AWS day 3

AWS platform
↓
services → serverless
↓
serverfull

Database server: oracle

Database engine: ~~Any~~ sQL

Database ~~eng~~ schema: RDBMS (Table)

Database server = ec2 (elastic complete ~~cloud~~)
2 is cloud

unorganized sector

organized sector

ec2 : ip : public/private

public IP: domain

Usecase of private IP:

database <> website

* Send file to ec2 (server)
Config the - config (database) file of db server (private
→e

* Aws management console → RDS search ~~करो~~

↓

← create database (sql) ← RDS ~~में~~ click ~~करो~~

ec2 instance →
Database instance →

RDS → ec2 + DBengine
↑
serverfull

AWS maintains DynamodB (serverless)
~~SQL~~ Dynamodb is Nosql

* Client side - Difference

nosql/sql → ec2 → Backend → client URL
nosql/sql → AWS → Backend → Client URL → quality high → auto maintained by AWS

Java
UI Background

* Discussion on data storage

There are 5-6 dedicated service in AWS which are data storage centric. Data storage can be done in 2 ways
A) Storage of data without schema → AWS(S3) (simple storage service)
→ EFS, EBS
B) Storage of data with schema
↕
- ~~RDMS~~ RDBMS (AWS - RDS)
- dynamodb (This is the only database which is maintained & supported by AWS community. This database is present only in AWS platform. ~~No Th~~ No third party involvement

CosmoDB (This is only database supported by Azure)
dynamoDB cannot be used in Azure

mongoDB is a database company that can be present at AWS or Azure platform, unlike DynamoDB & RDBMS.

Note: If you want to use mongoDB on AWS or Azure then there are 2 possible ways → install from server (ec2/vm)
→ get it from marketplace

* Which database architecture will be required for which type of data?

uber/ola cab book example:

Client/end user —assigned a driver→ [ ] Database

Continuous location update
Realtime location update

No SQL architecture doesn't use tables. It just uses json or document. (Hot data) Json = key-value pair
travels by ~~streaming~~

Cold data: SQL : travels by Batch/slots

~~text~~ size1 size1 > size2
~~fid~~
word x size 2 size2 > size1 ✓

* Agenda:

RDS → VPC, security, config → connecting to database
↓
esidb sql query

S3 → bucket → object storage system

S3 vs rds
• google drive vs mysql (db Engine)

* Simple storage device

storage : object

ex: image
folder
file
txt
json
csv

S3 → object or Bucket which stores object

object < > Data

Data → Cold, Hot

the storage for hot data, will be not same for cold data

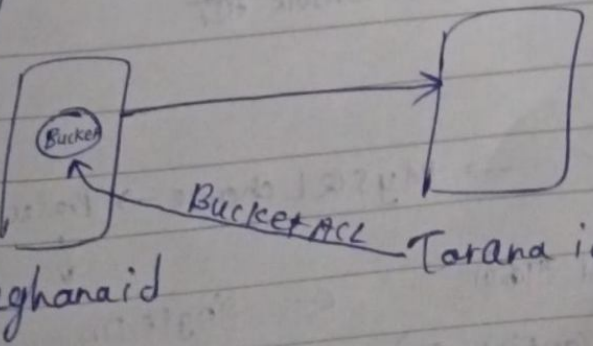
S3 → Glaciers → Archieves → Cold
S3 → Hot Teir → Hot Data

Two types of storage classes of S3

ACL → Access control list

Bucket name are Global in Nature

Bucket Name is Global → other aws account can fetch your bucket details if you allow them



Public : www
Private → IAM id, AWS email id
AWS account → Account id.

Bucket name is unique

Test conducted.

URI is superset URL >> URI

URI bucket with objects

S3 → Buckets (left side click) → create bucket → bucket name → ACLs enabled → block all public

create bucket ← click on bucket ← left side upload

add files → multiple files select → skip permission → ACL → choose from predi

Specify individual ACL

- ☐ objec read
- ☐ object ACL read
- ☐

window ← upload

Configuration

→ Screenshots Name taskbar

Test मेरी task

Day 3 lab 1 : create a database

RDS search करो AWS management console मेरो

create database

Standard create → MySQL choose → Production

single DB instance ← password टाको ← then,

confirm करो ← Meghana@123456

Allocated storage 100 करो

IOPS → 1500

Then after (iops → 1500)

↓

maximum storage threshold → 1000

↓

Public access → yes ✓

Now go to 'additional configuration (bold मराठी)

↓

'Give name → test'

- Then, VPC security group मधे ⊙ create ~~new~~ existing ✓

~~select launch wizard-2~~ ~~EC2~~ मधे ~~security groups~~ ~~17:12 पासून बघणे~~

open ~~new~~ duplicate tab

↓ search security groups

name → megnaa

Description → This is for Windows Env.

Inbound → All ~~TCP~~ Traffic select करणे, Anywhere IvPy → create security group

↓

← edit inbound rules

All traffic
HTTPS
HTTP
SSH
left side ची select करणे

आणि

सगळीकडे
0.0.0.0/0
select करणे
right side ची select करणे

Save rules.

Now go to RDS Management console tab

click on choose existing. double click (created ~~go~~ security group)

☐ ✗ Default ✗ असं येतं

uncheck the enable automated backups

↓

create database

open
download mysql workbench from google
↓
No thanks, just start my download
↓
choose developer default किंवा custom ✓
↓
MySQL servers मध्ये MySQL server → Mysql server run (पहिला)

Applications मध्ये MySQL workbench → पहिला select

- Mysql connectors → connector → पहिला

Documentation → mysql documentation → पहिला

next

execute

next

next

next

MySQL Root password टाकणे

repeat password

next

next

next

execute

हे सगळं personal laptop मध्ये possible आहे

Now go to RDS management console. Click on the created database.

Click on port number & endpoint

let us connect to database created via the workbench

Now open MySQL workbench

↓

click on ⊕ sign

↓

Connection name: MeghnaDB

Hostname : endpoint paste करो (of the database which was created)

port number तोच default.

Username → admin

Password → type the password which was used to create database.

Click on Test Connection

Click on MeghnaDB

Successfully made the MySQL connection असा message येतो.

Click on 'test' & start writing queries.

lab 2 : S3 bucket

search S3. in AWS.

↓

create bucket → name देतो 'mrp' → ACLs enabled ✓

↓

Bucket owner prefred ✓

↓

untick the Block all public access

↓

Bucket versioning → disable

Default encryption → Disable

Advance → disable

Bucket is created
Upload files
permissions मध्ये specify individual ACL permissions
& click grant public-read access

object owner read ☑ read ☑ write ☑

I understand ☑

Uploaded ✓

S3://{Bucket_name}/filename
↑
URI

जर करताना error आली तर,
permissions मध्ये
↳ ACL → Public access is blocked → click on Block Public Access settings for this account
↓
Save changes ← untick ← edit
↳ confirm

Now, duplicate tab → search ~~IAm~~ IAM
↓
Users मध्ये ← click IAM

Username → Meghana
↙ Access key
↙ Password

Custom password [ ]
~~meghana~~ ☑ show password

Create group
↓
Amazon S3 Full Access ✓
Administrator Access ✓
↓
Create group

group name → group

tick group created
Click on nex. tags

create user

download csv

Now put username & password

Day 3 completed