CSC590: Selected Topics
BIG DATA & DATA MINING

Lecture 3

Feb 26, 2014

Dr. Esam A. Alwagait

Agenda

- Introduction
- ETL : Extract, Transform, Load
- Big Data Analysis
- Big Data Analysis Tools
- Case Studies
- Assignments for next lectures

Introduction

- Last Lecture we discussed Big Data in general
  - Characteristics
  - Features
  - Challenges
- In this lecture we will dig a little deeper
- Big Data analysis
  - ETL: Extract, Transform, and Load
  - MapReduce
  - Tools
  - Case Studies/ Examples

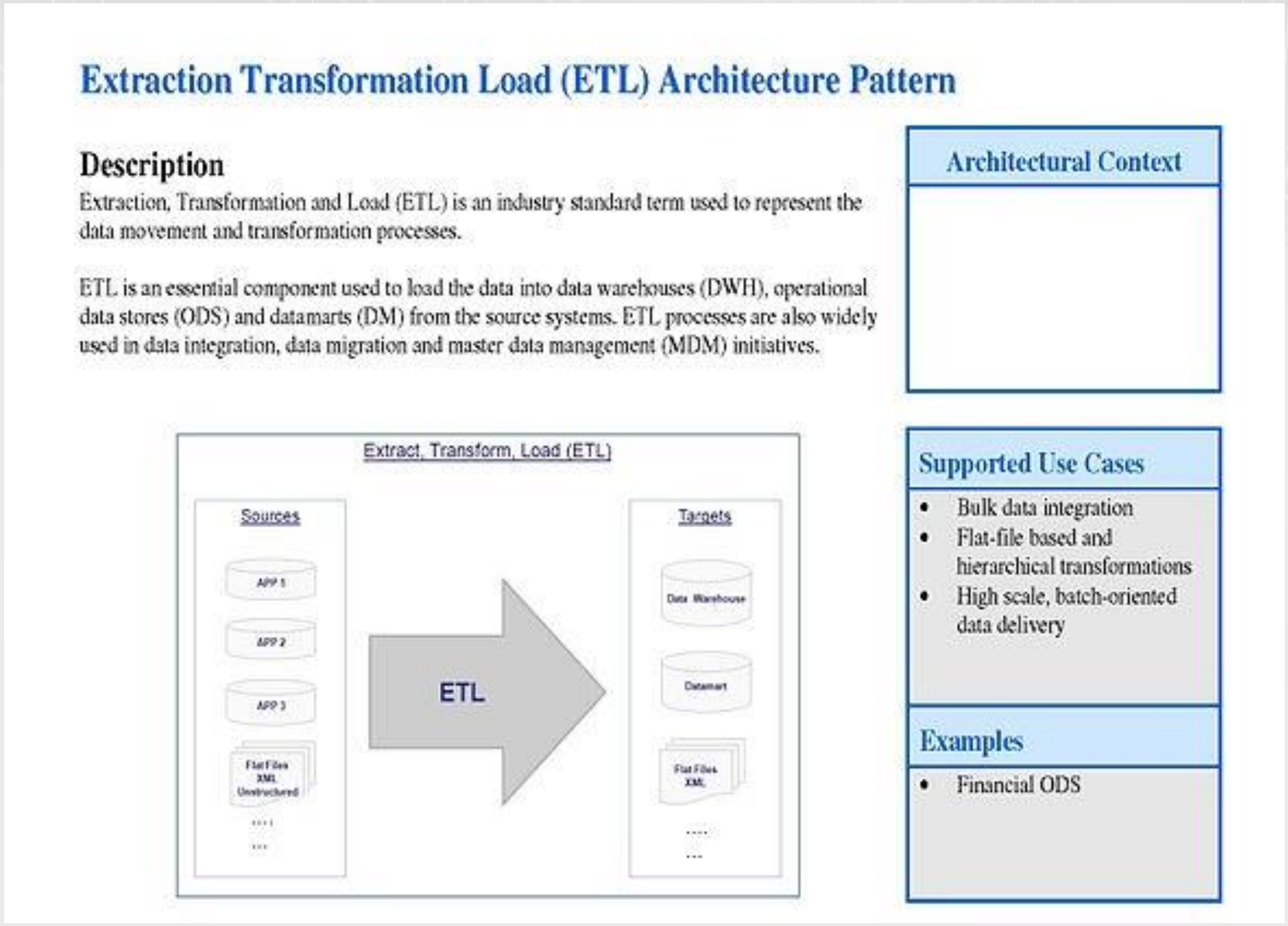
Big Data Analysis

- We know Big Data is BIG.. So, how can we analyze it ?
- Huge amount of data needs to be
  - Clusterd (Divided)
  - Manipulated
  - Analyzed
  - Summerized

