

Database Course

Ahmad Yoosofan

University of Kashan

Data or Quantitative data

Quantitative data are

- measures of values or counts and are expressed as numbers.
- data about numeric variables (e.g. how many; how much; or how often).
- measures of 'types' and may be represented by a name, symbol, or a number code.
- data about categorical variables (e.g. what type) `abs.gov.au` .

Quantitative Data Examples

- I updated my phone *6 times* in a quarter.
- My teenager grew by *3 inches* last year.
- *83 people* downloaded the latest mobile application.
- My aunt lost *18 pounds* last year.
- *150 respondents* were of the opinion that the new product feature will not be successful.
- There will be *30% increase* in revenue with the inclusion of a new product.
- *500 people* attended the seminar.
- *54% people* prefer shopping online instead of going to the mall.
- She has *10 holidays* in this year.
- Product X costs *\$1000*.

Advantages of Quantitative Data

- Conduct in-depth research: Since quantitative data can be statistically analyzed, it is highly likely that the research will be detailed.
- Minimum bias: There are instances in research, where personal bias is involved which leads to incorrect results. Due to the numerical nature of quantitative data, the personal bias is reduced to a great extent.
- Accurate results: As the results obtained are objective in nature, they are extremely accurate.

Models and their Usage

- What is a model?
- Why do we need model?
- A little history of model in mathetamtics
- Real life situations and models

Old models for Storing Data

- Why do we need secondary storage for preserving data?
- Direct Programming with Files (by Old languages like PL/I and Cobol)
- Hierarchical model
- Network model

„ :

<https://mariadb.com/kb/en/database-theory/>

Relational Model

- Table
- Field
- Record

Sales DataBase (1)

<u>product_id</u>	product_name	stock	quantity	<u>customer_id</u>	customer_name
10	Sugar	20	2	32	Ali
11	Salt	100	5	32	Ali
11	Salt	100	6	33	Reza
10	Sugar	20	3	34	Karim

Sales DataBase (2)

Product Table

<u>product_id</u>	product_name	stock
10	Sugar	20
11	Salt	100

Sales DataBase (3)

Customer Table

<u>customer_id</u>	customer_name
32	Ali
33	Reza
34	Karim

Sales DataBase (4)

Sales Table

<u>product_id</u>	<u>customer_id</u>	quantity
10	32	2
11	32	5
11	33	6
10	34	3

Sales DataBase (5)

- Product Table: ([product_id](#), product_name, stock)
- Customer Table: ([customer_id](#), customer_name)
- Sales Table: ([product_id](#), [customer_id](#) , quantity)

Parts Suppliers Database

- P(pn, pname, color, weight, city)
- S(sn, sname, status, city)
- SP(sn, pn, qty)

Parts, Suppliers and Projects Database

- S(sn ,sname,status,city)
- P(pn ,pname,color,weight,city)
- J(jn ,jname,budget,city)
- SPJ(sn , pn , jn ,qty)

جدول فروش یک فروشگاه

میزان فروش	شماره فاکتور فروش	موجودی	نام جنس	شماره جنس
12	1	1000	شکر	1
14	4	1000	شکر	1
5	5	1000	شکر	1
10	2	1500	نمک	2
20	4	1500	نمک	2
30	5	1500	نمک	2
40	3	1500	نمک	2

فروشگاه ۲

شماره جنس	نام جنس	موجودی
1	شکر	1000
2	نمک	1500

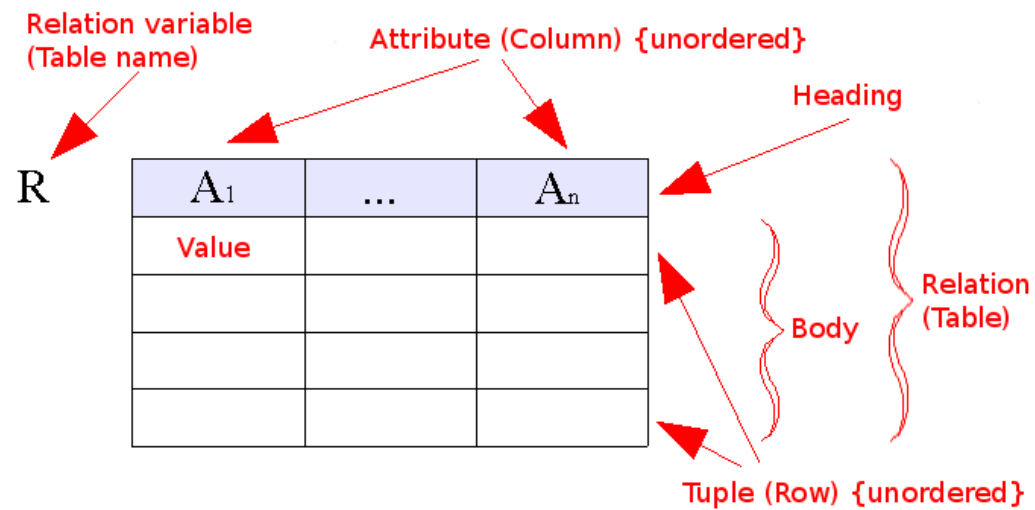
شماره جنس	شماره فاکتور فروش	میزان فروش
1	1	12
1	4	14
1	5	5
2	2	10
2	4	20
2	5	30
2	3	40

Types

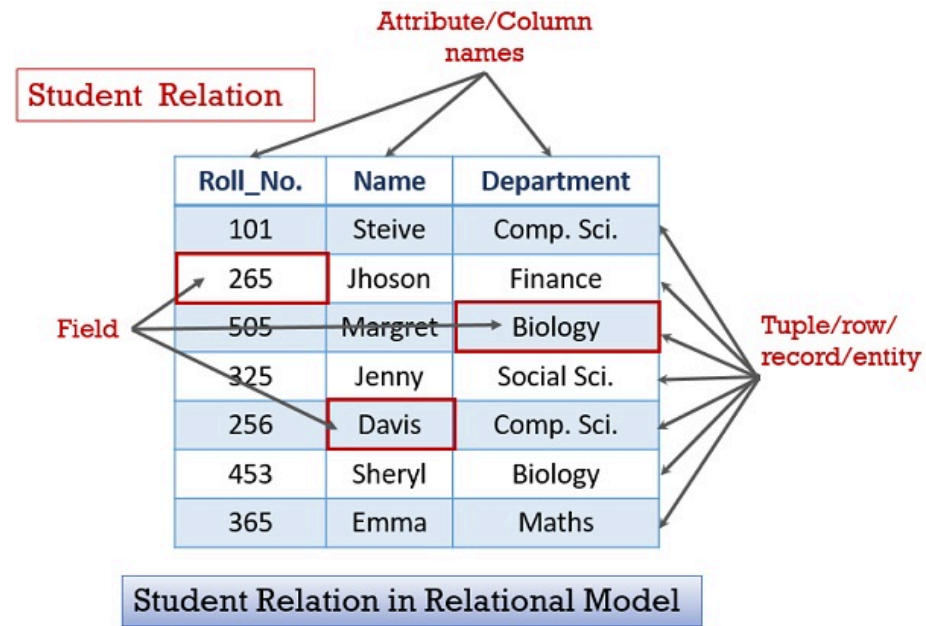
Not exact types

- product_id: positive number
- product_name: string
- stock: positive number
- quantity: positive number
- customer_id: positive number
- customer_name: string

