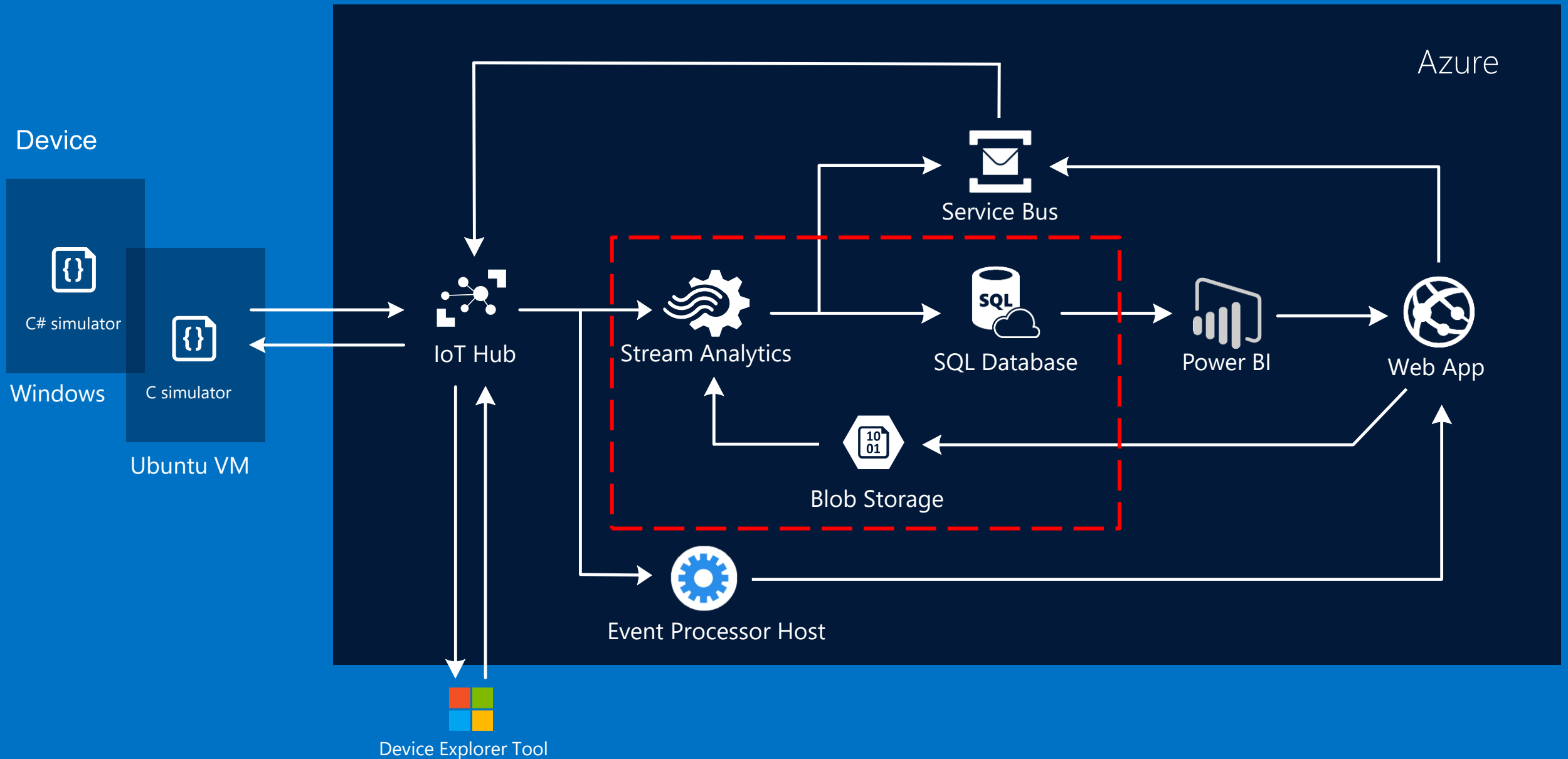


# Data Processing and Device Alert Part I

## Historic Data Processing in Stream Analytics

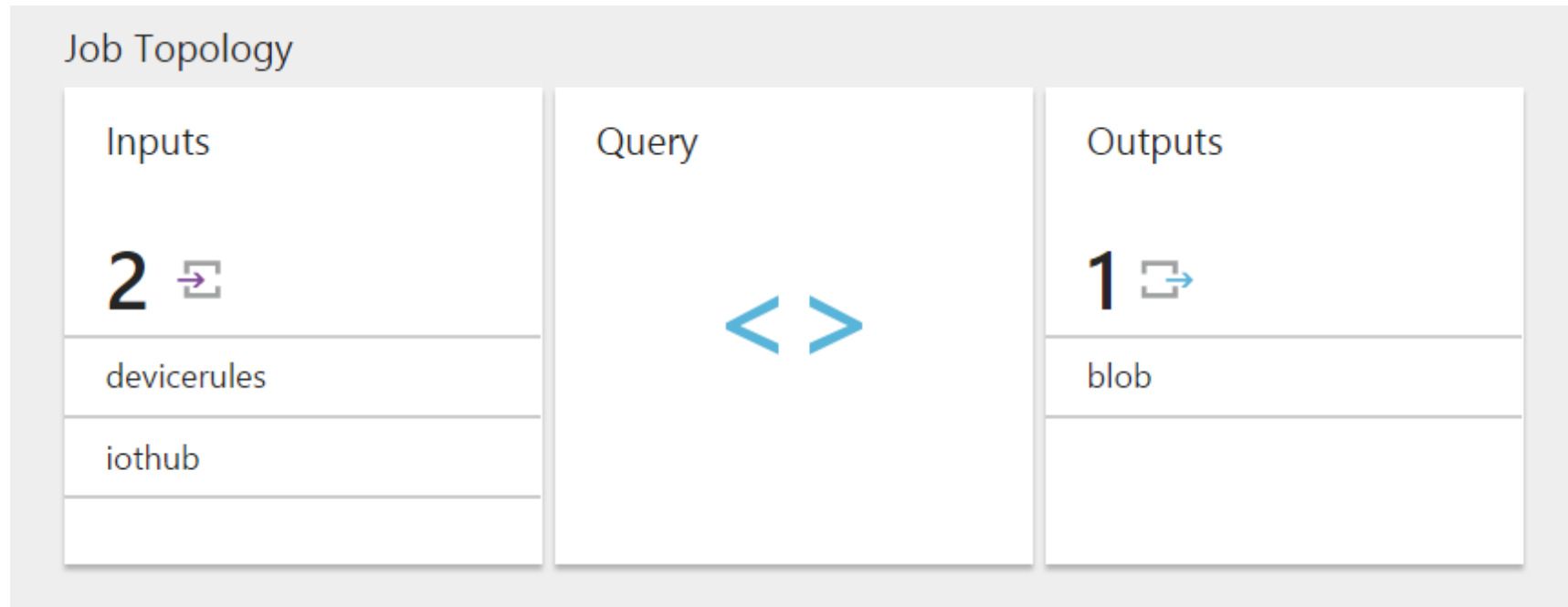


# HOL 4 – Historic Data Processing in Stream Analytics



# Azure Stream Analytics

# Concept of Azure Stream Analytics

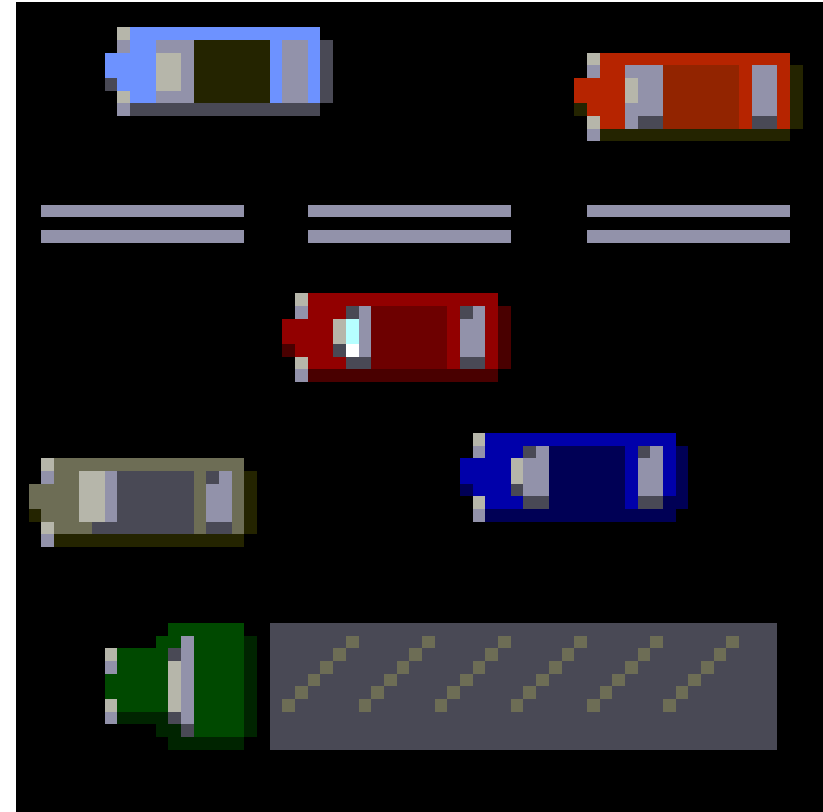


# What is Streaming Data?

Data at Rest



Data in Motion





# Introducing Azure Stream Analytics

Fully managed  
real-time analytics



Mission critical  
reliability and scale



Enables rapid  
development



# Real-time analytics

Fully managed  
real-time analytics



## Real-time Analytics

- Intake **millions of events per second** (up to 1 GB/s)
- **Low processing latency**, auto adaptive (sub-second to seconds)
- **Correlate** between different streams, or with reference data
- Find **patterns** or lack of patterns in data in real-time

## Fully Managed Cloud Service

- No hardware acquisition and maintenance
- No platform/infrastructure deployment and maintenance
- Easily **expand your business globally** leveraging Azure regions

# Mission critical

Mission critical  
reliability and scale



## Mission Critical Reliability

- **Guaranteed event delivery**
- **Guaranteed business continuity:** Automatic and fast recovery

## Effective Audits

- **Privacy and security** properties of solutions are evident
- Azure integration for **monitoring and ops alerting**

## Easy To Scale

- **Scale** from small to large on demand



# Rapid development

Enables rapid development



## Rapid Development with SQL like language

- **High-level:** focus on stream analytics solution
- **Concise:** less code to maintain
- **Fast test:** Rapid development and debugging
- First-class support for **event streams and reference data**

## Built in temporal semantics

- Built-in **temporal windowing and joining**
- Simple **policy configuration** to manage out-of-order events and late arrivals

# SAQL – Language & Library

## DML

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING
- CASE WHEN THEN ELSE
- INNER/LEFT OUTER JOIN
