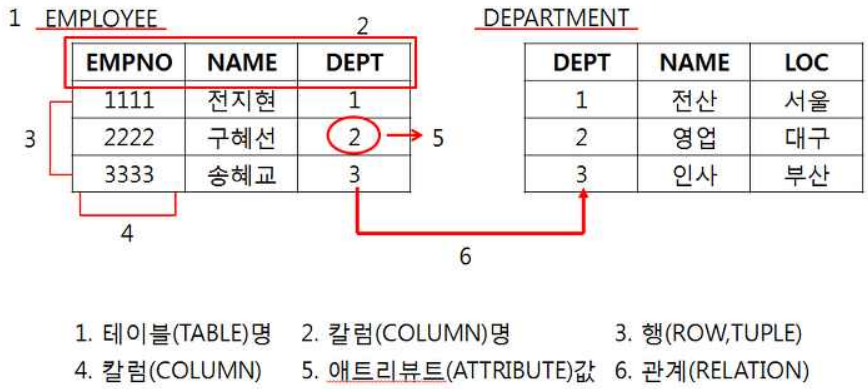
2장 관계 데이터 모델

데이터베이스 연구실

[목 차]

- 관계 데이터 모델
  - 1970년대 IBM 의 Codd박사에 의해 제안
  - 주류 데이터 베이스의 이론적인 모델을 제공

- 이전의 모델
  - 계층모델
  - 네트워크 모델
    - 레코드 단위를 처리



- 관계 데이터 모델
  - 테이블 형태의 집합의 개념으로 자료를 처리
  - 사용 자들이 쉽게 자료를 이해

2.1 관계 모델의 개념

- 릴레이션(REALTION): 사전적으로 '관계, 친척'등을 뜻
  - ✓ 관계 모델의 가장 중요한 핵심 개념
  - ✓ 관계 데이터 베이스는 관계들의 집합으로 표현
- 릴레이션은 '**릴레이션 스키마**' 와 '**릴레이션 인스턴스**'로 구성

2.1.1 릴레이션 스키마

- 릴레이션 스키마
  - BOOKID와 같이 각 열들의 이름과, 속성(애트리뷰트(ATTRIBUTE)을 가짐
- 애트리뷰트
  - 릴레이션 스키마가 가지고 있는 속성
- 도메인(DOMAIN)
  - 속성들이 가질 수 있는 값들의 범위가 지정되어있는 집합
- 차수 (DEGREE)
  - 릴레이션 스키마에서 애트리뷰트들은 적어도 1개 이상을 가져야 하며 이 애트리뷰트의 수를 말함

BOOKID	BOOKNAME	PUBLISHER	PRICE
1	축구의 역사	시공사	7000
2	축구를 아는 여자	나무수	13000
3	축구의 이해	대한미디어	22000

<표2.1> 도서 목록 TABLE

- 도서목록 (BOOKID : integer, BOOKNAME : char, PUBLISHER : char, PRICE : real)
- 릴레이션스키마 (애트리뷰트1 : 도메인1, 애트리뷰트2 : 도메인2,
  애트리튜트3 : 도메인2, 애트리뷰트4 : 도메인3)
- 릴레이션이름 (속성이름1 : 정수집합, 속성이름2 : 문자열집합,
  속성이름3 : 문자열집합, 속성이름4 : 실수집합)
- 차수 : 4

2.1.2 릴레이션 인스턴스

- 릴레이션 인스턴스(RELATION INSTANCE)
  - 표.2.1의 HEAD부분인 릴레이션 스키마의
    아래쪽의 행(2번째 행 이후) 들에 보여지는 실제 테이블의 자료

- 투플 (TUPLE)
  - 각 자료들의 행 하나 하나
  - 릴레이션 스키마의 차수와 같은 수의 필드를 가짐
  - 각 투플은 서로 중복이 없어야 함