Relational Model Like Table



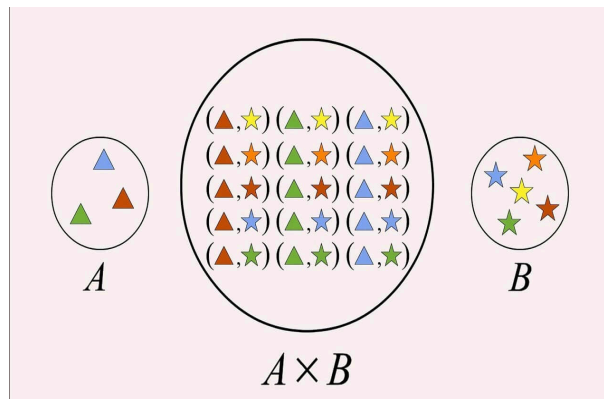
https://en.wikipedia.org/wiki/Relational_model



<https://binaryterms.com/relational-data-model.html>

Cartesian Product(I)

$$X \times Y = \{ (x, y) \mid x \in X \wedge y \in Y \}$$

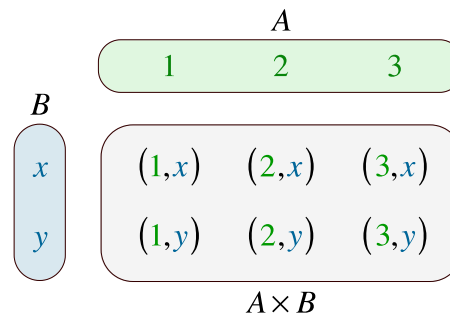


<https://smartinsight.ir/ضرب-دکارتی-دو-مجموعه/>

Cartesian Product(I)

$$A \times B = \{ (a, b) \mid a \in A \wedge b \in B \}$$

$$A = \{1, 2, 3\} \text{ and } B = \{x, y\}$$



$$A \times B = \{(1, x), (1, y), (2, x), (2, y), (3, x), (3, y)\}$$

$$B \times A = \{(x, 1), (x, 2), (x, 3), (y, 1), (y, 2), (y, 3)\}$$

$$A \times B \neq B \times A$$

<https://math24.net/cartesian-product-sets.html>

Set of Tuples (1)

```
{  
  ( a11, a12, a13, ... , a1n ) , # Tuple 1  
  ( a21, a22, a23, ... , a2n ) , # Tuple 2  
  .....  
  ( am1, am2, am3, ... , amn )   # Tuple m  
}
```

- Domain of $(\{a_{11}, a_{21}, \dots, a_{m1}\}) = S_1$
- Domain of $(\{a_{12}, a_{22}, \dots, a_{m2}\}) = S_2$
- ...
- Domain of $(\{a_{1n}, a_{2n}, \dots, a_{mn}\}) = S_n$

$$R_1 \subseteq S_1 * S_2 * \dots * S_n$$

- $\{a_{11}, a_{21}, \dots, a_{m1}\} \subseteq S_1$
- $\{a_{12}, a_{22}, \dots, a_{m2}\} \subseteq S_2$
- $\dots \subseteq S_i$
- $\{a_{1n}, a_{2n}, \dots, a_{mn}\} \subseteq S_n$

Set of Tuples (2)

```
{  
  { a11 : Label_1 , a12 : Label_2 , a13 : Label_3 , ... , a1n : Label_n } , # Tuple 1  
  { a21 : Label_1 , a22 : Label_2 , a23 : Label_3 , ... , a2n : Label_n } , # Tuple 2  
  .....  
  { am1 : Label_1 , am2 : Label_2 , am3 : Label_3 , ... , amn : Label_n } , # Tuple m  
}
```

Set of Tuples (3)

```
{  
  { a21 : Label_1 , a22 : Label_2 , a23 : Label_3 , ... , a2n : Label_n } , # Tuple 2  
  { a11 : Label_1 , a12 : Label_2 , a13 : Label_3 , ... , a1n : Label_n } , # Tuple 1  
  .....  
  { am1n : Label_1 , am2n : Label_2 , am3n : Label_3 , ... , amn : Label_n } , # Tuple m  
}
```


Set of Tuples (4)

```
{  
  { a21 : Label_1 , a22 : Label_2 , a23 : Label_3 , ... , a2n : Label_n } , # Tuple 2  
  { a12 : Label_2 , a13 : Label_3 , a11 : Label_1 , ... , a1n : Label_n } , # Tuple 1  
  .....  
  { am1 : Label_1 , am2 : Label_2 , am3 : Label_3 , ... , amn : Label_n } , # Tuple m  
}
```

Relational Algebra

<http://yoosofan.github.io/webrel/index.html>

<http://yoosofan.github.io/webrel/help/help.html>

- Schema
- Table Schema
- Database Schema

```

p;
-----
s minus s ;
-----
RELATION {
  TUPLE{
    PN PN("s4"), SNAME SNAME("Clark"),
    STATUS STATUS(20), CITY CITY("London")
  }
};
-----

p
minus
RELATION {
  TUPLE{
    PN("p1"), PNAME("Nut"),
    COLOR("Red"), WEIGHT(12), CITY("London")
  }
};
-----

p
union
RELATION {
  TUPLE{
    PN("p7"), PNAME("Bult2"),
    COLOR("White"), WEIGHT(15), CITY("Kashan")
  },
  TUPLE{
    PN("p8"), PNAME("Clark"),
    COLOR("Red"), WEIGHT(20), CITY("London")
  }
};

```

```

RELATION {
  TUPLE{
    SN SN("s7"), SNAME SNAME("Clark"),
    STATUS STATUS(20), CITY CITY("London")
  },
  TUPLE{
    SN SN ("s8"), SNAME SNAME("John"),
    STATUS STATUS(25), CITY CITY("Shiraz")
  }
};

```

```

-----
RELATION {
  TUPLE{
    SN("s4"), SNAME("Clark"),
    STATUS(20), CITY("London")
  },
  TUPLE{
    SN("s4"), SNAME("Clark"),
    STATUS(20), CITY("London")
  }
};

```

```

-----
RELATION {
  TUPLE{SN SN("s4"), SNAME SNAME("Clark"), STATUS STATUS(20), CITY CITY("London")},
  TUPLE{SN SN ("s5"), SNAME SNAME("Adam"), STATUS STATUS(40), CITY CITY("London")}
}

```

union

```

RELATION {
  TUPLE{SN SN("s6"), SNAME SNAME("Clark"), STATUS STATUS(20), CITY CITY("London")},
  TUPLE{SN SN ("s7"), SNAME SNAME("Adam"), STATUS STATUS(40), CITY CITY("London")}
}
};

```

Assignment

```
a := s
  union
    RELATION{
      TUPLE{SN("s16"), SNAME("Clark"), STATUS(20), CITY("London")},
      TUPLE{SN("s17"), SNAME("Adam"), STATUS(40), CITY("London")}
    };
```

SN	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens
s16	Clark	20	London
s17	Adam	40	London

Projection

```
s{city};
```

CITY
London
Paris
Athens

```
p{city} minus s{city};
```

CITY
Oslo

```
a minus s ;
```

SN	SNAME	STATUS	CITY
s16	Clark	20	London
s17	Adam	40	London

```
p{city, pname};
```

