



Query Rewriting in Itemset Mining



Rosa Meo, Marco Botta, Roberto Esposito
University of Torino



Outline



- Motivations
 - constraint-based mining
 - iterative and interactive mining, inductive databases
- Query rewriting and reuse of result sets
- Constraints properties
 - Item dependent constraints
 - Context dependent constraints
- Examples
- Conclusions



Motivations

- Knowledge Discovery from Databases (KDD) is usually an interactive and iterative process
- This sequence consists in constraint-based queries which are very often a refinement of previous ones
- **Problem**
Each new query cannot be executed from scratch: unfeasible workload for the extraction engine
- **Solution**
The problem can be solved materializing previous queries and adopting an “incremental” engine, in the sense that it can derive the result of a query Q reusing the result of previous, “correlated” queries.



Optimizing Mining Queries

- The novel languages for data mining need for optimizers
- can exploit the available informations in the database:
 - database schema (keys and functional dependencies)
 - the indices (*mining indices*)
 - results of previous, correlated queries



A Generic Mining Language



A very generic constraint-based mining query requests

- extraction from a source table



- $R=Q(T,G,I,\Gamma(M),\Xi)$

A Generic Mining Language

A very generic constraint-based mining query requests

- extraction from a source table
- of a set of items (on some schema)

■ $R=Q(T,G,I,\Gamma(M),\Xi)$

A Generic Mining Language

A very generic constraint-based mining query requests

- extraction from a source table
- of a set of items (on some schema)
- satisfying some user defined constraints (mining constraints)

■ $R=Q(T,G,I,\Gamma(M),\Xi)$





A Generic Mining Language



A very generic constraint-based mining query requests

- extraction from a source table
- of a set of items (on some schema)
- satisfying some user defined constraints (mining constraints)
- from the groups of the database (grouping constraints)



■ $R=Q(T,G,I,\Gamma(M),\Xi)$

A Generic Mining Language

A very generic constraint-based mining query requests

- extraction from a source table
- of a set of items (on some schema)
- satisfying some user defined constraints (mining constraints)
- from the groups of the database (grouping constraints)
- The number of such groups must be sufficient (user defined statistical evaluation measure, such as support)

■ $R=Q(T,G,I,\Gamma(M),\Xi)$

An Example

■ $R=Q(\text{purchase}, \text{customer}, \text{item}, \text{price} > 100, \text{support_count} \geq 2)$

purchase

transaction	customer	item	date	price	quantity
1	1001	ski_pants	12/7/98	140	1
1	1001	hiking_boots	12/7/98	180	1
3	1001	jacket	13/7/98	300	1
2	2256	col_shirt	13/7/98	25	2
2	2256	brown_boots	13/7/98	150	1
2	2256	jacket	13/7/98	300	1
4	2256	col_shirt	14/7/98	25	3
4	2256	jacket	14/7/98	300	2

Mining Query

R

itemset	support_count
{jackets}	2


Relationships Between Two Queries Q_1 and Q_2

- **Equivalence:** no computation is needed, because $R_1 = R_2$
- **Inclusion:** R_2 can be generated from R_1 by means of a simple selection – common elements have the same statistical measures
- **Dominance:** R_2 can be generated from R_1 by means of some accesses to the database, needed in order to determine the correct statistical measures in R_2

Database tight-coupling

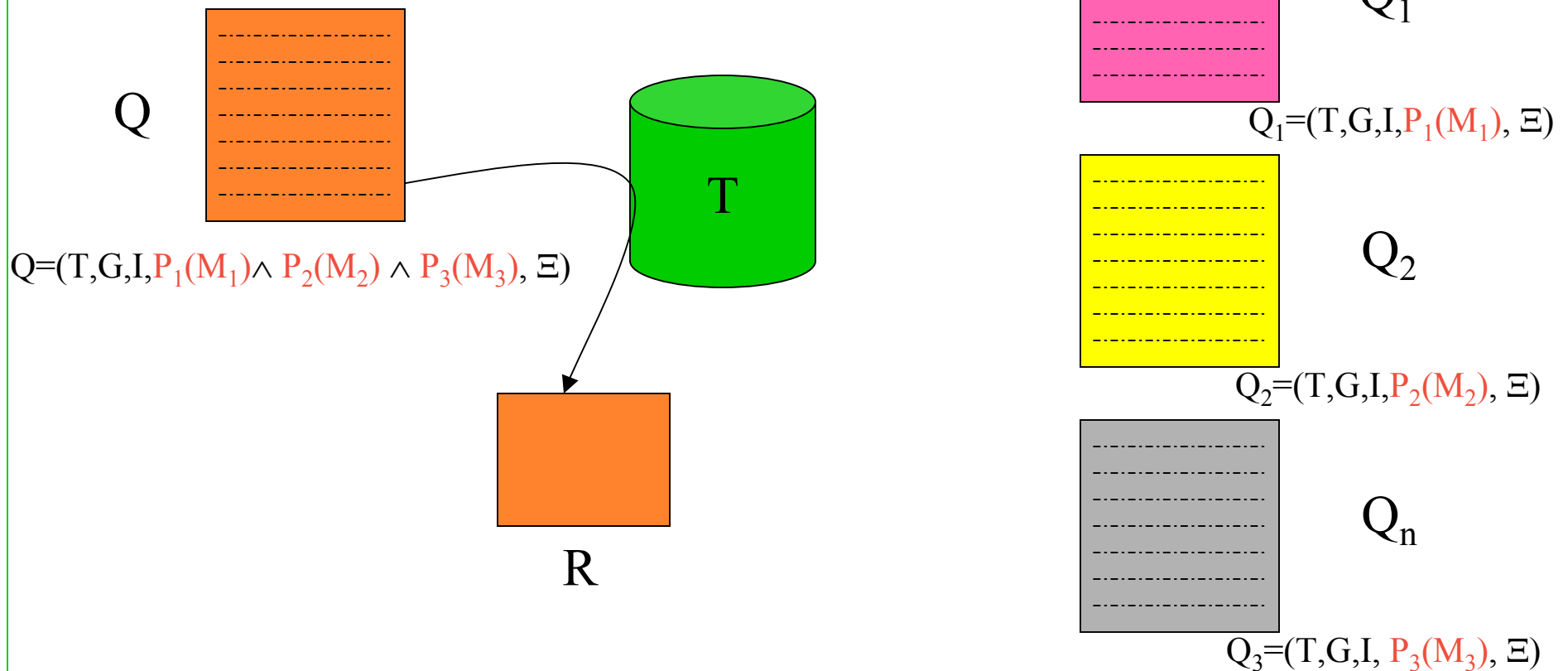


Searching for Equivalence

- 
- In order to allow the optimizer to recognize equivalent queries, we allow *query rewriting*
 - Query rewriting:
determination of a relational expression on a set of other queries whose result is equivalent to the result of the rewritten query but is better in terms of execution costs