- UNION
- CROSS/OUTER APPLY
- CAST
- INTO
- ORDER BY ASC, DSC

## Scaling Extensions

- WITH
- PARTITION BY
- OVER

## Date and Time Functions

- DateName
- DatePart
- Day
- Month
- Year
- DateTimeFromParts
- DateDiff
- DateAdd

## Temporal Functions

- Lag, IsFirst
- CollectTop

## Windowing Extensions

- TumblingWindow
- HoppingWindow
- SlidingWindow

## Aggregate Functions

- Sum
- Count
- Avg
- Min
- Max
- StDev
- StDevP
- Var
- VarP

## String Functions

- Len
- Concat
- CharIndex
- Substring
- PatIndex

# ASA - Inputs

## Data stream inputs

- **Event Hub:** Feed events into Azure in real time, and Stream Analytics jobs can process them
- **IoT Hub:** Stream information collected from connected devices directly into a streaming analytics job
- **Blob Storage:** Can be used as an input source for ingesting bulk data

## Reference Data

- Static or slow changing used for performing look-ups
- Azure Blob Storage is the only supported reference input
- Source blobs limited to 50MB in size

## Input Data Types

- CSV, JSON and Avro Serialized

# ASA - Queries

- Performs transformations and computations over streams of events
- Stream Analytics Query Language
  - *Subset of T-SQL syntax*
- Multiple built-in functions
- Supports multiple data types as well as complex
- 3 types of windowing
  - *Tumbling*
  - *Hopping*
  - *Sliding*
- Queries can be created via Azure Portal or via API calls



```
1 WITH [StreamData] AS (SELECT * FROM [IoTHubStream1])
2
3 SELECT *
4 INTO [Telemetry]
5 FROM [StreamData]
6
7 SELECT
8     DeviceId,
9     AVG (Humidity) AS [AverageHumidity],
10    MIN(Humidity) AS [MinimumHumidity],
11    MAX(Humidity) AS [MaxHumidity],
12    5.0 AS TimeframeMinutes
13 INTO
14     [TelemetrySummary]
15 FROM
16     [StreamData]
17 WHERE
18     [Humidity] IS NOT NULL
19 GROUP BY
20     DeviceId, slidingWindow (mi, 5)
```



# ASA - Outputs

- Multiple options for storing output and viewing analysis results  
Route to one or more outputs
- Flexibility in the consumption and storage of the job output for data warehousing and other purposes

