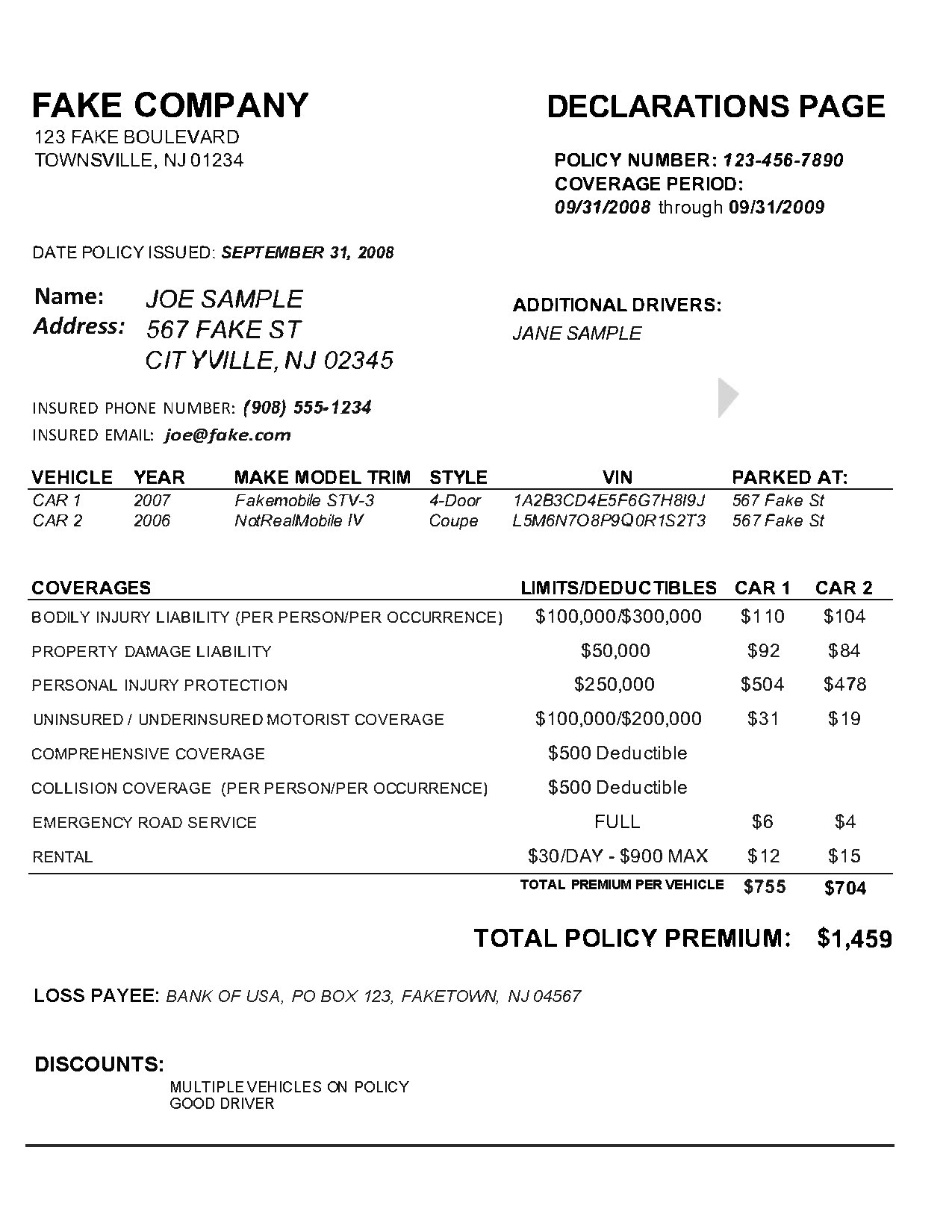
# Accelerating Auto Insurance Quote Generation

Using an auto insurance online quote generator can be a long painstaking process as several different pieces of information are required. The goal of this project is to transition the process to a short and seamless format; only requiring a previous declaration page to be uploaded by a company’s customer.

