



# Data Mining with the SAP Netweaver BI Accelerator

Thomas Legler [t.legler@sap.com](mailto:t.legler@sap.com)

Wolfgang Lehner [lehner@inf.tu-dresden.de](mailto:lehner@inf.tu-dresden.de)

Andrew Ross [a.ross@sap.com](mailto:a.ross@sap.com)

32nd Int'l Conf. on VLDB, 2006, Seoul



## **SAP Netweaver BI Accelerator**

- **Architecture**
- **Index Structures**

## **BIA Association Rule Mining**

- **Distributed Mining**
- **Frequent Pattern Mining**

## **Summary**

## The Problem for a BI system

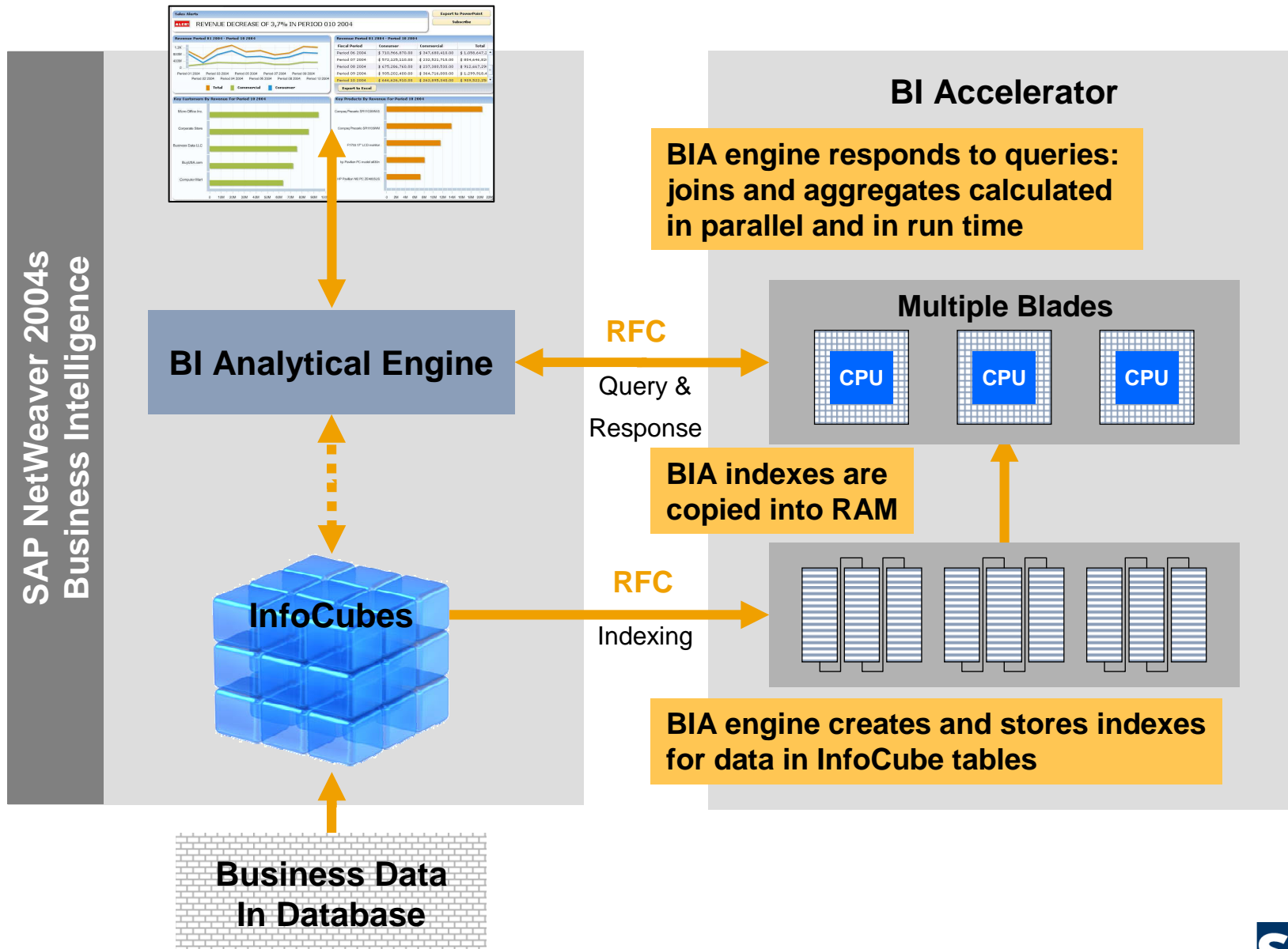
- Report queries can be very slow (some minutes, up to hours)
- Hard to predict how long a query execution may take

## Current Solution

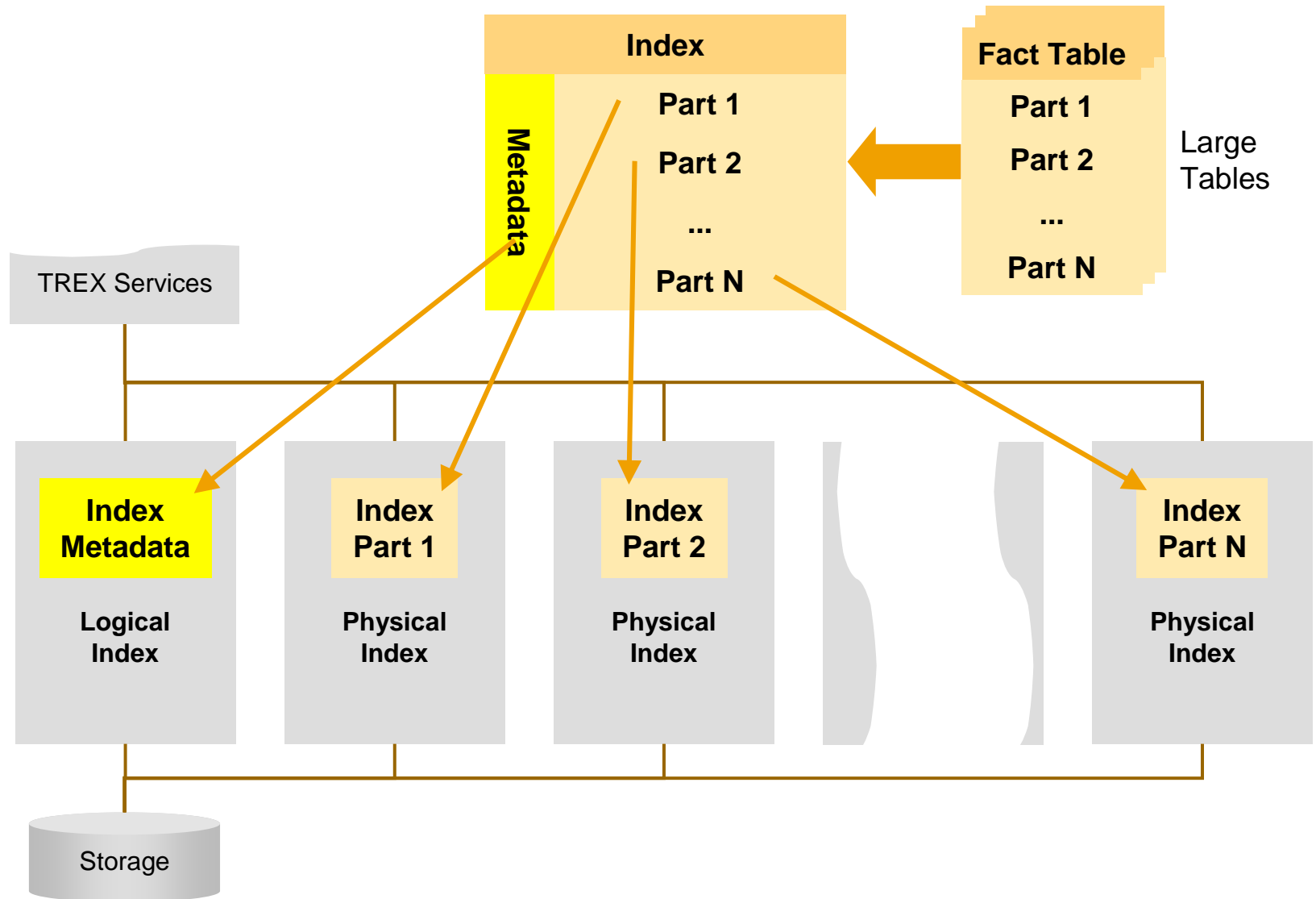
- Build materialized views (SAP: aggregates) for top X queries
  - Top X queries fast
  - Other queries slow
- You never cover every possible query
- You increase aggregate maintenance effort (for data loads, master data changes, ...)
- You increase consulting/know-how required at customer site
- You increase space consumption on database

**Aggregates: Flexibility ↔ Performance**

# BI Accelerator Architecture



# BI Accelerator Horizontal Partitioning



# Physical Index Structure

## ■ Column-based approach

- Only needed attributes in memory
- Fast attribute scan and aggregation

## ■ Compressed

- Integer-coded
- Several compression methods for AttributeTable and Dictionaries
- Compression factor ~1:20

## ■ Read optimized structures

- Additional small and write optimized index
- Queries use both indices and merge results
- Periodical merge of delta and main index

## Single valued attribute data structure

Attribute Table

DocId	ValueId
1	24
2	3
3	7
4	17

Dictionary

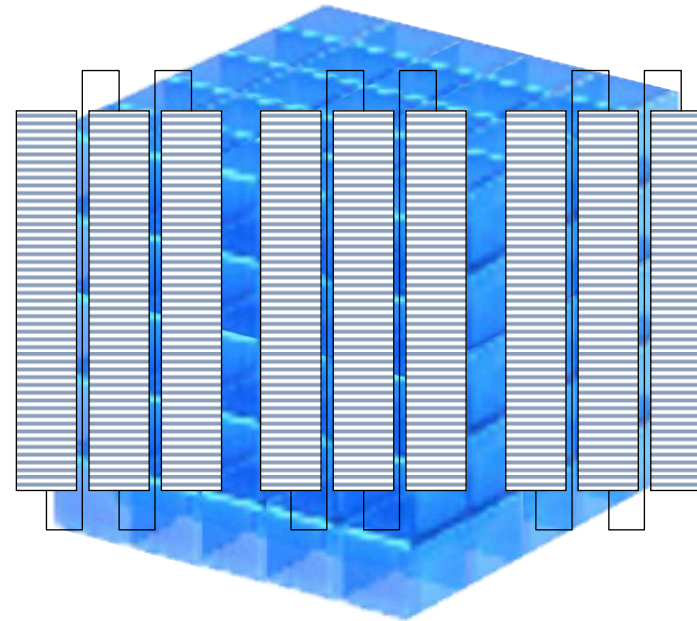
ValueId	Value
1	IBM
2	Microsoft
...	
17	SAP

