DB07

Relational algebra

Relational Database에 적용할 수 있는 Operation의 집합

- 보통 Relational Database에서 **원하는 것을 Retrieve**하기 위해 사용한다.
- **Relational algebra를 적용한 결과 역시 Relation (Table)**이다.

Select

$\sigma_{<selectionCondition>}(R)$ 로 표현한다.

- R: Relation
- Selection Condition에 맞는 것을 Select한다.

Relation R로부터 Select Condition을 만족하는 **Subset of the tuples을 골라낸다.**

- **Selection condition**은 **Boolean expression으로 표현한다.**
  - <attribute name> <comparison op> <constant value>
  - <attribute name> <comparison op> <atrribute name>
- $r(R)$으로부터 **Selection condition을 만족하는 Tuple을 포함하는 Relation**을 반환한다.

Company database example

$$\sigma_{\text{Dno}=4}(\text{EMPLOYEE})$$
$$\sigma_{\text{Salary}>30000}(\text{EMPLOYEE})$$

$$\sigma_{(\text{Dno}=4\ \textbf{AND}\ \text{Salary}>25000)\ \textbf{OR}\ (\text{Dno}=5\ \textbf{AND}\ \text{Salary}>30000)}(\text{EMPLOYEE})$$

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5

PROJECT

$\pi_{<attributeList>}(R)$

- Relation R에서 attiribute list에 존재하는 attribute만 보여준다.
- **Tuple의 개수는 유지**한다.
- **Column**만 선택한다.

$r(R)$으로부터 **Attribute list에 있는 Attribute를 갖는 Tuples를 포함한 Relation**을 반환한다.

PROJECT의 결과로 몇몇 **Column만 선택되기 때문에 서로 다른 Tuple의 모든 값이 중복될 수 있다.**

- 이 경우 **중복되는 Tuple은 하나만 사용한다**
- 이 경우에는 Tuple의 개수가 변경될 수 있다.
    - Attribute list에 두 Tuple을 구별하던 Attribute가 포함되지 않았기 때문에 발생한다.

Company database example

$\pi_{Lname,\ Fname,\ Salary}(EMPLOYEE)$

Lname	Fname	Salary
Smith	John	30000
Wong	Franklin	40000
Zelaya	Alicia	25000
Wallace	Jennifer	43000
Narayan	Ramesh	38000
English	Joyce	25000
Jabbar	Ahmad	25000
Borg	James	55000

$\pi_{Sex,\ Salary}(EMPLOYEE)$

Sex	Salary
M	30000
M	40000
F	25000
F	43000
M	38000
M	25000
M	55000

- 이 경우 중복이 생겨서 원래의 Tuple의 개수보다 감소한 것을 확인할 수 있다.

Relational Algebra Expressions

Relational algebra operation의 Sequence

복잡한 Query도 Relational algebra expression으로 표현할 수 있다.

- **하나의 Relational algebra의 결과 역시 Relation이기 때문**이다.
- **Relational algebra가 Relation을 입력받아 Relation을 출력**하기 때문에 연쇄적인 적용이 가능하다.

Company database example

- 자연어로 된 Query를 Relational algebra expression으로 표현할 수 있다.

"Retrieve the first name, last name, and salary of all employees who work in department number 5"

- $\pi_{\text{Fname, Lname, Salary}}(\sigma_{\text{Dno}=5}(\text{EMPLOYEE}))$

Each intermediate result can have a relation name

- $\text{DEP5\_EMPS} \leftarrow \sigma_{\text{Dno}=5}(\text{EMPLOYEE})$
  $\text{RESULT} \leftarrow \pi_{\text{Fname, Lname, Salary}}(\text{DEP5\_EMPS})$

- **Intermediate relation**을 두고 Aggisn해도 된다.
- 임시 테이블을 만든다.
  - 임시 테이블 역시 Relation name을 가질 수 있다.

