ABD315

Serverless ETL with AWS Glue

Mehul A. Shah

Software Manager, AWS Glue

November 27, 2017

AWS re:Invent

aws

Today's Agenda

Intro to AWS Glue

Construct an ETL flow in 4 steps

Under the hood: customize AWS Glue scripts

Merck – customer testimonial

What is AWS Glue?

Fully-managed, serverless extract-transform-load (ETL) service

for developers, built by developers

1000s of customers and jobs

Select AWS Glue customers

NTT docomo

News Corp

AUTODESK.

MediaMath

MERCK

myTomorrows

OLX Group

Vost

Expedia®

amazon Prime Air

21ST CENTURY FOX

AWS re:Invent

aws

There are many tools already

Amazon Redshift Partner Page for Data Integration

ATTUNITY CloudBeam

ironSource .atom

infoteria

bryte Systems

alooma

MATILLION

informatica Cloud

talend* integration at any scale

Segment

Fivetran

etleap

xplenty

FlyData

TREASURE DATA

Qubole

RJMETRICS

snapLogic

AWS re:Invent

aws

Still, ETL developers hand-code

Canvas-based tools are hard to extend

Code is flexible, powerful, and easy to share

Familiar tools and development pipelines

- IDEs, version control, testing, continuous integration

This talk is for developers!

Hand-coding is laborious

schemas change

data formats change

add or change sources

data volume grows

makes hand-coding
error-prone & brittle

AWS Glue does the undifferentiated heavy lifting
so developers can easily customize

AWS Glue Components

Data Catalog

Discover

Automatic crawling

Apache Hive Metastore compatible

Integrated with AWS analytic services

Job Authoring

Develop

Auto-generates ETL code

Python and Apache Spark

Edit, Debug, and Explore

Job Execution

Deploy

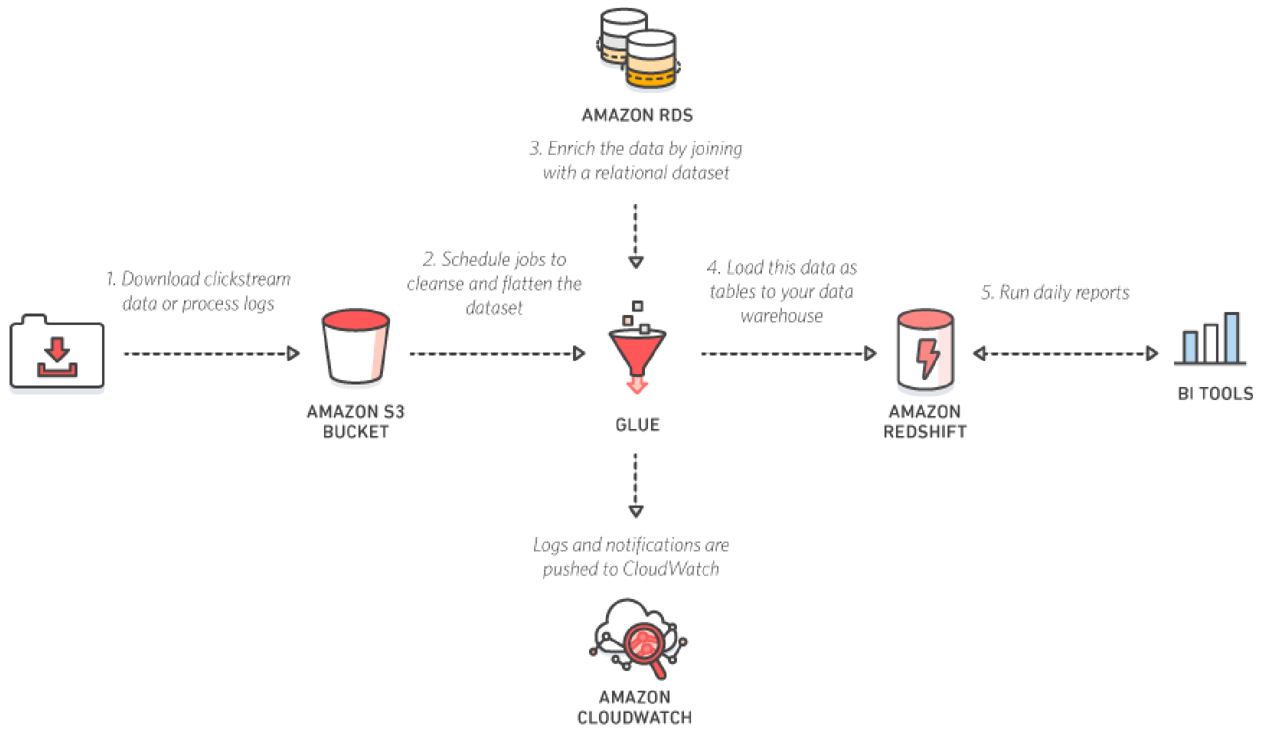
Serverless execution

Flexible scheduling

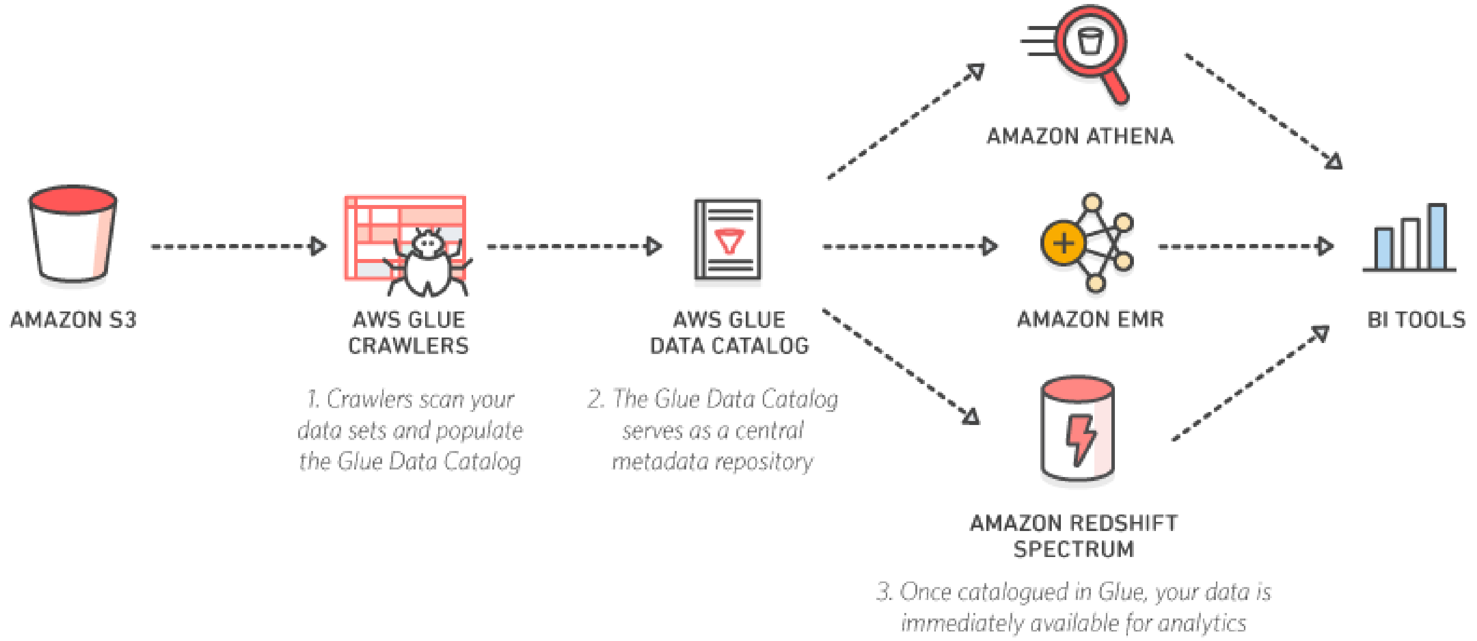
Monitoring and alerting

Common use-cases

Load data warehouses


AMAZON RDS
3. Enrich the data by joining with a relational dataset
1. Download clickstream data or process logs
2. Schedule jobs to cleanse and flatten the dataset
4. Load this data as tables to your data warehouse
5. Run daily reports
AMAZON S3 BUCKET
GLUE
AMAZON REDSHIFT
BI TOOLS
Logs and notifications are pushed to CloudWatch
AMAZON CLOUDWATCH

Build a data lake on Amazon S3

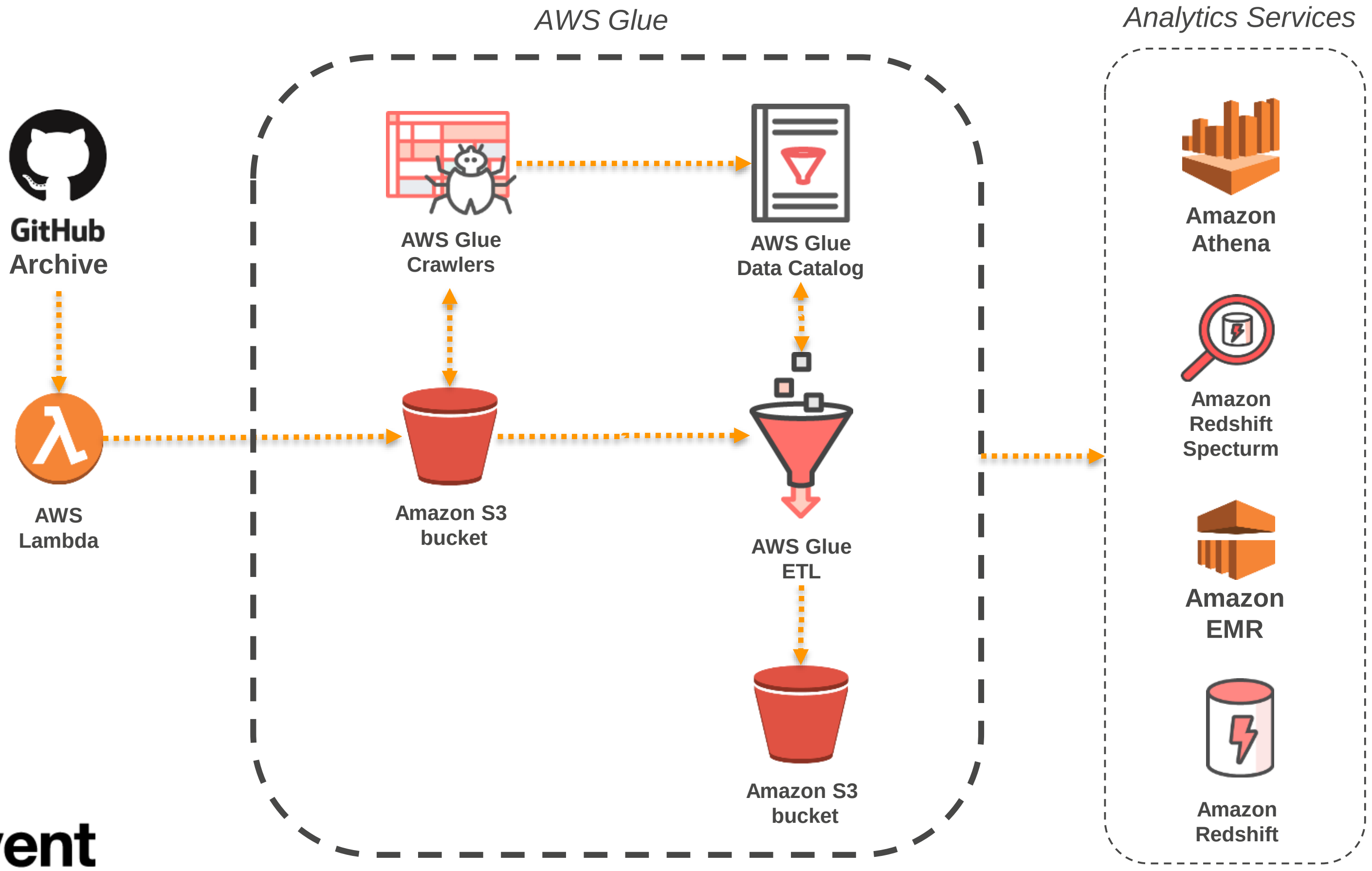

AMAZON S3
AWS GLUE CRAWLERS
1. Crawlers scan your data sets and populate the Glue Data Catalog
AWS GLUE DATA CATALOG
2. The Glue Data Catalog serves as a central metadata repository
AMAZON ATHENA
AMAZON EMR
AMAZON REDSHIFT SPECTRUM
3. Once catalogued in Glue, your data is immediately available for analytics
BI TOOLS

