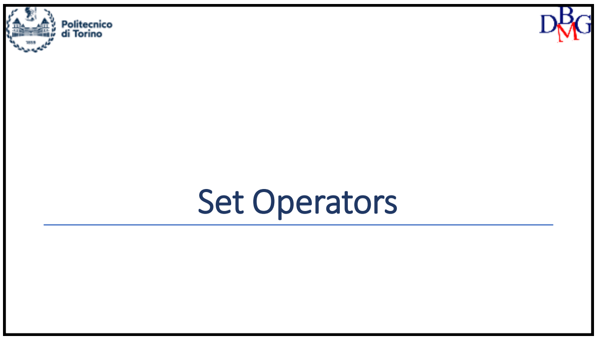
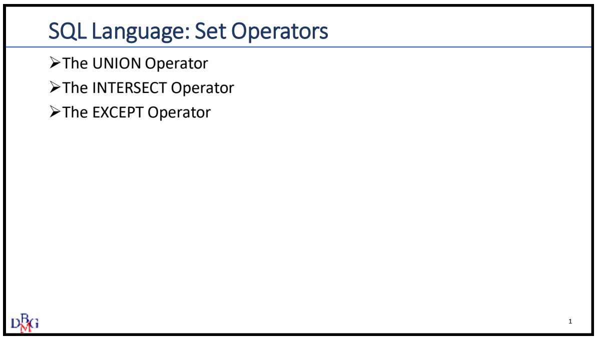


# Set Operators

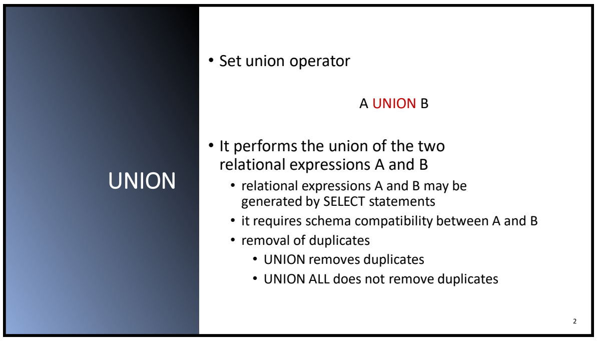
0



## SQL Language: Set Operators

- The UNION Operator
- The INTERSECT Operator
- The EXCEPT Operator

1



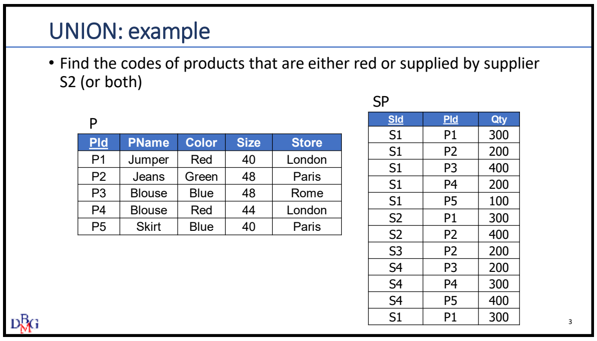
UNION

- Set union operator

A UNION B

- It performs the union of the two relational expressions A and B
  - relational expressions A and B may be generated by SELECT statements
  - it requires schema compatibility between A and B
  - removal of duplicates
    - UNION removes duplicates
    - UNION ALL does not remove duplicates

2



## UNION: example

- Find the codes of products that are either red or supplied by supplier S2 (or both)

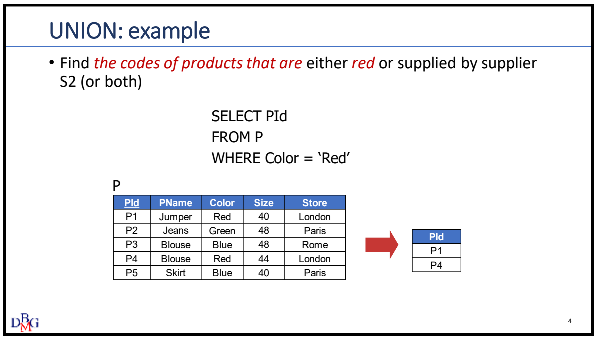
P

| PId | PName | Color | Size | Store |
|---|---|---|---|---|
| P1 | Jumper | Red | 40 | London |
| P2 | Jeans | Green | 48 | Paris |
| P3 | Blouse | Blue | 48 | Rome |
| P4 | Blouse | Red | 44 | London |
| P5 | Skirt | Blue | 40 | Paris |