- 카디널리티 (CARDINALITY)
  - 투플의 수(행들의 수) , 표2.1 의 카디널리티 3

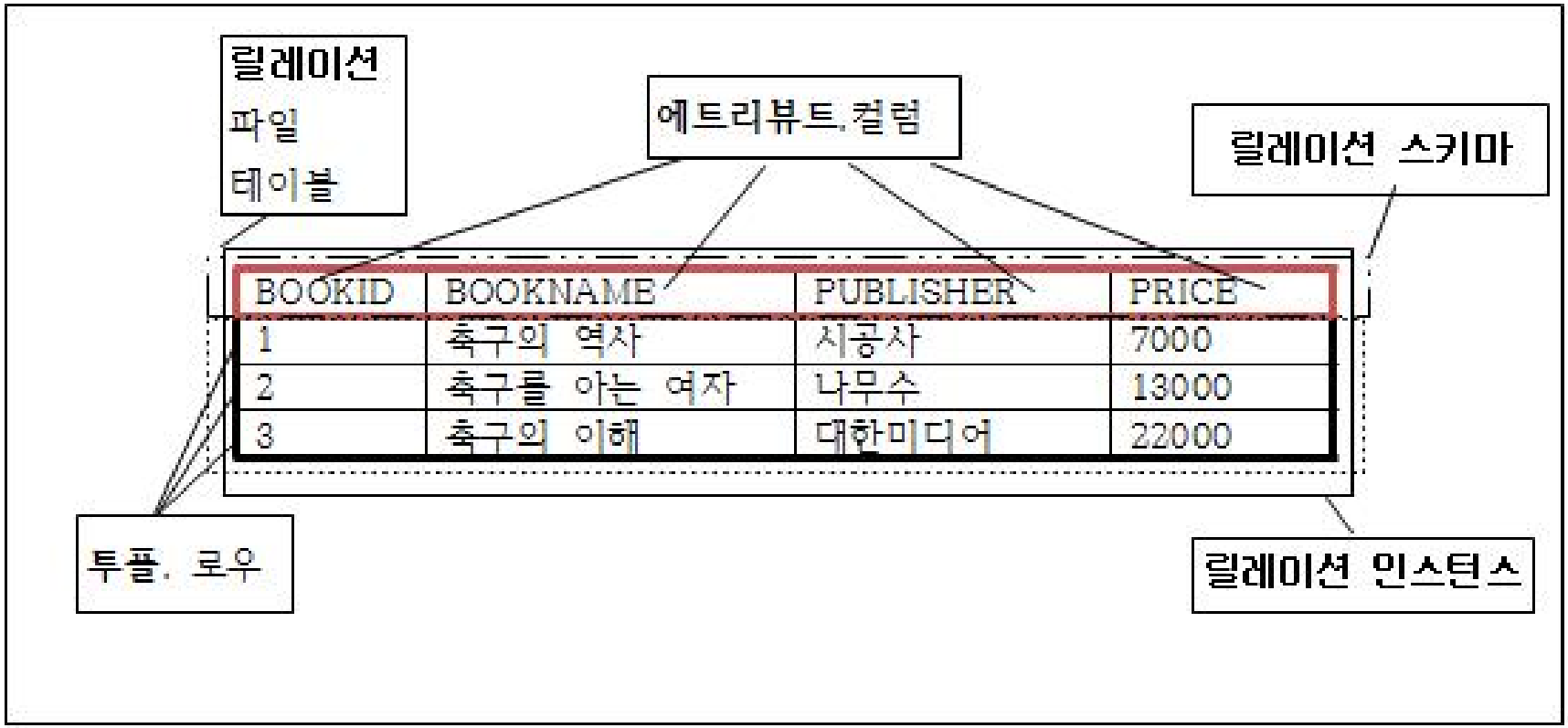


그림 2.1. 관계 데이터 모델의 주요 용어

2.1.3 관계 데이터 베이스

- 관계 데이터 베이스(RELATION DATABASE)
  - 릴레이션 스키마 (RELATION SCHEMA)와 릴레이션 인스턴스(REALTION INSTANCE)로 구성된 서로 다른 이름을 가진 릴레이션 (RELATION)들의 집합

- 관계 데이터 베이스 스키마 (RELATION DATABASE SCHEMA)
  - 릴레이션들을 위한 릴레이션 스키마(RELATION SCHEMA) 들의 집합

- 릴레이션 스키마(RELATION SCHEMA)
  - 릴레이션의 이름과 도메인(DOMAIN)에 정의된 값들을 가진 애트리뷰트(ATTRIBUTE)들의 집합
- 차수 (DEGREE)
  - 애트리뷰트의 수
- 릴레이션 인스턴스(RELATION INSTANCE)
  - 릴레이션 스키마에 정의된 값들의 집합인 투플 (TUPLE) 들의 집합
- 카디널리티(CARDINALITY)
  - 투플들의 수
- 투플과 애트리뷰트의 특징
  - 애트리튜트(열)은 각각 릴레이션 스키마의 애트리뷰트들의 도메인에 포함된 원자값
  - 애트리뷰트(열)과 투플(행)들의 순서는 서로 일치 하지 않아도 되지만 각 투플(행)들의 값은 서로 중복되지 않아야 함

릴레이션 스키마 R 의 경우

R(f1:D1,….,fn:Dn)

