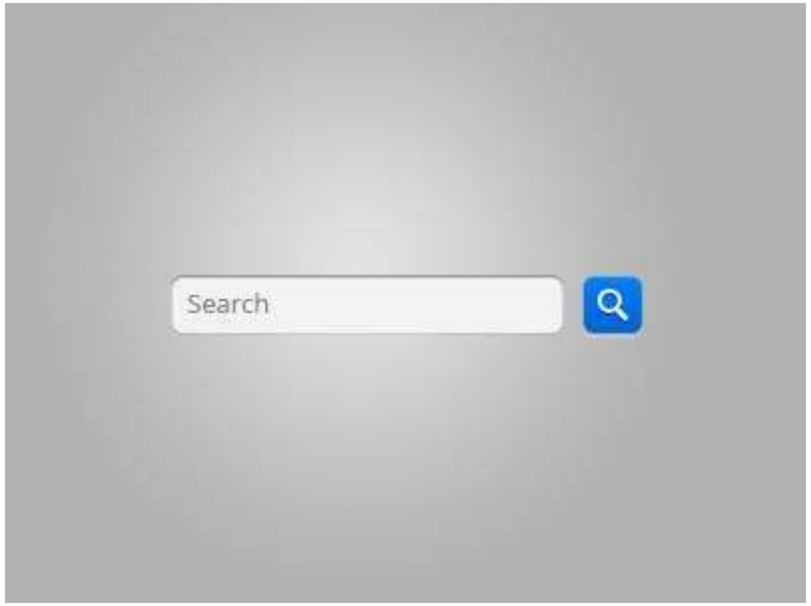
RELATIONAL ALGEBRA

QUERY LANGUAGE

- A query language is a language in which a user requests information from the database
- A Language which is used to store and retrieve data from database
- E.g.
  - SQL
- Two types
  - Procedural Query language
  - Non-procedural query language



QUERY LANGUAGE

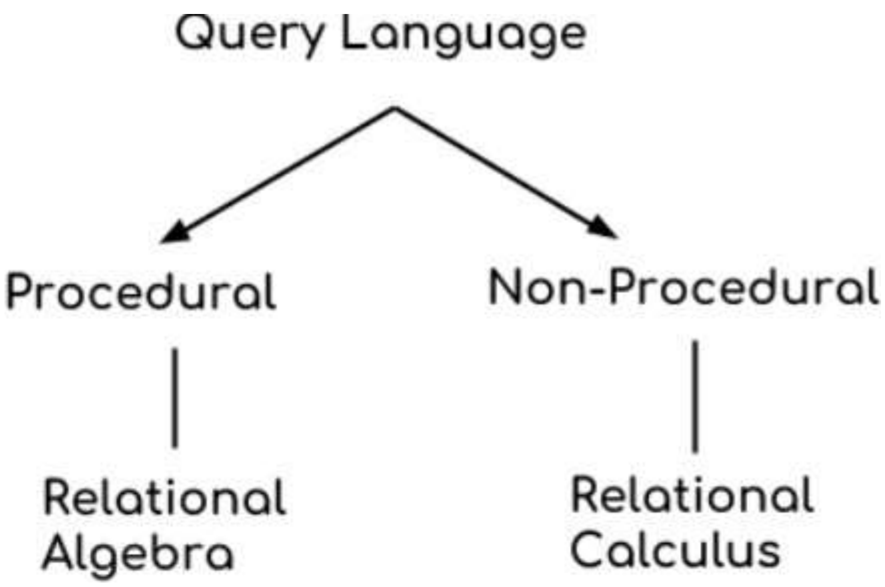
Procedural Query language

- user instructs the system to perform a sequence of operations on the database to compute the desired result
- users tells what data to be retrieved from database and how to retrieve it.

Non-procedural query language

- user describes the desired information without giving a specific procedure for obtaining that information
- users tells what data to be retrieved from database but doesn’t tell how to retrieve it.

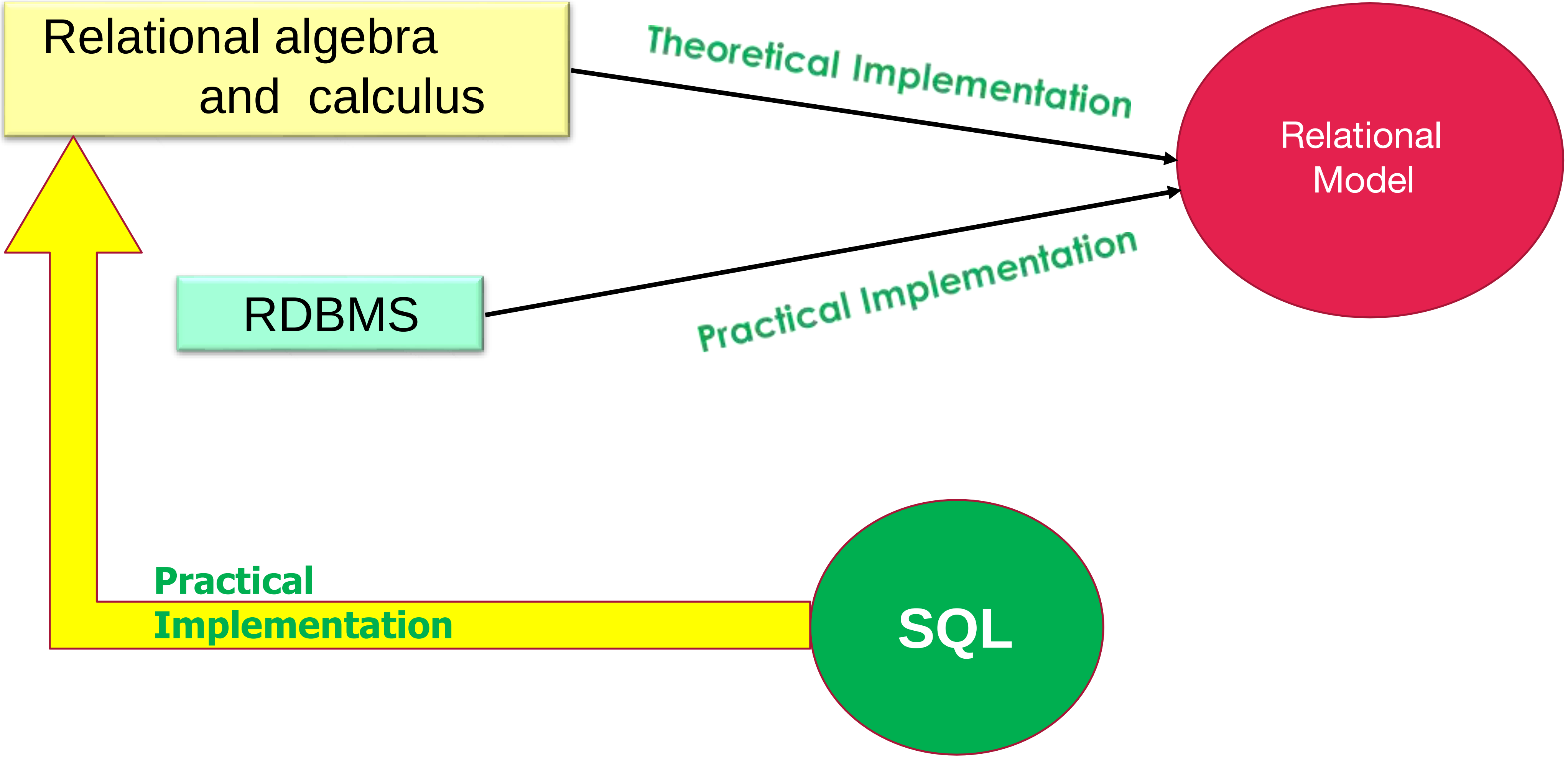
QUERY LANGUAGE



- Relational Algebra
  - conceptual procedural query language used on relational model
- Relational Calculus
  - conceptual non-procedural query language used on relational model

Relational algebra and calculus are the theoretical concepts used on relational model

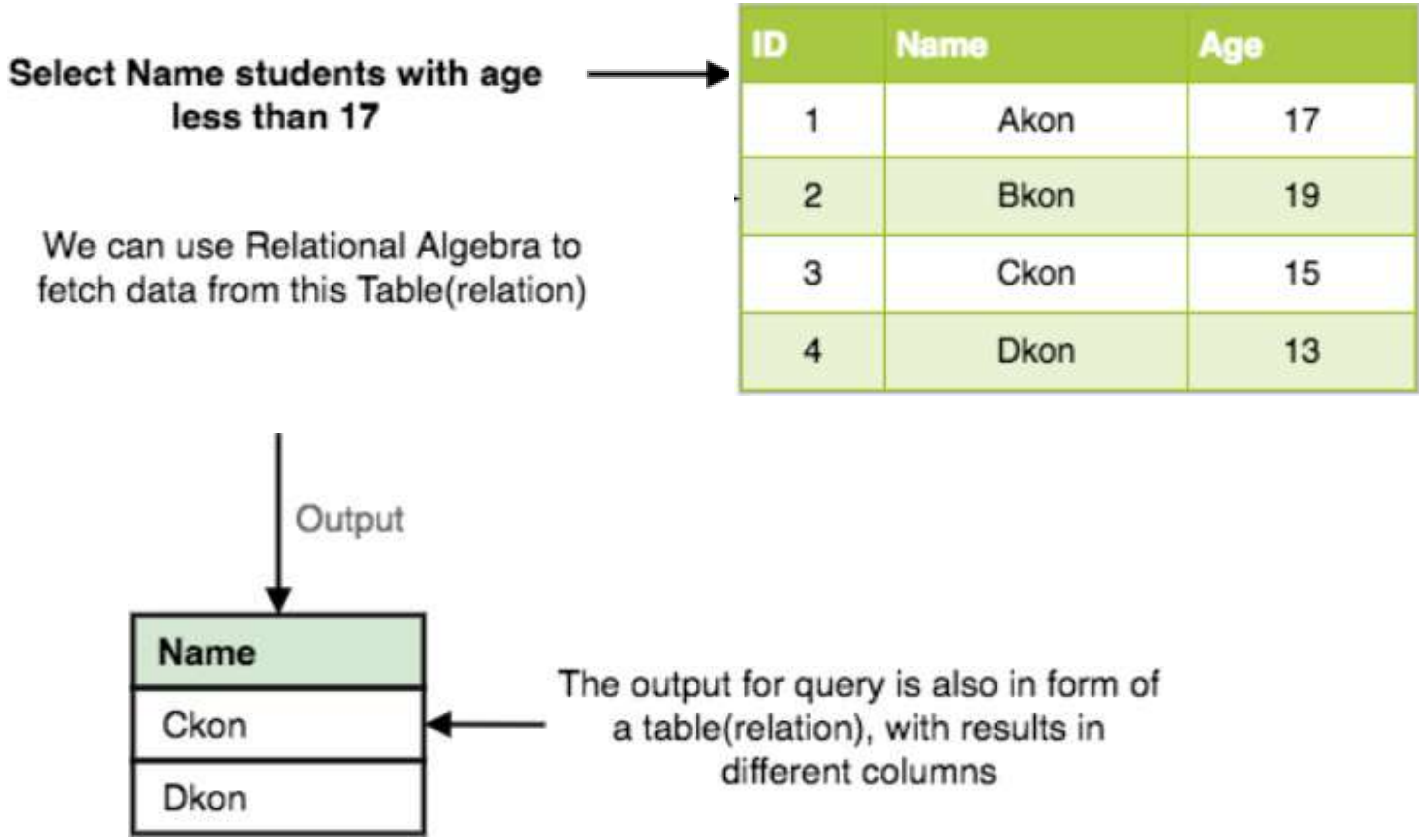
SQL is a practical implementation of relational algebra and relational calculus


Relational algebra and calculus
Theoretical Implementation
Relational Model
RDBMS
Practical Implementation
Practical Implementation
SQL

RELATIONAL ALGEBRA

- procedural query language that works on relational model
- it tells what data to be retrieved and how to be retrieved
- takes Relation as input and generate relation as output
- It uses operators to perform queries
- Relational Algebra works on the whole table at once, so we do not have to use loops etc to iterate over all the rows(tuples) of data one by one
- specify the table name from which we need the data, and in a single line of command, relational algebra will traverse the entire given table to fetch data for you

RELATIONAL ALGEBRA



Select Name students with age less than 17

ID	Name	Age
1	Akon	17
2	Bkon	19
3	Ckon	15
4	Dkon	13

We can use Relational Algebra to fetch data from this Table(relation)

Name
Ckon
Dkon

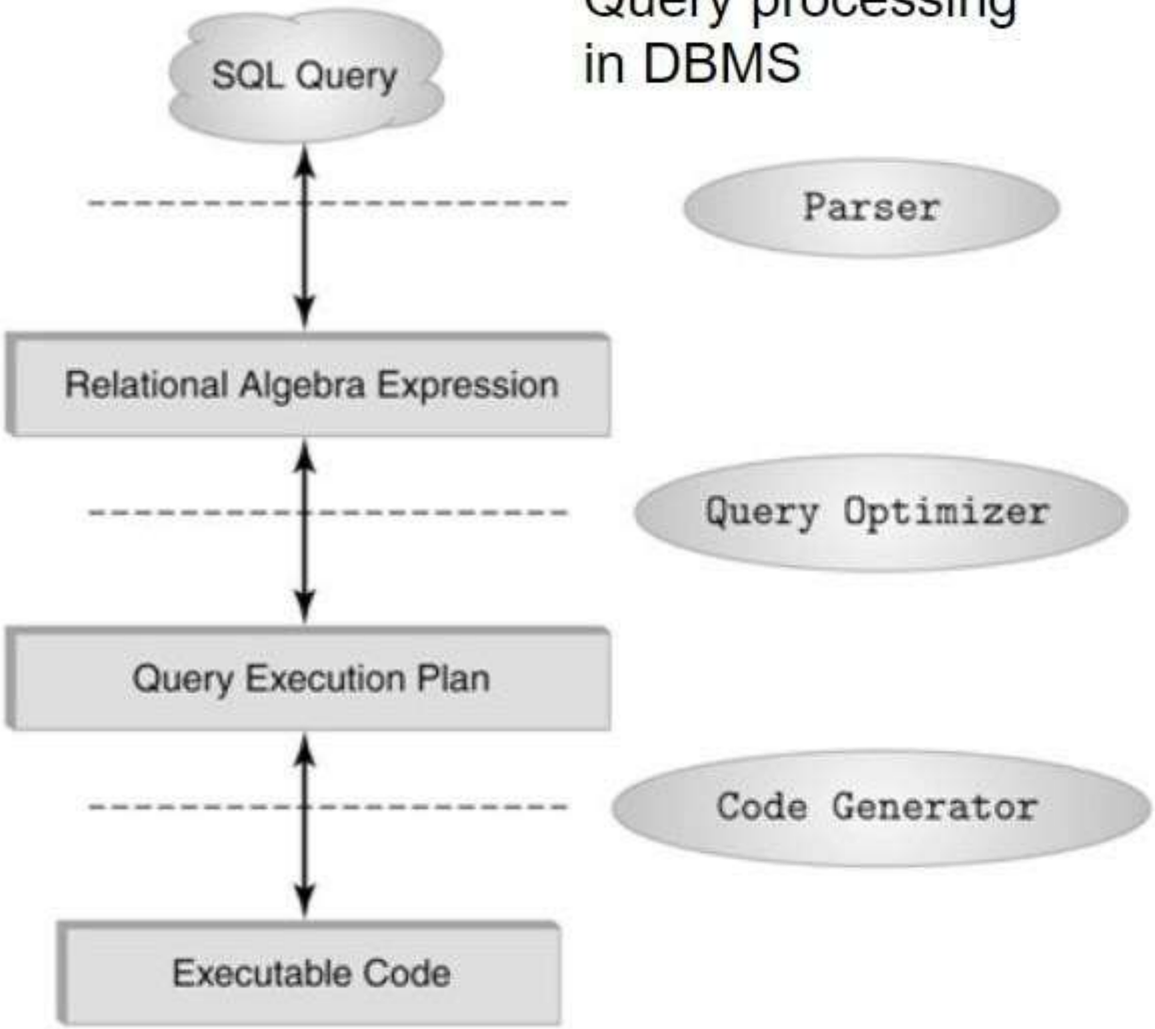
The output for query is also in form of a table(relation), with results in different columns

Query processing in DBMS

```
SELECT A.Name, B.Grade
FROM A, B
WHERE A.Id = B.Id
```

↓

$\pi_{\text{Name, Grade}}(\sigma_{\text{Id,Name}}(A \times B))$



SQL Query

Parser

Relational Algebra Expression

Query Optimizer

Query Execution Plan

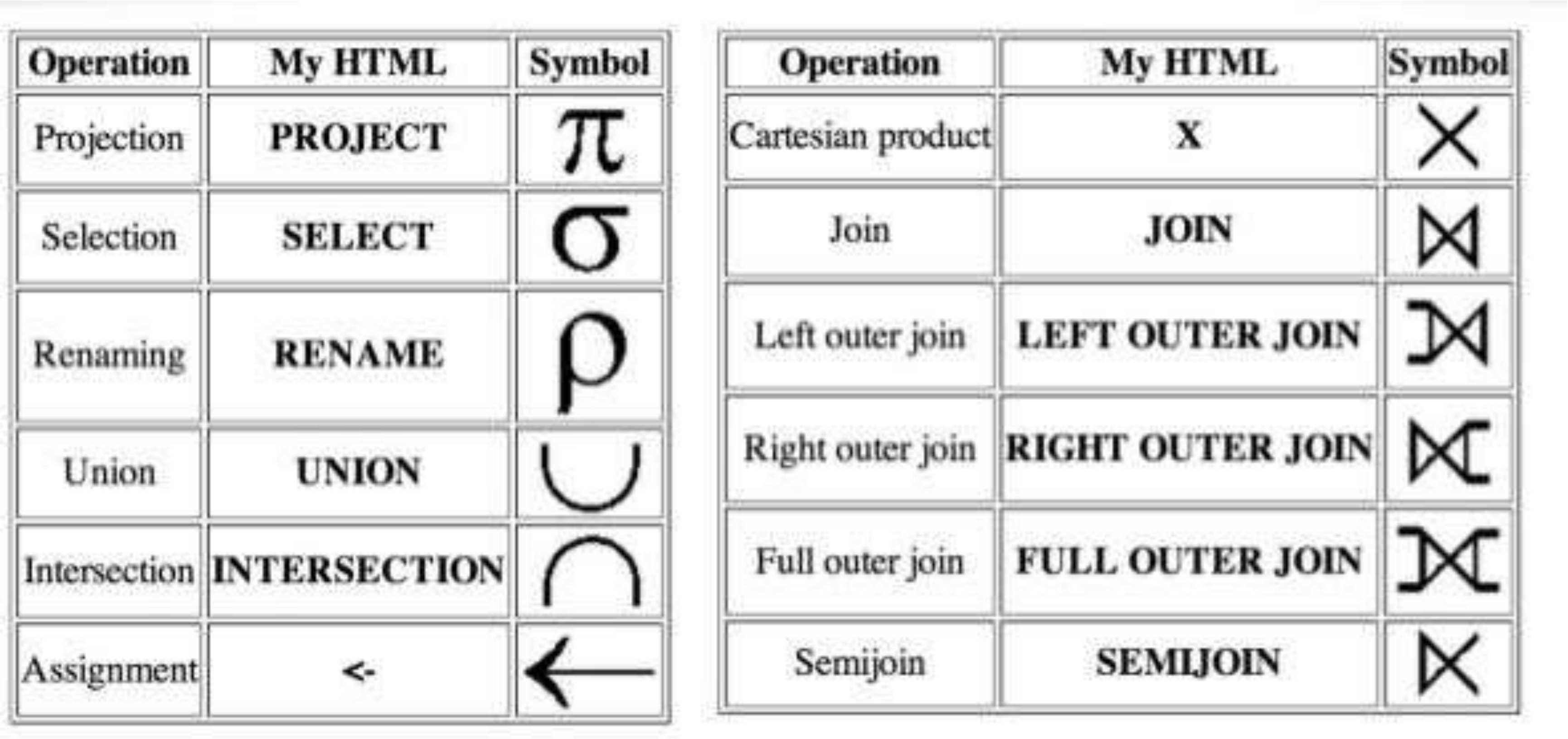
Code Generator

Executable Code

RELATIONAL ALGEBRA-OPERATIONS

- Select
- Project
- Rename
- Union
- Intersect
- Set Difference
- Cartesian Product
- Join

RELATIONAL ALGEBRA-OPERATIONS

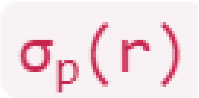


Operation	My HTML	Symbol
Projection	PROJECT	π
Selection	SELECT	σ
Renaming	RENAME	ρ
Union	UNION	∪
Intersection	INTERSECTION	∩
Assignment	<-	←

Operation	My HTML	Symbol
Cartesian product	X	×
Join	JOIN	⋈
Left outer join	LEFT OUTER JOIN	⟕
Right outer join	RIGHT OUTER JOIN	⟖
Full outer join	FULL OUTER JOIN	⟗
Semijoin	SEMIJOIN	⋉

SELECT

- used to select the required tuples of data from a relation.
- denoted by sigma (σ)
- During selection, we can specify certain conditions that the data must satisfy
- Syntax :



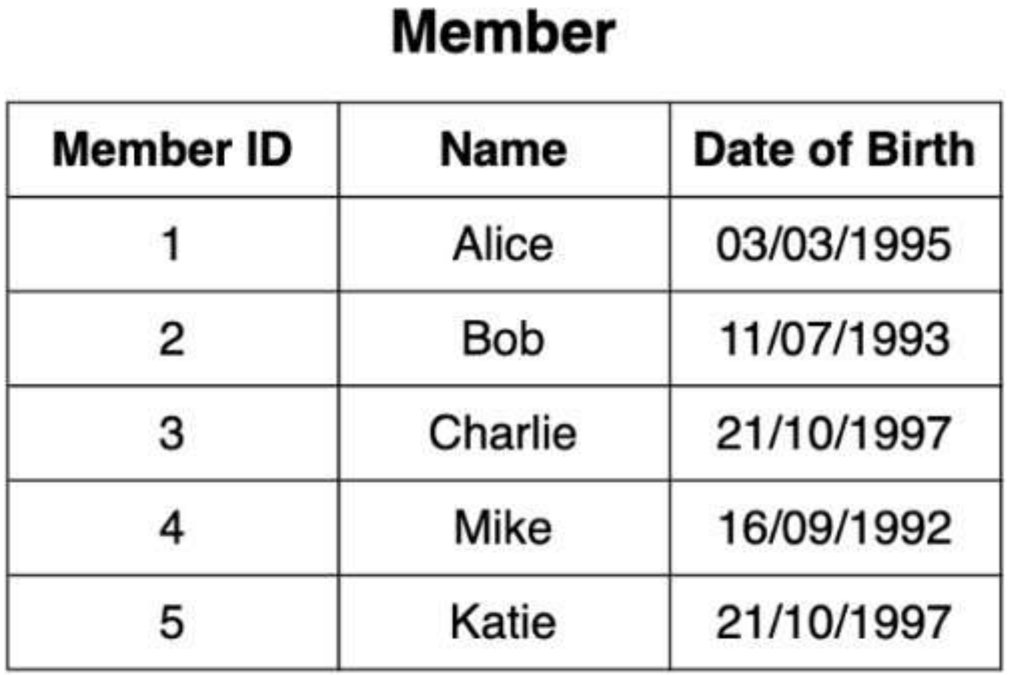
$\sigma_p(r)$

- σ – Selection Predicate
- p – propositional logic (where we specify the conditions - may use connectors like:

  AND OR and NOT. These relational can use as relational operators like =,

  ≠, ≥, <,>, ≤)
- r - Relation

SELECT - EXAMPLE



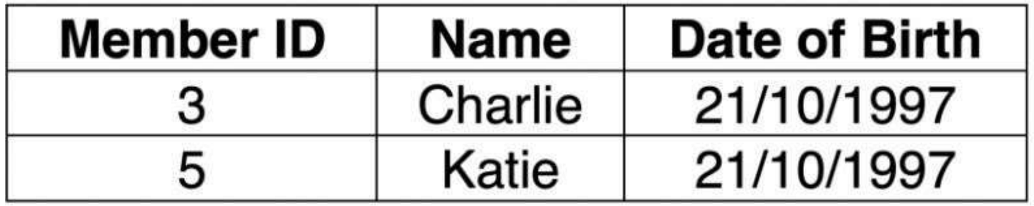
Member

Member ID	Name	Date of Birth
1	Alice	03/03/1995
2	Bob	11/07/1993
3	Charlie	21/10/1997
4	Mike	16/09/1992
5	Katie	21/10/1997

Query:

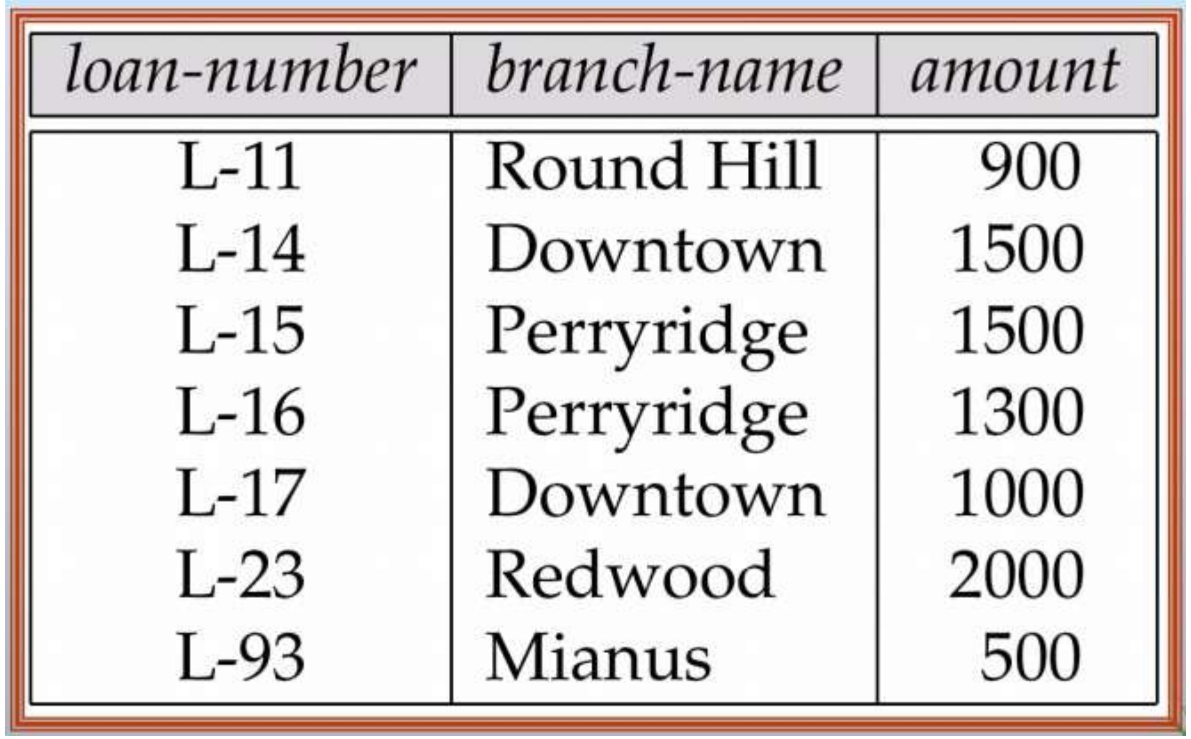
Details of the members who were born on 21/10/1997.

$$\sigma_{Date\ of\ Birth=21/10/1997}(Member)$$



Member ID	Name	Date of Birth
3	Charlie	21/10/1997
5	Katie	21/10/1997

SELECT - QUIZ

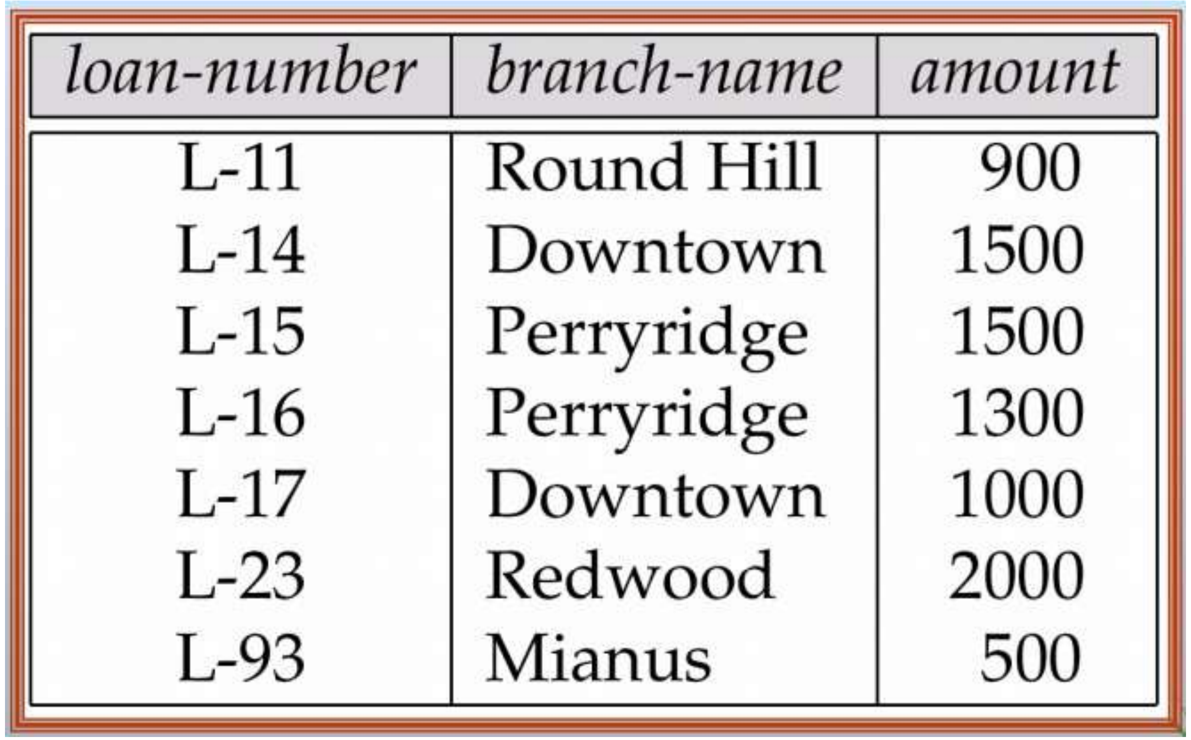


loan-number	*branch-name*	*amount*
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

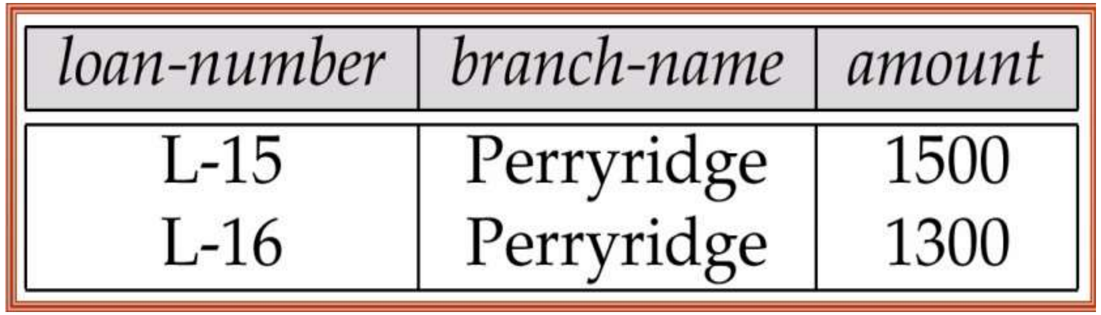
Query:

Select the tuples of the loan relation whose branch belongs to perryridge?