ETL

- Extract, Transform, and Load (ETL)
- ETL systems are commonly used to integrate data from multiple applications, typically developed and supported by different vendors or hosted on separate computer hardware
- It is a conceptual term that creates the fundamentals of Big Data

ETL

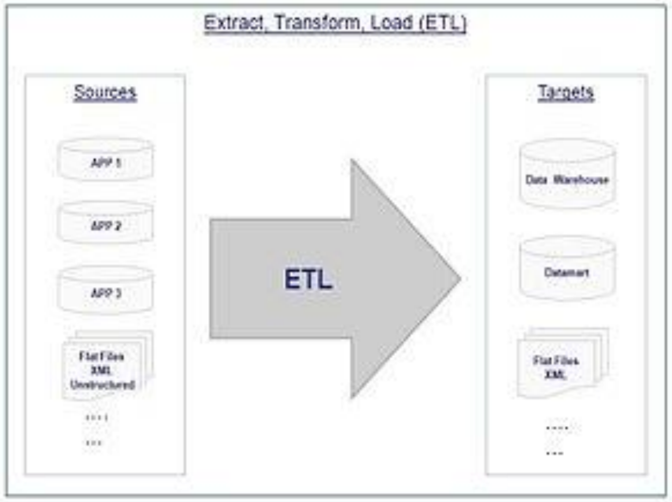


Extraction Transformation Load (ETL) Architecture Pattern

Description

Extraction, Transformation and Load (ETL) is an industry standard term used to represent the data movement and transformation processes.

ETL is an essential component used to load the data into data warehouses (DWH), operational data stores (ODS) and datamarts (DM) from the source systems. ETL processes are also widely used in data integration, data migration and master data management (MDM) initiatives.



Architectural Context

Supported Use Cases

- Bulk data integration
- Flat-file based and hierarchical transformations
- High scale, batch-oriented data delivery

Examples

- Financial ODS

Extract

- extracting the data from the source systems
- Most challenging as it affects all subsequent stages
- Usually requires consolidation from different sources… Duh!
- Sources could be both structured or non-structured
  - – List of data structures

Extract (Cont'd)

- Could be done in bluk (rarely)
- Or, Streaming from data sources (most common use)
- The goal of the extraction phase is to convert the data into a single format appropriate for transformation processing
- It might involve parsing to check if data is ready for the subsequent stages

Transform

- Different sources → different formats
- Transform stage applies several “rules” to extracted data to make it ready for the last stage
  - The level of “transformation” depends on the source of the data. Little or no manipulation of the data is requires in some cases

Transform (cont'd)

- Examples:
  - Filtering columns of data (depending on the need)
  - Translation of Codes (e.g. Male →M)
  - Calculation (e.g. Total price = unit price*qty)
  - Sorting
  - Joining
  - Aggregation (e.g. count)
  - Splitting columns into several columns
  - Etc.

Load

- As the name means.. Loading the transformed data into a Data warehouse
- Huge amount of data is reduced into small amount of data that could be interpreted easily

Big Data Analysis

- ETL concepts is utilized to analyze Big Data
- The most common used way is MapReduce
- MapReduce = Divide and Conqure
  - Parallelism
  - Reliability
  - scalability

MapReduce

- It is a programming model for processing large data sets in a parallel, distributed way
- Originated from Google to handle their search queries by querying Google File System (GFS)
- It has been written by many libraries
  - Most famous example is Hadoop (reverse engineered GFS, but it’s gotten better)

MapReduce Overview

- How does it solve our previously mentioned problems?
  - MapReduce is highly scalable and can be used across many computers.
  - Many small machines can be used to process jobs that normally could not be processed by a large machine.

