Lab 4: DynamoDB
DSCI 551 – Spring 2024

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

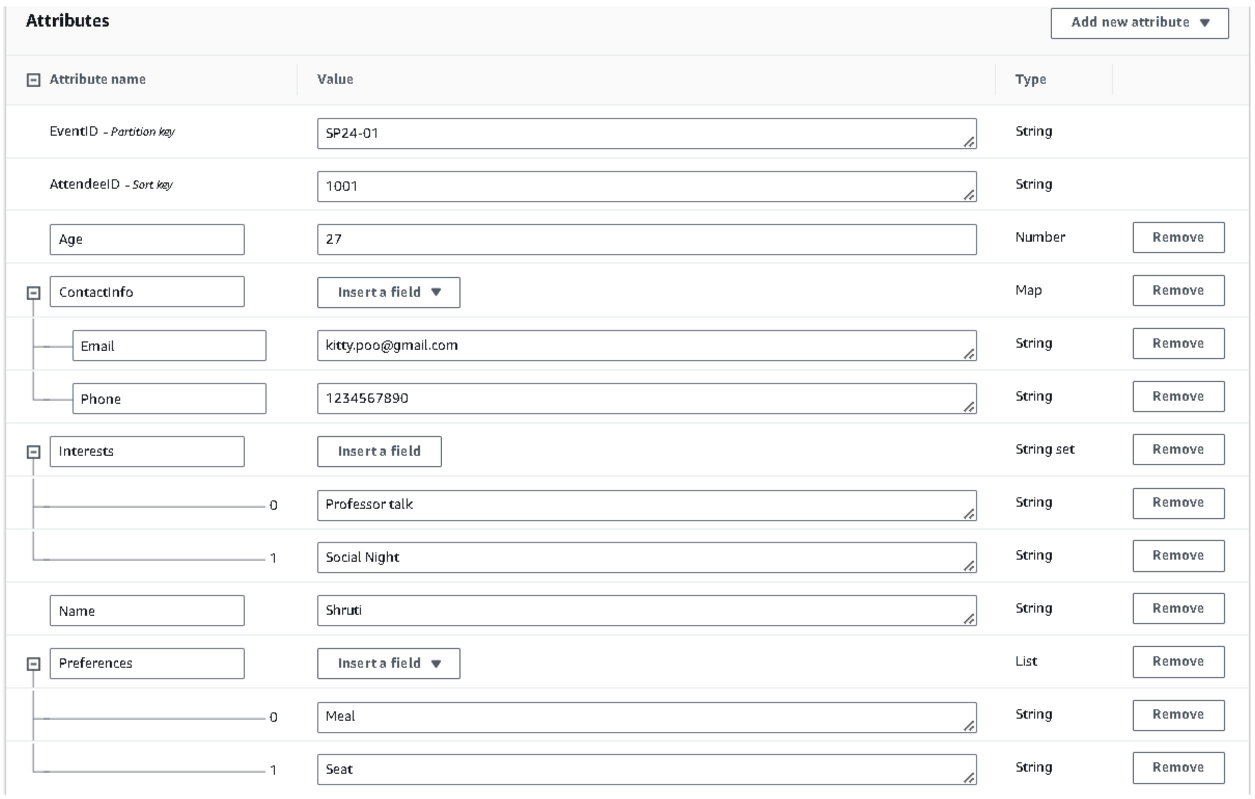
Solution:
For the "GridsEventRegistrations" table, the selection of EventID as the partition key and AttendeeID as the sort key is based on the nature of the data and its intended use.

Choosing EventID as the **partition key** ensures that data is distributed based on different events. This is effective because events naturally segment the data, assuming that the application will have a relatively large number of events with a fairly even distribution of registration records across these events. The access to the "GridsEventRegistrations" table would likely involve queries to retrieve all attendees for a specific event. Using EventID as the partition key optimizes the table for this access pattern, allowing efficient queries for all registrations associated with a given event.

The sort key in DynamoDB not only provides a way to ensure uniqueness within a partition but also orders the data. By using AttendeeID as the **sort key**, it ensures that each attendee within an event is unique and can be efficiently sorted or queried. This is particularly useful for scenarios where applications might need to retrieve specific attendees' details within an event or list all attendees in a sorted manner. The combination of EventID (partition key) and AttendeeID (sort key) forms a composite primary key for the table. This composite key uniquely identifies each item, which in this case, is each registration record. The choice supports a broad range of query operations, including efficient retrieval of individual registrations if both the event ID and

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.