으로 나타내며
릴레이션 스키마 R의 투플은

{<f1:d1,….,fn:dn> | d1 ∈ Dom1, …. ,dn ∈ Domn}

로 표기

참조)

fi : 1≤i≤n
<….> : 한 투플의 필드
Domi : Di라는 이름의 도메인이 가진 값의 집합
| : 다음을 만족 하면
∈ : 포함한다

2.1.4 관계 모델의 표현 예

- 표현
  - 관계모델의 정의: 수학의 표현법을 따름
  - 물리적 데이터 베이스: SQL 이라는 언어에 의해 표현

- SQL 과 관계 모델
  - 릴레이션 → 테이블
  - 애트리 뷰트 → 컬럼
  - 투플 → 레코드

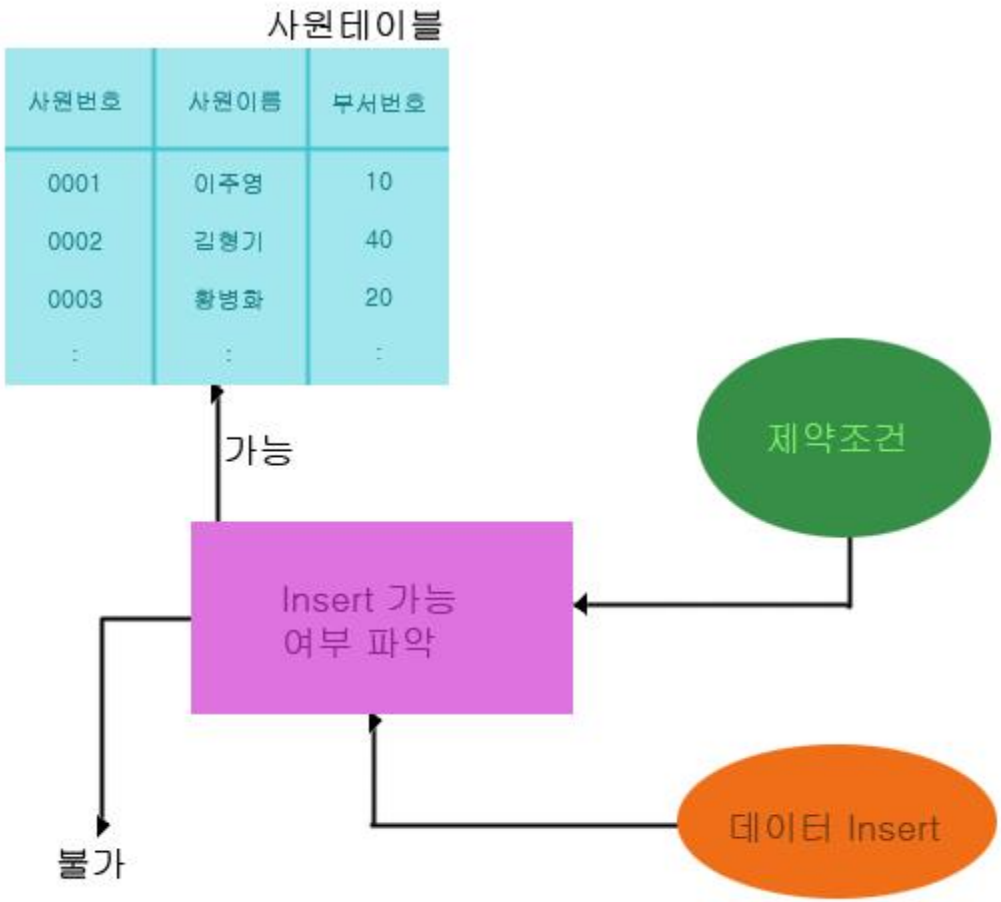
- 릴레이션 스키마
  - 릴레이션 인스턴스를 담기 위한 구조를 정의

- 릴레이션 스키마
  - 릴레이션 인스턴스를 담기 위한 구조를 정의
- 릴레이션 인스턴스
  - 릴레이션 스키마에 정의 된 구조에 맞게 저장된 투플들의 집합

SQL

- CREATE TABLE
  - 데이터를 저장하기 위한 테이블의 구조를 정의
- INSERT
  - 생성된 테이블에 데이터(투플)을 삽입
- DELETE
  - 테이블에 저장된 데이터(투플)을 삭제
- UPDATE
  - 테이블에 저장된 데이터(투플)의 수정

2.2 제약조건


사원테이블
사원번호
사원이름
부서번호
0001
이주영
10
0002
김형기
40
0003
황병화
20
가능
제약조건
Insert 가능
여부 파악
데이터 Insert
불가

2.2.1 키

- 데이터 베이스
  - 릴레이션들의 집합
- 릴레이션
  - 중복되지 않은 투플들의 집합
- 키
  - 릴레이션에서 특정한 투플을 찾기 위해서는 그 투플을 특정할 필요가 있음
  - 릴레이션내의 투플들 중에서 찾고자 하는 투플을 유일하게 표현할 수 있어야 함

