BDT303

Construct Your ETL with AWS Data Pipeline, Amazon EMR, and Amazon Redshift

Roy Ben-Alta (Big Data Analytics Practice Lead, AWS)
Thomas Barthelemy (Software Engineer, Coursera)

November 13, 2014 | Las Vegas | NV

AWS Big Data Portfolio

Collect


AWS Direct Connect


AWS Import/Export


Amazon Kinesis

Store


Amazon S3


DynamoDB


Amazon Glacier

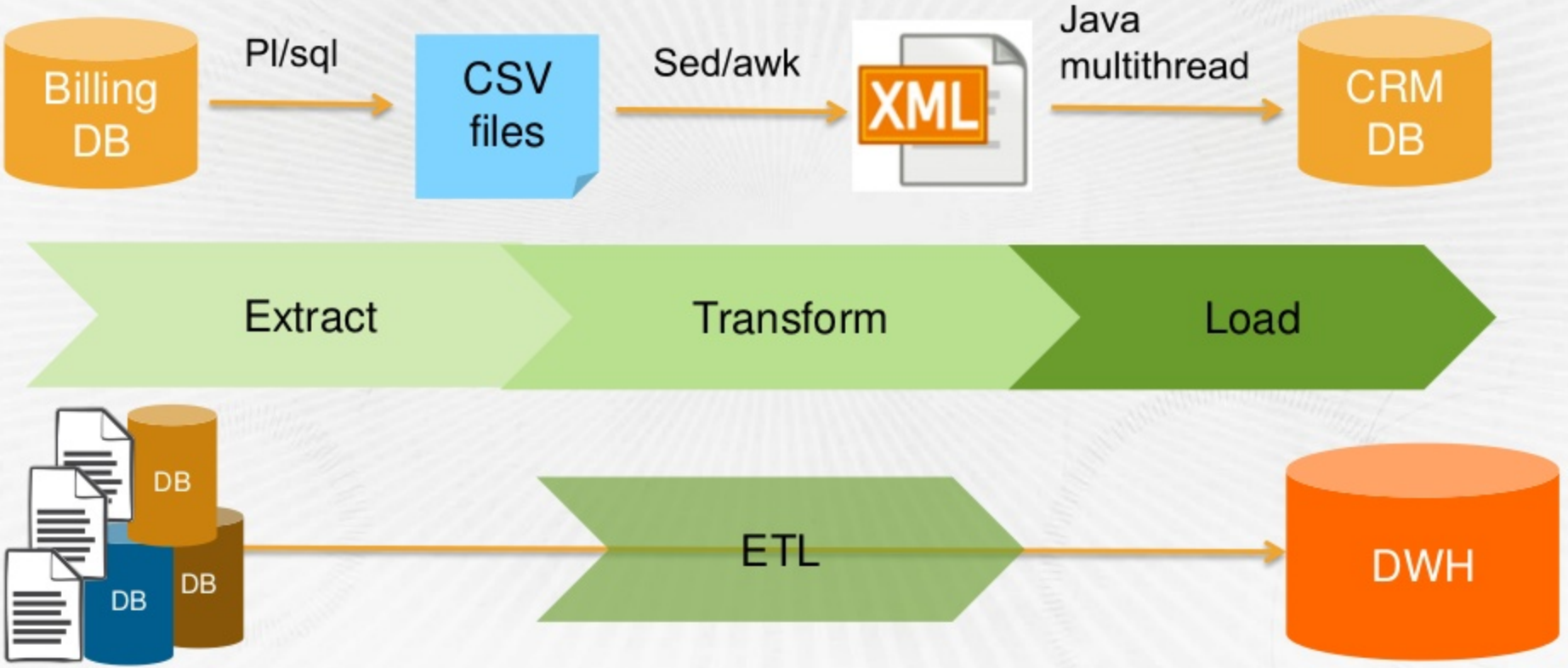
Process & Analyze


Amazon EMR


Amazon Redshift


Amazon EC2

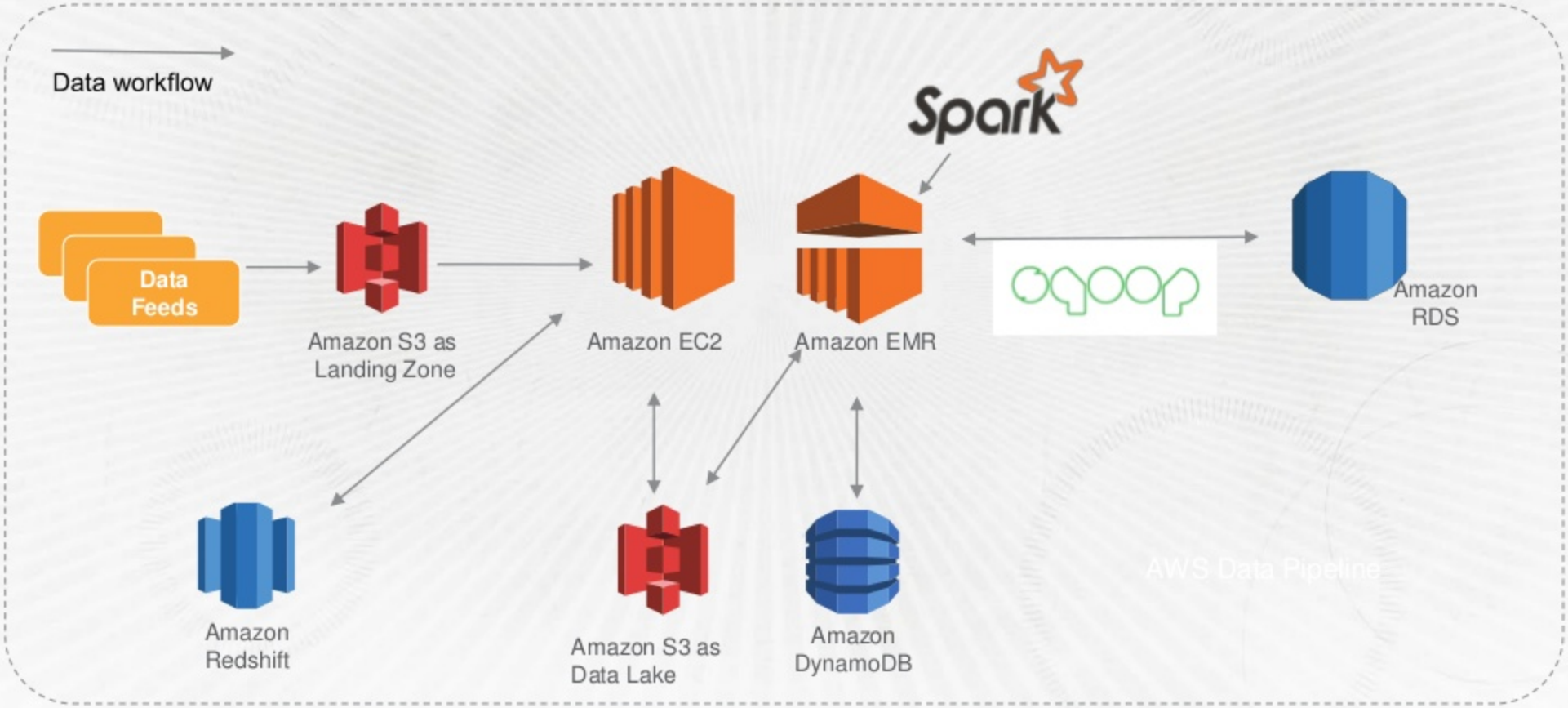
Hello ETL World



ETL as Data Driven Workflow

- What happens if your data doesn’t arrive on time?
- What happens if one of your processing steps fails?
- What happens if a transient error takes out one of your processing nodes?
- How do you operate/monitor?
- How do you control Scheduling?

Data Flow on AWS



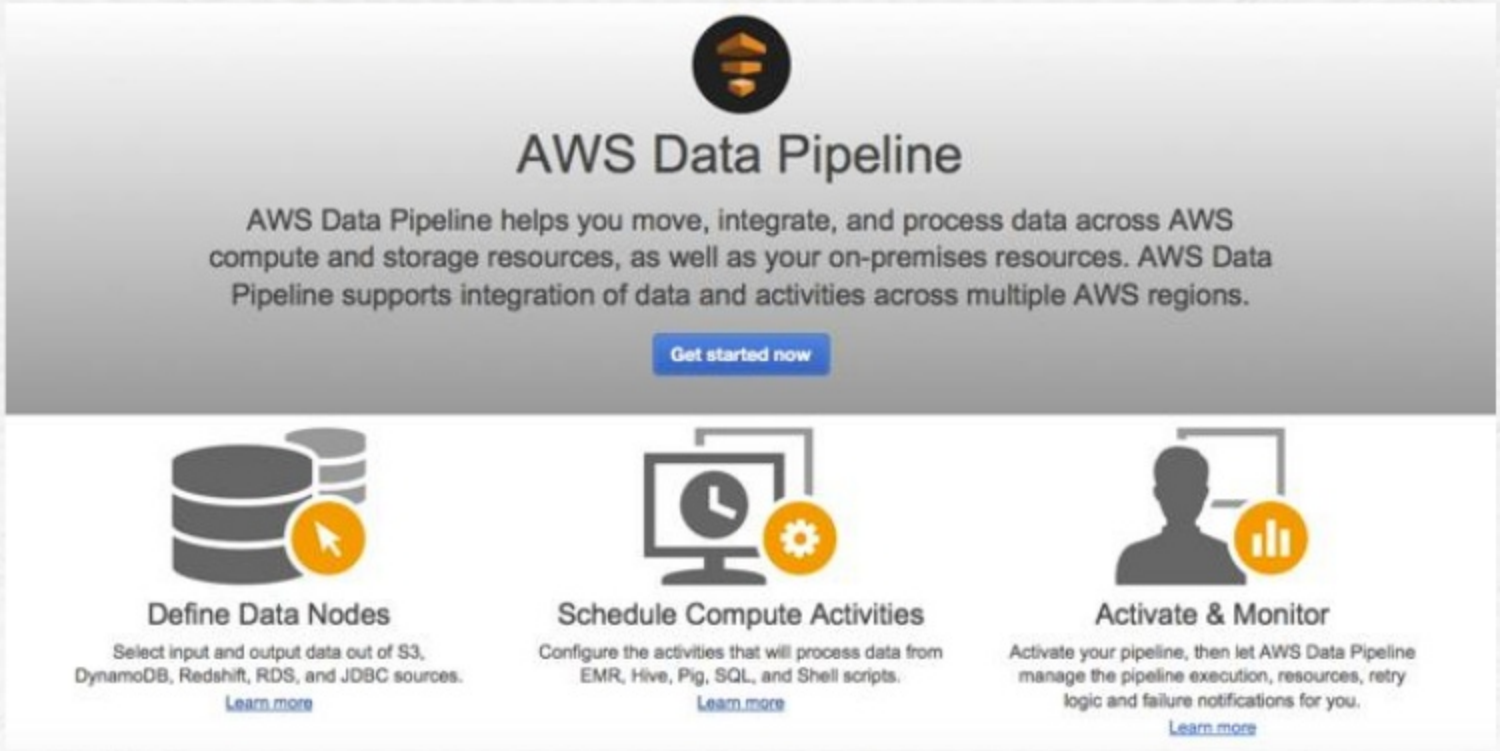
AWS Data Pipeline is…

ETL of Things

AWS Data Pipeline

- Templates Repository
- Manages your workflow data orchestration for you based on activities you define
- Automates creation and management of your resources
- Manages dependencies and automated scheduling for you
- Setup notification/alerts using Amazon SNS (On Job Failure/Success

