AZURE SCENARIOUS BASED QUESTIONS

NAME : Dineshbabu M.R.
ACE NO: ACE12507

Scenarios:-

1. Your team needs to deploy a virtual machine in Azure portal to test a new software application. HERE, the team has reuested both windows and linux virtual machine?

how could you setup this virtual machine and what considaration are needed for pricing and OS licensing.

Steps to Set Up Windows and Linux VMs in Azure:

1. Create a Windows Virtual Machine:

1. **Log in to the Azure portal**: Navigate to https://portal.azure.com and log in with your credentials.
2. **Create a new resource**:
    a. Click on **"Create a resource"**.
    b. In the search bar, type **“Windows Virtual Machine”**.
    c. Select **"Windows Server"** or **“Windows 10”** based on your need for the VM type.
3. **Configure the VM:**
    a. Choose a **subscription** and **resource group**.
    b. Provide a **VM name**.
    c. Select a **region** where you want the VM to be deployed.
    d. Choose a **size** (the pricing depends on the selected size, which impacts CPU, memory, and storage).
    e. For **authentication**, you can choose **password** or **SSH key** for access.
    f. Provide the **username** and **password** if choosing password authentication.
4. **Configure networking**:
    a. Select a **virtual network** or create a new one if needed.
    b. Configure **public IP** settings, like static or dynamic IP address.
5. **Review and Create**:
    a. Review all configurations.
    b. Click **Create** to deploy the VM.

Not the

2. Create a Linux Virtual Machine:

1. **Create a new resource**:
    a. Click on **"Create a resource"** again.
    b. Search for **“Linux Virtual Machine”** or select from available distributions like Ubuntu, CentOS, or Red Hat.
2. **Configure the VM**:
    a. Follow similar steps to configure the VM as you did with the Windows VM:
        i. Select **subscription**, **resource group**, and **VM name**.
        ii. Choose **size** and **region**.
        iii. Select the **Linux distribution** image (e.g., Ubuntu 20.04 LTS).
        iv. For authentication, provide **SSH key** or **password** authentication.
3. **Configure networking** as you did with the Windows VM (with the option for a virtual network and public IP).
4. **Review and Create**:
    a. Review all settings.
    b. Click **Create** to deploy the Linux VM.

Pricing

Azure pricing for virtual machines depends on several factors:

- **VM Size**: Larger VMs (with more CPU, RAM, and storage) are more expensive.
- **Region**: Pricing varies by region, depending on where you deploy the VM.
- **Operating System**: The cost for Windows VMs is generally higher than Linux VMs due to licensing fees for the OS.
- **Storage**: Consider the type of storage you need (Standard HDD, SSD, Premium SSD) for the OS disk and data disks. Premium storage costs more but offers better performance.
- **Network Costs**: Data transfer costs could apply if you have outbound data or use services like load balancers or VPNs.

You can use the **Azure Pricing Calculator** to estimate costs based on the configurations you choose.

2. Operating System Licensing:

- **Windows VMs**:

- The cost of a Windows VM includes **Windows Server licensing**.
- Azure provides **pay-as-you-go** Windows Server licenses bundled with the VM cost, so you don't need to purchase Windows licenses separately.
- If you have existing **Windows Server licenses with Software Assurance**, you may be eligible for the **Azure Hybrid Benefit**, which can save up to 40% on Windows VM pricing.

- **Linux VMs**:
  - Azure does **not charge** for Linux operating system licensing. You only pay for the infrastructure (compute, storage, networking) costs.
  - If you use a commercial Linux distribution (e.g., Red Hat or SUSE), you will incur additional licensing costs, which can either be bundled in the Azure VM price or need to be handled separately.

2.

The IT security team has requested the sensitive data stored in Azure storage account be encrypted to meets complaince requirements.how could you ensure the data stored is azure storage is encrypte , and what encryption types are available.

Azure provides multiple layers of encryption to protect data at rest and in transit.

Encryption Options for Azure Storage

1. **Storage Service Encryption (SSE) for Data at Rest:**
   a. Azure Storage automatically enables **Storage Service Encryption (SSE)** for data at rest by default. This ensures that all data stored in your Azure Storage account (Blobs, Files, Tables, Queues) is encrypted without requiring any user intervention.
   b. **Types of Encryption**:
      i. **Microsoft-managed keys**: By default, Azure uses Microsoft-managed keys to encrypt the data.
      ii. **Customer-managed keys (CMK)**: You can use your own keys stored in Azure Key Vault to encrypt the data if you require more control over key management.
2. **Encryption in Transit:**

    a. Azure Storage uses **SSL/TLS encryption** for data in transit between the client and the Azure Storage service, ensuring that data is securely transmitted over the network.
3. **Azure Key Vault Integration:**
    a. If your compliance requirements demand a higher level of control over encryption keys, you can integrate **Azure Storage with Azure Key Vault** to manage your own encryption keys for more granular control. You can configure **customer-managed keys (CMK)** to encrypt the data stored in your storage accounts.

Encryption Types Available in Azure Storage:

1. **Azure Storage Service Encryption (SSE) with Microsoft-managed keys:**
    a. Default encryption for all data in Azure Storage.
    b. Transparent and seamless encryption that is fully managed by Microsoft.
    c. No need for user intervention or management of keys.
2. **Azure Storage Service Encryption (SSE) with Customer-managed keys (CMK):**
    a. Provides the ability to control the encryption keys used for encrypting your data.
    b. You store and manage the encryption keys in **Azure Key Vault**.
    c. You can rotate, revoke, or audit access to these keys, meeting more stringent compliance requirements.
3. **Encryption for Azure Files (Azure File Share):**
    a. Supports both SSE with Microsoft-managed keys and CMK for Azure File Shares.
    b. This is important if you are using Azure Files for shared storage.
4. **Azure Disk Encryption:**
    a. For virtual machines with Azure Managed Disks, you can use **Azure Disk Encryption** (ADE) to encrypt the OS and data disks.
    b. This uses BitLocker for Windows or DM-Crypt for Linux to provide full disk encryption.
5. **Blob Encryption:**
    a. **Blob Storage** also supports encryption using the same SSE options, ensuring that files stored as blobs are automatically encrypted with either Microsoft-managed keys or customer-managed keys.
6. **Azure SQL Database Encryption:**
    a. If you're storing data in **Azure SQL Database**, it has built-in encryption called **Transparent Data Encryption (TDE)**, which protects data at rest.

Steps to Enable Customer-Managed Keys for Azure Storage:

1. **Create a Key Vault**:
    a. Create an Azure Key Vault to manage your encryption keys.
    b. Set access policies to control who can access the keys.
2. **Generate or Import Encryption Key**:
    a. Either generate a new encryption key within the Key Vault or import an existing one.
3. **Enable Encryption with Customer-Managed Keys**:
    a. In the Azure Storage account, go to **Encryption** settings.
    b. Select the option to use **Customer-Managed Keys**.
    c. Choose the Key Vault and encryption key you wish to use.
    d. This will ensure that the data in your storage account is encrypted with the specified key.
4. **Configure Key Rotation and Access Control**:
    a. Set up key rotation policies and access control in the Azure Key Vault to ensure ongoing compliance.

Microsoft also provides encryption to protect Azure SQL Database, Azure Cosmos DB, and Azure Data Lake. Data encryption at rest using **AES 256 data encryption** is available for services across the software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS) cloud models.

3.

Your responsible for setting up Devops pipe line in azure devops for your appication. The pipe line must deplay code to an azure app service and notify the team if the deploymnet fails.

How could you configure this pipe lines ti meets the requirments.

Sure! Here's a shorter version of how to set up the pipeline and notification in Azure DevOps:

Here’s a streamlined version of how to set up the pipeline and notifications in Azure DevOps without the code:

1. Create the Pipeline

- **Create a new pipeline** in Azure DevOps and link it to your code repository.
- Choose either **YAML** or **Classic Editor** (YAML is more flexible, but Classic is easier for beginners).
- **Configure build steps** to:
    - Checkout the code from your repository.
    - Restore dependencies (e.g., NuGet for .NET apps).
    - Build the application.
    - Publish the build artifacts.
- **Configure deployment** to your Azure App Service using the **Azure Web App task**. This will automatically deploy the built application to your specified Azure App Service.

2. Set Up Notifications

Option 1: Use Azure DevOps Native Notifications

- Go to **Project Settings** > **Notifications**.
- Set up a **Custom Subscription** to send notifications when the pipeline **fails** (you can specify either **Build Failed** or **Release Failed**).
- Choose how you want to receive notifications (email, Teams, etc.), and set the recipients (team members, user groups).

Option 2: Use Notifications Within the Pipeline

- Add a task to send an email or notify via a service like **Teams** or **Slack** if the deployment fails.
- Use a **failure condition** to trigger the notification only if the deployment task fails.

3. Test the Pipeline

- Push changes to trigger the pipeline.
- Simulate a failure to verify that the notification is sent correctly when the deployment fails.

This setup ensures that your application is deployed automatically to Azure App Service, and the team is notified if the deployment fails.

4.

Your organization is moving to onPrimes sql data to azure. the database must remains accessable during migration with minimul downtime. which azure service could you use, and how could perform the migration.

To migrate an on-premises SQL database to Azure with minimal downtime, use **Azure Database Migration Service (DMS)**.

Steps:

1. **Prepare**: Backup the on-prem SQL database, ensure network connectivity to Azure, and set up the target Azure SQL Database or Managed Instance.
2. **Create DMS Instance**: In the Azure portal, create a new **Database Migration Service** instance.
3. **Configure Migration Project**: Set source as **SQL Server** and target as **Azure SQL Database** or **Managed Instance**.
4. **Perform Migration**: Use **Online migration mode** to keep the source database accessible. DMS performs an initial full data load, followed by continuous replication.
5. **Cutover**: When migration is complete, perform a brief cutover to finalize the transfer and ensure everything is synced.
6. **Post-Migration**: Validate the data and optimize the Azure SQL instance for performance.

Key Consideration: Use **online migration** to keep the database accessible during migration, minimizing downtime.