Course Overview

Course Overview

Hi everyone. My name is Hugo Barona, and welcome to my course, Managing Azure SQL Server Database Performance. I am an independent contractor and Azure Solutions Architect with a passion for Azure. In this course, we are going to see how we can leverage Azure services and features to manage the performance of your Azure SQL database in a smooth and easy way. Some of the major topics that we will cover include options you have available towards your SQL database in Azure, using Query Performance Insight for basic performance monitoring, using Azure SQL Analytics for advanced performance monitoring, and leveraging automatic tuning to ensure the best performance of your database. By the end of this course, you will understand which service and features you can leverage to tune the performance of your database. Also, you will understand how you can use automatic tuning to ensure the best performance of your database in an effortless manner. Before beginning the course, you should be familiar with the different options and methods you have available to tune the performance of your SQL database authored in SQL Server. I hope you will join me on this journey to learn more about these amazing Azure services and features to ensure the best performance for your Azure SQL database, with the Managing Azure SQL Server Database Performance course, at Pluralsight.

Hosting SQL Server Databases in Azure

Introduction

Hi there. My name is Hugo Barona, and welcome to this course about Managing Azure SQL Database Performance. In this course, you will learn how to manage the performance of your database hosted in Azure by leveraging the Azure service available. We will start by doing a recap of the different hosting options available for SQL database on Azure, considering the Infrastructure as a Service and Platform as a Service options so you can understand the main difference between these options. Then, we will iterate the different factors that should be considered when deciding for the service tools to your SQL database. And I will close this section by showing you a demo with all the steps to create a single SQL database on Azure. The next section, we will talk about the migration methods available, including online and offline methods. And then I will close this section by showing you a demo, explaining all the steps to migrate your SQL database to Azure using one of the migration tools available, Data Migration Assistant. So let's start.

Hosting Options for SQL Databases

In this section, we will talk about the hosting options available for your SQL database. In terms of options to host your SQL database, you have two types of offerings, Infrastructure as a Service and Platform as a Service. Infrastructure as a Service is purely a host model that provides you full control over your environments and the database engine, thereby you are also responsible to manage the infrastructure hosting your database, manage any software updates, and manage configuration to ensure a secure access to your database. Also, in case you need to provide a highly available database, it can become a really expensive option since you would need to increase the amount of resources of your VM by scaling up or creating additional instances of your VM by scaling out. Platform as a Service is a model oriented to build. This means you have an abstraction of the environment hosting your database, and you can focus more on what really matters, your database and your data. With this model, you do not need to set up an infrastructure or even manage software updates, and you have a secure environment available to host your applications. The service in this model is a fully managed service, therefore, you have several useful features available, such as autoscale, track detection, performance monitoring, and more. This offering can become a really cost-effective solution since you can allocate the least required resource and then scale depending on the needs and the load. So in terms of Infrastructure as a Service, we have the option to use Azure Virtual Machines with SQL Server installed. This typically is an option used in scenarios that clients want to migrate their database without

performing any changes or configuration to them, so it can be basically a lift and shift migration. Consequently, you can perform a fast migration of your database to Azure with minimal or no change required. Also, this option provides you full control over the database engine and the environment. In terms of Platform as a Service, we have at least three options available. The first is single database, and typically this option fits the scenario where you need to migrate just one database. This is a fully managed service with built-in artificial intelligence to support administration tasks for your database. Also, you have several useful features available that allow you to easily, if highly available, secure and perform in a database with minimal effort required from your site. In terms of cost, you have an amazing option for a pricing tier called serverless. This means you use a pay-as-you-go pricing model and you end up only paying for what you use. Then you have the Elastic Pool. Elastic Pool provides compute and storage resources that are shared between all the databases hosted in the pool. Elastic pools provide a simple and cost-effective solution for managing the performance of multiple databases within a fixed budget with all the benefits of Azure SQL Database. This option is especially useful in scenarios where you have predefined usage patterns of your database, and typically they are not all using resources at the same time, so you distribute those resources to the different databases being used by time to time. Elastic Pool also provides useful features to make it easier to perform your administration tasks, such as elastic jobs where you are able to execute jobs to perform tasks against your database in an automated way. The last, but not least option, is SQL managed instance. In case you are managing complex databases using features such as SQL CLR, SQL Server Agent, or even cross-database queries, you would face some limitations in case you plan to migrate your database to Azure. Managed instance is the solution to overcome those limitations since it is one of the most intelligent database services that requires 0 code changes and provides near 100% compatibility for you to migrate your SQL database to Azure. This option is a bit concurrent of VMs hosting your database since you can easily and quickly migrate your SQL database with near no code changes and assuring almost 100% of compatibility with your database considering that you do not need full control over your environment since with this option you have full abstraction of the environment hosting your database. Also, it provides you mechanisms to fully isolate the environment hosting your database and enhance the security of the environment hosting your database.

Factors to Consider When Choosing the Data Offering

So without these options available towards your database, sometimes it can be challenging to decide which option you should use. I compiled this list of factors that I consider relevant to a device when you are trying to decide between the different options. The first factor is cost. Both Platform as a Service and Infrastructure as a Service options cover underlying infrastructure and licensing in the base price. However, using Infrastructure as a Service options, you need to invest additional time and resources to manage your database, while in the Platform as a Service option, you get these administration features included in the price. Then we have the administration. Platform as a Service options minimize the amount of time that you need to invest in administration tasks for your database.

However, it also limits you in terms of custom administration tasks and scripts that you can perform or run. Then we have the service-level agreements, SLA. This is important, especially if you have requirements around availability of your database. Both Infrastructure as a Service and Platform as a Service provide high SLA and align with the industry standards. Infrastructure as a Service warranties a smaller SLA compared to Platform as a Service options. You need to implement additional mechanisms to ensure availability of your database in case you need higher SLA in your IaaS solution. And the last factor is time to move to Azure. This means how quick and easy you want to move your database to Azure. SQL Server in Azure VM can be the exact match of your environment, so migration from on-premise to Azure SQL VM is practically the same as moving the database from one on-premise server to another. Managed instance also enables easy migration, ensuring the compatibility of SQL Server features you are using in your on-premise environment. However, there might be some changes that you need to perform to your database before you migrate to a managed instance.

