TEXAS ADVANCED COMPUTING CENTER
WWW.TACC.UTEXAS.EDU

TEXAS
The University of Texas at Austin

DATA MANIPULATION

John Gentle
Data Management & Collections Group
Texas Advanced Computing Center
jgentle@tacc.utexas.edu

OVERVIEW

Data is fundamental to all modern research and computation. It is the "currency" of the digital economy that envelops our modern society.

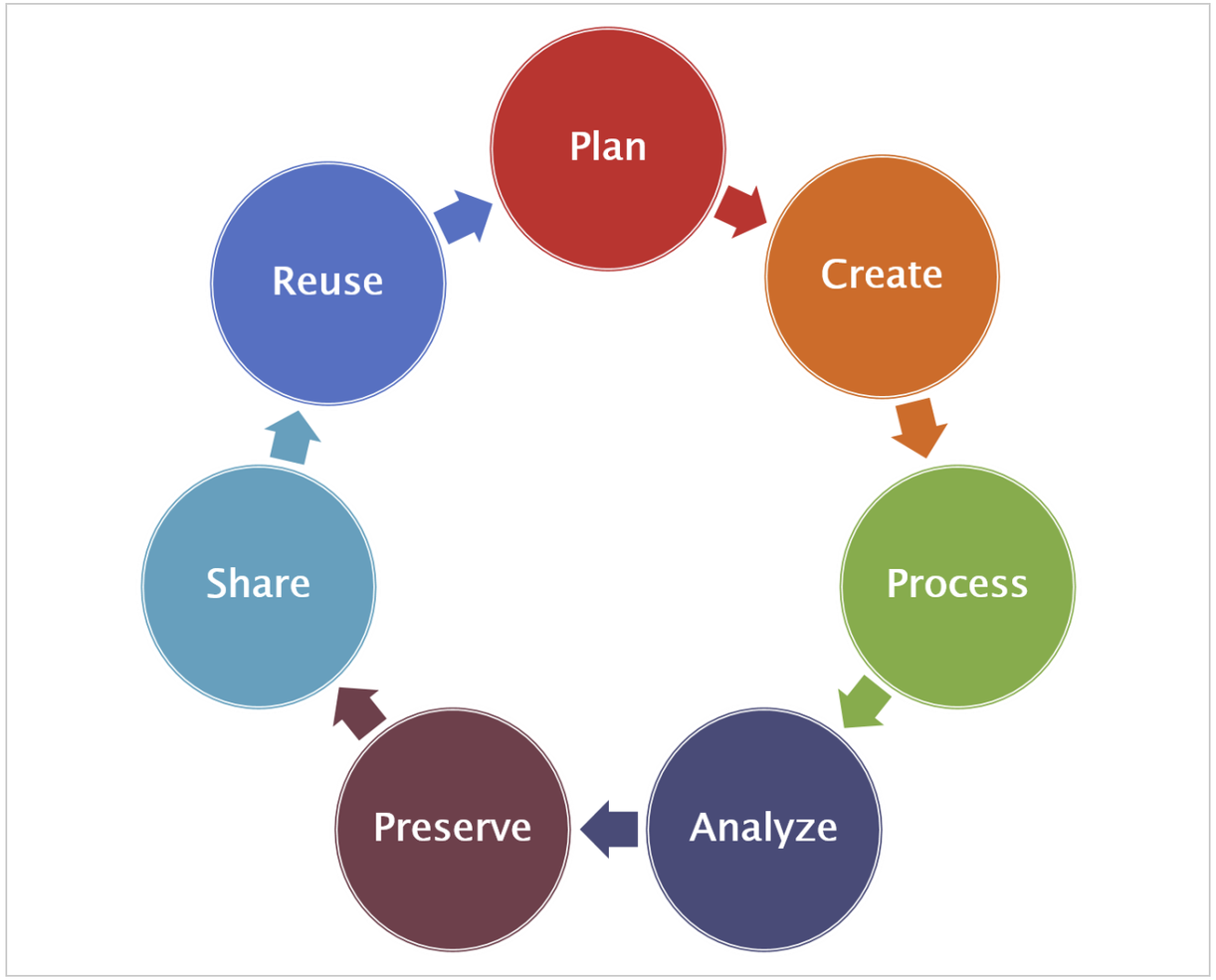
The formal processes for collecting, analyzing, storing and distributing this data are called *Data Management*.

These processes facilitate the discovery of useful information in the data, known as *Data Mining*, and coordinate the workflows and systems used in the extraction process, which are often implemented as a reproducible and reusable *Data Pipeline*.

These procedures also govern the data's structure and properties, called *Metadata*, as well as the monitoring and logging of the history of the data as it moves through the data pipeline, used in *Provenance*.

In order to fully grok the relationship between these components, it is important to start with an understanding of the *Data Lifecycle*.

DATA LIFECYCLE



Note that *Create* could be more accurately described as *Acquire*.

DLC PHASE 1: PLAN

A documented sequence of intended actions to identify and secure resources and gather, maintain, secure, and utilize data holdings comprise a *Data Management Plan*.

This also includes the procurement of funding and the identification of technical and staff resources for full lifecycle data management.

Once the data needs are determined, a system to store and manipulate the data can then be identified and developed.

DLC PHASE 2: CREATE (ACQUIRE)

Acquisition involves collecting or adding to the data holdings.

