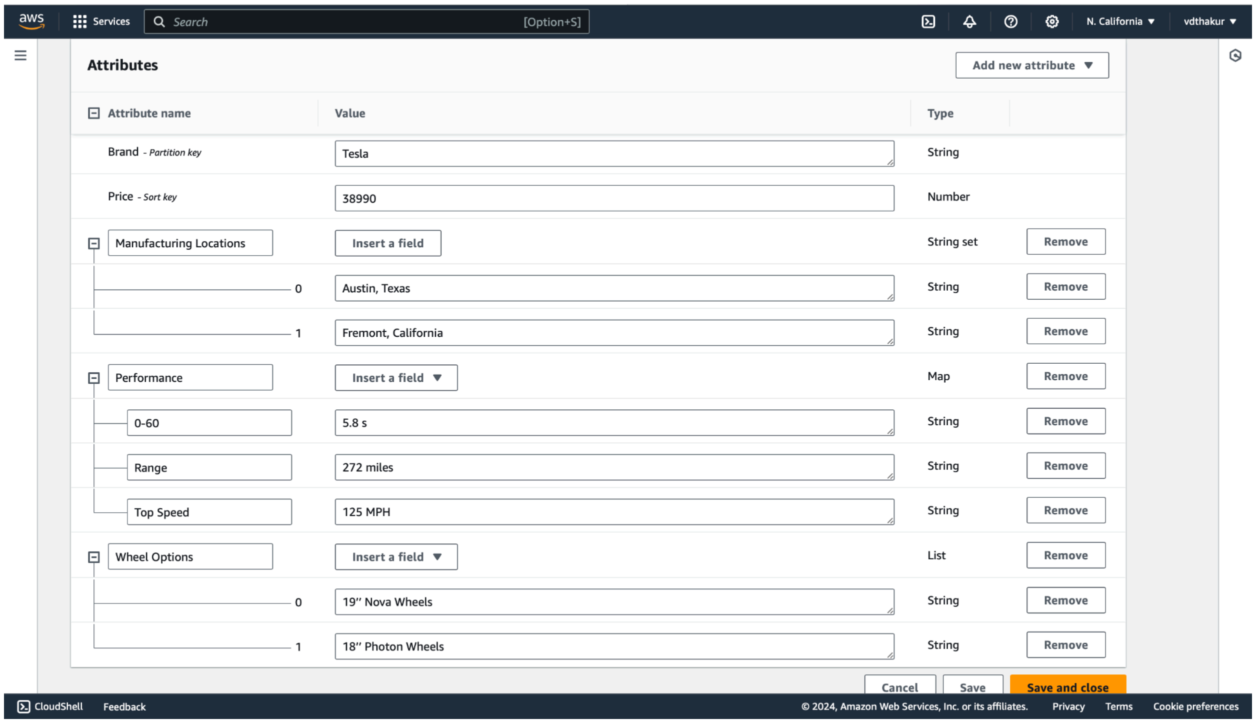
Lab 4 - Varun Thakur

1) Explain why you selected the particular attributes as the partition key and sort key.

I selected Brand to be the partition key because each car brand is unique and they create more than one vehicle. This way, all of the cars of the same Brand can be found in the same place as queries can be done on Brand.

The price was chosen as the sort key because price point is a very relevant factor in relation to evaluating cars as it can be indicative of being a high end or low end vehicle. Sorting by price allows for the ability to see the cars in order of their price point and thus allows for comparison of other attributes against cars of similar price points.

2) Insert at least one item into the table that meets all the type requirements mentioned above. Show a screenshot of this item in the DynamoDB console and identify each data type in the item.