## **Available Outputs:**

- SQL Database
- Blob Storage
- Event Hub
- Power BI
- Table Storage
- Service Bus Queues and Topics
- Document DB

# ASA – Management

## Portal

- Inputs/Queries/Outputs (stop first)
- Scale & Resource Management
- Start/Stop Jobs
- Other admin options

## Code

- .NET SDK reference
- REST endpoint available
- ARM via Azure PowerShell
- No extensibility for custom analysis

The screenshot displays the 'Running' status of an IoT Hub job. The top section, titled 'Essentials', provides key details: Resource group (iotpreviewtest), Status (Running), Created (Wednesday, November 11, 2015, 3:26:38 P...), Last output (-), Subscription id (7af011da-932a-4fff-83fe-ee66e0b4ee7b), Region (East US), Started (Tuesday, January 19, 2016, 5:11:59 PM), and Subscription name (IntergenUSA). A 'Send feedback UserVoice' link is also present. Below this, the 'Job Topology' section shows a visual representation of the job's components: Inputs (2), Query (represented by a blue double arrow), and Outputs (2). The inputs are listed as 'IoTHubStream' and 'IoTHubStream1', and the outputs are 'Telemetry' and 'TelemetrySummary'. An 'Add tiles' button is visible in the top right of the Job Topology section.

| Essentials  |  |
|---|--|
| Resource group<br>iotpreviewtest                        | Send feedback<br><a href="#">UserVoice</a>       |
| Status<br>Running                                       | Region<br>East US                                |
| Created<br>Wednesday, November 11, 2015, 3:26:38 P...   | Started<br>Tuesday, January 19, 2016, 5:11:59 PM |
| Last output<br>-  | Subscription name<br>IntergenUSA                 |
| Subscription id<br>7af011da-932a-4fff-83fe-ee66e0b4ee7b |  |

[All settings →](#)

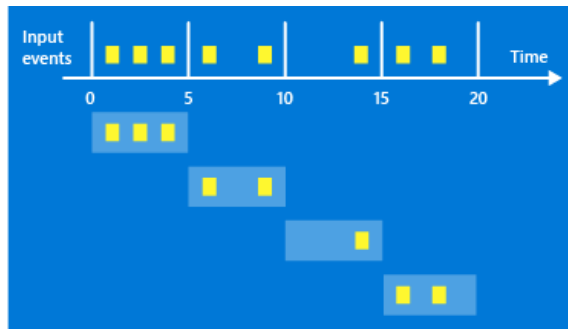
### Job Topology

Add tiles ⊕

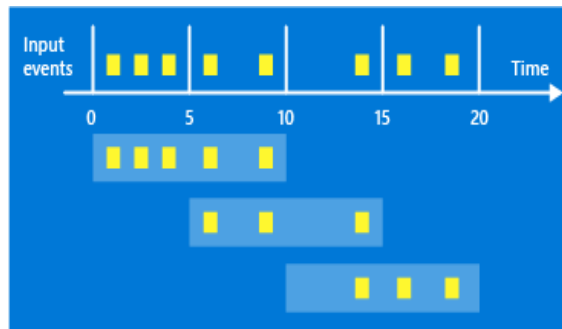
| Inputs        | Query | Outputs          |
|---------------|-------|------------------|
| 2 ➡           | < >   | 2 ➡              |
| IoTHubStream  |       | Telemetry        |
| IoTHubStream1 |       | TelemetrySummary |

# Three types of windows

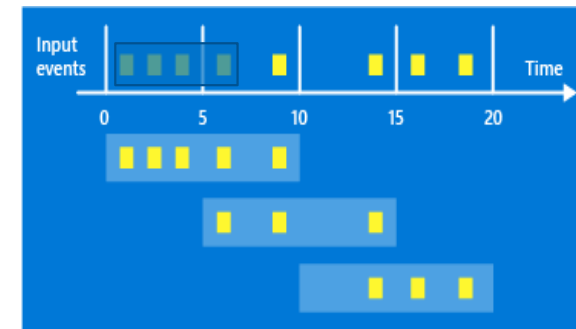
- Every window operation outputs events at the end of the window
- Will have the time stamp of the window
- All windows have a fixed length



Tumbling window  
*Aggregate per time interval*



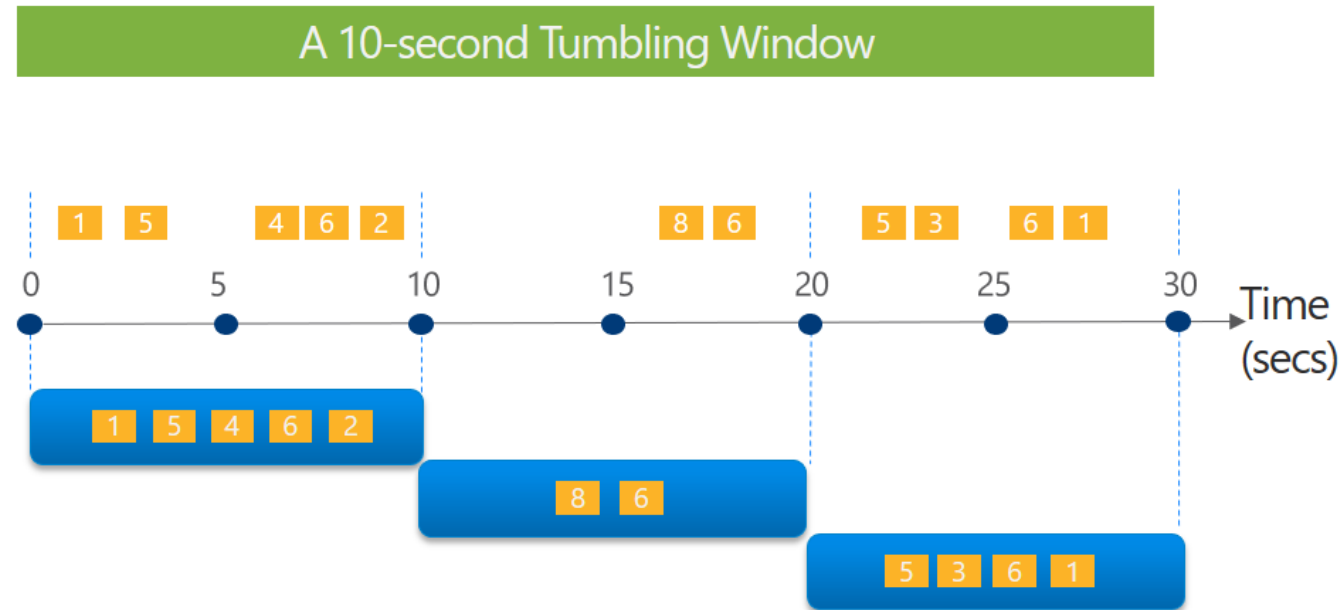
Hopping window  
*Schedule overlapping windows*