SP

| SId | PId | Qty |
|---|---|---|
| S1 | P1 | 300 |
| S1 | P2 | 200 |
| S1 | P3 | 400 |
| S1 | P4 | 200 |
| S1 | P5 | 100 |
| S2 | P1 | 300 |
| S2 | P2 | 400 |
| S3 | P2 | 200 |
| S4 | P3 | 200 |
| S4 | P4 | 300 |
| S4 | P5 | 400 |
| S1 | P1 | 300 |

3



## UNION: example

- Find *the codes of products that are* either *red* or supplied by supplier S2 (or both)

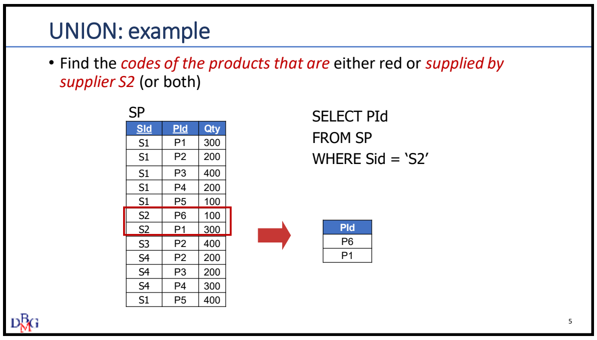
```
SELECT PId
FROM P
WHERE Color = ‘Red’
```

P

| PId | PName | Color | Size | Store |
|---|---|---|---|---|
| P1 | Jumper | Red | 40 | London |
| P2 | Jeans | Green | 48 | Paris |
| P3 | Blouse | Blue | 48 | Rome |
| P4 | Blouse | Red | 44 | London |
| P5 | Skirt | Blue | 40 | Paris |

| PId |
|---|
| P1 |
| P4 |

4



## UNION: example

- Find the *codes of the products that are* either red or *supplied by supplier S2* (or both)

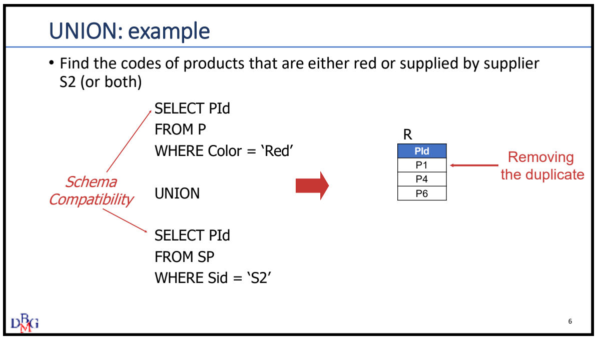
SP

| SId | PId | Qty |
|---|---|---|
| S1 | P1 | 300 |
| S1 | P2 | 200 |
| S1 | P3 | 400 |
| S1 | P4 | 200 |
| S1 | P5 | 100 |
| S2 | P6 | 100 |
| S2 | P1 | 300 |
| S3 | P2 | 400 |
| S4 | P2 | 200 |
| S4 | P3 | 200 |
| S4 | P4 | 300 |
| S1 | P5 | 400 |

```
SELECT PId
FROM SP
WHERE Sid = ‘S2’
```

| PId |
|---|
| P6 |
| P1 |

5



## UNION: example

- Find the codes of products that are either red or supplied by supplier S2 (or both)

```
SELECT PId
FROM P
WHERE Color = ‘Red’

UNION

SELECT PId
FROM SP
WHERE Sid = ‘S2’
```

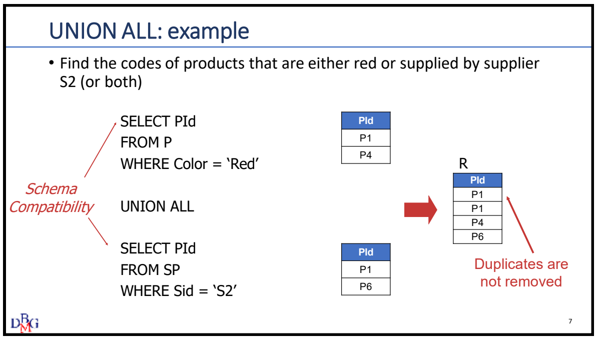
*Schema Compatibility*

R

| PId |
|---|
| P1 |
| P4 |
| P6 |

Removing the duplicate

6



## UNION ALL: example

- Find the codes of products that are either red or supplied by supplier S2 (or both)

```
SELECT PId
FROM P
WHERE Color = ‘Red’

UNION ALL

SELECT PId
FROM SP
WHERE Sid = ‘S2’
```