There are four methods of acquiring data:

- Collecting new data
- Converting/transforming legacy data
- Sharing/exchanging data
- Purchasing data

We will explore this phase in greater detail later.

DLC PHASE 3: PROCESS

Processing denotes actions or steps performed on data to verify, organize, transform, integrate, and extract data in an appropriate output form for subsequent use.

This includes data files and content organization, and data synthesis or integration, format transformations, and may include calibration activities (of sensors and other field and laboratory instrumentation).

Both raw and processed data require complete metadata to ensure that results can be duplicated.

Methods of processing must be rigorously documented to ensure the utility and integrity of the data.

DLC PHASE 4: ANALYZE

Analysis involves actions and methods performed on data that help describe facts, detect patterns, develop explanations, and test hypotheses.

- Data quality assurance
- Statistical data analysis
- Modeling
- Interpretation of analysis results

DLC PHASE 5: PRESERVE

Preservation involves actions and procedures to keep data for some period of time and/or to set data aside for future use, and includes data archiving and/or data submission to a data repository.

A primary goal for any data scientist or researcher is to preserve well-organized and documented datasets that support research interpretations that can be re-used by others.

All research publications should be supported by associated, accessible datasets (preferably with a minted DOI and linked data).

DLC PHASE 6: PUBLISH (SHARE)

The ability to prepare and issue, or disseminate, quality data to the public and to other researchers or data scientists is an important part of the lifecycle process.

The data should be medium- and agent-independent, with an understanding that transfer may occur via automated or non-automated mechanisms.

We need to ensure that data are shared, but (ideally) with controls to protect proprietary and pre-decisional data and the integrity of the data itself.

Data sharing also requires complete metadata to be useful to those who are receiving the data.

DLC PHASE 7: RE-USE

While there is no way to directly control the re-use of data, it is one of the objectives of proper data management to produce and maintain data that can be discovered and utilized by other researchers during their planning and acquisition phases.

This is where the data lifecycle begins anew with existing data being folded into the data pipeline alongside newly created or collected data.

METADATA & DOCUMENTATION

Throughout the data lifecycle process, documentation must be updated to reflect actions taken upon the data. This includes acquisition, processing, and analysis, but may touch upon any stage of the lifecycle.

Updated and complete metadata are critical to maintaining data quality. The key distinction between metadata and documentation is that metadata, in the standard sense of "data about data," formally describes various key attributes of each data element or collection of elements, while documentation makes reference to data in the context of their use in specific systems, applications, settings.

Documentation also includes ancillary materials (e.g., field notes) from which metadata can be derived. In the former sense, it's "all about the data;" in the latter, it's "all about the use."

PROVENANCE

Provenance of a resource is a record that describes entities and processes involved in producing and delivering or otherwise influencing that resource. Provenance provides a critical foundation for assessing authenticity, enabling trust, and allowing reproducibility. Provenance assertions are a form of contextual metadata and can themselves become important records with their own provenance.

Metadata is used to represent properties of objects. Many of those properties have to do with provenance. Descriptive metadata only becomes part of provenance when one also specifies its relationship to deriving an object. *Provenance is often represented as metadata, but not all metadata is necessarily provenance*.

Trust is derived from provenance information, and typically is a subjective judgment that depends on context and use. Provenance is effectively a platform for trust algorithms and approaches on the web.

Provenance information may be used for authentication purposes as well (ex. signature verification).

QUALITY ASSURANCE & MANAGEMENT

Protocols and methods must be employed to ensure that data are properly collected, handled, processed, used, and maintained at all stages of the scientific data lifecycle.

This is commonly referred to as "QA/QC" or *Quality Assurance/Quality Control*.

QA focuses on building-in quality while QC focuses on testing for quality.

- QA makes sure you are doing the right things, the right way (e.g., prevent defects).
- QC makes sure the results of what you've done are what you expected (e.g., detecting defects).

BACKUPS & ARCHIVING

Steps must be taken to protect data from accidental data loss, corruption, and unauthorized access.

This includes routinely making a *backup* (additional copies of data files or databases that can be used to restore the original data) or an *archive* (for recovery of earlier instances of the data and long-term storage for infrequently used data).

DATA MANAGEMENT

Data Management is the development and execution of architectures, policies, practices and procedures in order to manage the information (data) lifecycle needs in an effective manner.

The concept of "Data Management" arose in the 1980s as technology moved from sequential processing (first cards, then tape) to random access processing.

As applications moved into real-time, interactive applications, it became obvious to most practitioners that both data and process management were important.

If the data is not well defined, the data will be mis-used in applications.

If the process isn't well defined, it is impossible to meet user needs.

