**DynamoDB and RDS**

* SQL with relational database service and NoSQL with dynamoDB.
* DynamoDB and RDS

**Rds Managed task eg:** Software upgrades, nighlty db backups, monitoring  
  
When DB inst is created on RDS it creates a EC2 ins, installing an appropriate OS and DB engine. because of this nned to choose the size of EC2 instances that is powering your DB. Important becoz under powered EC2 inst wont be able to process queries very well and will impact app

EC2 instance type can be changed after inst has been created.

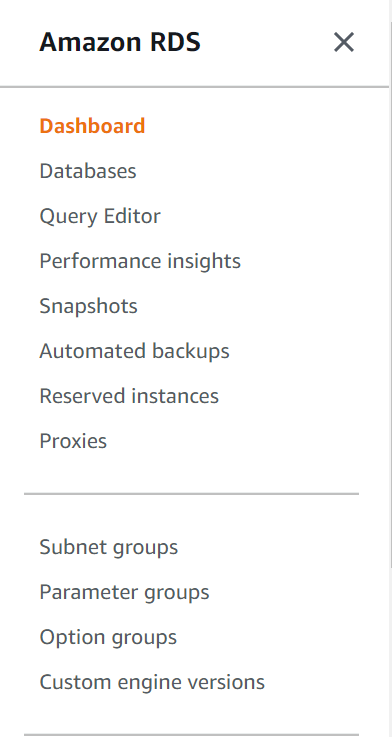
**RDS feature :**

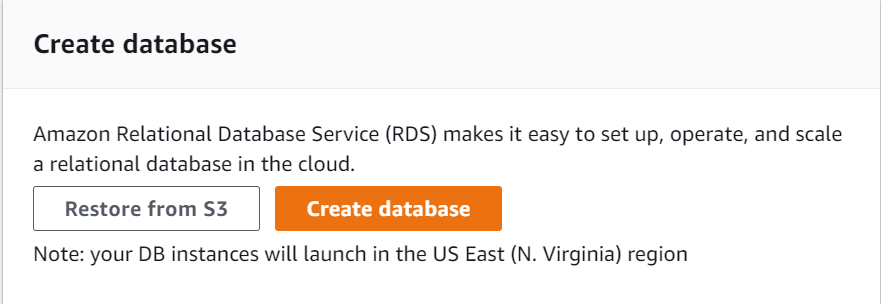
* **Backups :** Occurs daily, config backup windows, backups stores 1-35 days, restore from backup
* **Multi-AZ deploymnet (AZ- Availability zone) -** DB replication to different AZ, automatic failover in case of catastrophic event.
* **DB Read replica-** Non- prod copy of DB, copying of data to this replica is done with eventual consistency with source, useful for running queries on data, will not be use as failover.