*Schema Compatibility*

| PId |
|---|
| P1 |
| P4 |

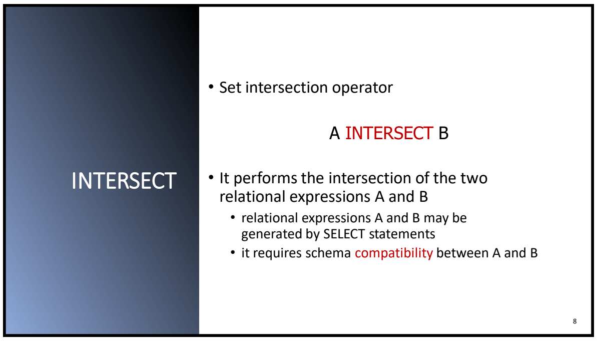
| PId |
|---|
| P1 |
| P6 |

R

| PId |
|---|
| P1 |
| P1 |
| P4 |
| P6 |

Duplicates are not removed

7



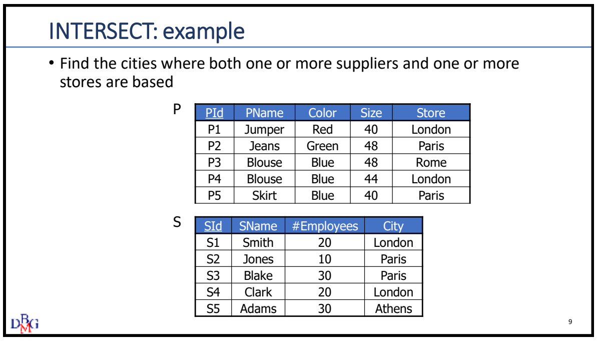
## INTERSECT

- Set intersection operator

A INTERSECT B

- It performs the intersection of the two relational expressions A and B
  - relational expressions A and B may be generated by SELECT statements
  - it requires schema compatibility between A and B

8



## INTERSECT: example

- Find the cities where both one or more suppliers and one or more stores are based

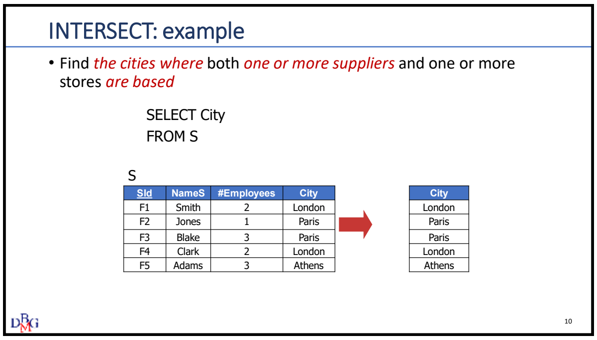
P

| PId | PName | Color | Size | Store |
|---|---|---|---|---|
| P1 | Jumper | Red | 40 | London |
| P2 | Jeans | Green | 48 | Paris |
| P3 | Blouse | Blue | 48 | Rome |
| P4 | Blouse | Blue | 44 | London |
| P5 | Skirt | Blue | 40 | Paris |

S

| SId | SName | #Employees | City |
|---|---|---|---|
| S1 | Smith | 20 | London |
| S2 | Jones | 10 | Paris |
| S3 | Blake | 30 | Paris |
| S4 | Clark | 20 | London |
| S5 | Adams | 30 | Athens |

9



## INTERSECT: example

- Find *the cities where* both *one or more suppliers* and one or more stores *are based*

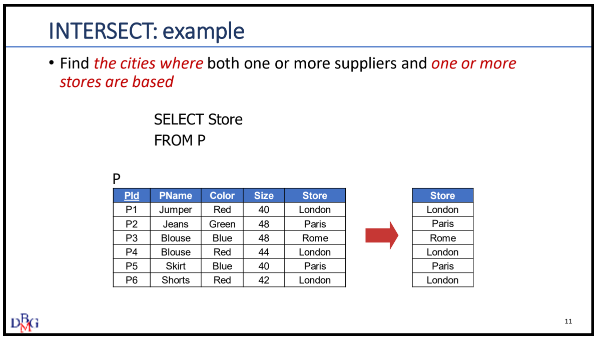
```
SELECT City
FROM S
```

S

| SId | NameS | #Employees | City |
|---|---|---|---|
| F1 | Smith | 2 | London |
| F2 | Jones | 1 | Paris |
| F3 | Blake | 3 | Paris |
| F4 | Clark | 2 | London |
| F5 | Adams | 3 | Athens |

| City |
|---|
| London |
| Paris |
| Paris |
| London |
| Athens |

10



## INTERSECT: example

- Find *the cities where* both one or more suppliers and *one or more stores are based*