In this project, we are attempting to demonstrate the use of Textract, a machine learning service provided by AWS, to automate the online quote generating process in serverless architecture. Textract will collect the required information from a declaration form and save the data as CSV files in S3. From here, a quote can be generated through the use of companies algorithm and be sent to through user as either a text message or email.

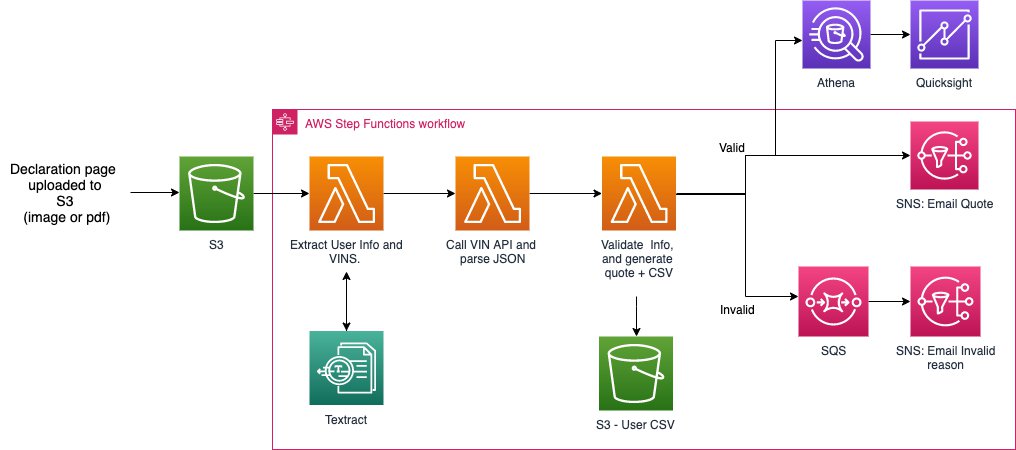
## Inspected Document:



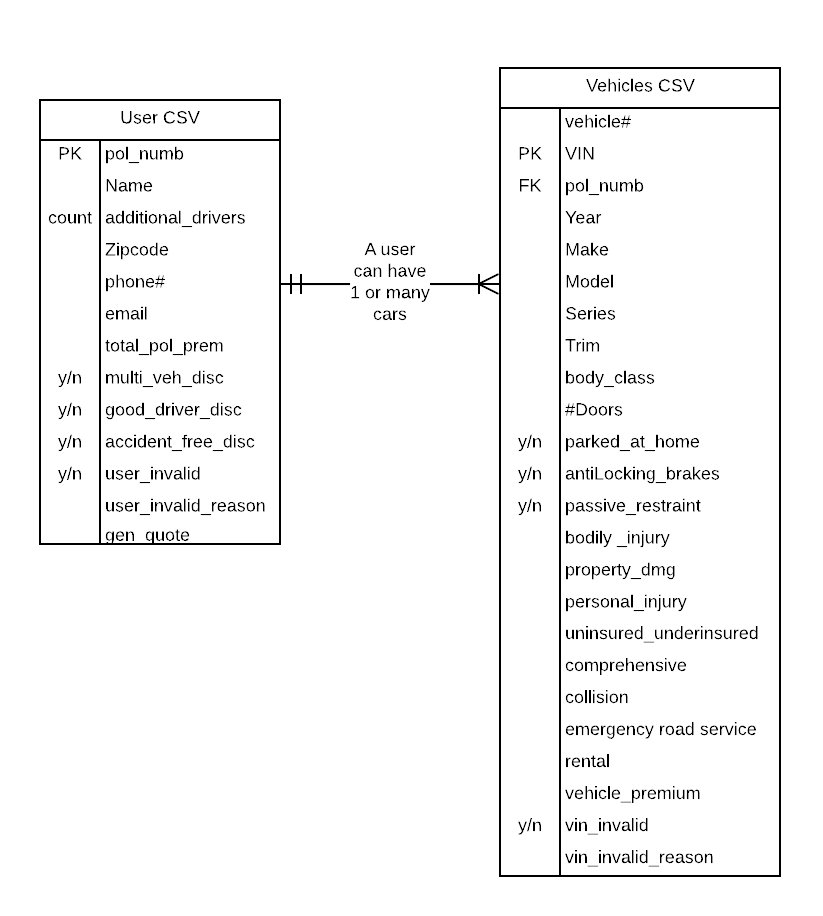
**Assumptions**:

