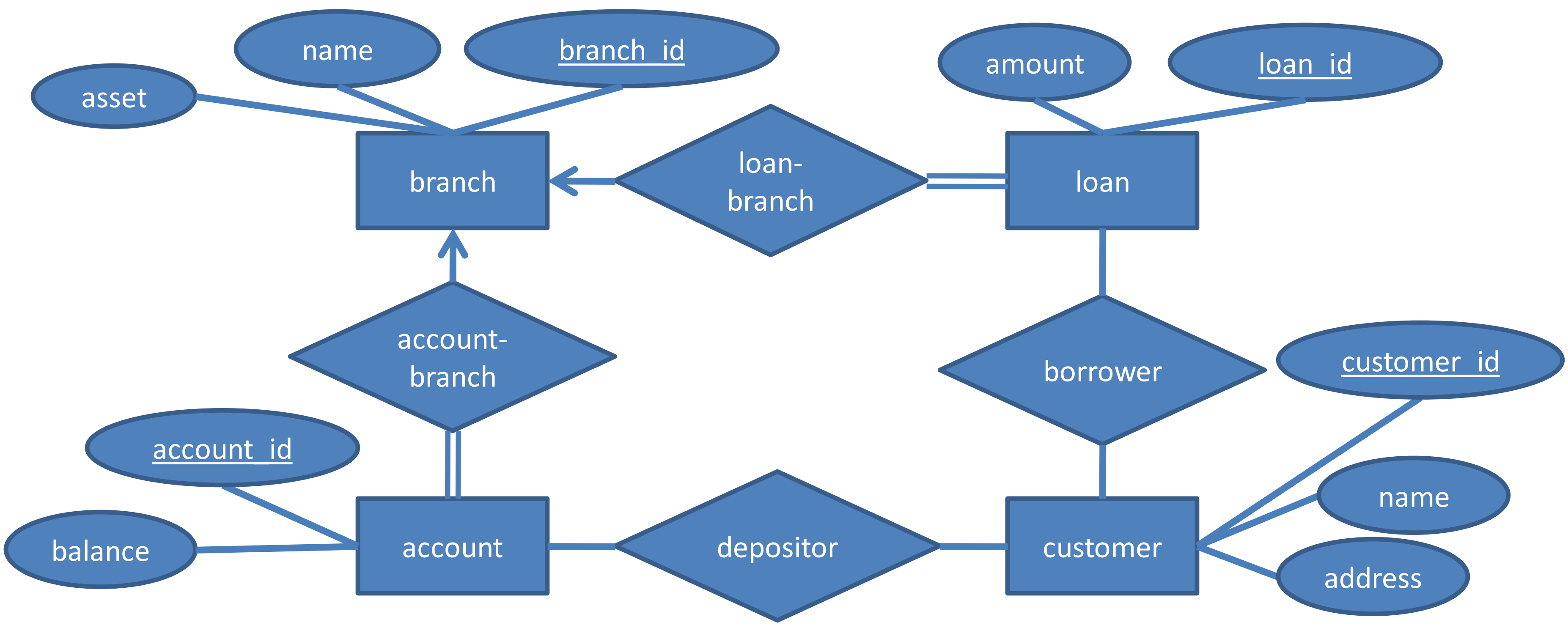
Banking example

- **Let's consider the banking enterprise example we used in the previous chapter.**



Banking example

Foreign key

Foreign key

Branch

branch_id	name	asset
B1	Central	7100000
B2	Causeway Bay	9000000
B3	Aberdeen	400000
B4	North Point	3700000

Loan

branch_id	loan_id	amount
B3	L1	900
B1	L2	1500
B1	L3	1000

Borrower

customer_id	loan_id
C1	L3
C4	L2
C2	L1

Foreign key

Foreign key

Foreign key

Account

branch_id	account_id	balance
B1	A1	500
B2	A2	400
B2	A3	900
B1	A4	700

Depositor

account_id	customer_id
A1	C1
A1	C2
A2	C2
A3	C4
A4	C4

Foreign key

Customer

customer_id	name	address
C1	Kit	CB320
C2	Ben	CB326
C3	Jolly	CB311
C4	Yvonne	CB415

Question 1

- Query: find the **customer_id** of all customers who have an account at a **branch** located in **Central**.

