Create a collection sites(url, dateofaccess). Write a MapReduce function to find the no. of times a site was accessed in a month.

```
db.sites.insertMany([
  { url: "https://example.com", dateofaccess: new Date("2024-10-01") },
  { url: "https://example.com", dateofaccess: new Date("2024-10-15") },
  { url: "https://example.org", dateofaccess: new Date("2024-10-10") },
  { url: "https://example.com", dateofaccess: new Date("2024-09-25") },
  { url: "https://example.org", dateofaccess: new Date("2024-09-15") },
  { url: "https://example.net", dateofaccess: new Date("2024-10-10") }
]);
var mapFunction = function() {
  var month = this.dateofaccess.getMonth() + 1;  // Months are 0-based
  var year = this.dateofaccess.getFullYear();
  var monthYear = year + "-" + (month < 10 ? "0" + month : month);  // Format as YYYY-MM
  emit({ url: this.url, month: monthYear }, 1);
};
var reduceFunction = function(key, values) {
  return Array.sum(values);
};
db.sites.mapReduce(
  mapFunction,
  reduceFunction,
  { out: "site_access_summary" }
);
db.site_access_summary.find();
```

ANS=

Step 1: Create the sites Collection

First, you'll need to insert the data into your sites collection. Here's the provided code to do that:

javascript

Copy code

```
// Insert sample data into the 'sites' collection
db.sites.insertMany([
 { url: "https://example.com", dateofaccess: new Date("2024-10-01") },
 { url: "https://example.com", dateofaccess: new Date("2024-10-15") },
 { url: "https://example.org", dateofaccess: new Date("2024-10-10") },
 { url: "https://example.com", dateofaccess: new Date("2024-09-25") },
 { url: "https://example.org", dateofaccess: new Date("2024-09-15") },
 { url: "https://example.net", dateofaccess: new Date("2024-10-10") }
]);
```

Step 2: Define the Map and Reduce Functions

Next, you will define the Map function to emit the counts for each site accessed during each month and the Reduce function to sum these counts.

javascript

Copy code

```
// Define the map function
var mapFunction = function() {
 var month = this.dateofaccess.getMonth() + 1; // Months are 0-based
 var year = this.dateofaccess.getFullYear();
 var monthYear = year + "-" + (month < 10 ? "0" + month : month); // Format as YYYY-MM
 emit({ url: this.url, month: monthYear }, 1); // Emit the site URL and month
};

// Define the reduce function
var reduceFunction = function(key, values) {
 return Array.sum(values); // Sum the counts for each URL and month
};
```

Step 3: Execute the MapReduce Operation

Now, you can execute the MapReduce operation on the sites collection, outputting the results to a new collection called site_access_summary.

javascript

Copy code

```
// Execute the MapReduce operation
db.sites.mapReduce(
 mapFunction,
 reduceFunction,
 { out: "site_access_summary" } // Output to a new collection
);
```

Step 4: Retrieve and View the Results

Finally, you can retrieve the results from the site_access_summary collection to see how many times each site was accessed in each month.

javascript

Copy code

```
// Find and display the results
db.site_access_summary.find();
```

Expected Output

When you run the above code, the site_access_summary collection should contain documents similar to the following:

json

Copy code

```
{ "_id": { "url": "https://example.com", "month": "2024-10" }, "value": 2 }
{ "_id": { "url": "https://example.org", "month": "2024-10" }, "value": 1 }
{ "_id": { "url": "https://example.net", "month": "2024-10" }, "value": 1 }
{ "_id": { "url": "https://example.com", "month": "2024-09" }, "value": 1 }
{ "_id": { "url": "https://example.org", "month": "2024-09" }, "value": 1 }
```