CITY	PNAME
London	Nut
Paris	Bolt
Oslo	Screw
london	Screw
Paris	Cam
London	Cog

شهرهایی را بیابید که هم عرضه کننده و هم قطعه در آنها هست.

```
s{city} minus (s{city} minus p{city});
```

```
p{city} minus (p{city} minus s{city});
```

```
s{city} intersect p{city};
```

CITY
London
Paris

شهرهای عرضه کنندگانی را بیابید که در آن شهرها قطعه‌ای نیست.

```
s{city} minus p{city};
```

CITY
Athens

شهرهایی را بیابید که یا عرضه کننده یا قطعه یا هر دو در آنهاست.

```
s{city} union p{city};
```

شهرهایی را بیابید که یا عرضه کننده یا قطعه در آنها باشد ولی هر قطعه یا عرضه کننده دو در آن شهرها نباشد

```
(s{city} union p{city}) minus (s{city} intersect p{city});
```

```
(s{city} minus p{city}) union (p{city} minus s{city});
```

CITY
Athens
Oslo

Condition

```
p where city = "Paris";
```

PN	PNAME	COLOR	WEIGHT	CITY
P2	Bolt	Green	17	Paris
P5	Cam	Blue	12	Paris

نام عرضه کنندگان با وضعیت بیشتر از ۴۰ را بیابید.

```
s where status > 40;
```

```
(s where status > 40){sname};
```

نام عرضه کنندگانی را بیابید که وضعیت آنها (status) بیشتر از ۲۵ است.

```
(s where status > 25){sname};
```

Where

SN	SNAME	STATUS	CITY
S3	Blake	30	Paris
S5	Adams	30	Athens

Project

SNAME
Blake
Adams

Get part names of P2

نام قطعه P2 را بیابید.

```
(P where pn = "P2") {pname};
```

نام شهرهایی را بیابید که یا قطعه‌ای با وزن بیشتر از 13 در آنها هست یا
عرضه‌کننده‌ای با وضعیت بیشتر از ۳۴ در آنها هست.

```
(p where weight > 13){city} union (s where status > 34){city};
```

CITY
Paris
Oslo
london

نام شهرهایی را بیابید که یا قطعه‌ای با وزن بیشتر از ۲۴ در آنها هست یا
عرضه‌کننده‌ای با وضعیت بیشتر از ۲۱ در آنها هست.

```
(p where weight > 24){city} union (s where status > 21){city};
```

CITY
Paris
Athens

Times

```
p{city} times s{status};  
p{pname, weight} times s{sname} ;  
p{pn} times s{sn} ;  
p{city} times s{city} -- wrong  
  
p{pn, pname, color}  
times  
s{sn, sname, status};
```

CITY	STATUS
London	20
London	10
London	30
Paris	20
Paris	10
Paris	30
Oslo	20
Oslo	10
Oslo	30

چه عرضه‌کنندگانی چه قطعاتی را عرضه نکرده‌اند.

زوج شمارهٔ عرضه‌کنندگان و قطعاتی را بیابید که آن عرضه‌کنندگان آن قطعات را عرضه نکرده باشند.

PN	SN	PN	SN	PN	SN
P1	S3	P3	S4	P5	S3
P1	S4	P3	S5	P5	S5
P1	S5	P4	S2	P6	S2
P2	S5	P4	S3	P6	S3
P3	S2	P4	S5	P6	S4
P3	S3	P5	S2	P6	S5

```
(p{pn} times s{sn})  
minus  
sp{sn, pn};
```

Rename

تغییر نام

```
S rename SN as SN1;
```

SN1	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

قطعه‌های عرضه شده را بیابید

```
p times sp;  
-- wrong
```

```
p  
times  
(sp rename pn as sppn)  
-- need another step
```

```
(  
  p  
  times  
  (sp rename pn as sppn)  
) where pn = sppn;
```

```
(  
  (  
    p  
    times  
    (sp rename pn as sppn)  
  ) where pn = sppn  
) {pn, pname, weight, color};
```

WEIGHT	PN	PNAME	COLOR
12	P1	Nut	Red
17	P2	Bolt	Green
17	P3	Screw	Blue
14	P4	Screw	Red
12	P5	Cam	Blue
19	P6	Cog	Red

نام قطعه‌های عرضه شده را بیابید

```
p
times
(sp rename pn as sppn)
```

```
(
  p
  times
  (sp rename pn as sppn)
) where pn = sppn
```

```
(
  (
    p
    times
    (sp rename pn as sppn)
  ) where pn = sppn
) {pname}
```

PNAME
Nut
Bolt
Screw
Cam
Cog

زوج نام عرضه‌کنندگان و نام قطعاتی را بیابید که آن عرضه‌کننده آن قطعه را عرضه نکرده

باشد (I)

```
(p{pn} times s{sn})  
minus  
sp{sn,pn};
```

```
( (p{pn} times s{sn}) minus sp{sn,pn} )  
times  
(s{sn, sname} rename sn as ssn)  
;
```

```
( (p{pn} times s{sn}) minus sp{sn,pn} )  
times  
(s{sn, sname} rename sn as ssn)  
times  
(p{pn, pname} rename pn as ppn)  
;
```

```
(  
  (  
    (p{pn} times s{sn}) minus sp{sn,pn} )  
    times  
    (s{sn, sname} rename sn as ssn)  
  ) where sn == ssn  
)  
times  
(p{pn, pname} rename pn as ppn)  
;
```

زوج نام عرضه‌کنندگان و نام قطعاتی را بیابید که آن عرضه‌کننده آن قطعه را عرضه نکرده باشد

(II)

```
(
  (
    ( (p{pn} times s{sn}) minus sp{sn,pn} )
    times
    (s{sn, sname} rename sn as ssn)
  ) where sn == ssn
)
times
(p{pn, pname} rename pn as ppn)
) where pn == ppn
;
```

```
(
  (
    ( (p{pn} times s{sn}) minus sp{sn,pn} )
    times
    (s{sn, sname} rename sn as ssn)
    times
    (p{pn, pname} rename pn as ppn)
  ) where pn == ppn and sn == ssn
)
{sname, pname}
;
```

زوج نام عرضه‌کنندگان و نام قطعاتی را بیابید که آن عرضه‌کننده آن قطعه را عرضه

نکرده باشد (III)

```
1 (
2   (
3     (p{pn} times s{sn}) minus sp{sn,pn} )
4     times
5     (s{sn, sname} rename sn as ssn)
6     times
7     (p{pn, pname} rename pn as ppn)
8   ) where pn == ppn and sn == ssn
9 )
10 {sname, pname}
11 ;
```

```
1 A := p{pn, pname} times s{sn, sname};
2 B := (sp{sn, pn} times (s{sn, sname} rename sn as ssn1) where sn = ssn1;
3 C := (B times (p{pn, pname} rename pn as pn1) where pn = pn1;
4 A minus (C{sn, sname, pn, pname}))
```

