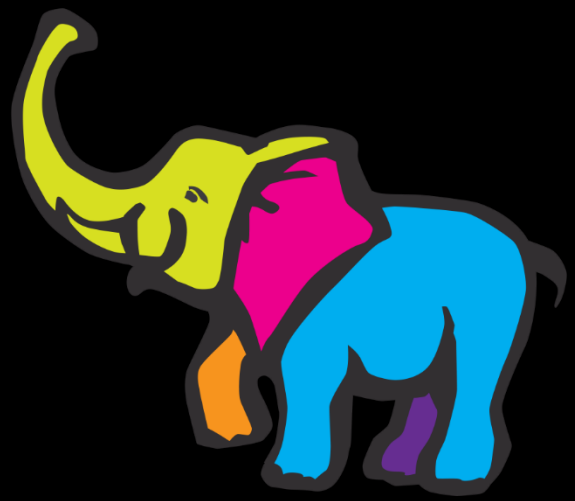
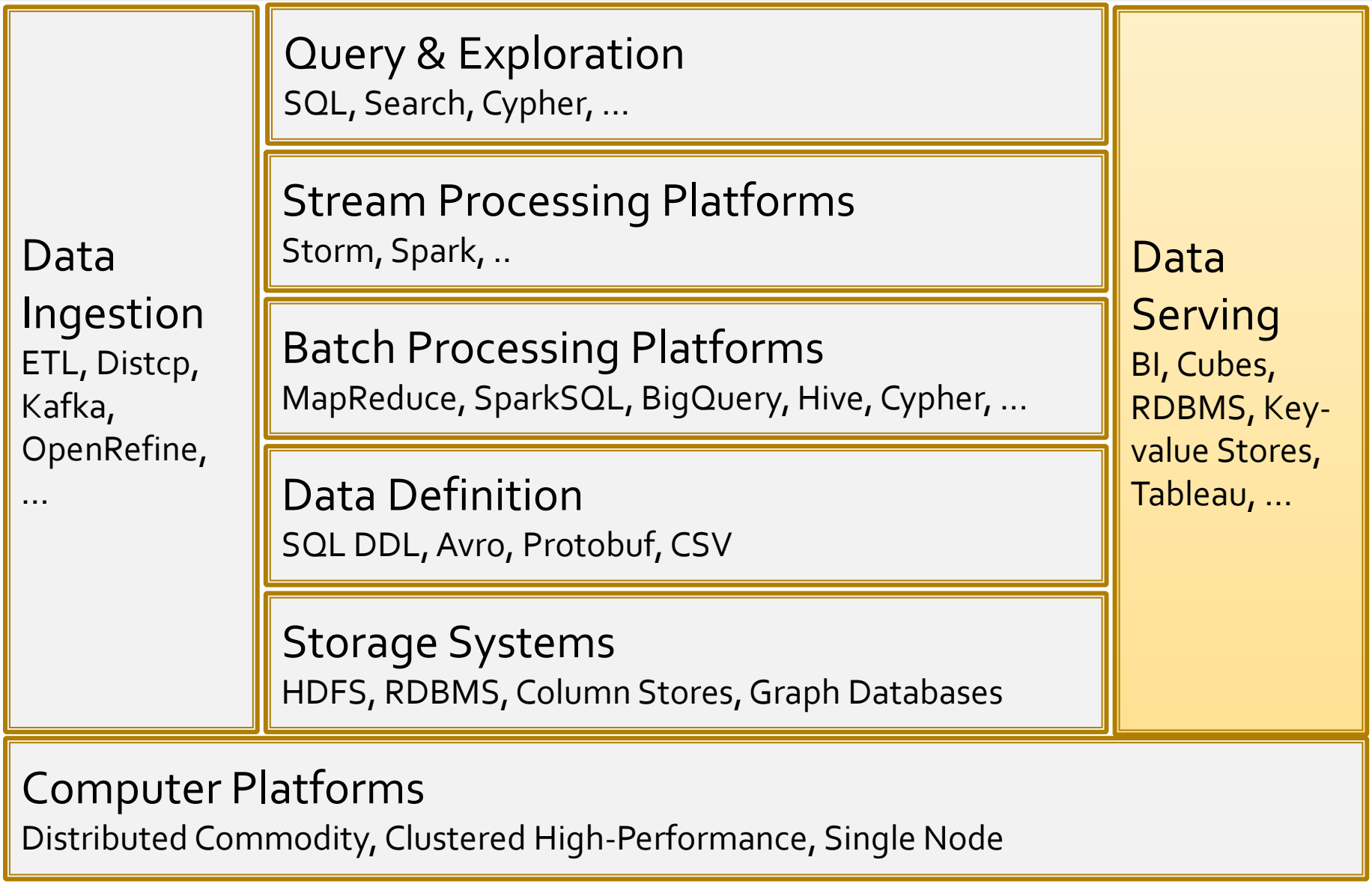
EECS4415: Big Data Systems



Serving Data

Big Data Technology & Analytics

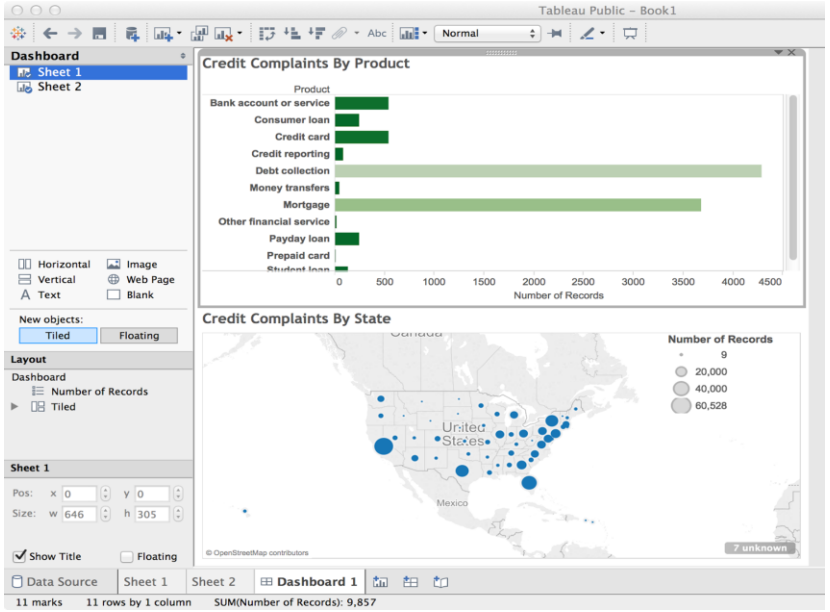


Introduction

- Purpose
  - To enable reporting
  - To power real-time analytics in services/applications (recommendation, fraud det.)
- Architectures for serving data depend on
  - The consuming system (technical, non technical)
  - The size of data (dashboards)
  - The number of consumers (concurrency)
- Considerations
  - Human/Machine?
  - Scale?