ETL with AWS Data Pipeline



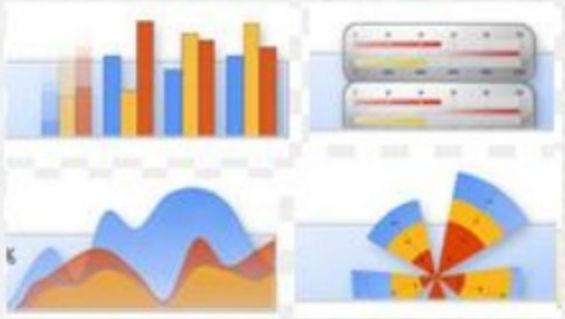
Build and Deploy AWS Data Pipeline

- Console
- CLI
- APIs
  - AWS Java SDK
  - AWS Python SDK
  - AWS Ruby SDK

  http://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-program-pipeline.html

Use Case - Clickstreams

```
 tongue-twisting food terms  5243470 Jenni
 tongue-twisting food terms  5243470 Jenni
ist post leads to stolen bike&amp;#8217;s
38703008004  285369353059736088069175308218
ist post leads to stolen bike&amp;#8217;s
31528812854  285929534262353237469175301695
et Worth Plus:How much income do you need
100899516966502446502850886974850 61  28607
tosses huge tip from &amp;#8216;Tips for Je
60842068623  286071031959919612546116871893
 Beckham: &amp;#8216;I had my breast implan
```



Amazon S3 as Data Lake/Landing Zone

Amazon EMR as ETL Grid

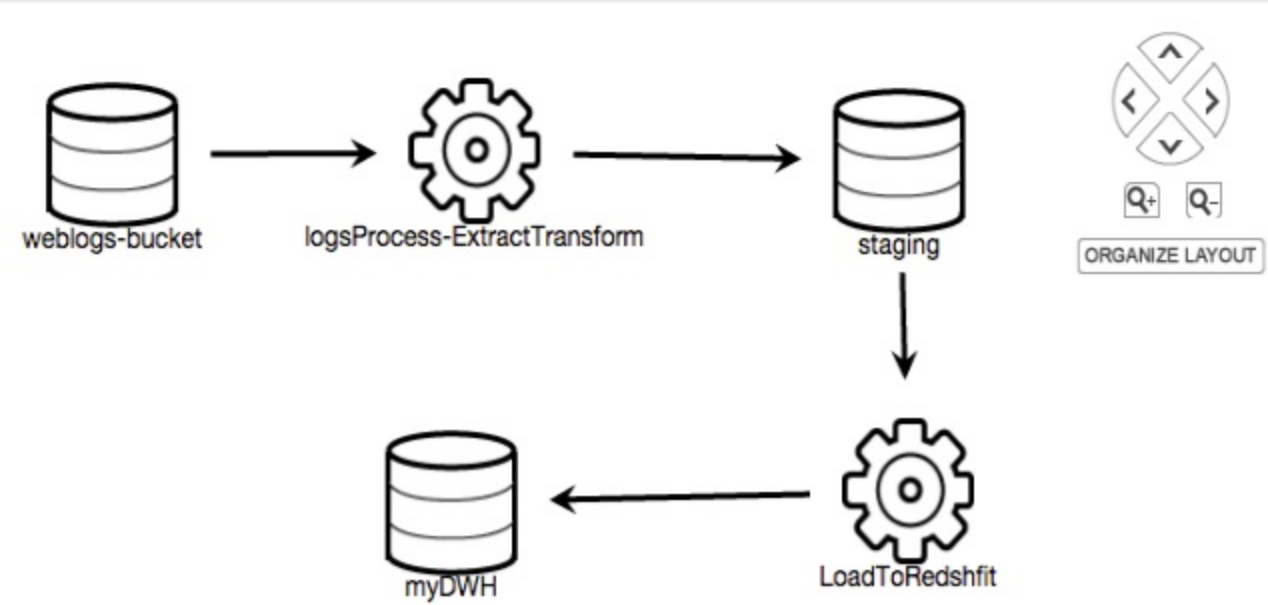
Amazon Redshift – Production DWH

Visualization

Services | S3 | Elastic MapReduce | EC2 | Data Pipeline | Redshift | Edit | Roy Ben-Alta | N. Virgini

DataPipeline > List Pipelines > Execution Details > Architect: my ETL (df-051320818664UI6HLEXW) [Active]

Add activity | Add data node | Save pipeline | Activate | Export | Templates



Activities

logsProcess-ExtractTransform

Name:	logsProcess-ExtractTransform
Type:	PigActivity
Schedule:	RunOnce
Input:	weblogs-bucket
Generated Scripts Path:	s3://reinventbdt303/scripts/
Script Uri:	s3://reinventbdt303/parseItNow

DataNodes

Schedules

Resources

Preconditions

Others

Errors/Warnings

AWS Data Pipeline CLI

Define

aws datapipeline create-pipeline –name myETL --unique-id token
--output: df-09222142C63VXJU0HC0A

Import

aws datapipeline put-pipeline-definition --pipeline-id df-09222142C63VXJU0HC0A --pipeline-definition /home/repo/etl_reinvent.json

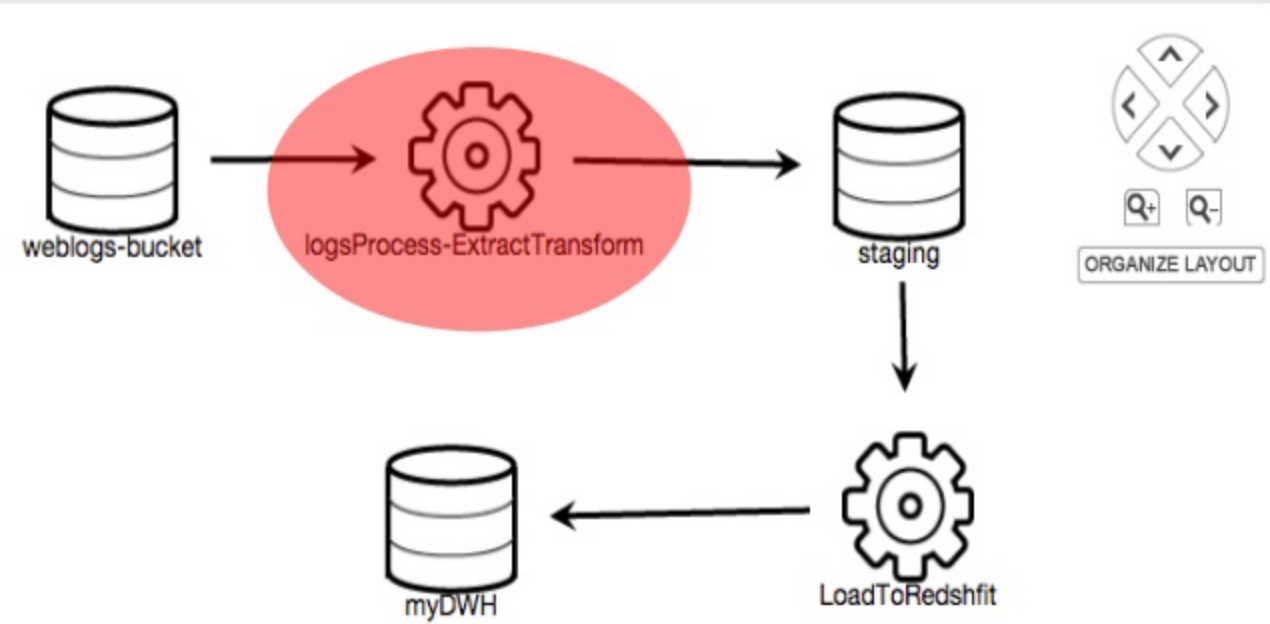
Activate

aws datapipeline activate-pipeline --pipeline-id df-09222142C63VXJU0HC0A

Services | S3 | Elastic MapReduce | EC2 | Data Pipeline | Redshift | Edit | Roy Ben-Alta | N. Virgini

DataPipeline | List Pipelines > Execution Details > Architect: my ETL (df-051320818664UI6HLEXW) [Active]

