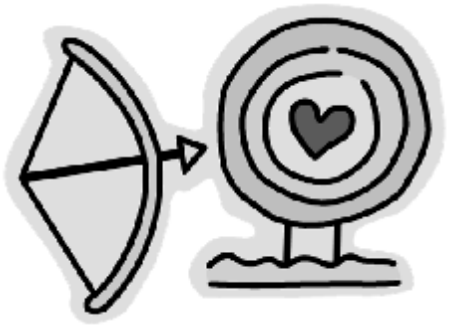
5장. 관계 대수(Relational Algebra)

- 관계 대수
- 백에 대한 관계 연산
- 관계 모델에 대한 그 밖의 확장
- 릴레이션에 대한 제약 조건



예제 데이타베이스

- **Movie**
  - **TITLE : string, YEAR : integer, length : integer, inColor : boolean**
  - **studioName : string, producerC# : integer**
- **StarsIn**
  - **MOVIETITLE : string, MOVIEYEAR : integer, STARNAME : string**
- **MovieStar**
  - **NAME : string, address : string, gender : char, birthdate : date**
- **MovieExec**
  - **name : string, address : string,**
  - **CERT# : integer, netWorth : integer**
- **Studio**
  - **NAME : string, address : string, presC# : integer**

관계 대수

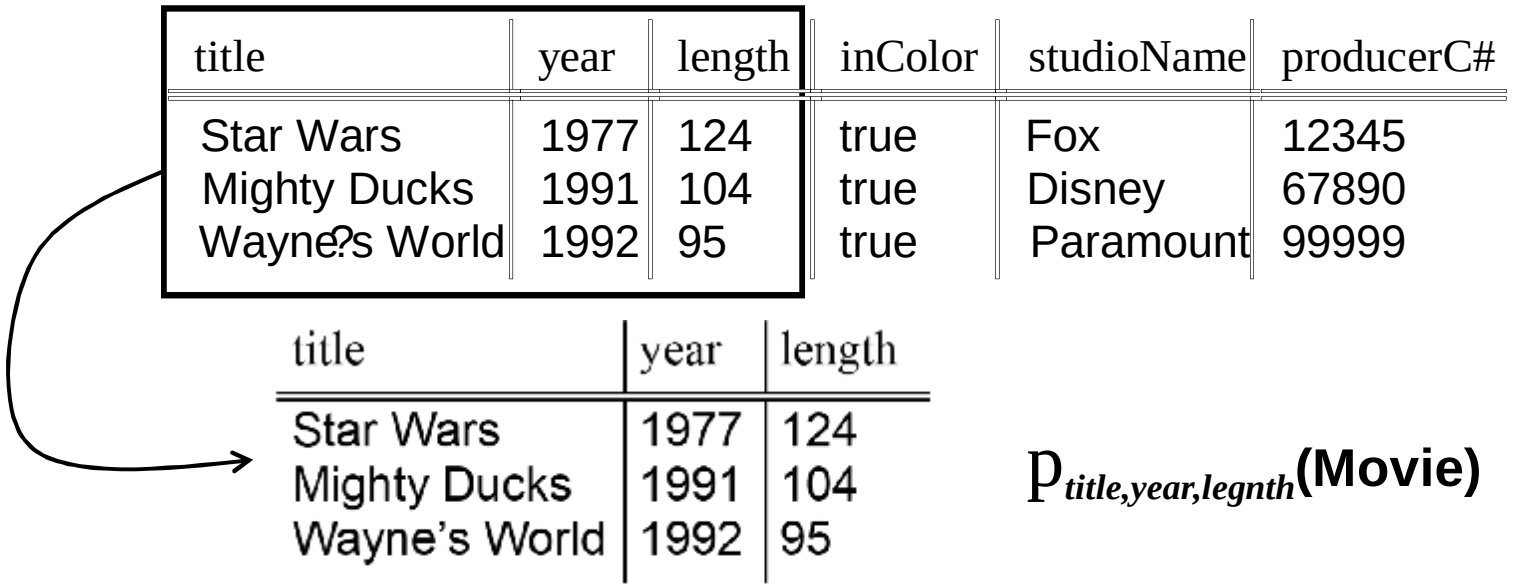
- 관계 대수**(relational algebra)**
  - 기존의 릴레이션에서 새로운 릴레이션을 만들어낼 수 있는 간단하고도 강력한 방법을 제공
  - 연산자와 피연산자를 사용한 수식
    - » 피연산자**(operand)**는 릴레이션임
- 관계 대수 연산
  - 보통의 집합 연산
  - 릴레이션의 일부를 제거하는 연산
  - 두 릴레이션의 튜플을 결합하는 연산
  - 이름 변경**(renaming)** 연산 **:** 스키마를 변경

