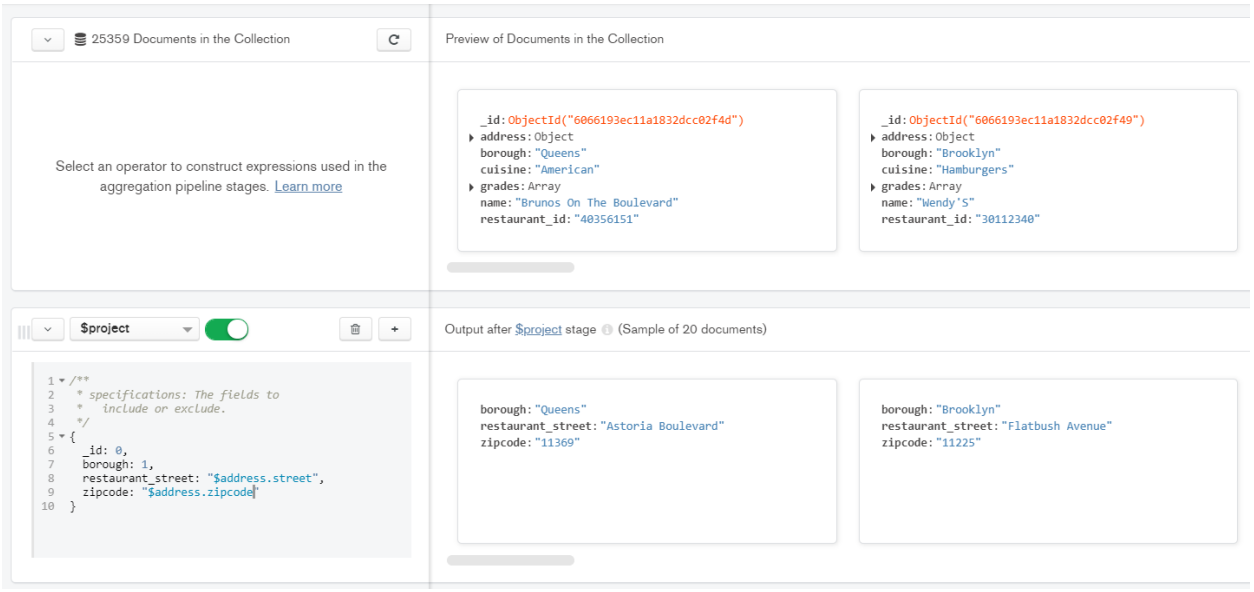
Mini-project #3: MongoDB Aggregate Pipeline Queries- Nicholas Gincley & Lauren Kahrs

- Show the aggregation pipeline in screen captures. Show 2-3 stages in each screenshot, and include the first 1-2 sample results.
- Only show the requested fields in the final answer; there should not be any nested fields (arrays or objects).

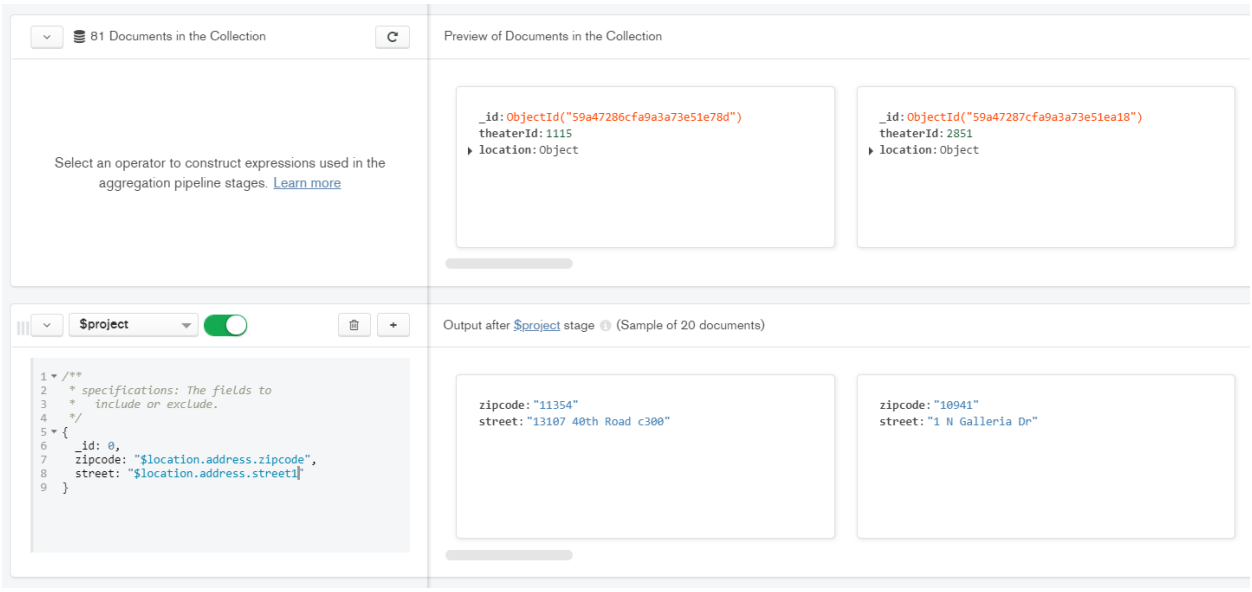
Part 1: Querying Provided Collections

1. Use *restaurants* and *nyTheaters*. Display the restaurant street, borough, and zipcode along with the theater street where the restaurant zipcode matches the theater zipcode. Sort the results by borough and zipcode (both in ascending order). The solution has 6,811 documents.

