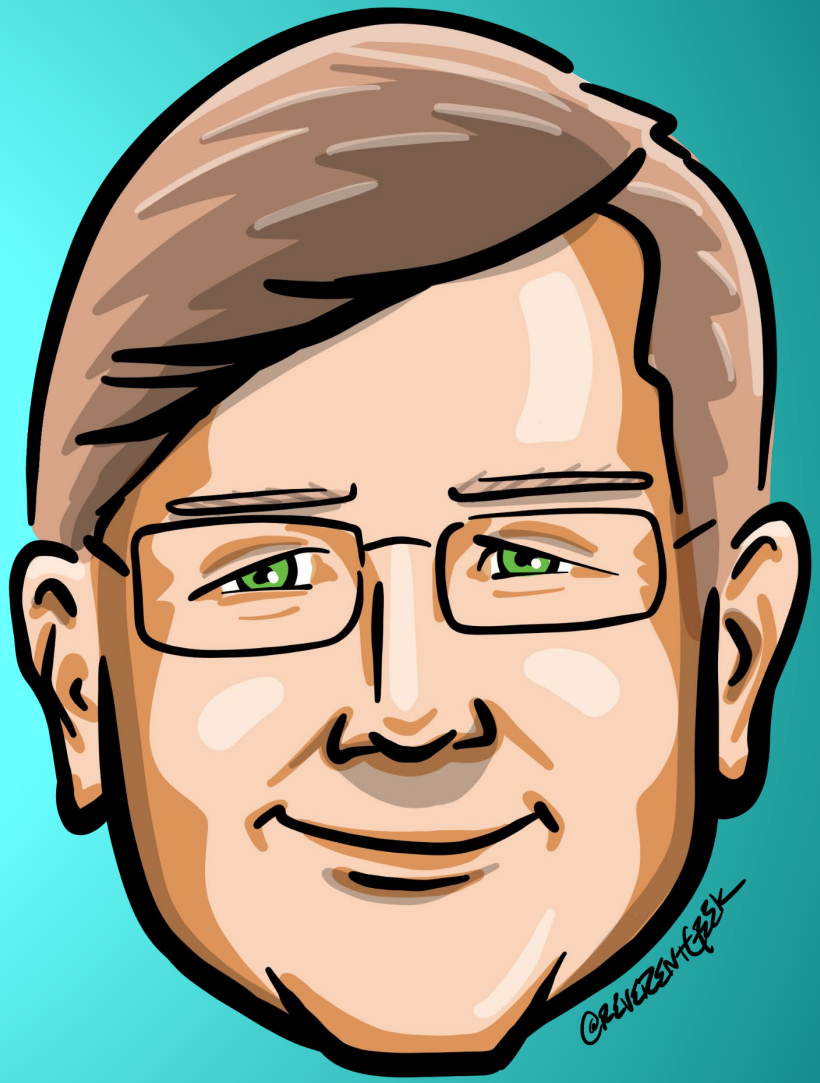
THE HITCHHIKER'S GUIDE TO THE COSMOS

MUSIC CITY TECH

Chad Green

Who is Chad Green

- chadgreen@chadgreen.com
- TaleLearnCode
- TaleLearnCode.com
- ChadGreen & TaleLearnCode
- ChadwickEGreen



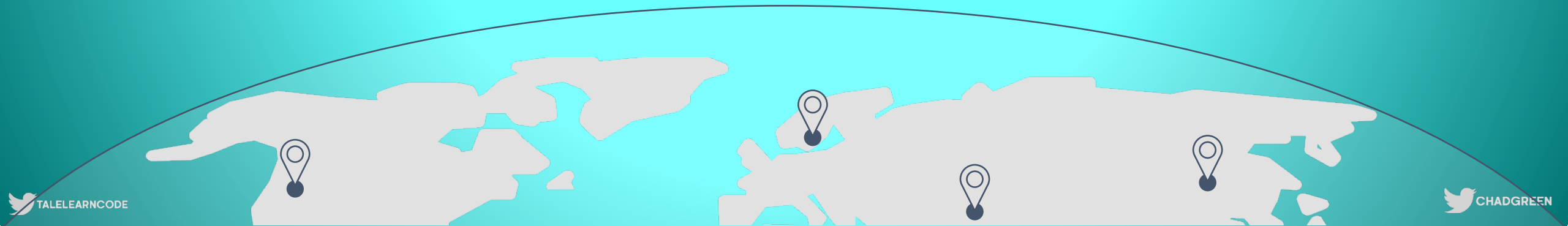
What is Cosmos DB

The Hitchhiker's Guide to the Cosmos

Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

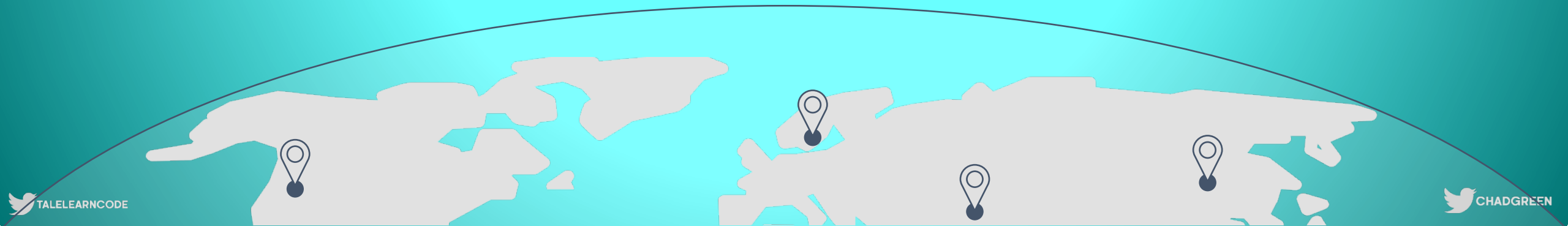
Turnkey global distribution



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Turnkey global distribution

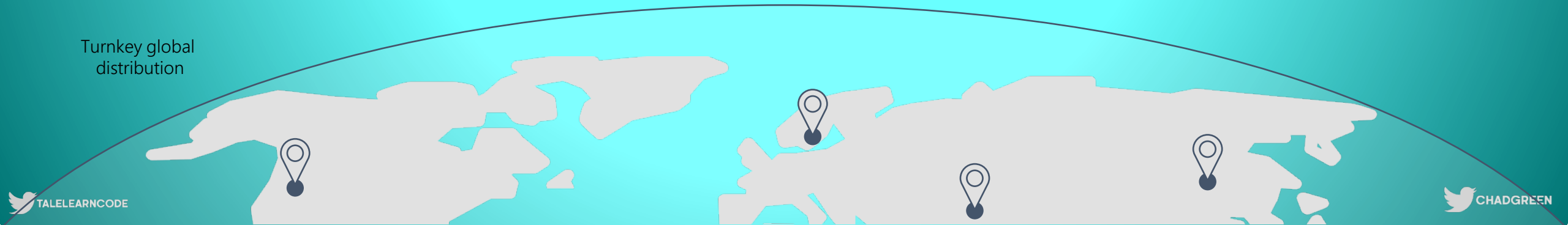


Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Elastic scale out of storage & throughput

Turnkey global distribution

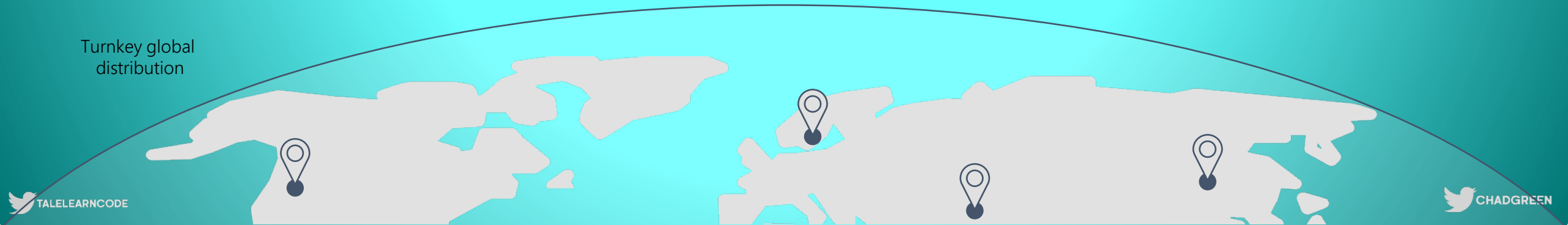


Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Elastic scale out of storage & throughput

Turnkey global distribution



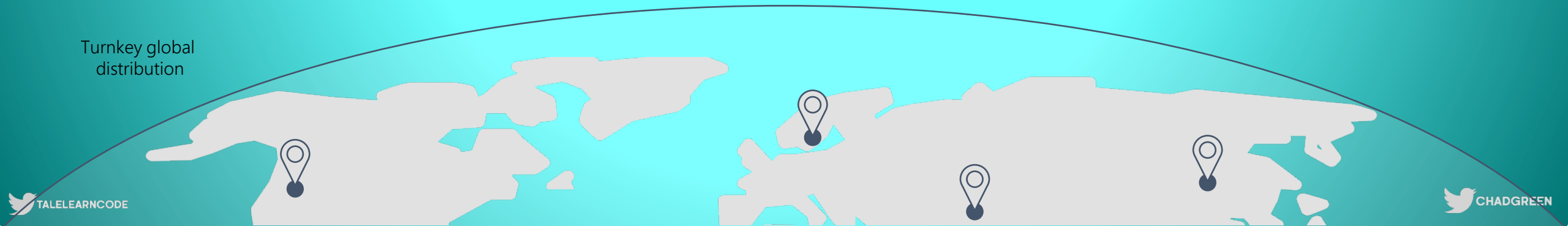
Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Guaranteed low latency at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



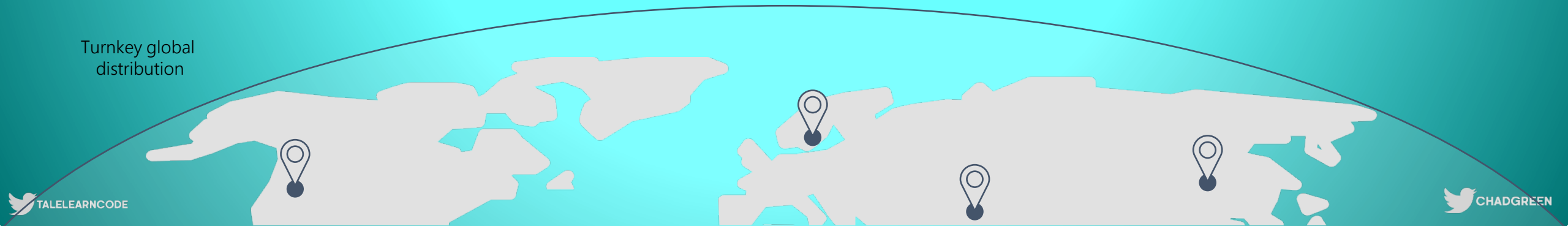
Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Guaranteed low latency at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

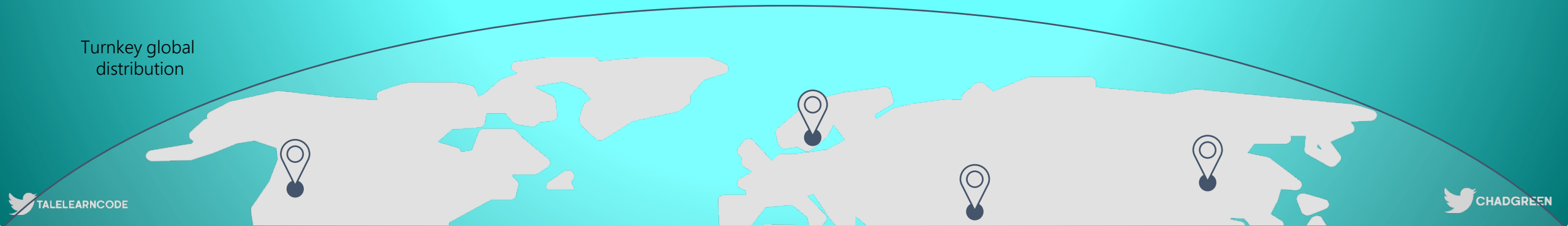
A globally distributed, massively scalable, multi-model database service

Five well-defined consistency models

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

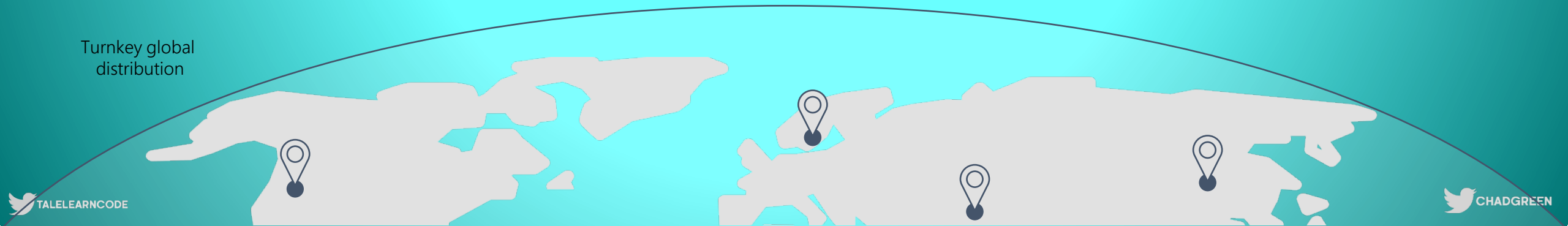
A globally distributed, massively scalable, multi-model database service

Strong

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

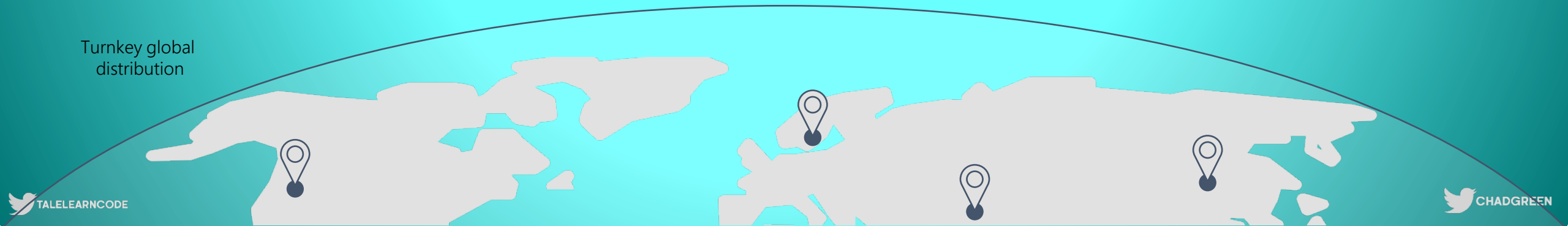
A globally distributed, massively scalable, multi-model database service