관계 대수 (계속)

- **집합**
  - **합집합(union) : R È S**
  - **교집합(intersection) : R Ç S**
  - **차집합(difference) : R - S**
- **조건**
  - **R과 S의 스키마는 동일한 애트리뷰트 집합으로 구성되어야 한다.**
  - **R과 S의 열(column)은 두 릴레이션에서 모두 같은 순서로 정렬되어 있어야 한다.**
  - **R과 S는 각각 애트리뷰트가 1:1로 대응되어야 한다.**

관계 대수 (계속)

- **프로젝션(projection): p**
  - 릴레이션 **R**로부터, **R**의 열들 중 일부만으로 구성된 새로운 릴레이션을 생성한다.
  - **vertical(attributed-based) rearrangement**
  - $p_{A1,A2,...,An}(R)$



title	year	length	inColor	studioName	producerC#
Star Wars	1977	124	true	Fox	12345
Mighty Ducks	1991	104	true	Disney	67890
Wayne?s World	1992	95	true	Paramount	99999

title	year	length
Star Wars	1977	124
Mighty Ducks	1991	104
Wayne's World	1992	95

$p_{title,year,legnth}(Movie)$

관계 대수 (계속)

- **선택 연산(selection): σ**
  - **R**에 포함된 튜플들의 부분 집합으로 이루어진 주어진 조건식을 만족하는 릴레이션을 생성한다.
  - **horizontal(tuple-based) rearrangement**
  - $\sigma_C(R)$

title	year	length	inColor	studioName	producerC#
Star Wars	1977	124	true	Fox	12345
Mighty Ducks	1991	104	true	Disney	67890

$$\sigma_{length>100 \wedge inColr = true}(\text{Movie})$$

관계 대수 (계속)

- **카티션 프로덕트(cartesian product): ×**
  - 단순히 프로덕트라고도 한다.
  - 다음과 같이 구성되는 원소 쌍의 집합이다.
    - » 원소 쌍의 첫 번째 원소: **R**의 원소
    - » 원소 쌍의 두 번째 원소: **S**의 원소
  - **R×S**

A	B
1	2
3	4

R

×

B	C	D
2	5	6
4	7	8
9	10	11

S

Þ

A	R.B	S.B	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

R×S

관계 대수 (계속)

- **자연(natural) 조인: ⋈**
  - **R과 S에 공통으로 존재하는 애트리뷰트들이 동일한 값을 갖는 튜플들만을 결합한다.**
  - **R⋈S**

A	B	C	D
1	2	5	6
3	4	7	8

R ⋈ S

중복된 열들은
하나의 열로 남는다

A	B	C
1	2	3
6	7	8
9	7	8

U

⋈

B	C	D
2	3	4
2	3	5
7	8	10

V

Þ

A	B	C	D
1	2	3	4
1	2	3	5
6	7	8	10
9	7	8	10

U ⋈ V

관계 대수 (계속)

- **natural join의 용도**
  - 분해된 릴레이션의 재결합에 사용

[예] Movie(title, year, length, filmType, studioName, starName)을 다음과 같이 분해한 경우 두 릴레이션의 natural join의 결과는 Movie 릴레이션과 같다.