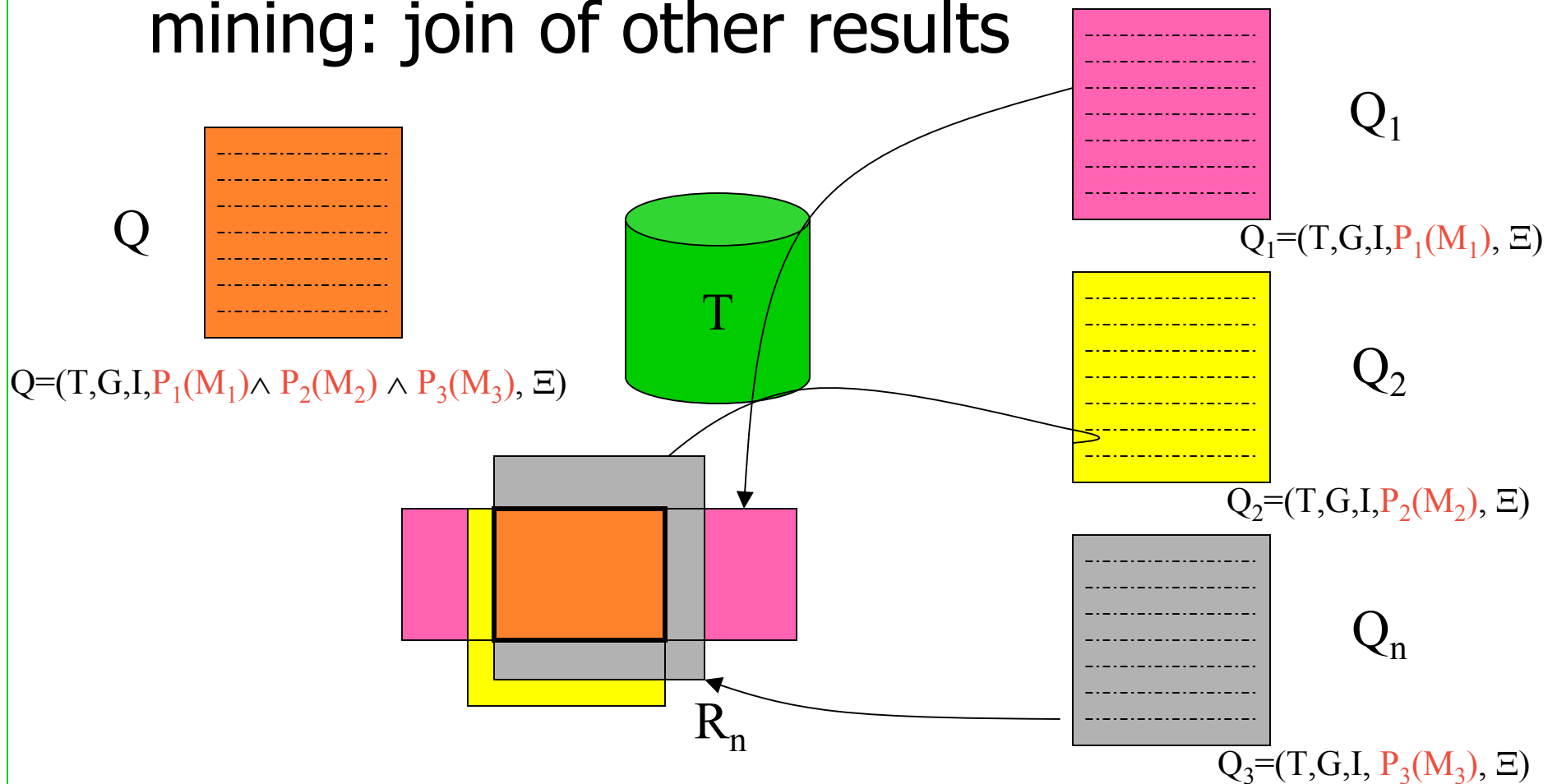
Query Rewriting

- Query rewriting of a query for itemset mining: join of other results



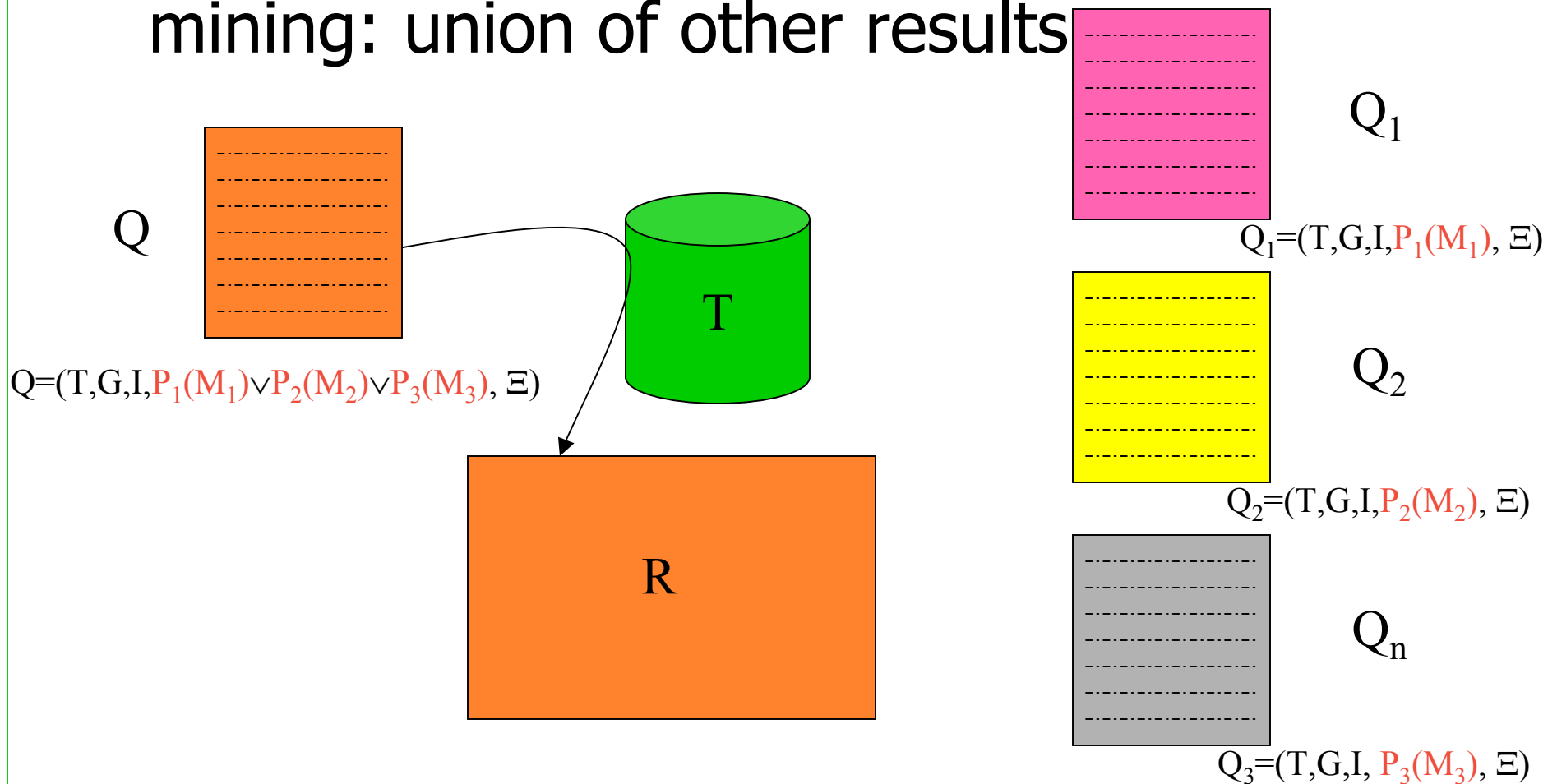
Query Rewriting

