTP,WAP, …
ETL
• Data Extraction
• Filtering & Aggregation
• Transform & Loading
• Meta Data
Data Warehouse
Data Mart
• Modeling & Building
Campaign Management
• Targeting
• Effect Analysis
OLAP
• Mult-Dimensional Analysis
• Reporting
Recommend
• Real-time Recommendation
• Contents Personalization
E-mail Marketing
Call Center
Event
Advertising
Web Sales Guide
Web Sales Recommendation
Customer Response

1.5 데이터 웨어하우스 모델링

주제영역

- 관심의 대상이 분명하게 구분되는 데이터의 논리적 그룹

Business Case

- 효율적인 업무수행을 위한 정보요구 사항들의 집합
- Business Process를 지원하는 정보요구 사항들의 집합

Fact

- 중심테이블(major 테이블)로서 관련성이 높은 Measure들의 집합
- Measure는 사업의 Performance와 주요 사업성공 요인들을 관찰할 수 있는 수단으로 사업분석에 필요로 하는 Data
- 두 가지의 Type의 Measures
- Raw(Base) fact / Derived(Calculated) metric

Dimension

- 부속 테이블(minor 테이블)
- 각 Fact를 분석하는 하나의 관점
- Dimension은 복수의 속성을 가지고 있으며 이것은 Drill-up과 Drill-down등 데이터분석의 요약수준에 대한 항해를 가능하게 하는 경로를 지님

1.6 데이터 웨어하우스 모델과 운영시스템 모델

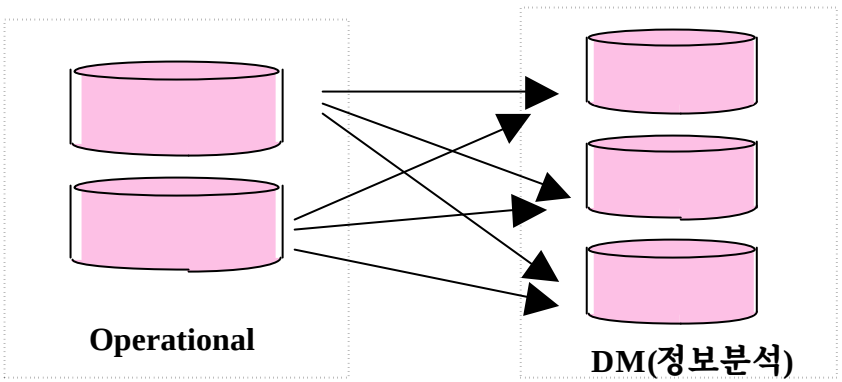
비교항목	ER모델/OLTP	DW모델/OLAP
용도	▣ On-Line Transaction Processing	▣ On-Line Analytical Processing
지원영역	▣ 업무처리 지원	▣ 의사결정(<=비교/분석) 지원
주요기능	▣ 정보의 기록	▣ 정보의 분석
주이용자	▣ 업무담당자	▣ 관리자 및 경영진
데이터 모델	▣ ER Model(=Entity Relation Model)	▣ Multi-Dimensional Model
키워드	▣ Entity, Relation, Attribute	▣ Fact, Dimension, Measure
Reporting	▣ 정형/고정적/전산실 지원	▣ 비정형/사용자취향/Ad hoc
데이터 측면	▣ 입력/수정/삭제 ▣ 통합/이력 부재 ▣ 상세데이터 ▣ 중복배제 ▣ 정확성/무결성 ▣ 현재	▣ 접근(=조회) ▣ 통합/이력 관리 ▣ 요약정보 ▣ 중복허용 ▣ 경향/추이 ▣ 과거/미래(Forecasting)

2. 데이터 마트 (Data Mart)

2.1 E)DW/DM/ODS - DM(Data Mart)

- 특정 부문의 의사결정 프로세싱 지원을 목적
- 부서별 또는 부문별 DW
  - 부서의 분석 수행에 가장 적합한 데이터 구조
  - EDW로 부터 필요한 데이터 획득
  - 일반적으로 한 기업내에 복수개의 DM존재

- DM의 특징
  - 분석요건 중심
  - 요약데이터로 구성
  - 제한된 규모의 이력 데이터 포함
  - OLAP지원
  - 다차원모델



2.2 E)DW/DM/ODS -ODS (Operating Data System)

- 배경
  - 통합적/전사적인 On-Line 질의 필요성
  - 현 운영계를 통합적으로 재 구축하는 데는 많은 시간/비용 투여
- 정의 : 운영 환경의 데이터 웨어하우스
  - 주제 중심적
  - 통합적
  - near current detail(30-60일 저장)
  - 트랜잭션 발생(Insert,Update,Delete)
  - non-historical

