SMARTARRAYS®

In-Memory Analytics

Welcome!

This seminar will start promptly at 11:00 Eastern Time.

To connect to the audio, please dial

1-650-429-3300 and enter the meeting number 947 965 759

(If you do not know your attendee ID, just press #.)

Click the phone icon on the WebEx control panel for dial-in instructions

In-Memory Analytics

Agenda

- Structuring Data for Effective Analysis
- Building Blocks for Analytic Software
- SmartArrays Application Architecture
- Case Studies; brief live demo
- Shrinking Data and Slicing with Bitmaps
- Continuous Measure Data (“Buckets”)
- Summary

SmartArrays, Inc.

Incorporated in 1999 by:

James Wheeler: Array technology head at Manugistics

Jim Brown: IBM veteran and inventor of APL2

Pat Buteux: Founding executive at Manugistics

Customer Quote

"SmartArrays is an impressive technology. Anybody with web applications that do number crunching will find it useful. SmartArrays can be used as a **powerful back-end analytics server.**

Clients should use different tools, like DB 2 for the transactional data store, and a snazzy web page formatting tool for the glitzy front end, but **put the heart of analytics for the application in SmartArrays."**

Mike Van Der Meulen
IBM, WebAhead Technologies

Customer Quote

"The TDD Tool Application that SmartArrays Developed for DOD, based on their SmartArrays technology, **far exceeded our expectations for the project.**

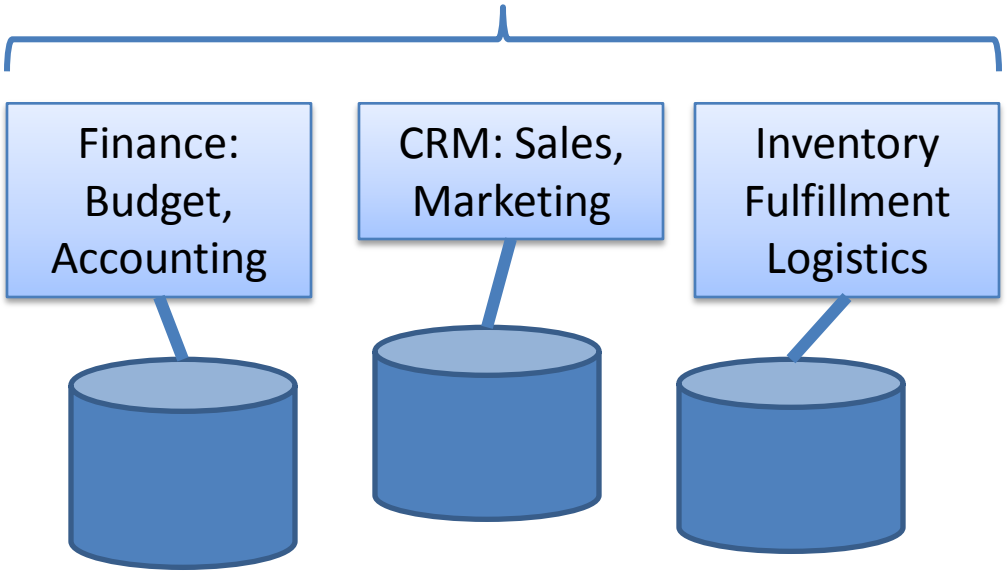
With the new information that we learned from our data we have recommended changes in procedures."

Office of Supply Chain Integration,
Office of the Secretary of Defense

Consulting Services	Software Development Kit
Database Model / Data Warehouse Design	Redistributable Array Engine
Application Architecture	Toolkit for data management and analytics
Collaborative Development or Full Implementation	More than 250 optimized array operations
Developer Training	

Meeting Analytical Needs

Enterprise BI = Generic Reports



Enterprise (Operational) Systems

Analytical Users

- Deep analytical questions
- Repeated, ad hoc querying

Analysts' Frustrations

- Too much data in too many places
- Too slow to get answers
- Partial or not-quite-right answers

Potential of In-Memory Technology

Most of the power of today's computers is wasted – waiting for data

- CPU can perform 3 billion cycles per second
- Access level 1 cache in 1-2 cycles
- Access level 2 cache in 6-20 cycles
- RAM memory in 100-400 cycles
- Flash memory in 5000 cycles
- Disk in 1,000,000 cycles

The Industry is Awakening

The Power of Speed

Keynote address by Hasso Plattner, Co-Founder, SAP AG at the Sapphire conference, May 2009

- The increase of computing power is transforming the software industry. It allows significant parallelization of application processes with new data management paradigms, as well as **in-memory, real-time analytics** over massive amounts of data. **Increased operational speed**, clearer **real-time insight** into businesses and markets, **new user-experiences**, and **reduced cost** of IT operations make this technology deliver the true power of speed