Attributes

Attribute name	Value	Type
Brand - *Partition key*	Tesla	String
Price - *Sort key*	38990	Number
Manufacturing Locations		String set
0	Austin, Texas	String
1	Fremont, California	String
Performance		Map
0-60	5.8 s	String
Range	272 miles	String
Top Speed	125 MPH	String
Wheel Options		List
0	19'' Nova Wheels	String
1	18'' Photon Wheels	String

identify each data type in the item:

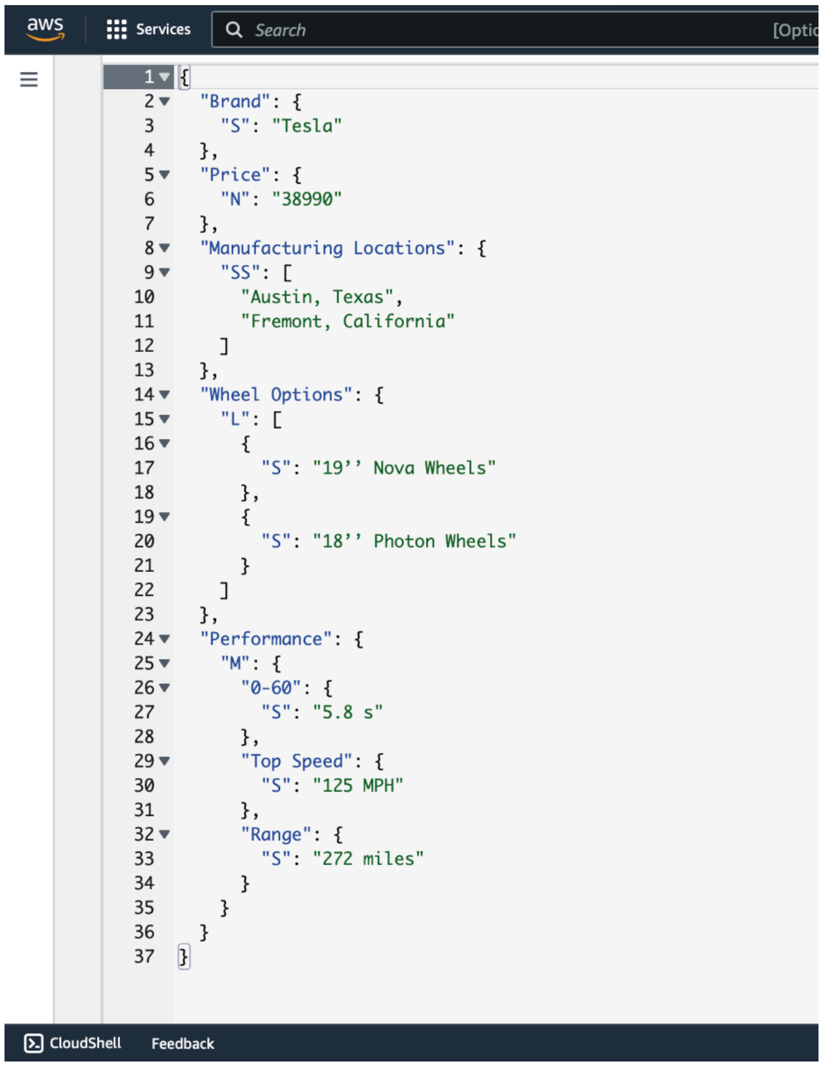
Brand (string)
Price (Number)
Manufacturing Locations (String Set)
- 0 (String)
- 1 (String)

Performance (Map)
- 0-60 (string)
- Top Speed (string)
- Range (string)

Wheel Options (List)
- 0 (String)
- 1 (String)

3) **Show the `JSON view` of the above item and explain its format (e.g., how each attribute’s data type is presented in JSON view)**

```
{
 "Brand": {
  "S": "Tesla"
 },
 "Price": {
  "N": "38990"
 },
 "Manufacturing Locations": {
  "SS": [
   "Austin, Texas",
   "Fremont, California"
  ]
 },
 "Wheel Options": {
  "L": [
   {
    "S": "19” Nova Wheels"
   },
   {
    "S": "18” Photon Wheels"
   }
  ]
 },
 "Performance": {
  "M": {
   "0-60": {
    "S": "5.8 s"
   },
   "Top Speed": {
    "S": "125 MPH"
   },
   "Range": {
    "S": "272 miles"
   }
  }
 }
}
```



```
{
  "Brand": {
    "S": "Tesla"
  },
  "Price": {
    "N": "38990"
  },
  "Manufacturing Locations": {
    "SS": [
      "Austin, Texas",
      "Fremont, California"
    ]
  },
  "Wheel Options": {
    "L": [
      {
        "S": "19’’ Nova Wheels"
      },
      {
        "S": "18’’ Photon Wheels"
      }
    ]
  },
  "Performance": {
    "M": {
      "0-60": {
        "S": "5.8 s"
      },
      "Top Speed": {
        "S": "125 MPH"
      },
      "Range": {
        "S": "272 miles"
      }
    }
  }
}
```