Strong

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

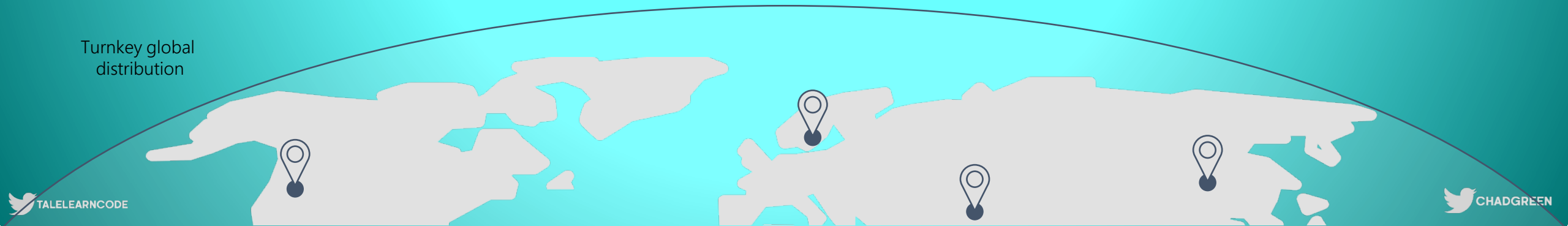
Strong

Bounded Staleness

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

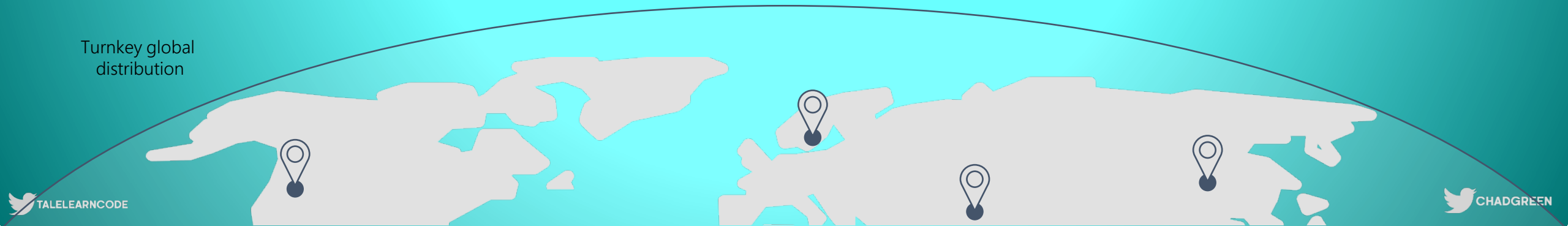
Strong

Bounded Staleness

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

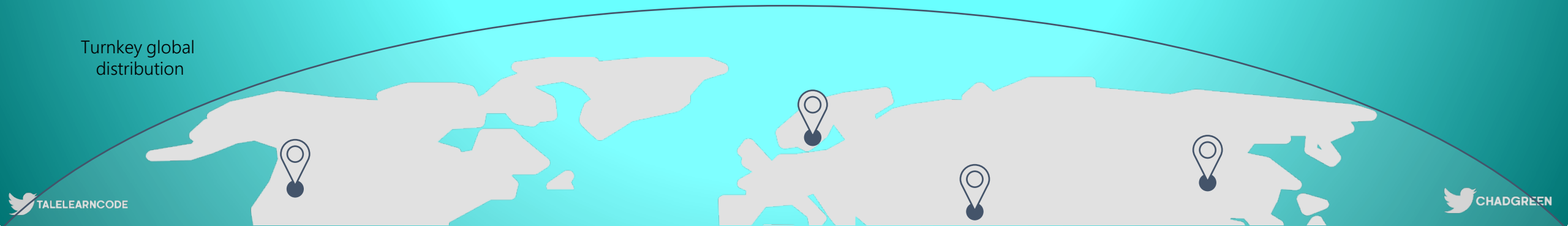
A globally distributed, massively scalable, multi-model database service

Strong **Session** **Bounded Staleness**

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

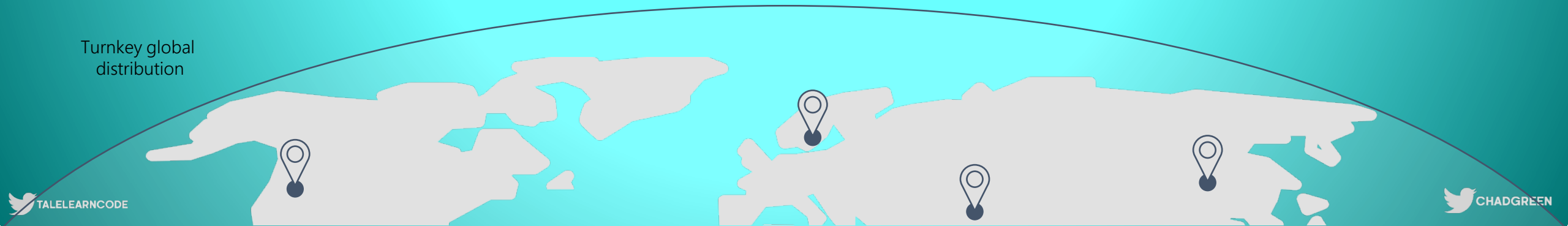
A globally distributed, massively scalable, multi-model database service

Strong **Session** **Bounded Staleness**

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Strong

Consistent Prefix

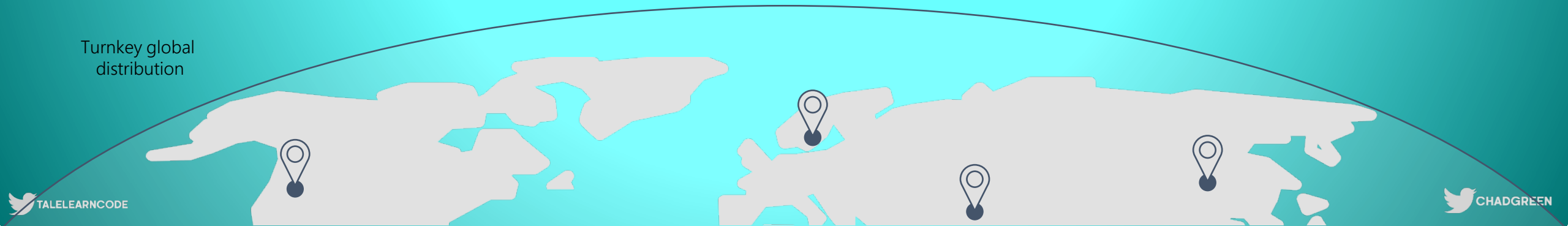
Bounded Staleness

Session

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Strong

Consistent Prefix

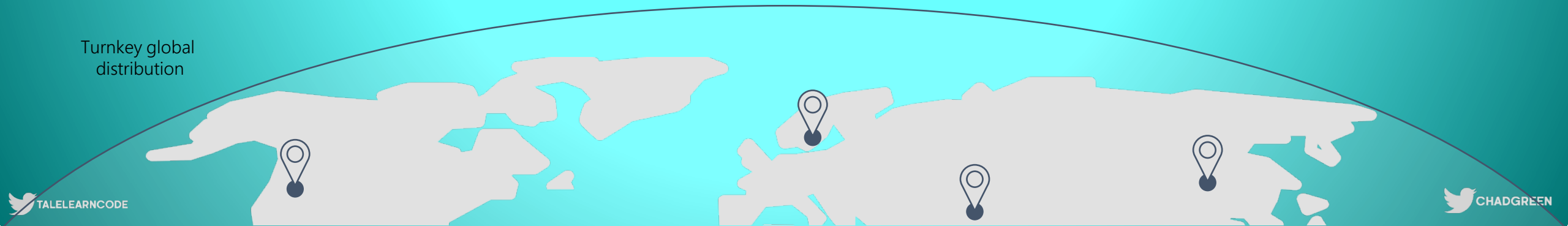
Bounded Staleness

Session

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Strong

Eventual

Bounded Staleness

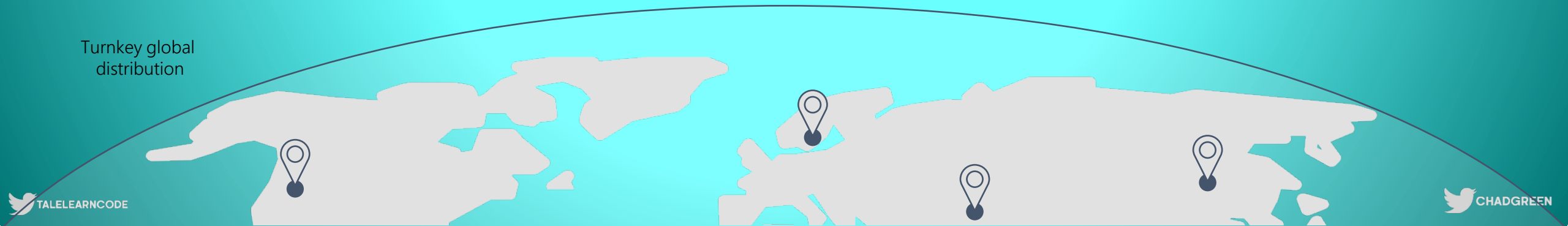
Session

Consistent Prefix

Guaranteed low latency
at the 99th percentile

Elastic scale out
of storage & throughput

Turnkey global
distribution



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

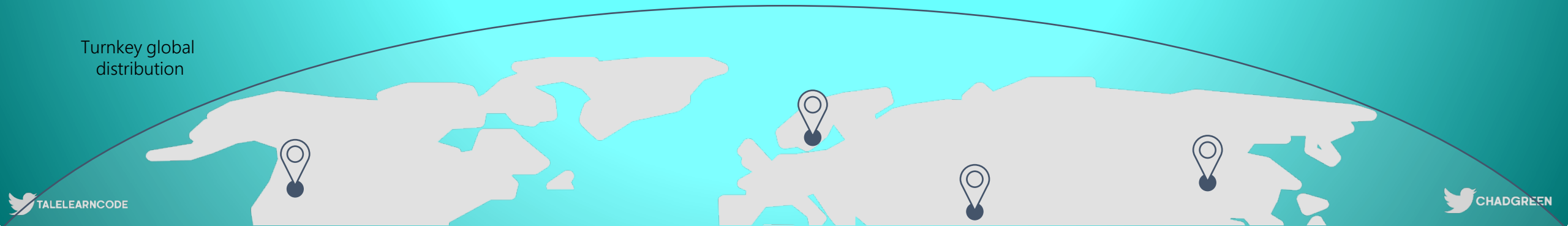
Comprehensive SLAs

Guaranteed low latency at the 99th percentile

Elastic scale out of storage & throughput

Five well-defined consistency models

Turnkey global distribution



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

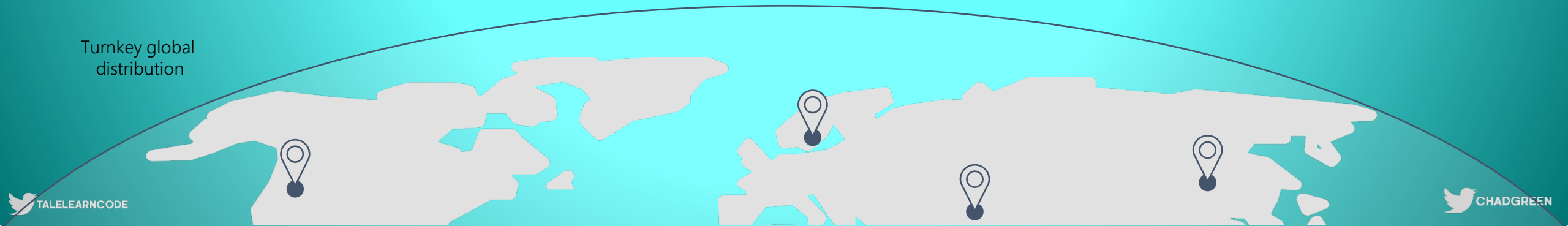
Comprehensive SLAs

Guaranteed low latency at the 99th percentile

Elastic scale out of storage & throughput

Five well-defined consistency models

Turnkey global distribution



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Key-value

Column-family

Document

Graph

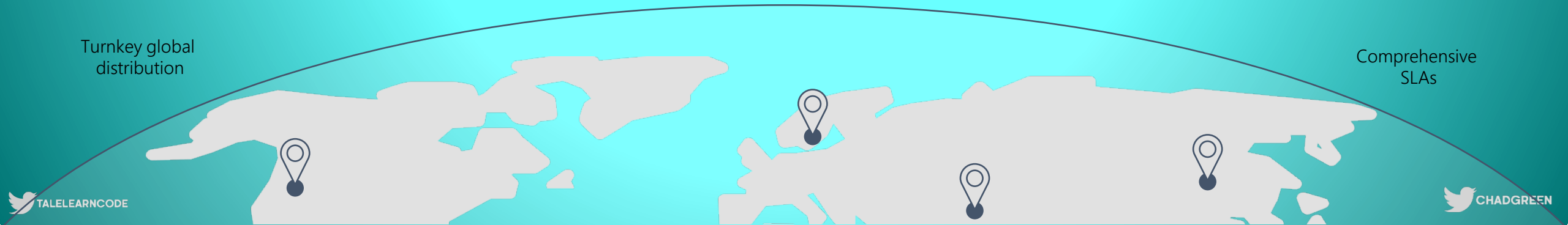
Guaranteed low latency at the 99th percentile

Elastic scale out of storage & throughput

Five well-defined consistency models

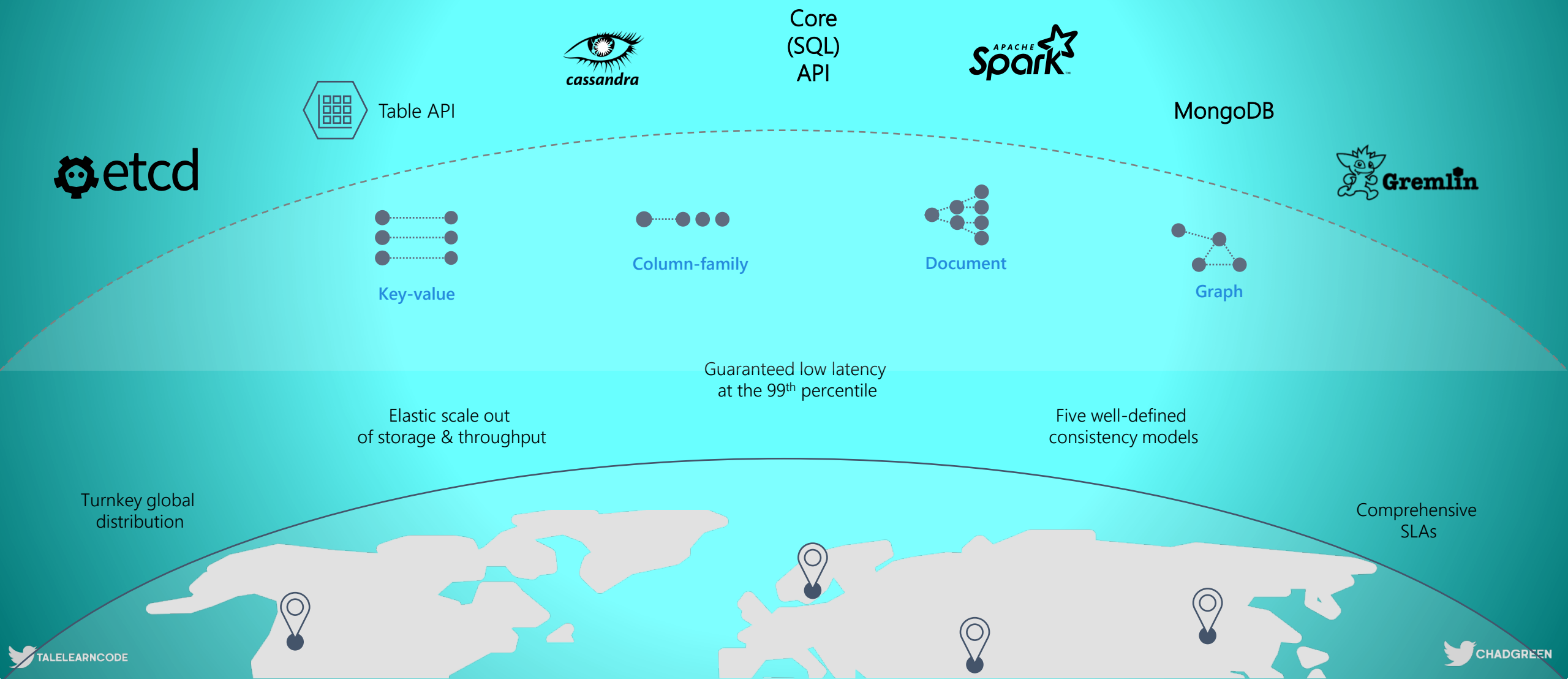
Turnkey global distribution

Comprehensive SLAs



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service



Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

What if we have REALLY large data requirements?