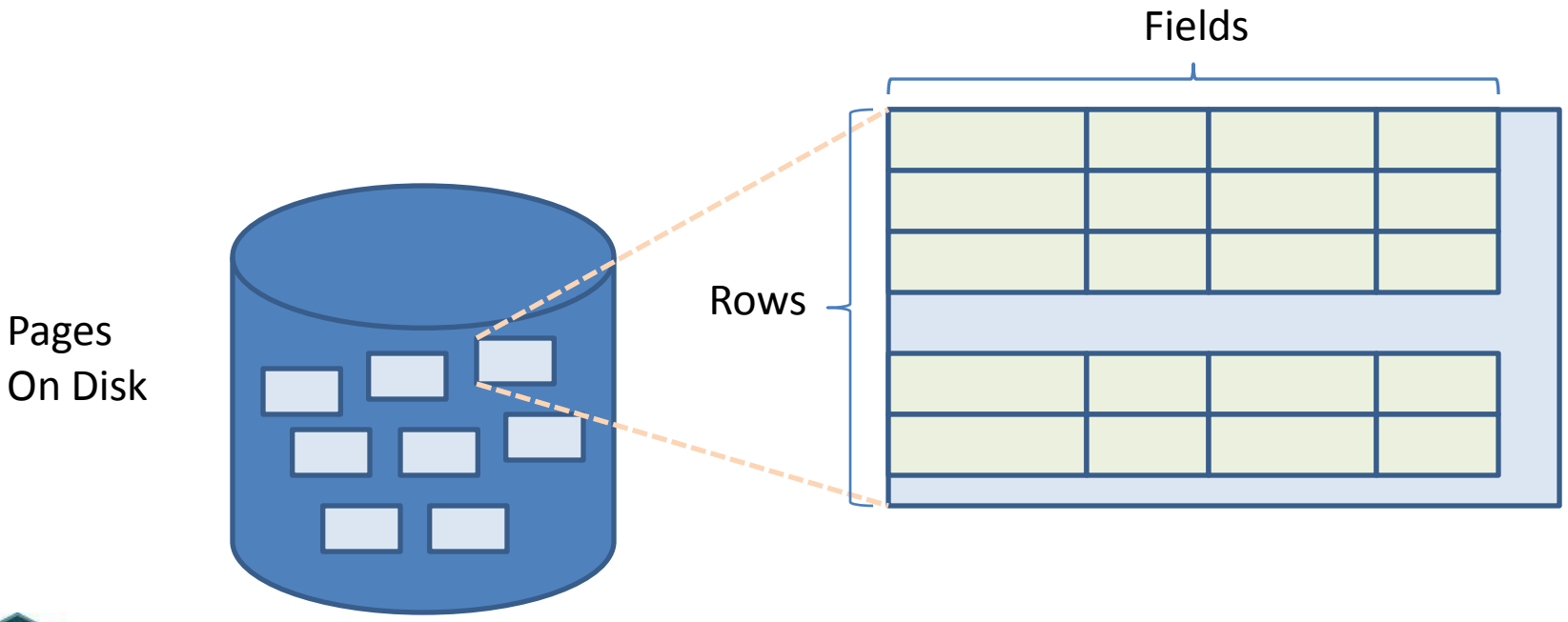
From Data to a Web-based Analytical Tool

Concepts

- Analytical Limitations of Relational Databases
- Structuring Data for Efficient Analysis
- A Toolkit for Analytical Software
- Memory Resident Data
- Data Flow in the a Typical Web-Based Tool
- Software Architecture

Value of Relational Databases

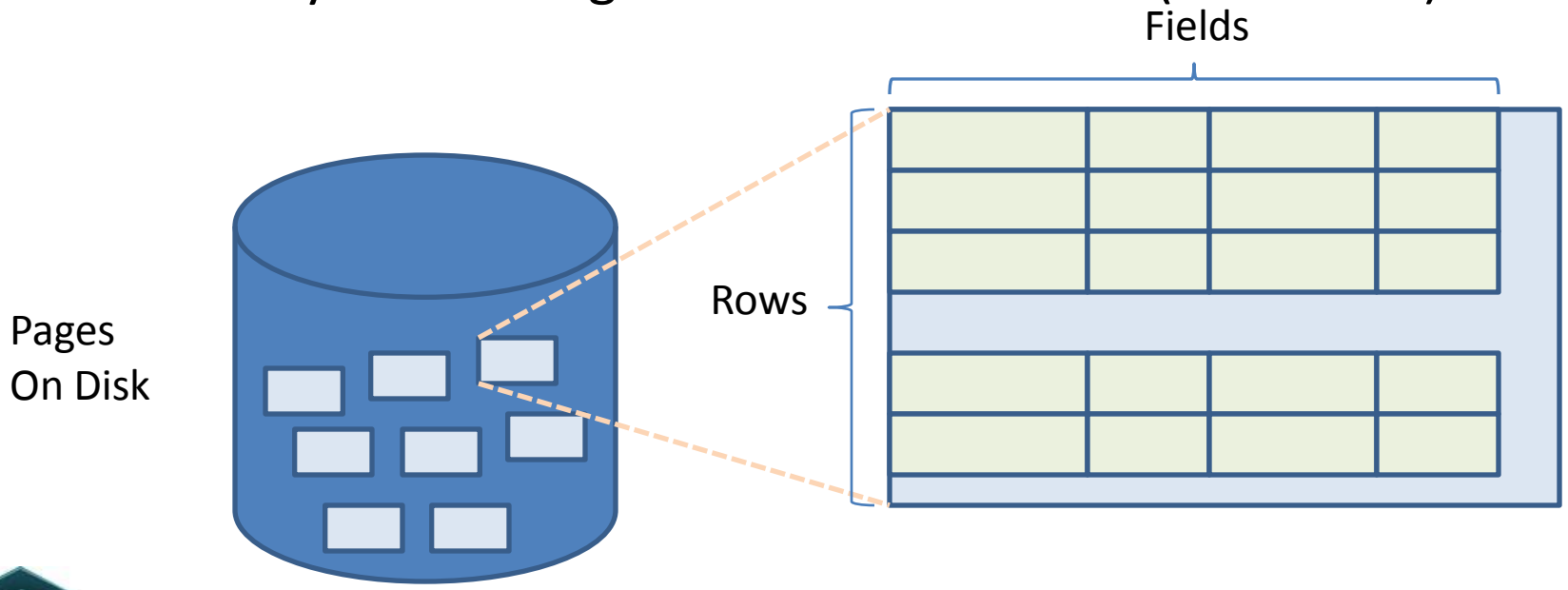
- Store data as rows (records) and fields.
- Multiple rows packed into a page.
- Reliable for holding records of data.
- Efficient for modifying/adding/deleting records.