DATA MINING

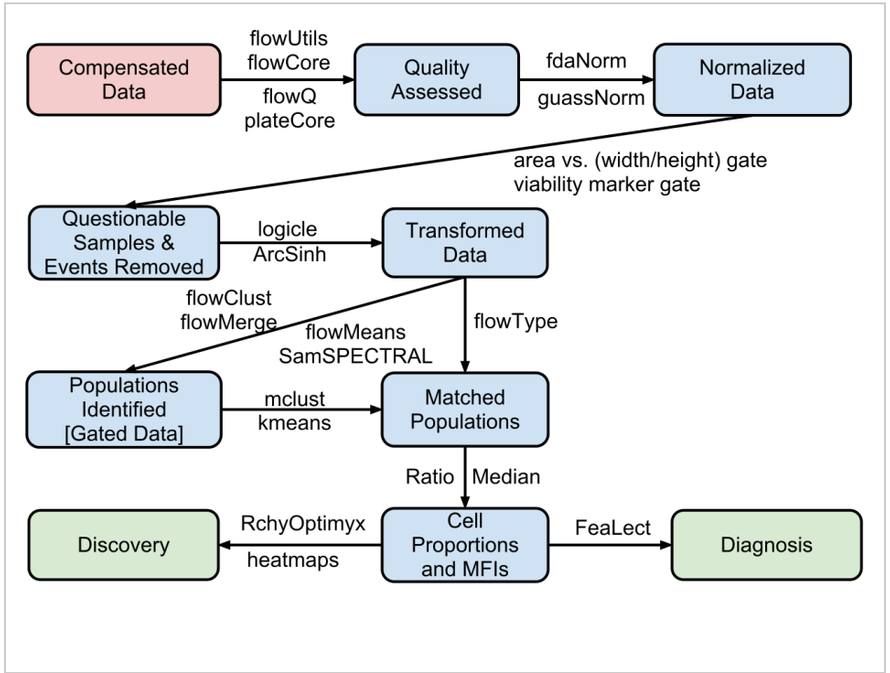
The manual extraction of patterns from data has occurred for centuries. The proliferation and increasing power of computers has dramatically increased data collection, storage, and manipulation ability.

- Bayes' theorem (1700s)
- Regression analysis (1800s).
- Automated data processing, neural networks, cluster analysis, genetic algorithms (1950s)
- Decision trees and decision rules (1960s)
- Support vector machines (1990s)

Data mining is the process of applying these methods with the intention of uncovering hidden patterns in large data sets. It bridges the gap from applied statistics and artificial intelligence (which usually provide the mathematical background) to database management by exploiting the way data is stored and indexed in databases to execute the actual learning and discovery algorithms more efficiently, allowing such methods to be applied to ever larger data sets.

The term is a misnomer, because the goal is the extraction of patterns and knowledge from large amounts of data, not the extraction (mining) of data itself.

DATA PIPELINE



A data pipeline is a set of data processing elements connected in series, where the output of one element is the input of the next one. The elements of a pipeline are often executed in parallel or in time-sliced fashion; in that case, some amount of buffer storage is often inserted between elements.

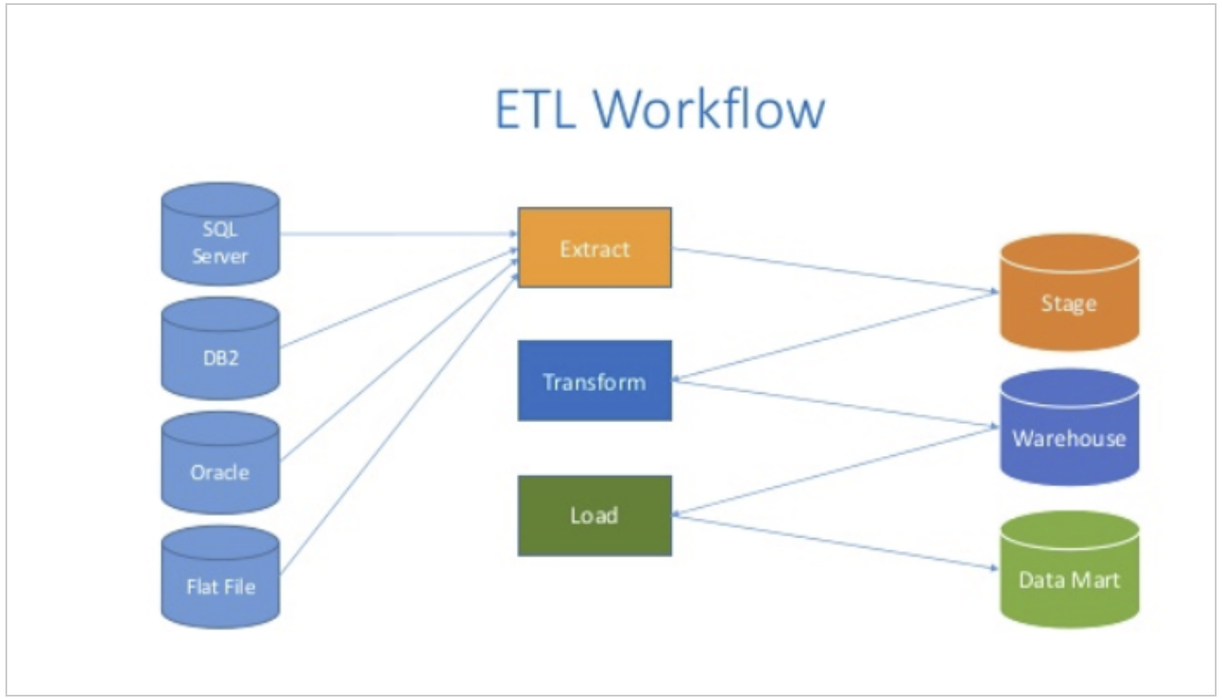
A data pipeline encompasses the complete journey of data through the data lifecycle.

EXTRACT, TRANSFORM & LOAD (ETL)

Extract-Transform-Load (ETL) refers to the process of transferring data in bulk from one system to another. ETL systems extract and transform large volumes of source data into a new dataset, and then load that data into a new data store, for subsequent querying.

