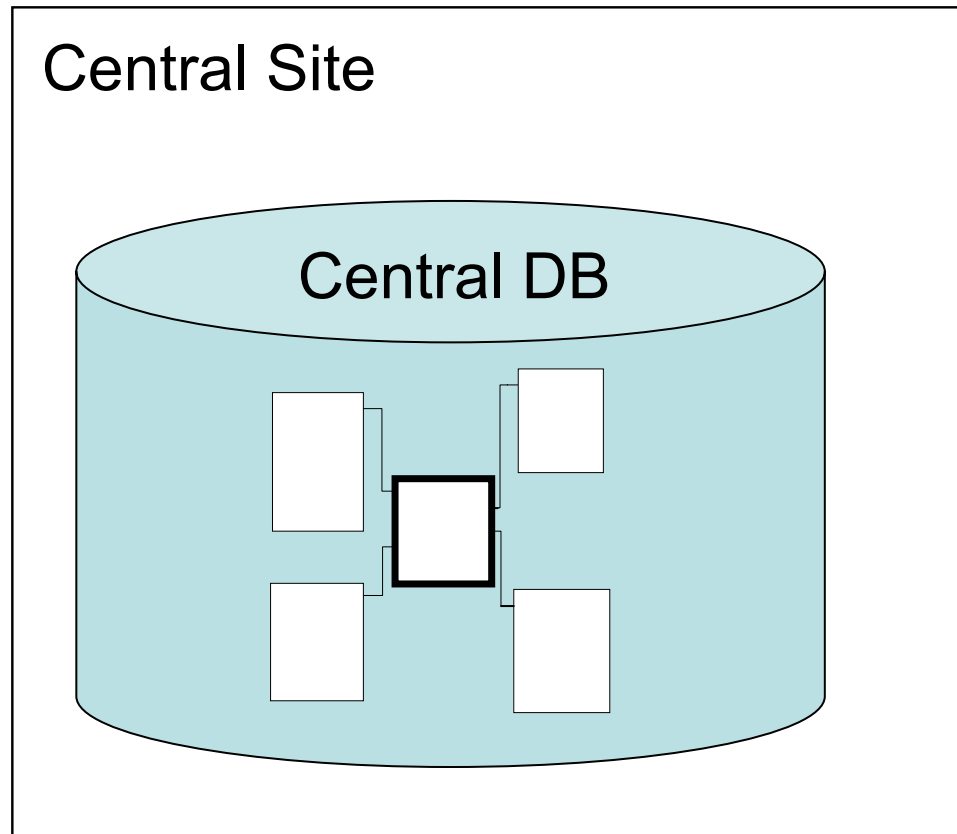


# Mining Qualified Association Rules in Distributed Databases

Svetlozar Nestorov  
The University of Chicago

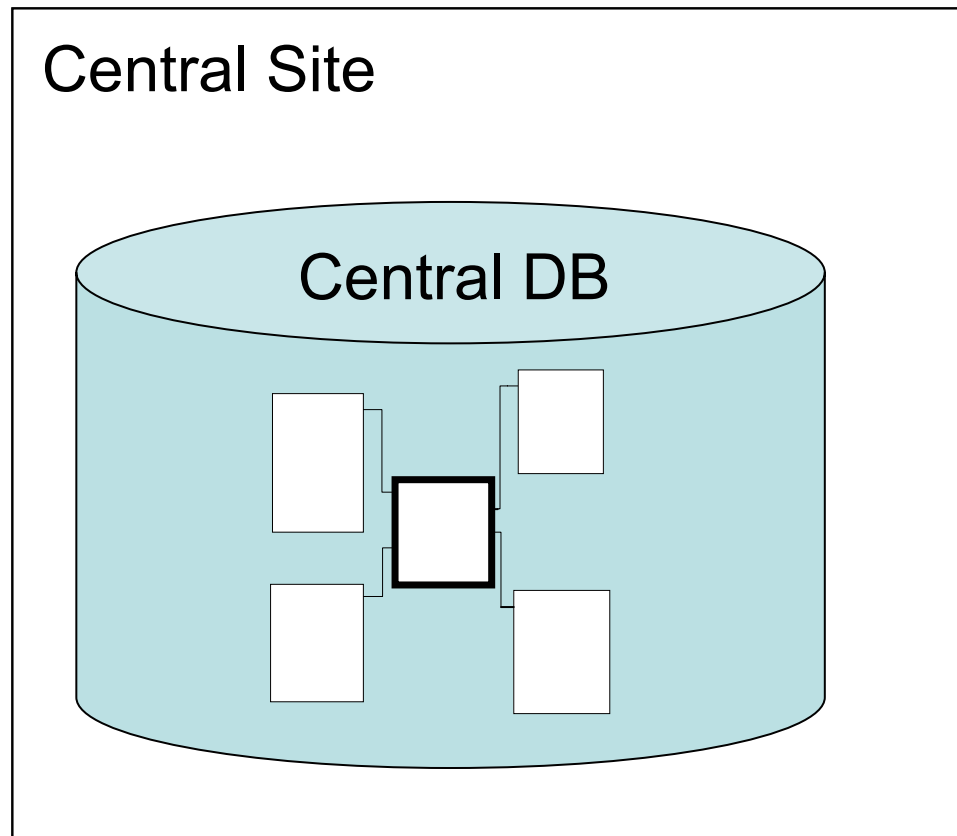
# Example Scenario

- What products are frequently bought together...



