ETL on Hadoop – What is Required

Keith Kohl
Director, Product Management
October 2012
Agenda

- **Who is Syncsort**
- **Extract, Transform, Load (ETL)**
  - Overview and conventional challenges
  - Use Case
  - Challenges on Hadoop
  - Requirements on Hadoop
- **DMExpress Hadoop**
  - Overview and Architecture
  - Capture and extract data into/out of Hadoop
  - Pluggable Sort
  - Improve user productivity
  - Seamless improve Hadoop MapReduce performance
  - Use Case using the plug-in
- **Summary**
What’s Needed to Drive Success for Hadoop

- **Enterprise tooling to become a complete data platform**
  - Open deployment & provisioning
  - Higher quality data loading
  - Monitoring and management
  - APIs for easy integration

- **Ecosystem needs support & development**
  - Existing infrastructure vendors need to continue to integrate
  - Apps need to continue to be developed on this infrastructure
  - Well defined use cases and solution architectures need to be promoted

- **Market needs to rally around core Apache Hadoop**
  - To avoid splintering/market distraction
  - To accelerate adoption

Source: Hortonworks, 2012
### About Syncsort

**Multinational Software Company**
- Founded 1968, operating today in North America, Europe and Asia
- 40+ Years of Performance Innovation
- 25+ Issued & Pending Patents
- Investors:

  ![Investor Logos]

**Large Global Customer Base**
- Leading provider of Data Integration and Data Protection solutions to businesses and governments worldwide
- 15,000+ deployments in 68 countries

**Syncsort Data Integration Offerings**
- DMExpress™ family of high-performance, purpose-built Data Integration solutions for integrating, optimizing and migrating Big Data
- MFX™ high-performance sort solutions for z/OS and SAS mainframe environments

<table>
<thead>
<tr>
<th>DATA SERVICES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experian</td>
</tr>
<tr>
<td>Global Payments</td>
</tr>
<tr>
<td>comScore</td>
</tr>
<tr>
<td>D&amp;B</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FINANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barclays Capital</td>
</tr>
<tr>
<td>BBVA Compass</td>
</tr>
<tr>
<td>JPMorganChase</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>INSURANCE &amp; HEALTHCARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nationwide</td>
</tr>
<tr>
<td>HCA</td>
</tr>
<tr>
<td>Presbyterian</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PUBLIC SECTOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMS</td>
</tr>
<tr>
<td>NEA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TELECOMMUNICATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orange</td>
</tr>
<tr>
<td>MetroPCS</td>
</tr>
<tr>
<td>Sprint</td>
</tr>
<tr>
<td>Verizon Wireless</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RETAIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CarQuest</td>
</tr>
<tr>
<td>Overstock.com</td>
</tr>
<tr>
<td>Shopzilla</td>
</tr>
<tr>
<td>The co-operative</td>
</tr>
</tbody>
</table>
What is ETL

Extract Transform Load

Data Warehouse

ETL

Oracle
File XML
ERP
Mainframe
Real-Time

ETL

Data Mart

ETL

Data Mart

ETL

Data Mart

Hadoop

Syncsort
Big Data Is Pushing the Limits of Conventional Data Integration

“Big Data” + Conventional DI Solutions =

- **VOLUME**: Unprecedented creation of data
- **VELOCITY**: Shrinking batch windows
- **VARIETY**: Unstructured data, mobile devices, cloud

- **UNDERPERFORMING**: Never designed or optimized for today’s increasing performance requirements
- **INEFFICIENT**: Requires massive investments in hardware, storage and database capacity
- **COMPLICATED**: Heavy architecture, disparate tools makes it difficult to develop, tune, and maintain

- Exponential hardware and storage costs
- More IT effort to develop and tune
- Longer development cycles
ETL Use Case using Hadoop

- Hadoop is replacing existing or conventional ETL processes
  - ETL layer and/or Data Warehouse can not handle data volumes or processing
- Hadoop is a huge sink of cheap storage and processing
- Hadoop is used to as ETL layer to process the very large volumes of data prior to the loading in the data warehouse
  - Web log processing
  - Filtering/Cleansing
  - Aggregations: Aggregates are built in Hadoop and exported

[http://www.slideshare.net/brocknoland/common-and-unique-use-cases-for-apache-hadoop](http://www.slideshare.net/brocknoland/common-and-unique-use-cases-for-apache-hadoop)
ETL on Hadoop Challenges: Load and Extract into Hadoop

Structured data stores: RDBMS, Enterprise Data Warehouses, NoSQL
- Sqoop: A tool to automate data transfer between structured data stores and Hadoop
- Command-line interface, no GUI
  
```bash
  sqoop import --connect jdbc:mysql://localhost/acmedb --table ORDERS --username test --password ****
```
- JDBC-based connectivity

Files
- Hadoop shell commands: put and get
- Command-line interface, no GUI
- Serial interface to Hadoop
  
```bash
  hadoop dfs -put localfile /user/hadoop/hadoopfile
```