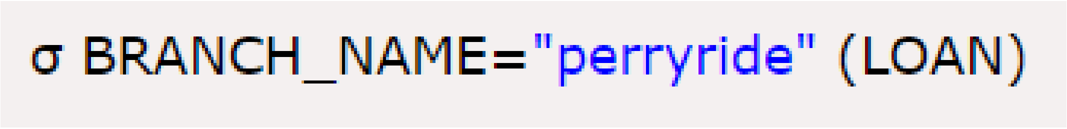
SELECT – QUIZ ANS



loan-number	*branch-name*	*amount*
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500



loan-number	*branch-name*	*amount*
L-15	Perryridge	1500
L-16	Perryridge	1300



σ BRANCH_NAME="perryride" (LOAN)

SELECT - QUIZ

Query:

Students with **age** more than 17

Query:

Selects tuples from **Tutorials** where **topic = 'Database'**

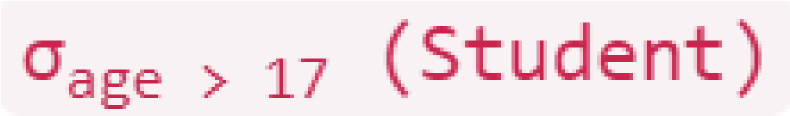
Query:

Select **male Students**, for which **age** will be **greater than** 17

SELECT – QUIZ ANS

Query:

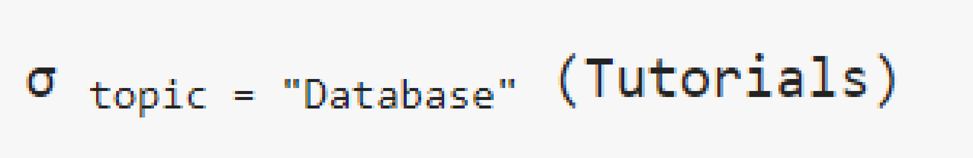
Students with **age** more than 17



$$\sigma_{\text{age} > 17}\ (\text{Student})$$

Query:

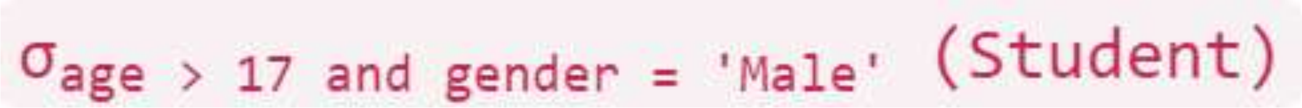
Selects tuples from **Tutorials** where **topic = 'Database**'



$$\sigma_{\text{topic = "Database"}}\ (\text{Tutorials})$$

Query:

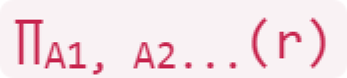
Select **male Students**, for which **age** will be **greater than** 17



$$\sigma_{\text{age > 17 and gender = 'Male'}}\ (\text{Student})$$

PROJECT

- used to select the required columns of data from a relation
- projection removes duplicate data
- Denoted by ∏
- Syntax:



$\Pi_{A1,\ A2...}(r)$

- A1, A2 etc are attribute names

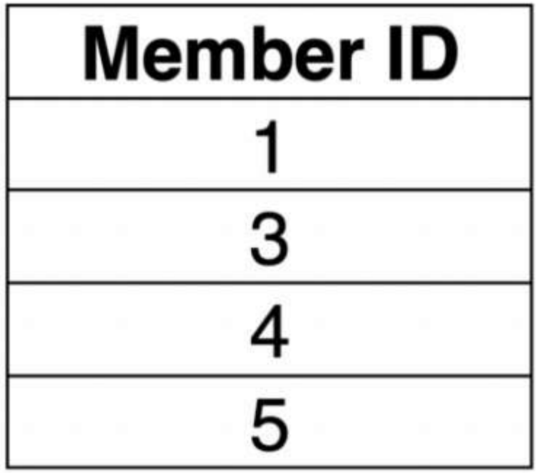
PROJECT - EXAMPLE

Query:

Member IDs of members who have borrowed books.

$$\pi_{Member\ ID}(Borrow)$$

Member ID	Book ID	Book Name
1	1	OOPS
3	5	DBMS
4	3	DS
5	2	Java



Member ID
1
3
4
5

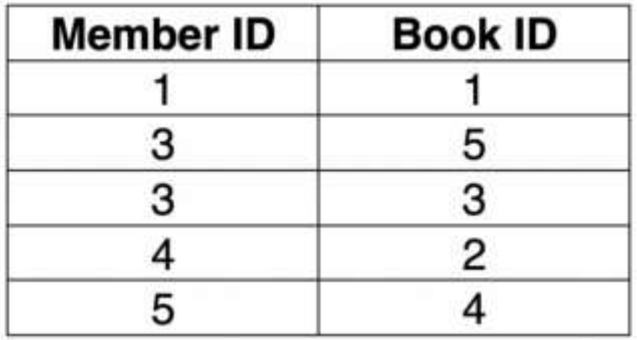
PROJECT - EXAMPLE

Query:

Member IDs of members and the Book IDs of the books they have borrowed books.

Member ID	Book ID	Book Name
1	1	OOPS
3	5	AI
3	3	DBMS
4	2	DS
5	4	Java

$$\pi_{Member\ ID, Book\ ID}(Borrow)$$

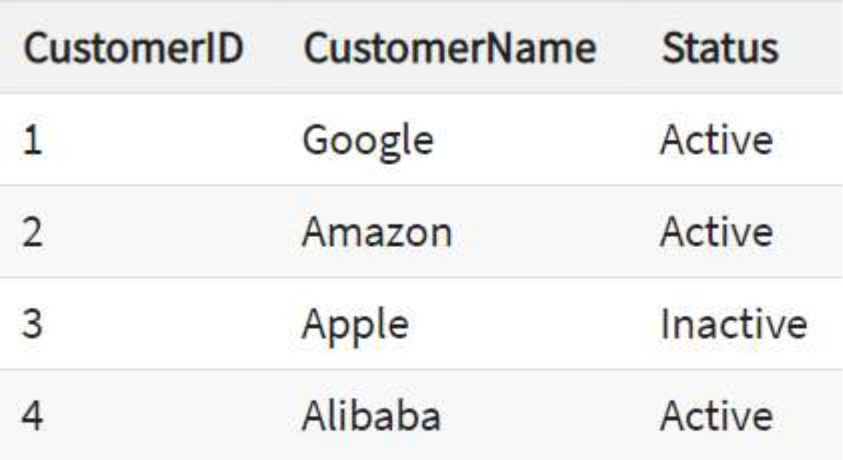


Member ID	Book ID
1	1
3	5
3	3
4	2
5	4

PROJECT - QUIZ

Query:

Select the columns customer Name and status from the table Customers

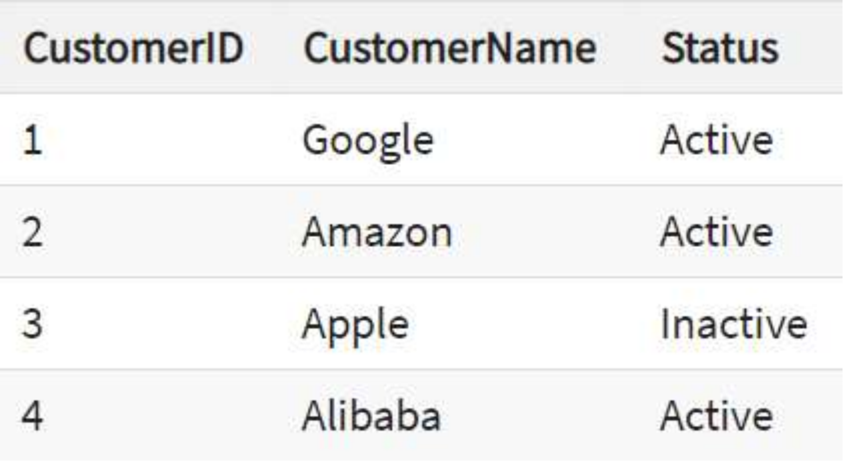


CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

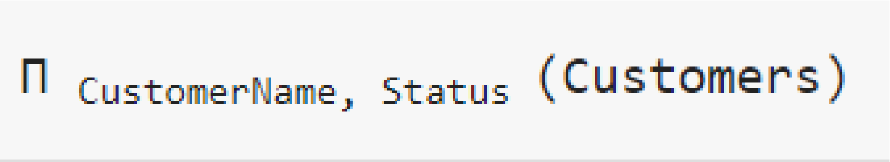
PROJECT – QUIZ ANS

Query:

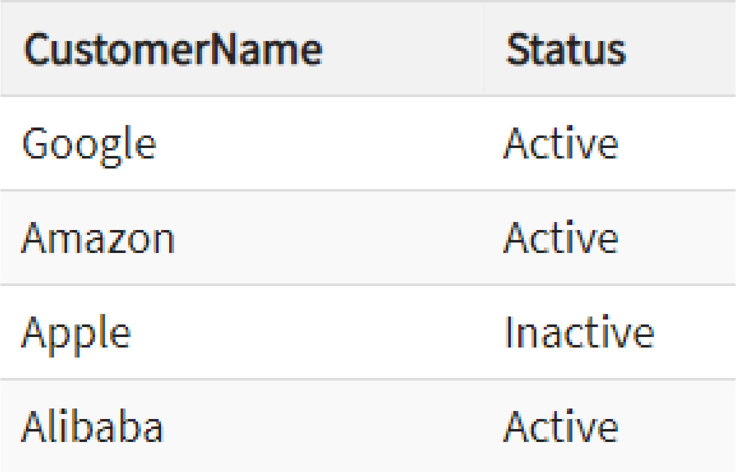
Select the columns customer Name and status from the table Customers



CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active



```
Π CustomerName, Status (Customers)
```

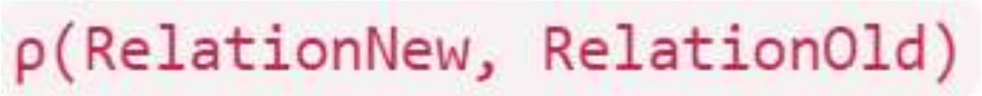


CustomerName	Status
Google	Active
Amazon	Active
Apple	Inactive
Alibaba	Active

RENAME

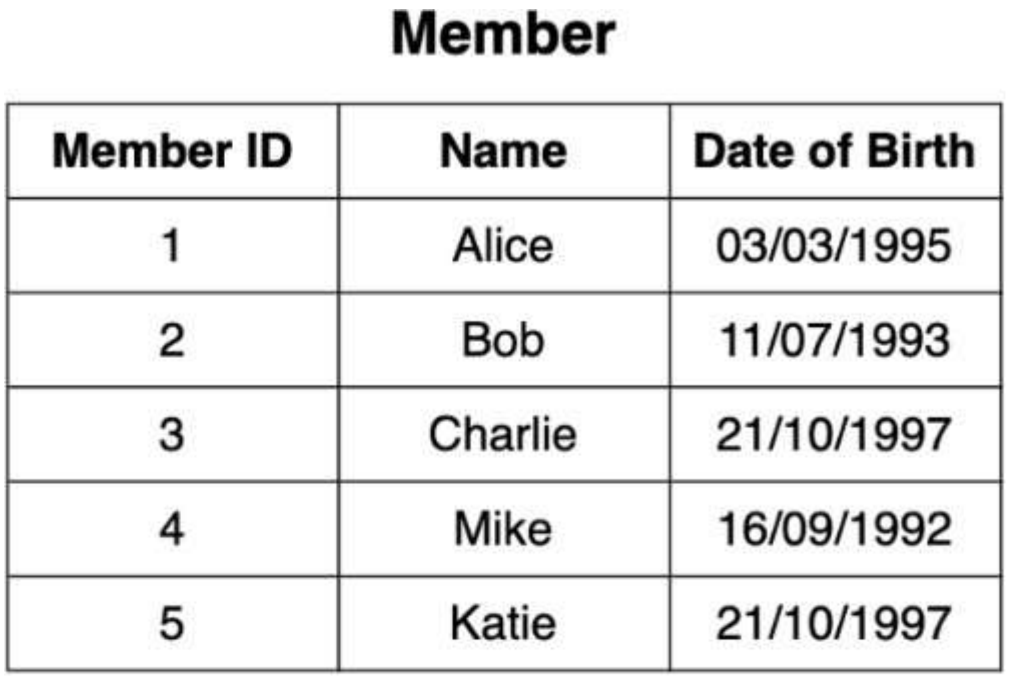
- Rename operation allows renaming a certain output relation
- It is denoted using small **Greek letter rho (ρ)**.
- Syntax:

ρ x (E)



```
ρ(RelationNew, RelationOld)
```

RENAME - EXAMPLE



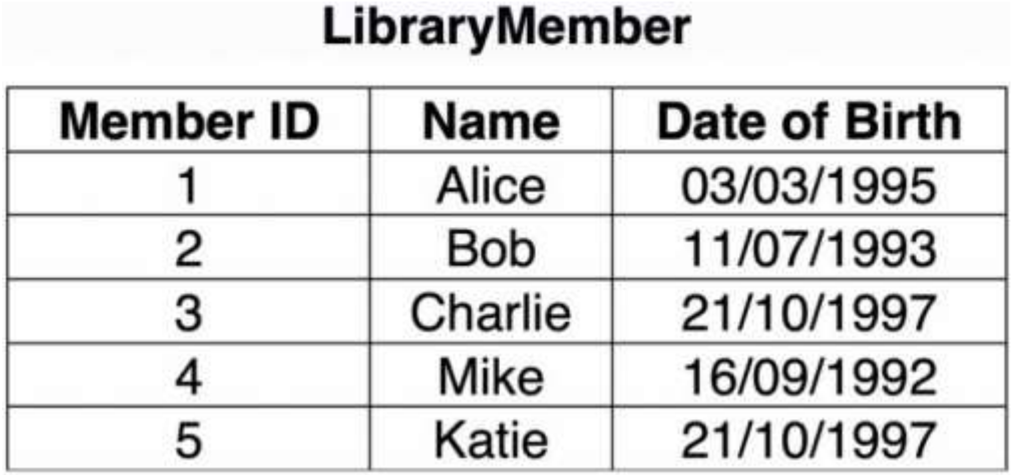
Member

Member ID	Name	Date of Birth
1	Alice	03/03/1995
2	Bob	11/07/1993
3	Charlie	21/10/1997
4	Mike	16/09/1992
5	Katie	21/10/1997

Query:

Rename the Member relation as Library Member.

$$\rho_{LibraryMember}(Member)$$



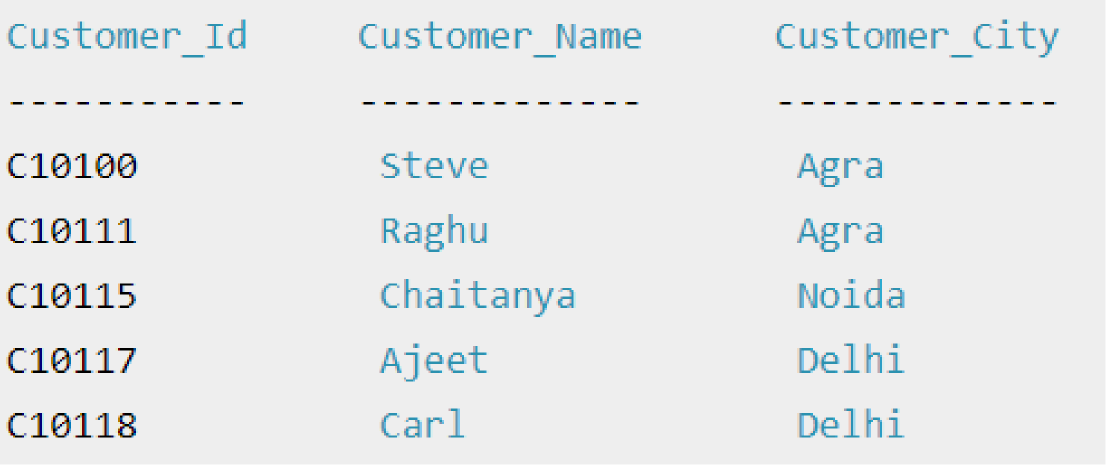
LibraryMember

Member ID	Name	Date of Birth
1	Alice	03/03/1995
2	Bob	11/07/1993
3	Charlie	21/10/1997
4	Mike	16/09/1992
5	Katie	21/10/1997

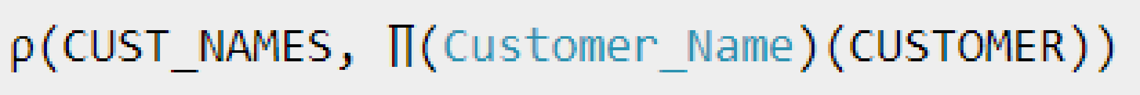
RENAME - EXAMPLE

You can select particular column and rename it and use as a relation

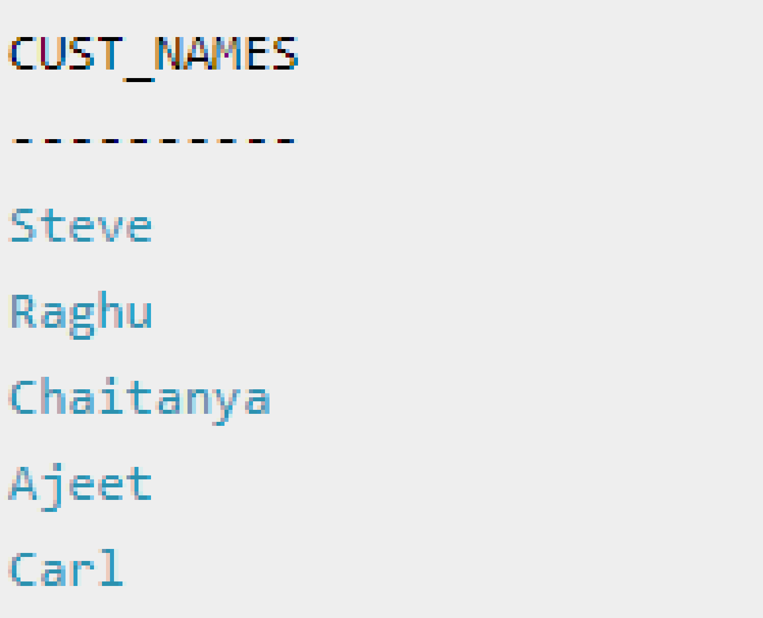
Table: CUSTOMER



```
Customer_Id     Customer_Name      Customer_City
-----------     -------------      -------------
C10100           Steve              Agra
C10111           Raghu              Agra
C10115           Chaitanya          Noida
C10117           Ajeet              Delhi
C10118           Carl               Delhi
```



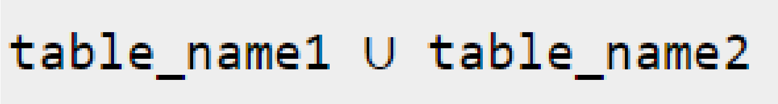
```
ρ(CUST_NAMES, ∏(Customer_Name)(CUSTOMER))
```



```
CUST_NAMES
----------
Steve
Raghu
Chaitanya
Ajeet
Carl
```

UNION

- used to fetch data from two relations(tables) or temporary relation(result of another operation)
- both the relations must have the same set of attributes
- Duplicate tuples should be automatically removed
- denoted by U symbol
- Syntax:



```
table_name1 ∪ table_name2
```

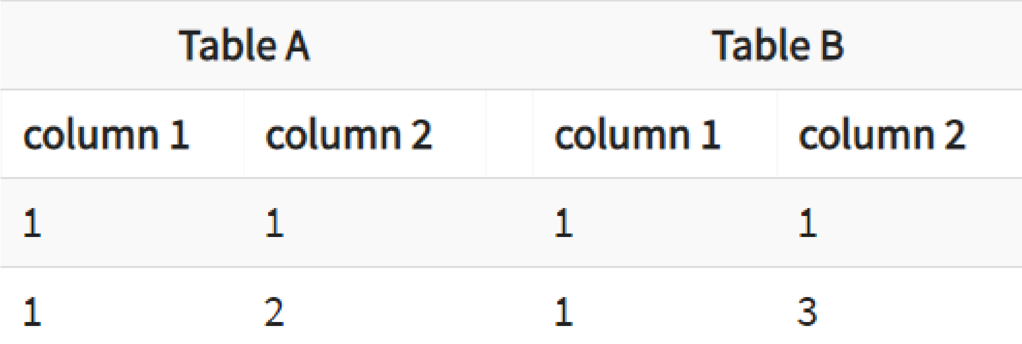
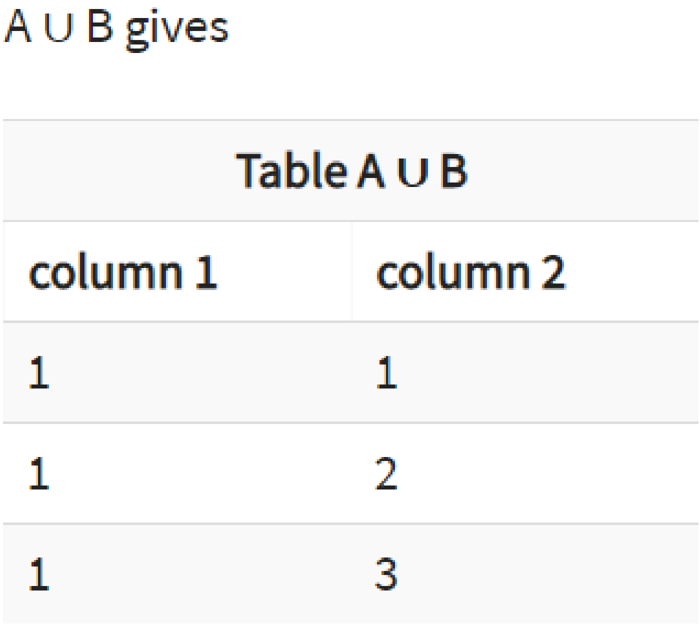


Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3

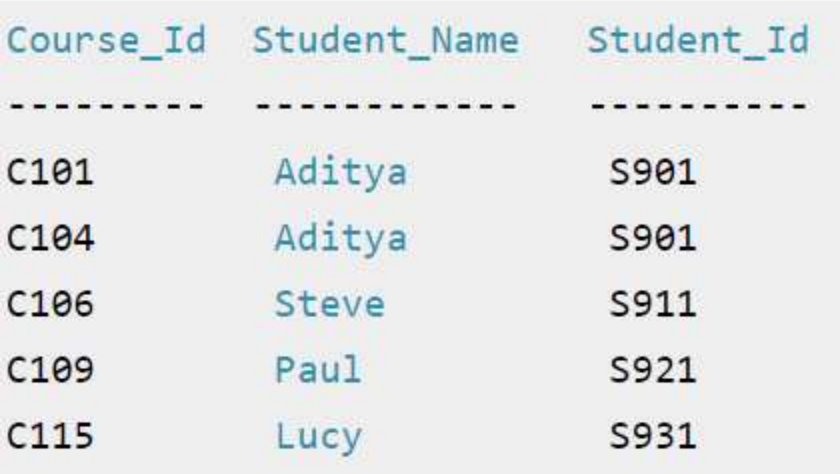


A ∪ B gives

Table A ∪ B	
column 1	column 2
1	1
1	2
1	3

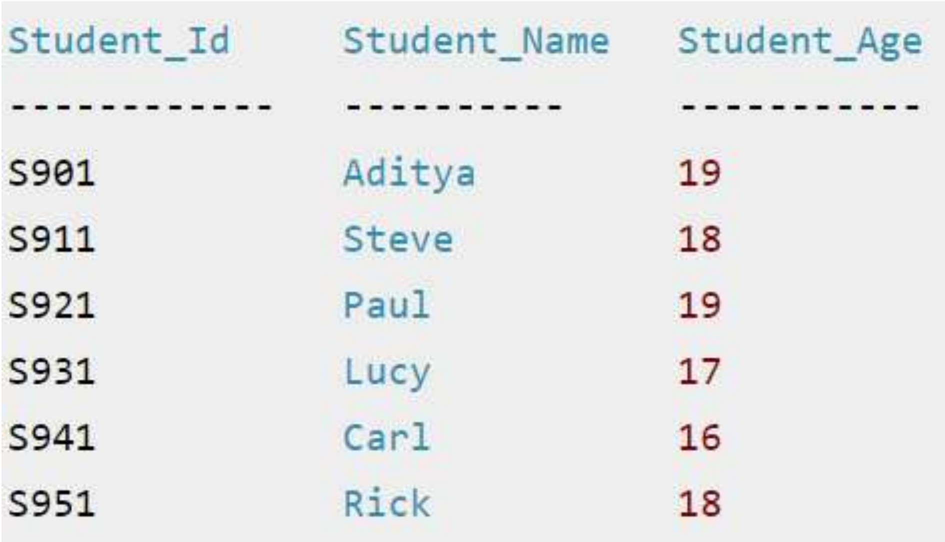
UNION - EXAMPLE

Table 1: COURSE

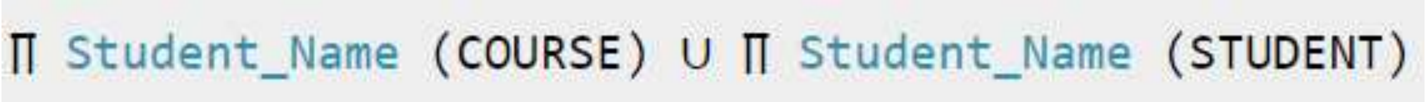


```
Course_Id  Student_Name   Student_Id
---------  ------------   ----------
C101        Aditya         S901
C104        Aditya         S901
C106        Steve          S911
C109        Paul           S921
C115        Lucy           S931
```