Sliding window  
*Windows constant re-evaluated*

# Tumbling Window

Tell me the count of tweets per time zone every 10 seconds



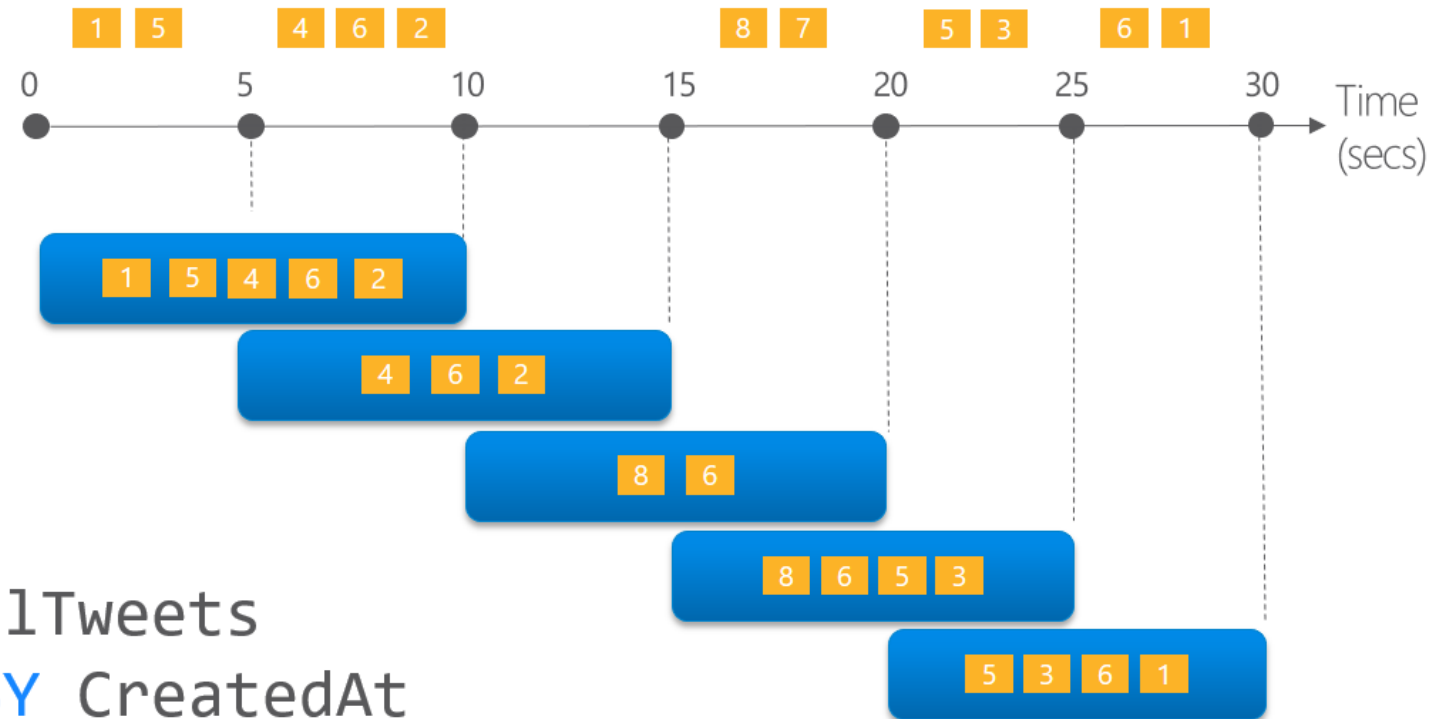
```
SELECT TimeZone, COUNT(*) AS Count
FROM TwitterStream TIMESTAMP BY CreatedAt
GROUP BY TimeZone, TumblingWindow(second,10)
```



# Hopping Window

Every 5 seconds give me the count of tweets over the last 10 seconds

A 10-second Hopping Window with a 5-second "Hop"

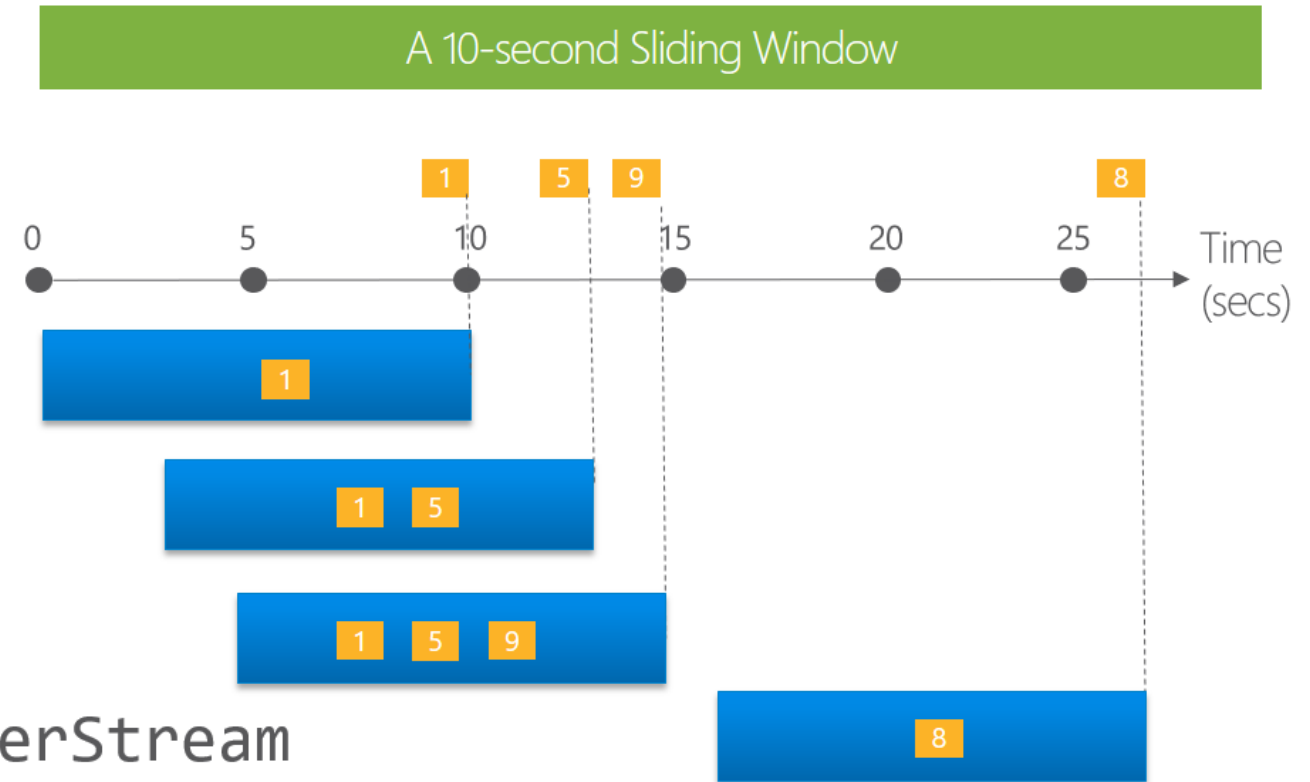


```
SELECT Topic, COUNT(*) AS TotalTweets
FROM TwitterStream TIMESTAMP BY CreatedAt
GROUP BY Topic, HoppingWindow(second, 10 , 5)
```