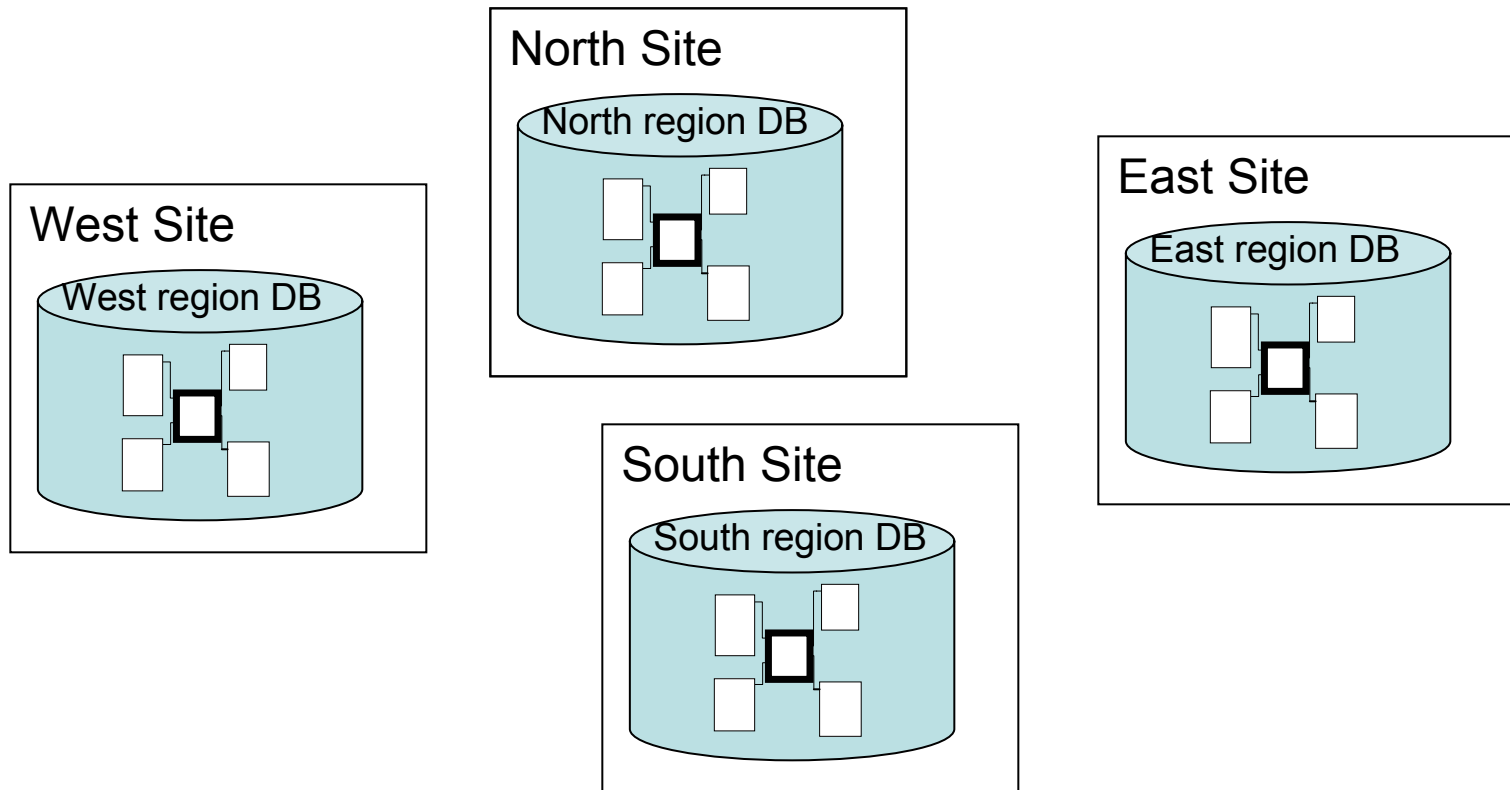
# Example Scenario

- What products are frequently bought together...  
in the same season, by customers of the same age group?



# Example Scenario

- What products are frequently bought together...  
in the same season, by customers of the same age group?



# Outline

- Association Rules
- Star Schema and Data Warehousing
- Qualified Association Rules (QAR)
- Tightly-Coupled Framework
- QAR for Distributed Databases ...  
    **and the GRID**
- Future Work