Table 2: STUDENT



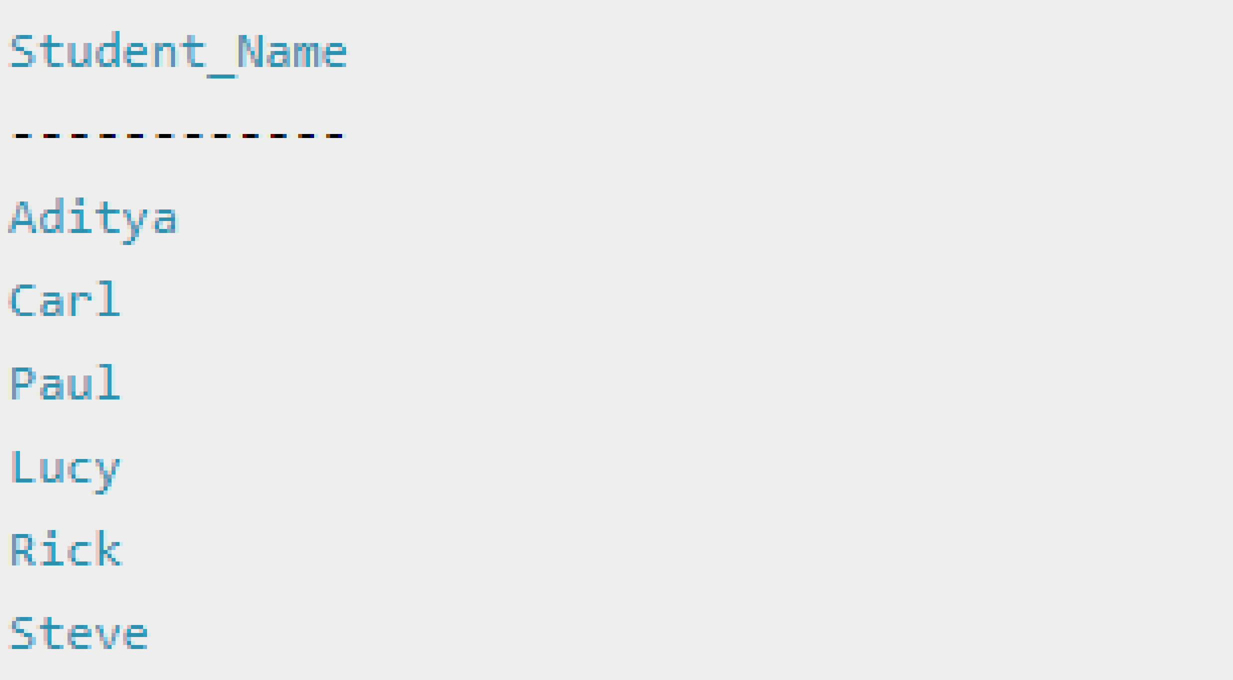
```
Student_Id     Student_Name   Student_Age
------------   ----------     -----------
S901           Aditya         19
S911           Steve          18
S921           Paul           19
S931           Lucy           17
S941           Carl           16
S951           Rick           18
```



```
∏ Student_Name (COURSE) ∪ ∏ Student_Name (STUDENT)
```

UNION - OUTPUT

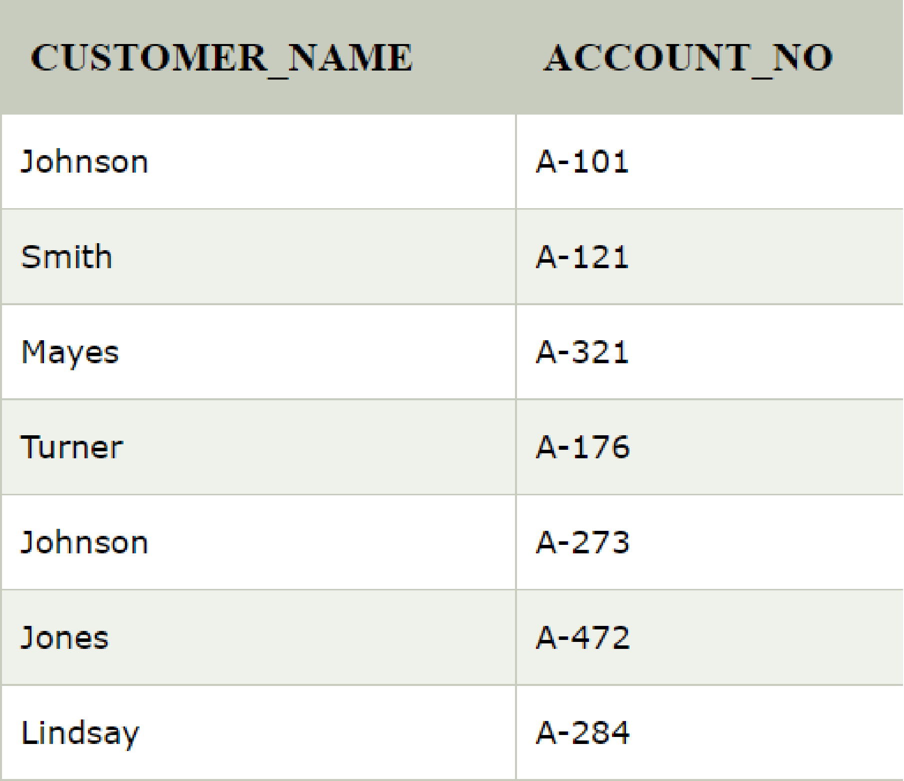
OUTPUT



```
Student_Name
-------------
Aditya
Carl
Paul
Lucy
Rick
Steve
```

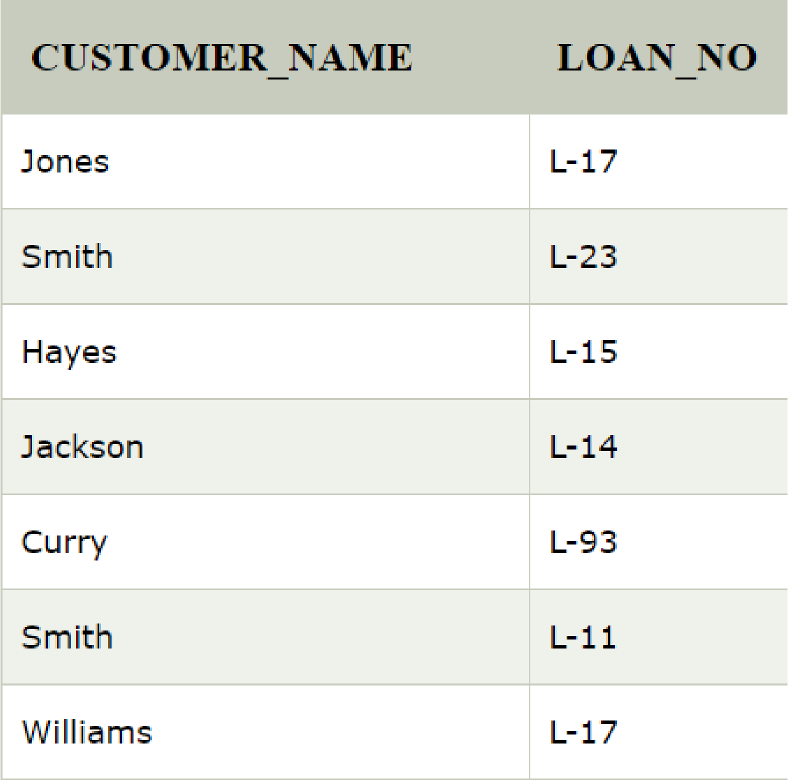
UNION - EXAMPLE

DEPOSITOR RELATION



CUSTOMER_NAME	ACCOUNT_NO
Johnson	A-101
Smith	A-121
Mayes	A-321
Turner	A-176
Johnson	A-273
Jones	A-472
Lindsay	A-284

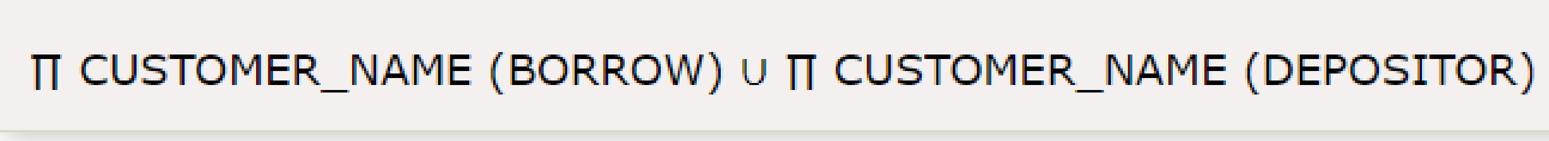
BORROW RELATION



CUSTOMER_NAME	LOAN_NO
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17

∏ CUSTOMER_NAME (BORROW) ∪ ∏ CUSTOMER_NAME (DEPOSITOR)

UNION - EXAMPLE



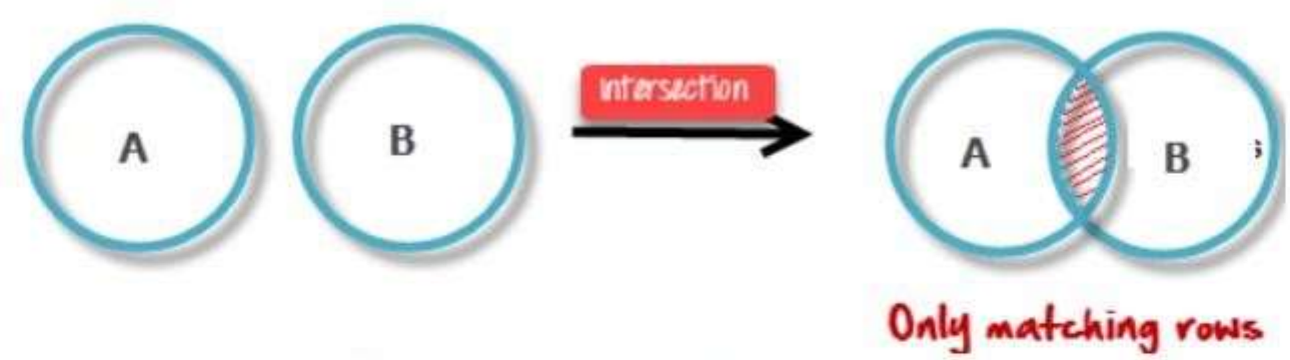
Π CUSTOMER_NAME (BORROW) ∪ Π CUSTOMER_NAME (DEPOSITOR)

- gives the customer name from both relation Depositor and Borrower by eliminating duplication.

INTERSECTION

- defined by the symbol ∩
- Suppose there are two tuples A and B. The set intersection operation contains all tuples that are in both A & B
- Syntax:

A ∩ B



Visual Definition of Intersection

INTERSECTION

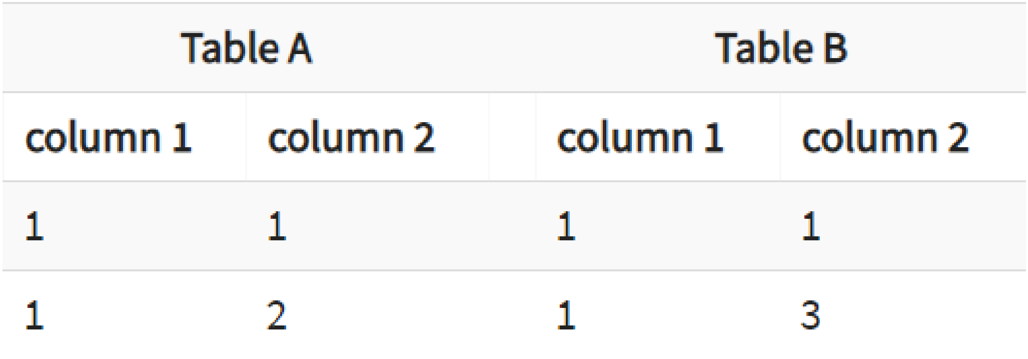


Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3

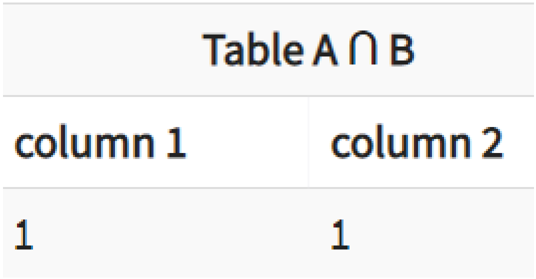
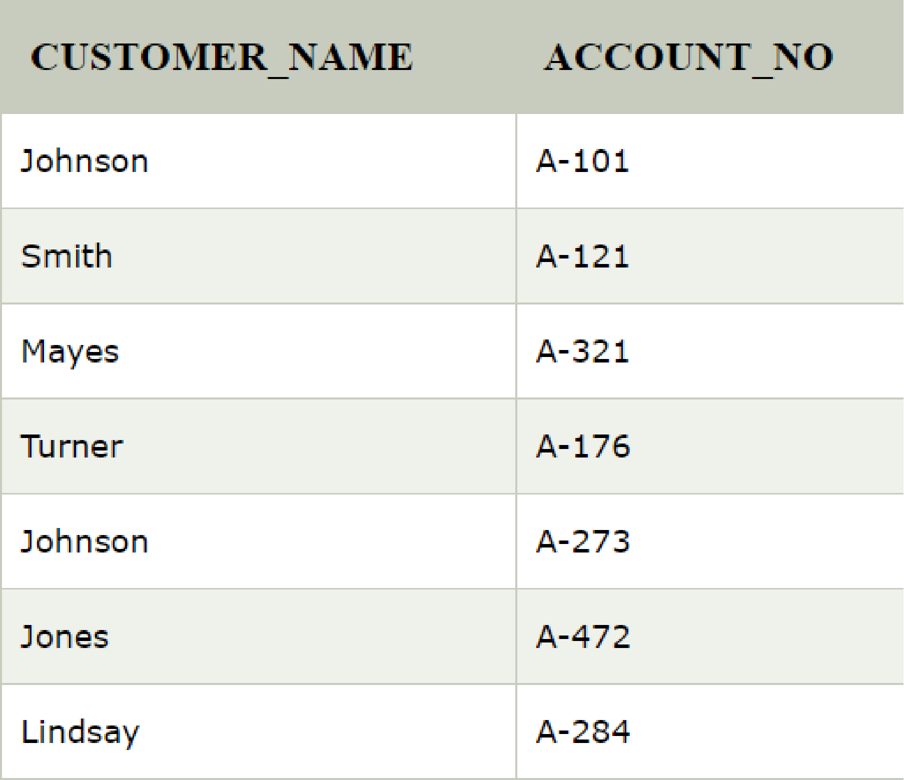


Table A ∩ B	
column 1	column 2
1	1

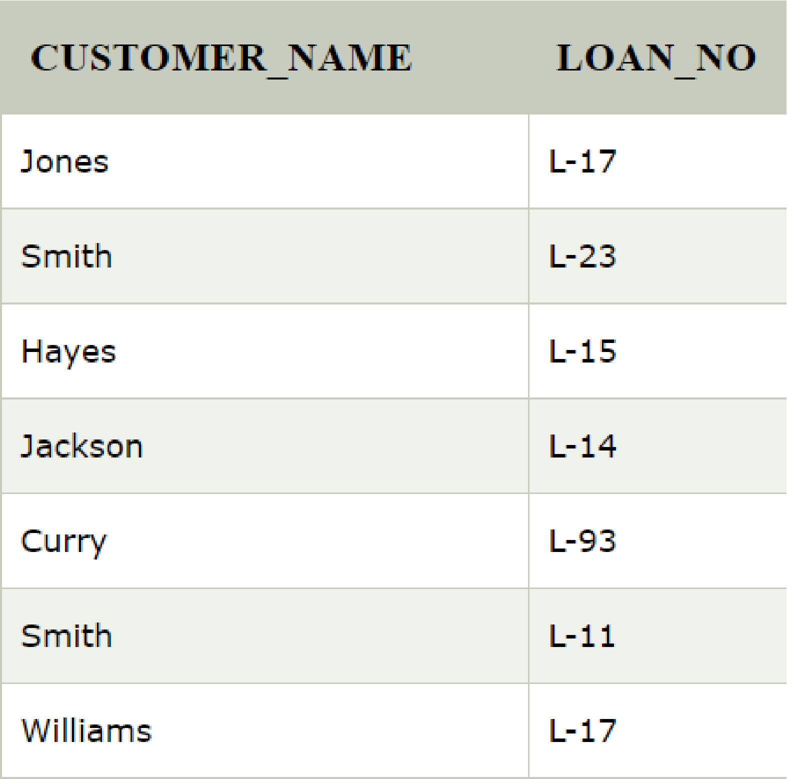
INTERSECTION

DEPOSITOR RELATION

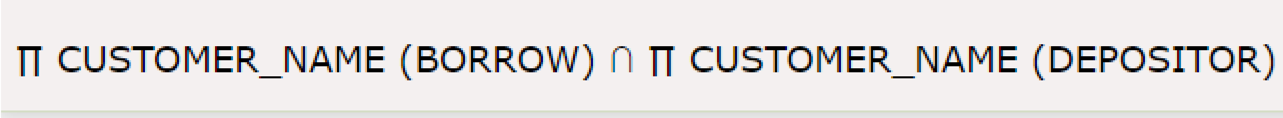


CUSTOMER_NAME	ACCOUNT_NO
Johnson	A-101
Smith	A-121
Mayes	A-321
Turner	A-176
Johnson	A-273
Jones	A-472
Lindsay	A-284

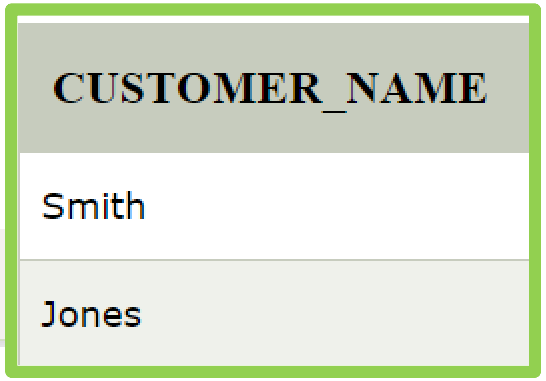
BORROW RELATION



CUSTOMER_NAME	LOAN_NO
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17



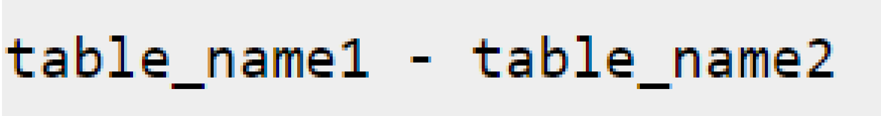
∏ CUSTOMER_NAME (BORROW) ∩ ∏ CUSTOMER_NAME (DEPOSITOR)



CUSTOMER_NAME
Smith
Jones

SET DIFFERENCE

- we have two relations R1 and R2 and selects all those tuples(rows) that are present in Relation R1 but not present in Relation R2
- denoted by – symbol
- both the relations must have the same set of attributes
- Syntax:



```
table_name1 - table_name2
```

SET DIFFERENCE

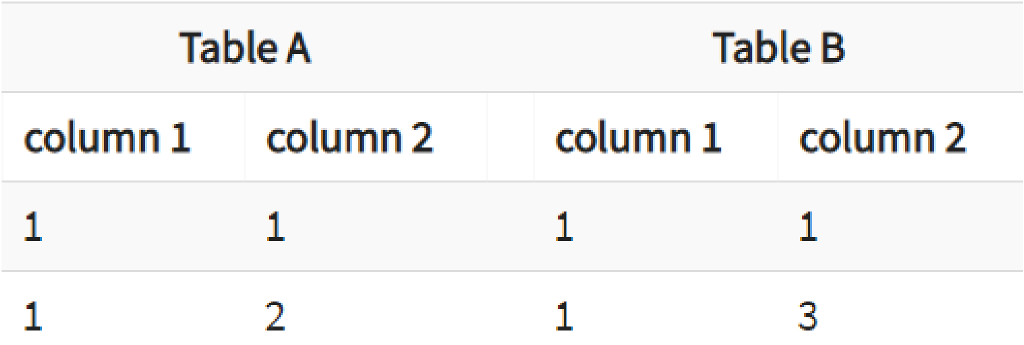
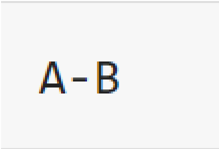


Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3



A - B

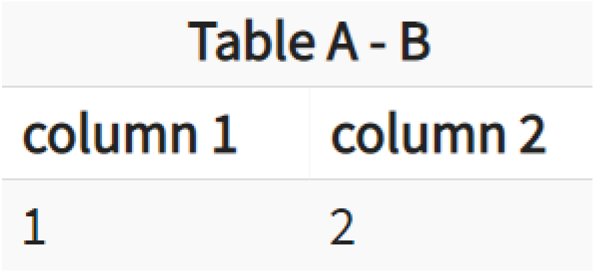
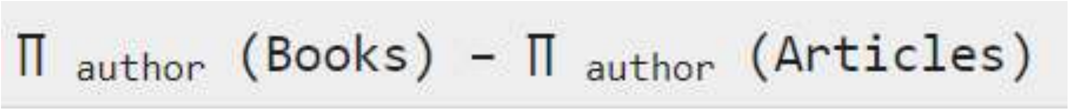


Table A - B	
column 1	column 2
1	2

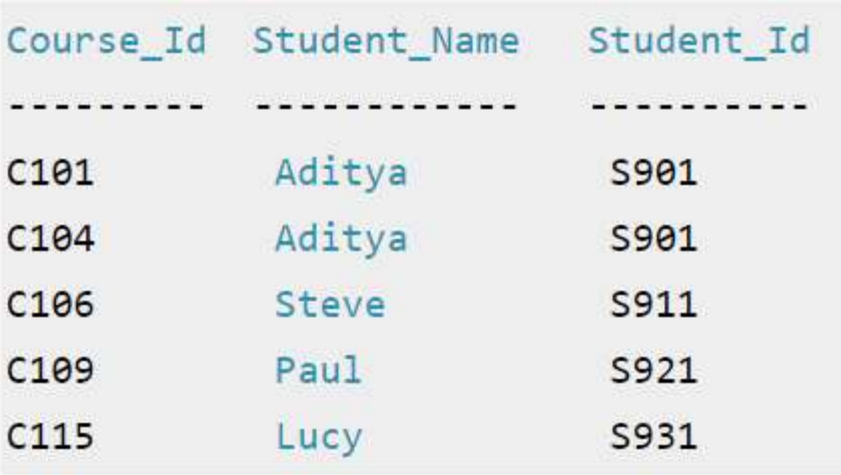


Π_{author} (Books) - Π_{author} (Articles)

Provides the name of authors who have written books but not articles

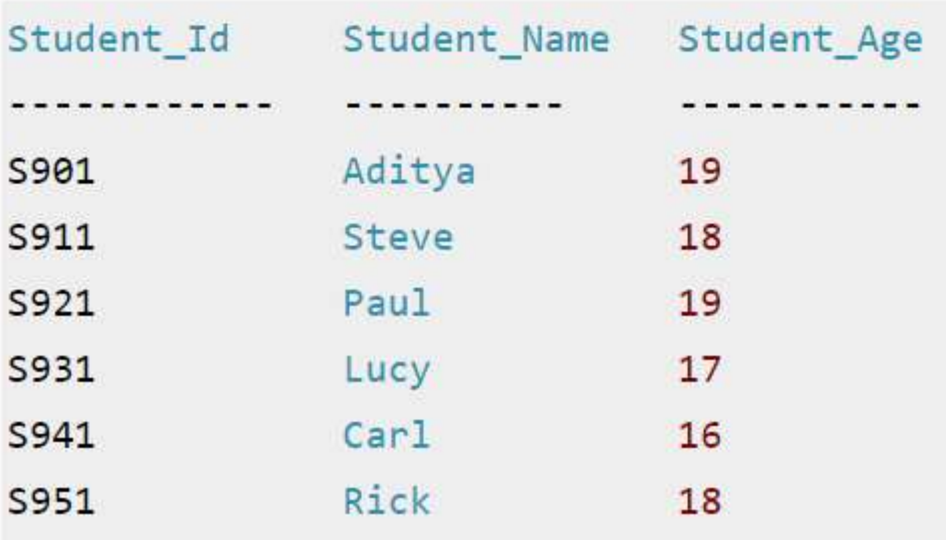
SET DIFFERENCE -QUIZ

Table 1: COURSE



Course_Id	Student_Name	Student_Id
C101	Aditya	S901
C104	Aditya	S901
C106	Steve	S911
C109	Paul	S921
C115	Lucy	S931