Limitations of Relational Databases

- Adding/changing a column is a violent act.
- Rows stored in no particular order.
- Pages scattered on disk.
- Use more space than data really requires.

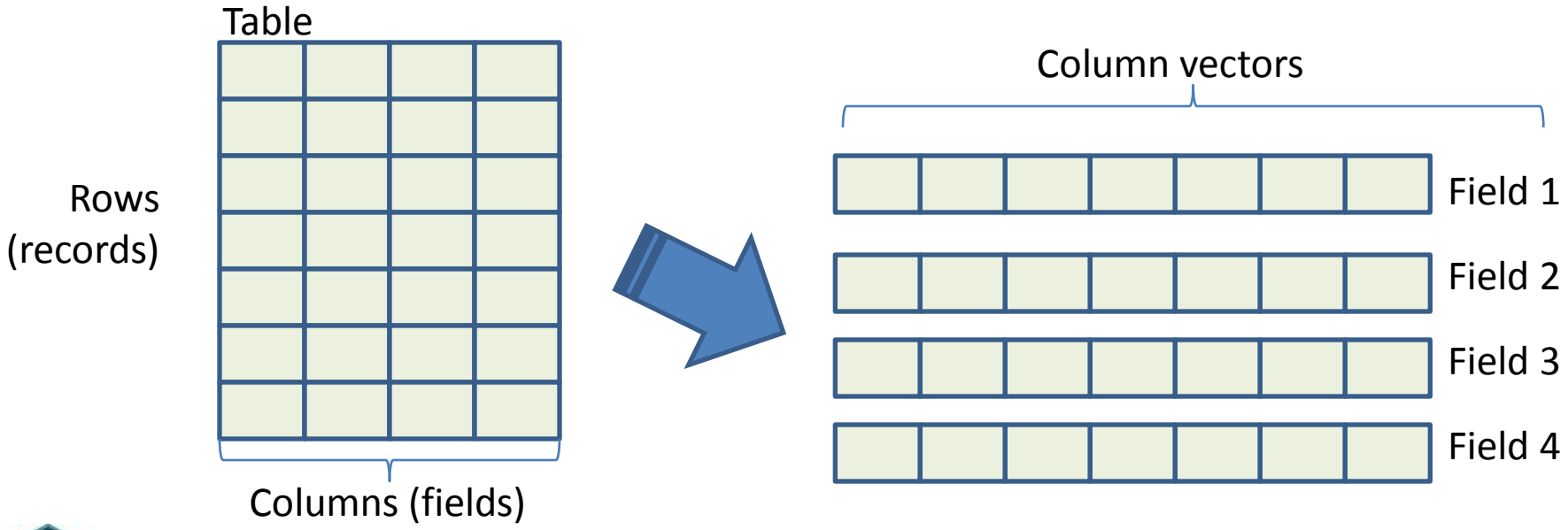
Result: Analysis - looking at each row – is slow (table scan).



SmartArrays as a Data Store: Structuring Data for Analysis

- Organize data as columns (field values stored adjacently).
- Store data in dense form.
- Hold data in memory.