Construct an ETL flow in 4 steps

The 4 Steps

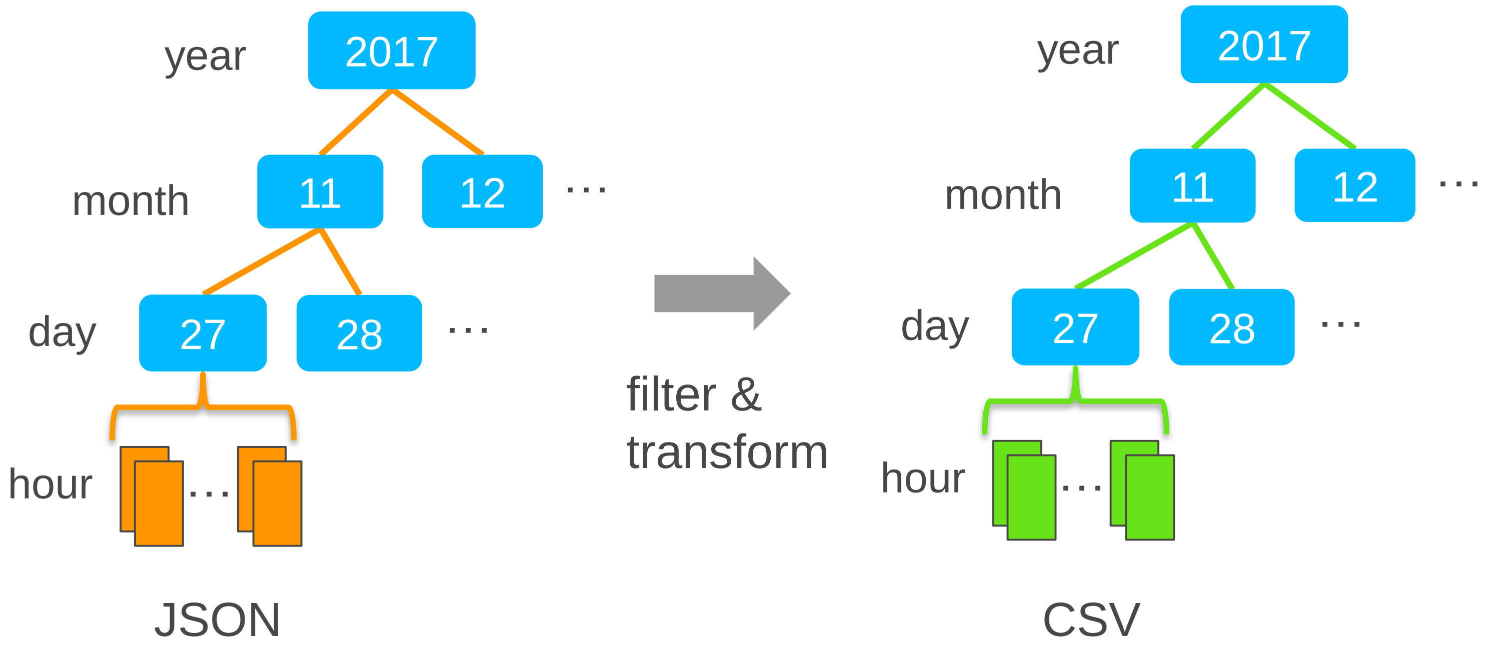
1. Crawl and catalogue your data
2. Specify mappings to generate scripts
3. Interactively edit and explore with dev-endpoints
4. Schedule a job for running in production

ETL example

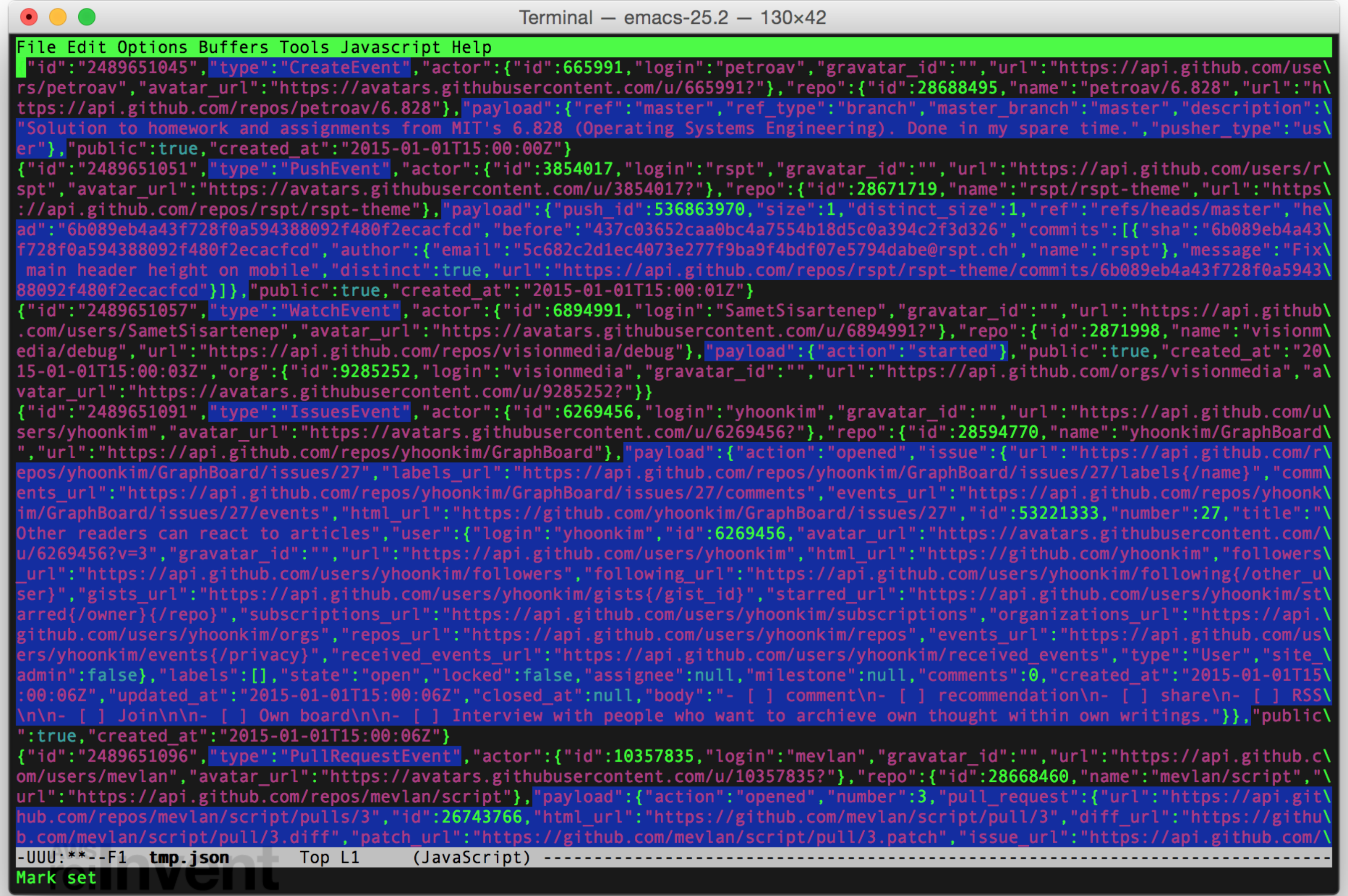

AWS Glue
Analytics Services
GitHub Archive
AWS Lambda
AWS Glue Crawlers
AWS Glue Data Catalog
Amazon S3 bucket
AWS Glue ETL
Amazon S3 bucket
Amazon Athena
Amazon Redshift Specturm
Amazon EMR
Amazon Redshift

ETL example (con't)

Organize data in Apache Hive-style partitions



Public GitHub timeline


Terminal — emacs-25.2 — 130×42
File Edit Options Buffers Tools Javascript Help

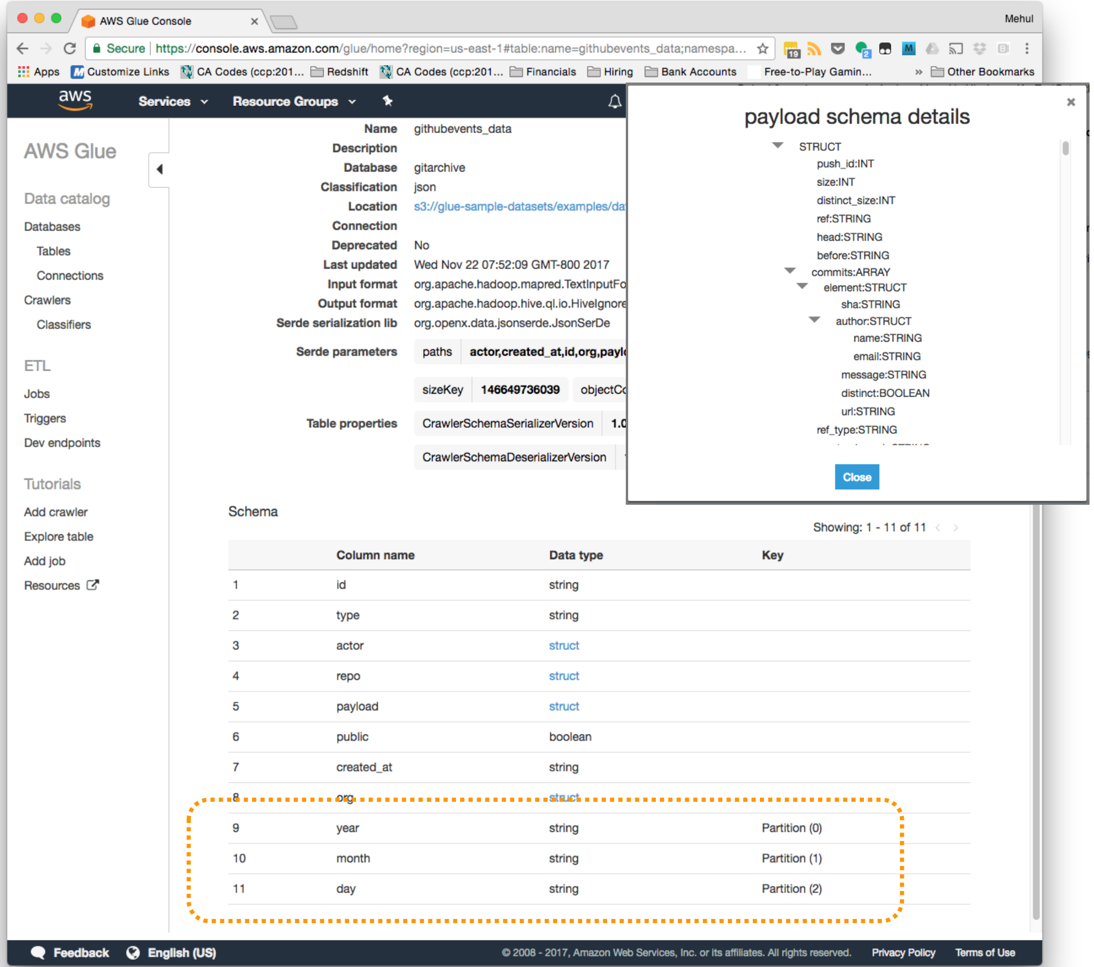
```
 "id":"2489651045","type":"CreateEvent","actor":{"id":665991,"login":"petroav","gravatar_id":"","url":"https://api.github.com/use\
rs/petroav","avatar_url":"https://avatars.githubusercontent.com/u/665991?"},"repo":{"id":28688495,"name":"petroav/6.828","url":"h\
ttps://api.github.com/repos/petroav/6.828"},"payload":{"ref":"master","ref_type":"branch","master_branch":"master","description":\
"Solution to homework and assignments from MIT's 6.828 (Operating Systems Engineering). Done in my spare time.","pusher_type":"us\
er"},"public":true,"created_at":"2015-01-01T15:00:00Z"}
{"id":"2489651051","type":"PushEvent","actor":{"id":3854017,"login":"rspt","gravatar_id":"","url":"https://api.github.com/users/r\
spt","avatar_url":"https://avatars.githubusercontent.com/u/3854017?"},"repo":{"id":28671719,"name":"rspt/rspt-theme","url":"https\
://api.github.com/repos/rspt/rspt-theme"},"payload":{"push_id":536863970,"size":1,"distinct_size":1,"ref":"refs/heads/master","he\
ad":"6b089eb4a43f728f0a594388092f480f2ecacfcd","before":"437c03652caa0bc4a7554b18d5c0a394c2f3d326","commits":[{"sha":"6b089eb4a43\
f728f0a594388092f480f2ecacfcd","author":{"email":"5c682c2d1ec4073e277f9ba9f4bdf07e5794dabe@rspt.ch","name":"rspt"},"message":"Fix\
 main header height on mobile","distinct":true,"url":"https://api.github.com/repos/rspt/rspt-theme/commits/6b089eb4a43f728f0a5943\
88092f480f2ecacfcd"}]},"public":true,"created_at":"2015-01-01T15:00:01Z"}
{"id":"2489651057","type":"WatchEvent","actor":{"id":6894991,"login":"SametSisartenep","gravatar_id":"","url":"https://api.github\
.com/users/SametSisartenep","avatar_url":"https://avatars.githubusercontent.com/u/6894991?"},"repo":{"id":2871998,"name":"visionm\
edia/debug","url":"https://api.github.com/repos/visionmedia/debug"},"payload":{"action":"started"},"public":true,"created_at":"20\
15-01-01T15:00:03Z","org":{"id":9285252,"login":"visionmedia","gravatar_id":"","url":"https://api.github.com/orgs/visionmedia","a\
vatar_url":"https://avatars.githubusercontent.com/u/9285252?"}}
{"id":"2489651091","type":"IssuesEvent","actor":{"id":6269456,"login":"yhoonkim","gravatar_id":"","url":"https://api.github.com/u\
sers/yhoonkim","avatar_url":"https://avatars.githubusercontent.com/u/6269456?"},"repo":{"id":28594770,"name":"yhoonkim/GraphBoard\
","url":"https://api.github.com/repos/yhoonkim/GraphBoard"},"payload":{"action":"opened","issue":{"url":"https://api.github.com/r\
epos/yhoonkim/GraphBoard/issues/27","labels_url":"https://api.github.com/repos/yhoonkim/GraphBoard/issues/27/labels{/name}","comm\
ents_url":"https://api.github.com/repos/yhoonkim/GraphBoard/issues/27/comments","events_url":"https://api.github.com/repos/yhoonk\
im/GraphBoard/issues/27/events","html_url":"https://github.com/yhoonkim/GraphBoard/issues/27","id":53221333,"number":27,"title":"\
Other readers can react to articles","user":{"login":"yhoonkim","id":6269456,"avatar_url":"https://avatars.githubusercontent.com/\
u/6269456?v=3","gravatar_id":"","url":"https://api.github.com/users/yhoonkim","html_url":"https://github.com/yhoonkim","followers\
_url":"https://api.github.com/users/yhoonkim/followers","following_url":"https://api.github.com/users/yhoonkim/following{/other_u\
ser}","gists_url":"https://api.github.com/users/yhoonkim/gists{/gist_id}","starred_url":"https://api.github.com/users/yhoonkim/st\
arred{/owner}{/repo}","subscriptions_url":"https://api.github.com/users/yhoonkim/subscriptions","organizations_url":"https://api.\
github.com/users/yhoonkim/orgs","repos_url":"https://api.github.com/users/yhoonkim/repos","events_url":"https://api.github.com/us\
ers/yhoonkim/events{/privacy}","received_events_url":"https://api.github.com/users/yhoonkim/received_events","type":"User","site_\
admin":false},"labels":[],"state":"open","locked":false,"assignee":null,"milestone":null,"comments":0,"created_at":"2015-01-01T15\
:00:06Z","updated_at":"2015-01-01T15:00:06Z","closed_at":null,"body":"- [ ] comment\n- [ ] recommendation\n- [ ] share\n- [ ] RSS\
\n\n- [ ] Join\n\n- [ ] Own board\n\n- [ ] Interview with people who want to archieve own thought within own writings."}},"public\
":true,"created_at":"2015-01-01T15:00:06Z"}
{"id":"2489651096","type":"PullRequestEvent","actor":{"id":10357835,"login":"mevlan","gravatar_id":"","url":"https://api.github.c\
om/users/mevlan","avatar_url":"https://avatars.githubusercontent.com/u/10357835?"},"repo":{"id":28668460,"name":"mevlan/script","\
url":"https://api.github.com/repos/mevlan/script"},"payload":{"action":"opened","number":3,"pull_request":{"url":"https://api.git\
hub.com/repos/mevlan/script/pulls/3","id":26743766,"html_url":"https://github.com/mevlan/script/pull/3","diff_url":"https://githu\
b.com/mevlan/script/pull/3.diff","patch_url":"https://github.com/mevlan/script/pull/3.patch","issue_url":"https://api.github.com/\
-UUU:**--F1  tmp.json      Top L1     (JavaScript) ----------------------------------------------------------------------------
Mark set
```