زوج نام عرضه‌کنندگان و نام قطعاتی را بیابید که آن عرضه‌کننده آن قطعه را عرضه
نکرده باشد (IV)

```
1 A := p{pn, pname} times s{sn, sname};
2 B := (
3     sp{sn, pn}
4     times
5     (
6         s{sn, sname} rename sn as ssn1
7     ) where sn = ssn1
8 )
9 ;
10 C := (
11     B
12     times
13     (
14         p{pn, pname} rename pn as pn1
15     ) where pn = pn1
16 );
17 A minus (C{sn, sname, pn, pname})
```


نام عرضه‌کنندگانی را بیابید که هیچ قطعه‌ای عرضه نکرده‌اند.

```
s{sn} minus sp{sn} ;
```

```
( s{sn} minus sp{sn} ) times s ; -- wrong
```

```
( s{sn} minus sp{sn} ) times (s rename sn as ssn) ;
```

```
( ( s{sn} minus sp{sn} ) times (s rename sn as ssn) ) where sn = ssn;
```

```
( ( ( s{sn} minus sp{sn} ) times (s rename sn as ssn) ) where sn = ssn) {sname};
```

```
(  
  ( -- better format  
    (s{sn} minus sp{sn})  
    times  
    (s rename sn as ssn)  
  ) where sn = ssn  
) {sname}  
;
```

زوج شهرهای عرضه‌کنندگان و شهرهای قطعات را بیابید.

```
s times p;
```

```
s{city} times p{city};
```

```
( s rename city as scity ){scity}  
times  
p{city};
```

scity	CITY
London	London
London	Paris
London	Oslo
Paris	London
Paris	Paris
Paris	Oslo
Athens	London
Athens	Paris
Athens	Oslo

زوج شهرهای عرضه‌کنندگان و شهرهای قطعات غیرهمشهری را بیابید.

```
( s rename city as scity ){scity}  
times  
p{city};
```

```
(  
  ( s rename city as scity ){scity}  
  times  
  p{city}  
) where scity <> city;
```

scity	city
London	Paris
London	Oslo
Paris	London
Paris	Oslo
Athens	London
Athens	Paris
Athens	Oslo

نام عرضه کنندگانی را بیابید که قطعه‌ای عرضه کرده باشند.

```
s times sp
```

```
(s rename sn as sn1) times sp
```

```
((s rename sn as sn1) times sp) where sn1=sn;
```

```
((s rename sn as sn1) times sp) where sn1=sn){sname};
```

sname
Smith
Jones
Blake
Clark

نام عرضه کنندگانی را بیابید که عرضه‌ای (qty) بیشتر از ۳۰۰ داشته باشند.

```
s  
times  
sp
```

```
(s rename sn as sn1)  
times  
sp
```

```
(s rename sn as sn1)  
times  
(sp where qty > 300)
```

```
(  
  (s rename sn as sn1)  
  times  
  (sp where qty > 300)  
) where sn1 = sn
```

```
(  
  (  
    (s rename sn as sn1)  
    times  
    (sp where qty > 300)  
  ) where sn1 = sn  
) {sname};
```

```
(  
  (  
    (s rename sn as sn1)  
    times  
    sp  
  ) where sn1 = sn and qty > 300  
) {sname};
```

نام عرضه کنندگانی را بیابید که قطعه P4 را عرضه کرده باشد

```
(
  (
    (S rename sn as sn1)
    times
    (sp where pn = "P4")
  ) where sn1=sn
){sname};
```

sname
Smith
Clark

شهر عرضه کنندگانی را بیابید که قطعه قرمزی را عرضه کرده باشند (I).

شهر عرضه کنندگانی را بیابید که دست کم یک عرضه کننده در آن شهر دست کم یک قطعه قرمز را عرضه کرده باشد

```
(s rename sn as sn1)
times
sp
```

```
(
  (s rename sn as sn1)
  times
  sp
) where sn1 = sn
```

```
(
  (
    (s rename sn as sn1)
    times
    sp
  ) where sn1 = sn
)
times
(p rename pn as pn1)
```

```
(
  (
    (s rename sn as sn1)
    times
    sp
  ) where sn1 = sn
)
times
( (p rename pn as pn1) where color = "Red")
```

شهر عرضه کنندگانی را بیابید که قطعه قرمزی را عرضه کرده باشند (II).

```
1 (
2   (
3     (s rename sn as sn1)
4     times
5     sp
6   ) where sn1 = sn
7 )
8 times
9 (
10  (
11    (p rename pn as pn1)
12    where color = "Red"
13  ){pn1}
14 )
```

```
1 (
2   (
3     (
4       (
5         (s rename sn as sn1)
6         times
7         sp
8       ) where sn1 = sn
9     )
10    times
11    (
12      (
13        (p rename pn as pn1)
14        where color = "Red"
15      ){pn1}
16    ) where pn1 = pn
17  )
18 ){city}
19 ;
```


شهر عرضه کنندگانی را بیابید که قطعه قرمزی را عرضه کرده باشند (III).

```
(
  (
    (
      (
        (s rename sn as sn1)
        times
        sp
      ) where sn1=sn
    ){pn, city}
  )
  times
  (
    (
      (
        (p where color = "Red")
      ){pn}
    )
    ) rename pn as pn1
  ) where pn1=pn
){city};
```

CITY
London
Paris

شهر عرضه کنندگانی را بیابید که قطعه قرمزی با وزن بیشتر از ۱۳ را عرضه کرده باشند.

```
1 (
2   (
3     (
4       (
5         (s rename sn as sn1)
6         times
7         sp
8       ) where sn=sn1
9     ){pn, city}
10   )
11   times
12   (
13     (
14       (p where color = "Red")
15       where weight > 13
16     ){pn}
17   ) rename pn as pn1
18   )
19   ) where pn1=pn
20   )
21   ) where pn1=pn
22 ){city};
```

```
1 (
2   (
3     (
4       (s rename sn as sn1)
5       times
6       (sp rename pn as pn1)
7     ) where sn1=sn
8   ){pn1, city}
9   times
10  (
11    (
12      (p where color = "Red")
13      where weight > 13
14    ){pn}
15  )
16  ) where pn1=pn
17  ){city};
```

شهر عرضه کنندگانی را بیابید که در شهر آنها قطعه‌ای نیز وجود داشته باشد.

```
s{city} intersect p{city};
```

نام عرضه کنندگانی را بیابید که در شهر آنها قطعه‌ای وجود ندارد.

```
s{city} minus p{city};
```

```
(s{city} minus p{city})  
times  
(s rename city as scity)
```

```
(  
  (s{city} minus p{city})  
  times  
  (s rename city as scity)  
) where city = scity
```

```
1 (  
2 (  
3   (s{city} minus p{city})  
4   times  
5   (s rename city as scity)  
6   ) where city = scity  
7 ){sname};
```

SNAME
Adams

نام قطعه‌های عرضه شده را بیابید.

```
1 (
2   (
3     p{{pn, pname}}
4     times
5     (sp rename pn as pn1)
6   ) where pn1 = pn
7 ) {{pname}};
```

PNAME
Nut
Bolt
Screw
Cam
Cog

شهر قطعه‌های عرضه شده با وزن بیشتر از ۱۵ را بیابید.

```
1 (
2   (
3     (p where weight > 15)
4     times
5     (sp rename pn as pn1)
6   ) where pn1 = pn
7 ){city};
```