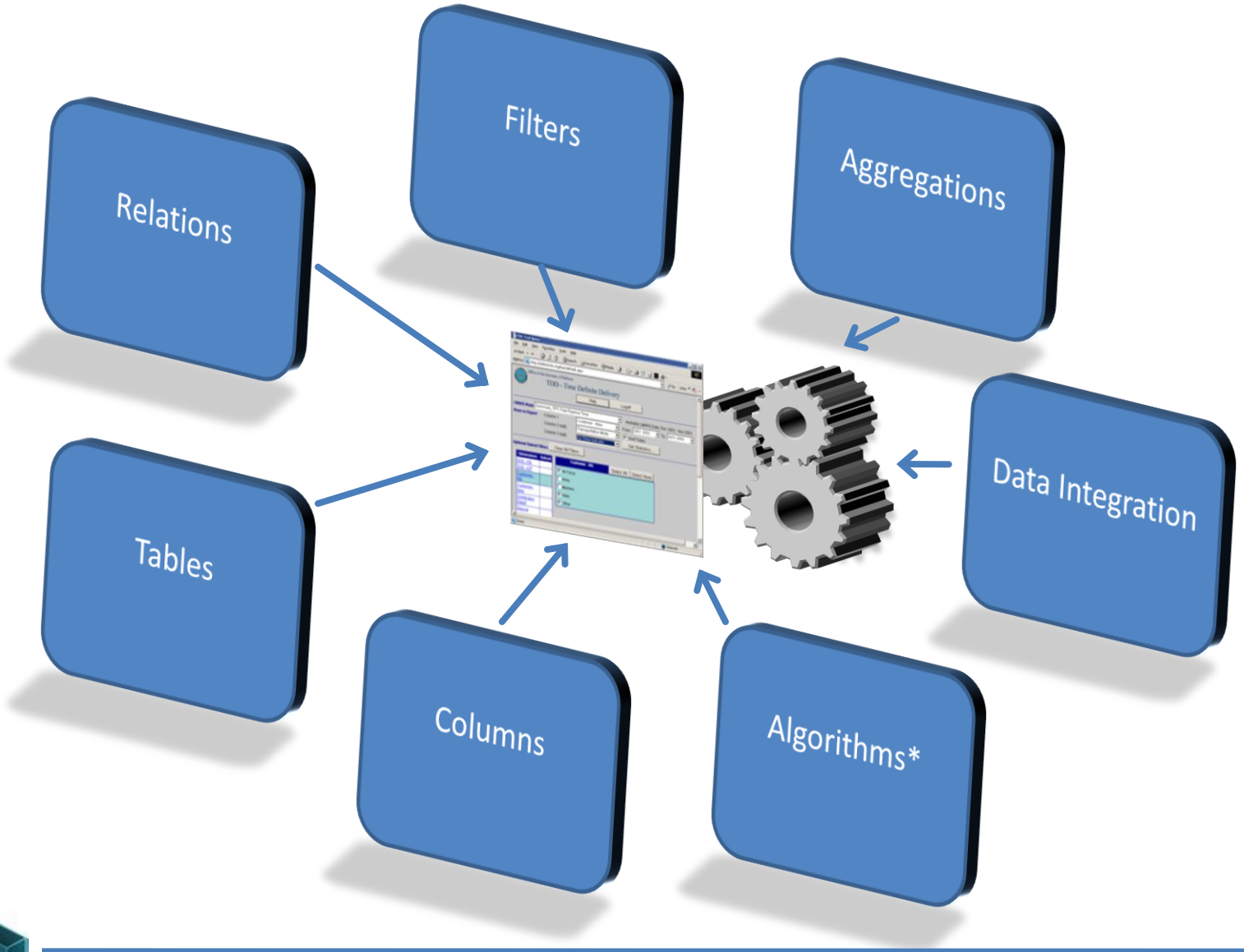
Result: Much faster analysis of data values.



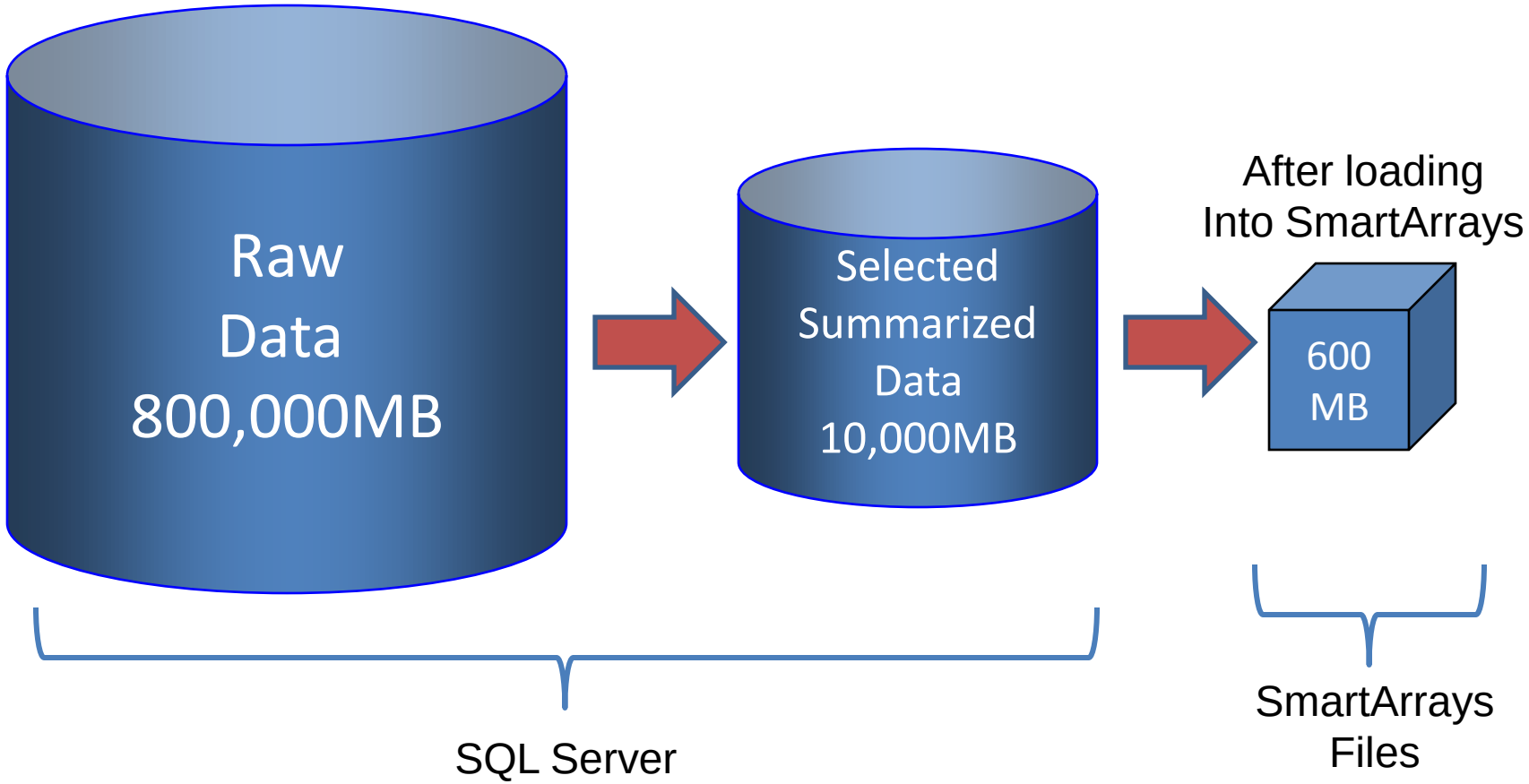
SmartArrays – A Toolkit for Analytic Software

