

RDS Hands-on

Problem Statement:

You work for XYZ Corporation. Your company has multiple branch offices all over the country. It wants to deploy its application on AWS that reads data from users' devices and sends reports to all the branches. The application also requires executing complex relational joins and updates. The company wants high availability for its application.

You are asked to perform the following tasks:

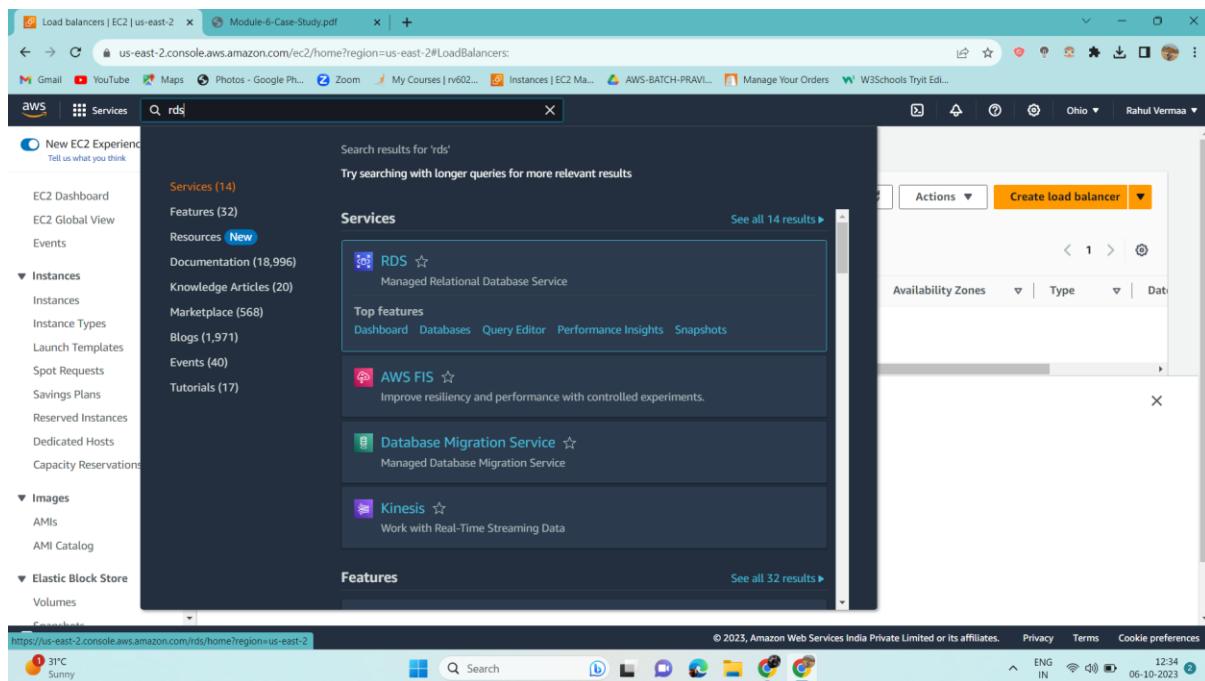
1. Build a highly scalable database; make sure that it is always available and accessible, and manage automated backups for this database with a retention period of 2 days
2. Build a database architecture, which lets your application read data from different regions
3. Establish a database architecture that processes the data collected for near real-time analysis
4. Upload data to a DynamoDB table that you have created
5. Take a backup of the table; delete the table, and then restore it

As the user base of the application increases, your company observes a significant latency in the transfer of data to the branches. It realizes that most of the time, the same kind of data is being fetched and sent to the branches.

Therefore, you are asked to:

6. Find and implement the solution to resolve the latency issues

Search RDS in AWS console

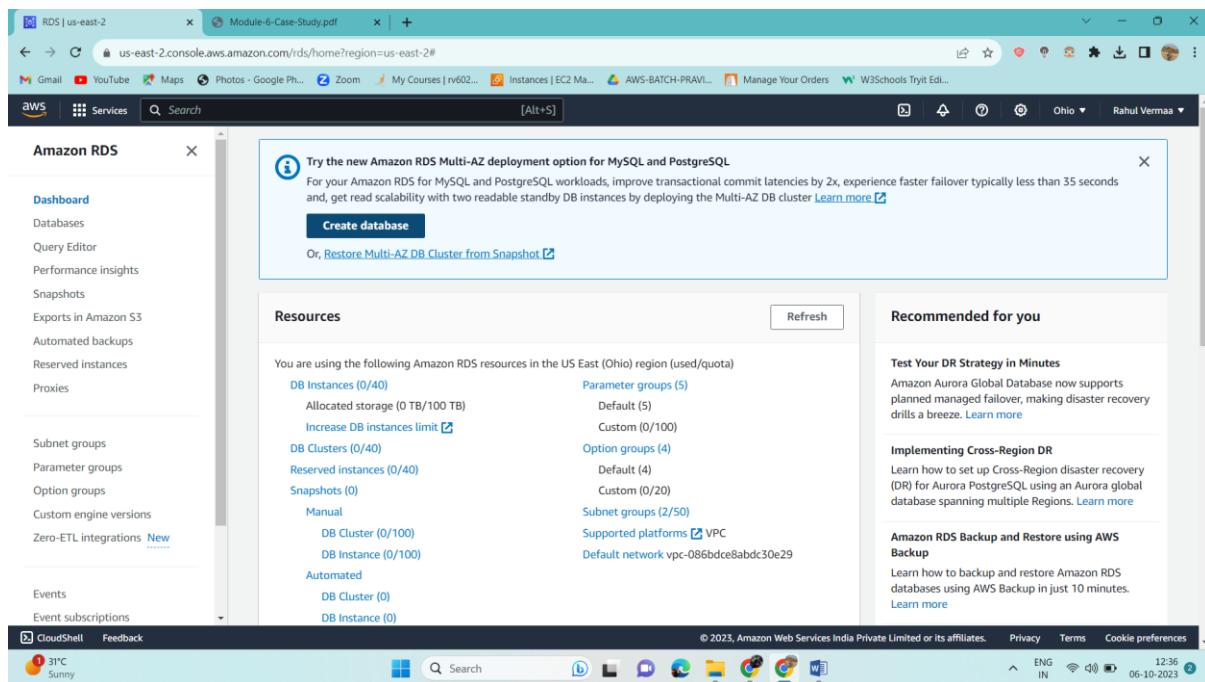


The screenshot shows the AWS console search results for 'rds'. The search bar at the top contains 'rds'. The results are categorized into 'Services' and 'Features'.

- Services (14)**
 - RDS** ☆
Managed Relational Database Service
 - AWS FIS** ☆
Improve resiliency and performance with controlled experiments.
 - Database Migration Service** ☆
Managed Database Migration Service
 - Kinesis** ☆
Work with Real-Time Streaming Data
- Features (32)**
 - Documentation (18,996)
 - Knowledge Articles (20)
 - Marketplace (568)
 - Blogs (1,971)
 - Events (40)
 - Tutorials (17)

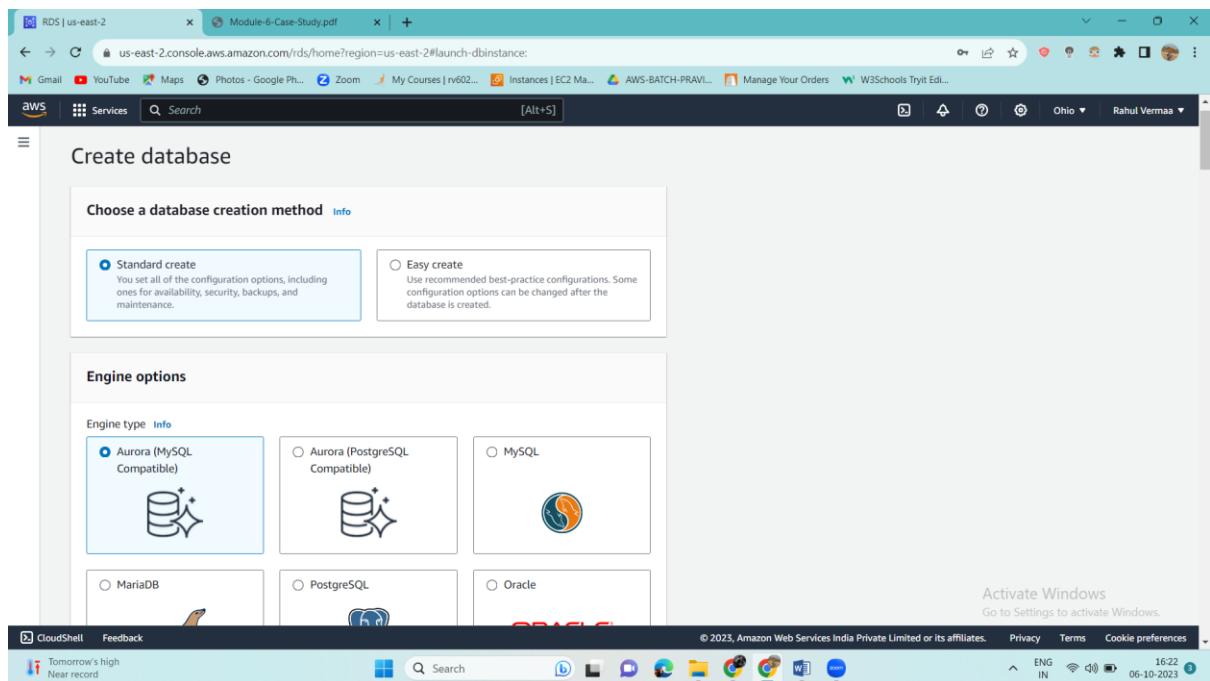
The right side of the screen shows a detailed view of the RDS service, including tabs for Actions, Create load balancer, Availability Zones, Type, and Data.

Now click on create database



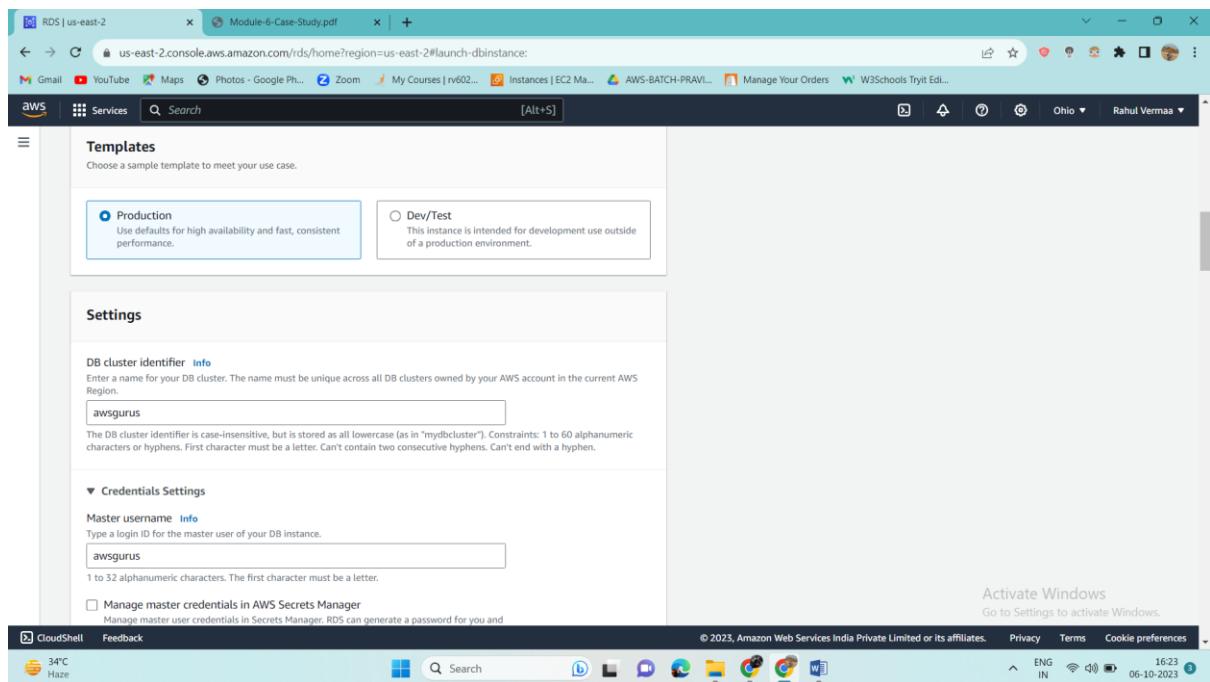
The screenshot shows the Amazon RDS home page for the US East (Ohio) region. The left sidebar includes links for Dashboard, Databases, Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Zero-ETL integrations, Events, and Event subscriptions. The main content area features a callout for the Multi-AZ deployment option and a 'Create database' button. The 'Resources' section lists various Amazon RDS resources, and the 'Recommended for you' section includes links for testing DR strategies, implementing cross-region DR, and using AWS Backup.