XBOX 360™

skype™

Office 365

Microsoft Azure

Azure Cosmos DB Capabilities

Resource	Default Limit
Maximum RUs per container	1,000,000
Maximum RUs per database	1,000,000
Maximum RUs per (logical) partition key	20-Gb
Maximum number of distinct (logical) partition keys	Unlimited
Maximum storage per container	Unlimited
Maximum storage per database	Unlimited
Maximum size of an item	2-Mb

Cosmos Use Cases

The Hitchhiker's Guide to the Cosmos

CHADGREEN

IoT and Telematics

Azure Cosmos DB Use Cases

Ingest Bursts of Data

Process and Analyze Streaming Data

Archive Data to Cold Storage

IoT and Telematics

Azure Cosmos DB Use Cases

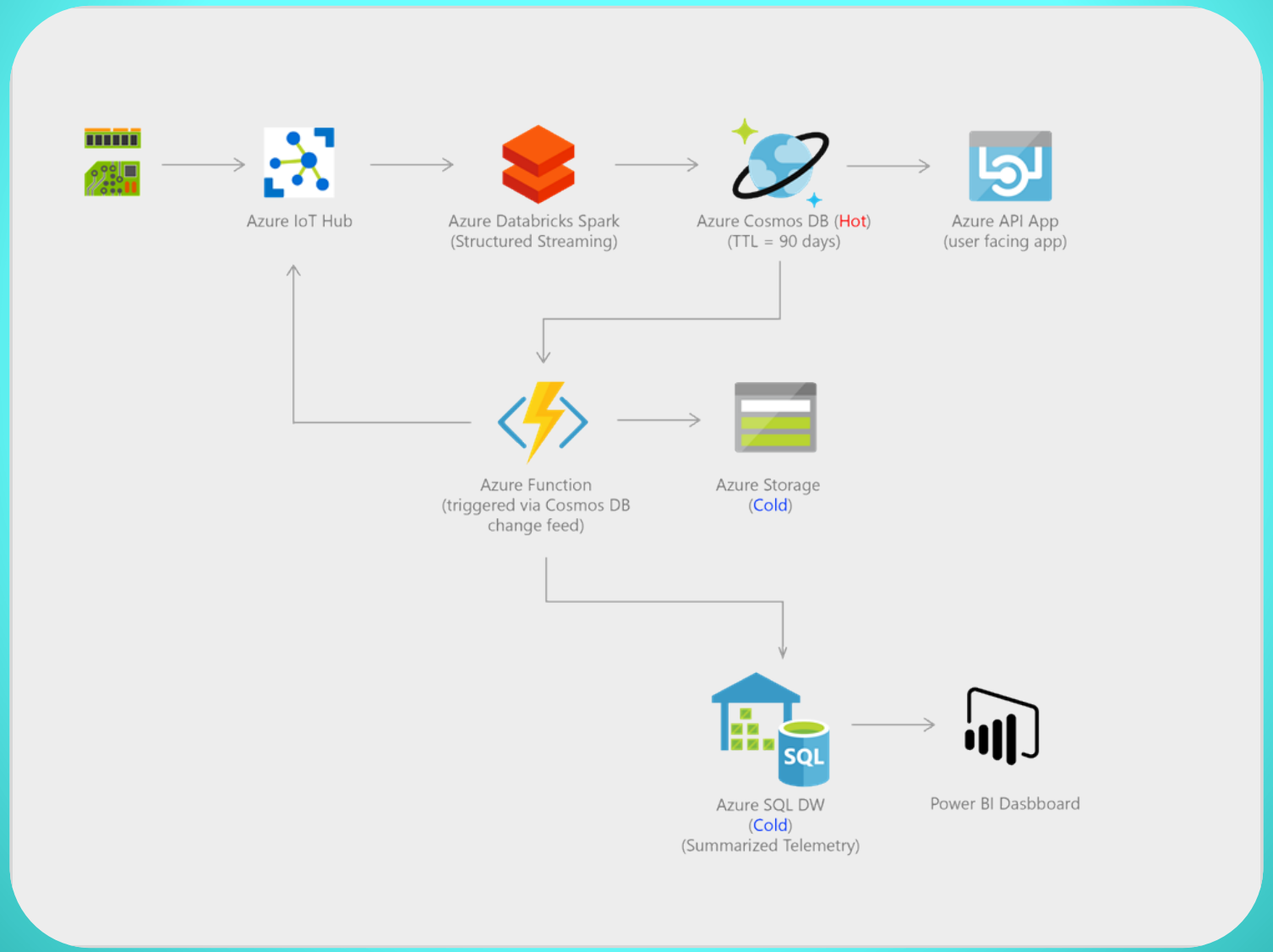
Ingest Bursts of Data

Process and Analyze Streaming Data

Archive Data to Cold Storage

IoT and Telematics

Azure Cosmos DB Use Cases



Retail Marketing

Azure Cosmos DB Use Cases

Storing and querying sets of attributes

User Accounts

Product Catalogs

IoT Device Registers

Retail Marketing

Azure Cosmos DB Use Cases

Storing and querying sets of attributes

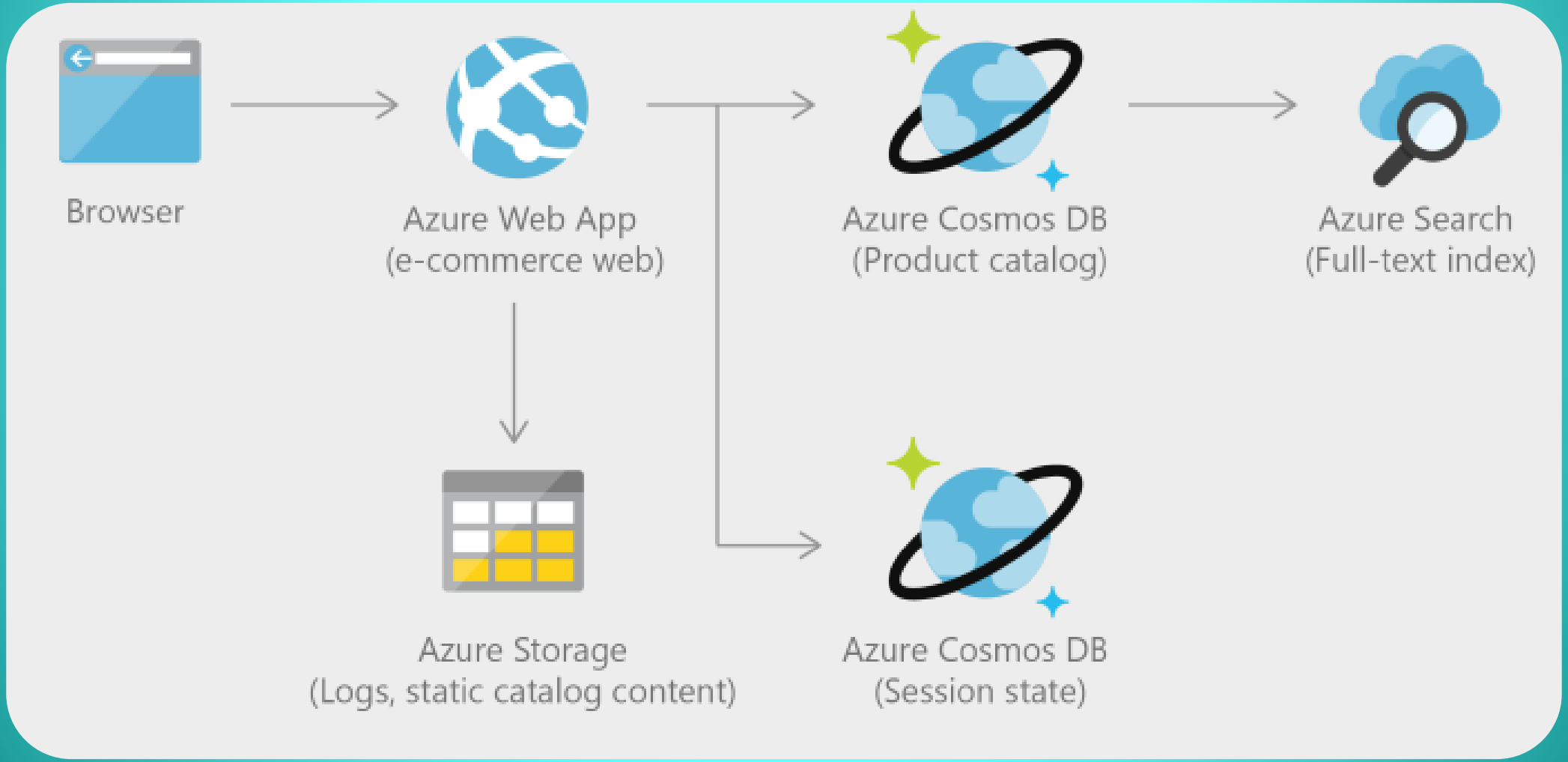
User Accounts

Product Catalogs

IoT Device Registers

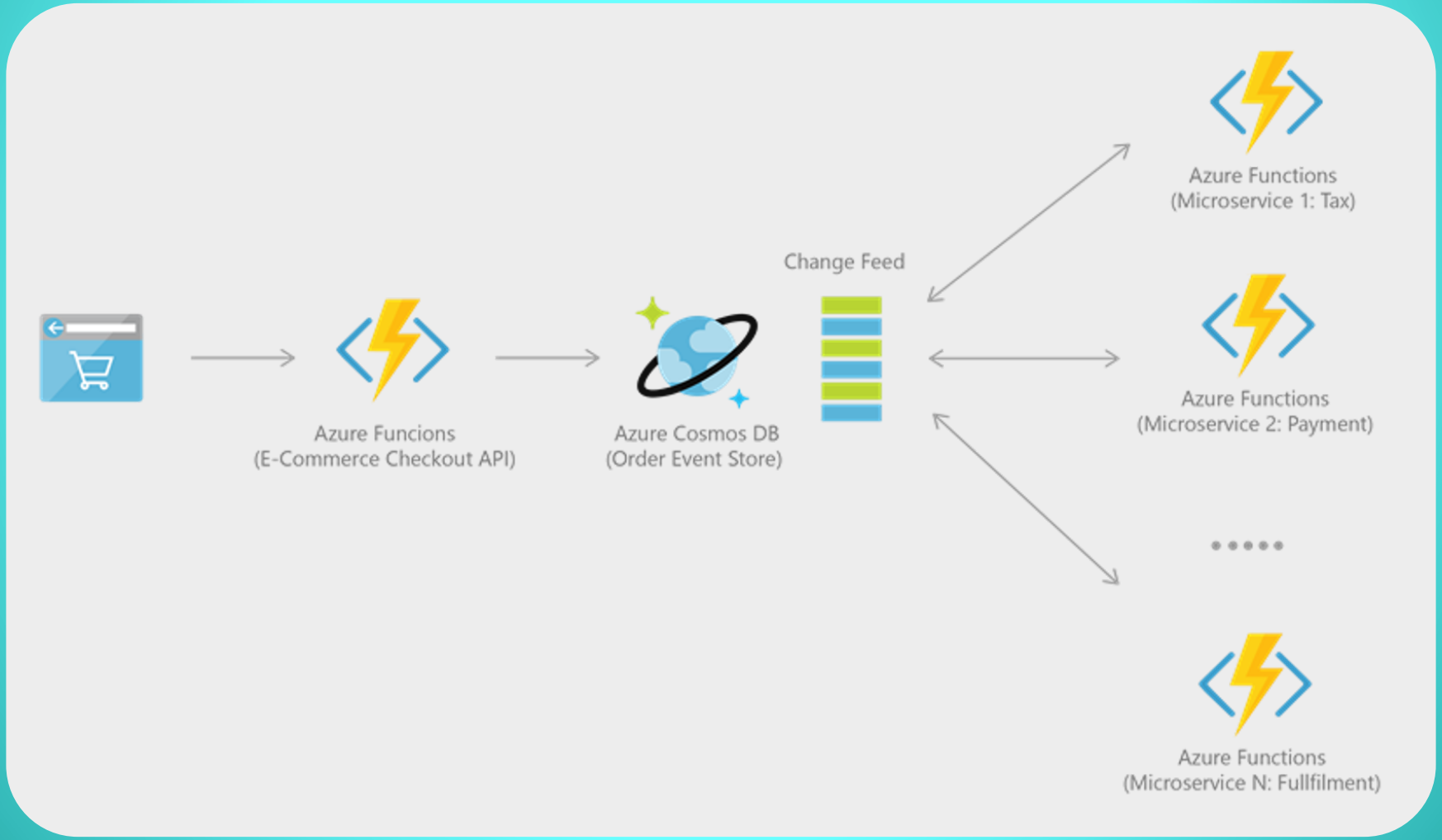
Retail and Marketing

Azure Cosmos DB Use Cases



Retail and Marketing

Azure Cosmos DB Use Cases



Gaming

Azure Cosmos DB Use Cases

Single-Millisecond Latencies

Handle Massive Spikes

Gaming

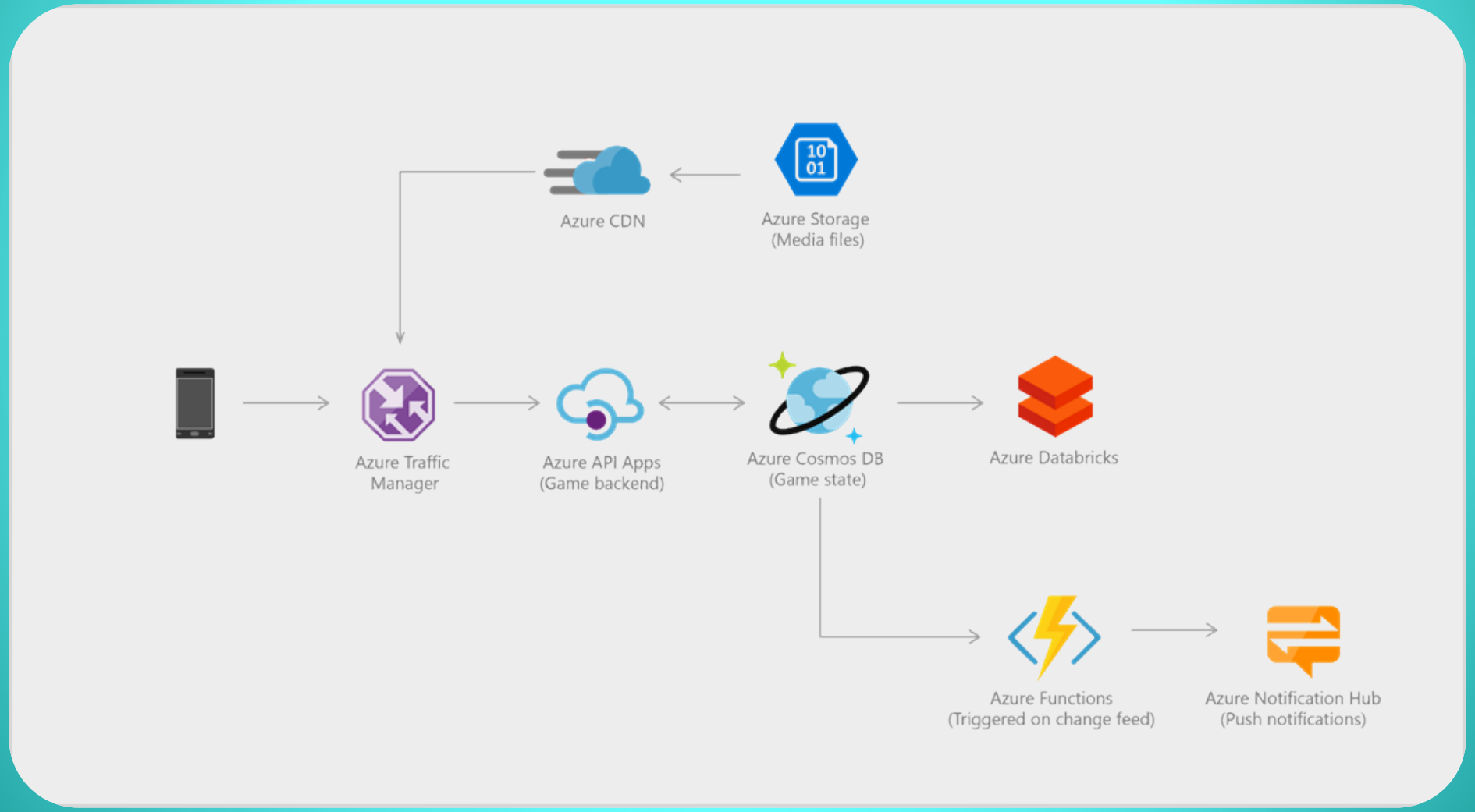
Azure Cosmos DB Use Cases

Single-Millisecond Latencies

Handle Massive Spikes

Gaming

Azure Cosmos DB Use Cases



Web & Mobile Applications

Azure Cosmos DB Use Cases

- Modeling Social Interactions
- Integrating with Third-Party Services
- Building Rich Personalized Experiences