Each attribute of a result relation can be *renamed*

- $\text{TEMP} \leftarrow \sigma_{\text{Dno}=5}(\text{EMPLOYEE})$
  $R(\text{First\_name, Last\_name, Salary}) \leftarrow \pi_{\text{Fname, Lname, Salary}}(\text{TEMP})$

- TEMP라는 **임시 Table**을 만들고, Attribute를 **Rename**도 할 수 있다.
  - Fname → First_name, Lname → Last_name, Salary → Salary

Set Operations

UNION operation

- $R_1 \cup R_2$

INTERSECTION operation

- $R_1 \cap R_2$

SET DIFFERENCE (MINUS) operation

- $R_1 - R_2$

CARTESIAN PRODUCT (CROSS PRODUCT) operation

- $R_1 \times R_2$

Company database example

$$DEP5\_EMPS \leftarrow \sigma_{Dno=5}(EMPLOYEE)$$
$$RESULT1 \leftarrow \pi_{Ssn}(DEP5\_EMPS)$$
$$RESULT2(Ssn) \leftarrow \pi_{Super\_ssn}(DEP5\_EMPS)$$
$$RESULT \leftarrow RESULT1 \cup RESULT2$$

RESULT1

Ssn
123456789
333445555
666884444
453453453

RESULT2

Ssn
333445555
888665555

RESULT

Ssn
123456789
333445555
666884444
453453453
888665555

- RESULT1은 Dno가 5인 Ssn값을 뽑은 것이다.
- RESULT2는 Super_ssn을 Ssn으로 Rename하여 뽑은 것이다.

- RESULT3는 RESULT1과 RESULT2의 **UNION**이다.
    - UNION은 **중복되는 수 (333445555)는 하나만 표시**한다.

Union compatibility (type compatibility)

UNION, INTERSECTION, SET DIFFERENCE에 적용된다.

- **각 Set operation이 의미가 있으려면** Union compatibility를 만족해야 한다.

위 세 가지 Set operation을 적용하기 위해선 **두 Relation(Table)이 같은 Type의 Tuple을 가져야 한다.**

1. **두 Relation은 같은 개수의 Attribute를 가져야 한다.**
2. **두 Relation에서 서로 대응되는 각 Attribute 간의 Domain이 같아야 한다.**
    - $Dom(A_i) = DOM(B_i)(1 <= i <= n)$
    - **Data type이 Compatible해야 한다.**
    - Attribute의 이름 자체는 달라도 되고 **값의 범위 자체**는 조금 달라도 **Compatible**하다고 한다.

호환되는 (Compatible) 예시:

- `SMALLINT` (2바이트 정수) 와 `INTEGER` (4바이트 정수)
    - → 둘 다 '숫자'이며, 시스템이 더 큰 범위인 `INTEGER`로 결과를 합칠 수 있습니다.
- `CHAR(10)` (고정 길이 10) 과 `VARCHAR(20)` (가변 길이 20)
    - → 둘 다 '문자열'이며, 시스템이 더 유연한 `VARCHAR(20)`으로 결과를 합칠 수 있습니다.
- `DECIMAL(10, 2)` (소수점 2자리) 와 `INTEGER` (정수)
    - → 둘 다 '숫자' 계열이며, 시스템이 정수를 소수점 (예: 123 → 123.00)으로 변환하여 합칠 수 있습니다.

UNION

Two union compatible relations

STUDENT

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

- Fn과 Fname 모두 String type / Ln과 Lname 모두 String type이므로 **Compatible** 하다.

STUDENT ∪ INSTRUCTOR

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert
John	Smith
Ricardo	Browne
Francis	Johnson

- UNION의 결과로 생성되는 **Relation의 Attribute name은 첫 번째 Relation의 attribute name으로 결정된다.**

INTERSECTION

STUDENT ∩ INSTRUCTOR

Fn	Ln
Susan	Yao
Ramesh	Shah

- Attribute name은 첫 번째 Table 기준이다.

SET DIFFERENCE

STUDENT - INSTRUCTOR

Fn	Ln
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

- SET DIFFERENCE는 **Non-Symmetric하다.**
- A - B와 B - A의 결과가 다르다.
- Attribute name은 첫 번째 Table 기준이다.