Add activity | Add data node | Save pipeline | Activate | Export | Templates



Activities

logsProcess-ExtractTransform

Name: logsProcess-ExtractTransform

Type: PigActivity

Schedule: RunOnce

Input: weblogs-bucket

Generated Scripts Path: s3://reinventbdt303/scripts/

Script Uri: s3://reinventbdt303/parseItNow

DataNodes

Schedules

Resources

Preconditions

Others

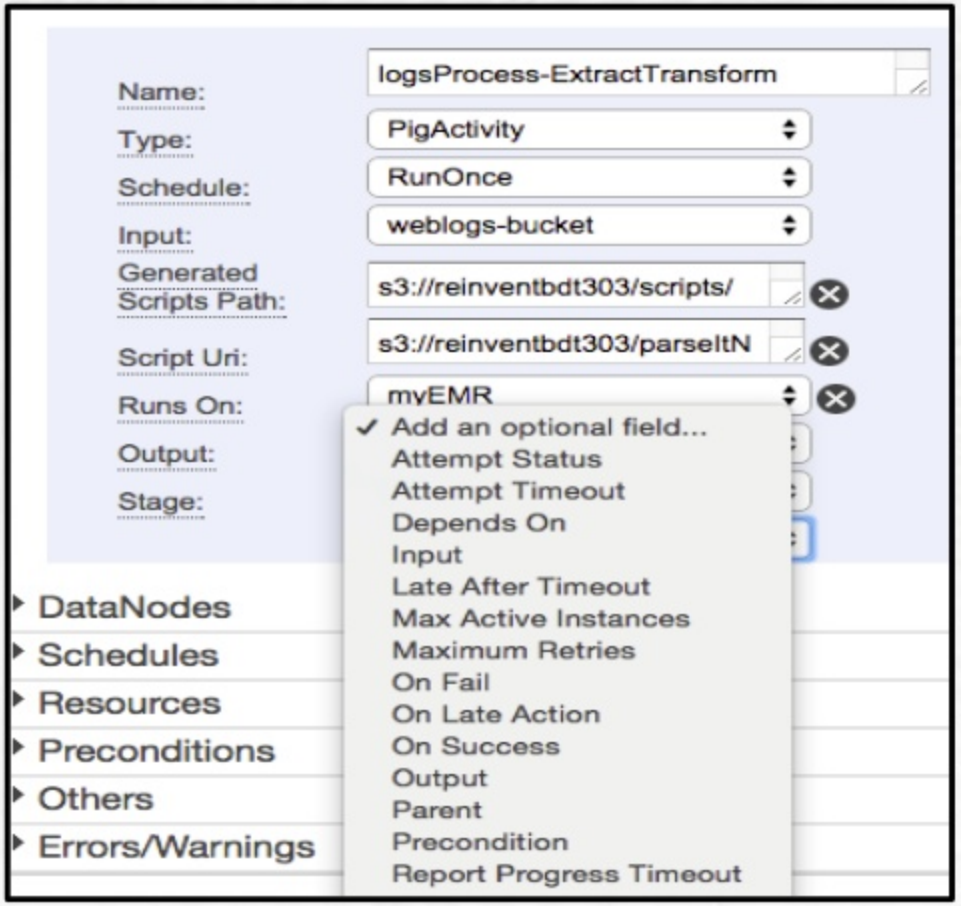
Errors/Warnings

Extract and Transform Using Pig

```
register /home/hadoop/lib/pig/piggybank.jar;
DEFINE EXTRACT org.apache.pig.piggybank.evaluation.string.EXTRACT;


RAW_LOGS = LOAD 's3://elasticmapreduce/samples/pig-apache/input/' USING
TextLoader as (line:chararray);
LOGS_BASE = FOREACH RAW_LOGS GENERATE
    FLATTEN(
      EXTRACT(line, '^(\\S+) (\\S+) (\\S+) \\[([\\w:/]+\\s[+\\-]\\d{4})\\] "(.
      +?)" (\\S+) (\\S+) "([^"]*)" "([^"]*)"')
    )
    as (
      host:    chararray,
      identity: chararray,
      user:          chararray,
      request_time:          chararray,
      request:       chararray,
      status:        int,
      size:  chararray,
      referrer:      chararray,
      agent:       chararray
  );
PARS = FOREACH LOGS_BASE GENERATE host,user,referrer,request_time;
STORE PARS into 's3://reinventbdt303/staging3/' USING PigStorage(',');
```

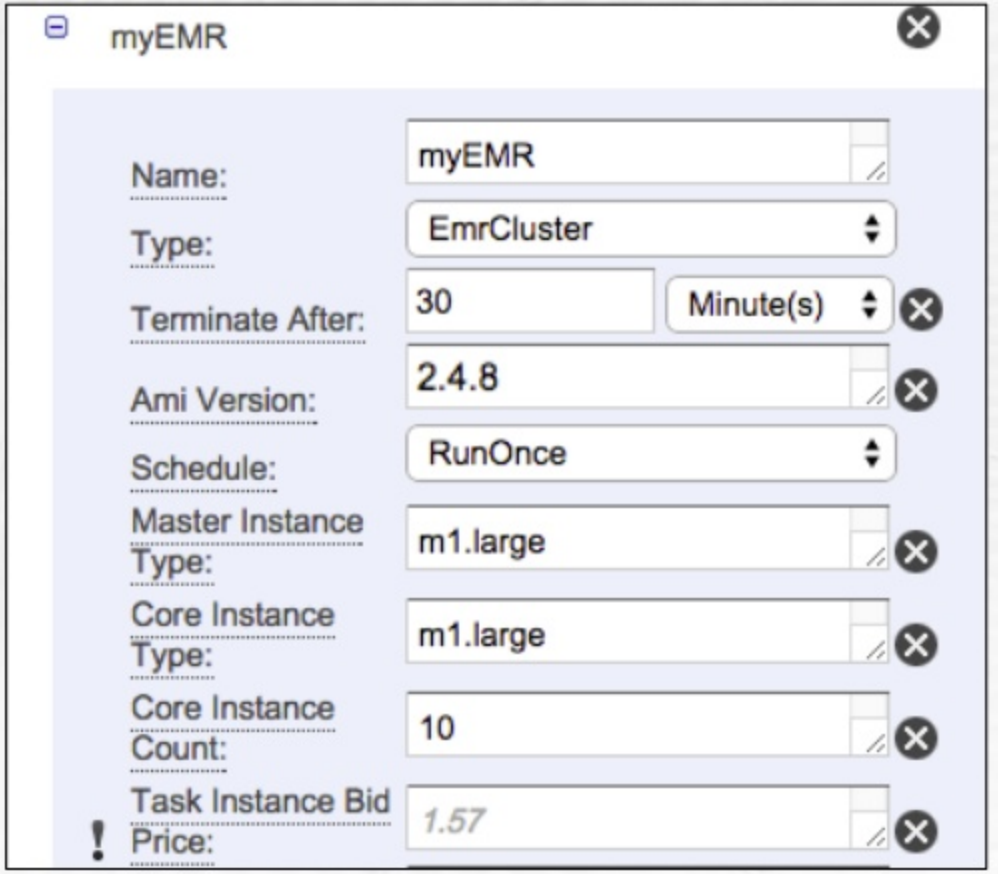
PigActivity

- Setup Schedule
- Retry
- On Fail/On Success
- Define Input
- Define Output
- **Runs On**



Amazon EMR Resource

- Transient Cluster
- Use Spot Instance
- Select Instance type
- Other 20+ Optional attributes you can define
- Bootstrap EMRFS



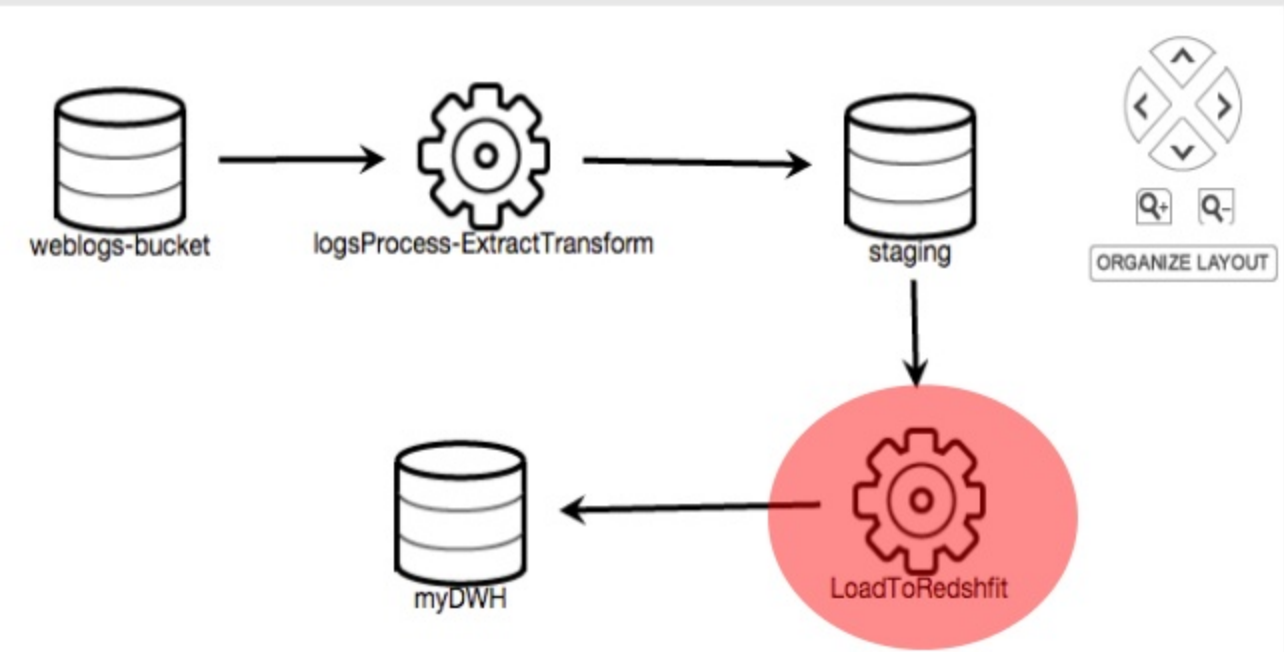
Amazon EMR Resource - EMRFS

{"terminateAfter": "1 Hour",
"id": "EmrClusterId_tX1ME",
"amiVersion": "3.2.1",
"schedule": {
"ref": "ScheduleId_E8jhD"},
"keyPair": "myKeyPair",
"masterInstanceType": "m1.large",
"bootstrapAction": "s3://elasticmapreduce/bootstrap-actions/configure-hadoop,-e,fs.s3.consistent=true,-e,
fs.s3.consistent.retryPeriodSeconds=10, -e, fs.s3.enableServerSideEncryption=true",
"coreInstanceType": "m3.xlarge",
"enableDebugging": "true",
"name": "DefaultEmrCluster1",
"coreInstanceCount": "3",
"type": "EmrCluster"}

Services | S3 | Elastic MapReduce | EC2 | Data Pipeline | Redshift | Edit | Roy Ben-Alta | N. Virgi

DataPipeline > List Pipelines > Execution Details > Architect: my ETL (df-051320818664UI6HLEXW) [Active]

Add activity | Add data node | Save pipeline | Activate | Export | Templates



Activities

LoadToRedshfit

Name:	LoadToRedshfit
Type:	RedshiftCopyActivity
Schedule:	RunOnce
Input:	staging
Precondition:	checkTrigger
Command Options:	copy weblogs\nfrom