Select standard create and engine type- Aurora



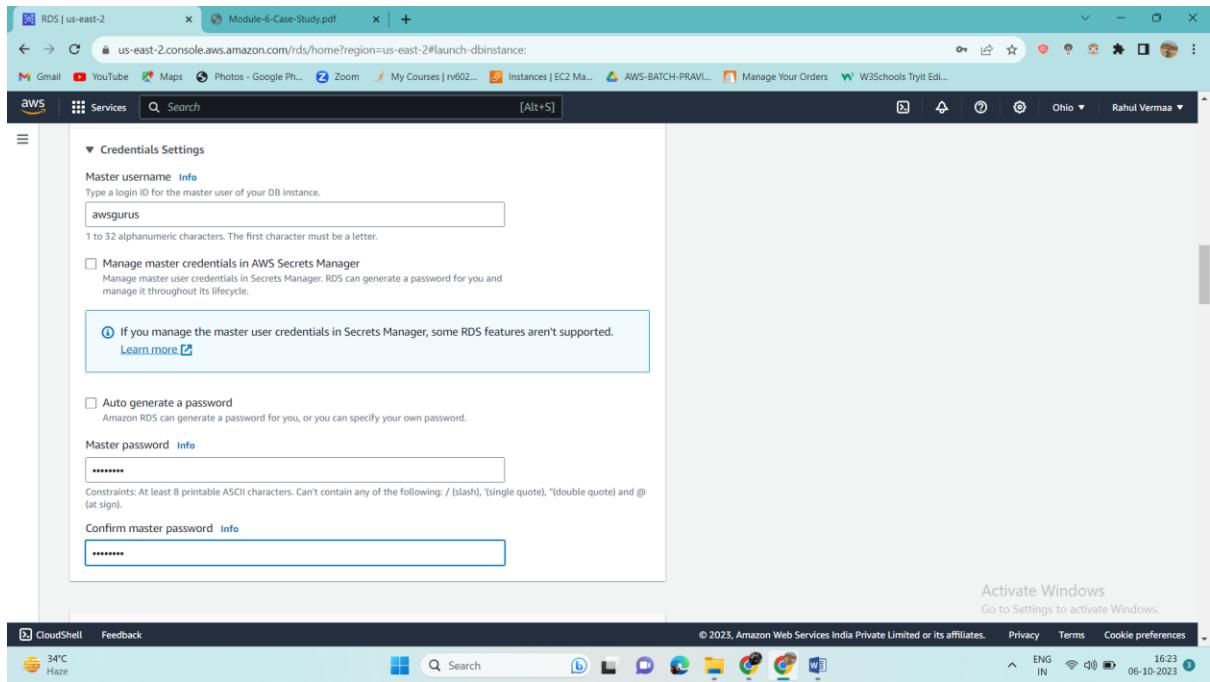
The screenshot shows the 'Create database' wizard in the AWS RDS console. The 'Choose a database creation method' section has 'Standard create' selected. The 'Engine options' section shows various engine types: Aurora (MySQL Compatible) is selected, followed by Aurora (PostgreSQL Compatible), MySQL, MariaDB, PostgreSQL, and Oracle. The Oracle option is highlighted in red. The bottom of the screen shows the Windows taskbar with various pinned icons and the date/time as 06-10-2023.

Templates- Production and DB instance Identifier- awsgurus



The screenshot shows the 'Templates' section in the AWS RDS console. 'Production' is selected as the template. The 'Settings' section shows the 'DB cluster identifier' field set to 'awsgurus'. The 'Master username' field is set to 'awsgurus'. The 'Manage master credentials in AWS Secrets Manager' checkbox is unchecked. The bottom of the screen shows the Windows taskbar with various pinned icons and the date/time as 06-10-2023.

Username- awsgurus and password is also same awsgurus



Master username [Info](#)
Type a login ID for the master user of your DB instance.

1 to 32 alphanumeric characters. The first character must be a letter.

Manage master credentials in AWS Secrets Manager
Manage master user credentials in Secrets Manager. RDS can generate a password for you and manage it throughout its lifecycle.

ⓘ If you manage the master user credentials in Secrets Manager, some RDS features aren't supported. [Learn more](#)

Auto generate a password
Amazon RDS can generate a password for you, or you can specify your own password.

Master password [Info](#)

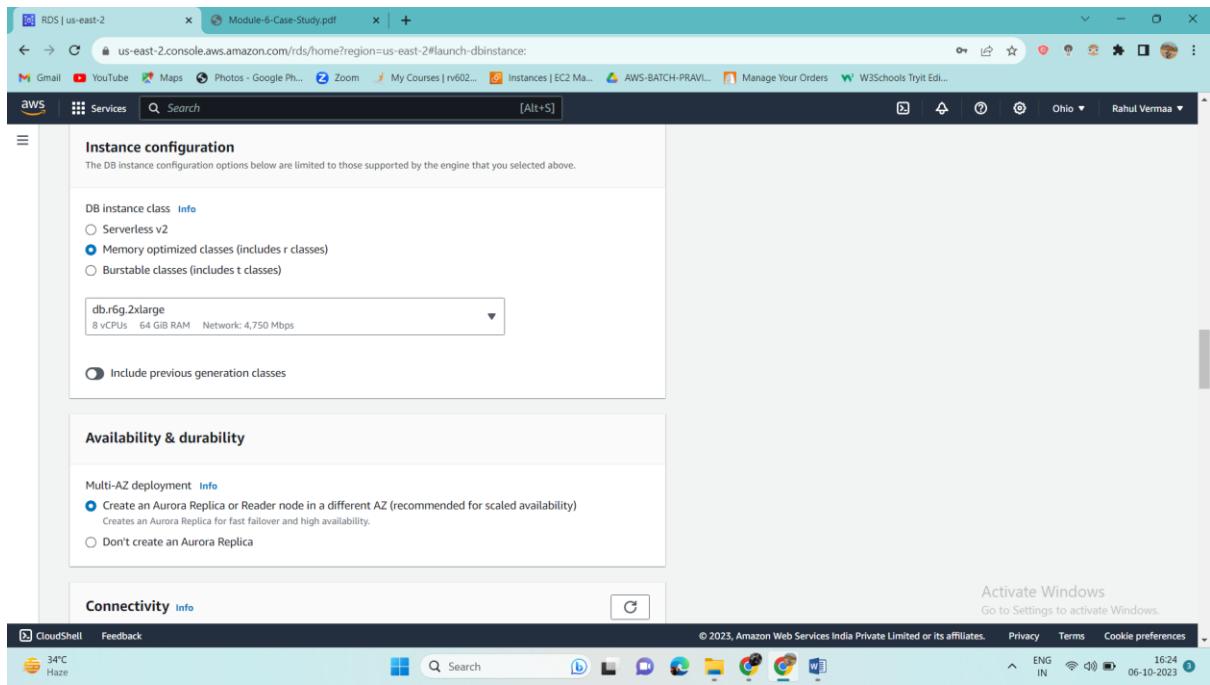
Constraints: At least 8 printable ASCII characters. Can't contain any of the following: / (slash), '(single quote), "(double quote) and @ (at sign).

Confirm master password [Info](#)

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback 34°C Haze 34°C Haze 16:23 06-10-2023

Select multi AZ deployment



DB instance class [Info](#)
 Serverless v2
 Memory optimized classes (includes r classes)
 Burstable classes (includes t classes)

8 vCPUs 64 GB RAM Network: 4,750 Mbps

Include previous generation classes

Availability & durability

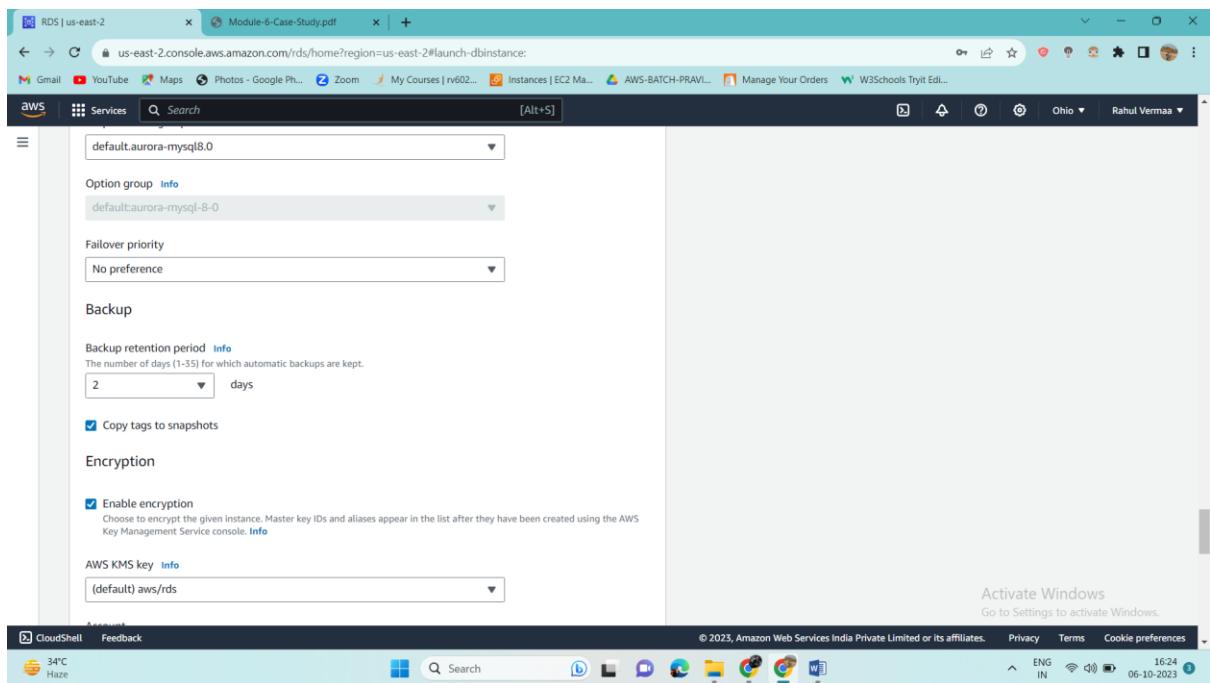
Multi-AZ deployment [Info](#)
 Create an Aurora Replica or Reader node in a different AZ (recommended for scaled availability)
Creates an Aurora Replica for fast failover and high availability.
 Don't create an Aurora Replica

Connectivity [Info](#)

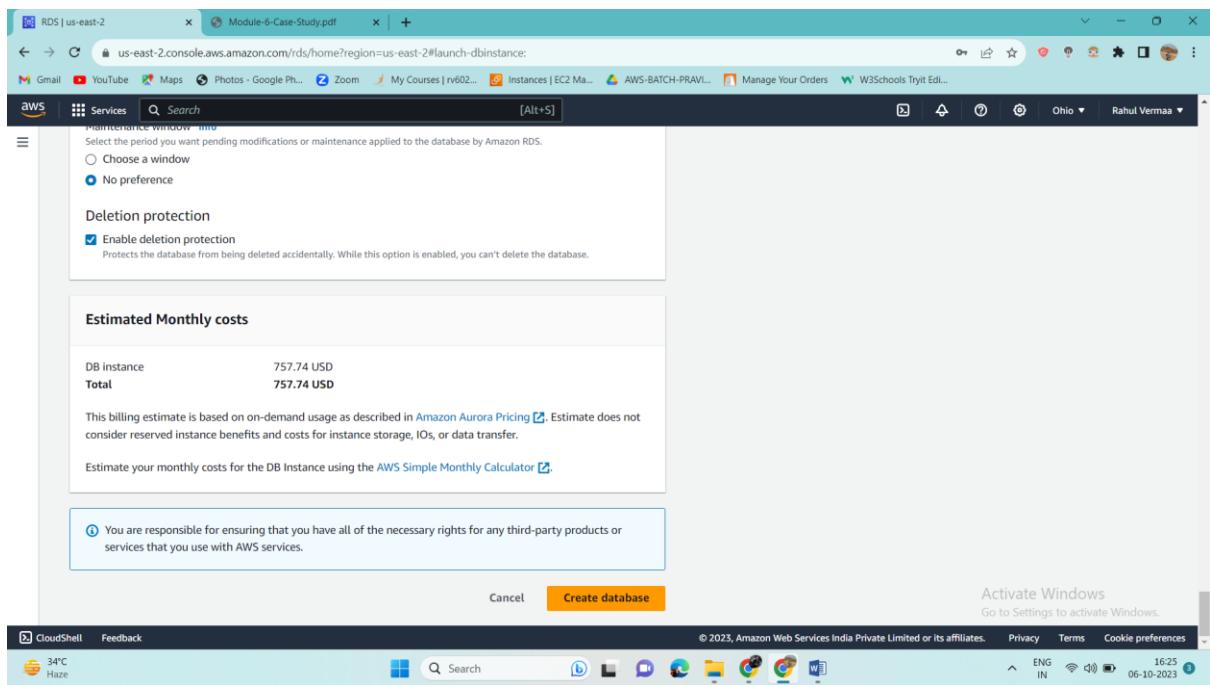
Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback 34°C Haze 34°C Haze 16:24 06-10-2023