- Query rewriting of a query for itemset mining: join of other results



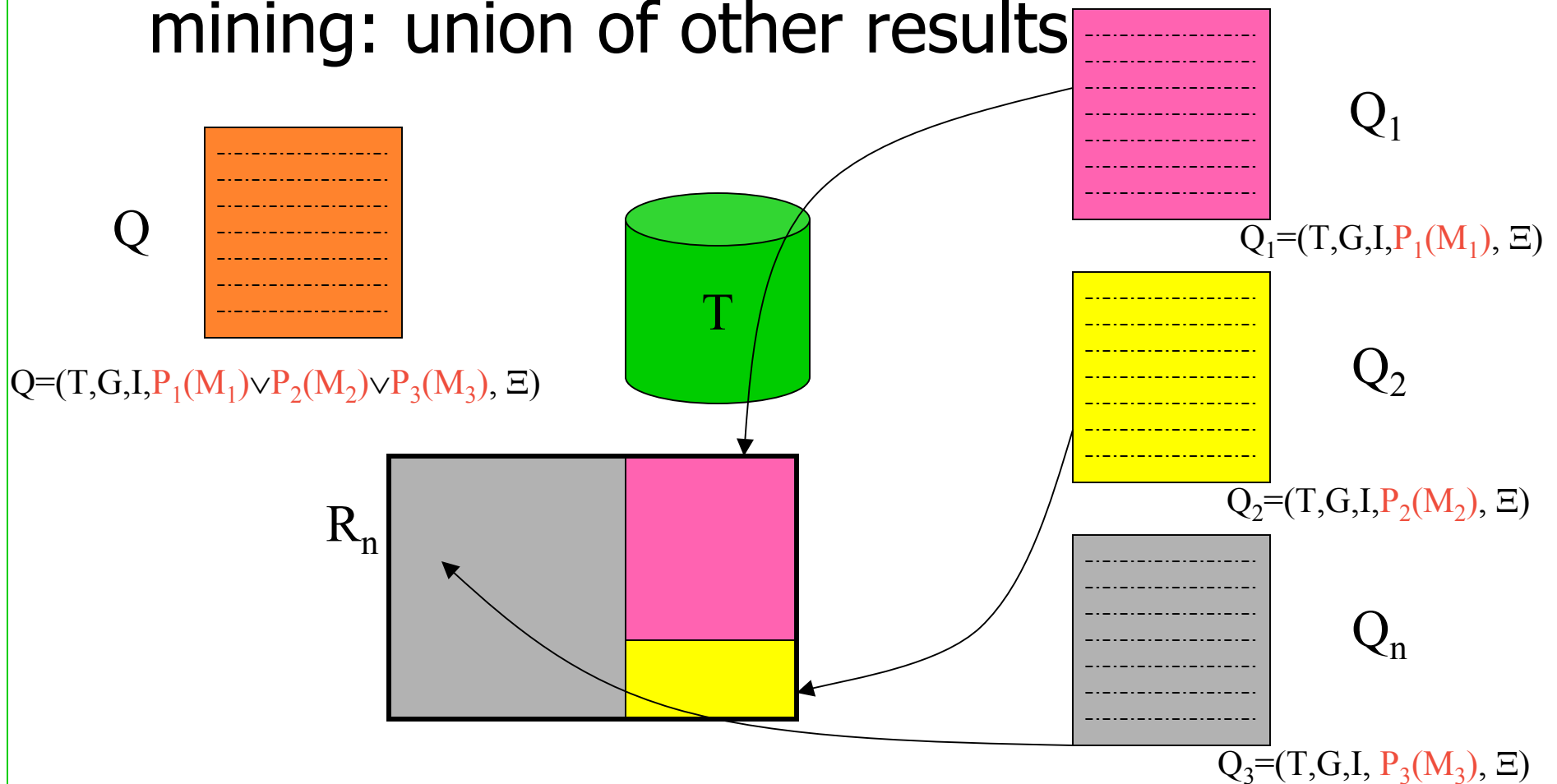
Query Rewriting (2)

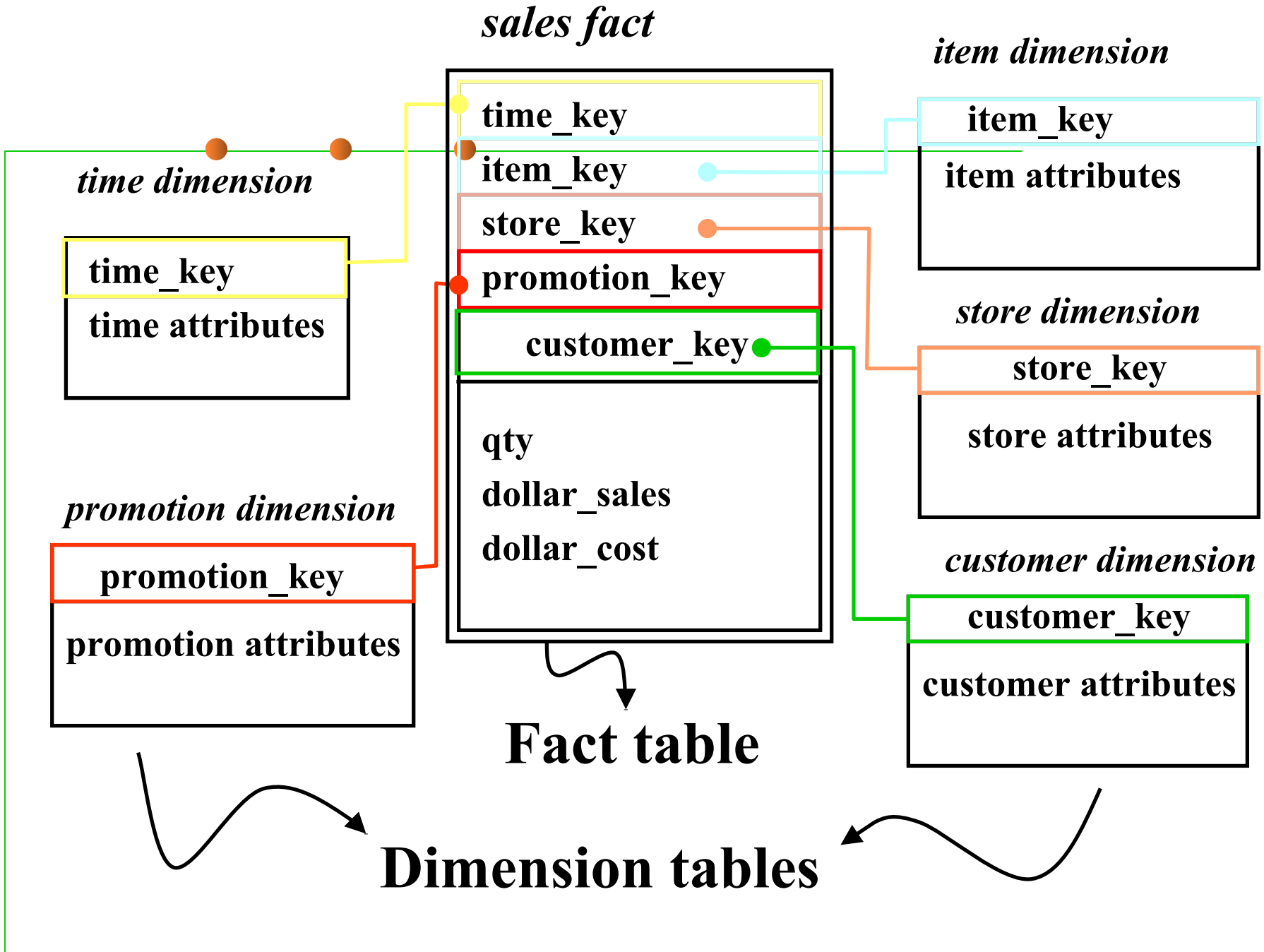
- Query rewriting of a query for itemset mining: union of other results



