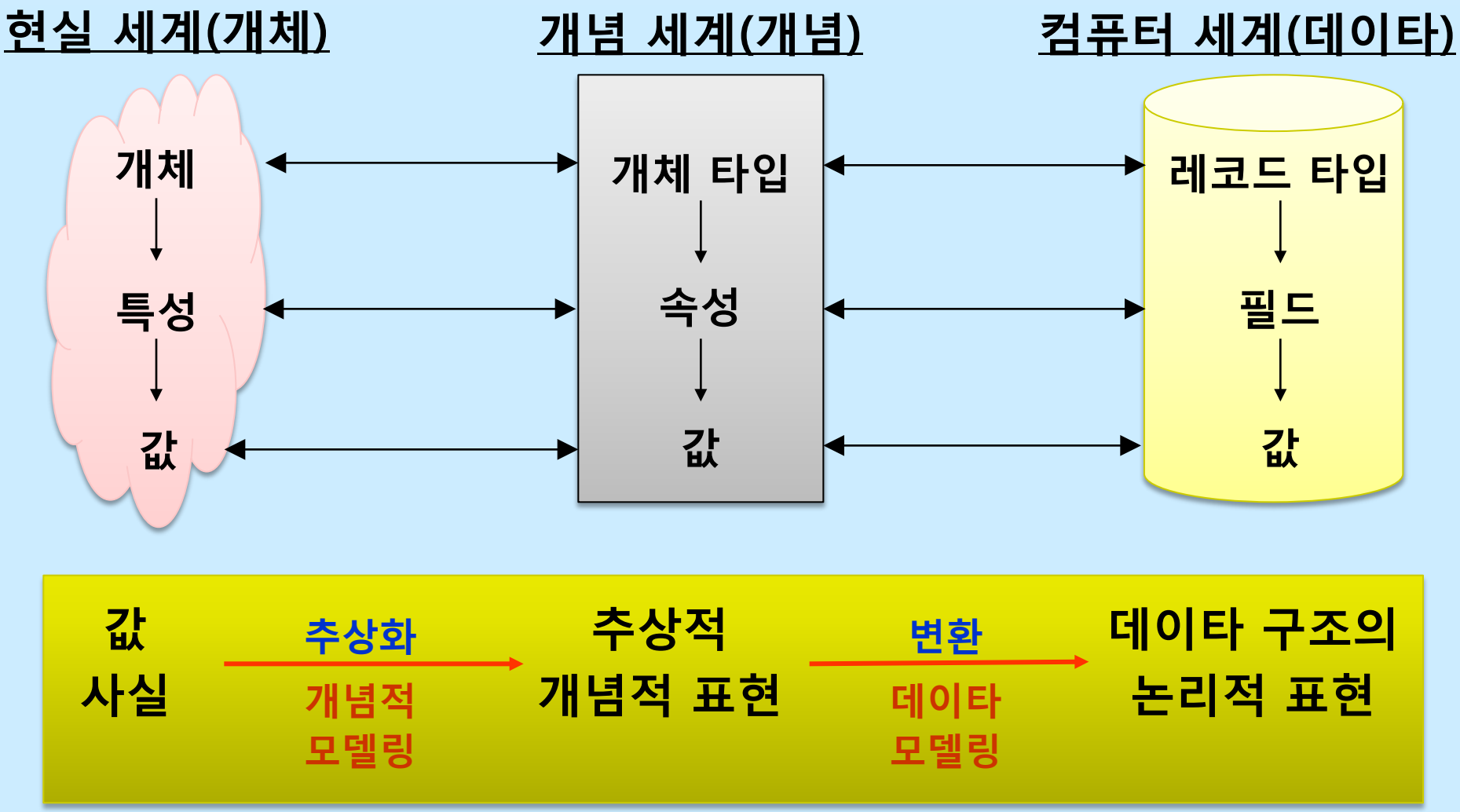
컴퓨터가 관리하는 데이터베이스는 계속적으로 변화하는 현실 세계를 표현

컴퓨터에 저장할 데이터의 구조를

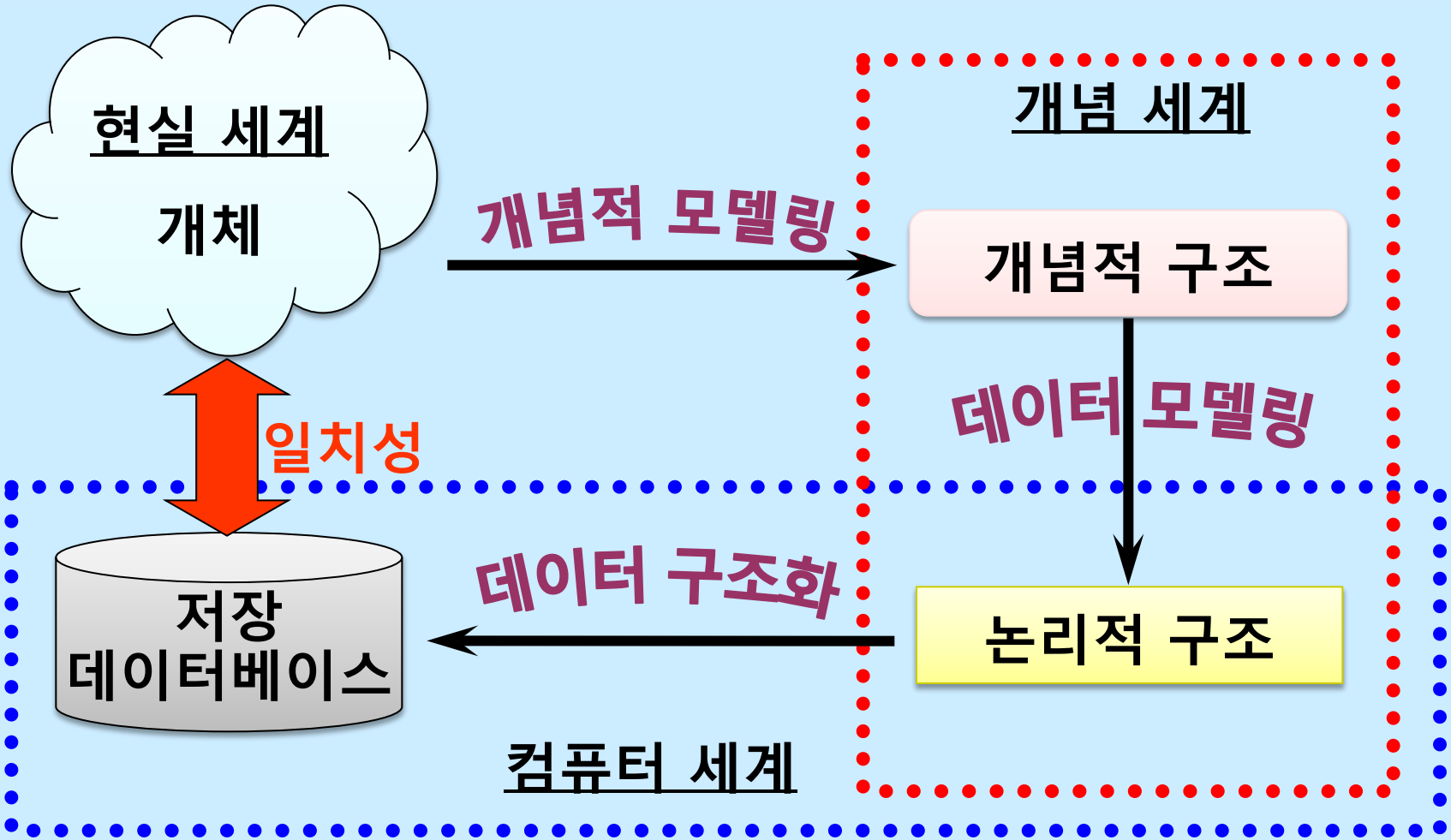
논리적으로 표현하기 위해 사용하는 도구

데이터 모델

데이터 : 현실세계 vs. 컴퓨터



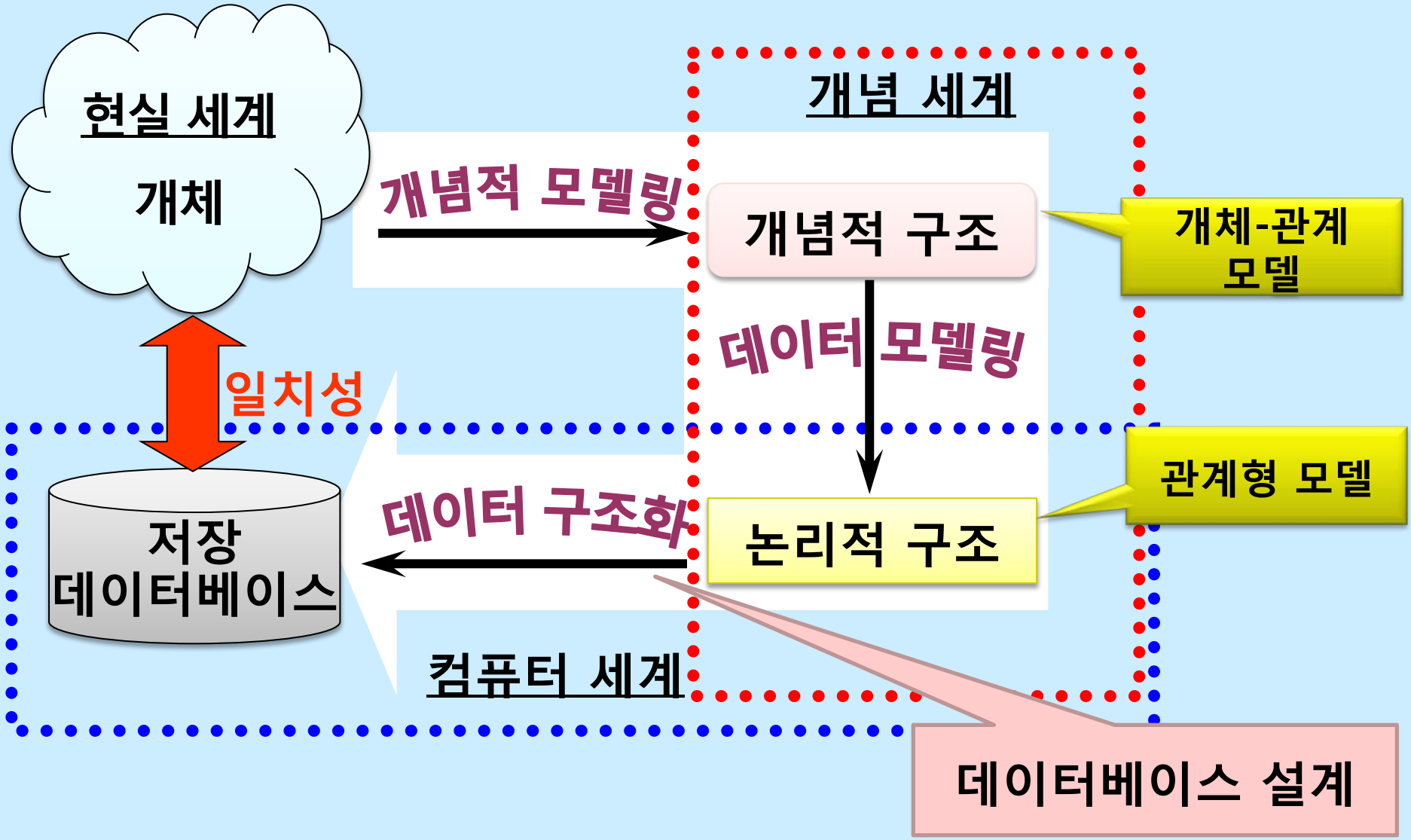
