~~Week 5~~ WEEK 12

Relational Algebra and SQL
(Cartesian Product & Join Operation)

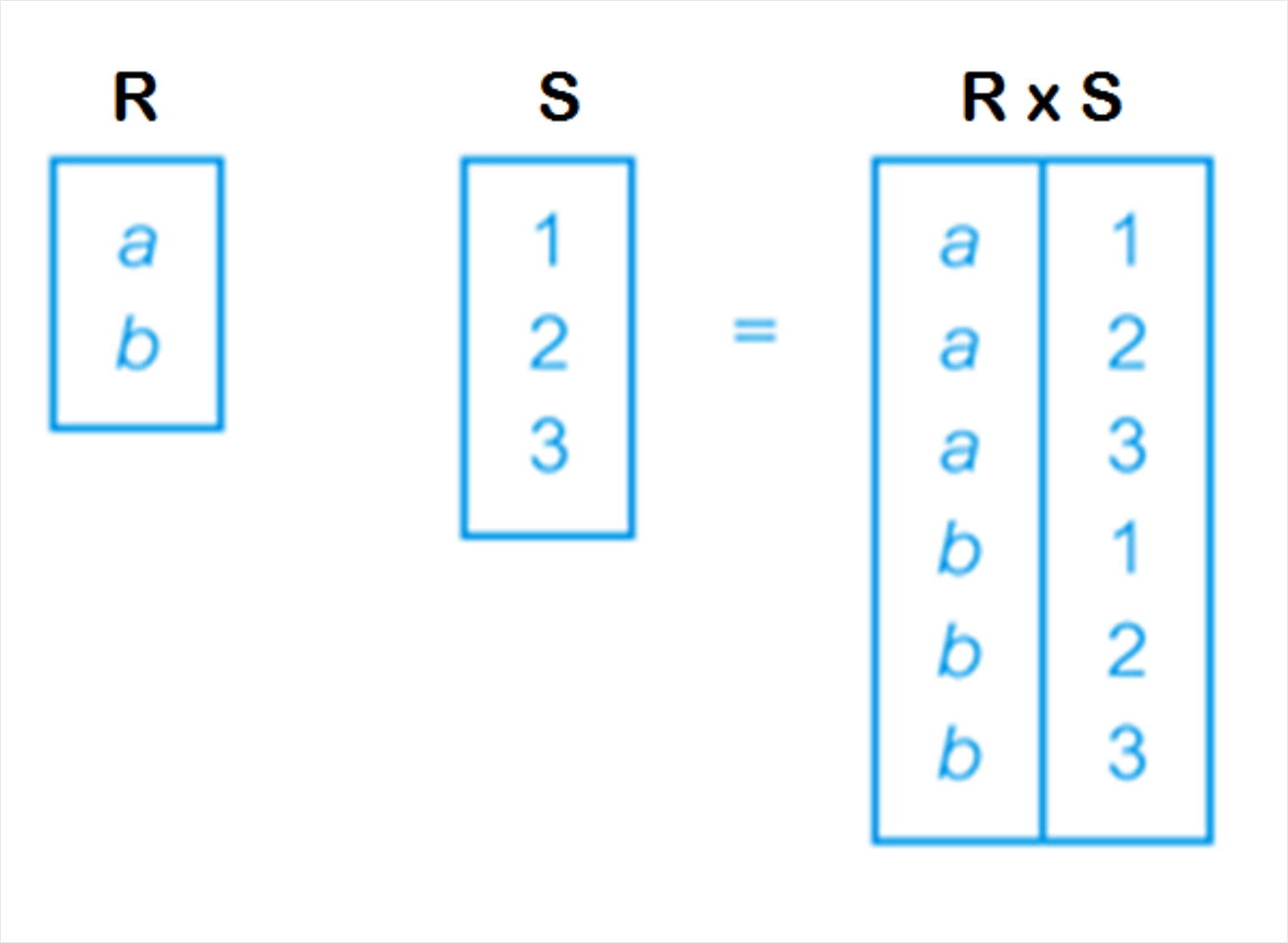
Objectives

- Relational Algebra : Cartesian Product
- Relational Algebra : Join Operation – Theta Join, Equijoin and Natural Join
- SQL on Cartesian Product, Theta Join, Equijoin and Natural Join.