Enable autoscaling and backup retention period should be 2 days

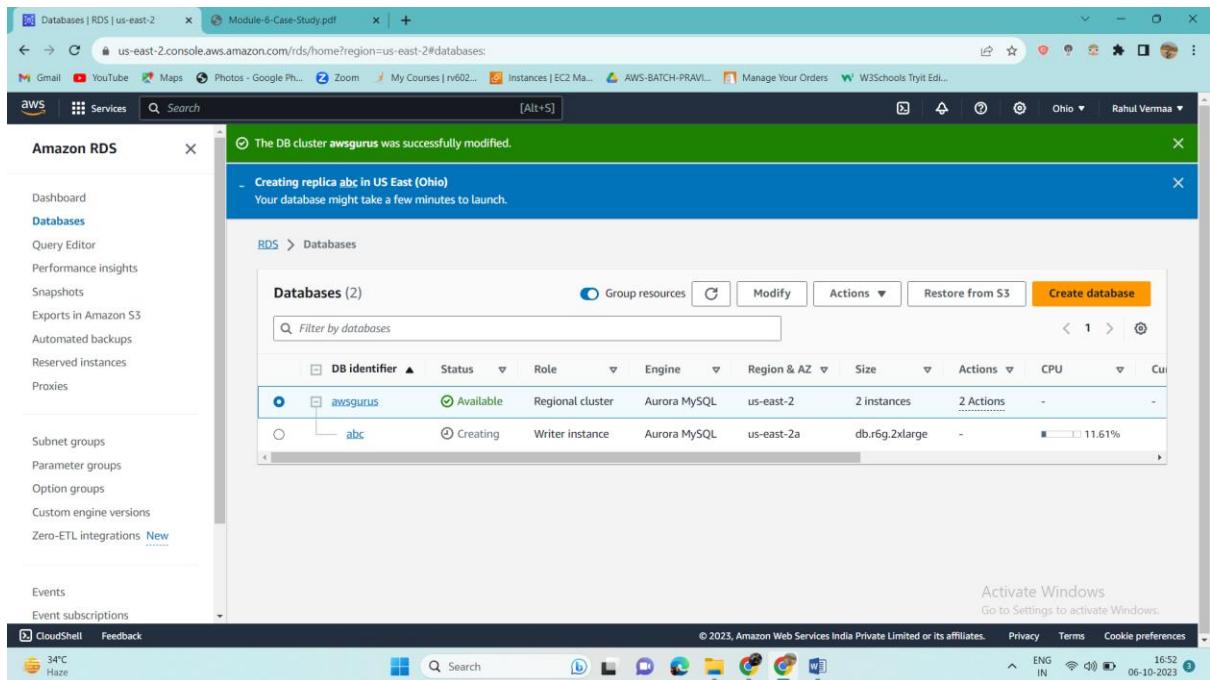


Now click on create database



Our database is created successfully

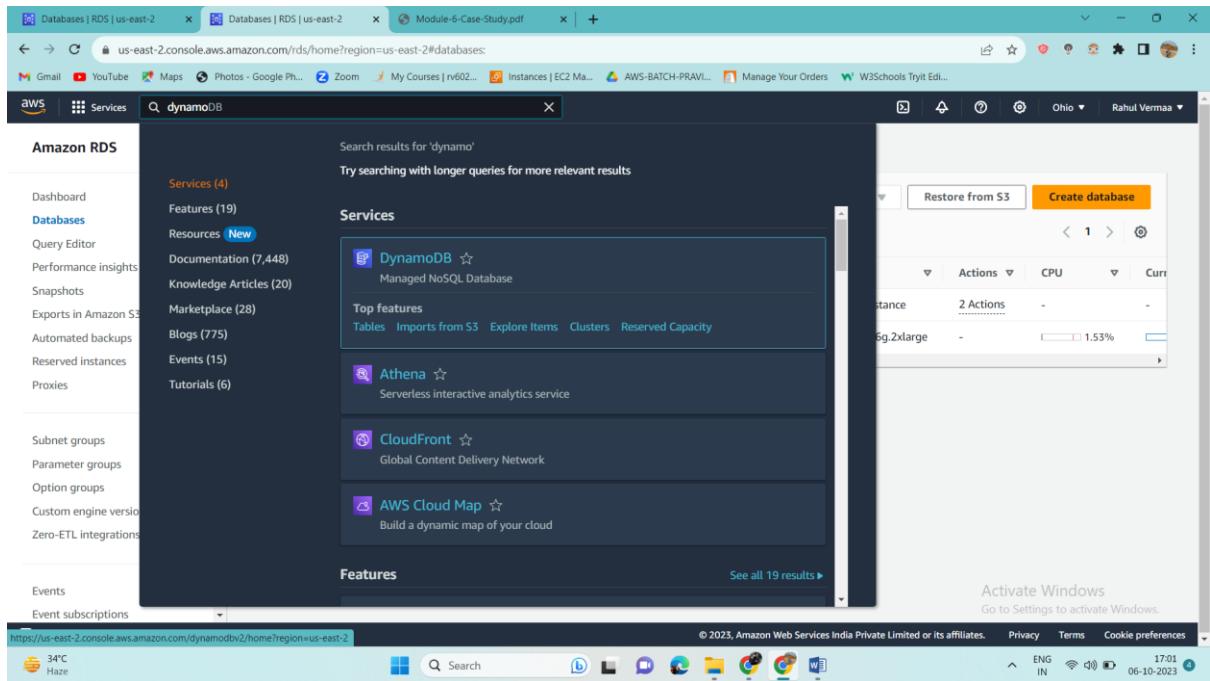
Now will create read replica for that just select your database and go to actions and click add reader



The screenshot shows the AWS RDS console with the following details:

- Region:** us-east-2
- Databases:** 2 (awsgurus, abc)
- awsgurus:** Available, Regional cluster, Aurora MySQL, us-east-2, 2 instances, 2 Actions, CPU 11.61%
- abc:** Creating, Writer instance, Aurora MySQL, us-east-2a, db.r6g.2xlarge

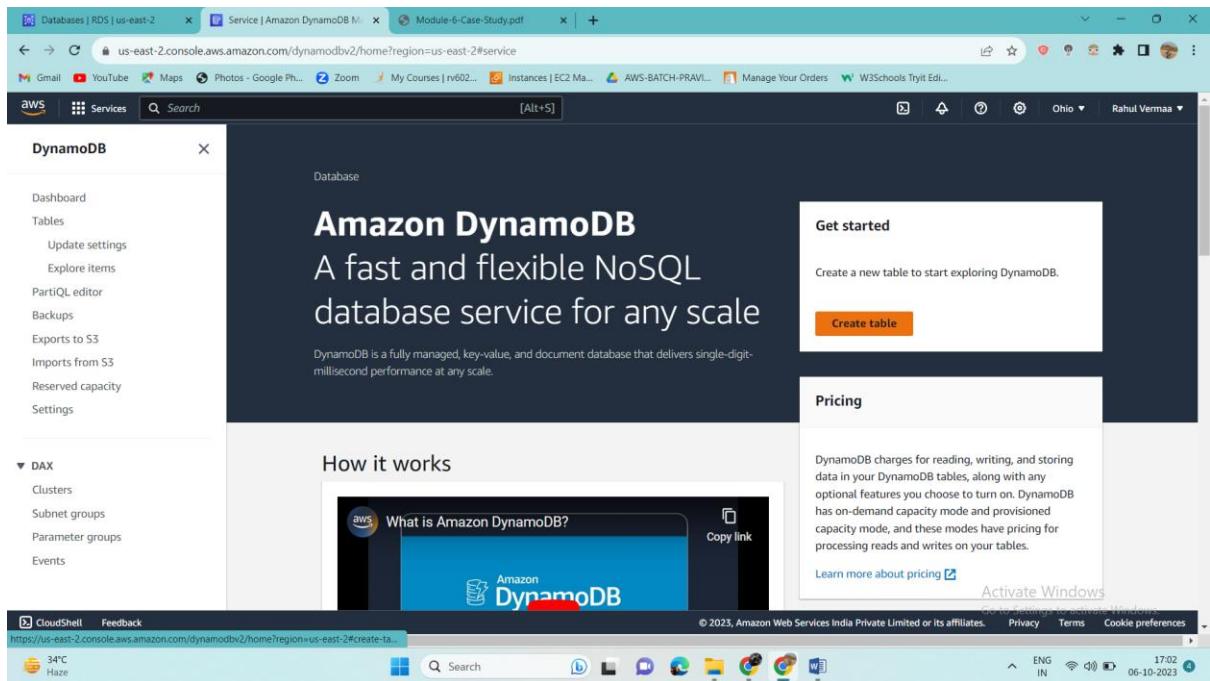
Establish a Database Architecture to process the data collected for near real-time analysis. Now for this we have to create DynamoDB



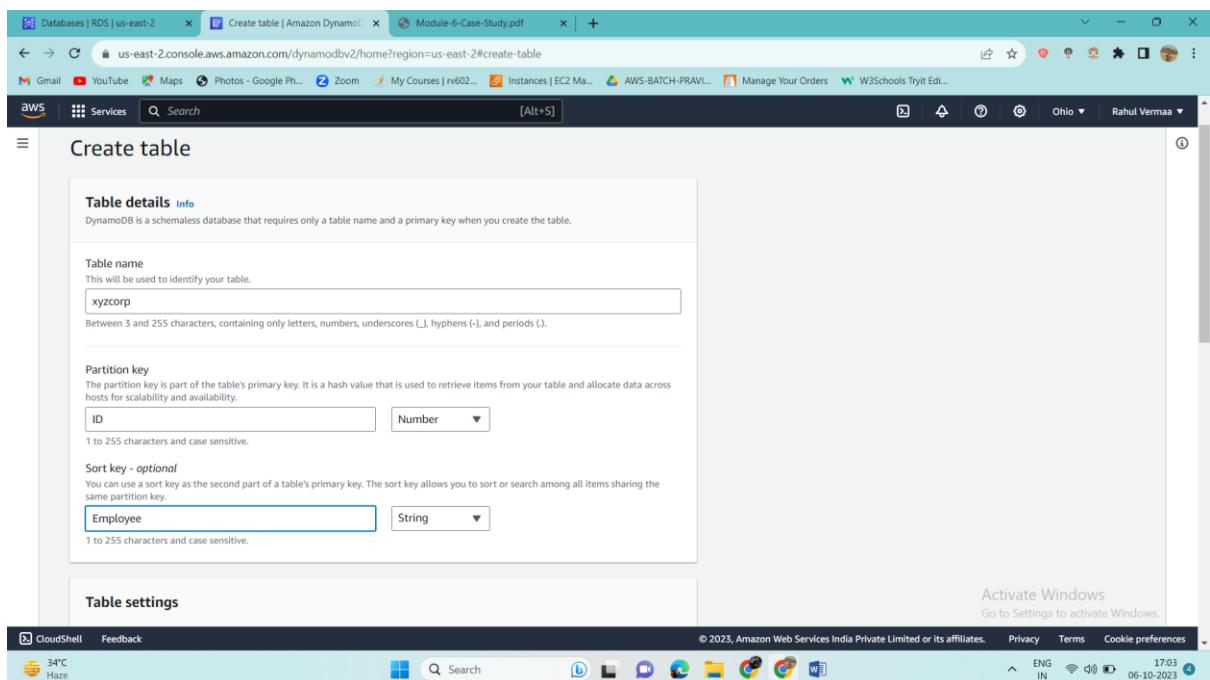
The screenshot shows the AWS Services search results for 'dynamoDB' with the following details:

- Services:** 4 (DynamoDB, Athena, CloudFront, AWS Cloud Map)
- DynamoDB:** Managed NoSQL Database, Top features: Tables, Imports from S3, Explore Items, Clusters, Reserved Capacity