DataNodes

Schedules

Resources

Preconditions

Others

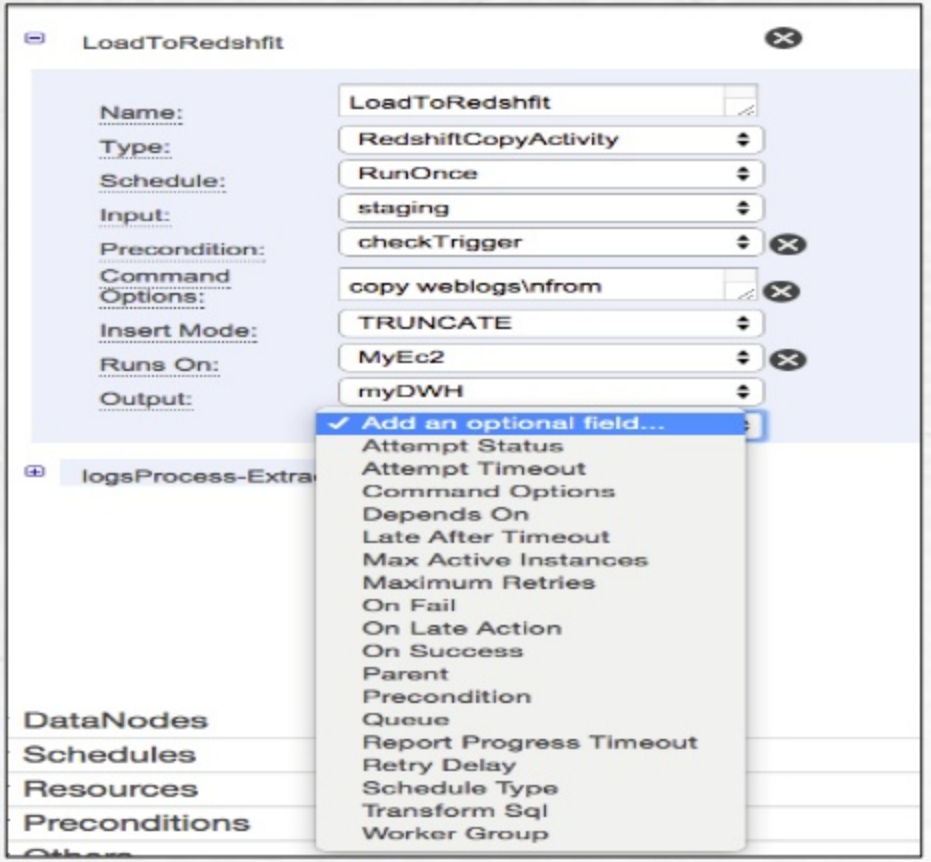
Errors/Warnings

Load to Amazon Redshift

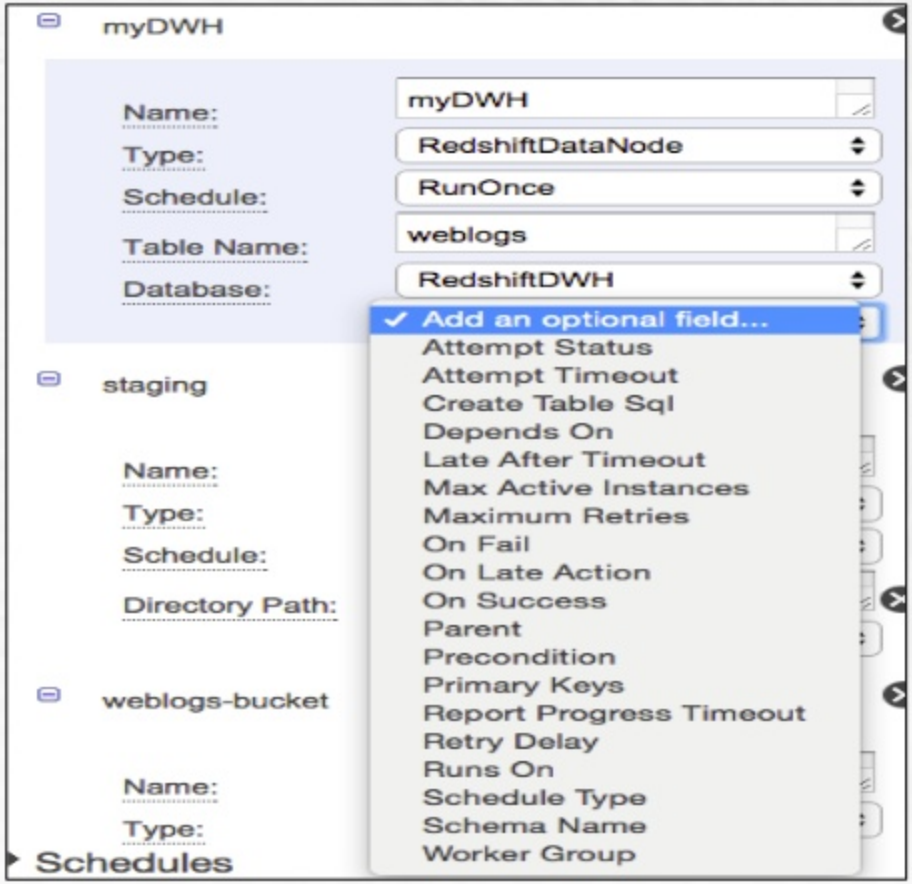
```
copy weblogs
from 's3://staging/'
credentials 'aws_access_key_id=<my access
key>;aws_secret_access_key=<my secret key>'
delimiter ','
```

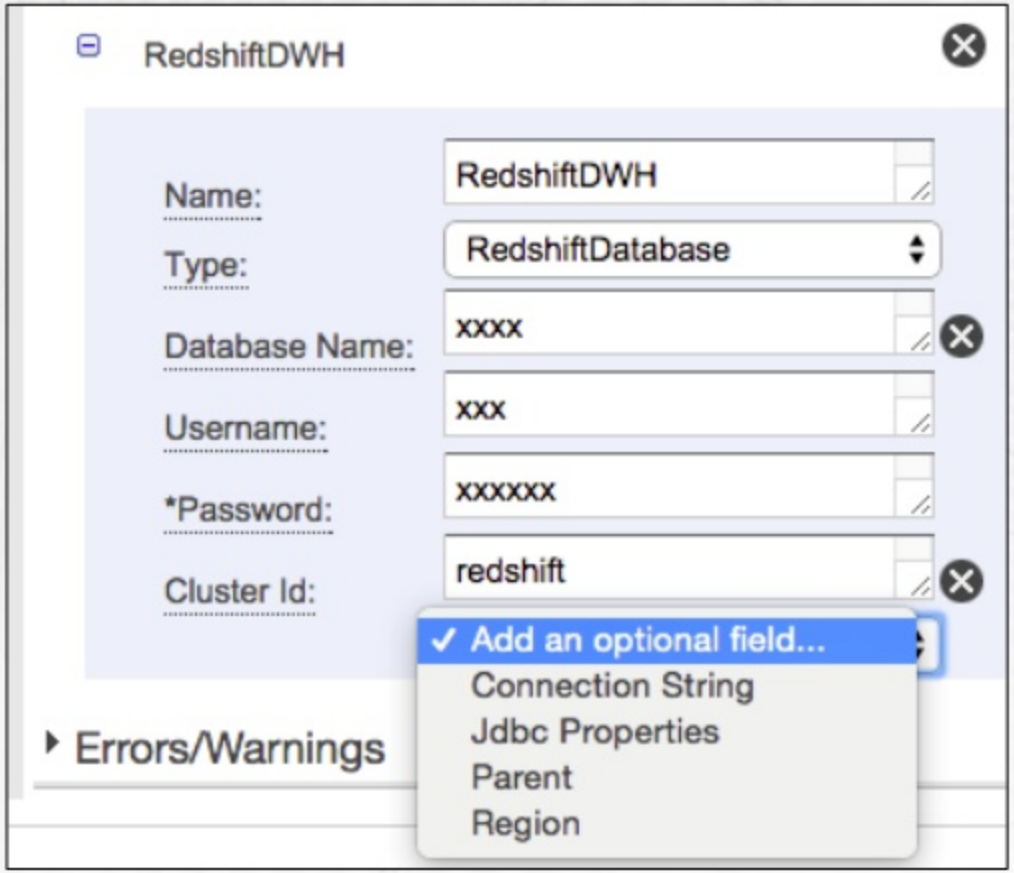
RedshiftCopyActivity

- Copy directly from Amazon S3
- Copy directly from Amazon DynamoDB
- Schedule
- **Precondition**
- Runs On – Set your Amazon EC2 engine to copy data

