# Database Systems

# Lecture #07

Sang-Wook Kim
Hanyang University

Data
And
Knowledge
Engineering

# Objectives

- To learn relational algebra
  - Operations and operators of relational algebra
  - Query expressions with relational algebra

# Outline

- Relational Algebra
- SELECT Operation
- PROJECT Operation
- Relational Algebra Expressions
- Set Operations
- CARTESIAN PRODUCT Operation
- JOIN Operation
- A Complete Set of Relational Algebra Operations

# Relational Algebra

- Basic set of *operations* for the relational model
- Used for specifying *retrieval requests*
- A result of a retrieval is represented as a new *relation*

# Relational Algebra

- Types
  - SELECT
  - PROJECT
  - Set operations
  - JOIN
  - …

# SELECT Operation

- Format

$$\sigma_{<\text{selection condition}>}(R)$$

# SELECT Operation

- Meaning
  - Selects a *subset of the tuples* from a relation *R* that satisfies a selection condition *c*
  - Selection condition *c* is a Boolean expression
    - <attribute name> <comparison op> <constant value>
    - <attribute name> <comparison op> <attribute name>
  - A resulting relation contains *only* the tuples which satisfies *c* from *r*(*R*)

# SELECT Operation

- Examples

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

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

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

| 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 Operation

- Format

$$\pi_{<\text{attribute list}>}(R)$$

# PROJECT Operation

- Meaning
  - *Selects columns* specified in attribute list L from table R
    - Discards the other columns
  - A resulting relation contains those tuples with attributes specified in *L* from *r*(*R*)
  - The result of PROJECT operation is a set of *distinct tuples*

project operation 후에 tuple 끼리 중복되지 않는다.
ex) πsex(EMPLOYEE) - tuple 2개 존재 (male,female)

# PROJECT Operation

- Examples

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

| 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 |

# PROJECT Operation

- Examples

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

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

# Relational Algebra Expressions

- Sequence of relational algebra operations
- Complex queries can be represented as a relational algebra expression
  - A relational algebra expression *results in a relation*

# Relational Algebra Expressions

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

# Relational Algebra Expressions

- Examples
  - "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 attribute of a result relation can be *renamed*
    - $\text{TEMP} \leftarrow \sigma_{\text{Dno}=5}(\text{EMPLOYEE})$
      $R(\text{First\_name, Last\_name, Salary}) \leftarrow \pi_{\text{Fname, Lname, Salary}}(\text{TEMP})$

# Set Operations

- Relational algebra operations from set theory
  - 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
    - $R_1 \times R_2$

# Set Operations

- 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 |

# Set Operations

- Union compatibility (type compatibility)
  - Two relations should have the same *type of tuples*
  - For set operations ∪, ∩, and -
    - The two relation $R(A_1, A_2, \ldots, A_n)$, $S(B_1, B_2, \ldots, B_n)$ should have
      - *Same number of attributes n*
    - Two attributes in the corresponding attribute pair *should have the domains compatible (similar) to each other*
      - $\mathrm{dom}(A_i) = \mathrm{dom}(B_i)$ $(1 \le i \le n)$

Data And Knowledge Engineering

# Set Operations

- Examples
  - 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 |

# Set Operations

- Examples
  - STUDENT ∪ INSTRUCTOR

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

# Set Operations

- Examples
  - STUDENT ∩ INSTRUCTOR

| Fn | Ln |
|---|---|
| Susan | Yao |
| Ramesh | Shah |

# Set Operations

- Examples
  - STUDENT - INSTRUCTOR

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

# Set Operations

- Examples
  - INSTRUCTOR - STUDENT

| Fname | Lname |
|---|---|
| John | Smith |
| Ricardo | Browne |
| Francis | Johnson |

# CARTESIAN PRODUCT Operation

- Format

$$R \times S$$

or

$$R(A_1, A_2, \ldots, A_n) \times S(B_1, B_2, \ldots, B_m)$$

# CARTESIAN PRODUCT Operation

- Meaning
  - Produces a new relation by *combining* every tuple from one relation with every tuple from the other relation
  - $Q(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$
    - Number of attributes: $n + m$
    - Number of tuples: $n_R * n_S$
  - Useful only when followed by a selection that matches values of attributes

# CARTESIAN PRODUCT Operation

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

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

# CARTESIAN PRODUCT Operation

◆ Example

$$\text{FEMALE\_EMPS} \leftarrow \sigma_{\text{Sex}=\text{'F'}}(\text{EMPLOYEE})$$

$$\text{EMPNAMES} \leftarrow \pi_{\text{Fname, Lname, Ssn}}(\text{FEMALE\_EMPS})$$

**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 |

# CARTESIAN PRODUCT Operation

- Example

$$\text{EMP\_DEPENDENTS} \leftarrow \text{EMPNAMES} \times \text{DEPENDENT}$$

**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 | . . . |
| Jennifer | Wallace | 987654321 | 123456789 | Elizabeth | F | 1967-05-05 | . . . |
| Joyce | English | 453453453 | 333445555 | Alice | F | 1986-04-05 | . . . |

