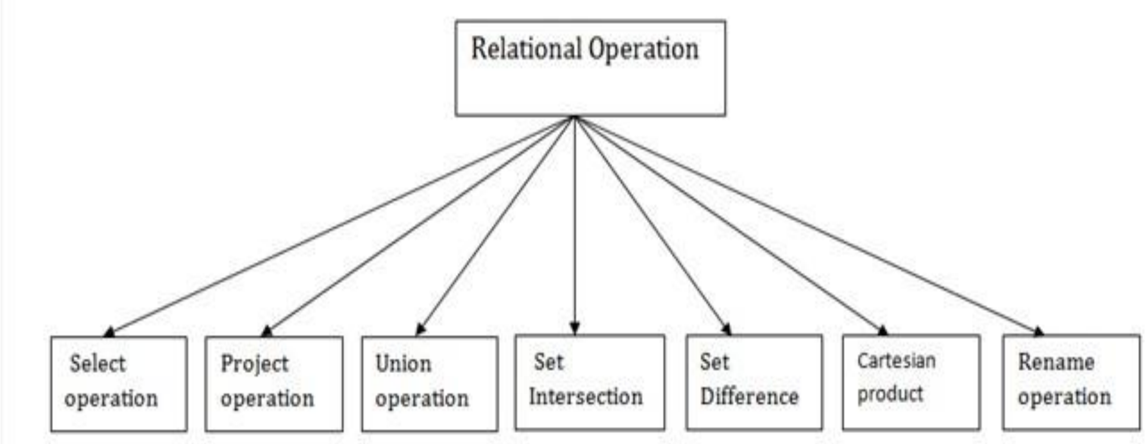
8. RELATIONAL ALGEBRA

Relational algebra is a procedural query language. It gives a step by step process to obtain the result of the query. It uses operators to perform queries. which takes instances of relations as input and yields instances of relations as output. It uses operators to perform queries.



1. Select Operation:

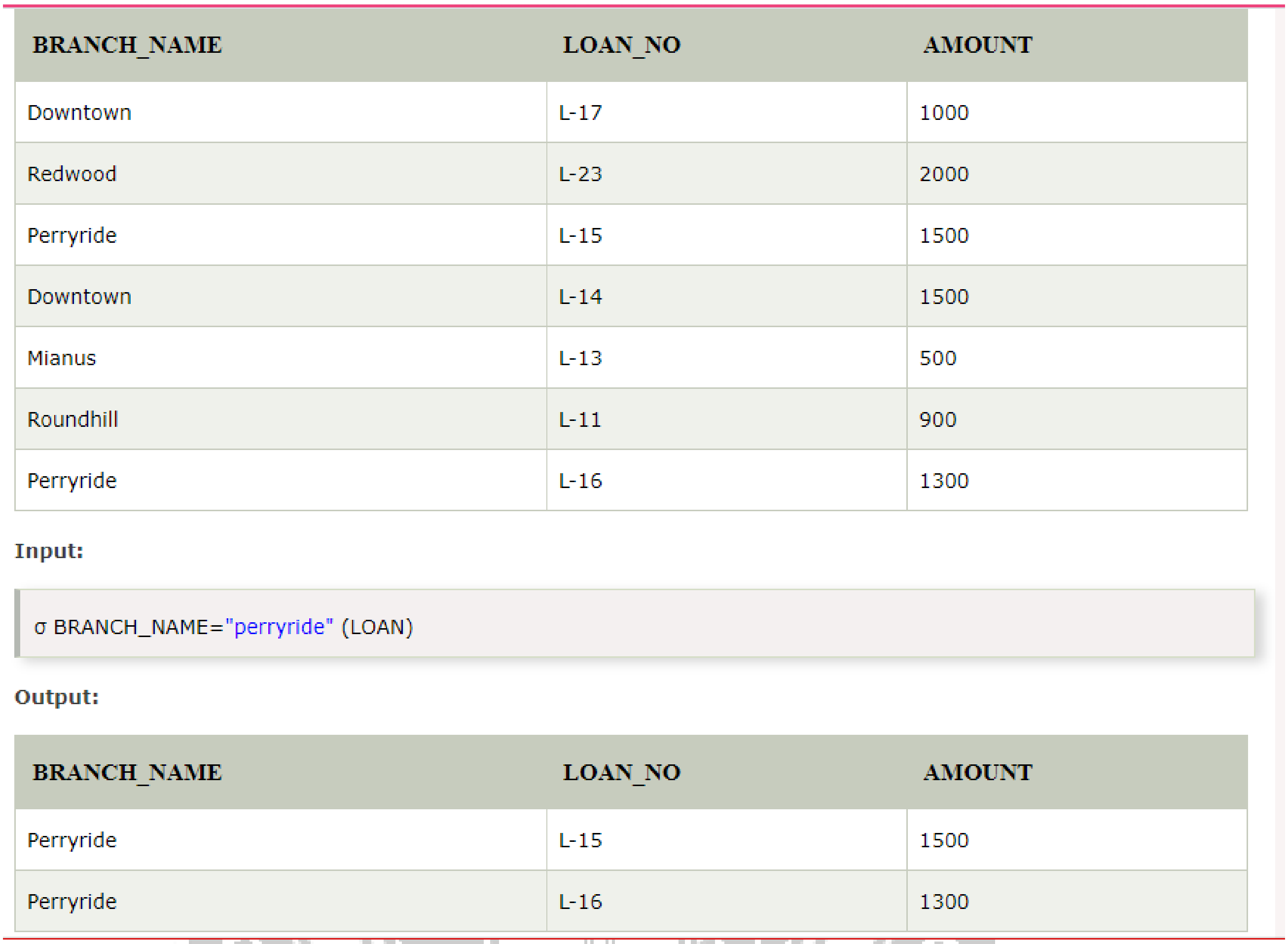
- The select operation selects tuples that satisfy a given predicate.
- It is denoted by sigma (σ).
- Notation: σ p(r)

Where: **σ** is used for selection prediction

r is used for relation

p is used as a propositional logic formula which may use connectors like: AND OR and NOT. These relational can use as relational operators like =, ≠, ≥, <, >, ≤.