Q3 continued on the next page

Format Explanation:

The format seen above shows that for every attribute present, the data type is presented right after. For example, for the string set (Manufacturing Locations) the data is represented below. We can see that the attribute Manufacturing Location is followed by the data type ({ "SS":) which is then followed by a list of the values for the Manufacturing Location (["Austin, Texas", "Fremont, California"])

```
 "Manufacturing Locations": {
   "SS": [
     "Austin, Texas",
     "Fremont, California"
   ]
 },
```

For a Map attribute such as Performance:

```
 "Performance": {
   "M": {
     "0-60": {
       "S": "5.8 s"
     },
     "Top Speed": {
       "S": "125 MPH"
     },
     "Range": {
       "S": "272 miles"
     }
   }
 }
}
```

The data type is still represented right after Performance within a JSON object. The key value pairs within the Map are also represented as JSON objects with the key name coming before the value which is again stored in a JSON object that contains the value's datatype.

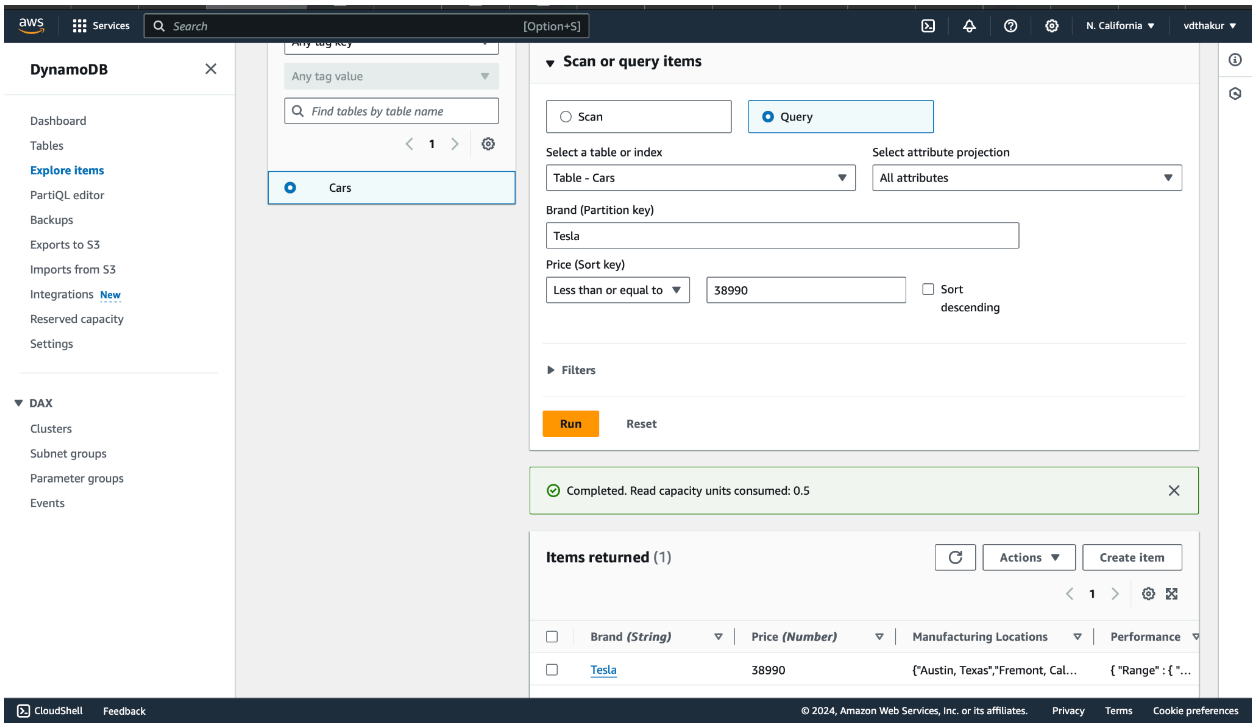
For the list, Wheel Options as seen below, the data is represented with the Attribute name followed by a JSON object which contains the Data Type (L for list) and is followed by a list of JSON objects that represent the list items. In this case, the different types of wheels offered on a Model 3.

```
 "Wheel Options": {
   "L": [
     {
       "S": "19” Nova Wheels"
     },
     {
       "S": "18” Photon Wheels"
     }
   ]
 }
```