Table 2: STUDENT

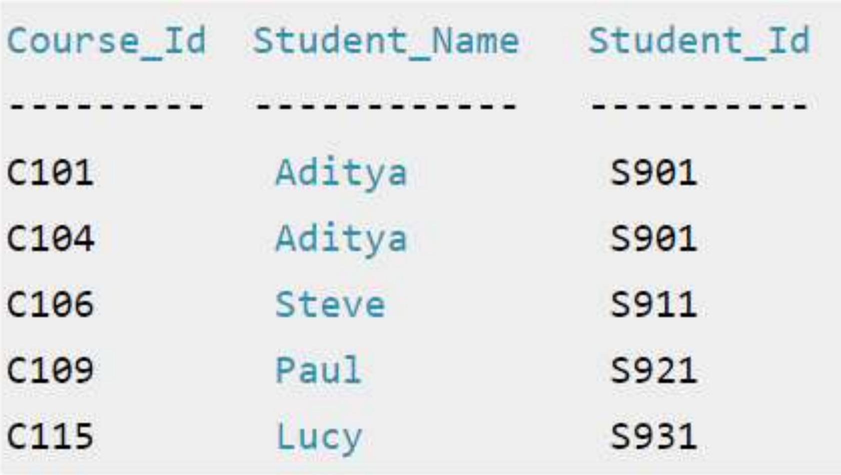


Student_Id	Student_Name	Student_Age
S901	Aditya	19
S911	Steve	18
S921	Paul	19
S931	Lucy	17
S941	Carl	16
S951	Rick	18

write a query to select those students who have not enrolled their courses

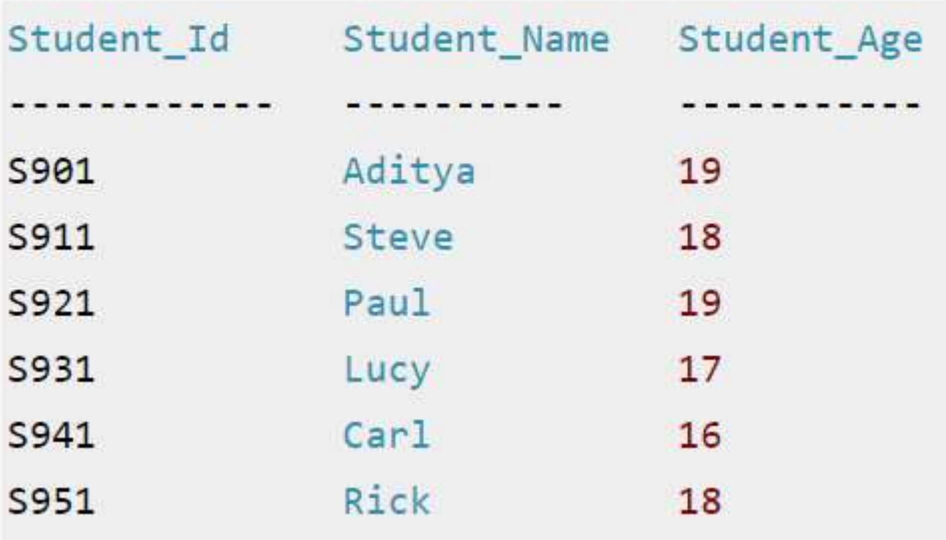
SET DIFFERENCE –QUIZ ANS

Table 1: COURSE



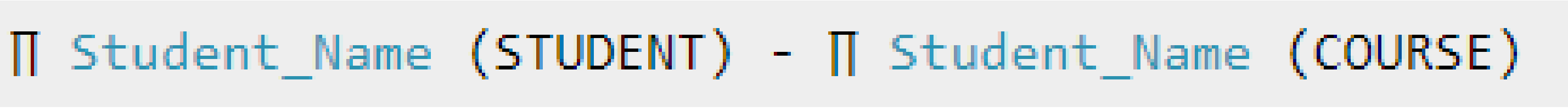
Course_Id	Student_Name	Student_Id
C101	Aditya	S901
C104	Aditya	S901
C106	Steve	S911
C109	Paul	S921
C115	Lucy	S931

Table 2: STUDENT



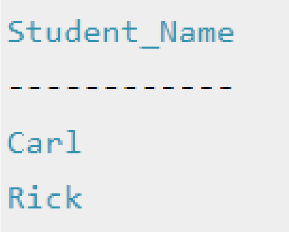
Student_Id	Student_Name	Student_Age
S901	Aditya	19
S911	Steve	18
S921	Paul	19
S931	Lucy	17
S941	Carl	16
S951	Rick	18

write a query to select those students who have not enrolled their courses



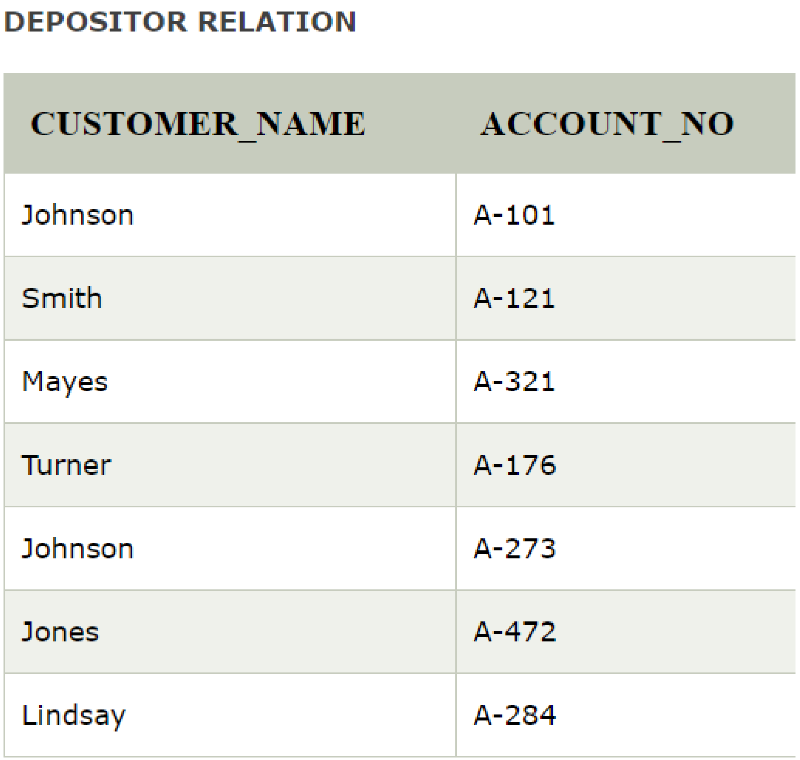
```
∏ Student_Name (STUDENT) - ∏ Student_Name (COURSE)
```

Output:



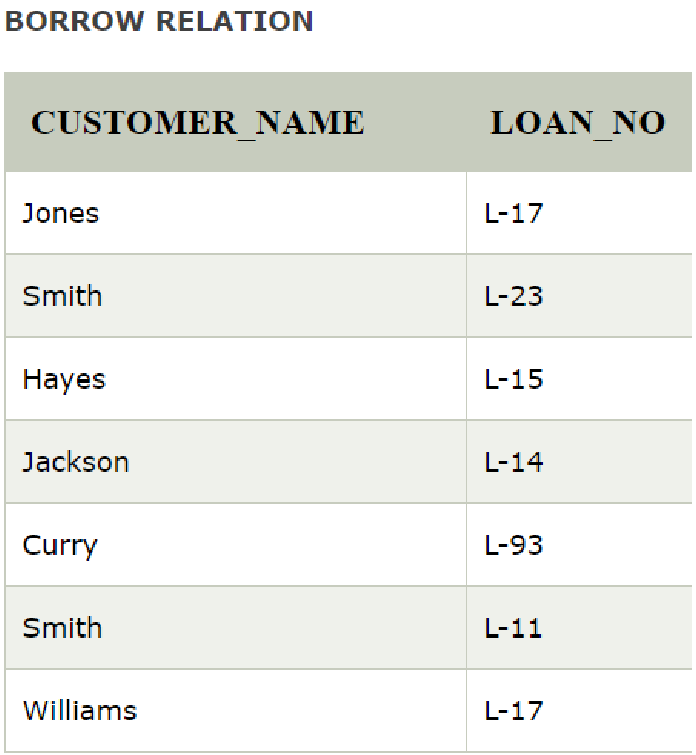
Student_Name
Carl
Rick

SET DIFFERENCE -QUIZ



DEPOSITOR RELATION

CUSTOMER_NAME	ACCOUNT_NO
Johnson	A-101
Smith	A-121
Mayes	A-321
Turner	A-176
Johnson	A-273
Jones	A-472
Lindsay	A-284



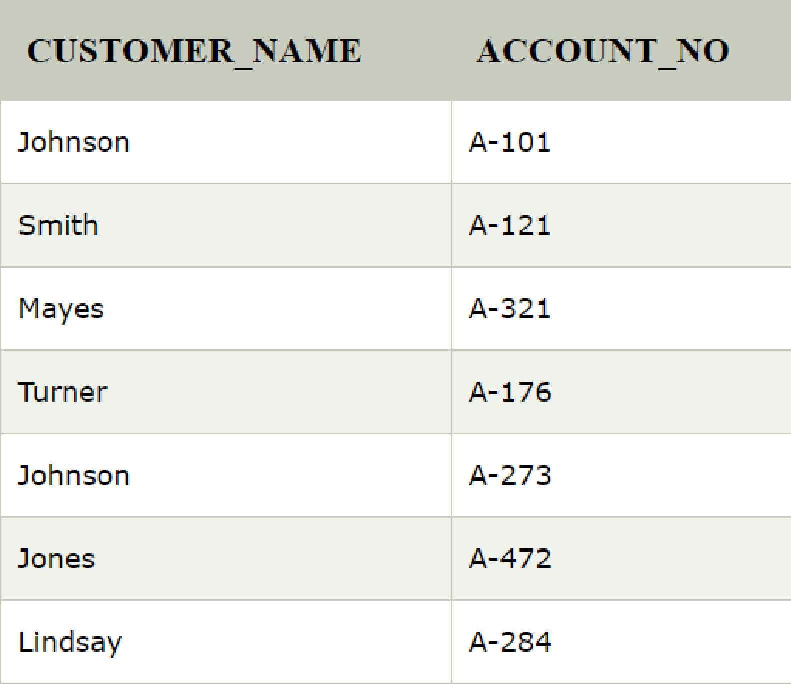
BORROW RELATION

CUSTOMER_NAME	LOAN_NO
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17

write a query to select customers who have loan but does not maintain a deposit in the bank

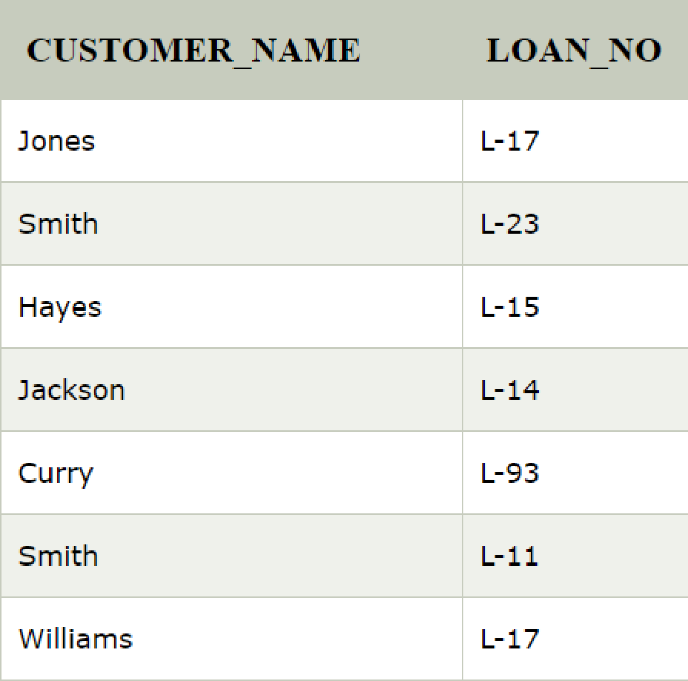
SET DIFFERENCE –QUIZ ANS

DEPOSITOR RELATION



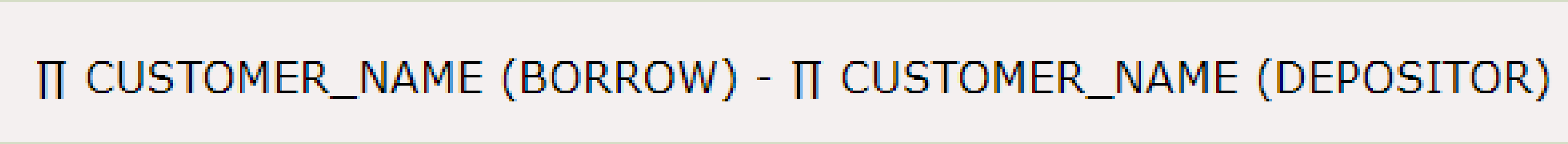
CUSTOMER_NAME	ACCOUNT_NO
Johnson	A-101
Smith	A-121
Mayes	A-321
Turner	A-176
Johnson	A-273
Jones	A-472
Lindsay	A-284

BORROW RELATION



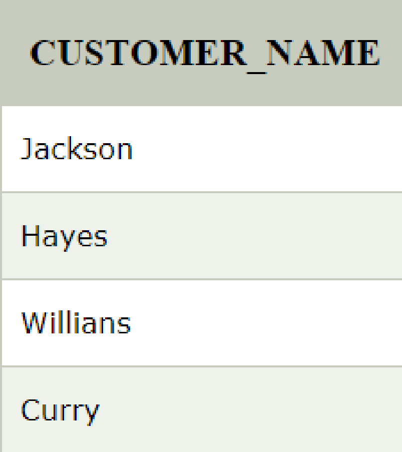
CUSTOMER_NAME	LOAN_NO
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17

write a query to select customers who have loan but does not maintain a deposit in the bank



∏ CUSTOMER_NAME (BORROW) - ∏ CUSTOMER_NAME (DEPOSITOR)

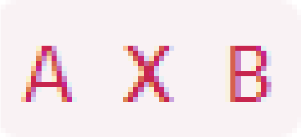
Output:



CUSTOMER_NAME
Jackson
Hayes
Willians
Curry

CARTESIAN PRODUCT

- operation used to merge columns from two relations
- Combines information of two different relations into one
- denoted by X symbol
- A X B will results all the attributes of A followed by each attribute of B
- Each record of A will pairs with every record of B
- It is also called Cross Product or Cross Join
- Syntax:

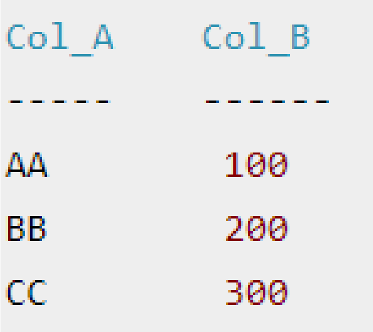


A X B

meaningful operation
when it is followed
by other operations

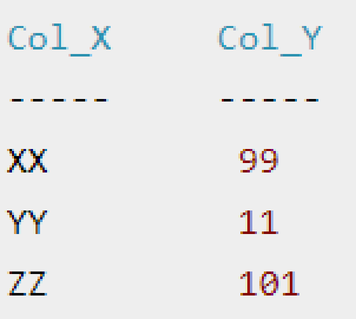
CARTESIAN PRODUCT

Table 1: R



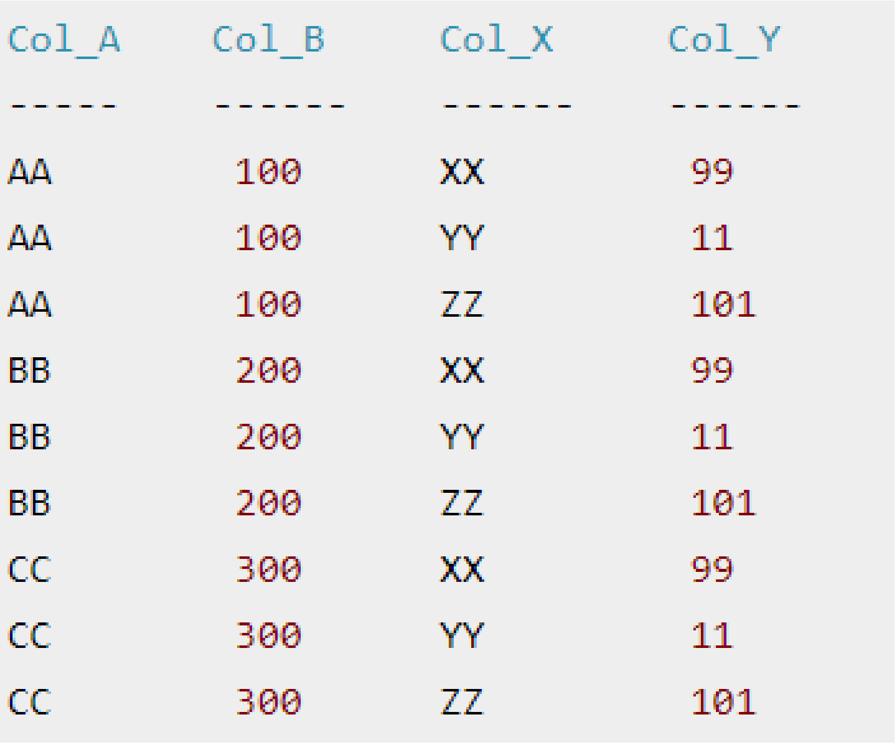
```
Col_A     Col_B
-----     ------
AA         100
BB         200
CC         300
```

Table 2: S



```
Col_X      Col_Y
-----      -----
XX          99
YY          11
ZZ          101
```

R X S



```
Col_A     Col_B     Col_X     Col_Y
-----     ------    ------    ------
AA         100      XX         99
AA         100      YY         11
AA         100      ZZ         101
BB         200      XX         99
BB         200      YY         11
BB         200      ZZ         101
CC         300      XX         99
CC         300      YY         11
CC         300      ZZ         101
```

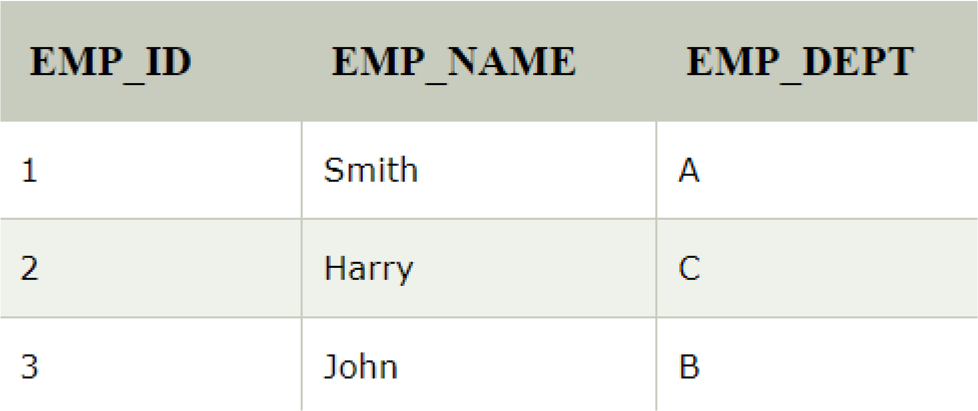
Total rows in R X S = no of rows in R x no of rows in S

= 3 x 3

= 9

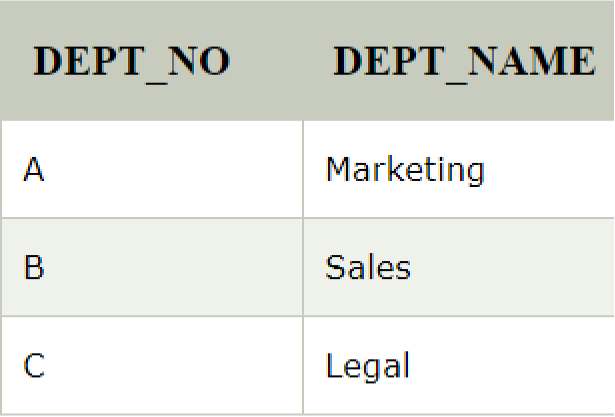
CARTESIAN PRODUCT

EMPLOYEE



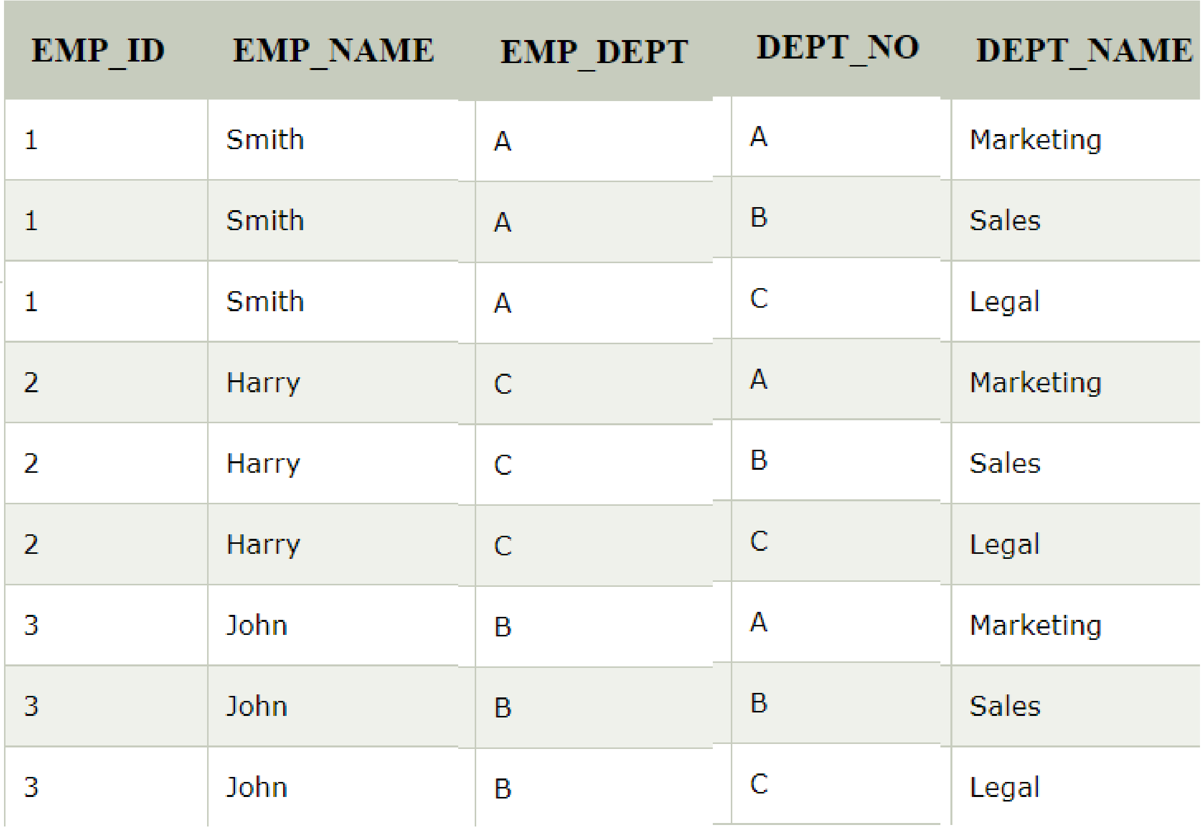
EMP_ID	EMP_NAME	EMP_DEPT
1	Smith	A
2	Harry	C
3	John	B

DEPARTMENT



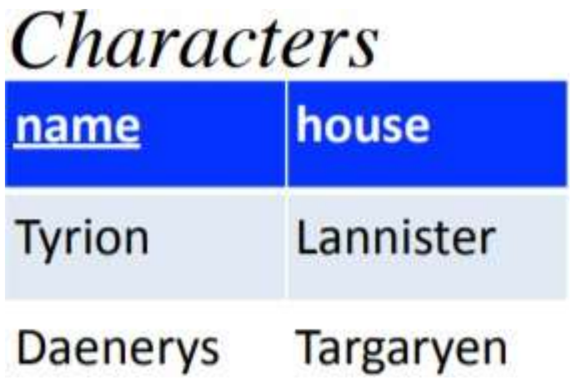
DEPT_NO	DEPT_NAME
A	Marketing
B	Sales
C	Legal

EMPLOYEE X DEPARTMENT



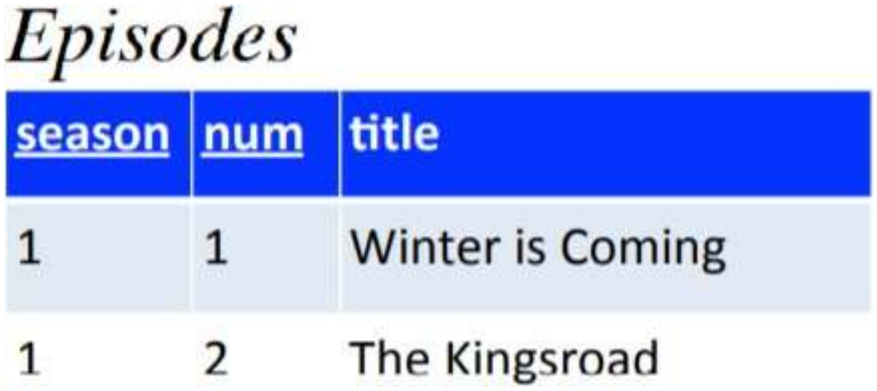
EMP_ID	EMP_NAME	EMP_DEPT	DEPT_NO	DEPT_NAME
1	Smith	A	A	Marketing
1	Smith	A	B	Sales
1	Smith	A	C	Legal
2	Harry	C	A	Marketing
2	Harry	C	B	Sales
2	Harry	C	C	Legal
3	John	B	A	Marketing
3	John	B	B	Sales
3	John	B	C	Legal

CARTESIAN PRODUCT



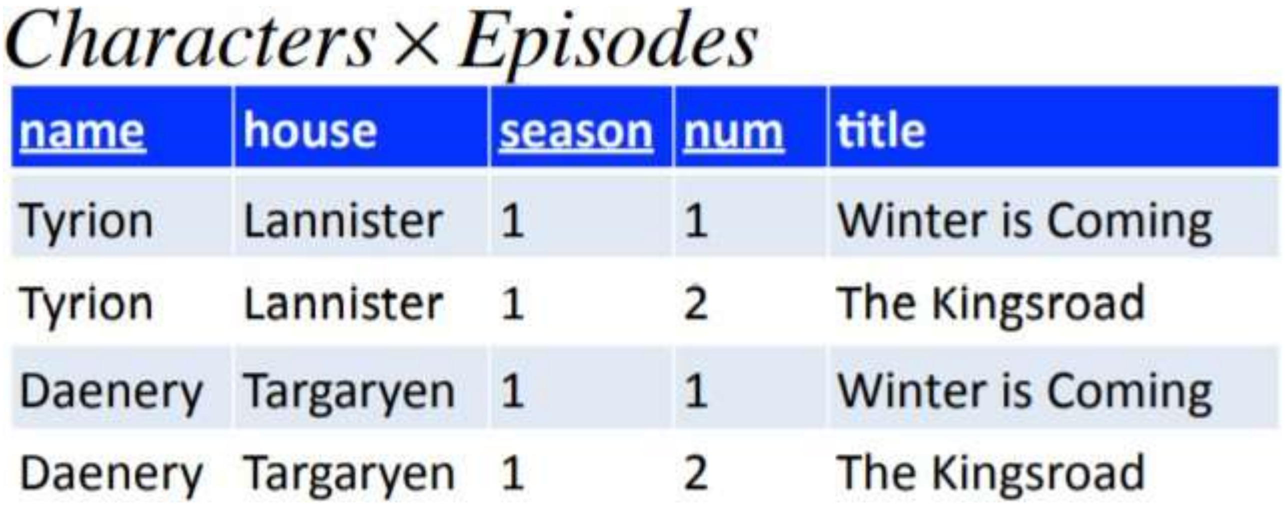
Characters

name	house
Tyrion	Lannister
Daenerys	Targaryen



Episodes

season	num	title
1	1	Winter is Coming
1	2	The Kingsroad



Characters × *Episodes*

name	house	season	num	title
Tyrion	Lannister	1	1	Winter is Coming
Tyrion	Lannister	1	2	The Kingsroad
Daenery	Targaryen	1	1	Winter is Coming
Daenery	Targaryen	1	2	The Kingsroad

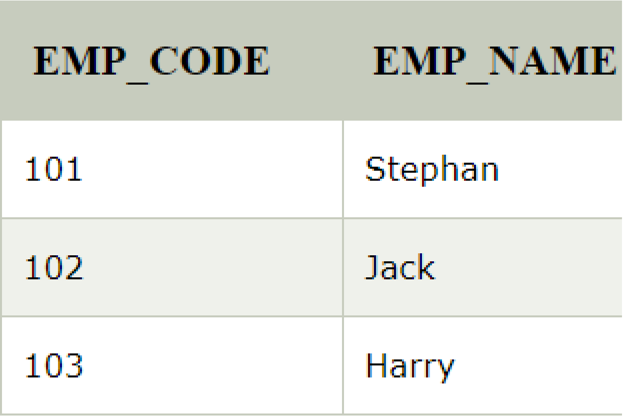
JOINS

- selectively pairs up tuples from two relations
- Join operation is essentially a cartesian product followed by a selection criterion.
- denoted by ⋈.
- combines related tuples from different relations, if and only if a given join condition is satisfied
- Syntax:

Relation1 $\bowtie_{\text{condition}}$ **Relation2**

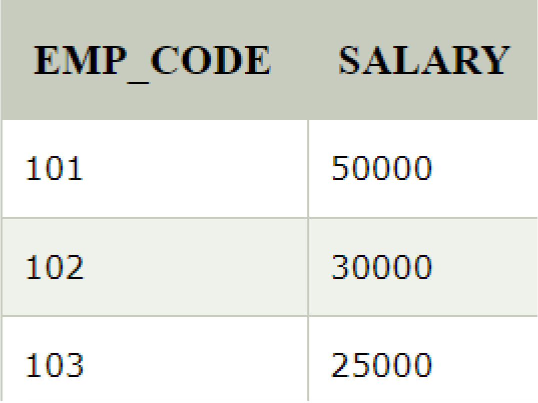
JOINS - EXAMPLE

EMPLOYEE



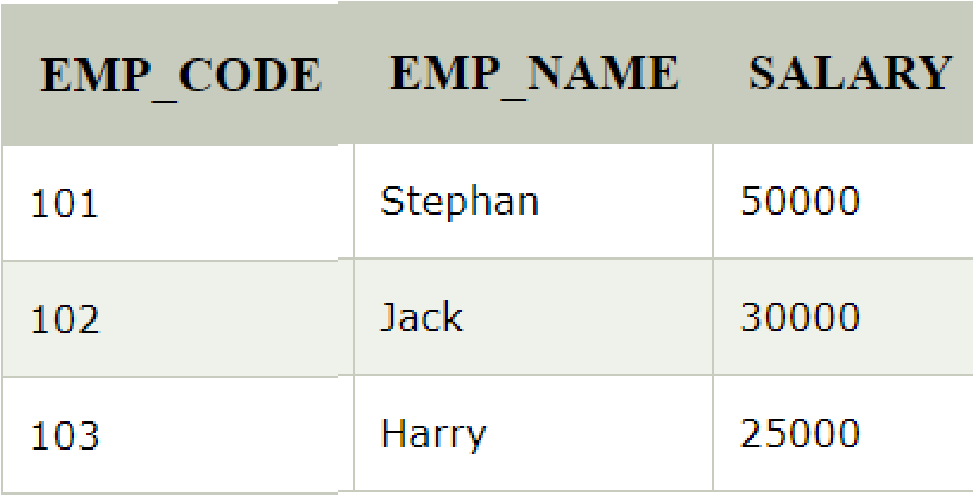
EMP_CODE	EMP_NAME
101	Stephan
102	Jack
103	Harry

SALARY



EMP_CODE	SALARY
101	50000
102	30000
103	25000

EMPLOYEE ⋈ SALARY



EMP_CODE	EMP_NAME	SALARY
101	Stephan	50000
102	Jack	30000
103	Harry	25000

JOINS - EXAMPLE

Characters

name	house
Tyrion	Lannister
Daenerys	Targaryen

Appearances

name	season	num
Jon Snow	2	1
Tyrion	1	1
Tyrion	2	2
Daenerys	1	2

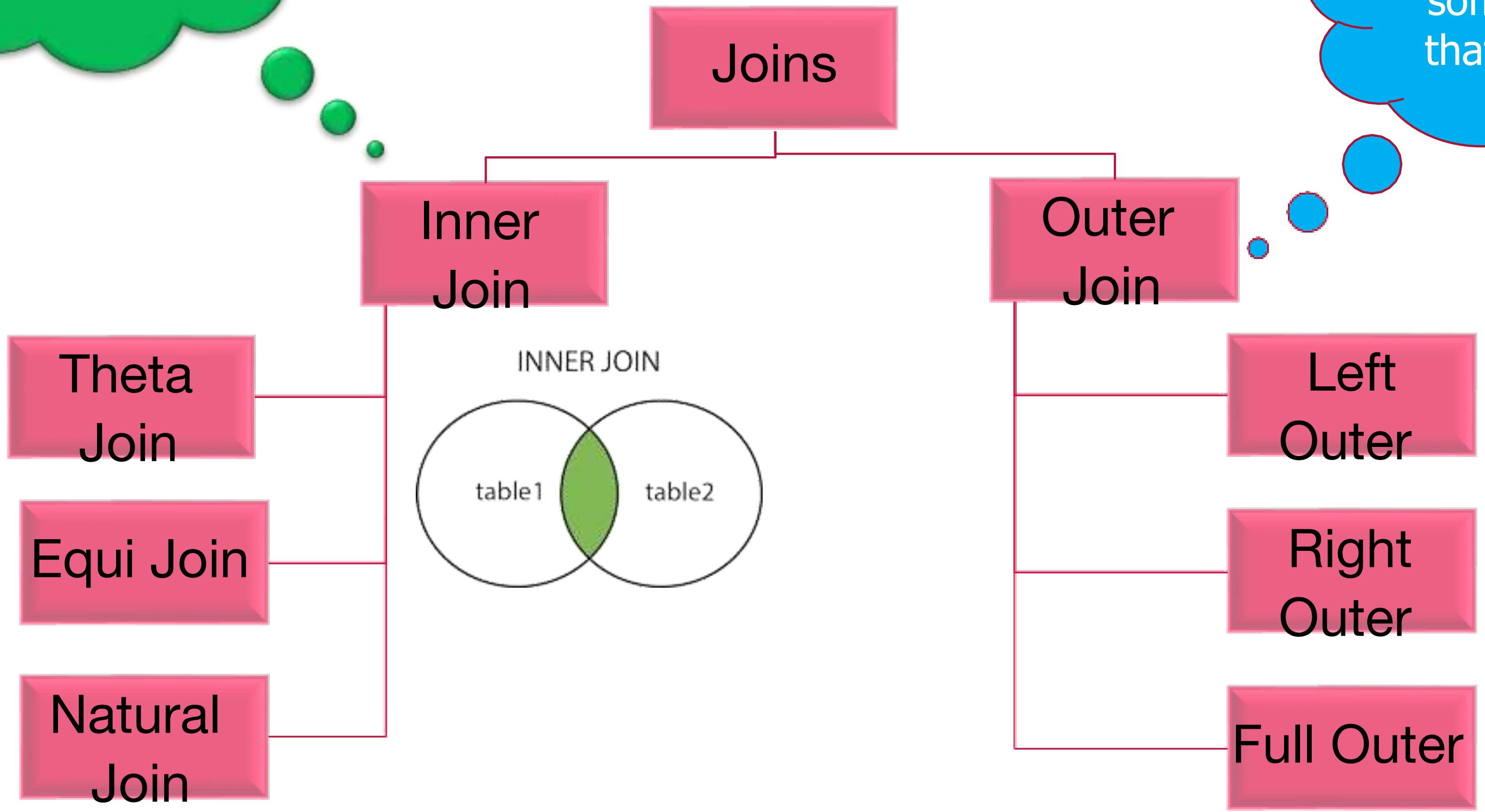
$$Characters \bowtie_{name} Appearances$$

name	house	name	season	num
Tyrion	Lannister	Tyrion	1	1
Tyrion	Lannister	Tyrion	2	2
Daenerys	Targaryen	Daenery	1	2

JOINS - TYPES

Includes tuples that satisfy the matching criteria

Include tuples that satisfy the criteria, some or all tuples that do not match the criteria



Joins

Inner Join

Outer Join

Theta Join

Equi Join

Natural Join

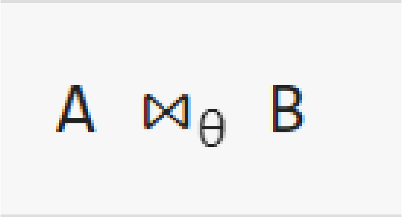
Left Outer

Right Outer

Full Outer

INNER JOIN – THETA JOIN

- general case of JOIN operation
- denoted by symbol θ
- combines tuples from different relations provided they satisfy the theta condition
- Syntax



$$A \bowtie_{\theta} B$$

INNER JOIN – THETA JOIN

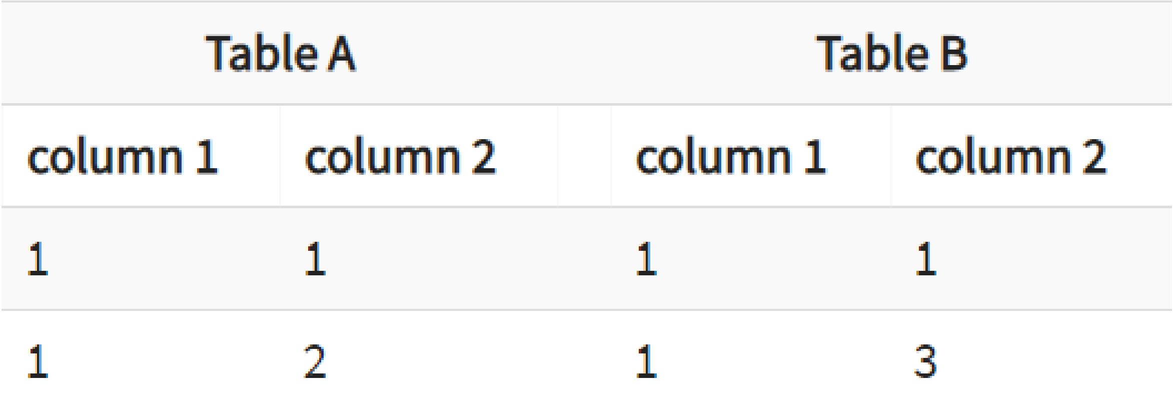
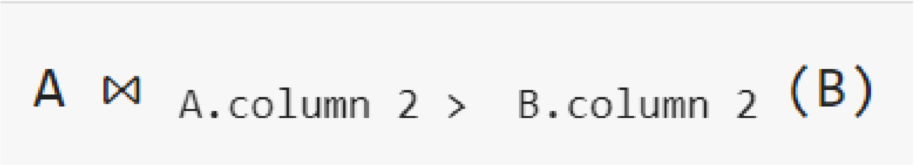
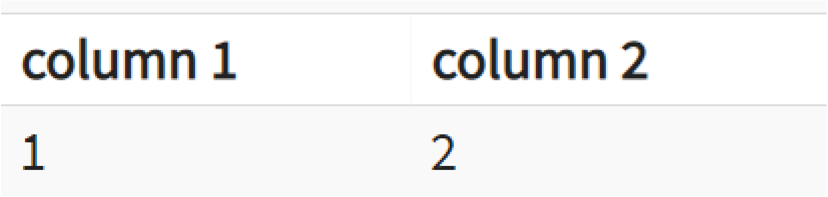


Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3

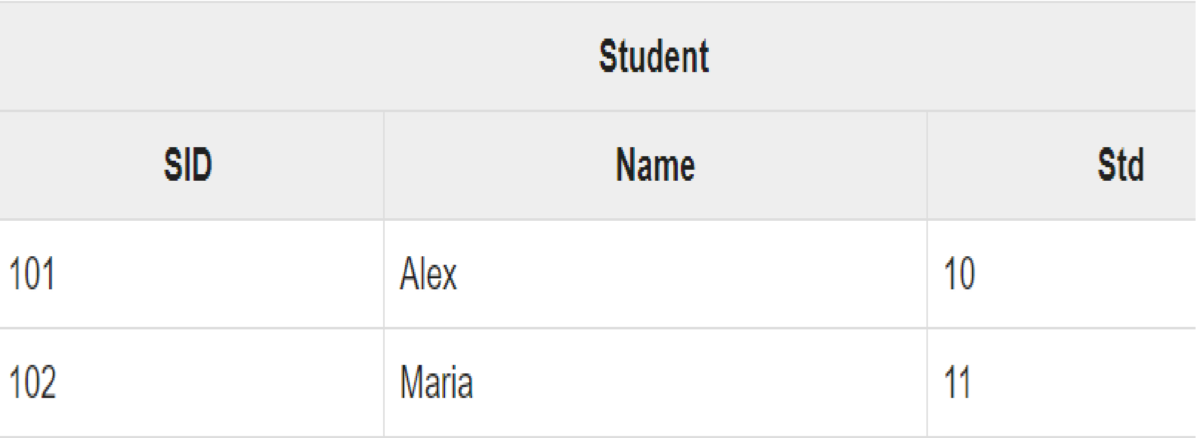


$A \bowtie_{\texttt{A.column 2 > B.column 2}} (B)$

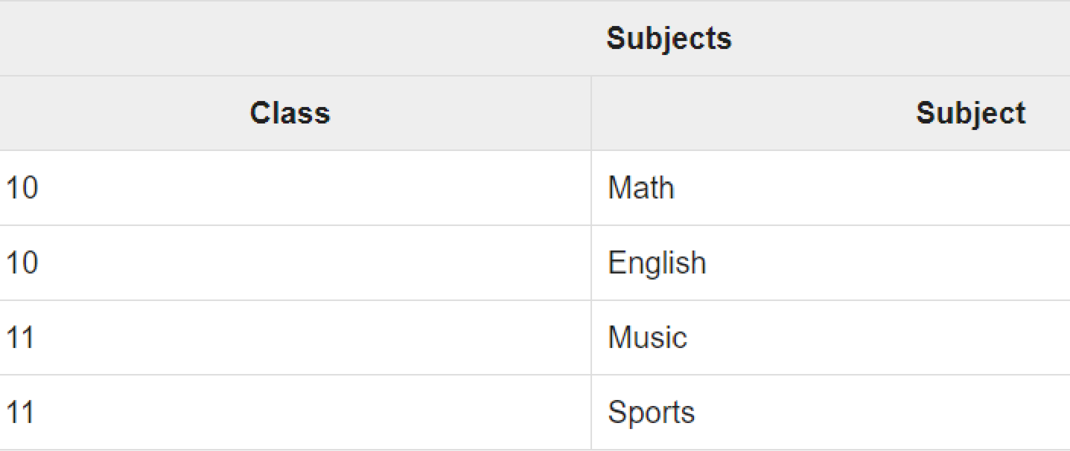


column 1	column 2
1	2

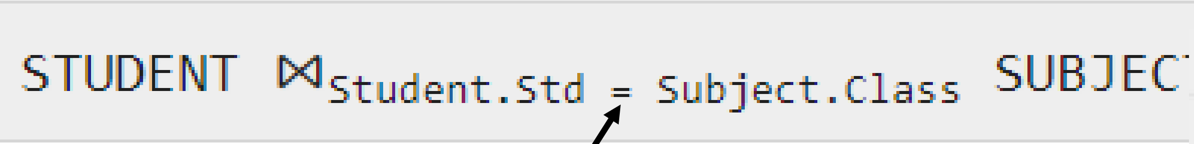
INNER JOIN – THETA JOIN



Student		
SID	Name	Std
101	Alex	10
102	Maria	11

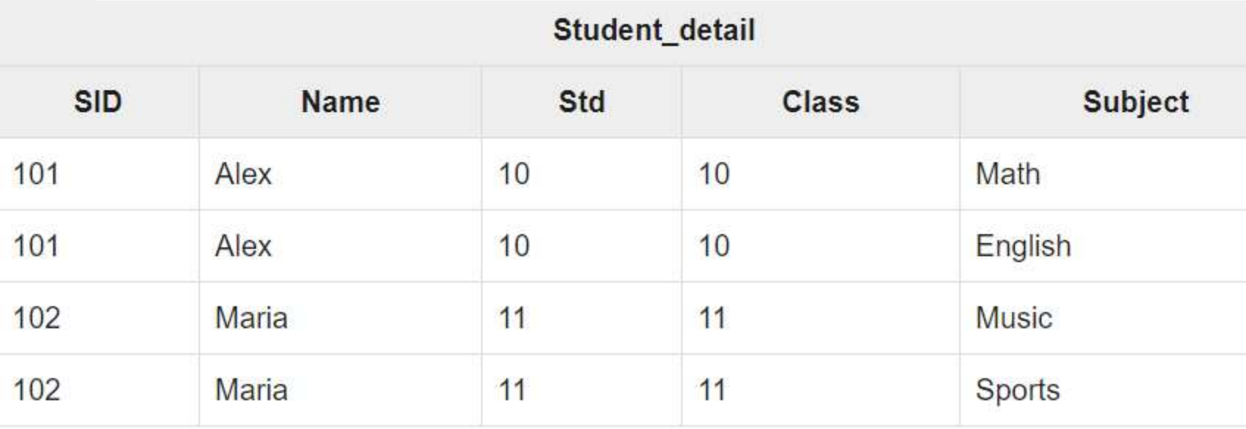


Subjects	
Class	Subject
10	Math
10	English
11	Music
11	Sports



$\text{STUDENT} \bowtie_{\text{Student.Std = Subject.Class}}$ SUBJEC

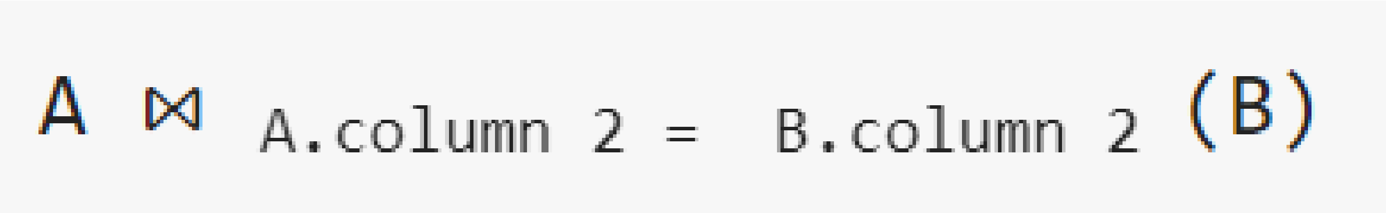
Used a = opeartor → **Equi Join**



Student_detail				
SID	Name	Std	Class	Subject
101	Alex	10	10	Math
101	Alex	10	10	English
102	Maria	11	11	Music
102	Maria	11	11	Sports

INNER JOIN – EQUI JOIN

- When Theta join uses only equality comparison operator, it is said to be equijoin
- special case of conditional join where only equality condition holds between a pair of attributes
- As values of two attributes will be equal in result of equijoin, only one attribute will be appeared in result



$$A \bowtie_{A.\text{column } 2 \;=\; B.\text{column } 2} (B)$$

INNER JOIN – EQUI JOIN

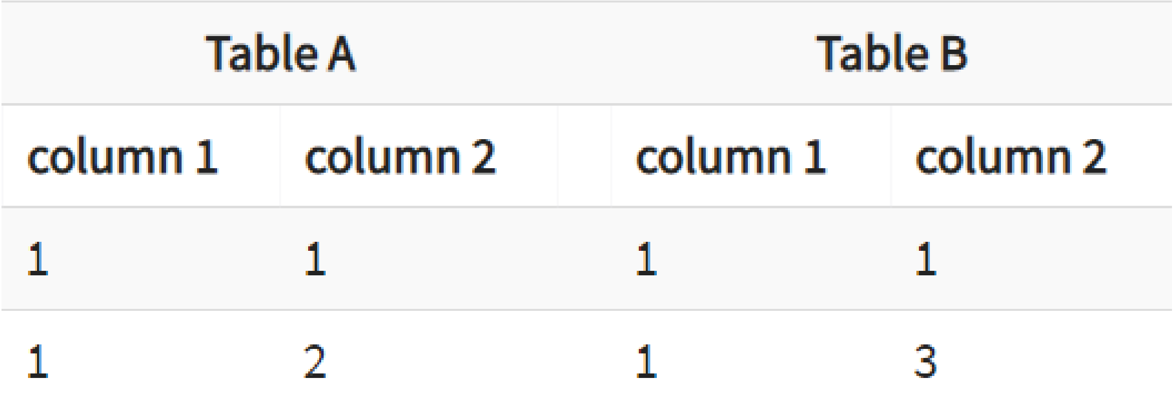
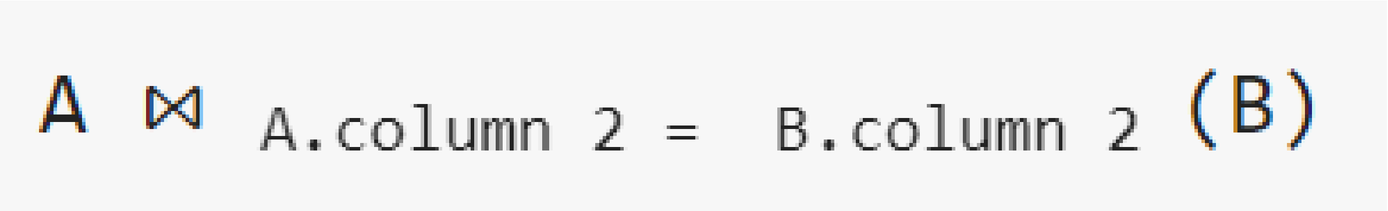
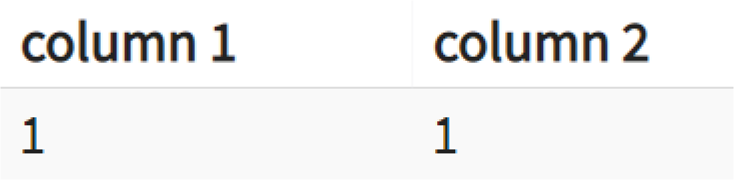


Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3



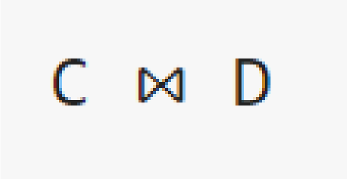
$$A \bowtie_{\texttt{A.column 2 = B.column 2}} (B)$$



column 1	column 2
1	1

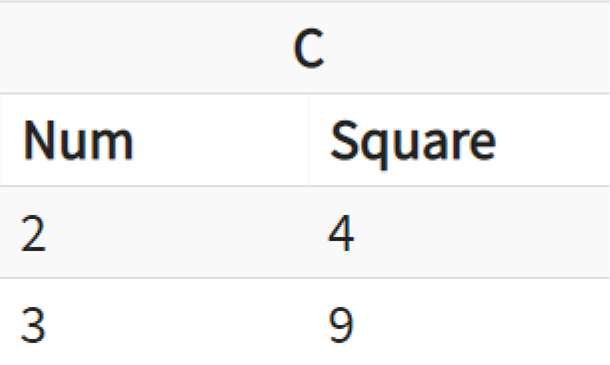
INNER JOIN – NATURAL JOIN

- binary operator
- can only be performed if there is a common attribute (column) between the relations.
- set of tuples of all combinations in R and S that are equal on their common attribute names
- does not use any comparison operator. It does not concatenate the way a Cartesian product does
- name and type of the attribute must be same.
- Syntax:

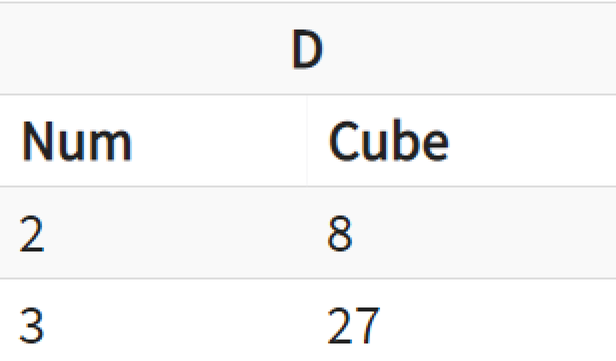


$C \bowtie D$

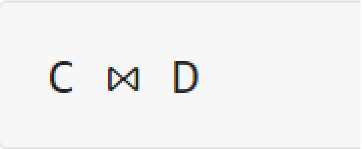
INNER JOIN – NATURAL JOIN



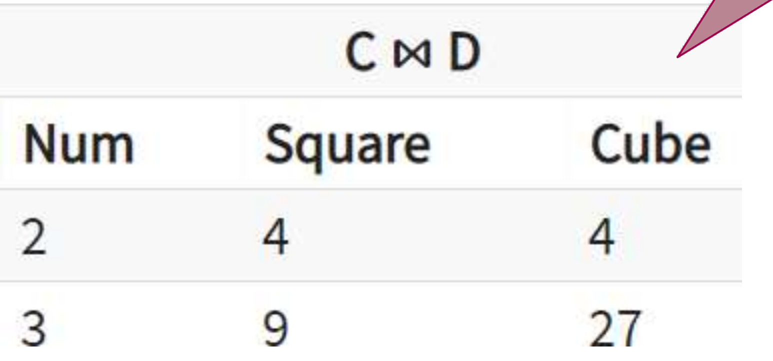
C	
Num	Square
2	4
3	9



D	
Num	Cube
2	8
3	27



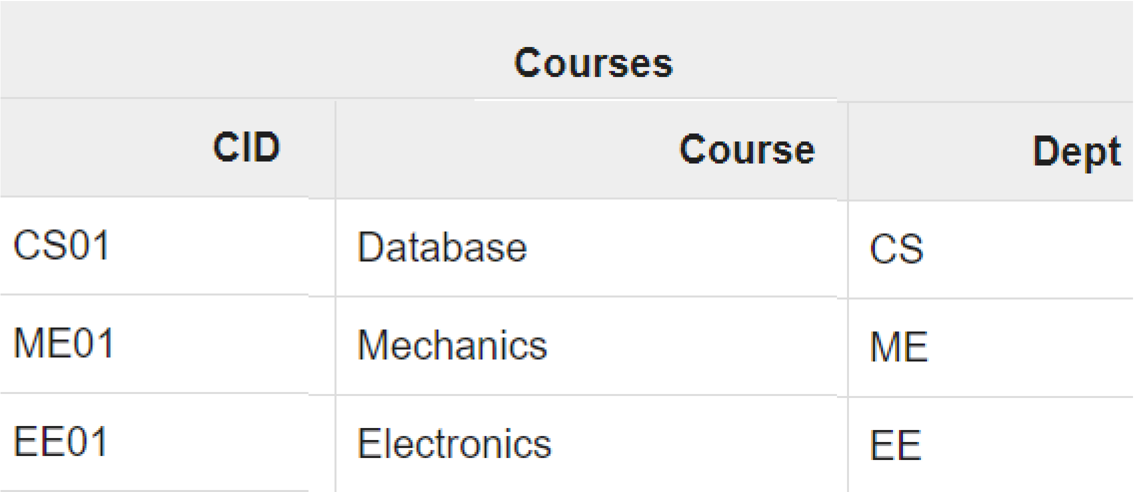
C ⋈ D



C ⋈ D		
Num	Square	Cube
2	4	4
3	9	27

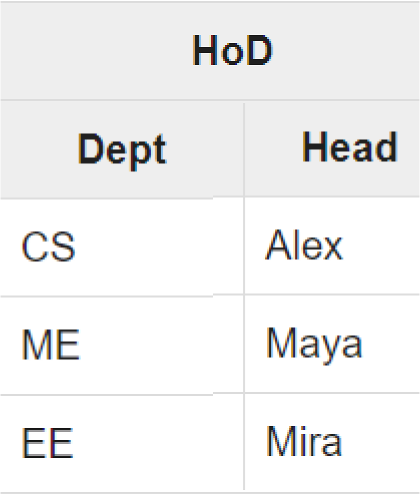
acts on those matching attributes where the values of attributes in both the relations are same