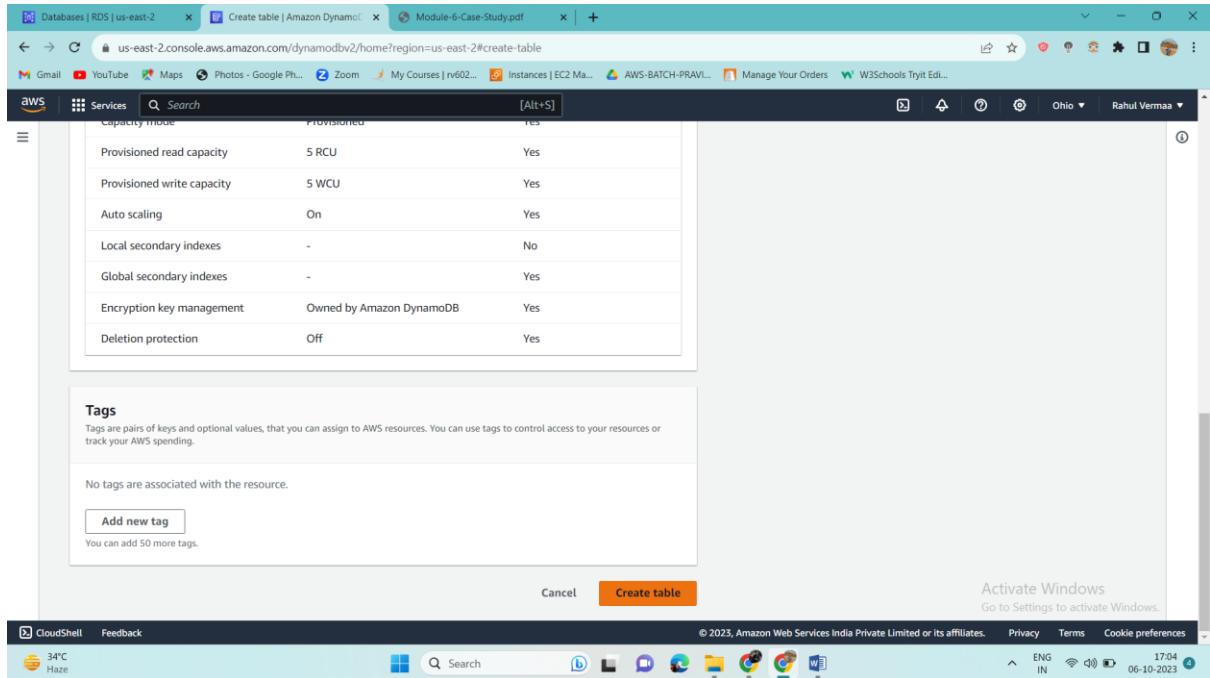
Click on create table



Enter the details and select options



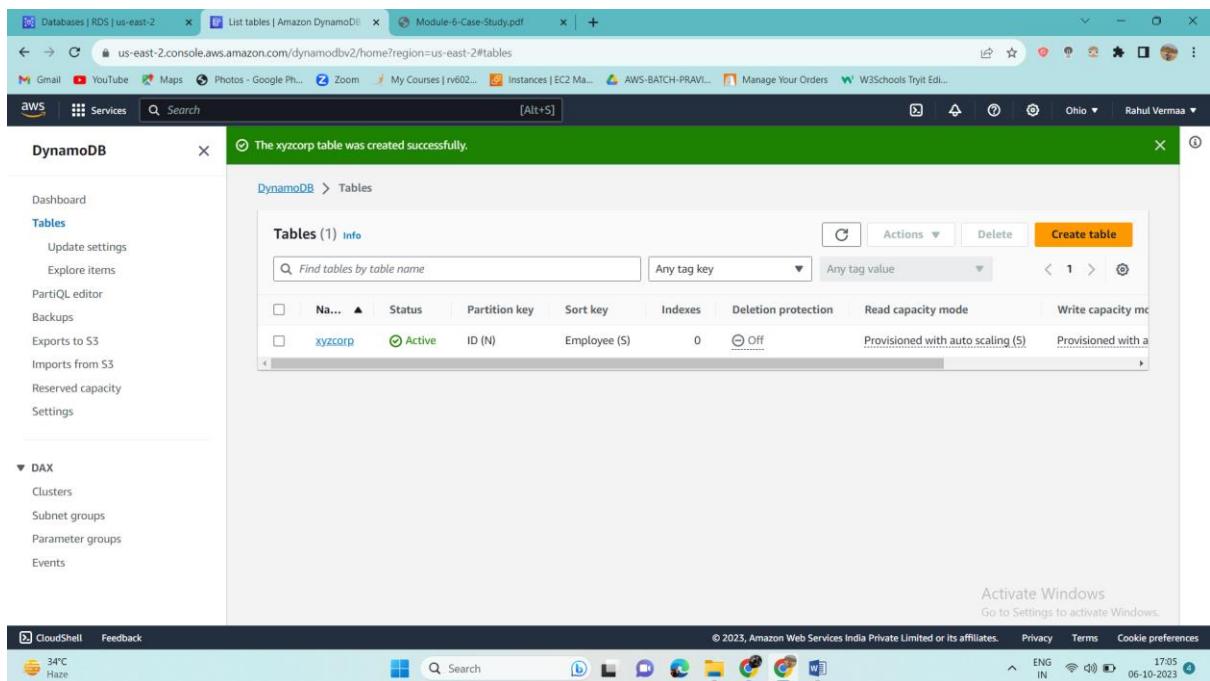
Now click on create table



The screenshot shows the 'Create table' wizard in the AWS DynamoDB console. The 'Capacity mode' is set to 'Provisioned'. The table has 5 RCU for read capacity and 5 WCU for write capacity. Auto scaling is enabled. Local and global secondary indexes are disabled. Encryption key management is set to 'Owned by Amazon DynamoDB'. Deletion protection is off. The 'Tags' section shows that no tags are associated with the resource. A 'Create table' button is at the bottom right.

Our table is successfully created.

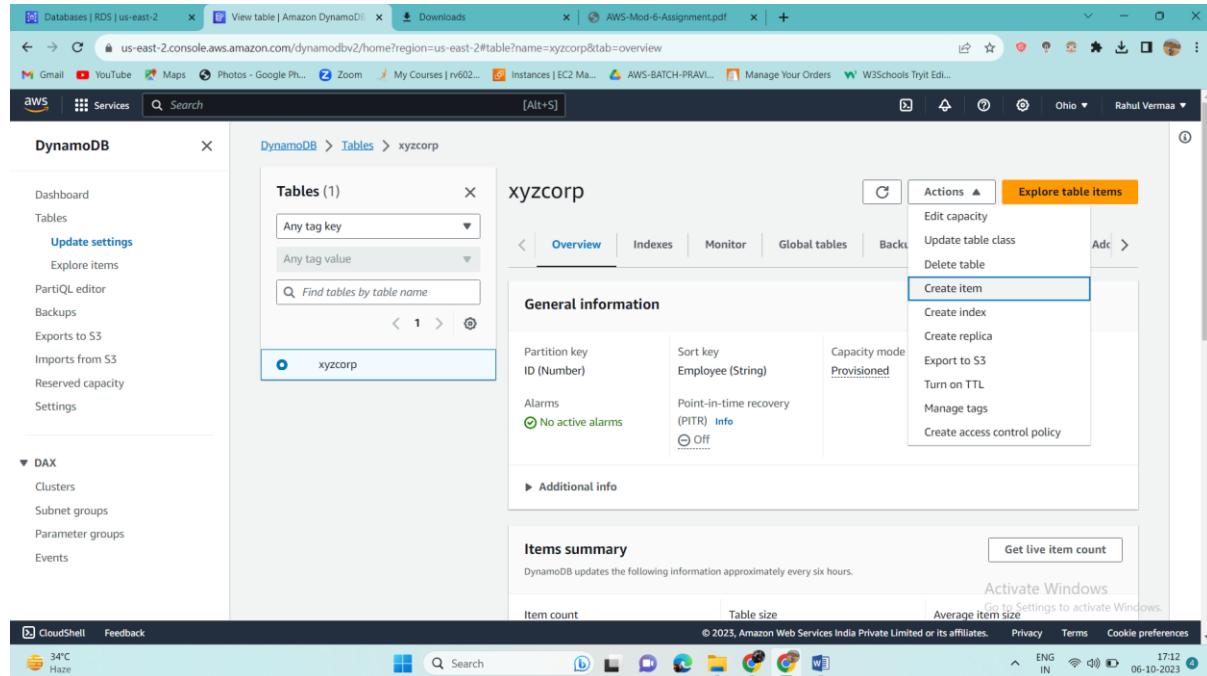
As it was realized that most of the time, same kind of data is being fetched and sent to the branches, so we can do multi-AZ deployment as shown in point# 2. This can also be achieved by enabling **Auto Scaling option**.



The screenshot shows the 'Tables' page in the AWS DynamoDB console. A green success message at the top states 'The xyzcorp table was created successfully.' The 'xyzcorp' table is listed in the table list, showing it is active, has an ID partition key, and is provisioned with auto scaling. The left sidebar shows the 'Tables' section with options like 'Update settings', 'Explore items', and 'Settings'.

Now we will upload data to a DynamoDB table

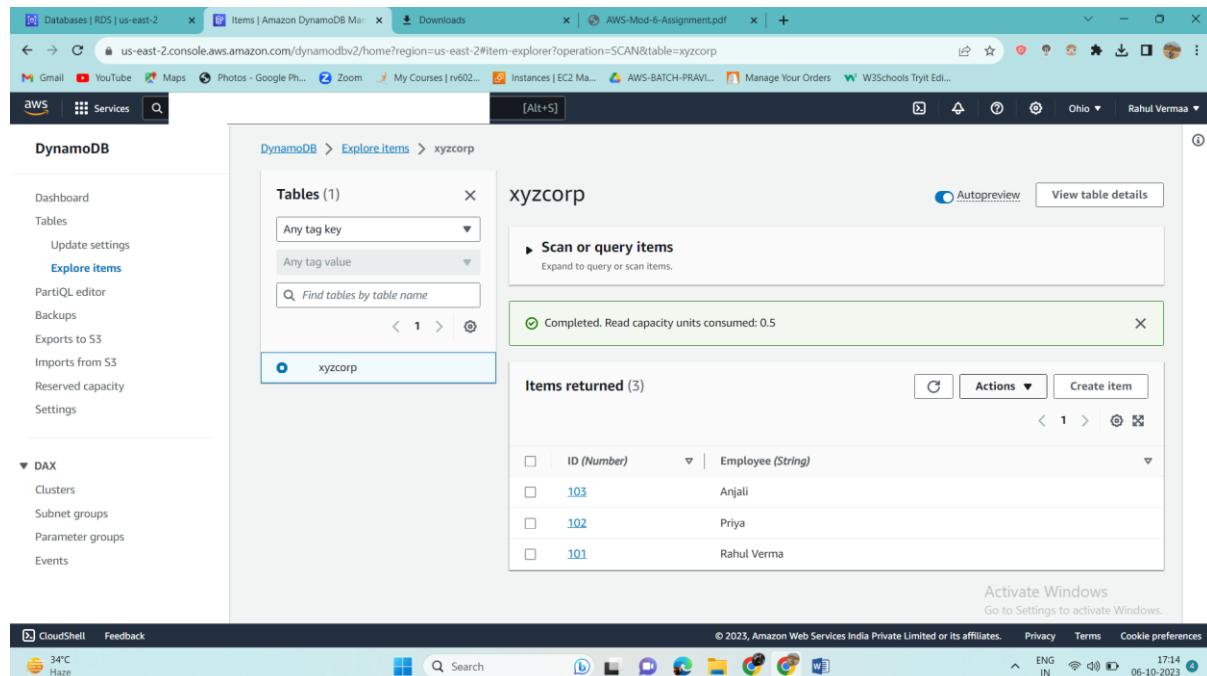
For that select your table go to actions and click on create items



The screenshot shows the AWS DynamoDB console with the 'xyzcorp' table selected. The 'Actions' menu is open, and 'Create item' is highlighted. The table details pane shows the following information:

- General information: Partition key 'ID (Number)', Sort key 'Employee (String)', Capacity mode 'Provisioned'.
- Alarms: 'No active alarms'.
- Items summary: Item count, Table size, Average item size.