- Used in Java or .NET applications.
- Software building blocks for tables and columns and analytic components.
- Keep data in memory as much as possible
- Advanced techniques: memory mapping from disk.
- Reduce size by shrinking data to its essence.
- Use databases (etc) as source of data.
- Library of powerful bulk-data algorithms.
- Fits into IT environment.

Building Blocks of Analytical Software

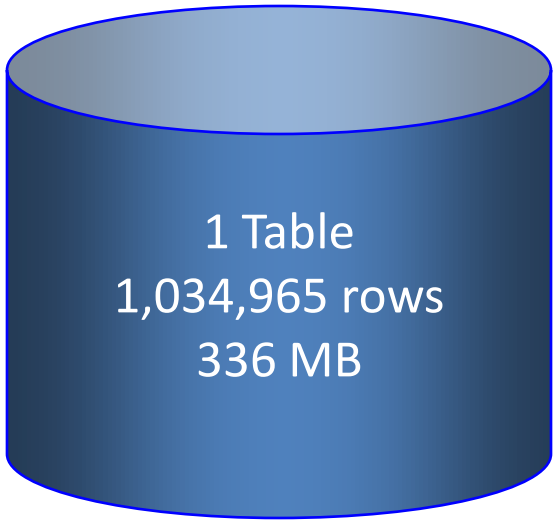


Real-Life Example:
Distilling Dept of Defense Data Into Analytical Form



Example:
Compact Data Representation

Original Table
In Database



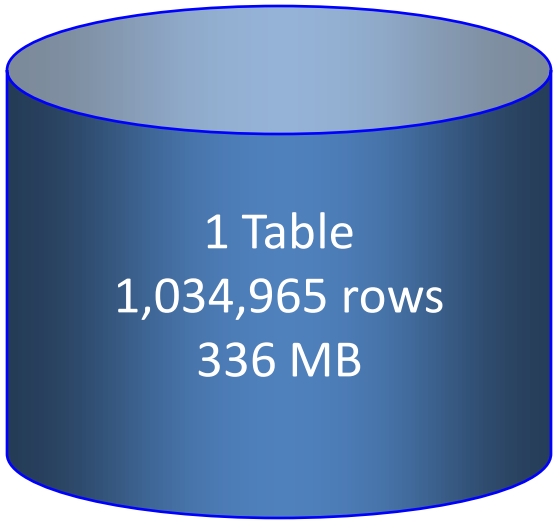

```
SummaryLast12_CIT
  [YRMO] [int],
  [YR] [smallint],
  [MO] [tinyint,
  [SOS - ICP] [nvarchar] (50),
  [SOS - HQ] [nvarchar] (50,
  [Project Code] [nvarchar] (3),
  [Customer - HQ] [nvarchar] (50,
  [Fill Type] [nvarchar] (50),
  [RDD Type] [nvarchar] (6),
  [Class of Supply] [nvarchar] (15),
  [CONUS OCONUS] [nvarchar] (1),
  [Customer - Area] [nvarchar] (20),
  [Issue Priority Group] [float],
  [Commodity Group] [nvarchar] (255),
  [Distribution Depot] [nvarchar] (50),
  [Transportation Mode] [nvarchar] (255),

  [LRT] [int],
  [Reqs] [int]
```

16 dimension columns

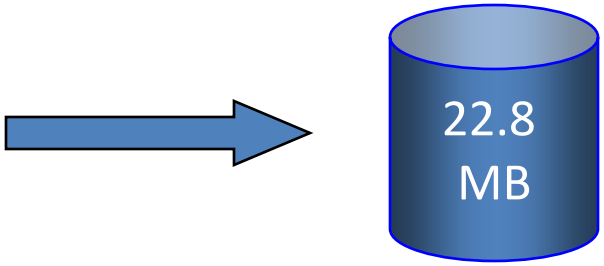
2 “fact” columns

Example:
Compact Data Representation

Original Table
In Database



Imported into SmartArrays
Table Object and Saved to Disk



- 15x denser information.
- Much faster to access.
- Easily fits in memory

Speed Compared to SQL Server

Database table has 6.5 million rows, each describing a sale in a 1-year period.

Query to select the total sales by month:

```
SELECT month,sum(amount)
FROM sales_2008
GROUP BY month
```

2 minutes with SQL Server .