githubarchive.org

35+ event types

unique payload per event type

Step 1: Run crawler



Name	githubevents_data
Description	
Database	gitarchive
Classification	json
Location	s3://glue-sample-datasets/examples/da
Connection	
Deprecated	No
Last updated	Wed Nov 22 07:52:09 GMT-800 2017
Input format	org.apache.hadoop.mapred.TextInputFo
Output format	org.apache.hadoop.hive.ql.io.HiveIgnore
Serde serialization lib	org.openx.data.jsonserde.JsonSerDe
Serde parameters	paths actor,created_at,id,org,payl
	sizeKey 146649736039 objectC
Table properties	CrawlerSchemaSerializerVersion 1.0
	CrawlerSchemaDeserializerVersion

payload schema details

```
STRUCT
  push_id:INT
  size:INT
  distinct_size:INT
  ref:STRING
  head:STRING
  before:STRING
  commits:ARRAY
    element:STRUCT
      sha:STRING
      author:STRUCT
        name:STRING
        email:STRING
      message:STRING
      distinct:BOOLEAN
      url:STRING
  ref_type:STRING
```

Schema

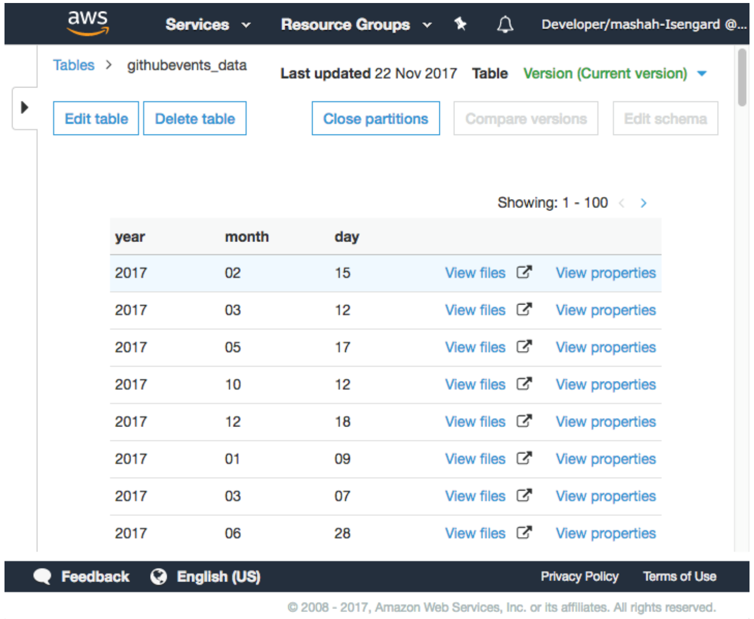
Showing: 1 - 11 of 11

	Column name	Data type	Key
1	id	string	
2	type	string	
3	actor	struct	
4	repo	struct	
5	payload	struct	
6	public	boolean	
7	created_at	string	
8	org	struct	
9	year	string	Partition (0)
10	month	string	Partition (1)
11	day	string	Partition (2)

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.

200+ fields

Groups files into Apache Hive-style partitions



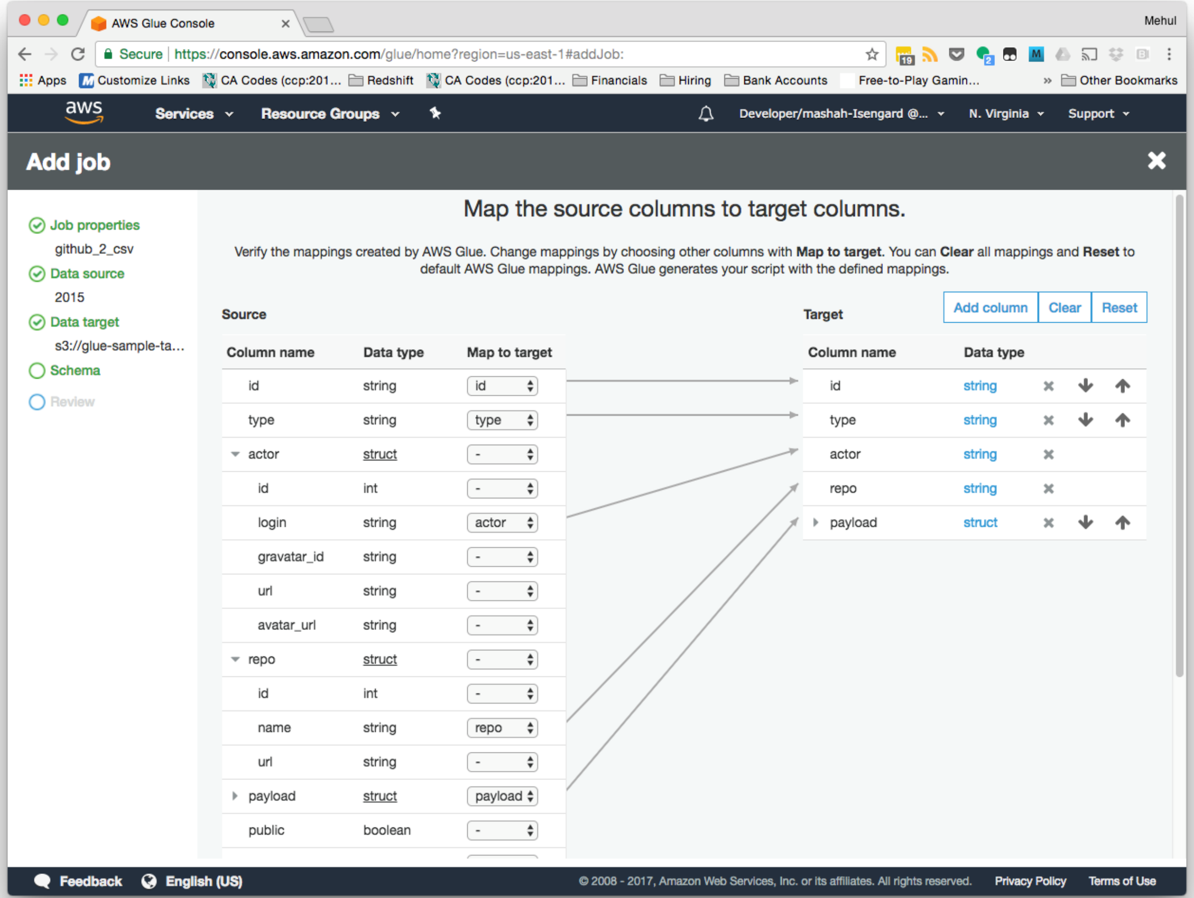
Tables > githubevents_data Last updated 22 Nov 2017 Table Version (Current version)

Showing: 1 - 100

year	month	day		
2017	02	15	View files	View properties
2017	03	12	View files	View properties
2017	05	17	View files	View properties
2017	10	12	View files	View properties
2017	12	18	View files	View properties
2017	01	09	View files	View properties
2017	03	07	View files	View properties
2017	06	28	View files	View properties

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Step 2: Specify mappings



Map the source columns to target columns.

Verify the mappings created by AWS Glue. Change mappings by choosing other columns with **Map to target**. You can **Clear** all mappings and **Reset** to default AWS Glue mappings. AWS Glue generates your script with the defined mappings.