1. A quote is generated using: zip code, loss payee, additional drivers, current discounts, vehicle year make model style, if the vehicle is parked at home, and the current coverage limits and premiums.
2. Coverages and vehicles are identified as table 1 and 2 respectively by textract.
3. Total policy premium, name, address, email, loss payee are identified as a form.
4. The only coverages provided are those located on the document.
5. Only discounts that are offered are: good driver, multiple vehicle, accident free.
6. Each driver can have at least two cars.
7. The Address provided is considered as the home address.

## Workflow summary and architecture:



1. Amazon S3 will be used to store the declaration page uploaded. An upload will trigger a step function workflow.
2. The first lambda in the workflow will use Amazon Textract to extract the tables and forms from the document. This text is parsed and data on the driver and vehicles are stored in a dictionary.
3. The second lambda will take the VIN data from the first lambda and use a VIN API (<https://vpic.nhtsa.dot.gov/api/>) to add additional vehicle information to our collected data.
4. The third lambda will validate if the driver information such as zipcode and vehicle information is valid.If so, a quote is generated and sent to the user though SNS, additionally the data is split between driver and vehicle information and saved in two different CSV’s.
5. If at any point the information is invalid, a message is sent to the user through SNS containing the reasons why their page is invalid, and a message is sent to SQS with the invalid reasons. The file are still saved in CSV’s as the information and are approiately tagged as invalid. We do this as the data can still be analyzed.
6. Using Athena, the CSV’s located in S3 can be queried with SQL and inner joined by policy number as demonstrated on the graph below. Lastly, this data can visualized using AWS Quicksight.



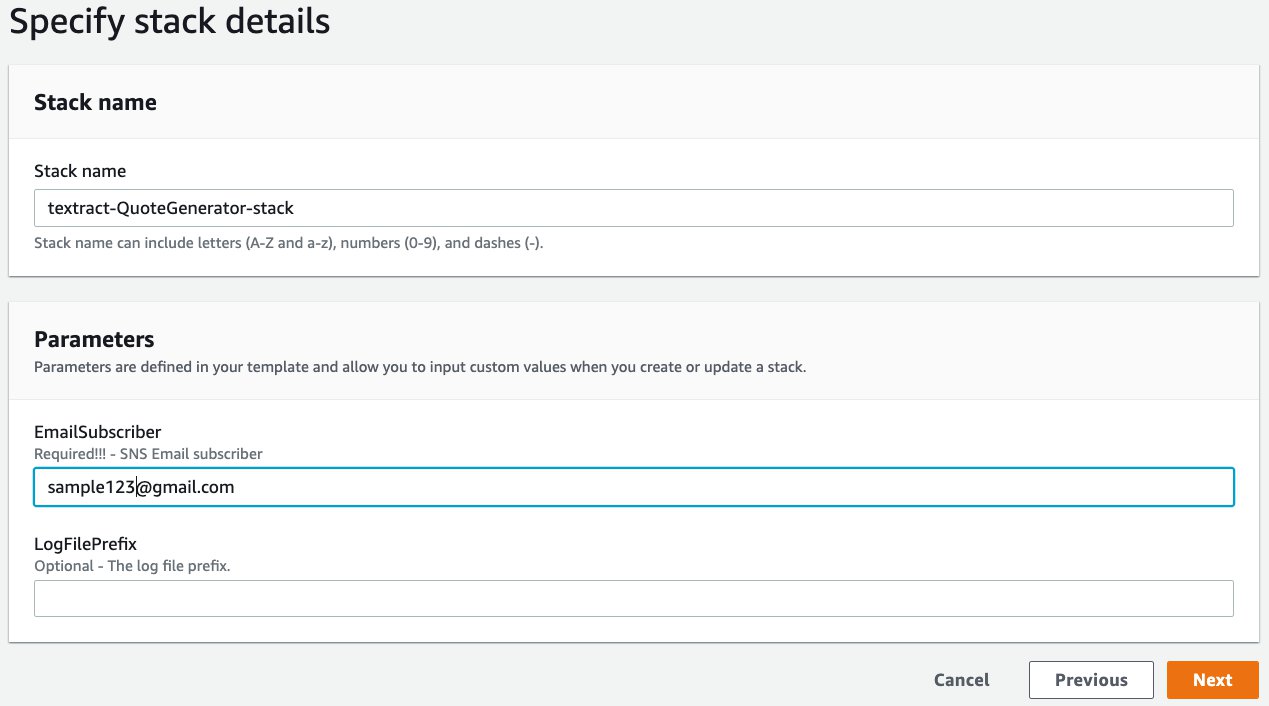
## Deploying the solution:

If any of the hyperlinks fail, all documents can be found here:

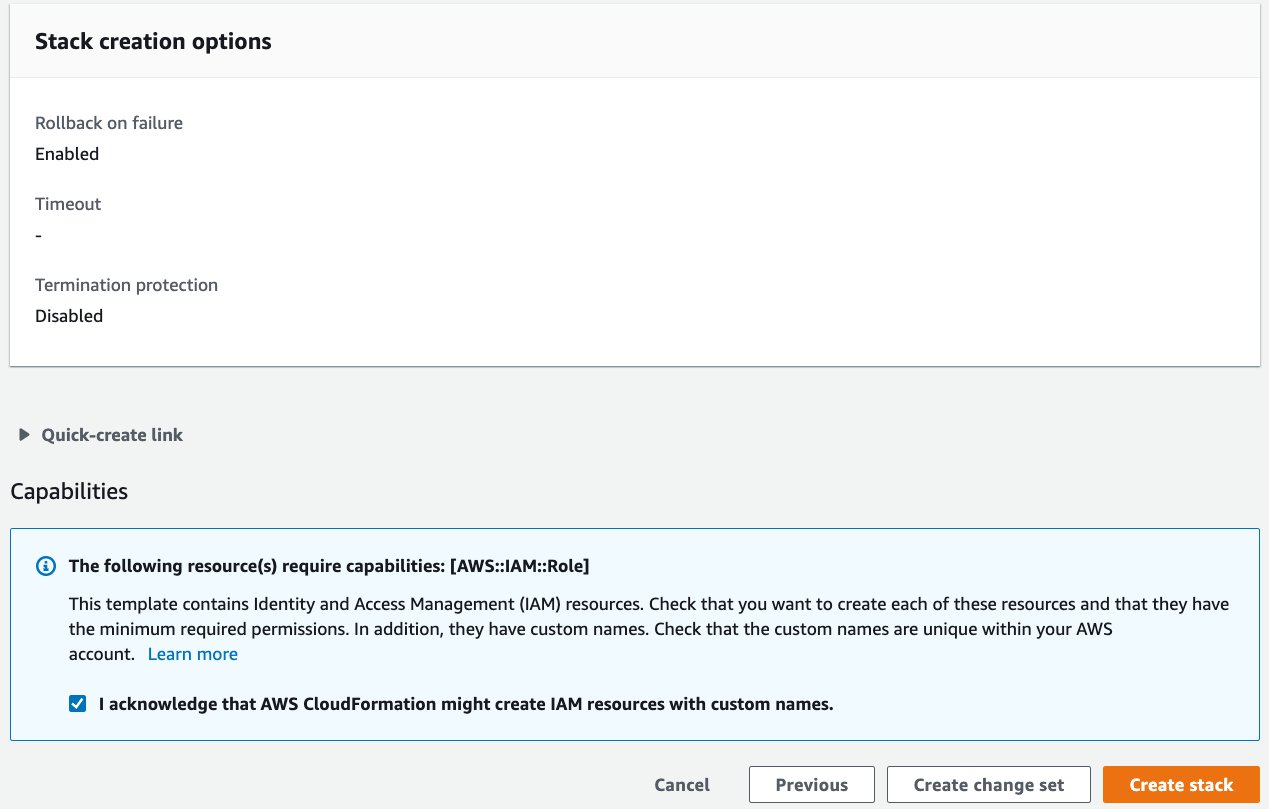
### Launching your AWS CloudFormation stack

With an AWS CloudFormation stack, we can easily deploy the solution. When deployed, the stack will create three SQS queues, two SNS topics, three S3 buckets, three Lambda functions, a Cloud Trail, a CloudWatch rule, and a Identity and Access Management (IAM) role.

