Serving Data
<table>
<thead>
<tr>
<th>Computer Platforms</th>
<th>Data Ingestion</th>
<th>Stream Processing Platforms</th>
<th>Batch Processing Platforms</th>
<th>Data Definition</th>
<th>Storage Systems</th>
<th>Data Serving</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed Commodity, Clustered High-Performance, Single Node</td>
<td>ETL, Distcp, Kafka, OpenRefine, ...</td>
<td>Storm, Spark, ..</td>
<td>MapReduce, SparkSQL, BigQuery, Hive, Cypher, ...</td>
<td>SQL DDL, Avro, Protobuf, CSV</td>
<td>HDFS, RDBMS, Column Stores, Graph Databases</td>
<td>BI, Cubes, RDBMS, Key-value Stores, Tableau, ...</td>
</tr>
</tbody>
</table>

**Query & Exploration**
- SQL, Search, Cypher, ...

**Storage Systems**
- HDFS, RDBMS, Column Stores, Graph Databases

**Big Data Technology & Analytics**

- **ETL**, Distcp, **Kafka**, OpenRefine, ...
- **Data Ingestion**
- **Stream Processing Platforms**
- **Batch Processing Platforms**
- **Data Definition**
- **Query & Exploration**
- **Data Serving**
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
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
Pick specific value along multiple dimensions
Creates a smaller cube (all dimensions)
Change of level of granularity along a dimension, for example product, time etc.
“Rotation” of cube for presentation of different views of the data
**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
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.

```sql
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*

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

Applications

Consumer
  - Twitter
  - Uber
  - Facebook
  - ebay
  - shopping

B2B
  - Marketing
  - Sales
  - Support
  - Advertising
Feature Examples

Face detection (FB tag friends)

User Engagement (retweets, likes...)

Recommendations (books, friends, ...)

WHAT SHOULD I READ NEXT?
Recommendations

Collaborative Filtering (CF)
- Similar items/users
- Recommend Item

CF Challenges
- Accuracy
- Scalability
- Sparsity
- Cold-start
Serving at-Scale
Analytics Processing:
produce analytical results that can be used by applications

Serving:
Make analytics result available for quick and easy access to applications that are serving end users (Information Retrieval System)
Scaling Principles

- Distributing (static) Content {CDN}
- Loadbalancing
- Distributing Applications
- Loadbalancing
- Caching Data
- Loadbalancing
- Distributed Data Storage
Distributed Data Storage/Serving

Sharding or Partitioning

Loadbalancing

Thread Pool
Thread Pool
Thread Pool
Thread Pool
## Data Sharing (Horizontal Partition)

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

---

<table>
<thead>
<tr>
<th>Key</th>
<th>Name</th>
<th>Description</th>
<th>Stock</th>
<th>Price</th>
<th>LastOrdered</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARC1</td>
<td>Arc welder</td>
<td>250 Amps</td>
<td>8</td>
<td>119.00</td>
<td>25-Nov-2013</td>
</tr>
<tr>
<td>BRK8</td>
<td>Bracket</td>
<td>250mm</td>
<td>46</td>
<td>5.66</td>
<td>18-Nov-2013</td>
</tr>
<tr>
<td>BRK9</td>
<td>Bracket</td>
<td>400mm</td>
<td>82</td>
<td>6.98</td>
<td>1-Jul-2013</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Key</th>
<th>Name</th>
<th>Description</th>
<th>Stock</th>
<th>Price</th>
<th>LastOrdered</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOS8</td>
<td>Hose</td>
<td>1/2&quot;</td>
<td>27</td>
<td>27.50</td>
<td>18-Aug-2013</td>
</tr>
<tr>
<td>WGT4</td>
<td>Widget</td>
<td>Green</td>
<td>16</td>
<td>13.99</td>
<td>3-Feb-2013</td>
</tr>
<tr>
<td>WGT6</td>
<td>Widget</td>
<td>Purple</td>
<td>76</td>
<td>13.99</td>
<td>31-Mar-2013</td>
</tr>
</tbody>
</table>
# Data Sharing (Vertical Partition)

<table>
<thead>
<tr>
<th>Key</th>
<th>Name</th>
<th>Description</th>
<th>Stock</th>
<th>Price</th>
<th>Last Ordered</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARC1</td>
<td>Arc welder</td>
<td>250 Amps</td>
<td>8</td>
<td>119.00</td>
<td>25-Nov-2013</td>
</tr>
<tr>
<td>BRK8</td>
<td>Bracket</td>
<td>250mm</td>
<td>46</td>
<td>5.66</td>
<td>18-Nov-2013</td>
</tr>
<tr>
<td>BRK9</td>
<td>Bracket</td>
<td>400mm</td>
<td>82</td>
<td>6.98</td>
<td>1-Jul-2013</td>
</tr>
<tr>
<td>HOS8</td>
<td>Hose</td>
<td>1/2&quot;</td>
<td>27</td>
<td>27.50</td>
<td>18-Aug-2013</td>
</tr>
<tr>
<td>WGT4</td>
<td>Widget</td>
<td>Green</td>
<td>16</td>
<td>13.99</td>
<td>3-Feb-2013</td>
</tr>
<tr>
<td>WGT6</td>
<td>Widget</td>
<td>Purple</td>
<td>76</td>
<td>13.99</td>
<td>31-Mar-2013</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Key</th>
<th>Name</th>
<th>Description</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARC1</td>
<td>Arc welder</td>
<td>250 Amps</td>
<td>119.00</td>
</tr>
<tr>
<td>BRK8</td>
<td>Bracket</td>
<td>250mm</td>
<td>5.66</td>
</tr>
<tr>
<td>BRK9</td>
<td>Bracket</td>
<td>400mm</td>
<td>6.98</td>
</tr>
<tr>
<td>HOS8</td>
<td>Hose</td>
<td>1/2&quot;</td>
<td>27.50</td>
</tr>
<tr>
<td>WGT4</td>
<td>Widget</td>
<td>Green</td>
<td>13.99</td>
</tr>
<tr>
<td>WGT6</td>
<td>Widget</td>
<td>Purple</td>
<td>13.99</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Key</th>
<th>Stock</th>
<th>Last Ordered</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARC1</td>
<td>8</td>
<td>25-Nov-2013</td>
</tr>
<tr>
<td>BRK8</td>
<td>46</td>
<td>18-Nov-2013</td>
</tr>
<tr>
<td>BRK9</td>
<td>82</td>
<td>1-Jul-2013</td>
</tr>
<tr>
<td>HOS8</td>
<td>27</td>
<td>18-Aug-2013</td>
</tr>
<tr>
<td>WGT4</td>
<td>16</td>
<td>3-Feb-2013</td>
</tr>
<tr>
<td>WGT6</td>
<td>76</td>
<td>31-Mar-2013</td>
</tr>
</tbody>
</table>
Data Caching

In-memory cache

Thread Pool

Thread Pool

Thread Pool

Thread Pool
Caching Layer

Without Memcached

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

With Memcached

```php
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

Loadbalancing

Loadbalancing

http://www
http://www
http://www
http://www
http://www
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?