Source

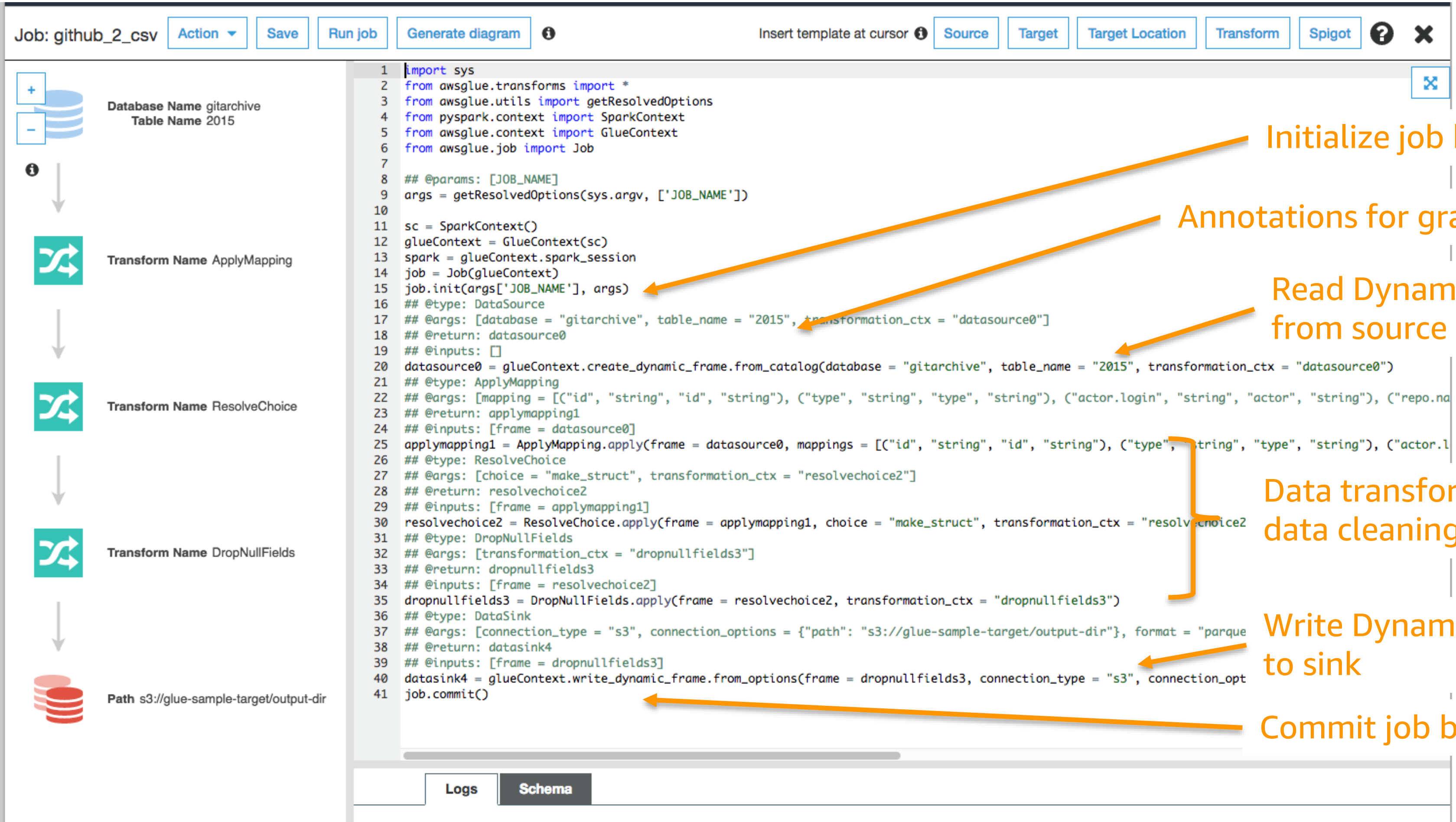
Column name	Data type	Map to target
id	string	id
type	string	type
actor	struct	-
id	int	-
login	string	actor
gravatar_id	string	-
url	string	-
avatar_url	string	-
repo	struct	-
id	int	-
name	string	repo
url	string	-
payload	struct	payload
public	boolean	-

Target

Add column | Clear | Reset

Column name	Data type
id	string
type	string
actor	string
repo	string
payload	struct

Anatomy of a generated script


Job: github_2_csv
Action
Save
Run job
Generate diagram
Insert template at cursor
Source
Target
Target Location
Transform
Spigot
Database Name gitarchive
Table Name 2015
Transform Name ApplyMapping
Transform Name ResolveChoice
Transform Name DropNullFields
Path s3://glue-sample-target/output-dir
Logs
Schema

```python
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
## @type: DataSource
## @args: [database = "gitarchive", table_name = "2015", transformation_ctx = "datasource0"]
## @return: datasource0
## @inputs: []
datasource0 = glueContext.create_dynamic_frame.from_catalog(database = "gitarchive", table_name = "2015", transformation_ctx = "datasource0")
## @type: ApplyMapping
## @args: [mapping = [("id", "string", "id", "string"), ("type", "string", "type", "string"), ("actor.login", "string", "actor", "string"), ("repo.na
## @return: applymapping1
## @inputs: [frame = datasource0]
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [("id", "string", "id", "string"), ("type", "string", "type", "string"), ("actor.l
## @type: ResolveChoice
## @args: [choice = "make_struct", transformation_ctx = "resolvechoice2"]
## @return: resolvechoice2
## @inputs: [frame = applymapping1]
resolvechoice2 = ResolveChoice.apply(frame = applymapping1, choice = "make_struct", transformation_ctx = "resolvechoice2
## @type: DropNullFields
## @args: [transformation_ctx = "dropnullfields3"]
## @return: dropnullfields3
## @inputs: [frame = resolvechoice2]
dropnullfields3 = DropNullFields.apply(frame = resolvechoice2, transformation_ctx = "dropnullfields3")
## @type: DataSink
## @args: [connection_type = "s3", connection_options = {"path": "s3://glue-sample-target/output-dir"}, format = "parque
## @return: datasink4
## @inputs: [frame = dropnullfields3]
datasink4 = glueContext.write_dynamic_frame.from_options(frame = dropnullfields3, connection_type = "s3", connection_opt
job.commit()
```

Initialize job bookmark

Annotations for graphical DAG

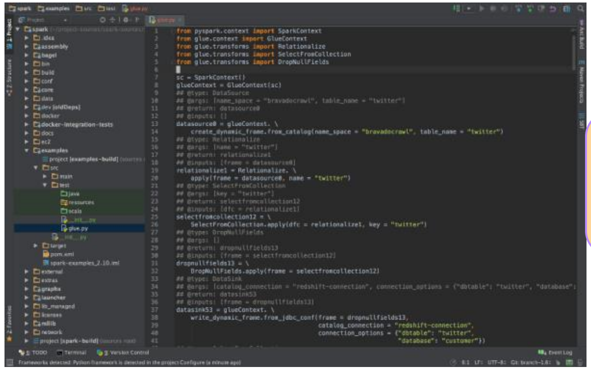
Read Dynamic Frame from source

Data transformation + data cleaning functions

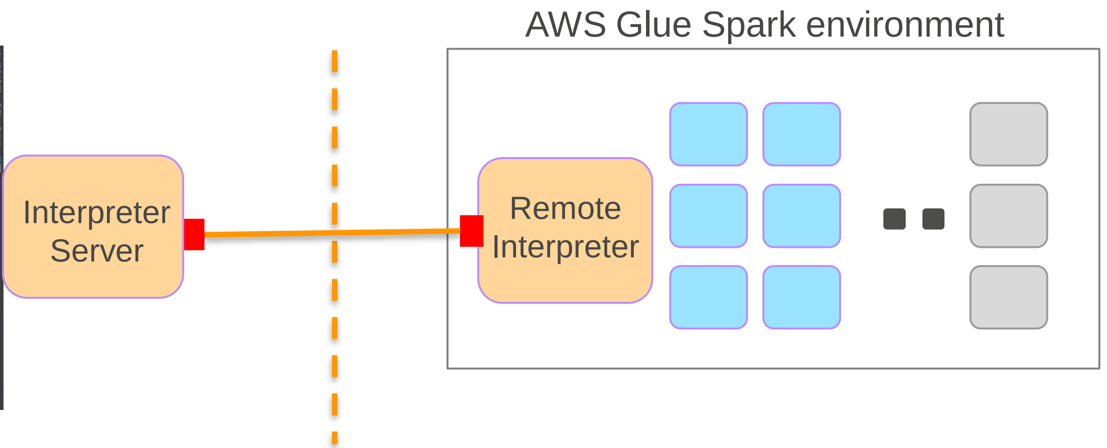
Write Dynamic Frame to sink

Commit job bookmark

Step 3: Edit + Test with Dev-Endpoints



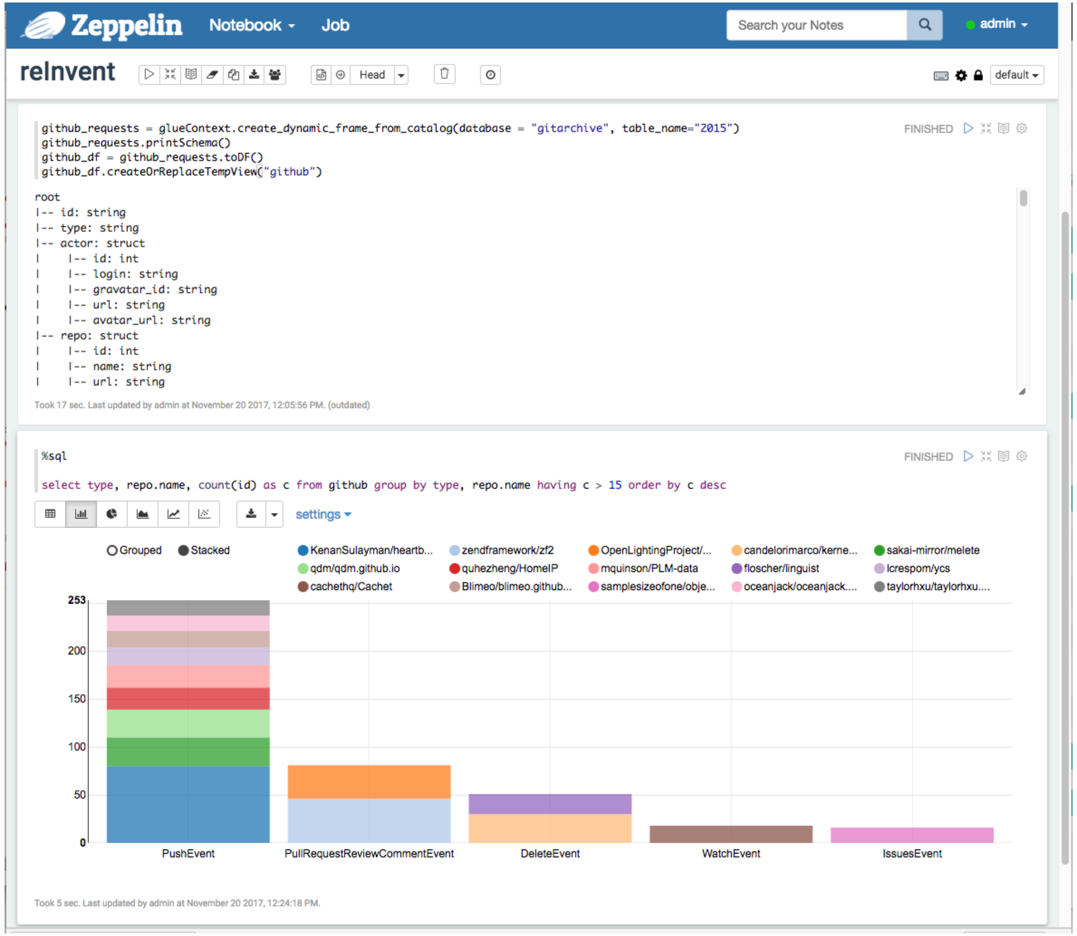
```python
from pyspark.context import SparkContext
from glue.context import GlueContext
from glue.transforms import Relationalize
from glue.transforms import SelectFromCollection
from glue.transforms import DropNullFields

sc = SparkContext()
glueContext = GlueContext(sc)
## @type: DataSource
## @args: [name_space = "bravadocrawl", table_name = "twitter"]
## @return: datasource0
## @inputs: []
datasource0 = glueContext. \
    create_dynamic_frame.from_catalog(name_space = "bravadocrawl", table_name = "twitter")
## @type: Relationalize
## @args: [name = "twitter"]
## @return: relationalize1
## @inputs: [frame = datasource0]
relationalize1 = Relationalize. \
    apply(frame = datasource0, name = "twitter")
## @type: SelectFromCollection
## @args: [key = "twitter"]
## @return: selectfromcollection12
## @inputs: [dfc = relationalize1]
selectfromcollection12 = \
    SelectFromCollection.apply(dfc = relationalize1, key = "twitter")
## @type: DropNullFields
## @args: []
## @return: dropnullfields13
## @inputs: [frame = selectfromcollection12]
dropnullfields13 = \
    DropNullFields.apply(frame = selectfromcollection12)
## @type: DataSink
## @args: [catalog_connection = "redshift-connection", connection_options = {"dbtable": "twitter", "database":
## @return: datasink53
## @inputs: [frame = dropnullfields13]
datasink53 = glueContext. \
    write_dynamic_frame.from_jdbc_conf(frame = dropnullfields13,
                                       catalog_connection = "redshift-connection",
                                       connection_options = {"dbtable": "twitter",
                                                             "database": "customer"})
```



Connect your IDE to an AWS Glue development endpoint.

Environment to **interactively develop,** debug, and test ETL code.