Query Rewriting (2)

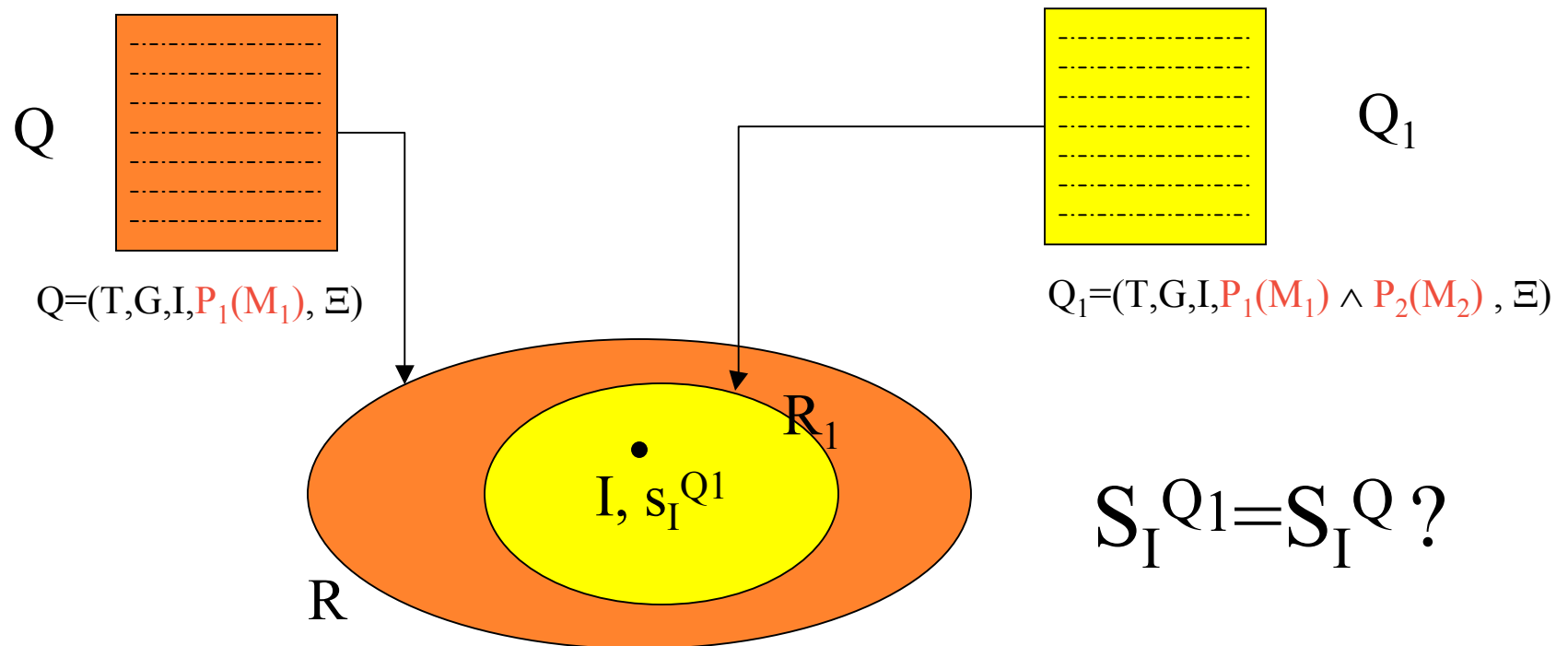
- Query rewriting of a query for itemset mining: union of other results





Query Rewriting

- Can also be used to recognize inclusion and dominance



Answer

- $Q_1 = (T, G, I, P_1(M_1), \Xi)$
- $Q_2 = (T, G, I, P_2(M_2), \Xi)$
- Support of an itemset in results of Q_1 and Q_2 is the same if there exist

$$I \rightarrow M_1 \quad \text{and} \quad I \rightarrow M_2$$

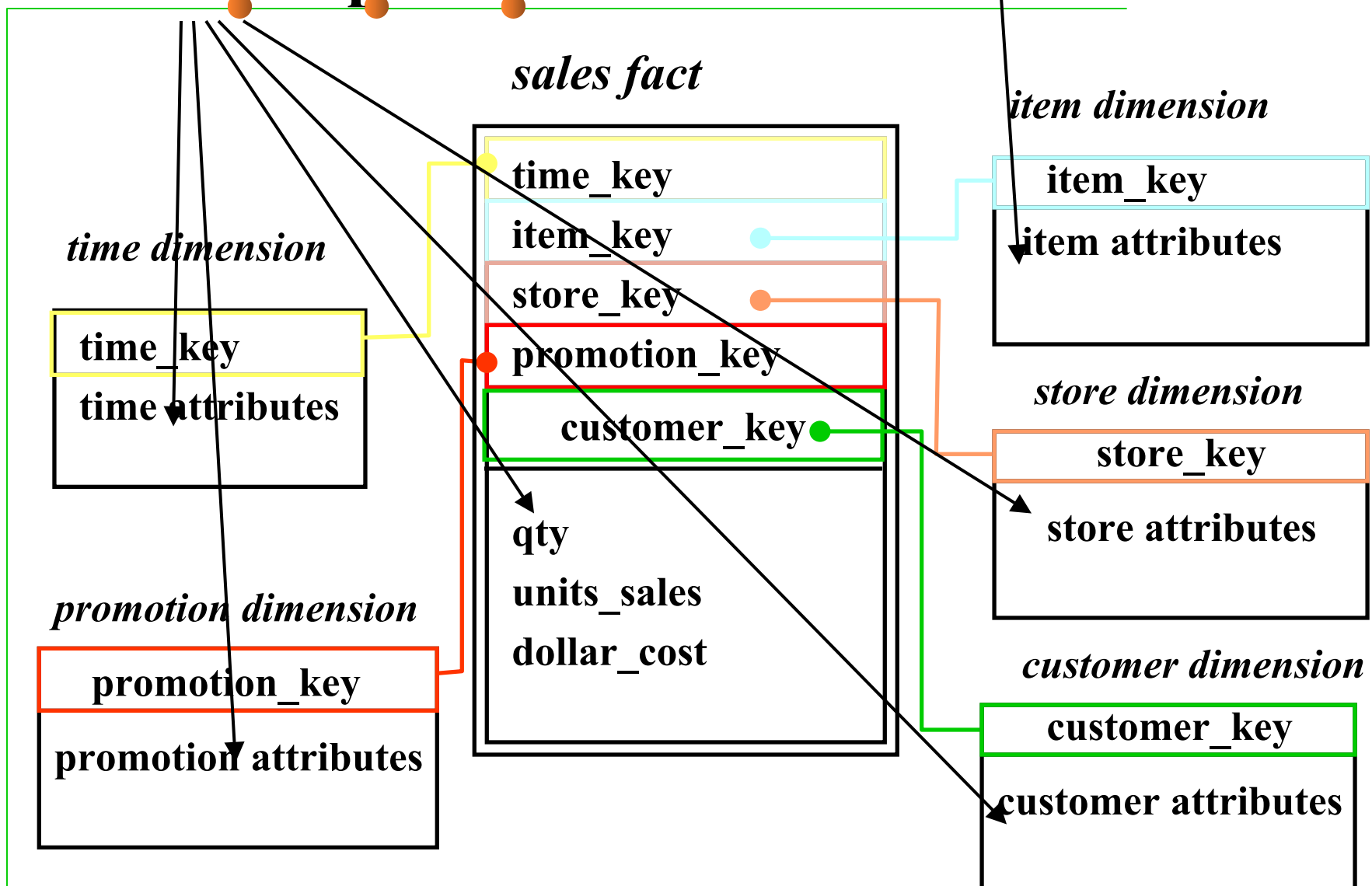
- We call $P_1(M_1)$ and $P_2(M_2)$
item dependent constraints