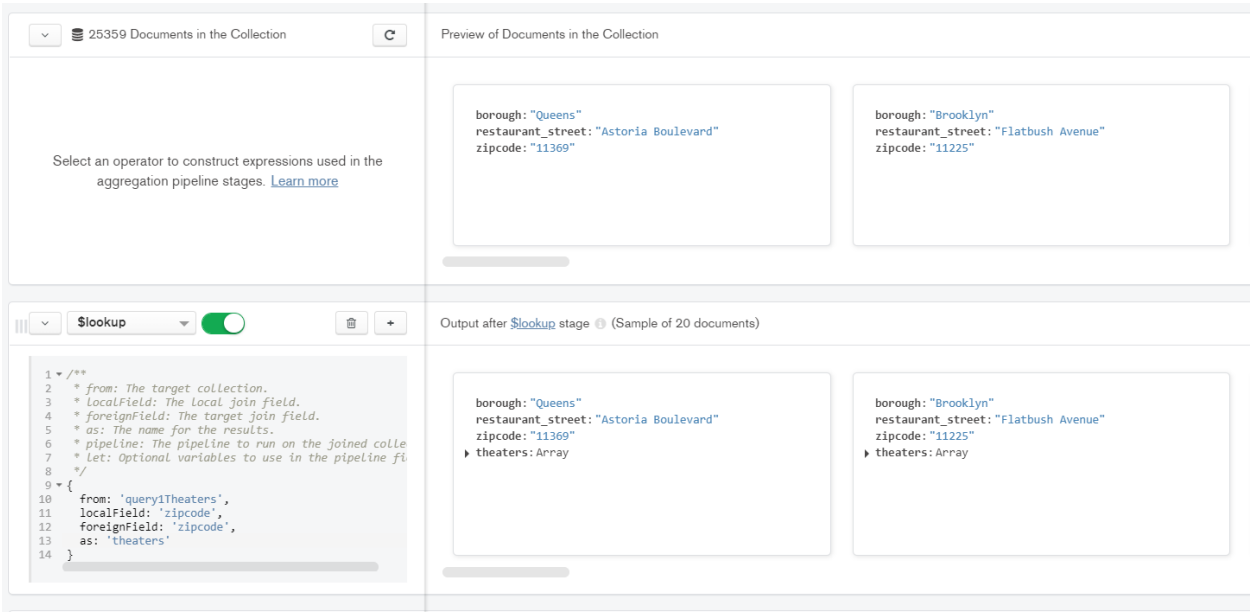
Getting query1Restaurants view

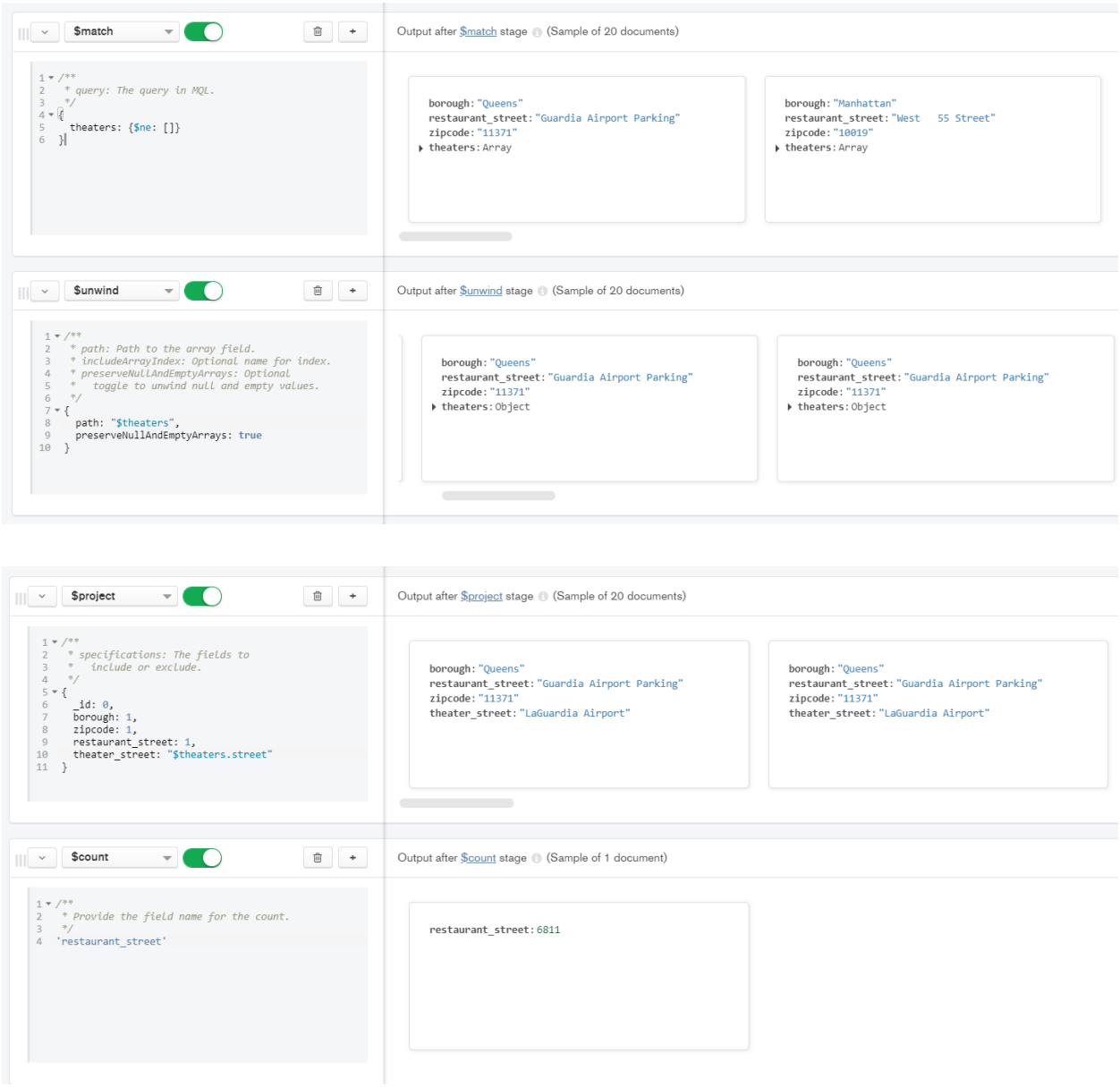


Getting query1Theaters view

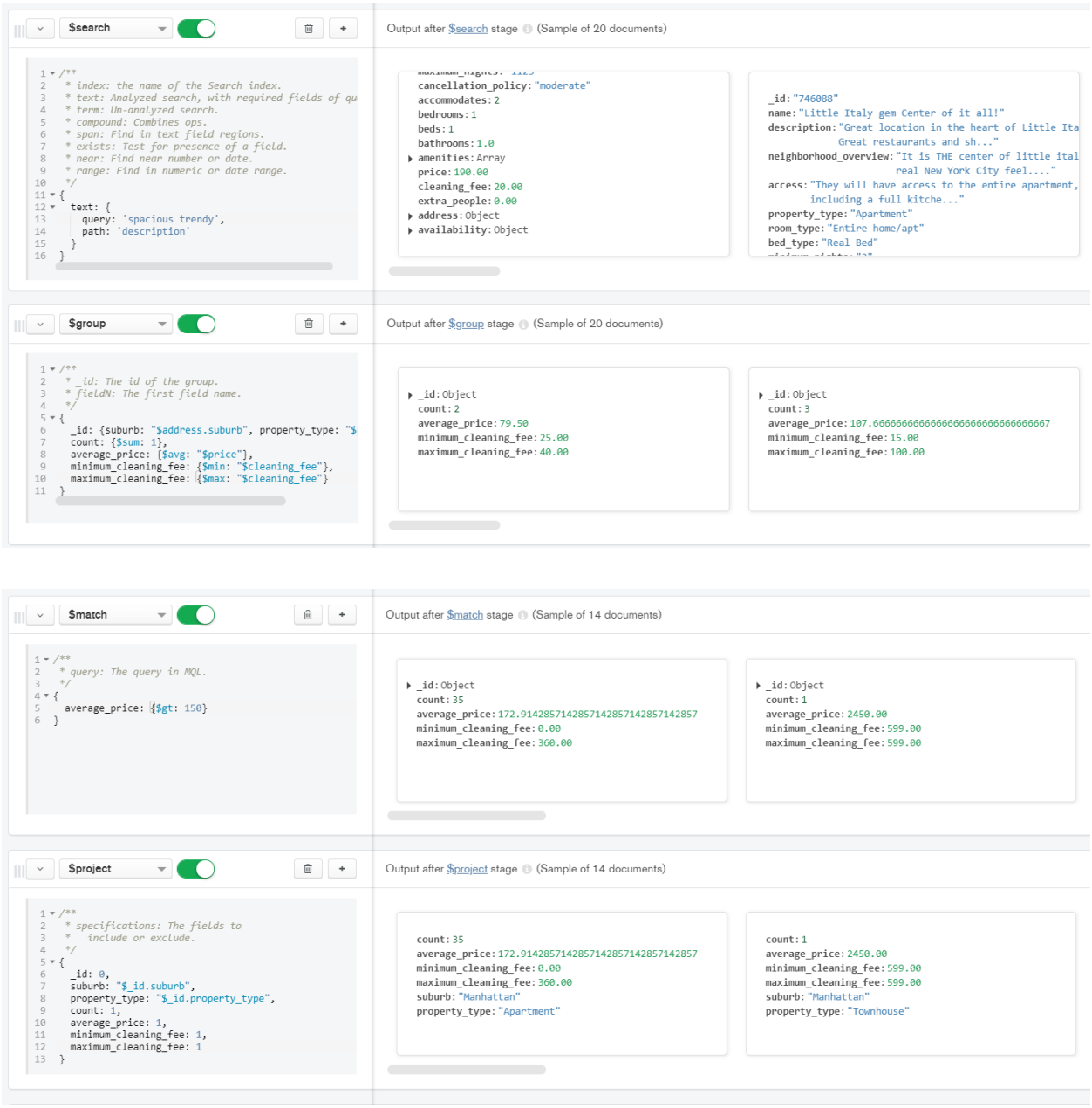


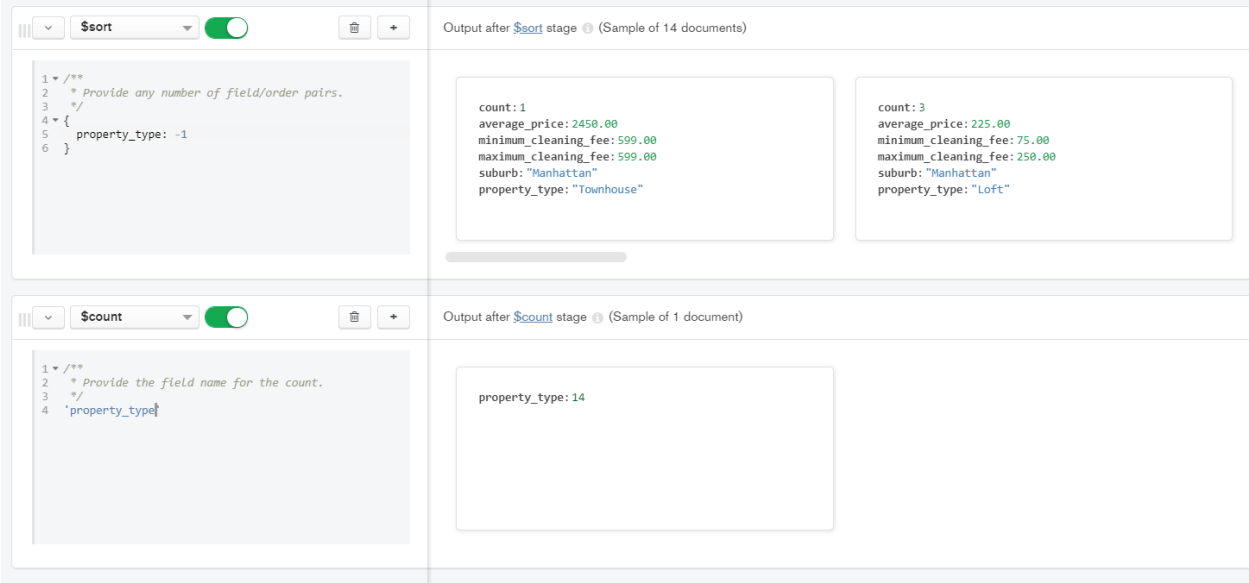
Getting query





2. Use *nyBnB* to find lodgings that have ‘spacious’ or ‘trendy’ in their description. Group by suburb and property type. There should be 164 documents at this stage. Display the suburb, property type, count, average price, and minimum and maximum cleaning fees where the average price is greater than or equal to 150. Sort by property type in descending order. The final result has 14 documents.


$search
/**
* index: the name of the Search index.
* text: Analyzed search, with required fields of qu
* term: Un-analyzed search.
* compound: Combines ops.
* span: Find in text field regions.
* exists: Test for presence of a field.
* near: Find near number or date.
* range: Find in numeric or date range.
*/
{
text: {
query: 'spacious trendy',
path: 'description'
}
}
Output after $search stage (Sample of 20 documents)
cancellation_policy: "moderate"
accommodates: 2
bedrooms: 1
beds: 1
bathrooms: 1.0
amenities: Array
price: 190.00
cleaning_fee: 20.00
extra_people: 0.00
address: Object
availability: Object
_id: "746088"
name: "Little Italy gem Center of it all!"
description: "Great location in the heart of Little Ita
Great restaurants and sh..."
neighborhood_overview: "It is THE center of little ital
real New York City feel...."
access: "They will have access to the entire apartment,
including a full kitche..."
property_type: "Apartment"
room_type: "Entire home/apt"
bed_type: "Real Bed"
$group
/**
* _id: The id of the group.
* fieldN: The first field name.
*/
{
_id: {suburb: "$address.suburb", property_type: "$
count: {$sum: 1},
average_price: {$avg: "$price"},
minimum_cleaning_fee: {$min: "$cleaning_fee"},
maximum_cleaning_fee: {$max: "$cleaning_fee"}
}