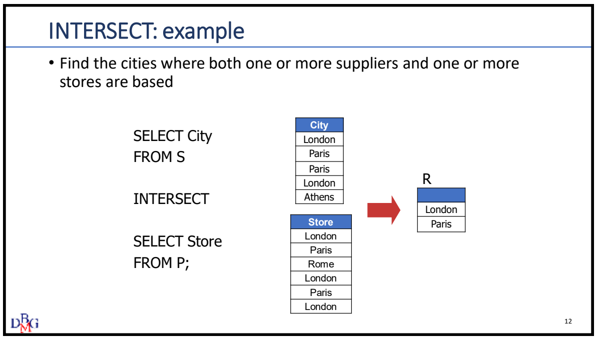
```
SELECT Store
FROM P
```

P

| PId | PName | Color | Size | Store |
|---|---|---|---|---|
| P1 | Jumper | Red | 40 | London |
| P2 | Jeans | Green | 48 | Paris |
| P3 | Blouse | Blue | 48 | Rome |
| P4 | Blouse | Red | 44 | London |
| P5 | Skirt | Blue | 40 | Paris |
| P6 | Shorts | Red | 42 | London |

| Store |
|---|
| London |
| Paris |
| Rome |
| London |
| Paris |
| London |

11



## INTERSECT: example

- Find the cities where both one or more suppliers and one or more stores are based

```
SELECT City
FROM S

INTERSECT

SELECT Store
FROM P;
```

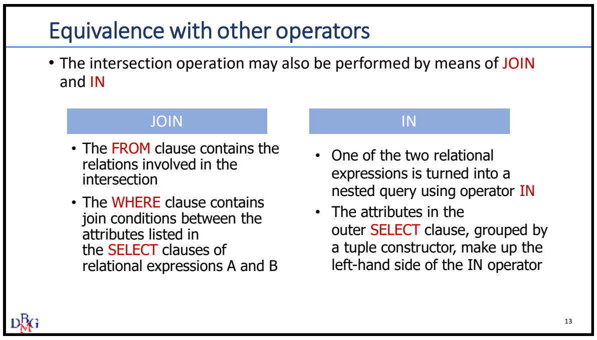
| City |
|---|
| London |
| Paris |
| Paris |
| London |
| Athens |

| Store |
|---|
| London |
| Paris |
| Rome |
| London |
| Paris |
| London |

R

| |
|---|
| London |
| Paris |

12



## Equivalence with other operators

- The intersection operation may also be performed by means of JOIN and IN

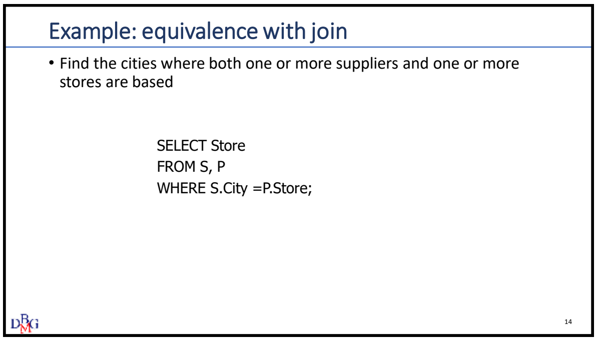
JOIN

- The FROM clause contains the relations involved in the intersection
- The WHERE clause contains join conditions between the attributes listed in the SELECT clauses of relational expressions A and B

IN

- One of the two relational expressions is turned into a nested query using operator IN
- The attributes in the outer SELECT clause, grouped by a tuple constructor, make up the left-hand side of the IN operator