Demo: Create Azure SQL Database

Now, it's time for a demo. So we will see how can you easily create your SQL database in Azure, and after the creation, we will talk about some relevant features available in Azure SQL. So we start our demo by accessing the Azure portal. On the Azure portal, we will select SQL databases, and then we will create our first SQL database. Under the Basics tab, basically you have the ability to select the subscription where you want to create your database. Also, you can create a new resource group or select an existing one. In this case, I will leave it as pluralsight-demo. Then we need to provide a name for the database. I will call it demo-database. On the servers, you can create again a new server to host your database or select an existing one. I will just use this one that I created previously. Under the elastic pool setting, we'll leave the option as No since we don't want to use an elastic pool and we will just create a single database. And under Compute and storage, we can configure it and select, under General Purpose, Serverless, Gen5, so we can use the pay-as-you-go model. Then let's move to the Next tab, Networking. Under the Networking, you have the ability to configure your firewall rules. These firewall rules you can configure at the database level or server level. Also, you have the private endpoints. This feature is in preview mode. This feature basically allows you to create private links between your database and various Azure service. So moving to the Next tab, Additional settings. So in terms of the data source, you have the ability to basically create your database from a backup, an existing backup, or from a sample database, or you can leave as None and create a blank database. That's what we are going to do. Also, you have the ability to configure the database collection. In this case, I will leave as the default, SQL_Latin. And also, you can configure advanced data security settings. These settings you can configure again at the database level and server level. So moving to the next tab. Under the Tags, so you have the ability to take your resource, including your SQL database. This can be useful, for example, in scenarios where you share your subscription to multiple environments and you want to then drill down your costs per environment, so consequently one approach can be you take your resource with the different environments and then you can use

those tags to drill down the cost. So the last tab is basically the Review and creation. So here, you have estimated costs of your database that you're going to create, and also you have access to the terms of the service, and also summary of all the configurations that you enable to create your database. So let's create our database. The deployment will take a while, so I will pause this demo and we'll come back when the deployment is finished. Okay, so the deployment is finished and everything went well, so let's access our new SQL database. So accessing the resource, our SQL database, I will explain you a few features that you have available here that I think they are relevant in context of this course. So the first one is the Query Editor. This feature basically allows you to run queries against your database using the portal. Then you have under the Settings, you have the feature called Configure. This feature basically allows you to increase resources in your database by scaling up or decrease resources in your database by scaling down. Also, you have the Connection strings so you can access your connection string and start to connect your applications to your database. And then you have a section dedicated to performance. In the next modules, I will explain all these features in detail, and also I will show you a few demos explaining how to use these features. And also, another important section is Monitoring. So again, you have a few features available that allows you to monitor your database. Again, on the next modules, I will explain these features in detail and I will give you some demos explaining how to use them. So, it's the end of our demo showing how easy it is to create your Azure SQL Database in the Azure portal.

Migration Options

In this section of the module, I will explain the different options you have available to migrate your data or schema to your database hosted in Azure. The first option is the transactional replication. Typically, this option is used in scenarios with distributed systems where you have several databases and you need to synchronize changes between the different databases in real time. You can also use this method to migrate your data by continuously publishing changes from one database to another. This method uses a publisher/subscriber model. The publisher is the database that publishes changes made on some tables to the distributor. The distributor then collects all changes from the publisher and distributes them to the subscribers. You have many types of replications available, such as standard transactional, snapshots, merge replication, and more. The other option available is the bulk loads. This means you are able to bulk import your database by using a BACPAC file. Also, you can migrate data using bulk insert statements in Transact-SQL, or even migrate your data by using the Bulk Copy Program utility that allows you to export a given database table to a data file and then import from the file to a destination table in another database. This is a fully manual and offline option, and you need to be responsible to manage the data files and data formats being used. Another common option used is the Deploy Database to SQL Azure Wizard. Available since the version of SQL Server 2012, this option is a quick and easy option to export a single database, but still is a manual process, and there are some requirements and limitations. So all the tables in the database you are planning to migrate should have a clustered index, otherwise, your migration process will fail. Also, your database should not contain

extended properties since it's not supported by this wizard. The Azure Data Migration Assistant is a tool built to facilitate the migration process from your SQL Server Database hosted on-premise to Azure. It allows you to migrate database schema or data, or both. Also, apart from the migration option, you have as well the option to perform assessments to your migration process, so this tool will assess if your database has any issues that can bulk to migration, such as compatibility issues, also analyze if there exists any unsupported features being used. This tool has a recommendations feature that will identify and suggest the right SQL database queue that you should use when migrating your database to Azure. The migration process is done by using a single source database and a single target database. This is an amazing option to consider when migrating a small number of databases to Azure at no cost. When we have scenarios with a large number of databases to migrate, and especially if we need to do it quickly but without compromising the success of the migration, then we should consider using Azure Database Migration Service. This is a fully managed service that allows you to easily migrate your database from multiple sources to multiple targets in Azure with minimal downtime. It supports offline and online migration. The service uses the Data Migration Assistant tool to generate assessment reports to provide recommendations to guide you through the changes required before performing the migration. After you apply the changes required, when you are ready to begin the migration process, Azure Database Migration Service performs all the required steps automatically. The service handles all the migration process, applying the best practice as determined by Microsoft. It can be used in many scenarios, such as continuous synchronization between databases hosted on-premise and databases hosted in Azure.

Demo: Migrate SQL Database to Azure Using Data Migration Assistant

Now, it's time for a demo. In this demo, I will show you how to migrate a SQL Server Database to Azure using the Data Migration Assistant. We will also export some of the features available in this tool, including the assessment reports. So I will start my demo by showing you the Microsoft official page that allows you to download the Data Migration Assistant. Once you download it, you can install it in your local machine or in your on-premise environment. So let's open the Data Migration Assistant. On the assistant, you have two main sections, one related to assessments and one related to migrations. So let's start by creating our first assessment. On the assessment, let's give a name. I will call it demo-assessment. I will call it demo-assessment2 since the demo-assessment already exists. In terms of assessment type, it will be Database Engine since we want to assess the migration of my database located in my local machine in my SQL Server to Azure. And the source will consequently be SQL Server and the target Azure SQL Database. So let's create the assessment. During the creation of the assessment, you have the possibility to select the options in terms of the report types and all the checks that basically the assessment will do. In this case, it will check the database compatibility and also the feature parity. Then on the source, I will select my SQL Server, connect to the server, and then I will be able to select the database, in this case AdventureWorks2016. And then once you select the source, you are able to start the assessment. So once this