2.2.1 키

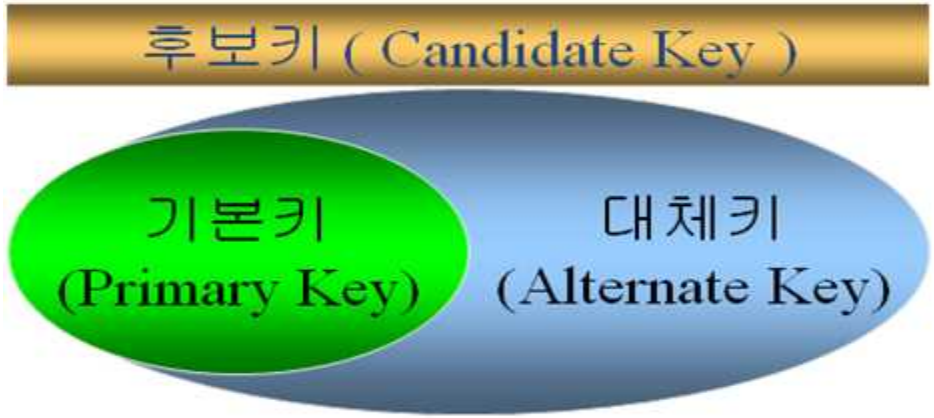
- **슈퍼 키 (super key)**
  - 투플의 애트리뷰트들 중 투플을 유일하게 나타낼 수 있는 집합
  - 투플의 모든 애트리뷰트의 집합일수도 있고 특정 값의 조합으로 이루어질 수도 있음
  - 키를 포함하는 필드들의 집합

If)
    다른 제약조건이 있다면
Then)
    이 필드들의 어떤 부분집합이 키를 형성가능
Else)
    모든 필드들의 집합이 키가 됨

2.2.1 키

- **후보키 (candidate key)**
  - 키 제약조건에 의하여 투플을 유일하게 식별하는 필드 집합
  - 그 릴레이션에 대한 후보키
  - **key** 라고 함

- ex) Student 릴레이션의 경우, sid 필드가 후보키



2.2.1 키

- **기본키 (primary key)**
  - 모든 가능한 **후보키 (candidate keys)**중에서 지정된 키
  - 하나의 투플을 데이터베이스 내 다른 곳에서 언급할 때 효율적으로 활용 가능
  - 한 릴레이션에는 하나 밖에 존재하지 않음
  - 한 릴레이션 내의 모든 투플을 고유하게 식별할 수 있어야 함
  - 널(null) , 중복값이 없음
  - 기본키를 찾기 어려울때는 레코드번호와 같은 값을 만들고 이를 **대리키 (surrogate key)**라고 함

2.2.1 키

- **대체키 (alternate key)**
  - 기본키로 선정되지 않는 후보키

- **외래키(foreign key)**
  - 한 릴레이션에 저장된 정보가 다른 릴레이션에 저장된 정보에 링크된 경우
  - 이중 한곳에서 수정이 되면 연관된 릴레이션에 대한 정보 역시 수정되어야 **"자료의 일관성"**을 유지 할 수 있음
  - 외래키는 동일한 릴레이션을 참조할 수도 있음

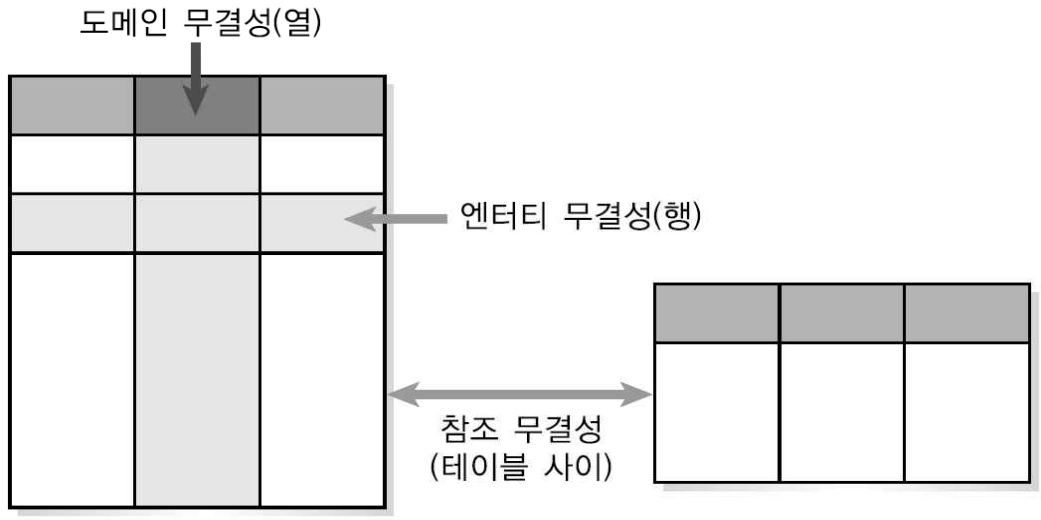
Null (널)

•한 투플의 어떤 필드에 null값을 사용하는 것은 그 필드의 값을 모르거나 적용 할 수 없다는 것을 의미

•null은 기본키에는 허용되지 않음

2.2.2 제약조건

- **데이터 무결성(integrity, 無缺性)**
  - 데이터베이스에 저장된 데이터의 일관성과 정확성을 지키는 것을 말하며, 데이터가 무결성을 가지도록 하는 행위를 무결성 강화(enforcement)라고 함