Table Name : LOAN



BRANCH_NAME	LOAN_NO	AMOUNT
Downtown	L-17	1000
Redwood	L-23	2000
Perryride	L-15	1500
Downtown	L-14	1500
Mianus	L-13	500
Roundhill	L-11	900
Perryride	L-16	1300

Input:

```
σ BRANCH_NAME="perryride" (LOAN)
```

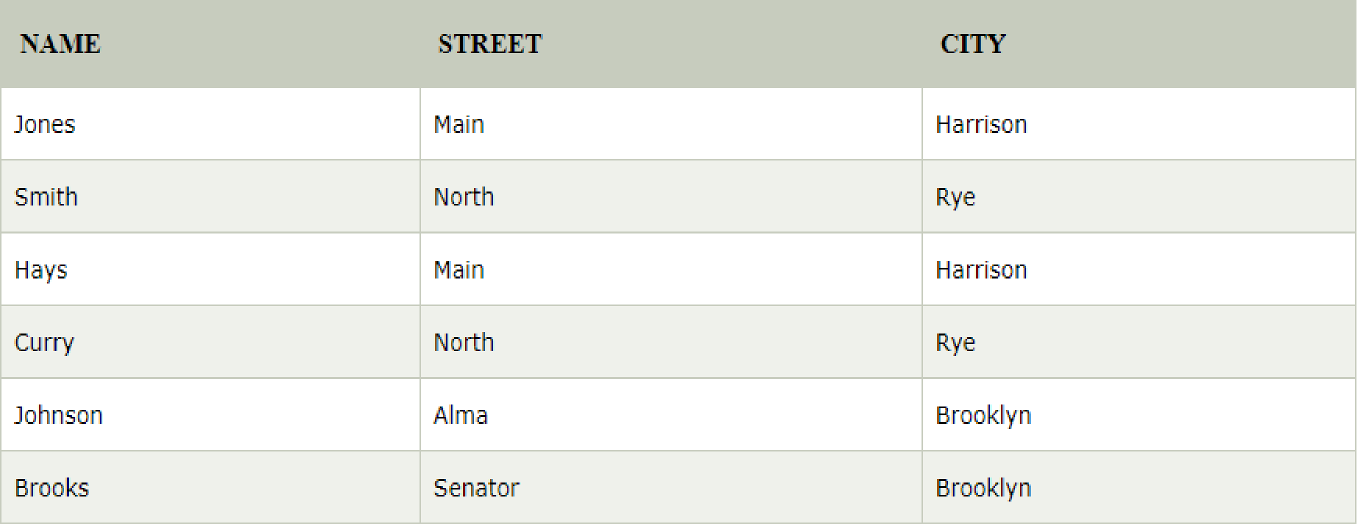
Output:

BRANCH_NAME	LOAN_NO	AMOUNT
Perryride	L-15	1500
Perryride	L-16	1300

2. Project Operation:

- This operation shows the list of those attributes that we wish to appear in the result. Rest of the attributes are eliminated from the table.
- It is denoted by ∏.
- Notation: ∏ A1, A2, An (r)
- **Where A1**, **A2**, **A3** is used as an attribute name of relation **r**.

Example: CUSTOMER RELATION

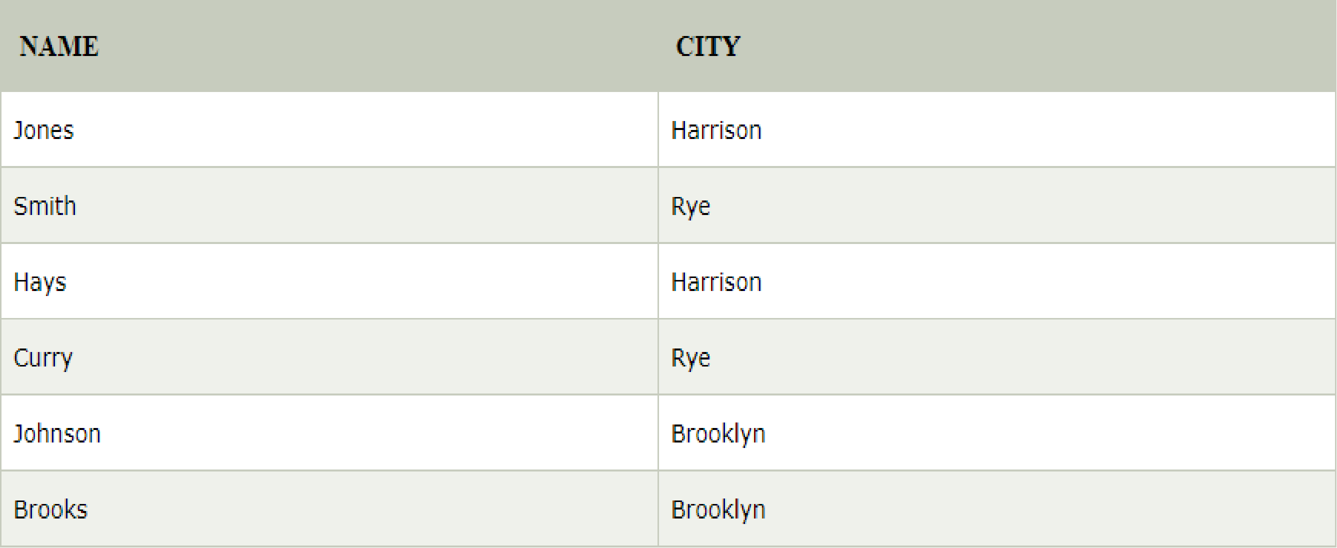


NAME	STREET	CITY
Jones	Main	Harrison
Smith	North	Rye
Hays	Main	Harrison
Curry	North	Rye
Johnson	Alma	Brooklyn
Brooks	Senator	Brooklyn