데이터베이스 : 현실세계 vs. 컴퓨터



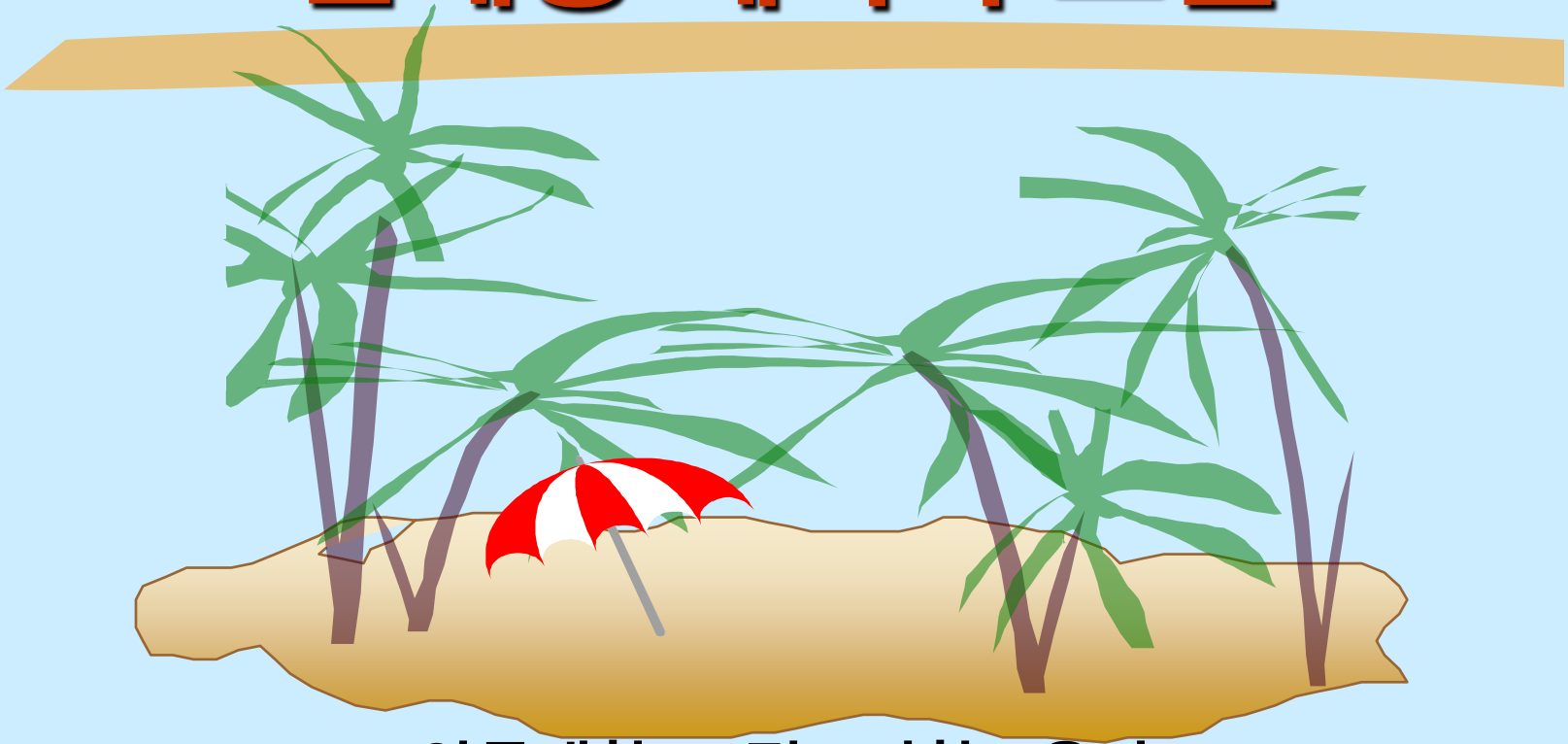
데이터 모델(Data Model) - I

- 다음 사항들을 기술하기 위한 개념적 표현
  - 데이터
  - 데이터들 간의 관계
  - 데이터의 의미
  - 일관성 제약 조건
- 데이터 모델 : D = < S, O, C >
  - S : 데이터의 구조(Structure)
    - 정적 성질 (추상적 개념) - 개체타입과 이들 간의 관계를 명세
  - O : 연산(Operations)
    - 동적 성질 - 개체 인스턴스를 처리하는 작업(데이터 조작)에 대한 명세
  - C : 제약 조건(Constraints)
    - 데이타의 논리적 제약 - 개체 인스턴스의 허용 조건
    - 데이타 조작의 한계를 표현한 규정