0.92 seconds with SmartArrays = over 100 times faster

Why are SmartArrays Tables So Much Smaller and Faster?

- Keeps data in memory as much as possible
- Column-oriented structure.
- Column values stored as dense arrays, adjacent memory addresses.
- Low-cardinality columns can be represented compactly.
- Varchar() columns turn into SmartArrays strings, 4 bytes each.
- No indexes needed - table-scans are fast

Myth:
My Data Can't Fit in Memory

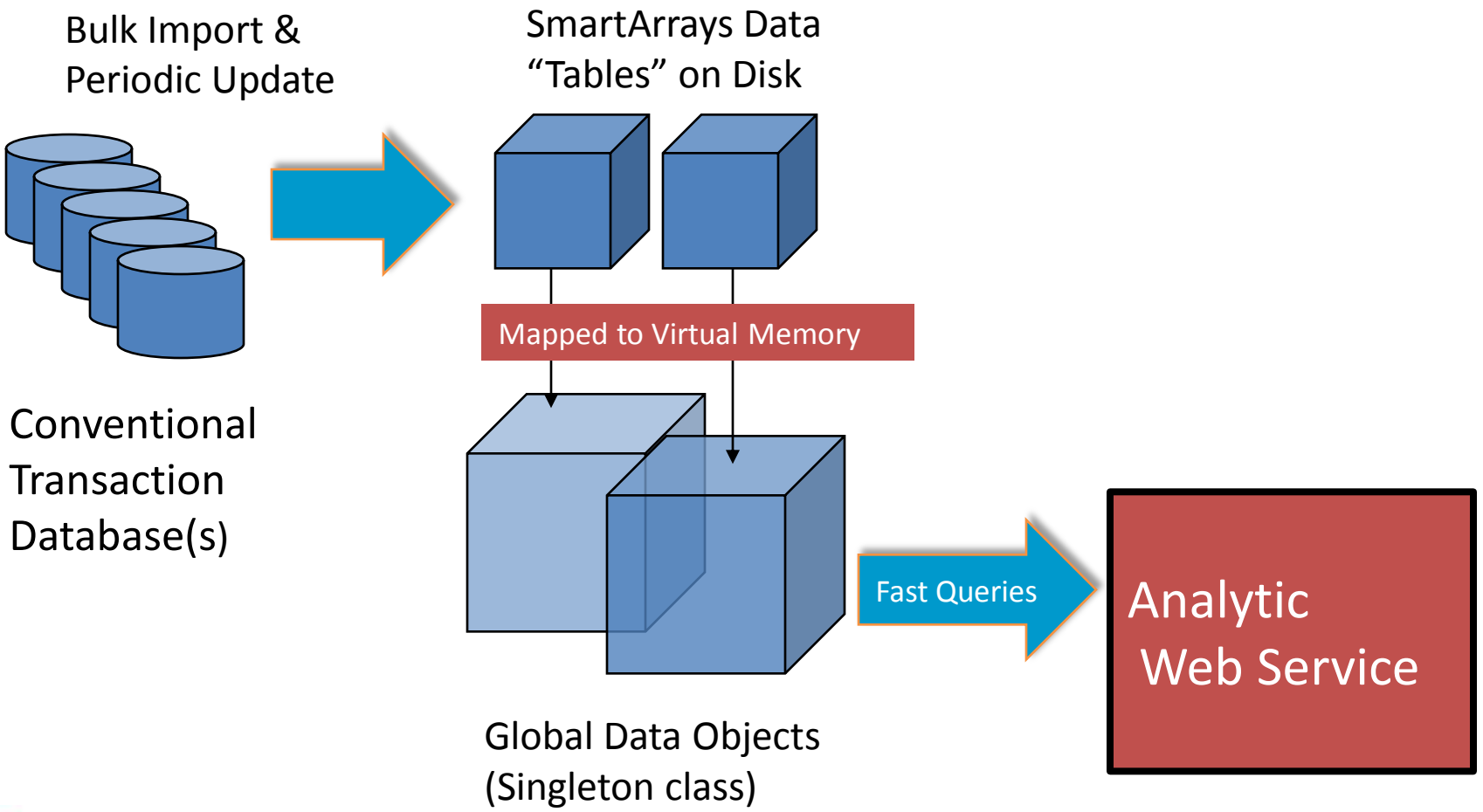
Consider: Database of Reported Financials for Public Companies

All the quarterly data for 100,000 companies for 10 years will fit in about 1.2GB of memory using SmartArrays