Input:

```
∏ NAME, CITY (CUSTOMER)
```

Output:

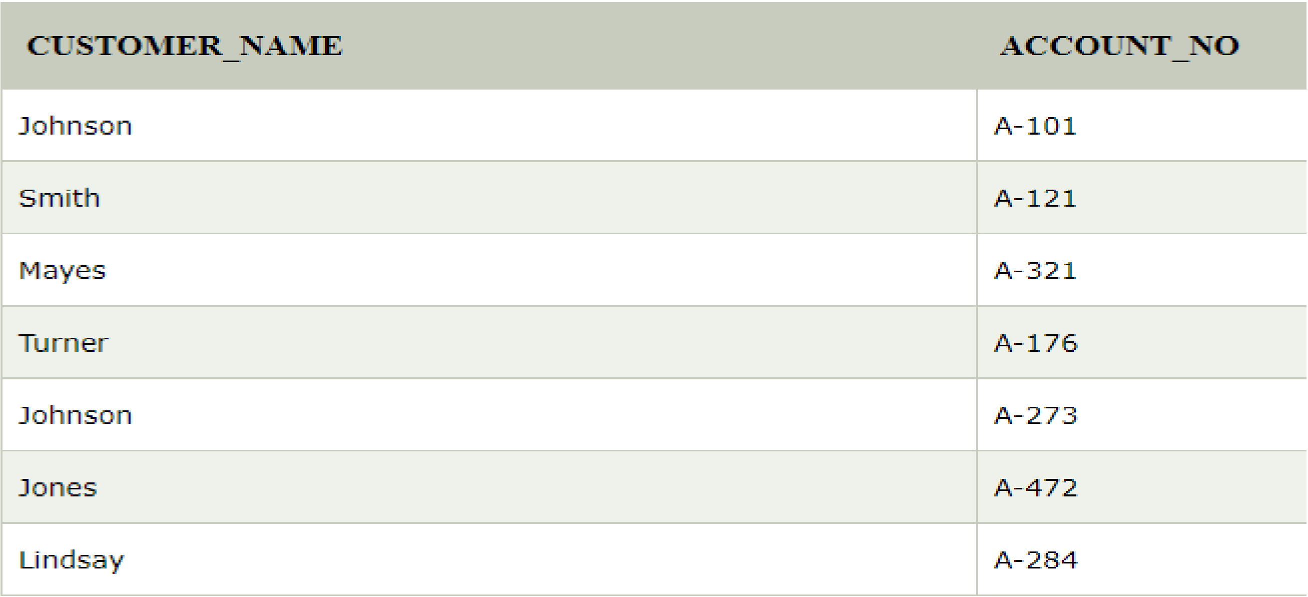


NAME	CITY
Jones	Harrison
Smith	Rye
Hays	Harrison
Curry	Rye
Johnson	Brooklyn
Brooks	Brooklyn

3. Union Operation:

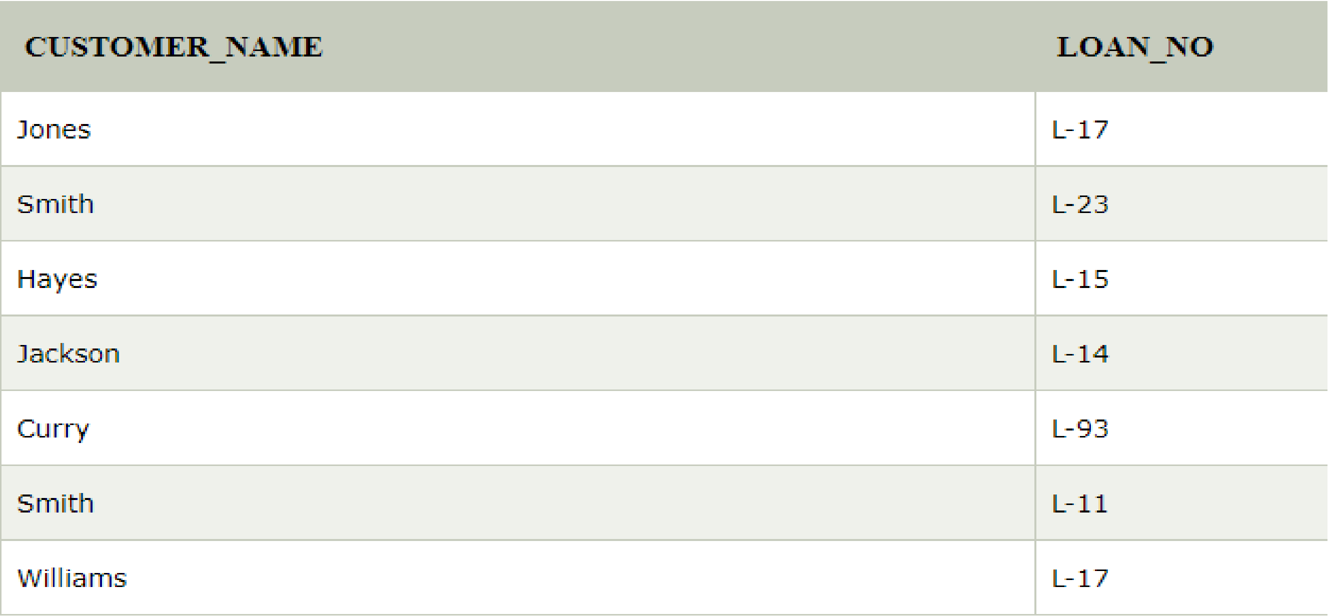
- Suppose there are two tuples R and S. The union operation contains all the tuples that are either in R or S or both in R & S.
- It eliminates the duplicate tuples. It is denoted by U.
- Notation: R U S