- Relational expression

```
SELECT customer_id
FROM branch, account, depositor
WHERE
    account.account_id = depositor.account_id AND
    branch.branch_id = account.branch_id AND
    branch.location = "Central"
```

SQL

$$\Pi_{customer\_id}\Big(\sigma_{branch.location = \text{"Central"}}\big(branch \bowtie (account \bowtie depositor) \big) \Big)$$

Relational algebra expression

branch

branch_id	location	asset
B1	Central	7100000
B2	Causeway Bay	9000000
B3	Aberdeen	400000
B4	North Point	3700000

account

branch_id	account_id	balance
B1	A1	500
B2	A2	400
B2	A3	900
B1	A4	700

depositor

account_id	customer_id
A1	C1
A1	C2
A2	C2
A3	C4
A4	C4

Question 1

- Query: find the **customer_id** of all customers who have an account at a **branch** located in **Central**.
- Relational expression

$$\Pi_{customer\_id}(\sigma_{branch.location = \text{“Central”}}(branch \bowtie (account \bowtie depositor)))$$

Note that we can transform the relational expression by using the equivalence rules in Chapter 4B.
E.g., **Performing selection as early as possible reduces the size of the relation to be joined**.

Question 1

- We can reduce the size of temporary relation by transforming the expression according to using rule 5a.
  - **Rule 5a.** The selection operation distributes over the natural join operation when all the attributes in selection condition **involve only the attributes of one of the expressions** (say, E_1) being joined.

$$\Pi_{customer\_id}(\sigma_{branch.location = \text{"Central"}}(branch \bowtie (account \bowtie depositor)))$$

$$\sigma_p(E_1 \bowtie E_2) = (\sigma_p(E_1) \bowtie E_2)$$

$$\Pi_{customer\_id}((\sigma_{branch.location = \text{"Central"}}(branch)) \bowtie (account \bowtie depositor))$$

Question 2a

- Find the **customer_id** of all customers with an account at **Central** branch whose account **balance is over $1000**

```
SELECT customer_id
FROM branch, account, depositor
WHERE
    account.account_id = depositor.account_id AND
    branch.branch_id = account.branch_id AND
    branch.location = “Central” AND balance > 1000
```

SQL

branch

branch_id	location	asset
B1	Central	7100000
B2	Causeway Bay	9000000
B3	Aberdeen	400000
B4	North Point	3700000

account

branch_id	account_id	balance
B1	A1	5000
B2	A2	4000
B2	A3	900
B1	A4	700

depositor

account_id	customer_id
A1	C1
A1	C2
A2	C2
A3	C4
A4	C4

- Relational expression

$$\Pi_{customer\_id}\Big(\sigma_{branch.location = \text{“Central”} \wedge balance > 1000}\big(branch \bowtie (account \bowtie depositor) \big) \Big)$$

Question 2b

- Transform the expression in question 2a to another equivalent expression with smaller temporary relation
  - **Rule 4.** Natural join operations are associative.

$$(E_1 \bowtie E_2) \bowtie E_3 = E_1 \bowtie (E_2 \bowtie E_3)$$

$$\Pi_{customer\_id}(\sigma_{branch.location = \text{“Central”} \wedge balance > 1000}(branch \bowtie (account \bowtie depositor)))$$

$$\Pi_{customer\_id}(\sigma_{branch.location = \text{“Central”} \wedge balance > 1000}((branch \bowtie account) \bowtie depositor))$$

As **branch.location** and **balance** are attributes of the **branch** and **account** relations, we would like to make them on one side of the natural join (the one pointed by the pointer) so that we can apply rules to push the selection down the natural join.

