



Global Knowledge®

Expert Reference Series of White Papers

Introducing Amazon RDS for Aurora

Introducing Amazon RDS for Aurora

Chris Littlefield, AWS Certified Solutions Architect – Associate Level, AWS Certified SysOps Administrator – Associate Level

Introduction

The following document provides an overview of one of the latest offerings from cloud leader Amazon Web Services (AWS). It's a product that will enable powerful, massively scalable relational databases in Amazon's cloud environment. The product was designed, built, and tested in secret over the past three years, and now it is nearly ready for production workloads. Some of its detailed design remains confidential, so we'll present the released features and functionality, as well as pricing for this exciting new product.

Background

In November 2014, at Amazon Web Services' annual conference re:Invent, Andy Jassy introduced a series of revolutionary new products. Among those new products is the product described in this paper, which is based on AWS's hugely successful Relational Database Service (RDS). So what's this new product, you ask? It's Amazon Aurora for RDS. Aurora is MySQL compatible database solution that will enable highly scalable databases running across multiple AWS availability zones, at a very low price point. It's designed to deliver the performance and availability of commercial grade databases at the simplicity and price point of open source databases.

What is Amazon Aurora?

Aurora is a MySQL compatible database engine that far exceeds the current size limitations of relational databases running on RDS. It spans three AWS availability zones within a region. The great news is that if you already use or are familiar with RDS, you're already going to be comfortable configuring Aurora. Aurora utilizes many of the same configuration options as other flavors of RDS. You simply have a new dimension to the implementation: greater scale and performance. Aurora will be a game changer in both the database and cloud markets.

An Introduction to Amazon RDS

For those readers who are unfamiliar with RDS, this section will give you a high level view of its features and functionality. (If you're familiar with RDS, please skip to the "Getting Started with Aurora" section.)

RDS provides a managed environment for customers to store their databases. At present there are four supported database options:

- Microsoft SQL Server
- Postgres SQL
- Oracle
- MySQL

The purpose of RDS is to provide a managed service for relational database use cases, and thus allow companies to offload the "heavy-lifting" of database management. In other words, AWS will manage the database admin tasks, while customers focus on their data.

RDS works similarly to Amazon Elastic Compute Cloud EC2. The customer chooses a virtual machine, or “instance” type for their database. Based on the use case, customers can choose instance types based on various levels of CPU, memory, and storage options. The customer can scale the instance type up or down as needed. Pricing is based on the instance type chosen and billed hourly. Aurora will be billed the similarly.

The database patching and backups are carried out by AWS on behalf of the customer. The backup retention period is customer configurable for backups (up to thirty-five days). In terms of point-in-time recovery, customers can choose to recover data up to five minutes in the past. In terms of patching, customers can choose whether to allow AWS to patch the database and when this should be done. The customer can then set this “maintenance window,” and it’s configurable via the management console and the API.

Although you can configure a single availability zone setup for RDS, AWS provides a multiple availability zone (multi-AZ) RDS option for high availability and disaster recovery purposes. In the multi-AZ configuration, RDS creates a primary and secondary database instance in two separate availability zones. A primary database instance is placed in one availability zone, and a secondary is placed in a second availability zone. In a third availability zone, a witness instance runs to observe the health of the primary instance. If that instance becomes unhealthy, the witness will cut over to the primary, switch the primary RDS endpoint address (or CNAME DNS record thereof) to the secondary instance. This means that there is very little downtime for the application using the database and there is no action required on the part of the customer to get the database back up and running.

In terms of security, customers should consider deploying RDS instances in private subnets within the Virtual Private Cloud. In terms of access control, security groups act like firewalls that allow for fine-grained control of access to the instance. Security groups utilize allow rules by port number to restrict access to only those users or processes requiring access. RDS endpoints should also be secured via TLS/SSL if they are addressable from the Internet. It is also possible to encrypt your data at rest for all the supported database platforms. Aurora will support encryption at rest once it’s fully launched.

Getting Started with Aurora

If you are already familiar with RDS, you are ahead of the game. Aurora can be configured very similarly to the current RDS offerings. The customer chooses and launches Aurora database instances, and then secures them with security groups. There are parameter groups as before, and you can easily modify the parameters as dictated by your particular use case.

MySQL Compatibility

Currently, the version supported by Aurora is MySQL 5.6 and the INNODB storage engine. Its MySQL compatibility means that your existing MySQL applications will run on Aurora without you having to change code. You’ll get the familiar backup, maintenance, and patching as well as the ability to choose a retention period of up to thirty-five days for point-in-time recovery. Aurora does not support the MyISAM database engine at this time. The great news is that your existing MySQL applications can easily be migrated to AWS with little to no changes to your code. Aurora also offers a push-button migration tool, which will convert your existing RDS MySQL DB Snapshots into running Aurora instances.