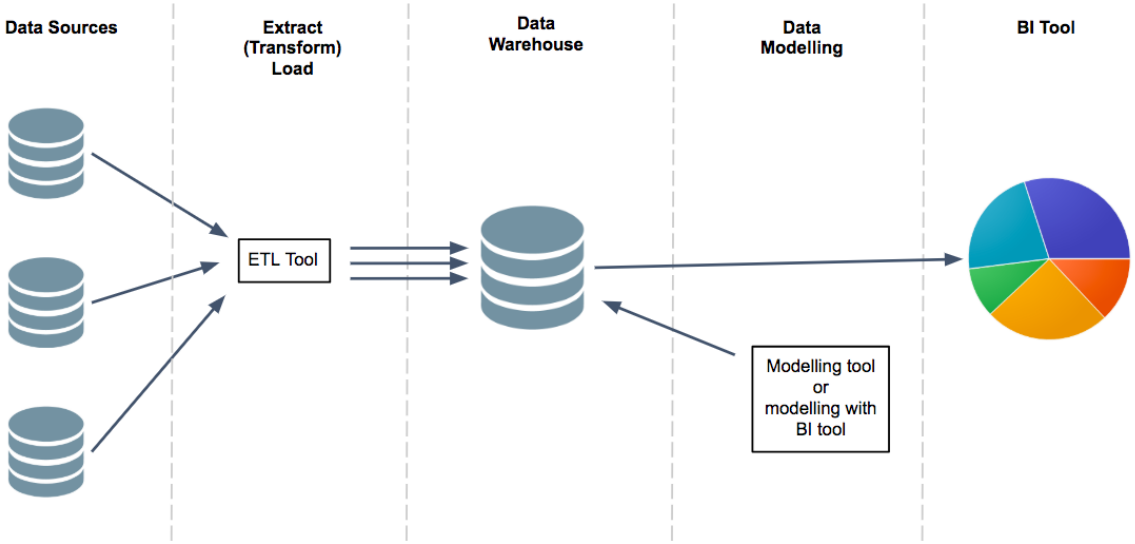
Reporting

- Reporting is accomplished by **Business Intelligence (BI) tools**
- Real-time analytics are accomplished by **In-application Analytics**



Business Intelligence (BI) Tools

How Does a BI Tool Work?



- Popular Tools
  - MicroStrategy
  - Tableau
  - Pentaho
  - Cognos
  - Spotfire
- Do-It-Yourself
  - HTML5
  - d3 and friends
  - API to get to data

OLAP/Data Cubes & Cuboids

Goals

Understand the concept of a **cube**

How are cubes computed

Pros and **cons** of cubes

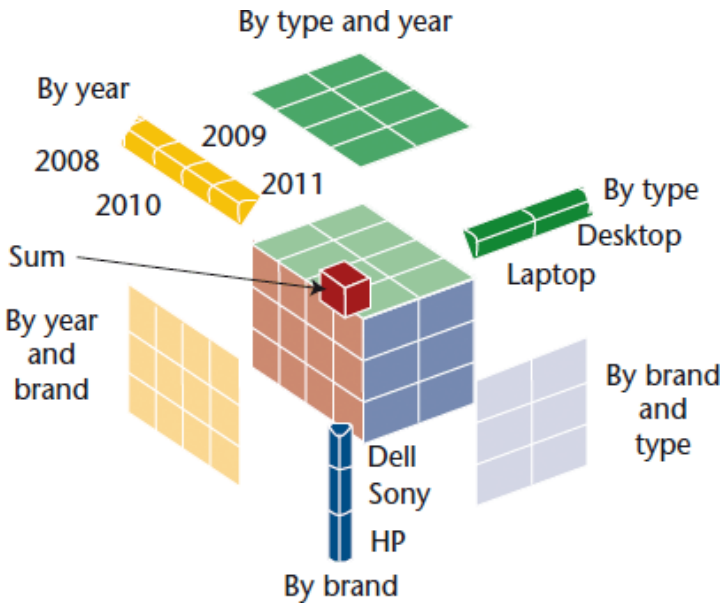
The Business Problem

A manufacturing company wants to be able to analyze and query information such as:

- How much did individual factories manufacture each: *day*, *week*, *month*?
- How much was manufactured per: *factory*, *state*, *country*?
- How much was manufactured across *different product lines*?

The Solution: Data Cubes

- An efficient solution for **OLAP** (online analytical processing)
- Computation and storage intensive
  - different implementations and optimizations

