BACS3183
Advanced Database Management

Chapter 4

Relational Model

Learning Outcomes

- Formulate queries in relational algebra.
- Understand the basic concepts of query processing and optimization.

Introduction

- **Relational algebra and calculus are the theoretical concepts used on relational model.**
- **RDBMS is a practical implementation of relational model.**
- **SQL is a practical implementation of relational algebra and calculus.**

1. Relational Algebra

- Five basic operations in relational algebra: **Selection, Projection, Cartesian product, Union, and Set Difference.**

- These perform most of the data retrieval operations needed.

- Also have **Join, Intersection, and Division** operations

Dream Home

BRANCH (**BranchNo**, Street, City, Postcode)

STAFF (**StaffNo**, FName, LName, Position, Sex, DOB, Salary, BranchNo)

PROPERTYFORRENT (**PropertyNo**, Street, City, Postcode, Type, Rooms, Rent, OwnerNo, StaffNo, BranchNo)

PRIVATEOWNER (**OwnerNo**, FName, LName, Address, TelNo)

CLIENT (**ClientNo**, FName, LName, TelNo, PrefType, MaxRent)

VIEWING (**ClientNo, PropertyNo,** ViewDate, Comments)

Dream Home Sample Relations

Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Dream Home Sample Relations

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

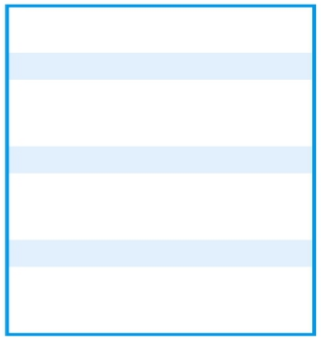
Client

clientNo	fName	lName	telNo	prefType	maxRent
CR76	John	Kay	0207-774-5632	Flat	425
CR56	Aline	Stewart	0141-848-1825	Flat	350
CR74	Mike	Ritchie	01475-392178	House	750
CR62	Mary	Tregear	01224-196720	Flat	600

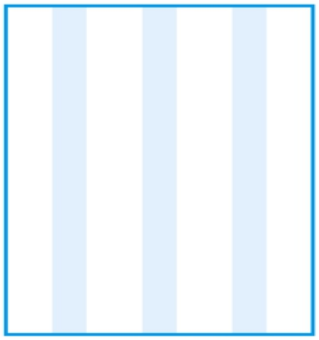
Viewing

clientNo	propertyNo	viewDate	comment
CR56	PA14	24-May-04	too small
CR76	PG4	20-Apr-04	too remote
CR56	PG4	26-May-04	
CR62	PA14	14-May-04	no dining room
CR56	PG36	28-Apr-04	