assessment is finished, you have a report similar to this one where you have all the issues identified in the assessment. In this case, we have one issue related with unsupported features and one issue related with partial supported features. Some of the issues are bulk into the migration and some of the issues do not bulk to migration, so the ones that bulk to migration, you need to basically act before you perform the migration. The ones that do not bulk, you can proceed with the migration, but you run the risk about facing some issues when you start to use your migrated database. Also, in terms of the generated report, you can always upload the report to Azure Migrate and use the report in case you plan to migrate your database using the Data Migration Service, so you can use the reports in the service itself. So this is the assessment itself. It's basically as simple as you can see. So now, let's create our migration. So for the migration, again, I will give it a name, demo-migration. Again, in terms of the source it will be my SQL Server, the target Azure SQL Database, and the migration scope. As I mentioned in the presentation, you have three options basically for the migration scope. You can migrate schema and data or schema only or data only. In this case, we'll migrate schema and data. Let's create the migration. So in terms of the source, I will select my SQL Server. Again, connect to the database, to the SQL Server, and then select the database, AdventureWorks2016. On the target, I will select my server and the database that I created in the previous demo. So again, I need to connect to the server. Once connected to the server, you are able to select the database, in this case demo-database. That's the database that we created in the previous demo. So once you select the source and target, you have the ability to select the objects that you want to migrate. This is the list of objects existing in the source database. So once you select all the objects, you can generate the SQL script, so let's generate the SQL script. Once the script is generated, basically you have two options to deploy your schema. The first option is you can save the script as a local file and then run the script manually against your target database, so you can save it here, and then I can run it. Or the second option is basically you can deploy the schema using the tool and continuing the migration process, so let's deploy the schema. So this deployment will take some time, so I will pause this demo and we will come back when the deployment is finished. So the schema deployment is finished, and as you can see here you have a summary with the deployment results about the total number of commands executed, and in case there are any errors each will be identified and you will have details about all the errors that happened during the deployment, so you cannot unfix the issues. And then in case you need, you can redeploy the schema again or you can proceed to migrate the data. So let's proceed to migrate the data. When you are trying to migrate the data, basically you have this ability to select the tables that you want to migrate. By default, all the tables are selected, but you can always exclude any table that you want to exclude from the migration. In this case, let's leave all the tables selected, and let's start the data migration. Again, this migration will take some time, so I'll pause the demo and we'll come back when the migration is finished. Okay, so the migration is finished, and as you can see, again, you have a summary with all the numbers. In this case, the total numbers successfully migrated. And also, in case you have any warnings or failed tables that fail to migrate, you'll be able to see it in the summary, the numbers, and also get the details about the issues that happened during the migration. So the migration is complete, and this is how easy it is to migrate your data from on-premise to Azure.

Overview

To recap, in this module, we explored the different options available to host your database on Azure and which option is the most recommended depending on the context. Also, we iterate the relevant factors that you should consider when trying to pick the best option to host your database on Azure. Then, we discussed the different options available to migrate your SQL database to Azure, the difference between them, and the scenarios where each of them are typically used. Finally, we learned how to create a single SQL database on Azure and how to migrate your SQL database from on-premise to Azure. I hope you enjoyed this module, and keep watching because in the next module we will talk about monitoring and troubleshooting your Azure SQL Database performance.

Monitoring and Troubleshooting Database Performance

Introduction

In this module, we will learn how to monitor and troubleshoot the performance of your database hosted in Azure by leveraging the Azure SQL features and tools available. We will start by comparing and identifying the main difference between the methods used on-premise and on Azure to monitor and troubleshoot your SQL database. Then, we will have a look to the different metrics and diagnostics logging capabilities available, how to configure diagnostics telemetry, and how you can use it to monitor the performance of your database. Next, we will see how we can use Query Performance Insight to perform basic performance monitoring in your database and explore the dashboards and metrics available. Last, but not least, we will see how to use Azure SQL Analytics to perform advanced database performance monitoring. So, let's start.

On-premises vs. PaaS Model for Database Monitoring and Troubleshooting

Typically, monitoring and troubleshooting SQL database performance can be hard and time consuming since it requires expertise to look to various dynamic management views and correlate the data in order to identify the issues and find a solution to mitigate them. SQL Server provides an extensive set of tools for monitoring, troubleshooting, and tuning your SQL Server database. The tool you should use will depend on the type of monitoring or tuning mechanisms you want to use. Some of these traditional mechanisms that used to be employed to monitor the SQL Server instance, such as SQL Profile and SQL Trace, are deprecated since SQL Server 2017. The monitoring and troubleshooting process is fully manual and you have no built-in graphical user interface with charts to show you the relevant performance metrics to support your analysis. Moving to a Platform as a Service model typically implies give up of a certain degree of control over your computing environment. Consequently, one of the main concerns related to this transition is the diminished level of transparency when managing and troubleshooting the performance of your workloads running in Azure. Fortunately, with Platform as a Service model, we have a comprehensive set of tools that deal with this concern, allowing you to easily identify and fix the majority of performance-related issues in an automated way. Also, you have an extensive set of charts available with relevant metrics and logs to support and simplify your analysis. Azure SQL Database also has built-in artificial intelligence available that simplifies and

automates this process since it allows you to constantly monitor your database and proactively identify the issues before it causes impact in your applications. In case you do not want to invest any time monitoring and troubleshooting your database, you can enable automatic tuning and let the artificial intelligence features monitor the performance of your database and fix issues automatically. Also, in case you want to enhance the performance of your database in Azure, we have always the options to automate scalability of your database using the auto-scale feature or using scripts to automate the scale-up and down of your database. So, unlike managing a troubleshooting database performance on-premise, which requires expertise and can be time consuming, in the Platform as a Service model you no longer need to be an expert in the subject and spend significant amounts of time to fix a majority of the performance issues that you can find in your database hosted in Azure. Along this module, we will see how we can leverage these features to monitor and tune the performance of your database hosted in Azure.