1. Download the [CloudFormation template](https://github.com/RinoDinoRino/soldo_project/blob/master/cfn/autoPolicyStack.yaml) and save it to your local disk.
2. Log into the AWS Management Console with your IAM username and password.
3. Go to CloudFormation either using the lookup bar or service list. When there, click Create stack.
4. On the Create stack page,  select Upload a template file and upload the CloudFormation template downloaded from Step 1. Choose Next.
5. On the next page, make sure to include a Stack name and enter an email under the EmailSubscriber parameter and click Next. If no email is provided, the stack will fail to deploy.



1. On the Configure stack options page, keep everything as default and choose Next.
2. On the Review page, scroll to the bottom and select the checkbox “I acknowledge that AWS CloudFormation might create IAM resources with custom names.” Click Create stack.

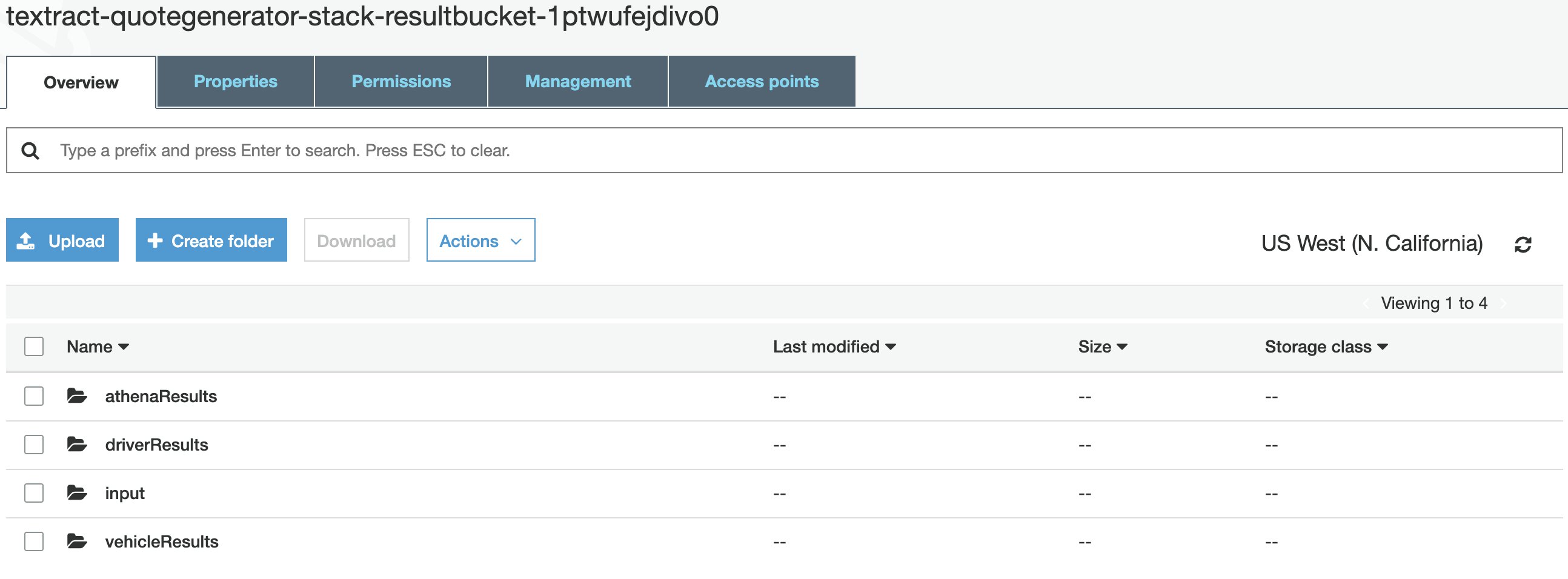


1. Wait for the stack to complete deploying, this might take a couple of minutes. Completion will be indicated by a “CREATE\_COMPLETE” status on the left menu. You can view all the resources provisioned by looking though the Resource tab.
2. The email subscriber will also recieve two emails from a no-reply email. Make sure to click the confirm subscription link in both emails.