데이터 모델(Data Model) - II



관계형 데이터 모델



안동대학교 정보과학교육과

배울 내용

- 관계 대수
  - 일반 집합 연산
  - 순수 관계 대수 연산
  - 확장 관계 대수 연산
- 데이터베이스의 수정
- 뷰

관계 대수(Relational Algebra)

- 릴레이션을 처리하기 위한 연산의 집합
  - 릴레이션 : 투플의 집합
- 기본 연산
  - 일반 집합 연산 :
    - 합집합 , 교집합, 차집합, 카티션 프로덕트
  - 순수 관계 연산 :
    - 실렉트, 프로젝트, 조인, 디비전
- 폐쇄성질 (closure property)
  - 피연산자와 연산 결과가 모두 릴레이션
  - 중첩(nested)된 수식의 표현이 가능

일반 집합 연산(I)

- 합집합 (union, ∪)
  - $R \cup S = \{ t \mid t \in R \vee t \in S \}$
  - $|R \cup S| \leq |R| + |S|$
- 교집합 (intersect, ∩)
  - $R \cap S = \{ t \mid t \in R \wedge t \in S \}$
  - $|R \cap S| \leq \min\{ |R|, |S| \}$
- 차집합 (difference, -)
  - $R-S = \{ t \mid t \in R \wedge t \notin S \}$
  - $|R-S| \leq |R|$
- 카티션 프로덕트 (cartesian product, ×)
  - $R \times S = \{ r \cdot s \mid r \in R \wedge s \in S \}$: 접속(concatenation)
  - $|R \times S| = |R| \times |S|$
  - 차수(degree) = R의 차수 + S의 차수

일반 집합 연산(II)

- 합집합 연산
  - 표기법 : r ∪ s
  - 정의 : $r \cup s = \{t \mid t \in r \text{ or } t \in s\}$
  - r ∪ s가 가능하려면,
    - r과 s는 같은 항(애트리뷰트의 수가 같음)을 가져야 함
    - 애트리뷰트의 도메인은 양립할 수 있어야 함
      - 즉, r의 두번째 열은 s의 두번째 열의 것과 같은 유형의 값을 다룸
  - 예

릴레이션 r

A	B
α	1
α	2
β	1

릴레이션 s

A	B
α	2
β	3

r ∪ s

A	B
α	1
α	2
β	1
β	3

일반 집합 연산(III)

- 교집합 연산
  - 표기법: r ∩ s
  - 정의 : $r \cap s = \{t \mid t \in r \text{ and } t \in s\}$
  - 가정:
    - r과 s는 같은 항을 갖는다.
    - r과 s의 애트리뷰트는 양립성이 있다.
  - 유의: r ∩ s = r - (r - s)
  - 예

릴레이션 r

A	B
α	1
α	2
β	1

릴레이션 s

A	B
α	2
β	3

r ∩ s

A	B
α	2

일반 집합 연산(IV)

- 차집합
  - 표기법 : r - s
  - 정의 : $r - s = \{t \mid t \in r \text{ and } t \notin s\}$
  - r - s가 가능하려면,
    - r과 s는 같은 항(애트리뷰트의 수가 같음)을 가져야 함
    - 애트리뷰트의 도메인은 양립할 수 있어야 함
      - 즉, r의 두번째 열은 s의 두번째 열의 것과 같은 유형의 값을 다룸
  - 예

릴레이션 r

A	B
α	1
α	2
β	1

릴레이션 s

A	B
α	2
β	3

r - s

A	B
α	1
β	1

일반 집합 연산(V)

- 카티젼곱 연산
  - 표기법 : r × s
  - 정의 : $r \times s = \{t\, q \mid t \in r \text{ and } q \in s\}$
  - r(R)과 s(S)의 애트리뷰트가 서로 다르다고 가정(즉, $R \cap S = \phi$).
  - r(R)과 s(S)의 애트리뷰트가 서로 같다면, 재명명을 사용해야 함
  - 예

릴레이션 r

A	B
α	1
β	2

릴레이션 s

C	D	E
α	10	+
β	10	+
β	20	-
γ	10	-

r × s

A	B	C	D	E
α	1	α	10	+
α	1	β	10	+
α	1	β	20	-
α	1	γ	10	-
β	2	α	10	+
β	2	β	10	+
β	2	β	20	-
β	2	γ	10	-

순수 관계 연산(I)

- 선택연산
  - 표기법 : $\sigma_P(r)$
  - 정의 : $\sigma_P(r) = \{ t \mid t \in r \text{ and } P(t)\}$

    여기서, P 는 다음과 같은 유형의 표현을 다루는 조건식

    <속성> = (또는 $\neq, >, \geq, <, \leq$) <속성> 또는 <상수>

    $\wedge$(and), $\vee$(or), $\neg$(not)으로 연결된다.
  - 예제

릴레이션 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

순수 관계 연산(II)

