RDS Cost Analysis

RDS high costs causes and recommendations:

The main leaks in the RDS costs are:

- Overprovisioned storage class (io1)
- Overprovisioned IOPS (1500 reserved IOPS but from the metrics it was maxed out about 200 IOPS per second.)
- On-demand instances (reservation can be made to 1 year, 3 year, and can be payed all upfront, partial upfront, or no upfront which is basically monthly pay)

First the easiest change would be to reserve the instances which could bring you about ~20% decrease in price:

1. Change Unreserved instances → Reserved instances:
   Currently 3 db.m5d.large is used. *(1 writer and 2 reader instances)*
   This currently costs approximately $1050 per month!
   With the currently used settings if 1 year reservation would be made:
   Total Upfront cost: 2,680.00 USD per year (approximately $223 per month)
   Total Monthly cost: 600.00 USD *(due to io2 storage type)*
   This would total out to $823 per month.

   Additional recommendation:
   Join Cloudvisor resell program so we can offer you additional savings on your reserved instances and other benefits!
   If interested let us know and sales will get in contact with you!

   ONLY APPLY RESERVED INSTANCES IF YOU DECIDED NOT TO DO THE BELOW STEPS OR AFTER APPLYING THEM!

The second option would be a more complex solution but as the end result combined with the first option you can push down your costs to about $200-400 dollars per month.

2. Storage type and instance type optimizations:
   Currently used RDS infra main properties:
   The currently used **Multi-AZ DB cluster does not support the changing of storage type**, and the io1 only supports the minimum 400 GB storage size.
   Current cost for the current RDS env:
   Approximately $1050 per month.

OPTION 1:

Changing to RDS MultiAZ DB Instace:

You can use the same database setup (1 instance and 2 read replicas) but the change of DB Cluster to DB Instance still would need to be done in order to change storage type and save costs.

In case you decide to use Multi AZ DB instance recommendations are:

- Use GP3 storage with the default 3000 IOPS and 100GB storage.
If it is not sufficient you can use 400GB which will default to 12000 IOPS

Forecasted cost:

DB costs would drop to approximately $450 per month *(instead of $1050)*

- Decrease storage size to 100 gb.

Metrics says 99% of the db storage is free. (not utilized)

Forecasted cost:

Approximately it would give you **minus** ~$50 per month.

So it would total out approximately $400 dollars per month.

If you are interested --> Read more here ← on how to select the best storage type for your database!

Note:
Reserving these new instances would lead to even lower costs!

Generic implementation considerations and steps

Changing to Multi AZ DB Instance would require DB migration.

To re-create the db into a new Multi-AZ DB instance:

The necessary steps would be to:

1. Create a new Multi-AZ DB instance (not cluster)
2. When db is created add the necessary read replicas.
3. Do a mysql dump & restore with some local mysql client where the source is the currently existing prod db (zaaprodclone) and the target database would be the one created in the previous steps.
4. Change the DB endpoints in your application to point to the new one.

Note:
Some downtime can be expected with the above solution!
Keep in mind that during MySQL Dump & Restore it is recommended to stop your application in order to the old \ new db to be consistent.

OPTION 2:
I would recommend **migrating the DB to Aurora Mysql** with the following settings:
- Aurora MySQL compatible
- Engine -> Aurora MySQL 3.04.0
- Storage -> Aurora standard (not the IOPS one as based on the metrics it is not needed).
- For the instance type I can recommend the **db.t3.large** *(approximately $294 per month)* which is a burstable instance. Or if burstable instance is not a preference, then use the **db.r6g.large** *(approximately $435 per month)*

Note:
This calculation was calculated with 2 nodes, where there is 1 regional cluster and 2 reader instances. All of them are supporting high availability, uses Multi-AZ in 2 Zones.
If you want to play around more with the calculations use:
https://calculator.aws/#/addService/AuroraMySQL

Note 2:
Reserving these instances for 1 year, all upfront option would result in:
db.t3.large *(approximately $208 per month)*
db.r6g.large *(approximately $263 per month)*
Also note that with cloudvisor resell you can get it even a bit lower!

To **migrate the DB to Aurora Mysql the recommended steps is the use of AWS DMS Service.**
https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/migrate-an-on-premises-mysql-database-to-aurora-mysql.html

https://aws.amazon.com/getting-started/hands-on/migrate-rdsmysql-to-auroramysql/