## Executing a quote generation workflow

### Generating a quote with a valid declaration page

1. Download the following [auto insurance declaration page PNG](https://github.com/RinoDinoRino/soldo_project/blob/master/testPages/successCase.png) on your local disk.
2. Using the services tab on the top of the console, locate S3.
3. In S3, locate the bucket containing “resultbucket” in its name. Alternatively, you can find out the name of the S3 bucket from the outputs tab of the CloudFormation stack.



1. Within the bucket’s “input” prefix, upload the successCase PNG from step-1 of this section. The process to generate a quote may take a minute.
2. Examine the CSV files located in the “driverResults” and “vehicleResults” prefix’s of the bucket. The first bucket will contain data relevant to the driver, while the latter is relevant to the vehicle.
3. Check the inbox of the email provided in the creation of the stack. There, you should have recieved an email from Amazon SNS with a generated quote. This may take a couple of minutes to send.

### Generating a quote with an invalid declaration page

For declaration pages that fail to validate, an email will be sent containing the reasons why it failed. Additionally, the data will still be saved in S3. To replicate this use case, download this [declaration page PNG](https://github.com/RinoDinoRino/soldo_project/blob/master/testPages/failCase.png) and use it for this section. This form contains an invalid zip code, and both vehicles have invalid VIN numbers.

1. Download the image above and save it to your local drive.
2. In the “input” prefix of the result bucket, upload the image. It may take a minute for the resulting CSV’s to be uploaded.
3. Check the “driverResults” and “vehicleResults” prefix of the bucket. New CSV files should have been uploaded, both ending with “\_invalid.”
4. Check the inbox of the email provided. There should be an email from Amazon SNS containing the invalid reasons. This may take a couple of minutes to send.