# Sliding Window

Give me the count of tweets for all topics which are tweeted more than 10 times in the last 10 seconds

```
SELECT Topic, COUNT(*) FROM TwitterStream
TIMESTAMP BY CreatedAt
GROUP BY Topic, SlidingWindow(second, 10)
HAVING COUNT(*) > 10
```



# ASA – Common Query Patterns

<https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-stream-analytics-query-patterns>

# Azure Blob Storage

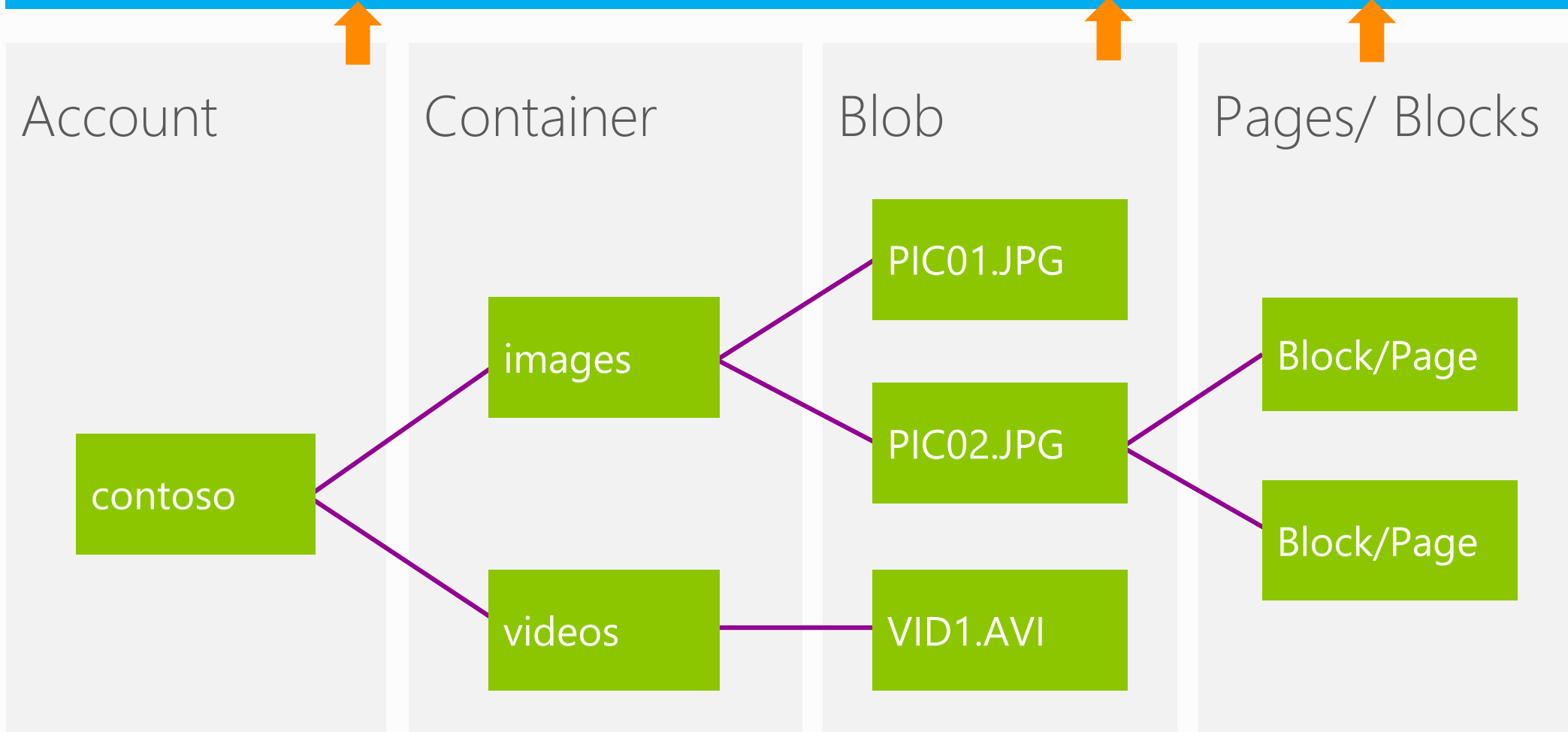


# Blob

- Binary Large object
- Unstructured data
  - audio, video, document etc.
- Up to 1TB each
- Scoped by the containers

# Blob Storage Concepts

`http://<account>.blob.core.windows.net/<container>/<blobname>`



# Azure SQL Database

# Database-as-a-Service



# Azure SQL Database

Fully managed database-as-a-service built on SQL with near zero administration



Built for SaaS and Enterprise applications

Predictable performance & Pricing

Elastic database pool for unpredictable SaaS workloads

99.99% availability built-in

Geo-replication and restore services for data protection

Secure and compliant for your sensitive data

Fully compatible with SQL Server 2014 databases

# SQL Database Service Tiers

|                     | Basic   | Standard                                    | Premium   |
|---------------------|---|---|---|
| Intended Use        | Light transactional workloads                   | Go-to option for most business applications | High throughput and business-critical databases |
| Workload Elasticity | Isolated databases and elastic database pools   |   |   |
| Performance         | •   | • •   | • • •   |
| Business Continuity | •   | • •   | • • •   |
| Programming Surface | Fully compatible with SQL Server 2014 databases |   |   |
| Availability        | 99.99%*   |   |   |