Output after $group stage (Sample of 20 documents)
_id: Object
count: 2
average_price: 79.50
minimum_cleaning_fee: 25.00
maximum_cleaning_fee: 40.00
_id: Object
count: 3
average_price: 107.66666666666666666666666666667
minimum_cleaning_fee: 15.00
maximum_cleaning_fee: 100.00
$match
/**
* query: The query in MQL.
*/
{
average_price: {$gt: 150}
}
Output after $match stage (Sample of 14 documents)
_id: Object
count: 35
average_price: 172.91428571428571428571428571428571
minimum_cleaning_fee: 0.00
maximum_cleaning_fee: 360.00
_id: Object
count: 1
average_price: 2450.00
minimum_cleaning_fee: 599.00
maximum_cleaning_fee: 599.00
$project
/**
* specifications: The fields to
* include or exclude.
*/
{
_id: 0,
suburb: "$_id.suburb",
property_type: "$_id.property_type",
count: 1,
average_price: 1,
minimum_cleaning_fee: 1,
maximum_cleaning_fee: 1
}
Output after $project stage (Sample of 14 documents)
count: 35
average_price: 172.91428571428571428571428571428571
minimum_cleaning_fee: 0.00
maximum_cleaning_fee: 360.00
suburb: "Manhattan"
property_type: "Apartment"
count: 1
average_price: 2450.00
minimum_cleaning_fee: 599.00
maximum_cleaning_fee: 599.00
suburb: "Manhattan"
property_type: "Townhouse"



Part 2: Preparing Your Own Collection

1. Find a data set at https://opendata.cityofnewyork.us/data/. You should choose a data set that contains either a zipcode or borough. These fields may be named something different, but the data should match up to one of these fields from the *restaurants* collection. There are over 2,000 data sets available.

   https://data.cityofnewyork.us/Business/Consumer-Services-Mediated-Complaints/nre2-6m2s/data

2. Give a high-level description of what is contained in the data set (one or two sentences.) This should not be a list of the fields; describe it meaningfully and concisely.