Compared to transactional systems, ETL systems do not have an ongoing ‘live’ state. Instead they typically operate on a data load cycle that might be daily, weekly, or monthly, usually containing a ‘critical’ period where data must be loaded within a specific timeframe in order to meet internal or external deadlines.

ETL WORKFLOW



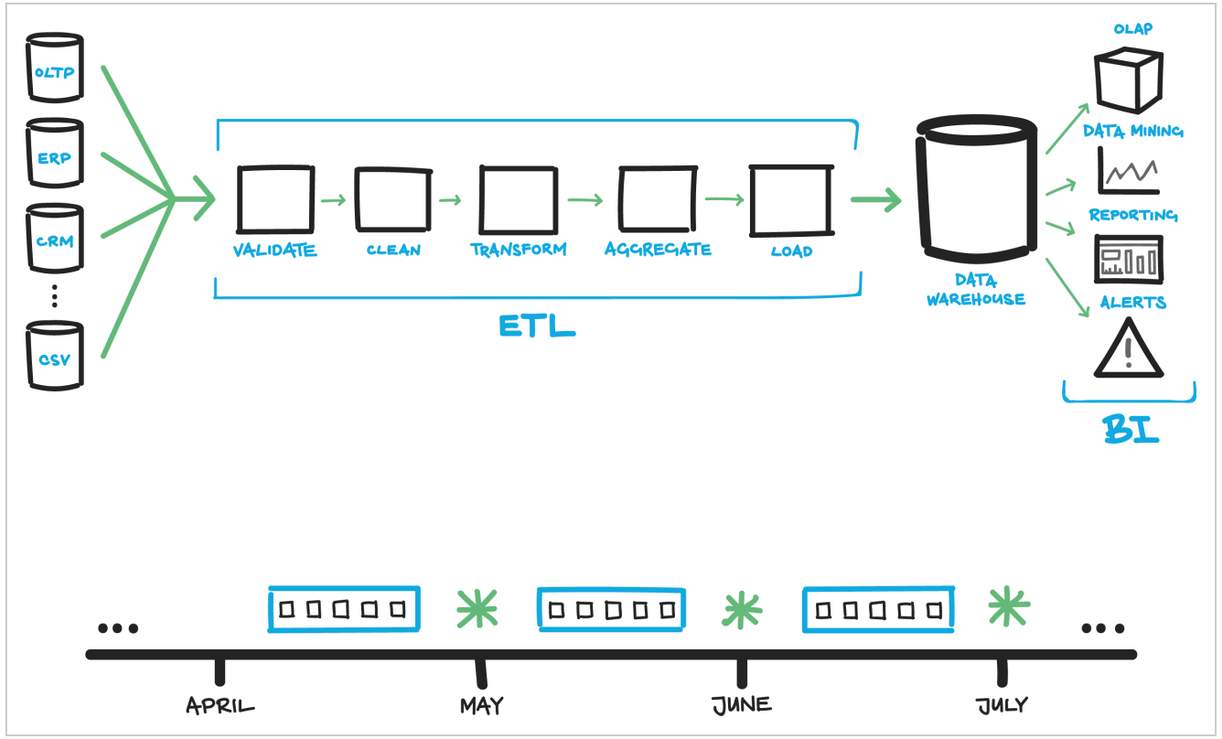
ETL sounds like three distinct phases but in fact the phases overlap and it’s not necessarily a serial process. The ‘users’ of ETL systems are always other software systems or people involved in data analysis, rather than interactive customers or end-users.

ETL == E(VC)T(A)L

The ETL Workflow actually consists of six distinct phases:

Extract . Validate . Clean . Transform . Aggregate . Load

These phases are not necessarily sequential in execution.



ETL: EXTRACT

The Extract step covers the data extraction from the source system and makes it accessible for further processing.

The main objective of the extract step is to retrieve all the required data from the source system with as little resources as possible.

The extract step should be designed in a way that it does not negatively affect the source system in terms or performance, response time or any kind of locking.

ETL KJB EXAMPLE SCRIPT

KJB files are jobs (in the Pentaho Kettle data integration pipeline) that wrap flow control around transformations.

```
<?xml version="1.0" encoding="UTF-8"?>
<job>
  <name>master_job</name>
    <description/>
    <extended_description/>
    <job_version/>
  <directory>&#x2f;</directory>
  <created_user>-</created_user>
  <created_date>2015&#x2f;07&#x2f;31 13&#x3a;39&#x3a;30.873</created_date>
  <modified_user>-</modified_user>
  <modified_date>2015&#x2f;07&#x2f;31 13&#x3a;39&#x3a;30.873</modified_date>
    <parameters>
    </parameters>
  <connection>
    <name>Biotica</name>
    <server>localhost</server>
    <type>POSTGRESQL</type>
    <access>Native</access>
    <database>biotica_large</database>
    <port>5432</port>
    <username>turban</username>
    <password>Encrypted 2be98afc86aa7f2dcf34e8671dcd7fe89</password>
    <servername/>
    <data_tablespace/>
    <index_tablespace/>
    <attributes>
      <attribute><code>FORCE_IDENTIFIERS_TO_LOWERCASE</code><attribute>N</attribute></attribute>
      <attribute><code>FORCE_IDENTIFIERS_TO_UPPERCASE</code><attribute>N</attribute></attribute>
      <attribute><code>IS_CLUSTERED</code><attribute>N</attribute></attribute>
      <attribute><code>PORT_NUMBER</code><attribute>5432</attribute></attribute>
      <attribute><code>PRESERVE_RESERVED_WORD_CASE</code><attribute>N</attribute></attribute>
```