INSTRUCTOR - STUDENT

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

CARTESIAN PRODUCT

R X S 또는 $R(A_1, A_2, \ldots, A_N)$ x $S(B_1, B_2, \ldots, B_M)$ 이라고 쓴다.

- 결과로 나오는 Relation은 $Q(A_1, A_2, \ldots, A_N, B_1, B_2, \ldots, B_M)$이다.
- **Attribute의 개수**는 **N+M**이다.
- **Tuple의 개수**는 **(R의 Tuple 개수) x (S의 Tuple 개수)**이다.

가능한 모든 Tuple의 조합이다.

CARTESIAN PRODUCT는 **보통 SELECTION과 같이 사용되어야 의미가 있다.**

- CARTESIAN PRODUCT는 가능한 모든 Tuple의 조합을 만드는데, **그 자체로는 아무 의미가 없다.**
- CARTESIAN PRODUCT**이후에 SELECTION을 함으로써 의미가 생긴다.**

SELECT는 하나의 Relation에만 적용된다. 이때, **서로 다른 두 Table에서 뭔가를 Select하고 싶다면 SET OPERATION을 이용하여 하나의 Table**로 만든 후, **SELECT를 적용**해야 한

다. UNION 등의 경우 **UNION compatibility 조건이 걸려있기 때문에 까다로워 보통 CARTESIAN PRODUCT**를 이용한다.

Company database example

- Dependent에는 Essn을 통해 어떤 EMPLOYEE인지 확인할 수 있는 식별 번호가 있지만, EMPLOYEE에 대한 구체적인 정보는 없다.
- 여기서 원하는 정보는 **여자 Employee의 dependent의 이름**이기 때문에, **EMPLOYEE table에서 여자인 사람을 고르고, DEPENDENT table과 CARTESIAN PRODUCT**를 한 이후에 **Ssn과 Essn이 같은 Tuple을 Select**하면 된다.

- "Retrieve a list of names of each female employee's dependents"

$$\text{FEMALE\_EMPS} \leftarrow \sigma_{\text{Sex='F'}}(\text{EMPLOYEE})$$
$$\text{EMPNAMES} \leftarrow \pi_{\text{Fname, Lname, Ssn}}(\text{FEMALE\_EMPS})$$
$$\text{EMP\_DEPENDENTS} \leftarrow \text{EMPNAMES} \times \text{DEPENDENT}$$
$$\text{ACTUAL\_DEPENDENTS} \leftarrow \sigma_{\text{Ssn=Essn}}(\text{EMP\_DEPENDENTS})$$
$$\text{RESULT} \leftarrow \pi_{\text{Fname, Lname, Dependent\_name}}(\text{ACTUAL\_DEPENDENTS})$$

이 과정을 하나씩 살펴보자.

$$\text{FEMALE\_EMPS} \leftarrow \sigma_{\text{Sex='F'}}(\text{EMPLOYEE})$$
$$\text{EMPNAMES} \leftarrow \pi_{\text{Fname, Lname, Ssn}}(\text{FEMALE\_EMPS})$$

FEMALE_EMPS

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Alicia	J	Zelaya	999887777	1968-07-19	3321Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291Berry, Bellaire, TX	F	43000	888665555	4
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5

EMPNAMES

Fname	Lname	Ssn
Alicia	Zelaya	999887777
Jennifer	Wallace	987654321
Joyce	English	453453453