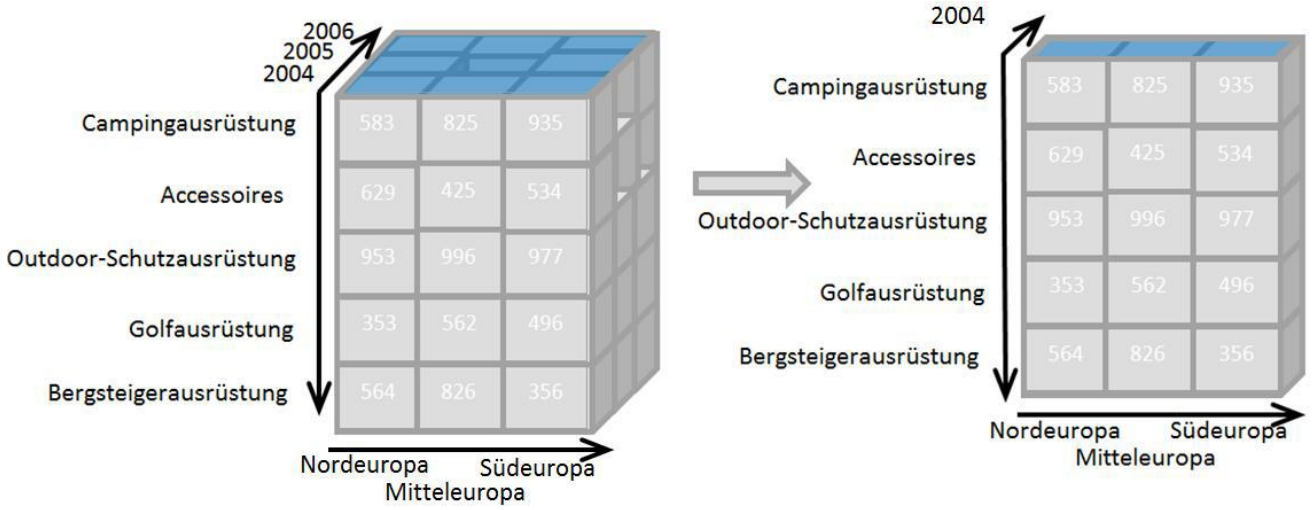


Operations on Data Cubes

- Slicing
- Dicing
- Drill down & Roll up
- Pivoting

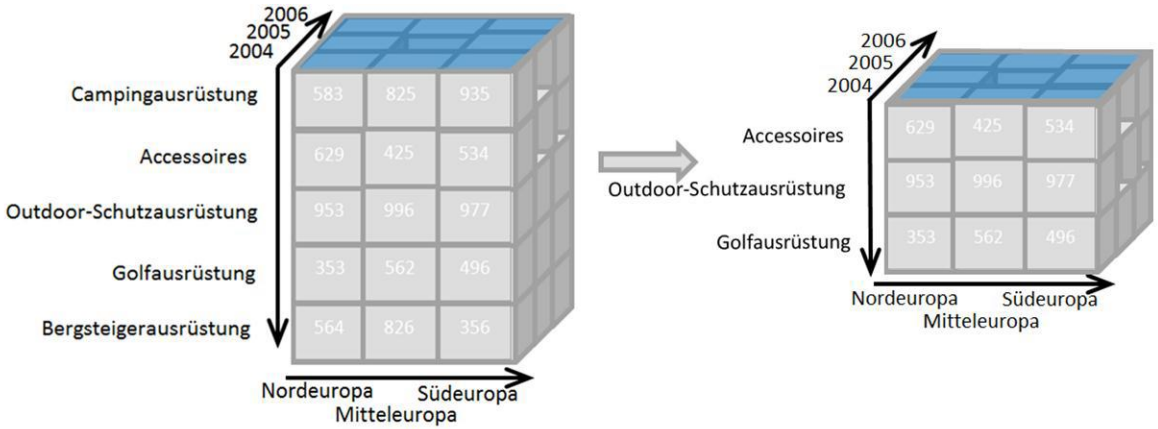
Slicing

Pick one value along one dimension
Creates a cube with one dimension less



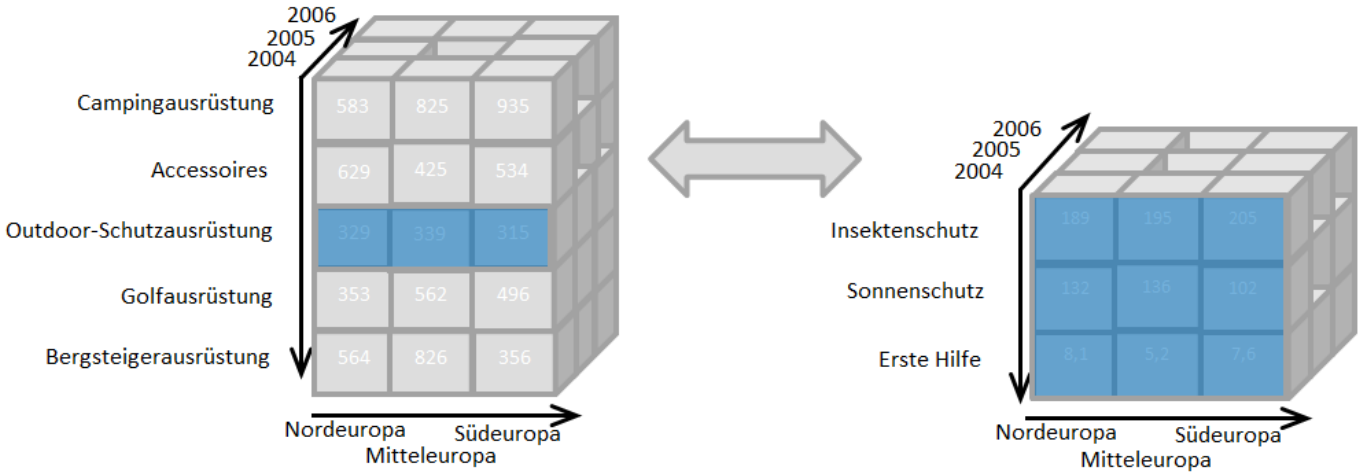
Dicing

Pick specific value along multiple dimensions
Creates a smaller cube (all dimensions)



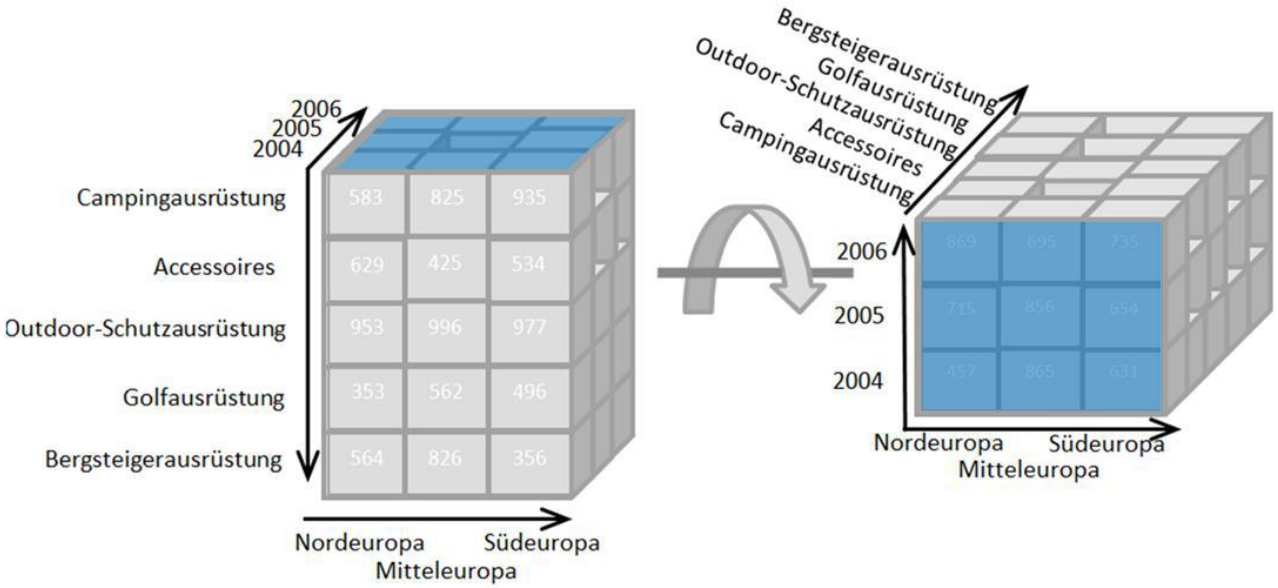
Drill down & Roll up

Change of level of granularity along a dimension, for example product, time etc.



Pivoting

"Rotation" of cube for presentation of different views of the data



ROLAP / MOLAP

ROLAP

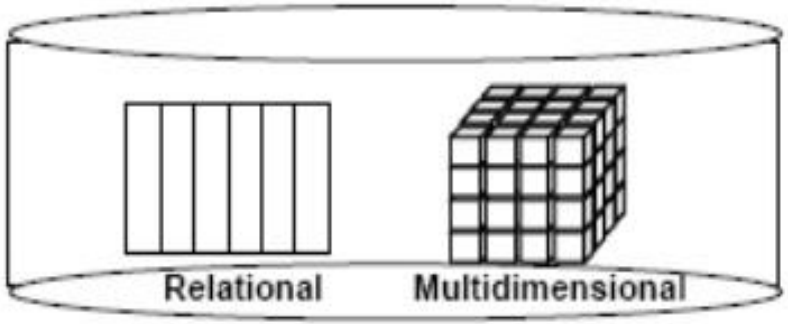
Data stored in relational database

- Performance depends on underlying query
- Generally slower than MOLAP
- Can be partially materialized and partially based on dynamic computation