Metrics and Diagnostics Logging

In this section, we will discuss how you can use metrics and diagnostics logging to monitor the performance of your Azure SQL Database. By default, you have access to the core performance metrics of Azure SQL Database directly in the Azure portal, accessible from the database way or from the Azure monitor way. In case you need access to custom metrics and logs, you need to configure your diagnostics telemetry. By default, this is disabled since there is a low cost associated with enabling it. You configure your diagnostics telemetry logging by creating diagnostics settings. While creating the diagnostics settings, you can specify the type of diagnostics logs and metrics that you want to collect and the destination where you want to store this data. You can configure and manage metrics and diagnostics logging by using one of the following methods, Azure Portal, PowerShell, Azure CLI, Azure Monitor REST API, and Azure Resource Manager template. The metrics are collected every minute and are retained by the platform for 93 days. If you need to maintain access to them for an extended period of time, you have a few options such as sending them to a Log Analytics workspace or streaming them to an event hub or archiving them to a storage account. The same options are available for storage of diagnostics logs, which provide visibility to a variety of aspects of database operations, such as blocks, deadlocks, timeouts, and errors. As a side note, it is important to highlight that elastic pools and managed instances have their own separate diagnostics telemetry from the database they contain. This is important to note as diagnostics telemetry is configured separately for each of these resources. In terms of diagnostics telemetry for SQL Database in Azure, there is a limited set of metrics and logs that you have available. We have the SQLInsights logs that represent output produced from intelligent insights related to the performance of your database. Then we have automatic tuning, and these logs are the output of all performance recommendations available, the ones applied, and the ones reverted in your database. Also, we have the Query Store runtime statistics and Query Store wait statistics that store the statistics produced by Query Store in case it is enabled in your database. We have access also to logs related to all errors happening in your database, and a few logs that can be used to report important performance metrics such as database wait statistics, timeouts, blocks, and

deadlocks. In terms of metrics, we have basic metrics that store relevant performance metrics such as DTU and CPU usage percentage, DTU and CPU limits, physical data read percentage, and more. So, what can you do with all these metrics and logs? To start, you can analyze them to identify potential issues occurring in your database. Also, you can visualize the metrics and aggregate values by time periods to have a better and easier understanding of the performance of your database. With alerts, you can proactively act to mitigate issues as soon as they occur since you will be using metric alerts that will notify you or will execute an automated action as soon as the alert criteria is met. You can also use metrics together with the auto-scale feature to increase resources, scale up, or decrease resources, scale down, of your database based on the metric value and thresholds defined. In case you need to store for extended periods your logs and metrics, you can always store them in Azure Storage and export them to your machine or any other place that you decide. Also, we have many ways to retrieve those metrics, such as from a common line using PowerShell cmdlets or from a custom application using the REST API, or even from the command line using Azure CLI. So when it comes to comparing metrics and logs, they have some differences and different ways to be used. Metrics are basically numerical values that represent some aspect of your database at specific points in time. They are composed by name value and a time stamp since they are collected at regular intervals. They are ideal for fast detection of issues, especially when combined with near real-time alerting. Metrics are available for analysis in the Metrics Explorer blade located in the Azure portal. In terms of logs, things are a little bit different. Logs can be text or numeric data, and we have rich query languages available, such as the Kusto query language that allows you to perform deep analysis to identify the root cause of the reported issue. In case you send the logs to Log Analytics, you can view and query them on your Azure Log Analytics workspace. Alerts is a feature that allows you to proactively monitor your Azure resource by receiving notifications when important conditions are found in your monitoring data. With alerts, you can identify and address issues before the users of your system notice them. You can create alerts for your metrics and logs. An alert has for main components. The first is the target resource, and here you just select the resource that you want to monitor. Then we have the condition. It is the components you select to signal metrics or activity logs that you want to track and you specify the condition to fire the alert. In case you select a metric signal, you have to specify a specific value and a condition, such as greater than or lower than, or the threshold can be dynamic and Azure will track if the metric value has deviations from the threshold intervals configured. Then you specify the actions that you want to perform when the alert fires. These actions are defined by action groups, and these represent the logical grouping of actions. You have an extensive set of actions available, such as send email, trigger a Logic App, and more. Last, but not least, comes the alert details where you define the alert rule name, the description, and you specify if you want to enable the alerts upon creation.

Using Query Performance Insight for Basic Monitoring

In this section, we will see how we can use Query Performance Insight to perform basic monitoring of the performance of your database. Query Performance Insight is a feature available in Azure SQL Database that helps

you understand the impact of queries running against your database and how they are related to the consumption of resources allocate to your database or database servers. This feature is recommended in scenarios of basic performance tuning and troubleshooting. You can access this feature directly from the Azure portal and have access to the detailed data exposing performance of individual queries, and in case it is available, the corresponding performance recommendations to fix a given performance issue provided by SQL Database Advisor. Query Performance Insight relies on Query Store to manage performance data. By default, Query Store is enabled for Azure SQL Database, but you can enable it anytime if it is not running in your database. In order for you to view information in Query Performance Insight, Query Store needs to capture a few hours of data. If your database has no activity or if the Query Store is not active during a certain period, the charts will not show any information during that time range. Query Performance Insight provides you an extensive set of features to support your analysis, such as review top CPU-consuming queries, review top queries per duration, review top queries per execution count, view individual query details, and understand the performance tuning annotations available.

Demo: Using Query Performance Insights

Now it's time for a demo. In this demo, we will see how we can access and use Query Performance Insight and we will explore some of the charts available in this feature. Okay, so we start our demo by accessing the Azure portal. Then we basically select the SQL database, and on the SQL database I have this AdventureWorks2016 sample database that I created for the purpose of this demo. So if we select it, then if you scroll down, you will find under the Intelligence Performance, you will find the Query Performance Insight blade. If you select it, then you have this default view. This default view basically is showing the consumption of DTUs of your SQL database. DTU stands for Database Transaction Units, and these units are represented by the consumption of CPU memory reads and writes. So, this is showing aggregated values in the time period of 24 hours. If you scroll down, you will see a list of the top five queries with the most consumption in your database running against your database. Also, you can customize this view by going to the Custom tab and then selecting the different values you want. So I can select, for example, the past week, and instead of the 5 top queries, I want to select the 10 top queries. So if you select Go, the graph should update and show different values according to the time period selected, and also if you scroll down, you have the list of your 10 top queries with most of the consumption in your database. If you select one of the queries, you will be able to see the details of the query, okay, and also all the consumption of DTUs along the time that you selected, the time period that you selected. And also, you have a list of all the executions of these queries and the different values associated. So basically, this is a basic way to troubleshoot and monitor your database in terms of the performance and all the queries running against your database, and you can quickly analyze the query and eventually decide to improve the performance in case you need to. And basically that's how easy you can use the Query Performance Insight feature available and built in on your Azure SQL Database.