13

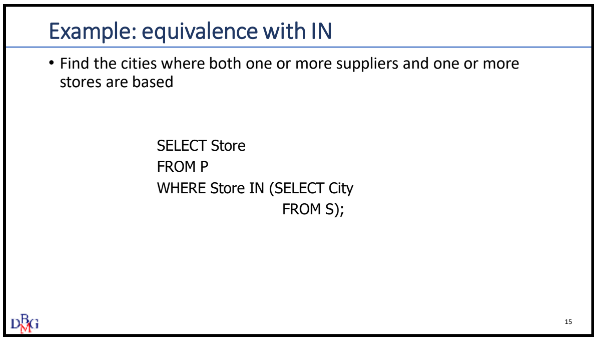


## Example: equivalence with join

- Find the cities where both one or more suppliers and one or more stores are based

```
SELECT Store
FROM S, P
WHERE S.City =P.Store;
```

14

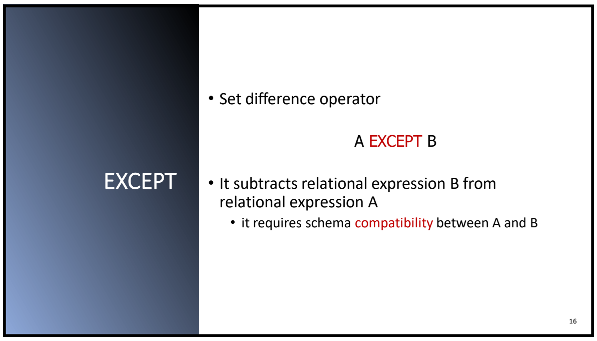


## Example: equivalence with IN

- Find the cities where both one or more suppliers and one or more stores are based

```
SELECT Store
FROM P
WHERE Store IN (SELECT City
                FROM S);
```

15



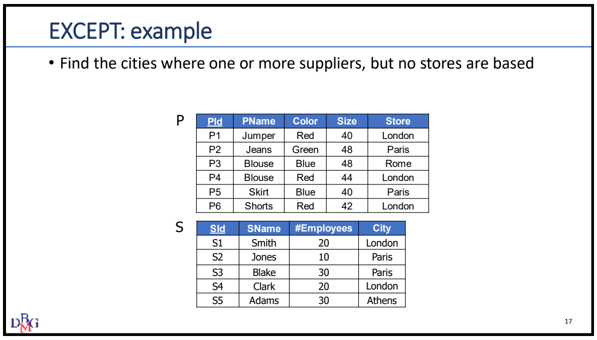
## EXCEPT

- Set difference operator

A EXCEPT B

- It subtracts relational expression B from relational expression A
  - it requires schema compatibility between A and B

16



## EXCEPT: example

- Find the cities where one or more suppliers, but no stores are based

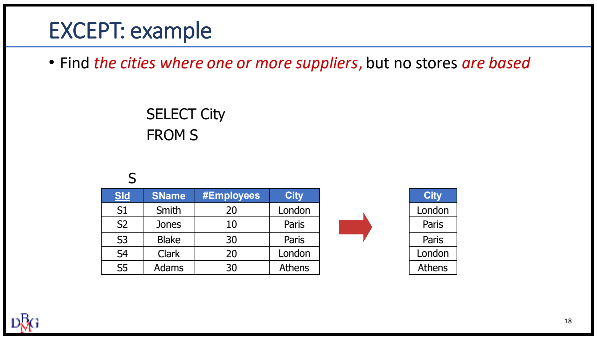
P

| PId | PName | Color | Size | Store |
|---|---|---|---|---|
| P1 | Jumper | Red | 40 | London |
| P2 | Jeans | Green | 48 | Paris |
| P3 | Blouse | Blue | 48 | Rome |
| P4 | Blouse | Red | 44 | London |
| P5 | Skirt | Blue | 40 | Paris |
| P6 | Shorts | Red | 42 | London |

S

| SId | SName | #Employees | City |
|---|---|---|---|
| S1 | Smith | 20 | London |
| S2 | Jones | 10 | Paris |
| S3 | Blake | 30 | Paris |
| S4 | Clark | 20 | London |
| S5 | Adams | 30 | Athens |