How to connect to  
SQL DB and start coding

# Your choice of language and tooling

C#  
VB.NET

Java

C/C++

PHP

JavaScript

Python

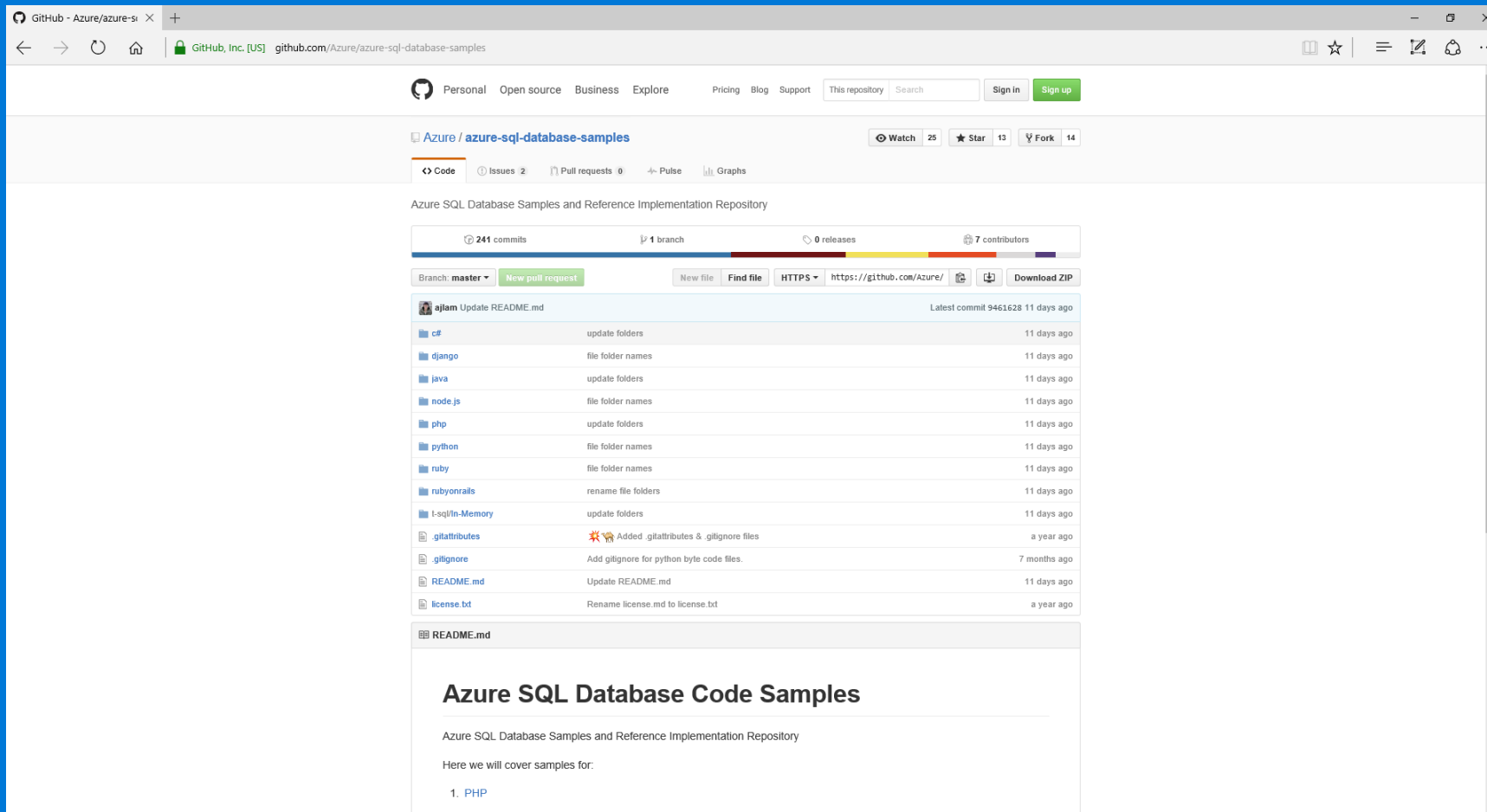
Ruby

SQL Server, Azure SQL Database, SQL DW



# Code Samples in GitHub

<https://github.com/Azure/azure-sql-database-samples>



The screenshot shows the GitHub repository page for `Azure/azure-sql-database-samples`. The repository is described as "Azure SQL Database Samples and Reference Implementation Repository". It has 241 commits, 1 branch, 0 releases, and 7 contributors. The latest commit is by `ajlam` updating the README.md, 11 days ago.

The file list shows the following structure:

| File/Folder                 | Description                               | Last Modified |
|-----------------------------|---|---------------|
| <code>c#</code>             | update folders                            | 11 days ago   |
| <code>django</code>         | file folder names                         | 11 days ago   |
| <code>java</code>           | update folders                            | 11 days ago   |
| <code>node.js</code>        | file folder names                         | 11 days ago   |
| <code>php</code>            | update folders                            | 11 days ago   |
| <code>python</code>         | file folder names                         | 11 days ago   |
| <code>ruby</code>           | file folder names                         | 11 days ago   |
| <code>rubyonrails</code>    | rename file folders                       | 11 days ago   |
| <code>i-sqlfn-Memory</code> | update folders                            | 11 days ago   |
| <code>.gitattributes</code> | Added .gitattributes & .gitignore files   | a year ago    |
| <code>.gitignore</code>     | Add gitignore for python byte code files. | 7 months ago  |
| <code>README.md</code>      | Update README.md                          | 11 days ago   |
| <code>license.txt</code>    | Rename license.md to license.txt          | a year ago    |

The README.md content is visible below the file list:

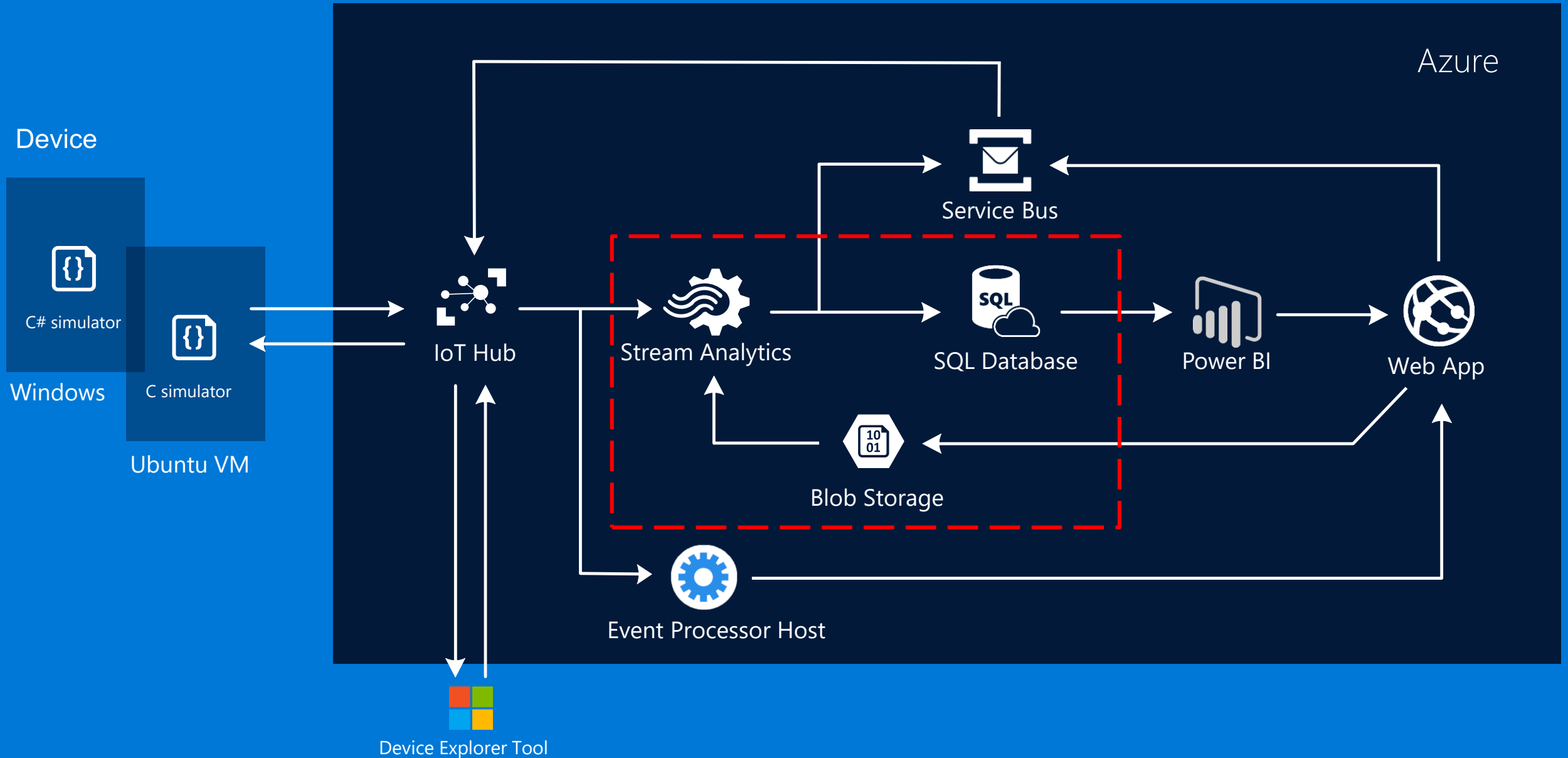
## Azure SQL Database Code Samples

Azure SQL Database Samples and Reference Implementation Repository

Here we will cover samples for:

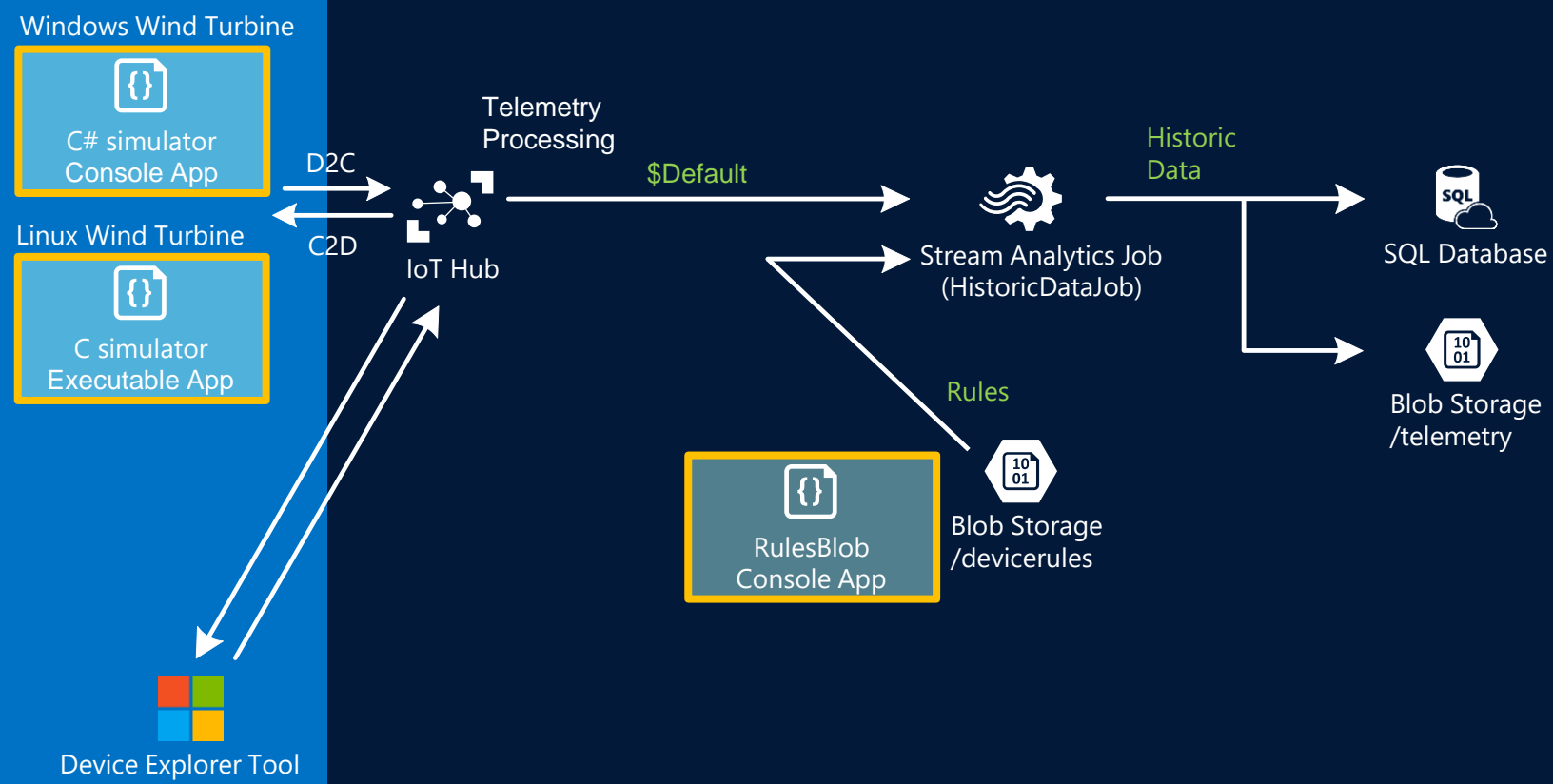
1. [PHP](#)