Small enough to provide instant results to any question in a web site

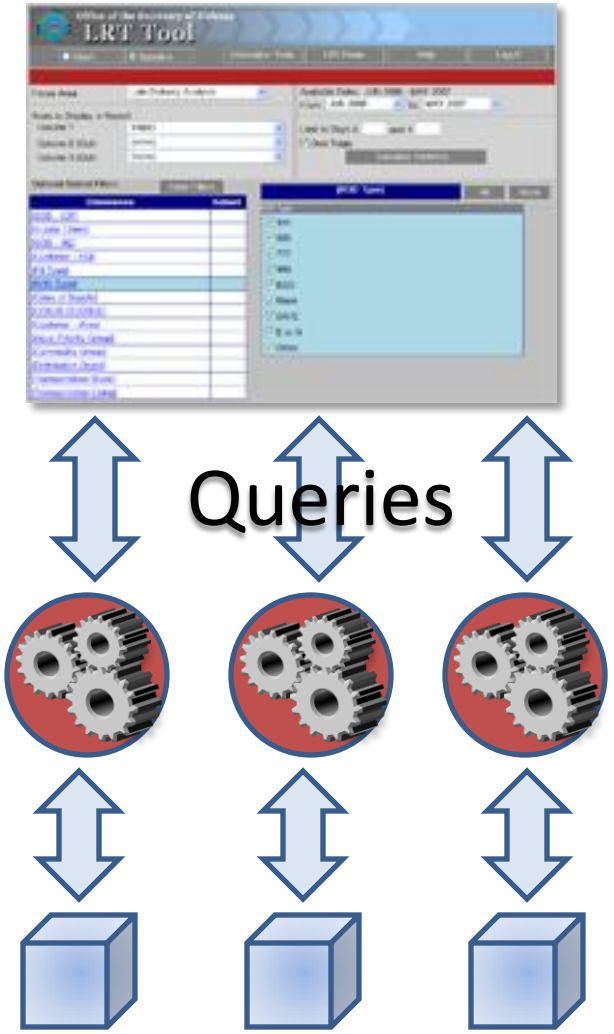
Even small enough to run on a laptop!

Typical Use of SmartArrays In a Service-Oriented Architecture



A Two-Tier Web Tool Architecture

- Web-based lightweight "Client" is the user interface
- Web client calls web services on other servers for calculations
- Web services provide central data repository



Integrating SmartArrays is Easy

- Fits comfortably into your existing application stack and data architecture.
- Just a library from the developer's perspective. No administration required.
- Package in any way appropriate:
  - Web hosted application
  - Web service
  - Client-side application.
  - Scriptable command line tools.

Anywhere application logic touches bulk data.

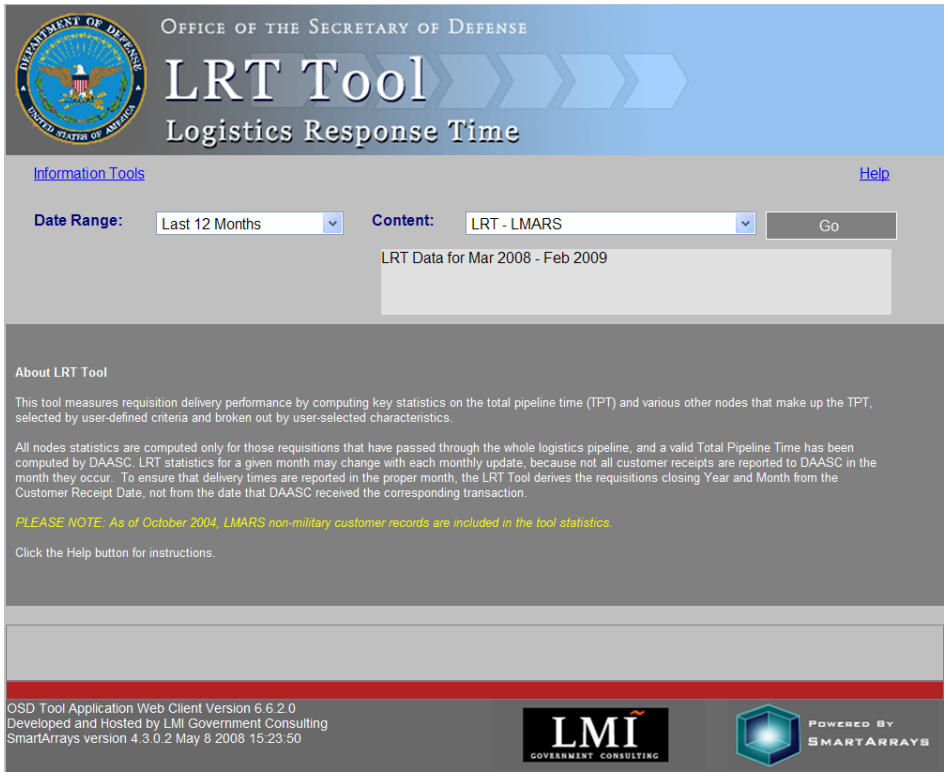
Case Study

A Statistical-OLAP Tool for the Dept of Defense

- 8 years at 35 million orders per year
- Each order record actually a time series with 200+ fields, 3KB of data.
- Need to drill-down on many dimensions with fine-grain filters on each.
- Tried other choices – nothing met requirements of atomic data and performance.

A Statistical-OLAP Tool for the Dept of Defense

(Live Demo)