Index

ValueId	DocIdList
1	...
2	2, 5
...	
17	4

## The BIA Index

- Store meta data of BI InfoCube
- Contains lists of
  - Index names
  - Join conditions
  - Join paths
  - View attributes
  - Semantic relations
  - Key figures
- Represent an interface for an InfoCube
- Executes multi-dimensional queries



## The BI accelerator

- Joins indexes in parallel wherever possible
- Aggregates index entries in parallel and on the fly



## SAP Netweaver BI Accelerator

- Architecture
- Index Structures

## BIA Association Rule Mining

- Distributed Mining
- Frequent Pattern Mining

## Summary



### Data Mining as a First-Class Citizen, but reporting has top priority

- Data distribution and partitioning optimized for reporting
- Use available structures to minimize additional resource consumption

### Algorithms must fit to the architecture

- Partitioning
- Shared-nothing architecture

#### We have:

- Big Tables with a lot of attributes

#### We search for rules like:

- If a customer is German and buys a black BMW, (s)he often requests manual transmission

#### We need two steps:

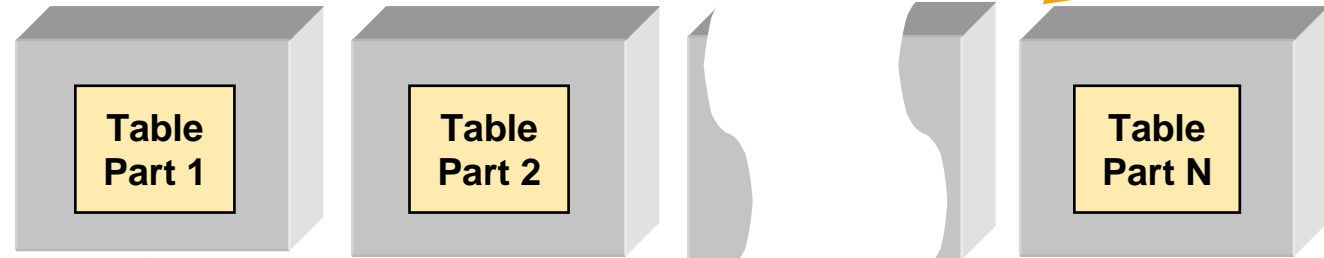
1. Filter frequent item sets to reduce the data to probably interesting rows
2. Discover (hopefully interesting) rules using these frequent item sets

# Distributed Association Rule Mining (PARTITION)

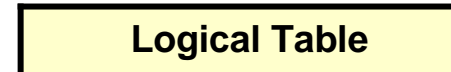
1. Request



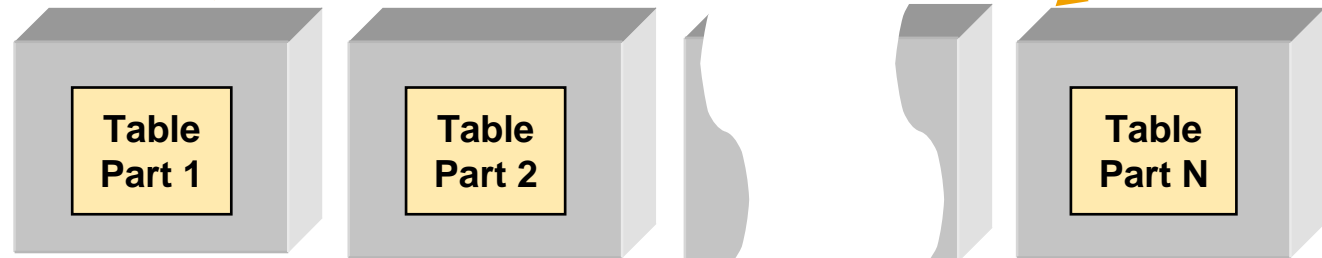
2. Local mining with relative support



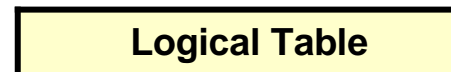
3. Merge results



4. Scan for missing item sets



5. Final merge

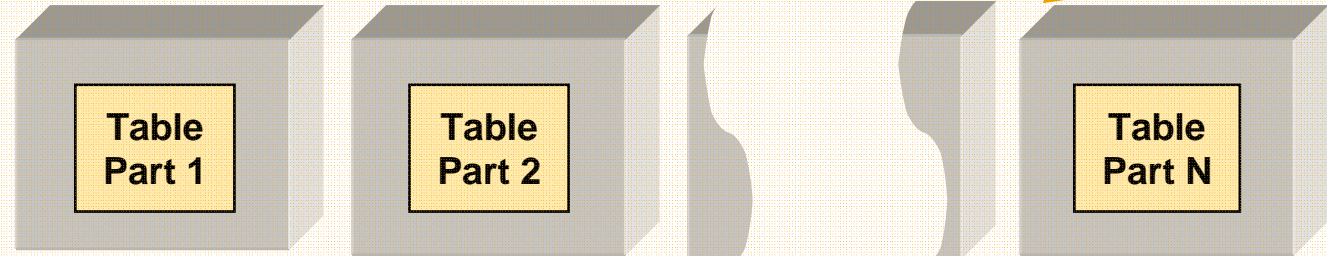


# Distributed Association Rule Mining (PARTITION)

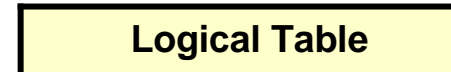
1. Request



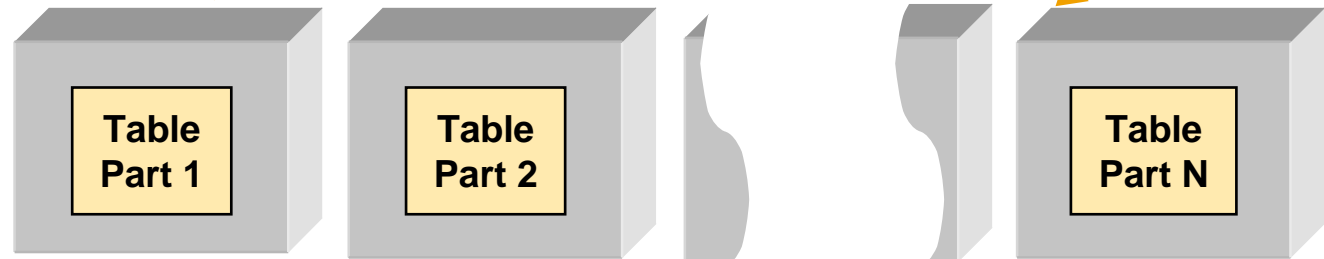
2. Local mining with relative support



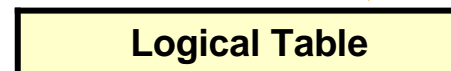
3. Merge results



4. Scan for missing item sets



5. Final merge



# Standard Algorithms for Frequent Pattern Mining

## Apriori by Agrawal et al.

- Generate  $(n+1)$ -item set candidates out of frequent  $n$ -item sets
- Count support for candidates
  - Candidate generation prevents slow database scans
  - But the table scans are fast for the BI accelerator

## FP-Growth by Han et al.

- Build frequent pattern tree and mine the tree
  - Needs a lot of additional memory to replicate the data

## Eclat by Zaki et al.

- Build Transaction-ID list for each value
- Mine depth-first
  - List comparisons needs transformation of the data to TID lists

## Modified Approach

Like BUC iceberg cubing by Beyer et al. 1999 (similar to Eclat)

- Find frequent 1-item sets
- Get their row-numbers
- Check each value of next attribute for these row-numbers
- If combinations are still frequent, check next attribute
- ...

### Our algorithm

- Is depth-first
- Does not need expensive list comparisons or transformations
- Needs a lot of short scans, but
  - We are in memory
  - We have direct access via row-numbers
  - Data is transformed to integers

So scanning the data and counting the support for all is cheaper than list comparisons and checks to find candidates

- Has very low additional memory consumption

# Example – Frequent Item Sets

Support = 2

country	car	color	trans.
GER	BMW	Red	MT
FRA	VW	Red	MT
PL	Audi	Black	AT
GER	VW	Red	AT
FRA	VW	Silver	MT
GER	BMW	Red	MT
GER	BWM	Silver	AT
FRA	VW	Black	AT
GER	BMW	Red	MT

- Country: (GER: 5, FRA: 3)
- Car: (BMW: 4, VW: 4)
- Color: (Red: 5, Black: 2, Silver: 2)
- Trans: (MT: 5, AT: 4)



- GER, BMW: 4, FRA, VW: 3
- GER, Red: 4
- GER, MT: 3 GER, AT: 2 FRA, MT: 2
- BMW, Red: 3 VW, Red: 2
- BMW, MT: 3 VW, MT: 2 VW, AT: 2
- Red, MT: 4 Black, AT: 2



...



- GER, BMW, RED, MT: 3

# Example – Frequent Item Sets

Support = 2

country	car	color	trans.
GER	BMW	Red	MT
FRA	VW	Red	MT
PL	Audi	Black	AT
GER	VW	Red	AT
FRA	VW	Silver	MT
GER	BMW	Red	MT
GER	BWM	Silver	AT
FRA	VW	Black	AT
GER	BMW	Red	MT

- Country: (GER: 5, FRA: 3)
- Car: (BMW: 4, VW: 4)
- Color: (Red: 5, Black: 2, Silver: 2)
- Trans: (MT: 5, AT: 4)



- GER, BMW: 4, FRA, VW: 3
- GER, Red: 4
- GER, MT: 3 GER, AT: 2 FRA, MT: 2
- BMW, Red: 3 VW, Red: 2
- BMW, MT: 3 VW, MT: 2 VW, AT: 2
- Red, MT: 4 Black, AT: 2



...



- GER, BMW, RED, MT: 3

# Example – Frequent Item Sets

Support = 2

country	car	color	trans.
GER	BMW	Red	MT
FRA	VW	Red	MT
PL	Audi	Black	AT
GER	VW	Red	AT
FRA	VW	Silver	MT
GER	BMW	Red	MT
GER	BWM	Silver	AT
FRA	VW	Black	AT
GER	BMW	Red	MT

- Country: (GER: 5, FRA: 3)
- Car: (BMW: 4, VW: 4)
- Color: (Red: 5, Black: 2, Silver: 2)
- Trans: (MT: 5, AT: 4)



- GER, BMW: 4, FRA, VW: 3
- GER, Red: 4
- GER, MT: 3 GER, AT: 2 FRA, MT: 2
- BMW, Red: 3 VW, Red: 2
- BMW, MT: 3 VW, MT: 2 VW, AT: 2
- Red, MT: 4 Black, AT: 2



...