Question 2b

- **Perform selection early**
  - **Rule 5a.** The selection operation distributes over the natural join operation when all the attributes in selection condition **involve only the attributes of one of the expressions** (say, E_1) being joined.

Now the selection predicates are all on the attributes of **branch** and **account** relations, i.e., on the L.H.S. of the natural join (the one pointed by the pointer), we can push the selection down the L.H.S. of that natural join.

$$\Pi_{customer\_id}(\sigma_{branch.location = \text{“Central”} \wedge balance > 1000}((branch \bowtie account) \bowtie depositor))$$

$$\Pi_{customer\_id}((\sigma_{branch.location = \text{“Central”} \wedge balance > 1000}(branch \bowtie account)) \bowtie depositor)$$

Question 2b

- **Perform selection early**
  - **Rule 5b.** The selection distributes over natural join when selection condition p1 involves only the attributes of E_1 and p2 involves only the attributes of E_2.

$$\sigma_{p1 \wedge p2}(\, E_1 \bowtie E_2) = (\sigma_{p1}(\, E_1\,) \bowtie \sigma_{p1}(\, E_2\,))$$

We can further push the selection predicates down the natural join between **branch** and **account**.

$$
\begin{aligned}
&\Pi_{customer\_id}(\\
&\qquad (\sigma_{branch.location\,=\,\text{“Central”}}\,(branch) \bowtie \sigma_{balance\,>\,1000}\,(account)\,)\\
&\qquad \bowtie depositor\\
&)
\end{aligned}
$$

$$
\begin{aligned}
&\Pi_{customer\_id}(\\
&\qquad (\sigma_{branch.location\,=\,\text{“Central”}\,\wedge\,balance\,>\,1000}\,(branch \bowtie account)\,)\\
&\qquad \bowtie depositor\\
&)
\end{aligned}
$$

Question 2b

- **Perform projection early**
  - **Rule 6.** The projection operation can distribute over the natural join operation.

$$\pi_{L1 \cup L2}(E_1 \bowtie E_2) = \pi_{L1 \cup L2}((\pi_{L1 \cup L3}(E_1)) \bowtie (\pi_{L2 \cup L3}(E_2)))$$

When pushing $\Pi_{customer\_id}$ down the natural join (the one pointed by pointer), we need to add the attribute that is used in the joining (i.e., **account_id**).

$$\Pi_{customer\_id}(\ (\sigma_{branch.location = \text{“Central”}}(branch) \bowtie \sigma_{balance > 1000}(account)) \bowtie depositor\ )$$

$$\Pi_{customer\_id}(\ \Pi_{account\_id}(\ \sigma_{branch.location = \text{“Central”}}(branch) \bowtie \sigma_{balance > 1000}(account)\ ) \bowtie \Pi_{customer\_id,\ account\_id}(depositor)\ )$$

Question 2b

- **Perform projection early**

When pushing $\Pi_{account\_id}$ down the natural join (the one pointed by pointer), we need to add the attribute that is used in the joining (i.e., **branch_id**).

$$
\begin{aligned}
&\Pi_{customer\_id}(\\
&\qquad \Pi_{account\_id}(\\
&\qquad\qquad \sigma_{branch.location = \text{“Central”}}(branch) \bowtie \sigma_{balance > 1000}(account)\\
&\qquad)\\
&\qquad \bowtie \Pi_{customer\_id,\ account\_id}(depositor)\\
&)
\end{aligned}
$$

$$
\begin{aligned}
&\Pi_{customer\_id}(\\
&\qquad \Pi_{account\_id}(\\
&\qquad\qquad \Pi_{branch\_id}(\\
&\qquad\qquad\qquad \sigma_{branch.location = \text{“Central”}}(branch)\\
&\qquad\qquad)\\
&\qquad\qquad \bowtie\\
&\qquad\qquad \Pi_{account\_id,branch\_id}(\\
&\qquad\qquad\qquad \sigma_{balance > 1000}(account)\\
&\qquad\qquad)\\
&\qquad)\\
&\qquad \bowtie\\
&\qquad \Pi_{customer\_id,\ account\_id}(depositor)\\
&)
\end{aligned}
$$

Question 3

- Find the **sID** and **name** of the employee who know all the IT skills in the company.

$$\left((\text{Has}) \div \pi_{\text{skillID}}(\text{IT\_skill}) \right)$$

The use of division to find the sID in Has that has all skillID appearing in IT_skill table.