INNER JOIN – NATURAL JOIN



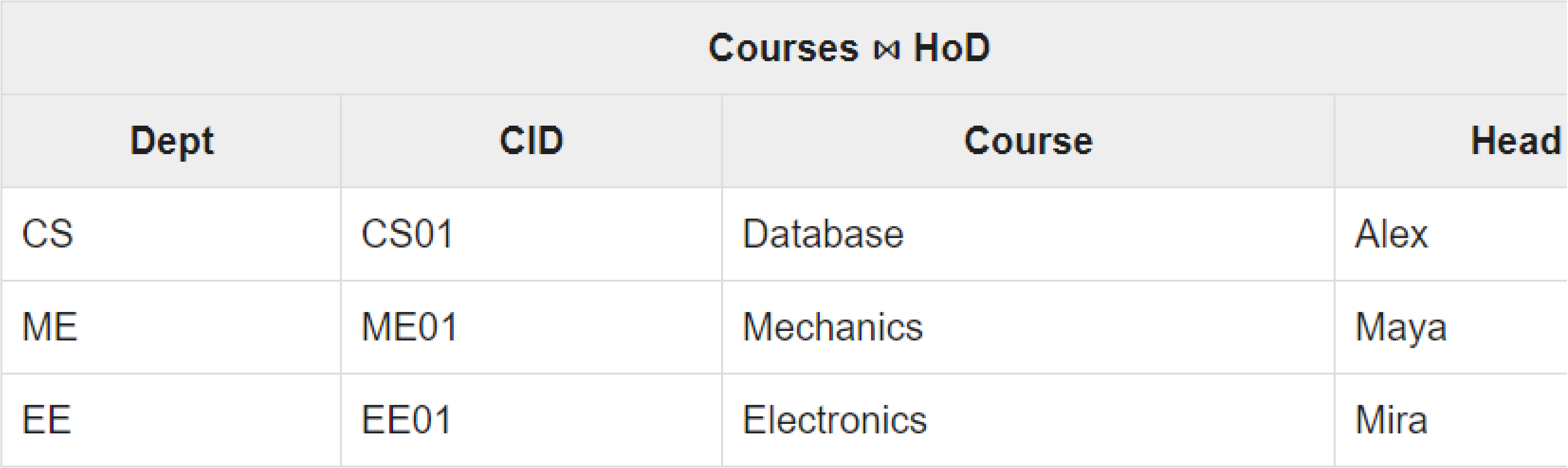
Courses

CID	Course	Dept
CS01	Database	CS
ME01	Mechanics	ME
EE01	Electronics	EE



HoD

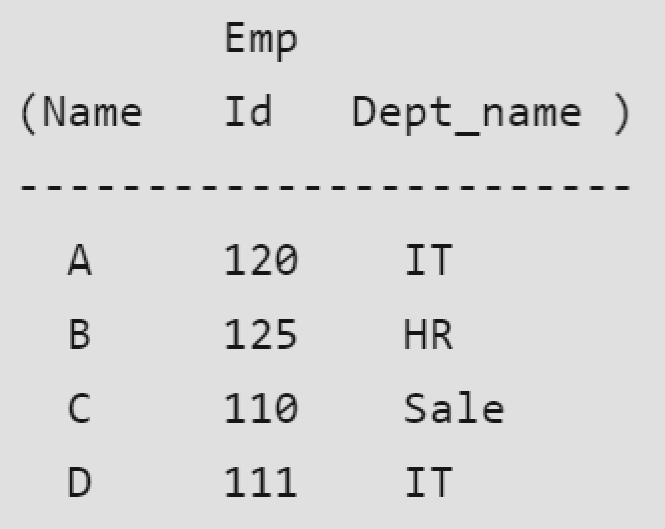
Dept	Head
CS	Alex
ME	Maya
EE	Mira



Courses ⋈ HoD

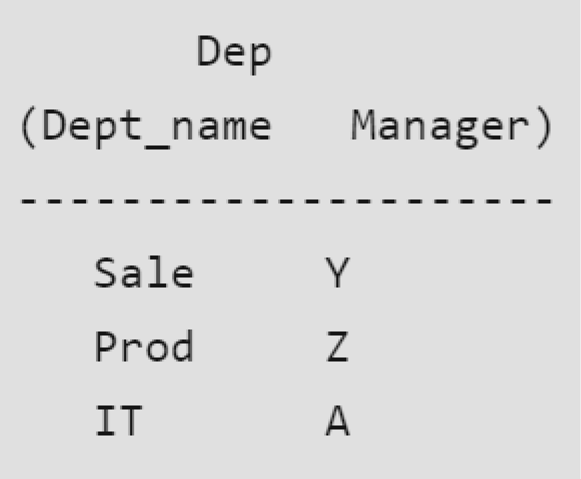
Dept	CID	Course	Head
CS	CS01	Database	Alex
ME	ME01	Mechanics	Maya
EE	EE01	Electronics	Mira

INNER JOIN - NATURAL JOIN



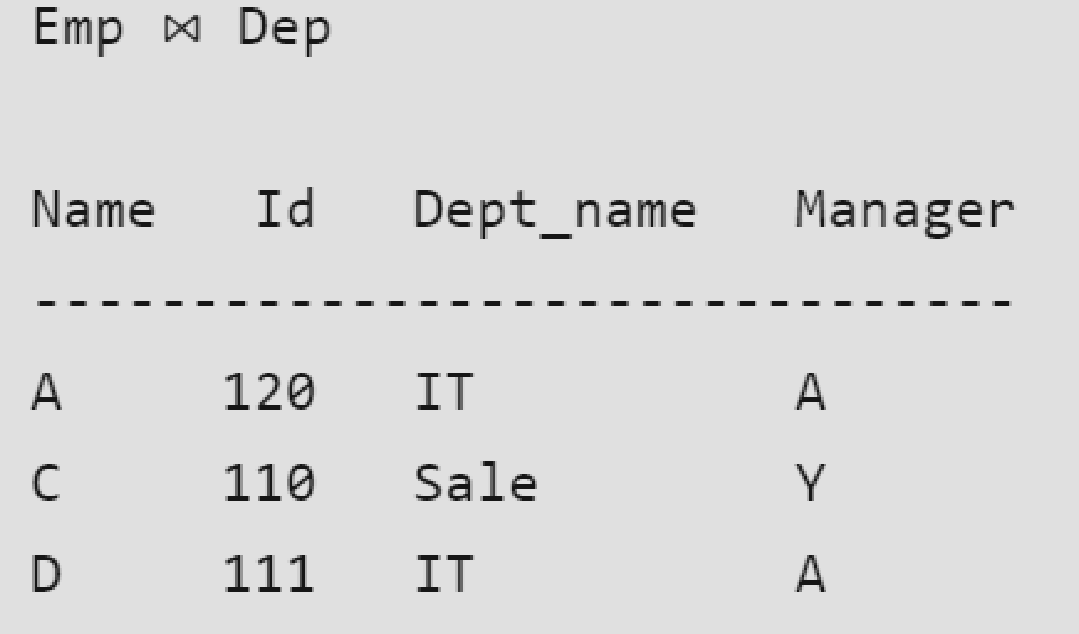
```
          Emp
(Name     Id    Dept_name )
------------------------

  A       120     IT
  B       125     HR
  C       110     Sale
  D       111     IT
```



```
          Dep
(Dept_name     Manager)
---------------------

   Sale        Y
   Prod        Z
   IT          A
```



```
Emp ⋈ Dep


Name    Id    Dept_name    Manager
--------------------------------

A      120    IT           A
C      110    Sale         Y
D      111    IT           A
```

JOIN

R

sid	name	gpa
1111	Joe	3.2
2222	Ann	4.0
3333	Mike	3.5

S

sid	did	cid	term	grade
1111	1	210	Fall 2012	A
2222	1	220	Winter 2013	

$R \bowtie S$

R.sid	R.name	R.gpa	S.sid	S.did	S.cid	S.term	S.grade
1111	Joe	3.2	1111	1	210	Fall 2012	A
2222	Ann	4.0	2222	1	220	Winter 2013	

What are the names of students who got an A in any course?

Students

sid	name	gpa
1111	Joe	3.2
2222	Ann	4.0
3333	Mike	3.5

Enrollment

sid	did	cid	term	grade
1111	1	210	Fall 2015	A
2222	1	220	Winter 2016	

$(\mathit{Students} \bowtie \mathit{Enrollment})$,

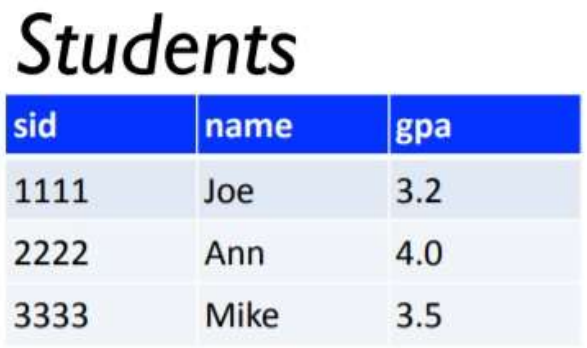
R.sid	R.name	R.gpa	S.sid	S.did	S.cid	S.term	S.grade
1111	Joe	3.2	1111	1	210	Fall 2012	A
2222	Ann	4.0	2222	1	220	Winter 2013	

$$(\ \sigma_{grade='A'}(\ \mathit{Students} \bowtie \mathit{Enrollment}))$$

$$\pi_{name}(\ \sigma_{grade='A'}(\ \mathit{Students} \bowtie \mathit{Enrollment}))$$

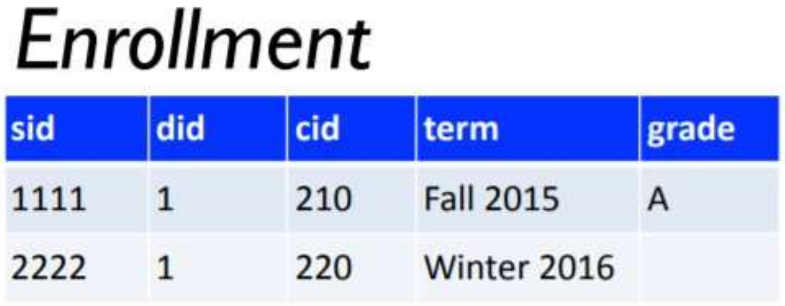
name
Joe

What are the names of students who got an A in any course?



Students

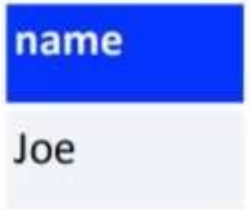
sid	name	gpa
1111	Joe	3.2
2222	Ann	4.0
3333	Mike	3.5



Enrollment

sid	did	cid	term	grade
1111	1	210	Fall 2015	A
2222	1	220	Winter 2016	

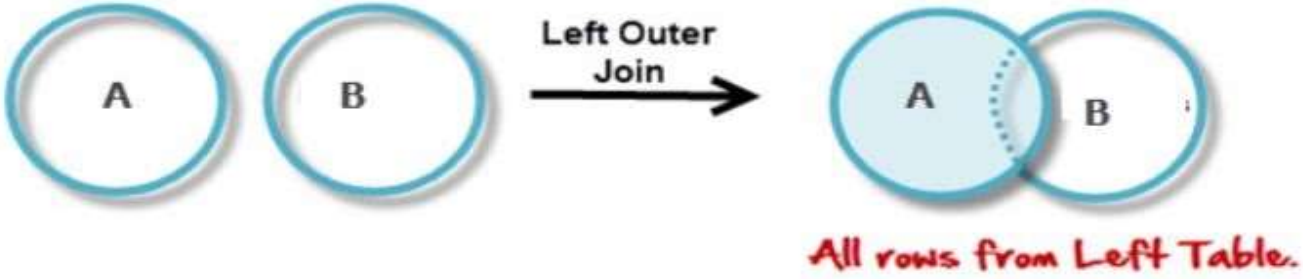
$$\pi_{name} \left(Students \bowtie \left(\sigma_{grade='A'} \; Enrollment \right) \right)$$



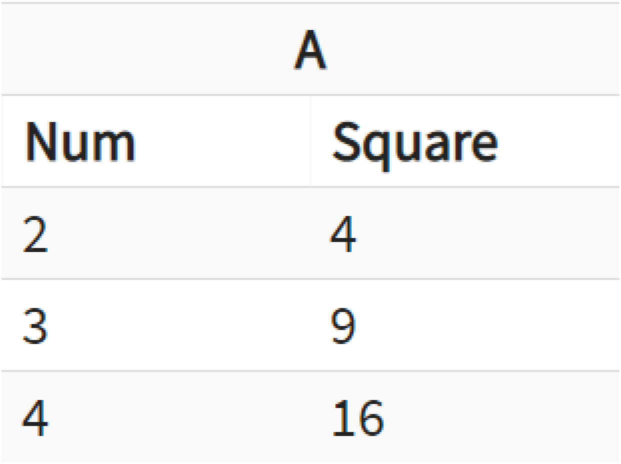
name
Joe

OUTER JOIN – LEFT JOIN

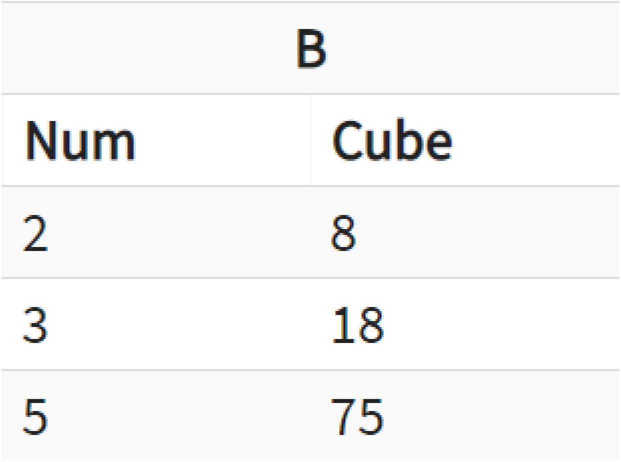
- Select records from the first (left-most) table with matching right table records
- join starting with the first (left-most) table.
- Then, any matched records from the second table (right-most) will be included
- there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values
- Syntax : A ⟕ B



LEFT JOIN - EXAMPLE



A	
Num	Square
2	4
3	9
4	16

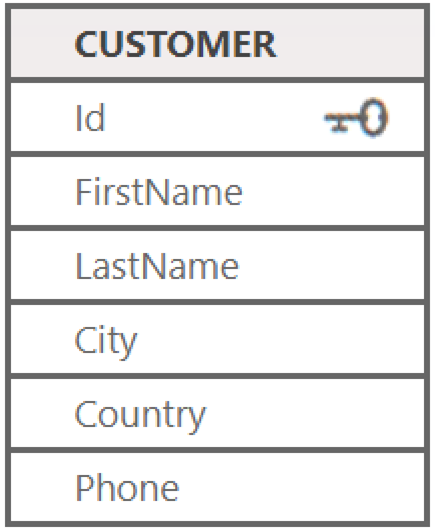


B	
Num	Cube
2	8
3	18
5	75

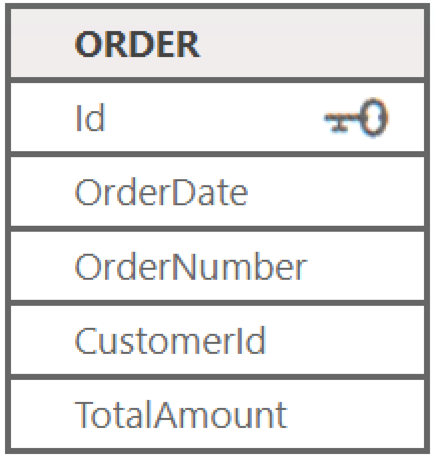
$A ⟕ B$

Num	Square	Cube
2	4	8
3	9	18
4	16	-

LEFT JOIN - EXAMPLE

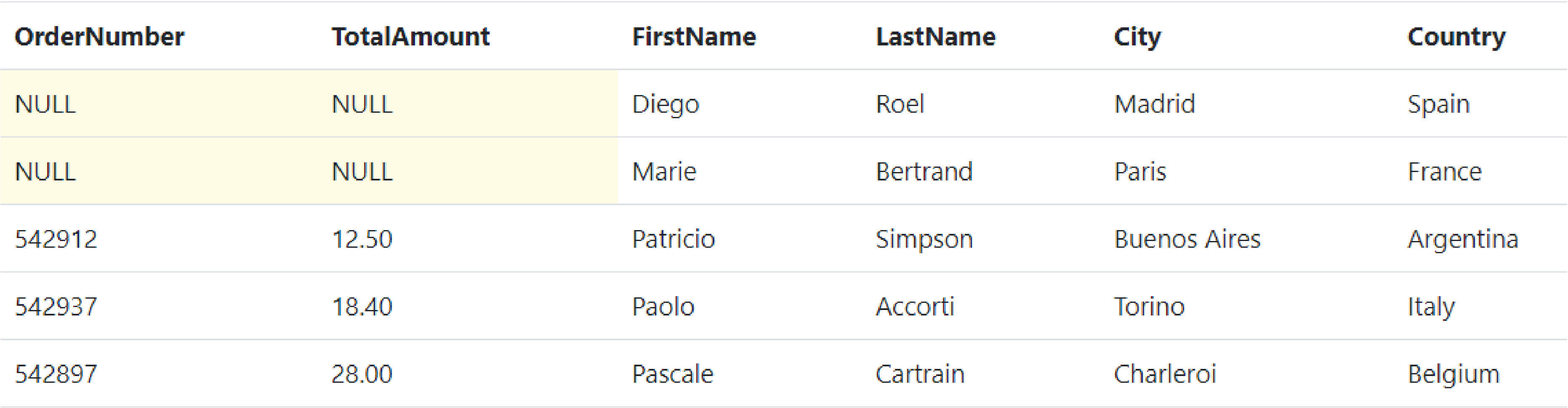


CUSTOMER
Id 🔑
FirstName
LastName
City
Country
Phone



ORDER
Id 🔑
OrderDate
OrderNumber
CustomerId
TotalAmount

List all customers and the total amount they spent irrespective whether they placed any orders or not.

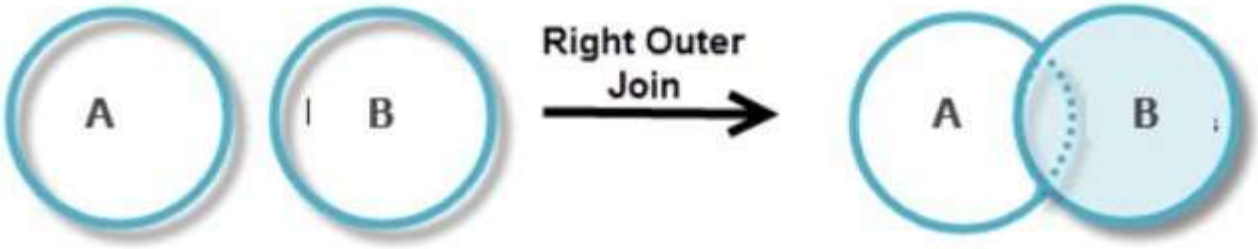


OrderNumber	TotalAmount	FirstName	LastName	City	Country
NULL	NULL	Diego	Roel	Madrid	Spain
NULL	NULL	Marie	Bertrand	Paris	France
542912	12.50	Patricio	Simpson	Buenos Aires	Argentina
542937	18.40	Paolo	Accorti	Torino	Italy
542897	28.00	Pascale	Cartrain	Charleroi	Belgium

RIGHT JOIN

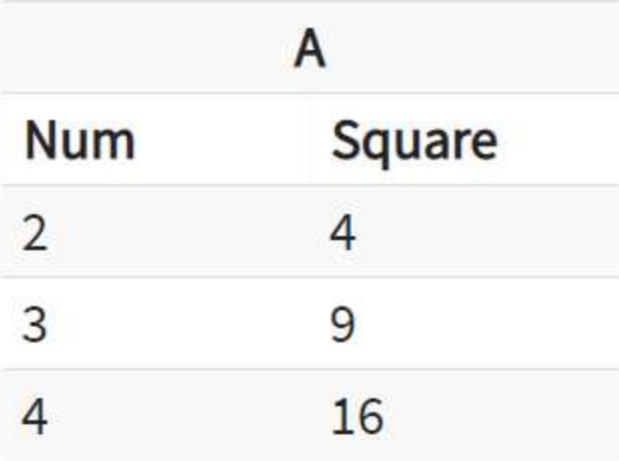
- operation allows keeping all tuple in the right relation
- join starting with the second (right-most) table and then any matching first (left-most) table records
- no matching tuple is found in the left relation, then the attributes of the left relation in the join result are filled with null values
- Syntax:

$A ⟖ B$

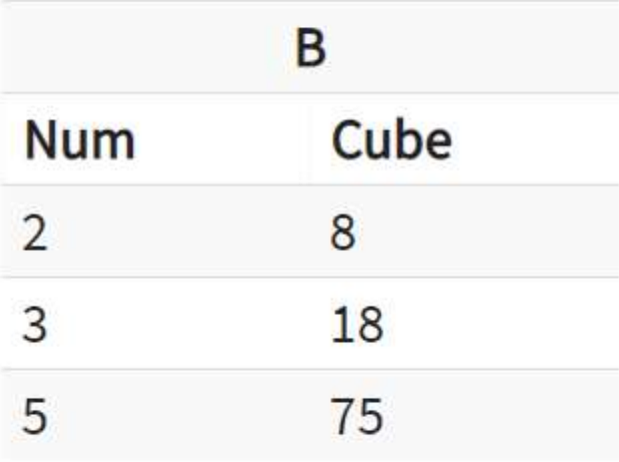


All rows from Right Table.

RIGHT JOIN - EXAMPLE



A	
Num	Square
2	4
3	9
4	16

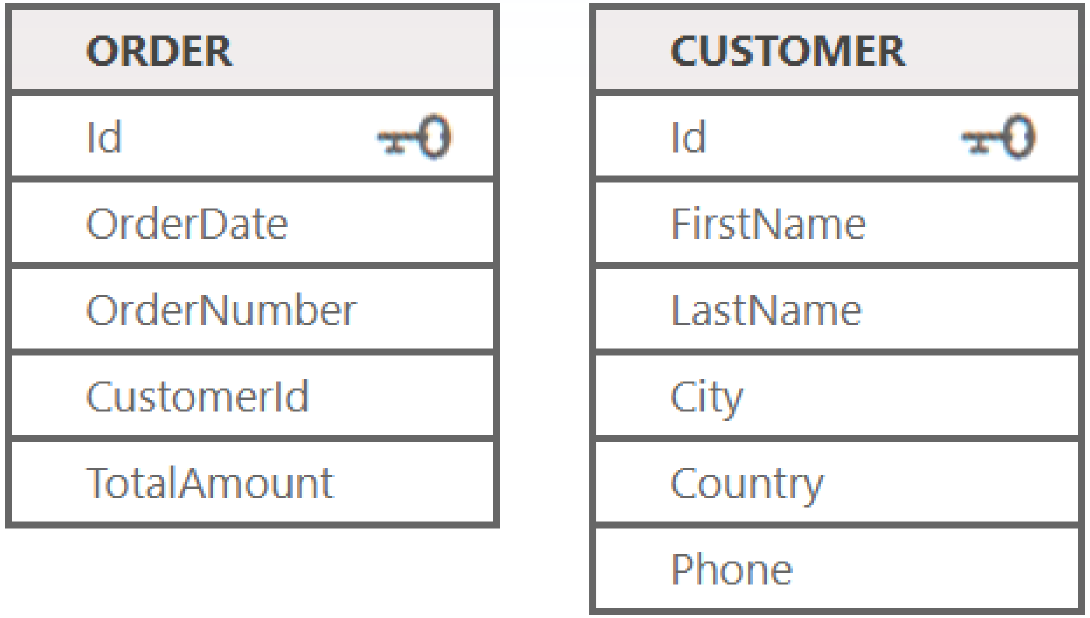


B	
Num	Cube
2	8
3	18
5	75

A ⟖ B

Num	Square	Cube
2	4	8
3	9	18
5	-	75

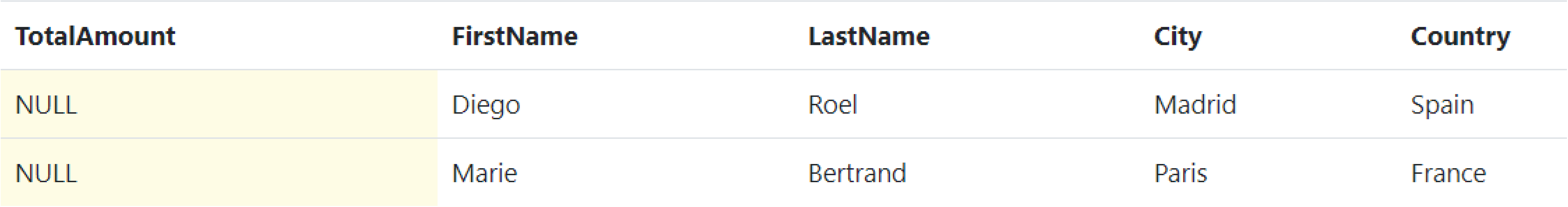
RIGHT JOIN - EXAMPLE



ORDER
Id 🔑
OrderDate
OrderNumber
CustomerId
TotalAmount

CUSTOMER
Id 🔑
FirstName
LastName
City
Country
Phone

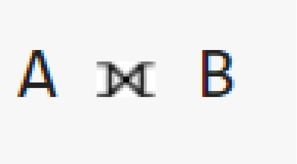
List customers that have not placed orders



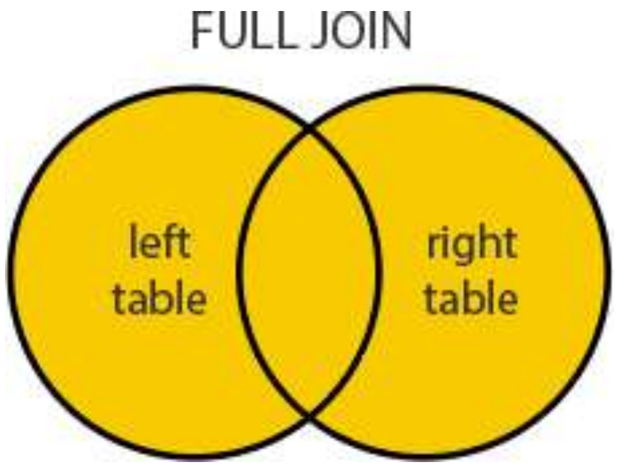
TotalAmount	FirstName	LastName	City	Country
NULL	Diego	Roel	Madrid	Spain
NULL	Marie	Bertrand	Paris	France

FULL JOIN

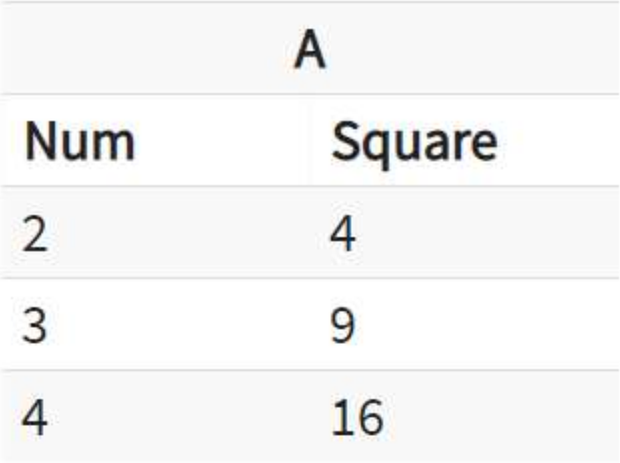
- all tuples from both relations are included in the result, irrespective of the matching condition.
- Syntax:



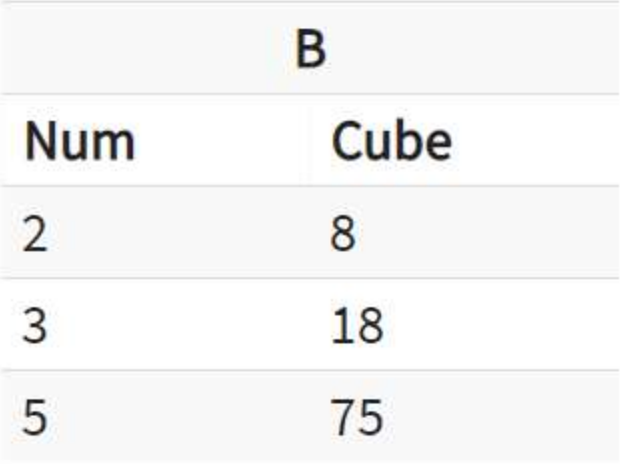
A ⟗ B



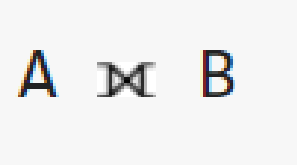
FULL JOIN - EXAMPLE



A	
Num	Square
2	4
3	9
4	16



B	
Num	Cube
2	8
3	18
5	75

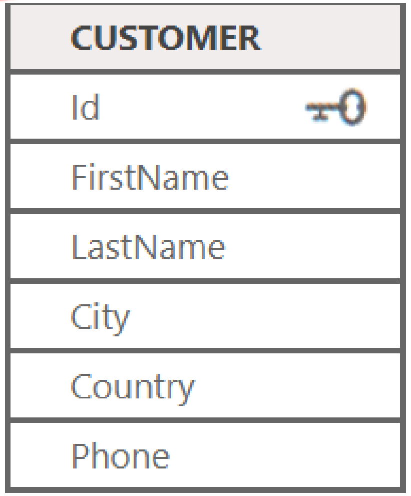


A ⟗ B

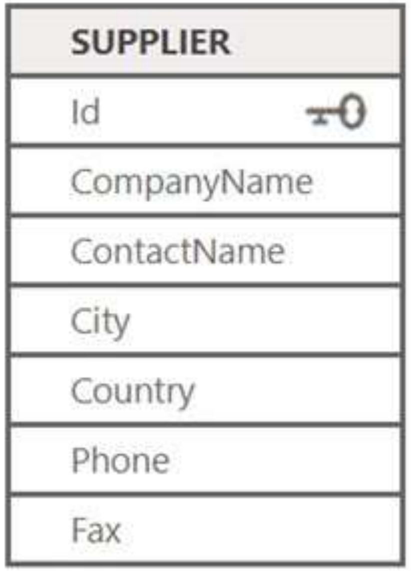
Num	Square	Cube
2	4	8
3	9	18
4	16	-
5	-	75