# CARTESIAN PRODUCT Operation

- Example

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

**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

- Necessity
  - Used to process *relationships* among tuples stored in different relations

# JOIN Operation

- Example
  - "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 | . . . |

# JOIN Operation

- Example
  - Using CARTESIAN PRODUCT
    - ALL_PRODUCT ← DEPARTMENT × EMPLOYEE
    - DEPT_MGR ← $\sigma_{Mgr_ssn=Ssn}$ALL_PRODUCT
    - RESULT ← $\pi_{Dname,\ Lname,\ Fname}$(DEPT_MGR)

# JOIN Operation

- Example
  - Using JOIN

$$DEPT\_MGR \leftarrow DEPARTMENT \bowtie_{Mgr\_ssn=Ssn} EMPLOYEE$$

$$RESULT \leftarrow \pi_{Dname,\ Lname,\ Fname}(DEPT\_MGR)$$

# THETA JOIN Operation

- Format

$$R \bowtie_{\text{<join condition>}} S$$

# THETA JOIN Operation

◆ Meaning

- Produces a new relation by combining every tuple from one relation with every tuple from the other relation
  - Whenever the combination *satisfies the join condition c*
- SELECT operation after CARTESIAN PRODUCT
- Join condition *c*:
  - <condition> **AND** <condition> **AND…AND** <condition>
  - <condition> = $A_i \, \theta \, B_j$
  - $\theta = \{=, <, \leq, >, \geq, \neq\}$

# EQUIJOIN Operation

- Meaning
  - *Only = comparison operator* used in a JOIN operation
  - The join result always has *a pair of attributes that have identical values* in every tuple

# NATURAL JOIN Operation

◆ Format

$$R * S$$

| R1A | FK | R2B | PK | -> equijoin | R1A | FK | R2B | PK | -> natural join에서는 해당 attribute 제외 |
|---|---|---|---|---|---|---|---|---|---|
| A1 | 1 | B1 | 1 | | A1 | 1 | B1 | 1 | |
| A2 | 2 | B2 | 2 | | A2 | 2 | B2 | 2 | |
| A3 | 3 | | | | | | | | |

# NATURAL JOIN Operation

- Meaning
  - EQUIJOIN
    - The join result always has *a pair of attributes that have identical values* in every tuple => *Unnatural!*
  - Second (superfluous) attribute in an EQUIJOIN condition is removed in the final result
  - Before joining, two join attributes must have the same name in both relations

identical 해야 한다.
why? join condition에서 join attribute가 명시되지 않기 때문에 identical 해야 프로그램이 파악 가능

# NATURAL JOIN Operation

- Examples
  - "Combine each PROJECT tuple with the DEPARTMENT tuple that controls the project"

$$DEPT \leftarrow \rho_{(Dname,\ Dnum,\ Mgr\_ssn,\ Mgr\_start\_date)}(DEPARTMENT)$$
$$PROJ\_DEPT \leftarrow PROJECT * DEPT$$

Makes attributes have the same name

**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 |

# NATURAL JOIN Operation

- Examples
  - "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 |

# JOIN Operation

- Two relations can be joined with more than one JOIN attribute
  - With different meanings

# JOIN Operation

◆ Examples

- "Associate each DEPARTMENT with its manager"

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

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

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

EMPLOYEE*DEPARTMENT 시키면 두 개의 join attribute 모두 같은 경우만 결과로 나온다.

Data
And
Knowledge
Engineering

# JOIN Operation

- A single relation can be used in both sides of JOIN operation
  - Treated as a JOIN between two identical copies of the same relation
  - Renaming is useful

# JOIN Operation

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

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

$$\text{TEMP} \leftarrow \text{EMPLOYEE} * \text{SUPERVISOR}$$

$= \text{EMPLOYEE} \bowtie \text{EMPLOYEE (super.ssn = ssn)}$

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

# A Complete Set of Relational Algebra Operations

## ◆ *Complete set* 최소 연산자의 집합

- Any relational algebra operation can be expressed
- As a sequence of operations from this set

DIVISION
R1(Z) div R2(Y)
- Produces a relation R(X) that includes all tuples t[X] in R1(Z) that appear in R1 in combination with every tuple from R2(Y), where Z = X ∪ Y.

ex) T <- R div S

R

| A | B |
|---|---|
| a1 | b1 |
| a2 | b1 |
| a3 | b1 |
| a4 | b1 |
| a1 | b2 |
| a3 | b2 |
| a2 | b3 |
| a3 | b3 |
| a4 | b3 |
| a1 | b4 |
| a2 | b4 |
| a3 | b4 |

S

| A |
|---|
| a1 |
| a2 |
| a3 |

->

T

| B |
|---|
| b1 |
| b4 |

# A Complete Set of Relational Algebra Operations

- Complete set of relational algebra
  - {SELECT, PROJECT, UNION, SET DIFFERENCE, CARTESIAN PRODUCT}
  - JOIN?
    - can be represented as a SELECT after CARTESIAN PRODUCT
  - INTERSECTION?
    - Meaning? $R \cap S = (R \cup S) - ((R-S) \cup (S-R))$
    - How to express by the complete set?
  - DIVISION?
    - Meaning?
    - How to express by the complete set?