The partition key Brand and sort key price are shown at the top of the JSON view with the attribute name followed by the data type (S for string, N for number) and the value for that data type enclosed in a JSON object.

```
{
 "Brand": {
   "S": "Tesla"
 },
 "Price": {
   "N": "38990"
 },
```

4) Execute the scan and query methods on your table. Show a screenshot for each operation.



Select a table or index: Table - Cars

Select attribute projection: All attributes

Brand (Partition key): Tesla

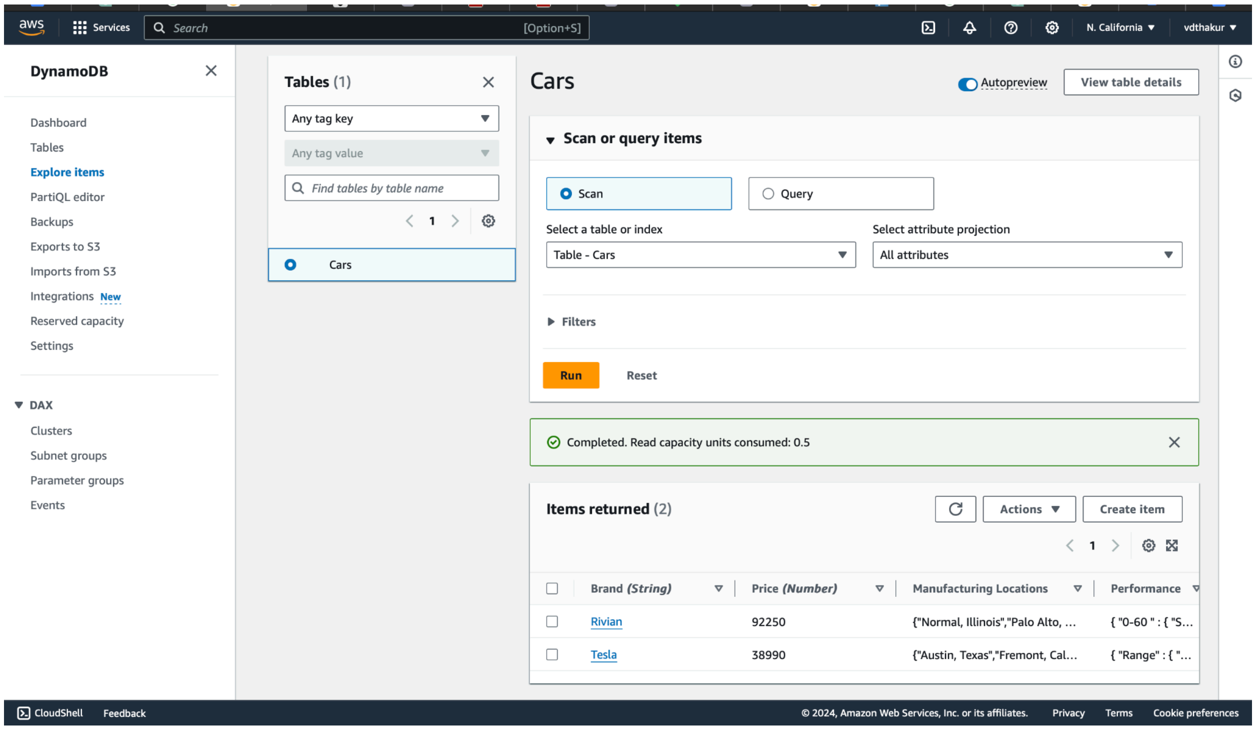
Price (Sort key): Less than or equal to 38990