- 추출 연산
  - 표기법: $\Pi_{A_1, A_2, \ldots, A_k}(r)$
  - 여기서 A_1, A_2는 애트리뷰트명이고 r은 릴레이션명
  - 결과는 명시하지 않은 열을 제외한 k 열의 릴레이션
  - 릴레이션은 집합이기 때문에 중복 행은 결과에서 제거
  - 예

릴레이션 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

순수 관계 연산(III)

- 죠인 연산
  - 표기법 : $r \bowtie_{\theta} s$ (= Theta join), $r \bowtie_{N} s$ (= Natural join)
  - r과 s를 각각 스키마 R과 S상의 릴레이션이라 하면, 죠인 연산 결과는 r의 튜플 t_r과 s의 튜플 t_s의 각 쌍을 고려해 얻은 스키마 $R \cup S$ 상의 릴레이션이다.
  - t_r과 t_s가 $R \cap S$의 애트리뷰트 각각에 같은 값을 가지면, 다음과 같이 튜플 t가 결과에 추가된다.
    - t 는 r 상에 t_r로서 같은 값을 갖는다.
    - t 는 s상에 t_s로서 같은 값을 갖는다.
  - 예 : R = (A, B, C, D), S = (E, B, D) ➔ 결과 스키마는 (A, B, C, D, E)
  - $r \bowtie s$는 다음과 같이 정의된다.

$$\Pi_{r.A,\ r.B,\ r.C,\ r.D,\ s.E}(\sigma_{r.B=s.B \wedge r.D=s.D}(r \times s))$$

순수 관계 연산(IV)

❑ 조인 연산(계속) - 예

릴레이션 r

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

릴레이션 s

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	ε

$r \bowtie s$

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

순수 관계 연산(V)

- 나누기 연산
  - 표기법: r ÷ s
  - "모두에 대한"이라는 구절을 내포한 질의에 적합하다.
  - r과 s를 각각이 다음과 같은 스키마 R, S상의 릴레이션이라 하면,
    - R = (A_1, …, A_m, B_1, …, B_n), S = (B_1, …, B_n)
    - r ÷ s의 결과는 스키마 R-S = (A_1, …, A_m)상의 릴레이션
    - $r \div s = \{t \mid t \in \Pi_{R-S}(r) \wedge \forall u \in s(tu \in r)\}$
  - 예

릴레이션 r

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
δ	6
ε	1
ε	2

릴레이션 s

B
1
2

r ÷ s

A
α
ε

순수 관계 연산(V-1)

❑ 나누기 연산 예

학과목(SC)

학번 (SNO)	과목번호 (CNO)
100	C413
100	E412
200	C123
300	C312
300	C324
300	C413
400	C312
400	C324
400	C413
400	E412
500	C312

과목1(C1)

과목번호 (CNO)
C413

과목2(C2)

과목번호 (CNO)
C312
C413

과목3(C3)

과목번호 (CNO)
C312
C413
E412

SC ÷ C1

학번 (SNO)
100
300
400

SC ÷ C2

학번 (SNO)
300
400

SC ÷ C3

학번 (SNO)
400

순수 관계 연산(VI)

- 재명명 연산
  - 이름을 줄 수 있도록 하여 관계 대수 표현식의 결과를 참조하도록 함
  - 하나 이상의 이름으로 릴레이션을 참조하도록 함
  - 예1: $\rho_x(E)$ ..... 이름 x로 표현식 E를 돌려줌
  - 예2: 관계형 대수 표현식 E가 n항이면, $\rho_{x\ (A1,\ A2,\ \ldots,An)}(E)$
    - 이름 x하에 $A_1, A_2, \ldots, A_n$으로 재명명 된 애트리뷰트를 가진 표현식 E의 결과를 돌려줌

순수 관계 연산(VI-1)

- 재명명 연산 예

Account-number	Branch-name	balance
101	Downtown	500
102	Perryridge	400
201	Brighton	900
215	Mianus	700
217	Brighton	750
222	Redwood	700
305	Round hill	350

- 은행에서 가장 큰 잔고를 찾아라.

$$\Pi_{balance}(account) - \Pi_{account.balance}(\sigma_{account.balance < d.balance}(account \times \rho_{d}(account)))$$

순수 관계 연산(VII)

- 배정 연산
  - 배정 연산(←)은 복잡한 질의를 편리하게 표현하는 방법을 제공
  - 질의를 일련의 배정 연산과 질의의 결과 값이 출력되는 표현식으로 구성된 순차 프로그램으로 작성
  - 배정은 항상 임시 릴레이션 변수로 작성되어야 함
  - 예 : r ÷ s는 다음과 같이 작성한다.

$$temp1 \leftarrow \Pi_{R-S}(r)$$

$$temp2 \leftarrow \Pi_{R-S}((temp1 \times s) - \Pi_{R-S,S}(r))$$

$$result = temp1 - temp2$$

    - ←의 오른쪽 결과가 ←의 왼쪽의 릴레이션 변수에 배정
    - 연속 표현식내에 변수를 사용할 수 있음

순수 관계 연산(VIII)

- 복합연산(예, $\sigma_{A=C}(r \times s)$)
  - $(r \times s)$

릴레이션 r

A	B
α	1
β	2

릴레이션 s

C	D	E
α	10	+
β	10	+
β	20	-
γ	10	-

$r \times s$

A	B	C	D	E
α	1	α	10	+
α	1	β	10	+
α	1	β	20	-
α	1	γ	10	-
β	2	α	10	+
β	2	β	10	+
β	2	β	20	-
β	2	γ	10	-