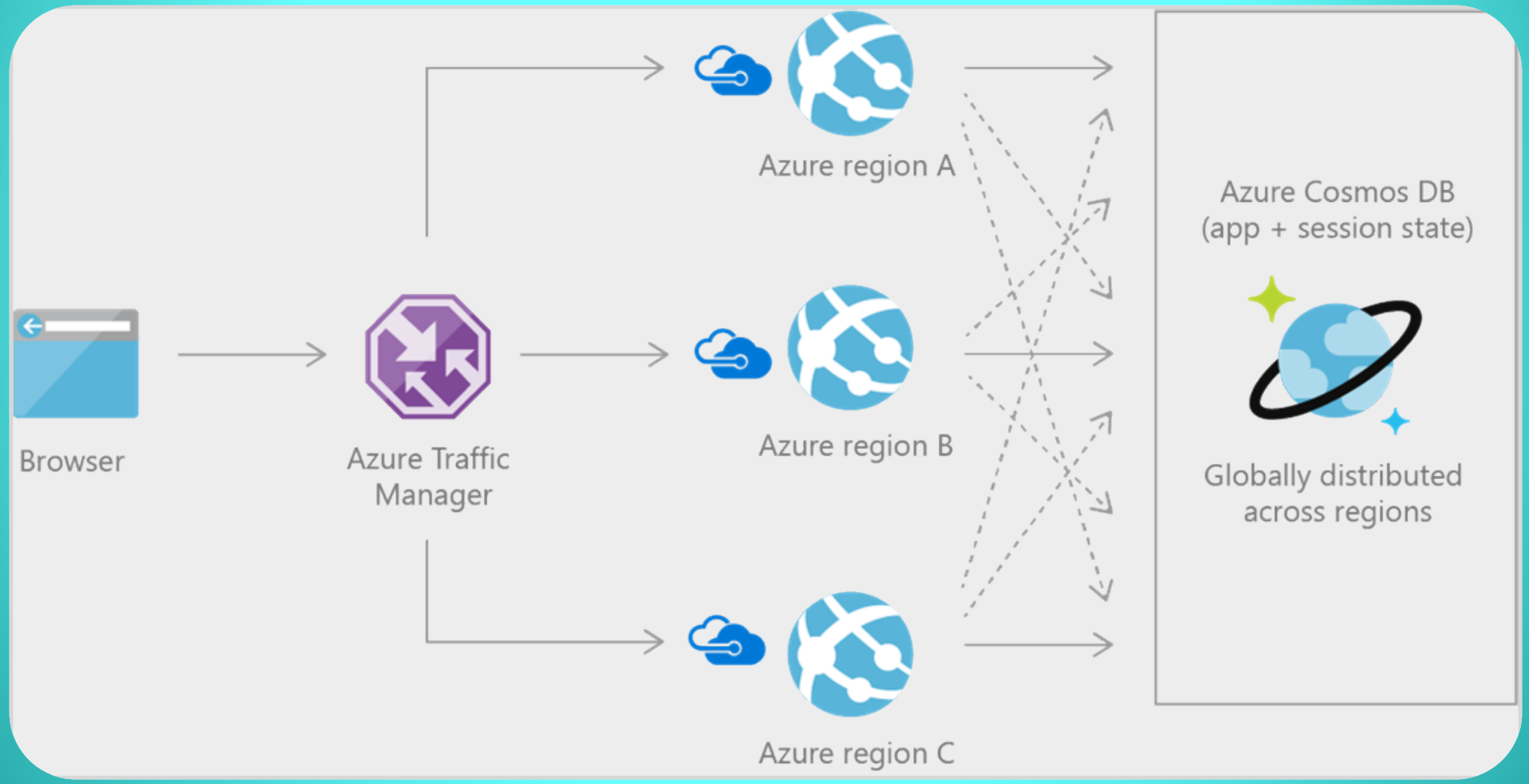
Web & Mobile Applications

Azure Cosmos DB Use Cases

- Modeling Social Interactions
- Integrating with Third-Party Services
- Building Rich Personalized Experiences