**BUT...**
- Requires manual scripting/coding
- Relatively slow
- No pre-processing capabilities

✓ Without ETL (cleanse, reformat, filter), “garbage in, garbage out”
✓ How to prepare data for optimum performance in Hadoop?
ETL on Hadoop Challenges: MapReduce Development & Maintenance

MapReduce Development
- Lots of manual coding
  - Java, Pig
- Limited skills available
- Heavy learning curve
- No GUI
- Limited metadata capabilities
- No mainframe connectivity
ETL on Hadoop Challenges: Performance Tuning

- Developer and Operations need to be aware of Hadoop configurations
  - Memory and output size (especially mapper output)
- Hundreds of configuration tuning parameters
  - Hadoop configuration
  - JVM configuration
  - Understanding disk I/O speeds, memory allocations, cache and spill over
  - Scheduling (i.e., simultaneous tasks)
ETL Requirements on Hadoop

- **Comprehensive transformation capabilities**
  - Aggregations, cleansing, filtering, reformatting, lookups, data type conversions, date/time, string manipulations, arithmetic, and so on

- **Connectivity**
  - Native mainframe data access and conversion capabilities
  - Heterogeneous database management system (DBMS) access on the cluster for reading into Hadoop, loading warehouses from Hadoop, as well as for sourcing data for lookups and other transformations

- **Improves user productivity**
  - Graphical development environment that minimizes the need for Java coding and Pig script development
  - Built-in metadata capabilities, which enable greater transparency into impact analysis, data lineage, and execution flow; facilitates re-use

- **Optimizes MapReduce processing, seamlessly**
  - Reduces tuning for developers and operations
Agenda

- Who is Syncsort
- Extract, Transform, Load (ETL)
  - Overview and conventional challenges
  - Use Case
  - Challenges on Hadoop
  - Requirements on Hadoop
- DMExpress Hadoop
  - Overview and Architecture
  - Capture and extract data into/out of Hadoop
  - Pluggable Sort
  - Improve user productivity
  - Seamless improve Hadoop MapReduce performance
  - Use Case using the plug-in
- Summary
DMExpress Hadoop Overview

Install in Minutes. Deploy in Weeks. Never Tune Again.

User Interface

- Task Editor
- Job Editor
- SDK

Shared File-based Metadata Repository
- Shared File-based Metadata Interchange
- Impact Analysis
- Data Lineage
- Global Search

DMExpress Server Engine
- High Performance Transformations
- High Performance Functions
- Automatic Continuous Optimization

High Performance Connectivity
- Appliances
- RDBMS
- CRM / ERP
- Real Time
- Cloud
- Hadoop
- Files / XML
- Native, Direct I/O
- Mainframe

Template-driven Design
Small Footprint ETL Engine
Self-tuning Optimizer
Native, Direct I/O Access

Syncsort
DMExpress delivers high-performance connectivity and pre-processing capabilities to optimize Hadoop environments.

**Extract**
- Files
- CRM/ERP
- Cloud
- Appliances
- XML
- RDBMS
- Mainframe

**Preprocess & Compress**
- Sort
- Aggregate
- Join
- Compress
- Partition

**Load**
- Data Node
- Data Node
- Data Node
- Data Node
- HDFS

Connect to virtually any source

Pre-process data to cleanse, validate, & partition for better and faster Hadoop processing and significant storage savings.

Load data up to 6x faster!
Connecting to HDFS

Remote File Connection

- Name: Cloudera-DMX-HDFS
- File server: hadoop-1.vm
- Current DMExpress server: 192.168.91.145
- Connection type: Hadoop Distributed File System (HDFS)
- Authentication: Hadoop Distributed File System (HDFS)
- User name: dmxdemo
- Password: 
- Key set: 

[Verify connection]
HDFS Connectivity

- Partition the output for parallel loading
- Makes full use of network bandwidth with reduced elapsed time
- Hadoop/DMExpress can process wildcard input files from HDFS
DMExpress Accelerates HDFS Loading

HDFS Load using DMExpress

- **HDFS Load**
  - 20 partitions
  - Uncompressed input file size from 100GB to 2100GB

- **Cluster Specifications**
  - Size: 10+1+1 nodes
  - Hadoop distribution: CDH4
  - HDFS block size: 256 MB

- **Hardware Specifications (Per Node)**
  - Red Hat EL 5.8
  - Intel Xeon x5670 *2
  - Memory: 94 GB

---

**DMExpress HDFS Load vs Hadoop put**
(data sizes 100GB and larger)

6x Faster!

Elapsed time (Seconds)

File Size (GB)
Syncsort MapReduce Contribution to Apache Hadoop

- Allow external sort to be plugged in
- Seamlessly accelerate MapReduce performance
  - Replace Map output sorter
  - Replace Reduce input sorter
- Improve developer productivity
  - Develop MapReduce jobs via DMExpress GUI
    - Aggregations, cleansing/filtering, reformatting, etc.