- $\sigma_{A=C}(r \times s)$

A	B	C	D	E
α	1	α	10	+
β	2	β	10	+
β	2	β	20	-

배울 내용

- 관계 대수
  - 일반 집합 연산
  - 순수 관계 대수 연산
  - 확장 관계 대수 연산

- 데이터베이스의 수정
- 뷰

확장 관계 대수 연산(I)

- 일반화 추출 연산
  - 추출 리스트에 산술 함수를 사용 하도록 함으로써 추출 연산을 확장

$$\Pi_{F1,\ F2,\ \ldots,\ Fn}(E)$$

  - E는 관계형 대수 표현식이다.
  - $F_1,\ F_2,\ \ldots,\ F_n$ 각각은 E의 스키마 내에 상수와 애트리뷰트를 내포하고 있는 산술 표현식이다.
  - 주어진 릴레이션 *credit-info(customer-name, limit, credit-balance)* 에 대해 각 개인이 얼마까지 사용할 수 있는지를 찾아라.

$$\Pi_{\text{customer-name, limit} \ - \ \text{credit-balance}}(\text{credit-info})$$

확장 관계 대수 연산(II)

- 외부 조인(I)
  - 정보의 손실을 피하기 위한 조인 연산의 확장
  - 조인을 계산하고 다른 릴레이션의 튜플과 부합하지 않는 어떤 릴레이션의 튜플들을 조인의 결과에 추가
  - 널 값을 사용
    - 널은 알려지지 않은 값이나 존재하지 않는 값을 의미
    - 널을 내포한 모든 비교는 정의에 의해 거짓

확장 관계 대수 연산(III)

외부 조인 예제(I)

loan 릴레이션

branch-name	loan-number	amount
Downtown	L-170	3000
Redwood	L-230	4000
Perryridge	L-260	1700

borrower 릴레이션

customer-name	loan-number
Jones	L-170
Smith	L-230
Hayes	L-155

확장 관계 대수 연산(IV)

외부 조인 예제(II)

- *loan* ⟕ *borrower*

branch-name	loan-number	amount	customer-name
Downtown	L-170	3000	Jones
Redwood	L-230	4000	Smith

- *loan* ⟖ *borrower*

branch-name	loan-number	amount	customer-name
Downtown	L-170	3000	Jones
Redwood	L-230	4000	Smith
Perryridge	L-260	1700	null

확장 관계 대수 연산(V)

❑ 외부 조인 예제(III)

❑ *loan* ⟕ *borrower*

branch-name	loan-number	amount	customer-name
Downtown	L-170	3000	Jones
Redwood	L-230	4000	Smith
null	L-155	null	Hayes

❑ *loan* ⟖ *borrower*

branch-name	loan-number	amount	customer-name
Downtown	L-170	3000	Jones
Redwood	L-230	4000	Smith
Perryridge	L-260	1700	null
null	L-155	null	Hayes

확장 관계 대수 연산(VI)

i . 세미 조인 (Semijoin, ⋉)

- R(X), S(Y)의 조인 애트리뷰트를 Z=X∩Y라 하면

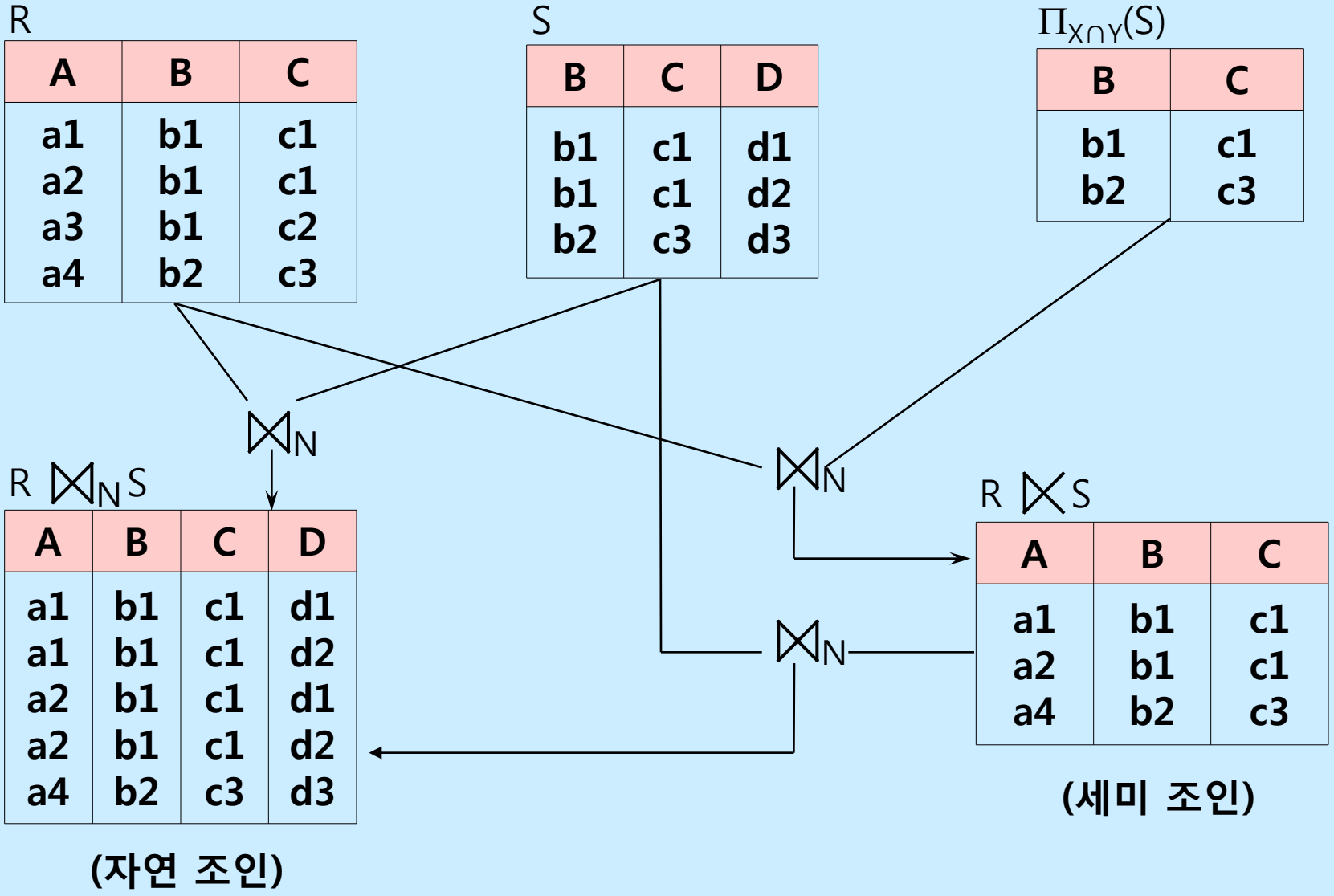
  $R \ltimes S = R \bowtie_N (\Pi_Z(S)) = \Pi_X(R \bowtie_N S)$

