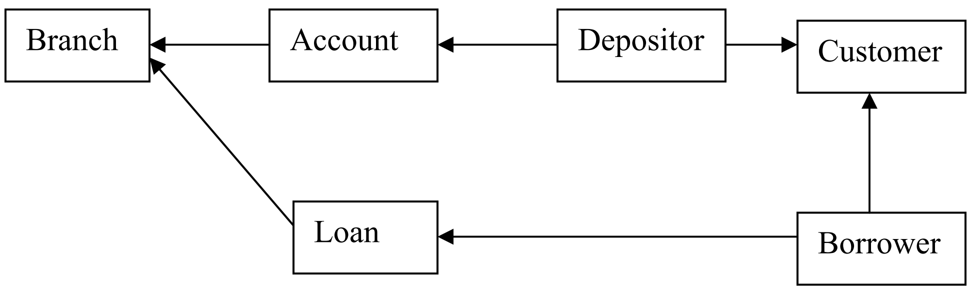
RELATIONAL MODEL LECTURE (Relational Algebra Exercise – Banking)

Consider the following schema for a bank accounts and loan enterprise system.

Branch (branch-no, branch-name, branch-city, assets)
Customer (customer-no, name, street, city)
Account (account-no, branch-no, balance)
Loan (loan-no, branch-no, amount)
Depositor (customer-no, account-no)
Borrower (customer-no, loan-no)

Note: In this example validity of Account's information is tested with Depositor and validity of Loan is tested with Borrower. It means there are some Accounts (account-no) that do not belong to customers, so we cannot consider as active accounts. Same thing is applied for Loan as well.



1. Get the number, name, city and assets of all Branches in city "Brooklyn".

   Result = $\sigma_{\text{branch-city = "Brooklyne"}}$ (Branch)

Points to note: This expression defines all tuples in relation Branch where attribute city = "Brooklyn".

2. Get names and addresses of all Customers in city "Princeton".

   Result = $\prod_{\text{name, street}} (\sigma_{\text{city = "Princeton"}}$ (Customer)$)$

Points to note: Since in this case we want just names and addresses (or name, street and city) rather than all the attributes of Customer, we need the project operation as well as select operation.

3. Get the number, name, city, assets of all Branches.

   Result = σ (Branch)

4. Get a list of customer names who are Borrowers.

   Result = $\prod_{\text{name}}$ (Customer $\bowtie$ Borrower)

5. Get a list of account-no, branch name having branch number "B-25" and balance greater than 2560. (use Cartesian Product method)

$$\text{BrAcc} = \text{Branch} \times \left(\prod_{\text{account-no, branch-no}} \sigma_{\text{branch-no} = \text{"B-25"} \wedge \text{balance} > 2560} (\text{Account})\right)$$

$$\text{Result} = \prod_{\text{account-no, branch-name}} \left(\sigma_{\text{br.branch-no} = \text{acc.branch-no}} \text{ BrAcc}\right)$$

Points to note:

- Two relations are used here and so we need the join operation
- Before the join occurs attribute branch-no has been projected in the preceding sub expression.
- This projection is so that the required join becomes a natural join. Had branch-no not been projected in the sub expression, both relations to be joined would have contained not only common attribute branch-no but also attribute name, which has a different meaning in each of the relations.
- The projection of branch-no simplifies the join and subsequent final projection.

6. Get the list of customer's name, customer's city for loan numbers above or same as "L-15".

$$\text{Result} = \prod_{\text{name, city}} \left(\text{Customer} \bowtie \left(\prod_{\text{customer-no}} \sigma_{\text{loan-no} >= \text{"L-15"}} (\text{Borrower})\right)\right)$$

7. Get the list of customer's name, customer's city and Depositor's account number, who have branch in city "Perryidge".

$$\text{R1} \leftarrow \left(\prod_{\text{account-no, branch-no}} (\text{Account})\right) \times \left(\prod_{\text{branch-no}} \sigma_{\text{branch-city} = \text{"Perryidge"}} (\text{Branch})\right)$$

$$\text{R2} \leftarrow \prod_{\text{account-no, branch-no}} \left(\sigma_{\text{Account.branch-no} = \text{Branch.branch-no}} (\text{R1})\right)$$

$$\text{Result} = \prod_{\text{name, city, account-no}} \left(\text{Customer} \times \left(\prod_{\text{customer-no, account-no}} (\text{Depositor})\right) \bowtie \text{R2}\right)$$

Alternatively, using natural join (equi-join) above expression can be written as:

$$\text{Result} = \prod_{\text{name, city, loan-no}} \left(\left(\text{Customer} \bowtie \text{Depositor}\right) \times \text{R2}\right)$$

8. List the names of all bank customers who have either an account or a loan or both.

Names of all customers with a loan in the bank:

$$\text{R1} \leftarrow \prod_{\text{name}} \left(\text{Customer} \bowtie \left(\prod_{\text{customer-no}} (\text{Borrower})\right)\right)$$

Names of all customers with an account in the bank:

$$R2 \leftarrow \prod_{name} \left(Customer \blacktriangleright\blacktriangleleft \left(\prod_{customer\text{-}no} (Depositor) \right) \right)$$

To answer the query, we need the union of these two sets.