We have added data



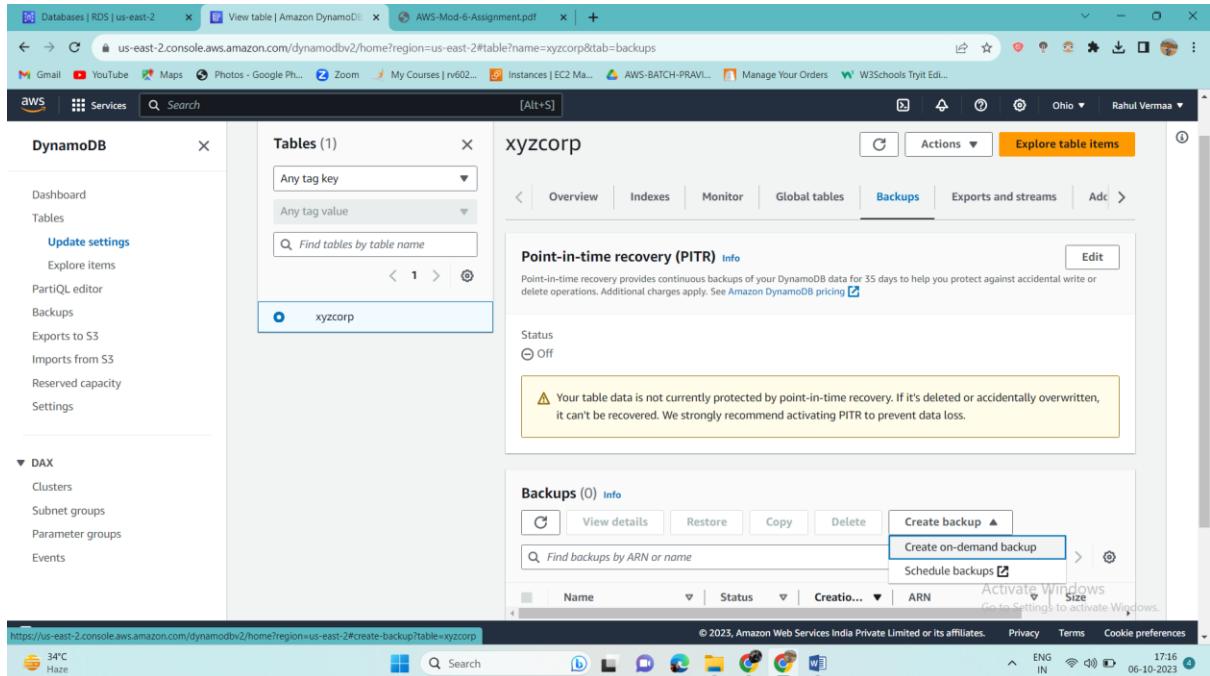
The screenshot shows the AWS DynamoDB console with the 'xyzcorp' table selected. The 'Explore items' section is active, showing the following results:

- Scan or query items: 'Completed. Read capacity units consumed: 0.5'.
- Items returned (3): A table with columns 'ID (Number)' and 'Employee (String)'. The data is as follows:

ID (Number)	Employee (String)
103	Anjali
102	Priya
101	Rahul Verma

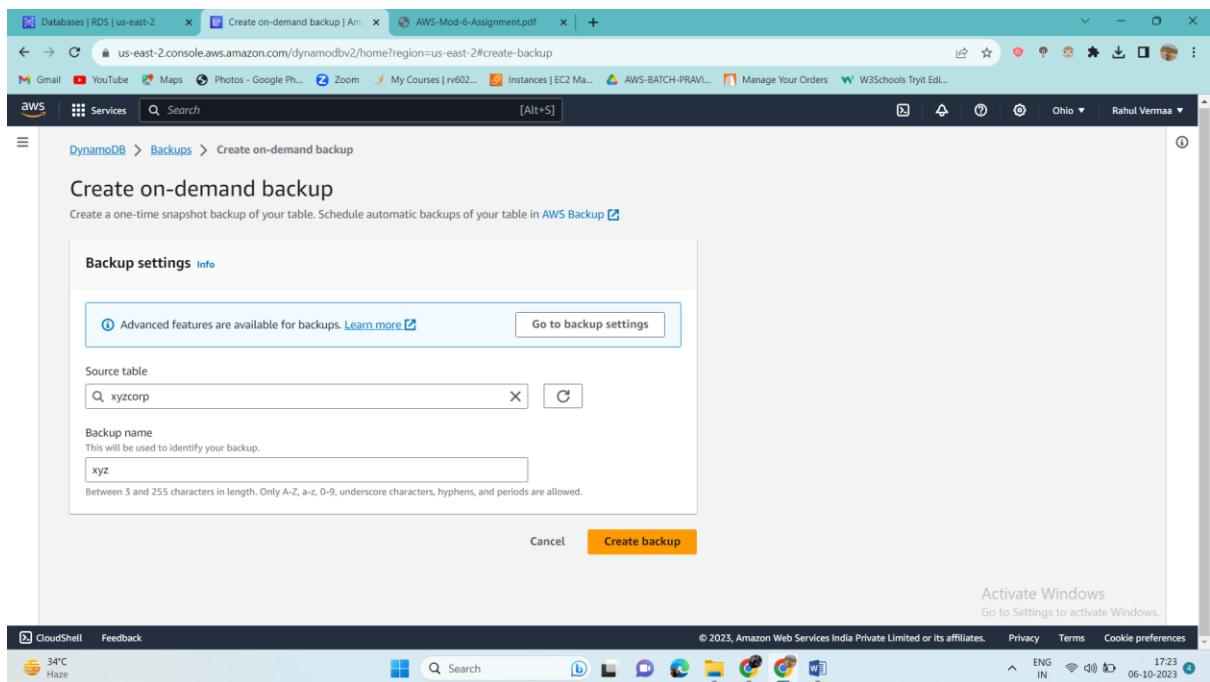
Now will take a backup of the table

Select your table and to backup option and click on- create backup



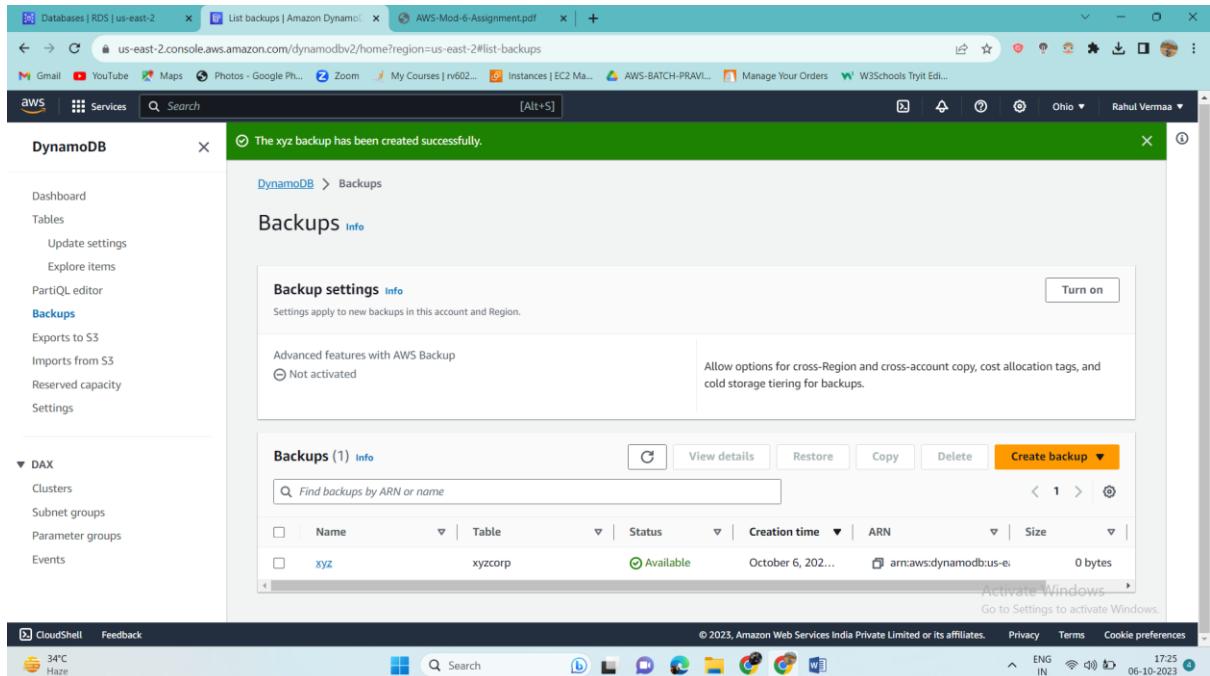
The screenshot shows the AWS DynamoDB console with the 'xyzcorp' table selected. The 'Backups' tab is active. A callout box highlights the 'Create backup' button in the 'Backups (0)' section. The status of the table is shown as 'Off' with a warning message: 'Your table data is not currently protected by point-in-time recovery. If it's deleted or accidentally overwritten, it can't be recovered. We strongly recommend activating PITR to prevent data loss.'

Select options and click on create backup



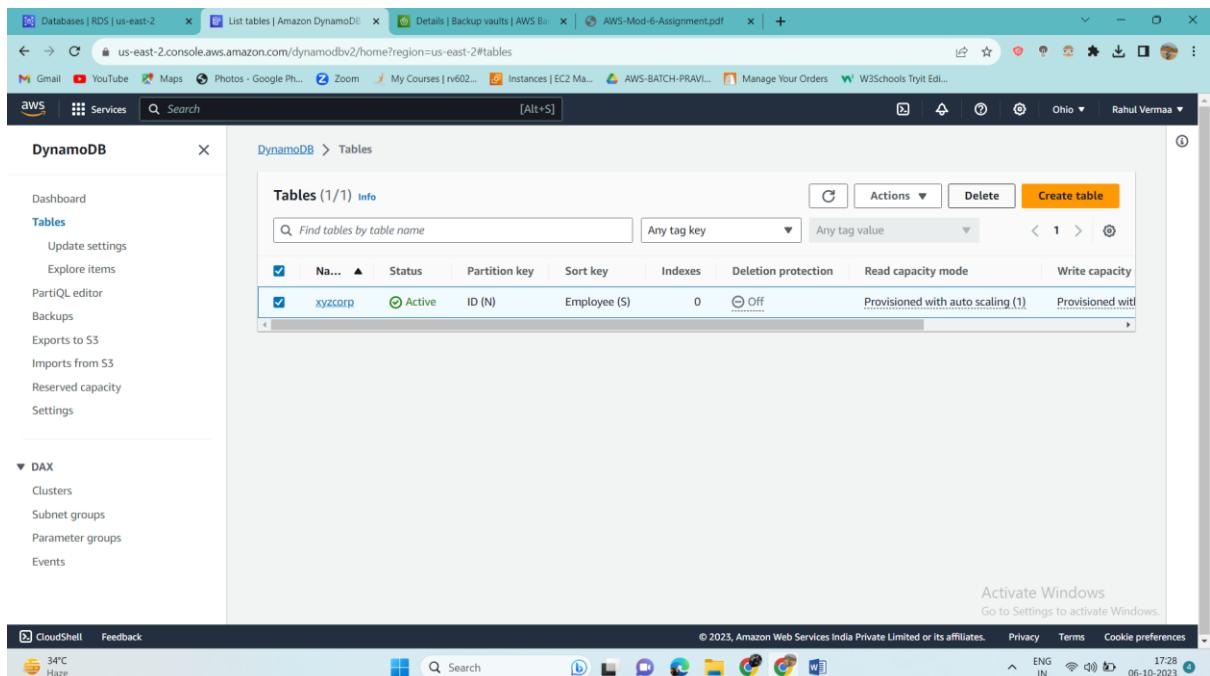
The screenshot shows the 'Create on-demand backup' dialog box. It includes fields for 'Source table' (set to 'xyzcorp') and 'Backup name' (set to 'xyz'). A note at the bottom states: 'Between 3 and 255 characters in length. Only A-Z, a-z, 0-9, underscore characters, hyphens, and periods are allowed.' The 'Create backup' button is highlighted with a callout box.

Backup is successfully created



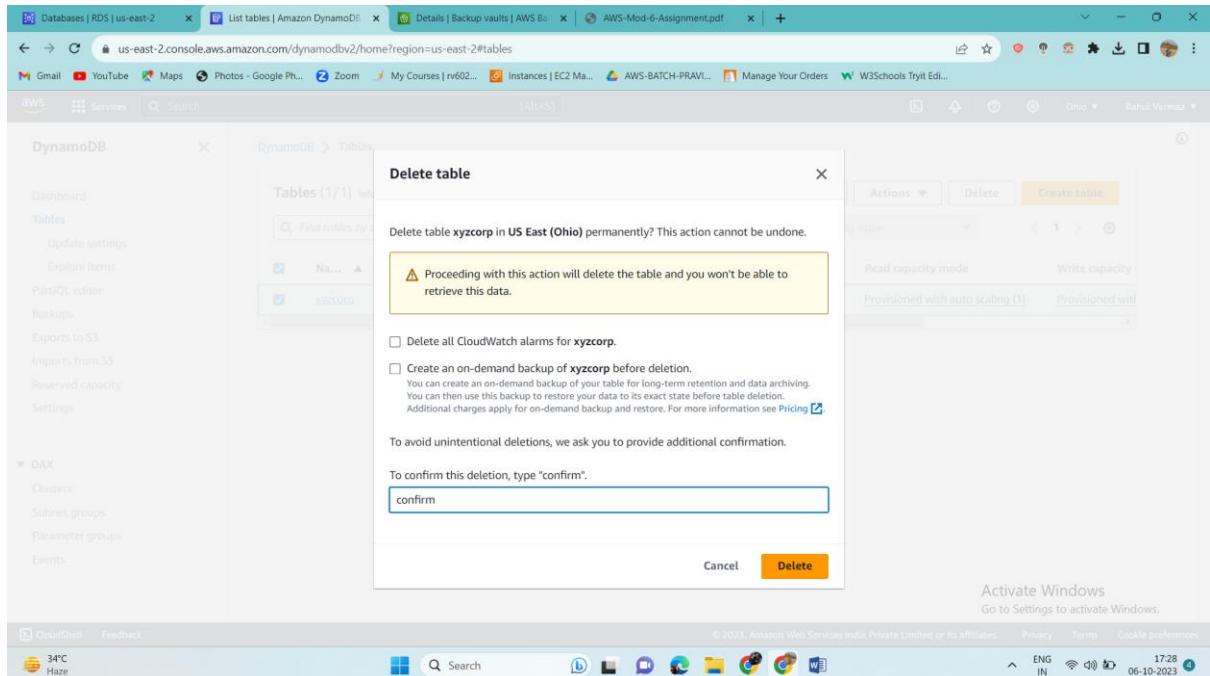
The screenshot shows the AWS DynamoDB console with a successful backup creation message: "The xyz backup has been created successfully." The left sidebar shows the navigation menu for DynamoDB, including "Tables", "Backups", and "DAX". The "Backups" section is selected, showing a table with one backup entry: "xyz" (Status: Available, Creation time: October 6, 2023, ARN: arn:aws:dynamodb:us-east-2:123456789012:table/xyzcorp, Size: 0 bytes). The top navigation bar shows the region as "us-east-2" and the AWS logo.

