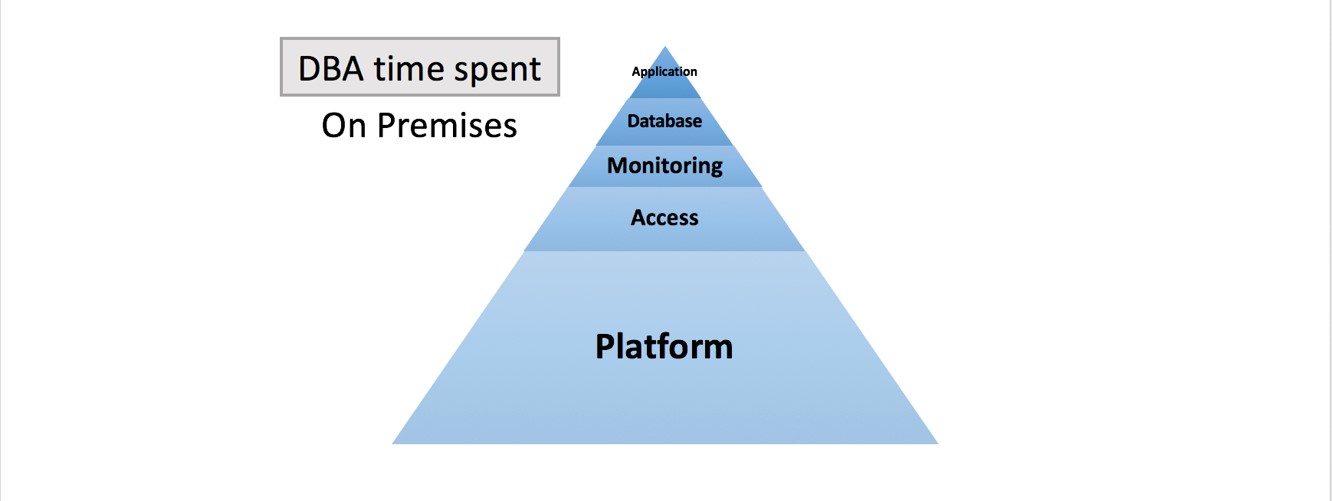
Part 1 – Role of the DBA When Moving to Amazon RDS: Responsibilities

Database administrators (DBAs) are under tremendous pressure every day to deliver value to the business across a variety of fronts. In general, a business’s goals for using the data it gathers are to better understand the business, reduce costs, increase revenue, and deliver improvement and results. If you’re spending most of your time each day installing software, preparing systems, or any number of redundant tasks, that leaves less time to work toward actual business goals.



The more time you spend in moving the business forward, the more likely that you will become a recognizable force for progress.

This blog post is the first in a two-part series. In this first post, we discuss how moving to [Amazon Relational Database Service (Amazon RDS)](https://aws.amazon.com/rds/) can change your role as a traditional DBA and bring more value to you, the business, key projects, and end users. In the next post, we will discuss how to use other AWS products to automate any remaining regular tasks in Amazon RDS.

Traditional DBA role  
According to [Wikipedia](https://en.wikipedia.org/wiki/Database_administrator), the role of the traditional, on-premises DBA typically includes many of the responsibilities that are listed in the following table. For the purposes of this post, we have grouped those duties into five management categories: Application, Access, Database, Monitoring, and Platform.

|  |  |  |
| --- | --- | --- |
| *Category* | DBA responsibility | Traditional DBA role |
| *Application* | Designing schema, access patterns, locking strategy, SQL development, and tuning. | ✓ |
| Modifying the database structure | ✓ |
| Optimizing application and end-user queries (reactive tuning) | ✓ |
| Archiving data | ✓ |
| Generating needed ad hoc reports by querying from the database | ✓ |
| Proactive performance tuning | ✓ |
| *Access* | Enrolling users and maintaining system security | ✓ |
| Controlling user access to the database | ✓ |
| Locking down host access | ✓ |
| Securing database privileged credentials  (SYSDBA or SYSTEM for Oracle; sa for SQL Server) | ✓ |
| *Database* | Parameter configuration and tuning | ✓ |
| Cache management | ✓ |
| Job scheduling | ✓ |
| *Monitoring* | Monitoring performance metrics, response times, and request rates | ✓ |
| Alerting | ✓ |
| Object access | ✓ |
| Logs | ✓ |
| *Platform* | Ensuring compliance with database vendor license agreement | ✓ |
| Designing and implementing disaster recovery (DR) solutions | ✓ |
| Allocating system storage and planning future storage requirements for the database system | ✓ |
| Installing and upgrading the database software | ✓ |
| Performing data backups | ✓ |
| Creating, managing, and monitoring high-availability (HA) systems | ✓ |
| Patching the software that powers your database | ✓ |
| Troubleshooting DB errors and potentially contacting vendors for technical support | ✓ |