- GER, BMW, RED: 3



# Example – Frequent Item Sets

Support = 2

country	car	color	trans.
GER	BMW	Red	MT
FRA	VW	Red	MT
PL	Audi	Black	AT
GER	VW	Red	AT
FRA	VW	Silver	MT
GER	BMW	Red	MT
GER	BWM	Silver	AT
FRA	VW	Black	AT
GER	BMW	Red	MT

- Country: (GER: 5, FRA: 3)
- Car: (BMW: 4, VW: 4)
- Color: (Red: 5, Black: 2, Silver: 2)
- Trans: (MT: 5, AT: 4)



- GER, BMW: 4, FRA, VW: 3
- GER, Red: 4
- GER, MT: 3 GER, AT: 2 FRA, MT: 2
- BMW, Red: 3 VW, Red: 2
- BMW, MT: 3 VW, MT: 2 VW, AT: 2
- Red, MT: 4 Black, AT: 2



...



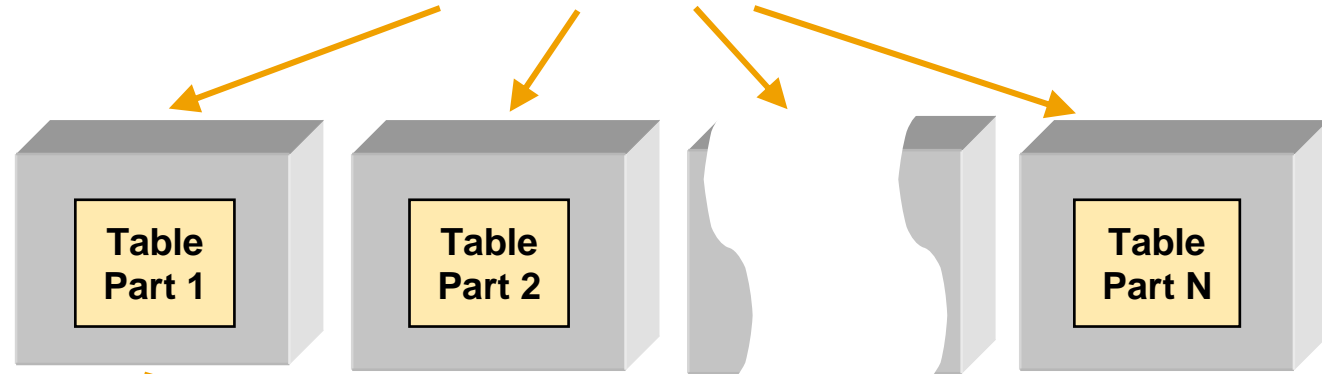
- GER, BMW, RED, MT: 3

# Distributed Association Rule Mining (PARTITION)

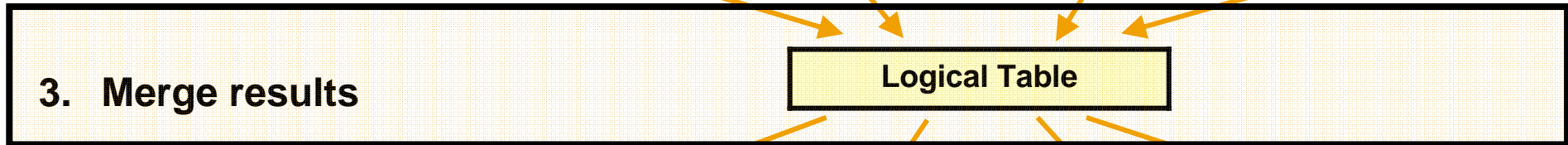
1. Request



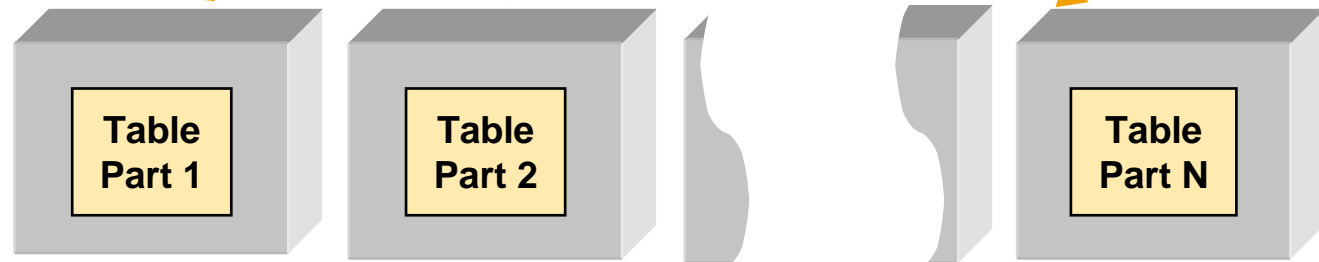
2. Local mining with relative support



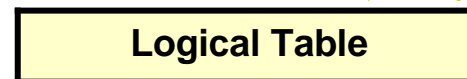
3. Merge results



4. Scan for missing item sets



5. Final merge



# Example – Distributed Association Rule Mining

country	car	color	trans.
GER	BMW	Red	MT
FRA	VW	Red	MT
PL	Audi	Black	AT
USA	Ford	Blue	MT
GER	VW	Red	AT

country	car	color	trans.
FRA	VW	Silver	MT
GER	BMW	Red	MT
GER	BWM	Silver	AT
FRA	VW	Silver	AT
GER	BMW	Red	MT

global support = 4  $\rightarrow$  local support =  $4/2 = 2$

•GER: 2, VW: 2, Red:3, MT: 3, AT:2

•GER: 3, FRA: 2, VW: 2, BMW: 3,  
Red:2, Silver: 3, MT: 3, AT:2



GER:  $2 + 3 = 5 \rightarrow$  OK!!

BMW: ?? + 3 = ??  $\rightarrow$  maybe OK

FRA: ?? + 2 = ???  $\rightarrow$  not OK (?? < 2)

...



BMW:  $1 + 3 = 4 \rightarrow$  OK!!

...