Now we will delete our table

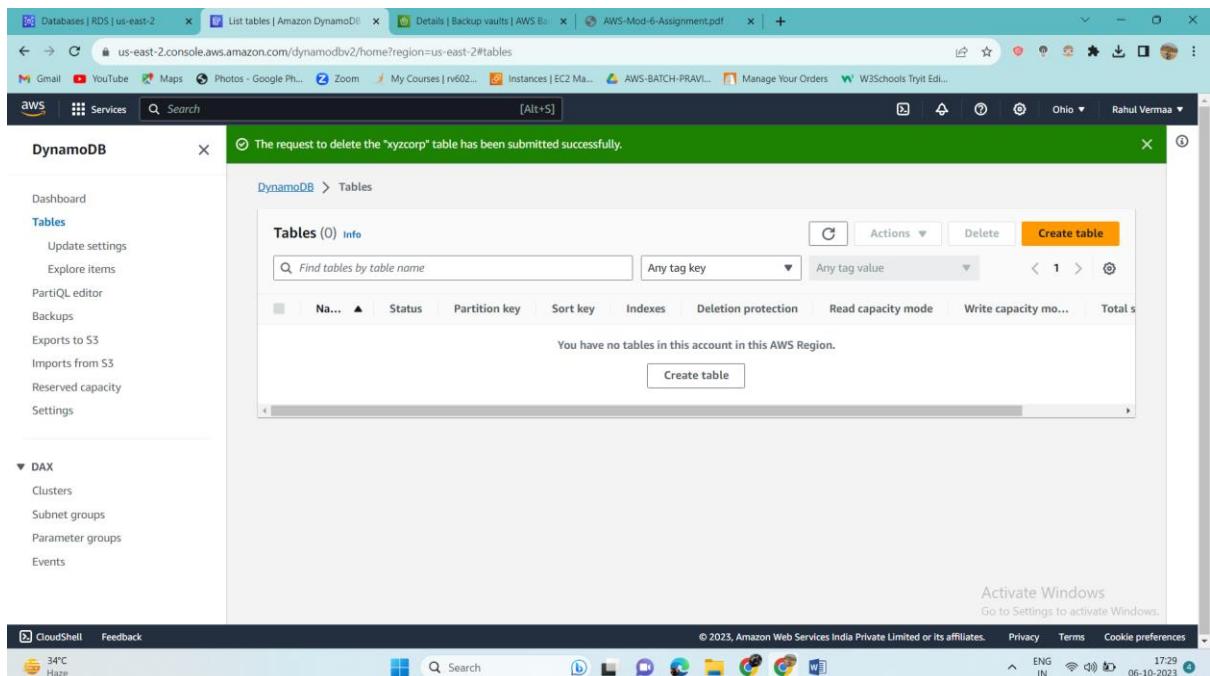


The screenshot shows the AWS DynamoDB console with the "Tables" section selected. The table "xyzcorp" is listed with the following details: Name: xyzcorp, Status: Active, Partition key: ID (N), Sort key: Employee (\$), Indexes: 0, Deletion protection: Off, Read capacity mode: Provisioned with auto scaling (1), Write capacity mode: Provisioned with auto scaling (1). The "Actions" button is highlighted in orange. The left sidebar shows the navigation menu for DynamoDB, including "Tables", "Backups", and "DAX". The top navigation bar shows the region as "us-east-2" and the AWS logo.

While deleting also we select backup option

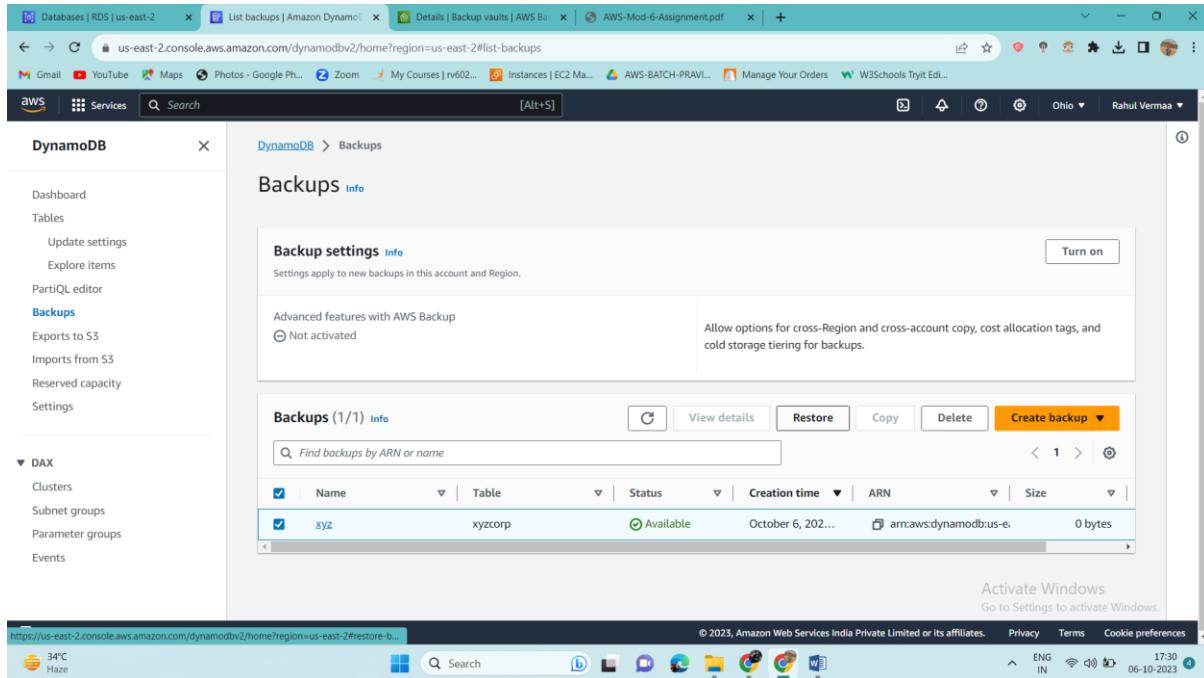


Our table is deleted



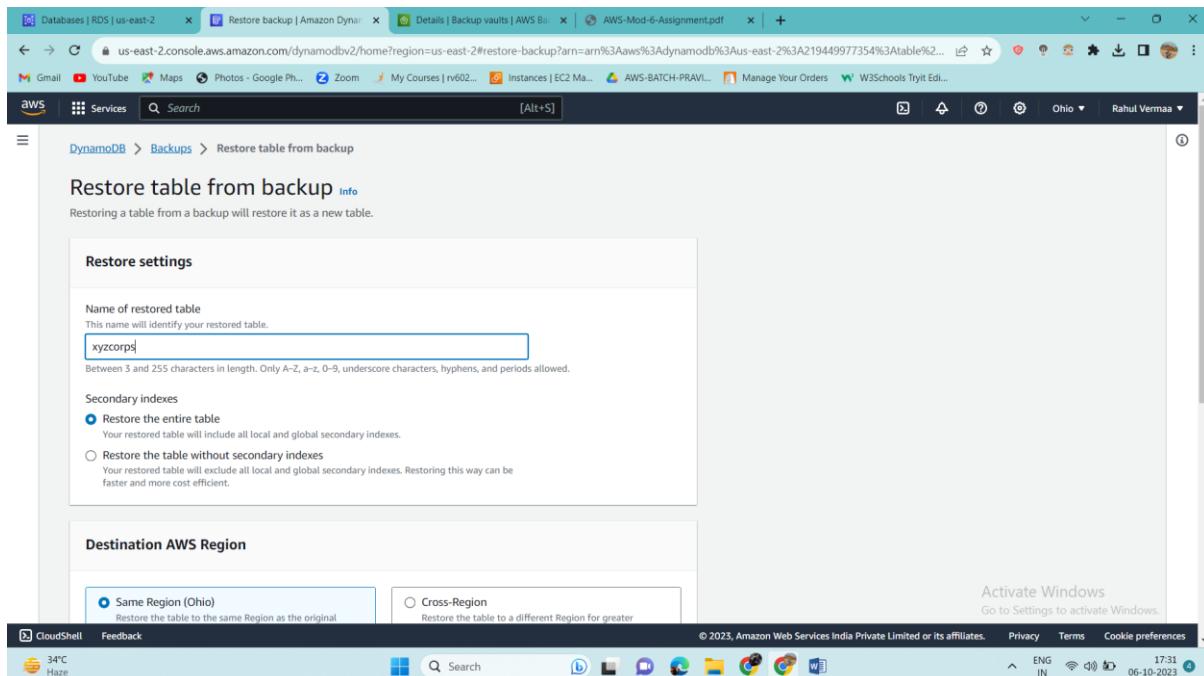
Now we will try to restore it using our backup

Will select our backup and now just click on Restore



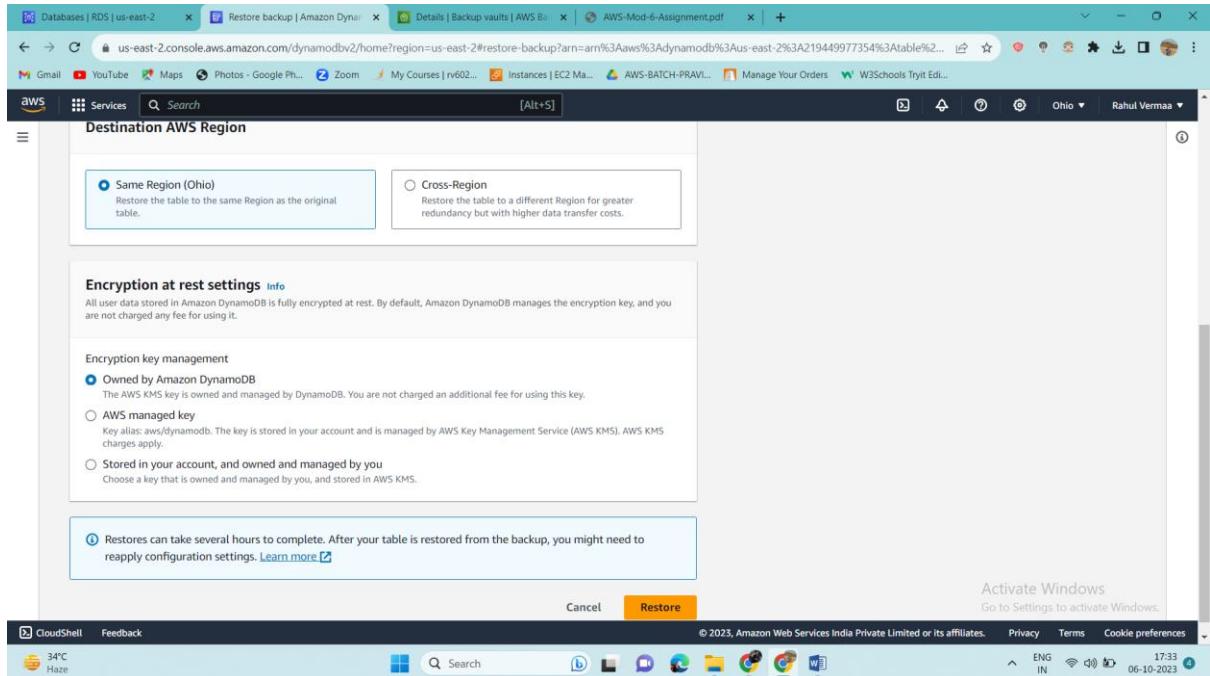
The screenshot shows the AWS DynamoDB Backups page. On the left, a sidebar lists options like Dashboard, Tables, Backups, and DAX. The main area is titled 'Backups' and shows a table with one item: 'xyz' (xyzcorp) - Available. A 'Restore' button is visible in the table header. The status bar at the bottom indicates '34°C Haze' and the date '06-10-2023'.