Movie1(title, year, length, filmType, studioName)

Movie2(title, year, starName)

£ Movie = Movie1 ⋈ Movie2

관계 대수 (계속)

- **세타(theta) 조인: $\bowtie_C$**
  - 임의의 조건을 만족하는 두 릴레이션의 튜플들을 결합한다.
    1. **R**과 **S**의 프로덕트를 구한다.
    2. 프로덕트 결과로부터, 조건 **C**를 만족하는 튜플만을 선택한다.
    3. $\sigma_C(R \times S)$

A	B	C
1	2	3
6	7	8
9	7	8

U

B	C	D
2	3	4
2	3	5
7	8	10

V

Þ

A	U.B	U.C	V.B	V.C	D
1	2	3	2	3	4
1	2	3	2	3	5
1	2	3	7	8	10
6	7	8	7	8	10
9	7	8	7	8	10

U $\bowtie_{A<D}$ **V**

중복된 열이 제거되지 않는다.

관계 대수 (계속)

- 질의 작성을 위한 연산의 조합
  - 릴레이션에 관계 연산자를 적용하여 생성된 릴레이션에 또 다시 관계 연산자를 적용할 수 있다.

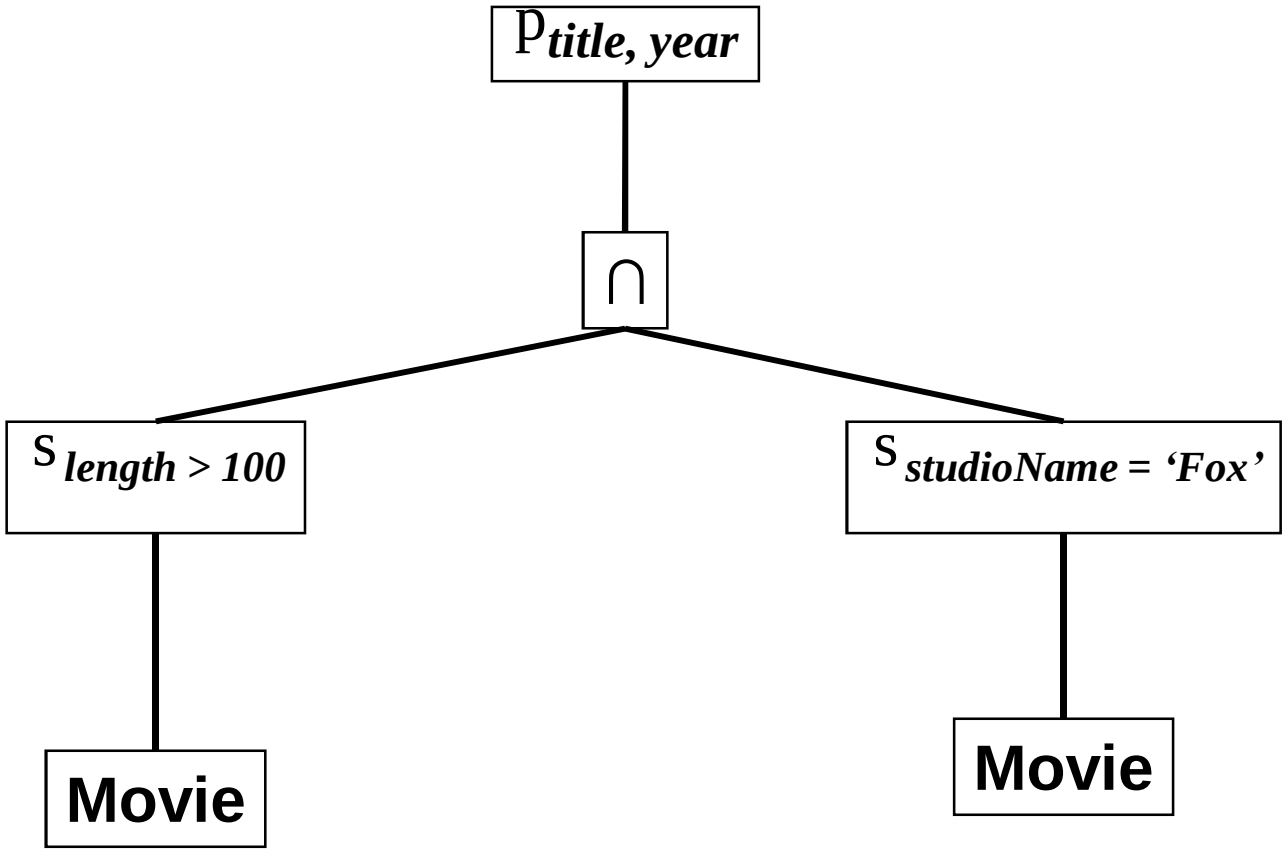
[예] 상영 시간이 100분 이상인 Fox사에서 제작된 영화의 제목과 연도를 찾아라.