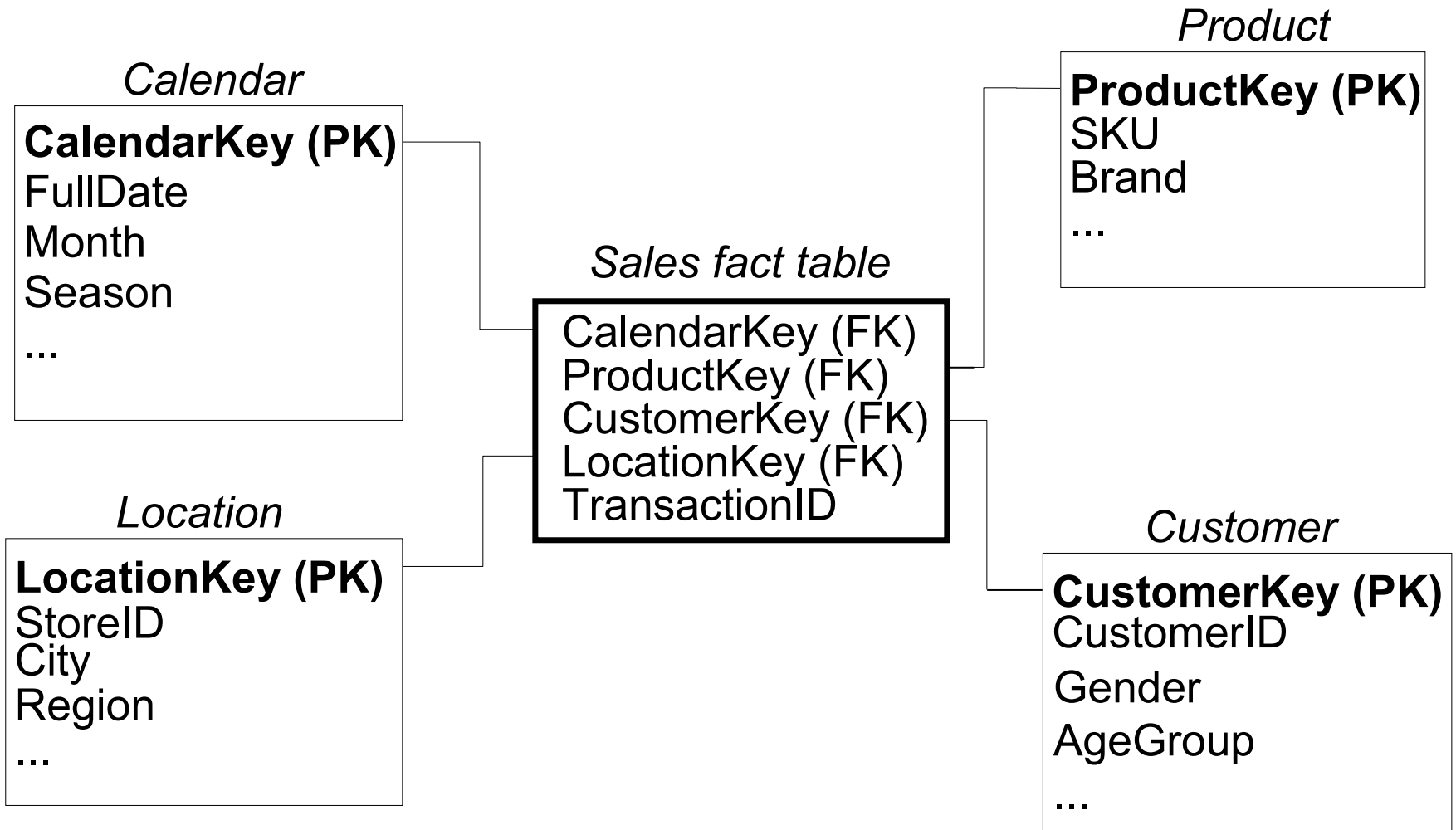
# Association Rules

- Market Basket Data
  - Sets (transactions, baskets) of items.
- Standard Definition
  - Example: {gloves, scarf} → {hat}
- Support and Confidence
- Frequent (high support) itemsets
  - What items appear together (in the same basket) frequently?

# Why Association Rules Work

- Appropriateness of purpose
  - Understandable results
- Wide applicability in many domains
  - Genomics, economics, astronomy, etc.
- Optimization techniques
  - Apriori (levelwise pruning)
- Where do market basket data come from?
  - Databases, often with a star schema