Will define our table name and we want our entire table to be restored so will select first option

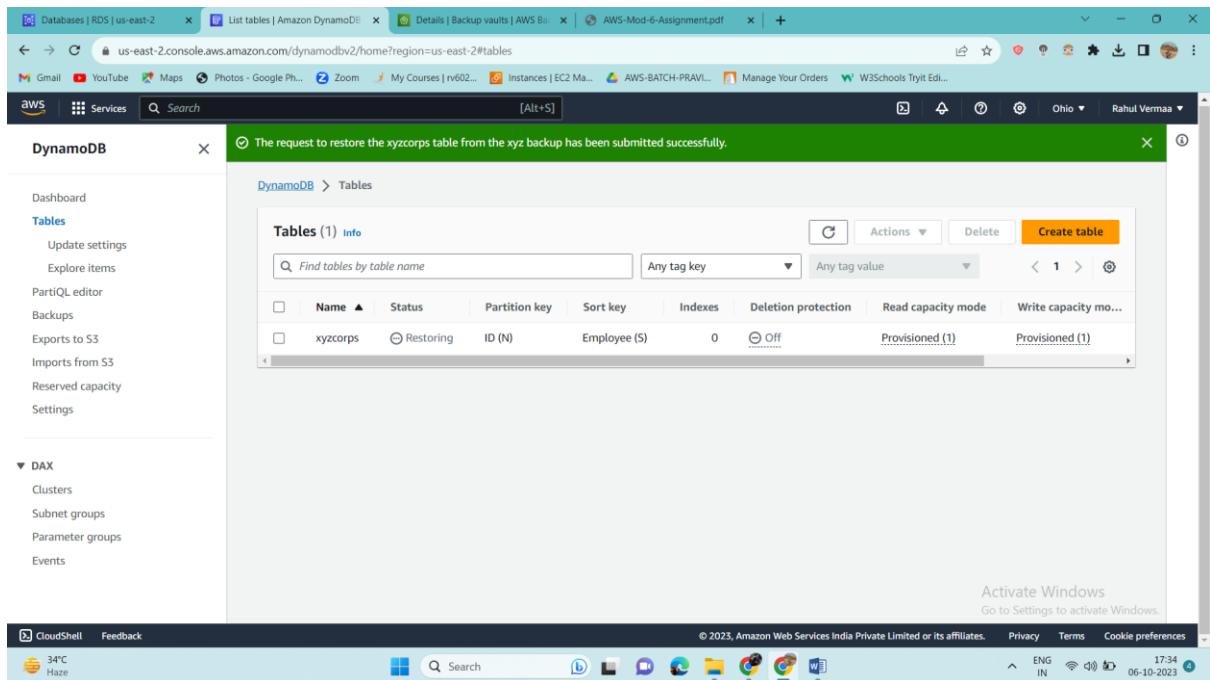


The screenshot shows the 'Restore table from backup' page. It has two main sections: 'Restore settings' and 'Destination AWS Region'. In 'Restore settings', the 'Name of restored table' is set to 'xyzcorps'. Under 'Secondary indexes', the 'Restore the entire table' option is selected. In 'Destination AWS Region', the 'Same Region (Ohio)' option is selected. The status bar at the bottom indicates '34°C Haze' and the date '06-10-2023'.

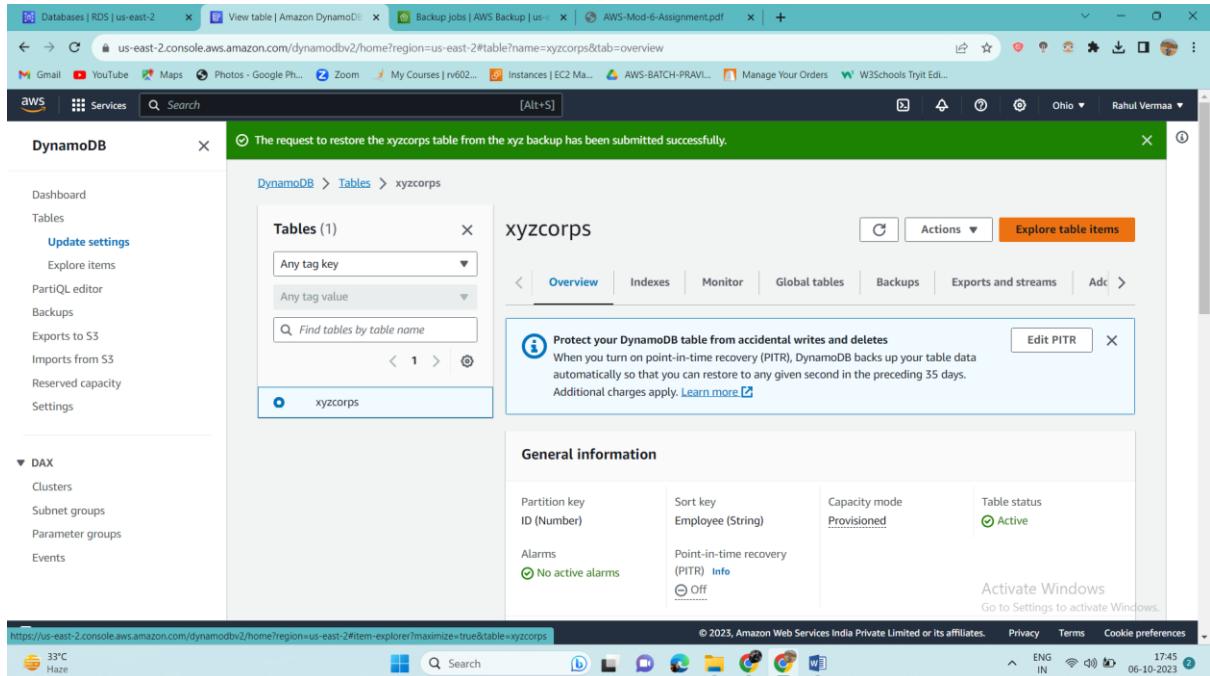
Now we will select same region rest default option and just click on restore



As you can see our table is created and it's status is showing restoring

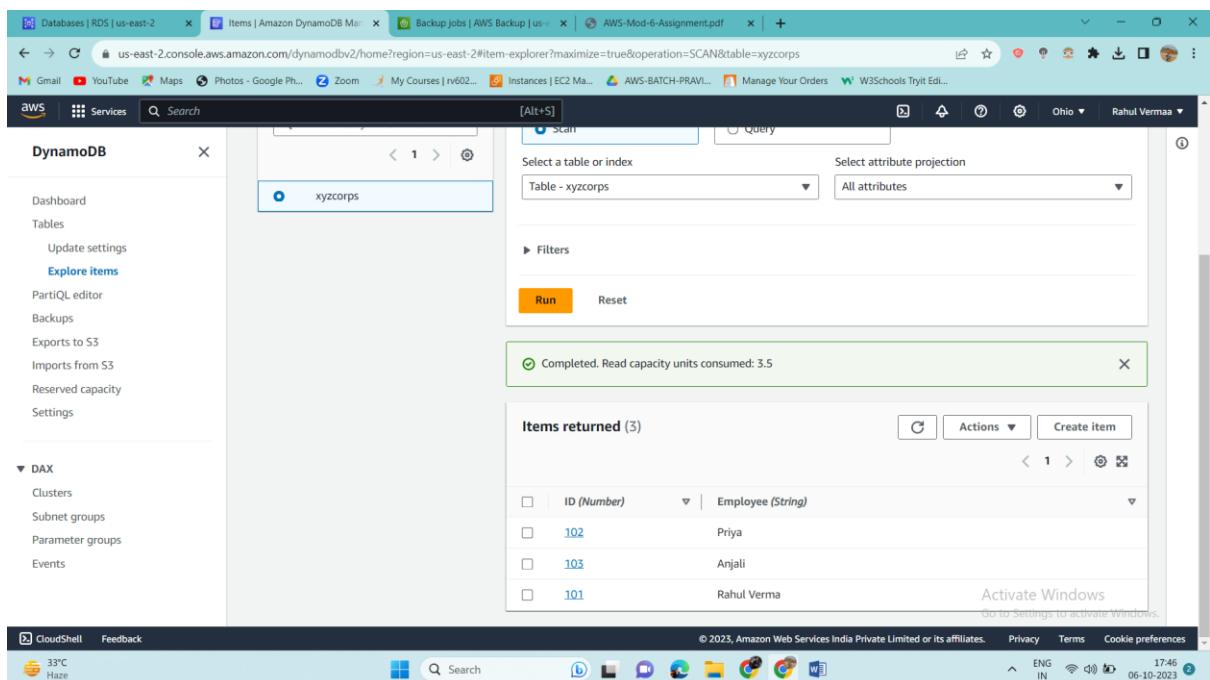


Now to check items select your table when it's status turned to active state and click on expand items



The screenshot shows the AWS DynamoDB console. On the left, the navigation menu is visible with 'Tables' selected. Under 'Tables', 'Update settings' is highlighted. The main content area shows the 'xyzcorps' table. A green banner at the top states: 'The request to restore the xyzcorps table from the xyz backup has been submitted successfully.' Below this, the 'xyzcorps' table is listed in a table structure with columns: 'Tables (1)', 'Any tag key', 'Any tag value', and a search bar. The table row for 'xyzcorps' is selected. To the right, the 'xyzcorps' table details are shown under 'Overview'. It includes a section for 'Protect your DynamoDB table from accidental writes and deletes' with a 'Edit PITR' button. Below this is the 'General information' section, which shows the partition key as 'ID (Number)', sort key as 'Employee (String)', capacity mode as 'Provisioned', and table status as 'Active'. There is also a note to 'Activate Windows' with a link to do so. The bottom of the screen shows the browser's address bar with the URL <https://us-east-2.console.aws.amazon.com/dynamodbv2/home?region=us-east-2#table=xyzcorps>, the AWS logo, and the user 'Rahul Verma'.

And our item is there

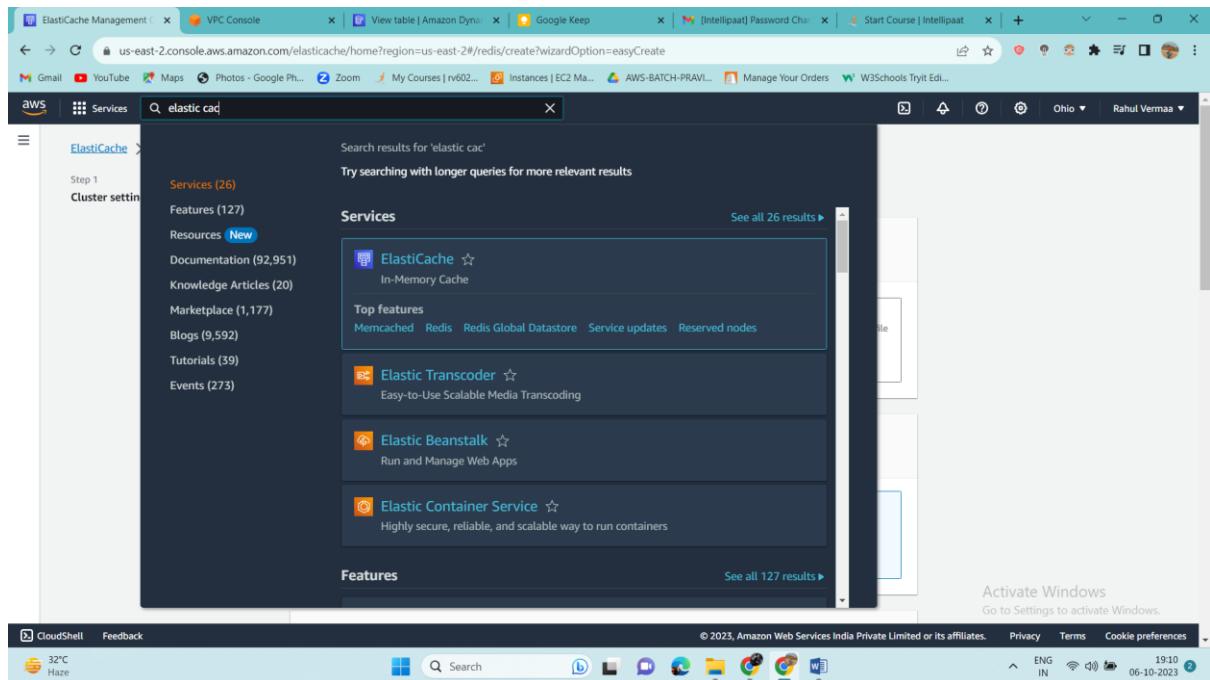


The screenshot shows the AWS DynamoDB console. The navigation menu on the left has 'Explore items' selected under 'Tables'. The main content area shows the 'xyzcorps' table. At the top, there are filters for 'Select a table or index' (set to 'Table - xyzcorps') and 'Select attribute projection' (set to 'All attributes'). Below this is a 'Filters' section with a 'Run' button. A green banner at the bottom of the table area states: 'Completed. Read capacity units consumed: 3.5'. The 'Items returned' section shows three items: ID (Number) 102 (Employee String) Priya, ID (Number) 103 (Employee String) Anjali, and ID (Number) 101 (Employee String) Rahul Verma. The bottom of the screen shows the browser's address bar with the URL <https://us-east-2.console.aws.amazon.com/dynamodbv2/home?region=us-east-2#item-explorer?maximize=true&table=xyzcorps>, the AWS logo, and the user 'Rahul Verma'.