Map Abstraction

- Inputs a key/value pair
  - Key is a reference to the input value
  - Value is the data set on which to operate
- Evaluation
  - Function defined by user
  - Applies to every value in value input
    - Might need to parse input
- Produces a new list of key/value pairs
  - Can be different type from input pair

Map Example

```
def map(key, value):
    list = []
    for x in value:
        if test:
            list.append( (key, x) )
    return list
```

Reduce Abstraction

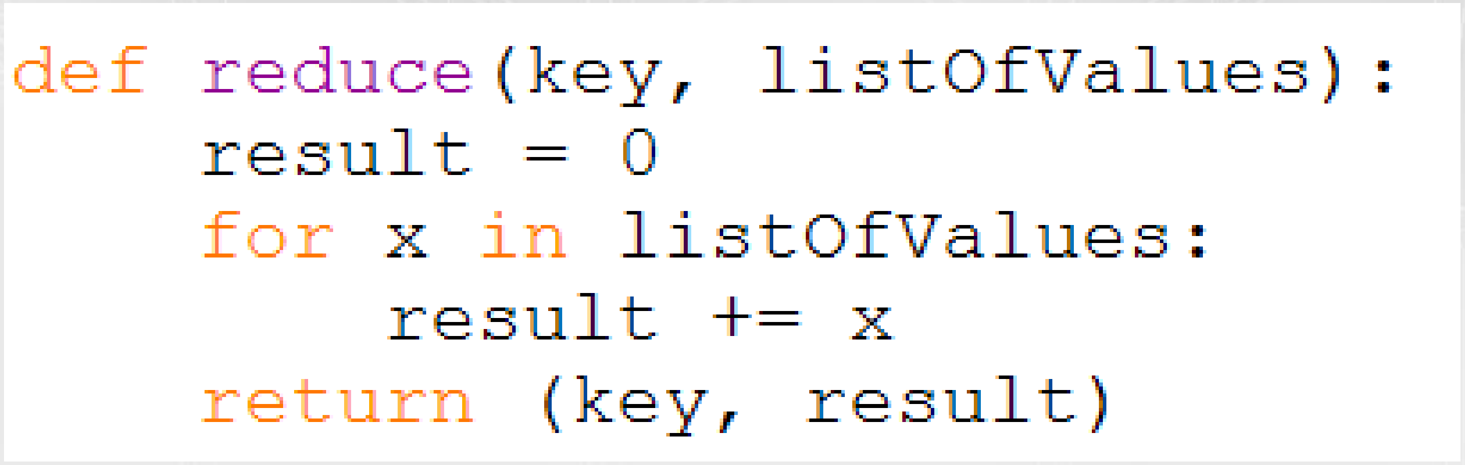
- Starts with intermediate Key / Value pairs
- Ends with finalized Key / Value pairs

- Starting pairs are sorted by key
- Iterator supplies the values for a given key to the Reduce function.

Reduce Abstraction

- Typically a function that:
  - Starts with a large number of key/value pairs
    - One key/value for each word in all files being greped (including multiple entries for the same word)
  - Ends with very few key/value pairs
    - One key/value for each unique word across all the files with the number of instances summed into this entry
- Broken up so a given worker works with input of the same key.

Reduce Example



```
def reduce(key, listOfValues):
    result = 0
    for x in listOfValues:
        result += x
    return (key, result)
```

How Map and Reduce Work Together

Map returns information

Reduces accepts information

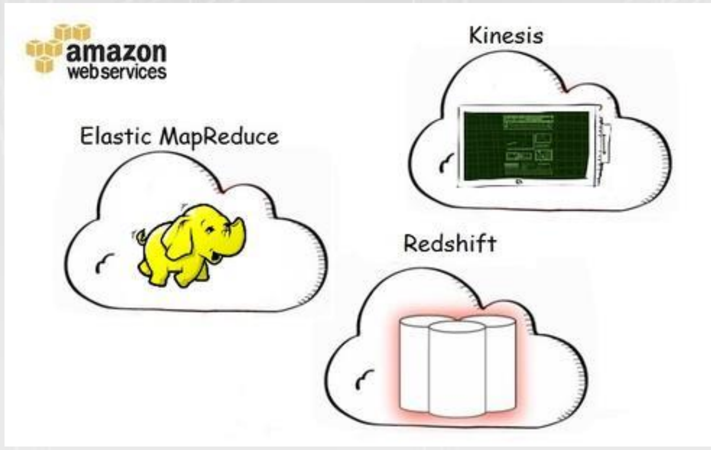
Reduce applies a user defined function to reduce the amount of data

Case Studies

- American Express and Airline loyality
- http://www.ibm.com/analytics/us/en/case-studies/
- California Fights drought with Big Data
- How analytics helped Ford turn its fortunes

