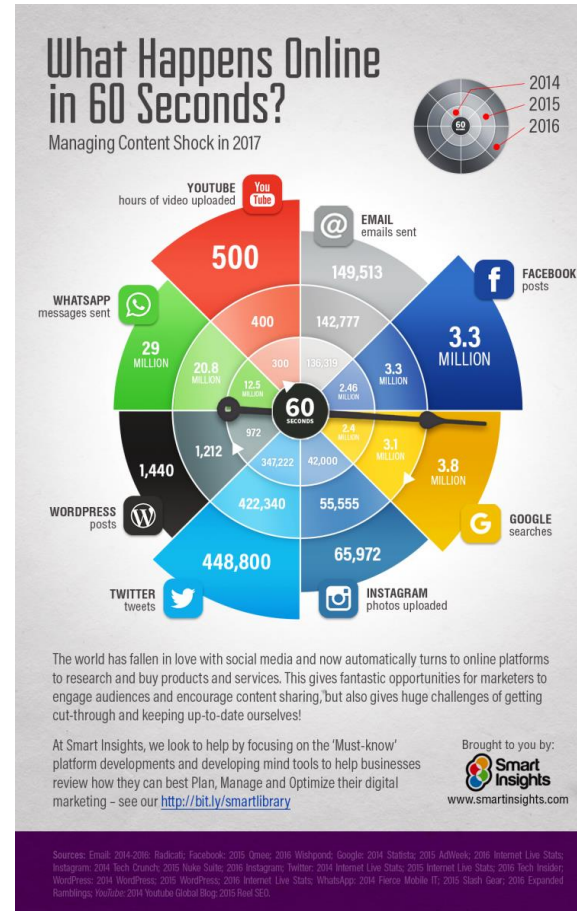


Real-time Analytics

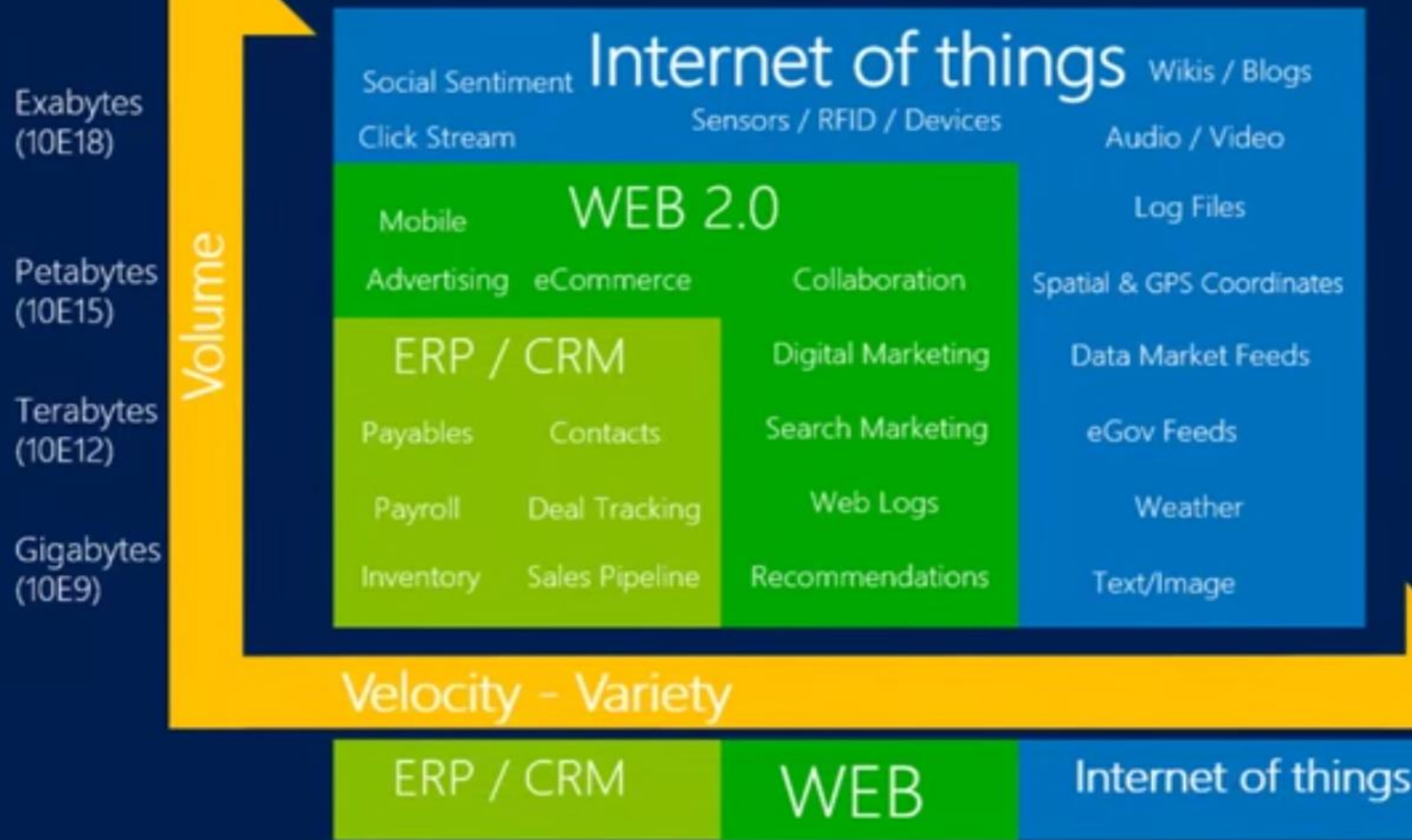


What happens in 60 seconds?