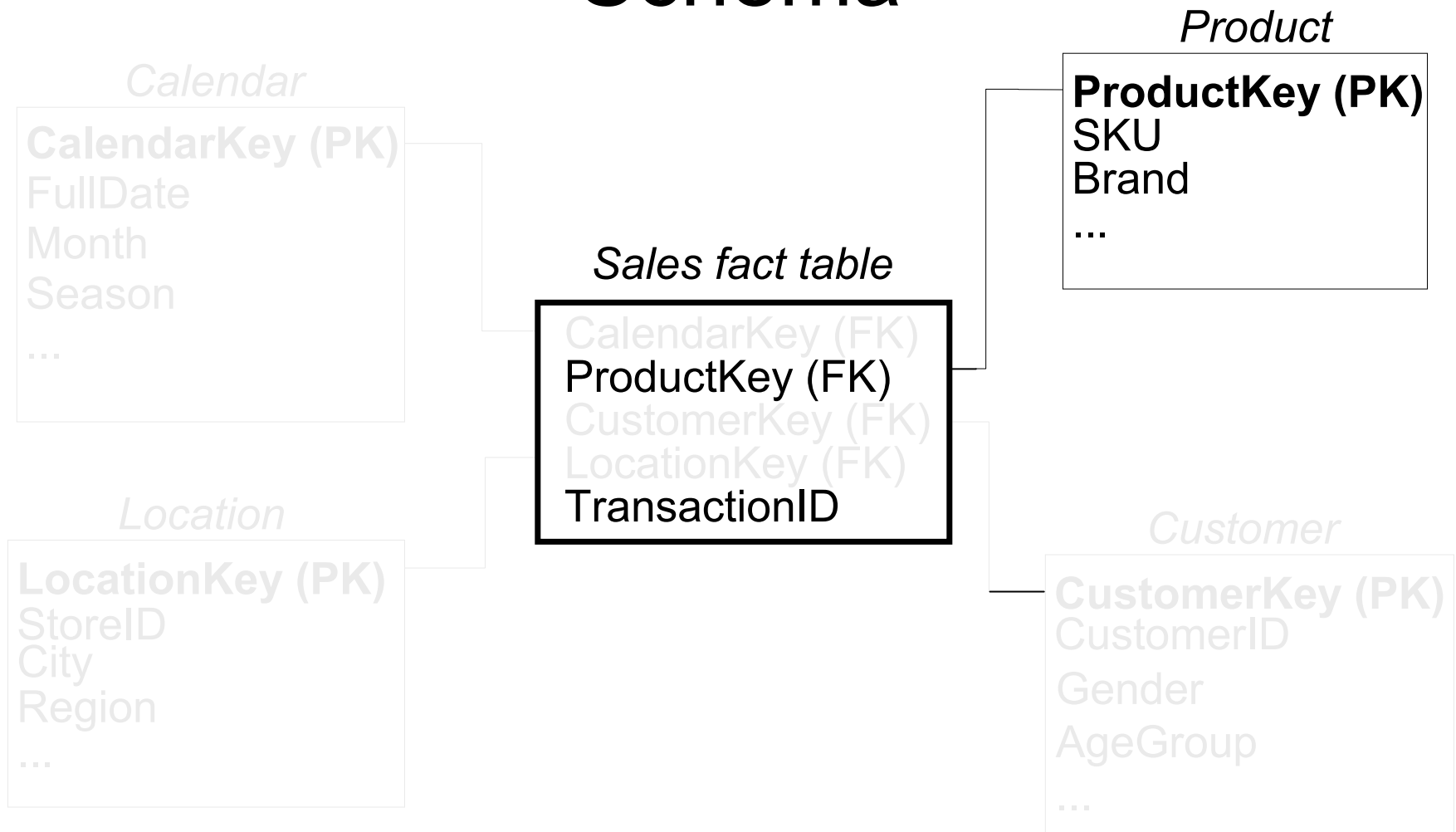
# Star Schema



# Data Warehousing

- Standard practice
- Relatively complex and expensive
  - Powerful hardware and software, often underutilized
- Dimensional modeling
  - Results in star schema databases
- Analyzed with OLAP

# Association Rules over Star Schema



# Qualified Association Rules (QAR)

- Association rules with qualified dimensions
  - Correlations among values of multiple attributes in multiple tables.
- Example
  - $\{\text{gloves, scarf}\} \rightarrow \{\text{hat}\}$  qualified with  
 $\text{Season} = \text{winter}, \text{AgeGroup} = 35\_45$
- Support and Confidence

# Mining Qualified Association Rules

- Specify dimensional attributes to be qualified.
- Specify support and confidence thresholds.
- Example:

What **products** are frequently bought together in the same **season**, by customers of the same **age group**? (with  $\text{supp} = 0.5\%$ ,  $\text{conf} = 50\%$ )

# QAR Advantages

- Add details (context) to association rules.
- Enable ad-hoc mining from different perspectives.
- May help overcome common problems:
  - too many rules,
  - too few rules.

# Optimization Technique

- More complex than Apriori (standard AR).
  - Partially ordered lattice of subqueries.
- Key principle:
  - If itemset  $X$  is frequent in transactions qualified with  $Z$  then Itemset  $Y$  is frequent in transactions qualified with  $W$ , for all subsets  $Y$  of  $X$  and subsets  $W$  of  $Z$ .

# Example

- {hat, scarf} is frequent for {Season = winter, AgeGroup = 35\_45}, then:
  - {hat, scarf} is frequent
  - {hat, scarf} is frequent for {Season = winter}
  - {hat} is frequent for {Season = winter, AgeGroup = 35\_45}
  - {scarf} is frequent for {AgeGroup = 35\_45}
  - ...