Gaming

Azure Cosmos DB Use Cases



Integrations

The Hitchhiker's Guide to the Cosmos

CHADGREEN

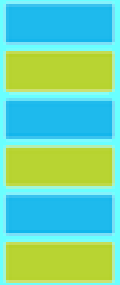
Cosmos DB Integrations

Cosmos DB Integrations

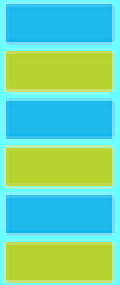
Cosmos DB Integrations

Cosmos DB Integrations

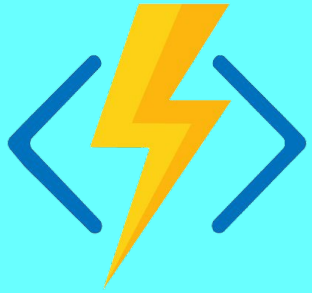
Cosmos DB Integrations

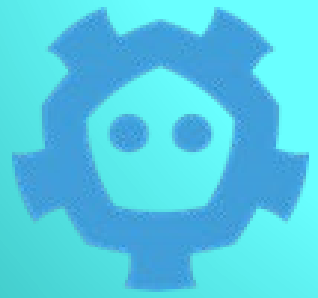


Cosmos DB Integrations

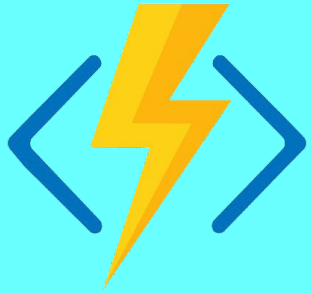


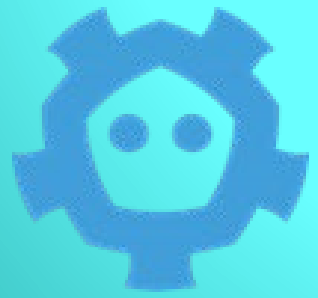
Cosmos DB Integrations



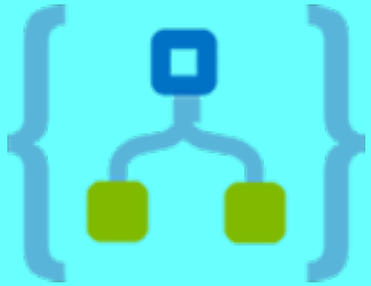


Cosmos DB Integrations

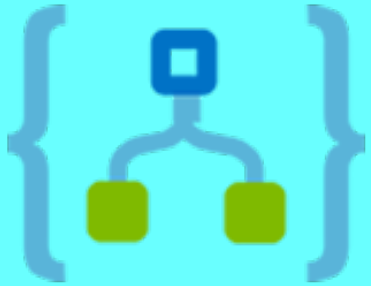




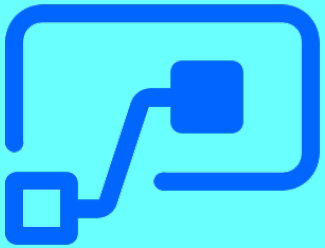
Cosmos DB Integrations

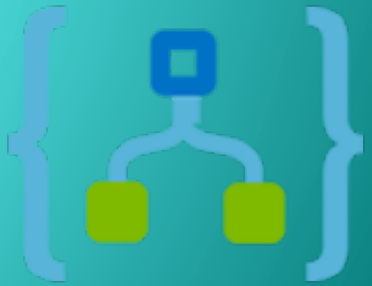


Cosmos DB Integrations



Cosmos DB Integrations



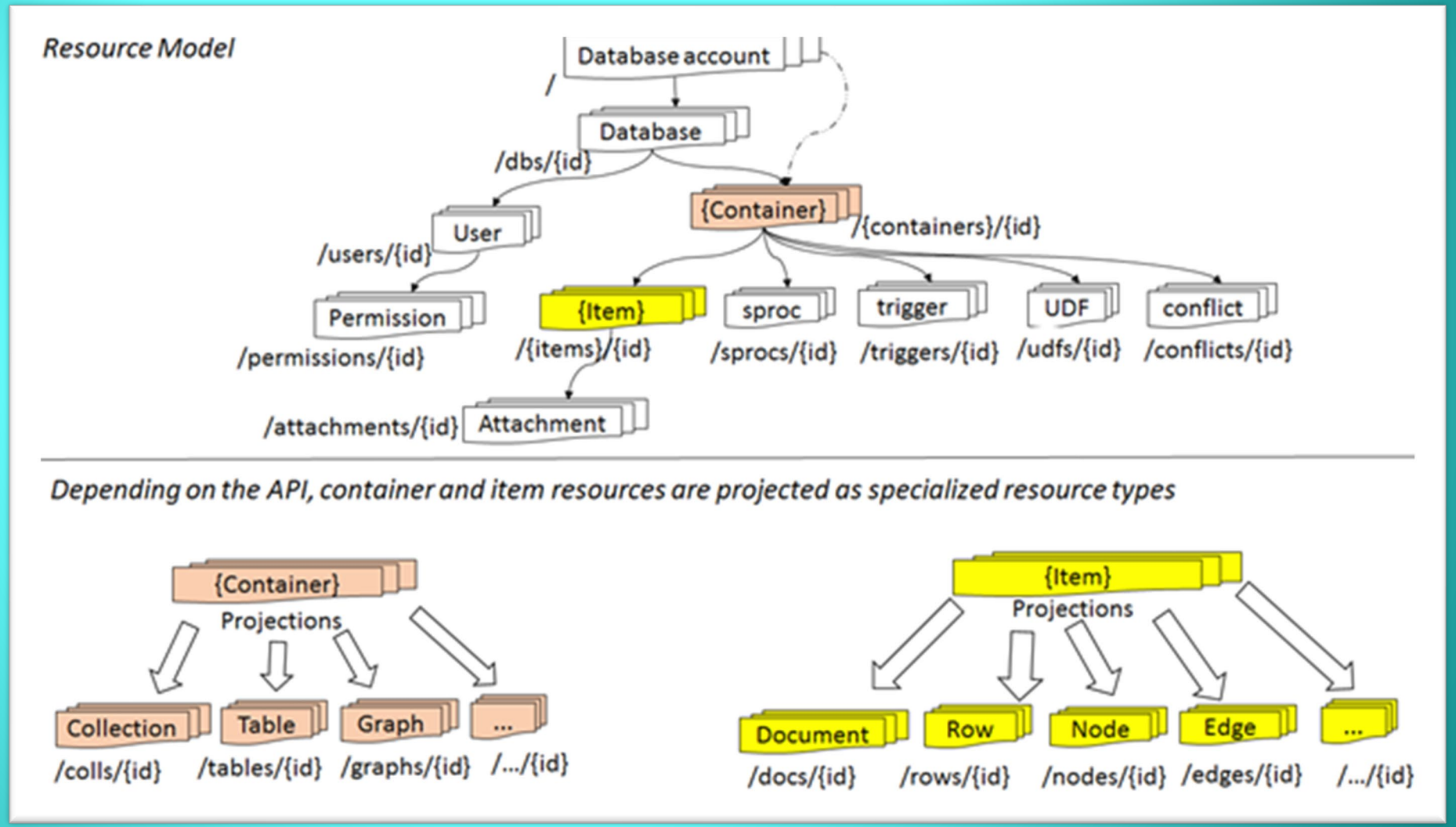


Wide-Range of Data APIs

The Hitchhiker's Guide to the Cosmos

Atom Resource Sequence

Wide-Range of Data APIs



SQL API

Document Database

SQL API – What

Wide-Range of Data APIs

Document Database

SQL to Query JSON Documents

JavaScript Programming Model

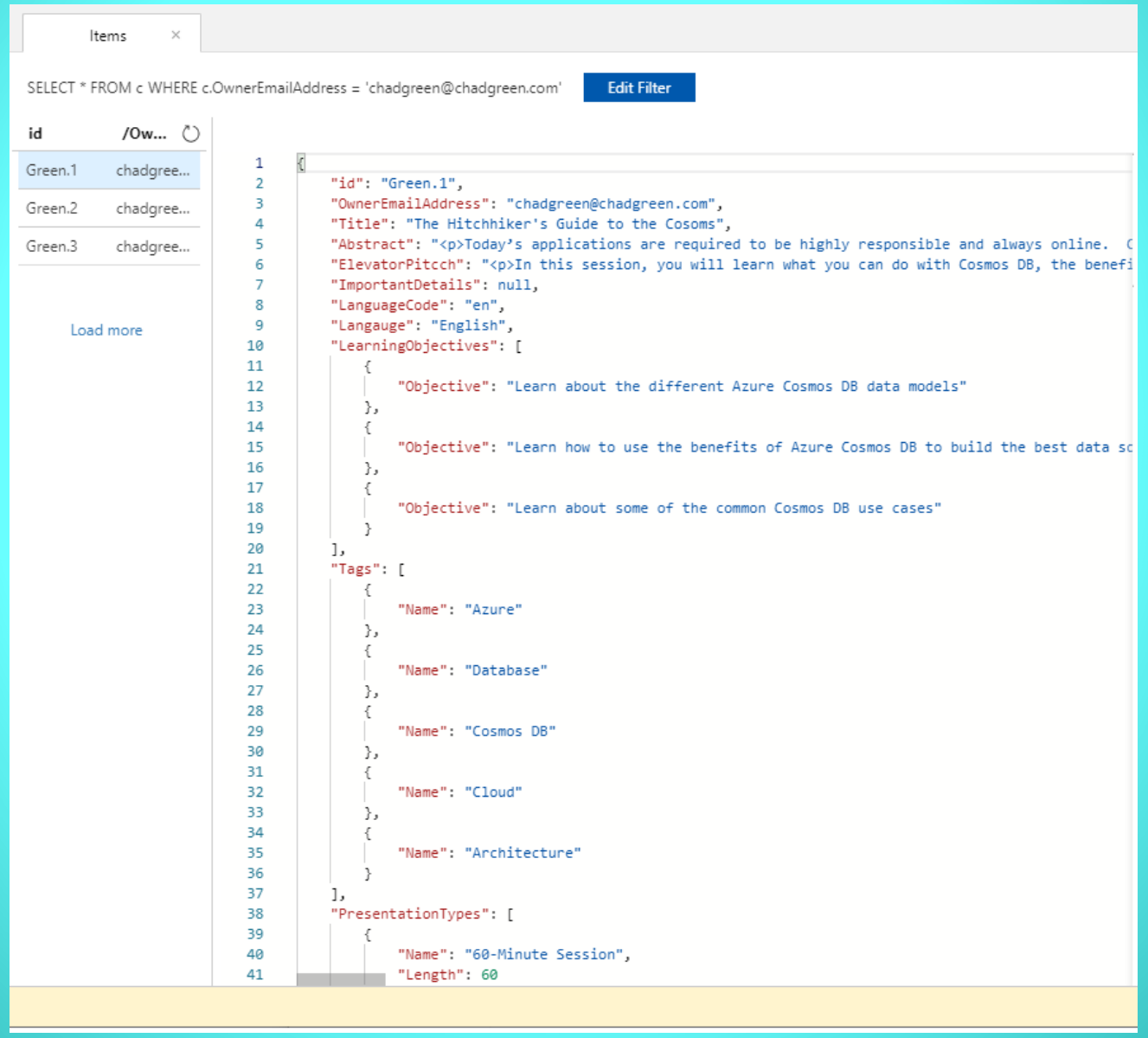
SQL API – When

Wide-Range of Data APIs

Building a new non-relational document database and want to query using SQL

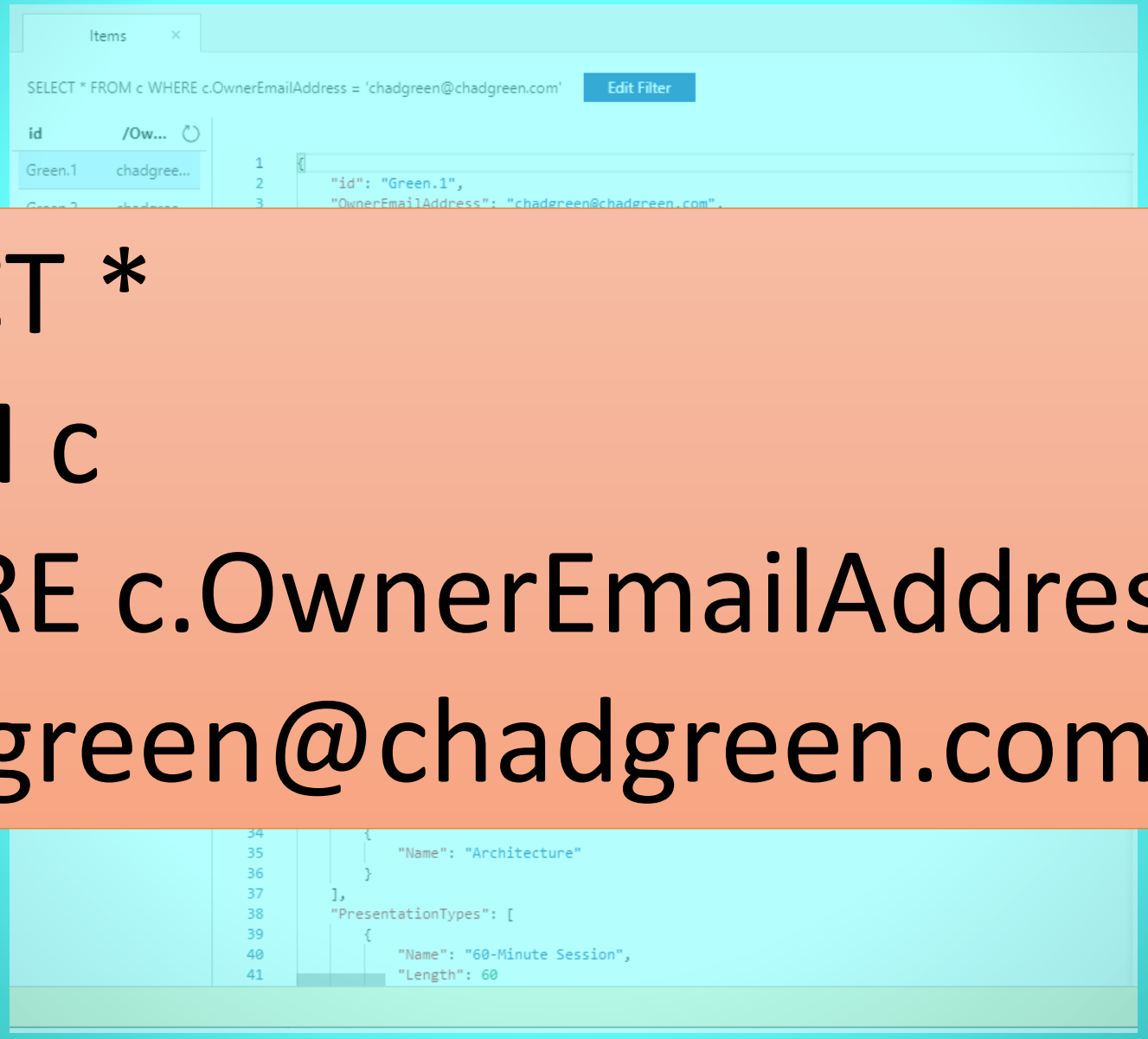
SQL API – How: Data Model

Wide-Range of Data APIs



SQL API – How: Data Model

Wide-Range of Data APIs




```
SELECT *
FROM c
WHERE c.OwnerEmailAddress =
'chadgreen@chadgreen.com'
```

SQL API – How: Insert

Wide-Range of Data APIs

```
public class Presentation
{
  [JsonProperty(PropertyName = "id")]
  public string Id { get; set; }

  public string OwnerEmailAddress { get; set; }
  public string Title { get; set; }
  public string Abstract { get; set; }
  public string ElevatorPitcch { get; set; }
  public string ImportantDetails { get; set; }
  public string LanguageCode { get; set; }
  public string Langauge { get; set; }
  public LearningObjective[] LearningObjectives { get; set; }
  public Tag[] Tags { get; set; }
  public PresentationType[] PresentationTypes { get; set; }
}
```

SQL API – How: Insert

Wide-Range of Data APIs

```
public class Presentation
{
  [JsonProperty(PropertyName = "id")]
  public string Id { get; set; }

  public string OwnerEmailAddress { get; set; }
  public string Title { get; set; }
  public string Abstract { get; set; }
  public string ElevatorPitcch { get; set; }
  public string ImportantDetails { get; set; }
  public string LanguageCode { get; set; }
  public string Langauge { get; set; }
  public LearningObjective[] LearningObjectives { get; set; }
  public Tag[] Tags { get; set; }
  public PresentationType[] PresentationTypes { get; set; }
}
```

SQL API – How: Insert

Wide-Range of Data APIs

```
private async Task AddPresentation(ExistingPresentation existingPresentation)
{
  Presentation presentation = Generate.Presentation(existingPresentation);
  try
  {
    ItemResponse<Presentation> presentationResponse =
        await this.container.ReadItemAsync<Presentation>(presentation.Id,
        new PartitionKey(presentation.OwnerEmailAddress));
    Console.WriteLine("Item in database with id: {0} already exists\n",
        presentationResponse.Resource.Id);
  }
  catch
  (
    CosmosException ex) when (ex.StatusCode == HttpStatusCode.NotFound)
    {
      ItemResponse<Presentation> presentationResponse =
          await this.container.CreateItemAsync<Presentation>(presentation,
          new PartitionKey(presentation.OwnerEmailAddress));
      Console.WriteLine("Created item with id: {0} Using {1} RUs.\n",
      presentationResponse.Resource.Id, presentationResponse.RequestCharge);
    }
  }
}
```

SQL API – How: Insert

Wide-Range of Data APIs

```
private async Task AddPresentation(ExistingPresentation existingPresentation)
{
  Presentation presentation = Generate.Presentation(existingPresentation);
   try
   {
     ItemResponse<Presentation> presentationResponse =
         await this.container.ReadItemAsync<Presentation>(presentation.Id,
         new PartitionKey(presentation.OwnerEmailAddress));
     Console.WriteLine("Item in database with id: {0} already exists\n",
         presentationResponse.Resource.Id);
   }
  catch
  (
    CosmosException ex) when (ex.StatusCode == HttpStatusCode.NotFound)
    {
      ItemResponse<Presentation> presentationResponse =
          await this.container.CreateItemAsync<Presentation>(presentation,
          new PartitionKey(presentation.OwnerEmailAddress));
      Console.WriteLine("Created item with id: {0} Using {1} RUs.\n",
      presentationResponse.Resource.Id, presentationResponse.RequestCharge);
    }
  }
}
```

SQL API – How: Insert

Wide-Range of Data APIs

```
private async Task AddPresentation(ExistingPresentation existingPresentation)
{
  Presentation presentation = Generate.Presentation(existingPresentation);
  try
  {
    ItemResponse<Presentation> presentationResponse =
        await this.container.ReadItemAsync<Presentation>(presentation.Id,
        new PartitionKey(presentation.OwnerEmailAddress));
    Console.WriteLine("Item in database with id: {0} already exists\n",
        presentationResponse.Resource.Id);
  }
   catch
   (
     CosmosException ex) when (ex.StatusCode == HttpStatusCode.NotFound)
     {
       ItemResponse<Presentation> presentationResponse =
           await this.container.CreateItemAsync<Presentation>(presentation,
           new PartitionKey(presentation.OwnerEmailAddress));
       Console.WriteLine("Created item with id: {0} Using {1} RUs.\n",
       presentationResponse.Resource.Id, presentationResponse.RequestCharge);
     }
  }
}
```

SQL API – How: Query

Wide-Range of Data APIs

```
private async Task QueryPresentationsAsync()
{
  var sqlQueryText = "SELECT * FROM c WHERE c.OwnerEmailAddress = 'chadgreen@chadgreen.com'";
  Console.WriteLine("Running query: {0}\n", sqlQueryText);

  QueryDefinition queryDefinition = new QueryDefinition(sqlQueryText);
  FeedIterator<Presentation> queryResultSetIterator
      = this.container.GetItemQueryIterator<Presentation>(queryDefinition);

  List<Presentation> presentations = new List<Presentation>();
  while (queryResultSetIterator.HasMoreResults)
  {
    FeedResponse<Presentation> currentResultSet = await queryResultSetIterator.ReadNextAsync();
    foreach (Presentation presentation in currentResultSet)
    {
      presentations.Add(presentation);
      Console.WriteLine($"\t{presentation.Title}");
    }
  }
}
```

SQL API – How: Query

Wide-Range of Data APIs

```
private async Task QueryPresentationsAsync()
{
  var sqlQueryText = "SELECT * FROM c WHERE c.OwnerEmailAddress = 'chadgreen@chadgreen.com'";
  Console.WriteLine("Running query: {0}\n", sqlQueryText);

  QueryDefinition queryDefinition = new QueryDefinition(sqlQueryText);
  FeedIterator<Presentation> queryResultSetIterator
      = this.container.GetItemQueryIterator<Presentation>(queryDefinition);

  List<Presentation> presentations = new List<Presentation>();
  while (queryResultSetIterator.HasMoreResults)
  {
    FeedResponse<Presentation> currentResultSet = await queryResultSetIterator.ReadNextAsync();
    foreach (Presentation presentation in currentResultSet)
    {
      presentations.Add(presentation);
      Console.WriteLine($"\t{presentation.Title}");
    }
  }
}
```

SQL API – How: Query

Wide-Range of Data APIs

```
private async Task QueryPresentationsAsync()
{
  var sqlQueryText = "SELECT * FROM c WHERE c.OwnerEmailAddress = 'chadgreen@chadgreen.com'";
  Console.WriteLine("Running query: {0}\n", sqlQueryText);

  QueryDefinition queryDefinition = new QueryDefinition(sqlQueryText);
  FeedIterator<Presentation> queryResultSetIterator
      = this.container.GetItemQueryIterator<Presentation>(queryDefinition);

  List<Presentation> presentations = new List<Presentation>();
  while (queryResultSetIterator.HasMoreResults)
  {
    FeedResponse<Presentation> currentResultSet = await queryResultSetIterator.ReadNextAsync();
    foreach (Presentation presentation in currentResultSet)
    {
      presentations.Add(presentation);
      Console.WriteLine($"\t{presentation.Title}");
    }
  }
}
```

SQL API – How: Query

Wide-Range of Data APIs

```
private async Task QueryPresentationsAsync()
{
  var sqlQueryText = "SELECT * FROM c WHERE c.OwnerEmailAddress = 'chadgreen@chadgreen.com'";
  Console.WriteLine("Running query: {0}\n", sqlQueryText);

  QueryDefinition queryDefinition = new QueryDefinition(sqlQueryText);
  FeedIterator<Presentation> queryResultSetIterator
      = this.container.GetItemQueryIterator<Presentation>(queryDefinition);

  List<Presentation> presentations = new List<Presentation>();
  while (queryResultSetIterator.HasMoreResults)
  {
    FeedResponse<Presentation> currentResultSet = await queryResultSetIterator.ReadNextAsync();
    foreach (Presentation presentation in currentResultSet)
    {
      presentations.Add(presentation);
      Console.WriteLine($"\t{presentation.Title}");
    }
  }
}
```