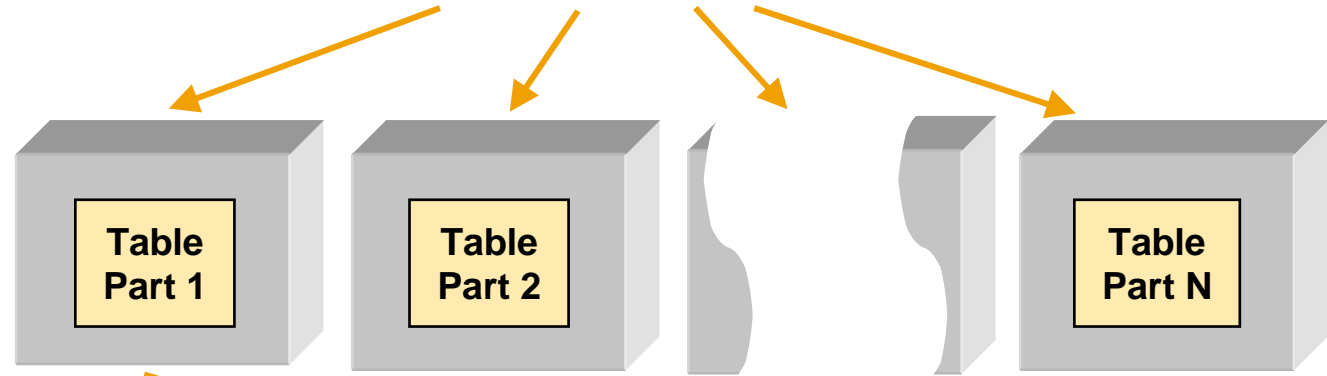


# Distributed Association Rule Mining (PARTITION)

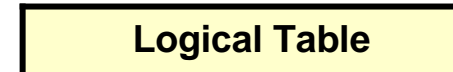
1. Request



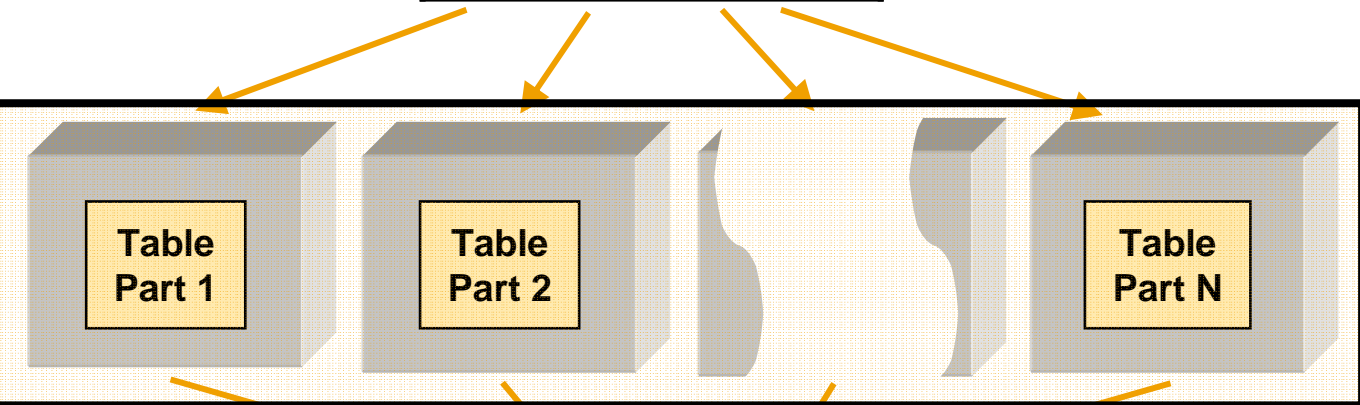
2. Local mining with relative support



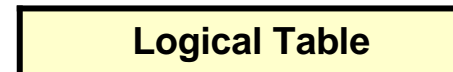
3. Merge results



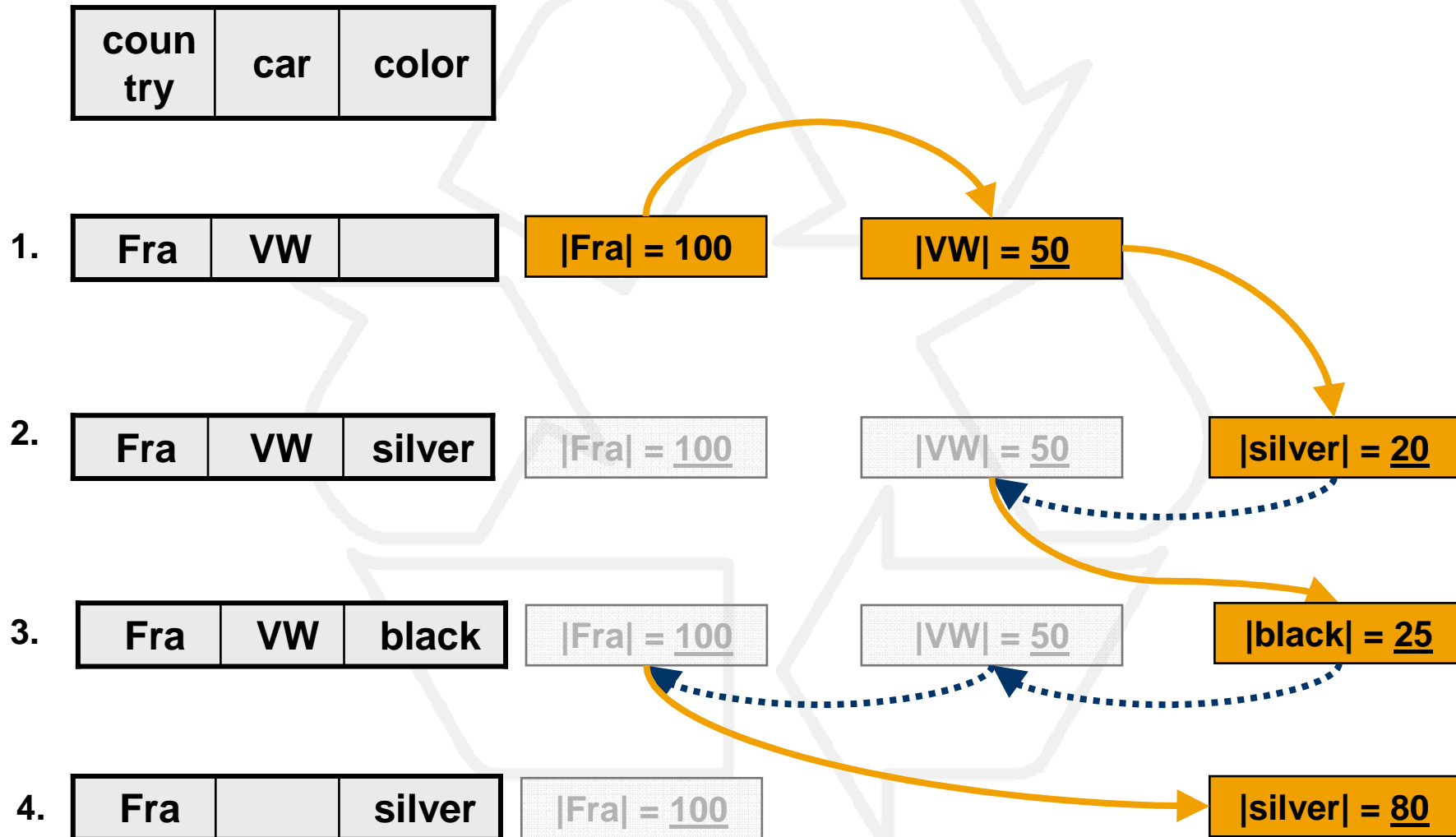
4. Scan for missing item sets



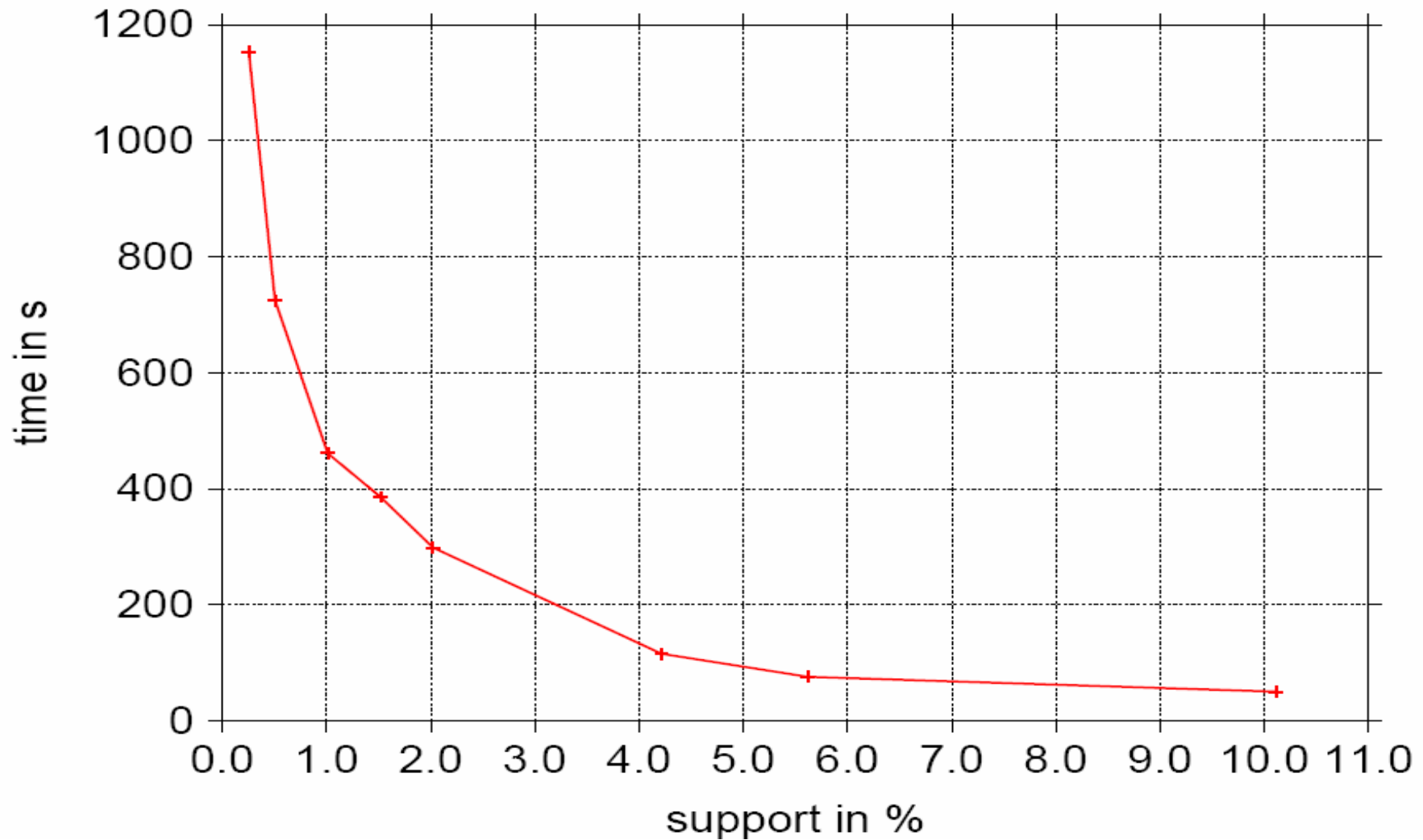
5. Final merge



# Scan for missing item sets

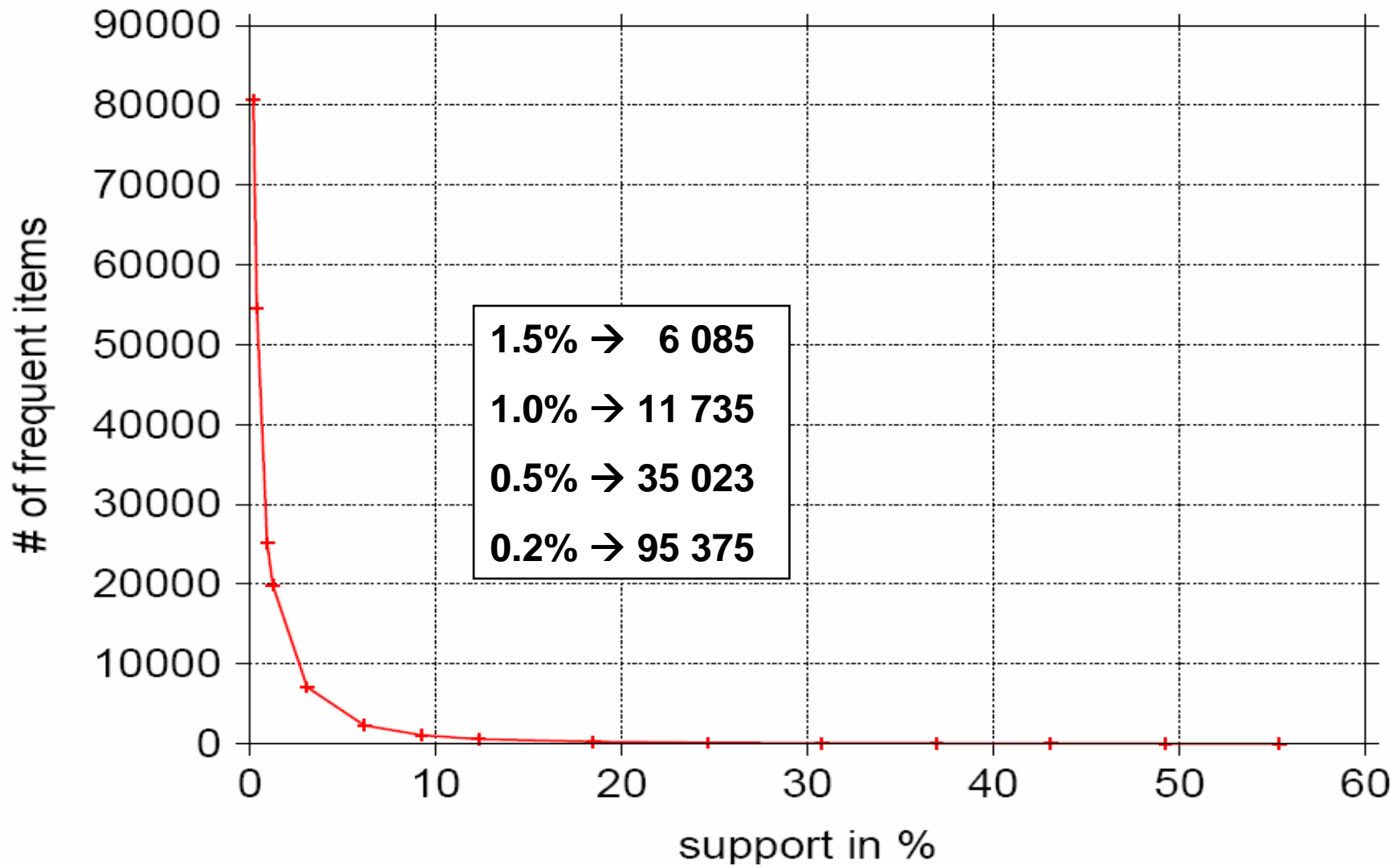


## Performance - Runtime



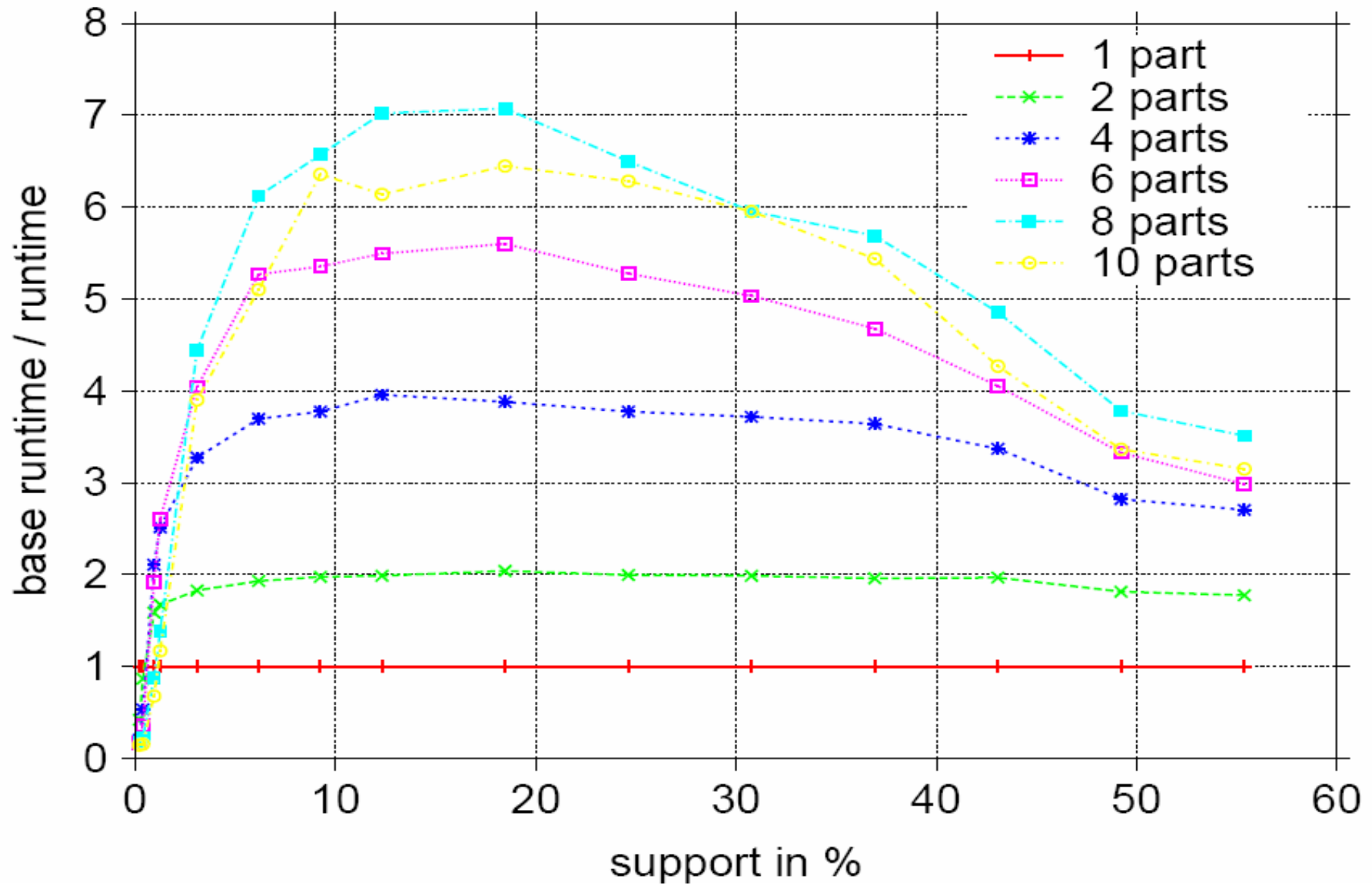
For index with 180 million rows distributed on 10 blades (32 bit)