Introducing Big Data

Continued



Defining Real-time

Within seconds...

or...

Within minutes...

of an event occurring

Up to 2 hours



Timeliness of Information



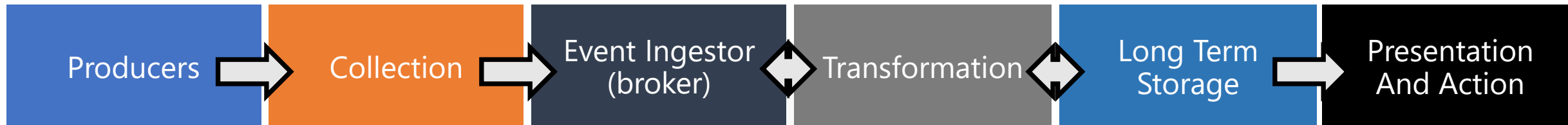
What was trending in the past 5 minutes?

Amber alert car detected!





A tornado will form in the next 30 minutes.

Typical Event Processing




Applications



Cloud Gateways
(WebAPIs)


Scalable
Event Broker



Real-Time Analytics


External
Data Sources



Web/Thick
Client Dashboards


Devices


Field Gateways


Event Hub


Search And Query


Data Analytics

ETL Timespan
(Extract, Transform, Load)

DATA INGESTION

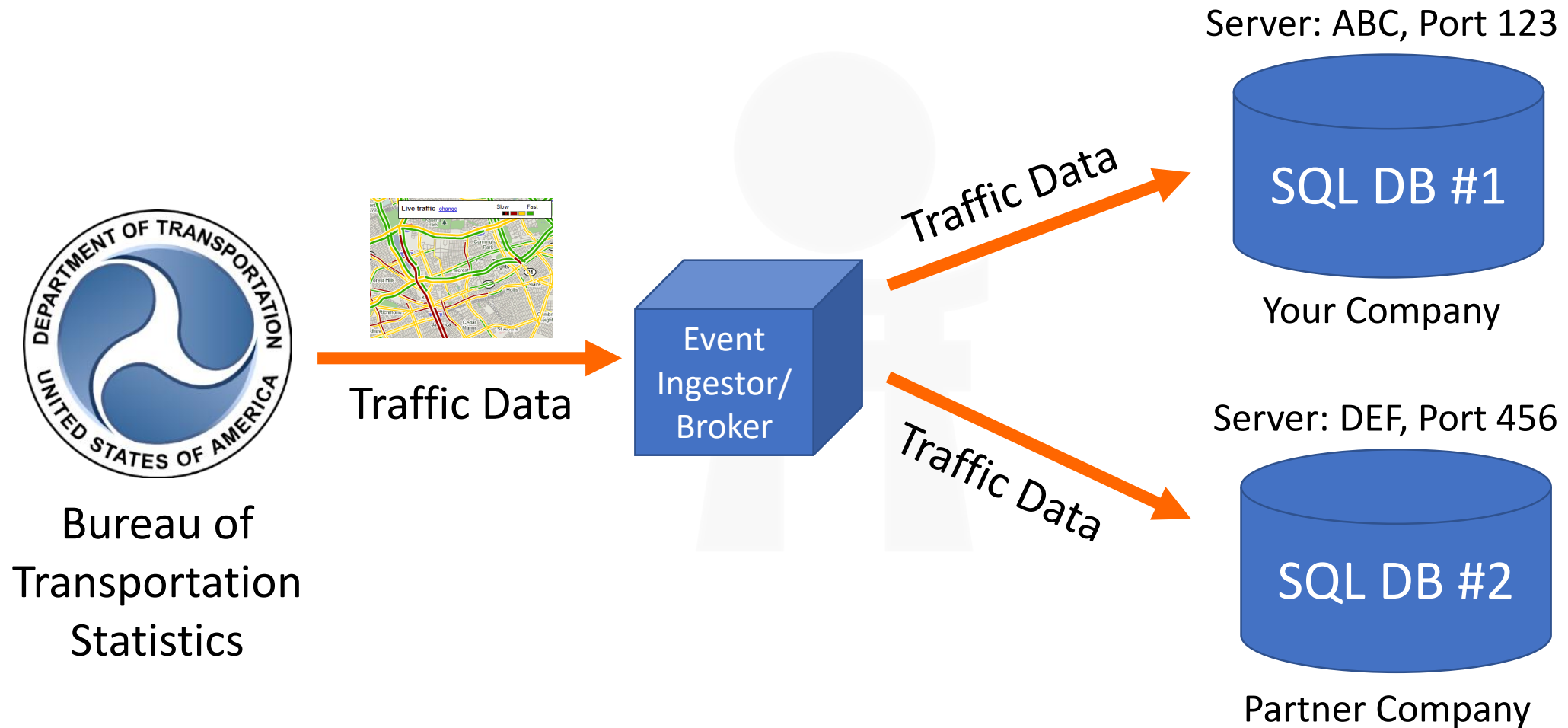


The Post Office & Shipping Centers



- Tracks address changes
- Tries again tomorrow if send failed
- Holds packages in short term
 - Too many failed deliveries
 - Vacations
- Reduces complexity through specialization
- Optimized to send, receive, and temporarily house packages