DEPOSITOR RELATION



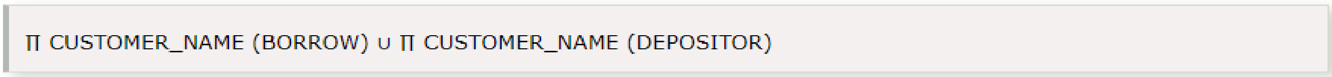
CUSTOMER_NAME	ACCOUNT_NO
Johnson	A-101
Smith	A-121
Mayes	A-321
Turner	A-176
Johnson	A-273
Jones	A-472
Lindsay	A-284

BORROW RELATION



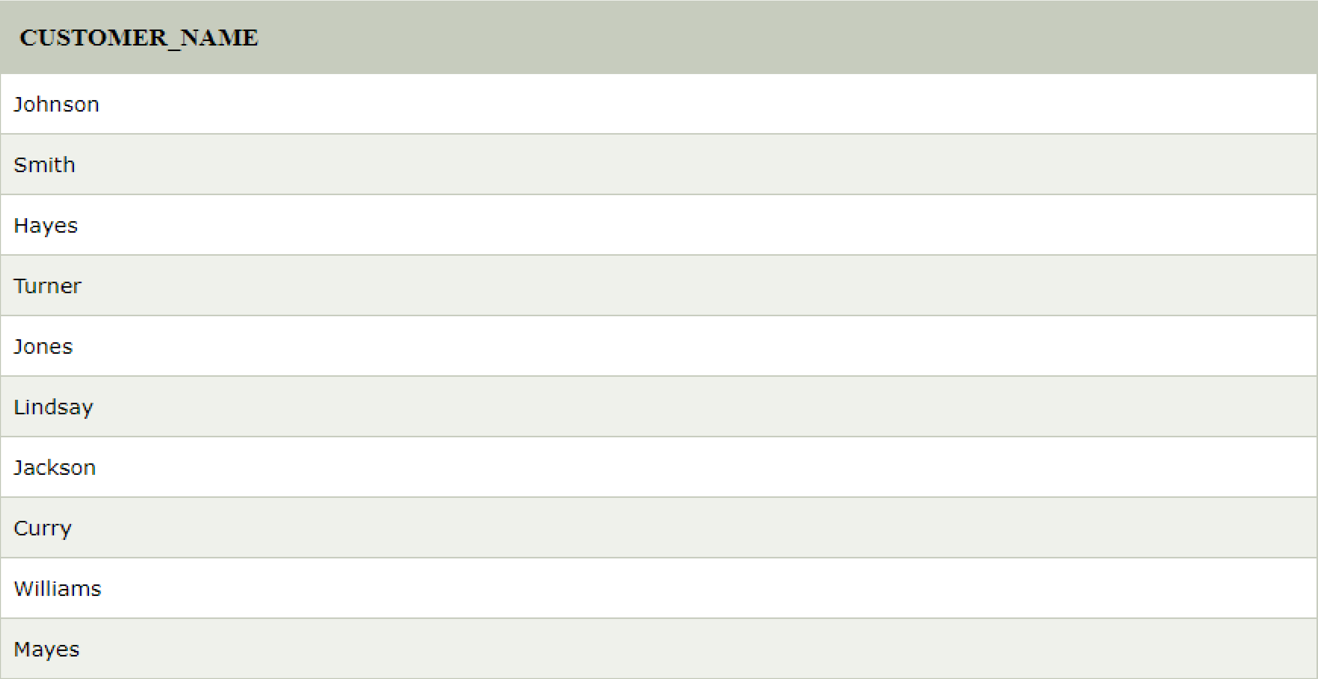
CUSTOMER_NAME	LOAN_NO
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17

Input:



```
∏ CUSTOMER_NAME (BORROW) ∪ ∏ CUSTOMER_NAME (DEPOSITOR)
```

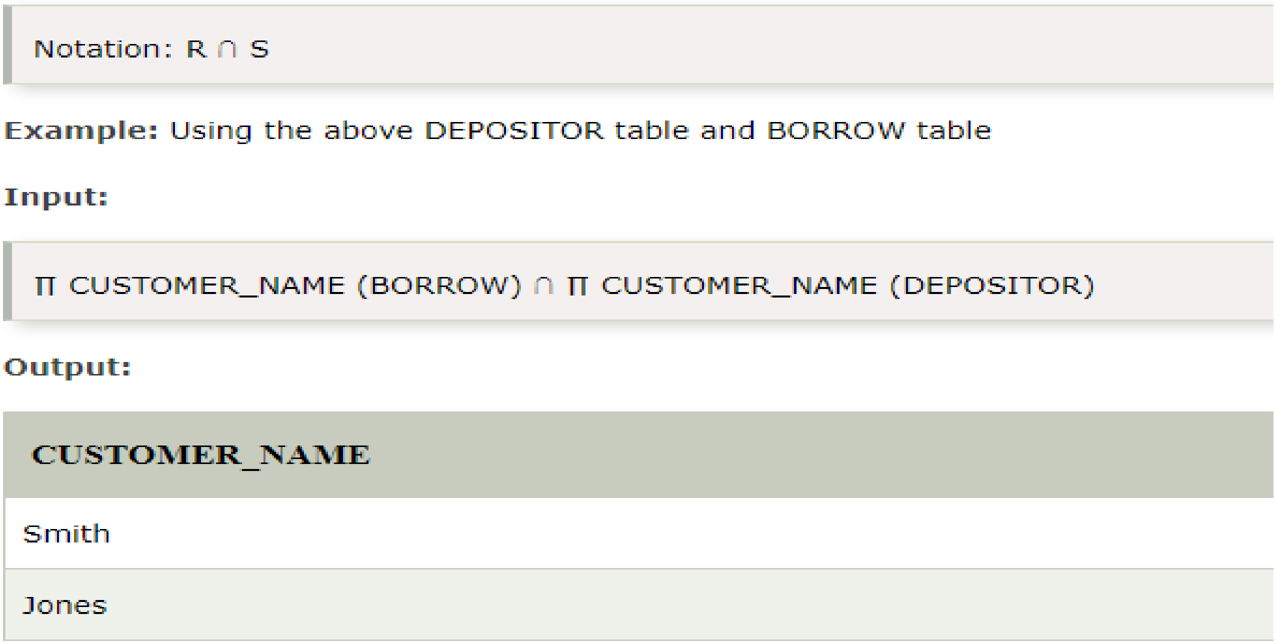
Output:



CUSTOMER_NAME
Johnson
Smith
Hayes
Turner
Jones
Lindsay
Jackson
Curry
Williams
Mayes

4. Set Intersection:

- Suppose there are two tuples R and S. The set intersection operation contains all tuples that are in both R & S.
- It is denoted by intersection ∩.



