**Data Engineer Coding Challenge - Architectural Solution**

This entire process can be completed utilizing AWS Glue, Python, and Tableau by manual creation within AWS Glue to setup the framework. The framework would consist of databases, jobs, triggers, and python scripts. Tableau would be used to connect to AWS to visualize the data.

**Databases:**

1. pizza\_source

2. pizza\_prepared

**Jobs:**

## Download layer (1 job)

1**. pizza\_download\_transactions\_data job**

- triggered every 15 minutes

- connects to REST API

- checks for last\_updated.txt file in S3 bucket (if this file is not here then job has never been run and will pull all orders)

- reads date(timestamp in UTC) from last\_updated.txt file

- if date is newer than max(order\_date) from REST API JSON then job will be completed

- if date is older than max(order\_date) from REST API JSON then job will download all transactions to S3 after date from last\_updated.txt file

-- if new data is downloaded to S3 it will trigger the pizza\_source\_transactions\_data job

## Source Layer (3 jobs)

1. **pizza\_source\_customer\_data job**

- ingests customer data from a RDBMS into a customer\_table

- job will run once since we are assuming no customers are added or removed over the lifetime of this pipeline

- table schema:

• customer\_id (integer): primary key

• name (string): customer name

• address (string): street address

• city (string): city

• state (string): state

• zip\_code (string): five-digit US zip code

2. **pizza\_source\_transactions\_data job**

- ingests ordering data from REST API JSON and appends data to existing data into the transactions\_table

- this job will be triggered if pizza\_download\_transactions\_data job completes successfully and if a new json file has been downloaded

- table schema:

• order\_id (integer): unique identifier for the order

• customer\_id (integer): unique identifier for customer (foreign key to the customers table)

• type (string): "cheese", "pepperoni", "supreme", "meat lover", or "veggie" - the flavor of pizza ordered

• qty (integer): - the number of pizzas ordered

• retail\_price (float): the total retail price of the order, including taxes

• order\_date (timestamp in UTC): the time that the order was placed

3. **pizza\_source\_population\_data job**

- ingests adult population by US state from census.gov API for the past 12 months into a state\_population\_table

- triggered monthly and will be a full overwrite for past 12 months

- table schema:

• state (string): state

• population (integer): population

## Prepared Layer (2 jobs)

1. **pizza\_prepared\_customer\_pop\_data job**

- join customer\_table and state\_population\_table by state to create the customer\_population\_table

- this job will be triggered monthly if pizza\_source\_population\_data job is successful

2. **pizza\_prepared\_data\_warehouse job**

- join transactions\_table and customer\_population\_table by customer\_id to create the data\_warehouse table

- this job will be triggered if pizza\_source\_transactions\_data job completes succesfully

**Triggers:**

1. pizza\_download\_transactions\_trigger

- will trigger every 15 minutes

2. pizza\_source\_transactions\_trigger

- will trigger if pizza\_download\_transactions\_data job completes successfully and REST API JSON data is downloaded to S3

3. pizza\_data\_warehouse\_trigger

- will trigger if pizza\_source\_transactions\_data job completes succesfully

4. pizza\_source\_population\_trigger

- will trigger monthly

5. pizza\_prepared\_customer\_pop\_trigger

- will trigger if pizza\_source\_population\_data job completes successfully

**Python scripts:**

- View attached for code samples

**Tableau:**

- connect Tableau to AWS for data visualization to pull data from the semantic layer (data warehouse)

- Tableau can schedule a refresh from the data warehouse daily at 2AM or more often if needed