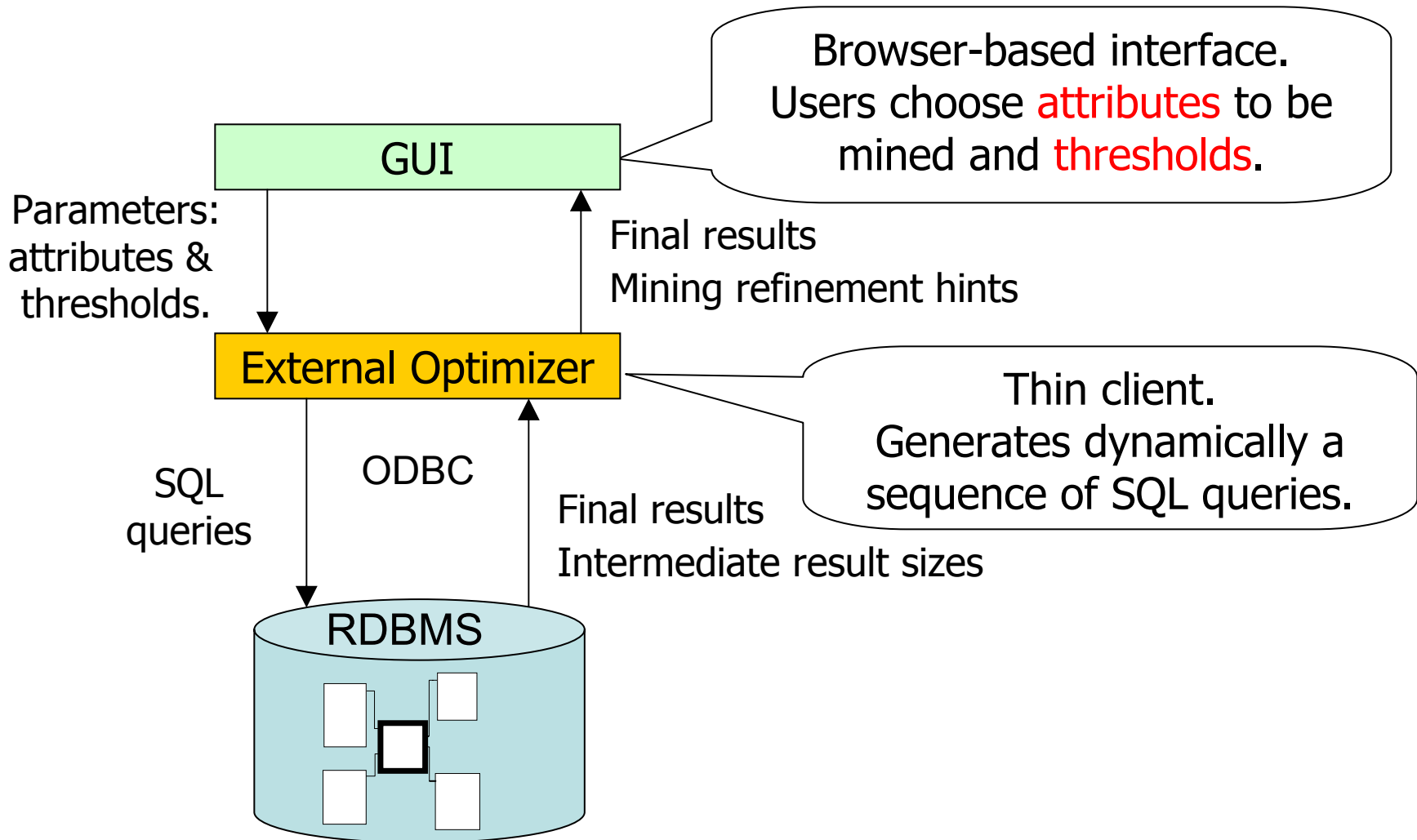
# Levelwise algorithm

- A plan is a sequence of subqueries.
- Doubly exponential search space
- Levelwise greedy heuristic:
  - Level 1: Find a plan to compute all frequent qualified items
  - Level 2: Find a plan to compute all frequent qualified pairs
  - ...