Under the Hood

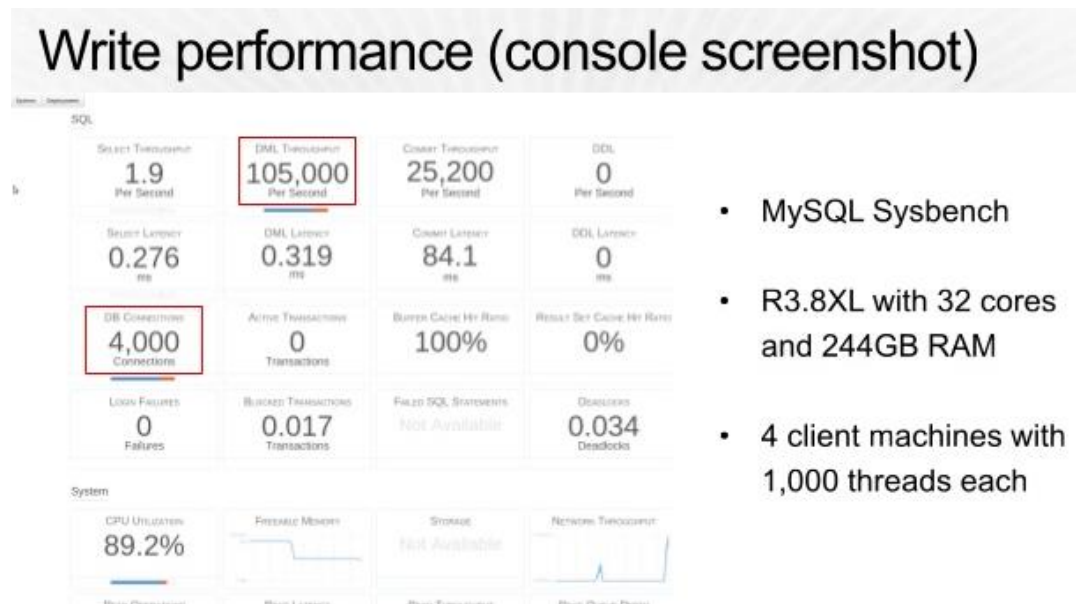
Databases have some standard functions: SQL, Transactions, Caching, and Logging. Although there are some alternative architectures, traditional database components tend to be stored entirely on the same server. If you need more performance you get a more powerful server. The aim of the Aurora team is to split up the functions of the database in to horizontally scalable tiers for increased performance and reliability.

Aurora's design incorporates a log-structured storage model. The nodes in Aurora are SSD backed instances that are autoscaled. You don't have to manage provisioning your storage, or your IO for your databases. The instance types allow for 10 Gb/sec network bandwidth, and AWS offer hotspot management for Aurora customers. Read and write quorums are utilized to get four out of six for writes, and three out of six reads which reduces latency volatility.

Aurora makes use of other AWS services as part of its architecture: Elastic Compute Cloud (EC2) for the nodes and S3 for highly durable backup storage. Aurora also integrates with other AWS services, some of which will discuss later in this paper.

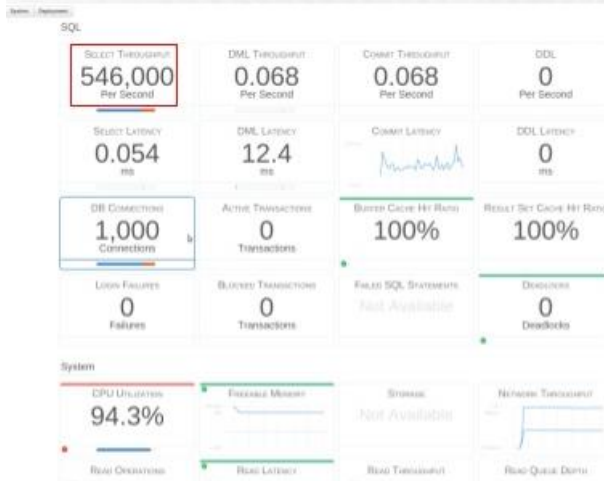
Performance

The performance of Aurora looks to be five times that of any other MySQL implementation. There have been tests showing millions of transactions per minute, and Aurora handles them with ease. It's unclear if Aurora will be released for other vendor platforms, but for now it promises to rapidly increase MySQL application performance whilst dramatically reducing costs for customers.