[그림 8-5] 데이터 무결성의 종류

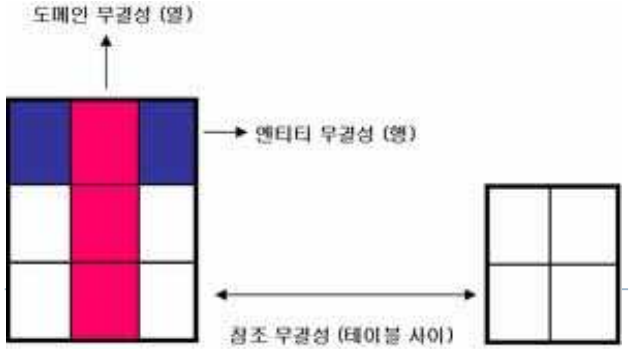
- **데이터 베이스가 중요하게 취급되는 이유**
  - 데이터베이스 SW (DBMS)의 기술적인 가치 보다 그 데이터 베이스 SW에 의해 취급되어 지는 **데이터의 가치가 중요**

- **키 제약조건 (key constraint)**
  - 릴레이션에 속한 필드들의 어떤 최소 부분집합이 각 투플에 대한 '고유한 식별자' 라는 선언

- **외래키 제약조건 (foreign key constraint)**
  - 두개의 릴레이션을 포함하는 가장 일반적인 IC

2.2.3 일반적인 제약조건

- **도메인(Domain) 무결성**
  - 열의 값에 관련된 무결성으로, 이를 강화시키는 방법에는 데이터 형식, NULL / NOT NULL, 기본값, 체크 / 규칙 등이 있다.
  - 열 무결성이라고도 한다.
- **엔터티(Entity) 무결성**
  - 테이블의 모든 행들이 유일하게 식별되는 무결성으로, 대부분의 경우 기본 키에 의해 강화된다. 테이블 무결성이라고도 한다.
- **참조(Referential) 무결성**
  - 서로 관계를 맺은 두 테이블 사이의 무결성으로, 외래 키에 의해 강화된다. 자식 테이블의 외래 키는 반드시 부모테이블의 기본 키 값으로 존재하는 값을 가져야 하며, 외래 키가 참조하는 기본 키 값은 변경 또는 삭제가 금지된다.

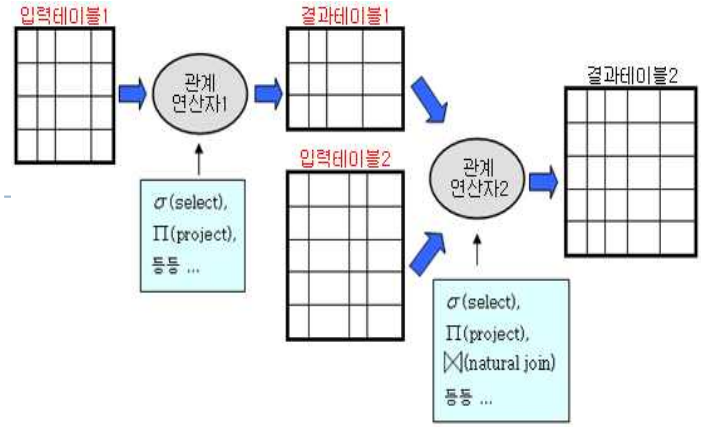


2.3 관계 대수

- SQL과 같은 관계 질의어의 기초?
- 절차적인 언어와 선언적인 언어의 차이
- 관계대수는 무엇이며, 왜 그것이 중요한가?
- 기본 대수 연산자들은 무엇이며 어떻게 그들이 복잡한 질의를 작성하기 위해 조합되는가?

