1. Why This Is Critical

Azure Cosmos DB bills you for compute (RU/s), storage (GB/mo) and bandwidth. With 2 million records at ~300 KB each, storage alone is ~600 GB—and growing. Offloading cold records frees up both storage capacity and RUs for your hot workload.

2. Core Idea: Invisible Storage Tiering

Hot Tier (≤ 90 days): remains in Cosmos DB

Cold Tier (> 90 days): lives in compressed blobs

Index: a tiny map in Azure Table Storage (`recordId → blobUri`)

APIs: unchanged—calls still go to `/getBillingRecord` or `/saveBillingRecord`

All routing happens in your backend code (or Function App), never at the API gateway.

3. End‐to‐End Flow

1.Write Path:

- Exactly as today: writes always land in Cosmos DB.

2.Archive Function - (Timer‐triggered, e.g. daily or hourly):

CODE

3 Read Path - (inside your DAL or HTTP Function):

- \*\*APIs and JSON schemas stay identical\*\*—only your implementation changes.

CODE

4. No API‐Contract Changes

Endpoints - (`/read`, `/write`, etc.) remain as is.

Request/response JSON shapes don’t change.

Clients see the same URLs, the same status codes, the same payload fields.

All the “archive/fallback” logic is tucked into your service’s data‐access module.

5. Zero Downtime & Zero Data Loss

Idempotent archive: delete from Cosmos DB only after blob + index succeed.

Dual‐write phase (optional): temporarily write new records both to Cosmos and to blob if ≥ 90 days—guarantees no gap.

Phased rollout: start with small backfill batches, monitor, then scale up.

6. Simplicity & Maintainability

4 PaaS resources: Cosmos DB, Blob Storage, Table Storage, one Function App.

<150 LOC of azure-functions code in Python/JS/C#.

Serverless\* → no VM or container infra.

Schema-free index in Table Storage.

Auto-scaling and pay-per-use for both Functions and Blob.

7 Architecture Diagram

Clients

↓

┌───────────┐

│ HTTP API │ — existing endpoints, unchanged →

└───────────┘

↓

┌───────────────────┐

│ Data‐Access Layer │ ← contains “try Cosmos; on 404, fallback to Blob”

└───────────────────┘

↓ ↓

┌────────┐ ┌─────────────┐

│Cosmos │ │Table Index │

│ (Hot) │ │(Id → Blob) │

└────────┘ └─────────────┘

↓

┌────────┐

│ Blob │

│(Cold) │

└────────┘