Step 3: Explore and experiment with data



```
github_requests = glueContext.create_dynamic_frame_from_catalog(database = "gitarchive", table_name="2015")
github_requests.printSchema()
github_df = github_requests.toDF()
github_df.createOrReplaceTempView("github")
```

```
root
|-- id: string
|-- type: string
|-- actor: struct
|    |-- id: int
|    |-- login: string
|    |-- gravatar_id: string
|    |-- url: string
|    |-- avatar_url: string
|-- repo: struct
|    |-- id: int
|    |-- name: string
|    |-- url: string
```

Took 17 sec. Last updated by admin at November 20 2017, 12:05:56 PM. (outdated)

```
%sql
select type, repo.name, count(id) as c from github group by type, repo.name having c > 15 order by c desc
```

● KenanSulayman/heartb...
● zendframework/zf2
● OpenLightingProject/...
● candelorimarco/kerne...
● sakai-mirror/melete
● qdm/qdm.github.io
● quhezheng/HomeIP
● mquinson/PLM-data
● floscher/linguist
● lcrespom/ycs
● cachethq/Cachet
● Blimeo/blimeo.github...
● samplesizeofone/obje...
● oceanjack/oceanjack....
● taylorhxu/taylorhxu....
253
200
150
100
50
0
PushEvent
PullRequestReviewCommentEvent
DeleteEvent
WatchEvent
IssuesEvent

Took 5 sec. Last updated by admin at November 20 2017, 12:24:18 PM.

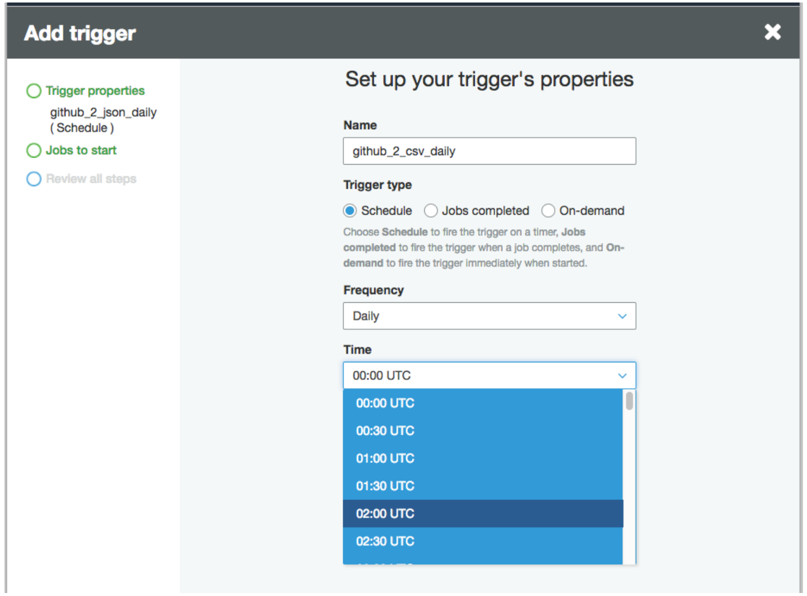
Connect your notebook (e.g. **Zeppelin**) to an AWS Glue development endpoint.

Interactively experiment and **explore** datasets and data sources

Deploy to production

- Push scripts to S3
- Create or register with ETL job

Step 4: Schedule a job



Add trigger

Trigger properties
github_2_json_daily
(Schedule)
Jobs to start
Review all steps

Set up your trigger's properties

Name: github_2_csv_daily

Trigger type: [✓ Schedule] Jobs completed On-demand

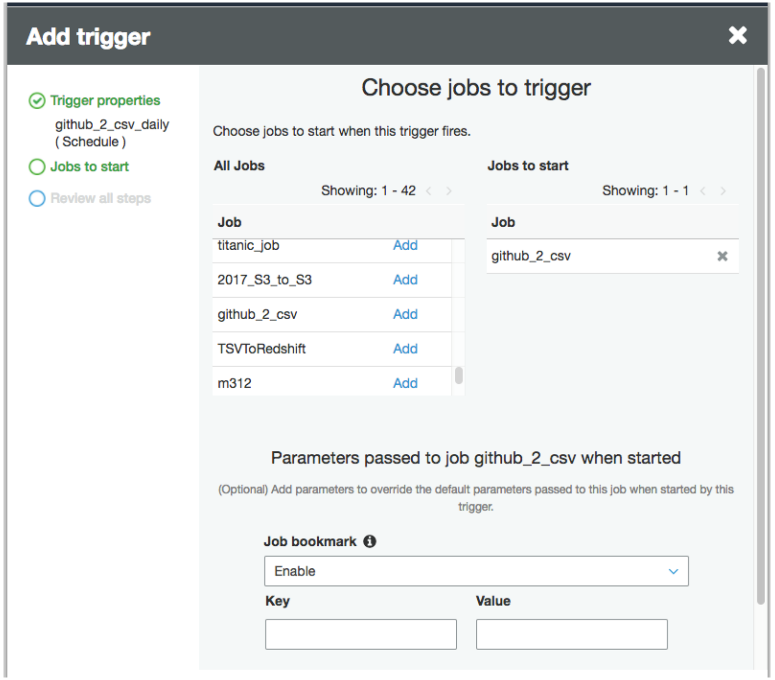
Choose **Schedule** to fire the trigger on a timer, **Jobs completed** to fire the trigger when a job completes, and **On-demand** to fire the trigger immediately when started.

Frequency: Daily

Time: 00:00 UTC

- 00:00 UTC
- 00:30 UTC
- 01:00 UTC
- 01:30 UTC
- 02:00 UTC
- 02:30 UTC

several event types



Add trigger

Trigger properties
github_2_csv_daily
(Schedule)
Jobs to start
Review all steps

Choose jobs to trigger

Choose jobs to start when this trigger fires.

All Jobs — Showing: 1 - 42

Job	
titanic_job	Add
2017_S3_to_S3	Add
github_2_csv	Add
TSVToRedshift	Add
m312	Add

Jobs to start — Showing: 1 - 1

Job
github_2_csv

Parameters passed to job github_2_csv when started

(Optional) Add parameters to override the default parameters passed to this job when started by this trigger.

Job bookmark: Enable

Key	Value

pass parameters

Serverless job execution

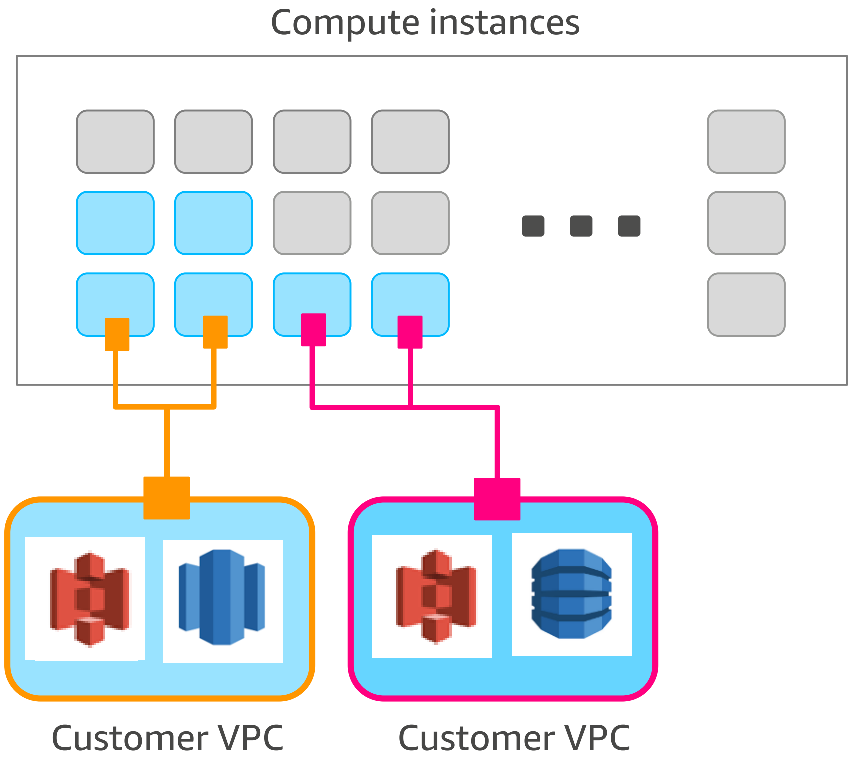
No need to provision, configure, or manage servers

Auto-configure VPC & role-based access
security & isolation preserved

Customers can specify job capacity (DPU)

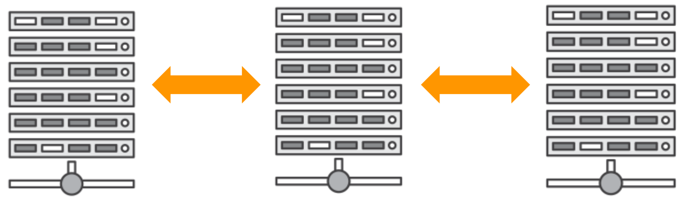
Automatically scale resources

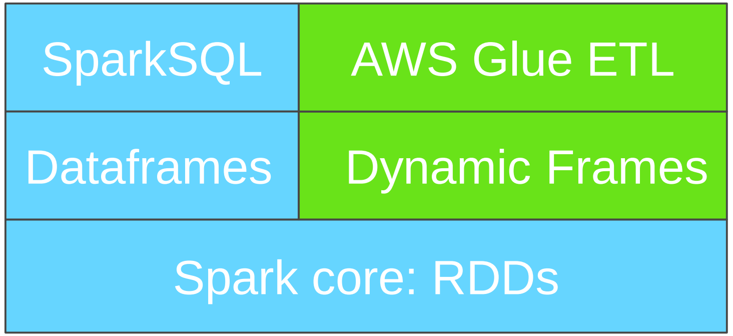
Only pay for the resources you consume
per-second billing (10-minute min)



Under the hood: customize AWS Glue scripts

Apache Spark and AWS Glue ETL





What is Apache Spark?

Parallel, scale-out data processing engine

Fault-tolerance built-in

Flexible interface: Python scripting, SQL

Rich eco-system: ML, Graph, analytics, …

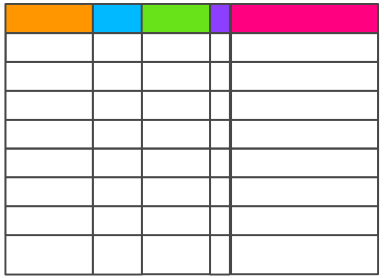
AWS Glue ETL libraries

Integration: Data Catalog, job orchestration, code-generation, job bookmarks, S3, RDS

ETL transforms, more connectors & formats

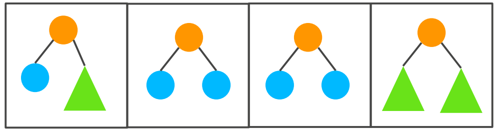
New data structure: Dynamic Frames

Dataframes and Dynamic Frames



Dataframes

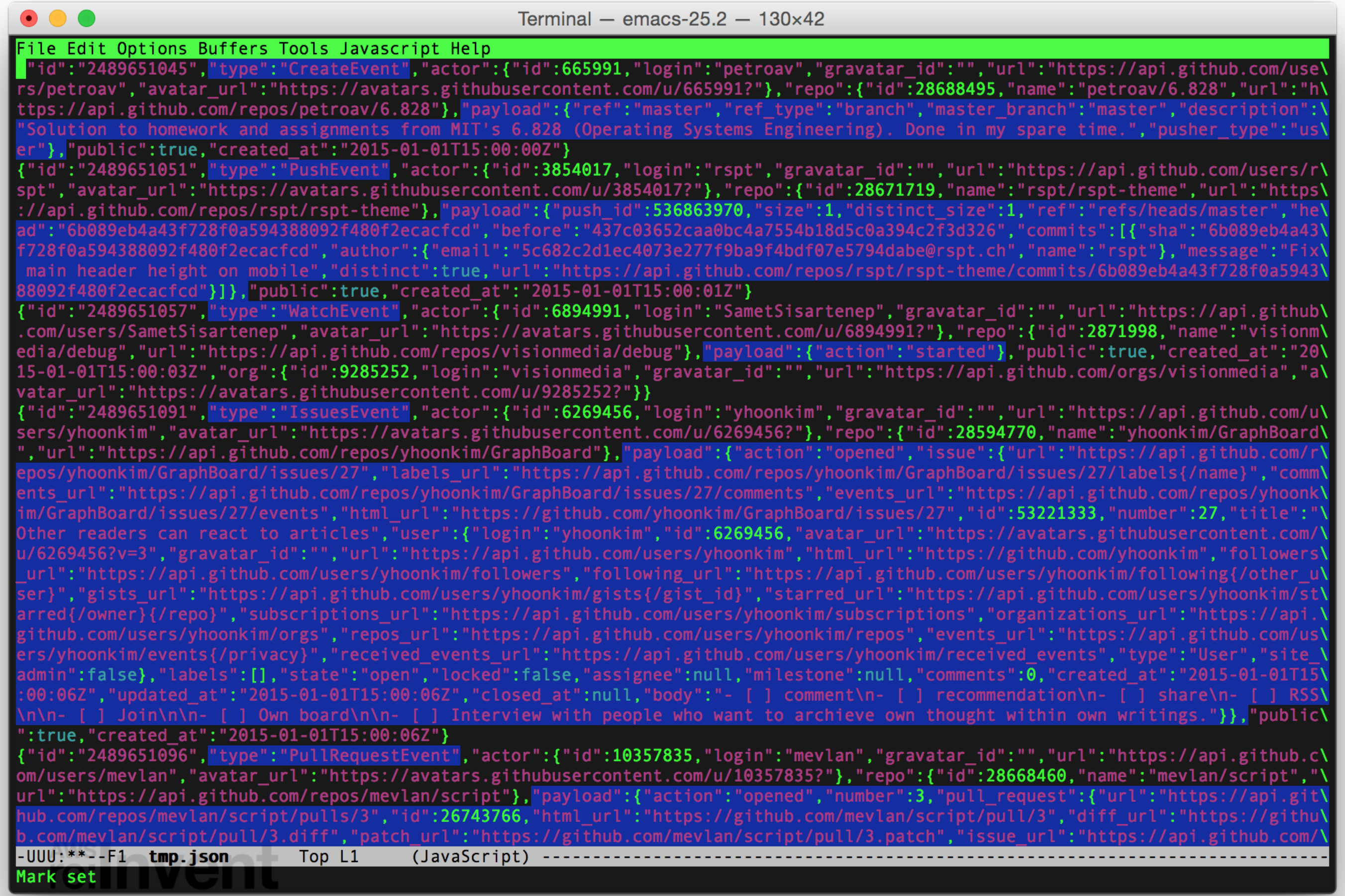
- Core data structure for SparkSQL
- Like structured tables
  - Need schema up-front
  - Each row has same structure
- Suited for SQL-like analytics