As a DBA, you probably do capacity planning, create databases, and install software. You might also spend a great deal of time understanding the growth patterns of your data to plan for future storage requirements. You probably evaluate and apply patches, upgrade point release database software, and perform backups. And you go to great lengths to set up and manage HA systems that can meet steep Recovery Point Objective (RPO) and Recovery Time Objective (RTO) SLAs. The work of database creation, configuration, backups, patching, upgrades, and DR can be time consuming and repetitive. And some of this work is performed during off hours to prevent interference with running production systems.

You are also probably being asked to control access to the database, help application teams draft and apply changes to database structures, and perform reactive and proactive performance tuning. The more time you spend performing proactive tuning and application improvement generally makes the business systems run faster and better. Too often, these improvements take a back seat to the day-to-day managing of the database itself. The business might have to wait for available DBA time to get new features and functionality.

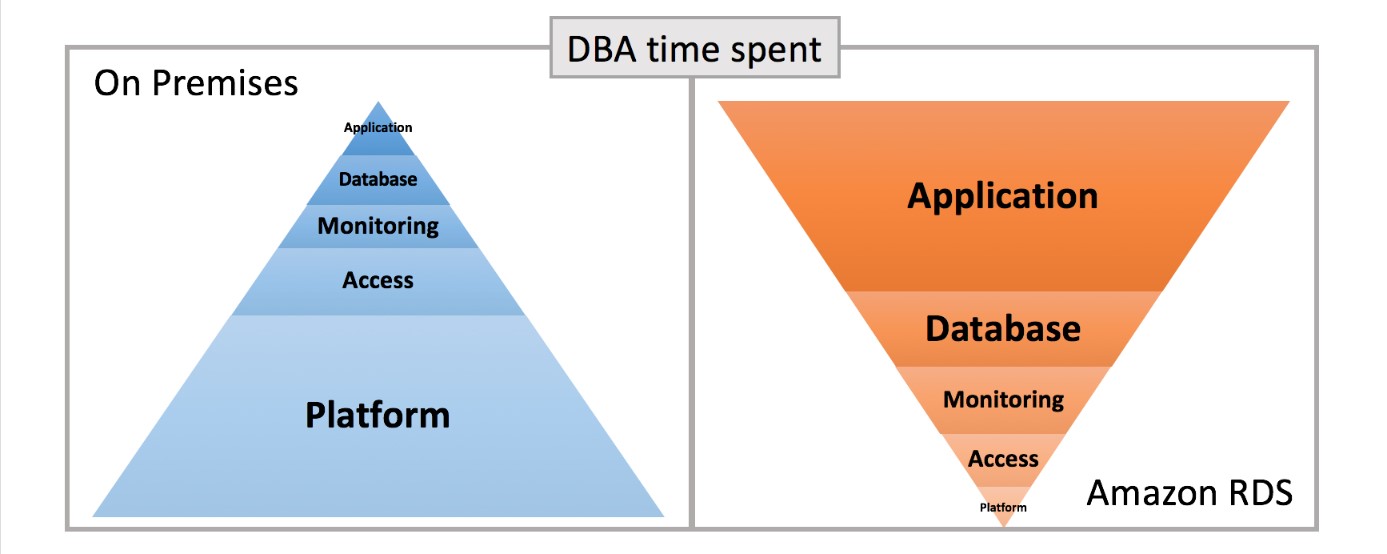
Amazon RDS value  
Amazon RDS makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks, such as hardware provisioning, database setup, patching, and backups. It frees you to focus on your applications, so you can give them the fast performance, high availability, security, and compatibility that they need.