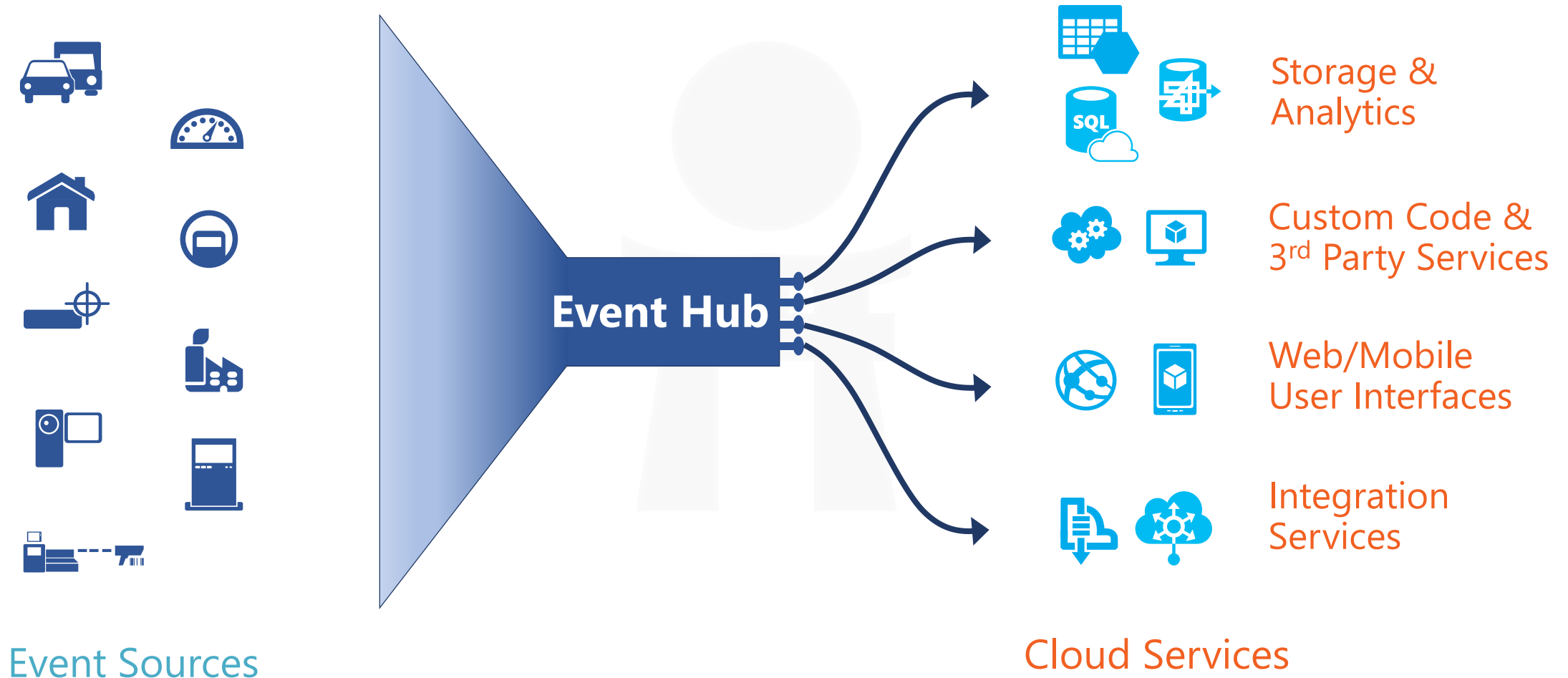
Preventative Solution: Middleware



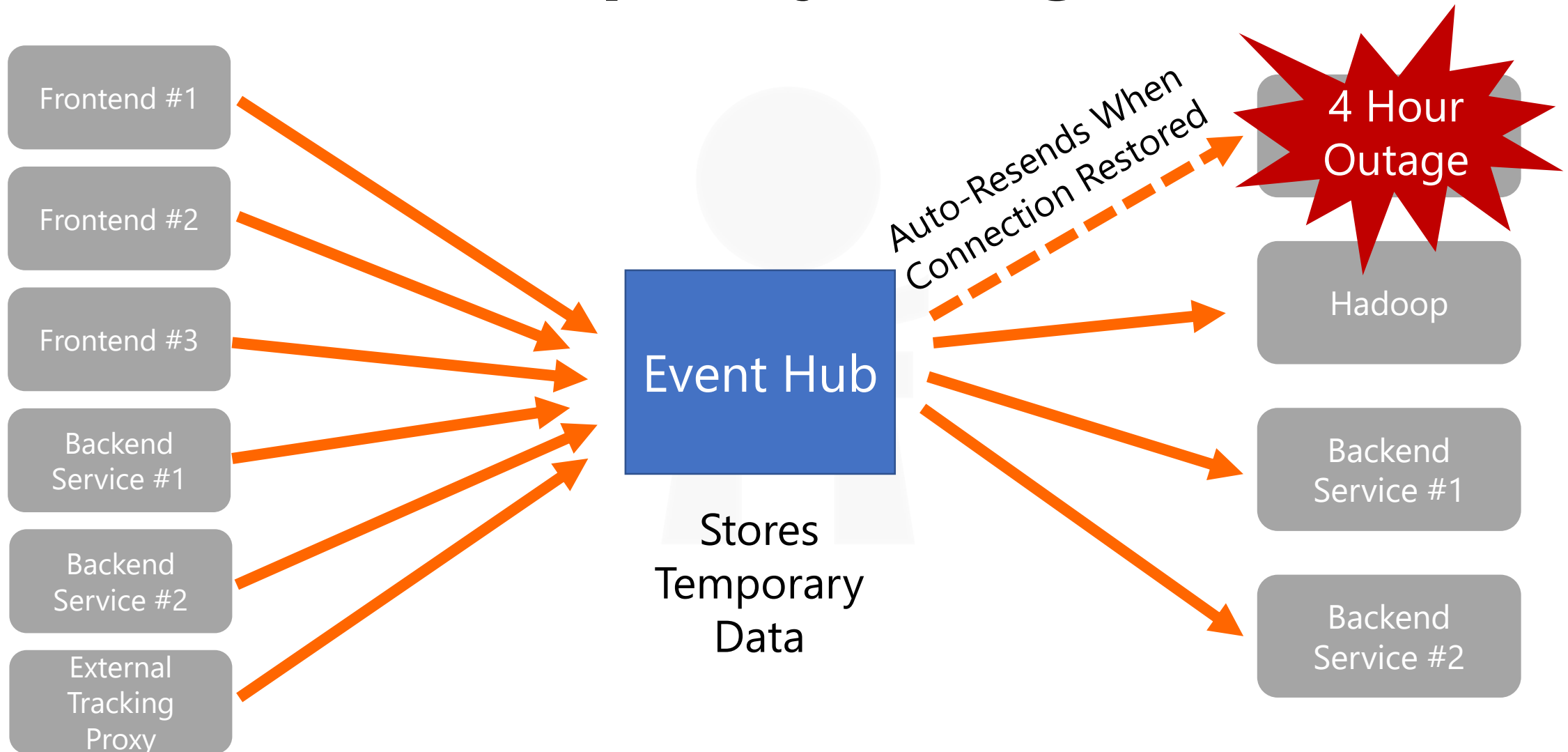
Popular Event Brokers



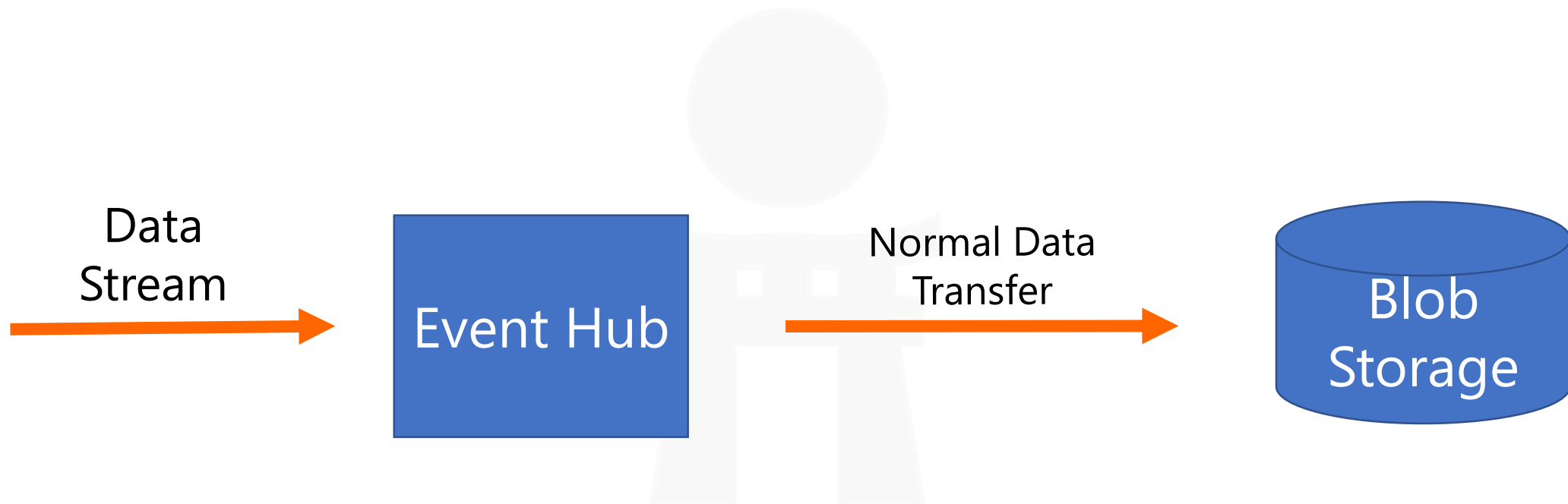
Event Hub for IoT: Big Data Ingestion