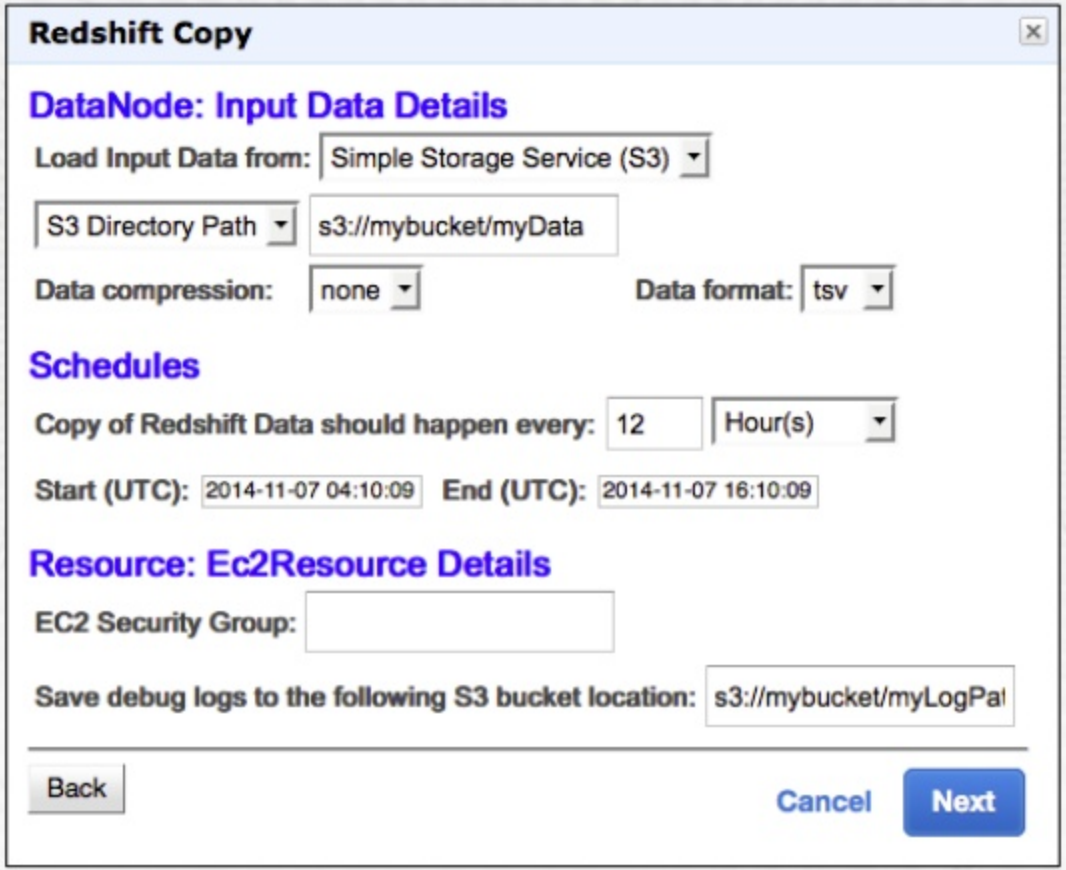


Amazon Redshift Data Node

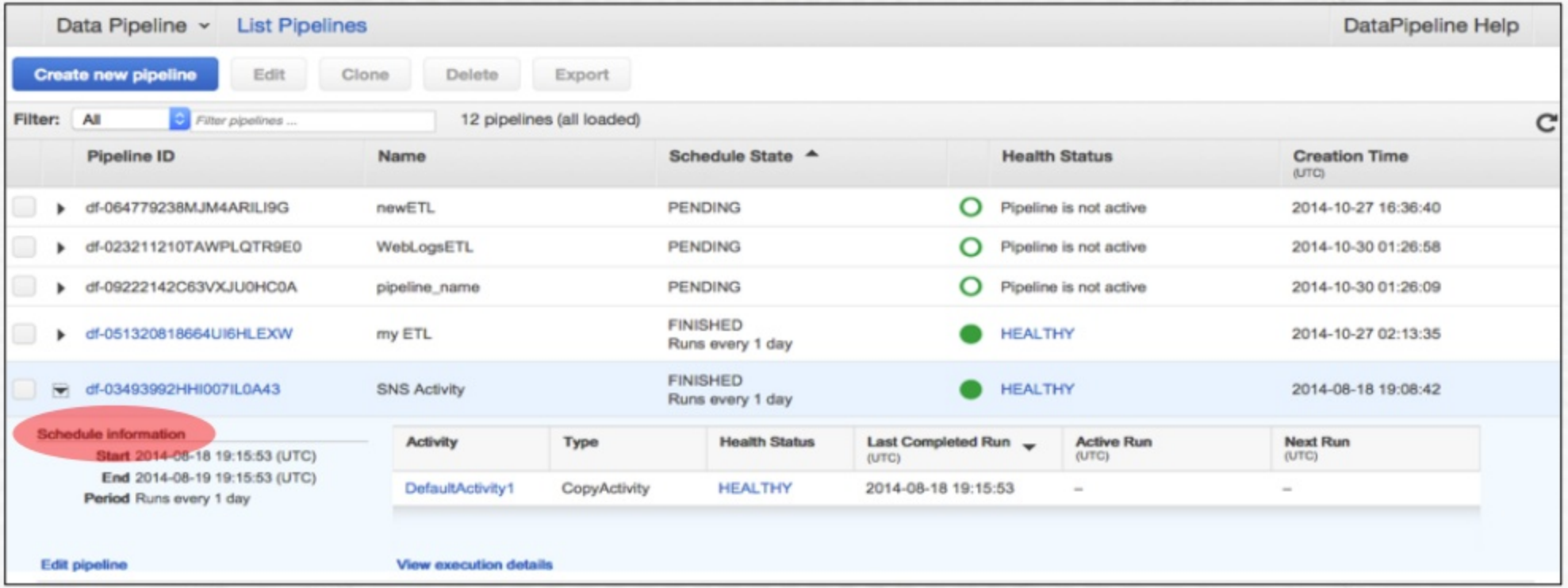




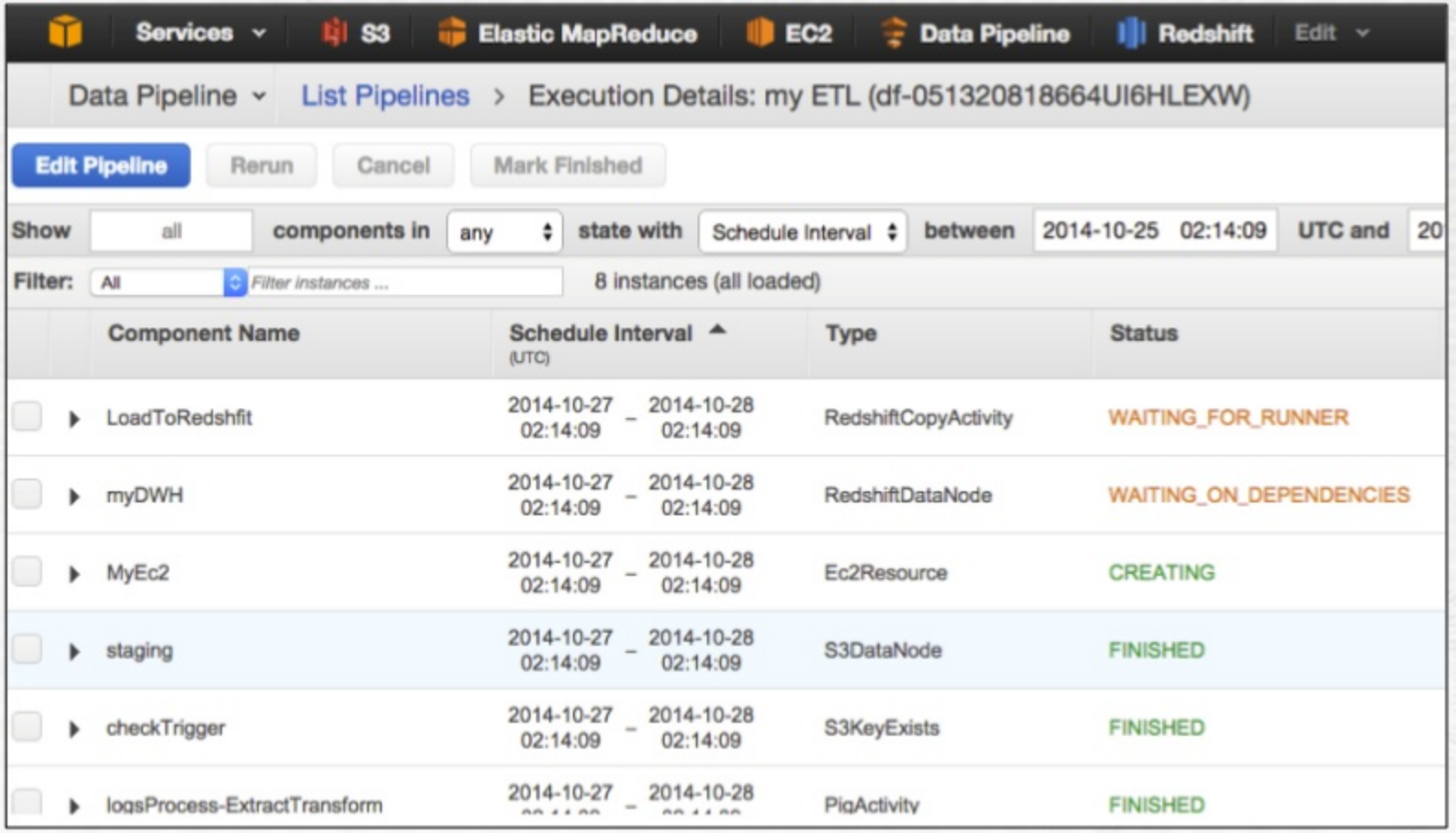
Amazon Redshift Copy Wizard



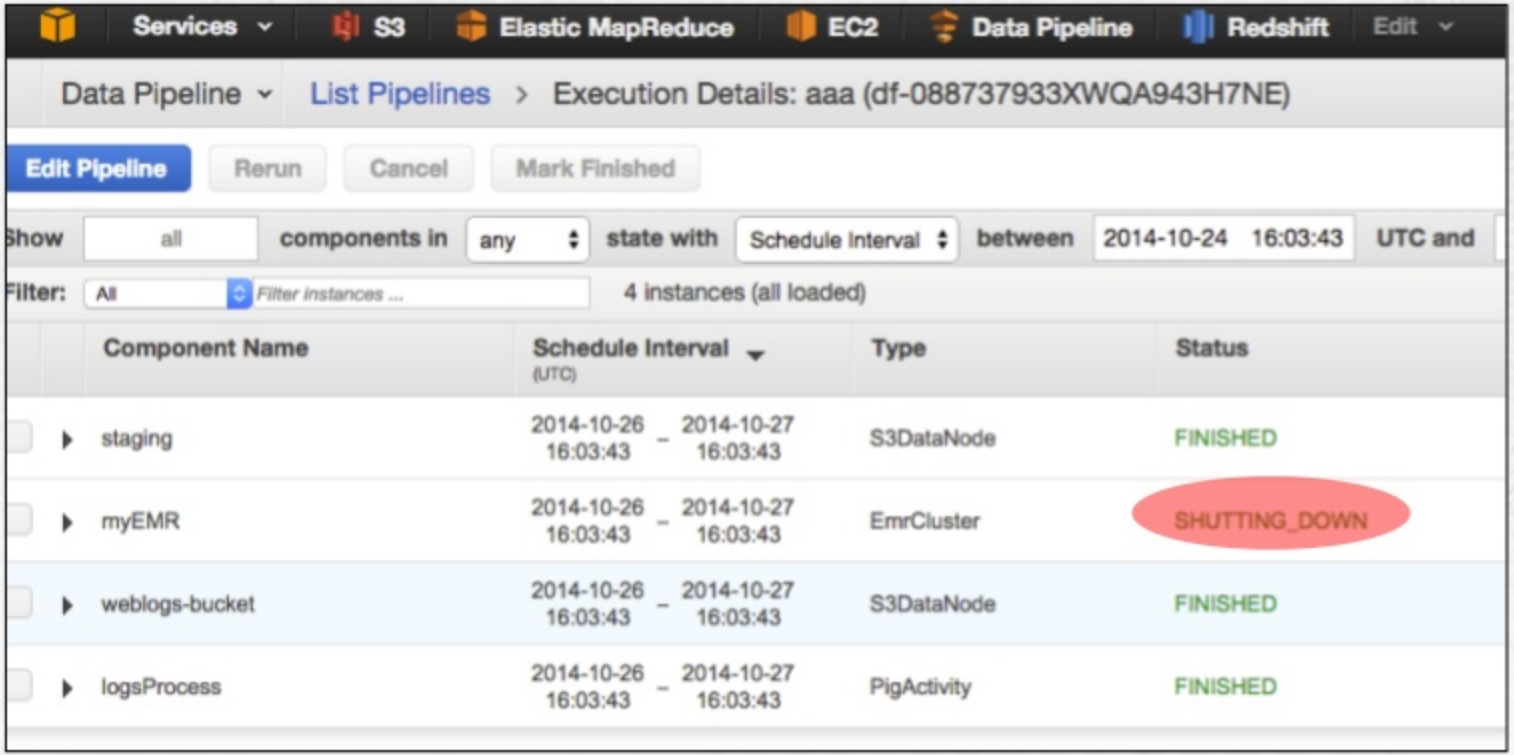
AWS Data Pipeline Dashboard



AWS Data Pipeline Operation



AWS Data Pipeline Monitoring



Tips

- Focus on business logic first
- Design for Failure
- Scheduling – (Min interval 15 minutes)
  - Run Once to test
  - Use Backfill
- SLA
- Build Once, Deploy many

Coursera

Coursera Big Data Analytics Powered by AWS

Thomas Barthelemy
SWE, Data Infrastructure
thomas@coursera.org

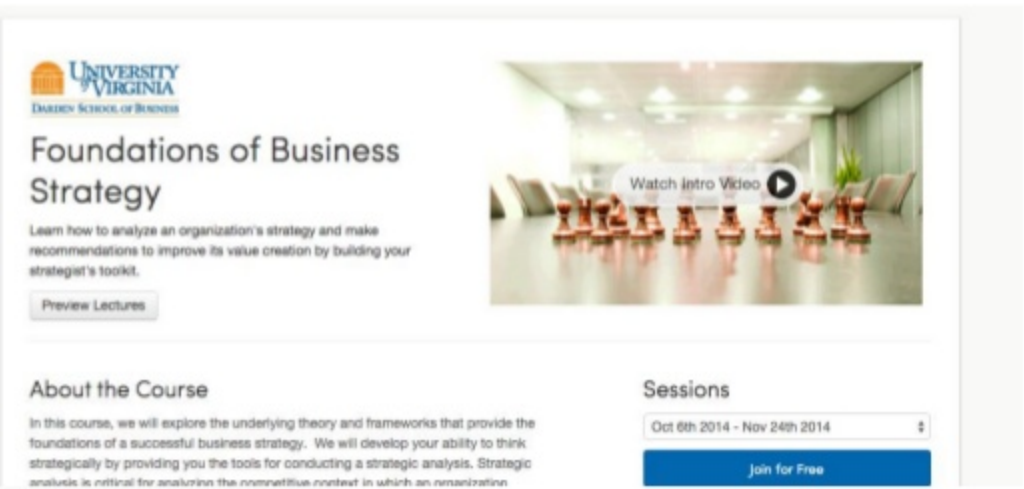
Overview

- About Coursera
- Phase 1: Consolidate data
- Phase 2: Get users hooked
- Phase 3: Increase reliability
- Looking forward