$$EMP\_DEPENDENTS \leftarrow EMPNAMES \times DEPENDENT$$

EMP_DEPENDENTS

Fname	Lname	Ssn	Essn	Dependent_name	Sex	Bdate	...
Alicia	Zelaya	999887777	333445555	Alice	F	1986-04-05	...
Alicia	Zelaya	999887777	333445555	Theodore	M	1983-10-25	...
Alicia	Zelaya	999887777	333445555	Joy	F	1958-05-03	...
Alicia	Zelaya	999887777	987654321	Abner	M	1942-02-28	...
Alicia	Zelaya	999887777	123456789	Michael	M	1988-01-04	...
Alicia	Zelaya	999887777	123456789	Alice	F	1988-12-30	...
Alicia	Zelaya	999887777	123456789	Elizabeth	F	1967-05-05	...
Jennifer	Wallace	987654321	333445555	Alice	F	1986-04-05	...
Jennifer	Wallace	987654321	333445555	Theodore	M	1983-10-25	...
Jennifer	Wallace	987654321	333445555	Joy	F	1958-05-03	...
Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	...
Jennifer	Wallace	987654321	123456789	Michael	M	1988-01-04	...
Jennifer	Wallace	987654321	123456789	Alice	F	1988-12-30	...

$$ACTUAL\_DEPENDENTS \leftarrow \sigma_{Ssn=Essn}(EMP\_DEPENDENTS)$$
$$RESULT \leftarrow \pi_{Fname,\ Lname,\ Dependent\_name}(ACTUAL\_DEPENDENTS)$$

ACTUAL_DEPENDENTS

Fname	Lname	Ssn	Essn	Dependent_name	Sex	Bdate	...
Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	...

RESULT

Fname	Lname	Dependent_name
Jennifer	Wallace	Abner

JOIN Operation

서로 다른 Relation에 저장되어져 있는 Tuple간의 Relationship을 처리하기 위해 필요한 Operation

- JOIN이 없다면 **항상 CARTESIAN → SELECT을 이용**해야 한다.
- JOIN Operation을 통해 위 작업을 한번에 처리한다.

Company database example

- DEPARTMENT Relation에는 Manager를 식별할 수 있는 Mgr_ssn이 있지만 대응되는 Manager의 정보를 담는 Attribute는 존재하지 않는다.
- 때문에 기존 CARTESIAN PRODUCT를 사용한다고 가정하면 EMPLOYEE와 DEPARTMENT를 CARTESIAN PRODUCT한 이후에 $\sigma_{Mgrssn=Ssn}$을 하면 된다.

- "Retrieve the name of the manager of each department"

DEPT_MGR

Dname	Dnumber	Mgr_ssn	...	Fname	Minit	Lname	Ssn	...
Research	5	333445555	...	Franklin	T	Wong	333445555	...
Administration	4	987654321	...	Jennifer	S	Wallace	987654321	...
Headquarters	1	888665555	...	James	E	Borg	888665555	...

- JOIN을 이용하면 **조건에 맞게 DEPARTMENT와 EMPLOYEE를 JOIN**하고 **원하는 Column만 얻기 위해 PROJECT**하면 된다.

$$DEPT\_MGR \leftarrow DEPARTMENT \bowtie_{Mgr\_ssn=Ssn} EMPLOYEE$$
$$RESULT \leftarrow \pi_{Dname,\ Lname,\ Fname}(DEPT\_MGR)$$

JOIN의 장점

1. **Query 작성 시 빠르다.**
2. JOIN 대신 CARTESIAN PRODUCT를 사용하지 않기 때문에 **CARTESIAN PRODUCT의 결과로 나오는 Table을 추가로 디스크에 저장할 필요가 없어 처리 속도가 빠르다.**

THETA JOIN Operation

$R \bowtie_{<joincondition>} S$

Join condition을 만족하는 모든 Tuple의 쌍을 갖는 새로운 Relation을 생성한다.

- CARTESION PRODUCT → SELECT와 동일하다.

Join condition

- <condition> AND <condition> AND <condition>
- <condition> = $A_i\ \theta\ B_j$
- θ = {= , <, ≤, >, ≥, ≠}

EQUIJOIN Operation

= Comparison만 사용하는 JOIN OPERATION

- THETA Operatoin의 θ = {=}인 특별한 케이스이다.

= Comparison이기 때문에 결과로 나오는 Relation은 같은 값을 가지는 두 개 Attribute를 갖는다.

- 중복을 제거하거나 하지 않는다.

NATURAL JOIN Operation

$R * S$으로 표현한다.