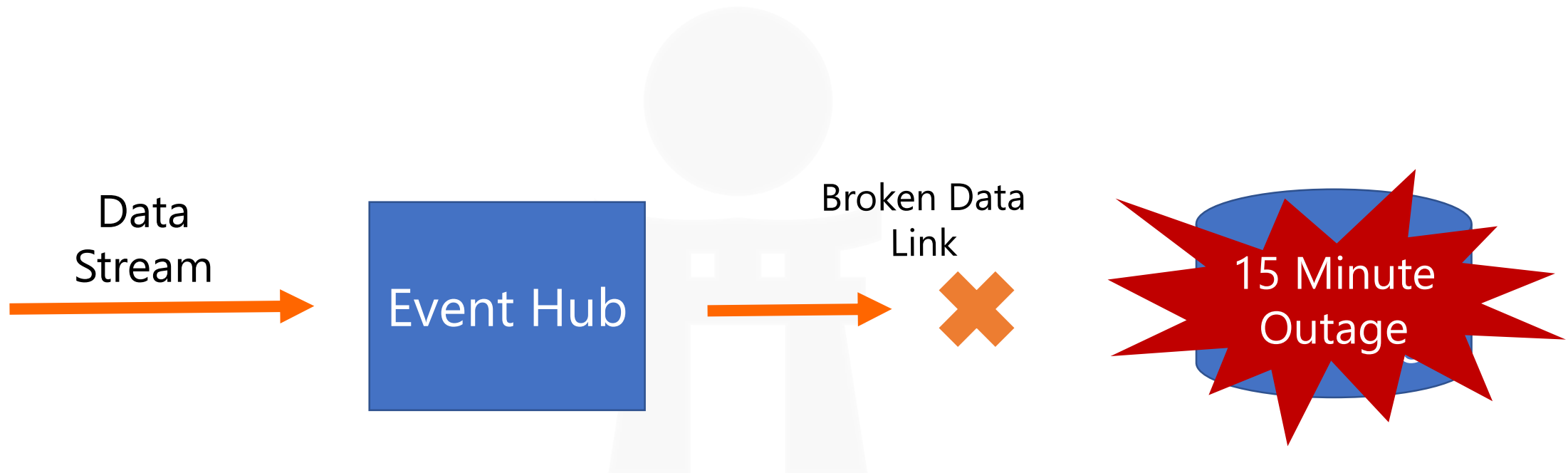
Temporary Storage



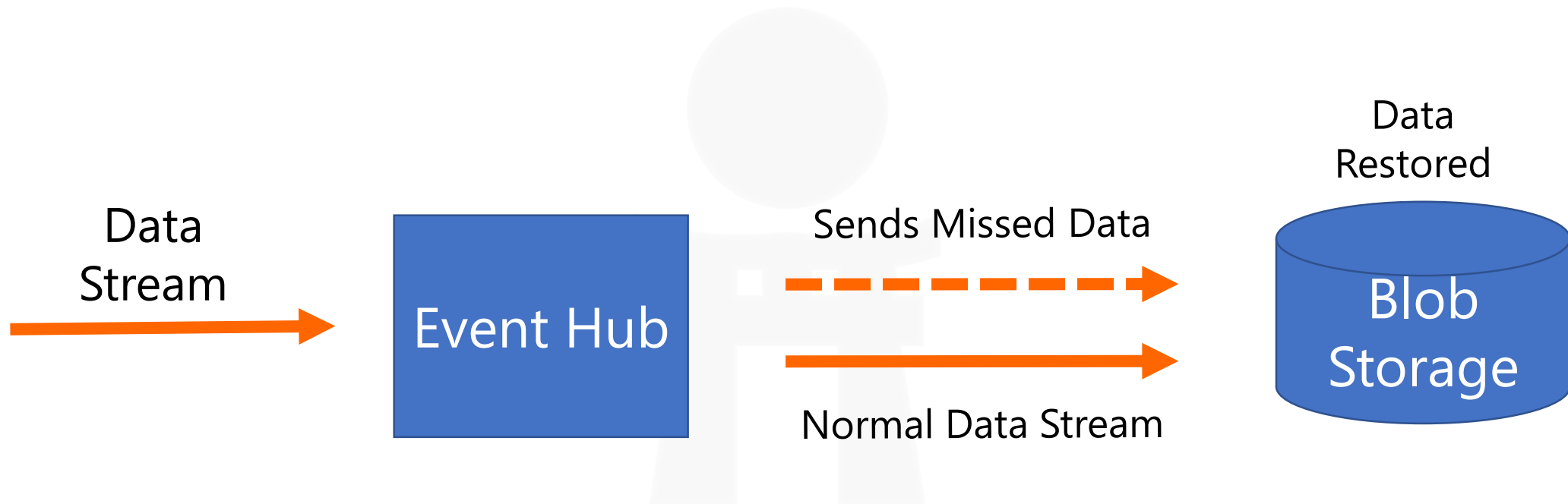
Demo: Normal Scenario



Demo: Output Downage



Demo: Output Restored

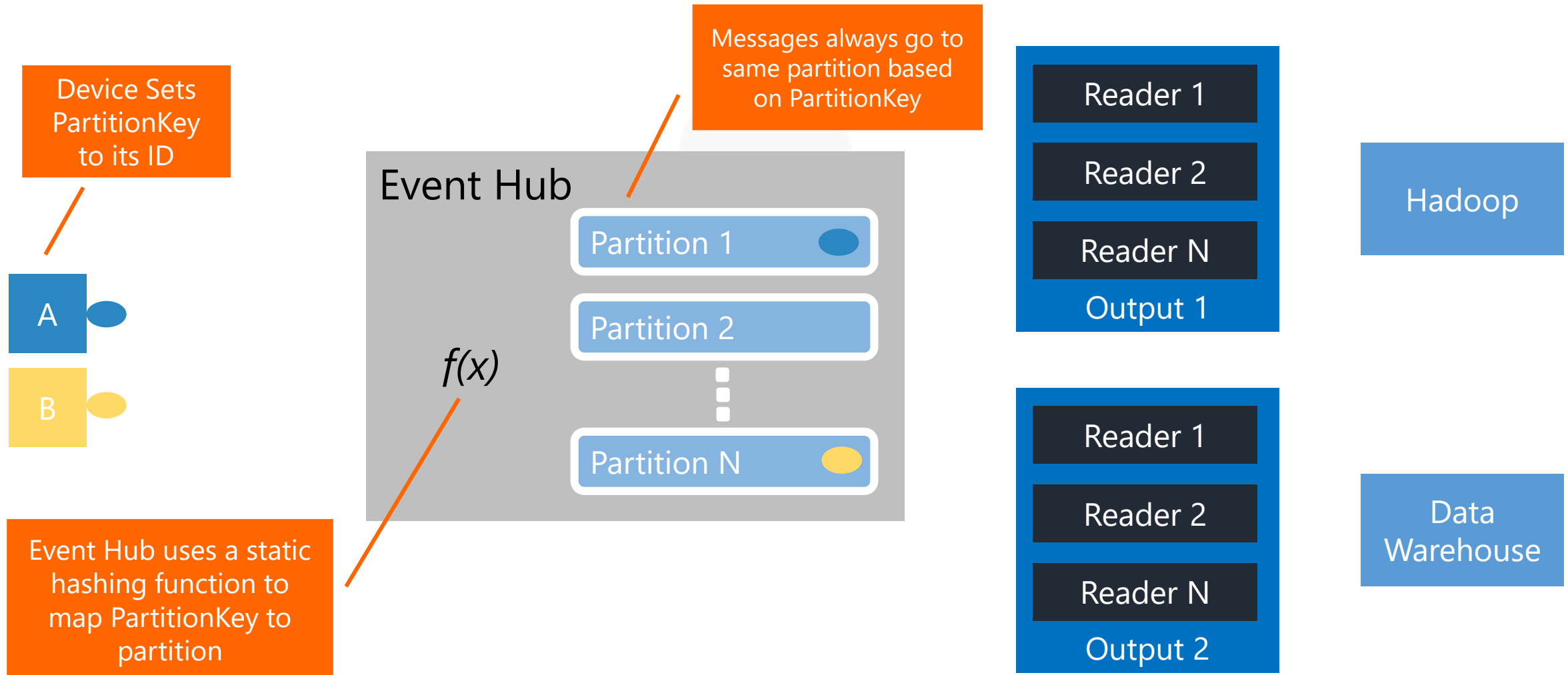


The Post Office



- Tracks address changes
- Tries again tomorrow if send failed
- Holds packages in short term
 - Too many failed deliveries
 - Vacations
- Reduces complexity through specialization