Testing by AWS was performed using Sysbench. The results showed that Aurora can handle 500,000 selects per second and 100,000 updates per second.

Read performance (console screenshot)



- MySQL Sysbench
- R3.8XL with 32 cores and 244GB RAM
- Single client with 1,000 threads

Scale

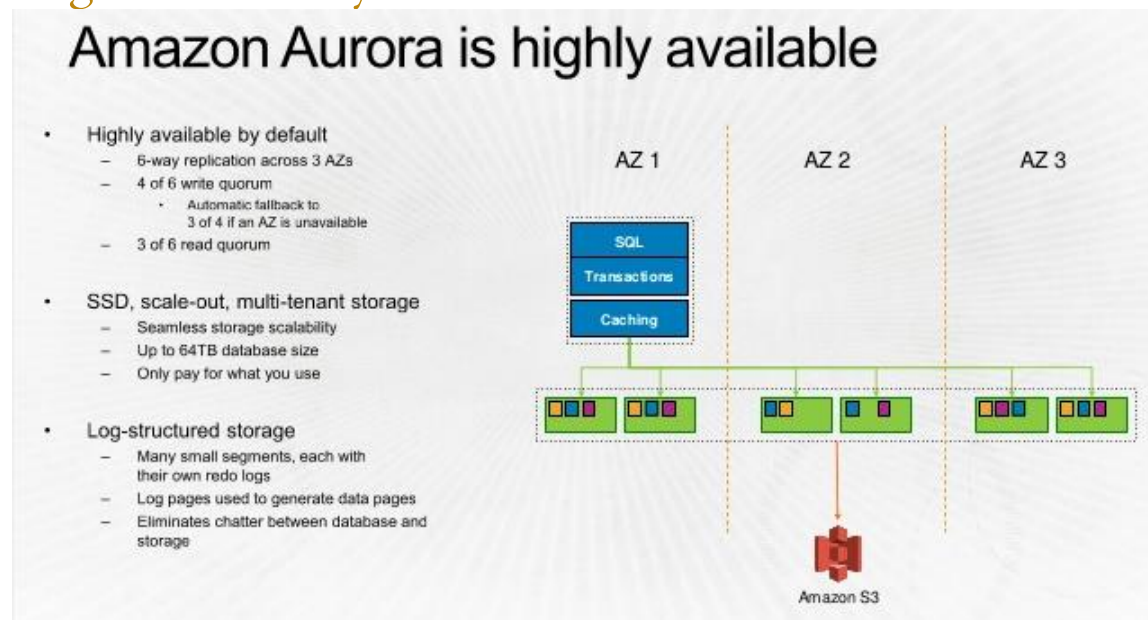
The minimum size for a database in Aurora is 10 GB. Up until the advent of Aurora the maximum RDS database size possible was 3 TB. Unlike the other RDS offerings, Aurora will scale beyond 3 TB. It will automatically grow to up to 64 TB in 10 GB increments. As a result, you only pay for what you use and no longer have to pay for overprovisioned storage.

As before, the scaling of the RDS instance is achieved by changing the underlying instance type. However, Aurora instance types are much greater in size than the biggest existing RDS instances. The 64 TB size and superior processing and memory settings on the big Aurora instances will allow for enterprise scale workloads running at AWS.

Read Scaling

With Aurora you can have your database scale automatically via autoscaling up to the aforementioned 64 TB without doing a thing on your end. Aurora allows you to scale your database(s) up to 15 replicas (or read-slaves). The replicas enable excellent read performance and thus taking your read scaling capabilities to a much higher within Aurora, as compared with standard MySQL within RDS.

High Availability



In Amazon Aurora data is broken into 10 GB chunks, and replicated six ways across three availability zones. The data is stored across three availability zones within a single AWS region. What this means in practice is that you can maintain read availability of your data even if three read copies are lost. In terms of writes, two copies of the data could be lost without affecting the write availability of your database. The data is also backed up to S3 to ensure it can be recovered in the event of a catastrophic failure. As part of the service, AWS are constantly monitoring the health of your data within your Aurora databases. When drives or storage nodes fail, they automatically repair the data making the storage self-healing and fault tolerant.

Database Replicas

Amazon Aurora provides two types of replicas: shared replicas where the data is stored on the same volume as your database data and MySQL read replicas. These replicas can be referenced by your application for read heavy workloads. These replicas are generally within tens of milliseconds behind the primary copies of your data. You can have up to fifteen read replicas per Aurora instance spread across 3 AZs.