- u $\Pi_{title, year}\ (\sigma_{length>100}(Movie) \cap \sigma_{studioName = 'Fox'}(Movie))$
- v $\Pi_{title, year}\ (\sigma_{length>100\ \text{AND}\ studioName = 'Fox'}(Movie))$

- 동치 수식(equivalent expression)
  - » 피연산자로 동일한 릴레이션이 주어지면, 항상 동일한 결과를 생성하는 수식들
  - » 질의 최적화(query optimization)

관계 대수 (계속)

Â 관계수식 트리



관계 대수 (계속)

- 이름 변경(renaming): $r_{S(A1,A2,...,An)}$(R)
  - 결과 릴레이션의 이름은 S가 되고, 애트리뷰트는 $A_1, A_2, ..., A_n$ 이 되지만, 튜플은 R과 동일하다.

A	B
1	2
3	4

R

B	C	D
2	5	6
4	7	8
9	10	11

S

Þ

A	B	X	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

R × $r_{S(X,C,D)}$(S)

관계 대수 (계속)

- 이름 변경의 사용 예
  - **MovieStar(name, address)** 스키마에 대하여
    - » 주소가 같은 스타의 이름을 찾아라.

$$\pi_{name,\ name2}\ (\sigma_{address=A}(\text{MovieStar} \times \rho_{M(name2,A)}\ (\text{MovieStar})))$$

name	name2
C. Fisher	M. Hamill
M. Hamill	C. Fisher
C. Fisher	C. Fisher
M. Hamill	M. Hamill

관계 대수 (계속)

- 2중으로 나타난 스타의 이름을 제거

$$\pi_{name,\, name2}\,(\sigma_{address=A \,\wedge\, name < name2}\,(\text{MovieStar} \times r_{M(name2,A)}\,(\text{MovieStar})))$$

name	name2
C. Fisher	M. Hamill
M. Hamill	C. Fisher
C. Fisher	C. Fisher
M. Hamill	M. Hamill

조건에 의해서 제거될 튜플들

관계 대수 (계속)

- **종속(dependent) 연산**
  - **R ∩ S = R - (R - S)**
  - **$R \bowtie_C S = \sigma_C(R \times S)$**
  - **$R \bowtie S = \pi_L(\sigma_C(R \times S))$ 등**
- **독립(independent) 연산**
  - **선택 연산, 프로젝션, 합집합, 차집합, 프로덕트, 이름 변경 연산을 위한 여섯 가지 연산자들은 하나의 독립적인 집합 을 구성한다.**