Event Hub, Stream Management



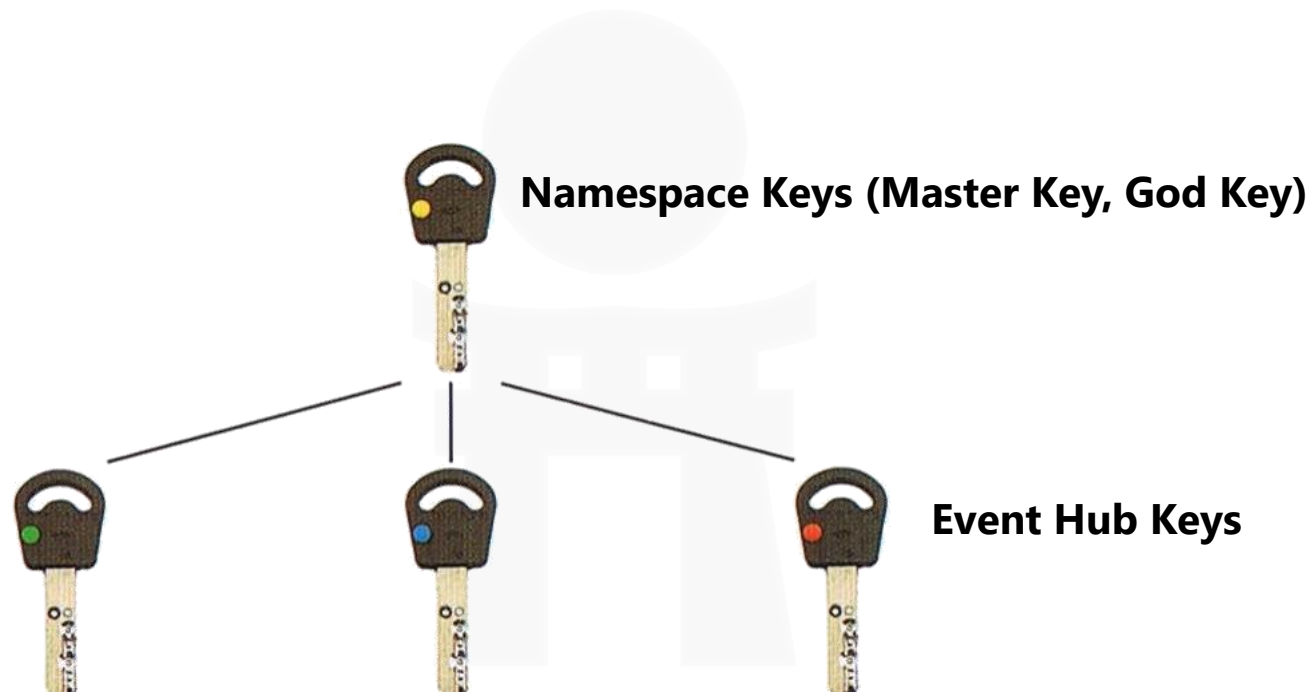
Service Bus Namespace

Service Bus Namespace

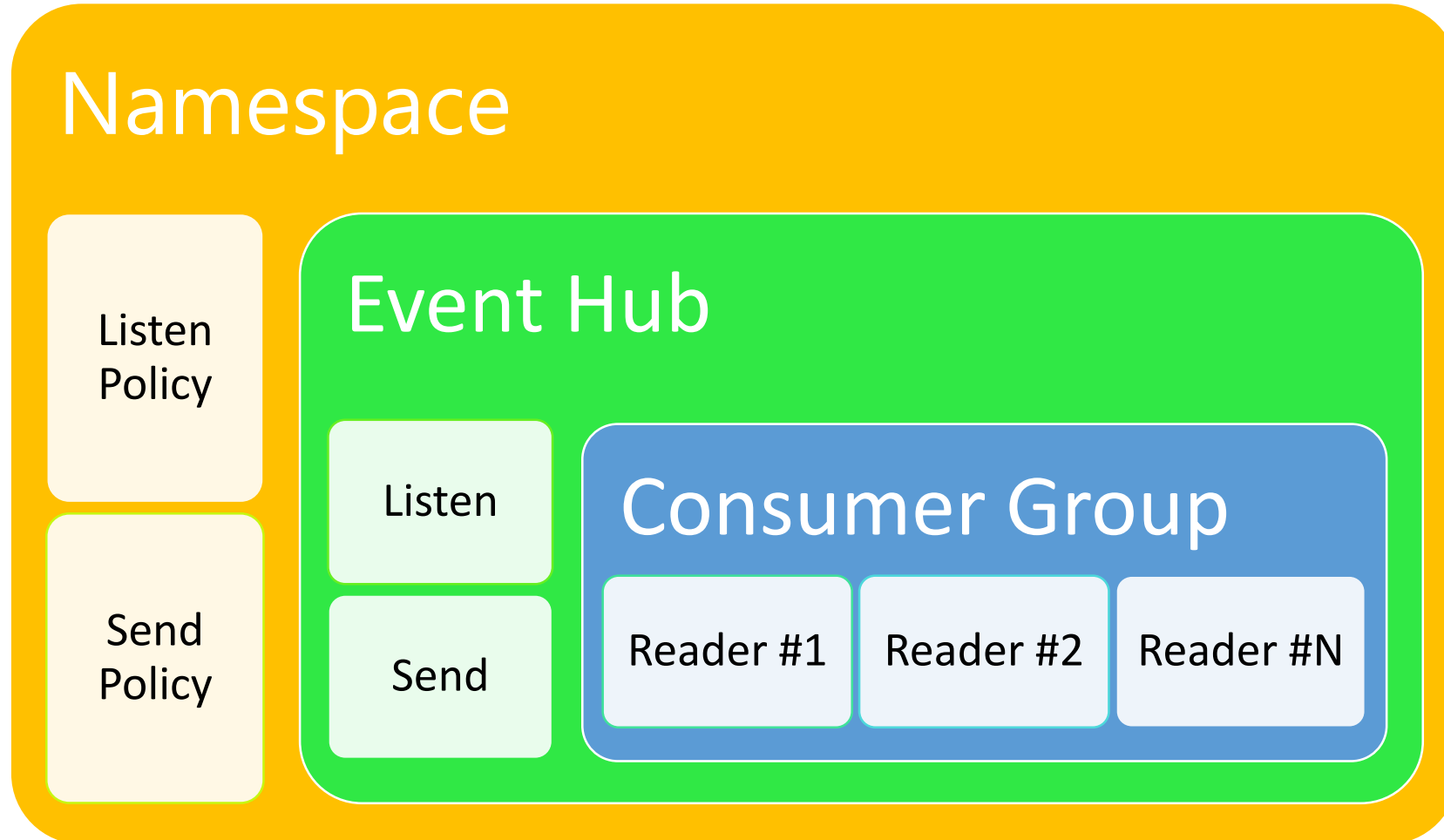
Event Hub 1