Native Sort:
- Not modular
- Limited capabilities
- Difficult to fine-tune & configure (requires coding & compilation)

Contribution:
- Modular
- Extensible
- Configurable through use of external sorters on MapReduce nodes

[https://issues.apache.org/jira/browse/MAPREDUCE-2454](https://issues.apache.org/jira/browse/MAPREDUCE-2454)
Apache Hadoop Pluggable Sort

- The main MapReduce code changes enable sorting to be done by external implementations
  - MapTask.java: modified the class MapOutputCollector (which does the sorting on the Map side)
    - The intermediate output of the external sorter complies with the Hadoop framework’s sorter output format
  - ReduceTask.java: modified such that when an external sorter is plugged in, the shuffled data will be sent to the external sorter
  - The Hadoop framework code for shuffling is not modified

- Replaces up to 4 phases in MapReduce flow
  - Mapper output sorter (seamless)
  - Partitioner (optional)
  - Combiner (optional)
  - Reduce input sorter (merger) (seamless)

https://issues.apache.org/jira/browse/MAPREDUCE-2454
Improve User Productivity

Use DMExpress to Accelerate Development and Maintenance of MapReduce Jobs

MapReduce Development:
- X Lots of manual coding:
  - X MapReduce, Pig, Java
- X Limited skills supply
- X Heavy learning curve

DMExpress Hadoop Edition:
- ✓ No coding required – full ETL capabilities
- ✓ Leverages the same skills most IT organizations already have
- ✓ New resources can be trained in just 3 days
Extracting from Hadoop to Load Data Warehouse

- Reducer to write/load to warehouse
  - Load data to analytic database/warehouse
  - No need to land data in HDFS and then extract
  - Use native load utilities from Hadoop
    - Teradata TTU/TPT, GP, Vertica, Netezza, Oracle, etc.
  - Note: Control number of reducers

Note: Read right to left
DMExpress Hadoop runs **native** on each data node on the cluster
- DMExpress is installed on each data node
- Same benefits as High-performance ETL

DMExpress Hadoop is **not** generating MR code (i.e., Java, Pig, Python)

**Issues with code generation**
- Can’t change code without breaking link
- Requires re-compilation with every change
- May still require MR skills
- Always issues with efficiency of generated code
DMExpress Hadoop – TeraSort Benchmark

- **TeraSort Benchmark**
  - GZIP compression
  - Uncompressed input file size from 100GB to 4TB

- **Cluster Specifications**
  - Size: 10+1+1 nodes
  - Hadoop distribution: CDH3
  - HDFS block size: 256 MB

- **Hardware Specifications (Per Node)**
  - Red Hat EL 5.8
  - Intel Xeon x5670 *2
  - Memory: 94 GB

**Over 2x Faster!**

![Graph showing TeraSort Benchmark results]

File Size (GB) vs. Elapsed Time (min)
DMExpress Hadoop – ETL Benchmark

ETL Benchmark
- Filter & Aggregation
- TPC-H data
- Uncompressed input file size from 100GB to 2.4TB

Cluster Specifications
- Size: 10+1+1 nodes
- Hadoop distribution: CDH3u2
- HDFS block size: 256 MB

Hardware Specifications (Per Node)
- Red Hat EL 5.8
- Intel Xeon x5670 *2
- Memory: 94 GB

Almost 2x Faster than Java; Over 2x Faster Pig
Example of MR using the External Sort Plug-in

- **Goal:** Aggregate tagged tracks
- **Published listening data from last.fm via JSON**
  - Per track listened to, timestamp, list of similar tracks (big list), list of tags, artist, track title, etc.
  - Tags (Romantic, Ambient, Love, Pop, 90s, Bicycle, Hard-workout, etc.)
- **Wrote a Mapper in Java to parse JSON and Pivot the data**
  - 1 record/tag with track info (multiple tags per track)
- **Used Map-Sort step via plug-in to call DMExpress**
  - Aggregate by tag
- **Reduce-Sort step**
  - Aggregation to pull all of the combined map outputs together
- **Still have a Reduce step which could call Java to do something (convert to XML)**

This is not a customer reference, purely a use case of the external plug-in and DMExpress
Faster Connectivity. Optimized Processing. Easier Development

Load & Extract from HDFS
- Partition output for parallel loading
- 16:1 compression (Gzip)
- Optimum network I/O utilization
- Process wildcard input files from HDFS

Easily Implement & Process Optimized MapReduce
- No coding
- Leverage existing skills
- Increased throughput by more than 2X
- No code generation
- No compiling
Thank you!

RETHINK THE ECONOMICS OF DATA

syncsort
DMExpress Hadoop Edition

Accelerate Development. Optimize Performance. Reduce TCO.

- **Fast**
  - Reduce elapsed processing time for existing Hadoop deployments

- **Efficient**
  - Reduce resource utilization (CPU, memory, I/O)
  - Enhance Scalability

- **Simple**
  - No tuning
  - No coding
  - No MapReduce scripting
  - Seamless plug-in

- **Cost Effective**
  - ✓ Less Hardware Costs
  - ✓ Less IT Staff hours wasted on constant coding & tuning
  - ✓ Increased IT staff productivity

Accelerate Development. Optimize Performance. Reduce TCO.