FULL JOIN - EXAMPLE

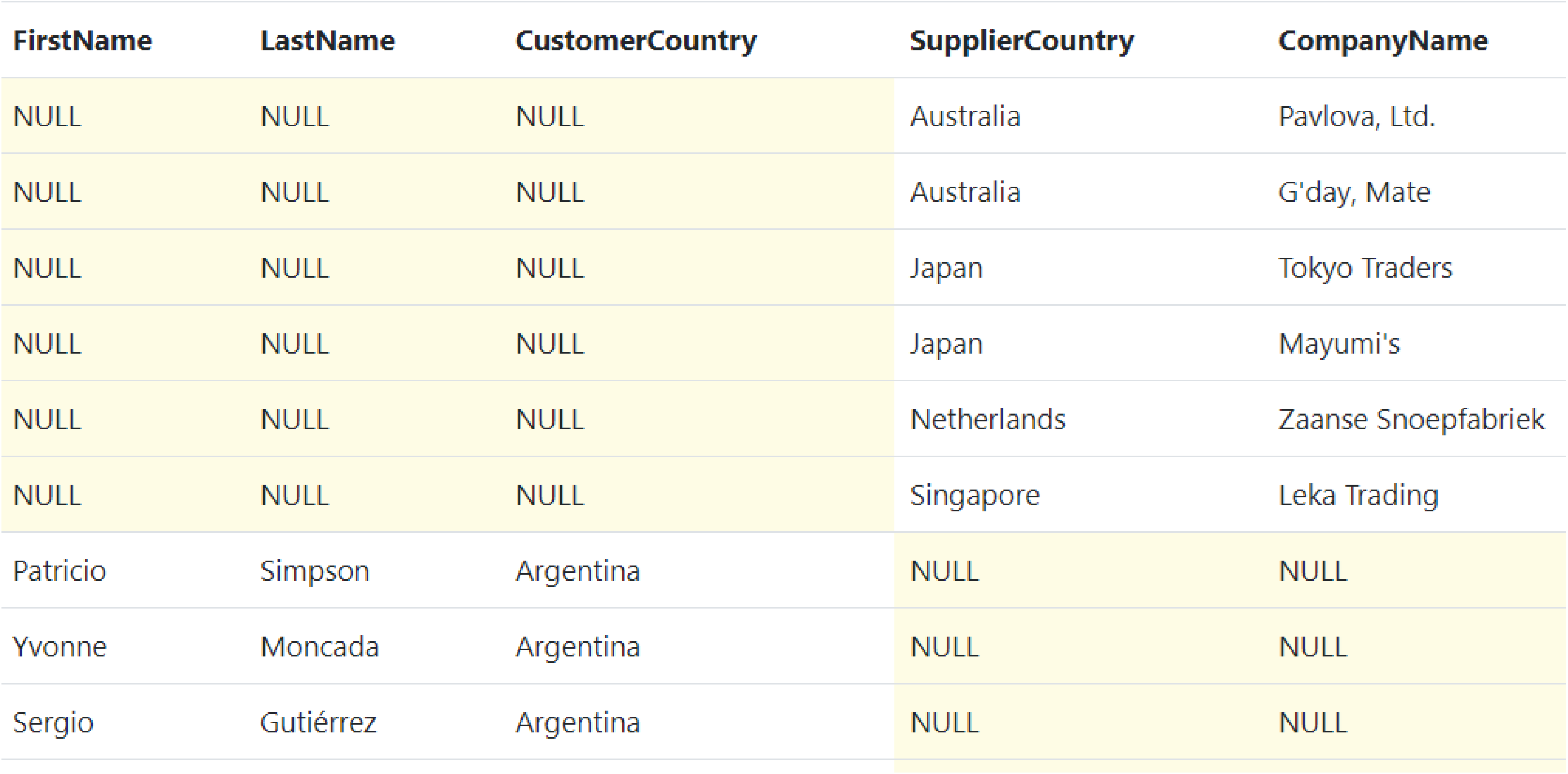
Match all customers and suppliers by country



CUSTOMER
Id 🔑
FirstName
LastName
City
Country
Phone



SUPPLIER
Id 🔑
CompanyName
ContactName
City
Country
Phone
Fax



FirstName	LastName	CustomerCountry	SupplierCountry	CompanyName
NULL	NULL	NULL	Australia	Pavlova, Ltd.
NULL	NULL	NULL	Australia	G'day, Mate
NULL	NULL	NULL	Japan	Tokyo Traders
NULL	NULL	NULL	Japan	Mayumi's
NULL	NULL	NULL	Netherlands	Zaanse Snoepfabriek
NULL	NULL	NULL	Singapore	Leka Trading
Patricio	Simpson	Argentina	NULL	NULL
Yvonne	Moncada	Argentina	NULL	NULL
Sergio	Gutiérrez	Argentina	NULL	NULL

Semi Join

- Semi-Join matches the rows of two relations and then show the matching rows of the relation whose name is mentioned to the left side of ⋉ Semi Join operator.

Relation Teacher

ID	Rank	Salary
101	Assistant Professor	80,000
102	Associate Professor	90,000
103	Lecturer	70,000

Relation Student

ID	RollNo	Marks
103	2017 – 01	80
104	2017 – 02	90
105	2017 – 03	70

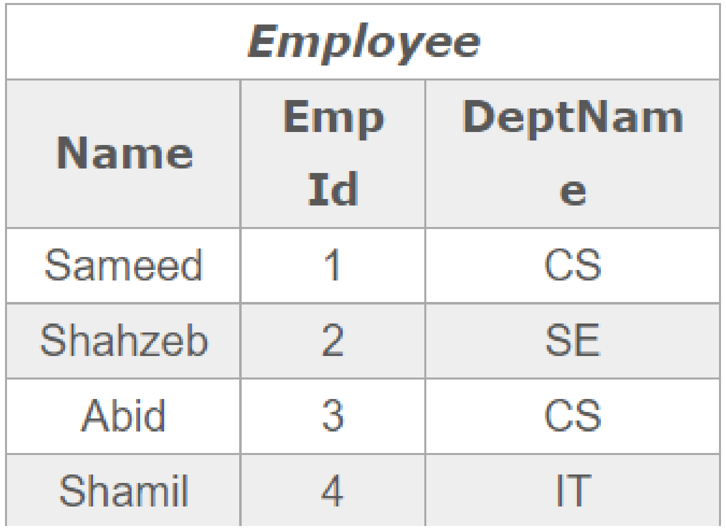
ID	RollNo	Marks
103	2017 – 01	80

Student ⋉ Teacher OR Student Semi Join Teacher

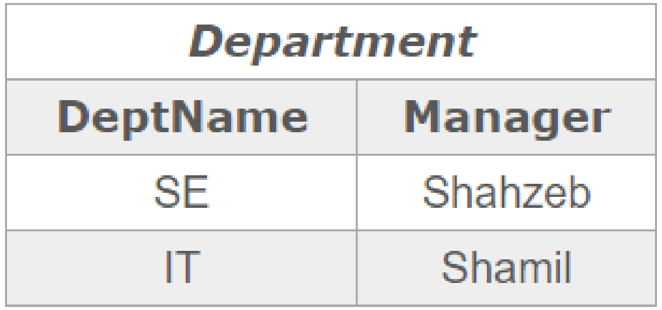
ID	Rank	Salary
103	Lecturer	70,000

Teacher ⋉ Student OR Teacher SEMI-JOIN Student

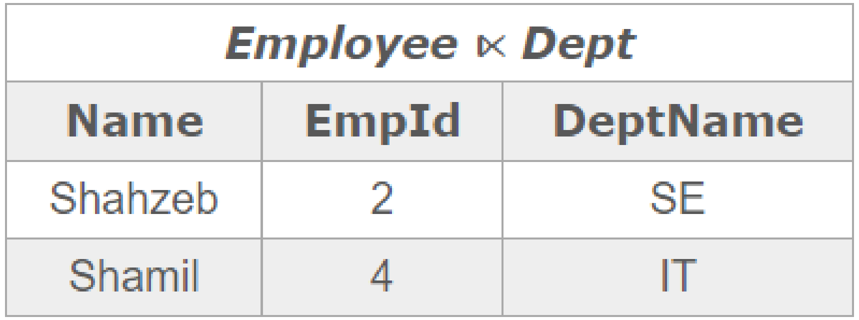
Semi Join - Examples



Employee		
Name	**Emp Id**	**DeptName**
Sameed	1	CS
Shahzeb	2	SE
Abid	3	CS
Shamil	4	IT



Department	
DeptName	**Manager**
SE	Shahzeb
IT	Shamil



Employee $\ltimes$ *Dept*		
Name	**EmpId**	**DeptName**
Shahzeb	2	SE
Shamil	4	IT

Assignment Operator (←)

- We can write the operations as a single relational algebra expression by nesting the operations, or we can apply one operation at a time and create intermediate result relations.
- In the latter case, we must name the relations that hold the intermediate results.
- Here, we use the assignment operator (←).

Syntax:

Relational Variable ← Expression (or) R ← E.

R is a relation.

E is the Expression whose result we wish to assign to the relation variable R.

Example:

R1 ← πname(Customer)

R2 ← πname(Employee)

R = R1 – R2

Division Operator (÷)

- Division operation is denoted by ÷ sign.
- Let R (R-Schema) and S(S-Schema) be relations and any attribute of S – Schema is also in R – Schema.
- The relation R / S is a relation on schema R-Schema – S-Schema i.e. on the schema containing all the attributes of Schema R that are not in Schema S.

Syntax:

P = R ÷ S

Where,

P is result we get after applying division operator,

R and **S** stands for relation (name of the tables) on which division operation is applied.

Division Operator (÷) Examples

A tuple t is in r ÷ s if and only if both the conditions hold.

- T is in πR – S (r)
- For every tuple ts in S, there is a tuple tr in R satisfying both of the following:
  - tr[s] = ts[s]
  - tr[R-S] = t

Relation P

A	B
A1	B1
A1	B2
A2	B1
A3	B1
A4	B2
A5	B1
A5	B2

Relation Q

B
B1
B2

R = P ÷ Q is

Relation R

A
A1
A5

Division Operator (÷) Examples

Relation P

A	B
A1	B1
A1	B2
A2	B1
A3	B1
A4	B2
A5	B1
A5	B2

Relation Q

B
B1

R = P ÷ Q is

Relation R

A
A1
A2
A3
A5

Relation Q

B

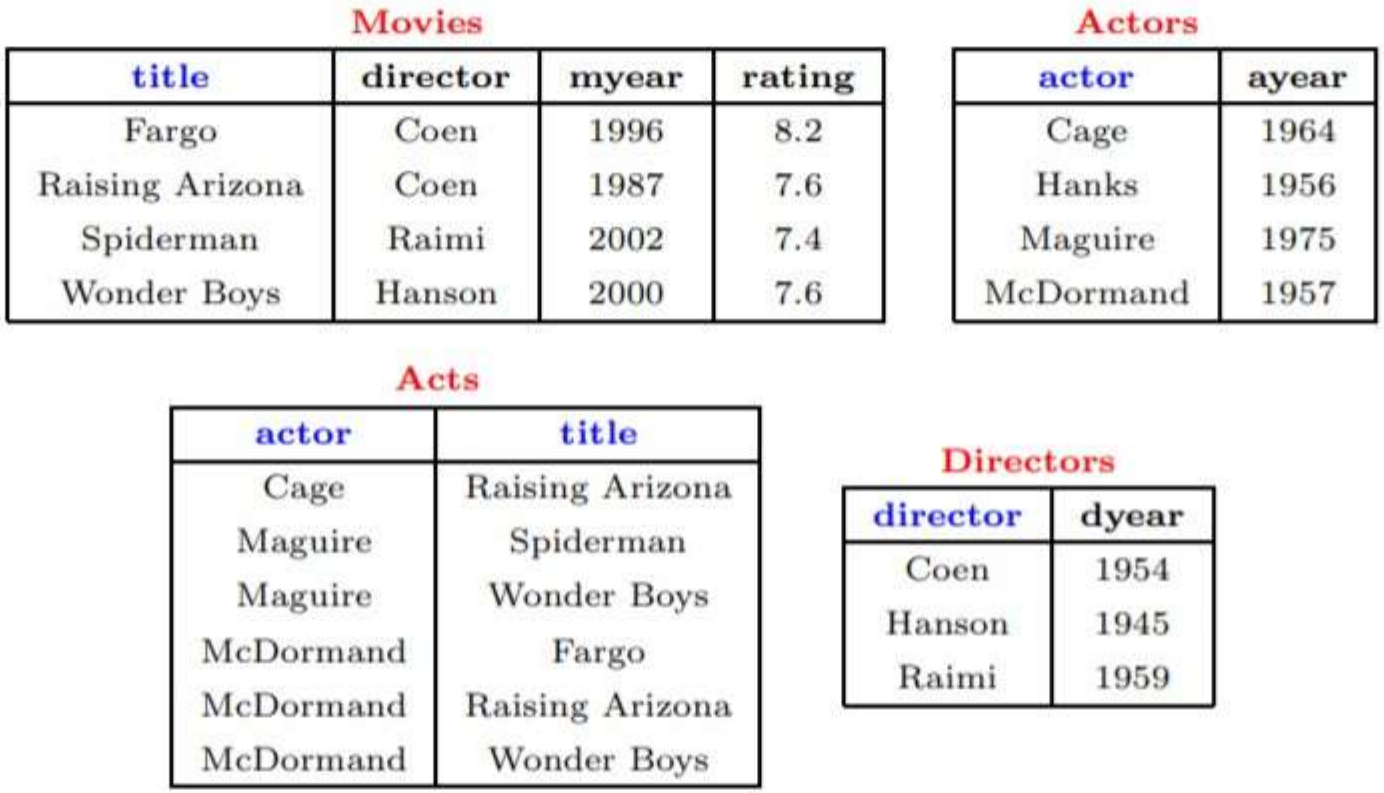
R = P ÷ Q is

Relation R

A
A1
A2
A3
A4
A5

EXAMPLES

Example Database



Movies

title	director	myear	rating
Fargo	Coen	1996	8.2
Raising Arizona	Coen	1987	7.6
Spiderman	Raimi	2002	7.4
Wonder Boys	Hanson	2000	7.6

Actors

actor	ayear
Cage	1964
Hanks	1956
Maguire	1975
McDormand	1957

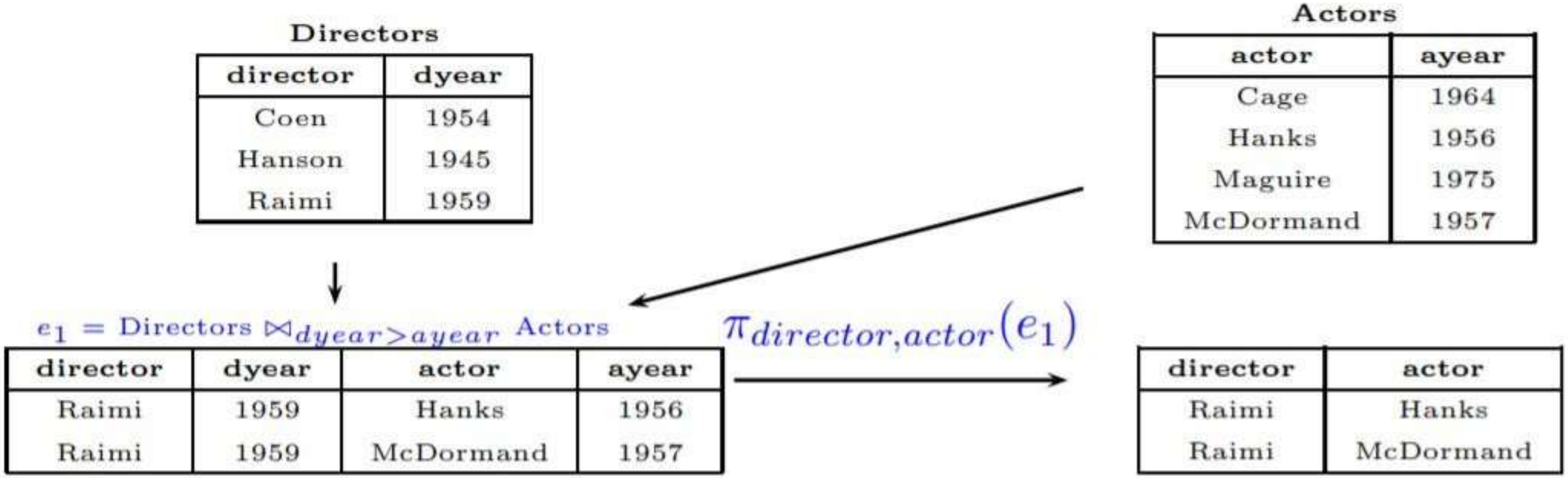
Acts

actor	title
Cage	Raising Arizona
Maguire	Spiderman
Maguire	Wonder Boys
McDormand	Fargo
McDormand	Raising Arizona
McDormand	Wonder Boys

Directors

director	dyear
Coen	1954
Hanson	1945
Raimi	1959

Example: Find (director,actor) pairs where the director is younger than the actor



Directors

director	dyear
Coen	1954
Hanson	1945
Raimi	1959

Actors

actor	ayear
Cage	1964
Hanks	1956
Maguire	1975
McDormand	1957

$e_1 = \text{Directors} \bowtie_{dyear>ayear} \text{Actors}$

director	dyear	actor	ayear
Raimi	1959	Hanks	1956
Raimi	1959	McDormand	1957

$\pi_{director,actor}(e_1)$

director	actor
Raimi	Hanks
Raimi	McDormand

Example: Find actors who have acted in some Coen's movie

$e_1 = \textbf{Acts} \bowtie_{Acts.title\ =\ Movies.title} \textbf{Movies}$

actor	**title**	**director**	**myear**	**rating**
Cage	Raising Arizona	Coen	1987	7.6
Maguire	Spiderman	Raimi	2002	7.4
Maguire	Wonder Boys	Hanson	2000	7.6
McDormand	Fargo	Coen	1996	8.2
McDormand	Raising Arizona	Coen	1987	7.6
McDormand	Wonder Boys	Hanson	2000	7.6

$$\Big\downarrow\ \pi_{actor}(\sigma_{director='Coen'}((e_1))$$

actor
Cage
McDormand

Sailors (sid, name, rating, age)

sid	name	rating	age
1	Dustin	7	45
2	Rusty	10	35
3	Horatio	5	35
4	Zorba	8	18
5	Julius		25

Boats (bid, name, color)

bid	name	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

Reserves (sid, bid, day)

sid	bid	day
1	101	10/10/12
1	102	10/10/12
1	101	10/7/12
2	102	11/9/12
2	102	7/11/12
3	101	7/11/12
3	102	7/8/12
4	103	19/9/12

List names of sailors who reserved boat 102

$$\pi_{name}\ (\mathit{Sailors} \bowtie (\ \sigma_{bid=102}\ \mathit{Reserves}))$$

Sailors (<u>sid</u>, name, rating, age)

sid	name	rating	age
1	Dustin	7	45
2	Rusty	10	35
3	Horatio	5	35
4	Zorba	8	18
5	Julius		25

Boats (<u>bid</u>, name, color)

bid	name	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

Reserves (<u>sid</u>, <u>bid</u>, <u>day</u>)

sid	bid	day
1	101	10/10/12
1	102	10/10/12
1	101	10/7/12
2	102	11/9/12
2	102	7/11/12
3	101	7/11/12
3	102	7/8/12
4	103	19/9/12

List names of sailors who reserved the red Interlake.

$$
\begin{aligned}
&\pi_{Sailors.name}\ (\\
&\qquad Sailors \bowtie (\\
&\qquad\qquad (\ \sigma_{name=Interlake\ and\ color=red}\ Boats) \bowtie Reserves))
\end{aligned}
$$

Sailors (sid, name, rating, age)

sid	name	rating	age
1	Dustin	7	45
2	Rusty	10	35
3	Horatio	5	35
4	Zorba	8	18
5	Julius		25

Boats (bid, name, color)

bid	name	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

Reserves (sid, bid, day)

sid	bid	day
1	101	10/10/12
1	102	10/10/12
1	101	10/7/12
2	102	11/9/12
2	102	7/11/12
3	101	7/11/12
3	102	7/8/12
4	103	19/9/12

List names of boats that were reserved by Horatio.

$$\pi_{Boats.name}\big((\sigma_{Sailors.name=Horatio}\ Sailors) \bowtie (Boats \bowtie Reserves)\big)$$

Sailors (sid, name, rating, age)

sid	name	rating	age
1	Dustin	7	45
2	Rusty	10	35
3	Horatio	5	35
4	Zorba	8	18
5	Julius		25

Boats (bid, name, color)

bid	name	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

Reserves (sid, bid, day)

sid	bid	day
1	101	10/10/12
1	102	10/10/12
1	101	10/7/12
2	102	11/9/12
2	102	7/11/12
3	101	7/11/12
3	102	7/8/12
4	103	19/9/12

List days on which some sailor with rating higher than 7 was at sea

$$\pi_{day}\,((\sigma_{rating>7}\ Sailors) \bowtie Reserves)$$

JOIN VS CARTESIAN PRODUCT

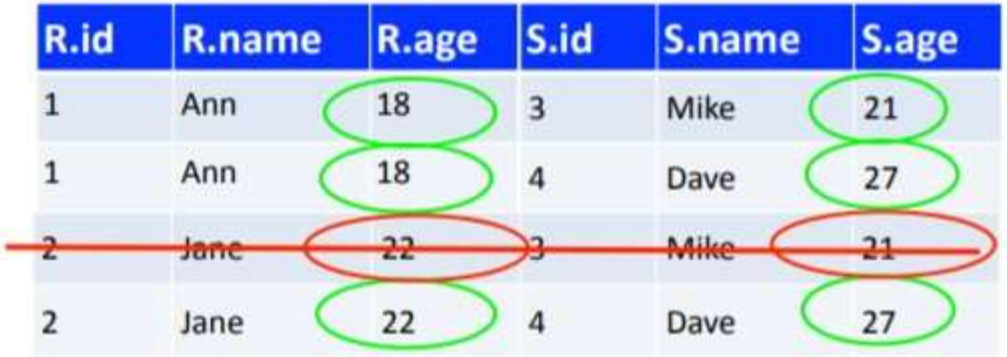
Conceptually, to compute $R \bowtie_C S$

1. compute a Cartesian product R × S

2. then compute a selection σ_C (R × S) using the join condition

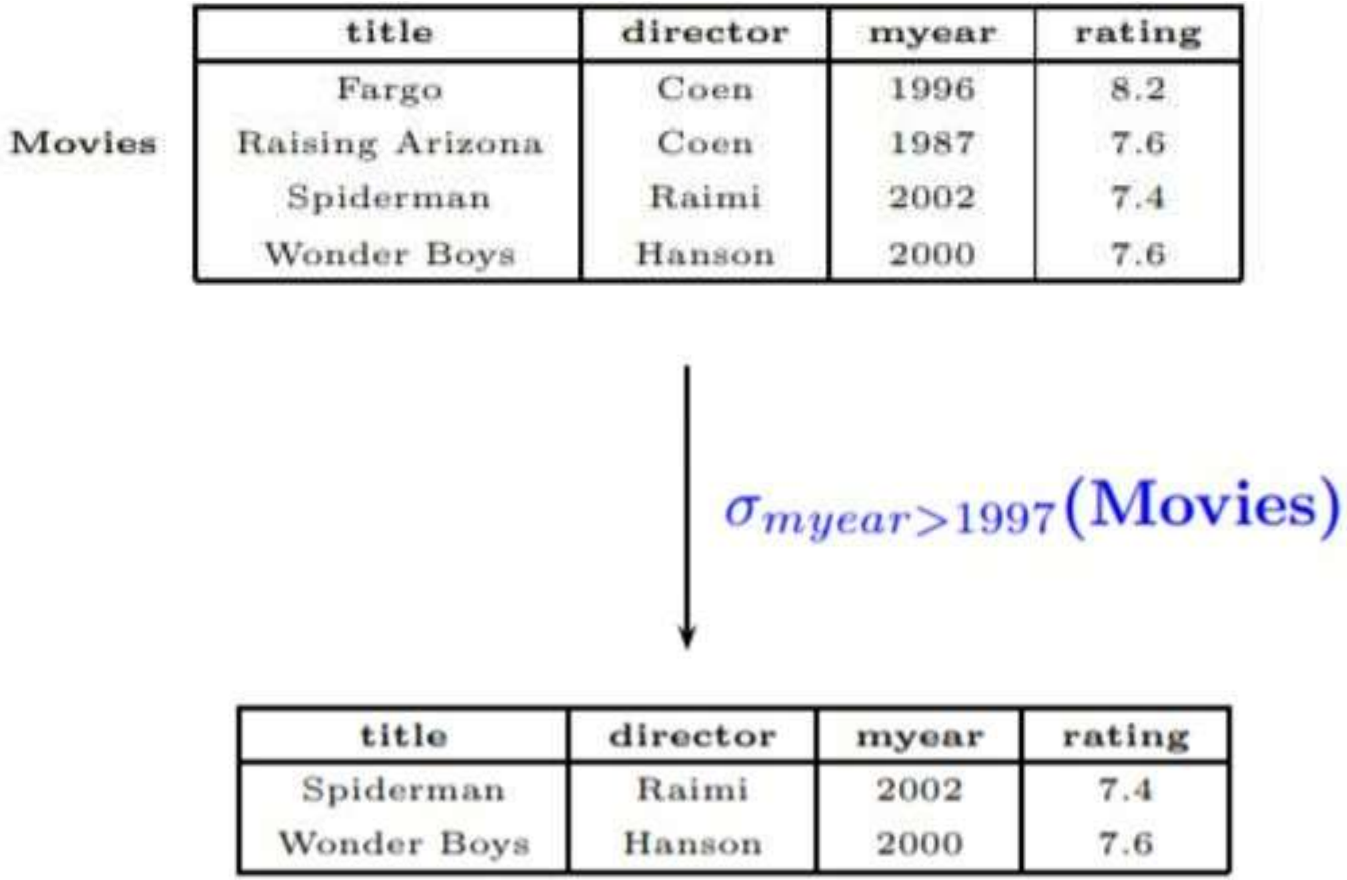
$$\boxed{R \bowtie_C S = \sigma_C (R \times S)}$$

$$R \bowtie_{R.age < S.age} S = \sigma_{R.age < S.age} (R \times S)$$



R.id	R.name	R.age	S.id	S.name	S.age
1	Ann	18	3	Mike	21
1	Ann	18	4	Dave	27
~~2~~	~~Jane~~	~~22~~	~~3~~	~~Mike~~	~~21~~
2	Jane	22	4	Dave	27