Using Azure SQL Analytics for Advanced Monitoring

In this section, we will see how we can use Azure SQL Analytics to perform advanced monitoring of the performance of your database. Azure SQL Analytics solution is the solution for advanced monitoring scenarios that allows you to collect and visualize important data related to the performance of your Azure SQL Database, elastic pools, and managed instances in a single place. It is based on Log Analytics, so it relies on the Log Analytics workspace available to manage all metrics and logs. It also allows you to leverage its functionality and create custom rules and alerts associated to the metrics collected, so you can proactively identify issues in your database and applications. In order to configure the type of data to show in Azure SQL Analytics, you need to configure the diagnostic settings of your Azure SQL Database and store the data in your Log Analytics workspace. The solution relies on the built-in intelligence to present relevant data and charts related to your Azure SQL Database performance, including database errors, timeouts, query durations, and query waits. Intelligent Insights is an artificial intelligence based solution that provides continuous database monitoring and relies on collecting SQLInsights logs. You can enable these logs by configuring their diagnostic settings and designating one or more destinations as the persistent store for log data, which can then be analyzed and reviewed. Intelligent Insights is capable of detecting temporary deviations from a long-term workload baseline, evaluating their impact, performing root cause analysis, and providing remediation options. Intelligent Insights is able to identify many database performance issues based on the following patterns. When a database is reaching resource limits. When there is an unusual workload increase. When there is unusual memory pressure. When there are locks happening in your database. When there is an increased MAXDOP. When there is a contention of pagelatch. When there is index missing in your tables. When new queries are executed against your database. When there is a significant increase of wait statistics. Also, when there is contention of TempDB. When you have a shortage of elastic pool DTUs. When you have plan regressions. When a database-scoped configuration value changes. When you have a slow client querying your database and taking too long to retrieve the data. And also, when there is a pricing tier downgrade that degrades the database performance. In this diagram, we have the architecture of database telemetry in Azure. As the source, we have Azure SQL Database, SQL managed instance, or Elastic Pool. Then we have Azure Diagnostics that enables us to configure the metrics and logs to collect and store them in the following target options, Log Analytics, Event Grid or Storage. It is important to note that each link from each source to Azure Diagnostics has a different color, and the links going from Azure Diagnostics to Log Analytics has all the colors combined. By this it means Azure Diagnostics is able to group the metrics and logs from multiple sources and send them to a specific target. Now that we have explored Query Performance Insight and Azure SQL Analytics as options to monitor and troubleshoot the performance of your database, it is important to understand some of the differences between them and what is the best option to the different scenarios. Starting with the Query Performance Insight, it is a built-in feature that comes with your Azure SQL Database and is based on the Query Store, so it requires the Query Store enabled. Also, it provides you many charts with relevant performance metrics and resource consumption. You are also able to visualize an individual

query's resource consumption and performance. It provides you an amazing graphical user interface that allows you to perform your basic performance analysis and use Azure Database Advisor to provide you performance recommendations to be applied to your database. Moving next to the Azure SQL Analytics, it is a solution based on Log Analytics, so it requires Log Analytics workspace with database telemetry available. It also requires an Azure SQL Analytics workspace so then you can use the solution. You can leverage the automated database performance monitoring using Intelligent Insights, and it also provides an amazing graphical user interface that allows you to easily perform advanced database performance analysis, and it provides you a single place to monitor and troubleshoot the performance of multiple databases.

Demo: Using Azure SQL Analytics

Now it's time for a demo. In this demo, we will see how we can create an Azure SQL Analytics workspace, and then we will explore some relevant features of Azure SQL Analytics. Okay, so we start our demo by accessing the Azure portal. Then let's create the resource. We just search for Azure SQL Analytics, and then we create the solution. So as I mentioned previously, the SQL Analytics requires a Log Analytics workspace available. So in this case, if I try to select this existing one, so this AdventureWorks workspace, since this workspace is being used in another SQL Analytics solution that I created previously, basically the creation will fail. So we need to, in this case, create a new workspace. So let's call it ps-demo-workspace3. In terms of resource group, we use the existing one, pluralsight-demo. Location we can leave as Australia Central. And let's create it. So the deployment just started, so I will pause this demo and we will come back when the deployment is finished. Okay, so the deployment is finished and we have already our Log Analytics workspace created. So let's create now our SQL Analytics solution. Again, this deployment tool starts, so it will take a while, so we will come back when the deployment is finished. Okay, so the deployment is finished and we have already created our resource successfully. So we if we go to the Resource groups, pluralsight-demo, and if we basically open the ps-demo-workspace3, the one that we created. So our SQL Analytics has here an option to view the summary. So you can see all the solutions associated with this SQL Analytics, and as expected, basically since we don't have any data, it gives you the message No data detected, okay. So as I mentioned previously, SQL Analytics, in order to provide you some insights and some data in relation with the metrics involved of your database, it requires a few hours of data collected previously. So we'll go back to the pluralsight-demo resource group and let's select the solution that I created previously. So in this solution, again if hit View Summary, I have an Azure SQL Database associated to this solution. So once you select the database, you come basically to the default view where it shows an extensive set of graphs with relevant metrics associated to your database. As you can see, for example, you can see the utilization bucket for your database. You can see the query durations in seconds, the timeouts, also in case there are errors happening in your database, you can see as well. And the most important, you have a graph related with performance issues. So this is actually Insights that is providing this data, Intelligent Insights. So once you select the graph, you basically come to this default view of Insights for your database, and if

you select the database, then you basically have a view in terms of all the insights related to your database. As you can see, for example, there is here a query that is causing locks. Also, another query that is causing high consumption of CPU. So, for example, if you select the query, you basically get details of the query. So as you can see, this is a query that is getting executed, and as you can see here, it's applying table lock, okay. So that's what is causing these reports to report the lock issue in relation with this query. So basically, it's quite detailed. It's really detailed, all the data that you have available, and these make it easier for you to troubleshoot and monitor the performance of your database and eventually fix any issues, because with this solution you basically get out of the box guidance to the issues that are happening with your database. So once again, this is how easy you can use Azure SQL Analytics, this time to basically perform a more advanced monitoring and troubleshooting of the performance of your database. As I mentioned previously, this requires you to configure a Log Analytics workspace, and also enables diagnostic settings in your database to basically collect the metrics and logs and report to the Log Analytics workspace. So then the SQL Analytics solution is able to pull that data and show you these graphs with relevant reports and metrics that basically allows you to easily identify issues and mitigate them.