F 관계 데이터베이스에서의 기본(fundamental) 연산자

- **선택 연산, 프로젝션, 합집합, 차집합, 프로덕트**

관계 대수 예

- 상영시간이 **120**분 이상인 영화를 제작했던 영화사와 주소는 **?**
  - $\pi_{name,address}(\text{Movie} \bowtie_{studioname=name \wedge length > 120} \text{Studio})$
- **1970**년 이후 출생한 남자 배우가 출연한 영화의 제목, 상영 년도 및 상영시간은 **?**
  - $\pi_{title,year,length}($
    $(\text{MovieStar} \bowtie_{gender='M' \wedge name=starname \wedge birthdate \geq '1970/01/01'}$
    $\text{StarsIn}) \bowtie_{title=movietitle \wedge year=movieyear} \text{Movie})$
- 가장 부자인 제작자의 이름, 주소 및 재산액수는 **?**
  - $\pi_{name,address,networth}\text{MovieExec} - p_{name,address,networth}$
    $(\text{MovieExec} \bowtie_{networth < W} \rho_{M(N,A,C\#,W)}(\text{MovieExec}))$

대수 수식의 선형 표기법

- 치환문을 이용한 관계 수식의 표현법
  - $\pi_{title,\ year}\ (\sigma_{length>100}\ (Movie) \cap \sigma_{studioName = \text{'Fox'}}\ (Movie))$
    1. R(t, y, l, i ,s, p) := $\sigma_{studioName = \text{'Fox'}}\ (Movie)$
    2. S(t, y, l, i, s, p) := $\sigma_{length>100}\ (Movie)$
    3. T(t, y, l, i, s, p) := R ∩ S
    4. Answer(title, year) := $\pi_{t,\ y}\ (T)$

    } Answer(title, year) := $\pi_{t,\ y}\ (R \cap S)$

백에 대한 관계 연산

- 관계 데이터베이스에서의 백(bag)
  - 중복된(duplicated) 튜플이 허용되는 릴레이션
- 백의 용도
  - 프로젝션, 선택 연산 등의 수행 속도를 높일 수 있다.
    - » 집합에 대한 projection시에 중복된 튜플들을 모두 제거해야 한다.
    - » 백에서는 중복된 튜플들을 제거하는 overhead가 없다.
  - 집단값(aggregate)을 처리할 수 있다.
    - » R(A,B)와 같은 스키마에서 A의 값이 'a'인 튜플의 수를 계산할 때 중복된 값들도 모두 고려되어야 한다.
    - » 집합으로 처리하면 이런 연산이 힘들다.

백에 대한 관계 연산 (계속)

- **백의 합집합, 교집합, 차집합**
  - **R, S:** 튜플 **t**가 각각 **n, m**번 나타나는 백
    - **R ∪ S :** 튜플 **t**는 **n + m** 번 나타난다.
    - **R ∩ S :** 튜플 **t**는 **min(n, m)** 번 나타난다.
    - **R - S :** 튜플 **t**는 **max(0, n-m)** 번 나타난다.

백에 대한 관계 연산 (계속)

A	B
1	2
3	4
1	2
1	2

R

A	B
1	2
3	4
3	4
5	6

S

R∪S

A	B
1	2
1	2
1	2
1	2
3	4
3	4
3	4
5	6

A	B
1	2
3	4

R∩S

A	B
1	2
1	2

R - S

A	B
3	4
5	6

S - R

집합에 대한 백 연산

- 집합 R과 S를 생각해 보자. 모든 집합은 각 튜플이 한번씩만 나타나는 백으로 생각할 수 있다.
  - R과 S에 백 교집합 과 백 차집합을 적용한 결과는, R과 S에 집합에 기반을 둔 교집합과 차집합의 결과와 동일하다.
  - 집합을 백으로 표현시 아래의 n과 m은 0 또는 1
    - » R ∩ S : 튜플 t는 min(n, m) 번 나타난다
    - » R - S : 튜플 t는 max(0, n-m) 번 나타난다
  - 합집합 은 R과 S가 집합인지, 혹은 백인지 여부에 따라 결과가 달라진다.
    - » R∪S : 튜플 t는 n + m 번 나타나야 하므로.. n=m=1인 경우 튜플 t는 집합에서 2번 나타날 수 있다.

백에 대한 관계 연산 (계속)

- 백의 프로젝션
  - 프로젝션 시 여러 튜플들로부터 동일한 튜플이 중복되어 생성될 수 있다.
    - » 중복된 튜플은 결과에서 제거되지 않는다.
  - 집합 프로젝션에서는 한 튜플은 오직 한 번만 나타난다.
- 백의 선택 연산
  - 각 튜플은 독립적으로 선택 조건에 적용된다.