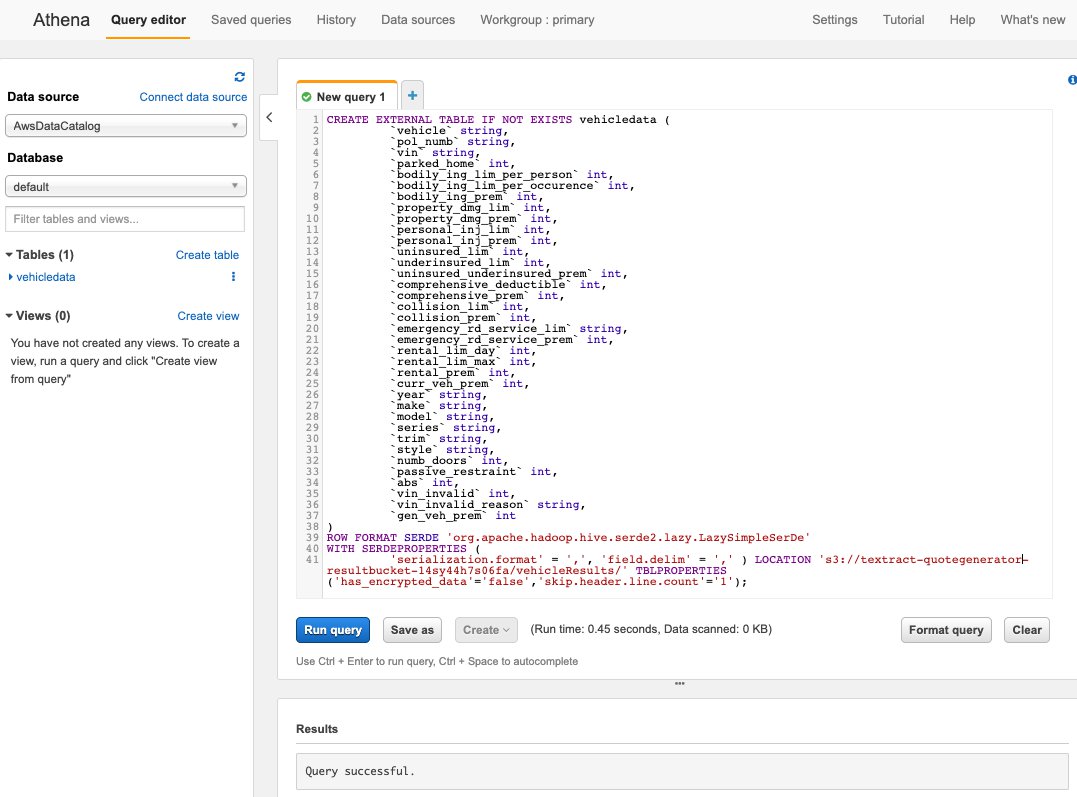
## Analyzing and Visualizing Claim data

Here we will use Athena to query the data and Quicksight to visualize it.

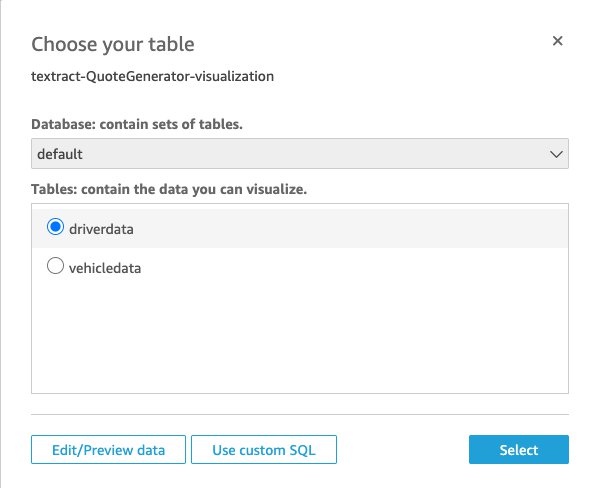
### To setup analytics from declaration page data

