**Cost Optimization Solution: Managing Billing Records in Azure Serverless Architecture**

**Problem Overview**

* **Database**: Azure Cosmos DB
* **Current Load**: >2 million records, each up to 300 KB
* **Challenge**: High storage costs due to growing read-heavy data with aging access patterns

**Goals**

* Maintain API contract
* Avoid data loss or downtime
* Ensure old records are still accessible (with seconds-level latency)
* Simple to implement and maintain

**Proposed Solution: Tiered Storage with Azure Blob Storage for Archival**

**Improved High-Level Architecture**

+----------------------------+

| Client / API |

+----------------------------+

|

v

+----------------------------+

| Azure API Management |

+----------------------------+

|

v

+----------------------------+

| Azure Function App (API) |

+----------------------------+

|

+------------------+------------------+

| |

v v

+-----------------------+ +---------------------------------+

| Azure Cosmos DB | | Azure Blob Storage (Archive) |

| (Hot Billing Data) | | (Cold Billing Records) |

+-----------------------+ +---------------------------------+

^ ^

| |

+-------------------------+ +-------------------------------+

| Timer-triggered Function| | Metadata pointer (Cosmos DB) |

| (Archival/Transition) | +-------------------------------+

+-------------------------+

**Improvements Explained**

* **Azure API Management**: Acts as a secure and scalable front door for APIs, enabling versioning, throttling, and analytics.
* **Separation of Concerns**: API Function App and Timer Function are isolated.
* **Metadata Pointer Layer**: Keeps minimal record in Cosmos DB for archived blobs.
* **Improved Scalability**: Clean architecture allows for caching, logging, and diagnostics per module.

**Solution Design**

**1. Data Tiering Strategy**

* **Hot Tier (Cosmos DB)**: Retain billing records from the last 3 months
* **Cold Tier (Blob Storage)**: Move records older than 3 months to Azure Blob Storage (Cool or Archive tier)

**2. Archival Process (Automated)**

* Scheduled Azure Function (daily or weekly)
* Moves records older than 3 months from Cosmos DB to Blob
* Stores a reference (pointer metadata) in Cosmos DB

**3. On-Demand Retrieval Process**

* API logic:
  + Check Cosmos DB first
  + If not found, check pointer and retrieve from Blob
  + Optionally cache result

**Sample Pseudocode & Scripts:**

**Terraform Template (Azure Resources)**

provider "azurerm" {

features {}

}

resource "azurerm\_resource\_group" "main" {

name = "billing-opt-rg"

location = "East US"

}

# Cosmos DB Account

resource "azurerm\_cosmosdb\_account" "main" {

name = "billingcosmosacct"

location = azurerm\_resource\_group.main.location

resource\_group\_name = azurerm\_resource\_group.main.name

offer\_type = "Standard"

kind = "GlobalDocumentDB"

consistency\_policy {

consistency\_level = "Session"

}

geo\_location {

location = azurerm\_resource\_group.main.location

failover\_priority = 0

}

}

# Cosmos DB SQL Database and Container

resource "azurerm\_cosmosdb\_sql\_database" "billing" {

name = "billing-db"

resource\_group\_name = azurerm\_resource\_group.main.name

account\_name = azurerm\_cosmosdb\_account.main.name

}

resource "azurerm\_cosmosdb\_sql\_container" "records" {

name = "records"

resource\_group\_name = azurerm\_resource\_group.main.name

account\_name = azurerm\_cosmosdb\_account.main.name

database\_name = azurerm\_cosmosdb\_sql\_database.billing.name

partition\_key\_path = "/partitionKey"

throughput = 400

}

# Storage Account and Blob Container for Archival

resource "azurerm\_storage\_account" "archive" {

name = "billingarchivestg01"

resource\_group\_name = azurerm\_resource\_group.main.name

location = azurerm\_resource\_group.main.location

account\_tier = "Standard"

account\_replication\_type = "LRS"

}

resource "azurerm\_storage\_container" "archive" {

name = "billing-archive"

storage\_account\_name = azurerm\_storage\_account.archive.name

container\_access\_type = "private"

}

# App Service Plan for Function App

resource "azurerm\_app\_service\_plan" "plan" {

name = "billing-function-plan"

location = azurerm\_resource\_group.main.location

resource\_group\_name = azurerm\_resource\_group.main.name

kind = "FunctionApp"

reserved = true

sku {

tier = "Dynamic"

size = "Y1"

}

}

# Function App (API + Archival Timer)

resource "azurerm\_function\_app" "billing\_fn" {

name = "billing-function-app"

location = azurerm\_resource\_group.main.location

resource\_group\_name = azurerm\_resource\_group.main.name

app\_service\_plan\_id = azurerm\_app\_service\_plan.plan.id

storage\_account\_name = azurerm\_storage\_account.archive.name

storage\_account\_access\_key = azurerm\_storage\_account.archive.primary\_access\_key

os\_type = "linux"

version = "~4"

app\_settings = {

"AzureWebJobsStorage" = azurerm\_storage\_account.archive.primary\_connection\_string

"COSMOS\_DB\_URI" = azurerm\_cosmosdb\_account.main.endpoint

"COSMOS\_DB\_KEY" = azurerm\_cosmosdb\_account.main.primary\_key

"BLOB\_CONN\_STR" = azurerm\_storage\_account.archive.primary\_connection\_string

"WEBSITE\_RUN\_FROM\_PACKAGE" = "1"

}

}

# (Optional) API Management for Gateway & Throttling

resource "azurerm\_api\_management" "apim" {

name = "billing-apim"

location = azurerm\_resource\_group.main.location

resource\_group\_name = azurerm\_resource\_group.main.name

publisher\_name = "Contoso"

publisher\_email = "admin@contoso.com"

sku\_name = "Consumption\_0"

}