   This dataset contains information complaints about businesses in the US (majority in the NYC area) that have been mediated by the DCA (Department of Consumer Affairs) and it's outcome, satisfaction of the complainer and any restitution.

3. List how many instances (e.g., rows) and how many fields (e.g., columns) are in the data set. Give the URL for the data set.

   There are 2506 rows and 17 columns in this table.

   https://data.cityofnewyork.us/Business/Consumer-Services-Mediated-Complaints/nre2-6m2s/data

4. Load the data into BigQuery and create a subset of the data with 1,000-2,000 instances.

   Done (If you wanna do this too I downloaded the data from the link above as a csv then in BigQuery click create new table, then select import data from the dropdown list at the

top, select csv as the type on the right, select the file, tell it to auto create the schema then create.)

5. Describe in English in what ways you modified the data set (to reduce the number of instances and fields, if you choose to reduce the number of fields.) Give the BigQuery query or queries that you use to reduce the data set.

   First I removed all rows not pertaining the New York.

```
delete from tactical-curve-258016.ginclene_dw.Customer_Complaints

where Business_State != 'NY';
```

   This removed 460 rows leaving us with 2046 rows.

   Then I removed all entries where the city is null (as this is what we will turn into Borough)

```
delete from tactical-curve-258016.ginclene_dw.Customer_Complaints

where Business_City is null;
```

   This removed another 59 rows leaving us with 1987 rows

   Then I removed many of the unnecessary columns in the dataset

```
ALTER TABLE  tactical-curve-258016.ginclene_dw.Customer_Complaints

DROP COLUMN Longitude,

DROP COLUMN Latitude,

DROP COLUMN Building_Address_Unit,

DROP COLUMN Business_Building,

DROP COLUMN Mediation_Close_Date,

DROP COLUMN Mediation_Start_Date,

DROP COLUMN Business_State;
```

   We dropped state because now we know all the businesses are in New York.