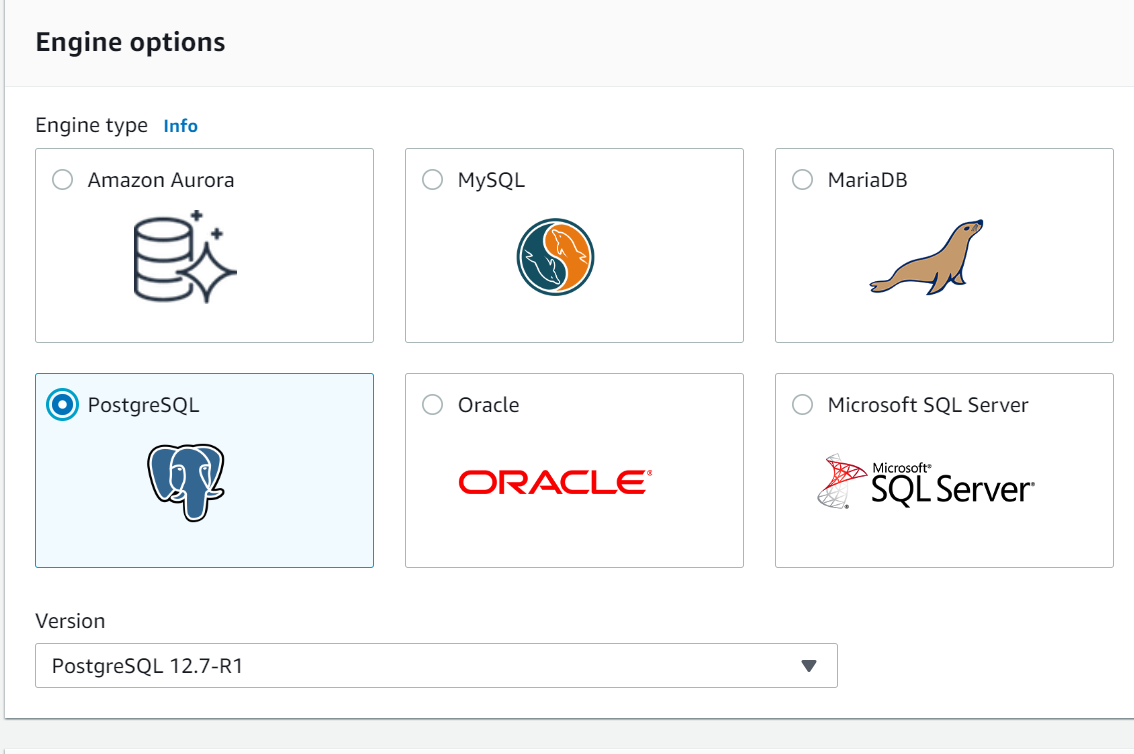
**RDS DB Option :** Amazon aurora, MySQL, PostgreSQL, SQLServer, Oracle DB, MariaDB