Fail-Over

When you choose the multi-AZ option for Aurora failover occurs within minutes. The CNAME of the primary is automatically mapped to the secondary copy of your data. The secondary copy of your database is automatically promoted to the primary in the event of a failure. No customer intervention is required to allow this to take place. Any of your Aurora Replicas (not MySQL replicas) are potential failover targets. Failovers take place without any data loss.

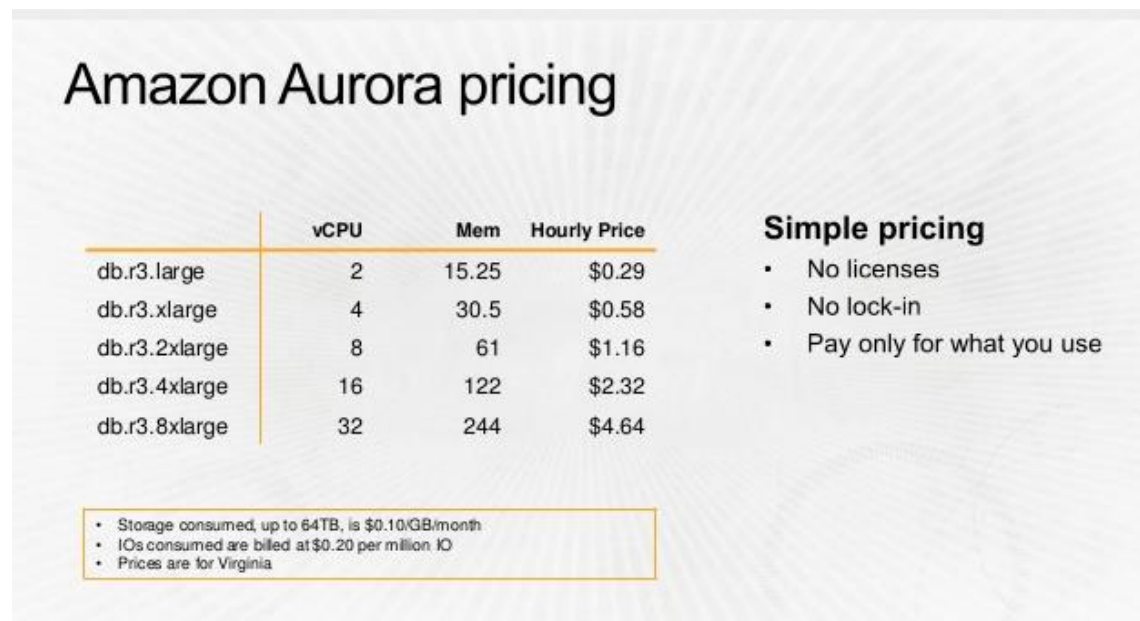
VPC Requirement

Aurora instances must be launched within the AWS Virtual Private Cloud (VPC). This is mandated to ensure security of customer data. Utilizing the VPC gives the customer control of network access to the Aurora instances as well as other server instances that they choose to deploy within their AWS VPC.

Pricing

The pricing for Aurora will be very granular. Just like other services at AWS the model will give customers fine-grained control over the cost of their database storage. Customers will be charged for the instance hour, plus storage. The storage will be in 10 GB chunks, and customers will only pay for what they use.

During the preview, Aurora is free of charge. Once fully released AWS will charge based on the server, or instance type chosen. Ranging from twenty-nine cents per hour up to 4.64 per instance hour. Allowing for a range from 2 vCPUs and 15.25 GiB of RAM up to 32 vCPUs and 244 GiB of RAM. At the time of this writing Aurora does not provide micro instances, and therefore will not have a free-usage tier option available. You can also choose to purchase reserved instances, which can lower prices by up to 64%.



The slide features a title 'Amazon Aurora pricing' at the top left. Below it is a table with four columns: instance type, vCPU, Mem, and Hourly Price. To the right of the table is a section titled 'Simple pricing' with three bullet points. At the bottom left, there is a box containing three additional pricing details.

	vCPU	Mem	Hourly Price
db.r3.large	2	15.25	\$0.29
db.r3.xlarge	4	30.5	\$0.58
db.r3.2xlarge	8	61	\$1.16
db.r3.4xlarge	16	122	\$2.32
db.r3.8xlarge	32	244	\$4.64

Simple pricing

- No licenses
- No lock-in
- Pay only for what you use

• Storage consumed, up to 64TB, is \$0.10/GB/month
• IOs consumed are billed at \$0.20 per million IO
• Prices are for Virginia

Reliability

RDS already provided multi-AZ redundancy options, but Aurora is raising the bar in terms of reliability. Your data is automatically spread across three AZs and you can have up to 15 failover targets (replicas) across those 3 AZs. This means an entire AZ can go down without affect their database availability.