   Next we changed all instances where the business city is New York to it's borough of Manhattan along with normalizing so other city names and reformatting them to later match the restaurant table.

```
update tactical-curve-258016.ginclene_dw.Customer_Complaints
set Business_City = "Manhattan"
where Business_City = "NEW YORK";

update tactical-curve-258016.ginclene_dw.Customer_Complaints
set Business_City = "Queens"
where Business_City = "QUEENS VLG"
or Business_City = "QUEENS VILLAGE"
or Business_City = "QUEENS";

update tactical-curve-258016.ginclene_dw.Customer_Complaints
set Business_City = "Bronx"
where Business_City = "BRONX";


update tactical-curve-258016.ginclene_dw.Customer_Complaints
set Business_City = "Brooklyn"
where Business_City = "BROOKLYN";

update tactical-curve-258016.ginclene_dw.Customer_Complaints
set Business_City = "Staten Island"
where Business_City = "STATEN ISLAND";
```

Finally we will now remove all entries that are not 1 of our 5 boroughs.

```
delete from tactical-curve-258016.ginclene_dw.Customer_Complaints
```

```
where Business_City != "Manhattan" AND

Business_City != "Queens" AND

Business_City != "Staten Island" AND

Business_City != "Brooklyn" AND

Business_City != "Bronx";
```

This leaves us with a total of 1269 rows in our dataset

6. Export the data to either JSON or CSV and give a Google Drive link to the final data set.

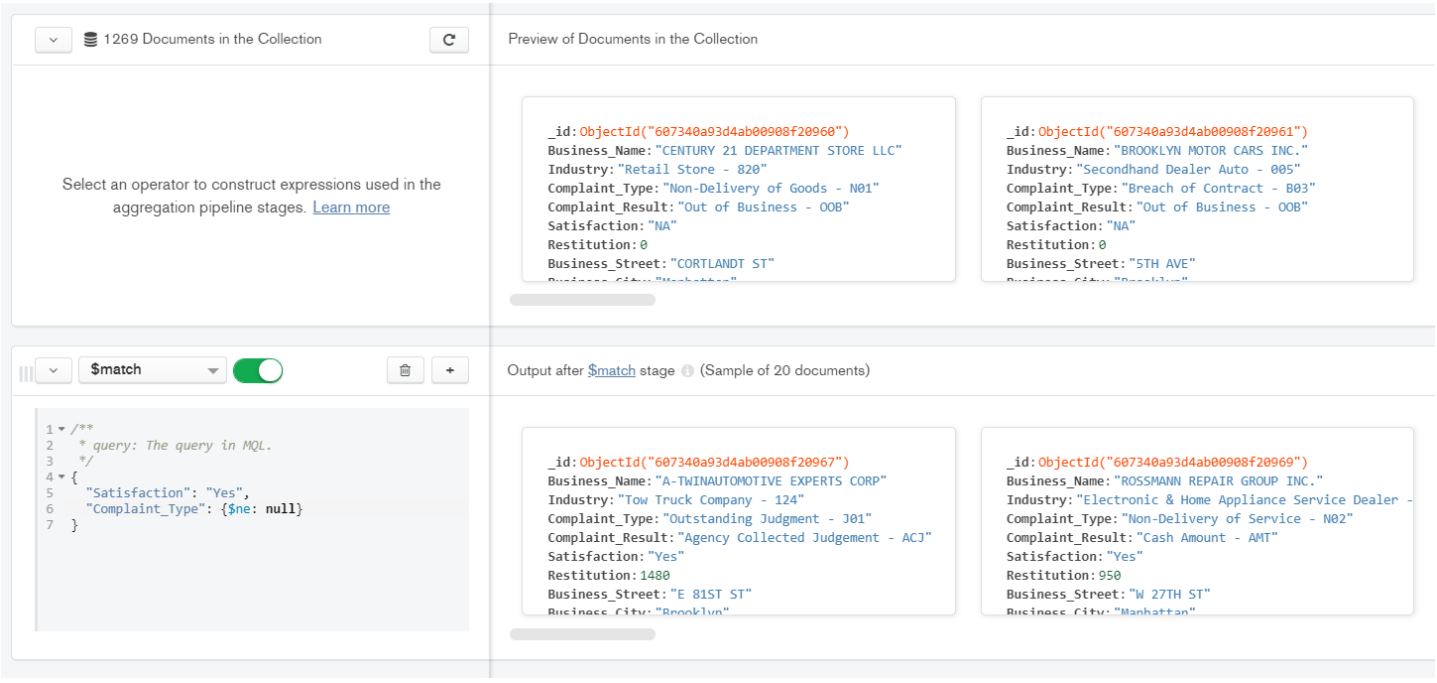
   https://drive.google.com/file/d/1nCNGvHpRPufAsR6n88Mvw0Ak7lK4bpFd/view?usp=sharing