R

A	B	C
1	2	5
3	4	6
1	2	7
1	2	7

Þ

A	B	C
3	4	6
1	2	7
1	2	7

$\sigma_{C \geq 6}(R)$

백에 대한 관계 연산 (계속)

- **백의 프로덕트**
  - 중복에 상관없이 릴레이션의 한 튜플은 다른 릴레이션의 각 튜플들과 쌍을 이룬다.
  - R의 튜플 *r*이 *m*번 나타나고 S의 튜플 *s*가 *n* 번 나타나면, R×S에는 튜플 *rs*가 m×n 번 나타난다.

A	B
1	2
1	2

R

B	C
2	3
4	5
4	5

S

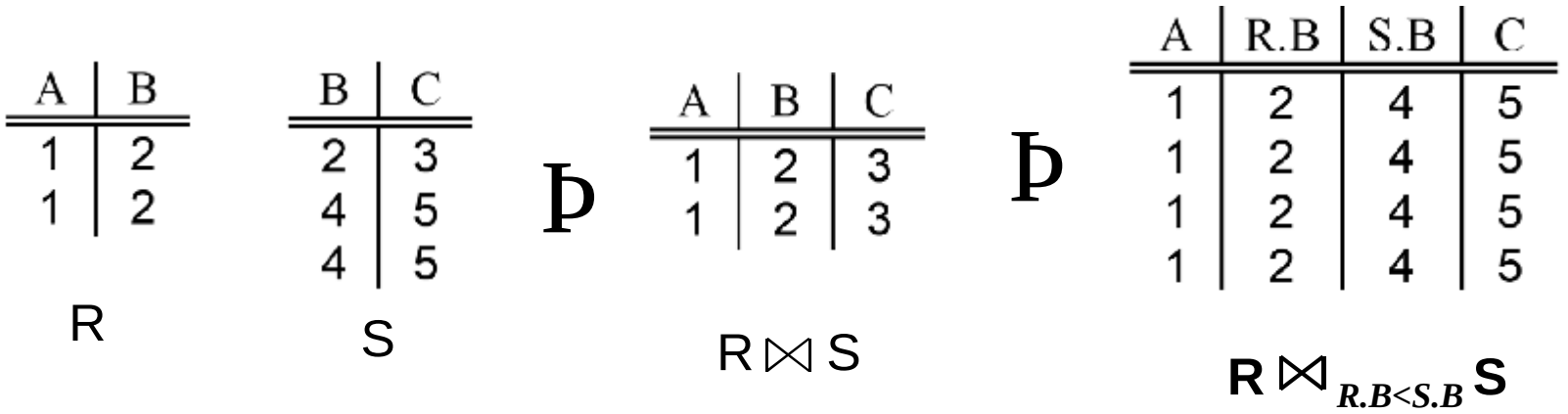
Þ

A	R.B	R.C	C
1	2	2	3
1	2	2	3
1	2	4	5
1	2	4	5
1	2	4	5
1	2	4	5

R×S

백에 대한 관계 연산 (계속)

- **백의 조인연산**
  - 릴레이션의 한 튜플을 다른 릴레이션의 각 튜플들과 비교하여, 두 튜플이 조인될 수 있는지 여부를 조사한다.



참고: 백에 대한 대수 법칙

- 대수 법칙**(algebraic law)**이란 릴레이션을 나타내는 변수를 인수로 갖는 두 관계 대수 식 사이의 동치**(equivalence)**를 의미한다.
- 어떤 대수 법칙은 집합이나 백에 상관없이 성립한다.