  - Level N: Find a plan to compute all frequent qualified N-itemsets

# Implementation Framework

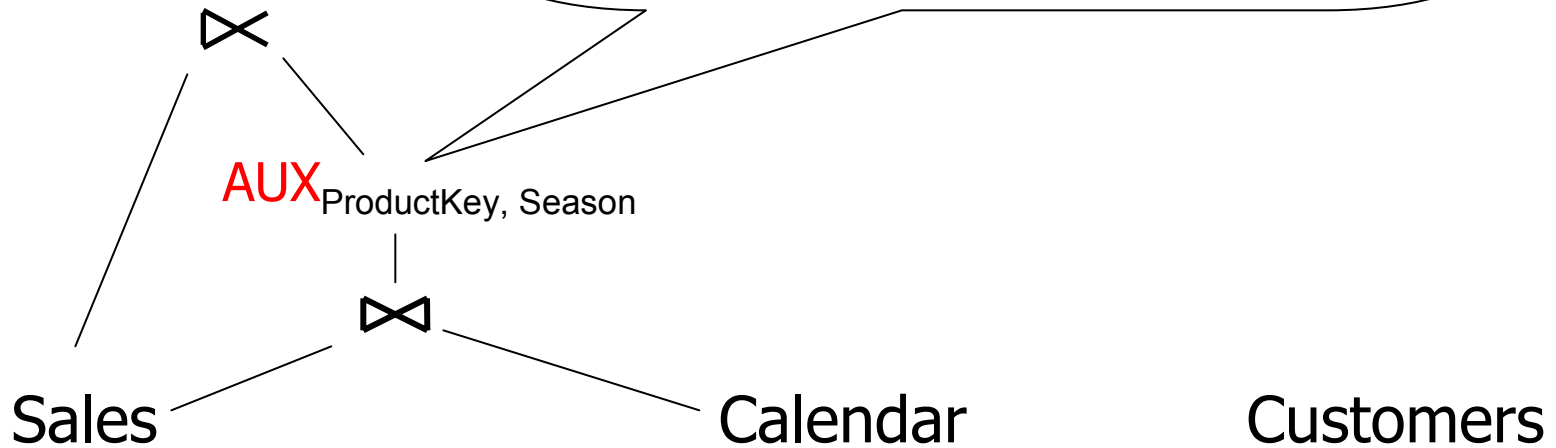
- Tightly-coupled with database systems
- Advantages:
  - Powerful hardware and software
  - Privacy
  - Security
  - Fast deployment
- Disadvantages
  - Performance?