Context Dependent Constraints

- $Q_1 = (T, G, I, P_1(M_1), \Xi)$
- $Q_2 = (T, G, I, P_2(M_2), \Xi)$
- Support of an itemset in results of Q_1 and Q_2 is not necessarily the same if
$$I \not\rightarrow M_1 \text{ or } I \not\rightarrow M_2$$
- If $I \not\rightarrow M_1$ $P_1(M_1)$ is a *context dependent constraint*

Item dependent constraints

Context dependent constraints



Candidate Rewriting Sets

- $Q = (T, G_1, I, \Gamma(M), \Xi)$
- $Q_1 = (T, G_1, I_1, \Gamma_1(M_1), \Xi_1)$
- $Q_2 = (T, G_2, I_2, \Gamma_2(M_2), \Xi_2)$
- $\{Q_1, Q_2\}$ is a candidate rewriting set for Q if
 - I_1, I_2 and I are in the same grouping equivalence class
 - G_1, G_2 and G are in the same grouping equivalence class
 - Ξ_1, Ξ_2 and Ξ are logically equivalent expressions

Main Result

- $Q=(T,G_1,I,\Gamma(M),\Xi)$
- $Q_1=(T,G_1,I_1,\Gamma_1(M_1),\Xi_1)$
- $Q_2=(T,G_2,I_2,\Gamma_2(M_2),\Xi_2)$
 - If there exist
$$\left\{ \begin{array}{l} I' \subseteq I, I' \rightarrow M \\ I'_1 \subseteq I_1, I'_1 \rightarrow M_1 \\ I'_2 \subseteq I_2, I'_2 \rightarrow M_2 \end{array} \right.$$
- $\Gamma(M)=\Gamma_1(M_1)\wedge\Gamma_2(M_2) \rightarrow R=R_1\cap R_2$
- $\Gamma(M)=\Gamma_1(M_1)\vee\Gamma_2(M_2) \rightarrow R=R_1\cup R_2$
 - and result of Ξ_1, Ξ_2 and Ξ , evaluated for each common element in R, R_1 and R_2 , are the same.



Example

- $Q_1 = (\text{Fact table}, tr, \text{item}, \emptyset, \text{support_count} \geq 2)$
- $Q_2 = (\text{Fact table}, tr, \text{item}, \text{category} = \text{computer}, \text{support_count} \geq 2)$

Fact table

item dimension

Tr	item_k	qty	data
1	hd	✓	
	video		
	...		
2	hd	✓	
	video		
	jacket	F	
	pants		
...			
3	jacket	F	
	pants		
	...		
4	hd		
	video	✓	
...			

item_k	price	category	...
hd	500		
video	700	computer	
jacket	300		
pants	200	wear	
....			

category=computer ?