**Creating a PostgreSQL DB Ins**

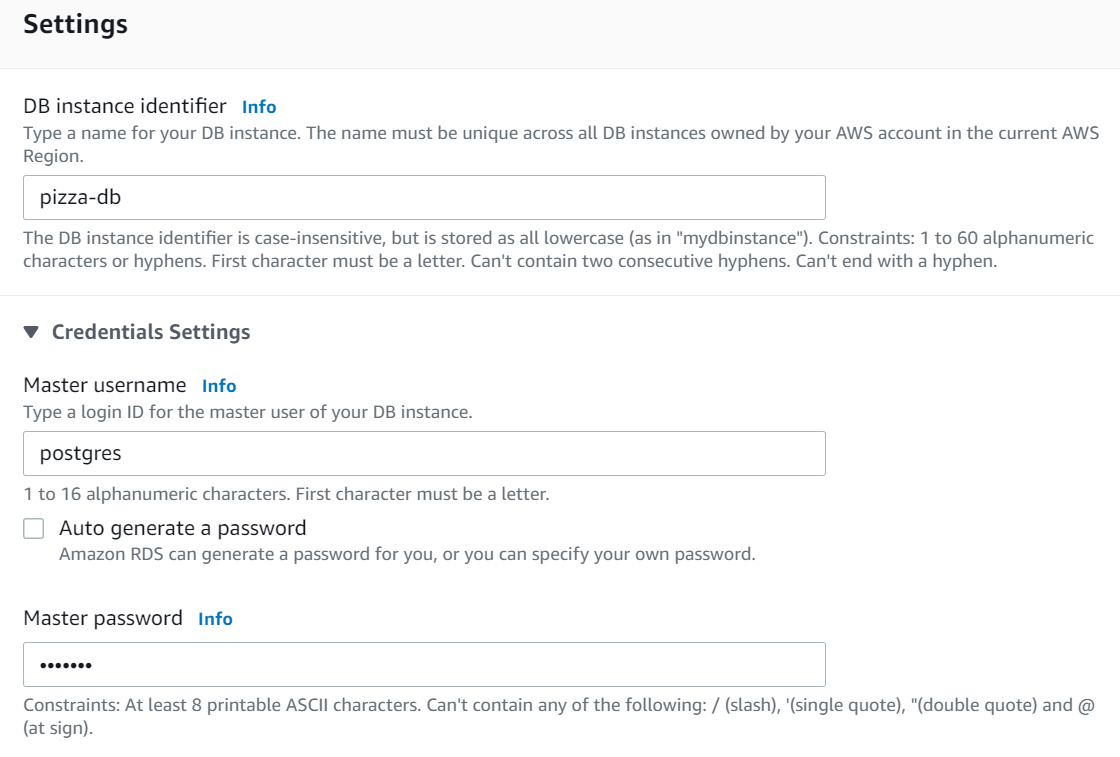
Search - RDS

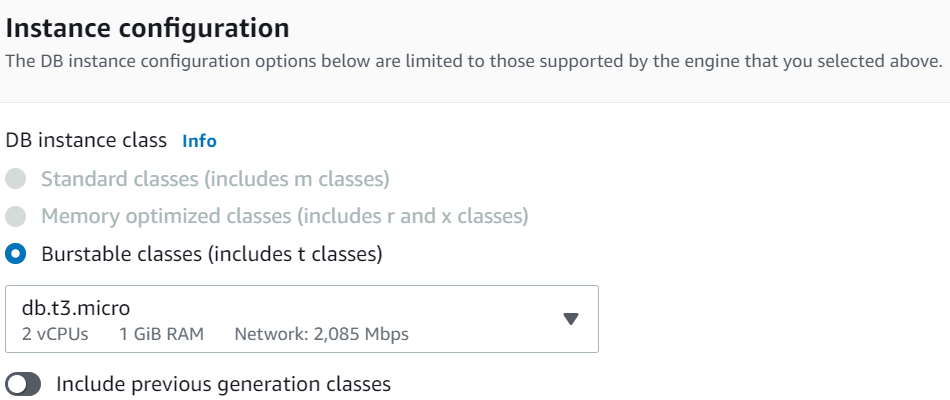