In addition, database crash recovery is virtually instantaneous and the database page cache is fault tolerant allowing it to live on after database crashes.

Security

Security Groups can be configured to lock down access to your Aurora databases. This means that you may want to restrict connections by port number and/or IP address, or IP address blocks (CIDR ranges). In addition, Aurora supports advanced SSL to your endpoint address. In preview mode, Aurora does not support encryption of data in the database. However, stay tuned as some form of encryption of data at rest is planned for the full release.

Migration

Migrating data to Aurora is straightforward. Customers can use the MySQL dump function to get data out of existing MySQL databases, and use MySQL import to upload the data into to Aurora. You can use existing tools like MySQL Workbench with RDS for Aurora. Migration will generally occur in less than one hour depending on the database size. There is also a push button migration for RDS MySQL customers via DB Snapshots.

Backups

Aurora supports database snapshots. This means that your data can be backed up to Amazon's Simple Storage Service (S3). S3 is natively online and provides 99.99999999% of durability of your data. Although AWS already backup your data in RDS, snapshots give you direct control of your data, and taking snapshots will not affect database performance. Snapshots can also be used when terminating your Aurora instances, and remain in S3 for future use. Backups are incremental and continuous and have no performance impact on the data since the work is done by the segregated storage tier.

Access Control

Aurora integrates with AWS Identity and Access Management (IAM). IAM integration allows administrators to fine-grain control user and group access at the resource level in Aurora. In addition, it supports the use of tags in IAM policies to restrict access to resources. So if a resource has been tagged "Dev," "Test," "Prod," for example, you can specify those tags in allow or deny rules in the policy in IAM. Again, if you're already comfortable with Amazon RDS, this process will remain the same, you simply need to specify the Amazon Resource Name (ARN) for your Aurora instances.

Monitoring

Amazon Aurora integrates with Amazon Cloudwatch to enable monitoring of your databases. Customers can configure metrics that will fire alarms when triggered and alert operations teams of any issues within Aurora. The "everything is an API" philosophy will also mean that your existing monitoring tools can pull data from Aurora via api calls in scripts.

Notifications

Aurora also integrates with Amazon's Simple Notification Service. This allows the automation of events to other individuals, or systems. For example an SNS message could be used to place a message in a queue using the Simple Queue Service and fire off another part of a workflow. Alternatively, you could have the notification sent to a user or group of users via text message, email or mobile push notifications.

The Future

Amazon has developed Aurora to support MySQL based on the vast amount of existing customers that use MySQL within their AWS environments. Amazon is open to future expansion of the service into a multi-region architecture. They will also consider other database platforms for Aurora based on demand and feasibility.

More Information

- To sign-up for the preview and for more information please check out:
<http://aws.amazon.com/aurora>
- AWS General Manager, Anurag Gupta presents Aurora at re:Invent 2014:
<https://www.youtube.com/watch?v=HAITkZr9pUk>

Conclusion

Amazon Aurora is a game changer. The enhanced performance and reliability of Aurora will help AWS customers reduce performance bottlenecks in their applications, and blow through the previous limits of database scaling within RDS. The relatively low cost of Aurora will tempt many customers to migrate workloads to this implementation of RDS.

Aurora is currently available under a “Preview” release, and you can reach out to AWS for free access during the preview period. There is no cost for Aurora in the Preview phase.

Learn More

Learn more about how you can improve productivity, enhance efficiency, and sharpen your competitive edge through training.

[AWS Essentials](#)

[Architecting on AWS](#)

[Architecting on AWS - Advanced Concepts](#)

[Systems Operations on AWS](#)

[Advanced Operations on AWS](#)

[Developing on AWS](#)

[Big Data on AWS](#)

Visit www.globalknowledge.com or call **1-800-COURSES (1-800-268-7737)** to speak with a Global Knowledge training advisor.

About the Author

Chris Littlefield has been training and consulting in the IT space for the past 18 years. His career has spanned many IT disciplines. Starting as a network engineer, and Microsoft Certified Trainer, Chris consulted and taught throughout the US, and Australia. He’s completed large scale financial IT projects in the US, Australia, India and China. His background includes projects in infrastructure, business intelligence and cloud architecture. Chris began working with the AWS platform as an architect and developer in 2010. He’s been training Amazon Web Services courses for Global Knowledge for the past year.