ETL KTR EXAMPLE SCRIPT

KTR files are transformations (in the Pentaho Kettle data integration pipeline) that act on the data.

```
<?xml version="1.0" encoding="UTF-8"?>
<transformation>
  <info>
    <name>migrate_institutions</name>
    <description/>
    <extended_description/>
    <trans_version/>
    <trans_type>Normal</trans_type>
    <trans_status>0</trans_status>
    <directory>&#x2f;</directory>
    <parameters>
    </parameters>
    <log>
<trans-log-table><connection/>
<schema/>
<table/>
<size_limit_lines/>
<interval/>
<timeout_days/>
<field><id>ID_BATCH</id><enabled>Y</enabled><name>ID_BATCH</name></field><field><id>CHANNEL_ID</id><enabled>Y</enabled><name>CHANNEL_ID<
<perf-log-table><connection/>
<schema/>
<table/>
<interval/>
<timeout_days/>
<field><id>ID_BATCH</id><enabled>Y</enabled><name>ID_BATCH</name></field><field><id>SEQ_NR</id><enabled>Y</enabled><name>SEQ_NR</name></
<channel-log-table><connection/>
<schema/>
<table/>
<timeout_days/>
<field><id>ID_BATCH</id><enabled>Y</enabled><name>ID_BATCH</name></field><field><id>CHANNEL_ID</id><enabled>Y</enabled><name>CHANNEL_ID<
```

ETL: VALIDATE

Validation is the process of assessing how well your mining models perform against real data. Data validation is fundamental to ensuring you have a reliable and reproducible ETL workflow.

There are many approaches for assessing the quality and characteristics of a data mining model. Common validation criteria for data include *accuracy, reliability, and usefulness*.

Accuracy is a measure of how well the model correlates an outcome with the attributes in the data that has been provided. There are various measures of accuracy, but all measures of accuracy are dependent on the data that is used.

Reliability assesses the way that a data mining model performs on different data sets. A data mining model is reliable if it generates the same type of predictions or finds the same general kinds of patterns regardless of the test data that is supplied.

Usefulness includes various metrics that tell you whether the model provides useful information. You might also find that a model that appears successful in fact is meaningless, because it is based on cross-correlations in the data.

Validation is commonly run after extraction (*post-extraction validation*), before load (*pre-load validation*) or after load (*post-load validation*).