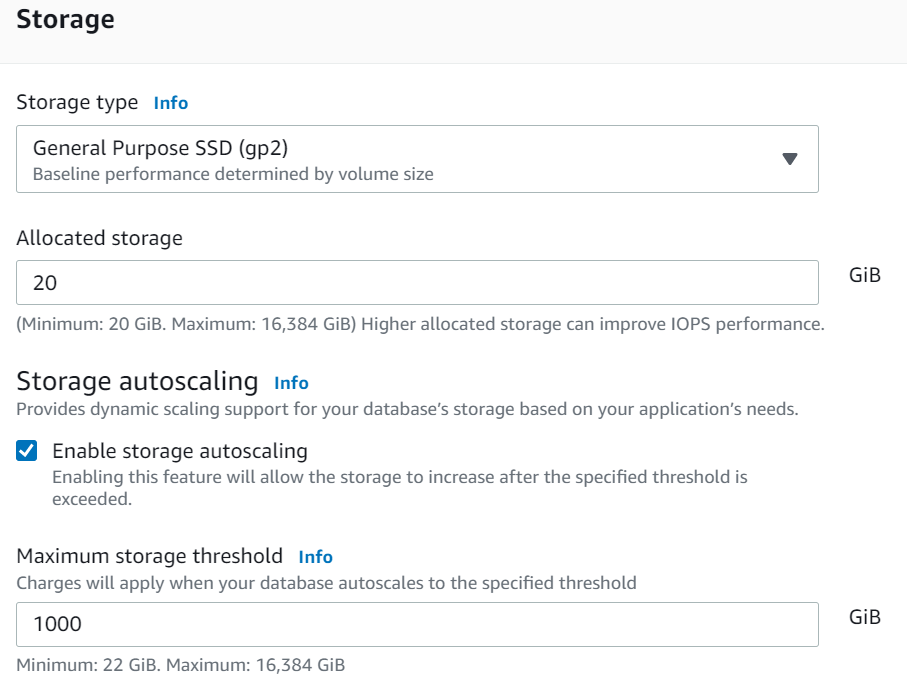


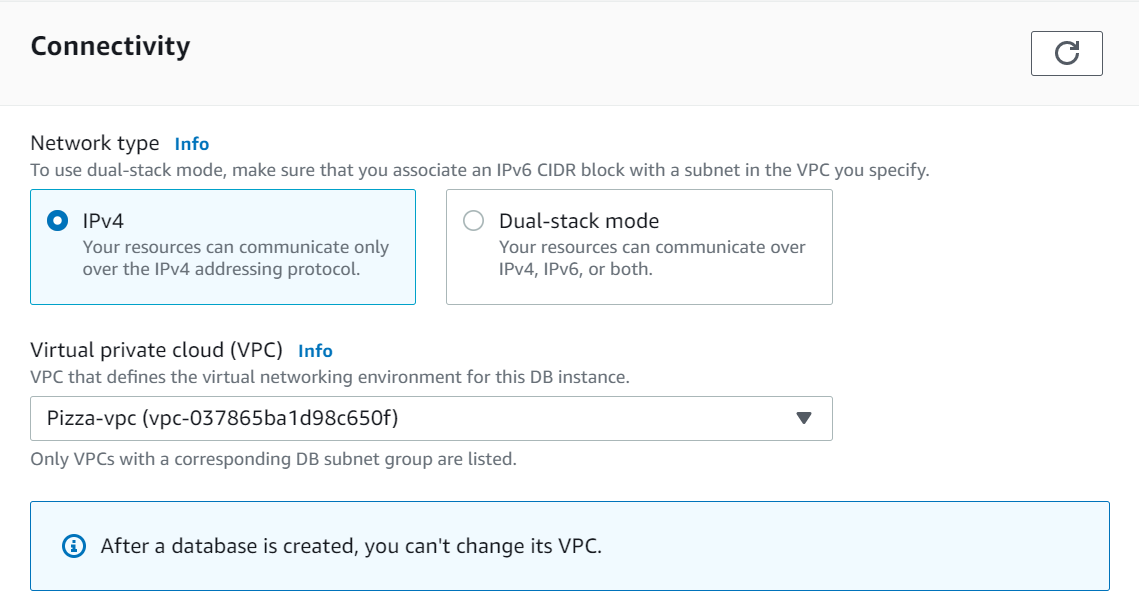


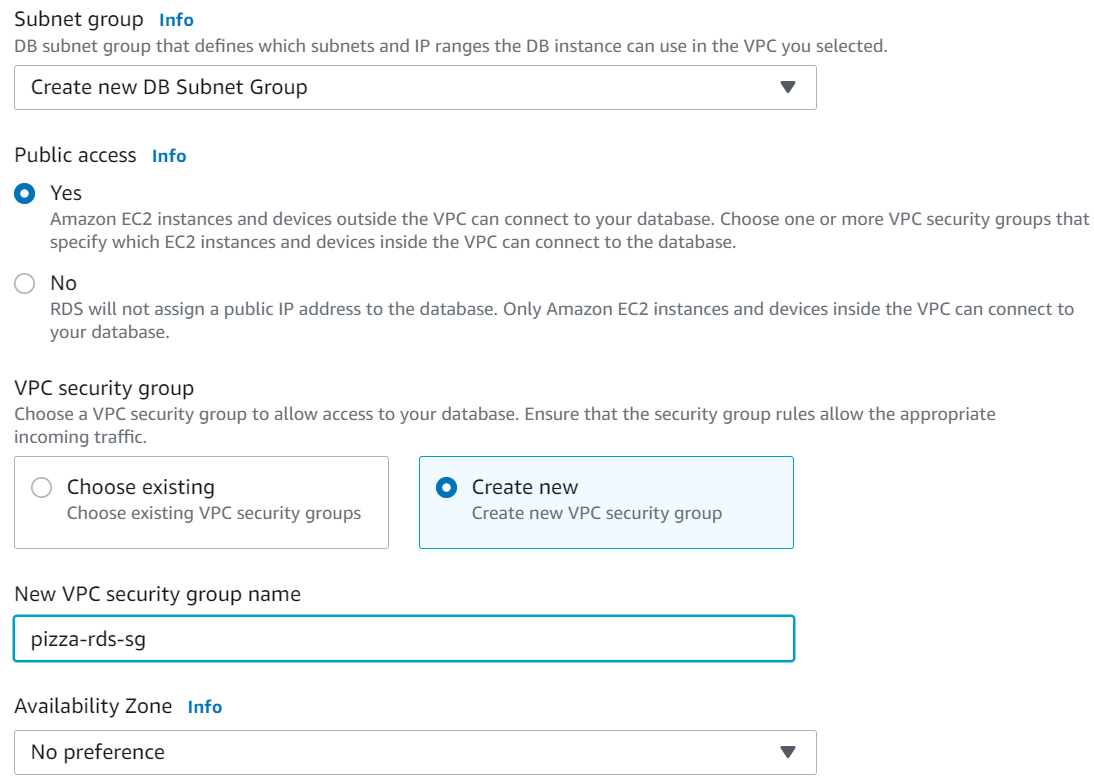