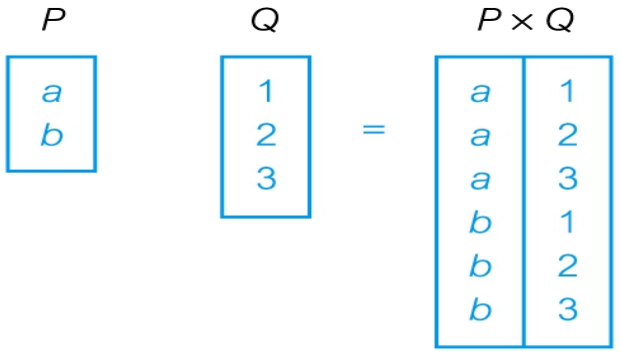
Relational Algebra Operations



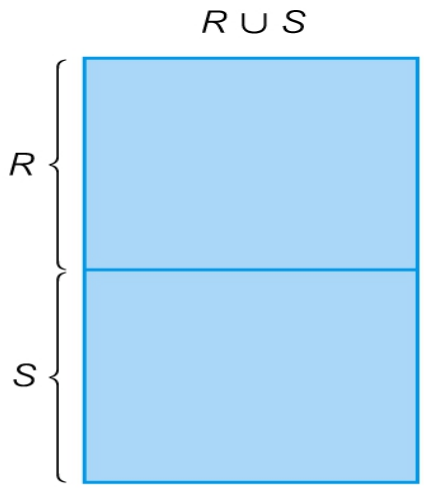
(a) Selection



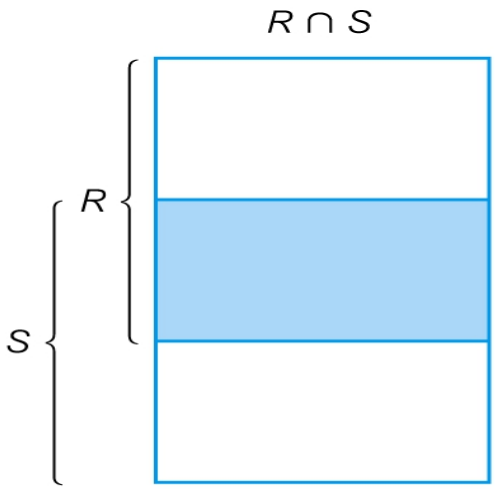
(b) Projection



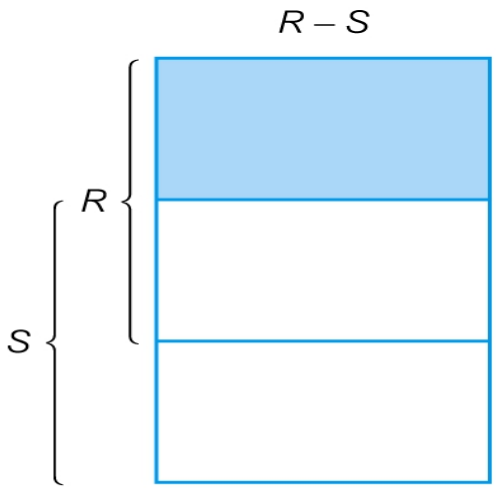
(c) Cartesian product



(d) Union



(e) Intersection



(f) Set difference

Relational Algebra Operations

T

A	B
a	1
b	2

U

B	C
1	x
1	y
3	z

$T \bowtie U$

A	B	C
a	1	x
a	1	y

(g) Natural join

$T \triangleright_B U$

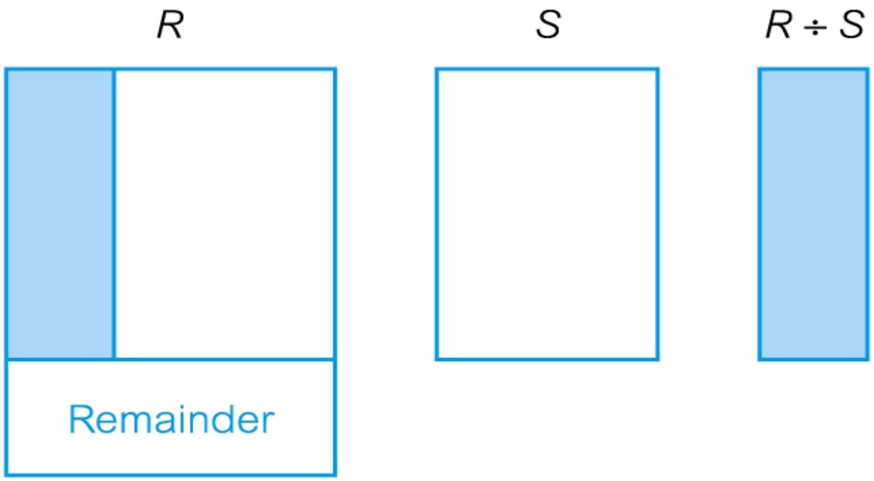
A	B
a	1

(h) Semijoin

$T ⟕_C U$

A	B	C
a	1	x
a	1	y
b	2	

(i) Left Outer join



(j) Divis on (shaded area)

V

A	B
a	1
a	2
b	1
b	2
c	1

W

B
1
2

$V \div W$

A
a
b

Example of division

Relational Algebra Symbols

- **Projection (π)**
- **Selection (σ)**
- **Rename (ρ)**
- **Natural join (⋈)**
- **Left outer join (⟕)**
- **Right outer join (⟖)**
- **Semijoin (▷)**
- **Division (÷)**

- **Union (u)**
- **Intersection (∩)**
- **Difference (-)**

Selection (or Restriction)

- $\sigma_{predicate}$ **(R)**
  - **Works on a single relation R and defines a relation that contains only those tuples (rows) of R that satisfy the specified condition (*predicate*).**

Will display all attributes of the relation

Example - Selection (or Restriction)

- List all staff with a salary greater than £10,000.

$\sigma_{salary > 10000}$ **(Staff)**

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24- Mar-58	18000	B003
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003

Projection

- $\Pi_{col1, \ldots, coln}(R)$
  - **Works on a single relation R and defines a relation that contains a vertical subset of R, extracting the values of specified attributes and eliminating duplicates.**

Project yields values for selected attributes

Example - Projection

- Produce a list of salaries for all staff, showing only staffNo, fName, lName, and salary details.

$\Pi_{staffNo, fName, lName, salary}(Staff)$

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

staffNo	fName	lName	salary
SL21	John	White	30000
SG37	Ann	Beech	12000
SG14	David	Ford	18000
SA9	Mary	Howe	9000
SG5	Susan	Brand	24000
SL41	Julie	Lee	9000

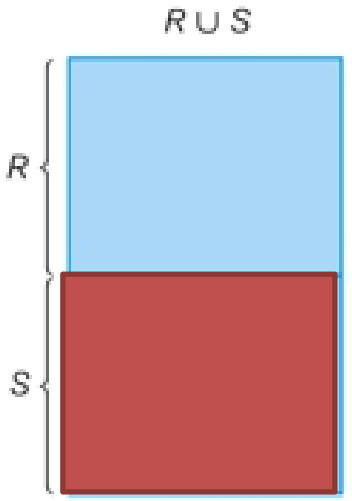
Question

Given the following SQL statements, write the related relational algebra

```
Select StaffNo,Fname, Lname, Salary
From Staff
Where salary < 1000;
```

Union

- **$R \cup S$**
  - **Union of two relations R and S defines a relation that contains all the tuples of R, or S, or both R and S, duplicate tuples being eliminated.**
  - **R and S must be union-compatible.**



Same attributes

- **If R and S have *I* and *J* tuples, respectively, union is obtained by concatenating them into one relation with a maximum of (*I* + *J*) tuples.**

UNION

- The Company A has bought another company, Company B. Thus, the company has 2 customer tables that require merging.
- Merge the customer table with customer_2 table using union