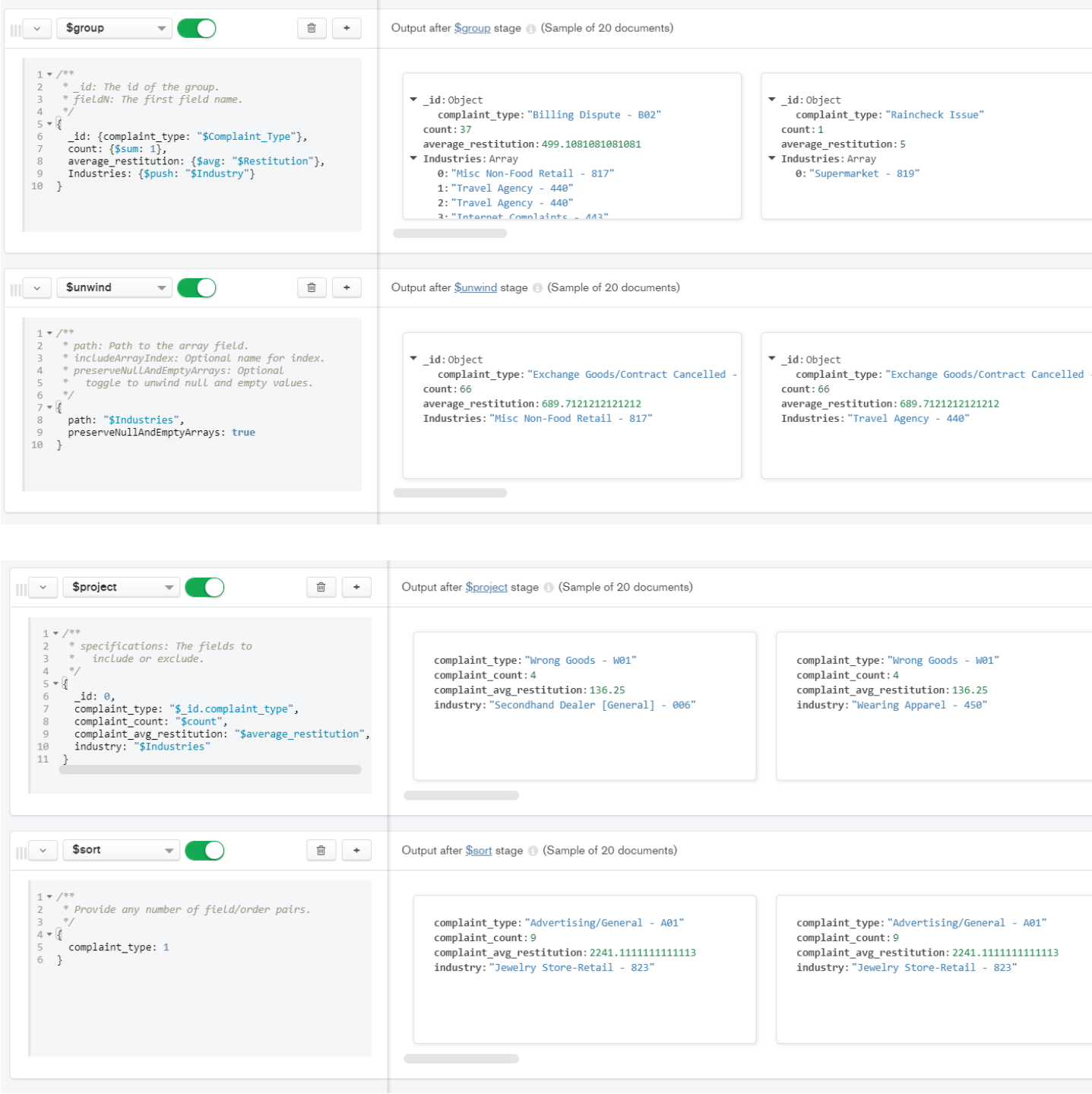
Part 3: Creating Data Demands and Solutions

Create two data demands and solutions that meet the given criteria. Your data demands and solutions should be presented in the same style as the data demands and solutions in Part 1.

1. Using your collection prepared in Part 2, write a data demand and solution that requires using $group, $match, $project, and $sort stages.

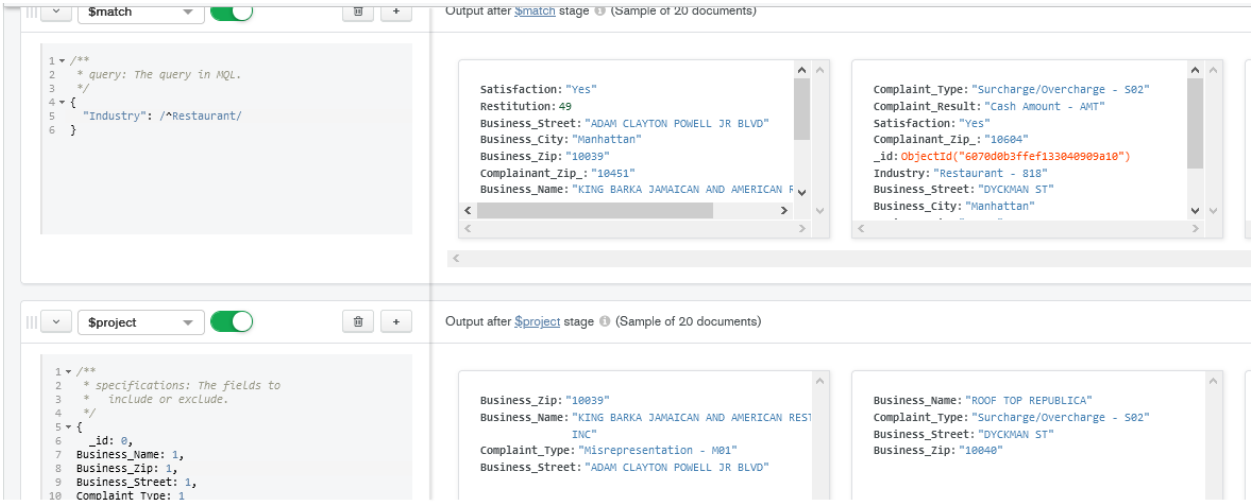
   Give each complaint type, how many of each complaint, average restitution for each complaint, and each industry associated with the complaint. Leave out any null values.