# Performance - Complexity



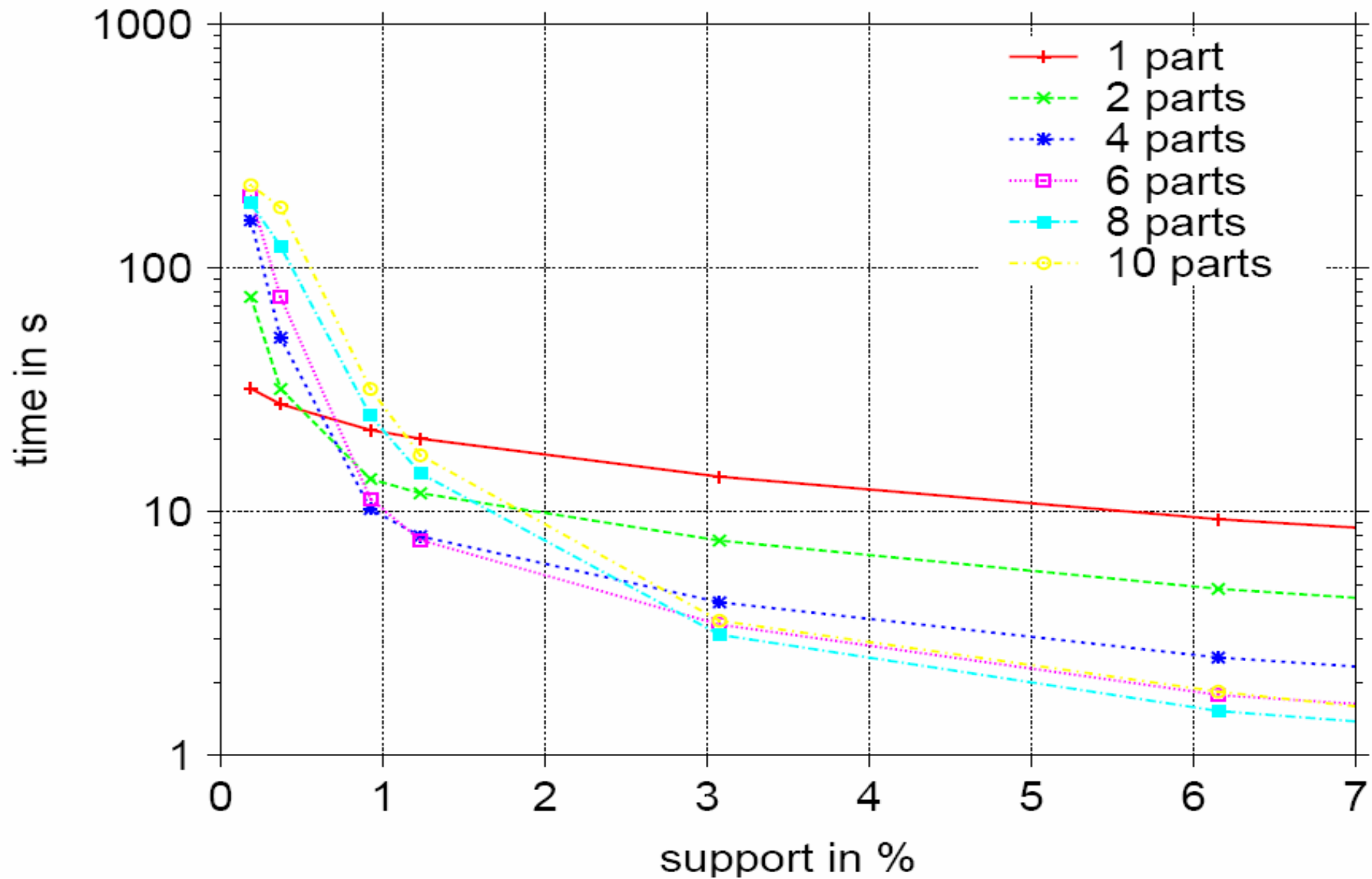
8 million row version of mushroom data set

# Performance - Scaling





# The Negative Effect of Distribution



### The SAP BI accelerator

- Distributed, compressed, and column-based in-memory data structures
- One new “virtual aggregate” replaces all previous relational aggregates
- Reduced aggregate maintenance effort
- Stable, appreciable, and fast query response time

### BIA Association Rule Mining

- Mine standard-SAP BI scenarios
- No additional replication of data for mining
- PARTITION to handle distribution
- BUC for frequent pattern mining

