Scalability Vs Elasticity	▪ **Elasticity** - Scale with Demand (Short Term). ▪ **Scalability -** Scale out Infrastructure (Long Term). **EC2** ▪ **Scalability** - Increase instance size as required, using reserved instances. ▪ **Elasticity** - Increase the number of EC2 instances, based on Autoscaling. **DynamoDB** ▪ **Scalability** - Unlimited amount of storage. ▪ **Elasticity** - Increase additional IOPS for additional spikes in traffic. Decrease that IOPS after the spikes. **RDS** ▪ **Scalability** - Increase instance size. ▪ **Elasticity** - Not very elastic, cant scale RDS based on demand. **Aurora** ▪ **Scalability** - Modify the instance type. ▪ **Elasticity** - Aurora Server less.
RDS Multi-AZ Failover	Multi AZ (Availability Zone) keeps a copy of your production database in a separate Availability Zone in case of a failure or disaster. AWS manage the failure from one AZ to another automatically. **Multi-AZ RDS** Multi-AZ is for Disaster Recovery Only. It is not primarily used for improving performance. For performance improvement, you need Read Replicas. Multi-AZ allows you to have an exact copy of your production data base in another AZ. AWS handle the replication for you, so when your production database is written to, this write will automatically be synchronized to the stand by database. In the event of planned database maintenance, DB instance failure, or an AZ failure, Amazon RDS will automatically failover to the standby so that database operations can resume quickly without administrative intervention. **RDS Multi-AZ Failover Advantages** High Availability Backups & Restores are taken from the secondary which avoids I/O suspension to the primary. **Exam Tips** ▪ RDS Multi-AZ Failover is not a **Scaling Solution**. ▪ Amazon handles the failover for you. Done by updating the private DNS for the database endpoint.

	▪ Backups & Restores are taken from the secondary Multi-AZ instances. ▪ Read Replicas are used to scale. ▪ You can force a failover from one AZ to another by rebooting your instance. This can be done through the AWS management console or by using RebootDBInstance API call.
Read Replicas	Read Replicas make it easy to take advantage of supported engine's built in replication functionality to elastically scale out beyond the capacity constraints of a single DB Instance for read-heavy database workloads. *** Read only copies of your database**. *** Replica of your production database is read only** *** Once Read Replica is created, database updates on the source DB Instances will be replicated using a support engine's native, asynchronous replication. You can create multiple Read Replicas for a given source DB Instance and distribute your application's read traffic amongst them.** **Exam Tips** • You can have up to 5 read replicas for MySQL, PostgreSQL & Maria DB. • You can have read replicas in different REGIONS for all engines. • Replication is Asynchronous only, not synchronous. • Read Replicas can be built off Multi-AZ's databases. • Read Replica's themselves can now be Multi-AZ. • You can have Read Replica's of Read Replica's beware of latency. • DB Snapshots and Automated backups cannot be taken of read replicas. • Key Metric to look for is REPLICA LAG.
Aurora	**Amazon Aurora provides up to five times better performance than MySQL (3 times better performance than PostgreSQL).** **Aurora comes in 2 flavours** ▪ Aurora ▪ Aurora Serverless **Redundancy** 2 copies of your data is contained in 3 separate availability zones with a total of 6 copies. **Storage is Self Healing** Data blocks and disks are continuously scanned for errors and repaired automatically. **Aurora at 100% CPU utilization?** Is it writes causing the issue? If so Scale Up (increase instance size) Is it reads causing the issue? If so Scale Out (increase the number of read replicas)