CITY
Paris
Oslo
London

نام عرضه کنندگانی را بیابید که قطعه‌ای در شهر پاریس عرضه کرده باشند.

```
(
  (
    (
      (
        s{sn, sname}
        times
        (sp rename sn as sn1)
      ) where sn1 = sn
    )
    times
    (p rename pn as pn1)
  ) where pn1 = pn
) where city = "Paris"
){sname} ;
```

SNAME
Smith
Jones
Blake
Clark

پیوند Join

```
( p times (sp rename pn as sppn) ) where sppn = pn
```

```
(  
  p  
  times  
  (sp rename pn as sppn)  
) where sppn = pn
```

```
p join sp
```

- Join does not have order
- It will apply equals for each shared attributes
- It will apply times for two relations without shared attribute

نام قطعه‌های عرضه شده را بیابید.

```
(  
  (  
    p  
    times  
    (sp rename pn as sppn)  
  ) where sppn = pn  
) {pname};
```

```
(p join sp){pname}
```

نام قطعه‌های قرمز عرضه شده را بیابید.

```
(
  (
    (p where color="Red")
    times
    (sp rename pn as sppn)
  ) where sppn = pn
){pname};
```

```
(
  (p where color = "Red")
  join
  sp
){pname}
```

شهر قطعه‌های عرضه شده با وزن بیشتر از ۱۵ را بیابید.

```
(  
  (p where weight > 15)  
  join  
  sp  
) {city};
```

CITY
Paris
Oslo
London

نام عرضه کنندگانی را بیابید که در شهر آنها قطعه‌ای وجود ندارد.

```
s{city} minus p{city};
```

```
(s{city} minus p{city})  
join  
s
```

```
(  
  (s{city} minus p{city})  
  join  
  s  
) {sname};
```

```
s{sname}  
minus  
((s join p){sname});
```

SNAME
Adams

نام عرضه کنندگانی را بیابید که قطعه‌ای در شهر پاریس عرضه کرده باشند.

```
( (s{sn, sname} join sp) join p) where city = "Paris"){sname} ;
```

```
(
  (
    (s{sn, sname} join sp)
    join
    p
  ) where city = "Paris"
){sname} ;
```

SNAME
Smith
Jones
Blake
Clark

```
(
  --- راه حل نادرست
  (
    (s join sp)
    join p
  ) where city = "Paris"
){sname} ;
```

```
(
  -- راه حل نادرست
  (s join sp){pn, sname}
  join p
) where city = "Paris"
){sname} ;
```

```
(
  -- راه حل نادرست
  (p{city} where city='Paris') join s
){sname} ;
```

نام عرضه کنندگانی را بیابید که قطعه‌ای در شهر خودشان عرضه کرده باشند.

```
(
  (s join sp) join p
){sname} ;
```

SNAME
Smith
Jones
Blake
Clark

```
-- نادرست ۱
(
  (s join sp)
  join
  (p{city})
){sname} ;
```

```
-- نادرست ۲
(
  (s join sp)
  join
  p{city}
){sname} ;
```

```
-- نادرست ۳
(s join sp join p{city}){sname} ;
```

```
-- نادرست ۴
((s join sp{sn})join p){sname};
```

```
-- نادرست ۵
((sp join s){sn, sname} join p){sname};
```

```
-- نادرست ۶
((sp join s){sn, sname, city} join p){sname};
```

```
-- نادرست ۷
(((s join sp ){city}) join p){sname};
```

نام قطعاتی را بیابید که وزنشان بیشتر از ۱۰ است و عرضه‌کننده‌ای در شهر پاریس
با وضعیت بیشتر از ۲۰ آنها را عرضه کرده باشد.

```
( p where weight > 10 ){pn, pname}
```

```
(s where status > 20) where city = "Paris"
```

```
( p where weight > 10 ){pn, pname}  
join  
sp
```

```
(  
  (  
    ( p where weight > 10 ){pn, pname}  
    join  
    sp  
  )  
  join  
  (  
    (s where status > 20) where city = "Paris"  
  )  
) {pname} ;
```

PNAME
Bolt

Keys

ابر کلیدها Super Keys

- P(pn, pname, color, weight, city)
 - {pn}, {pn, pname}, {pn, color}, {pn, pname, weight}
 - {pn, city, color} {pn, pname, color, weight, city}
 - نادرست‌ها
 - {city}
 - {city, weight}
 - {pname, city, weight}
 - {pname, color}
 - هیچ دو قطعه‌ای در یک شهر نام یکسانی ندارند.
 - {pn}, {pname, city}, {pn, city}
 - {pname, city, weight}
 - اگر در هر شهر فقط یک قطعه بتواند باشد.
 - {pn}, {city}, {pn, city} , {city, pname}
 - {city, pname, color}

کلیدهای نامزد Candidate Keys

- S(sn, sname, status, city)
 - {sn}
 - هیچ دو عرضه‌کننده‌ای در یک شهر نام یکسانی ندارند.
 - {sn}
 - {sname, city}
 - اگر در هر شهر فقط یک عرضه‌کننده بتواند باشد.
 - {sn}
 - {city}
- SP(sn, pn, qty)
 - {sn, pn}

کلید اصلی Primary key

- SP(sn, pn, qty)
- S(sn, sname, status, city)
 - {sn}
 - هیچ دو عرضه‌کننده‌ای در یک شهر نام یکسانی ندارند.
 - {sn}
 - {sname, city}
 - فقط یکی از این دو بالایی کلید اصلی گذاشته شود.
 - اگر در هر شهر فقط یک عرضه‌کننده بتواند باشد.
 - {sn}
 - {city}
 - فقط یکی از این دو بالایی کلید اصلی گذاشته شود.

Foreign key

- SP([sn](#), [pn](#), qty)
- sn from s
- pn from p

Project/Department/Employee

Design 1

- Employee(SSN, name, salary, Dn)
- Department(DN, DeptName, MgrSSN)
- Project(PN, location, ProjName)
- HourLog(SSN, PN, hours)

Design 2

- Employee(SSN, name, salary, DeptName)
- Department(DeptName, MgrSSN)
- Project(PN, location, ProjName)
- HourLog(SSN, PN, hours)

Design 3

- Employee(SSN, name, salary, DeptName)
- Department(DeptName, MgrSSN)
- Project(ProjName, location)
- HourLog(SSN, ProjName, hours)

