**The lecture 9**

Managing data is a critical component of any business. Relational databases, and specifically Microsoft SQL Server, have been among the most common tools for handling that data for decades.

If we want to manage our data using the cloud, we *can* just use Azure virtual machines to host our own Microsoft SQL Server instances. Sometimes that's the right solution, but Azure offers another way that is often much easier and more cost effective. Azure SQL databases are a Platform-as-a-Service (PaaS) offering, meaning much less infrastructure and maintenance to manage yourself.

To understand better, let's consider a scenario: you're a software development lead at a transportation logistics company, Contoso Transport.

The transportation industry requires tight coordination among everyone involved: schedulers, dispatchers, drivers, and even customers.

Your current process involves piles of paper forms and hours on the phone to coordinate shipments. You find that paperwork is often missing signatures and dispatchers are frequently unavailable. These holdups leave drivers sitting idle; and as a result, some important shipments arrive late.

Customer satisfaction and repeat business are crucial to your bottom line, so your team decides to move from paper forms and phone calls to digital documents and online communication. Going digital will enable everyone to coordinate and track shipment times through their web browser or mobile app.

You want to quickly prototype something to share with your team. Your prototype will include a database to hold driver, customer, and order information. Your prototype will be the basis for your production app, so the technology choices you make now should carry over to what your team delivers.

Azure provides Platform as a Service (PaaS) services to help you manage all kinds of data, from highly structured relational data to unstructured data.

Here you'll learn why Azure SQL Database is a convenient, cost-effective, and secure way to host your relational databases.

## Why choose Azure SQL Database?

Your transportation logistics application requires stored procedures that run basic CRUD (Create, Read, Update, and Delete) operations. You have experience working with SQL Server and other relational databases.

You consider two choices for your database:

1. Host SQL Server on-premises. Your IT team runs a small in-house data center to support the finance department and a few other teams. You can work with IT to host a SQL Server deployment in their data center.
2. Host Azure SQL Database in the cloud. Azure SQL Database is based on SQL Server and provides the relational database functionality you need.

You've decided to build the web and application tiers for your logistics app on Azure. So it makes sense to also host your database there. But there are some other reasons why Azure SQL Database is a smart choice, and why it's even easier than using virtual machines.

* **Convenience**

Setting up SQL Server on a VM or on physical hardware requires you to know about hardware and software requirements. You'll need to understand the latest security best practices and manage operating system and SQL Server patches on a routine basis. You also need to manage backup and data retention issues yourself.

With Azure SQL Database, we manage the hardware, software updates, and OS patches for you. All you specify is the name of your database and a few options. You'll have a running SQL database in minutes.

You can bring up and tear down Azure SQL Database instances at your convenience. Azure SQL Database comes up fast and is easy to configure. You can focus less on configuring software and more on making your app great.

* **Cost**

Because we manage things for you, there are no systems for you to buy, provide power for, or otherwise maintain.

Azure SQL Database has several pricing options. These pricing options enable you to balance performance versus cost. You can start for just a few dollars a month.

* **Scale**

You find that the amount of transportation logistics data you must store doubles every year. When running on-premises, how much excess capacity should you plan for?

With Azure SQL Database, you can adjust the performance and size of your database on the fly when your needs change.

* **Security**

Azure SQL Database comes with a firewall that's automatically configured to restrict connections from the Internet.

You can allow access to specific IP addresses that you trust. Doing so allows you to use Visual Studio, SQL Server Management Studio, or other tools to manage your Azure SQL database.

To recap - with Azure SQL Database, Microsoft Azure manages the hardware, software updates, and OS patches for you. We provide buying options to help you get the performance you need at a predictable cost. Azure SQL Database also comes with a firewall so that you can control access to your data.

Although you don't need to be a DBA to use Azure SQL Database, there are a few concepts you should understand before you start. We'll cover these concepts in the next unit.

**Create your Azure SQL Database**

Your transportation company wants to set itself apart from other companies, without breaking the bank. You must have a good handle on how to set up the database to provide the best service while controlling costs.

Here, you'll learn:

* What considerations you need to make when creating an Azure SQL database, including:
	+ How a logical server acts as an administrative container for your databases.
	+ The differences between purchasing models.
	+ How elastic pools enable you to share processing power among databases.
	+ How collation rules affect how data is compared and sorted.
* How to bring up Azure SQL Database from the portal.
* How to add firewall rules so that your database is accessible from only trusted sources.

Let's take a quick look at some things you need to consider when you create an Azure SQL database.

**One server, many databases**

When you create your first Azure SQL database, you also create an *Azure SQL logical server*. Think of a logical server as an administrative container for your databases. You can control logins, firewall rules, and security policies through the logical server. You can also override these policies on each database within the logical server.

For now, you need just one database. But a logical server enables you to add more later and tune performance among all your databases.

**Choose performance: DTUs versus vCores**

Azure SQL Database has two purchasing models: *DTU* and *vCore*.

* **What are DTUs?**

DTU stands for *Database Transaction Unit*, and is a combined measure of compute, storage, and IO resources. Think of the DTU model as a simple, preconfigured purchase option.

Because your logical server can hold more than one database, there's also the idea of eDTUs, or *elastic Database Transaction Units*. This option enables you to choose one price, but allow each database in the pool to consume fewer or greater resources depending on current load.

* **What are vCores?**

vCores are *Virtual cores*, which give you greater control over the compute and storage resources that you create and pay for.

While the DTU model provides fixed combinations of compute, storage, and IO resources, the vCore model enables you to configure resources independently. For example, with the vCore model you can increase storage capacity but keep the existing amount of compute and IO throughput.

Your transportation and logistics prototype only needs one Azure SQL Database instance. You decide on the DTU option because it provides a good balance of compute, storage, and IO performance and is less expensive to get started.

**What are SQL elastic pools?**

When you create your Azure SQL database, you can create a *SQL elastic pool*.

SQL elastic pools relate to eDTUs. They enable you to buy a set of compute and storage resources that are shared among all the databases in the pool. Each database can use the resources they need, within the limits you set, depending on current load.

For your prototype, you won't need a SQL elastic pool because you need only one SQL database.

**What is collation?**

Collation refers to the rules that sort and compare data. Collation helps you define sorting rules when case sensitivity, accent marks, and other language characteristics are important.

Let's take a moment to consider what the default collation, **SQL\_Latin1\_General\_CP1\_CI\_AS**, means.

* **Latin1\_General** refers to the family of Western European languages.
* **CP1** refers to code page 1252, a popular character encoding of the Latin alphabet.
* **CI** means that comparisons are case insensitive. For example, "HELLO" compares equally to "hello".
* **AS** means that comparisons are accent sensitive. For example, "résumé" doesn't compare equally to "resume".

Because you don't have specific requirements around how data is sorted and compared, you choose the default collation.

**Create your Azure SQL database**

Here you'll set up your database, which includes creating your logical server. You'll choose settings that support your transportation logistics application. In practice, you would choose settings that support the kind of app you're building.

Over time if you realize you need additional compute power to keep up with demand, you can adjust performance options or even switch between the DTU and vCore performance models.

1. Sign into the [Azure portal](https://portal.azure.com/learn.docs.microsoft.com) using the same account you activated the sandbox with.
2. From the Azure portal menu or the **Home** page, select **Create a resource**. Select **Databases**, then select **SQL Database**.



1. Use these values to fill out the rest of the form.

| **Setting** | **Value** |
| --- | --- |
| **Subscription** | *Concierge Subscription* |
| **Resource group** | *[sandbox resource group name]* |
| **Database name** | *Logistics* |
| **Server** | *[See below]* |
| **Want to use SQL elastic pool?** | *No* |
| **Compute + storage** | *[See below]* |

1. Under **Server**, click **Create new**, fill out the form, then click **OK**. Here's more information on how to fill out the form:

| **Setting** | **Value** |
| --- | --- |
| **Server name** | A globally unique [server name](https://docs.microsoft.com/azure/architecture/best-practices/naming-conventions). |
| **Server admin login** | A [database identifier](https://docs.microsoft.com/sql/relational-databases/databases/database-identifiers) that serves as your primary administrator login name. |
| **Password** | Any valid password that has at least eight characters and contains characters from three of these categories: uppercase characters, lowercase characters, numbers, and non-alphanumeric characters. |
| **Location** | Any valid location from the available list below. |
| **Allow Azure services to access server** | This checkbox should be checked. |

1. The free sandbox allows you to create resources in a subset of the Azure global regions. Select a region from the following list when you create resources:
	* West US 2
	* South Central US
	* Central US
	* East US
	* West Europe
	* Southeast Asia
	* Japan East
	* Brazil South
	* Australia Southeast
	* Central India
2. Under **Compute + storage**, click **configure database**, then use the following steps:
	* To configure your database to use DTUs, click **Looking for basic, standard, premium?**
	* Depending on your application needs, choose **Basic**, **Standard**, or **Premium**.
	* Click **Apply**.
3. Click **Next : Additional settings**, then use these values to fill out the form.

| **Setting** | **Value** |
| --- | --- |
| **Data source** | *None* |
| **Database Collation** | *SQL\_Latin1\_General\_CP1\_CI\_AS* |
| **Advanced Data Security** | *Not now* |

1. Click **Review + Create** and then **Create** to create your Azure SQL database.

Important

Remember your server name, admin login, and password for later.

1. On the toolbar, click **Notifications** to monitor the deployment process.

When the process completes, click **Pin to dashboard** to pin your database server to the dashboard so that you have quick access when you need it later.



**Set the server firewall**

Your Azure SQL database is now up and running. You have many options to further configure, secure, monitor, and troubleshoot your new database. You can also specify which systems can access your database through the firewall. Initially, the firewall prevents all access to your database server from outside of Azure.

For your prototype, you only need to access the database from your laptop. Later, you can add additional systems, such as your mobile app.

For now, let's enable your development computer to access the database through the firewall.

1. Go to the overview pane of the Logistics database. If you pinned the database earlier, you can click the **Logistics** tile on the dashboard to get there.
2. Click **Set server firewall**.



1. Click **Add client IP**, this will automatically add the IP address for your development computer.



1. Click **Save**.

In the next unit, you'll get some hands-on practice with your new database and with Azure Cloud Shell. You'll connect to the database, create a table, add some sample data, and execute a few SQL statements.