Solution:



Attribute name	Value	Type
EventID - *Partition key*	SP24-01	String
AttendeeID - *Sort key*	1001	String
Age	27	Number
ContactInfo		Map
Email	kitty.poo@gmail.com	String
Phone	1234567890	String
Interests		String set
0	Professor talk	String
1	Social Night	String
Name	Shruti	String
Preferences		List
0	Meal	String
1	Seat	String

EventID: **String**
The EventID attribute is represented as a string. It uniquely identifies the event. In this example, it's "SP24-01".

AttendeeID: **String**
The AttendeeID attribute is also represented as a string. It uniquely identifies the attendee within an event. In this example, it's "1001".

Name: **String**
The Name attribute is represented as a string. It stores the name of the attendee. In this example, it's "Shruti".

Age: **Number**
The Age attribute is represented as a number. It stores the age of the attendee. In this example, it's 27.

Interests: **String Set**
The Interests attribute is represented as a string set. It stores a collection of unique interests of the attendees. In this example, it's ["Professor Talk", "Social Night"].

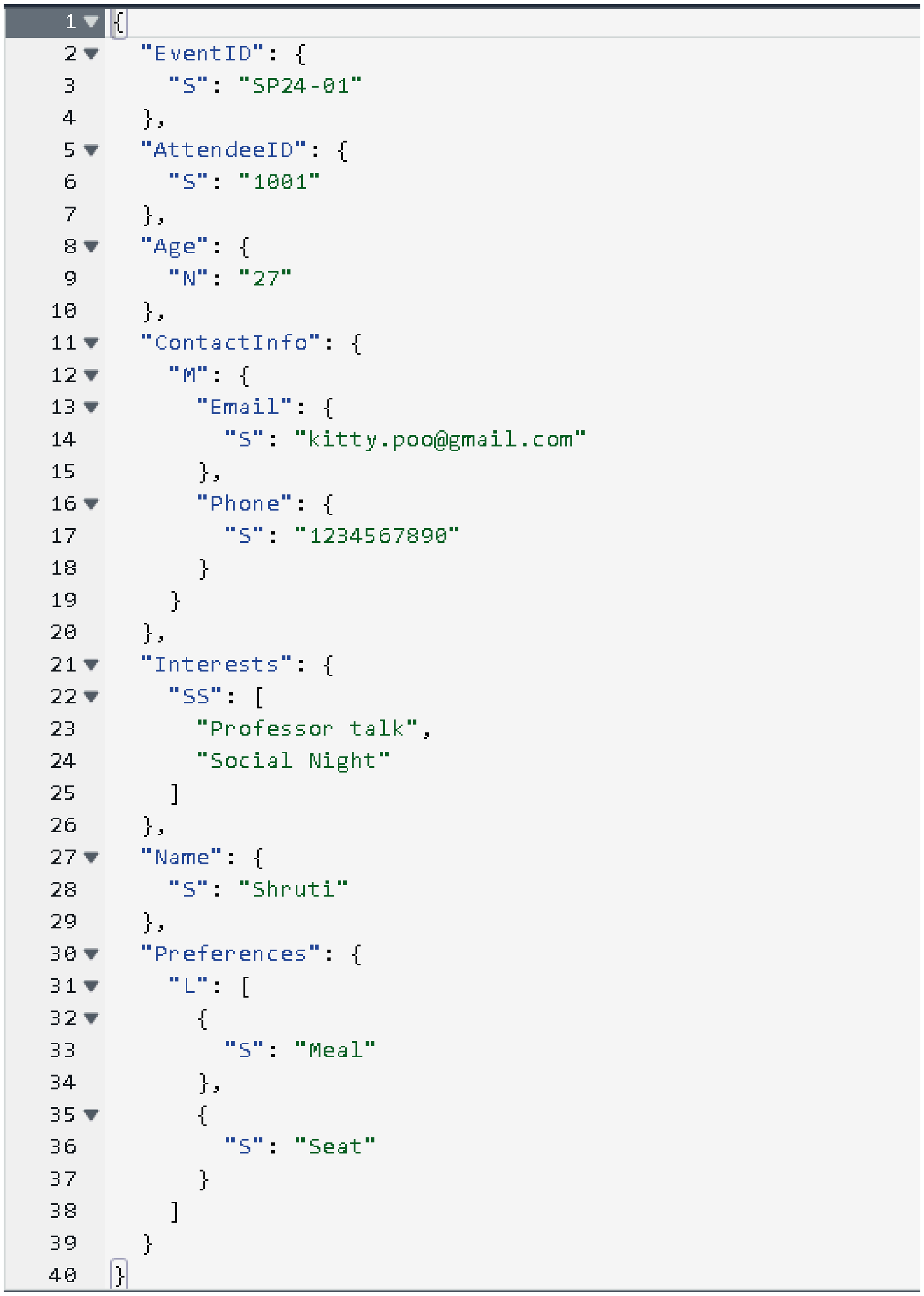
Preferences: **List**

The Preferences attribute is represented as a list. It stores a list of preferences or special accommodations the attendee has requested. In this example, it's ["Meal", "Seat"].

ContactInfo: **Map**
The ContactInfo attribute is represented as a map. It stores contact information of the attendee, such as email and phone number. In this example, it's {"Email": "kitty.poo@gmail.com", "Phone": "1234567890"}.

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)