Dynamic Frames

- Like dataframes for ETL
- Designed for processing **semi-structured** data, e.g. JSON, Avro, Apache logs ...

Public GitHub timeline is …


Terminal — emacs-25.2 — 130×42
File Edit Options Buffers Tools Javascript Help

```
{"id":"2489651045","type":"CreateEvent","actor":{"id":665991,"login":"petroav","gravatar_id":"","url":"https://api.github.com/use\
rs/petroav","avatar_url":"https://avatars.githubusercontent.com/u/665991?"},"repo":{"id":28688495,"name":"petroav/6.828","url":"h\
ttps://api.github.com/repos/petroav/6.828"},"payload":{"ref":"master","ref_type":"branch","master_branch":"master","description":\
"Solution to homework and assignments from MIT's 6.828 (Operating Systems Engineering). Done in my spare time.","pusher_type":"us\
er"},"public":true,"created_at":"2015-01-01T15:00:00Z"}
{"id":"2489651051","type":"PushEvent","actor":{"id":3854017,"login":"rspt","gravatar_id":"","url":"https://api.github.com/users/r\
spt","avatar_url":"https://avatars.githubusercontent.com/u/3854017?"},"repo":{"id":28671719,"name":"rspt/rspt-theme","url":"https\
://api.github.com/repos/rspt/rspt-theme"},"payload":{"push_id":536863970,"size":1,"distinct_size":1,"ref":"refs/heads/master","he\
ad":"6b089eb4a43f728f0a594388092f480f2ecacfcd","before":"437c03652caa0bc4a7554b18d5c0a394c2f3d326","commits":[{"sha":"6b089eb4a43\
f728f0a594388092f480f2ecacfcd","author":{"email":"5c682c2d1ec4073e277f9ba9f4bdf07e5794dabe@rspt.ch","name":"rspt"},"message":"Fix\
 main header height on mobile","distinct":true,"url":"https://api.github.com/repos/rspt/rspt-theme/commits/6b089eb4a43f728f0a5943\
88092f480f2ecacfcd"}]},"public":true,"created_at":"2015-01-01T15:00:01Z"}
{"id":"2489651057","type":"WatchEvent","actor":{"id":6894991,"login":"SametSisartenep","gravatar_id":"","url":"https://api.github\
.com/users/SametSisartenep","avatar_url":"https://avatars.githubusercontent.com/u/6894991?"},"repo":{"id":2871998,"name":"visionm\
edia/debug","url":"https://api.github.com/repos/visionmedia/debug"},"payload":{"action":"started"},"public":true,"created_at":"20\
15-01-01T15:00:03Z","org":{"id":9285252,"login":"visionmedia","gravatar_id":"","url":"https://api.github.com/orgs/visionmedia","a\
vatar_url":"https://avatars.githubusercontent.com/u/9285252?"}}
{"id":"2489651091","type":"IssuesEvent","actor":{"id":6269456,"login":"yhoonkim","gravatar_id":"","url":"https://api.github.com/u\
sers/yhoonkim","avatar_url":"https://avatars.githubusercontent.com/u/6269456?"},"repo":{"id":28594770,"name":"yhoonkim/GraphBoard\
","url":"https://api.github.com/repos/yhoonkim/GraphBoard"},"payload":{"action":"opened","issue":{"url":"https://api.github.com/r\
epos/yhoonkim/GraphBoard/issues/27","labels_url":"https://api.github.com/repos/yhoonkim/GraphBoard/issues/27/labels{/name}","comm\
ents_url":"https://api.github.com/repos/yhoonkim/GraphBoard/issues/27/comments","events_url":"https://api.github.com/repos/yhoonk\
im/GraphBoard/issues/27/events","html_url":"https://github.com/yhoonkim/GraphBoard/issues/27","id":53221333,"number":27,"title":"\
Other readers can react to articles","user":{"login":"yhoonkim","id":6269456,"avatar_url":"https://avatars.githubusercontent.com/\
u/6269456?v=3","gravatar_id":"","url":"https://api.github.com/users/yhoonkim","html_url":"https://github.com/yhoonkim","followers\
_url":"https://api.github.com/users/yhoonkim/followers","following_url":"https://api.github.com/users/yhoonkim/following{/other_u\
ser}","gists_url":"https://api.github.com/users/yhoonkim/gists{/gist_id}","starred_url":"https://api.github.com/users/yhoonkim/st\
arred{/owner}{/repo}","subscriptions_url":"https://api.github.com/users/yhoonkim/subscriptions","organizations_url":"https://api.\
github.com/users/yhoonkim/orgs","repos_url":"https://api.github.com/users/yhoonkim/repos","events_url":"https://api.github.com/us\
ers/yhoonkim/events{/privacy}","received_events_url":"https://api.github.com/users/yhoonkim/received_events","type":"User","site_\
admin":false},"labels":[],"state":"open","locked":false,"assignee":null,"milestone":null,"comments":0,"created_at":"2015-01-01T15\
:00:06Z","updated_at":"2015-01-01T15:00:06Z","closed_at":null,"body":"- [ ] comment\n- [ ] recommendation\n- [ ] share\n- [ ] RSS\
\n\n- [ ] Join\n\n- [ ] Own board\n\n- [ ] Interview with people who want to archieve own thought within own writings."}},"public\
":true,"created_at":"2015-01-01T15:00:06Z"}
{"id":"2489651096","type":"PullRequestEvent","actor":{"id":10357835,"login":"mevlan","gravatar_id":"","url":"https://api.github.c\
om/users/mevlan","avatar_url":"https://avatars.githubusercontent.com/u/10357835?"},"repo":{"id":28668460,"name":"mevlan/script","\
url":"https://api.github.com/repos/mevlan/script"},"payload":{"action":"opened","number":3,"pull_request":{"url":"https://api.git\
hub.com/repos/mevlan/script/pulls/3","id":26743766,"html_url":"https://github.com/mevlan/script/pull/3","diff_url":"https://githu\
b.com/mevlan/script/pull/3.diff","patch_url":"https://github.com/mevlan/script/pull/3.patch","issue_url":"https://api.github.com/\
-UUU:**--F1  tmp.json      Top L1     (JavaScript) ----------------------------------------------------------------------------
Mark set
```

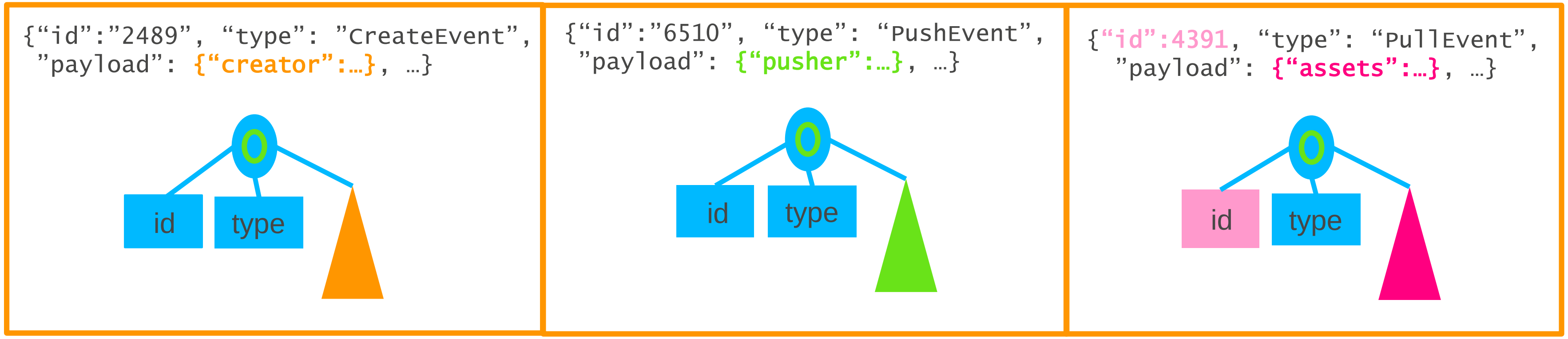
semi-structured

35+ event types

payload structure and size varies by event type

Dynamic Frame internals

Dynamic Records

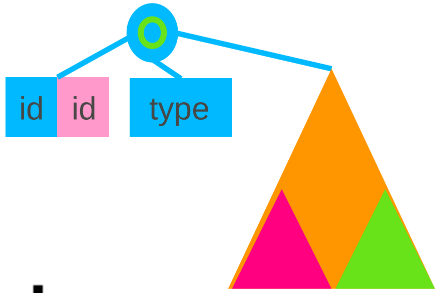


```
{"id":"2489", "type": "CreateEvent",
 "payload": {"creator":…}, …}
```

```
{"id":"6510", "type": "PushEvent",
 "payload": {"pusher":…}, …}
```

```
{"id":4391, "type": "PullEvent",
 "payload": {"assets":…}, …}
```

Dynamic Frame Schema



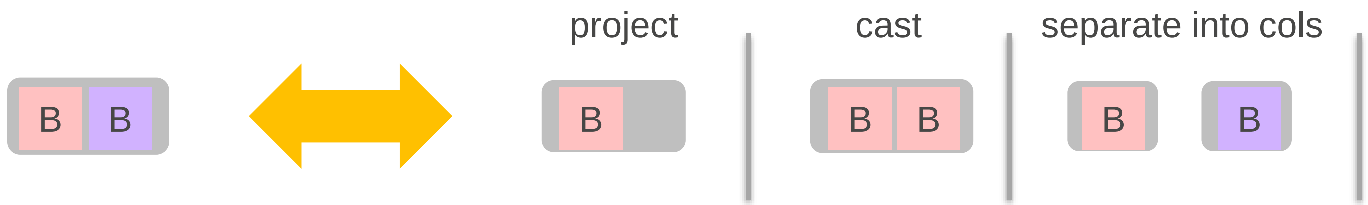
schema per-record, no up-front schema needed

- Easy to restructure, tag, modify
- Can be more compact than dataframe rows
- Many flows can be done in single-pass