itemset₁ = {jacket, pants}

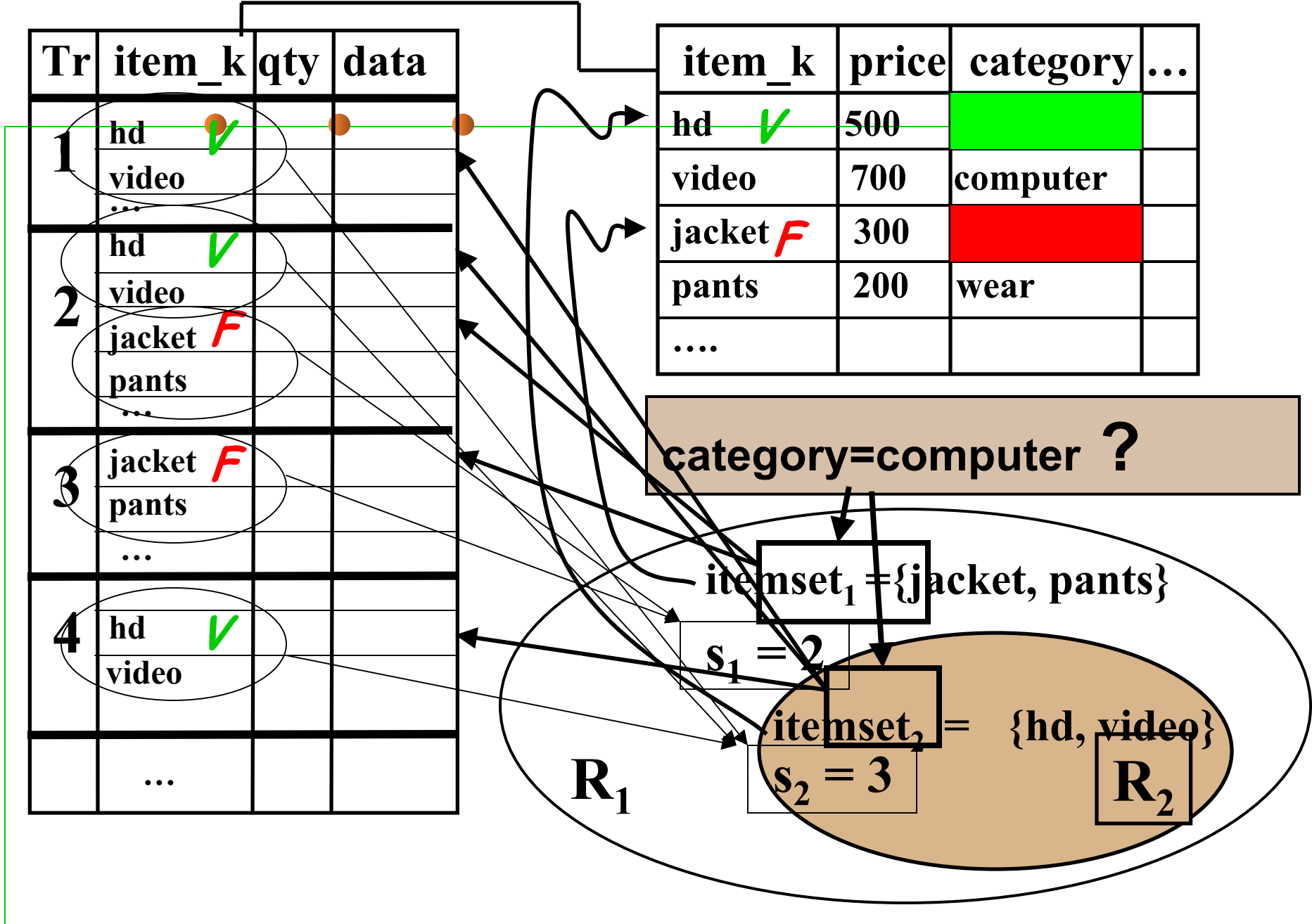
s₁ = 2

itemset₂ = {hd, video}

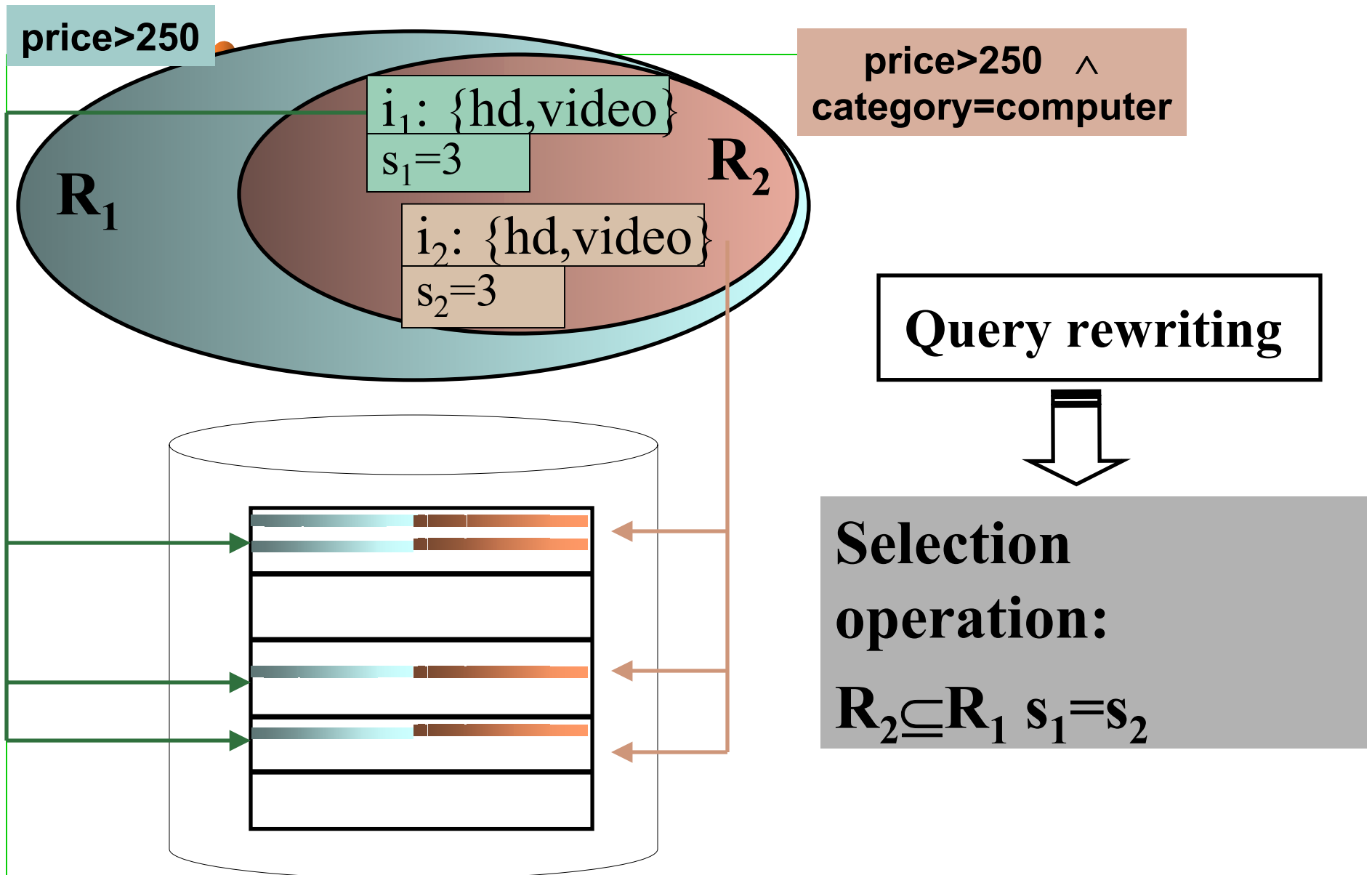
s₂ = 3

R₁

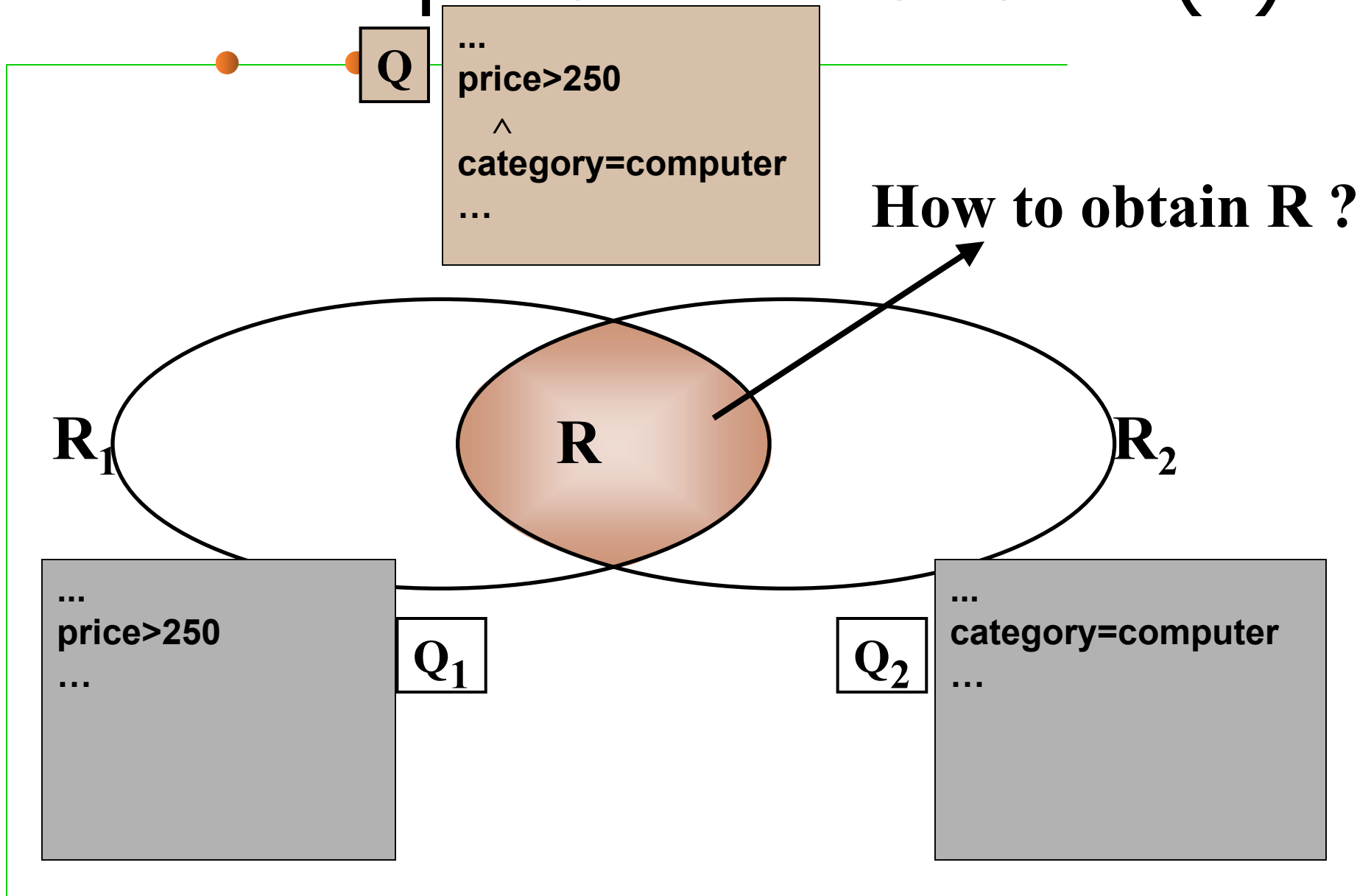
R₂



Item Dependent Predicates



Item Dependent Predicates (2)