Sort descending

Completed. Read capacity units consumed: 0.5

Items returned (1)

Brand (String)	Price (Number)	Manufacturing Locations	Performance
Tesla	38990	{"Austin, Texas","Fremont, Cal…	{ "Range" : { "…



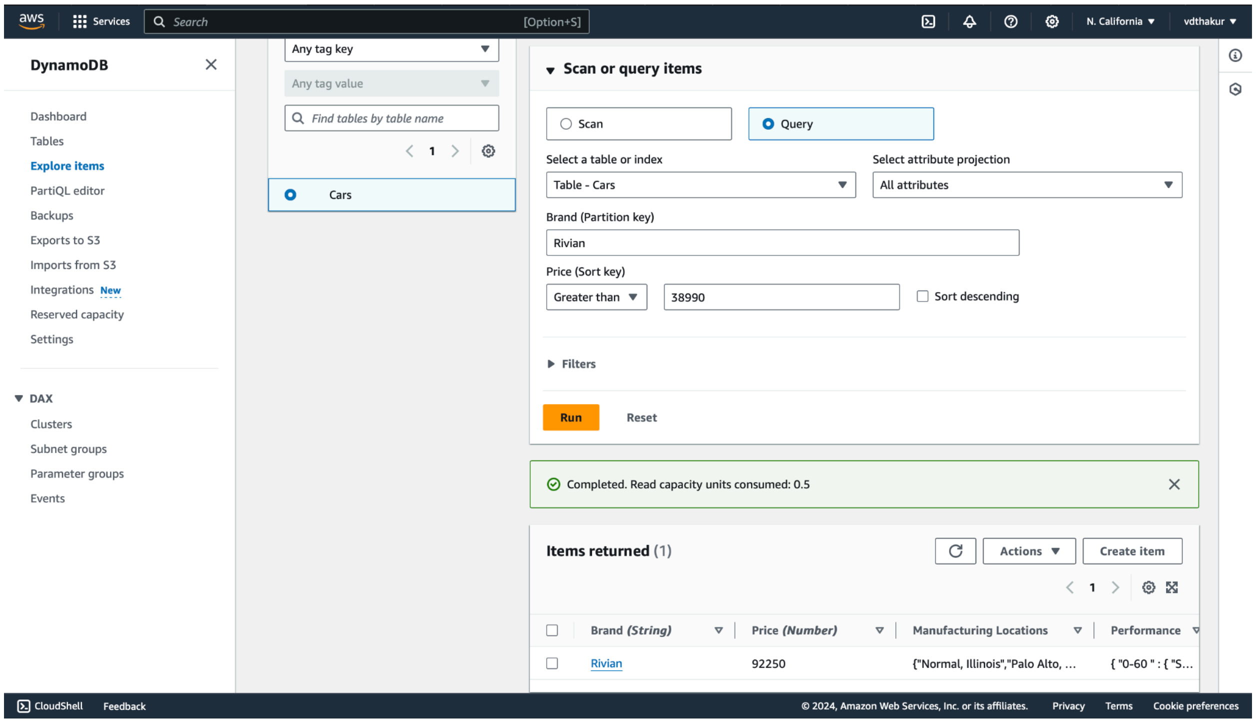
Select a table or index: Table - Cars

Select attribute projection: All attributes

Completed. Read capacity units consumed: 0.5

Items returned (2)

Brand (String)	Price (Number)	Manufacturing Locations	Performance
Rivian	92250	{"Normal, Illinois","Palo Alto, …	{ "0-60 " : { "S…
Tesla	38990	{"Austin, Texas","Fremont, Cal…	{ "Range" : { "…



Scan or query items

Scan [✓ Query]

Select a table or index: Table - Cars

Select attribute projection: All attributes

Brand (Partition key): Rivian

Price (Sort key): Greater than 38990

☐ Sort descending

Filters

Completed. Read capacity units consumed: 0.5

Items returned (1)

Brand (String)	Price (Number)	Manufacturing Locations	Performance
Rivian	92250	{"Normal, Illinois","Palo Alto, …	{ "0-60 " : { "S…