The following table summarizes the benefits of using Amazon RDS.

|  |  |
| --- | --- |
| Easy to administer | Amazon RDS makes it easy to go from project conception to deployment. Access the capabilities of a production-ready relational database in minutes using the AWS Management Console, the AWS Command Line Interface, or simple API calls. No need for infrastructure provisioning or installing and maintaining database software. |
| Highly scalable | Scale your database’s compute and storage resources independently, often with no downtime. Many Amazon RDS engine types enable you to launch one or more Read Replicas to offload read traffic from your primary database instance. |
| Available and durable | Amazon RDS runs on the same highly reliable infrastructure used by other AWS products. When you provision a Multi-AZ DB instance, Amazon RDS synchronously replicates the data to a standby instance in a different Availability Zone. It also includes automated backups, database snapshots, and automatic host replacement. |
| Fast | Amazon RDS supports the most demanding database applications. Choose between two SSD-backed storage options: one optimized for high-performance OLTP applications, and the other for cost-effective general-purpose use. |
| More secure | Amazon RDS makes it easier to control network access to your database. It also lets you run your database instances in Amazon Virtual Private Cloud (Amazon VPC). This lets you isolate your database instances and connect to your existing IT infrastructure through an industry-standard encrypted IPsec virtual private network (VPN). Many Amazon RDS engine types offer encryption at rest and encryption in transit. |
| Inexpensive | You pay low rates, and only for the resources that you actually consume. You also benefit from the option of On-Demand pricing with no upfront or long-term commitments, and even lower hourly rates via Reserved Instance pricing. |

To create an RDS database, follow the tutorial in the [Amazon RDS User Guide](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_GettingStarted.html).

Amazon RDS DBA role  
Amazon RDS flips your role as a DBA around, so you spend only a fraction of your time on routine management tasks. This leaves the rest of your time for aligning your work more closely with the business and value derived from the data assets that you manage. You can then focus on lending your DBA skills to application teams and end users—helping deliver new features, functionality, and proactive tuning value to the core business.



Platform  
Creating a highly available system with minimal RPO is difficult to do correctly. When you provision a Multi-AZ DB instance, high availability is baked in. Amazon RDS automatically creates a primary DB instance and synchronously replicates the data to a standby instance in a different Availability Zone. Each Availability Zone runs on its own physically distinct, independent infrastructure and is engineered to be highly reliable. In case of an infrastructure failure, Amazon RDS performs an automatic failover to the standby or replica so that you can resume database operations as soon as the failover is complete. This important recovery functionality no longer requires an enormous investment in setup and management on the part of DBAs.

Note: If your client application caches the Domain Name Service (DNS) data, set a time-to-live (TTL) value of less than 30 seconds because the underlying IP address of a DB instance can change after a failover.

Although the amount of platform work in Amazon RDS is greatly reduced, there are still a few tasks that a DBA performs. You still might occasionally need to take explicit action to restore your database to another environment (for example, restoring development or test environments) or restore it from a logical corruption (for example, a table dropped or data was deleted accidentally). The Amazon RDS automated backup feature delivers point-in-time recovery (PITR), which makes this restore process easy for the DBA. Now your role is to take fast, targeted action to initiate the restore and perhaps manage table recovery. For more information about point-in-time recovery, see [Restoring a DB Instance to a Specified Time](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_PIT.html) in the Amazon RDS documentation.

Another example of a platform task is tracking license compliance, depending on whether you have a license-included model or a Bring Your Own License model. You can find details about [Microsoft SQL Server license options](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/SQLServer.Concepts.General.Licensing.html) and [Oracle license options](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Oracle.html) in the Amazon RDS documentation.