Event Hub 2

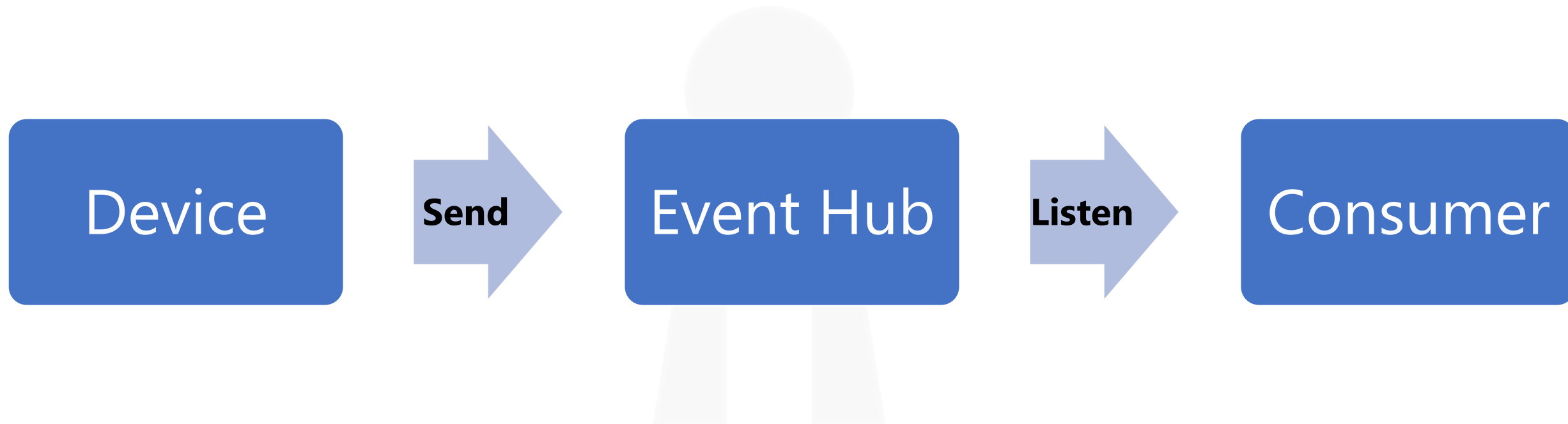
Access Rights, Policy, Keys



Access Rights



Access Rights



HANDS-ON LAB



Credit Card Transactions (swipes)