MOLAP

Data stored in multidimensional array

- Good performance
- Pre-computed
- Proprietary query language and structures



Data cubes and cuboids

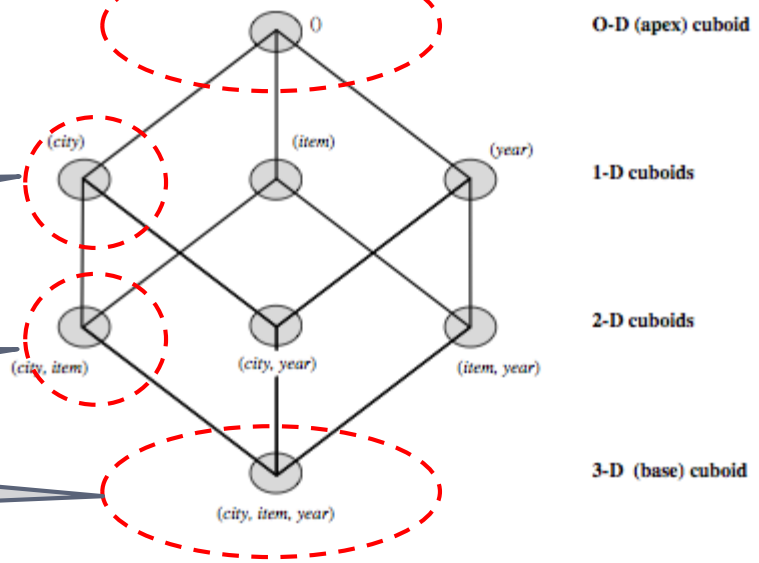
A data cube can be viewed as a lattice of **cuboids**

Most generalized, 1 value with complete aggregate (all cities, all items, all years)

Per city, all items and all years

Per city, per item items, all years

least generalized, each base value: (Chicago, Peppers, 2015)



The Technical Problem

Full cube computation of ***n*-dimensional cube** requires **2^n cuboids** (exponential to the number of dimensions) and is thus very expensive

Questions:

- How can we reduce the cost of computing a cube?
- What are the trade-offs?

Strategies to address scale issue

- Only compute cuboids satisfying defined thresholds (**iceberg cuboids**)
- Compute cuboids for a fixed number of dimensions (**cuboid shells**)
- Compute cuboid shells with fixed granularity for each dimension (**shell fragments**)

Iceberg cubes

An Iceberg-Cube contains only those cells of the data cube that meet an aggregate condition. The aggregate condition could be, minimum support, lower bound on average, min or max. The purpose of the Iceberg-Cube is to identify and compute only those values that will most likely be required for decision support queries.

```
COMPUTE CUBE sales_iceberg as
SELECT monthly, city, customer_grp, count(*)
FROM salesinfo
CUBE BY month, city, customer_group
HAVING count(*) >= min_sup
-- min_sup is min expected count
```

Iceberg cubes

Pros

- Computation can be reduced
- Storage can be reduced

Cons

- Some queries cannot be answered
- Difficult to find/identify the right threshold
- Incremental update is costly (requires recomputation)

Cube Shell and Shell Fragments

Assumption: most queries are on a subset of the dimensions ***d***

Idea: compute a **cube shell** of all cuboids of ***k*** dimension or less, where ***k*** **<<** ***d***

*Ex: Assume a 60 dimensions cube; compute all cuboids with 3 or less dimensions. Would require to compute 36,050 cuboids**

* *(60 choose 3) + (60 choose 2) + (60 choose 1) << 2^{60} = 1.1529215e+18*

Shell Fragment

Cube Shells still very expensive, many cuboids to calculate

Idea: Only a few dimensions are used in practice; fix some dimensions (from drill down)

Shell Fragments

Shell Fragment is a Shell Cuboid with fixed dimensions

Compute fragments offline

Have a fragment-aware query engine, compute full cubes online

Shell Fragments

Pros

- Can trade-off offline and online processing

Cons

- Identify the right fragments

Summary

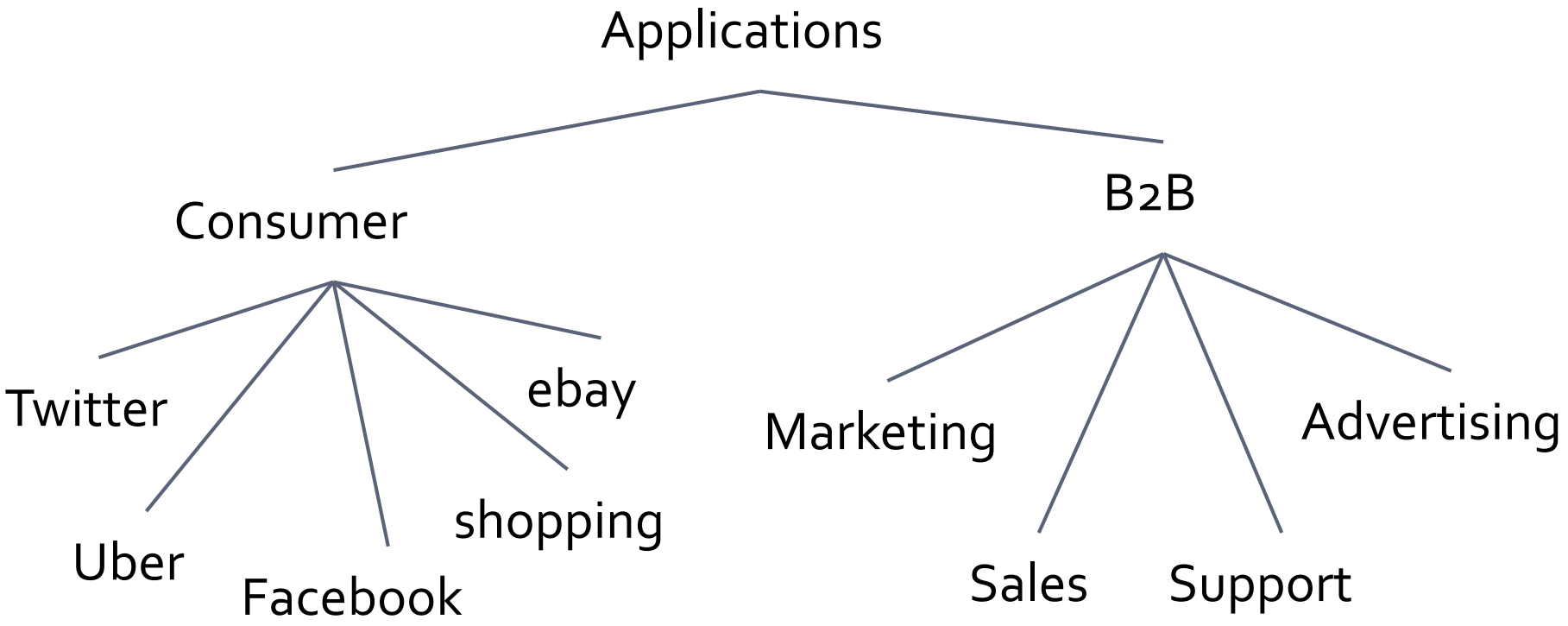
- **Data cubes** are very powerful for **online analytics processing**
- There are **ROLAP**, **MOLAP** & **HOLAP** methods
  - HOLAP stands for Hybrid OLAP
- Computing cubes is of **exponential complexity**
  - There are various ways of reducing storage and computation requirements