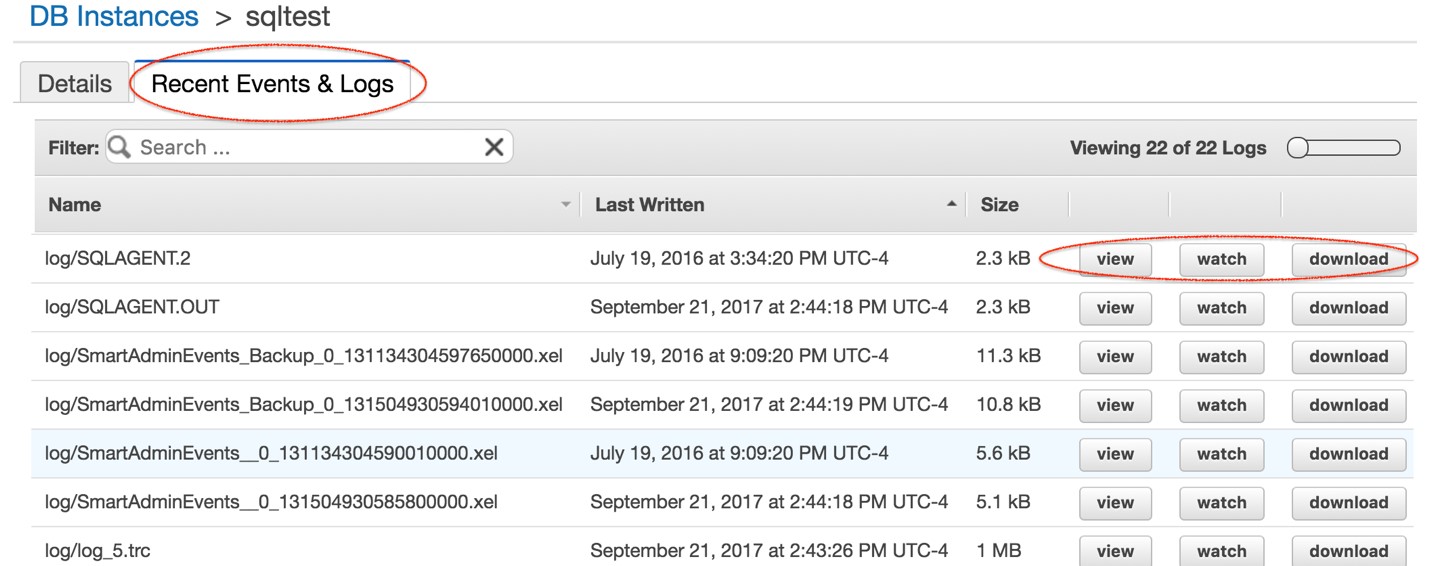
Access  
One of the key roles of a DBA is to implement the corporate security policy in the database to protect against accidental or malicious destruction of data. When you use Amazon RDS, your role in managing database access through users, privileges, roles, and profiles remains the same. You continue to enable users to connect to your database to access objects and data securely.

Note: For additional data protection, you can choose to [encrypt your RDS DB instance and snapshots](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Overview.Encryption.html) using industry standard AES-256 encryption algorithm. RDS SQL Server and Oracle DB instances also support Transparent Data Encryption (TDE).

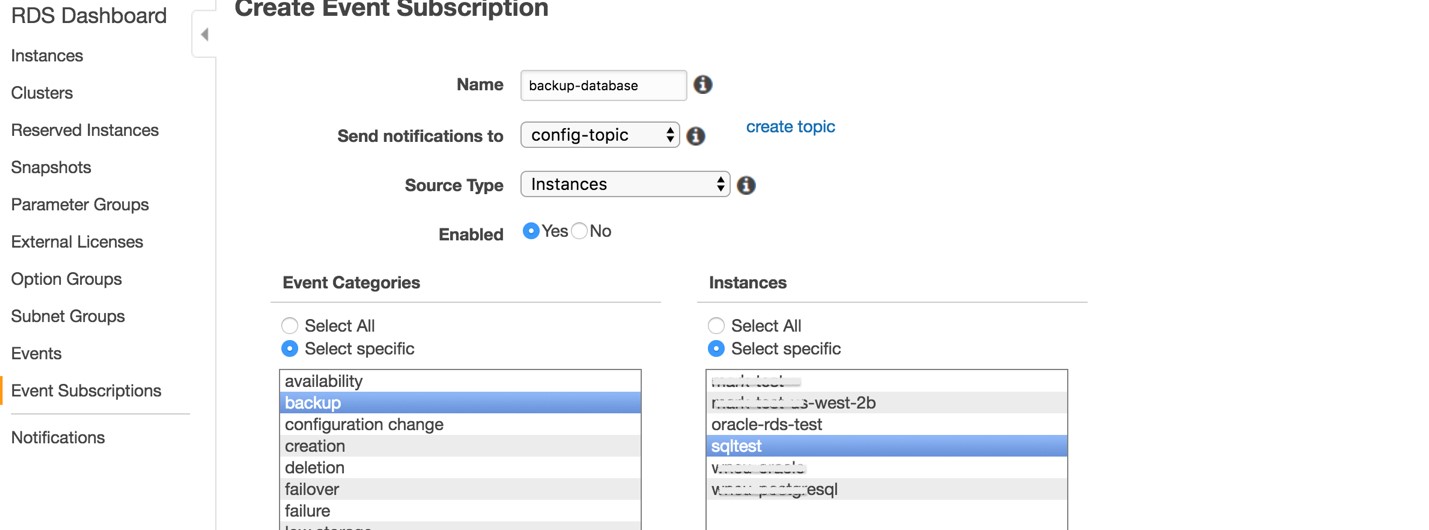
Monitoring  
In Amazon RDS, the database log files are still exposed to you. But instead of connecting to the host to get to the logs, you access the console or the AWS CLI. To view the log data on the console, choose one of your RDS instances and choose Logs to open the list of log files for that instance. In the list, look for the log that you want, and do one of the following.