질의어(query language)
데이터베이스에 들어있는 데이터에 대한 질의(query)를 하기 위한 특수 언어

2.3.1 관계 대수



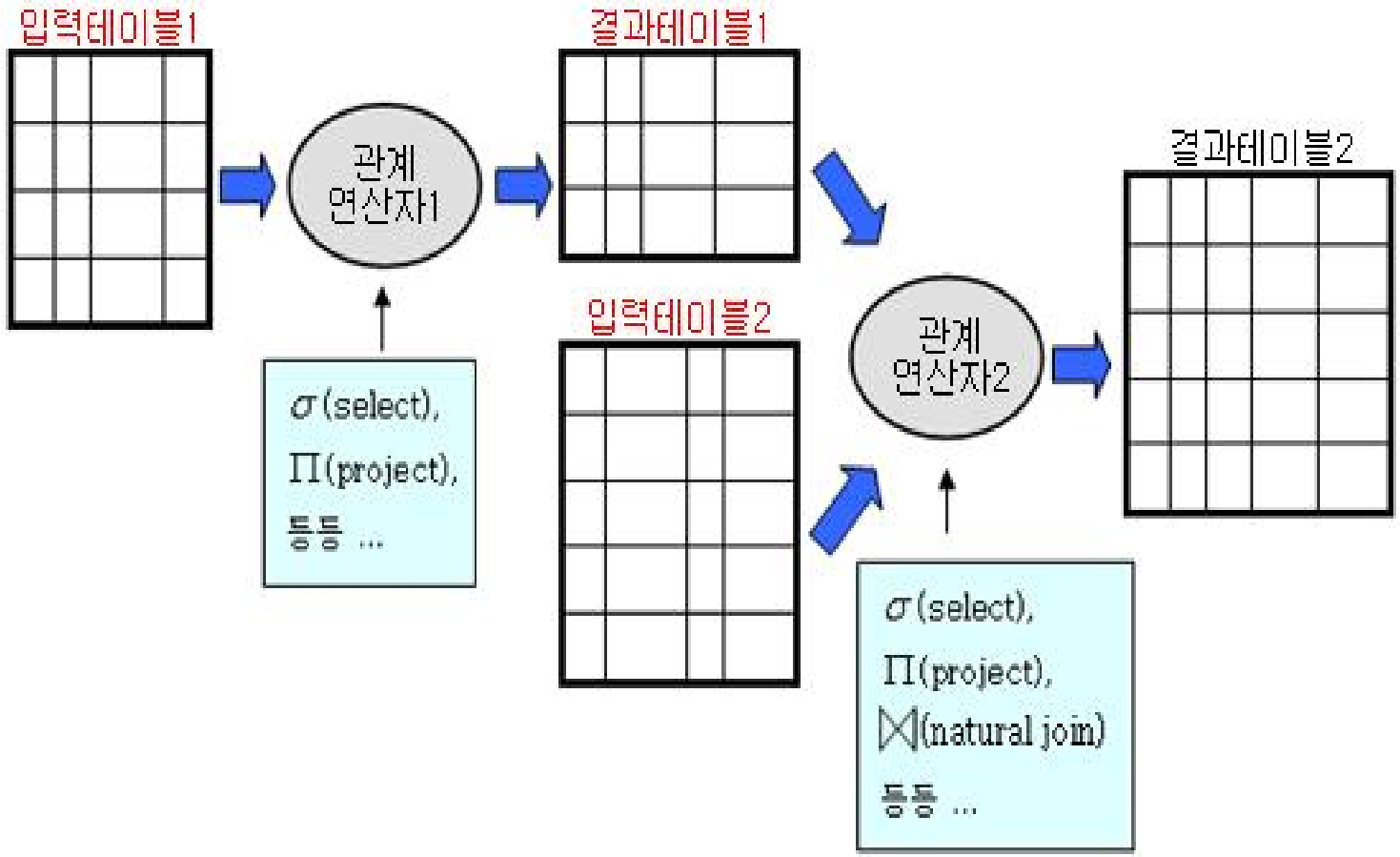
- **관계 대수 (RELATIONAL ALGEBRA)**
  - 연산자들을 사용하여 질의를 대수로 표현

- **기본연산자**
  - 셀력션, 프로젝션, 합집합, 크로스 프로덕트, 차집합 의 몇가지 추가 연산자 (기본연산자로 정의 가능하나 자주사용)
  - 연산자들은 하나 또는 두 개의 릴레이션 인스턴스를 매개 변수로 받아서 결과로 하나의 인스턴스를 반환

- **관계대수식 (relational algebra expression)**
  - 단항 (unary) 대수 연산자 : 한 릴레이션, 단일 식에 적용
  - 이항 (binary) 대수 연산자 : 두 개의 식에 적용, 순환적으로 정의

- **관계질의**
  - 연산자의 적용 순서에 의거하여, 원하는 답을 계산하기 위한 **단계적 절차를 기술**

대수식 : 질의 수행을 위한 일종의 계획
일부 dbms : 질의 수행 계획을 표현하기 위해 대수식을 이용


입력테이블1
관계
연산자1
결과테이블1
σ(select),
Π(project),
등등 ...
입력테이블2
관계
연산자2
결과테이블2
σ(select),
Π(project),
⋈(natural join)
등등 ...

연산자	기호	표기법	의미
선택 (selection)	σ	$\sigma_p(r)$	Selection 조건(Predicate)을 만족하는 Tuple들의 부분 집합
추출 (projection)	$\prod$	$\prod_{A_1, A_2 \ldots A_k}(r)$	Attribute들의 부분 집합. 중복은 제거됨.
합집합 (set union)	$\cup$	$r \cup s$	합집합 호환인 두 개의 Relation의 합이 추출. 중복은 제거됨. ※ 합집합 호환 : 두 Relation의 차수가 같고, 대응되는 Attribute들의 Domain이 같음을 의미
차집합 (set difference)	$-$	$r - s$	차집합 수행.
카티션 곱 (cartesian product)	$\times$	$r \times s$	두 Relation의 가능한 모든 Tuple들의 집합. 차수는 더하고, Cardinality는 곱함.
재명명 (rename)	ρ	$\rho_x(E)$	Relation의 이름을 변경. ※ 기본연산으로 분류하지 않는 자료도 있음.