SPJ

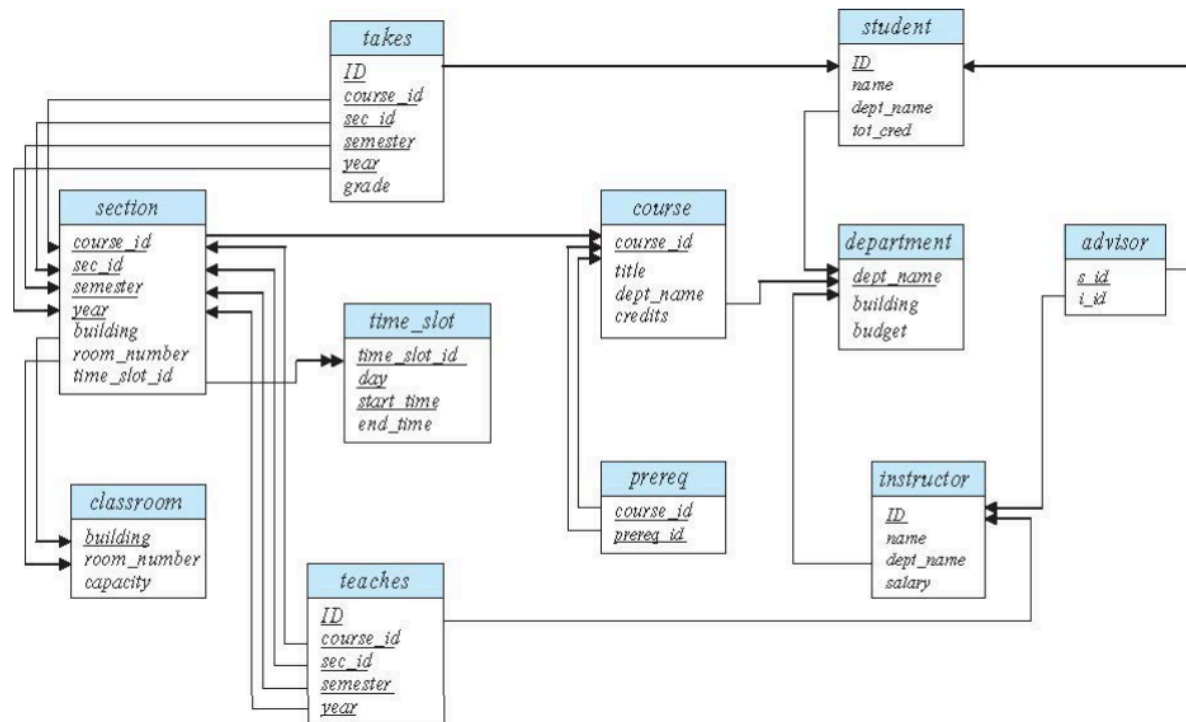
- S(sn,sname,status,city) ,
- P(pn,pname,color,weight,city) ,
- J(jn,jname,budget,city)
- SPJ(sn, pn, jn, qty)

Library

- book(bn , title, category, fpd, author)
- member(mn , name , category, bn)
- borrow(bn , mn , nd , rdt, ret)

University Database

- student(ID, name, dept_name, tot_cred)
- instructor(ID, name, dept_name, salary)
- department (dept_name, building, budget)
- course(course_id, title, dept_name, credits)
- classroom (building, room_number, capacity)
- section(course_id, sec_id, semester, year, building, room_number, time_slot_id)
- teaches(ID, course_id, sec_id, semester, year)
- takes(ID, course_id, sec_id, semester, year, grade)
- advisor(s_ID, i_ID)
- time_slot(time_slot_id, day, start_time, end_time)
- prereq(course_id, prereq_id)



Relational Calculus

Calculation on tuples instead of relations

- $\{ px \in P \mid px.city = 'kashan' \}$
- \forall forall
- \exists exists
- $\forall a, b \in R \mid a+b = b + a$
- $\exists a \in R, \forall b \in R \mid a * b = b$
- $a = 1$
- $\forall x p(x) \equiv \sim \exists x \sim p(x)$

- RANGEOVER sx RANGES over S;
- RANGEOVER sy RANGES over S;
- RANGEOVER px RANGES over P;
- RANGEOVER py RANGES over P;
- RANGEOVER spx RANGES over SP;
- RANGEOVER spy RANGES over SP;

$\{ \exists px \in P \mid px.city = 'kashan' \}$

$\{ \exists sx \in S \mid sx.city = 'kashan' \}$

$\{ \forall sx \in S \mid sx.status > 1 \}$

- px where $px.city = 'kashan'$
- sx where $sx.city = 'kashan'$
- sx where $sx.status > 1$

1. $sx.sname$ where $sx.sn = 's2'$
 - $\{sx.sname \mid sx \in S \mid sx.sn = 's2'\}$
2. $sx.sname, sx.staus$ where $sx.status > 100$
 - $\{(sx.sname, sx.status) \mid sx \in S \mid sx.status > 100\}$
3. $sx.sn, sx.sname$
4. $spx.sn, spx.pn$
5. $spx.sn, spy.pn$

نام قطعات را بیابید.

1. $px.pname$

نام عرضه‌کنندگان را بیابید

1. $sx.sname$

نام قطعاتی را بیابید که وزنشان بیشتر از ۲۰ باشد

- px
- px.pname
- px.pname where
- px.pname where px.weight > 20

نام قطعاتی را بیابید که وزنشان بیشتر از ۲۰ باشد و در شهر کاشان باشند.

- px
- px.pname
- px.pname where
- px.pname where px.weight > 20
- px.pname where px.weight > 20 and
- px.pname where px.weight > 20 and px.city = 'kashan'

نام قطعات قرمز رنگ با وزن بیشتر از ۱۲ را بیابید

1. px.pname
2. px.pname where
3. px.pname where px.weight > 12
4. px.pname where px.weight > 12 and
5. px.pname where px.weight > 12 and px.color = 'red'

نام و شماره قطعاتی را بیابید که وزن بیشتر از ۲۰ داشته باشند.

1. px.pname
2. px.pname, px.pn
3. px.pname, px.pn where px.weight > 20
4. {px.pname, px.pn} where px.weight > 20

نام عرضه‌کنندگانی را بیابید که وضعیت بیشتر از ۱۰ داشته باشند.

1. `sx.sname`
2. `sx.sname where`
3. `sx.sname where sx.status > 10`

زوج شمارهٔ عرضه‌کنندگانی را بیابید که در یک شهر باشند.

1. $sx.sn$
2. $sx.sn, sy.sn$
3. $\{sx.sn, sy.sn\}$
4. $\{sx.sn, sy.sn \text{ as } sn1\}$
5. $\{sx.sn, sy.sn \text{ as } sn1\}$ where
6. $\{sx.sn, sy.sn \text{ as } sn1\}$ where $sx.city = sy.city$
7. $\{sx.sn, sy.sn \text{ as } sn1\}$ where $sx.city = sy.city$ and
8. $\{sx.sn, sy.sn \text{ as } sn1\}$ where $sx.city = sy.city$ and $sx.sn < sy.sn$

نام شهر قطعاتی را بیابید که عرضه شده باشند.

- px
- px.city
- px.city where
- px.city where exists spx
- px.city where exists spx(px.pn = spx.pn)

نام قطعات عرضه شده را بیابید

1. px.pname
2. px.pname where
3. px.pname where exists spx
4. px.pname where exists spx (spx.pn = px.pn)

1. px.pname where spx.pn = px.pn
2. Error
3. px.pname where forall spx (spx.pn = px.pn)
4. نام همه قطعاتی را بیابید که همه عرضه کنندگان آنها را عرضه کرده باشند
5. نام همه قطعاتی که عرضه شده‌اند

نام شهرهای عرضه‌کنندگانی را بیابید که قطعه با شماره p2 را عرضه کرده باشند.

- sx.city
- sx.city where
- sx.city where exists spx (
 - spx.sn = sx.sn
 - and
 - spx.pn = 'p2'
 -)
- sx.city where exists spx(spx.sn = sx.sn and spx.pn = 'p2')
- اشکال
- ```
sx.city where exists spx (
 spx.sn = sx.sn
) and exists px (px.pn = 'P2')
```
- ```
sx.city where exists spx (  
    spx.sn = sx.sn  
) and exists spy (spy.pn = 'P2')
```

زوج شمارهٔ عرضه‌کنندگانی را بیابید که در یک شهر باشند و قطعه یا قطعه‌هایی عرضه کرده باشند.