# Tightly-Coupled Architecture

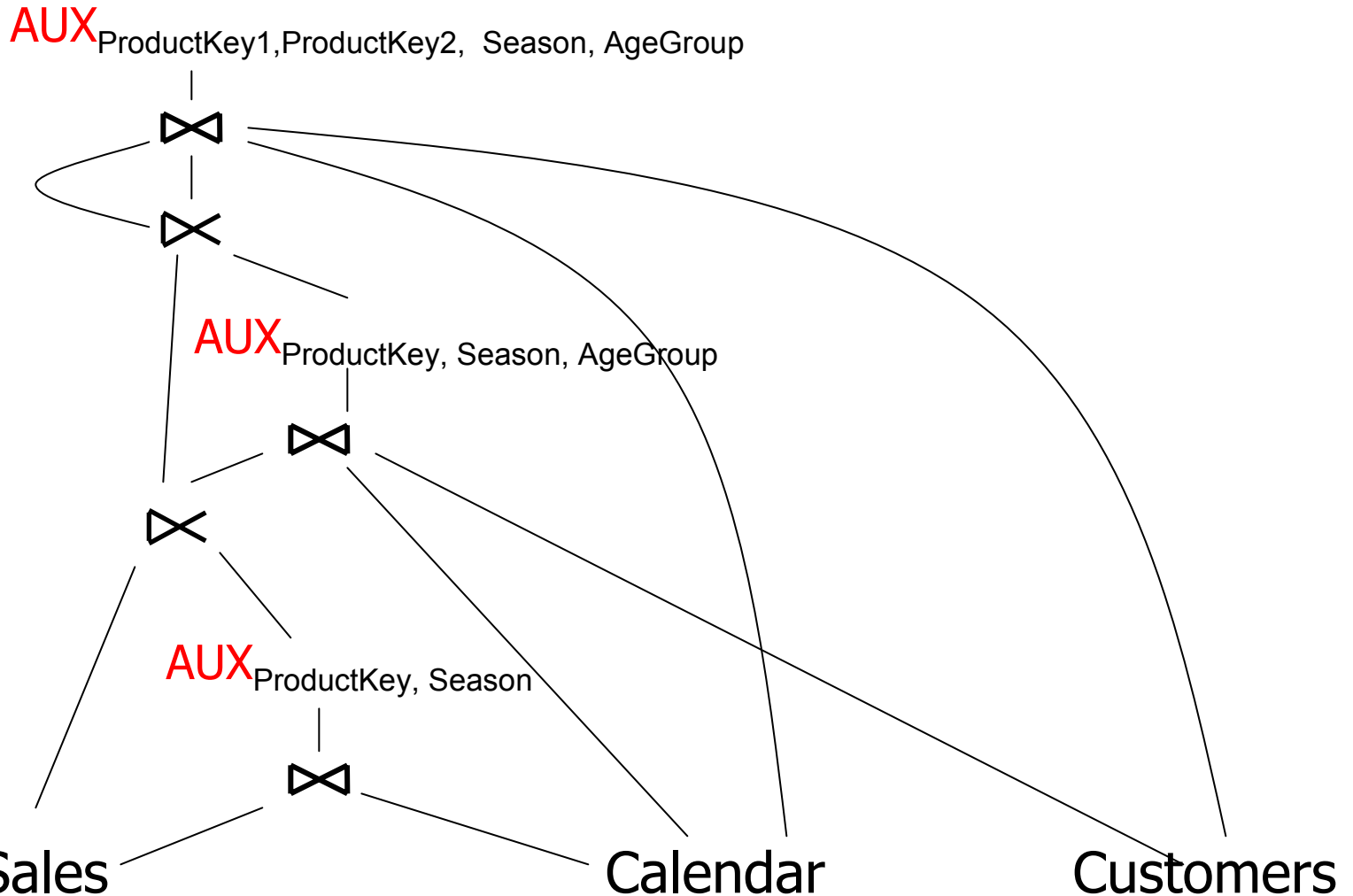


# Example Plan

```
select  S.ProductKey, Season
from    Sales S, Calendar C
where   C.CalendarKey = S.CalendarKey
group  by S.ProductKey, Season
having  count(TransactionID) >= :minsup
```