Coursera at a Glance

About Coursera

- Platform for Massive Open Online Courses
- Universities create the content
- Content free to the public



Coursera Stats

- ~10 million learners
- +110 university partners
- >200 courses open now
- ~170 employees

The value of data at Coursera

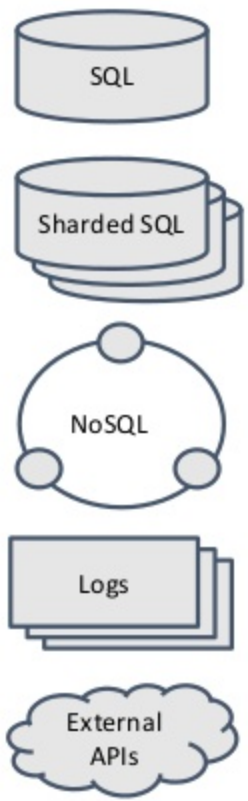
- Making strategic and tactical decisions
- Studying pedagogy

Becoming More Data-Driven

- Since the early days, Coursera has understood the value of data
  - Founders came from machine learning
  - Many of the early employees researched with the founders
- Cost of data access was high
  - each analysis requires extraction, pre-processing
  - data only available to data scientists + engineers

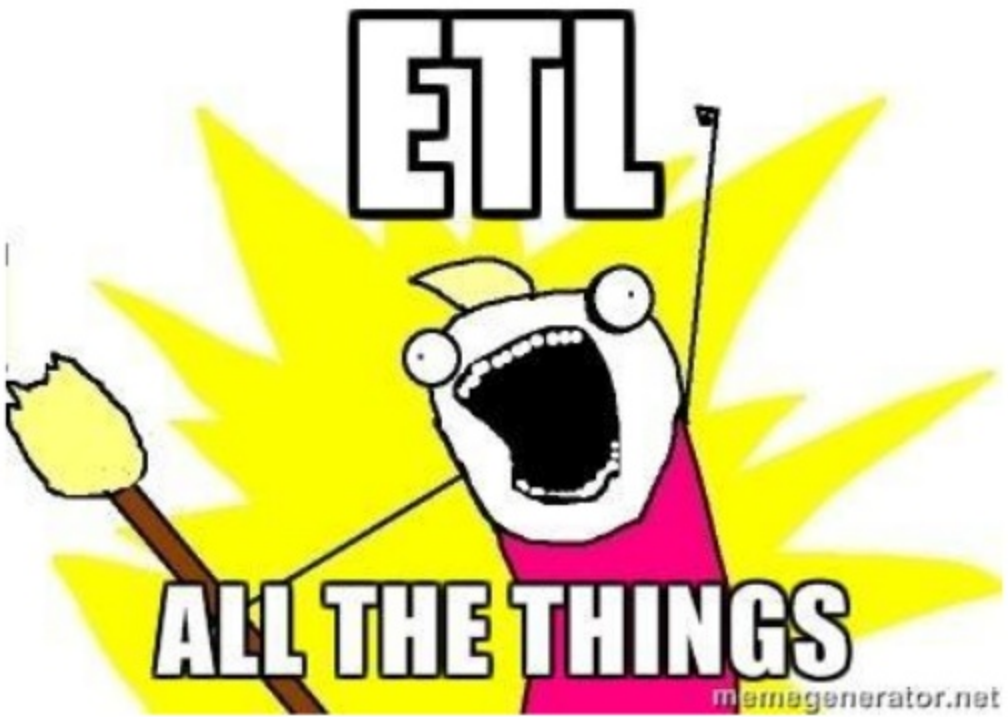
Phase 1:
Consolidate data

Sources



- MySQL
  - Site data
  - Course data sharded across multiple database
- Cassandra increasingly used for course data
- Logged event data
- External APIs

(Obligatory meme)



What platforms to use?

- **Amazon Redshift** had glowing recommendations
- **AWS Data Pipeline** has native support for various Amazon services

ETL development was slow :(

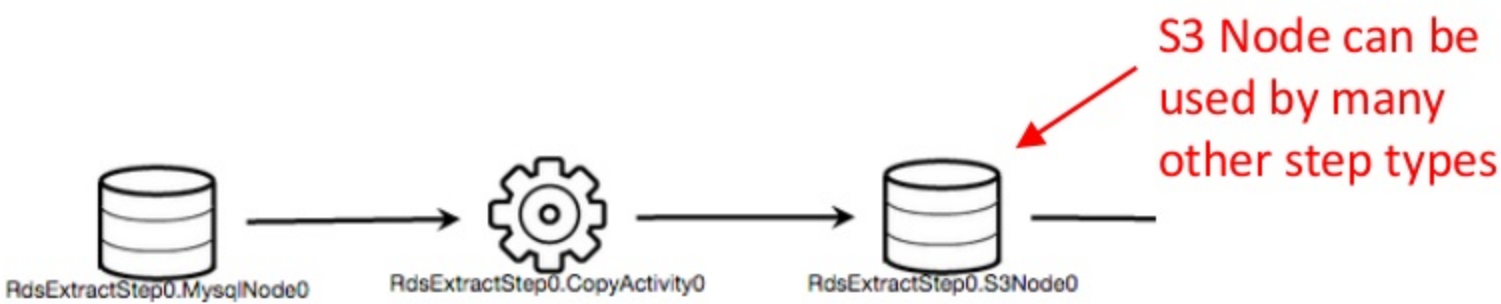
- Slow to do the following in the console:
  - create one pipeline
  - create similar pipelines
  - update existing pipelines

Solution: Programmatically create pipelines

- Break ETL into reusable **steps**
  - Extract from variety of sources
  - Transform data with Amazon EMR, Amazon EC2, or within Amazon Redshift
  - Load data principally into Amazon Redshift
- Use Amazon S3 as intermediate state for Amazon EMR- or Amazon EC2-based transformations
- Map **steps** to set of AWS Data Pipeline objects

Example step: extract-rds

- Parameters: hostname, database name, SQL
- Creates pipeline objects:



Example load definition

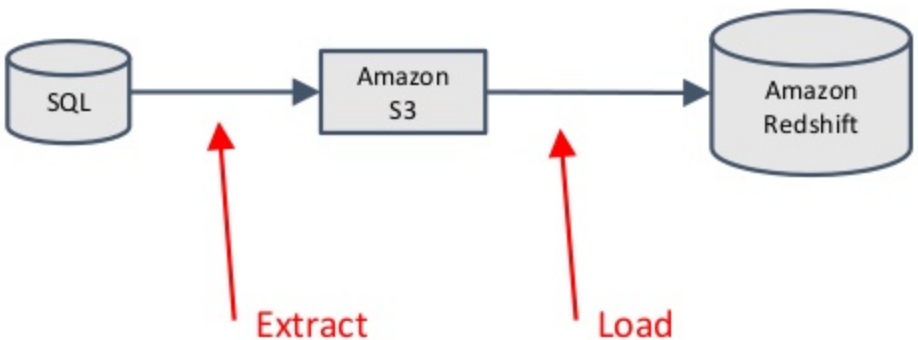
```
steps:
-   type: extract-from-rds
    sql: | SELECT instructor_id, course_id, rank
             FROM courses_instructorincourse;
    hostname: maestro-read-replica
    database: maestro

-   type: load-into-staging-table
    table: staging.maestro_instructors_sessions

-   type: reload-prod-table
    source: staging.maestro_instructors_sessions
    destination: prod.instructors_sessions
```

Extract data from Amazon RDS

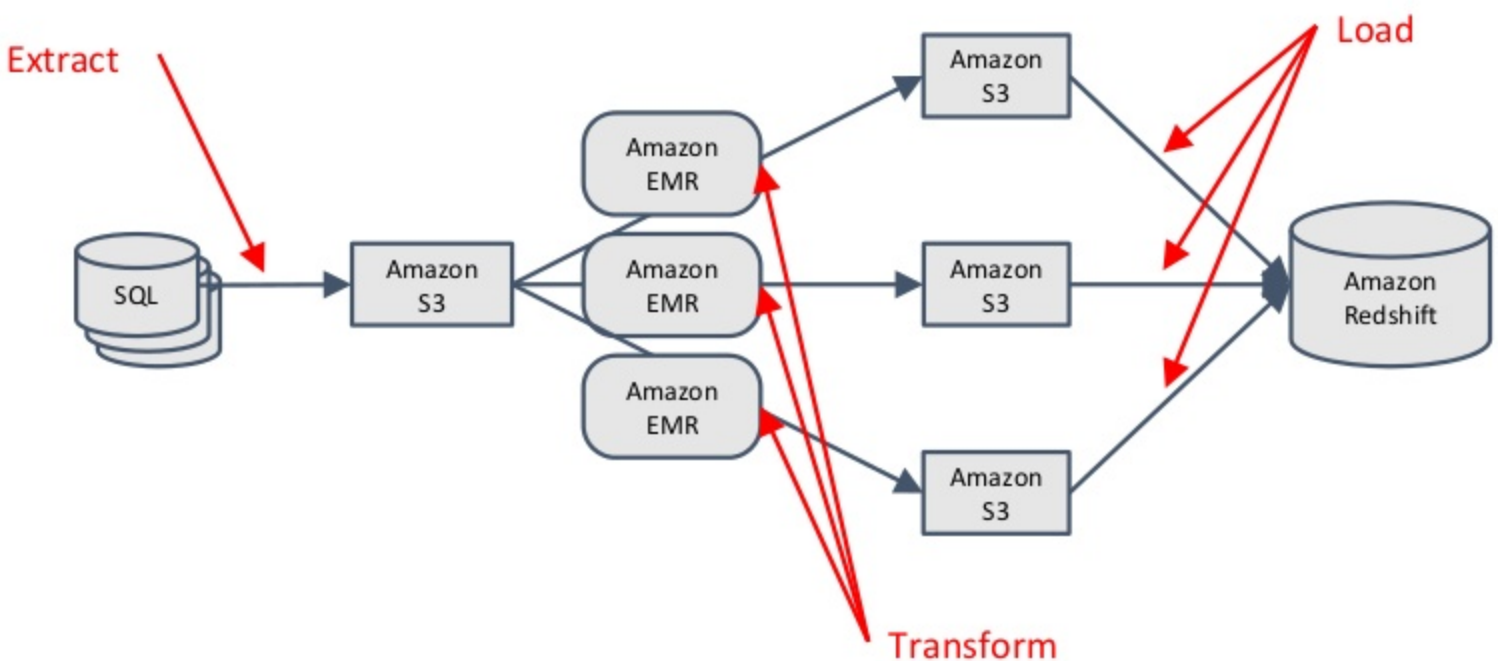
Load intermediate table in Amazon Redshift