Find movies made after 1997



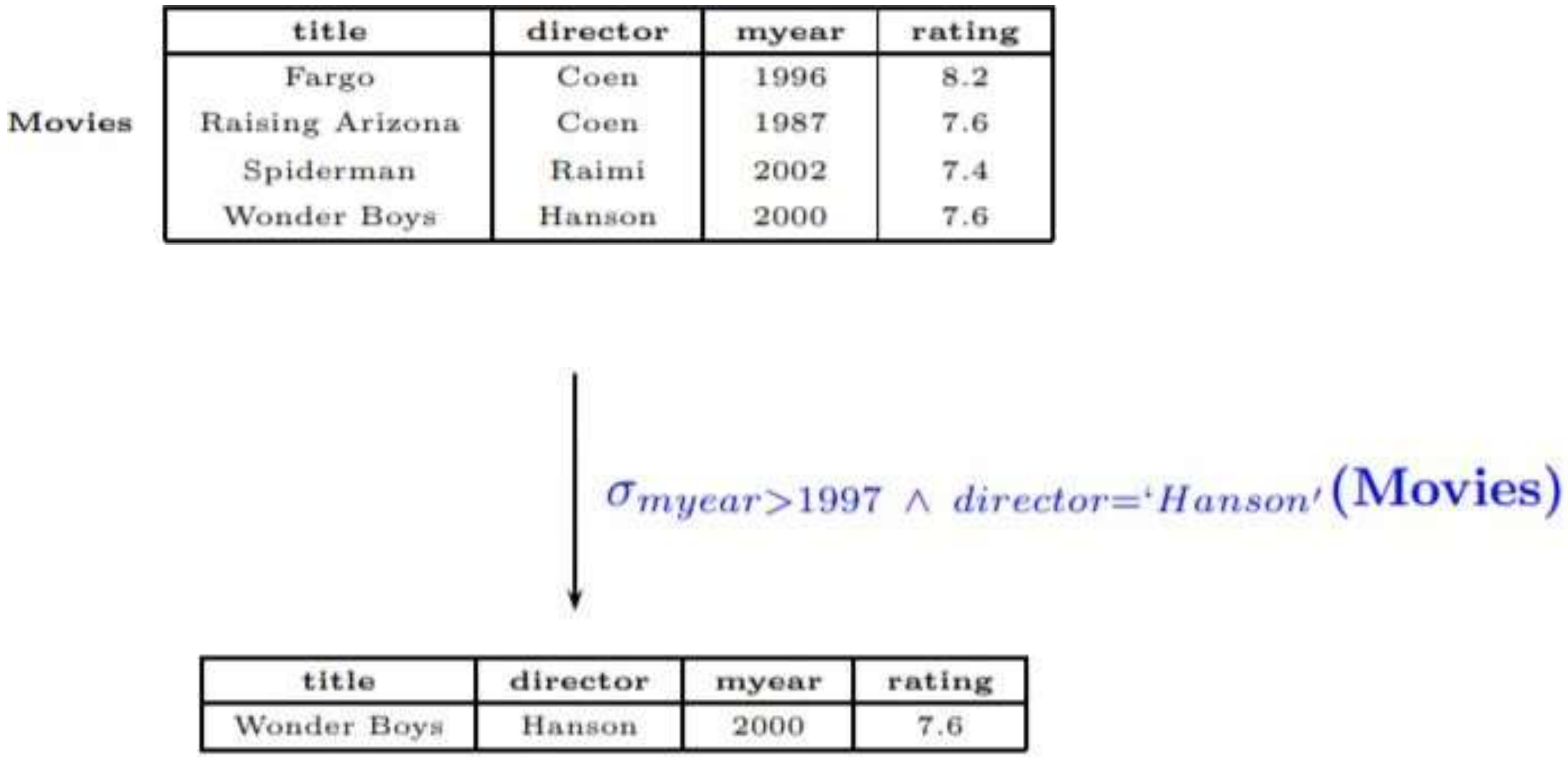
Movies

title	director	myear	rating
Fargo	Coen	1996	8.2
Raising Arizona	Coen	1987	7.6
Spiderman	Raimi	2002	7.4
Wonder Boys	Hanson	2000	7.6

$\sigma_{myear>1997}(\mathbf{Movies})$

title	director	myear	rating
Spiderman	Raimi	2002	7.4
Wonder Boys	Hanson	2000	7.6

Find movies made by Hanson after 1997



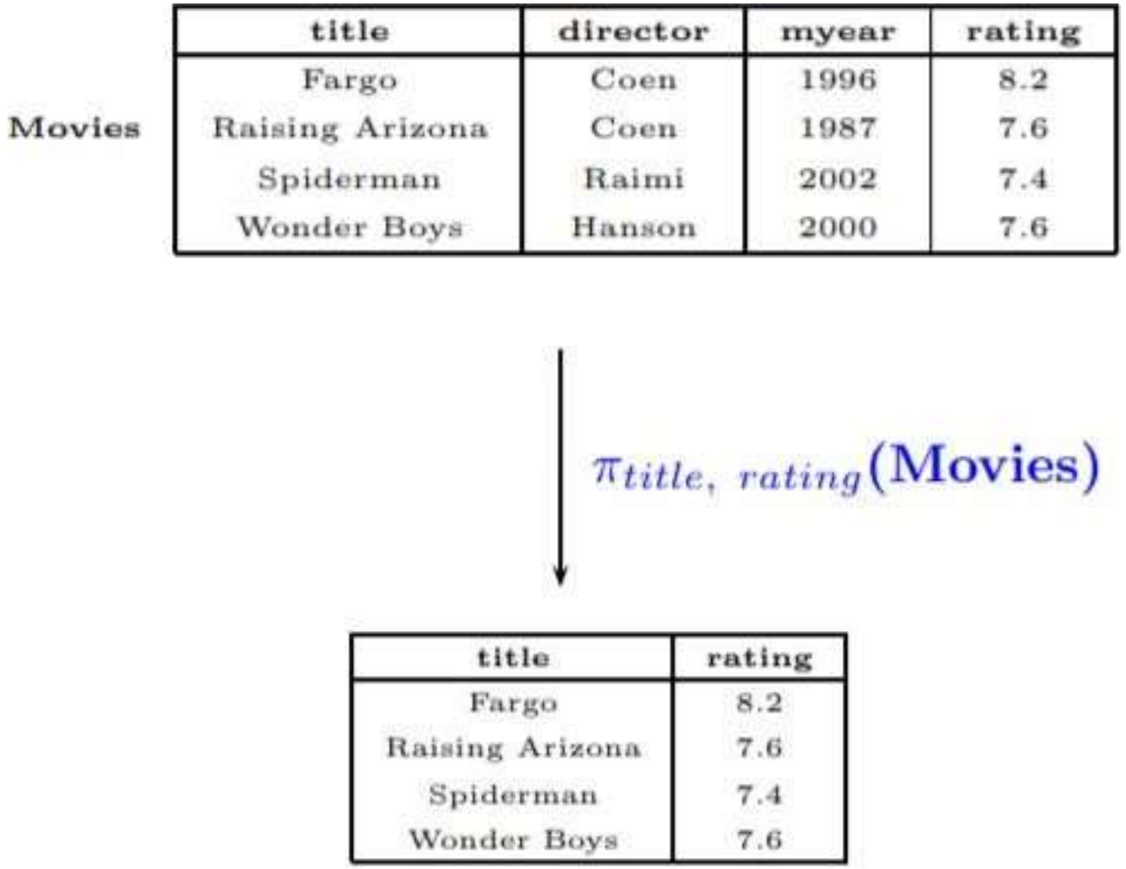
Movies

title	director	myear	rating
Fargo	Coen	1996	8.2
Raising Arizona	Coen	1987	7.6
Spiderman	Raimi	2002	7.4
Wonder Boys	Hanson	2000	7.6

$\sigma_{myear>1997\ \wedge\ director='Hanson'}(\text{Movies})$

title	director	myear	rating
Wonder Boys	Hanson	2000	7.6

Find all movies and their ratings



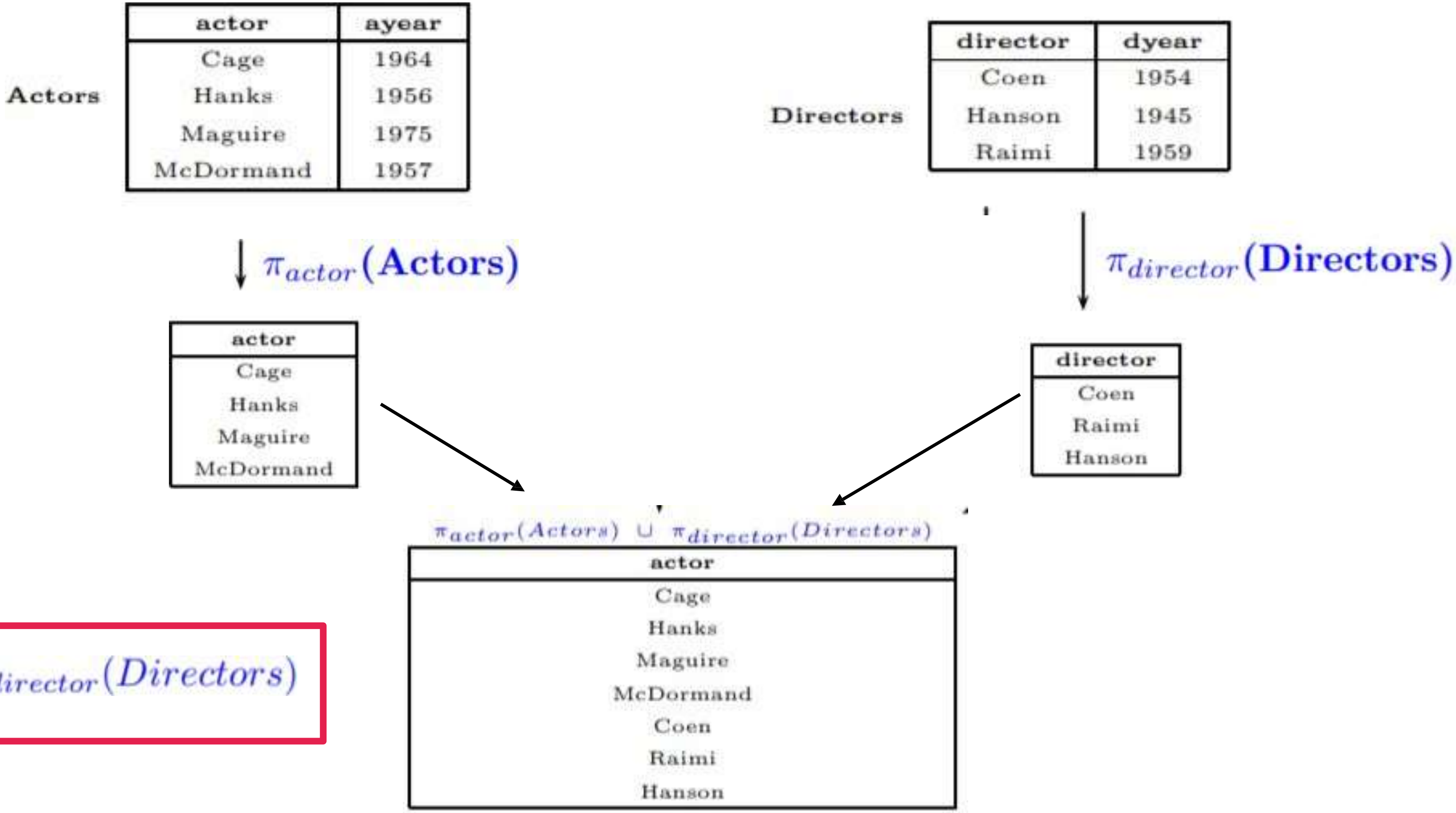
Movies

title	director	myear	rating
Fargo	Coen	1996	8.2
Raising Arizona	Coen	1987	7.6
Spiderman	Raimi	2002	7.4
Wonder Boys	Hanson	2000	7.6

$$\downarrow \; \pi_{title,\ rating}(\textbf{Movies})$$

title	rating
Fargo	8.2
Raising Arizona	7.6
Spiderman	7.4
Wonder Boys	7.6

Find all actors & directors



Actors

actor	ayear
Cage	1964
Hanks	1956
Maguire	1975
McDormand	1957

Directors

director	dyear
Coen	1954
Hanson	1945
Raimi	1959

$\pi_{actor}(\text{Actors})$

actor
Cage
Hanks
Maguire
McDormand

$\pi_{director}(\text{Directors})$

director
Coen
Raimi
Hanson

$\pi_{actor}(Actors) \cup \pi_{director}(Directors)$

actor
Cage
Hanks
Maguire
McDormand
Coen
Raimi
Hanson

$$\pi_{actor}(Actors) \;\cup\; \pi_{director}(Directors)$$

Find Coen’s movies with McDormand

Acts

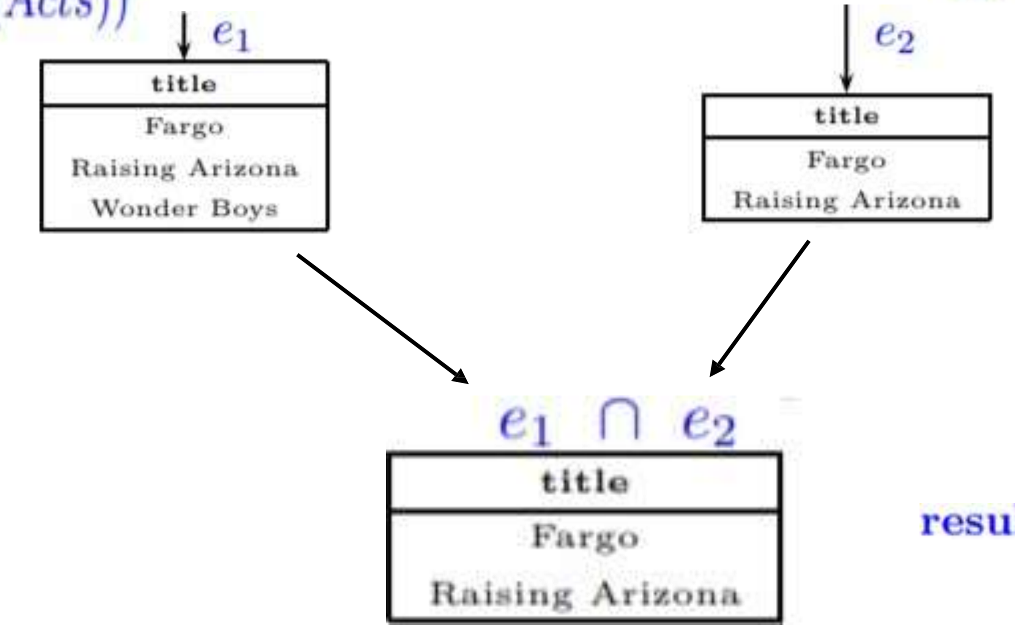
actor	title
Cage	Raising Arizona
Maguire	Spiderman
Maguire	Wonder Boys
McDormand	Fargo
McDormand	Raising Arizona
McDormand	Wonder Boys

Movies

title	director	myear	rating
Fargo	Coen	1996	8.2
Raising Arizona	Coen	1987	7.6
Spiderman	Raimi	2002	7.4
Wonder Boys	Hanson	2000	7.6

$$e_1 = \pi_{title}(\sigma_{actor=\text{'}McDormand\text{'}}(Acts))$$

$$e_2 = \pi_{title}(\sigma_{director=\text{'}Coen\text{'}}(Movies))$$



e_1

title
Fargo
Raising Arizona
Wonder Boys

e_2

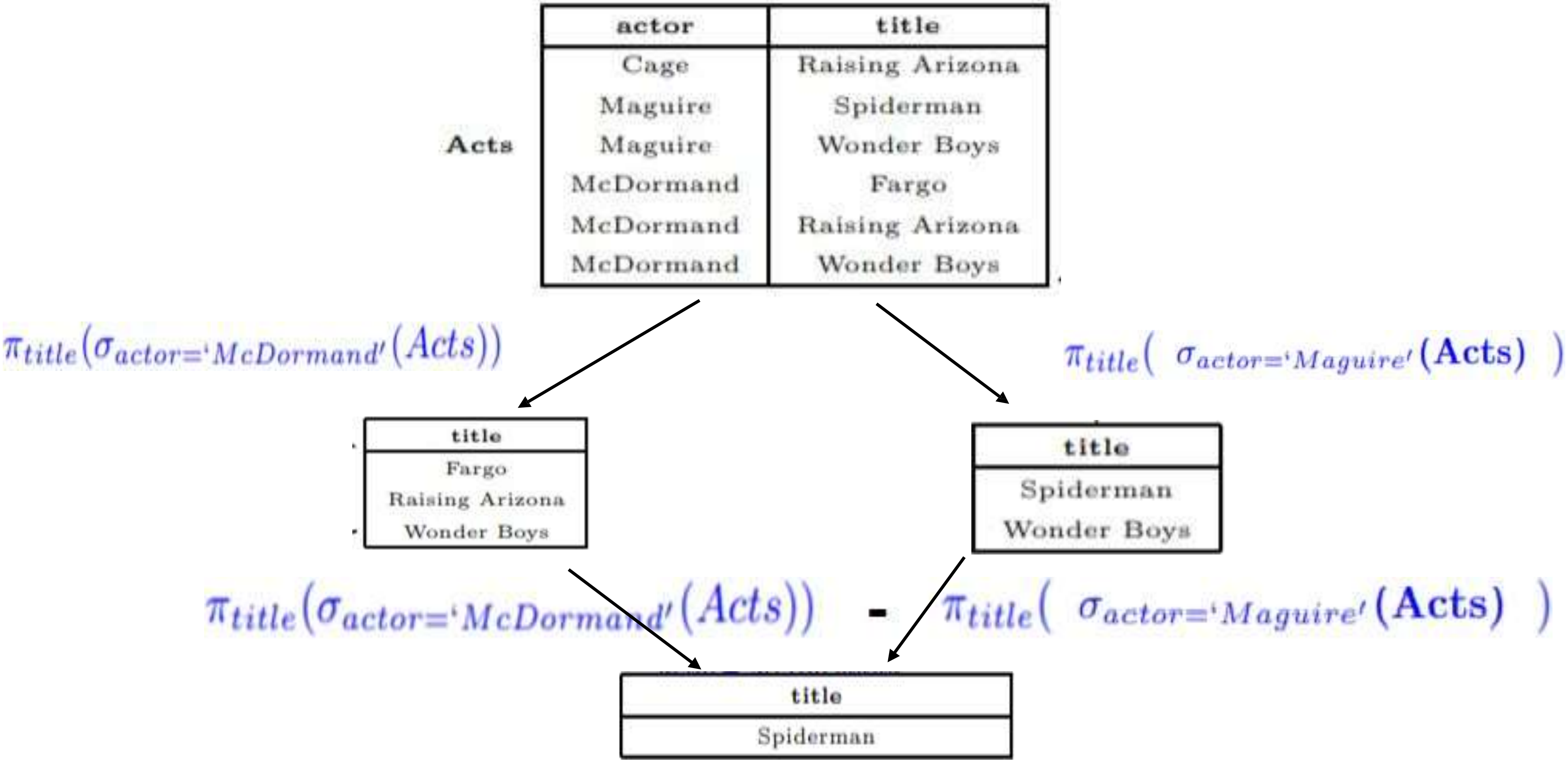
title
Fargo
Raising Arizona

$e_1 \cap e_2$

title
Fargo
Raising Arizona

$$\textbf{result} = e_1 \cap e_2$$

Find movies with Maguire but not McDormand



Acts

actor	title
Cage	Raising Arizona
Maguire	Spiderman
Maguire	Wonder Boys
McDormand	Fargo
McDormand	Raising Arizona
McDormand	Wonder Boys

$\pi_{title}(\sigma_{actor='McDormand'}(Acts))$

$\pi_{title}(\ \sigma_{actor='Maguire'}(\mathbf{Acts})\ \ )$

title
Fargo
Raising Arizona
Wonder Boys

title
Spiderman
Wonder Boys

$\pi_{title}(\sigma_{actor='McDormand'}(Acts))$ - $\pi_{title}(\ \sigma_{actor='Maguire'}(\mathbf{Acts})\ \ )$

title
Spiderman

Find actors who have acted in some Coen’s movies

Movies

title	director	myear	rating
Fargo	Coen	1996	8.2
Raising Arizona	Coen	1987	7.6
Spiderman	Raimi	2002	7.4
Wonder Boys	Hanson	2000	7.6

$$e_1 = \rho_{T(title2)}(\pi_{title}(\sigma_{director=\text{‘}Coen\text{’}}(Movies)))$$

$$\xrightarrow{e_1}$$

T

title2
Fargo
Raising Arizona

Find actors who have acted in some Coen’s movies

$e_2 =$

Acts

actor	title
Cage	Raising Arizona
Maguire	Spiderman
Maguire	Wonder Boys
McDormand	Fargo
McDormand	Raising Arizona
McDormand	Wonder Boys

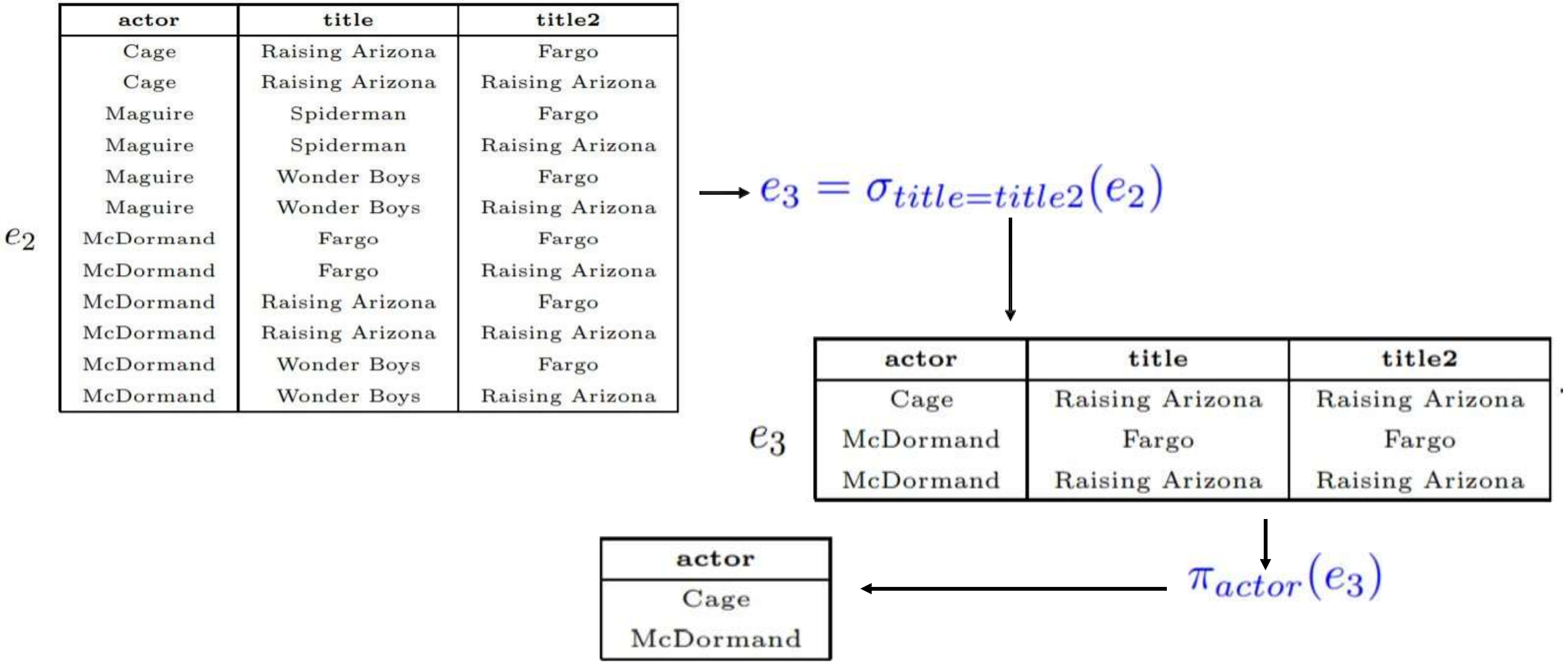
×

T

title2
Fargo
Raising Arizona

actor	title	title2
Cage	Raising Arizona	Fargo
Cage	Raising Arizona	Raising Arizona
Maguire	Spiderman	Fargo
Maguire	Spiderman	Raising Arizona
Maguire	Wonder Boys	Fargo
Maguire	Wonder Boys	Raising Arizona
McDormand	Fargo	Fargo
McDormand	Fargo	Raising Arizona
McDormand	Raising Arizona	Fargo
McDormand	Raising Arizona	Raising Arizona
McDormand	Wonder Boys	Fargo
McDormand	Wonder Boys	Raising Arizona

Find actors who have acted in some Coen's movies



e_2

actor	title	title2
Cage	Raising Arizona	Fargo
Cage	Raising Arizona	Raising Arizona
Maguire	Spiderman	Fargo
Maguire	Spiderman	Raising Arizona
Maguire	Wonder Boys	Fargo
Maguire	Wonder Boys	Raising Arizona
McDormand	Fargo	Fargo
McDormand	Fargo	Raising Arizona
McDormand	Raising Arizona	Fargo
McDormand	Raising Arizona	Raising Arizona
McDormand	Wonder Boys	Fargo
McDormand	Wonder Boys	Raising Arizona

$$e_3 = \sigma_{title=title2}(e_2)$$

e_3

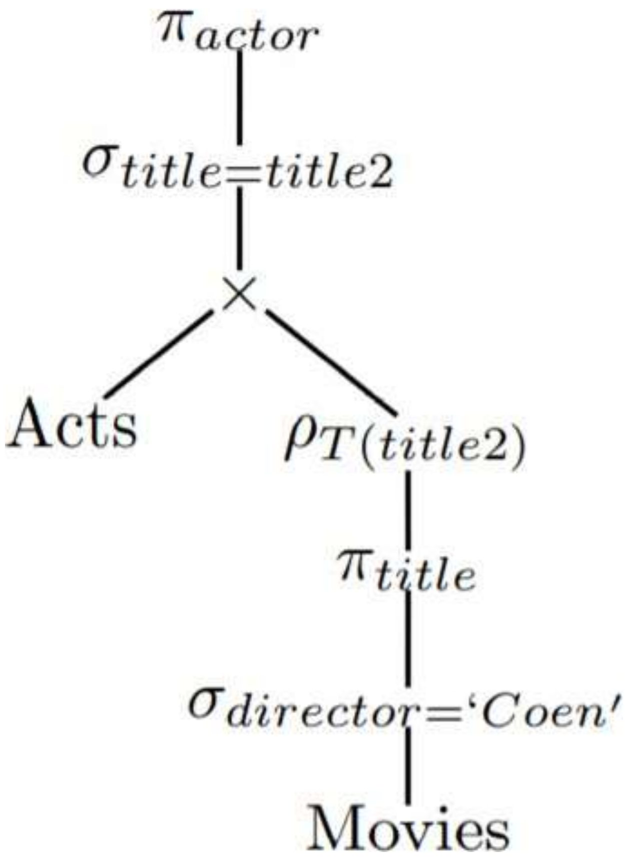
actor	title	title2
Cage	Raising Arizona	Raising Arizona
McDormand	Fargo	Fargo
McDormand	Raising Arizona	Raising Arizona

$$\pi_{actor}(e_3)$$

actor
Cage
McDormand

Find actors who have acted in some Coen’s movies

$$\pi_{actor}\left(\sigma_{title=title2} \left(\text{Acts} \times \rho_{T(title2)}\left(\pi_{title}\left(\sigma_{director=\text{`}Coen'}(\text{Movies}) \right) \right) \right) \right)$$



π_{actor}

$\sigma_{title=title2}$

$\times$

Acts

$\rho_{T(title2)}$

π_{title}

$\sigma_{director=\text{`}Coen'}$

Movies

What are the names of students whose GPA is at least 3.5?

Students

sid	name	gpa
1111	Joe	3.2
2222	Ann	4.0
3333	Mike	3.5

$$\downarrow$$

$$\sigma_{gpa \geq 3.5}(Students)$$

sid	name	gpa
2222	Ann	4.0
3333	Mike	3.5

$$\downarrow$$

$$\pi_{name}(\ \sigma_{gpa \geq 3.5}(Students))$$

name
Ann
Mike

THANK YOU