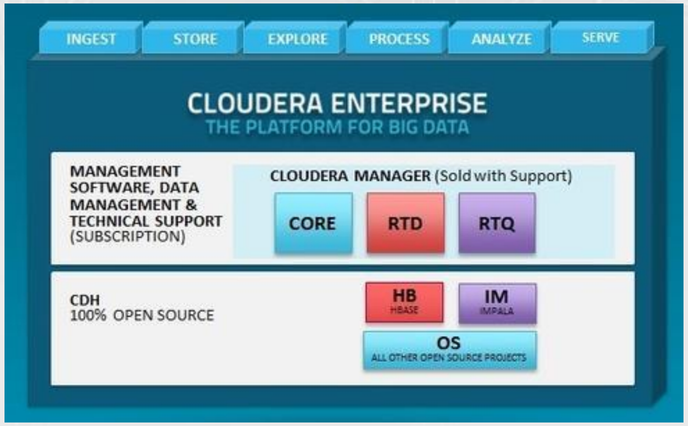
Big Data Analysis Tools

- Amazon Web Services (AWS)
- Elastic Compute Cloud (EC2) and Simple Storage Service (S3) storage infrastructure
- Amazon launched its Hadoop-based Elastic MapReduce service way back in 2009
- In 2013, AWS added the Redshift Data Warehousing service



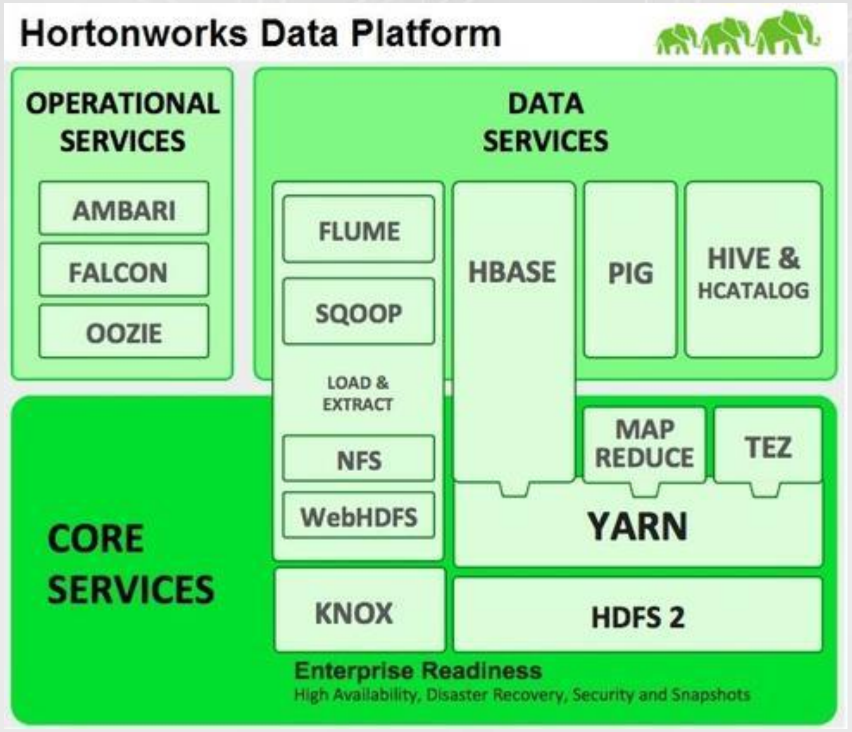
Big Data Analysis Tools (Cont’d)

- Cloudera
- Biggest distributor of Hadoop


INGEST
STORE
EXPLORE
PROCESS
ANALYZE
SERVE
CLOUDERA ENTERPRISE
THE PLATFORM FOR BIG DATA
MANAGEMENT SOFTWARE, DATA MANAGEMENT & TECHNICAL SUPPORT (SUBSCRIPTION)
CLOUDERA MANAGER (Sold with Support)
CORE
RTD
RTQ
CDH
100% OPEN SOURCE
HB
HBASE
IM
IMPALA
OS
ALL OTHER OPEN SOURCE PROJECTS

HortonWorks

- Hortonworks
- Hadoop



Assignments

- Dean, J. and Ghemawat, S. 2008. MapReduce: simplified data processing on large clusters. Communication of ACM 51, 1 (Jan. 2008), 107-113.