Cartesian product

- **R x S**
  - **Defines a relation that is the concatenation of every tuple of relation R with every tuple of relation S.**

Cartesian product



Example - Cartesian product

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

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

client.clientNo	fName	lName
CR76	John	Kay
CR56	Aline	Stewart
CR74	Mike	Ritchie
CR62	Mary	Tregear

X

Viewing.clientNo	propertyNo	comment
CR56	PA14	too small
CR76	PG4	too remote
CR56	PG4	
CR62	PA14	no dining room
CR56	PG36	

Example - Cartesian product

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

client.clientNo	fName	lName	Viewing.clientNo	propertyNo	comment
CR76	John	Kay	CR56	PA14	too small
CR76	John	Kay	CR76	PG4	too remote
CR76	John	Kay	CR56	PG4	
CR76	John	Kay	CR62	PA14	no dining room
CR76	John	Kay	CR56	PG36	
CR56	Aline	Stewart	CR56	PA14	too small
CR56	Aline	Stewart	CR76	PG4	too remote
CR56	Aline	Stewart	CR56	PG4	
CR56	Aline	Stewart	CR62	PA14	no dining room
CR56	Aline	Stewart	CR56	PG36	
CR74	Mike	Ritchie	CR56	PA14	too small
CR74	Mike	Ritchie	CR76	PG4	too remote
CR74	Mike	Ritchie	CR56	PG4	
CR74	Mike	Ritchie	CR62	PA14	no dining room
CR74	Mike	Ritchie	CR56	PG36	
CR62	Mary	Tregear	CR56	PA14	too small
CR62	Mary	Tregear	CR76	PG4	too remote
CR62	Mary	Tregear	CR56	PG4	
CR62	Mary	Tregear	CR62	PA14	no dining room
CR62	Mary	Tregear	CR56	PG36	

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

client.clientNo	fName	lName	Viewing.clientNo	propertyNo	comment
CR76	John	Kay	CR56	PA14	too small
CR76	John	Kay	CR76	PG4	too remote
CR76	John	Kay	CR56	PG4	
CR76	John	Kay	CR62	PA14	no dining room
CR76	John	Kay	CR56	PG36	
CR56	Aline	Stewart	CR56	PA14	too small
CR56	Aline	Stewart	CR76	PG4	too remote
CR56	Aline	Stewart	CR56	PG4	
CR56	Aline	Stewart	CR62	PA14	no dining room
CR56	Aline	Stewart	CR56	PG36	
CR74	Mike	Ritchie	CR56	PA14	too small
CR74	Mike	Ritchie	CR76	PG4	too remote
CR74	Mike	Ritchie	CR56	PG4	
CR74	Mike	Ritchie	CR62	PA14	no dining room
CR74	Mike	Ritchie	CR56	PG36	
CR62	Mary	Tregear	CR56	PA14	too small
CR62	Mary	Tregear	CR76	PG4	too remote
CR62	Mary	Tregear	CR56	PG4	
CR62	Mary	Tregear	CR62	PA14	no dining room
CR62	Mary	Tregear	CR56	PG36	

Example - Cartesian product and Selection

- Use selection operation to extract those tuples where Client.clientNo = Viewing.clientNo.

$$\sigma_{\text{Client.clientNo} = \text{Viewing.clientNo}}((\Pi_{\text{clientNo, fName, lName}}(\text{Client})) \text{ X } (\Pi_{\text{clientNo, propertyNo, comment}}(\text{Viewing})))$$

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

- Cartesian product and Selection can be reduced to a single operation called a *Join*.

Join Operations

- **Join is a derivative of Cartesian product.**
- **Equivalent to performing a Selection, using join predicate as selection formula, over Cartesian product of the two operand relations.**
- **One of the most difficult operations to implement efficiently in an RDBMS and one reason why RDBMSs have intrinsic performance problems.**

Join Operations

- **Various forms of join operation**
  - **Inner Join**
    - **Theta join**
    - **Equijoin (a particular type of Theta join)**
    - **Natural join**
  - **Outer join**
  - **Semijoin**

Theta join (θ-join)

- $R \bowtie_F S$
  - Defines a relation that contains tuples satisfying the predicate F from the Cartesian product of R and S.
  - The predicate F is of the form $R.a_i \ \theta \ S.b_i$ where θ may be one of the comparison operators (<, ≤, >, ≥, =, ≠).