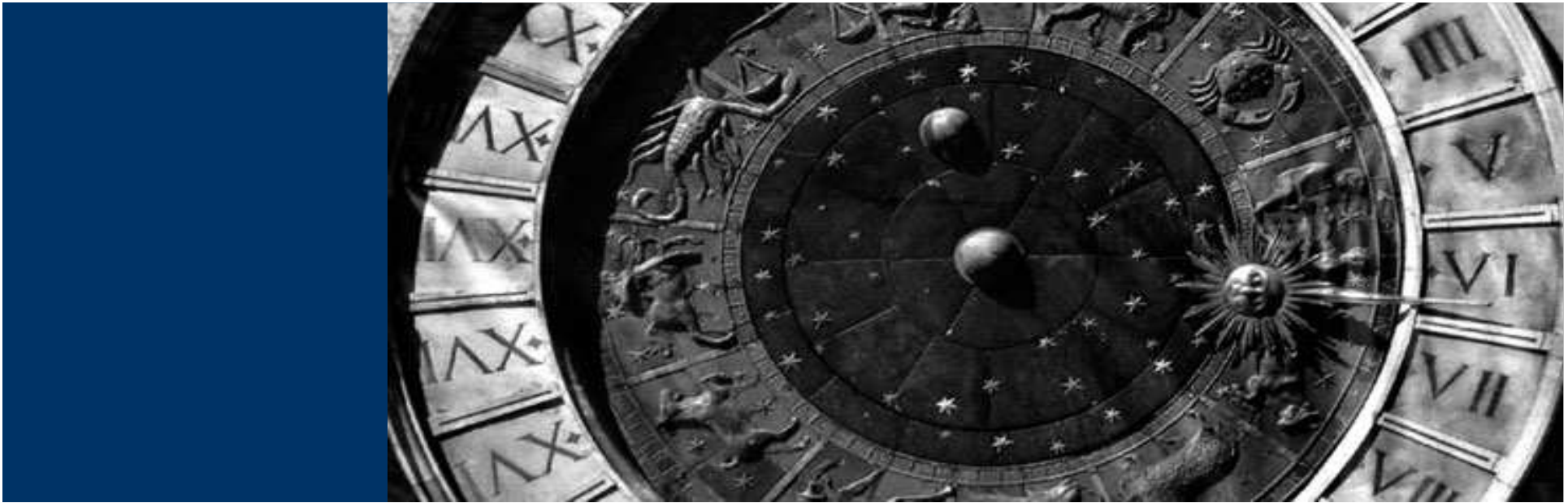
 **Thank You!**

# Q&A

**Thomas Legler ([t.legler@sap.com](mailto:t.legler@sap.com))**

**For more information about  
SAP BI accelerator please contact**

**Andrew Ross ([a.ross@sap.com](mailto:a.ross@sap.com))**

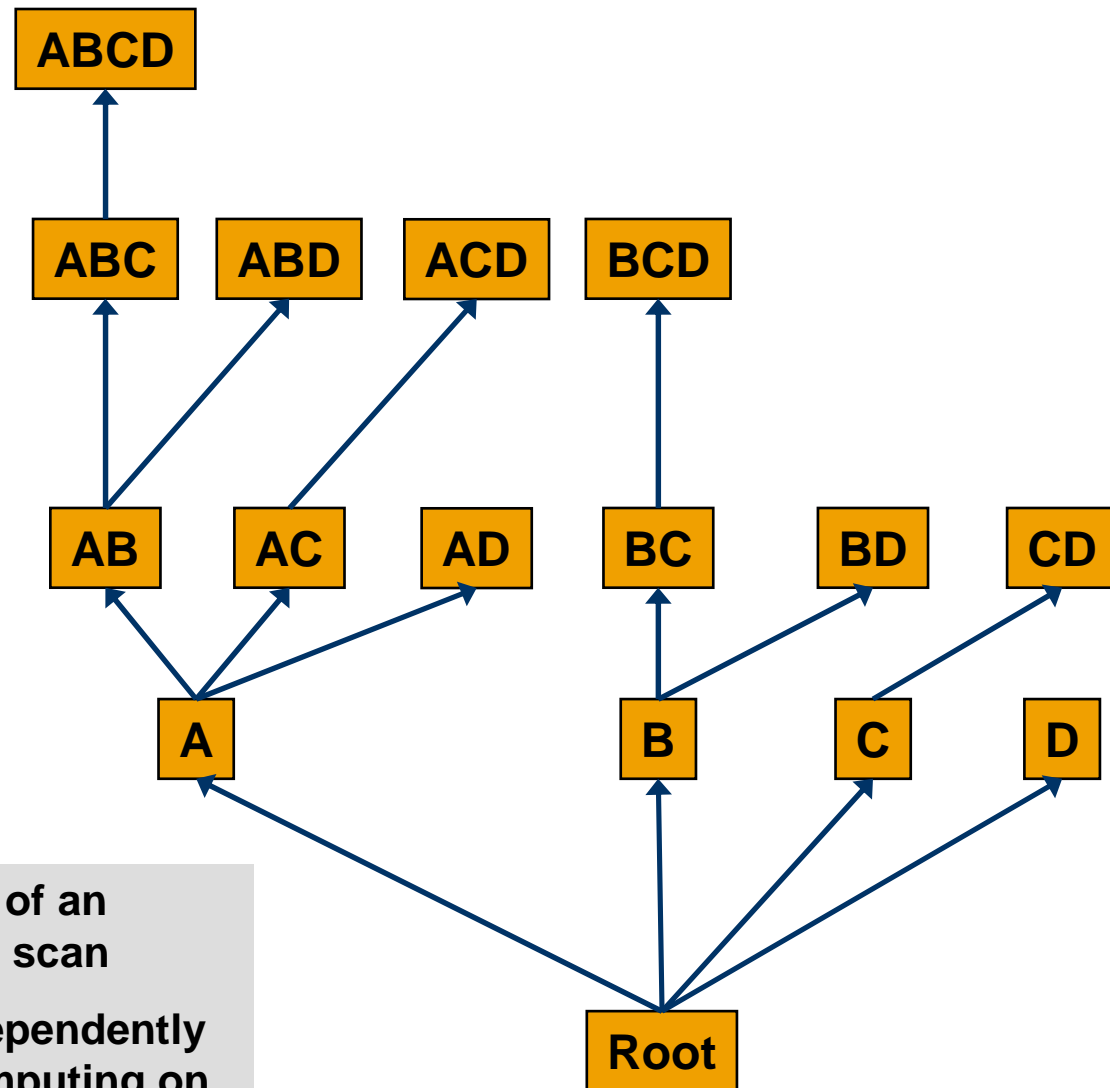


## Backups

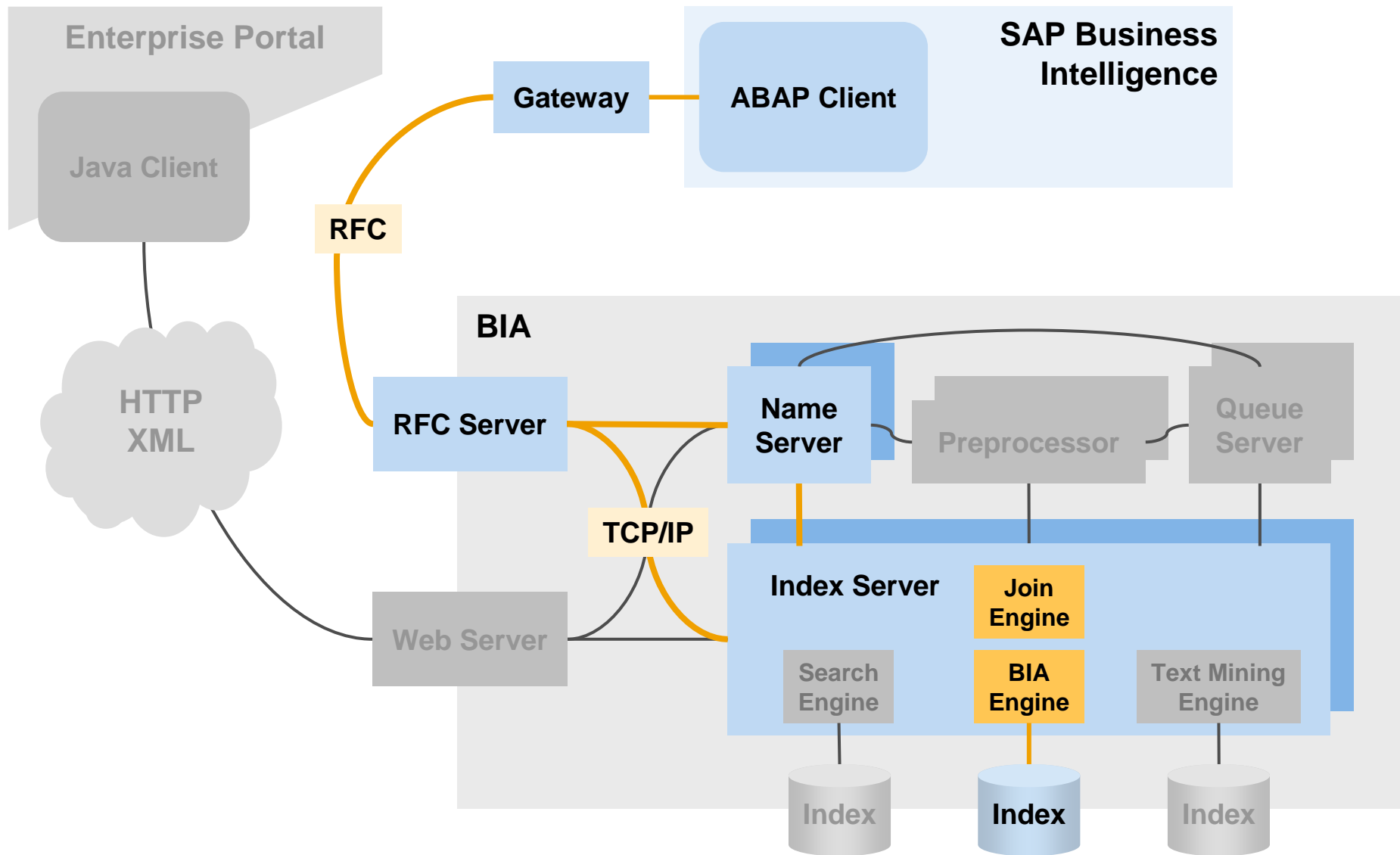
## Example - Association Rule Mining

1. Count values on A, B, C, and D
2. Check AB based on rows of A: if AB is still frequent, check ABC and ABCD
3. If all ABs are checked, test AC
4. Continue until all combinations are tested

- Count support for all values of an attribute combination in one scan
- Branches can be mined independently of each other → Parallel computing on multiple or multicore processors



# BI Accelerator Engine



# BI Accelerator Engine Details

## For each query, the engine

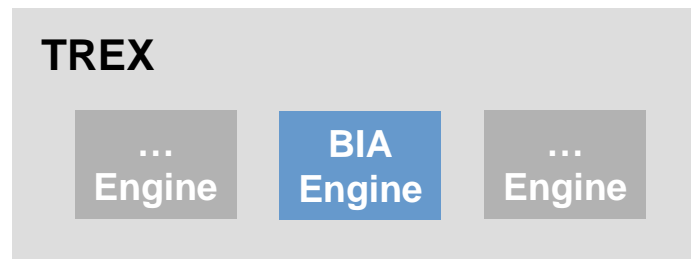
- Executes table joins as specified in the BIA index
- Processes boolean expressions
- Restricts rows for aggregation
- Aggregates data in parallel

## Very fast aggregation algorithm

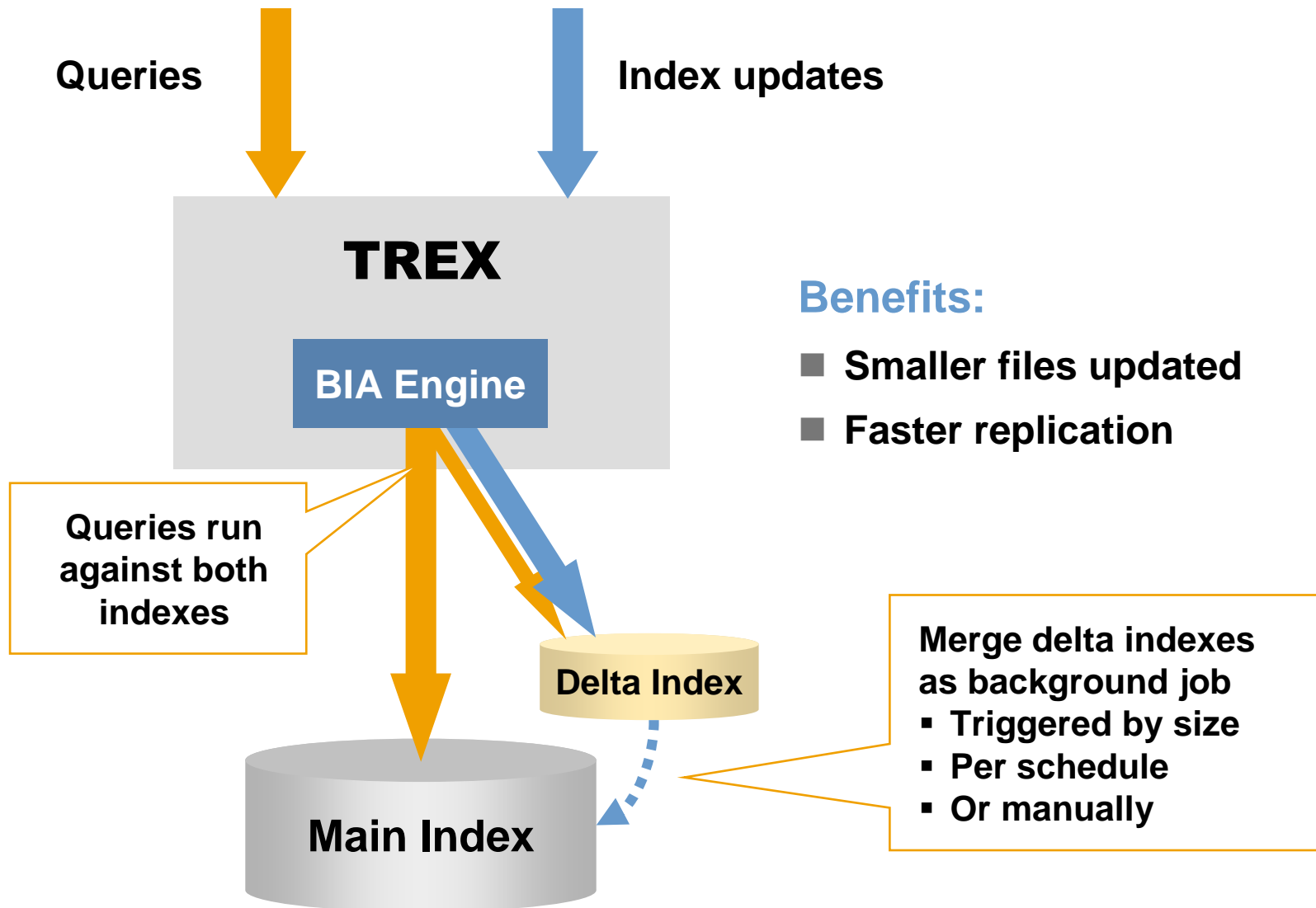
- Exploits integer coding for max speed and min I/O
- Runs on latest blade servers and grid landscapes
- Optimizes usage of memory and cache resources
- Is optimized for BI