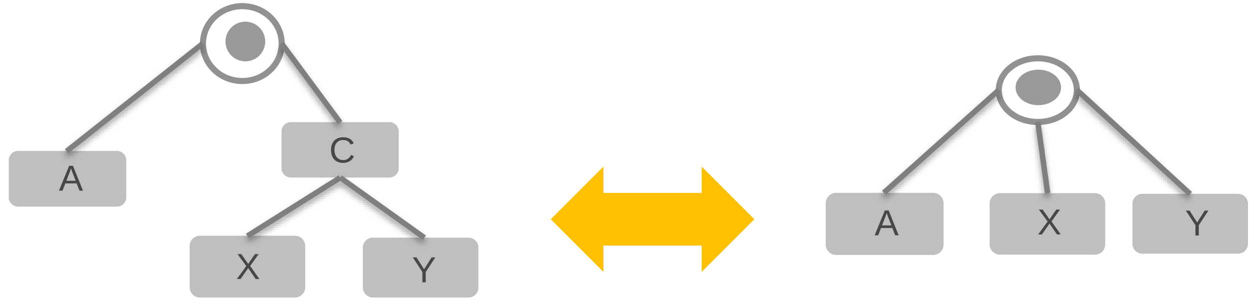
Dynamic Frame transforms

15+ transforms out-of-the box

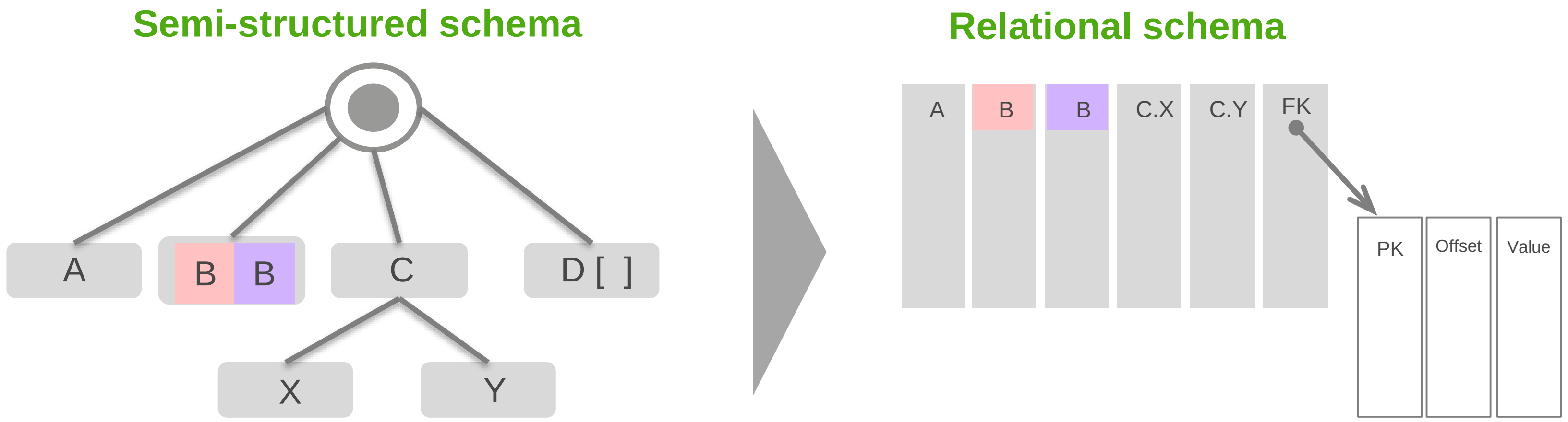
ResolveChoice()



ApplyMapping()



Relationalize() transform



Transforms and **adds new** columns, types, and tables on-the-fly

Tracks **keys** and **foreign keys** across runs

SQL on the relational schema is orders of **magnitude faster** than JSON processing

Useful AWS Glue transforms

toDF(): Convert to a Dataframe

Spigot(): Sample data of any Dynamic Frame to S3

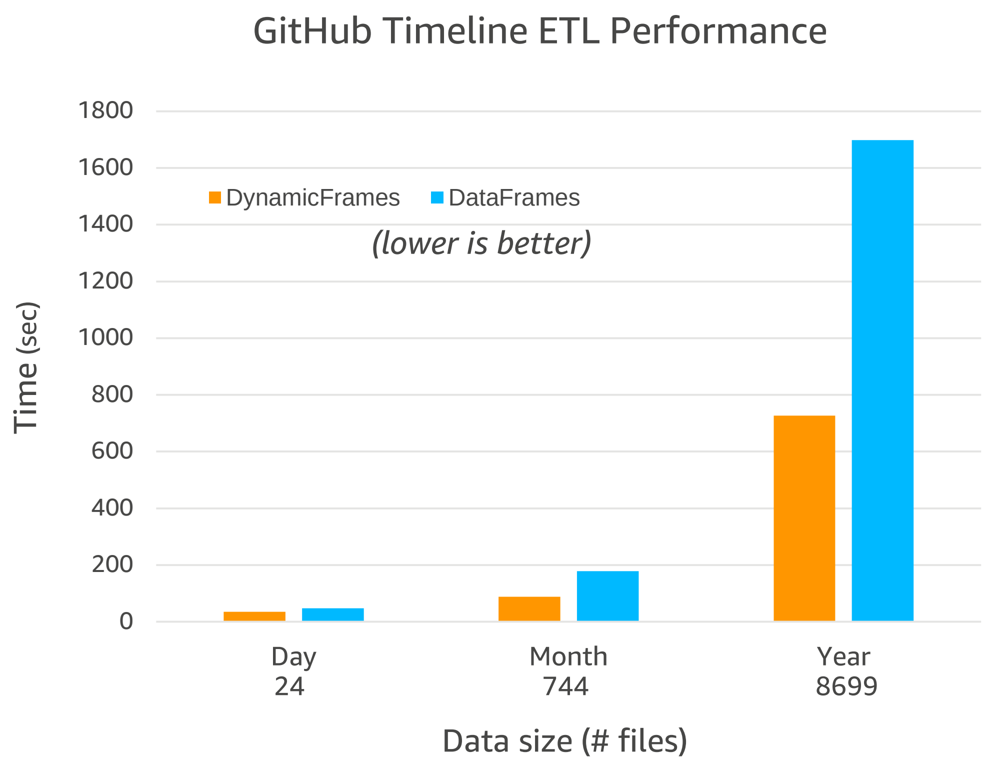
Unbox(): Parse string column as given format into Dynamic Frame

Filter(), Map(): Apply Python UDFs to Dynamic Frames

Join(): Join two Dynamic Frames

And more ….

Performance: AWS Glue ETL



Configuration

10 DPUs

Apache Spark 2.1.1

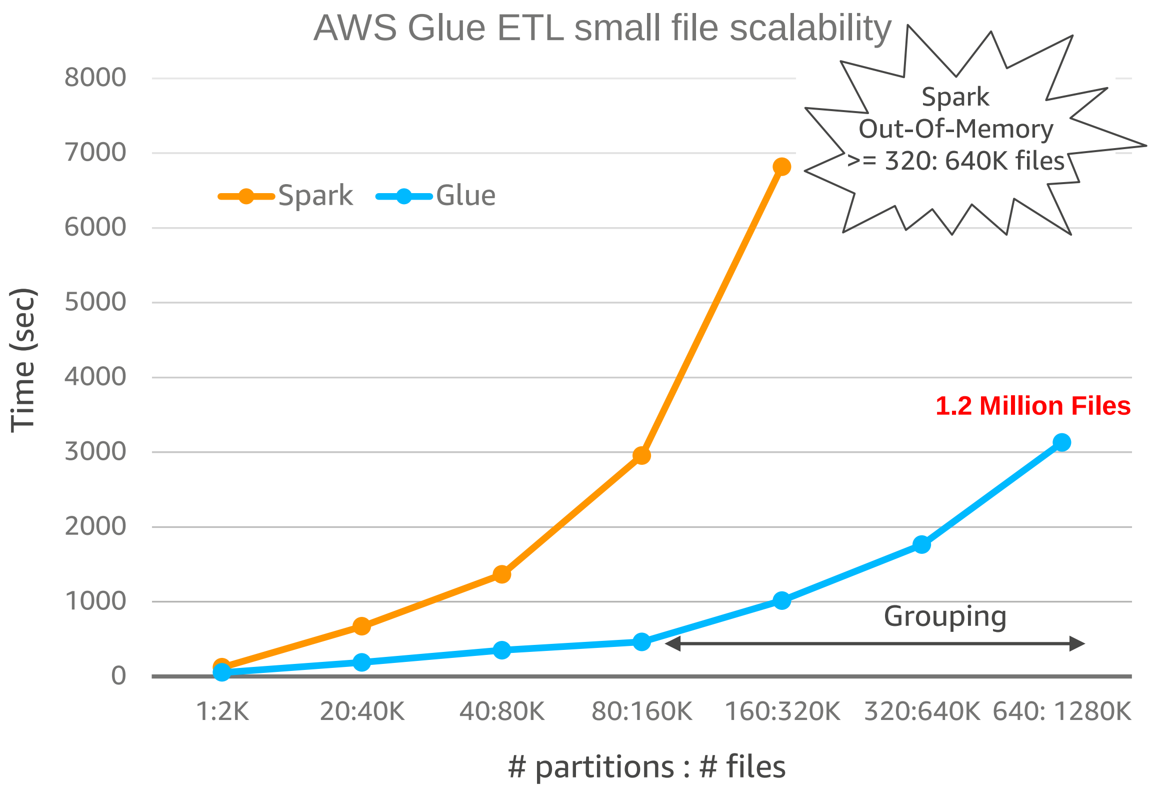
Workload

JSON to CSV

Filter for Pull events

On average: 2x performance improvement

Performance: Lots of small files



Lots of small files, e.g. Kinesis Firehose

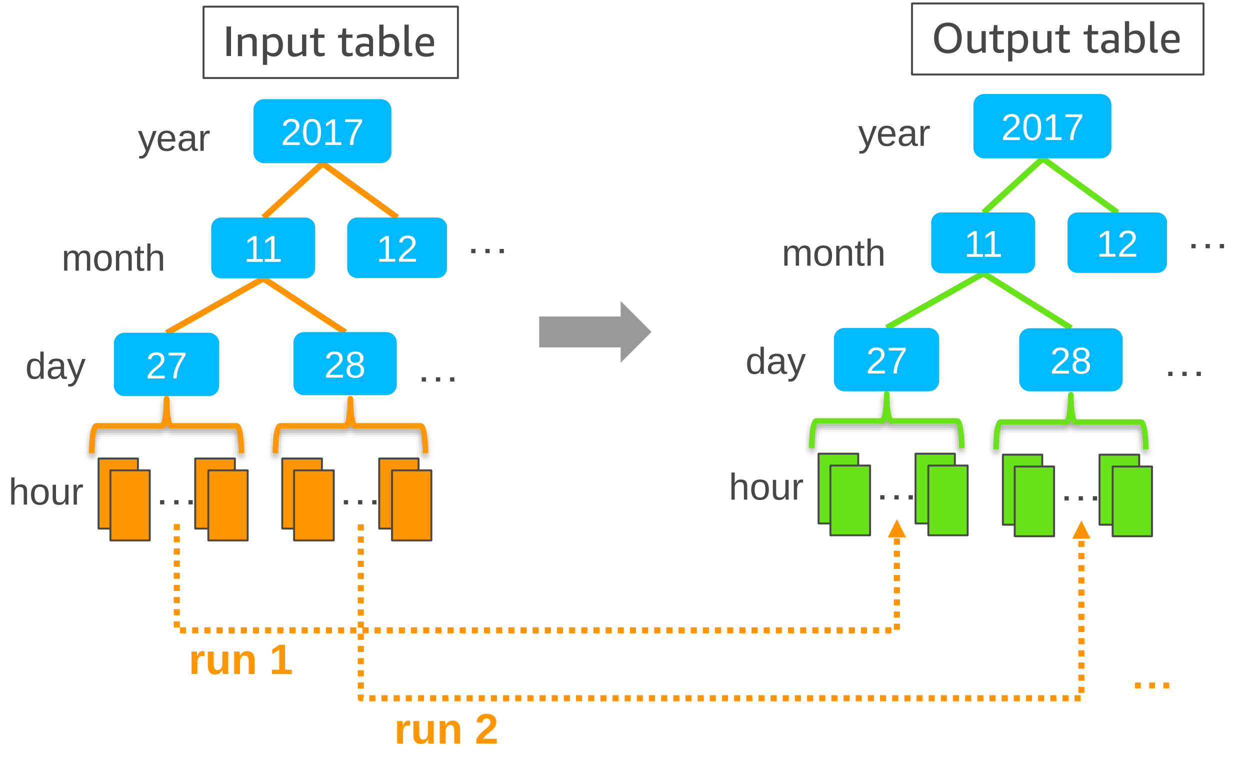
Vanilla Apache Spark (2.1.1) overheads

- Must reconstruct partitions (2-pass)
- Too many tasks: task per file
- Scheduling & memory overheads

AWS Glue Dynamic Frames

- Integration with Data Catalog
- Automatically group files per task
- Rely on crawler statistics

Job bookmark example



Periodically run a job

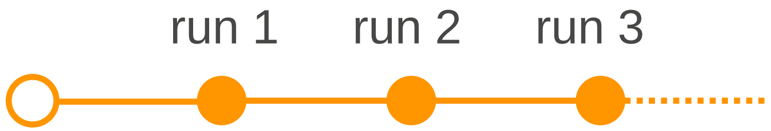
avoid reprocessing previous input

avoid generating duplicate output

Job bookmarks

Bookmarks are per-job checkpoints that track the work done in previous runs.

They persist the state of sources, transforms, and sinks on each run.