Theta join (θ-join)

- **Can rewrite Theta join using basic Selection and Cartesian product operations.**

$$R \bowtie_F S = \sigma_F(R \times S)$$

- **Degree of a Theta join is sum of degrees of the operand relations R and S. If predicate F contains only equality (=), the term *Equijoin* is used.**

Example - Equijoin

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

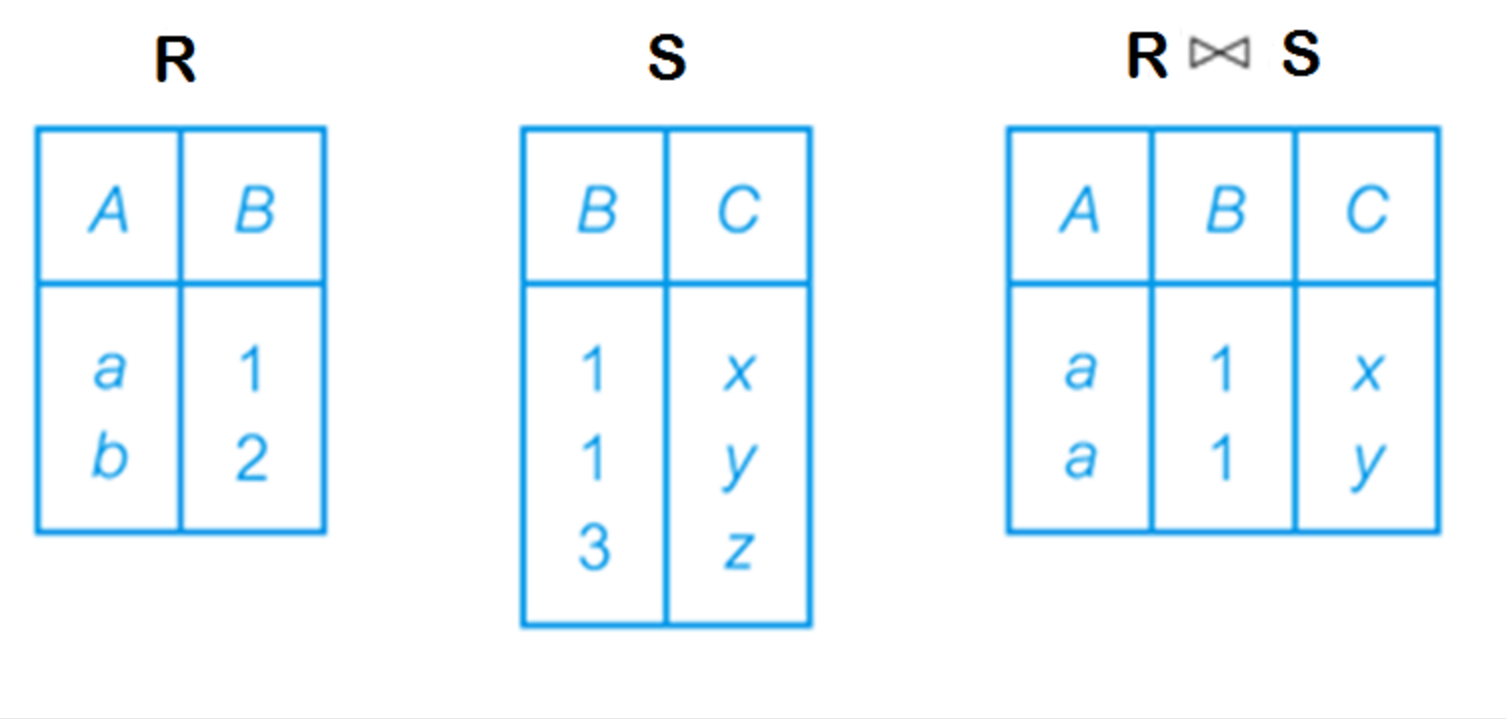
$$(\Pi_{clientNo,\ fName,\ lName}(Client)) \bowtie_{Client.clientNo\ =\ Viewing.clientNo} (\Pi_{clientNo,\ propertyNo,\ comment}(Viewing))$$

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

Natural join

- **R ⋈ S**
  - **An Equijoin of the two relations R and S over all common attributes *x*. One occurrence of each common attribute is eliminated from the result.**

Natural join



Example - Natural join

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

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

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

SQL :

Cartesian Product & Inner Join

Multi-Table Queries

- Can use subqueries provided result columns come from same table.
- If result columns come from more than one table must use a join.
- To perform join, include more than one table in FROM clause.
- Use comma as separator and typically include WHERE clause to specify join column(s).