Example - Union

- List all cities where there is either a branch office **or** a property for rent.

$$\Pi_{city}(Branch) \cup \Pi_{city}(PropertyForRent)$$

Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

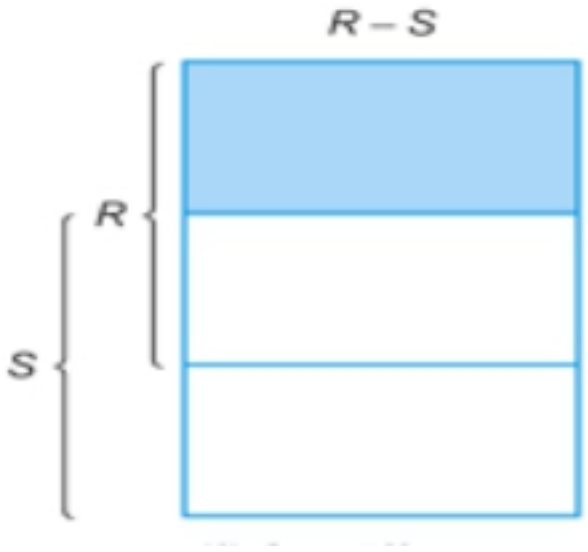
city
London
Aberdeen
Glasgow
Bristol

List all cities where there is either a branch office or a property.

```
(SELECT city
FROM Branch
WHERE city IS NOT NULL) UNION
(SELECT city
FROM PropertyForRent
WHERE city IS NOT NULL);
```

Set Difference

- **R – S**
  - **Defines a relation consisting of the tuples that are in relation R, but not in S.**
  - **R and S must be union-compatible.**



Example - Set Difference

- List all cities where there is a branch office **but no** properties for rent.

$$\Pi_{city}(Branch) - \Pi_{city}(PropertyForRent)$$

Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

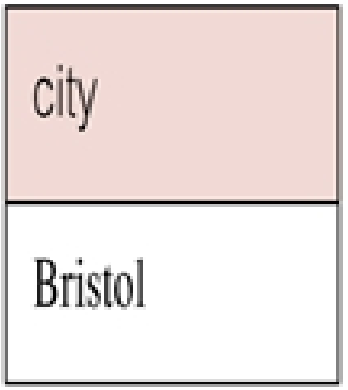
PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

city
Bristol

List of all cities where there is a branch office but no properties.

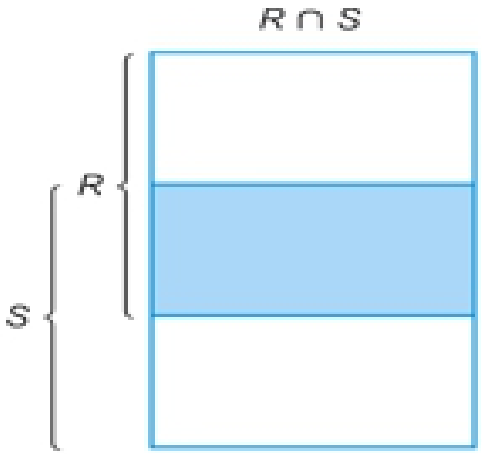
```
(SELECT city FROM Branch)
EXCEPT
(SELECT city FROM PropertyForRent);
```



city
Bristol

Intersection

- **R ∩ S**
  - **Defines a relation consisting of the set of all tuples that are in both R and S.**
  - **R and S must be union-compatible.**



INTERSECT

- The management wants to know which customers are dealing with both companies. Customers which deal with both companies are considered as not loyal.
- Use **INTERSECT** to combine rows from two queries, returning only the rows that appear in both sets.

Example - Intersection

- List all cities where there is both a branch office and at least one property for rent.

$$\Pi_{city}(Branch) \cap \Pi_{city}(PropertyForRent)$$

Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

city
Aberdeen
London
Glasgow

List all cities where there is both a branch office and a property.

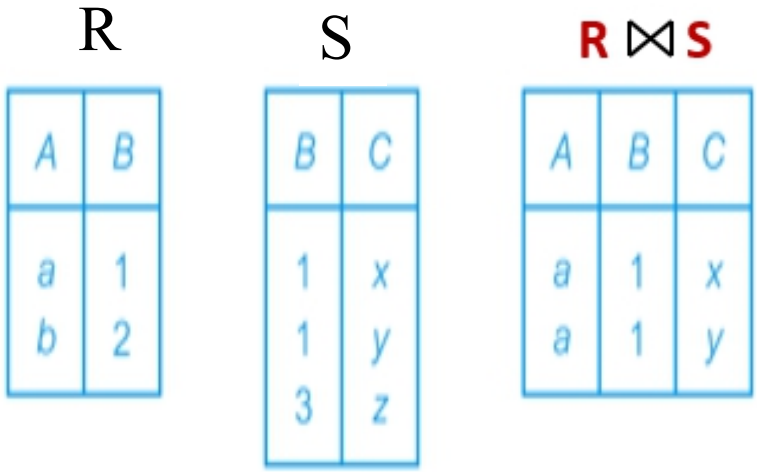
```
(SELECT city FROM Branch)
INTERSECT
(SELECT city FROM PropertyForRent);
```

Join Operations

- **Various forms of join operation**
  - **Natural join**
  - **Left Outer join**
  - **Right Outer join**
  - **Semijoin**

Natural join

- **R ⋈ S**
  - The result of the natural join is the set of all combinations of tuples in *R* and *S* that are **equal on their common attribute names**



R

A	B
a	1
b	2

S

B	C
1	x
1	y
3	z

R ⋈ S

A	B	C
a	1	x
a	1	y

Example - Natural Join

- List the clientno, names and comments of all clients who have viewed a property for rent.