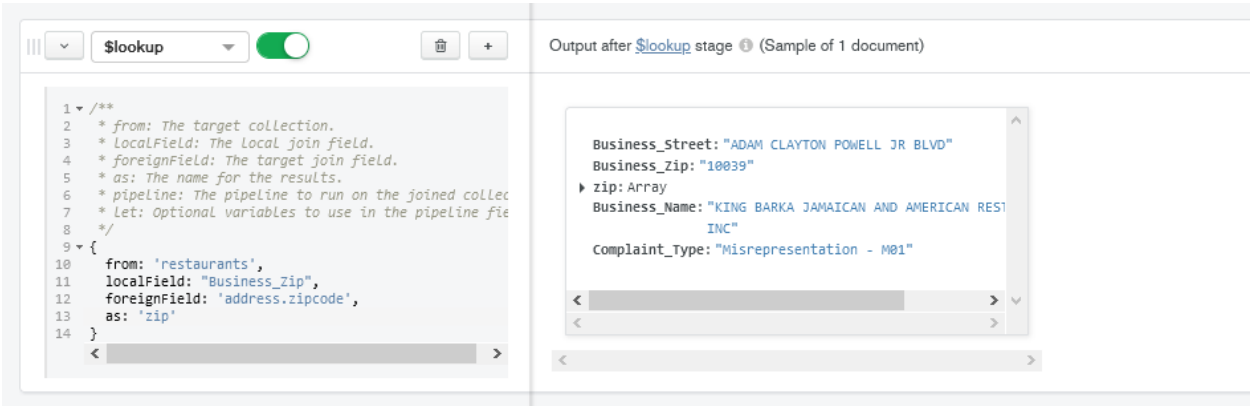


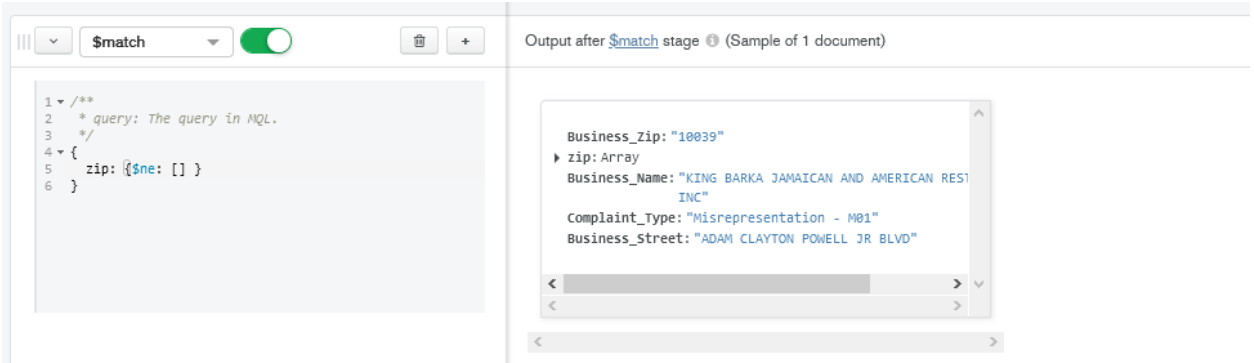


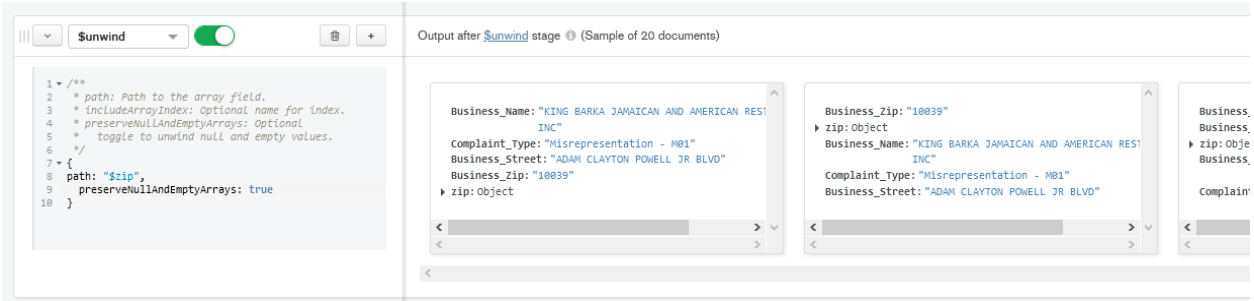
2. Using your collection prepared in Part 2 and the *restaurants* collection, write a data demand and solution that uses $lookup to achieve an inner join, $match, $project, and $sort stages.