Multi-Table Queries

- Also possible to use an alias for a table named in FROM clause.
- Alias is separated from table name with a space.
- Alias can be used to qualify column names when there is ambiguity.

Example : Simple Join

List names of all clients who have viewed a property along with any comment supplied.

```
SELECT c.clientNo, fName, lName,
                propertyNo, comment
FROM Client c, Viewing v
WHERE c.clientNo = v.clientNo;
```

Example : Simple Join

- Only those rows from both tables that have identical values in the clientNo columns (c.clientNo = v.clientNo) are included in result.
- Equivalent to equi-join in relational algebra.

Table 5.24 Result table for Example 5.24.

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

Alternative JOIN Constructs

- **SQL provides alternative ways to specify joins:**

  FROM Client c JOIN Viewing v ON c.clientNo = v.clientNo

  FROM Client JOIN Viewing USING clientNo

  FROM Client NATURAL JOIN Viewing

- **In each case, FROM replaces original FROM and WHERE. However, first produces table with two identical clientNo columns.**

Example : Sorting a join

For each branch, list numbers and names of staff who manage properties, and properties they manage.

```
SELECT s.branchNo, s.staffNo, fName, lName,
                propertyNo
FROM Staff s, PropertyForRent p
WHERE s.staffNo = p.staffNo
ORDER BY s.branchNo, s.staffNo, propertyNo;
```

Example : Sorting a join

Table 5.25 Result table for Example 5.25.

branchNo	staffNo	fName	lName	propertyNo
B003	SG14	David	Ford	PG16
B003	SG37	Ann	Beech	PG21
B003	SG37	Ann	Beech	PG36
B005	SL41	Julie	Lee	PL94
B007	SA9	Mary	Howe	PA14

Example : Three Table Join

For each branch, list staff who manage properties, including city in which branch is located and properties they manage.

```
SELECT b.branchNo, b.city, s.staffNo, fName, lName,
       propertyNo
FROM Branch b, Staff s, PropertyForRent p
WHERE b.branchNo = s.branchNo AND
       s.staffNo = p.staffNo
ORDER BY b.branchNo, s.staffNo, propertyNo;
```

Example : Three Table Join

Table 5.26 Result table for Example 5.26.

branchNo	city	staffNo	fName	lName	propertyNo
B003	Glasgow	SG14	David	Ford	PG16
B003	Glasgow	SG37	Ann	Beech	PG21
B003	Glasgow	SG37	Ann	Beech	PG36
B005	London	SL41	Julie	Lee	PL94
B007	Aberdeen	SA9	Mary	Howe	PA14

- **Alternative formulation for FROM and WHERE:**

FROM (Branch b JOIN Staff s USING branchNo) AS bs JOIN PropertyForRent p USING staffNo

Example : Multiple Grouping Columns

Find number of properties handled by each staff member.

```
SELECT s.branchNo, s.staffNo, COUNT(*) AS myCount
FROM Staff s, PropertyForRent p
WHERE s.staffNo = p.staffNo
GROUP BY s.branchNo, s.staffNo
ORDER BY s.branchNo, s.staffNo;
```

Example : Multiple Grouping Columns

branchNo	staffNo	myCount
B003	SG14	1
B003	SG37	2
B005	SL41	1
B007	SA9	1

Computing a Join

Procedure for generating results of a join are:

1. **Form Cartesian product of the tables named in FROM clause.**

2. **If there is a WHERE clause, apply the search condition to each row of the product table, retaining those rows that satisfy the condition.**

3. **For each remaining row, determine value of each item in SELECT list to produce a single row in result table.**

Computing a Join

4. If DISTINCT has been specified, eliminate any duplicate rows from the result table.

6. If there is an ORDER BY clause, sort result table as required.

- SQL provides special format of SELECT for Cartesian product:

```
SELECT [DISTINCT | ALL] {* | columnList}
FROM Table1 CROSS JOIN Table2
```