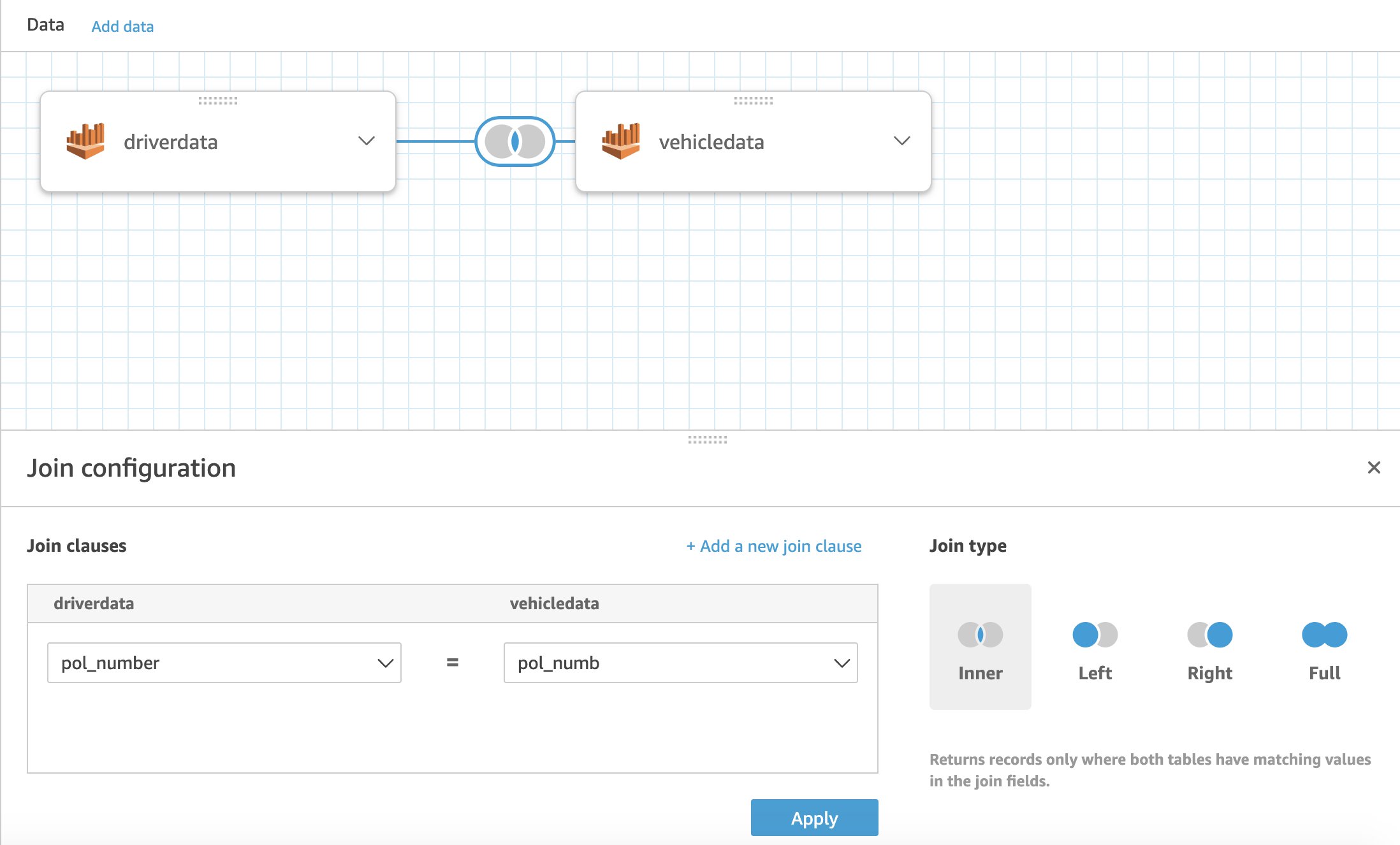
1. Use the services tab to locate Amazon Athena
2. Download the [Athena Query text file](https://github.com/RinoDinoRino/soldo_project/blob/master/AthenaQuery/Queries.txt) and open it.
3. In both part 1 and part 2 of the text file, change the “YOUR RESULT BUCKET NAME” to your bucket name. This will be located at the bottom of both parts.

1. Copy the first query from the text file into Athena’s query editor and run it.



### To setup visualization for the declaration page data

1. **(Optional)**: Download the [driver sample data and vehicle sample data](https://github.com/RinoDinoRino/soldo_project/tree/master/sampleData) and upload both into their respective prefix in the result bucket.
2. Use the services tab to locate Amazon QuickSight.
3. On the upper right hand of the screen, click the account icon. Check if the region you are currently in is the same as that of where your Athena table’s are located. Not every region supports Quicksight.
4. On the upper left, click New Analysis > New Dataset > Athena
5. Enter a name for your data source. Click create data source.
6. On the Choose your table page, chose the database your tables are located. Currently, they should be under default. Then select the “driverData” table. Followed by clicking Edit/Preview data.
   1. If you receive an Access Denied error. Click the account icon> Manage Quicksight > Security & permissions. Under quicksight access to AWS services, click Add or remove. Either add Athena if it is not on the list, or remove and then re-add Athena if it is already on the list. You will be prompted to select S3 buckets to allow access to. Add the result bucket created by the stack. This should resolve the issue.
7. Once at the data editor, on the top click “Add data.” Add the “vechileData” table, then click select.
8. Click the red circles that are between the two tables. This should open the join configuartion menu.
9. From the driverData select the “pol\_number” feild. From the vehicleData table, select the “pol\_numb” feild. Additionally, make sure that the join type is Inner. Then click apply.



1. From here additional filters can be added to ensure only valid data is visualized. If you want to only see data from valid pages, click “Add Filter” on the bottom left. Add the “vin\_invalid” and “user\_invalid” fields and set both filters to equal 0. The query should update and only show valid data.
2. Finally, click “Save & visualize” on the top of the window.
3. You can now visualize the data in any combination and in several different ways.