• branch (branch\_name, branch\_city, assets)

• customer (customer\_name, customer\_street, customer\_city)

• account (account\_number, branch\_name, balance)

• loan (loan\_number, branch\_name, amount)

• depositor (customer\_name, account\_number)

•borrower (customer\_name, loan\_number)

1. Find all loans of over $1200

σ amount > 1200 (loan)

1. Find the loan number for each loan of an amount greater than $1200

∏loan\_number (σamount > 1200 (loan))

1. Find the names of all customers who have a loan, an account, or both from the bank

∏customer\_name(depositor) U ∏customer\_name(borrower)

1. Find the names of all customers who have a loan and an account at the bank

∏customer\_name(depositor) ∩ ∏customer\_name(borrower)

1. Find the names of all customers who have a loan at the Perryridge branch

∏ customer\_name (σ branch \_name=” Perryridge” (loan∞borrower))

1. Find the names of all customers who have a loan at the Perryridge branch but do not have an account at any branch of the bank

∏ customer\_name (σ branch \_name=” Perryridge” (loan∞borrower)) -- ∏ customer\_name (depositor ∞ account))

***Cartesian Product***

∏ customer\_name (σ branch \_name=” Perryridge” ^ l.loan\_number=b.loan\_number (loan x borrower)) --

∏ customer\_name(σ d.account\_number = a.account\_number (depositor x account))

1. Find all customers who have an account from at least the “Downtown” and the “Uptown” branches

∏ c.customer\_name, c.customer\_street, c.customer\_city (σ branch \_name=” Downtown” (customer∞depositor∞account))

**∩**

∏ c.customer\_name, c.customer\_street, c.customer\_city (σ branch \_name=” Uptown” (customer∞depositor∞account))