## Dynamically loaded index

- Enables indexing of huge volumes of structured data
- Keeps only required columns in memory, others stay on disk



# BI Accelerator Fast Index Update

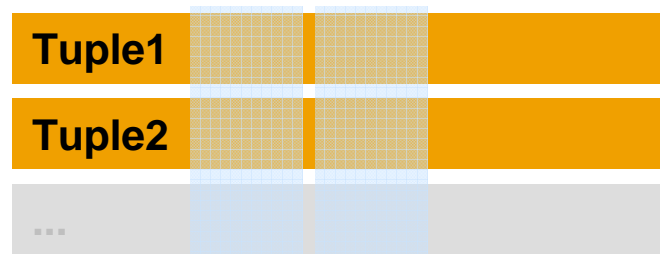




# BI Accelerator Vertical Decomposition

## Classical DB

stores tables by row



To find all instances of an attribute value:

- Go to the first row
- Check the attribute value
- Repeat for each row in the table

## BI accelerator

stores tables by column



To find all instances of an attribute value:

- Go to the attribute column
- Read its row values

Related data for aggregation  
is compact → use Caches

## BI accelerator storage for a single valued attribute

### ■ Dictionaries

Lists of all used values

Values stored in compressed form

### ■ Attribute tables

Minimal number of bits used to represent values

### ■ Indexes

Represented compactly using highly optimized compression coding

## Single valued attribute data structure

Attribute Table

DocId	ValueId
1	24
2	3
3	7
4	17
5	3

Dictionary

ValueId	Value
1	IBM
2	Microsoft
...	
17	SAP
...	

Index

ValueId	DocIdList
1	...
2	...
3	2, 5
4	...
...	
17	4

# Example: Front Coded Blocks

## Dictionary

Valued	Value
1	IBM
2	Microsoft
...	
20	SAP
21	SAP_Press
22	SAP_SI

## Blocks with x values

Valued	Value
1	IBM
2	Microsoft
...	

Valued	Value
20	SAP
21	SAP_Press
22	SAP_SI

Valued	Value
20	SAP
21	[3] + "_Press"
22	[4] + "SI"

Sorted by value

Good chance that similar values are in one block

Good Compression

Find Valued (or not)

Search in decompressed block

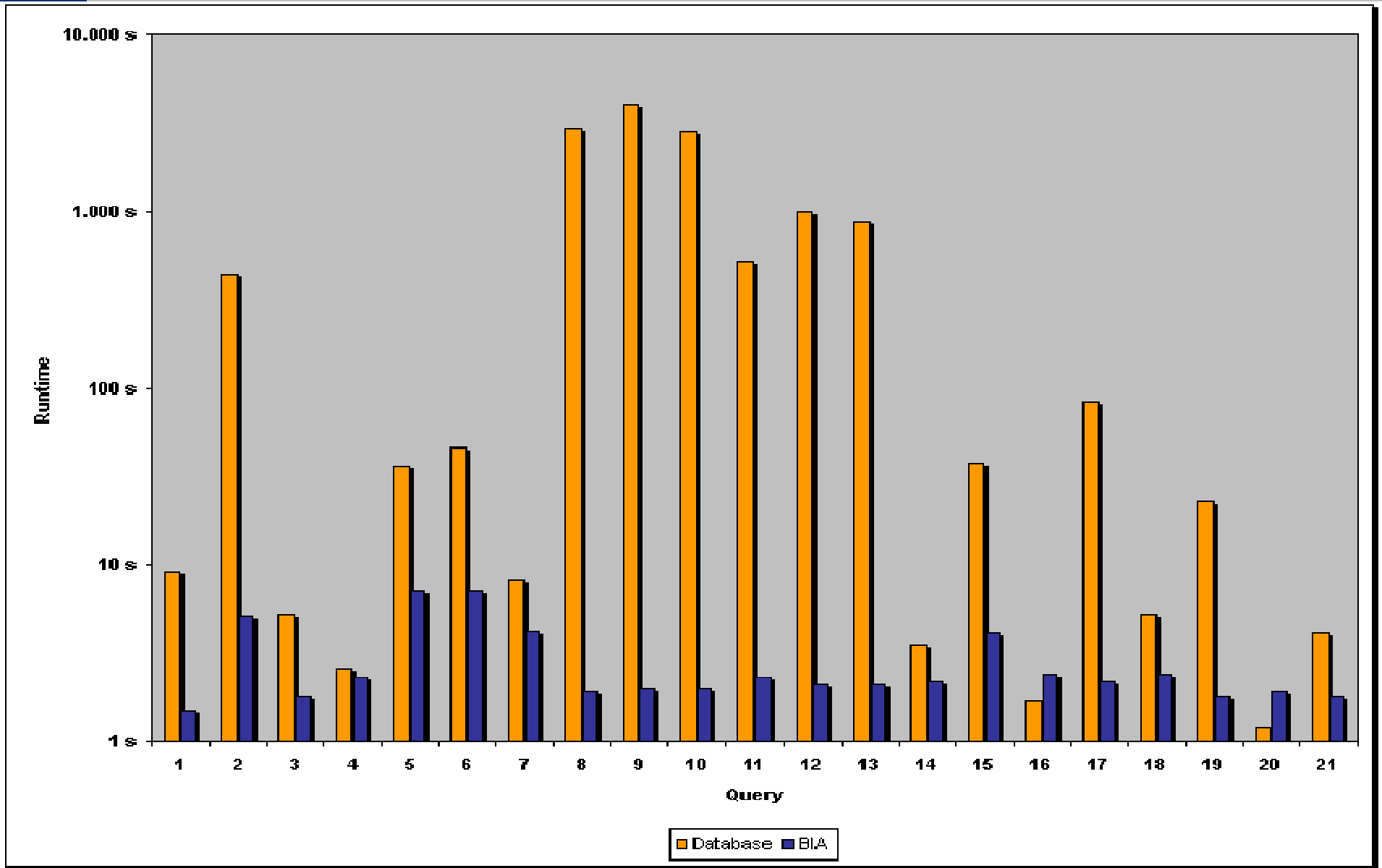
Find block

decompress

SEARCH

1. **Shortage/Overage Keep**
2. **Sales, Delivery, Billing Cubes over 2 years**
3. **Customer Net Sales**
4. **Handling Charge Analysis over 2 cubes**
5. **Return Value by Reason code**
6. **Handling Fee by National Grp, ChainId, Customer**
7. **Solution Management Optimization with a partner**

# Query Execution Times



### Queries over multiple InfoCubes

- 9 InfoCubes with approx. 850 million rows in fact tables
- 22 aggregates
- Storage on Oracle 9.2.0.4.0
- InfoCubes approx. 130 GB + aggregates 6 GB = 136 GB

### Storage on BI accelerator

- Overall: 30.4 GB
- Database used as backup for rebuilding indexes

#### BI accelerator

- 64-bit SUSE Linux SLES 9
- 6 \* Dual Xeon 3.6 GHz
- 6 \* 8 GB RAM

#### Versus

#### Database

- SunOS 5.9
- Sun V440 4 CPUs
- 16 GB RAM

## Query Execution Times (Top 7 Queries)

Query	Info Cubes	Aggr	RDBMS (sec)	BI accelerator (sec)	Improvement (factor)	Rows after filtering	Rows after aggregn.
Query 1	6		9.1	1.5	6	2 540	10
Query 2	8	*	435.3	5.2	84	13 434 508	1 322
Query 3	3	*	5.3	1.8	3	283 020	126
Query 4	4	*	2.6	2.3	1	96 712	5 771
Query 5	4		36.3	3.4	11	590 784	27 798
Query 6	4		46.1	3.2	15	590 784	27 798
Query 7	5		8.2	4.2	2	59 870	15 803
<b>Total</b>			<b>542.9</b>	<b>21.5</b>	<b>25</b>		

## Query Execution Times

Query	Info Cubes	Aggr	RDBMS (sec)	BI accelerator (sec)	Improvement (factor)	Rows after filtering	Rows after aggregn.
Query 8	1		2924.9	1.9	1538	67 318 176	281
Query 9	1		4015.3	2.0	2008	67 318 176	149
Query 10	2		516.4	2.3	224	33 801 513	32
Query 11	2		865.7	2.1	411	33 801 513	32
Query 12	4		37.5	4.1	9	88 435 773	6 280
Query 13	3	*	1.2	1.9	0	348 957	262
Query 14	3	*	5.3	2.4	2	348 957	209
<b>Total</b>			<b>8 366.0</b>	<b>16.7</b>	<b>501</b>		



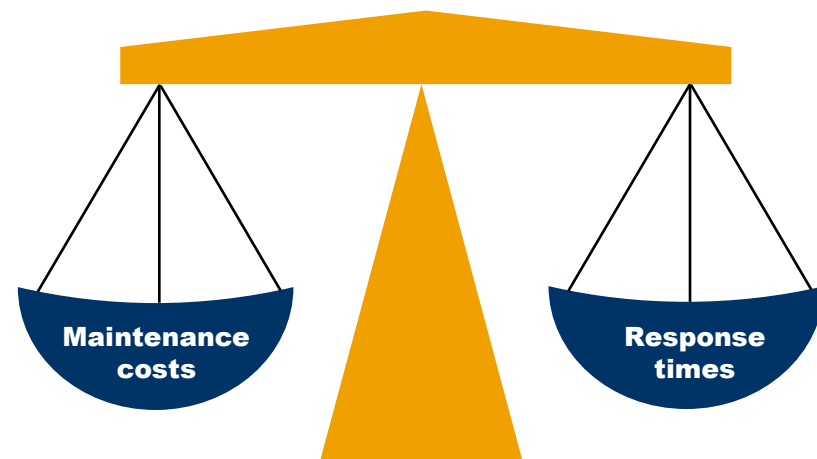
## Aggregates: Flexibility ↔ Performance