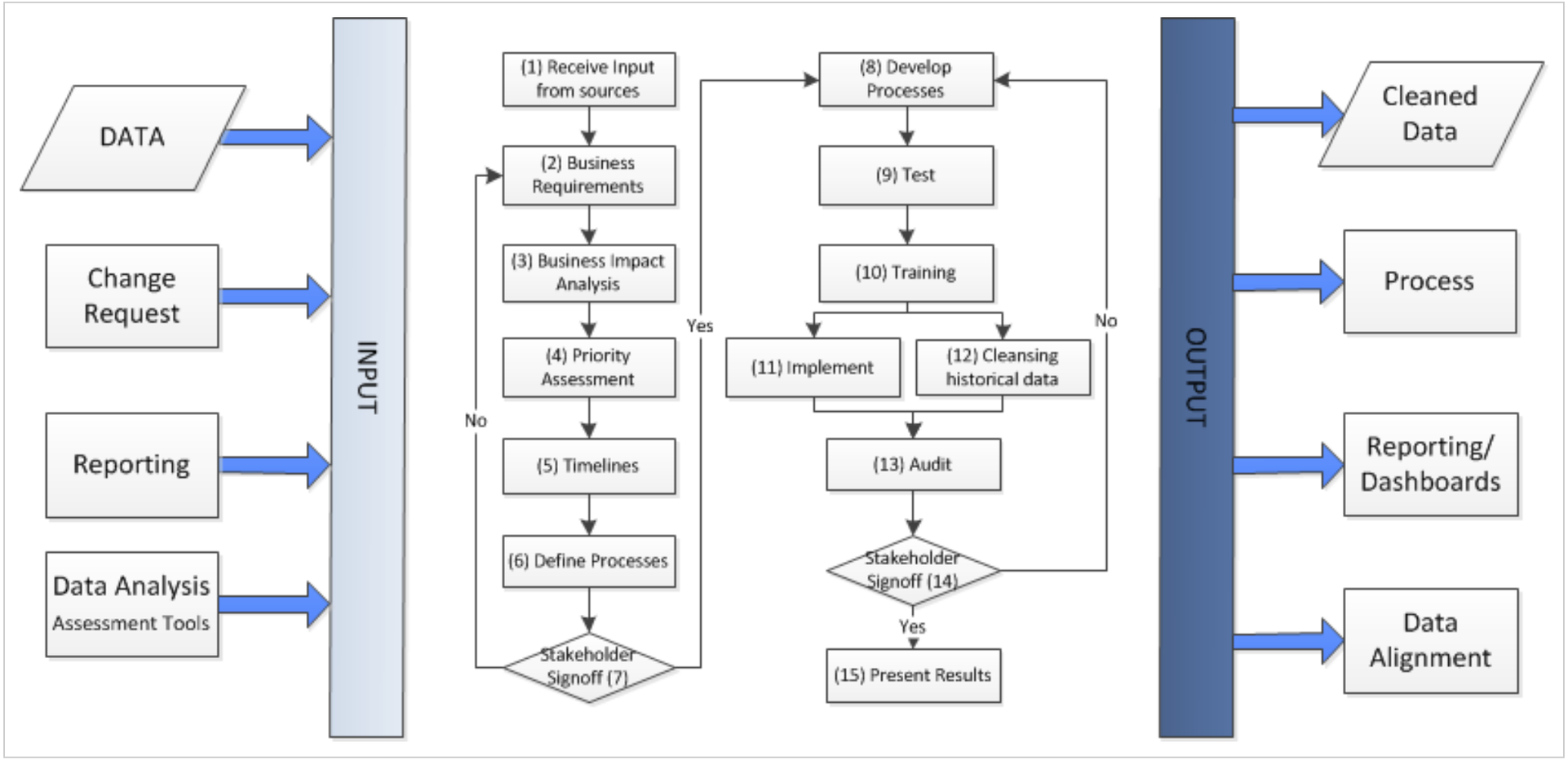
ETL: CLEAN

The cleaning step is one of the most important as it ensures the quality of the data in the data warehouse.

Cleaning should perform basic data unification rules such as:

- Identify and remove duplicates
- Convert numbers to a consistent representation
- Convert dates and times to a consistent representation
- Remove case sensitivity, or make it consistent throughout
- Normalize spelling for a given dictionary or term
- Making identifiers unique
- Convert null values into a standardized value
- Validate address fields, convert them into proper naming, e.g. Street/St/St./Str./Str
- Validate address fields against each other (State/Country, City/State, City/ZIP code, City/Street)

DATA CLEANSING DIAGRAM



Open Data Quality Cleansing Framework Diagram.

ETL: TRANSFORM

The transform step applies a set of rules to transform the data from the source to the target. This includes converting any measured data to the same dimension (i.e. conformed dimension) using the same units so that they can later be joined.

The transformation step also requires joining data from several sources, generating aggregates, generating surrogate keys, sorting, deriving new calculated values, and applying advanced validation rules.

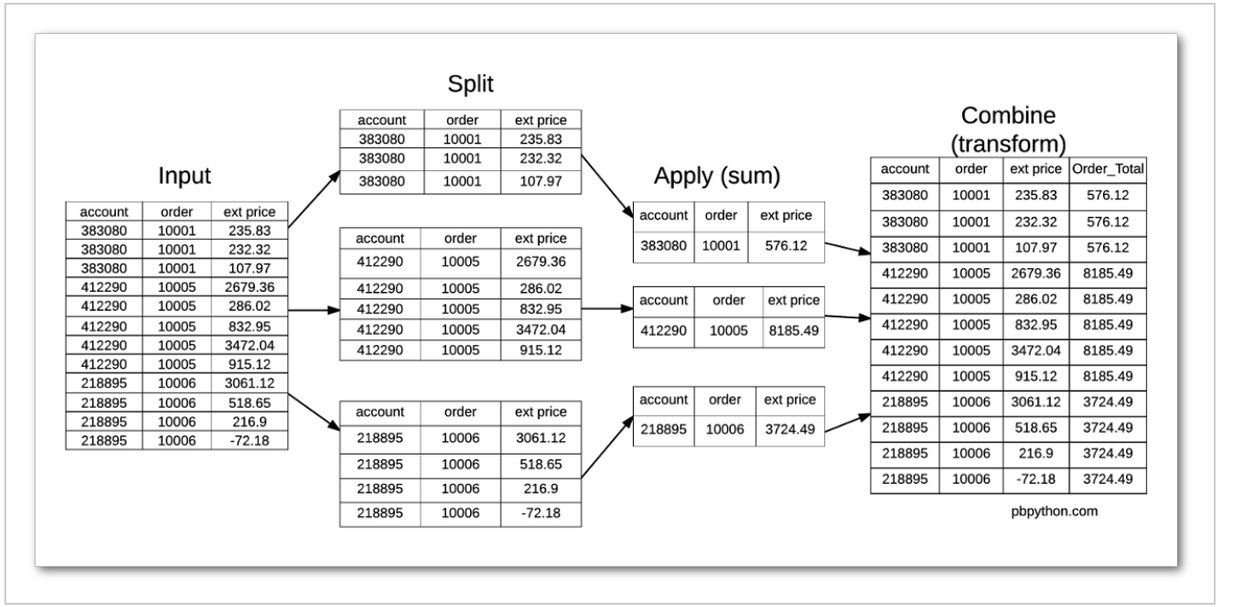
ETL TRANSFORM SCRIPT (PYTHON 2009)

```
# 7/28/2009 - Ryan Robitaille (http://ryrobes.com/)
# An easy Python DTS / ETL "package" script | Reading data from one source (Oracle) and writing it to another (MS SQL or Oracle)
# Please let me comments / fixes / changes on this post (http://ryrobes.com/featured-articles/using-a-simple-python-script-for-end-to-en

# DataSource:    Random Data generated from Oracle's DBMS_RANDOM.x functions (just as an example)

# Destination:    Very basic table on MSSQL and Oracle (see below - create statement should work on both platforms)
#        CREATE TABLE test --just a test destination table
#        (field1 VARCHAR(50), field2 VARCHAR(50),
#        field3 INT, field4 VARCHAR(50))


# First we need to import all the modules we will be using - for now its only 3 easy ones
import string, pymssql, cx_Oracle

## Define Oracle connection - format ("username/password@TNSNAME")
ora_conn = cx_Oracle.connect("xxxx/xxxx@XXXXX")

## Define the Oracle cursor objects
ora_cursor = ora_conn.cursor()        #Allocate a cursor to that particular database connection

## Define the MSSQL server connection - format is self-explanitory
mssql_conn = pymssql.connect(host='xxxxx', user='rrobitaille', password='xxxxx', database='XXXXXX')

## Define the MSSQL cursor objects
mssql_cursor = mssql_conn.cursor()    #Allocate a cursor to that particular database connection


## Truncate our destination tables
ora_cursor.execute("truncate table test")
```