Overview

To recap, in this module, we focused on understanding the main difference between monitoring and troubleshooting your database on-premise and using Platform as a Service model. Then we learned how to enable and use diagnostics telemetry in your Azure SQL Database to support you on monitoring and troubleshooting your database performance. Next, we'll learn how to use Query Performance Insights to perform basic database performance analysis and how to use Azure SQL Analytics to perform advanced database performance analysis. I hope you enjoyed this module, and keep watching, because in the next module, we will talk about how to manually tune you Azure SQL Database.

Tuning Azure SQL Performance Manually

Introduction

In this module, we'll talk about tuning your Azure SQL Database performance manually and understand how you can leverage Azure service to help you in these tasks while controlling the actions performed in your database. So we will start by talking about the different options you have available to help you to manually tune your database so you can understand when and why to use each of the options available. Also, we'll explore the Performance Recommendations feature in detail so we can see how you can leverage the feature to fix performance issues in your database. Then, we will talk about scenarios where you do not use any of the Azure services available and you basically perform the tuning by yourself.

Before Starting to Manually Tune Your Database

Azure provides you an extensive set of features and tools to support you in the daily tasks to manage your database, such as monitor and troubleshoot the performance of your database. These features and tools give you an abstraction of some or even all of these tasks and automatically performs them for you. One of the fully managed services that you have available to manage the performance of your database is automatic tuning. This service leverages the built-in artificial intelligence available and continuously monitors your database to identify any performance issues that occur. When the issues are identified, there are analyzed and actions are enumerated to fix these issues. In the next module, we'll see in detail how you can leverage this service to ensure the best performance for your database. Then we have the Performance Recommendations feature. This is a built-in feature available with your Azure SQL Database that provides you recommendations to improve the performance of your database. Along this module, we will see the capabilities of this feature. Then we have the Query Performance Insight that as we discussed in the previous module helps you monitor and identify issues in your database so then you can act and perform actions to mitigate the database issues including performance issues. Last but not least, we have Azure SQL Analytics, and as we discussed in the previous module, it helps you to monitor and troubleshoot your database performance typically used in more advanced scenarios. It is important to understand that all these features, services, and solutions discussed along this course can be unavailable depending on the host option you are using to host your database in Azure. For Azure SQL Database and Elastic Pool database, you have the following options available, automatic tuning, Performance Recommendations, Query Performance Insights, and SQL Analytics. Then we have managed instance database, and for this you have the following options available,

automatic tuning, but this service is only able to identify issues with the plans for your queries that run against your database and force the last good plan. Also, we have the SQL Analytics available to support you in the monitoring and troubleshooting your database. If none of these options enumerated is valid in your context and before you start to fix the issues by yourself, you can always consider the manual or the auto-scale features to increase the resource of your database since sometimes it can be the easier way for you to mitigate your database performance issues.

Performance Recommendations

Before we continue to talk in depth about the Performance Recommendations feature, it is important to understand its definition. Performance Recommendations is a feature based on SQL Database Adviser and available in your Azure SQL single or pooled database. This feature is enabled by default and continuously assess and analyzes the usage of your database and provides recommendations when it detects ways to improve the database performance. Performance Recommendations has an extensive set of capabilities to identify performance issues in your database and provide recommendations. The databases are monitored and assessed to identify issues by predefined patterns. We have the following group of recommendations provided. Create index recommendations. This group identifies indexes missing in the tables used in the queries that run against your database and then recommends the creation of the missing index. Also, we have the drop index recommendations. This group identifies indexes that are compromising somehow the performance of your database by identifying the tables being used by the queries running against your database and then identifying the index available on each of the tables. Then, it assesses if the index is not used for a long time, more than 90 days, or if it's duplicated, and if that's the case, then it recommends you to drop the index. This recommendation is not compatible in scenarios of using partitioned switching or index hints. Also, dropping unused indexes is not available for Premium and Business Critical service tiers. Then, we have the parametrize queries recommendations. And these recommendations are applied in scenarios that you have queries running against your database, constantly being recompiled and using the same query execution plan. In these scenarios, you will receive recommendations to parametrize your queries so they are not recompiled constantly. Also, we have the fixed schema issues recommendations. Yes, I said we had because this option is being deprecated by the Azure team since the Intelligent Insights already provides you this option and more. Last, you can also use Performance Recommendations in your applications by using its API. There are PowerShell cmdlets available that use the Azure SQL Database Adviser API to allow to manage performance recommendations.

Demo: Using Performance Recommendations

It is time for a demo. In this demo, we'll explore the capabilities of the Performance Recommendations feature, and also we will see how we can access the recommendations available and apply them. Okay, so we start our demo by accessing the Azure portal. Then we will access again our database. So the AdventureWorks2016. So the database

that we have been using in the previous demos. And then when you scroll down in the left menu, you have under the Intelligent Performance section, you have an option called Performance recommendations. So once you access this option, you have basically a list of available recommendations for your database. And also, you have a list of tuning history with all the recommendations that you applied or rejected. As you can see here, I have a recommendation applied for the drop index. So basically, this recommendation drops an index, a duplicated index, and as you can see, typically associated to a recommendation, you have estimated impact that explains how this issue is impacting your database performance. And then once you apply it, you have a validation report that basically expresses to you all the metrics relevant to explain how you improve the performance of your database. So if we go back to the lists again, in this case, as you can see here, I still have three more recommendations related with Drop index with the low impact, and then I have two recommendations with high impacts related with the creation of index. So there is missing index in these tables that basically are compromising the performance of the queries running against this database and trying to use these columns. So if we access one of these recommendations, basically as you can see, once again, you have estimated impact reports and in this case it says it needs this amount of disk space in order to create this index. Also, it identifies the schema, the table, and the index column associated to this recommendation. And you can always view the script and store the script in case you are planning to apply the recommendation by yourself, or basically you can hit the option to apply and just confirm it. Once you confirm it, basically this recommendation stays on status Pending and it will be validated and it will be applied. So you don't need to worry more about this. You just need to basically double check when this is finished in order to ensure that it was associated. Okay, so once you apply the recommendation, basically it will transit to the state of Validating, and by this it means once it's supplied, the Performance Recommendations feature will validate the impact of applying the recommendation. And it will basically transit only for the state of success if basically the feature realizes that there was an improve on the performance. Otherwise, the change will be reverted and you will get notified about the revert. It is important to understand that this step of validating sometimes can take up to a few days since it will be depending on the workloads running against your database because this artificial intelligence feature basically requires to analyze and investigate the workloads running against your database in order to identify if there was an improvement in the performance. So once it finished validating and transits to the state of Success, that's the stage that we can consider the recommendation as finalized and successfully applied to your database. So that's how easy you can inspect and investigate all the performance recommendations available for your database, and also apply some of these recommendations. You are not required to apply all the recommendations. It all depends on which ones you think they are relevant in context of your database. And that's how easy you can get basically and improve the performance of your database with this built-in feature available on your Azure SQL Database.

Manual Tune Azure SQL Database

In case you cannot use any of the options discussed before, or if you decide to perform the monitoring and tuning by yourself, then I would like to share with you a few scenarios and techniques to enhance the performance of your database. First, let's talk about the few scenarios related with your applications using your database. In case your applications make excessive data read and write operations to your database, this potentially is compromising your database performance and your application performance considering the network latency associated to each of the requests then to perform the operations. So in this scenario, you can potentially consider to use a different approach in case of reads. Instead of performing several single queries, you can try to combine the queries into few or even one query to get all the information you need or moving these queries to a sort procedure. Also, you can batch your ad hoc queries. In case of writes, you can again try to batch the operations to execute a single request to your database. Then we have applications that execute intensive workloads against your database. In this scenario, you could consider to increase resources by scaling up your database. In case you are using the highest Premium tier and the problem still persists, then you should consider increasing the number of instances by scaling out your database and using techniques such as cross-database sharding or functional partitioning. In case your application is running suboptimal queries, potentially you should investigate the queries being executed and identify any missing indexes or consider to tune your query or use query hinting. If your application is facing data access concurrency issues, such as deadlocking, you can consider to use some caching mechanism in your application that would reduce the round trips against your database. Now we have scenarios related with tuning your database. There are times that your performance issues are a result of missing an index or a duplicating index in your database. The scenario will require you to identify queries running in your database and then enumerate the tables being used, and for each of them analyze the existing or missing index. Other times, while analyzing the queries, we need to tune them or use query hinting. Also, sometimes you'll be required to consider to split your database data into multiple databases and then having your applications to perform operations against the required database. Sometimes this division per multiple database can be also done by functional partitioning, so you identify the functions performed in each of your databases by each of your applications and then create instances dedicated for each of the functions. So, in case the workload increases and you need to improve performance, you can scale independently by function and application. The last scenario is related with applications running multiple single database operations that result in multiple requests and an increase of the network latency. In this scenario, you should assess if you can batch queries and reduce the number of requests executed against your database. This slide just enumerates some of the most common scenarios related to database performance issues. In case you want to get more details and explore some of these scenarios in depth, I would recommend you watch the course of one of my colleagues. The course is Managing SQL Server Database Performance, available in Pluralsight.

Overview

To recap, in this module, we learned the different options you have available to manually tune your database performance while leveraging the Azure SQL Database features and tools available. We also learned about the Performance Recommendations feature and how to use it to improve the performance of your database. Then, we discussed some of the most common scenarios of databases or applications with poor performance and discussed possible techniques to use and fix the database performance issues. I hope you enjoyed this module, and keep watching because the next module is the last module of this course and we will talk about one of the most useful features available in Azure SQL Database that provides you automatic monitoring and troubleshooting of your Azure SQL Database performance and automatic fix of the performance issues.

Improving the Performance of Azure SQL Databases with Automatic Tuning

Introduction

In this module, we'll talk about improving the performance of Azure SQL Database using the automatic tuning feature. So, we will start by exploring the automatic tuning feature, explaining what is this feature, which capabilities it provides, and also we will talk about which Azure service allows you to use this feature. Then, we will iterate some of the major benefits of using automatic tuning when comparing to other features and tools available to improve the performance of your SQL database. We will also talk about some of the recommended scenarios to use automatic tuning and how you can benefit from this feature in those scenarios. This feature is disabled by default, so we will see the different options you can use to enable this feature, and in case you want to actively receive notifications of all performance recommendations available, applied by this feature, we will discuss a simple way you can use to enable these notifications. So let's start.

Exploring Automatic Tuning

Let's start by exploring the features of automatic tuning by looking to its definition, some of the features it provides, and the Azure service that provide this feature. Automatic tuning is a fully managed service available in your Azure SQL Database. The feature continuously monitors your Azure SQL Database and collects observations to be analyzed by the built-in intelligence and generate performance recommendations. Also, it leverages artificial intelligence to learn from all SQL databases on Azure and provides accurate recommendations. This feature also provides you continuous performance tuning while averaging the built-in artificial intelligence and machine learning to ensure the best performance and stable workloads in your database through continuous performance tuning. Automatic tuning provides you an extensive set of capabilities to ensure your database achieves the best performance. One of the capabilities is the automated performance tuning that continuously monitors and assesses the performance of your database including actively verifies the performance gains from the recommendations applied, and in case any of these recommendations are not resulting in any gains, the feature is able to roll back the recommendation applied and self-correct it. All these options performed by this feature are recorded, and it allows you access to the tuning history to view the history of actions. To ensure that your database achieves the best performance, this feature continuously monitors the performance the workloads running in your database, and

sometimes depending on the change of workloads the recommendations applied previously may no longer make sense since they are no longer reflecting any gains to your database, and in this case the feature is able to adapt to those changes and apply new or refer to existing recommendations previously applied to fix the issue. It is important to understand the different Azure services hosting options that provide this feature. This feature is provided with full support in Azure SQL Single Database and Azure SQL Pooled Database. It is important also to mention that you can always create your virtual machine with SQL Server 2017 or above installed, also the database there and enable automatic tuning since this feature is also available in SQL Server 2017 or newer versions. But there is a difference when we plan to use Azure Managed Instance database since this service only supports partially this feature since it only corrects the query plan and is forced to use the last good plan, so any actions related with creation of missing index or dropping duplicated or unused indexes are not supported in this service.

Benefits of Using Automatic Tuning When Compared to Other Performance Tuning Options

Now, let's highlight some of the benefits of using this feature so we can understand how we can benefit from it and in which scenarios would make sense to use this feature. One of the main the benefits of using this feature is the automated database performance tuning, so it requires no human interaction to perform the tuning actions. Also, this feature provides continuous database performance monitoring to ensure that all actions performed in your database reflect gains in terms of performance. Another important benefit of this feature, it is the fully managed service, so it requires no human interaction to work as expected, and you can expect to achieve great results using this feature since it was built by the Microsoft Engineers team and potentially the same team that masters the environment that hosts your database. So since it's a fully managed service, it does not require any database administration expertise from you or your team, and it also allows you to save time and effort related to all database administration tasks that you need to perform in case you would decide to manage and tune the performance of your database by yourself. Because this list of benefits, the feature is available with no additional costs. So after we review some of the main benefits of using this feature, I would like to also share some scenarios where it makes sense to use this feature. The first scenario is when you or your team has no database administration expertise to manage and tune the performance of your database. You could always decide to acquire third-party expert service or consulting service that would provide you this expertise. But this would definitely have a cost and most probably would not have the same quality of results as you have with this feature since this is designed by Microsoft the same way that they designed the database system that you are using to host your database. The second scenario is when you have a large number of databases to manage and you realize that this becomes feasible to manage the performance of your database. In this case, you can always consider to use this feature and let it automatically manage and tune the performance of your database. The third scenario is the common scenario when you have time or budget constraints or want to save time and money, so you can use this feature with no additional cost and save the time of doing all

those administration tasks by yourself. I believe there are many other scenarios that may make sense to use this feature, and it will depend on the context and conditions that you are using your database.

Options to Enable Automatic Tuning

So it's important to understand that this feature is disabled by default, so in order to use this feature the only action required is actually to enable it. Next, we will see the different options we have available to enable this feature. You can enable this feature at a database server level or a database level. In terms of server level, you have two options. The first is always the Azure portal. So from your database server features available, you have the automatic tuning feature available in Intelligence Performance section. Then, we have the REST API that provides two methods, GET and UPDATE. The GET method allows us to retrieve server automatic tuning options, and the UPDATE method allows us to update automatic tuning options on server. In terms of database level, we have three options. As well, you can enable it using the Azure portal by accessing your database and then accessing the Automatic Tuning option from the Intelligence Performance section. Then we have the REST API that provides two methods, GET and UPDATE, as well. The GET method to retrieve the database automatic tuning options and the UPDATE method to update the database options. Additionally, you can use Transact-SQL statements to manage the automatic tuning options and enable or disable this feature. It is important to understand that you are able to enable or disable the feature globally or enable or disable each of the three options that the feature provides, such as create index, drop index, and force quit plan. Also, it is important to know that the automatic tuning feature may be disabled by the system in case it needs to protect the workload performance. And in order to use this feature, you need to ensure the user you are using to manage the feature has one of the following RBAC roles, SQL DB Contributor, SQL Server Contributor, Contributor or Owner. In case you plan to manage automatic tuning by using Transact-SQL scripts so you can manage configuration by database, I want to share with you some useful statements you can use. The first statement allows us to query the current database options configured for automatic tuning. This query can be useful to validate which options are enabled or disabled. Also, in case the options are disabled by the system, we can have a look to the reason and reason desk columns and verify the option was disabled by the system or someone else. The second statement allows us to create all the performance recommendations associated to the database. And we can get relevant information associated to each recommendation, such as recommendation state, recommendation impact start, and reason of the current status. The third statement allows us to change the automatic tuning options in our database. We have three possible values, AUTO, INHERIT, and CUSTOM. The AUTO option will provide default values configured by Azure. The INHERIT option will inherit the options configured at the server level from the server hosting your database. The CUSTOM option will allow us to manually configure automatic tuning options. To configure manually the automatic tuning options, we can use this statement where you specify which options you want to change and the new value for each of the options. The possible values are ON, OFF, and DEFAULT. The ON

value enables the giving option and the value OFF disables it. The DEFAULT value will make the option to inherit the value from the configuration at server level.

Demo: Enable and Configure Automatic Tuning

Now it's time for a demo. In this demo, we will see how we enable automatic tuning at a server level, and also how we enable at the database level. Also, during this demo we will see how to configure the automatic tuning options at the server and database level. So we start our demo by accessing the Azure portal. So now we are going to access the database server, so I'll just filter by adventureworks in order to be able to easily find the database server. So on the database server page you basically, if you scroll down on the menu options, you have this Intelligent Performance section, and under this section you have two options, Automatic tuning and Recommendations. So selecting Automatic tuning, here on the automatic tuning you have the option to configure the inherit of the options, so you can inherit the options from the Azure defaults, and by this means by default force plan and the create index will be enabled and the drop index will be disabled. Or, you can set as don't inherit, and this means you need to enable and disable each one of these options manually. Once you configure the options and you apply, then basically the feature will start to analyze all the performance recommendations available to your database and will try to identify if it's possible or if it's able to apply any of those recommendations available. Once the recommendations are applied, then the feature will basically analyze the gains of applying those recommendations, and in case there was gains or the database performance for a given recommendation, then that recommendations will move to the status of Success. Otherwise if there are no gains, then the recommendation will move to the status of Reverted. So now let's move to the database itself. So selecting SQL databases, I'll select this AdventureWorks2016. This is the database we created in the previous demos. So, scrolling down again under the Intelligence Performance you have the Automatic tuning and you have the Performance recommendations options. So on the automatic tuning, in terms of the inherit, you have a new option for Server, and by this means if you select this option basically it will inherit the configurations from server level, so whatever configurations you configured at your database server, the database itself will inherit from them. And then again, you have the Azure defaults and don't inherit Once you configure all the options, again you apply and then the feature will basically start to analyze the performance recommendations available. And once again, we'll try to identify if it's possible to apply any of those recommendations available, and then we'll try to move to the Success state or to the Reverted state just in case there is no gains of applying a given recommendation. So this is how easy you can enable automatic tuning in your database server or even in your database level. And once again, if you don't have time, or even if you don't have expertise to manage and tune the performance of your database, then you can leverage this feature and basically use the Microsoft Engineers team knowledge to basically support you on these administration tasks in relation with the performance of your database.

Overview

To recap, in this module we looked at what is automatic tuning and the capabilities it provides. Also, we trained some of the main benefits of using automatic tuning and we explored some scenarios that are recommend to use automatic tuning. Since this feature is disabled by default, we had to look to the different options available to enable this feature, and then we explored the simple way to build the notification system to be notified about the different actions being performed by automatic tuning. I hope you enjoyed this module and this course. This is the last module of the course, but feel free to post any queries or questions you have and I will be happy to help you. Thank you for watching this course, and keep tuned to the new course coming soon.