$(\Pi_{clientNo,\ fName,\ lName}(Client)) \bowtie (\Pi_{clientNo,\ propertyNo,\ comment}(Viewing))$

Client

clientNo	fName	lName	telNo	prefType	maxRent
CR76	John	Kay	0207-774-5632	Flat	425
CR56	Aline	Stewart	0141-848-1825	Flat	350
CR74	Mike	Ritchie	01475-392178	House	750
CR62	Mary	Tregear	01224-196720	Flat	600

Viewing

clientNo	propertyNo	viewDate	comment
CR56	PA14	24-May-04	too small
CR76	PG4	20-Apr-04	too remote
CR56	PG4	26-May-04	
CR62	PA14	14-May-04	no dining room
CR56	PG36	28-Apr-04	

clientNo	fName	lName	propertyNo	comment
CR76	John	Kay	PG4	too remote
CR56	Aline	Stewart	PA14	too small
CR56	Aline	Stewart	PG4	
CR56	Aline	Stewart	PG36	
CR62	Mary	Tregear	PA14	no dining room

Outer Join

- **To display rows in the result that do not have matching values in the join column, use Outer join.**
  - **Left outer join** $R \bowtie S$
  - **Right outer join** $R \bowtie S$

Left Outer Join

Left outer join $R \mathbin{⟕} S$

– **(Left) outer join is join in which tuples from R that do not have matching values in common columns of S are also included in result relation.**

R

A	B
a	1
b	2

S

B	C
1	x
1	y
3	z

$R \mathbin{⟕} S$

A	B	C
a	1	x
a	1	y
b	2	

Example - Left Outer Join

Employee

Name	EmpId	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales
Tim	1123	Executive

Dept

DeptName	Manager
Sales	Harriet
Production	Charles

Employee ⟕ *Dept*

Name	EmpId	DeptName	Manager
Harry	3415	Finance	ω
Sally	2241	Sales	Harriet
George	3401	Finance	ω
Harriet	2202	Sales	Harriet
Tim	1123	Executive	ω

Example - Left Outer join

Produce a status report on property viewings.

$\Pi_{propertyNo,\ street,\ city}(PropertyForRent) \ ⟕ \ Viewing$

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Viewing

clientNo	propertyNo	viewDate	comment
CR56	PA14	24-May-04	too small
CR76	PG4	20-Apr-04	too remote
CR56	PG4	26-May-04	
CR62	PA14	14-May-04	no dining room
CR56	PG36	28-Apr-04	

propertyNo	street	city	clientNo	viewDate	comment
PA14	16 Holhead	Aberdeen	CR56	24-May-01	too small
PA14	16 Holhead	Aberdeen	CR62	14-May-01	no dining room
PL94	6 Argyll St	London	null	null	null
PG4	6 Lawrence St	Glasgow	CR76	20-Apr-01	too remote
PG4	6 Lawrence St	Glasgow	CR56	26-May-01	
PG36	2 Manor Rd	Glasgow	CR56	28-Apr-01	
PG21	18 Dale Rd	Glasgow	null	null	null
PG16	5 Novar Dr	Glasgow	null	null	null

List branches and properties that are in same city along with any unmatched branches.

```
SELECT b.*, p.*
FROM Branch1 b LEFT JOIN
     PropertyForRent1 p ON b.bCity = p.pCity;
```

Right Outer Join

- **R ⟖ S**

The right outer join of relations *R* and *S* is written as *R* ⟖ *S*.

The result of the right outer join is the set of all combinations of tuples in *R* and *S* that are equal on their common attribute names, in addition to tuples in *S* that have no matching tuples in *R*.

Example - Right Outer Join

Employee

Name	EmpId	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales
Tim	1123	Executive

Dept

DeptName	Manager
Sales	Harriet
Production	Charles

Employee ⟖ *Dept*

Name	EmpId	DeptName	Manager
Sally	2241	Sales	Harriet
Harriet	2202	Sales	Harriet
ω	ω	Production	Charles

List branches and properties in same city and any unmatched properties.

```
SELECT b.*, p.*
FROM Branch1 b RIGHT JOIN
  PropertyForRent1 p ON b.bCity = p.pCity;
```

Semijoin

- $R \triangleright_F S$
  - Defines a relation that contains the tuples of R that participate in the join of R with S.

R

A	B
a	1
b	2

S

B	C
1	x
1	y
3	z

$R \triangleright_B S$

A	B
a	1

Example - Semijoin

- **List complete details of all staff who work at the branch in Glasgow.**

$$Staff \triangleright_{Staff.branchNo=Branch.branchNo}(\sigma_{city='Glasgow'}(Branch))$$

Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24- Mar-58	18000	B003
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003

Find all staff who work in a Glasgow branch.

```
SELECT *
FROM Staff s
WHERE EXISTS
   (SELECT *
    FROM Branch b
    WHERE s.branchNo = b.branchNo AND
                city = 'Glasgow );
```

Division

- **R ÷ S**
  - Defines a relation over the attributes B that consists of set of tuples from R that match combination of *every* tuple in S.