Reload target table with new data

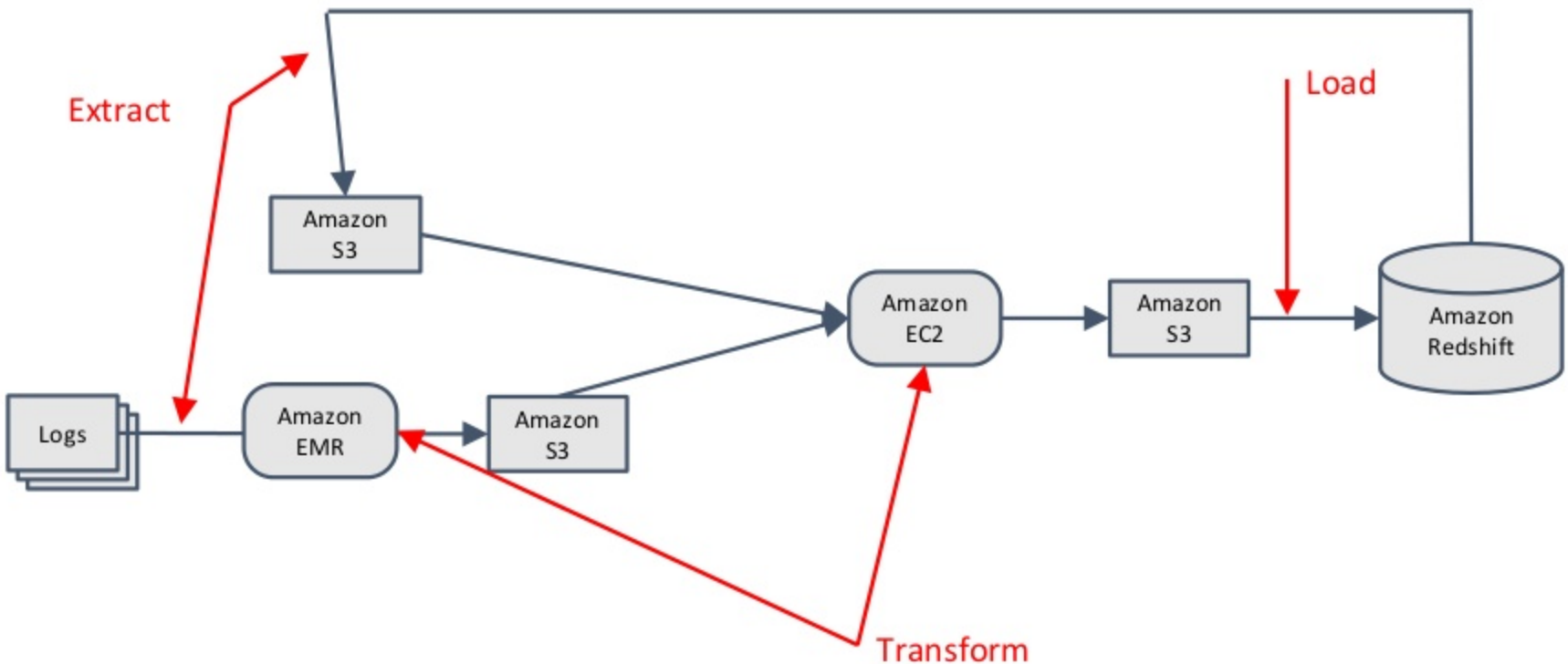
ETL – Amazon RDS



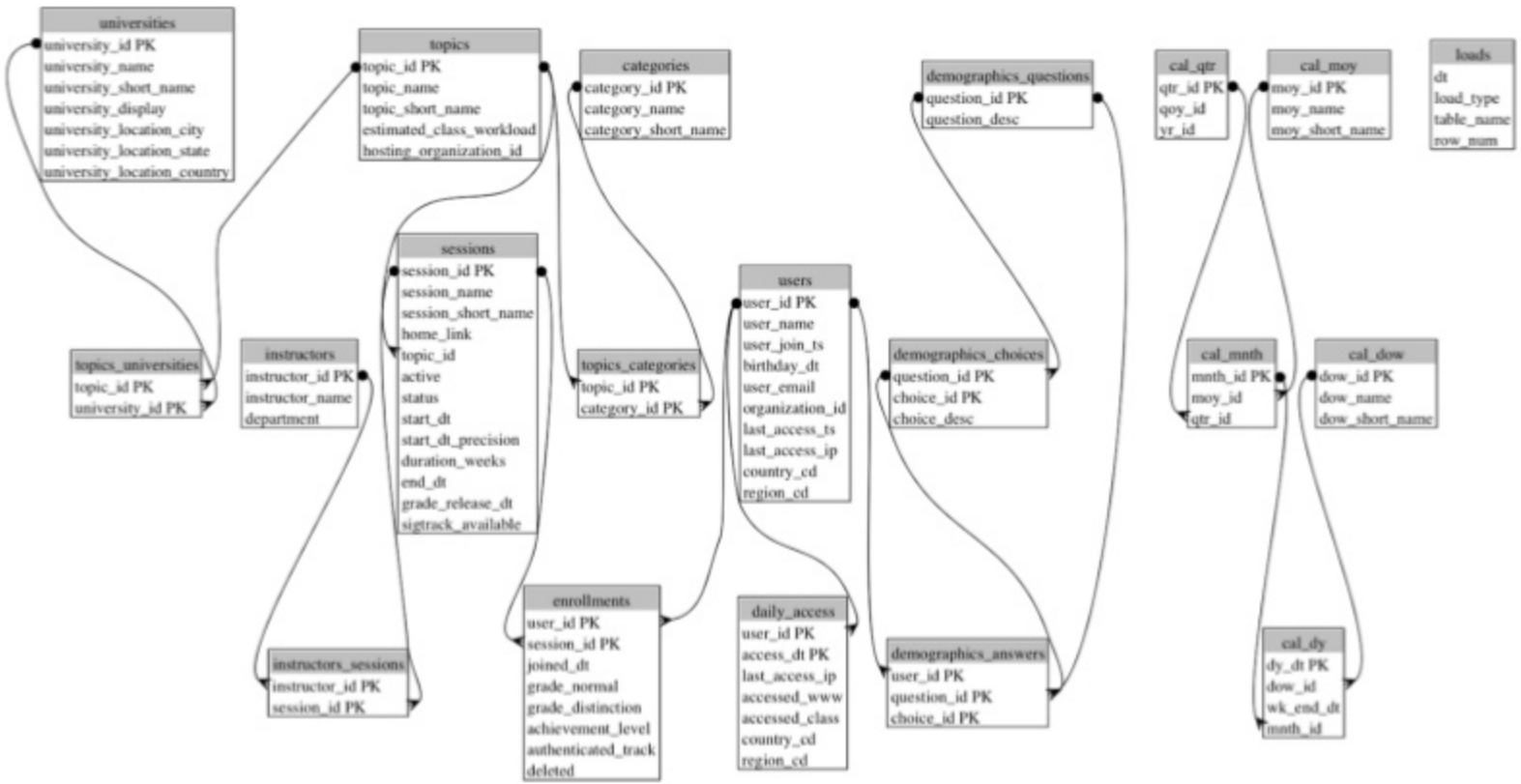
ETL – Sharded RDS



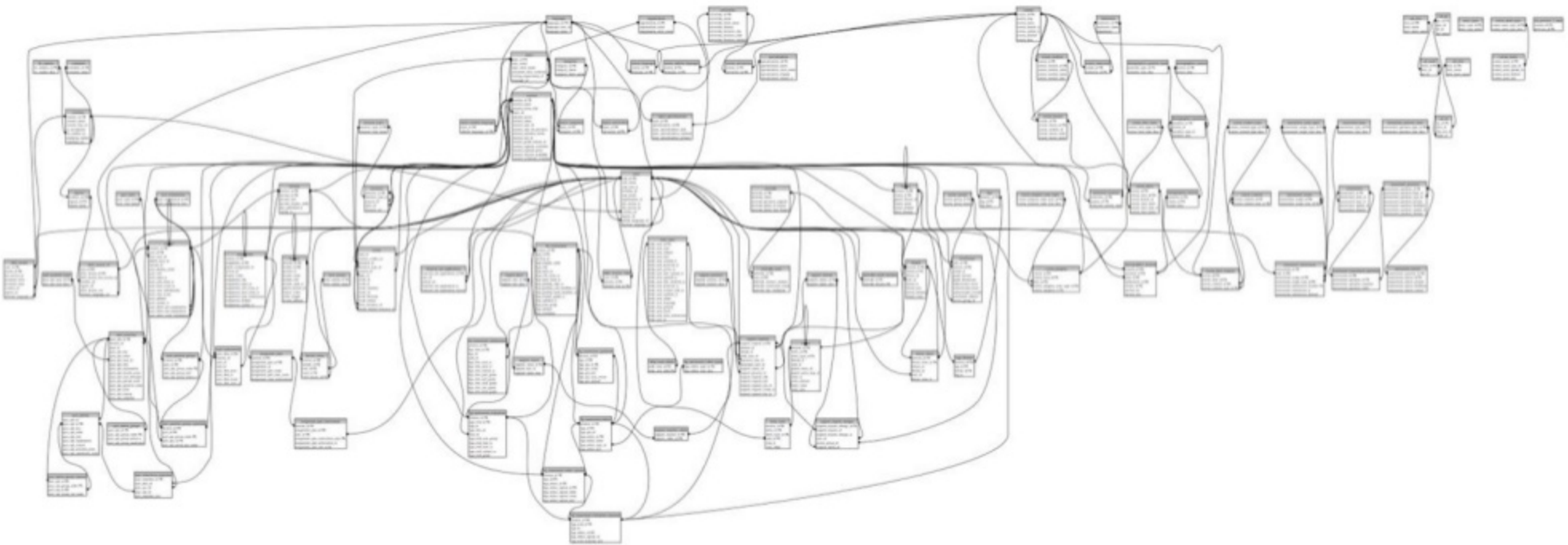
ETL – Logs


Extract
Load
Amazon S3
Amazon EC2
Amazon S3
Amazon Redshift
Logs
Amazon EMR
Amazon S3
Transform

Reporting Model, Dec 2013



Reporting Model, Sep 2014



AWS Data Pipeline

- Easily handles starting/stopping of resources
- Handles permissions, roles
- Integrates with other AWS services
- Handles “flow” of data, data dependencies

Dealing with large pipelines

- Monolithic pipelines hard to maintain
- Moved to making pipelines smaller
  - Hooray modularity!
- If pipeline B depended on pipeline A, just schedule it later
  - Add a time buffer just to be safe

Setting cross-pipeline dependencies

- Dependencies accomplished using a script that would wait until dependencies finished
  - **ShellCommandActivity** to the rescue

The beauty of ShellCommandActivity

- You can use it anywhere
  - Accomplish tasks that have no corresponding activity type
  - Override native AWS Data Pipeline support if it does not meet your needs

ETL library

- Install on machine as first step of each pipeline
  - With **ShellCommandActivity**
- Allows for even more modularity

Phase 2:
Get users hooked

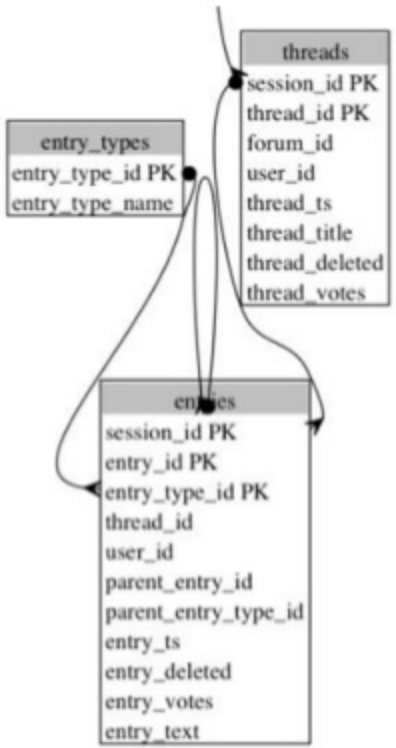
We have data. Now what?