3. CRM(Customer Relationship Management)

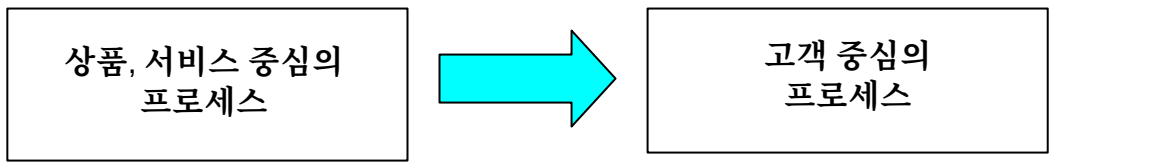
3.1 CRM의 정의

- "신규고객 획득과 기존고객 유지 및 고객 수익성 증대를 위하여 지속적인 커뮤니케이션을 통해 고객 행동을 이해하고 영향을 주기 위한 광범위한 접근"
  - 가트너 그룹

- 고객에 대한 "학습과 대응의 반복"

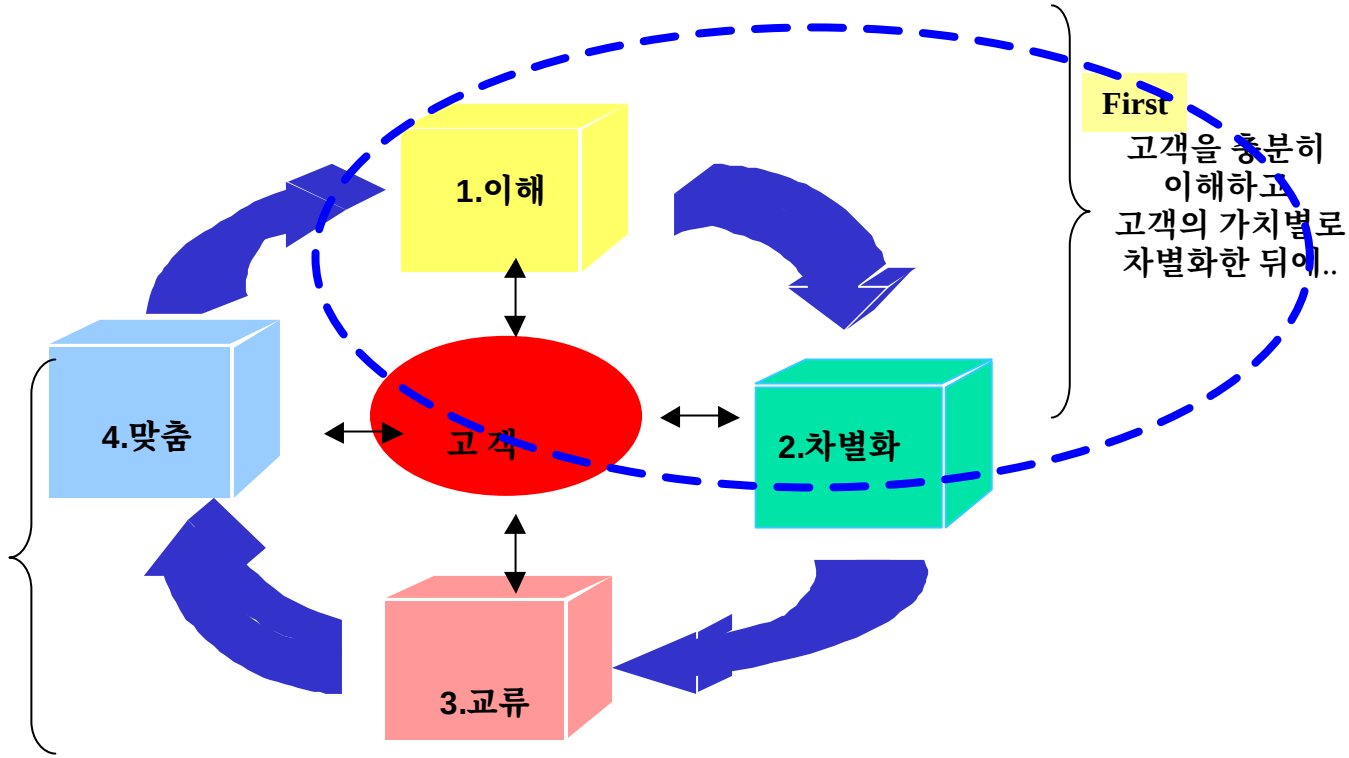
- 고객에 대한 내/외부 자료를 분석, 통합하여 고객들의 특성과 성향에 기초를 둔 마케팅활동을 계획/지원/평가하는 일련의 과정

- 기업의 업무 프로세스를 변화시키는 것



기업의 체질 변화를 통해
고객의 잠재된 가치를 발견하는 것이
CRM의 진정한 가치

3.2 CRM의 구축전략



* IDIC : Identify, Differentiate, Interacting, Customize (Don Peppers & Martha Rogers)

3.3 CRM의 요소기술 및 솔루션