Notation: R ∩ S

Example: Using the above DEPOSITOR table and BORROW table

Input:

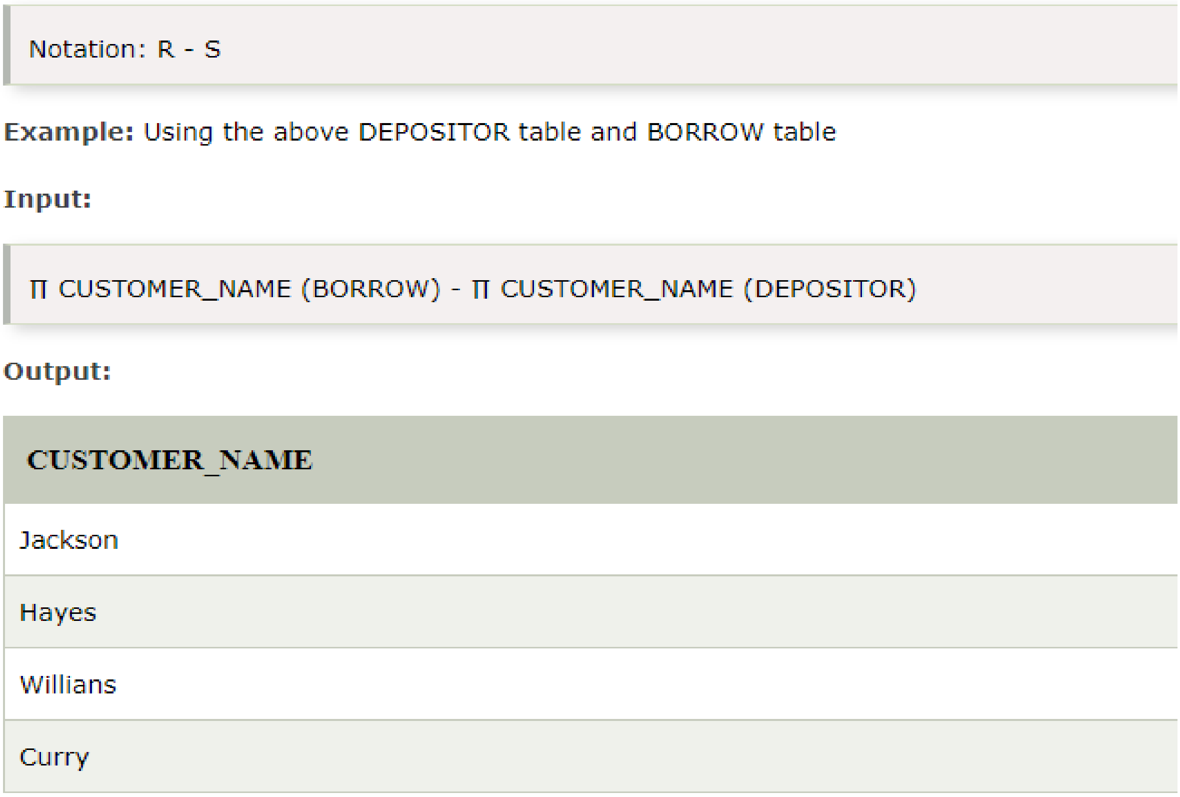
Π CUSTOMER_NAME (BORROW) ∩ Π CUSTOMER_NAME (DEPOSITOR)

Output:

CUSTOMER_NAME
Smith
Jones

5. Set Difference:

The result of set difference operation is tuples, which are present in one relation but are not in the second relation. Notation r – s .Finds all the tuples that are present in r but not in s.