- Simply collecting data will not make a company data-driven
- First step: make data easier for the analysts to use

Certifying a version of the truth

- Data Infrastructure team creates a 3NF model of the data
- Model is designed to be as interpretable as possible

Example: Forums



- Full-word names for database objects
- Join keys made obvious through universally-unique column names
  - e.g. session_id joins to session_id
- Auto-generated ER-diagram, so documentation is up-to-date

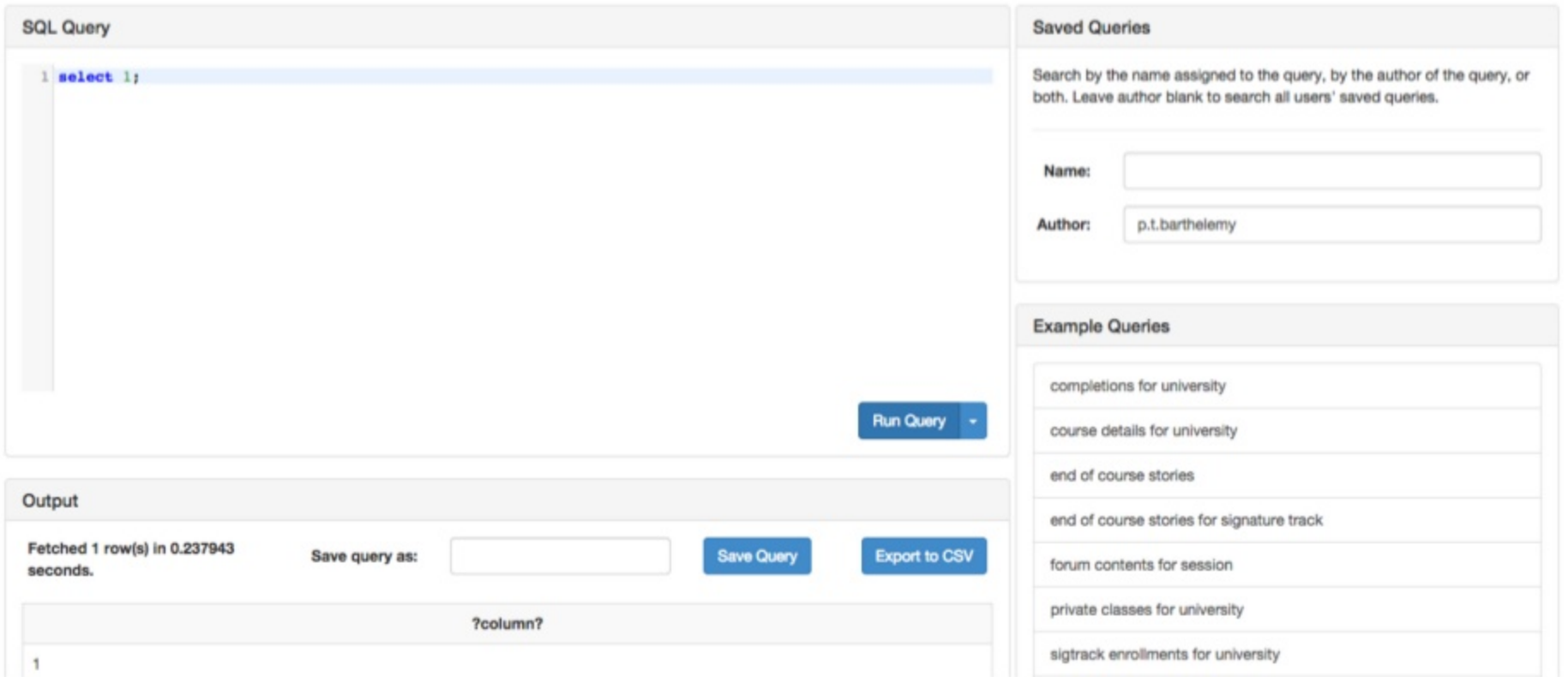
Some power users need data faster!

- Added new schemas for developers to load data
- Help us to remain data driven during early phases of…
  - Product release
  - Analyzing new data

Lightweight SQL interface

- Access via browser
  - Users don’t need database credentials or special software
- Data exportable as CSV
- Auto-complete for DB object names
- Took ~2 days to make

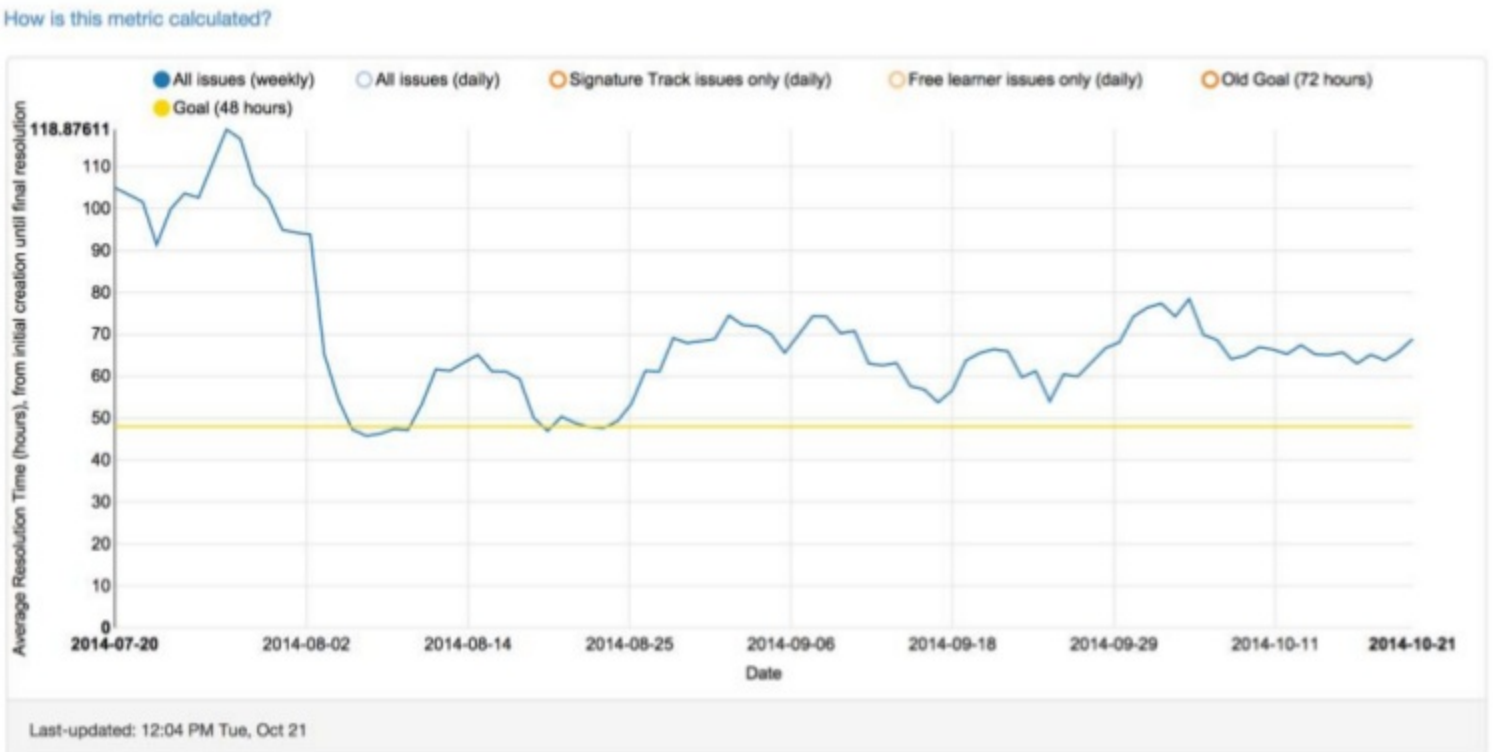
Lightweight SQL interface



But what about people who don't know SQL?

- Reporting tools to the rescue!
- In-house dashboard framework to support:
  - Support
  - Growth
  - Data Infrastructure
  - Many other teams...
- Tableau for interactive analysis

But what about people who don't know SQL?



Phase 3:
Increase reliability

If you build it, they will come

- Built ecosystem of data consumers
  - Analysts
  - Business users
  - Product team

Maintaining the ecosystem

- As we accumulate consumers, need to invest effort in...
  - Model stability
  - Data quality
  - Data availability

Persist Amazon Redshift logs to keep track of usage

- Leverage database credentials to understand usage patterns
- Know whom to contact when
  - Need to change the schema
  - Need to add a new set of tables similar to an existing set (e.g. assessments)

Data Quality

- Quality of source systems (GIGO)
  - Encourage source system to fix issues
- Quality of transformations
  - Responsibility of the data infrastructure team

Quality Transformations

- Automated QA checks as part of ETL
  - Again, use **ShellCommandActivity**
- Key factors
  - Counts
  - Value comparisons
  - Modeling rules (primary keys)

Ensuring High Reliability

- Step retries catch most minor hiccups
- On-call system to recover failed pipelines
- Persist DB logs to keep track of load times
  - Delayed loads can also alert on-call devs
  - Bonus: users know how fresh the data is
  - Bonus: can keep track of success rate
- AWS very helpful in debugging issues

Looking Forward

Current bottlenecks

- Complex ETL
  - How do we easily ETL complex, nested structures?
- Developer bottleneck
  - How do we allow users to access raw data more quickly?

Amazon S3 as principal data store

- Amazon S3 to store raw data as soon as it's produced (data lake)
- Amazon EMR to process data
- Amazon Redshift to remain as analytic platform

Thank you!

Want to learn more?

- github.com/coursera/dataduct
- http://tech.coursera.org
- thomas@coursera.org

Call To Action

- Use AWS Data Pipeline API
- Use AWS Data Pipeline for testing/QE
- Open source integration
- Reusable artifacts
- Amazon RDS, Amazon DynamoDB, Amazon S3, Amazon Redshift, Amazon EMR

Call To Action (2)

- AWS Big Data Blog
  - http://blogs.aws.amazon.com/bigdata/
- AWS Data Pipeline documentation
  - http://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-concepts-schedules.html
- AWS re:Invent Big Data Booth
- AWS Professional Service/SA/Support

AWS re:Invent

Please give us your feedback on this session.
Complete session evaluations and earn re:Invent swag.

BDT303

http://bit.ly/awsevals

amazon web services

Join the conversation on Twitter with #reinvent