In-Application Analytics

What is "in-application analytics"

- Present statistics, analytics
- Recommend content items
- Adapt features to behaviour
- Detect patterns, recognize information, auto label

Types -- Examples



Feature Examples

Face detection (FB tag friends)

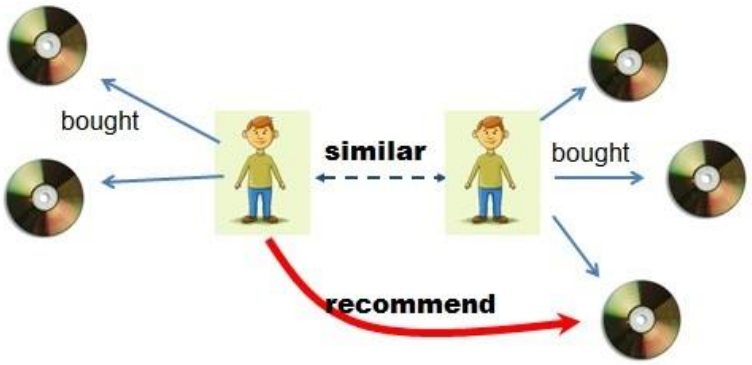
User Engagement (retweets, likes...)

Recommendations (books, friends, …)

Recommendations

Collaborative Filtering (CF)