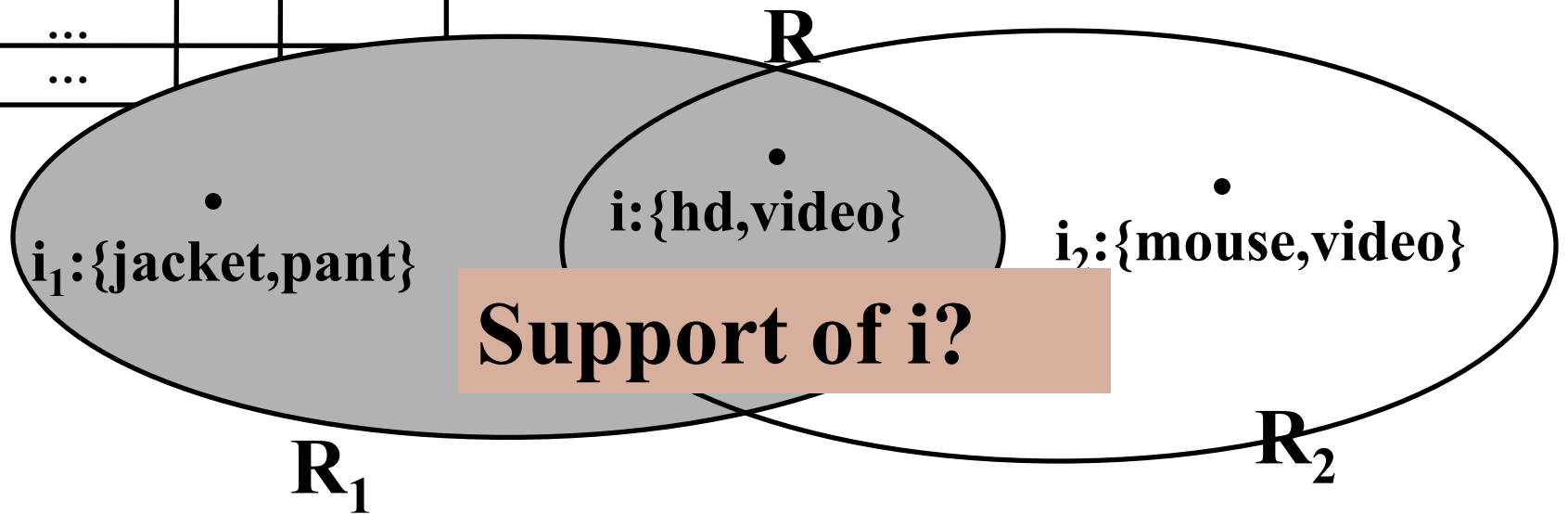


Fact table

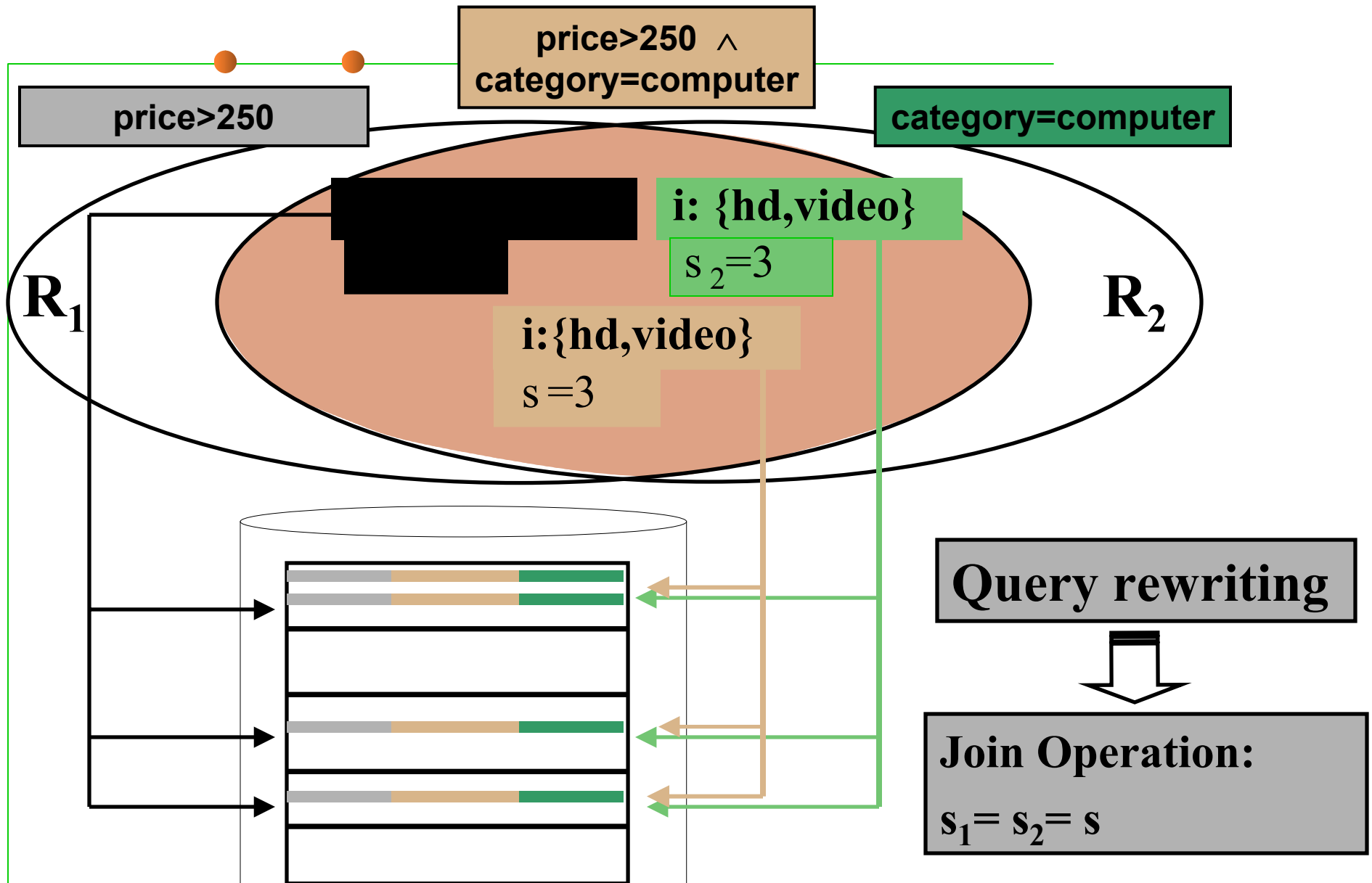
item dimension

Tr	item_k	qty	date
1	hd		
	video		
	mouse		
2	hd		
	video		
	jacket		
	pants		
3	...		
	jacket		
	pants		
	...		
	...		

item_k	price	category	...
hd	500	computer	
video	700	computer	
mouse	50	computer	
jacket	300	wear	
pants	200	wear	
...			