ETL TRANSFORM SCRIPT (PYTHON 2017)

Using the Pandas package.

```
import pandas as pd
df = pd.read_excel("sales_transactions.xlsx")
df.groupby('order')["ext price"].transform('sum')
df["Order_Total"] = df.groupby('order')["ext price"].transform('sum')
df["Percent_of_Order"] = df["ext price"] / df["Order_Total"]
# Or Combine them.
# df["Percent_of_Order"] = df["ext price"] / df.groupby('order')["ext price"].transform('sum')
```



account	order	ext price
383080	10001	235.83
383080	10001	232.32
383080	10001	107.97
412290	10005	2679.36
412290	10005	286.02
412290	10005	832.95
412290	10005	3472.04
412290	10005	915.12
218895	10006	3061.12
218895	10006	518.65
218895	10006	216.9
218895	10006	-72.18

account	order	ext price
383080	10001	576.12
412290	10005	8185.49
218895	10006	3724.49

account	order	ext price	Order_Total
383080	10001	235.83	576.12
383080	10001	232.32	576.12
383080	10001	107.97	576.12
412290	10005	2679.36	8185.49
412290	10005	286.02	8185.49
412290	10005	832.95	8185.49
412290	10005	3472.04	8185.49
412290	10005	915.12	8185.49
218895	10006	3061.12	3724.49
218895	10006	518.65	3724.49
218895	10006	216.9	3724.49
218895	10006	-72.18	3724.49

ETL: AGGREGATE

The single most dramatic way to affect performance in a large data warehouse is to provide a proper set of aggregate (summary) records that coexist with the primary base records. Aggregates can have a very significant effect on performance, in some cases speeding queries by a factor of one hundred or even one thousand. No other means exist to harvest such spectacular gains. ***~ Ralph Kimball***

An aggregate (in its simplest form) is a summary table that can be derived by performing a Group by SQL query.

```
SELECT <column_name(s)>
FROM <table_name>
WHERE <condition>
GROUP BY <column_name(s)>
ORDER BY <column_name(s)>;
```

A more common use of aggregates is to take a dimension and change the granularity of this dimension.

Aggregates are sometimes referred to as pre-calculated summary data, since aggregations are usually precomputed, partially summarized data, that are stored in new aggregated tables.

ETL: LOAD

During the load step, it is necessary to ensure that the load is performed correctly and with as little resources as possible.

The target of the Load process is often a database or data warehouse.

In order to make the load process efficient, it is helpful to disable any constraints and indexes before the load and enable them back only after the load completes.

The referential integrity needs to be maintained by the ETL tools to ensure consistency.

ETL: STAGING

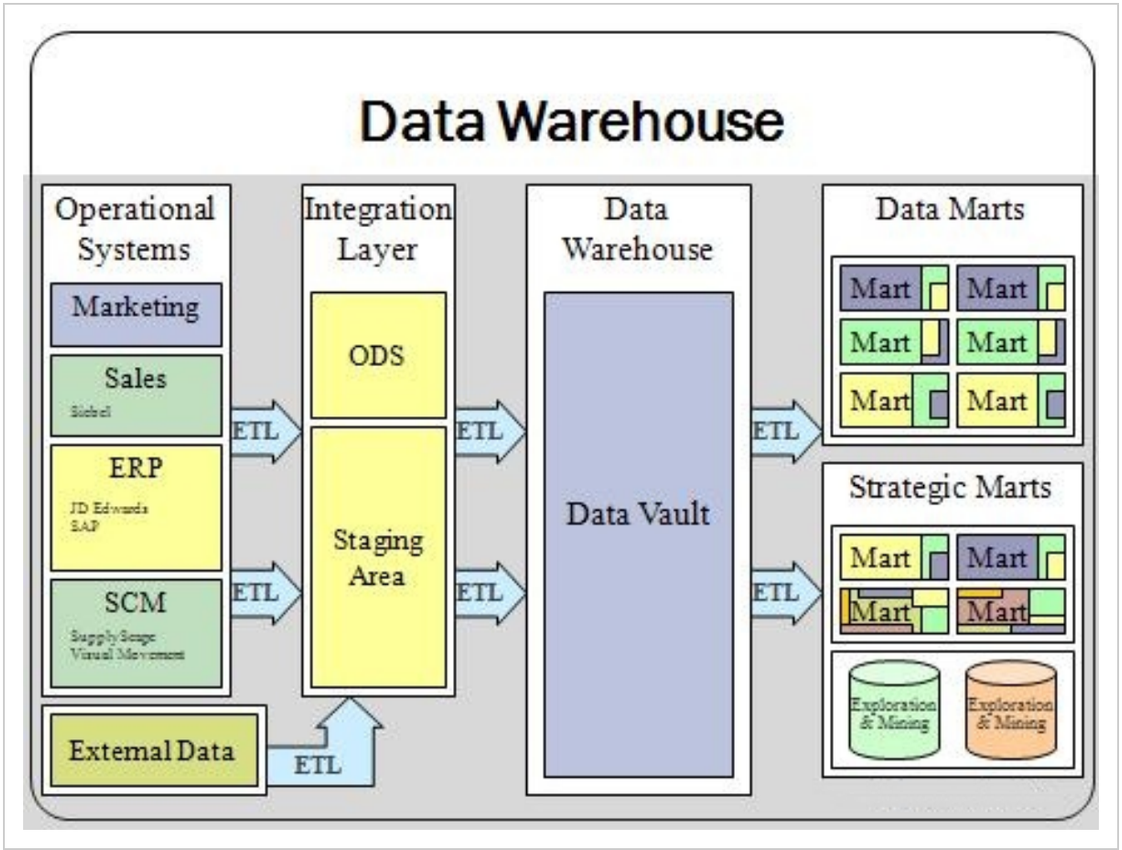
It should be possible to restart some of the ETL phases independently from the others. If the transformation step fails, it should not be necessary to restart the Extract step.