1. {sx.sn, sy.sn as sn2}
2. {sx.sn, sy.sn as sn2} where sx.city = sy.city
3. {sx.sn, sy.sn as sn2} where sx.city = sy.city and sx.sn < sy.sn
4. {sx.sn, sy.sn as sn2} where sx.city = sy.city and sx.sn < sy.sn and
5. {sx.sn, sy.sn as sn2} where sx.city = sy.city and sx.sn < sy.sn and exists spx(sx.sn = spx.sn)
6. {sx.sn, sy.sn as sn2} where sx.city = sy.city and sx.sn < sy.sn and exists spx(sx.sn = spx.sn) and
7. {sx.sn, sy.sn as sn2} where sx.city = sy.city and sx.sn < sy.sn and exists spx(sx.sn = spx.sn) and exists spy
8. {sx.sn, sy.sn as sn2} where sx.city = sy.city and sx.sn < sy.sn and exists spx(sx.sn = spx.sn) and exists spy(sy.sn = spy.sn)

پاسخ نادرست

1. {sx.sn, sy.sn as sn2} where sx.city = sy.city and sx.sn < sy.sn and exists spx(sx.sn = spx.sn and sy.sn = spx.sn)

نام عرضه‌کنندگانی را بیابید که دست کم یک قطعه قرمز عرضه کرده باشند.

1. `sx.sname`
2. `sx.sname where`
3. `sx.sname where exists spx(sx.sn=spx.sn and`
4. `sx.sname where exists spx(sx.sn=spx.sn and exists px`
5. `sx.sname where exists spx(sx.sn=spx.sn and exists px(px.pn=spx.pn`
6. `sx.sname where exists spx(sx.sn=spx.sn and exists px(px.pn=spx.pn and`
7. `sx.sname where exists spx(sx.sn=spx.sn and exists px(px.pn=spx.pn and px.color = 'red'))`

1. `sx.sname where exists px(px.color = 'red' and exists spx(px.pn=spx.pn and sx.sn=spx.sn))`

پاسخ نادرست

1. `sx.sname where exists spx(sx.sn=spx.sn) and exists px(px.pn=spx.pn and px.color = 'red')`
2. `sx.sname where exists spx(sx.sn=spx.sn) and exists px(exists spx(px.pn=spx.pn and px.color = 'red'))`
3. `sx.sname where exists spx(sx.sn=spx.sn and px.pn=spx.pn and px.color = 'red')`

زوج شماره عرضه‌کنندگانی را بیابید که در یک شهر باشند و دست کم یکی از آن دو عرضه‌کننده، قطعه یا قطعه‌هایی عرضه کرده باشند.

SN	SNAME	STATUS	CITY
S1	Smith	20	Paris
S2	Jones	10	Paris

SN	PN	qty
S2	P1	20

```
{sx.sn, sy.sn as sn2} where sx.city = sy.city and
sx.sn <> sy.sn and (exists spx(sx.sn = spx.sn))
```

sn	sn2
s2	s1

```
{sx.sn, sy.sn as sn2} where sx.city = sy.city and
sx.sn < sy.sn and exists spx(sx.sn = spx.sn)
```

sn	sn2
----	-----

```
{sx.sn, sy.sn as sn2} where sx.city = sy.city and
sx.sn < sy.sn and (exists spx(sx.sn = spx.sn) or
exists spy(sy.sn = spy.sn))
```

sn	sn2
s1	s2

زوج شمارهٔ عرضه‌کنندگانی را بیابید که در یک شهر باشند و دست کم یکی از آن دو عرضه‌کننده، قطعه یا قطعه‌هایی عرضه کرده باشند.

```
{sx.sn, sy.sn as sn2} where sx.city = sy.city and  
sx.sn < sy.sn and exists spx(sx.sn = spx.sn or sy.sn = spx.sn)
```


زوج شمارهٔ عرضه‌کنندگانی را بیابید که در یک شهر باشند و هر دو عرضه‌کننده،
قطعه یا قطعه‌هایی را عرضه کرده باشند.

هر دو عرضه کرده باشند (راه حل اشتباه)

-- راه حل نادرست

```
{sx.sn, sy.sn as sn2} where sx.city = sy.city and  
sx.sn < sy.sn and exists spx(sx.sn = spx.sn and sy.sn = spx.sn)
```

-- راه حل درست

```
{sx.sn, sy.sn as sn2} where sx.city = sy.city and  
sx.sn < sy.sn and (exists spx(sx.sn = spx.sn) and  
exists spy(sy.sn = spy.sn))
```

-- راه حل درست

```
{sx.sn, sy.sn as sn2} where sx.city = sy.city and  
sx.sn < sy.sn and (exists spx(sx.sn = spx.sn) and  
exists spx(sy.sn = spx.sn))
```

نام عرضه‌کنندگانی را بیابید که همهٔ قطعه‌ها را عرضه کرده باشند (I).

-- نادرست

```
sx.sname where forall spx(sx.sn=spx.sn)
```

-- نادرست

```
sx.sname where forall spx(sx.sn != spx.sn)
```

```
sx.sname where forall px(
```

-- نادرست

```
sx.sname where forall px(  
  px.pn=spx.pn  
)  sx.sn=spx.sn
```

نام عرضه‌کنندگانی را بیابید که همهٔ قطعه‌ها را عرضه کرده باشند (II).

```
sx.sname where forall px(  
  exists spx(  
    spx.pn=px.pn and sx.sn=spx.sn  
  )  
)
```

```
sx.sname where forall px(  
  exists spx(  
    spx.pn=px.pn and sx.sn=spx.sn  
  )  
)
```

```
sx.sname where forall px(  
  exists spx(  
    spx.pn=px.pn and sx.sn=spx.sn  
  )  
)
```

```
sx.sname where not exists px(  
  not exists spx(  
    spx.pn=px.pn and sx.sn=spx.sn  
  )  
)
```

```
sx.sname where forall px( -- ??  
  not exists spx(  
    spx.pn <> px.pn or sx.sn <> spx.sn  
  )  
)
```

```
-- نادرست  
sx.sname where forall px(  
  sx.sn=spx.sn and  
  exists spx( spx.pn=px.pn )  
)
```

شماره قطعاتی را بیابید که هیچ عرضه کننده‌ای آنها را عرضه نکرده باشد.

```
px.pn where not exists spx (spx.pn = px.pn)
```

```
px.pn where forall spx(spx.pn<>px.pn)
```

دستورهای اضافی

```
px.pn where not exists sx(  
    exists spx(spx.sn=sx.sn and spx.pn=px.pn  
)
```

```
-- نادرست  
px.pn where exists spx(spx.pn<>px.pn)  
spx.pn where not exists spx(spx.pn<>spx.pn)  
spx.pn where not exists spy(spx.pn<>spy.pn)  
px.pn where not exists sx(exists spx(sx.sn=spx.sn))
```

نام قطعاتی را بیابید که اگر عرضه‌کننده‌ای قطعهٔ p3 را عرضه کرده‌اند آن قطعه را نیز عرضه کرده باشند.

```
px.pname where not exists spx(  
  spx.pn = 'p3' and not exists spy(  
    spy.sn = spx.sn and spy.pn = px.pn  
  )  
)
```

```
px.pname where forall spx(  
  spx.pn <> 'p3' or exists spy(  
    spy.sn = spx.sn and spy.pn = px.pn  
  )  
)
```