* Choose view to display the entire log on your screen.
* Choose watch to display a tail of the end of the log showing the most recent events.
* Choose download to open a link that you can save as a file on your computer for review in an editor.

To learn more about log file access in Amazon RDS, review the AWS documentation on [Oracle log files](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_LogAccess.Concepts.Oracle.html) and [SQL Server log files](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_LogAccess.Concepts.SQLServer.html).



Amazon RDS exposes a significant number of events, which are grouped into categories that you can subscribe to using the console, AWS CLI, or API.  You are then notified when an event in that category occurs. You can subscribe to an event category for a DB instance, DB cluster, DB snapshot, DB cluster snapshot, DB security group, or DB parameter group. For example, if you subscribe to the Backup category for a given DB instance, you are notified whenever a backup-related event occurs that affects the DB instance. You also receive notification when an event notification subscription changes.



To read more about events in Amazon RDS, see [Using RDS Event Notification](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_Events.html) in the RDS User Guide.

Database  
Each proprietary instance on premises has important mechanisms for configuring the database. In Oracle, you change initialization files on the host database. In SQL Server, you change settings through the SQL Server Management Studio. In Amazon RDS, you manage the configuration of each of your database engines through a parameter group with [database parameters](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_WorkingWithParamGroups.html) that you can set or change using the Amazon RDS console, AWS CLI, or API. Once you establish a common set of parameters, you can then re-use that parameter group across multiple database instances.

Some DB engines offer additional features that make it easier to manage data and databases or provide additional security. In on-premises environments, you generally add features by installing or patching. Amazon RDS uses option groups to enable and configure these features. An option group can specify features, called options, that are available for a particular RDS DB instance. Options can have settings that specify how the option works. When you associate a DB instance with an option group, the specified options and option settings are enabled for that DB instance. You can set or make changes to [options and option groups](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_WorkingWithOptionGroups.html) using the Amazon RDS console, AWS CLI, or API.

Application  
Business and application teams look to the DBA or database developer to put together and manipulate data for efficiency using smart techniques. The better they are at this and the more time allotted to understanding the data that is being stored, the more value they can bring to the business. In on-premises environments, too often you don’t have much time to devote to your application teams for advice. DBA involvement can become either a bottleneck or an afterthought.

If you use Amazon RDS, you have time available to spend with application teams to focus on protecting an important asset—the data. You can work with teams on automating database changes in line with the application changes. Database changes are tricky. By suggesting optimizations and getting ahead of issues before they get to production, you can add value to the business and help prevent rework and rollbacks. As a DBA, you are ideally suited to this task. Instead of managing the infrastructure, it’s now possible for you to manage more of the business.

The revised list of responsibilities reflects a new focus on application management.

|  |  |  |
| --- | --- | --- |
| *Category* | DBA responsibility | With Amazon RDS |
| *Application* | Designing schema, access patterns, locking strategy, SQL development, and tuning. | ✓ |
| Modifying the database structure | ✓ |
| Optimizing application and end-user queries (reactive tuning) | ✓ |
| Archiving data | ✓ |
| Generating needed ad hoc reports by querying from the database | ✓ |
| Proactive performance tuning | ✓ |
| *Access* | Enrolling users and maintaining system security | ✓ |
| Controlling user access to the database | ✓ |
| *Database* | Parameter configuration and tuning | ✓ |
| Cache management |  |
| Job scheduling | ✓ |
| *Monitoring* | Alerting | ✓ |

Conclusion  
There is no question that management of your data affects the bottom line of your business. However, it’s unlikely that spending lots of time in the *Platform* category is helping your business deliver more features, reduce project time, or lower your budget. The biggest value add to the business is in providing end users and applications access to your data quickly and efficiently.