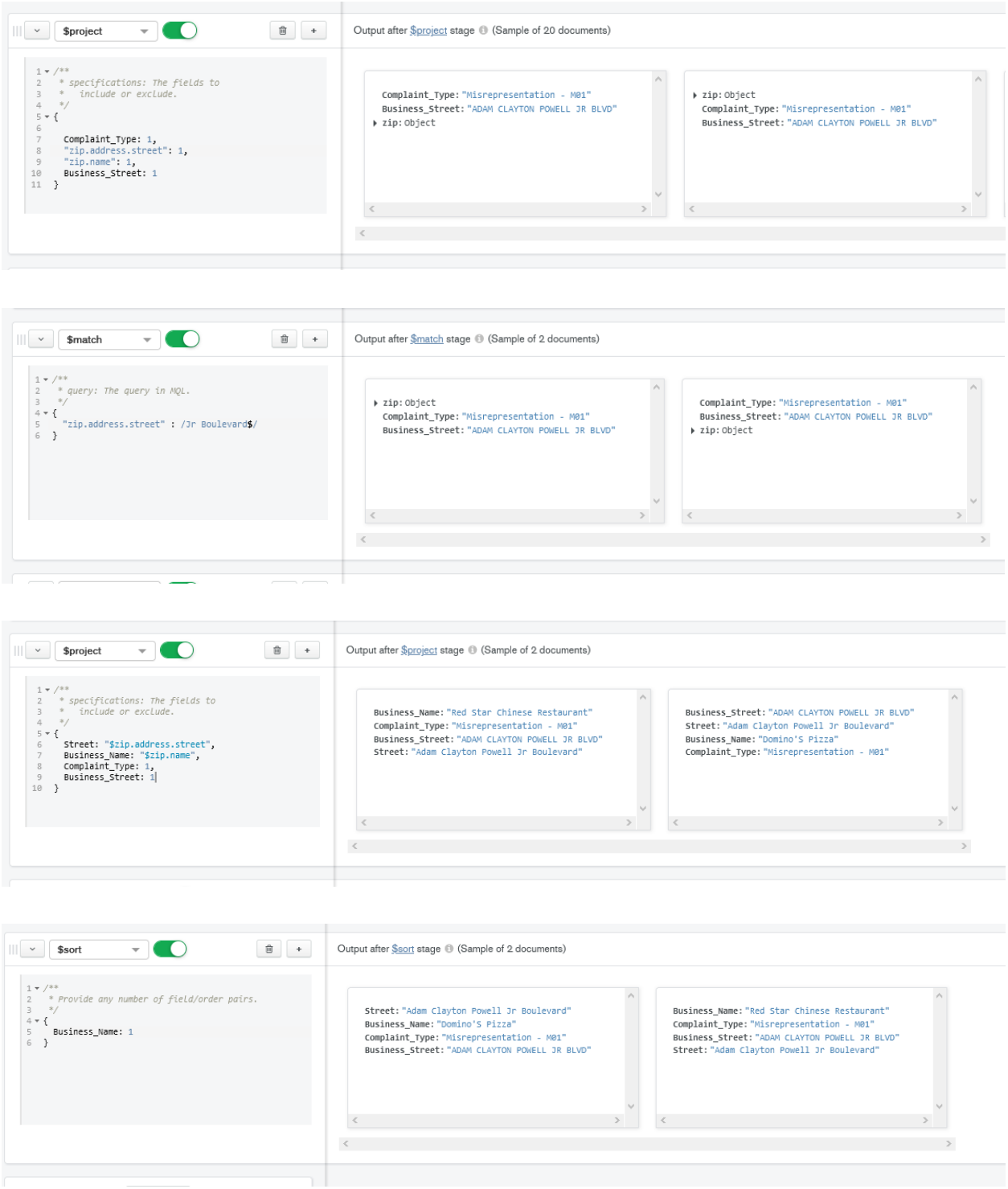
   Find any restaurants that have a consumer complaint, provide the type of complaint, the restaurant name, and the street it is on.


$match
Output after $match stage (Sample of 20 documents)
/**
* query: The query in MQL.
*/
{
"Industry": /^Restaurant/
}
Satisfaction: "Yes"
Restitution: 49
Business_Street: "ADAM CLAYTON POWELL JR BLVD"
Business_City: "Manhattan"
Business_Zip: "10039"
Complainant_Zip_: "10451"
Business_Name: "KING BARKA JAMAICAN AND AMERICAN R
Complaint_Type: "Surcharge/Overcharge - S02"
Complaint_Result: "Cash Amount - AMT"
Satisfaction: "Yes"
Complainant_Zip_: "10604"
_id: ObjectId("6070d0b3ffef133040909a10")
Industry: "Restaurant - 818"
Business_Street: "DYCKMAN ST"
Business_City: "Manhattan"
$project
Output after $project stage (Sample of 20 documents)
/**
* specifications: The fields to
* include or exclude.
*/
{
_id: 0,
Business_Name: 1,
Business_Zip: 1,
Business_Street: 1,
Complaint_Type: 1
Business_Zip: "10039"
Business_Name: "KING BARKA JAMAICAN AND AMERICAN REST
INC"
Complaint_Type: "Misrepresentation - M01"
Business_Street: "ADAM CLAYTON POWELL JR BLVD"
Business_Name: "ROOF TOP REPUBLICA"
Complaint_Type: "Surcharge/Overcharge - S02"
Business_Street: "DYCKMAN ST"
Business_Zip: "10040"


$lookup
Output after $lookup stage (Sample of 1 document)
/**
* from: The target collection.
* localField: The local join field.
* foreignField: The target join field.
* as: The name for the results.
* pipeline: The pipeline to run on the joined collec
* let: Optional variables to use in the pipeline fie
*/
{
from: 'restaurants',
localField: "Business_Zip",
foreignField: 'address.zipcode',
as: 'zip'
}
Business_Street: "ADAM CLAYTON POWELL JR BLVD"
Business_Zip: "10039"
zip: Array
Business_Name: "KING BARKA JAMAICAN AND AMERICAN REST
INC"
Complaint_Type: "Misrepresentation - M01"


$match
Output after $match stage (Sample of 1 document)
/**
* query: The query in MQL.
*/
{
zip: {$ne: [] }
}
Business_Zip: "10039"
zip: Array
Business_Name: "KING BARKA JAMAICAN AND AMERICAN REST
INC"
Complaint_Type: "Misrepresentation - M01"
Business_Street: "ADAM CLAYTON POWELL JR BLVD"


$unwind
Output after $unwind stage (Sample of 20 documents)
/**
* path: Path to the array field.
* includeArrayIndex: Optional name for index.
* preserveNullAndEmptyArrays: Optional
* toggle to unwind null and empty values.
*/
{
path: "$zip",
preserveNullAndEmptyArrays: true
}
Business_Name: "KING BARKA JAMAICAN AND AMERICAN REST
INC"
Complaint_Type: "Misrepresentation - M01"
Business_Street: "ADAM CLAYTON POWELL JR BLVD"
Business_Zip: "10039"
zip: Object
Business_Zip: "10039"
zip: Object
Business_Name: "KING BARKA JAMAICAN AND AMERICAN REST
INC"
Complaint_Type: "Misrepresentation - M01"
Business_Street: "ADAM CLAYTON POWELL JR BLVD"



Part 4: Time on Task

Record the time spent working individually and as a group on this project.

Lauren Kahrs - 4 hours

Nicholas Gincley - 4 hours