SQL API – Querying Options

Wide-Range of Data APIs

API

LINQ to SQL

JavaScript

Entity Framework

MongoDB

Document Database

{ LEAF }

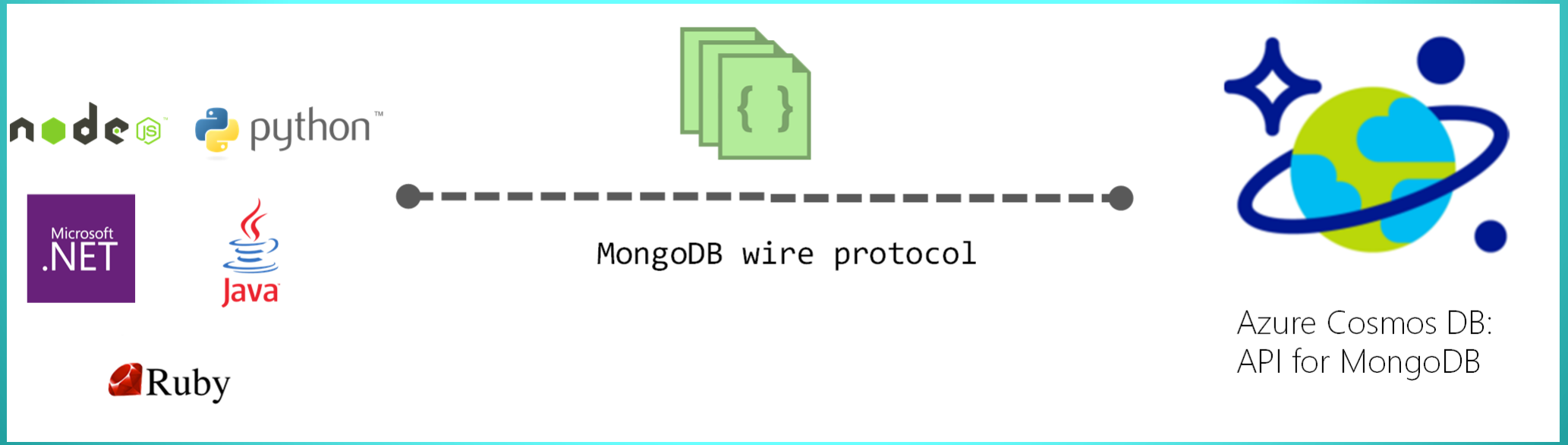
MongoDB – What

Wide-Range of Data APIs

Native Implementation

Interact Transparently

V3.6 Compatibility



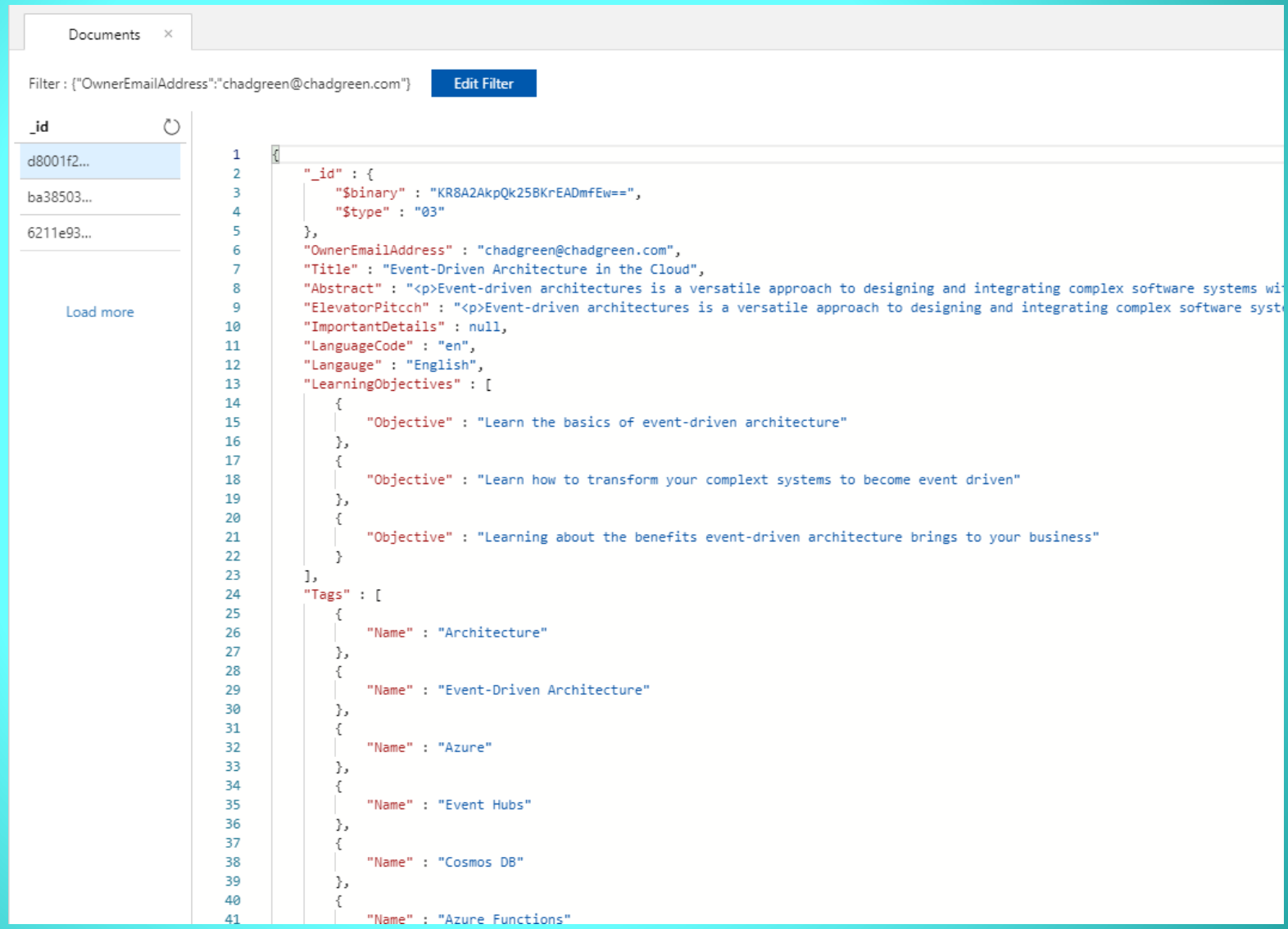
MongoDB – When

Wide-Range of Data APIs

Migrating data from a MongoDB database to Azure Cosmos DB's fully managed service

MongoDB – How: Data Model

Wide-Range of Data APIs



MongoDB – How: Data Model

Wide-Range of Data APIs

Filter: {"OwnerEmailAddress": "chadgreen@chadgreen.com"}

MongoDB – How: Insert

Wide-Range of Data APIs

```
public class Presentation
{
  [BsonId(IdGenerator =typeof(CombGuidGenerator))] public Guid Id { get; set; }

  [BsonElement("OwnerEmailAddress")] public string OwnerEmailAddress { get; set; }

  [BsonElement("Title")] public string Title { get; set; }

  [BsonElement("Abstract")] public string Abstract { get; set; }

  [BsonElement("ElevatorPitcch")] public string ElevatorPitcch { get; set; }

  [BsonElement("ImportantDetails")] public string ImportantDetails { get; set; }

  [BsonElement("LanguageCode")] public string LanguageCode { get; set; }

  [BsonElement("Langauge")] public string Langauge { get; set; }

  [BsonElement("LearningObjectives")] public LearningObjective[] LearningObjectives { get; set; }

  [BsonElement("Tags")] public Tag[] Tags { get; set; }

  [BsonElement("PresentationTypes")] public PresentationType[] PresentationTypes { get; set; }
}
```

MongoDB – How: Insert

Wide-Range of Data APIs

```
public void CreatePresentation(Presentation presentation)
{
  var collection = GetPresentationsCollection();
  try
  {
    collection.InsertOne(presentation);
  }
  catch (MongoCommandException ex)
  {
    string msg = ex.Message;
  }
}
```

MongoDB – How: Insert

Wide-Range of Data APIs

```
private IMongoCollection<Presentation> GetPresentationsCollection()
{
  MongoClientSettings settings = new MongoClientSettings();
  settings.Server = new MongoServerAddress(host, 10255);
  settings.UseTls = true;
  settings.SslSettings = new SslSettings();
  settings.SslSettings.EnabledSslProtocols = SslProtocols.Tls12;

  MongoIdentity identity = new MongoInternalIdentity(dbName, userName);
  MongoIdentityEvidence evidence = new PasswordEvidence(password);

  settings.Credential = new MongoCredential("SCRAM-SHA-1", identity, evidence);

  MongoClient client = new MongoClient(settings);
  var database = client.GetDatabase(dbName);
  var presetationCollection = database.GetCollection<Presentation>(collectionName);
  return presetationCollection;
}
```

MongoDB – How: Query

Wide-Range of Data APIs

```
public List<Presentation> GetAllPresentations()
{
  try { var collection = GetPresentationsCollection();
  return collection.Find(new BsonDocument()).ToList();
}
  catch (MongoConnectionException)
  {
    return new List<Presentation>();
  }
}
```

MongoDB – Cosmos or Atlas

Wide-Range of Data APIs

Cosmos DB

- Small documents – the smaller the cheaper
- Read more often than write
- Like idea to start small and pay-as-you-go
- Support included in Azure subscription
- Need a guaranteed latency despite usage

MongoDB Atlas

- Any size documents (only choice for documents over 2-Mb)
- Like to have a fixed budget for storage
- Use Mongo-API features that are not covered by Cosmos DB
- Freedom to create unique indexes
- Write data more often than read
- Want to decide backup policy

https://blog.chudinov.net/azure-cosmosdb-vs-mongodb-atlas/

Graph API

Graph Database

Graph API – What

Wide-Range of Data APIs

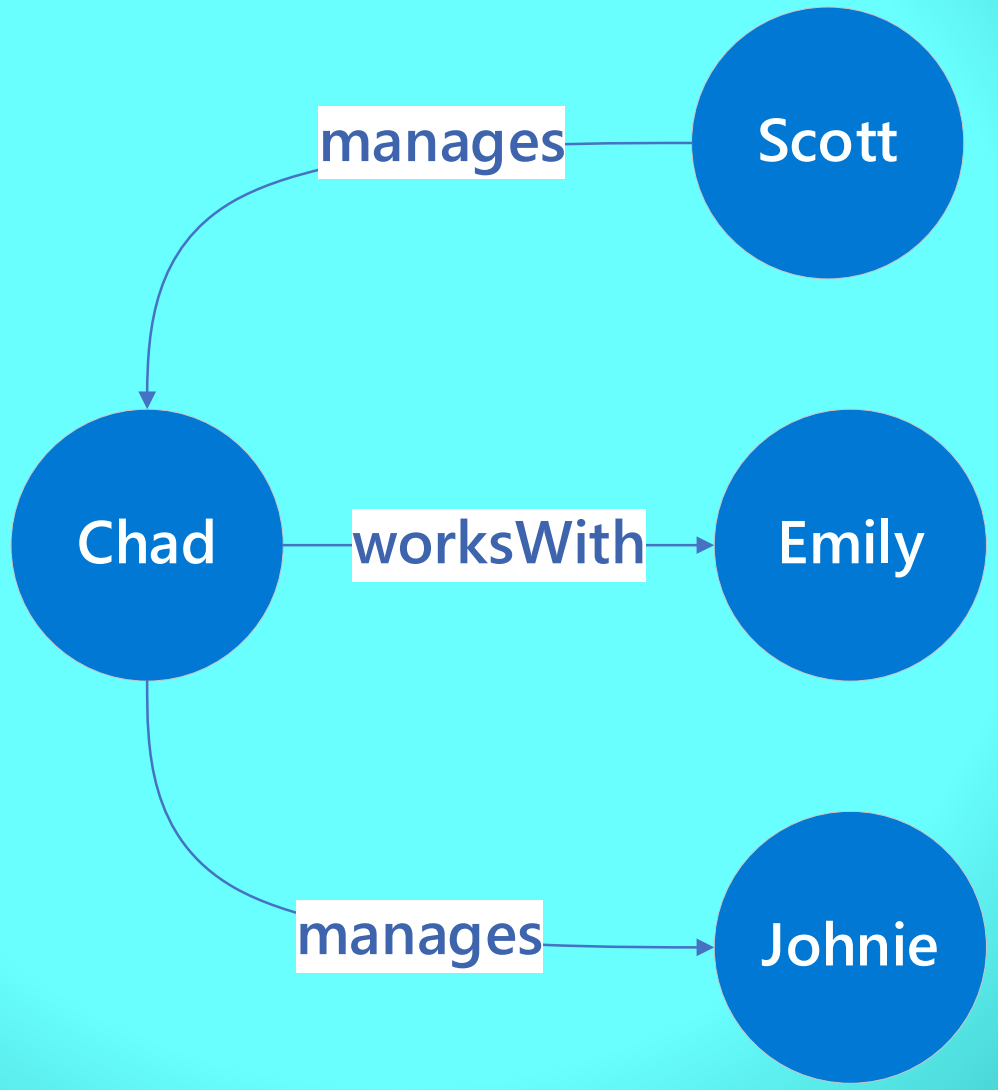
Vertices and Edges

Natural Data Representation

Model All Kinds of Scenarios

What is a Graph

Wide-Range of Data APIs



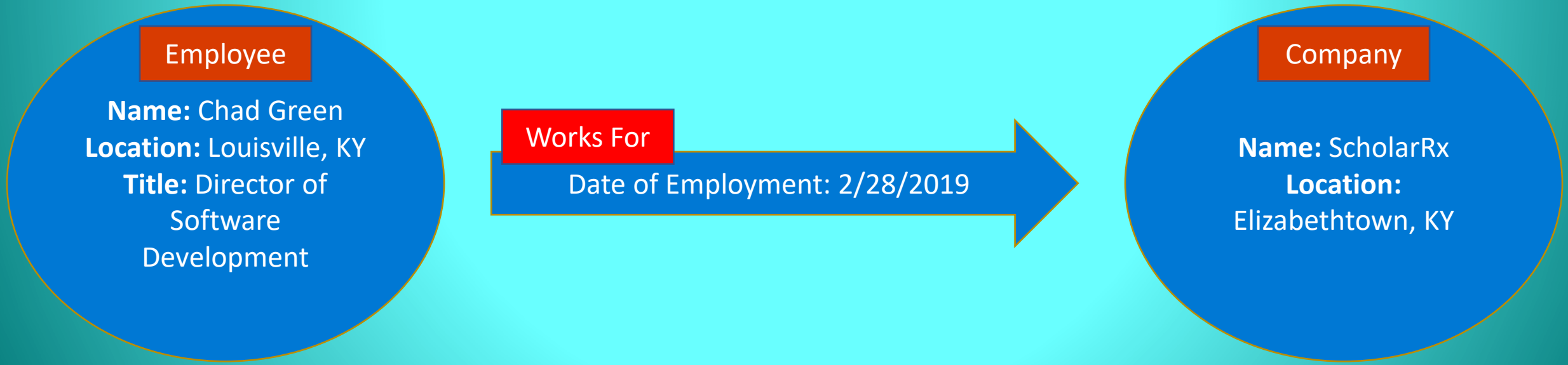
Property Graph Model

Wide-Range of Data APIs

Contains **nodes** (vertices) and **relationships** (edges)

Nodes and relationships contain **properties**

Relationships are **named** and **directed** with a **start** and **end** node



The Power of Graph Databases

Wide-Range of Data APIs

Performance

The Power of Graph Databases

Wide-Range of Data APIs

Performance

Flexibility

The Power of Graph Databases

Wide-Range of Data APIs

Performance

Flexibility

Agility

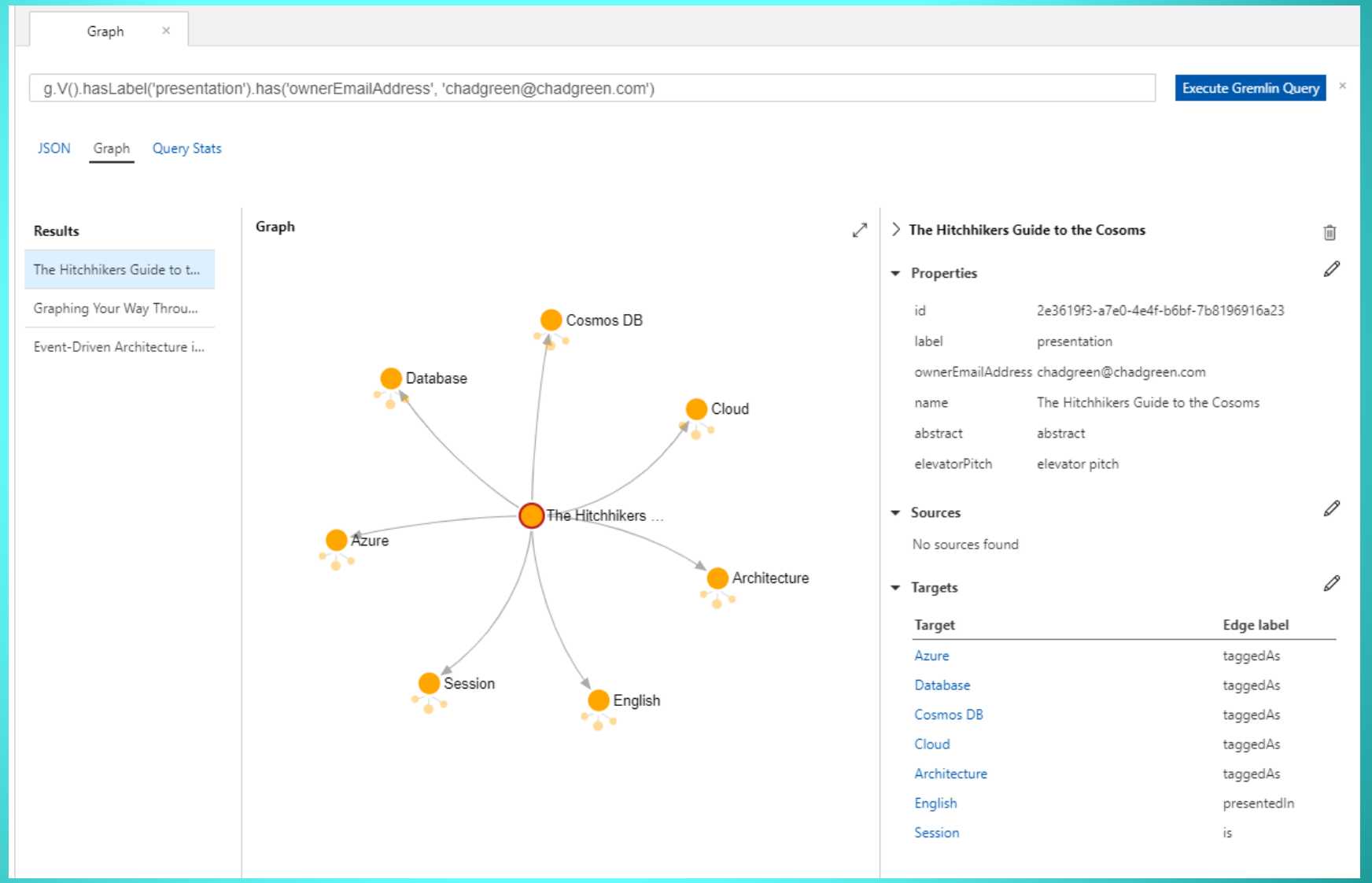
Graph API – When

Wide-Range of Data APIs

Building a graph database to model and traverse relationships among entities

Graph API – How: Data Model

Wide-Range of Data APIs



Graph API – How: Data Model

Wide-Range of Data APIs


```
g.V().hasLabel
('presentation')
.has('ownerEmailAddress',
'chadgreen@chadgreen.com')
```