Staff

sID	name	dpt_id
1	Peter	1
2	Sharon	1
3	David	2
4	Joe	3

Has

sID	skillID
1	1
1	2
1	3
2	3
3	3
4	1
4	2
4	3

IT_skill

skillID	skillName	desc
1	C++	…
2	JAVA	…
3	MySQL database	…

$(\text{Has}) \div \pi_{\text{skillID}}(\text{IT\_skill})$

sID
1
4

Question 3

- Find the **sID** and **name** of the employee who know all the IT skills in the company.

$\pi_{sID,\ name}$((**(Has)** $\div$ $\pi_{skillID}$**(IT_skill)**) $\bowtie$ **Staff**)

sID	name
1	Peter
4	Joe

$\pi_{sID,\ name}($

(**(Has)** $\div$ $\pi_{skillID}$**(IT_skill)**)

$\bowtie$

Staff

)

With the sID, we join with Staff and project the sID and name of the staffs.

Staff

sID	name	dpt_id
1	Peter	1
2	Sharon	1
3	David	2
4	Joe	3

Has

sID	skillID
1	1
1	2
1	3
2	3
3	3
4	1
4	2
4	3

IT_skill

skillID	skillName	desc
1	C++	…
2	JAVA	…
3	MySQL database	…

(Has) $\div$ $\pi_{skillID}$**(IT_skill)**

sID
1
4

Question 4

- Find the **sID** of the staffs who know about C++ or works in the IT department (dpt_id=2).

(

$\pi_{sID}($ **Has** $\bowtie$ $\sigma_{skillName=\text{"C++"}}($**IT_skill**$)$ $)$

$\cup$

$\pi_{sID}($ $\sigma_{dpt\_id=2}($ **Staff** $)$ $)$

)

Given the skillName as "C++", find the sID of the staffs who has this skill.

$\pi_{sID}($ **Has** $\bowtie$ $\sigma_{skillName=\text{"C++"}}($**IT_skill**$)$ $)$

sID
1
4

Find the sID of the staffs who works in the IT department.

$\pi_{sID}($ $\sigma_{dpt\_id=2}($ **Staff** $)$ $)$

sID
3

Staff

sID	name	dpt_id
1	Peter	1
2	Sharon	1
3	David	2
4	Joe	3

Has

sID	skillID
1	1
1	2
1	3
2	3
3	3
4	1
4	2
4	3

IT_skill

skillID	skillName	desc
1	C++	…
2	JAVA	…
3	MySQL database	…

Question 5

- List the name of the IT skills that the staffs named "Peter" and "David" know.

Staff

sID	name	dpt_id
1	Peter	1
2	Sharon	1
3	David	2
4	Joe	3
5	David	4

Has

sID	skillID
1	1
1	2
1	3
2	3
3	3
4	1
4	2
4	3

IT_skill

skillID	skillName	desc
1	C++	…
2	JAVA	…
3	MySQL database	…

Result

sID	name	skillName
1	Peter	C++
1	Peter	JAVA
1	Peter	MySQL database
3	David	MySQL database
5	David	***null***

Question 5

- List the name of the IT skills that the staffs named "Peter" and "David" know.

$\sigma_{name= \text{"Peter"} \vee name = \text{"David"}}$ (**Staff**)

Select the Staffs with name equal to "Peter" or "David".