  $R \cup S \equiv S \cup R$: 교환법칙**(commutative law)**

  $(R \cup S) \cup T \equiv R \cup (S \cup T)$: 결합법칙**(assosiative law)**

  $(R \bowtie S) \equiv (S \bowtie R)$, $(R \bowtie S) \bowtie T \equiv R \bowtie (S \bowtie T)$
- 어떤 대수 법칙은 릴레이션이 집합으로 해석되는 경우에는 성립하나, 백으로 해석되는 경우에는 성립하지 않는다. : 배분법칙**(distributive law)**

  $(R \cup S) - T \not\equiv (R - T) \cup (S - T)$

  $(R \cap S) - T \not\equiv R \cap (S - T)$

  $R \cap (S \cup T) \not\equiv (R \cap S) \cup (R \cap T)$

관계대수 연산의 확장

- 중복 제거
  - **δ(R) :** 릴레이션 **R**의 중복된 튜플들을 제거
- 집단화 연산**(aggregation)**
  - 한 릴레이션의 모든 튜플들을 결합하여 임의의 집단 값**(*aggregate* value)**을 생성
  - 집단화 연산 **:** 개수**(count),** 합**(sum),** 평균**(average),** 최소**(minimum),** 최대**(maximum)**

A	B
1	2
3	4
1	2
1	2

- u SUM(B) = 2+4+2+2 = 10
- v AVG(A) = (1+3+1+1)/4 = 1.5
- w MIN(A) = 1
- x MAX(B) = 4
- y COUNT(A) = 4

관계대수 연산의 확장(계속)

u 그룹화(grouping)
- 특정 애트리뷰트들에 대해서 같은 값을 가지는 튜플들로 그룹화한다.
- 그룹화된 각 그룹에 대해서 집단화 연산을 적용

A	B	C
a1	b1	1
a2	b2	2
a1	b3	3
a1	b4	2
a2	b2	2
a3	b0	3

u A값이 같은 튜플들의 개수(count)는 ?

a1 = 3, a2 = 2, a3 = 1

v A값이 같은 튜플들에 대해서 C의 평균값 (average)은 ?

a1 = 2, a2 = 2, a3 = 3

w A값이 같은 튜플들에 대해서 C의 평균값이 3이상인 각 튜플들의 개수는 ?

a3 = 1

관계대수 연산의 확장(계속)

u 그룹화 연산자

- g_L**(R) : 릴레이션 R에 대해서 그룹화 연산을 한다.**
  - » **엘리먼트 리스트 L의 구성**
    - ① **릴레이션 R에 대해서 그룹화할 애트리뷰트들**
    - ② **집단화 연산자와 집단화 연산 결과값을 의미하는 이름 부여**

u 예제

- **"최소한 3편의 영화에 출연했던 스타에 대해 그들이 처음으로 출연했던 영화의 상영년도는?"**
- **StarsIn(title, year, starName)**에 대해서 다음과 같은 그룹화 연산식

$$\pi_{\text{starName, minYear}}(\sigma_{\text{cTitle >= 3}}(g_{\text{starName, MIN(year)}\rightarrow\text{minYear, COUNT(title)}\rightarrow\text{cTitle}}(\text{StarsIn}))$$

관계대수 연산의 확장(계속)

- 프로젝션 연산의 확장
  - π_L(R) : L의 의미를 다음과 같이 확장
    - R의 단일 애트리뷰트
    - x → y : 애트리뷰트 x를 y로 이름 변경
    - E → z : E는 R의 애트비뷰트, 상수, 산술 연산, 문자열 연산 등을 포함하는 수식이고 계산 결과에 부여되는 이름은 z가 된다.
      - 예를 들어, a + b → x는 애트리뷰트 a와 b를 더해서 x로 이름을 부여한다.
      - c || d → e는 애트리뷰트 c와 d를 결합(concatination)하여 e로 이름을 부여한다.

$$\pi_{title \rightarrow MovieTitle,\ year,\ lengh/60 \rightarrow LenghinHour}(Movie)$$

- 정렬 연산자
  - 임의의 애트리뷰트들을 기준으로 튜플들을 정렬

$$\tau_{year,\ length}\ (Movie)$$

외부조인(outer join)