R

A	B
a	1
a	2
b	1
b	2
c	1

S

B
1
2

R ÷ S

A
a
b

Example of Division

Identify students who has completed Database1 and Database 2

Completed

Student	Task
Fred	Database1
Fred	Database2
Fred	Compiler1
Eugene	Database1
Eugene	Compiler1
Sarah	Database1
Sarah	Database2

DBProject

Task
Database1
Database2

Completed ÷ ***DBProject***

Student
Fred
Sarah

Example - Division

- Identify all clients who have viewed **all** properties with three rooms.

$$(\Pi_{\text{clientNo, propertyNo}}(\text{Viewing})) \div (\Pi_{\text{propertyNo}}(\sigma_{\text{rooms = 3}}(\text{PropertyForRent})))$$

$\Pi_{\text{clientNo,propertyNo}}(\text{Viewing})$

clientNo	propertyNo
CR56	PA14
CR76	PG4
CR56	PG4
CR62	PA14
CR56	PG36

$\Pi_{\text{propertyNo}}(\sigma_{\text{rooms=3}}(\text{PropertyForRent}))$

propertyNo
PG4
PG36

RESULT

clientNo
CR56

Aggregate Operations

- $\mathfrak{I}_{AL}(R)$
  - Applies aggregate function list, **AL**, to R to define a relation over the aggregate list.
  - **AL** contains one or more (<aggregate_function>, <attribute>) pairs .
- Main aggregate functions are: COUNT, SUM, AVG, MIN, and MAX.

Eg:

$$\mathfrak{I}_{\text{COUNT staffNo, SUM salary}}(\text{Staff})$$

Find number of Managers and sum of their salaries.

```
SELECT COUNT(staffNo) AS myCount,
       SUM(salary) AS mySum
FROM Staff
WHERE position = 'Manager';
```

myCount	mySum
2	54000.00

Example – Aggregate Operations

- How many properties cost more than £350 per month to rent?

$$\rho_R(\text{myCount})\ \Im_{\text{COUNT propertyNo}}\ (\sigma_{\text{rent} > 350}\ (\text{PropertyForRent}))$$

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

myCount
5

Grouping Operation

- $_{GA}\mathfrak{I}_{AL}(R)$
  - **Groups tuples of R by grouping attributes, GA, and then applies aggregate function list, AL, to define a new relation.**
  - **AL contains one or more (<aggregate_function>, <attribute>) pairs.**
  - **Resulting relation contains the grouping attributes, GA, along with results of each of the aggregate functions.**

Example – Grouping Operation

- Find the number of staff working in each branch and the sum of their salaries.

$$\rho_R(\text{branchNo, myCount, mySum})$$

$$\text{branchNo}\ \Im\ \text{COUNT staffNo, SUM salary}\ (\text{Staff})$$

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

branchNo	myCount	mySum
B003	3	54000
B005	2	39000
B007	1	9000

Find number of staff in each branch and their total salaries.

```
SELECT branchNo,
        COUNT(staffNo) AS myCount,
        SUM(salary) AS mySum
  FROM Staff
  GROUP BY branchNo
  ORDER BY branchNo;
```