- Similar items/users
- Recommend Item



CF Challenges

Accuracy

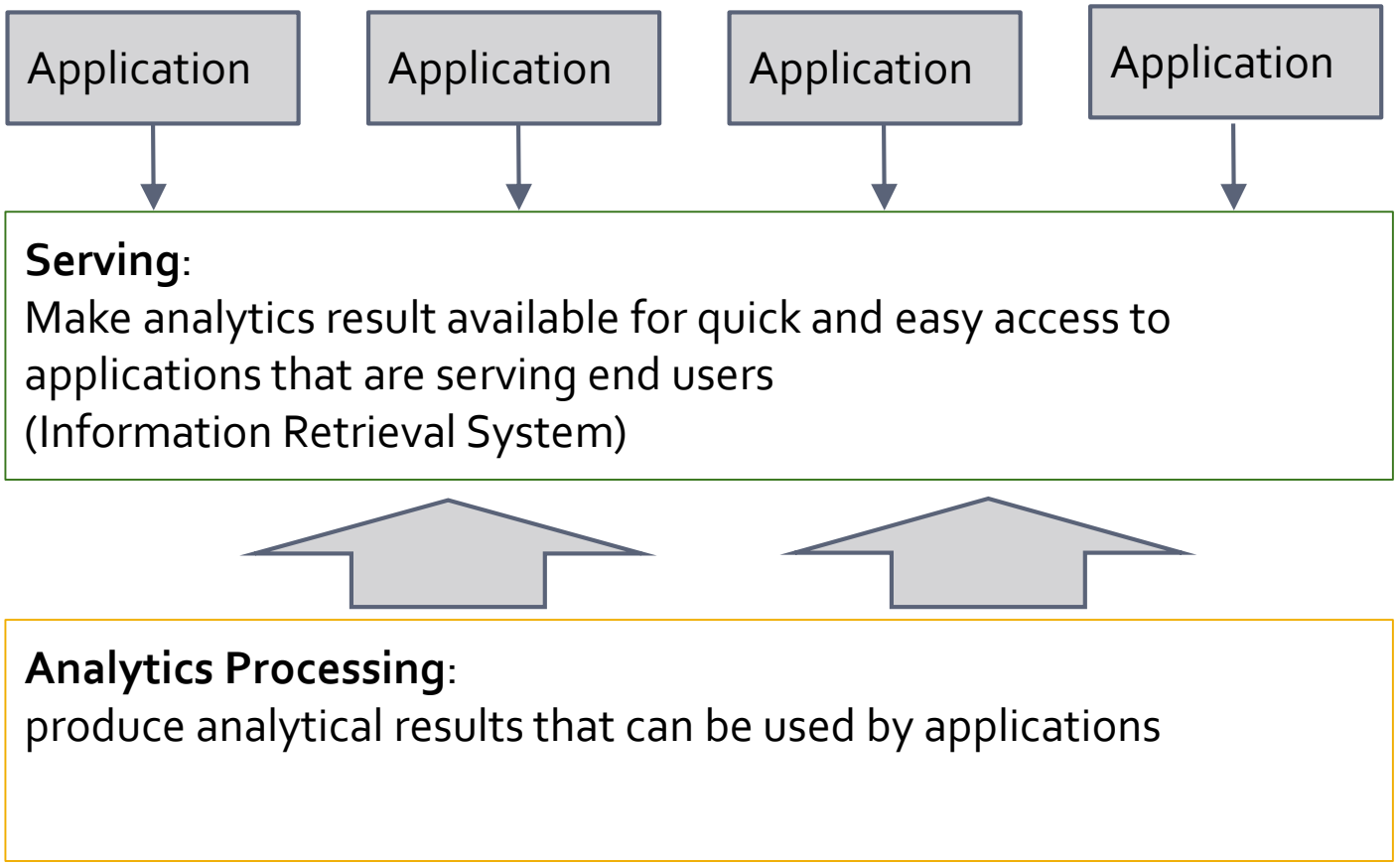
Scalability

Sparsity

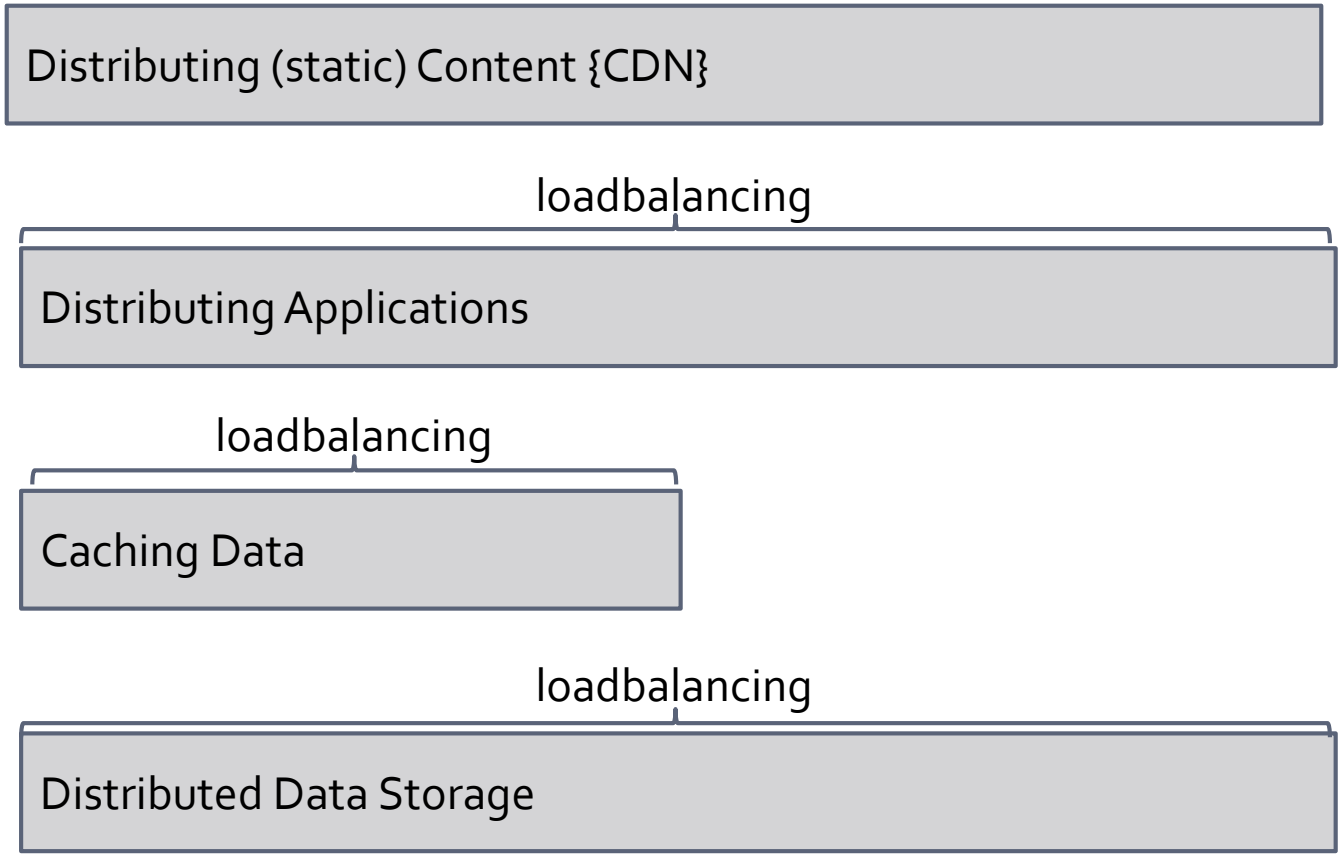
Cold-start

Serving at-Scale

Layers

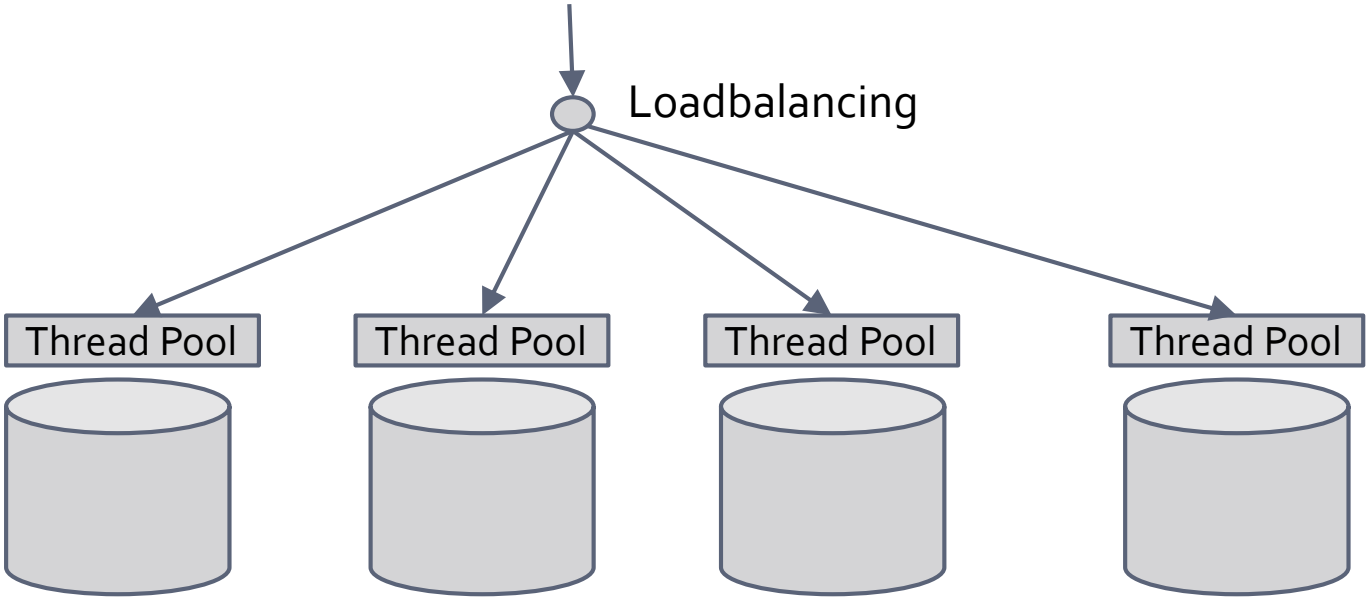

Application
Application
Application
Application
Serving:
Make analytics result available for quick and easy access to applications that are serving end users
(Information Retrieval System)
Analytics Processing:
produce analytical results that can be used by applications

Scaling Principles



Distributed Data Storage/Serving

Sharding or Partitioning



Data Sharing (Horizontal Partit.)

Key	Name	Description	Stock	Price	LastOrdered
ARC1	Arc welder	250 Amps	8	119.00	25-Nov-2013
BRK8	Bracket	250mm	46	5.66	18-Nov-2013
BRK9	Bracket	400mm	82	6.98	1-Jul-2013
HOS8	Hose	1/2"	27	27.50	18-Aug-2013
WGT4	Widget	Green	16	13.99	3-Feb-2013
WGT6	Widget	Purple	76	13.99	31-Mar-2013

Key	Name	Description	Stock	Price	LastOrdered
ARC1	Arc welder	250 Amps	8	119.00	25-Nov-2013
BRK8	Bracket	250mm	46	5.66	18-Nov-2013
BRK9	Bracket	400mm	82	6.98	1-Jul-2013

Key	Name	Description	Stock	Price	LastOrdered
HOS8	Hose	1/2"	27	27.50	18-Aug-2013
WGT4	Widget	Green	16	13.99	3-Feb-2013
WGT6	Widget	Purple	76	13.99	31-Mar-2013