Result = R1 U R2

9. Find all customers who have an account but not a loan. (use of minus)

   Result = R2 — R1

10. Get the list of all customers who have both a loan and an account.

   Result = R1 ∩ R2

11. Find the names of all customers who have a loan at the branch name "Perryidge". (use of Cartesian Products)

$$BL \leftarrow \sigma_{branch\text{-}name = \text{"Perryidge"}} (Borrower \times Loan)$$

$$CNO \leftarrow \prod_{customer\text{-}no} \left(\sigma_{Borrower.loan\text{-}no = Loan.loan\text{-}no} (BL) \right)$$

$$Result = \prod_{customer.customer\text{-}no} \left(\sigma_{Customer.customer\text{-}no = CNO.customer\text{-}no} (Customer \times CNO) \right)$$

12. Find the names of all customers in city "Princeton", who have a loan at the bank, and find the amount of the loan as well.

$$Result = \prod_{name,\ loan\text{-}no,\ amount} \left(Borrower \times Loan \times \left(\sigma_{city = \text{"Princeton"}} (Customer) \right) \right)$$

Points to note: Cartesian product of two arguments performs a selection forcing equality on those attributes that appear in both relation schemas and finally removes duplicate attributes.

Alternatively, last Cartesian product can be replaced by intersection operation. Also, verify this query is correct or not.

13. Find all customers who have an account at **all** the branches located in "Brooklyne".

   First obtain all branches in Brooklyne by the expression

$$R \leftarrow \prod_{branch\text{-}name} \left(\sigma_{branch\text{-}city = \text{"Brooklyne"}} (Branch) \right)$$

   Second, find all (customer-name, branch-name) pairs for which customer has an account at a branch by writing

$$S \leftarrow \prod_{\text{customer-no, branch-name}} \left(\left(\text{Depositor} \bowtie \text{Account} \right) \bowtie \text{Branch} \right)$$

Then to find customers who appear in S with every branch name in R.

$\text{Result} = S \div R$

14. List of customer-no, branch-no and loan amount of all borrower customers who withdraw loan amount or not.

$$\text{Result} = \prod_{\text{customer-no, branch-no, amount}} \left(\text{Loan} ⟕ \text{Borrower} \right)$$

15. Find the largest account balance in bank. (Use of rename column, cartesian product and minus).

Compute first a temporary relation consisting of those balances that are not the largest

$$R \leftarrow \prod_{\text{account.balance}} \left(\sigma_{\text{account.balance} < \text{d.balance}} \; (\text{Account x } \rho_d \; (\text{Account})) \right)$$

Take the difference

$$\text{Result} = \prod_{\text{account.balance}} \left(\text{Account} \right) \; — \; R$$

16. Delete all those customer records from Depositor where customer number is 56475.

$$\text{Depositor} \leftarrow \text{Depositor} — \sigma_{\text{customer-no} = 56475} \; (\text{Depositor}) \big)$$

17. Delete all accounts at branches located in "Needham"

$$R1 \leftarrow \sigma_{\text{branch-city} = \text{"Needham"}} \; (\text{Account} \bowtie \text{Branch})$$

$$R2 \leftarrow \prod_{\text{account-no, branch-no, balance}} \left(R1 \right)$$

$\text{Account} \leftarrow \text{Account} — R2$

18. Insert a deposit record for a customer-no 22341 having balance amount of $1200 for account number A-984 at branch number 2556

$\text{Account} \leftarrow \text{Account U } \{(\text{"A-984"}, 2556, 1200)\}$

$\text{Depositor} \leftarrow \text{Depositor U } \{(22341, \text{"A-984"})\}$

19. Give 6% raise to those account holders who have balance amount more than $1000. (Example of Update table Account)

$$\text{Account} \leftarrow \prod_{\text{account-no, branch-no, balance * 1.06}} \left(\sigma_{\text{balance} > 1000} \; (\text{Account}) \right)$$

20. List of customers' Customer-no, Address (rename attribute for street) where city is "Perryridge".

$$\text{Result} \leftarrow \prod_{\text{customer-no, address}} \left(\rho_{\text{address} \leftarrow \text{street}} \left(\sigma_{\text{city = "Perryridge"}} (\text{Customer})\right)\right)$$

MISC. NOTE: Difference between Cartesian product and natural join can be elaborate mathematically for two relations A and B as:

$$\sigma_{A.a = B.b} (A \times B) = A \bowtie B$$

Also,

ρ_d (Account) is same as $\rho_{d \leftarrow \text{balance}}$ (Account)

Writing up join condition while using σ with cartesian product.
Natural join ►◄ obviously indicates join on same column names from both relations.

NOTE: Unnecessary link of a relation is not correct.
For example, if it is required to join Borrower for Customer-no, then there is no need to join customer of some other attribute is not needed from Customer.

Please adopt the standards defined in this tutorial for future explanation of Relational Algebra

EXERCISE-1: List of all those employees who are working under managers only.

EXERCISE-2: Modify query 13, using customer-no and branch-no, showing their names too

EXERCISE-3: How to prove $\sigma_{A.a = B.b} (A \times B) = A \bowtie B$

A:

a	b
1	5
2	8
3	9

B:

a	c
1	6
2	7

A x B:

a	b	a	c
1	5	1	6
1	5	2	7
2	8	1	6
2	8	2	7
3	9	1	6
3	9	2	7

$\sigma_{A.a = B.a} (A \times B)$:

a	b	a	c
1	5	1	6
2	8	2	7