A	B	C
1	2	3
2	3	4
5	6	1

R

C	D	E
4	5	1
2	1	3
3	4	2

S

A	B	C	D	E
1	2	3	4	2
2	3	4	5	1
5	6	1	-	-

R ⟕ S : left outer join

A	B	C	D	E
-	-	2	1	3
1	2	3	4	2
2	3	4	5	1

R ⟖ S : right outer join

A	B	C	D	E
1	2	3	4	2
2	3	4	5	1

R ⋈ S : natural join

A	B	C	D	E
1	2	3	4	2
2	3	4	5	1
5	6	1	-	-
-	-	2	1	3

R ⟗ S : full outer join

릴레이션에 대한 제약

u 제약 조건을 표현하는 두 가지 방법

R, S : 관계 대수 수식

① **R = Æ : "R**의 결과에는 어떤 튜플도 존재하지 않는다"는 것을 의미하는 제약

② **R Í S: "R**의 결과에 속하는 모든 튜플은 **S**의 결과에도 속해야 한다"는 것을 의미하는 제약

- 위의 제약과 동치인 여러 다른 표현법이 존재한다.
  - » **R Í S** 는 **R - S = Æ** 로도 표현될 수 있다.
  - » **R = Æ** 는 **R Í Æ** 로도 표현될 수 있다.

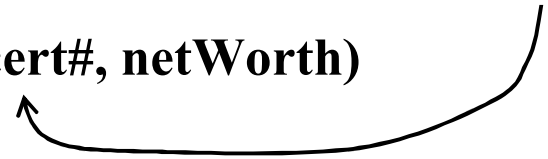
릴레이션에 대한 제약 (계속)

- **참조 무결성 제약**
  - **ODL :** 임의의 객체 **A**가 객체 **B**와 연관되어 있으면, **B**는 실제로 존재하여야 한다.
  - 관계 모델: 릴레이션 **R**의 어떤 애트리뷰트에 값 ***v***가 있으면, ***v***는 릴레이션 **S**에서 어떤 튜플의 특정 애트리뷰트 값으로 나타나야 한다.

[예] 각 영화의 제작자는 반드시 **MovieExec** 릴레이션에 나타나야 한다.

Movie(title, year, length, inColor, studioName, producerC#)

MovieExec(name, address, cert#, netWorth)



$$\pi_{producerC\#}(\text{Movie}) \subseteq \pi_{cert\#}(\text{MovieExec})$$

$$\pi_{producerC\#}(\text{Movie}) - \pi_{cert\#}(\text{MovieExec}) = \varnothing$$

릴레이션에 대한 제약 (계속)

- **함수적 종속성**

  [예] MovieStar(name, address, gender, birthdate)

  - **name ® address**

  $\sigma_{MS_1.name=MS_2.name \text{ AND } MS_1.address \neq MS_2.address}(MS_1 \times MS_2) = \emptyset$

  MS_i: $\rho_{MS_i(name,address,gender,birthdate)}(MovieStar)$

- **도메인 제약**

  [예] MovieStar 릴레이션에서 gender 애트리뷰트의 값으로 'F'와 'M'만 허용한다.

  $\sigma_{gender \neq 'F' \text{ AND } gender \neq 'M'}(MovieStar) = \emptyset$

릴레이션에 대한 제약 (계속)

- **그 밖의 제약**

  [예] 영화 스튜디오의 사장이 되기 위해서는 적어도 $10,000,000의 재산(net worth)을 소유하고 있어야 한다. (도메인, 단일 값, 혹은 참조 무결정 제약과는 다른 형태이다.)

MovieExec(name, address, cert#, netWorth)

Studio(name, address, presC#)

$$\sigma_{netWorth<10000000}(\text{Studio} \bowtie_{presC\#=cert\#} \text{MovieExec}) = \emptyset$$

$$\pi_{presC\#}(\text{Studio}) - \pi_{cert\#}(\sigma_{netWorth \geq 10000000}(\text{MovieExec}))$$