نام قطعاتی که برای‌شان وجود نداشته باشد عرضه‌ای که آن
عرضه برای قطعهٔ p3 باشد و وجود نداشته باشد عرضهٔ
دیگری از همان عرضه‌کننده که قطعهٔ آن همین قطعهٔ مورد
نظر ما نباشد.

```
-- نادرست  
px.pname where forall spx(  
  spx.pn = 'p3' or not exists spy(  
    spy.sn = spx.sn and spy.pn = px.pn  
  )  
)
```

نام قطعاتی را بیابید که اگر عرضه‌کننده‌ای قطعهٔ p3 را عرضه کرده باشد آن قطعه را

نیز عرضه کرده باشند. پاسخ‌های نادرست (I)

```
px.pname where exists spx(  
  spx.pn = 'p3' and exists spy(  
    spy.pn = px.pn and spy.sn = spx.sn  
  )  
)
```

```
px.pname where forall spx(  
  spx.pn='p3' and forall Spy(  
    Spy.sn=Spx.sn and Spy.pn <> 'p3' and exists Px(  
      px.pn=Spy.pn  
    )  
  )  
)
```

```
px.pname where exists spx(  
  Spx.pn='p3' and exists spy(  
    spy.sn=Spx.sn and Spy.pn<>'p3' and exists px(  
      px.pn=spy.pn)  
    )  
)
```

نام قطعاتی را بیابید که اگر عرضه‌کننده‌ای قطعهٔ p3 را عرضه کرده‌اند آن قطعه را نیز

عرضه کرده باشند. پاسخ‌های نادرست (II)

```
px.pname where forall spx(  
  spx.pn = 'p3' or exists spy(  
    spy.sn = spx.sn and spy.pn = px.pn  
  )  
)
```

```
px.pname where forall spx(  
  spx.pn = 'p3' and exists spy(  
    spy.pn = spx.pn and spy.pn = px.pn  
  )  
)
```

```
px.pname where forall spx(  
  spx.pn = 'p3' or exists spy(  
    spy.pn = spx.pn and spy.sn = spx.sn  
  )  
)
```

نام قطعاتی را بیابید که اگر عرضه‌کننده‌ای قطعهٔ p_3 را عرضه کرده باشد آن قطعه را نیز عرضه کرده باشند. پاسخ‌های نادرست (I)

نام قطعاتی را بیابید که اگر عرضه‌کننده‌ای قطعهٔ p_3 را عرضه کرده باشد فقط آن قطعه را نیز عرضه کرده باشند. پاسخ‌های نادرست (I)

نام قطعات عرضه شده‌ای را بیابید که همهٔ عرضه‌کنندگان آن قطعات قطعهٔ p_3 را هم حتماً عرضه کرده‌اند و اگر هیچ عرضه‌ای از p_3 نبود جواب باید خالی باشد. پاسخ‌های نادرست (I)

نام قطعاتی را بیابید که فقط عرضه‌کنندگانی که قطعهٔ p_3 را عرضه کرده‌اند آن قطعه را نیز عرضه کرده باشند.

- S(sn,sname,status,city) ,
- P(pn,pname,color,weight,city) ,
- J(jn,jname,budget,city)
- SPJ(sn, pn, jn, qty)

نام قطعاتی را بیابید که در همه پروژه‌ها به کار گرفته شده باشند.

نام قطعاتی را بیابید که برای همه پروژه‌ها عرضه‌ای از آن قطعه وجود داشته باشد.

نام قطعاتی را بیابید که پروژه‌ای وجود نداشته باشد که عرضه‌ای از آن قطعه برای آن

پروژه وجود نداشته باشد.

```
px.pname where forall jx(
  exists spjx(
    spjx.jn = jx.jn and px.pn = spjx.pn
  )
)
```

```
px.pname where not exists jx(
  not existst spjx(
    spjx.jn = jx.jn and px.pn = spjx.pn
  )
)
```

نام قطعاتی را بیابید که در همه پروژه‌ها به کار گرفته شده باشند. پاسخ‌های نادرست

```
px.pname where forall spjx(spjx.pn = px.pn)
```

```
px.pname where forall SPJx(  
  exists (  
    SPJx.pn=px.pn and px.pn=SPJx.pn  
  )  
)
```

```
px.pname where not exist spjx(  
  spjx.pn=px.pn and not exist spjy(  
    spjy.pn=spjx.pn and spjy.pn=px.pn  
  )  
)
```

```
px.pname where forall px(  
  exists spj(spj.pn = px.pn)  
)
```

نام نویسندگانی را بیابید که همه کتاب‌های‌شان را در این کتابخانه به امانت گرفته باشند.

- book([bn](#) , title, category, fpd, author)
- member([mn](#) , name , category, bn)
- borrow([bn](#) , [mn](#) , nd , rdt, ret)

پاسخ‌های نادرست

```
bookx.author where forall bookx(  
  exists borrowx(bookx.bn = borrowx.bn)  
)
```

```
bookx.author where not exist memberx(  
  memberx.bn=bookx.bn and  
  not exist borrowx(  
    borrowx.bn=memberx.bn and  
    borrowx.bn=bookx.bn  
  )  
)
```

نام نویسندگانی را بیابید که همه کتاب‌های‌شان را در این کتابخانه به امانت گرفته باشند.

پاسخ‌های نادرست

```
bookx.author where forall booky(  
  bookx.author=booky.author and  
  exist borrowx(borrowx.bn=bookx.bn)  
)
```

پاسخ‌های درست

```
bookx.author where not exists booky(  
  booky.author = bookx.author and  
  not exists borrowx(borrowx.bn = booky.bn)  
)
```

```
bookx.author where forall booky(  
  bookx.author <> booky.author or  
  exist borrowx(borrowx.bn=bookx.bn)  
)
```

References

abs.gov.au

<https://www.abs.gov.au/websitedbs/a3121120.nsf/home/statistical+language+-+quantitative+and+qualitative+>

questionpro

<https://www.questionpro.com/blog/quantitative-data/>

- https://www.ntu.edu.sg/home/ehchua/programming/sql/Relational_Database_Design.html
- <https://dev.to/lmolivera/everything-you-need-to-know-about-relational-databases-3ejl>
- <https://relational.fit.cvut.cz>
- <https://dzone.com/refcardz/from-relational-to-graph-a-developers-guide?chapter=1>
- https://www.ibm.com/support/knowledgecenter/zh/SSWU4L/Data/imc_Data/Data_q_a_watson_assistant/Relational
- <https://vincentarelbundock.github.io/Rdatasets/datasets.html>
- https://docs.oracle.com/cd/E15276_01/doc.20/e13677/dialogs_data_modeling.htm
- <http://www.acc.umu.se/~cthulhu/postgreSQL/tutorial/sql608.htm>
- https://fmhelp.filemaker.com/help/18/fmp/en/index.html#page/FMP_Help/planning-databases.html
- https://en.wikipedia.org/wiki/Relational_database
- <https://web.csulb.edu/colleges/coe/cecs/dbdesign/dbdesign.php?page=manymany.php>
- [A video of C.J.Date](#)

END

