Lab Exercise: Data Movement Using Azure SQL Geo-Replication, Blob Storage, and Azure Data Factory

---

Course

Azure Essentials

Lab Objective

The objective of this hands-on exercise is to demonstrate enterprise-level data movement by replicating an Azure SQL Database to another region, storing data in cost-optimized Blob Storage, and using Azure Data Factory to copy sales data from a readable replica to Blob Storage for analytics purposes.

Business Context

OLTP databases are optimized for transactional workloads and are not suitable for large analytical queries. To avoid performance impact on production systems, data is replicated to a secondary readable database and then moved to blob storage, which is optimized for analytics and cost efficiency.

Learning Outcomes

1. Azure SQL Database and Active Geo-Replication
2. Azure Blob Storage access tiers
3. Azure Data Factory pipelines and copy activities

Final Architecture Goal

- **Primary SQL Database** in Region 1
- **Readable Secondary SQL Replica** in Region 2
- **Blob Storage (Cool Tier)** in Region 2
- **Azure Data Factory** to copy data from SQL replica to Blob Storage

Regions Used:

- South India
- Central India
- South East Asia

Prerequisites

1. Active Azure subscription
2. Access to three Azure regions
3. Azure SQL Database
4. Azure Storage Account
5. Azure Data Factory
6. Azure Portal access

Step-by-Step Implementation

Step 1: Create Azure SQL Database

1. Logged in to the Azure Portal.
2. Created a new **Azure SQL Database** using the **AdventureWorks sample database**.
3. Configured database with default settings suitable for the lab.
4. Database deployed successfully in **Region 1** (Primary Region).

Step 2: Configure Active Geo-Replication

1. Opened the created SQL Database.
2. Selected **Geo-replication** from the left menu.
3. Added a **secondary replica** in **Region 2**.
4. Configured the replica as **Readable Secondary**.

This ensures read workloads can be executed without impacting the primary OLTP database.

Step 3: Create Azure Blob Storage Account

1. Created a new **Storage Account** in **Region 2**.
2. Selected **Blob Storage** with **Cool access tier** to reduce storage cost.
3. Verified container creation for storing copied data.

Step 4: Create Azure Data Factory

1. Created an **Azure Data Factory** instance in **Region 1**.
2. Opened **ADF Studio**.
3. Created a new **Pipeline**.

Step 5: Configure Data Factory Pipeline

Source Configuration

- Source type: Azure SQL Database
- Connected to **Readable Secondary Replica**
- Selected table: **SalesOrderDetail**

Destination Configuration

- Destination type: Azure Blob Storage
- Container configured in Cool tier

Activity Used

- **Copy Data Activity**

This ensures sales data is copied from SQL replica to Blob Storage.

Step 6: Execute the Pipeline

1. Validated the pipeline configuration.
2. Triggered the pipeline manually.
3. Monitored pipeline execution until successful completion.

Step 7: Verify Data in Blob Storage

1. Navigated to the Blob Storage account.
2. Opened the configured container.
3. Verified that **SalesOrderDetail** data was available in blob format.

Result

- Azure SQL Database was created successfully.
- Active Geo-Replication was configured to a secondary region.
- Blob Storage with Cool tier was created for cost-effective analytics storage.
- Azure Data Factory successfully copied sales data from SQL replica to Blob Storage.

Conclusion

This lab demonstrated an enterprise-grade data movement pattern using Azure services. By leveraging SQL Geo-Replication, Blob Storage, and Azure Data Factory, analytical workloads were isolated from OLTP systems, ensuring performance, scalability, and cost optimization.

Key Takeaway

Data movement for analytics should always use **read replicas** and **cost-optimized storage**, making the architecture scalable, efficient, and production-safe.