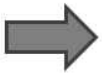
연산자	기호	표기법	의미
교집합 (set intersection)	$\cap$	$r \cap s$	교집합을 수행
조인 (join)	$\bowtie$	$r \bowtie s$	두 개의 Relation의 연관된 Tuple들을 결합.
나누기 (division)	$\div$	$r \div s$	r 에서 s의 Attribute들이 Domain값과 일치하는 r의 Tuple들을 찾아 내는 연산.
배정 (assignment)	$\leftarrow$	$temp \leftarrow \prod_{R-S}(r)$	프로그래밍 언어의 Assign과 비슷함. 복잡한 질의를 단순하게 표현하는 방법

2.3.2 셀렉션과 프로젝션

- 단일 릴레이션에 있는 데이터를 조작
- 관계 대수식의 결과는 하나의 릴레이션

Relation r

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10



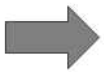
$\sigma_{A=B \wedge D > 5}(r)$

A	B	C	D
α	α	1	7
β	β	23	10

σ(시그마)
행들을 선택 연산자
셀렉션 연산자

Relation *r*:

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2



$\Pi_{A,C}(r)$

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

π(파이)
열들을 추출하기 위한 연산자
프로젝션 연산자

- **σ (select)**
  - 셀렉션 연산자, 셀렉션 조건을 통해 유지할 투플들을 명시
  - 투플들 중에 조건에 만족하는 투플들로 새로운 릴레이션을 결과로 만듬

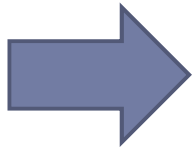
- **σ (조건) R**
  - R은 릴레이션

σ 금액 >= 10000 주문
주문 릴레이션에서 금액이 10000원이상인 투플 (행)

- **π (project)**
  - 프로젝션 연산자, 한 릴레이션으로부터 애트리뷰트(열)들을 추출
- **π(속성) R**
  - R은 릴레이션
- **π (이름,연락처,금액) 주문**
  - 주문 릴레이션에서 '이름, 연락처, 금액' 속성만 결과 릴레이션으로 출력

참조)
실제 dbms들은 중복제거라는 절차 생략 (리소스요구) 하여 중복으로 표현
관계대수나 관계해석에서는 릴레이션들이 투플들의 집합이 되도록 중복제거가 항상 수행 한다고 가정

릴레이션 정의: 투플들의 집합, 중복 허용하지 않음



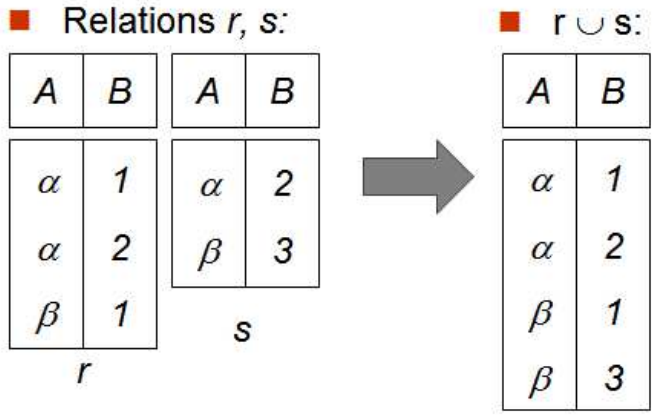
셀렉션 행, 프로젝션 열
결과 중복제거

2.3.3 집합연산

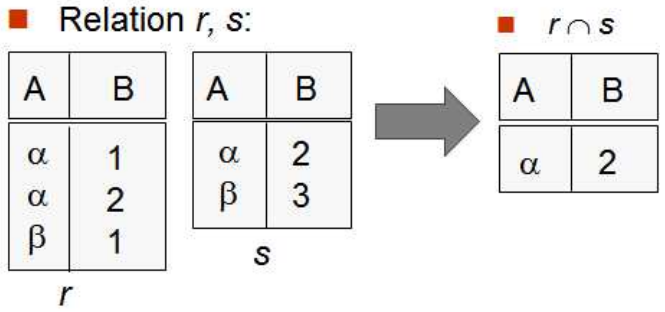
- 집합에 대한 표준 연산 사용
  - 합집합 (union : ∪), 교집합 (intersection:∩)
  - 차집합 (set-difference: -)
  - 크로스 프로덕트 (cross-product: x)

