



# Azure SQL – Transparent Data Encryption

## What is Azure SQL Database?

Azure SQL Database is a **fully managed, Platform-as-a-Service (PaaS)** cloud-based relational database service provided by Microsoft Azure. It is built on **Microsoft SQL Server engine** and eliminates the need for managing infrastructure, updates, backups, and security patches.

Unlike **SQL Server on Azure Virtual Machines (VMs)**, which requires manual configuration and maintenance, Azure SQL Database is a **serverless, auto-scaling, and highly available** database service designed for modern applications.

It offers multiple deployment models:

- **Single Database** (isolated database, fully managed)
- **Elastic Pool** (multiple databases sharing resources)
- **Managed Instance** (compatible with on-prem SQL Server, designed for easy migration)

## Key Benefits of Azure SQL Database

### 1. Fully Managed Service

- No need to handle software updates, backups, or patching.
- Automatic tuning and indexing improve performance without manual intervention.

### 2. High Availability & Reliability

- **99.99% SLA uptime** with built-in geo-redundancy and failover capabilities.
- Supports **Zone-redundant high availability** for business continuity.

### 3. Scalability on Demand

- **Auto-scale compute and storage** based on workload requirements.
- Serverless options available for cost-efficient usage.

### 4. Security & Compliance

- **Built-in Threat Detection, Advanced Data Security, and Transparent Data Encryption (TDE).**
- Compliance with **ISO, GDPR, HIPAA, and SOC** standards.

### 5. Cost-Effective

- Pay only for resources used (**DTU-based and vCore-based pricing models**).
- Supports **Azure Hybrid Benefit** for additional savings.

### 6. Easy Integration with Azure Services

- Works seamlessly with **Power BI, Azure Synapse, Logic Apps, and Azure Functions**.
- Supports **AI-driven performance tuning**.

## Use Cases of Azure SQL Database

### 1. Modern Web and Mobile Applications

- Used for SaaS applications requiring high availability and auto-scaling.

### 2. Multi-Tenant SaaS Applications

- Supports multiple customers by using **Elastic Pools** for cost-efficient database management.

### 3. Enterprise Data Warehousing

- Used for real-time analytics and business intelligence.

### 4. Disaster Recovery & Backup

- Ensures **automatic failover and geo-replication** for business continuity.

### 5. IoT and Real-Time Data Processing

- Stores **sensor data** and integrates with Azure services for analytics.

### 6. On-Premises to Cloud Migrations

- **Managed Instance** makes it easy to migrate on-prem SQL Server databases with minimal changes.

## Conclusion

Azure SQL Database is an **intelligent, secure, and scalable** PaaS offering that **removes infrastructure management overhead** while ensuring **high availability, security, and cost efficiency**. It is ideal for businesses looking to **modernize applications, migrate on-prem SQL workloads, and build cloud-native solutions**.

## What is Transparent Data Encryption (TDE)?

**Transparent Data Encryption (TDE)** is a security feature in **Azure SQL Database and SQL Server** that **encrypts data at rest** to protect it from unauthorized access. It ensures that stored data, including backups, is encrypted without requiring application-level changes.

## How TDE Works?

- **Encryption occurs at the storage level** (data files and log files) using **AES (Advanced Encryption Standard)** with a **256-bit key**.
- The encryption and decryption process is **transparent** to users and applications.
- The **Database Encryption Key (DEK)** is stored in the database but protected by **Azure Key Vault or a built-in server certificate**.
- When a user or application queries the database, data is automatically decrypted without manual intervention.

## Key Features of TDE

1. **Automatic Encryption of Data at Rest**
  - Encrypts **data files, log files, and backups** to prevent unauthorized access.
2. **No Application Changes Required**
  - Works at the database engine level, so applications don't need modifications.
3. **Azure Key Vault Integration**
  - Can use **Azure Key Vault Managed HSM** for **Bring Your Own Key (BYOK)** security.
4. **Protection Against Physical Theft**
  - Prevents data theft even if someone gains access to the storage or backups.
5. **Supports Compliance and Regulations**
  - Helps meet **GDPR, HIPAA, ISO 27001, and PCI DSS** security standards.

## Use Cases of TDE

1. **Protecting Sensitive Data**
  - Used in **financial, healthcare, and government** sectors to safeguard PII (Personally Identifiable Information).
2. **Securing Cloud Databases**
  - Ensures data security in **Azure SQL Database, Managed Instance, and SQL Server on VMs**.
3. **Preventing Data Breaches**
  - If storage media is stolen, **encrypted data remains unreadable** without the encryption key.
4. **Ensuring Regulatory Compliance**
  - Helps organizations meet compliance standards **without additional manual encryption efforts**.

## Conclusion

Transparent Data Encryption (TDE) is a **built-in security feature** in SQL Server and Azure SQL Database that **encrypts data at rest**, ensuring compliance and protection against unauthorized access. It provides **seamless security** without requiring application changes, making it a **critical feature for enterprise-grade security**.

## Process:

In this lab, we create an **Azure SQL Database**. First, navigate to **SQL databases** in the Azure Portal and click **Create**. Select your **subscription, resource group, and database name**. Create a new **SQL Server**, set **authentication credentials**, and configure **compute/storage**

with **Basic DTU**. Enable **public endpoint networking** and firewall rules, then review and create the database. After deployment, connect via **Query Editor, Azure Data Studio, or SSMS**.

### End Goal:

To set up an **Azure SQL Database with Transparent Data Encryption (TDE)** enabled by default, ensuring **secure, scalable, and managed cloud-based SQL storage** for applications.

## 😊 To begin with the Lab:

1. On your Azure Portal open All Resources. Then click on the hamburger icon.
2. There you will see the option for SQL databases. Click on it and from its dashboard click on create.

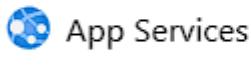
### — ★ FAVORITES —



All resources



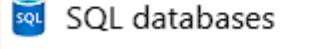
Resource groups



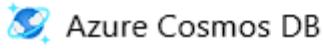
App Services



Function App



SQL databases



Azure Cosmos DB

3. As always first select your subscription and resource group.

### Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ⓘ

Azure Pass - Sponsorship (9e3f0cae-8274-4931-b16b-95242092e301) ✓

Resource group \* ⓘ

demo-resource-group ✓

[Create new](#)

4. Now you have to give the database name then for the server click on create new and create a new server.

## Database details

Enter required settings for this database, including picking a logical server and configuring the compute and storage resources

Database name *	demodb	✓
Server * ⓘ	Select a server	▼
	Create new	

✖ The value must not be empty.

5. While creating your server first give it a unique name the select your location.

## Server details

Enter required settings for this server, including providing a name and location. This server will be created in the same subscription and resource group as your database.

Server name *	demoserver102	✓
		.database.windows.net
Location *	(Europe) North Europe	▼

6. After that select the authentication method and create your user id and password.

Authentication method	<input type="radio"/> Use Microsoft Entra-only authentication <input type="radio"/> Use both SQL and Microsoft Entra authentication <input checked="" type="radio"/> Use SQL authentication	
Server admin login *	sqladmin	✓
Password *	*****	✓
Confirm password *	*****	✓

7. Now in the compute and storage option click on configure and select basic DTU for the workload.

## Configure ...

[Feedback](#)

### Service and compute tier

Select from the available tiers based on the needs of your workload. The vCore model provides a wide range of configuration controls and offers Hyperscale and Serverless to automatically scale your database based on your workload needs. Alternately, the DTU model provides set price/performance packages to choose from for easy configuration. [Learn more](#)

 SQL Database Hyperscale: Low price, high scalability, and best feature set. [Learn more](#)

Service tier

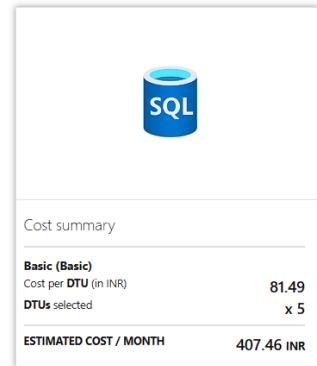
Basic (For less demanding workloads)

[Compare service tiers](#)

DTUs [Compare DTU options](#)

**5 (Basic)**

Data max size (GB)



## 8. Choose LRS for your backup storage.

### Backup storage redundancy

Choose how your PITR and LTR backups are replicated. Geo restore or ability to recover from regional outage is only available when geo-redundant storage is selected.

Backup storage redundancy 

Locally-redundant backup storage

Zone-redundant backup storage

Geo-redundant backup storage

## 9. Now in the networking enable the public endpoint and say yes to both of the firewall rules.

## 10. After that directly jump to the review page and create your database.

Basics **Networking** Security Additional settings Tags Review + create

Configure network access and connectivity for your server. The configuration selected below will apply to the selected server 'demoserver102' and all databases it manages. [Learn more](#)

### Network connectivity

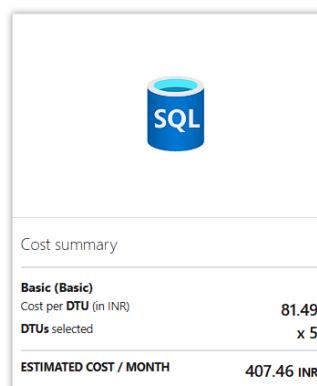
Choose an option for configuring connectivity to your server via public endpoint or private endpoint. Choosing no access creates with defaults and you can configure connection method after server creation. [Learn more](#)

Connectivity method \* 

No access

Public endpoint

Private endpoint



### Firewall rules

Setting 'Allow Azure services and resources to access this server' to Yes allows communications from all resources inside the Azure boundary, that may or may not be part of your subscription. [Learn more](#)  
Setting 'Add current client IP address' to Yes will add an entry for your client IP address to the server firewall.

Allow Azure services and resources to access this server \*

No  Yes

Add current client IP address \*

No  Yes

11. Now, this particular wizard is going to create two resources for us. One is a SQL database server and the other is a SQL database. This might take around three to five minutes.

12. Once the deployment is completed go to resources and open query editor.

13. Here you have to enter the password that you mentioned earlier while creating your database.

The screenshot shows the 'demodb (demoserver102/demodb) | Query editor (preview)' interface. On the left, there's a sidebar with navigation links: Overview, Activity log, Tags, Diagnose and solve problems, and the currently selected 'Query editor (preview)'. Below these are sections for Settings (Compute + storage, Connection strings, Properties, Locks), Data management (Replicas, Sync to other databases), and Integrations (Azure Synapse Link, Stream analytics (preview), Add Azure AI Search). The main area is titled 'Welcome to SQL Database Query Editor' and contains a 'SQL' icon. It shows a 'SQL server authentication' section with 'Login \*' set to 'sqladmin' and a 'Password \*' field, and a 'Microsoft Entra authentication' section with a 'Continue as behal.ritesh@gmail.com' button. There's also an 'OR' link and an 'OK' button at the bottom.

14. You can also connect your SQL Server with external applications like Azure Data Studio or SQL Server Management Studio.

15. Below I have used Azure Data Studio to connect my SQL Server. For that first you need to copy the server's name from the overview of your SQL Database and paste it in the Azure Data Studio.

Server name	: <a href="https://demodb121.database.windows.net">demodb121.database.windows.net</a>
Elastic pool	: <a href="#">No elastic pool</a>
Connection strings	: <a href="#">Show database connection strings</a>
Pricing tier	: <a href="#">Basic</a>
Earliest restore point	: No restore point available

16. Also, add the username and password then click on connect.

## Connection Details

Connection type	Microsoft SQL Server
Input type	<input checked="" type="radio"/> Parameters <input type="radio"/> Connection String
Server *	demodb121.database.windows.net
Authentication type	SQL Login
User name *	sqladmin
Password	*****
<input type="checkbox"/> Remember password	

17. You will be able to see the SQL Database. Also, if you expand the Demo DB then tables you will see the tables that come with the sample database which we enabled while creating the SQL Database.
18. You can right click on any table to select the top 1000 rows and columns of information.

The screenshot shows the SSMS interface with a query window open. The query window title is 'SQLQuery\_1 - (51 d...ladmin) X'. The query itself is:

```
1 SELECT TOP (1000) [AddressID]
2     ,[AddressLine1]
3     ,[AddressLine2]
4     ,[City]
5     ,[StateProvince]
6     ,[CountryRegion]
7     ,[PostalCode]
8     ,[rowguid]
9     ,[ModifiedDate]
10    FROM SalesLT.Address
```

The results pane displays the following table data:

	AddressID	AddressLine1	AddressLine2	City	StateProvince	CountryRegion
1	9	8713 Yosemite Ct.	NULL	Bothell	Washington	United States
2	11	1318 Lasalle Street	NULL	Bothell	Washington	United States
3	25	9178 Jumping St.	NULL	Dallas	Texas	United States
4	28	9228 Via Del Sol	NULL	Phoenix	Arizona	United States
5	32	26910 Indela Road	NULL	Montreal	Quebec	Canada
6	185	2681 Eagle Peak	NULL	Bellevue	Washington	United States
7	297	7943 Walnut Ave	NULL	Renton	Washington	United States
8	445	6388 Lake City Way	NULL	Burnaby	British Columbia	Canada
9	446	52560 Free Street	NULL	Toronto	Ontario	Canada
10	447	22580 Free Street	NULL	Toronto	Ontario	Canada

19. Now on the Azure Portal for your Database if you go to security and open the Data encryption for your SQL Server then you will see that the Transparent data encryption option is enabled by default and the security is being managed by the SQL Database itself. This means that the key used here is called PMK or platform-managed keys.
20. You can also use the CMK or Customer managed keys by enabling the database level customer-managed-key option and choosing the keys from your Azure Key Vault.

 demodb (demodb121/demodb) | Data Encryption ⭐ ...  
SQL database

Search Save Discard Feedback

Security

- Auditing
- Ledger
- Data Discovery & Classification
- Dynamic Data Masking
- Microsoft Defender for Cloud
- Identity
- Data Encryption**

Intelligent performance

Performance overview

**Transparent data encryption** Always Encrypted

 Transparent data encryption encrypts your databases, backups, and logs at rest without any changes to your application. To enable encryption, go to each database. [Learn more](#)

Data encryption ON OFF

Encryption status  Encrypted

Once database level customer-managed key is selected, switching to server level encryption key is only possible if the server is configured with service managed key.

Transparent data encryption ①

Server level encryption key  
 Database level customer-managed key (CMK)