a. The `JSON view` can be found in the DynamoDB console: DynamoDB > Explore table items > Edit item > JSON view

<u>Screenshot:</u>



```
{
  "EventID": {
    "S": "SP24-01"
  },
  "AttendeeID": {
    "S": "1001"
  },
  "Age": {
    "N": "27"
  },
  "ContactInfo": {
    "M": {
      "Email": {
        "S": "kitty.poo@gmail.com"
      },
      "Phone": {
        "S": "1234567890"
      }
    }
  },
  "Interests": {
    "SS": [
      "Professor talk",
      "Social Night"
    ]
  },
  "Name": {
    "S": "Shruti"
  },
  "Preferences": {
    "L": [
      {
        "S": "Meal"
      },
      {
        "S": "Seat"
      }
    ]
  }
}
```

<u>Solution:</u>

Strings (S): String values are represented as an object with the key "S" followed by the string value.

Example: "Name": {"S": "Shruti"}

Numbers (N): Numeric values are represented as an object with the key "N" followed by the numeric value in string format.

Example: "Age": {"N": "27"}

String Set (SS): Sets of string values are represented as an object with the key "SS" followed by an array of string values.

Example: "Interests": {"SS": ["Professor talk", "Social Night"]}

List (L): Lists are represented as an object with the key "L" followed by an array of elements, where each element is represented according to its data type.

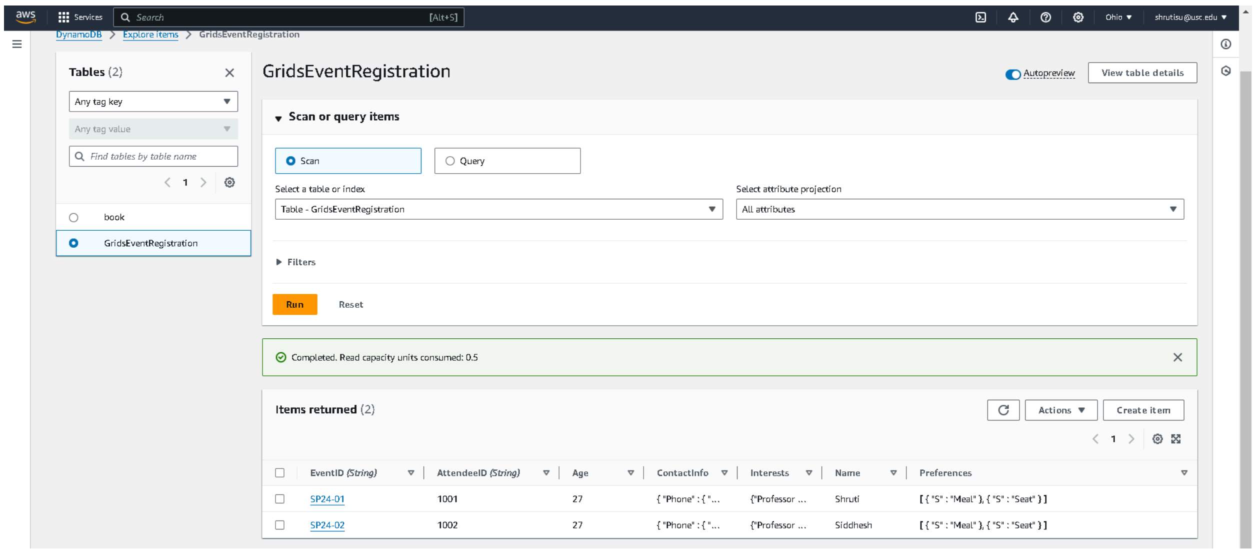
Example: "Preferences": {"L": [{"S": "Meal"}, {"S": "Seat"}]}

Map (M): Maps (nested objects) are represented as an object with the key "M" followed by nested key-value pairs, where each value is represented according to its data type.