T <- R % S
T1 <- πY(R)
T2 <- πY((S X T1)-R)
T <- T1 - T2

# A Complete Set of Relational Algebra Operations

- *Relationally complete* languages
  - Query languages capable of express *complete set of relational algebra operations*

# Summary

| OPERATION | PURPOSE | NOTATION |
|---|---|---|
| SELECT | Selects all tuples that satisfy the selection condition from a relation $R$. | $\sigma_{\text{<selection condition>}}(R)$ |
| PROJECT | Produces a new relation with only some of the attributes of $R$, and removes duplicate tuples. | $\pi_{\text{<attribute list>}}(R)$ |
| THETA JOIN | Produces all combinations of tuples from $R_1$ and $R_2$ that satisfy the join condition. | $R_1 \bowtie_{\text{<join condition>}} R_2$ |
| EQUIJOIN | Produces all the combinations of tuples from $R_1$ and $R_2$ that satisfy a join condition with only equality comparisons. | $R_1 \bowtie_{\text{<join condition>}} R_2$, OR $R_1 \bowtie_{(\text{<join attributes 1>}),\ (\text{<join attributes 2>})} R_2$ |
| NATURAL JOIN | Same as EQUIJOIN except that the join attributes of $R_2$ are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all. | $R_1 *_{\text{<join condition>}} R_2$, OR $R_1 *_{(\text{<join attributes 1>}),\ (\text{<join attributes 2>})} R_2$ OR $R_1 * R_2$ |

# Summary

| OPERATION | PURPOSE | NOTATION |
| --- | --- | --- |
| UNION | Produces a relation that includes all the tuples in $R_1$ or $R_2$ or both $R_1$ and $R_2$; $R_1$ and $R_2$ must be union compatible. | $R_1 \cup R_2$ |
| INTERSECTION | Produces a relation that includes all the tuples in both $R_1$ and $R_2$; $R_1$ and $R_2$ must be union compatible. | $R_1 \cap R_2$ |
| DIFFERENCE | Produces a relation that includes all the tuples in $R_1$ that are not in $R_2$; $R_1$ and $R_2$ must be union compatible. | $R_1 - R_2$ |
| CARTESIAN PRODUCT | Produces a relation that has the attributes of $R_1$ and $R_2$ and includes as tuples all possible combinations of tuples from $R_1$ and $R_2$. | $R_1 \times R_2$ |
| DIVISION | Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in $R_1$ in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$. | $R_1(Z) \div R_2(Y)$ |

# Summary

- Relational Algebra Operations
  - SELECT
  - PROJECT
  - Set Operations
    - UNION, INTERSECTION, SET DIFFERENCE, CARTESIAN PRODUCT
  - JOIN
    - THETA JOIN
    - EQUIJOIN
    - NATURAL JOIN

# References


1. Codd, Edgar F. "A relational model of data for large shared data banks."*Communications of the ACM* 13.6 (1970): 377-387.
2. Date, C. J. "The Outer Join." *Proceedings of the Second International Conference on Databases* (*ICOD*). 1983.
3. Carlis, John V. "HAS, a relational algebra operator or divide is not enough to conquer." *Proceedings of the Second International Conference on Data Engineering*. IEEE Computer Society, 1986.
4. Özsoyoğlu, G., Z. M. Özsoyoğlu, and Victor Matos. "Extending relational algebra and relational calculus with set-valued attributes and aggregate functions." *ACM Transactions on Database Systems (TODS)* 12.4 (1987): 566-592.
5. Cammarata, Stephanie, Prasadram Ramachandra, and Darrell Shane. *Extending a relational database with deferred referential integrity checking and intelligent joins*. Vol. 18. No. 2. ACM, 1989.

# References


6. Codd, Edgar F. "A data base sublanguage founded on the relational calculus."*Proceedings of the 1971 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control*. ACM, 1971.
7. Codd, Edgar F. *Relational completeness of data base sublanguages*. IBM Corporation, 1972.
8. Stonebraker, Michael. "Implementation of integrity constraints and views by query modification." *Proceedings of the 1975 ACM SIGMOD international conference on Management of data*. ACM, 1975.
9. Ullman J. *Principles of Database and Knowledge-Base Systems*, Vol. 1, Computer Science Press, 1988.
10. Abiteboul, Serge, Richard Hull, and Victor Vianu. *Foundations of databases*. Vol. 8. Reading: Addison-Wesley, 1995.
11. Atzeni, Paolo, and Valeria De Antonellis. *Relational database theory*. Benjamin-Cummings Publishing Co., Inc., 1993.

Have a nice day!