$\sigma_{name= \text{"Peter"} \vee name = \text{"David"}}$ (**Staff**)

sID	name	dpt_id
1	Peter	1
3	David	2
5	David	4

Staff

sID	name	dpt_id
1	Peter	1
2	Sharon	1
3	David	2
4	Joe	3
5	David	4

Has

sID	skillID
1	1
1	2
1	3
2	3
3	3
4	1
4	2
4	3

IT_skill

skillID	skillName	desc
1	C++	…
2	JAVA	…
3	MySQL database	…

Question 5

- List the name of the IT skills that the staffs named "Peter" and "David" know.

$\sigma_{name=\text{"Peter"} \vee name=\text{"David"}}(\textbf{Staff}) ⟕ \textbf{Has}$

Use left outer join to retain the staff "David" (sID = 5) who knows no IT_skills.

$\sigma_{name=\text{"Peter"} \vee name=\text{"David"}}(\textbf{Staff})$

sID	name	dpt_id
1	Peter	1
3	David	2
5	David	4

Staff

sID	name	dpt_id
1	Peter	1
2	Sharon	1
3	David	2
4	Joe	3
5	David	4

Has

sID	skillID
1	1
1	2
1	3
2	3
3	3
4	1
4	2
4	3

IT_skill

skillID	skillName	desc
1	C++	…
2	JAVA	…
3	MySQL database	…

$\sigma_{name=\text{"Peter"} \vee name=\text{"David"}}(\textbf{Staff}) ⟕ \textbf{Has}$

sID	name	dpt_id	skillID
1	Peter	1	1
1	Peter	1	2
1	Peter	1	3
3	David	2	3
5	David	4	***null***

Question 5

- List the name of the IT skills that the staffs named “Peter” and “David” know.

With skillID, we further join with IT_skill table to get the name of the skills

($\sigma_{name= \text{“Peter”} \vee name = \text{“David”}}$ (**Staff**)

⟕ **Has**)

⟕ **IT_skill**

$\sigma_{name= \text{“Peter”} \vee name = \text{“David”}}$ (**Staff**) ⟕ **Has** ⟕ **IT skill**

sID	name	dpt_id	skillID	skillName
1	Peter	1	1	C++
1	Peter	1	2	JAVA
1	Peter	1	3	MySQL database
3	David	2	3	MySQL database
5	David	4	***null***	***null***

Staff

sID	name	dpt_id
1	Peter	1
2	Sharon	1
3	David	2
4	Joe	3
5	David	4

Has

sID	skillID
1	1
1	2
1	3
2	3
3	3
4	1
4	2
4	3

IT_skill

skillID	skillName	desc
1	C++	…
2	JAVA	…
3	MySQL database	…

$\sigma_{name= \text{“Peter”} \vee name = \text{“David”}}$ (**Staff**) ⟕ **Has**

sID	name	dpt_id	skillID
1	Peter	1	1
1	Peter	1	2
1	Peter	1	3
3	David	2	3
5	David	4	***null***

Question 5

- List the name of the IT skills that the staffs named "Peter" and "David" know.

$$\pi_{sID,\ name,\ skillName}\Big(\big(\sigma_{name=\text{"Peter"} \vee name=\text{"David"}}(\textbf{Staff}) ⟕ \textbf{Has} \big) ⟕ \textbf{IT\_skill} \Big)$$

$\sigma_{name=\text{"Peter"} \vee name=\text{"David"}}(\textbf{Staff}) ⟕ \textbf{Has} ⟕ \textbf{IT skill}$

sID	name	dpt_id	skillID	skillName
1	Peter	1	1	C++
1	Peter	1	2	JAVA
1	Peter	1	3	MySQL database
3	David	2	3	MySQL database
5	David	4	***null***	***null***

Staff

sID	name	dpt_id
1	Peter	1
2	Sharon	1
3	David	2
4	Joe	3
5	David	4

Has

sID	skillID
1	1
1	2
1	3
2	3
3	3
4	1
4	2
4	3

IT_skill

skillID	skillName	desc
1	C++	…
2	JAVA	…
3	MySQL database	…

Result

sID	name	skillName
1	Peter	C++
1	Peter	JAVA
1	Peter	MySQL database
3	David	MySQL database
5	David	***null***