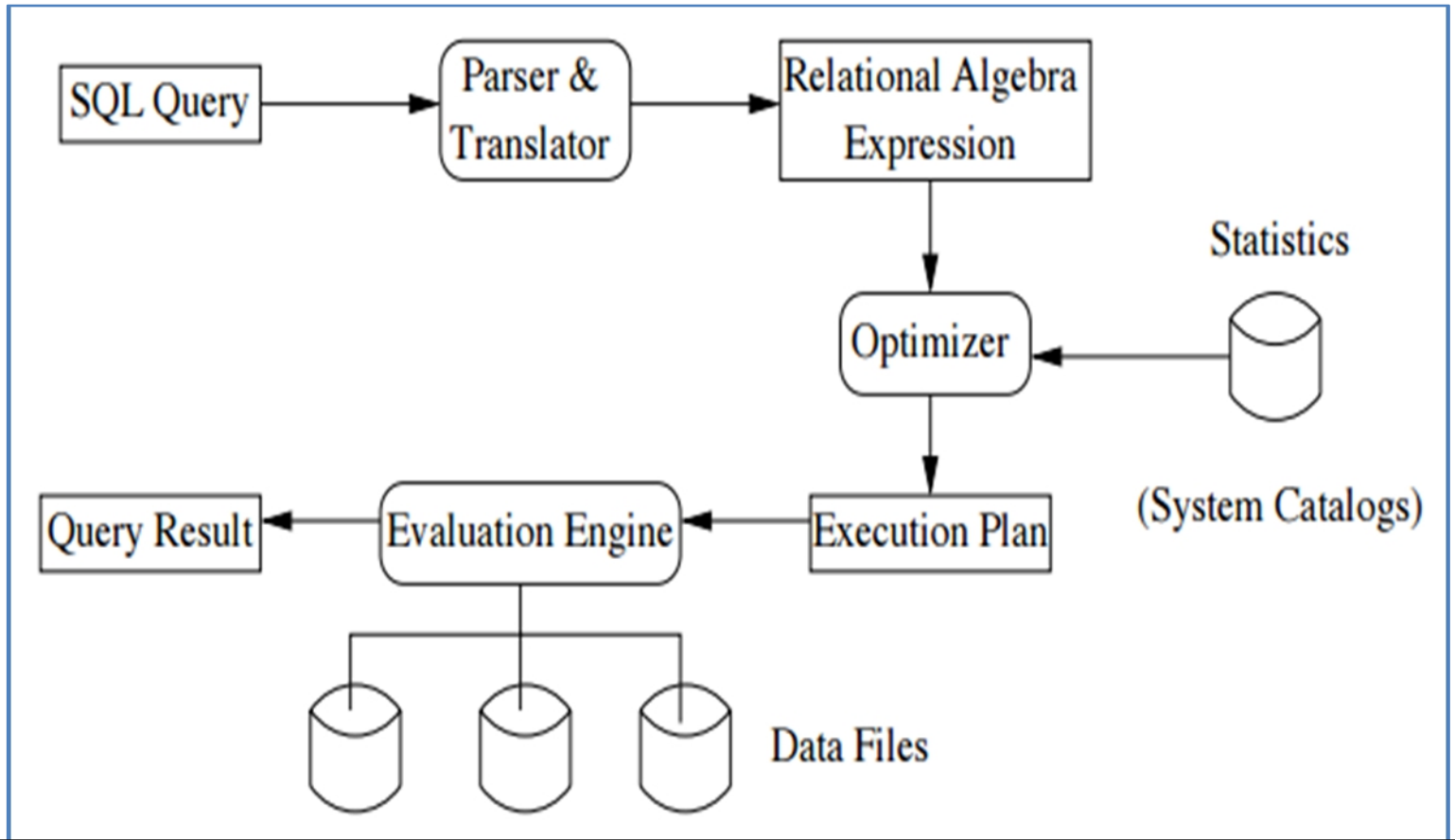
2. Query Optimization

Basic Steps in Query Processing

1. Parsing and translation

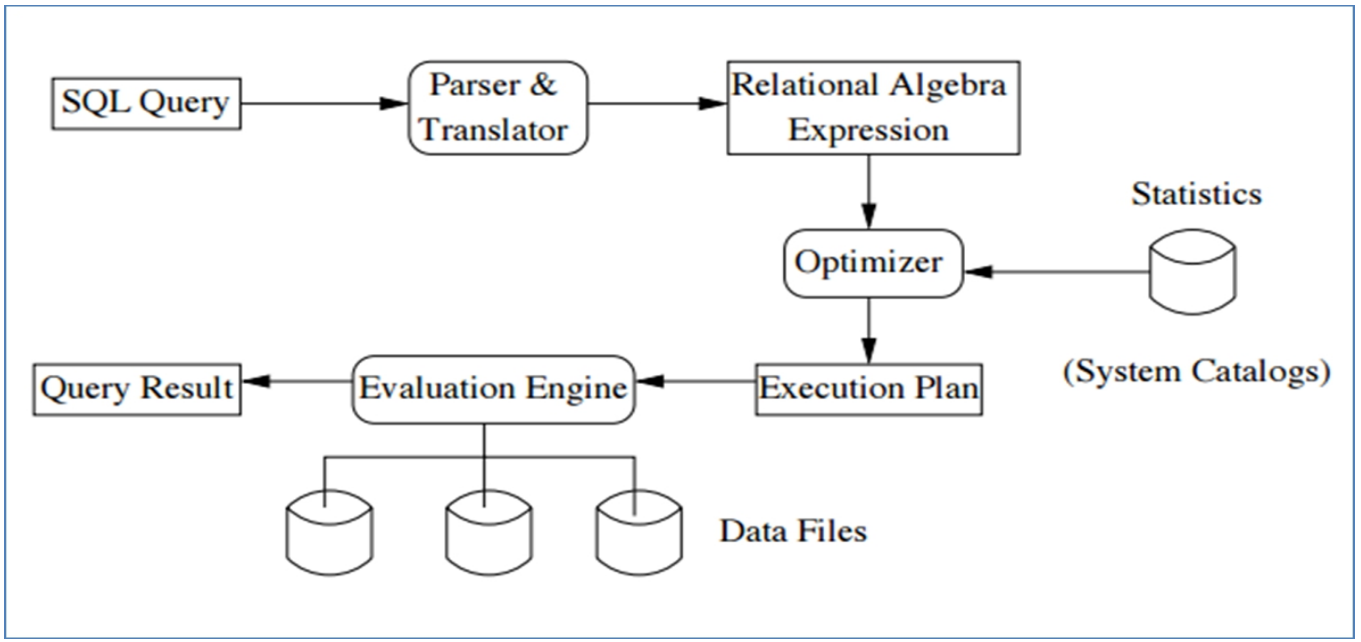
2. Optimization

3. Evaluation



Query Optimization

- Activity of choosing an efficient execution plan for processing an SQL query.
- As there are many equivalent transformations of same high-level query, aim of QO is to
  - ✓ minimize resource usage (generally refers to total execution time of the query)



Example

Find all Managers who work at a London branch.

```
SELECT *
  FROM Staff s, Branch b
  WHERE s.branchNo = b.branchNo AND
  (s.position = 'Manager' AND b.city = 'London');
```

Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Example

Find all Managers who work at a London branch.

```
SELECT *
  FROM Staff s, Branch b
  WHERE s.branchNo = b.branchNo AND
  (s.position = 'Manager' AND b.city = 'London');
```

Three equivalent RA queries are:

(1) $\sigma_{(position='Manager') \wedge (city='London') \wedge (Staff.branchNo=Branch.branchNo)}$ (Staff X Branch)

(2) $\sigma_{(position='Manager') \wedge (city='London')}$ (Staff $\bowtie_{Staff.branchNo=Branch.branchNo}$ Branch)

(3) $(\sigma_{position='Manager'}(Staff)) \bowtie_{Staff.branchNo=Branch.branchNo} (\sigma_{city='London'}(Branch))$

- **Assume:**
  - 1000 tuples in Staff; 50 tuples in Branch;
  - 50 Managers; 5 London branches;
  - tuples are accessed one at a time.
  - Results of intermediate operations are stored on disk.
- Comparison is based on the no. of disk accesses

Three equivalent RA queries are:

(1) $\sigma_{(position='Manager') \wedge (city='London') \wedge (Staff.branchNo=Branch.branchNo)}$ (Staff X Branch)

$(1000 + 50) + 2*(1000 * 50) = 101\ 050$

(2) $\sigma_{(position='Manager') \wedge (city='London')}$ (Staff $\bowtie_{Staff.branchNo=Branch.branchNo}$ Branch)

$2*1000 + (1000 + 50) = 3\ 050$

(3) $(\sigma_{position='Manager'}(Staff)) \bowtie_{Staff.branchNo=Branch.branchNo} (\sigma_{city='London'}(Branch))$

$1000 + 2*50 + 5 + (50 + 5) = 1\ 160$

(1) $\sigma_{(position='Manager') \wedge (city='London') \wedge (Staff.branchNo=Branch.branchNo)}$ (Staff X Branch)

(1000 + 50) + 2*(1000 * 50) = 101 050

(2) $\sigma_{(position='Manager') \wedge (city='London')}$(Staff $\bowtie_{Staff.branchNo=Branch.branchNo}$ Branch)

2*1000 + (1000 + 50) = 3 050

(3) ($\sigma_{position='Manager'}$(Staff)) $\bowtie_{Staff.branchNo=Branch.branchNo}$ ($\sigma_{city='London'}$ (Branch))

1000 + 2*50 + 5 + (50 + 5) = 1 160

(1) Read 1000 Staff, Read 50 Branch
Write 1000 x 50 records, read 1000 x 50 to search

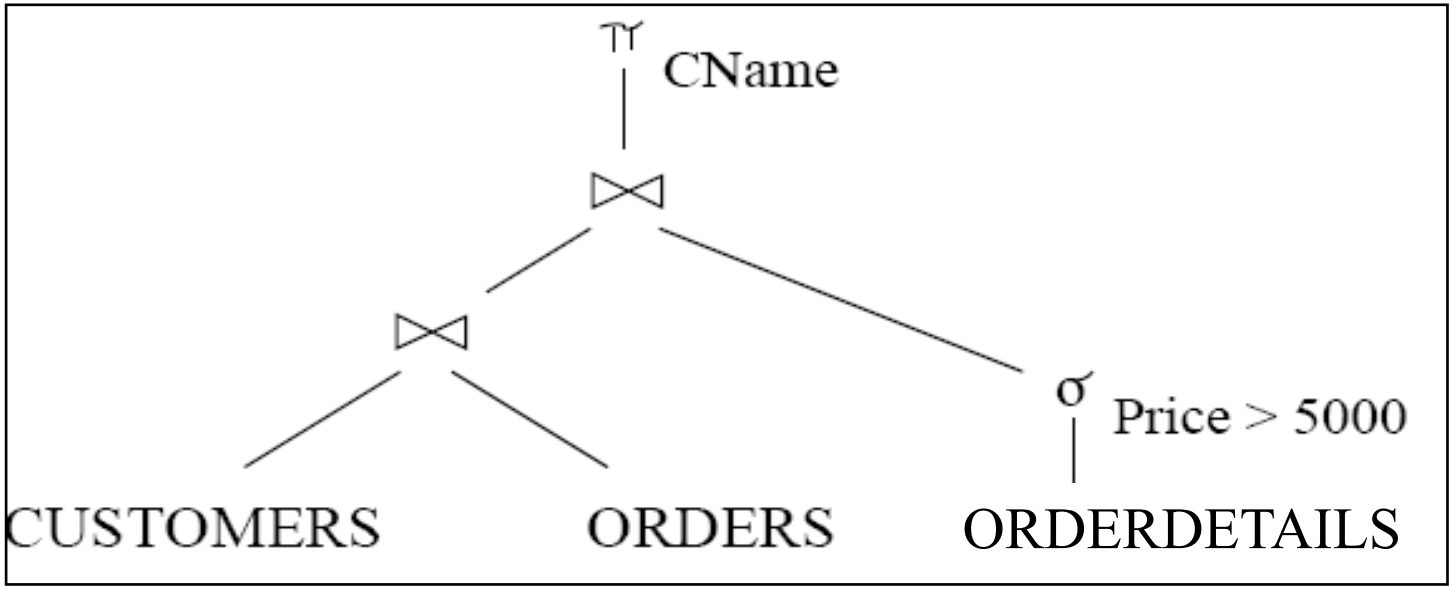
(2) Read 1000 Staff, Read 50 Branch
Join creates 1000 rows of Staff-Branch data; write & read