EQUIJOIN에서 = Comparison이기 때문에 결과로 나오는 Relation은 같은 값을 가지는 두 개 Attribute를 갖는 것을 **Unnatural하다.**

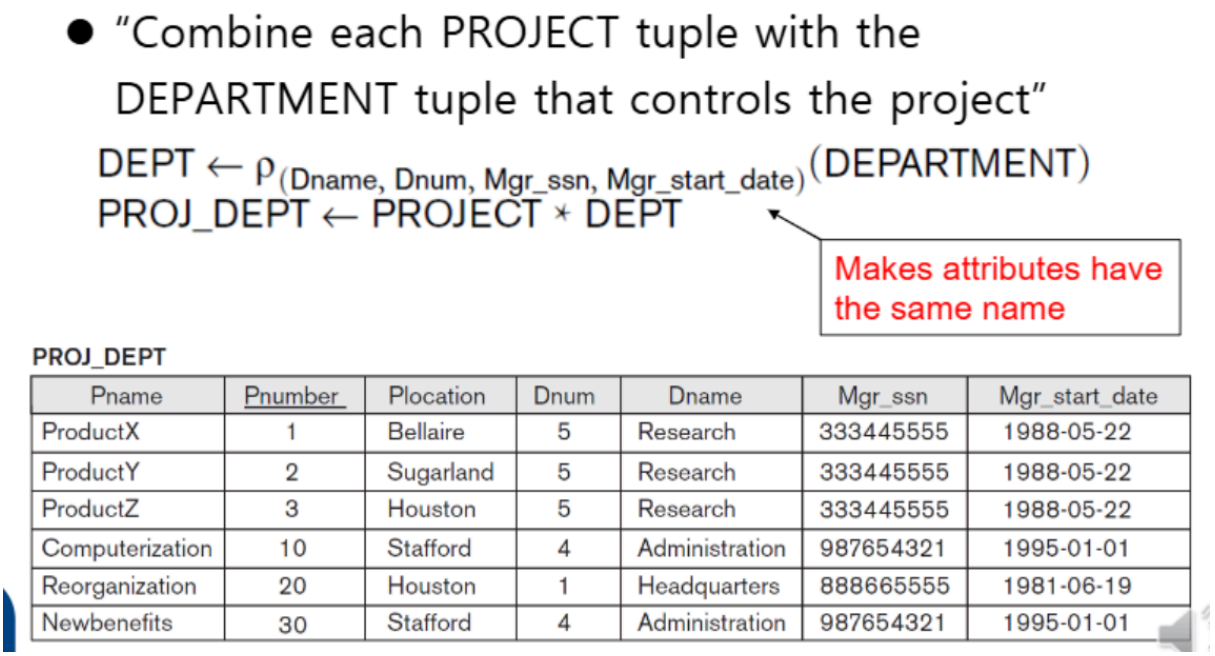
결과로 나오는 Relation에서 EQUIJOIN에서 중복되는 값의 두 번째 Attribute는 제거하는 방법이다.

- **이를 위해선 JOIN 하기 이전의 두 Table의 Attribute name까지 동일해야 한다.**
- Natural하다!

과정

1. 서로 다른 두 Table에서 먼저 Attribute name이 같은 모든 Attribute를 찾는다.
2. (1)에서 찾은 Attribute pair에서 값이 동일한 것을 찾는다.
    - 이때 찾은 **Attribute pair가 여러 쌍이라면 여러 쌍 모두 값이 같아야 한다.**
3. (2)에서 찾은 값이 동일한 Tuple만 남기고 두 번째 Attribute는 제거한다.