# HOL 4 – Historic Data Processing in Stream Analytics

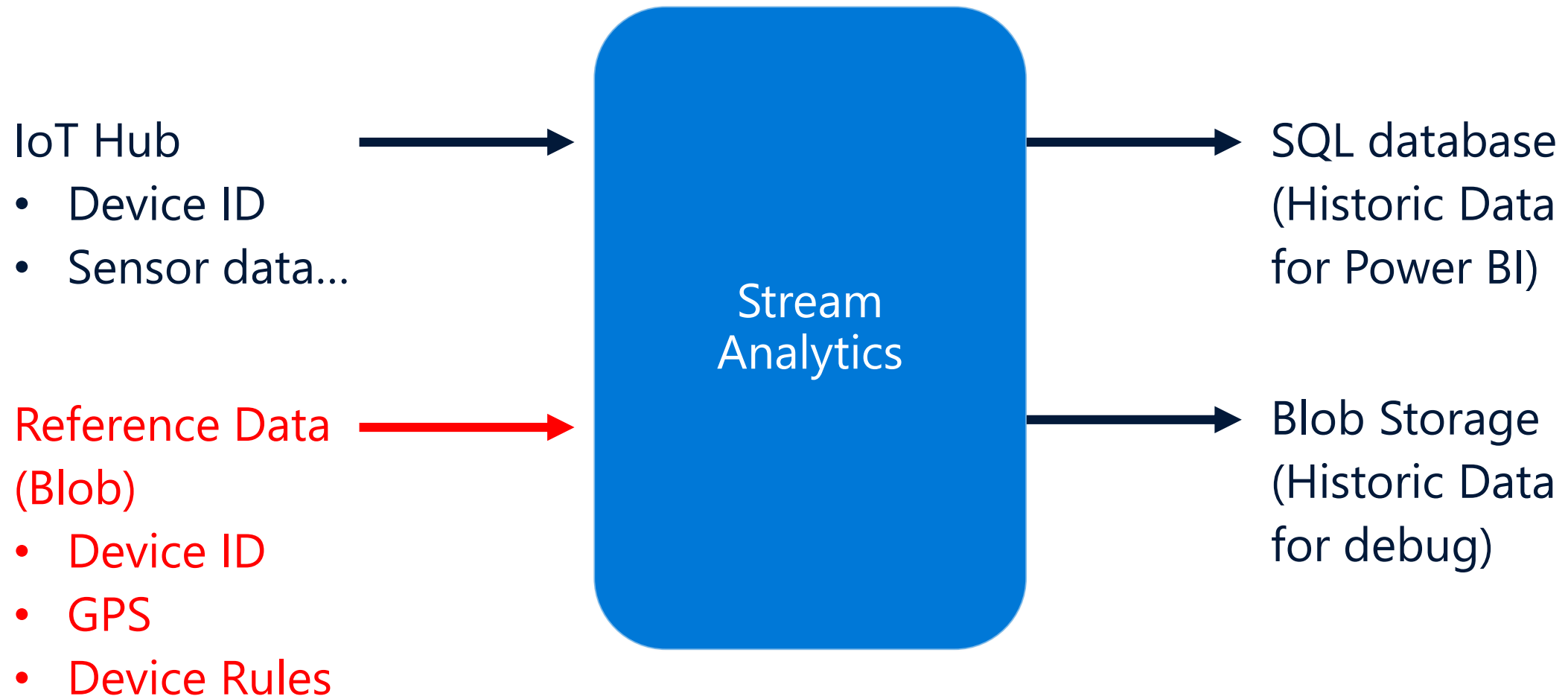




# Code & Modules



# Reference Data as input to Stream Analytics



## Developer Services



Visual Studio Team Services



Azure DevTest Labs\*



VS Application Insights\*



HockeyApp



Developer Tools

## Management & Security



Azure Portal



Scheduler



Automation



Log Analytics



Key Vault



Security Center\*

### Compute



Virtual Machines



Virtual Machine Scale Sets



Cloud Services



Batch



RemoteApp



Service Fabric



Azure Container Service



Web Apps



Mobile Apps



Logic Apps\*



API Apps



API Management



Notification Hubs



Mobile Engagement



Functions\*

### Data & Storage



SQL Database



DocumentDB



Redis Cache



Storage: Blobs, Tables, Queues, Files and Disks



StorSimple



Search



SQL Data Warehouse\*



SQL Server Stretch Database\*

### Analytics



Data Lake Analytics\*



Data Lake Store\*



HDInsight



Machine Learning



Stream Analytics



Data Factory



Data Catalog



Power BI Embedded\*

### Internet of Things & Intelligence



Azure IoT Suite



Azure IoT Hub



Event Hubs



Cortana Intelligence Suite



Cognitive Services\*

### Media & CDN



Media Services



Content Delivery Network

### Identity & Access Management



Azure Active Directory



B2C\*



Domain Services\*



Multi-Factor Authentication

## Hybrid Integration



BizTalk Services



Service Bus



Backup



Site Recovery

## Networking



Virtual Network



ExpressRoute



Traffic Manager



Load Balancer



Azure DNS\*



VPN Gateway



Application Gateway

# Let's Go

- Historic Data Processing in Stream Analytics

(Please refer the 04-HOL-Historic Data Processing in Stream Analytics file)

- Create an **Azure SQL Database** for historic data storing
- Create a reference **Blob** for device rules (Using a windows console App)
- Provision a **Azure Stream Analytics Job** for the historic data processing
  - **Input**
    - IoTHub
    - Reference Blob
  - **Output**
    - Azure SQL Database
    - Blob

BACKUP

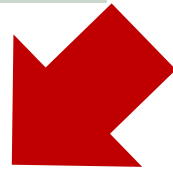
# SQL JOIN

customers

| CustomerId | Name   | Phone | E-mail |
|------------|--------|-------|--------|
| 1          | Walker | 123   | 123@   |
| 2          | Carol  | 456   | 456@   |
| 3          | LeAnn  | 789   | 789@   |

orders

| OrderID | OrderNo | CustomerId |
|---------|---------|------------|
| 1       | 1122    | 2          |
| 2       | 5566    | 1          |
| 3       | 8899    | 1          |
| 4       | 3344    | 3          |



| Name   | OrderNo |
|--------|---------|
| Walker | 5566    |
| Walker | 8899    |
| Carol  | 1122    |
| LeAnn  | 3344    |

**SELECT**

customers.Name, orders.OrderNo

**FROM**

customers

**LEFT JOIN** orders

**ON**

customers.CustomerId = orders.CustomerId;