(3) Read 1000 Staff, write 50 managers
Read 50 Branch, write 5 branches in London
Read 50 managers & 5 branches to create final join

Evaluation of Expressions

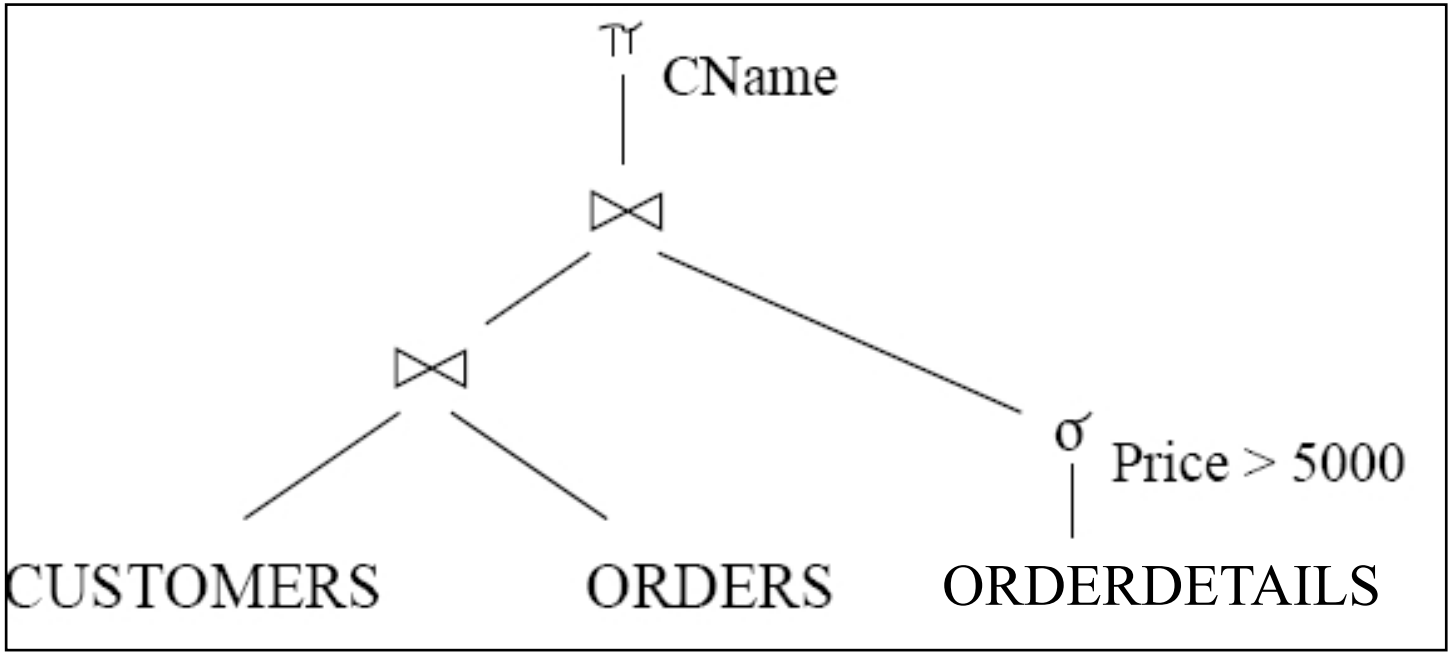
- **Strategy 1: Materialization.**
  - Evaluate one operation at a time, starting at the lowest level. Use intermediate results materialized in temporary relations to evaluate next level operation(s).
  - Generating and saving temporary files on disk is time consuming and expensive.
- **Strategy 2: Pipelining.**
  - Evaluate several operations simultaneously, and pass the result (tuple- or block-wise) on to the next operation.
  - Avoid constructing temporary results as much as possible.
  - Also known as stream-based or on-the-fly processing.

Materialization



1. Compute and store $\sigma_{Price>5000}$ (ORDERDETAILS);
2. Compute and store join of CUSTOMERS and ORDERS;
3. Join the two materialized relations and project on to CName.

Pipelining



Once a row from ORDERDETAILS satisfying selection condition has been found, pass it on to the join.

Similarly, don't store result of (final) join, but pass tuples directly to projection.

Exercise

The following tables form part of a Library database held in an RDBMS:

Book (ISBN, title, edition, year)
BookCopy (copyNo, ISBN*, available)
Borrower (borrowerNo, borrowerName, borrowerAddr)
BookLoan (copyNo*, dateOut, dateDue, borrowerNo*)

1. List all book titles.
2. List all borrower details.
3. List all book titles published in the year 2019.
4. List all copies of book titles that are available for borrowing.
5. List all copies of the book title “Lord of the Rings” that are available for borrowing.
6. List the names of borrowers who currently have the book title “Lord of the Rings” on loan.
7. List the names of borrowers with overdue books.
8. Produce a report of book titles that have been borrowed by “Peter Bloomfield”.
9. Produce a report with the details of borrowers who currently have books overdue

References

- *Database Systems: A Practical Approach to Design, Implementation and Management*. Connolly, T. M. and Begg, C. E. Pearson.

- Fundamentals of Database Systems. Elmasri, R. and Navathe, S.B. Pearson.