Notation: R - S

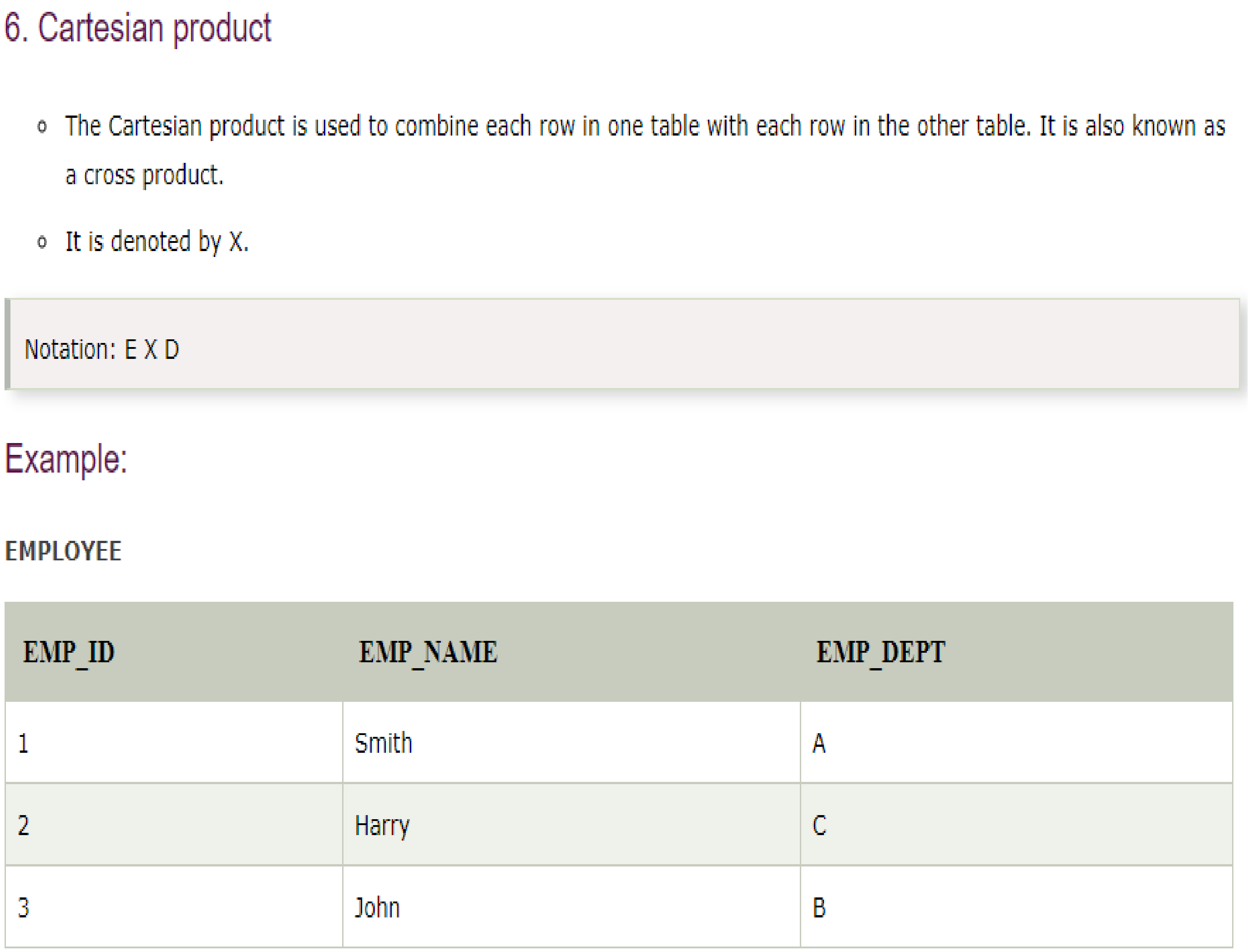
Example: Using the above DEPOSITOR table and BORROW table

Input:

Π CUSTOMER_NAME (BORROW) - Π CUSTOMER_NAME (DEPOSITOR)

Output:

CUSTOMER_NAME
Jackson
Hayes
Willians
Curry



6. Cartesian product

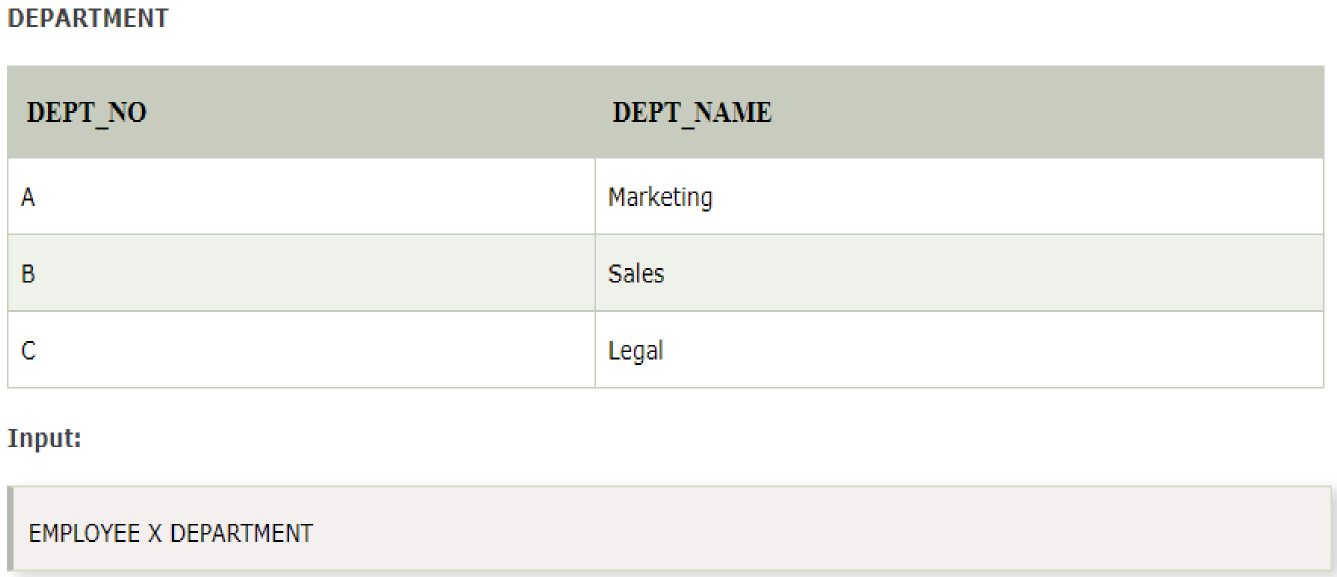
- The Cartesian product is used to combine each row in one table with each row in the other table. It is also known as a cross product.
- It is denoted by X.

```
Notation: E X D
```

Example:

EMPLOYEE

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



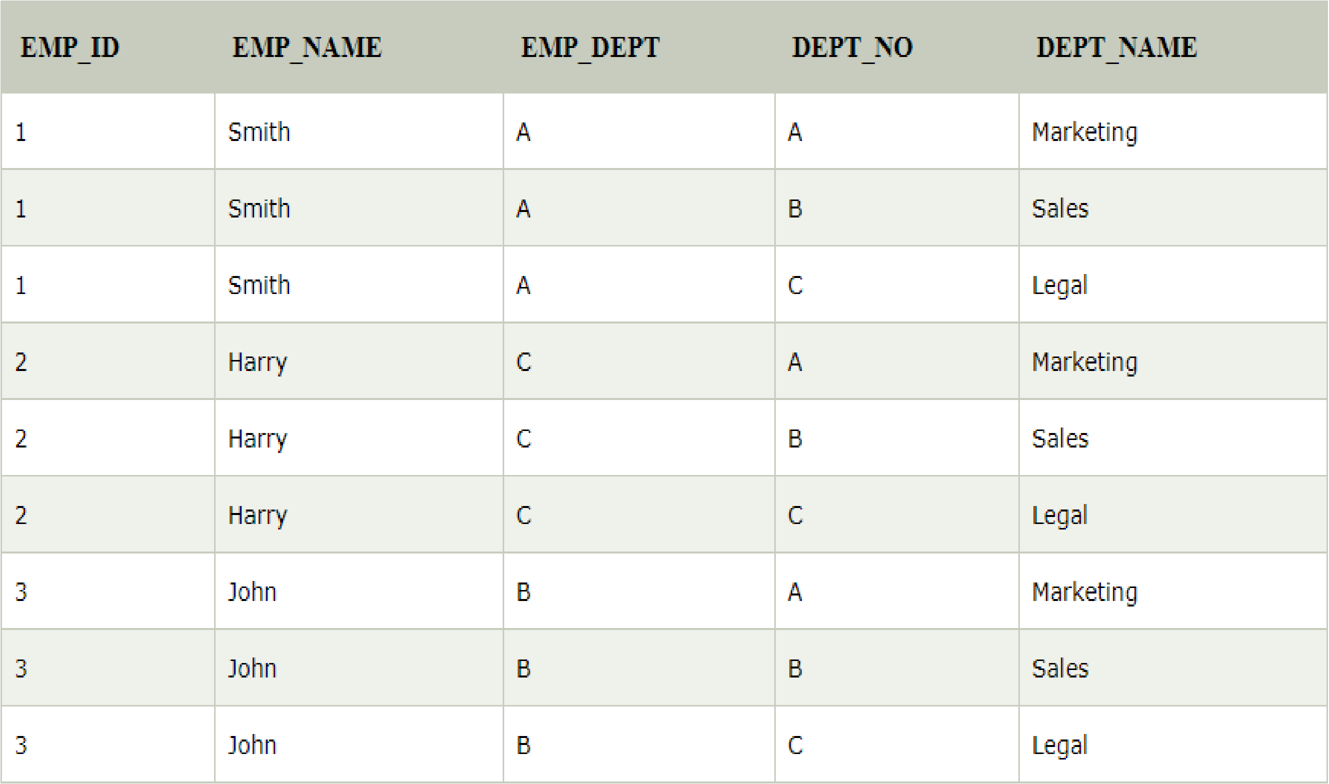
DEPARTMENT

DEPT_NO	DEPT_NAME
A	Marketing
B	Sales
C	Legal

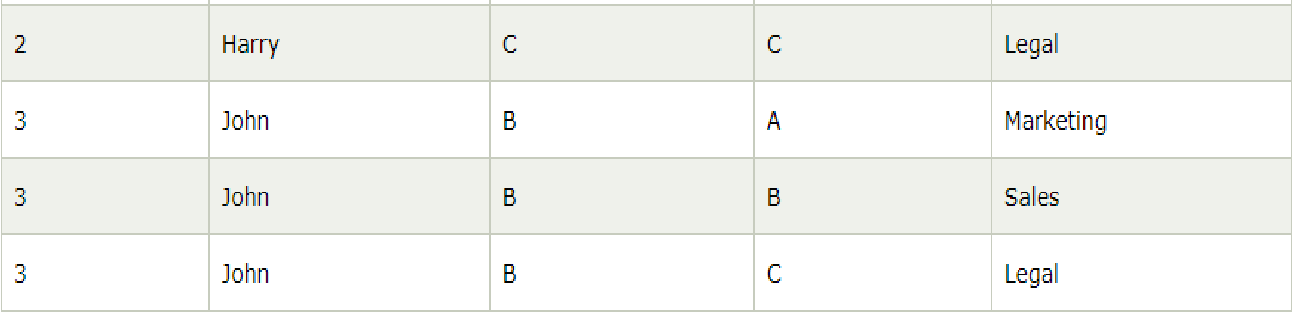
Input:

```
EMPLOYEE X DEPARTMENT
```

Output:



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



2	Harry	C	C	Legal
3	John	B	A	Marketing
3	John	B	B	Sales
3	John	B	C	Legal

7. Rename Operation:

The rename operation is used to rename the output relation. It is denoted by **rho** (ρ).

Example: We can use the rename operator to rename STUDENT relation to STUDENT1.

```
ρ(STUDENT1, STUDENT)
```

7. Joins operation in relational algebra

Join operation in relational algebra is a combination of a Cartesian product followed by which satisfy certain condition. A Join operation combines two tuples from two different relations, if and only if a given condition is satisfied.

There are different types of join operations.

(I) Natural Join (⋈) A result of natural join is the set of tuples of all combinations in R and S that are equal on their common attribute names.

It is denoted by ⋈.

Natural Join (⋈)

Natural join is a binary operator. Natural join between two or more relations will result set of all combination of tuples where they have equal common attribute.

Let us see below example

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	X

Emp ⋈ Dep

Name	Id	Dept_name	Manager
A	120	IT	X
C	110	Sale	Y
D	111	IT	X

Consider the following example to understand natural Joins.

EMPLOYEE

EMP_ID	**EMP_NAME**
1	Ram
2	Varun
3	Lakshmi

SALARY

EMP_ID	**SALARY**
1	50000
2	30000
3	25000

∏ EMP_NAME, SALARY (EMPLOYEE ⋈ SALARY)

Output:

EMP_NAME	**SALARY**
Ram	50000
Varun	30000
Lakshmi	25000

(II) Outer Join

Outer joins are used to include all the tuples from the relations included in join operation in the resulting relation.