Detailed Examples

Shrinking Data to Its Essence

Example: “Customer” Column in the database

- 4 million rows of varchar() data
- 100 MB on disk

“Customer” Column in SmartArrays

- 2 MB ***in memory***

SmBitmapColumn

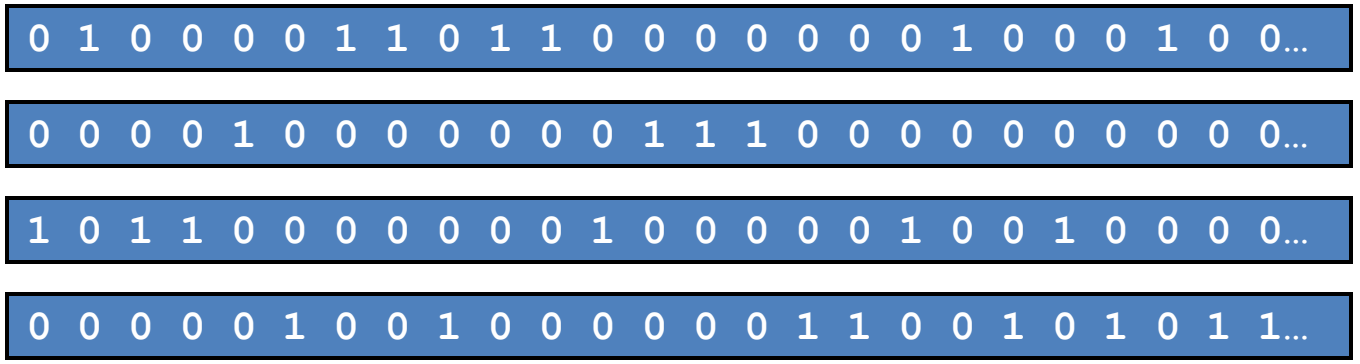
Customer column has 4 million rows but only 4 distinct values: “Air Force”, “Army”, “Navy”, “Marines”

Represent with Arrays:

- Vector of 4 strings: “Air Force”, “Army”, “Navy”, “Marines”

“Air Force”	“Army”	“Navy”	“Marines”

- Vector of 4 bitmaps, each with 4 million bits (500K)



Slicing Subsets

- Let the user pick any combination of characteristics to include in analysis.
- Let the user pick any set of dimensions to spread the data over.
- Select data subsets on-the-fly

Slicing with Bitmaps

- Select bit vector for each dimension
- Combine with “and” or “or” operations
- Use combined bitmap to “compress” data values of interest
- MUCH faster than SQL queries
- No limit on number of dimensions

Slicing with Bitmaps

Customer = “Navy”

“and”

Fill Type = “backordered”

Composite Bitmap

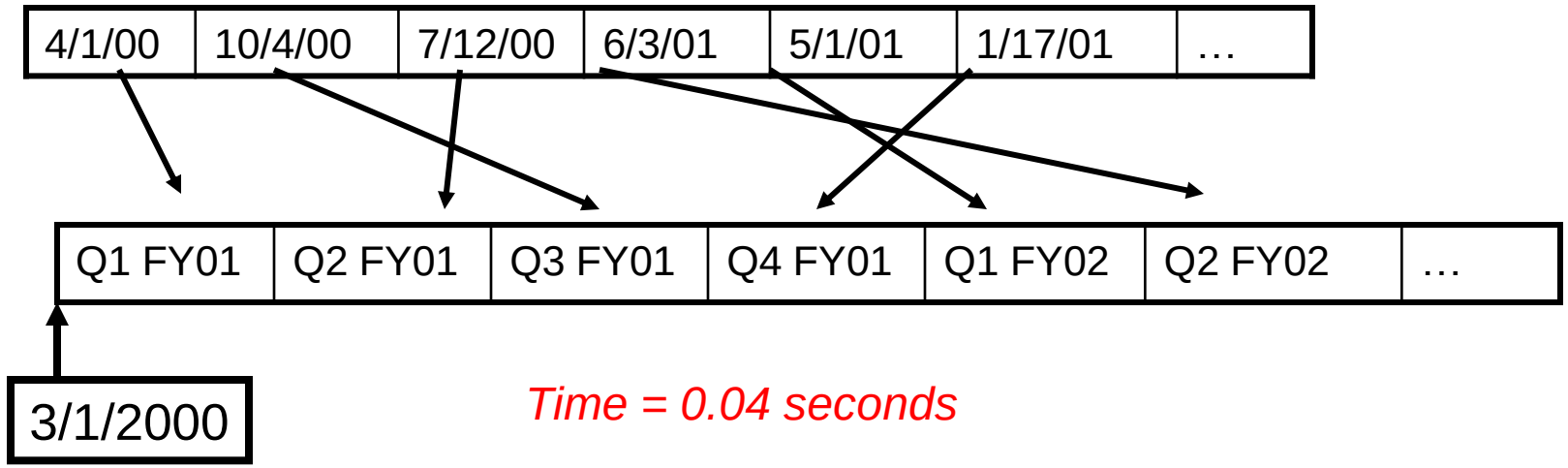
Select from a measure with “compress”

Days to Process

Subset of Interest to be analyzed

Mapping Continuous Measure Data into Buckets

- Difficult with SQL Databases
- Easy and very fast with arrays
- Example: map 1,000,000 dates into 12 quarters



Time = 0.04 seconds

```
SmArray bi = Buckets.lookupInterval(Dates);
0 2 1 5 4 3 …
```

A Full Range of Functions

Arithmetic, Comparison, Trigonometric

Mathematical: statistics, radix arithmetic, matrix inverse, etc.

Selection, filtering, subscripting

Sorting and Searching

Date and Time

Database and File I/O,

HTML, XML, formatting, string processing

More than 250 in all.

Questions

Thank You

A copy of these slides will be posted on our web site. We'll email the location.

We can repeat this session privately to your colleagues

Next steps:

- A technical "whiteboard" discussion with your requirements and your data
- A collaborative pilot project

Contact: Pat Buteux pat.buteux@smartarrays.com