Graph API – How: Insert

Wide-Range of Data APIs

```
g.addV('presentation')
.property('ownerEmailAddress', 'chadgreen@chadgreen.com')
.property('name', 'The Hitchhikers Guide to the Cosmos')
.property('abstract', 'This is a really cool talk!')
.property('elevatorPitch', 'Cool talk!')
```

Graph API – How: Insert

Wide-Range of Data APIs

```
private static Task<ResultSet<dynamic>> SubmitRequest(GremlinClient gremlinClient,
    KeyValuePair<string, string> query)
{
  try
  {
    return gremlinClient.SubmitAsync<dynamic>(query.Value);
  }
  catch (ResponseException e)
  {
    Console.WriteLine("\tRequest Error!");
    Console.WriteLine($"\tStatusCode: {e.StatusCode}");
    PrintStatusAttributes(e.StatusAttributes);
    Console.WriteLine($"\t[\"x-ms-retry-after-ms\"] :
        { GetValueAsString(e.StatusAttributes, "x-ms-retry-after-ms")}");
    Console.WriteLine($"\t[\"x-ms-activity-id\"] :
        { GetValueAsString(e.StatusAttributes, "x-ms-activity-id")}");

    throw;
  }
}
```

Graph API – How: Insert

Wide-Range of Data APIs

```
private static Task<ResultSet<dynamic>> SubmitRequest(GremlinClient gremlinClient,
    KeyValuePair<string, string> query)
{
  try
  {
     return gremlinClient.SubmitAsync<dynamic>(query.Value);
  }
  catch (ResponseException e)
  {
    Console.WriteLine("\tRequest Error!");
    Console.WriteLine($"\tStatusCode: {e.StatusCode}");
    PrintStatusAttributes(e.StatusAttributes);
    Console.WriteLine($"\t[\"x-ms-retry-after-ms\"] :
        { GetValueAsString(e.StatusAttributes, "x-ms-retry-after-ms")}");
    Console.WriteLine($"\t[\"x-ms-activity-id\"] :
        { GetValueAsString(e.StatusAttributes, "x-ms-activity-id")}");

    throw;
  }
}
```

Graph API – How: Insert

Wide-Range of Data APIs

```
private static Task<ResultSet<dynamic>> SubmitRequest(GremlinClient gremlinClient,
    KeyValuePair<string, string> query)
{
  try
  {
    return gremlinClient.SubmitAsync<dynamic>(query.Value);
  }
  catch (ResponseException e)
  {
    Console.WriteLine("\tRequest Error!");
    Console.WriteLine($"\tStatusCode: {e.StatusCode}");
    PrintStatusAttributes(e.StatusAttributes);
    Console.WriteLine($"\t[\"x-ms-retry-after-ms\"] :
        { GetValueAsString(e.StatusAttributes, "x-ms-retry-after-ms")}");
    Console.WriteLine($"\t[\"x-ms-activity-id\"] :
        { GetValueAsString(e.StatusAttributes, "x-ms-activity-id")}");

    throw;
  }
}
```

Graph API – How: Query

Wide-Range of Data APIs

```
g.V()
.hasLabel('tag’)
.has('name’, 'Azure’)
.in('taggedAs’)
.hasLabel('presentation')
```

Gremlin API – Querying Options

Wide-Range of Data APIs

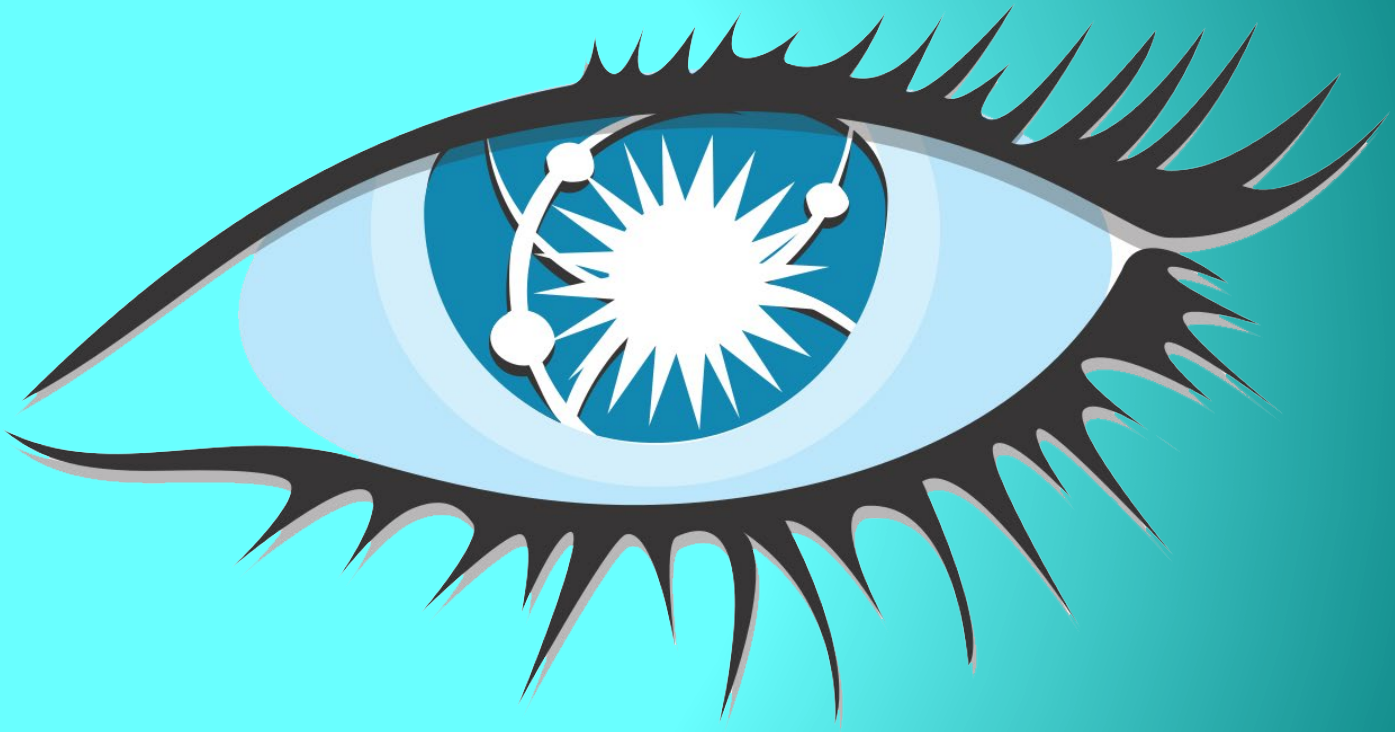
API

JavaScript

SQL API

Cassandra API

Wide Column Store



Cassandra API – What

Wide-Range of Data APIs

Handle Large Amounts of Data

Uses Commodity Servers

Developed by Facebook

Graph API – When

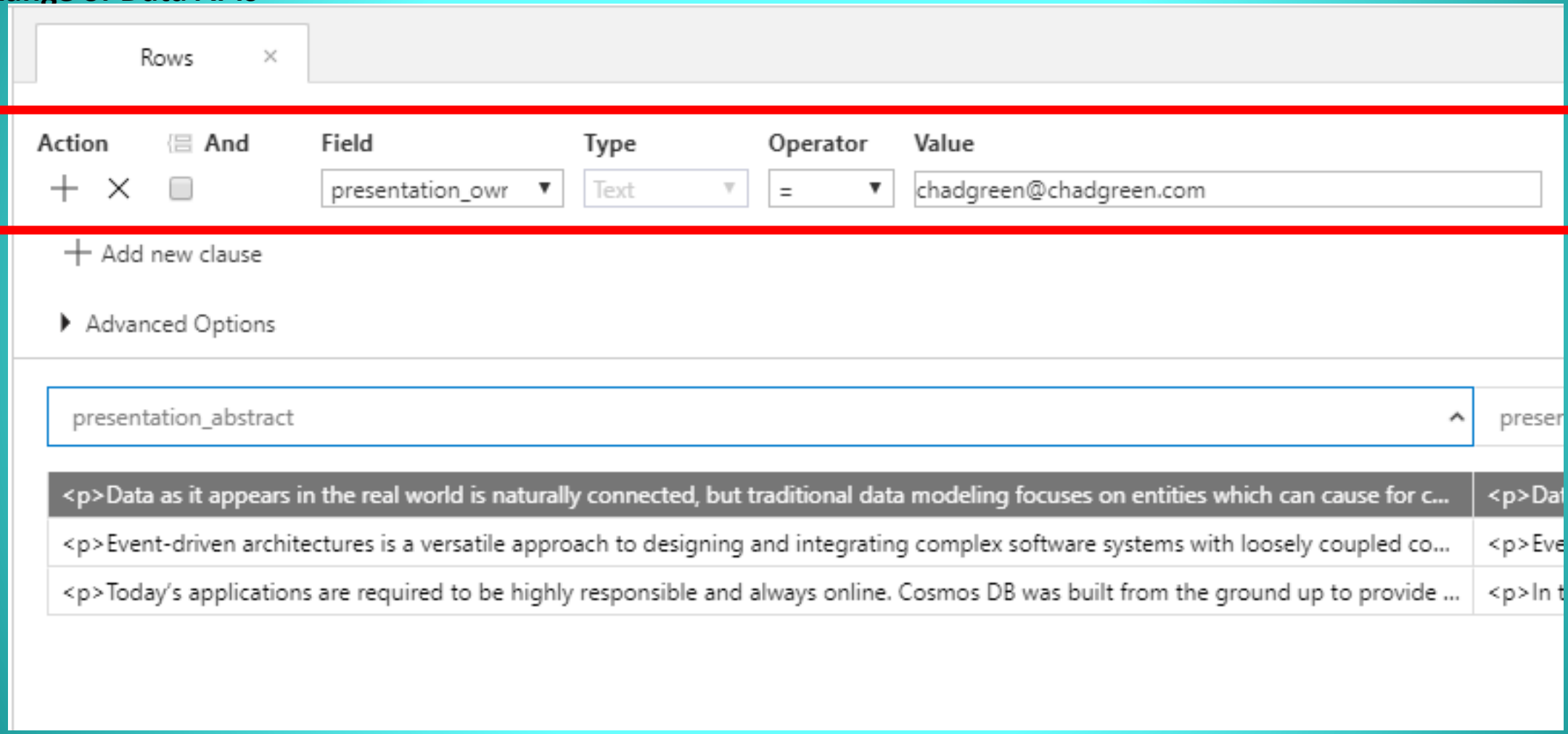
Wide-Range of Data APIs

Migrating data from Cassandra to

Azure Cosmos DB

Cassandra API – How: Data Model

Wide-Range of Data APIs



Cassandra API – How: Insert

Wide-Range of Data APIs

```
public class Presentation
{
  public string presentation_id { get; set; }
  public string presentation_owneremailaddress { get; set; }
  public string presentation_title { get; set; }
  public string presentation_abstract { get; set; }
  public string presentation_elevatorpitch { get; set; }
  public string presentation_importantinformation { get; set; }
  public string presentation_languagecode { get; set; }
  public string presentation_language { get; set; }
}
```

Cassandra API – How: Insert

Wide-Range of Data APIs

```
var options = new Cassandra.SSLOptions(SslProtocols.Tls12,
    true, ValidateServerCertificate);

options.SetHostNameResolver((ipAddress) => CassandraContactPoint);

Cluster cluster = Cluster.Builder()
    .WithCredentials(UserName, Password)
    .WithPort(CassandraPort)
    .AddContactPoint(CassandraContactPoint).WithSSL(options).Build();

ISession session = cluster.Connect();

session = cluster.Connect("speakingengagements");
IMapper mapper = new Mapper(session);

// Inserting Data into presentation table
mapper.Insert<Presentation>(Generate.Presentation(
    ExistingPresentation.EventDrivenArchitectureInTheCloud));
```

Cassandra API – How: Query

Wide-Range of Data APIs

```
Presentation presentationId3 = mapper.FirstOrDefault<Presentation>
    ("Select * from presentation where presentation_owneremailaddress = ?",
    "chadgreen@chadgreen.com");
Console.WriteLine(presentationId3.presentation_title);
```