합병가능 (union-compatible)
필드수가 같아야 함
왼쪽부터 오른쪽으로 순서대로 대응필드가 동일한 '도메인'
필드 이름은 상관없음

- **합집합: R ∪ S**
  - 릴레이션 인스턴스 R 이나 릴레이션 인스턴스 S (양쪽모두) 에 나타나는 모든 투플을 포함하는 릴레이션 인스턴스 반환
  - R과 S는 서로 합병 가능 (union - compatible) 하여야함
  - 결과 스키마는 R의 스키마와 동일

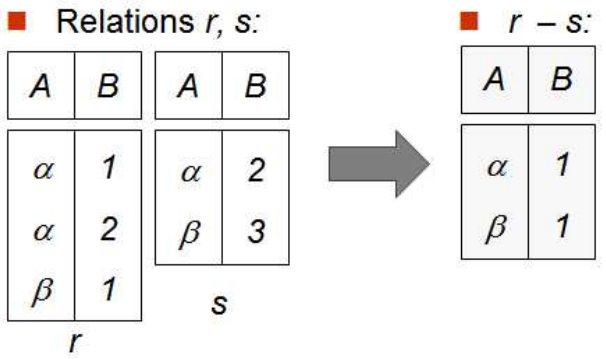


- **교집합 : R ∩ S**
  - R과 S에 모두 속하는 투플을 포함하는 릴레이션 인스턴스를 반환
  - 릴레이션 R과 S는 서로 합병가능 하여야 함
  - 결과 스키마는 R의 스키마와 동일

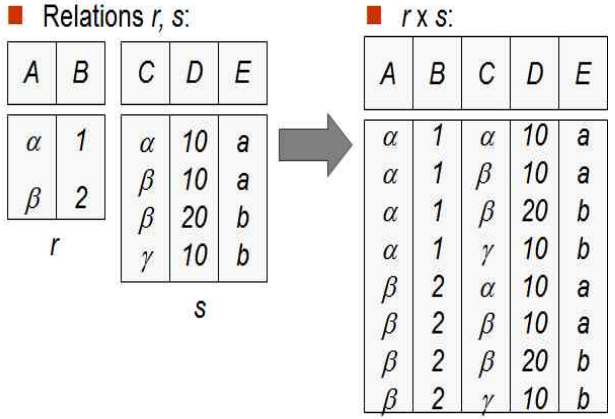


2.3.3. 집합연산

- **차집합: R-S**
  - R에는 속하고 S에는 속하지 않는 투플로 구성된 릴레이션 인스턴스 반환
  - 릴레이션 R과 S는 서로 합병가능 하여야 함
  - 결과 스키마는 R의 스키마와 동일



- **크로스 프로덕트: R×S**
  - R의 모든 필드 다음에 S의 모든 필드들을 순서대로 포함하는 릴레이션 인스턴스를 반환
  - 결과 r∈R, s∈S의 각 쌍에 대하여
  - 하나의 투플 <r, s> (투플 r과 투픈 s의 집합)를 포함



- **카티젼 프로덕트 (cartesian product)**
  - R x S 의 필드들은 R과 S의 대응 하는 필드로부터 이름 상속
  - R과 S중 동일한 이름의 경우 이름충돌 (naming conflict)이 발생
  - 이름이 만들어지지 않고 위치로만 언급

2.3.4 이름 바꾸기

- ρ (로우) : 개명 연산자
  - 식 ρ (R (),E)
  - 개명 리스트에 있는 필드를 제외하고 E 와 동일
  - oldname -> newname , position -> newname

ex) ρ(C(1-> sid1, 5-> sid2), S1 × R1)
=> C (sid1 : integer, sname: string, rating: integer, age: real, sid2: integer, bid:integer, day: dates)

2.3.5 조인

- 두 개 이상의 릴레이션들의 공통적인 속성(애트리뷰트)를 중심으로 합쳐 하나의 릴레이션으로 표현

- 두 개 이상의 릴레이션들로 부터 정보를 조합하기 위해 사용하는 가장 일반적인 방법

- **정의**
  - 크로스 프로덕트 후 셀렉션과 프로젝션
  - 크로스 프로덕트의 결과는 조인의 결과보다 더 크므로 셀렉션과 프로젝션을 바로 적용하여 구현하는 것이 중요

- **조인조건**
  - 조인조건 c와 한쌍의 릴레이션 인스턴스를 매개 변수로 받아 하나의 릴레이션 인스턴스로 반환
- **정의**
  - R ▷◁c S = σc (R×S)
  - R ▷◁(키속성r = 키속성s)S

■ Relations r, s:

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

r

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	∈

s

■ r ⋈ s

A	B	C	D	E
α	1	α	a	α
α	1	α	a	γ
α	1	γ	a	α
α	1	γ	a	γ
δ	2	β	b	δ

고객 ▷◁(고객.고객번호 = 주문.주문자)주문

고객 과 주문 릴레이션을 대상으로
고객 릴레이션의 고객번호와
주문 릴레이션의 주문자를 키로 하여
하나의 릴레이션을 반환