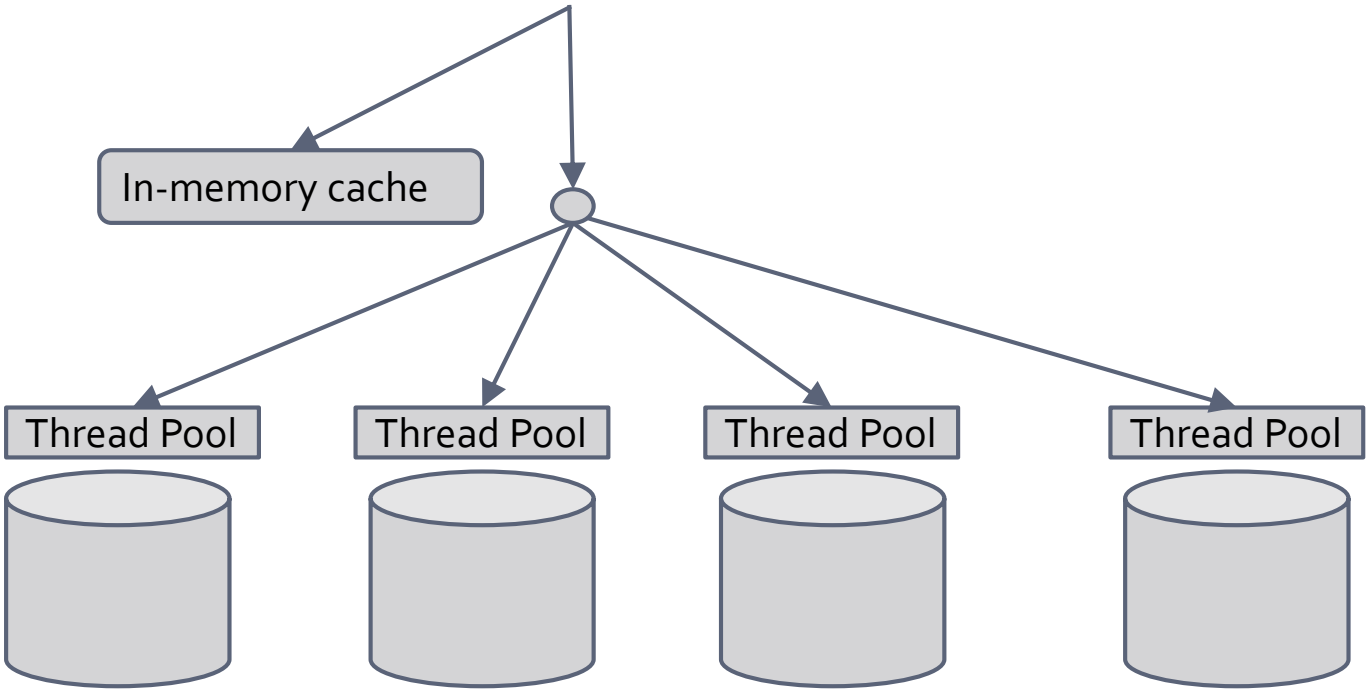
Data Sharing (Vertical Partit.)

Key	Name	Description	Stock	Price	LastOrdered
ARC1	Arc welder	250 Amps	8	119.00	25-Nov-2013
BRK8	Bracket	250mm	46	5.66	18-Nov-2013
BRK9	Bracket	400mm	82	6.98	1-Jul-2013
HOS8	Hose	1/2"	27	27.50	18-Aug-2013
WGT4	Widget	Green	16	13.99	3-Feb-2013
WGT6	Widget	Purple	76	13.99	31-Mar-2013

Key	Name	Description	Price
ARC1	Arc welder	250 Amps	119.00
BRK8	Bracket	250mm	5.66
BRK9	Bracket	400mm	6.98
HOS8	Hose	1/2"	27.50
WGT4	Widget	Green	13.99
WGT6	Widget	Purple	13.99

Key	Stock	LastOrdered
ARC1	8	25-Nov-2013
BRK8	46	18-Nov-2013
BRK9	82	1-Jul-2013
HOS8	27	18-Aug-2013
WGT4	16	3-Feb-2013
WGT6	76	31-Mar-2013

Data Caching



Caching Layer

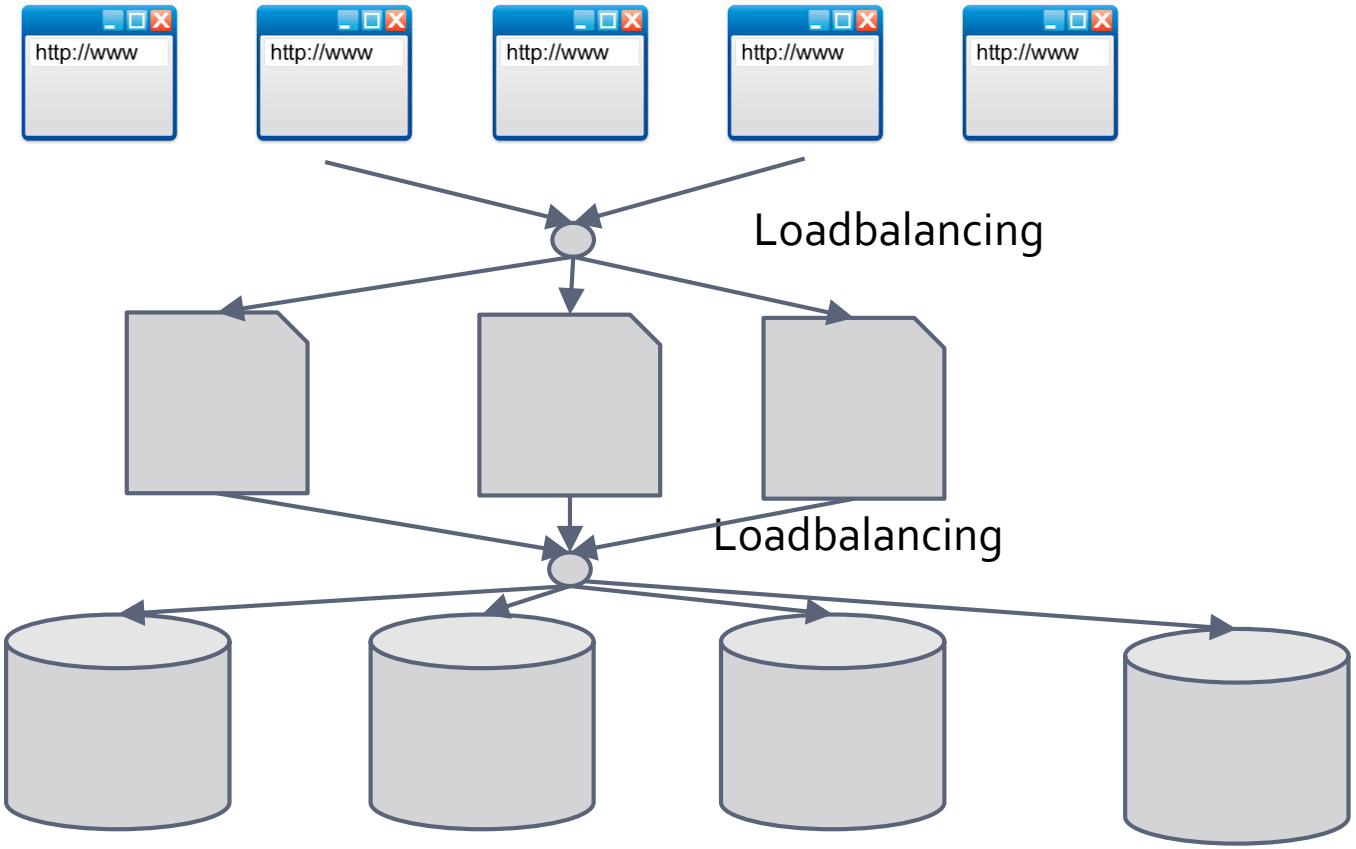
Without **Memcached**

```
function get_foo(int userid) {
   data = db_select("SELECT * FROM users WHERE userid = ?", userid);
   return data;
}
```