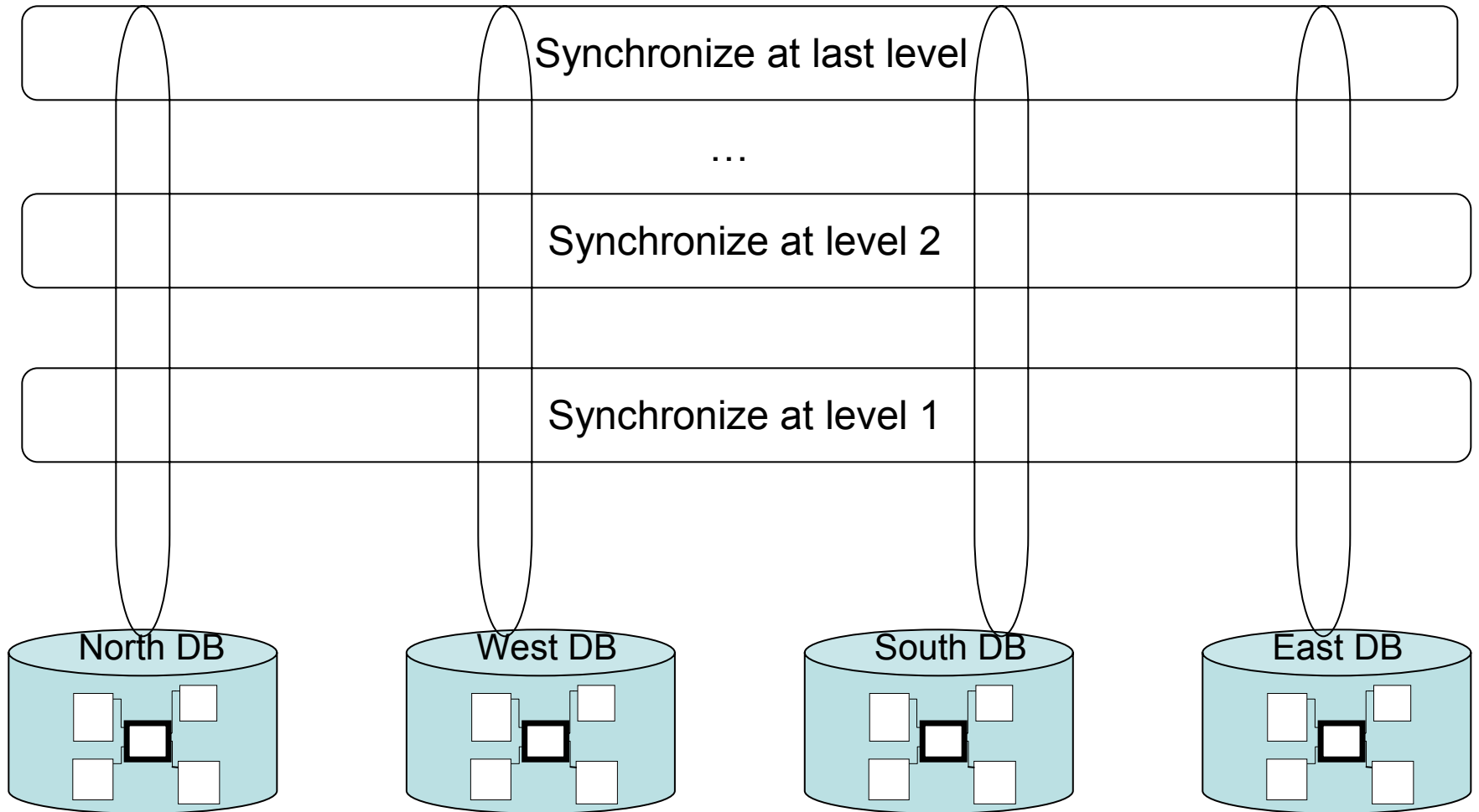
# Example Plan



# Distributed QAR

- Key observation: a globally frequent qualified itemset must be locally frequent at some site.
- Naïve Approach
  - Find frequent qualified itemsets at each site; then synchronize
  - Too many candidates
  - Too many messages
- Improvements:
  - Reduce the number of messages
  - Reduce candidates

# Levelwise Synchronization



# Future Work

- Different schemas, integration
- Different site capabilities
- Other implementations (not in SQL)
- **Scientific applications**