17



## EXCEPT: example

- Find *the cities where one or more suppliers,* but no stores *are based*

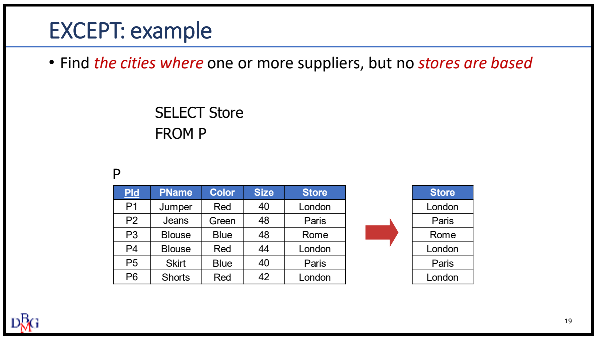
```
SELECT City
FROM S
```

S

| SId | SName | #Employees | City |
|---|---|---|---|
| S1 | Smith | 20 | London |
| S2 | Jones | 10 | Paris |
| S3 | Blake | 30 | Paris |
| S4 | Clark | 20 | London |
| S5 | Adams | 30 | Athens |

| City |
|---|
| London |
| Paris |
| Paris |
| London |
| Athens |

18



## EXCEPT: example

- Find *the cities where* one or more suppliers, but no *stores are based*

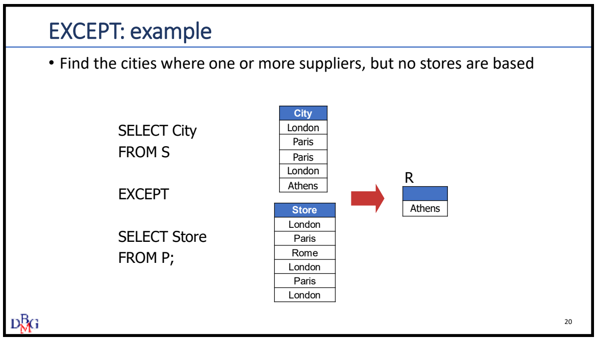
```
SELECT Store
FROM P
```

P

| PId | PName | Color | Size | Store |
|---|---|---|---|---|
| P1 | Jumper | Red | 40 | London |
| P2 | Jeans | Green | 48 | Paris |
| P3 | Blouse | Blue | 48 | Rome |
| P4 | Blouse | Red | 44 | London |
| P5 | Skirt | Blue | 40 | Paris |
| P6 | Shorts | Red | 42 | London |

| Store |
|---|
| London |
| Paris |
| Rome |
| London |
| Paris |
| London |

19



## EXCEPT: example

- Find the cities where one or more suppliers, but no stores are based

```
SELECT City
FROM S

EXCEPT

SELECT Store
FROM P;
```

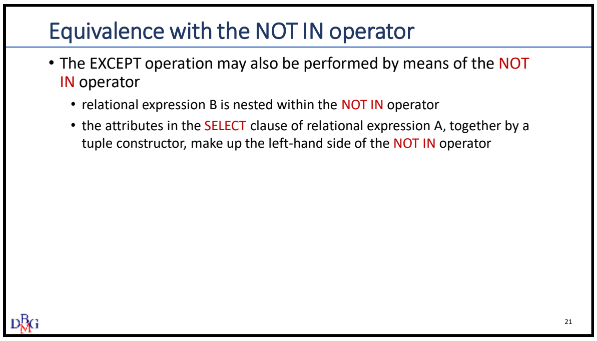
| City |
|---|
| London |
| Paris |
| Paris |
| London |
| Athens |

| Store |
|---|
| London |
| Paris |
| Rome |
| London |
| Paris |
| London |

R

| |
|---|
| Athens |

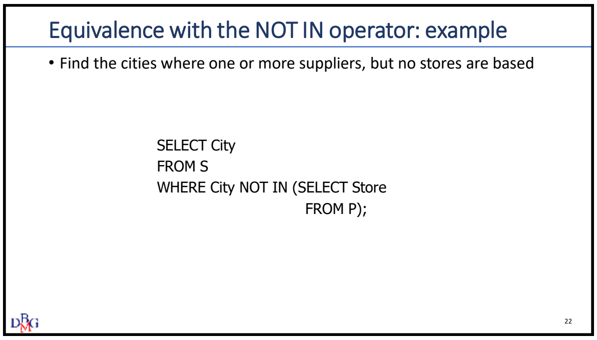
20



## Equivalence with the NOT IN operator

- The EXCEPT operation may also be performed by means of the NOT IN operator
  - relational expression B is nested within the NOT IN operator
  - the attributes in the SELECT clause of relational expression A, together by a tuple constructor, make up the left-hand side of the NOT IN operator

21



## Equivalence with the NOT IN operator: example

- Find the cities where one or more suppliers, but no stores are based

```
SELECT City
FROM S
WHERE City NOT IN (SELECT Store
                   FROM P);
```

22