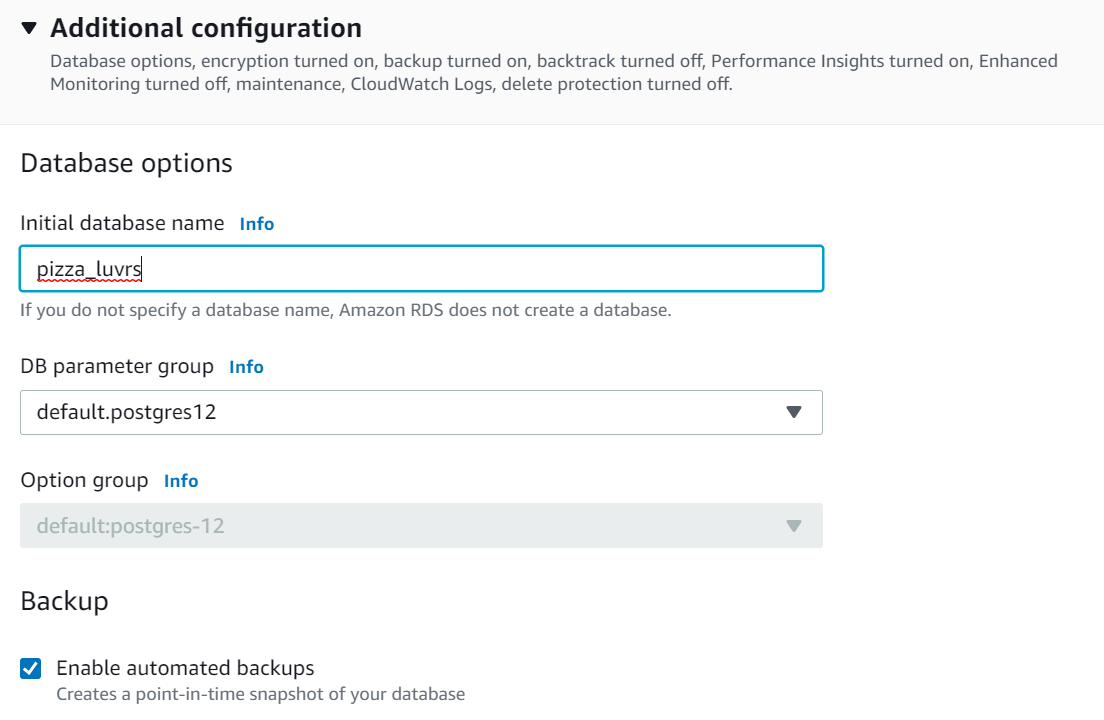












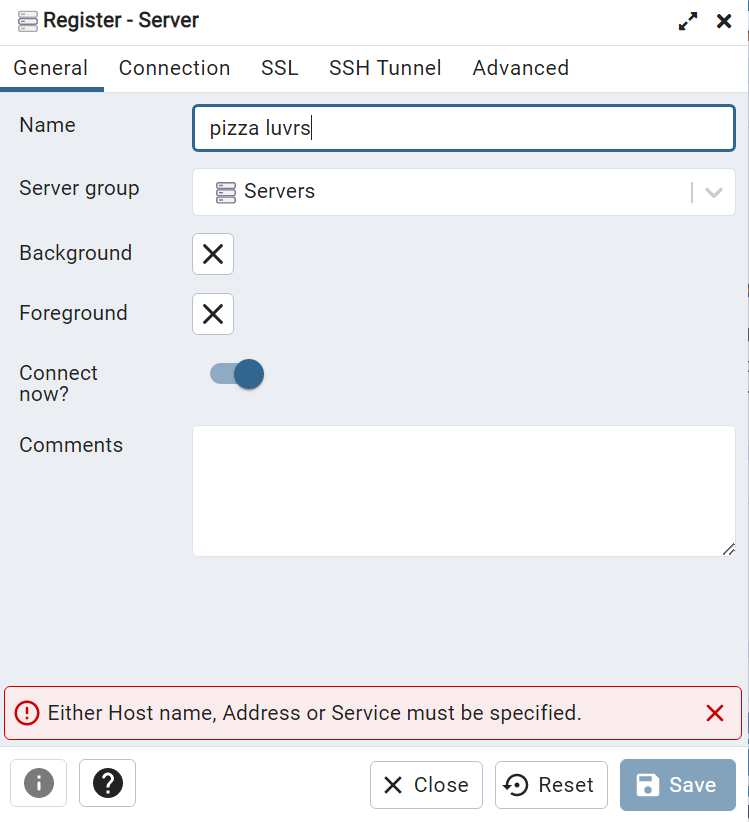
And create DB