Cassandra API – Query Options

Wide-Range of Data APIs

API

CQL

Cassandra-Based Tools

Table API

Table Storage

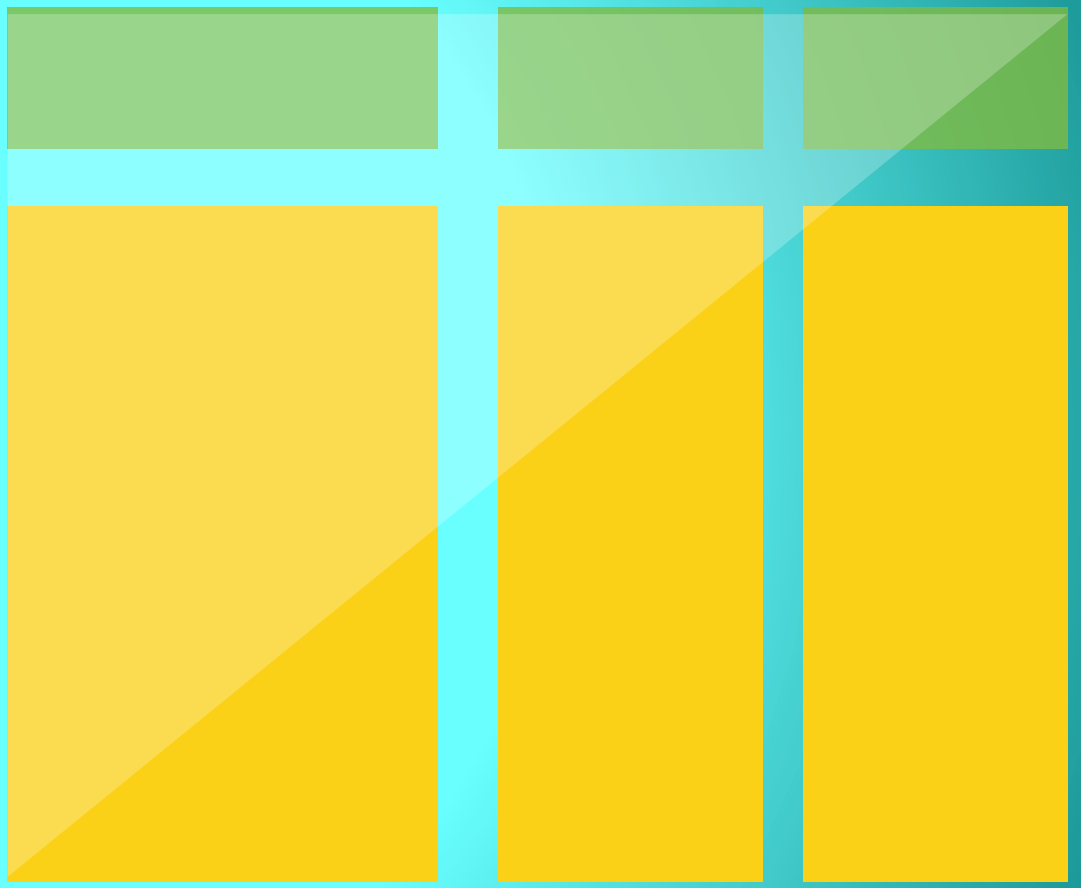


Table API – What

Wide-Range of Data APIs

Store Large Amounts of Data

Storing Structured Non-Related Data

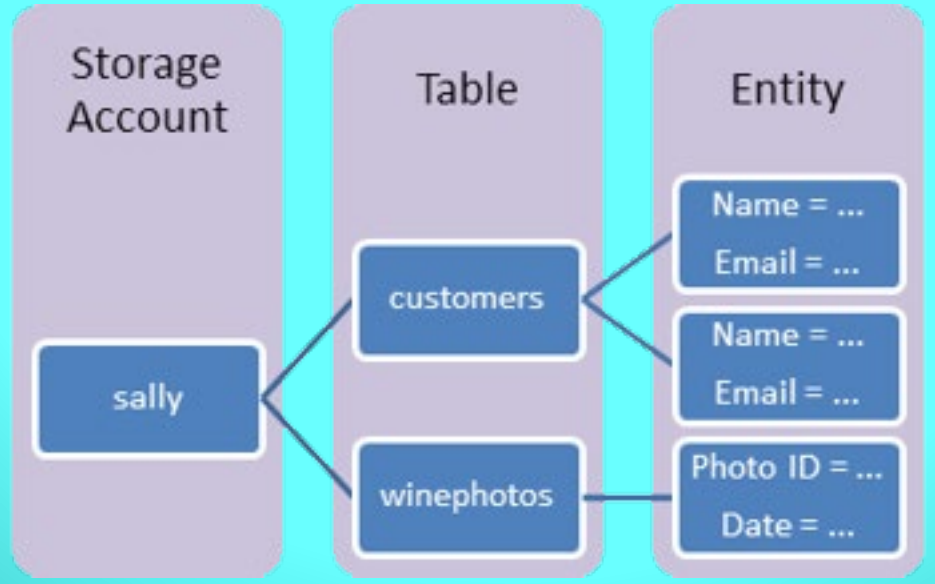


Table API – When

Wide-Range of Data APIs

Migrating data from Azure Table storage to Cosmos DB

Table API – How: Data Model

Wide-Range of Data APIs

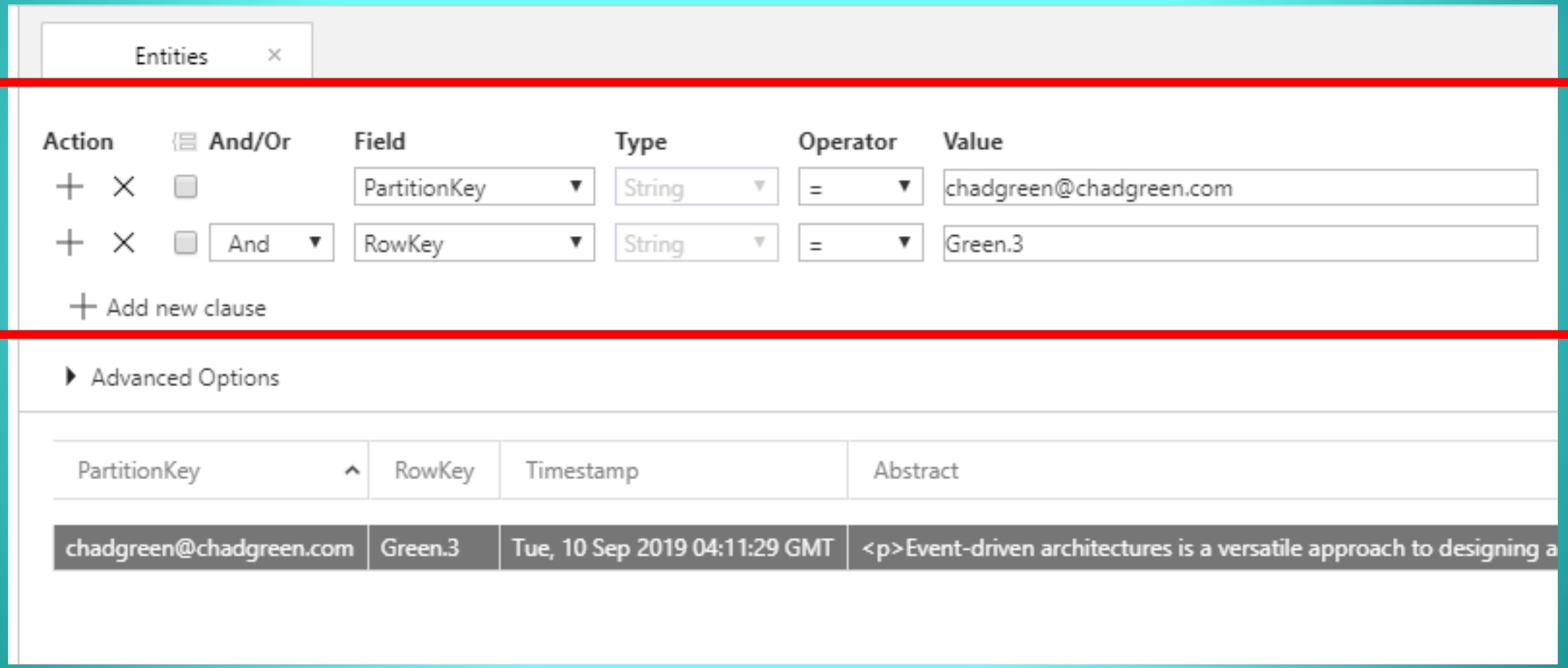


Table API – How: Insert

Wide-Range of Data APIs

```
public class Presentation : TableEntity
{
  public Presentation() { }

  public Presentation(string id, string ownerEmailAddress)
  {
    PartitionKey = ownerEmailAddress;
    RowKey = id;
  }

  public string Title { get; set; }
  public string Abstract { get; set; }
  public string ElevatorPitcch { get; set; }
  public string ImportantDetails { get; set; }
  public string LanguageCode { get; set; }
  public string Langauge { get; set; }
}
```

Table API – How: Insert

Wide-Range of Data APIs

```
public class Presentation : TableEntity
{
  public Presentation() { }

  public Presentation(string id, string ownerEmailAddress)
  {
    PartitionKey = ownerEmailAddress;
    RowKey = id;
  }

  public string Title { get; set; }
  public string Abstract { get; set; }
  public string ElevatorPitcch { get; set; }
  public string ImportantDetails { get; set; }
  public string LanguageCode { get; set; }
  public string Langauge { get; set; }
}
```

Table API – How: Insert

Wid

```
public static async Task<Presentation> InsertOrMergeEntityAsync(CloudTable table, Presentation entity)
{
  if (entity == null)
  {
    throw new ArgumentNullException("entity");
  }

  try
  {
    // Create the InsertOrReplace table operation
    TableOperation insertOrMergeOperation = TableOperation.InsertOrMerge(entity);

    // Execute the operation.
    TableResult result = await table.ExecuteAsync(insertOrMergeOperation);
    Presentation insertedCustomer = result.Result as Presentation;

    if (result.RequestCharge.HasValue)
    {
      Console.WriteLine("Request Charge of InsertOrMerge Operation: " + result.RequestCharge);
    }

    return insertedCustomer;
  }
  catch (StorageException e)
  {
    Console.WriteLine(e.Message);
    Console.ReadLine();
    throw;
  }
}
```

Table API – How: Insert

Wid

```
public static async Task<Presentation> InsertOrMergeEntityAsync(CloudTable table, Presentation entity)
{
  if (entity == null)
  {
    throw new ArgumentNullException("entity");
  }

  try
  {
    // Create the InsertOrReplace table operation
    TableOperation insertOrMergeOperation = TableOperation.InsertOrMerge(entity);

    // Execute the operation.
    TableResult result = await table.ExecuteAsync(insertOrMergeOperation);
    Presentation insertedCustomer = result.Result as Presentation;

    if (result.RequestCharge.HasValue)
    {
      Console.WriteLine("Request Charge of InsertOrMerge Operation: " + result.RequestCharge);
    }

    return insertedCustomer;
  }
  catch (StorageException e)
  {
    Console.WriteLine(e.Message);
    Console.ReadLine();
    throw;
  }
}
```

Table API – How: Query

Wi

```
public static async Task<Presentation> RetrieveEntityUsingPointQueryAsync(CloudTable table,
    string partitionKey, string rowKey)
{
  try
  {
    TableOperation retrieveOperation
        = TableOperation.Retrieve<Presentation>(partitionKey, rowKey);
    TableResult result = await table.ExecuteAsync(retrieveOperation);
    Presentation presentation = result.Result as Presentation;
    if (presentation != null)
    {
      Console.WriteLine("\t{0}\t{1}\t{2}", presentation.PartitionKey,
          presentation.RowKey, presentation.Title);
    }

    if (result.RequestCharge.HasValue)
    {
      Console.WriteLine("Request Charge of Retrieve Operation: " + result.RequestCharge);
    }

    return presentation;
  }
  catch (StorageException e)
  {
    Console.WriteLine(e.Message);
    Console.ReadLine();
    throw;
  }
}
```

Table API – How: Query

Wide Range of Data APIs

```
public static async Task<Presentation> RetrieveEntityUsingPointQueryAsync(CloudTable table,
    string partitionKey, string rowKey)
{
  try
  {
      TableOperation retrieveOperation
          = TableOperation.Retrieve<Presentation>(partitionKey, rowKey);
      TableResult result = await table.ExecuteAsync(retrieveOperation);
      Presentation presentation = result.Result as Presentation;
    if (presentation != null)
    {
      Console.WriteLine("\t{0}\t{1}\t{2}", presentation.PartitionKey,
          presentation.RowKey, presentation.Title);
    }

    if (result.RequestCharge.HasValue)
    {
      Console.WriteLine("Request Charge of Retrieve Operation: " + result.RequestCharge);
    }

    return presentation;
  }
  catch (StorageException e)
  {
    Console.WriteLine(e.Message);
    Console.ReadLine();
    throw;
  }
}
```

etcd API

Key-Value Storge

etcd

etcd API – What

Wide-Range of Data APIs

Simple

Secure

kubernetes

Fast

Reliable

etcd API – Why

Wide-Range of Data APIs

No etcd Operations Management

etcd API – Why

Wide-Range of Data APIs

No etcd Operations Management

etcd API – Why

Wide-Range of Data APIs

No etcd Operations Management

Global distribution & high availability

etcd API – Why

Wide-Range of Data APIs

No etcd Operations Management

Global distribution & high availability

etcd API – Why

Wide-Range of Data APIs

No etcd Operations Management

Elastic scalability

Global distribution & high availability

etcd API – Why

Wide-Range of Data APIs

No etcd Operations Management

Elastic scalability

Global distribution & high availability

etcd API – Why

Wide-Range of Data APIs

No etcd Operations Management

Elastic scalability

Security & enterprise readiness

Global distribution & high availability

etcd API – When

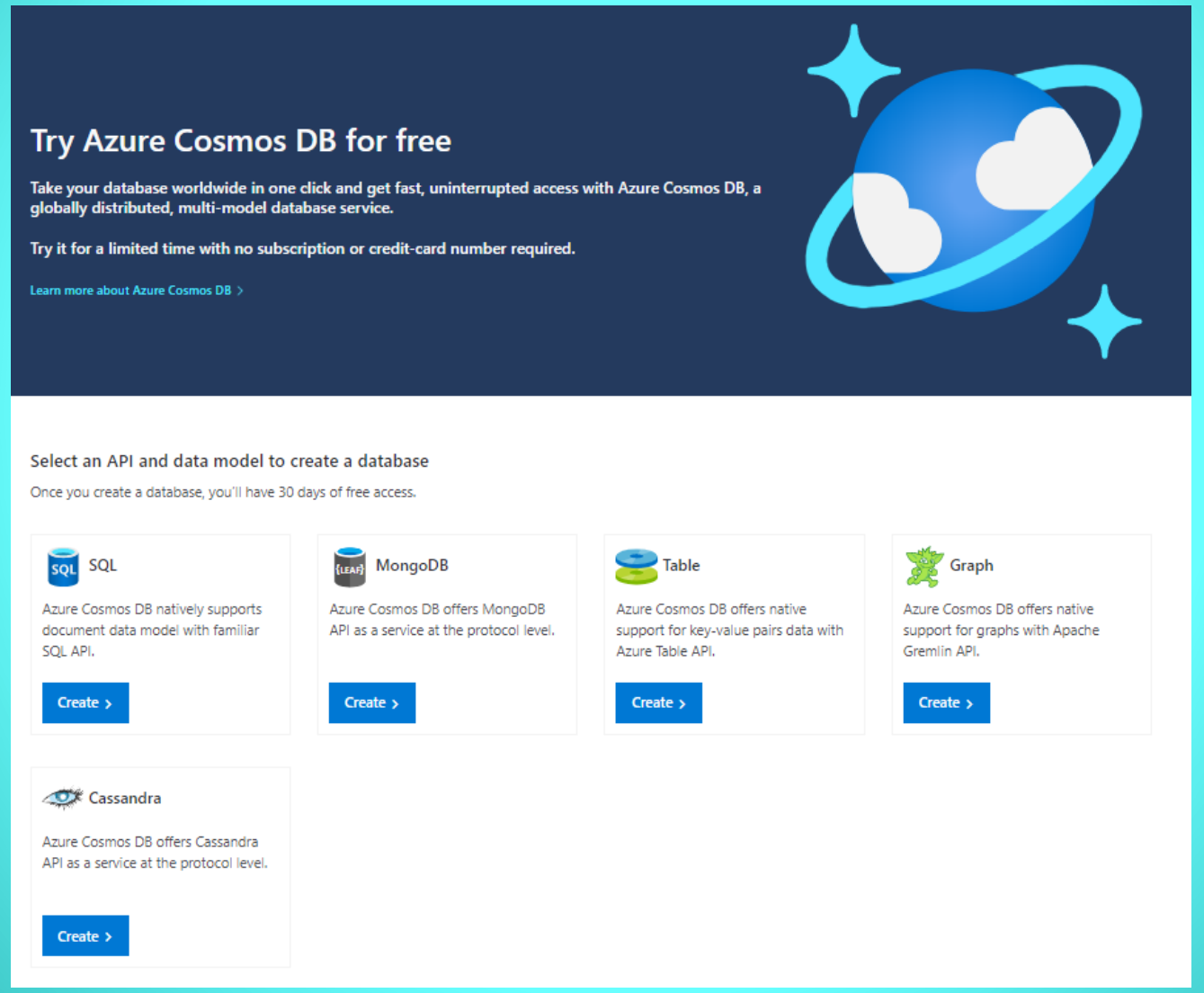
Wide-Range of Data APIs

Allows developers to scale Kubernetes state management on a fully managed cloud native PaaS service

Call to Action

The Hitchhiker's Guide to the Cosmos

azure.Microsoft.com/en-us/try/cosmosdb



Free Database

Free Database

Free Database

1000 RU/s

25-Gb

Serverless

Thank You!

chadgreen@chadgreen.com

TaleLearnCode

TaleLearnCode.com

ChadGreen & TaleLearnCode

ChadwickEGreen