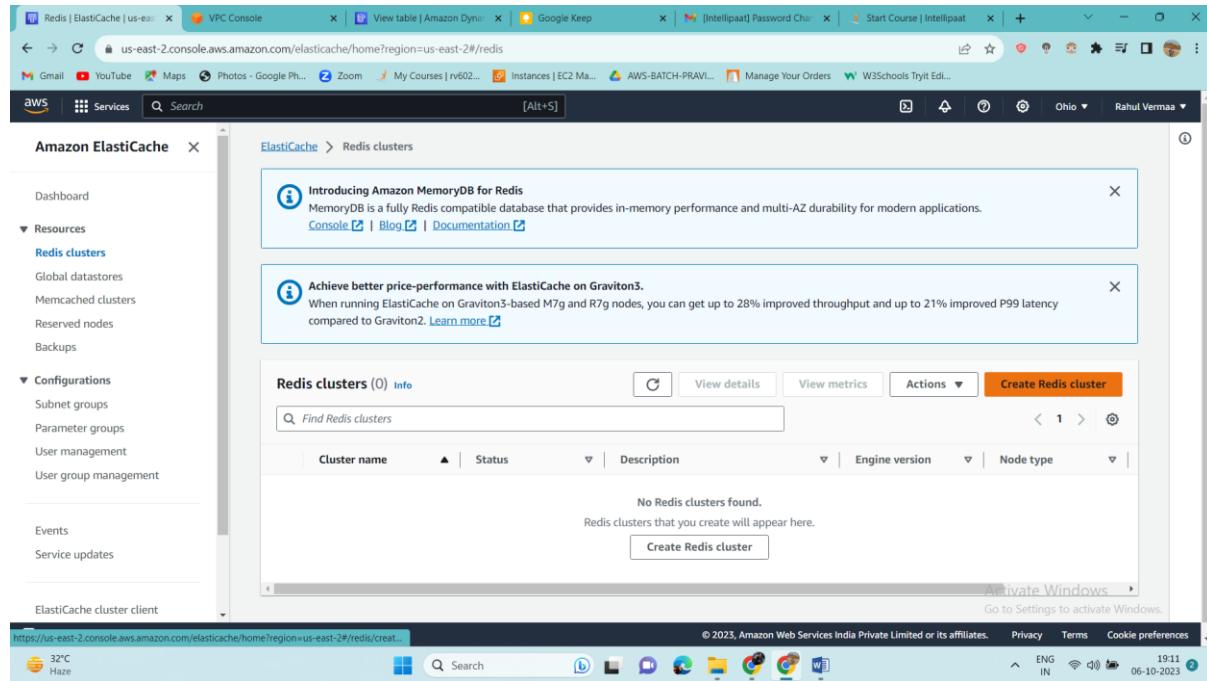
As the user base of the application increases, your company observes a significant latency in the transfer of data to the branches. It realizes that most of the time, the same kind of data is being fetched and sent to the branches. Therefore, you are asked to:

6. Find and implement the solution to resolve the latency issues

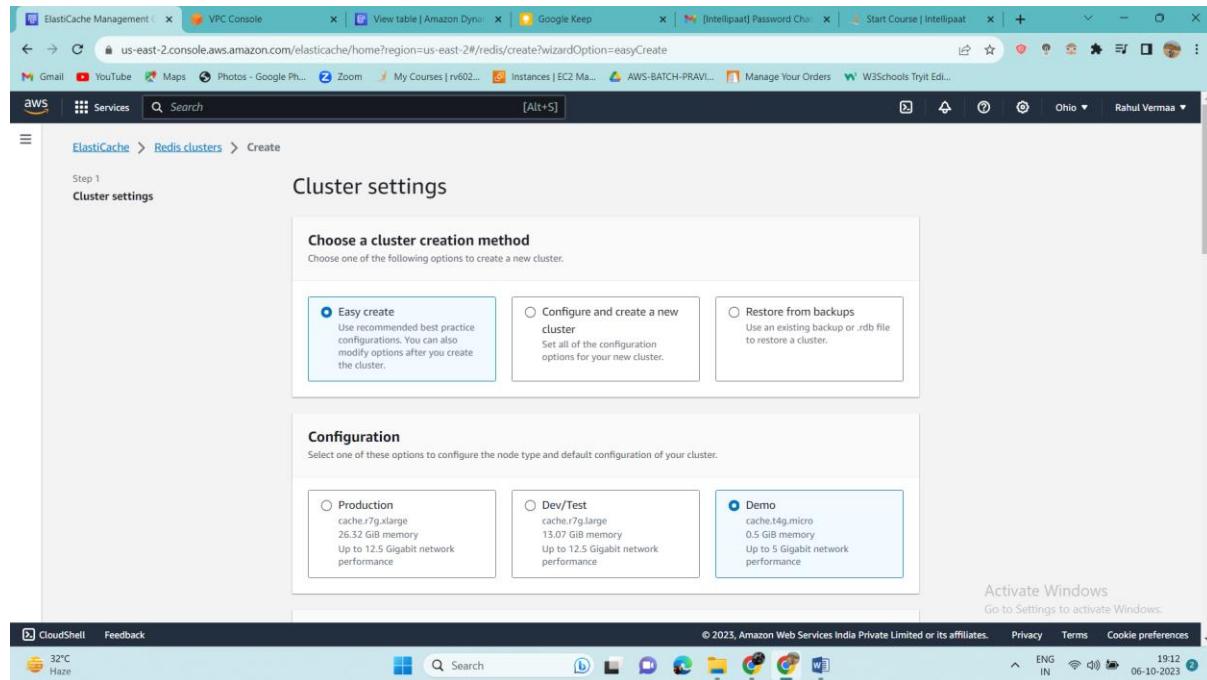
We can use elastic cache cluster in RDS



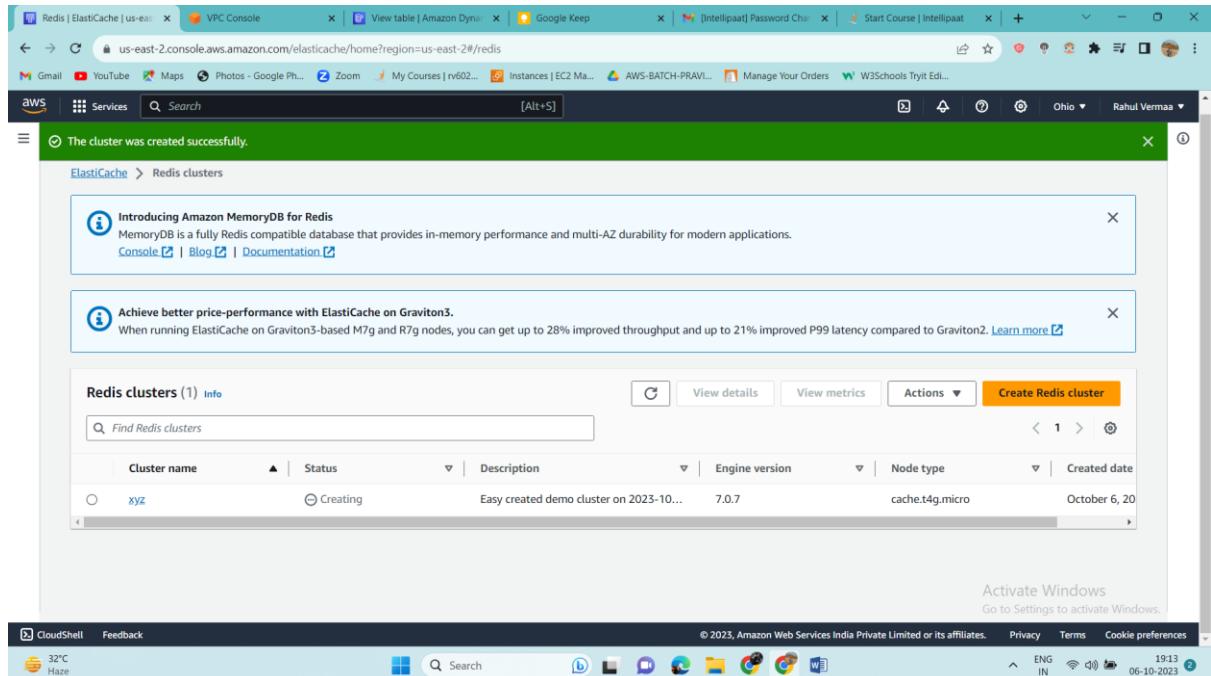
Now click on redis cluster



Select appropriate options and click on create button



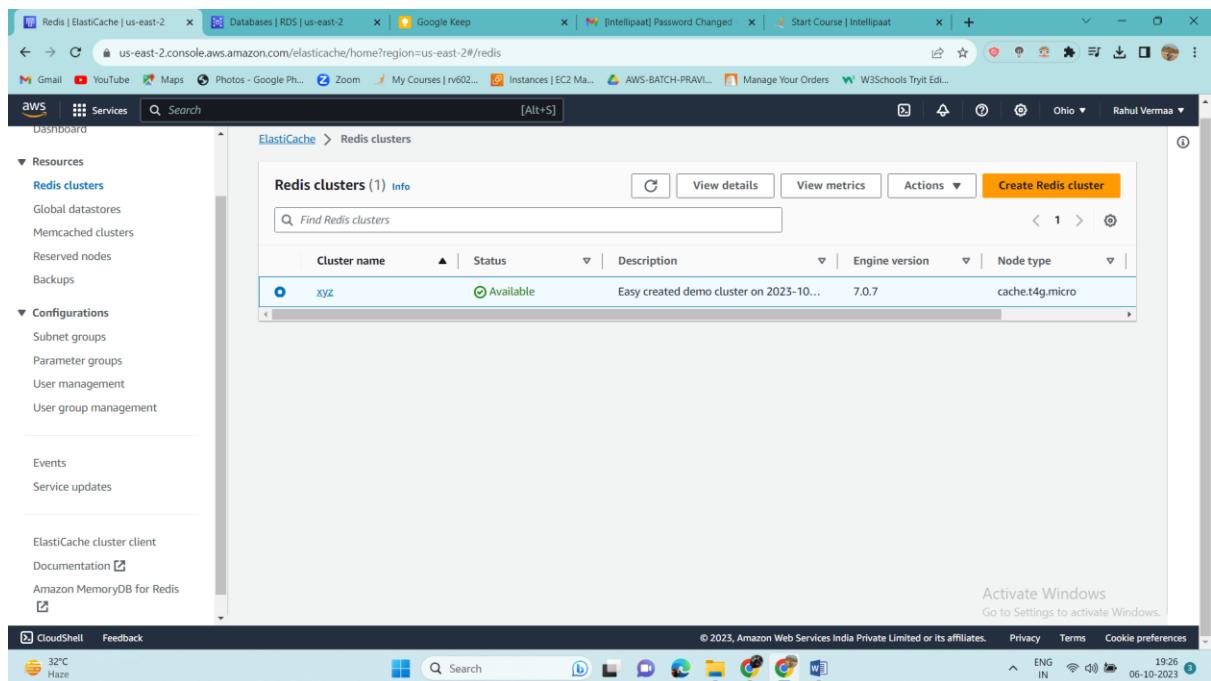
And it's created successfully



The screenshot shows the AWS ElastiCache console for the us-east-2 region. A green success message at the top states: "The cluster was created successfully." Below this, there are two informational banners: one about Amazon MemoryDB for Redis and another about price-performance with ElastiCache on Graviton3. The main table lists a single Redis cluster named "xyz" with the following details:

Cluster name	Status	Description	Engine version	Node type	Created date
xyz	Creating	Easy created demo cluster on 2023-10...	7.0.7	cache.t4g.micro	October 6, 2023

And it's done



The screenshot shows the AWS ElastiCache console for the us-east-2 region. The left sidebar is expanded to show "Resources" and "Configurations". The main table now shows the Redis cluster "xyz" with the status "Available". The cluster details are the same as in the previous screenshot.

Cluster name	Status	Description	Engine version	Node type
xyz	Available	Easy created demo cluster on 2023-10...	7.0.7	cache.t4g.micro