Examples uses:

Process githubarchive files daily

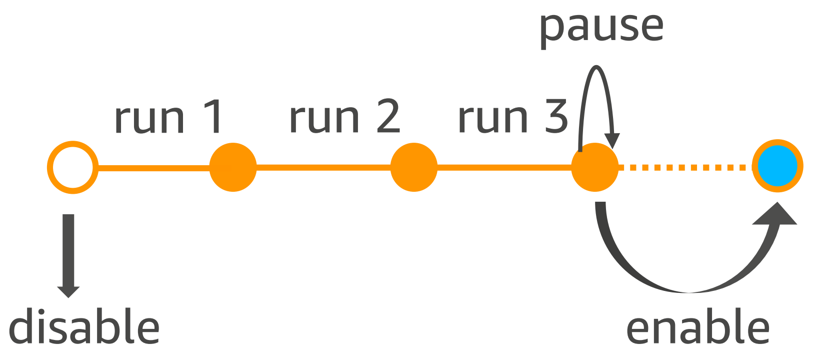
Process Firehose files hourly

Track timestamps or primary keys in DBs

Track generated foreign keys for normalization

Job bookmark options

Option	Behavior
Enable	Pick up from where you left off
Disable	Ignore and process the entire dataset every time
Pause	Temporarily disable advancing the bookmark



Examples:

Enable: Process the newest githubarchive partition

Disable: Process the entire githubarchive table

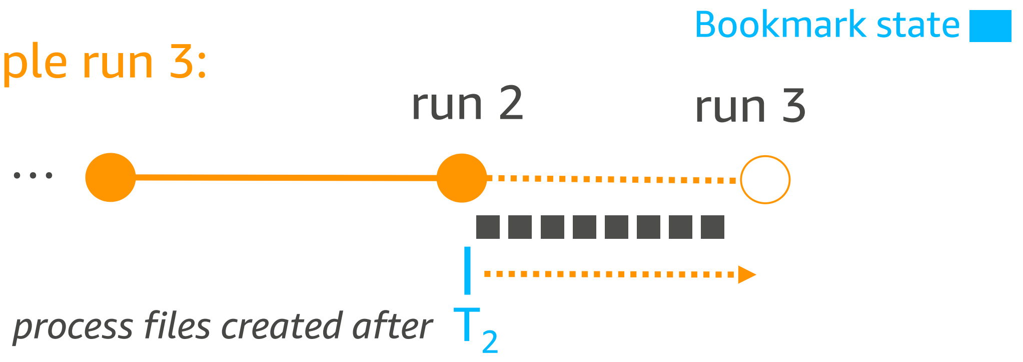
Pause: Process the previous githubarchive partition

Job bookmark internals

Example run 3:

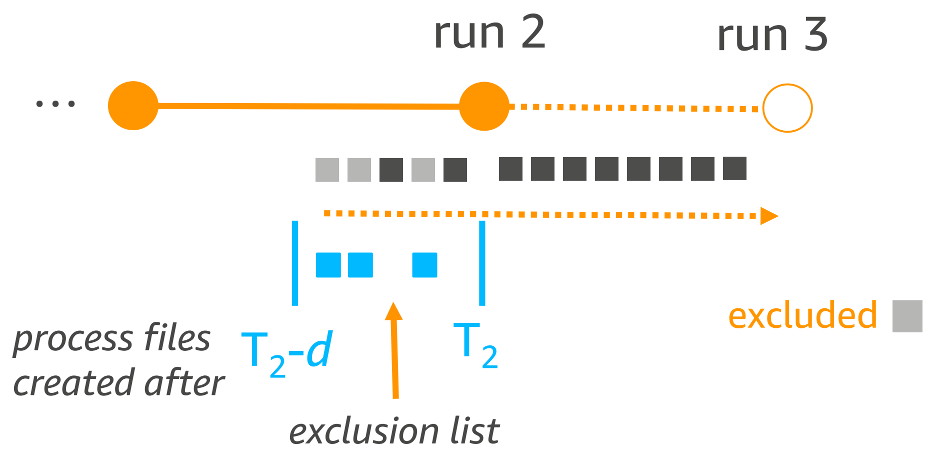
How do we avoid space blowup?

Use timestamps to filter already processed input



But S3 is eventually consistent?

Maintain exclusion list of files created in *inconsistency window* (size *d*) prior to start.



Wrap Up

4 steps to build a production ETL flow

AWS Glue features

- Dynamic frames
- Job bookmarks

AWS Glue Announcements

Scala support

New regions: Asia Pacific (Tokyo) & EU (Ireland)

Merck – customer testimonial

Keith Smola

Global Operations Management, Merck & Co.

AWS re:Invent

FOR MORE THAN A CENTURY, MERCK HAS BEEN INVENTING TO SOLVE SOME OF THE GREATEST CHALLENGES TO PEOPLE'S HEALTH AND WELL-BEING AROUND THE WORLD.

HEADQUARTERS
Kenilworth, NJ, U.S.A. operating in more than 140 countries

MRK

Merck & Co., Inc. is our legal name and is listed on the New York Stock Exchange under the symbol "MRK."

68,000

EMPLOYEES
approximately 68,000 worldwide (as of 12/31/16)

BUSINESSES
Prescription medicines, Vaccines, Biologic therapies, Animal Health products

$

2016 REVENUES
$39.8 billion, 54% of sales come from outside the United States

2016 R&D EXPENSE
$10.1 billion; 24 product pipeline programs in late-stage development

MERCK

Problem

We were challenged to quickly align extended data and metrics to a Enterprise Software Delivery Management system. One capability of the system is Environment Management, which is used to provide the knowledge regarding environments, what assets make them up and how they align to different software lifecycle efforts. Other capabilities include deployment management, delivery planning and scope management.

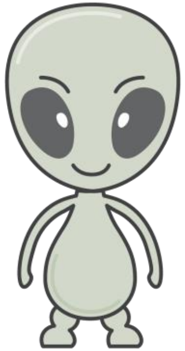
Limited integration with the Enterprise Software Delivery Management System

BI tools require a data source with application lifecycle context

Major projects require fast alignment to Enterprise Software Delivery Management System

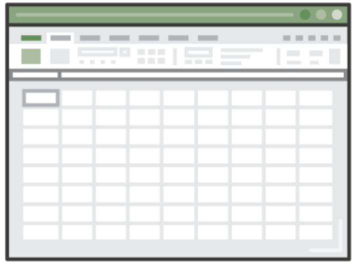
Challenges

Short timeline to produce a data layer.



Resources are not data scientists or ETL developers.

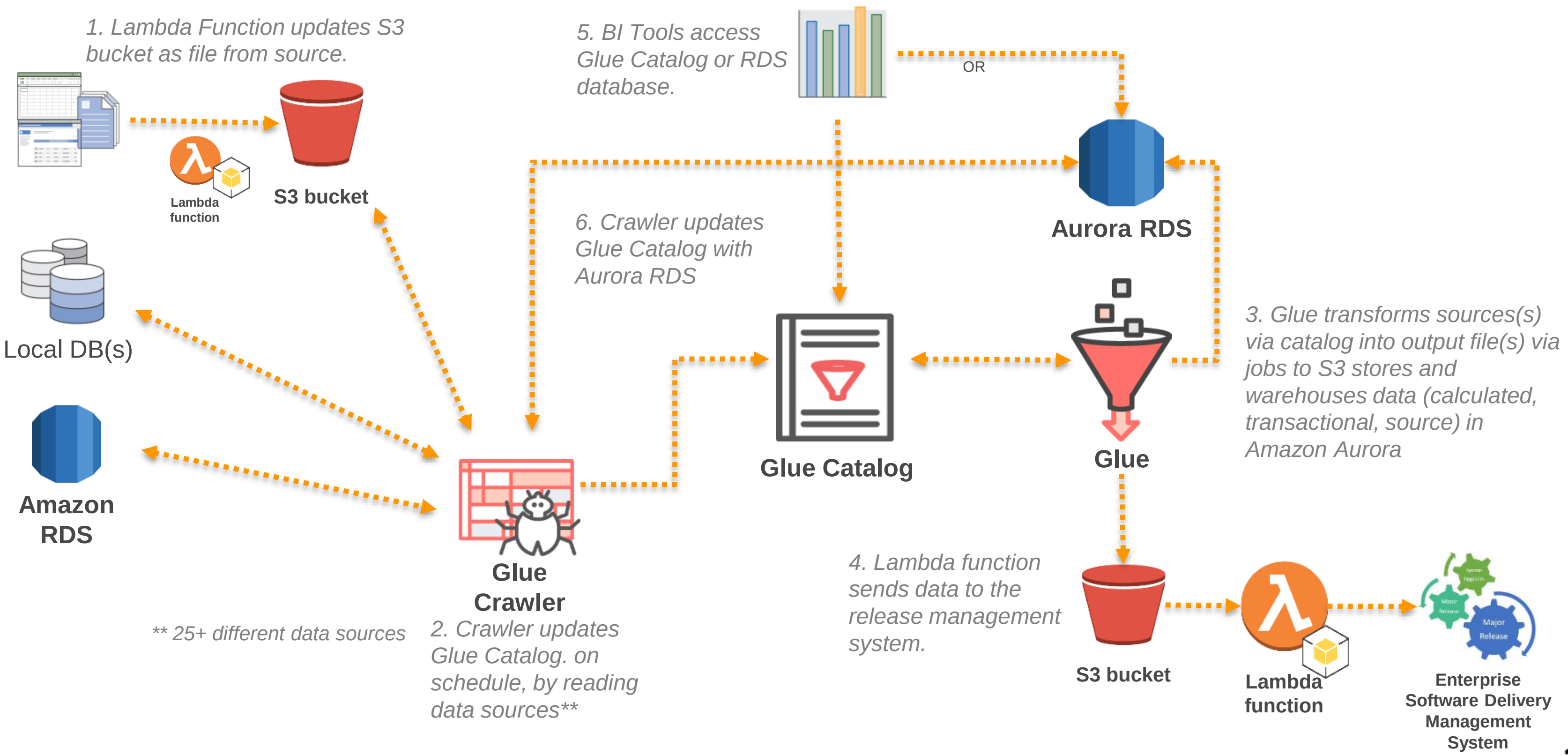
Established support does not go beyond AWS.



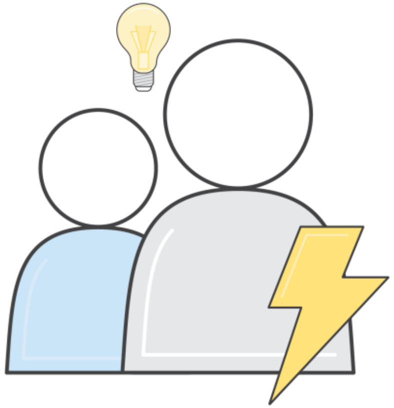
Data is in different spreadsheets or existing databases

Catalog vs. warehousing

Architecture


1. Lambda Function updates S3 bucket as file from source.
Lambda function
S3 bucket
5. BI Tools access Glue Catalog or RDS database.
OR
Aurora RDS
6. Crawler updates Glue Catalog with Aurora RDS
Local DB(s)
Amazon RDS
Glue Catalog
Glue
3. Glue transforms sources(s) via catalog into output file(s) via jobs to S3 stores and warehouses data (calculated, transactional, source) in Amazon Aurora
Glue Crawler
** 25+ different data sources
2. Crawler updates Glue Catalog. on schedule, by reading data sources**
4. Lambda function sends data to the release management system.
S3 bucket
Lambda function
System Upgrades
Minor Release
Major Release
Enterprise Software Delivery Management System

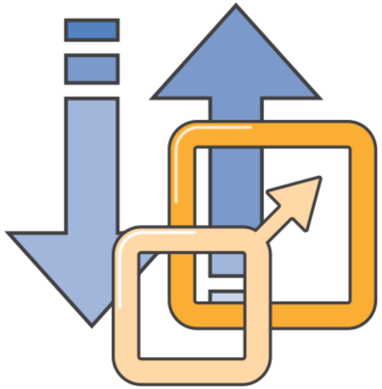
Value



Leverage existing resources.

Glue was quick to learn.

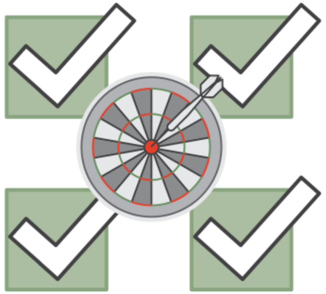
No servers means no additional resources to manage, procure or maintain.



Easy to adapt to new data sources and scale availability within short timeframe.

Provided a single source of data (aligned and calculated).

Enabled a scalable data layer/lake



Projects were able to use the data via our Release Management System.

Q&A

THANK YOU!