**Connect to DB**

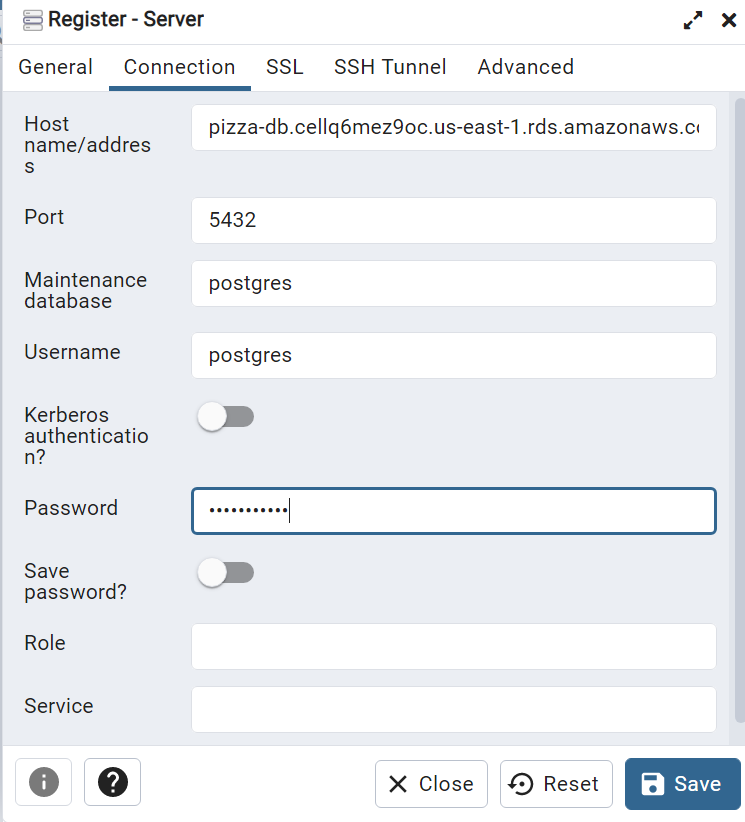
While creating a new DB instance create a new security grp for the Db instance.