Company database example



PROJ_DEPT

Pname	Pnumber	Plocation	Dnum	Dname	Mgr_ssn	Mgr_start_date
ProductX	1	Bellaire	5	Research	333445555	1988-05-22
ProductY	2	Sugarland	5	Research	333445555	1988-05-22
ProductZ	3	Houston	5	Research	333445555	1988-05-22
Computerization	10	Stafford	4	Administration	987654321	1995-01-01
Reorganization	20	Houston	1	Headquarters	888665555	1981-06-19
Newbenefits	30	Stafford	4	Administration	987654321	1995-01-01

- ρ**(Dname, Dnum, Mgr_ssn, Mge_start_date)(DEPARTMENT)**: DEPARTMENT의 Attribute name을 **rename**
    - PROJECT의 Dnum과 맞추기 위해 맞춘다.
    - Attribute의 이름이 동일해야 하기 때문이다.
- 결과로 나오는 Dnum attribute는 하나만 존재하는 것을 확인할 수 있다.

- "Combine each DEPARTMENT tuple with its location"

$$DEPT\_LOCS \leftarrow DEPARTMENT * DEPT\_LOCATIONS$$

Already have same name

DEPT_LOCS

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Location
Headquarters	1	888665555	1981-06-19	Houston
Administration	4	987654321	1995-01-01	Stafford
Research	5	333445555	1988-05-22	Bellaire
Research	5	333445555	1988-05-22	Sugarland
Research	5	333445555	1988-05-22	Houston

- DEPARTMENT와 DEPT_LOCATION에서 Dnumber가 이미 Attribute name이 같은 형태로 존재하기 때문에 바로 NATURAL JOIN할 수 있다.

JOIN OPERATION의 특징

1. **같은 두 TABLE이 JOIN될 수 있는 방법이 두 개 이상일 수 있다.**
   - 각각의 JOIN은 **다른 의미를 갖는다**

"Associate each DEPARTMENT with its manager"

$$DEPARTMENT \bowtie_{Mgr\_ssn=Ssn} EMPLOYEE$$

"Associate each EMPLOYEE with the department for which the EMPLOYEE works"

$$EMPLOYEE \bowtie_{Dno=Dnumber} DEPARTMENT$$

   - 위는 DEPARTMENT Manager를 찾는 JOIN, 아래는 DEPARTMENT에 속해 일하는 Employee를 찾는 JOIN이다.
   - **JOIN되는 Operation에 따라 의미가 다르다!**

2. **JOIN Operation은 하나의 Relation에 대해 적용될 수 있다.**
    - A Join op A
    - **하나의 Table의 두 Copy본을 이용하여 JOIN**을 하는 것처럼 생각하면 된다.
    - **Renaming**하는 것이 좋다.

"Retrieve the name of each EMPLOYEE and the name of its supervisor"

$$SUPERVISOR(Super\_ssn, Sfname, Slname) \leftarrow \pi_{Ssn,\ Fname,\ Lname}(EMPLOYEE)$$

$$TEMP \leftarrow EMPLOYEE * SUPERVISOR$$

$$RESULT \leftarrow \pi_{Fname,\ Lname,\ Sfname,\ Slname}(TEMP)$$

- EMPLOYEE Table에서 Ssn, Fname, Lname을 뽑는데 **이를 Super_ssn, Sfname, Slname으로 Renaming한다.**
    - Table도 SUPERVISOR라고 한다.
    - 기본 EMPLOYEE table에서 Super_ssn이 존재하기 때문이다.
- EMPLOYEE와 SUPERVISOR를 **NATURAL JOIN**하게 되면 **EMPLOYEE의 상사 번호와 SUPERVISOR의 번호가 일치하는 Tuple**만이 남게 된다.
    - **NATURAL JOIN을 사용하기 때문에 Renaming**을 하는 것이다.
- **ER Model에서 Recursive relationship에 대응되는 Relation을 JOIN할 때 유용**하다.

Complete set

위 집합 내의 Operation만을 이용해서 어떤 Relational algebra expressions도 표현할 수 있도록 하는 Operation의 집합

Relational algebra의 Complete set

- **{SELECT, PROJECT, UNION, SET DIFFERENCE, CARTESIAN PRODUCT}**
  - **INTERSECT**는 **A-((AUB)-B)**로 나타낼 수 있다.
  - **JOIN은 CARTESIAN PRODUCT → SELECT**으로 나타낼 수 있다.

Relationally complete languages

Complete set of relational algebra를 표현할 수 있는 Query language