We can ensure this by implementing proper *Staging*.

Staging means that the data is simply dumped to a location called the *Staging Area* so that it can then be read by the next processing phase.

The staging area is also used during ETL process to store intermediate results of processing. The staging area may therefore contain incomplete or in-the-middle-of-the-processing data and should be accessed by the load ETL process only. Staging should never be available to anyone beside the ETL system as it is not intended for data presentation.

ETL PIPELINE DIAGRAM



Logical architecture of a Data Warehouse data pipeline.

ETL VS ELT

Instead of transforming the data before it’s written, ELT leverages the target system to do the transformation. The data is copied to the target and then transformed in place.

ELT makes sense when the target is a high-end data engine, such as a data appliance, Hadoop cluster, or cloud installation.

ELT has no transformation engine – the work is done by the target system.

The ETL approach can provide drastically better performance under certain scenarios.

The training and development costs of ETL need to be weighed against the need for better performance.

Additionally, if you don’t have a target system powerful enough for ELT, ETL may be more economical as well.

ETLT

Data warehouse loader utilities have become so fast that it is practical to move data into the data warehouse at an early point in the data transformation process. Consequently, the performance bottleneck is shifting from the data warehouse to the ETL servers.

The idea behind *ETLT* (extract, format transformation, load, referential transformation) is to combine ETL and ELT to exploit the strengths of each approach to achieve optimum performance and scalability. The goal is to move certain transformation processing onto the data warehouse platform to take advantage of the inherent parallelism and the relational integration of data.

ETLT PHASES

ET Phase: In this arrangement, the ETL tool is responsible for extracting and consolidating data from the source systems, performing scalar data cleansing, and formatting the change data to look much like the target tables. At this point the "ET" phase data is loaded into the staging area of the data warehouse appliance. The loading operation is often performed by an appliance high speed loader.

LT Phase: Once that data is loaded into the appliance, the remaining "LT" transformation is completed using SQL-based set level operations. The *ETLT Framework Manager* handles end-to-end control including task dependencies, data archiving, restart/recovery, error-handling and event logging.

ETL: TOOLS

Commercial ETL Tools

- Ab Initio
- IBM InfoSphere DataStage
- Informatica
- Oracle Data Integrator
- SAP BI

Open Source ETL Tools

- Apatar
- CloverETL
- Pentaho
  - Kettle
  - GeoKettle
- Talend

RESOURCES

USGS: Data Lifecycle Overview

Data Lifecycle in a Big Data World

Database Lifecycle Management for ETL Systems

ORACLE Data Mining Concepts

Data Warehouse Tutorial Series (Youtube)

Thearling.com

Data Warehouse Info

RESOURCES (CONTINUED)

ETL Process

ETL Testing & Validation

5 Tips for Cleaning Dirty Data

ETL vs ELT: What's the Big Difference?

ETL and ELT Join Forces

Using a simple Python script for End-to-End Data Transformation and ETL (Part 1)

ETL Transformation Script Using Pandas

WC3 Wiki: Provenance

PRESENTATION SOURCE

The presentation requires NodeJS and Bower be installed on your system.

NodeJS: https://nodejs.org/en/

Bower.io (requires node):

```
npm install -g bower
```

Once these are installed...

Download the presentation here: https://github.com/jgentle/data-manipulation

Or clone it locally:

```
git clone https://github.com/jgentle/data-manipulation.git
```

Then...

```
cd data-manipulation
npm install
bower install
grunt serve
```

Enjoy the presentation!

PRINTING THIS PRESENTATION

Printing functionality requires the presentation to be running on a web server (either locally or online).

You can then print this presentation by appending `?print-pdf` to the end of the URL:

```
# localhost.
http://localhost:9000/?pdf-print/

# live online.
http://mydomain.com/my-presentation/?pdf-print/
```

Follow it then enter the standard CTRL/CMD+P (or use browser menu > Print...).

The content will look weird until you print preview... but it will print OKAY!

For convenience I have included the localhost link below.

Print Ready Version

THANK YOU