Example: "ContactInfo": {"M": {"Email": {"S": "kitty.poo@gmail.com"}, "Phone": {"S": "1234567890"}}}

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

<u>Screenshot for Scan:</u>



GridsEventRegistration

Scan or query items: [✓ Scan] Query

Select a table or index: Table - GridsEventRegistration

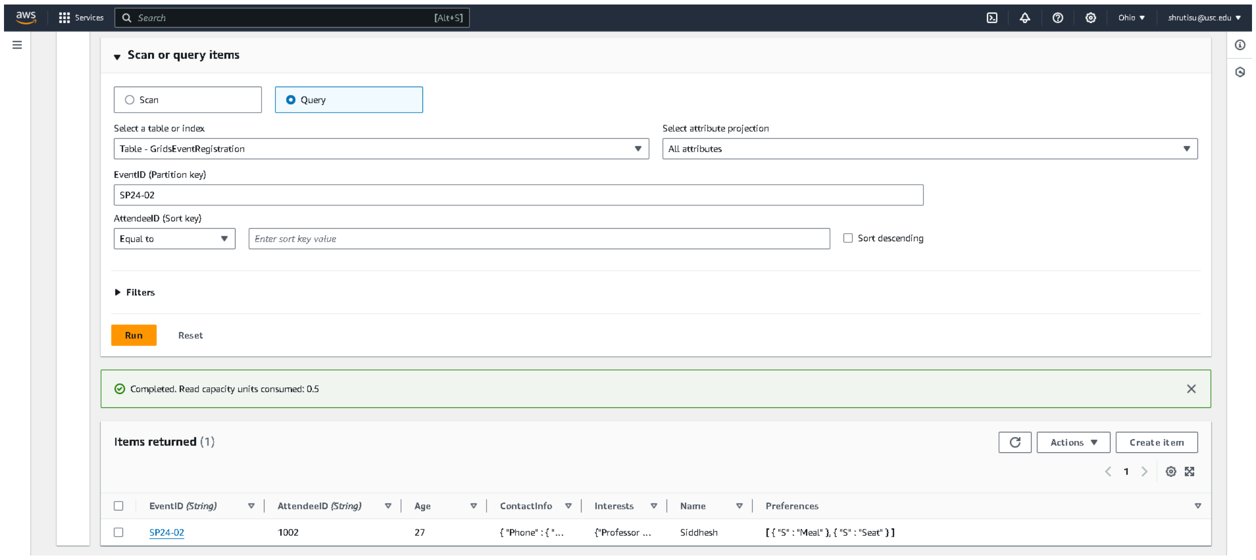
Select attribute projection: All attributes

Completed. Read capacity units consumed: 0.5

Items returned (2)

EventID (String)	AttendeeID (String)	Age	ContactInfo	Interests	Name	Preferences
SP24-01	1001	27	{ "Phone" : { "...	{"Professor ...	Shruti	[{ "S" : "Meal" }, { "S" : "Seat" }]
SP24-02	1002	27	{ "Phone" : { "...	{"Professor ...	Siddhesh	[{ "S" : "Meal" }, { "S" : "Seat" }]

<u>Screenshot for Query:</u>



Scan or query items

Scan [✓ Query]

Select a table or index: Table - GridsEventRegistration

Select attribute projection: All attributes

EventID (Partition key): SP24-02

AttendeeID (Sort key): Equal to | Enter sort key value | ☐ Sort descending

Completed. Read capacity units consumed: 0.5

Items returned (1)

	EventID *(String)*	AttendeeID *(String)*	Age	ContactInfo	Interests	Name	Preferences
☐	SP24-02	1002	27	{ "Phone" : { "...	{"Professor ...	Siddhesh	[{ "S" : "Meal" }, { "S" : "Seat" }]