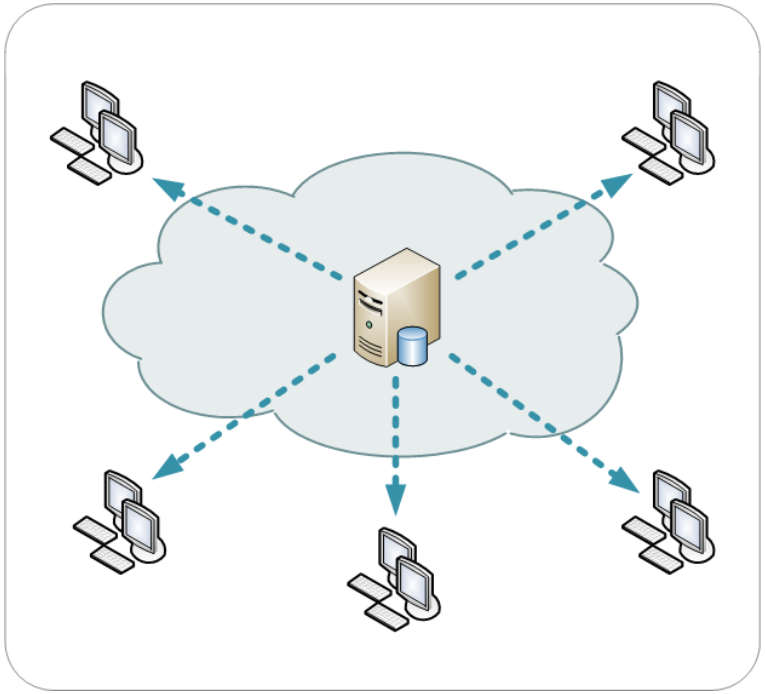
With **Memcached**

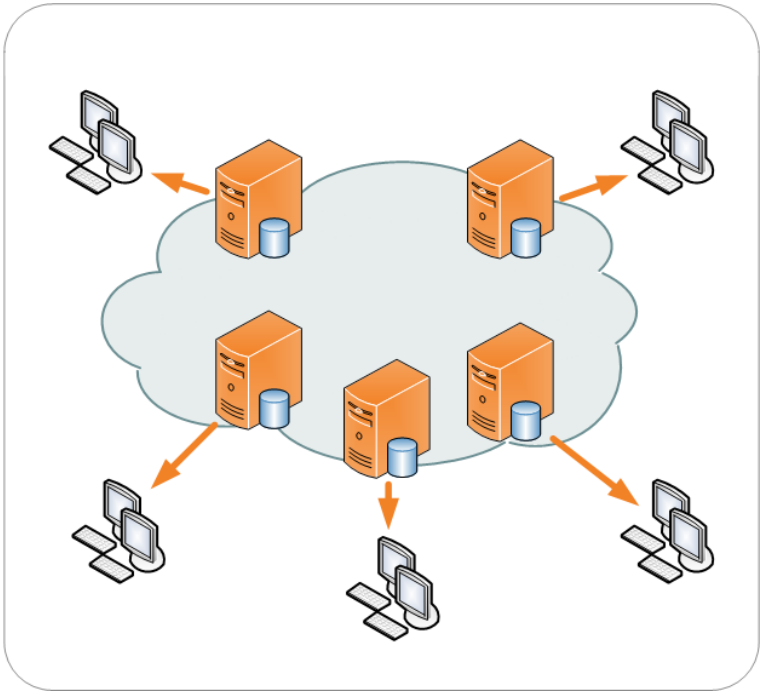
```
function get_foo(int userid) {
   /* first try the cache */
   data = memcached_fetch("userrow:" + userid);
   if (!data) {
      /* not found : request database */
      data = db_select("SELECT * FROM users WHERE userid = ?", userid);
      /* then store in cache until next get */
      memcached_add("userrow:" + userid, data);
   }
   return data;
}
```

Distributing Applications

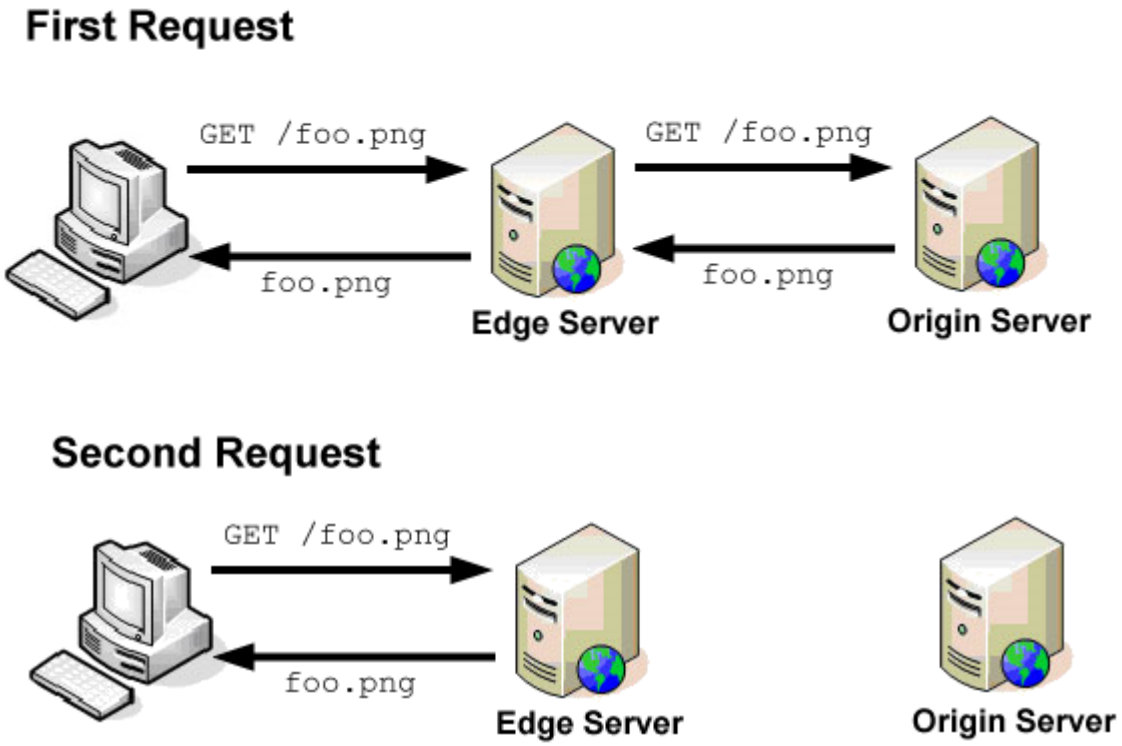


Content Delivery Network (CDN)





CDN Request, Edge Servers



- Things that need consideration
  - When to expire content
  - What content to cache

Problems

- Personalization
- Streaming data/Real-time updates

Technologies

Databases

- MySQL, Oracle DB, Postgres
- MongoDB, HBase, Cassandra

Caching

- memcached
- Redis

Load Balancing

- Various HW solutions
- HAProxy

CDNs

- Amazon CloudFront
- Azure CDN
- Akamai

Questions?