  - S와 자연조인을 할 수 있는 R의 투플

- 특징
  - $R \ltimes S \neq S \ltimes R$
  - $R \bowtie_N S = (R \ltimes S) \bowtie_N S = (S \ltimes R) \bowtie_N R$
    - 처리해야 될 데이타의 양이 다름

확장 관계 대수 연산(VII)

자연조인과 세미조인



확장 관계 대수 연산(VI)

- 집성 함수
  - 집성 연산자 $\mathcal{G}$ 는 값의 모임을 취해 하나의 값을 결과로 돌려준다.

    avg: 평균 값

    min: 최소 값

    max: 최대 값

    sum: 합계

    count: 값의 개수

$$G_1, G_2, \ldots, G_n \; \mathcal{G} \; F_1 A_1, F_2 A_2, \ldots, F_n A_n \; (E)$$

  - E는 관계형 대수 표현식
  - $G_1, G_2, \ldots, G_n$ 은 그룹핑할 애트리뷰트 리스트
  - F_i는 집성 함수
  - A_i는 애트리뷰트명

확장 관계 대수 연산(VII)

❑ 집성 함수 예(I)

릴레이션 r

A	B	C
α	α	7
α	β	7
β	β	3
β	β	10

$sum_c(r)$

sum-C
27

확장 관계 대수 연산(VIII)

❑ 집성 함수 예(II)

❑ 릴레이션 account를 branch-name으로 그룹핑한다.

branch-name	account-number	balance
Perryridge	A-102	400
Perryridge	A-201	900
Brighton	A-217	750
Brighton	A-215	750
Redwood	A-222	700

❑ $_{\text{branch-name}}\mathcal{G}_{\text{sum balance}}(\text{account})$

branch-name	sum-balance
Perryridge	1300
Brighton	1500
Redwood	700

배울 내용

- 관계 대수
  - 일반 집합 연산
  - 순수 관계 대수 연산
  - 확장 관계 대수 연산

- 데이터베이스의 수정
- 뷰

데이터베이스 수정(I)

- 데이터베이스의 내용은 다음 연산을 사용해 수정
  - 삭제
  - 삽입
  - 갱신
- 이들 모든 연산은 배정 연산자를 사용해 표현

데이터베이스 수정(II)

- 삭제
  - 삭제 요청은 질의와 유사하게 표현되나, 사용자에게 튜플을 출력하는 대신에, 선택한 튜플들이 데이터베이스에서 제거
  - 튜플을 통째로 삭제할 수 있지, 특정 애트리뷰트의 값 만을 삭제할 수는 없음
  - 삭제는 관계형 대수로 다음과 같이 표현

$$r \leftarrow r - E$$

여기서 r은 릴레이션이고, E는 관계형 대수 질의

데이터베이스 수정(III)

- 삭제 예
  - Perryridge 지점의 모든 계좌 레코드를 삭제하라.

    $account \leftarrow account - \sigma_{branch\text{-}name = \text{"Perryridge"}}(account)$

  - 0부터 50 사이의 금액을 가진 모든 대출 레코드를 삭제하라.

    $loan \leftarrow loan - \sigma_{amount \geq 0 \text{ and } amount \leq 50}(loan)$

  - Needham에 위치한 지점의 모든 계좌를 삭제하라.