An outer join is of three types:

1. Left outer join
2. Right outer join
3. Full outer join

Consider the example **EMPLOYEE**

EMP_NAME	STREET	CITY
Ram	Civil line	Mumbai
Varun	S.G.Road	Kolkata
Lakshmi	C.G.Road	Delhi
Hari	AnandNagar	Hyderabad

FACT_WORKERS

EMP_NAME	BRANCH	SALARY
Ram	Infosys	10000
Varun	Wipro	20000
Neha	HCL	30000
Hari	TCS	50000

(i) Left outer join (⟕)

In Left outer join, all the tuples from the Left relation, say R, are included in the resulting relation. If there are some tuples in relation R which are not matched with tuple in the Right Relation S, then the attributes of relation R of the resulting relation become NULL.

EMPLOYEE ⟕ FACT_WORKERS

Output:

EMP_NAME	STREET	CITY	BRANCH	SALARY
Ram	Civil line	Mumbai	Infosys	10000
Varun	S.G.Road	Kolkata	Wipro	20000
Hari	AnandNagar	Hyderabad	TCS	50000
Lakshmi	C.G.Road	Delhi	NULL	NULL

(ii) Right outer join (⟖)

In Right outer join, all the tuples from the Right relation, say S, are included in the resulting relation. If there are some tuples in relation S which are not matched with tuple in the Right Relation R, then the attributes of relation S of the resulting relation become NULL**.**

EMPLOYEE ⟖ FACT_WORKERS

Output :

EMP_NAME	BRANCH	SALARY	STREET	CITY
Ram	Infosys	10000	Civil line	Mumbai

Varun	Wipro	20000	S.G.Road	Kolkata
Hari	TCS	50000	AnandNagar	Hyderabad
Neha	HCL	30000	NULL	NULL

(iii)Full outer join

Full outer join is the combination of both left outer join and right outer join. It contains all the tuples from both relations.

For example

EMPLOYEE ⟗ FACT_WORKERS

EMP_NAME	STREET	CITY	BRANCH	SALARY
Ram	Civil line	Mumbai	Infosys	10000
Varun	S.G.Road	Kolkata	Wipro	20000
Hari	AnandNagar	Hyderabad	TCS	50000
Lakshmi	C.G.Road	Delhi	NULL	NULL
Neha	NULL	NULL	HCL	30000

(iv) Theta (θ) Join

Theta join is denoted by the symbol θ. It combines those tuples from different relations which satisfies the condition.

Notation – R1 $\bowtie_{\theta}$ R2

Where R1 and R2 are relations with n numbers of attributes such that the attributes do not have anything in common, it means R1 ∩ R2 = Φ.

Student		
Stud_ID	**Name**	**Std**
1	Ram	10
2	Shyam	11

Subjects	
Class	**Subject**
10	Math
10	English
11	Music
11	Sports

STUDENT $\bowtie_{\text{Student.Std = subject.Class}}$ SUBJECT

Output:

SID	**Name**	**Std**	**Class**	**Subject**
1	Ram	10	10	Math
1	Ram	10	10	English
2	Shyam	11	11	Music
2	Shyam	11	11	Sports

RELATIONAL ALGEBRA EXAMPLE SET 2

Player relation

Player Id	Team Id	Country	Age	Runs	Wickets
1001	101	India	25	10000	300
1004	101	India	28	20000	200
1006	101	India	22	15000	150
1005	101	India	21	12000	400
1008	101	India	22	15000	150
1009	103	England	24	6000	90
1010	104	Australia	35	1300	0
1011	104	Australia	29	3530	10
1012	105	Pakistan	28	1421	166
1014	105	Pakistan	21	3599	205

Deposit relation

Acc. No.	Cust-name
A 231	Rahul
A 432	Omkar
R 321	Sachin
S 231	Raj
T 239	Sumit

Borrower relation

Loan No.	Cust-name
P-3261	Sachin
Q-6934	Raj
S-4321	Ramesh
T-6281	Anil

1. SELECT

1. Find all tuples from player relation for which country is India.

   σ“country” = “India”(Player)

2. Select all the tuples for which runs are greater than or equal to 15000.

   σ“runs” >= “15000”(Player)

3. Select all the players whose runs are greater than or equal to 6000 and age is less than 25

   σ“runs” > “6000” ^ age < 25(Player)

2. PROJECT

1. List all the countries in Player relation.

   π“country”(Player)

2. List all the team ids and countries in Player Relation

   πTeam Id, Country (Player)

3. Set Difference Operation (-)

1. Find all the customers having an account but not the loan.

πcust-name(Depositor)–πcust-Name(Borrower)

Result –

Cust-name
Rahul
Omkar
Sumit

2. Find all the customers having a loan but not the account.

   πcust-name(Borrower) – πcust-name(Depositor)

Result

Cust-name
Ramesh
Anil

4. **Cartesian Product Operation (X)**

Example

Employee-Schema = { Emp-id, Name }

Emp-Id	Name
101	Sachin
103	Rahul
104	Omkar
106	Sumit
107	Ashish

Project-Schema = { Proj-name }

Proj-name
DBMS 1
DBMS 2

Find R = Employee X Project

Solution –

R-Schema = {Emp-id, Name, Proj-name}

Emp-Id	Name	Proj-name
101	Sachin	DBMS 1
101	Sachin	DBMS 2
103	Rahul	DBMS 1
103	Rahul	DBMS 2
104	Omkar	DBMS 1
104	Omkar	DBMS 2
106	Sumit	DBMS 1
106	Sumit	DBMS 2
107	Amit	DBMS 1
107	Amit	DBMS 2

6.Rename Operation

Notation of Rename Operation

ρ(NewName, OldName)

Question 1. Rename Customer relation to CustomerList.

Solution
ρ(CustomerList, Customer).

Set Intersection Operation (∩)

Relation A

Id	Name
101	Rahul
104	Anil

Relation B

Id	Name
101	Anil
104	Anil

1.Find all tuples that are common in relations A and B.

Solution – We have to find
P = A ∩ B

So, P is

Id	Name
104	Anil

2. Find names of customers having an account and loan (using Borrower and Deposit, given at start of this post).

Solution – We have to apply set intersection operation in Borrower and Deposit Relation (given at start of this post).

These can be represented as

P = Borrower ∩ Deposit.

So, P is

Cust-name
Sachin
Raj