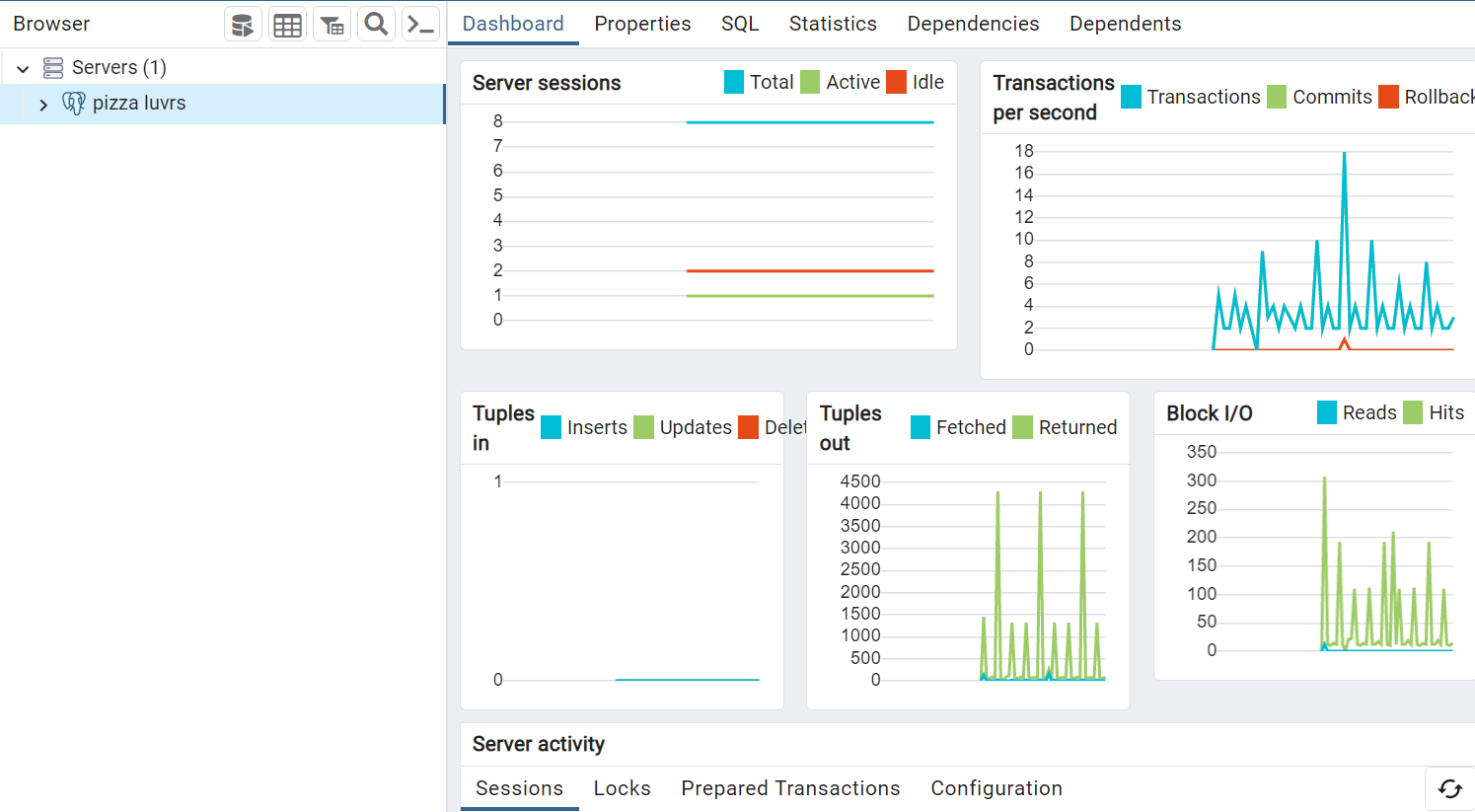
Make sure the RDS instance security group gives access to your ip on port 5432

will use postgres-specific GUI admin tool to connect : pgAdmin , <https://www.pgadmin.org/>

On pgAdmin console 🡪 add server  


Copy endpoint details from AWS console for the respective DB



Once connected will be shown on the left  


**ORM (object relation mapping) -** This transforms SQL queries and responses into basic function call and objects that makes DB is much easier to work with

Why do we need DB : Persistence b/w app restarts, scalability when activity increases

**DynamoDB**

* non- relation, NoSql
* The main structure in DynamoDb is a table, a table contains many rows of data called items .
* The tables need to include the same primary key used to index the data and make it efficient to query items based on the key.
* Primary key option- string, number or binary

Provisioned throughput capacity : R/W operation per sec provisioned for the DynamoDB table and tell aws how much h/w to provision for the table.

* 1 write Unit used for 4kb item
* 2 write Unit used for 8kb item

**Main features of DynamoDb capacity** :

* **On demand capacity :** Table capacity scales as needed, pay per table or index request, more expensive that provisioned capacity mode.
* **Provisioned capacity :** Increase or decrease provisioned capacity based on rules, works like EC2 auto scaling groups, cheaper than on-demand capacity mode.

**Create DynamoDB**On Dynamo DB Console 🡪 create table