    $r_1 \leftarrow \sigma_{branch\text{-}city = \text{"Needham"}}(account \bowtie branch)$

    $r_2 \leftarrow \Pi_{branch\text{-}name,\ account\text{-}number,\ balance}(r_1)$

    $r_3 \leftarrow \Pi_{customer\text{-}name,\ account\text{-}number}(r_2 \bowtie depositor)$

    $account \leftarrow account - r_2$

    $depositor \leftarrow depositor - r_3$

데이터베이스 수정(IV)

- 삽입
  - 릴레이션에 데이터를 삽입하려면
    - 삽입될 튜플을 지정
    - 결과가 삽입될 튜플의 집합인 질의를 작성
  - 관계형 대수에서, 삽입은 다음과 같이 표현

$$r \leftarrow r \cup E$$

여기서 r은 릴레이션이고, E는 관계형 대수 표현식

  - 단일 튜플의 삽입은 E를 하나의 튜플을 내포한 상수 릴레이션이 되도록하여 표현

데이터베이스 수정(V)

- 삽입 예
  - 데이터베이스에 Smith가 Perryridge 지점에 1,200불의 계좌 A-973을 가지고 있다는 정보를 삽입하라.

$$account \leftarrow account \cup \{\text{“Perryridge”}, A\text{-}973, 1200\}$$

$$depositor \leftarrow depositor \cup \{\text{“Smith”}, A\text{-}973\}$$

  - Perryridge 지점의 모든 대출 고객에게 200불의 저축 예금 계좌를 제공하고자 한다. 대출 번호를 새로운 저축 계좌의 계좌 번호로 하자.

$$r_1 \leftarrow (\sigma_{branch\text{-}name = \text{“Perryridge”}} (borrower \bowtie loan))$$

$$account \leftarrow account \cup \Pi_{branch\text{-}name,\ loan\text{-}number,\ 200}(r_1)$$

$$depositor \leftarrow depositor \cup \Pi_{customer\text{-}name,\ loan\text{-}number}(r_1)$$

데이터베이스 수정(VI)

- 갱신
  - 튜플 내의 모든 값을 바꾸지 않고 일부 값을 변경하는 기법
  - 일반화 추출 연산자를 사용

$$r \leftarrow \Pi_{F_1, F_2, \ldots, F_n}(r)$$

    - 각 F_i는 i번째 애트리뷰트가 갱신되지 않거나 갱신될 r의 i번째 애트리뷰트이다.
    - F_i는 애트리뷰트에 새로운 값을 주는 상수와 r의 애트리뷰트만을 내포하는 표현식이다.

데이터베이스 수정(VIII)

- 갱신 예
  - 모든 잔고에 5%의 이자를 지급하라.

$$account \leftarrow \Pi_{BN,\ AN,\ BAL \leftarrow BAL*1.05}(account)$$

    - 여기서 BAL, BN 및 AN은 각각 balance, branch-name 및 account-number를 나타낸다.

  - 10,000불을 초과하는 잔고에는 6%의 이자를 지급하고 그 외의 잔고에는 5%의 이자를 지급하라.

$$account \leftarrow \Pi_{BN,\ AN,\ BAL \leftarrow BAL*1.06}(\sigma_{BAL\ >\ 10000}(account))$$
$$\cup\ \Pi_{BN,\ AN,\ BAL \leftarrow BAL*1.05}(\sigma_{BAL\ \leq\ 10000}(account))$$

배울 내용

- 관계 대수
  - 일반 집합 연산
  - 순수 관계 대수 연산
  - 확장 관계 대수 연산

- 데이터베이스의 수정
- 뷰

뷰(View)

- 어떤 경우에도, 모든 사용자가 전체 논리 모델을 보는 것이 바람직하지 않음
  - 전체 논리 모델
    - 데이터베이스에 저장된 모든 실제 릴레이션

- 고객의 대출 번호는 알 필요가 있으나 대출 금액을 알 필요가 없는 개인을 고려해 보자. 이 사람은 다음과 같이 관계형 대수로 기술된 릴레이션을 보아야만 한다.

$$\Pi_{customer\text{-}name,\ loan\text{-}number}(borrower \bowtie loan)$$

- 개념적 모델의 일부는 아니지만 사용자에게 "가상 릴레이션"으로 보이는 릴레이션을 뷰라 한다

뷰(II)

- 뷰의 정의

  create view *v* as < 질의 표현식 >

  - 여기서 < 질의 표현식 >은 관계형 대수 질의 표현식
  - 뷰명은 *v*로 표현

- 일단 뷰가 정의되면, 뷰를 생성하는 가상 릴레이션을 참조하는데 뷰명이 사용될 수 있음

- 뷰 정의는 질의 표현식을 평가함으로써 새로운 릴레이션을 생성하는 것과는 다르게 뷰를 사용하는 질의내에 대치될 표현식을 저장

뷰(III)

- 뷰의 예
  - 지점과 고객으로 구성된 뷰(뷰명 : *all-customer*)를 고려해 보자.

    create view *all-customer* as

    $$\Pi_{branch\text{-}name,\ customer\text{-}name}\ (depositor \bowtie account)$$
    $$\cup\ \Pi_{branch\text{-}name,\ customer\text{-}name}\ (borrower \bowtie loan)$$

  - Perryridge 지점의 모든 고객은 다음과 같이 찾을 수 있다.

    $$\Pi_{customer\text{-}name}\ (\sigma_{branch\text{-}name\ =\ \text{“}Perryridge\text{”}}\ (all\text{-}customer))$$

요약

- 관계 대수
  - 일반 집합 연산
  - 순수 관계 대수 연산
  - 확장 관계 대수 연산
- 데이터베이스의 수정
- 뷰

다음 배울 내용 : 관계형 데이터베이스 설계