Item Dependent Predicates (2)



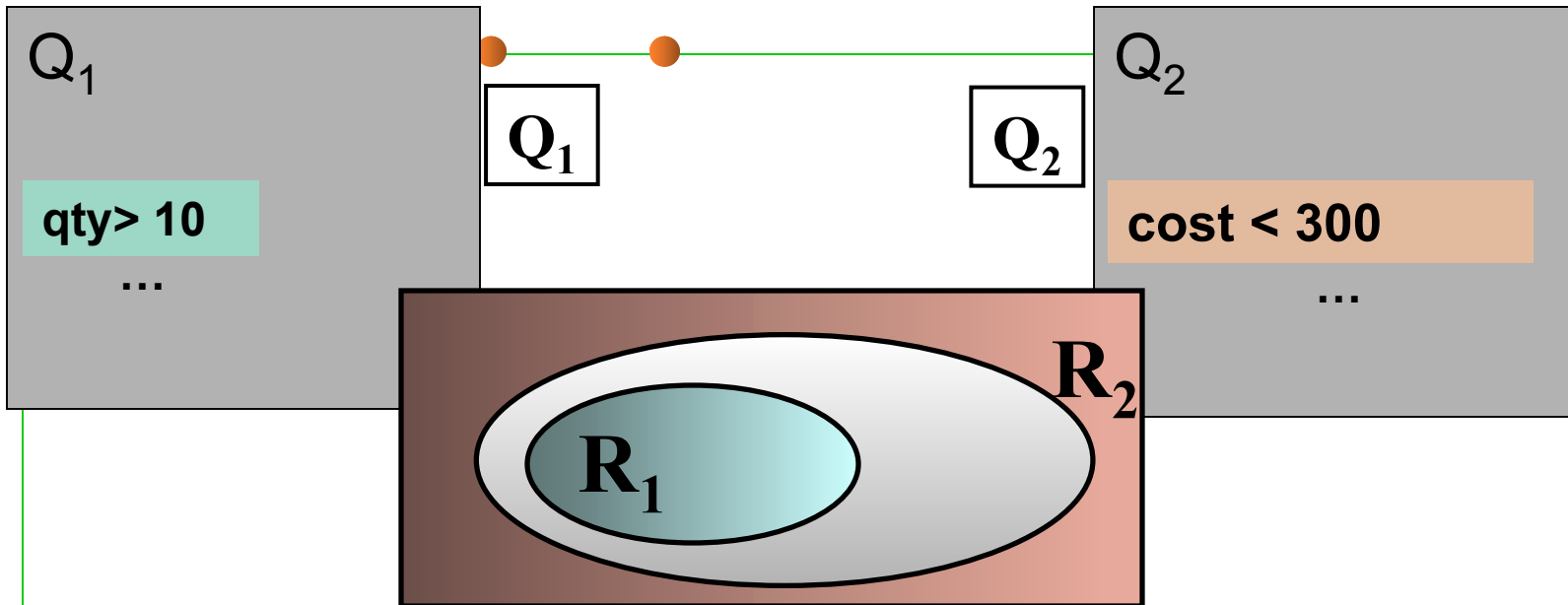
Main Result with Context Dependent Constraints

- $Q=(T,G_1,I,\Gamma(M),\Xi)$
- $Q_1=(T,G_1,I_1,\Gamma_1(M_1),\Xi_1)$
- $Q_2=(T,G_2,I_2,\Gamma_2(M_2),\Xi_2)$

- If $\Gamma_1(M_1) \rightarrow \neg\Gamma_2(M_2)$

- $\Gamma(M)=\Gamma_1(M_1)\wedge\Gamma_2(M_2) \rightarrow R=R_1\cap R_2= \emptyset$
- $\Gamma(M)=\Gamma_1(M_1)\vee\Gamma_2(M_2) \rightarrow R=R_1\cup R_2$
 - and $\Xi =\Xi_1+\Xi_2$ for each element in R

Context Dependent Predicates



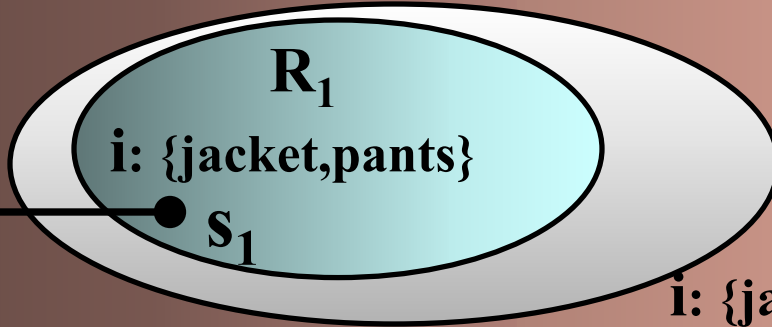
$item \not\rightarrow qty$ $item \not\rightarrow cost$

f.d. : $qty > 10 \rightarrow \neg cost < 300$

$R_{join} = Q(\text{Fact table, cust, item, } qty > 10 \wedge cost < 300)$
 $R_{union} = Q(\text{Fact table, cust, item, } qty > 10 \vee cost < 300)$

?

f.d. : qty>10 → ¬ cost<300



P₁: qty>10

P₂: cost<300

qty	cost	item
8	380	jacket
9	310	pants
12	400	jacket
11	380	pants
7	280	jacket
8	250	pants
6	250	jacket
9	270	pants

$R_{\text{union}} = Q(\dots, \text{item}, P_1 \vee P_2, \dots)$

Union of the results:
for common itemsets
 $s = s_1 + s_2$

Side Effects of Context Dependent Predicates

- $Q = (\text{Purchase}, \text{tr}, \text{item}, \min(\text{qty}) < 4, \text{support_count} \geq 2)$

tr	item	qty
1	A	5
1	B	3
2	A	2
2	B	4

Effects on
anti-monotonicity
of support

- $\{A\}$ and $\{B\}$ satisfying $\min(\text{qty}) < 4$ do not satisfy support_count constraint (count=1)
- $\{A, B\}$ satisfying $\min(\text{qty}) < 4$ satisfies support_count constraint (count=2)

Side Effects of Context

Dependent Predicates (2)

- $Q = (\text{Purchase}, \text{tr}, \text{item}, \max(\text{qty}) \geq 4, \text{support_count} \geq 2)$

tr	item	qty
1	A	5
1	B	3
2	A	4
2	B	4
3	A	3
3	B	4

Effects on
anti-monotonicity
of support

- $\{A\}$ and $\{B\}$ satisfying $\max(\text{qty}) \geq 4$ satisfy support_count constraint (count=2)
- $\{A, B\}$ satisfying $\max(\text{qty}) \geq 4$ satisfies support_count constraint with a *higher support* (count=3)



Conclusions

- We have proposed query rewriting for itemset mining as a promising strategy
- We defined item and context dependent constraints and studied their role in query rewriting
- Experiments on a preliminary version of an optimizer show that this approach is feasible and in many cases absolutely necessary to speed up execution times in inductive databases and data mining.

Experiments