- You never cover every possible query
- You increase aggregate maintenance effort (for data loads, master data changes, ...)
- You increase consulting/know-how required at customer site
- You increase space consumption on database

## The SAP solution:

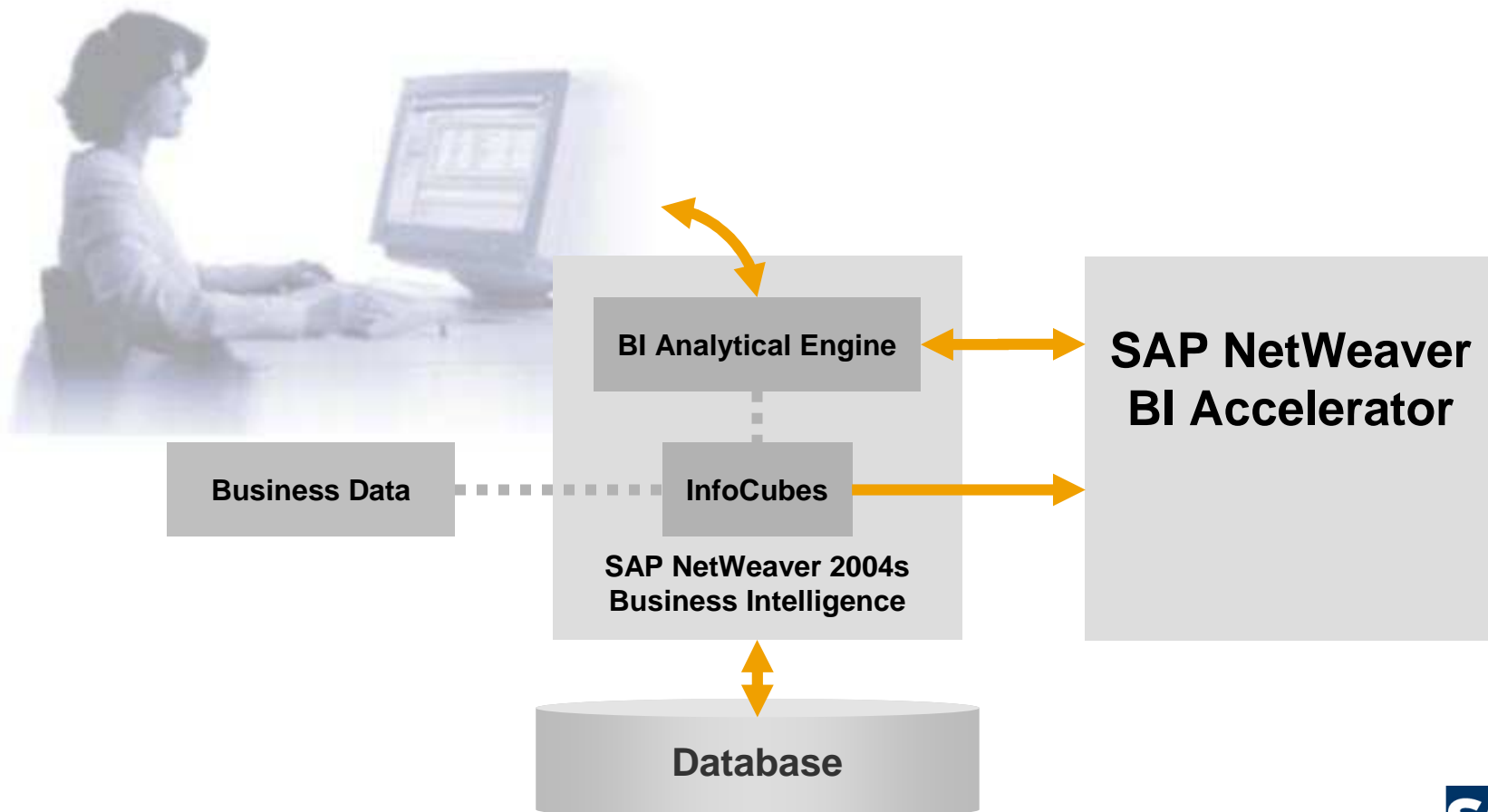
### The BI accelerator

- An aggregation engine based on search engine technology



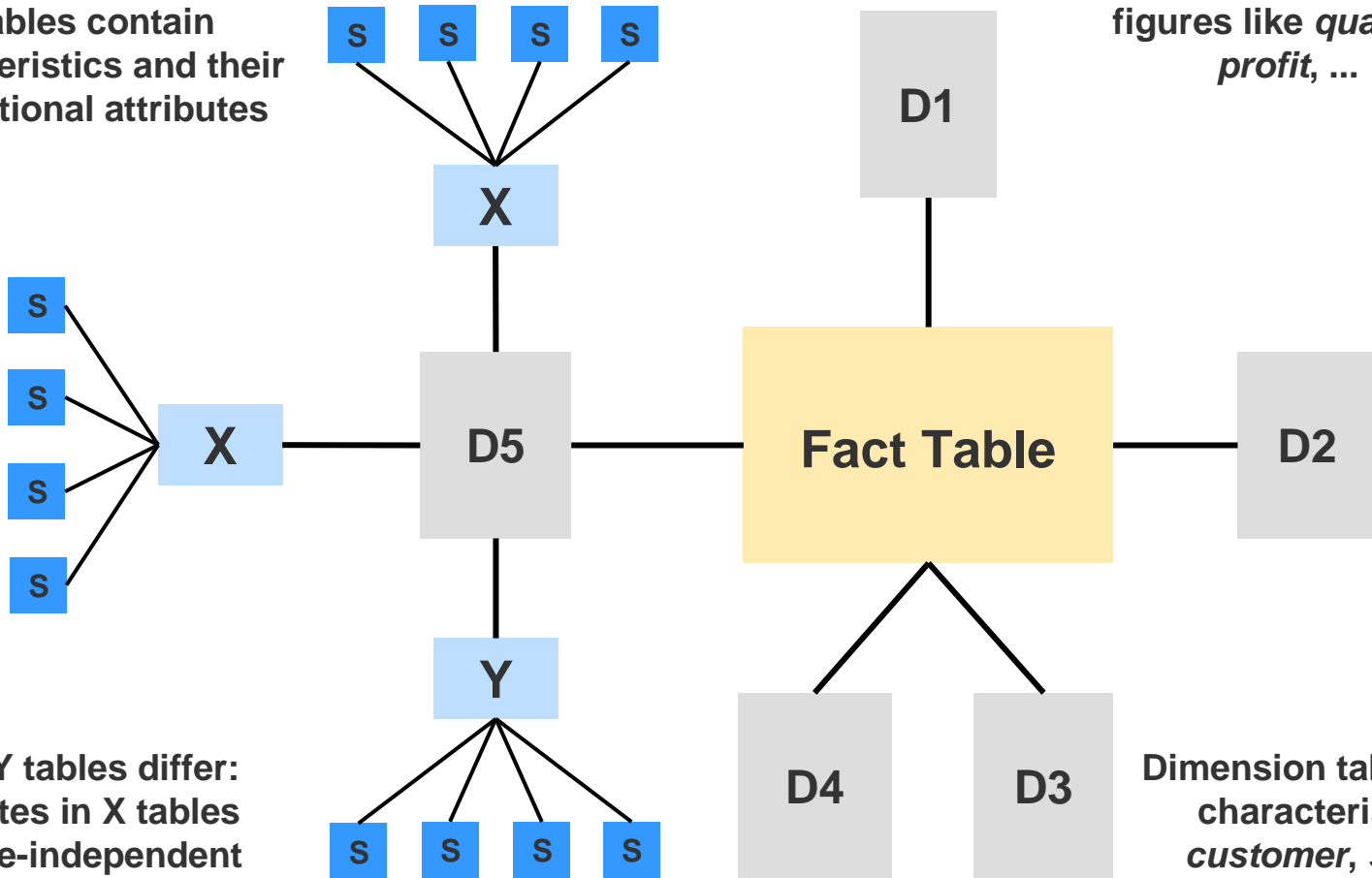
## A new accelerator functionality for SAP NW BI

- A new approach to boost BI query performance
- Performance speedup factor typically **between 10 and 100**



# Star Schema for BI InfoCube

S tables contain key values (strings)  
X tables contain characteristics and their navigational attributes



Fact table contains all characteristics and key figures like *quantity*, *profit*, ...

X and Y tables differ:  
Attributes in X tables are time-independent  
Attributes in Y tables are time-dependent

Dimension tables contain characteristics like *customer*, *sold to*, ...

## Two Nice Theorems

**Any subset of a frequent item set is frequent too**

### Example

- If you sold 4 BMWs in Germany → You sold at least 4 BMWs worldwide and you sold at least 4 cars in Germany

**→ You can build big frequent item sets out of smaller subsets**

**If an item set is globally frequent on partitioned data, the item set is frequent on at least one partition (relative to size of partition)**

### Example

- Overall 6 balls in 2 boxes → At least one box with  $\geq 6/2 = 3$  balls

**→ You never miss a frequent item set on distributed landscapes**

## The Problem: A Customer Scenario

Bob Boss, the CEO of WallMart Germany, wants to view the profit figures for all stores in South Germany for 2004 on a quarterly basis and to compare these numbers with the profits in Germany and Europe



How was the WallMart year 2004 in South Germany and how does it compare with Europe and all of Germany?

The challenge for BI is to calculate this aggregate table

Profit 2004	Q1	Q2	Q3	Q4
South Germany	???	???	???	???
Germany	???	???	???	???
Europe	???	???	???	???

### Data Mining as a First-Class Citizen, but reporting has top priority

- Data distribution and partitioning optimized for reporting
- Minimize additional resource consumption for mining

### Use already loaded data and structures

### Algorithms must fit the architecture

- Processing in memory
- Partitioning
- Shared-nothing architecture

# Copyright 2006 SAP AG. All Rights Reserved

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, System i, System i5, System p, System p5, System x, System z, System z9, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, POWER5+, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP AG.

This document is a preliminary version and not subject to your license agreement or any other agreement with SAP. This document contains only intended strategies, developments, and functionalities of the SAP® product and is not intended to be binding upon SAP to any particular course of business, product strategy, and/or development. Please note that this document is subject to change and may be changed by SAP at any time without notice.

SAP assumes no responsibility for errors or omissions in this document. SAP does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence.

The statutory liability for personal injury and defective products is not affected. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages.