**Module 2 – Governance and Compliance**

**Azure Regions**

* An Azure region is a set of datacenters deployed within a specific geographic area, connected by a low-latency network.
* Each region belongs to a geography, which acts as a data residency boundary.
* Many regions offer availability zones—physically separate locations within a region for high availability and fault tolerance.
* Most regions are paired with another within the same geography for disaster recovery and data residency.
* Some regions are restricted for compliance, such as Azure Government and Azure China.
* When choosing a region, consider latency, compliance, service availability, and resiliency options.
* Azure has over 70 regions, the largest among cloud providers.
* Not all services are available in every region; always check service availability before deployment.

**2. Azure Subscriptions**

* Subscriptions are units of management, billing, and scale in Azure.
* Types of subscriptions include Free, Pay-As-You-Go, Cloud Solution Provider (CSP), Enterprise, Dev/Test, and Student.
* Subscriptions are used to isolate workloads, manage costs, and apply governance.
* Best practices include separating subscriptions for production, non-production, and sandbox environments.
* Use management groups to organize subscriptions at scale.
* Define clear processes for subscription creation, naming, and lifecycle management.
* Monitor subscription limits and request increases as needed.
* Limit the number of subscription owners for security.

**3. Resource Groups**

* Resource groups are logical containers for related Azure resources.
* Resources in a group can be managed collectively (deploy, update, delete).
* Resources can be in different regions, but the group’s metadata is stored in a single region.
* Resource groups cannot be nested.
* Resources can be moved between groups.
* Group resources by lifecycle, project, or environment (development, testing, production).
* Use consistent naming conventions for resource groups.
* Apply policies and RBAC at the resource group level for granular control.
* Use tags within resource groups for further organization and cost tracking.
* Resource Group Insights can be used for monitoring health and performance.

**4. Service Limits and Quotas**

* Each Azure resource and subscription has default limits (quotas) to prevent overuse and ensure fair resource allocation.
* There are adjustable (soft) limits, which can be increased via support requests, and non-adjustable (hard) limits, which are fixed maximums.
* Monitor quotas using the Azure portal or Quota APIs.
* Set up alerts for approaching limits.
* Plan for regional quotas, such as vCPUs per region.
* Free trial subscriptions cannot request quota increases.
* Quotas protect against accidental overuse, fraud, and capacity constraints.

**5. Management Groups and Hierarchy**

* Management groups provide a scope above subscriptions for unified policy, access, and compliance management.
* The hierarchy can be up to six levels deep, excluding the root and subscription.
* All subscriptions are under a tenant root by default.
* Keep the hierarchy flat (three to four levels maximum) to reduce complexity.
* Use management groups for policy assignment, not for billing or RBAC unless necessary.
* Assign RBAC at the lowest necessary scope to avoid over-permissioning.
* Use management groups to aggregate policy and initiative assignments via Azure Policy.
* Protect the resource hierarchy with proper settings.
* Policies and RBAC assignments cascade down the hierarchy.

**6. Resource Tagging**

* Tags are key-value pairs for organizing, tracking, and automating resource management.
* Tags are used for cost allocation, chargeback, and showback.
* Tags enhance search, filtering, and automation.
* Tags support compliance and operational requirements.
* Define a consistent tagging strategy aligned with business needs.
* Use foundational tags such as cost center, environment, owner, and project.
* Enforce tagging compliance with Azure Policy.
* Automate tagging using Azure CLI, PowerShell, or ARM templates.
* Regularly audit and clean up unused or inconsistent tags.
* Some resources have tag limits or do not support tags.

**7. Cost Management**

* Azure Cost Management and Billing tools help analyze, monitor, and optimize costs.
* Azure Advisor provides cost optimization recommendations.
* Budgets and alerts can be set for spending thresholds and notifications.
* Cost data can be exported and analyzed using APIs.
* Plan costs before deployment using the Azure Pricing Calculator.
* Use reserved instances and savings plans for predictable workloads.
* Apply Azure Hybrid Benefit for eligible licenses.
* Regularly review and act on cost optimization recommendations.
* Attribute costs using tags and management groups.
* Limit resource sprawl by deallocating or deleting unused resources.
* Perform cost analysis at different scopes, such as subscription, resource group, and management group.

**8. Azure Policy**

* Azure Policy enforces organizational standards, assesses compliance, and automates remediation.
* Policy definitions are JSON rules describing allowed or denied actions.
* Policy initiatives are groups of policy definitions for easier management.
* Policies can be assigned at management group, subscription, resource group, or resource level.
* The compliance dashboard provides an aggregated view of compliance state.
* Remediation can be performed in bulk or automatically for non-compliant resources.
* Azure provides many built-in policies for common scenarios, and custom policies can be created for specific requirements.
* Policies can be managed as code and integrated with CI/CD pipelines.
* Common use cases include restricting allowed regions, SKUs, or resource types, enforcing tagging and diagnostic settings, and ensuring regulatory compliance and security baselines.
* Policies are evaluated at resource creation or update and periodically for ongoing compliance.

**9. Role-Based Access Control (RBAC)**

* RBAC provides fine-grained access management for Azure resources.
* Components include security principal (user, group, service principal, or managed identity), role definition (built-in or custom), scope (management group, subscription, resource group, resource), and role assignment.
* Grant least privilege—only the access needed for the job.
* Limit the number of subscription owners (maximum three recommended).
* Use Privileged Identity Management for just-in-time access.
* Assign roles at the lowest possible scope.
* Regularly review and remove unnecessary privileged roles.
* Use conditional access and multifactor authentication for sensitive roles.
* Built-in roles include Owner, Contributor, Reader, User Access Administrator, and many more.
* Custom roles can be defined for specific permissions.
* RBAC is an allow model; no access is granted unless explicitly assigned.

**10. Azure Resource Manager (ARM)**

* Azure Resource Manager is the deployment and management service for Azure.
* Provides a consistent management layer for all resources.
* Supports infrastructure as code using ARM templates and Bicep.
* Enables group operations such as deploy, update, and delete.
* Integrates with access control, locks, and tags.
* Provides auditing and security features.
* All management operations go through ARM, ensuring consistency across tools such as the Portal, CLI, PowerShell, and SDKs.
* Each Azure service is managed by a resource provider.
* Declarative syntax allows defining infrastructure in code for repeatable deployments.
* Locks can prevent accidental deletion or modification of resources.
* ARM integrates with security monitoring and supports complex scenarios for SaaS and ISVs.

**11. Resource Locks**

* Resource locks prevent accidental deletion or modification of critical resources.
* Types of locks include Read-Only (prevents changes) and Delete (prevents deletion but allows modifications).
* Locks can be applied at the subscription, resource group, or resource level.
* Locks are inherited by child resources.
* Only Owner and User Access Administrator roles can create or delete locks.
* Use locks for mission-critical resources, such as production databases.
* Combine locks with RBAC for layered protection.
* Regularly review and update locks as environments evolve.