- Credit card transactions are usually done in batch as an end-of-the-day send.
- Stream process for insights now.
- US mainland transactions



Streaming to Event Hub

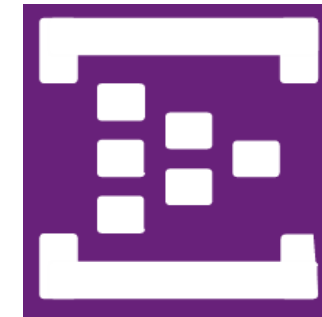


**Credit Card
Reader
(Synthetic)**

Swipes



**Message Broker
(DataScienceDojo's
Webpage)**



**Data Ingestor
(Azure Event Hub)**

The Data

```
{
  "swipe_date": "2015-05-22T20:16:27.122Z",
  "transaction_id": 3127484,
  "card_type": "VISA",
  "card_number": "4913419738164560",
  "expiration_month": "02",
  "expiration_year": "18",
  "cvv_code": "520",
  "user_id": "972288",
  "user_gender": "male",
  "user_first_name": "Alexander",
  "user_last_name": "Hamilton",
  "merchant": "McDonald's",
  "transaction_amount": 13.64,
  "balance": 336.48,
  "merchant_fee": .5,
  "swipe_city": "New York",
  "swipe_state": "New York",
  "swipe_city_state": "New York, NY",
  "InstanceNo": 1
}
```


The Streamer

- <http://demos.datasciencedojo.com/app/credit-card-streamer/>


Credit Card Streamer

This app will simulate the kind of data streams that banks would encounter, credit card swipe data. The app will generate synthetic data from a credit card transaction (swipe) and pushes it into a given Azure Event Hub as a JSON. The application logic for this app is written entirely in JavaScript so the speed and interval of the transactions is dependent on the processing power of the user device.


Event Hub Credentials

Event Hub Name (Need help? PDF Guide) 

field required

Service Bus Namespace (Need help? PDF Guide) 


field required


Shared Access Policy Name (Need help? PDF Guide) 

field required

Output Preview

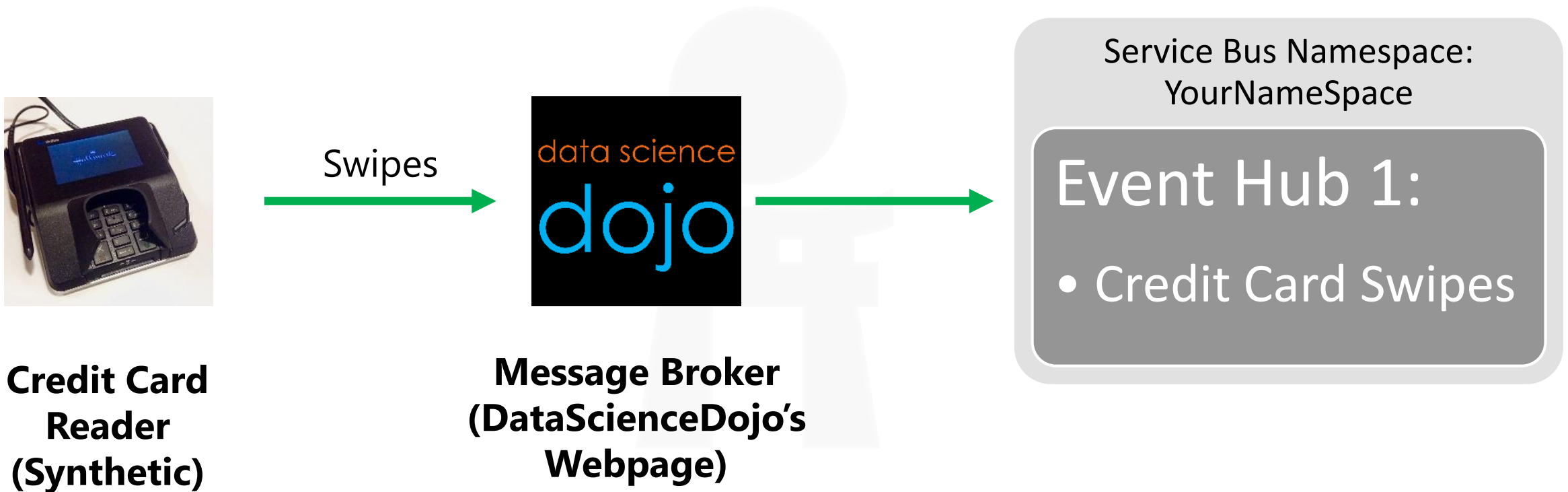
Display Format (Data is still sent as a JSON):

JSON 

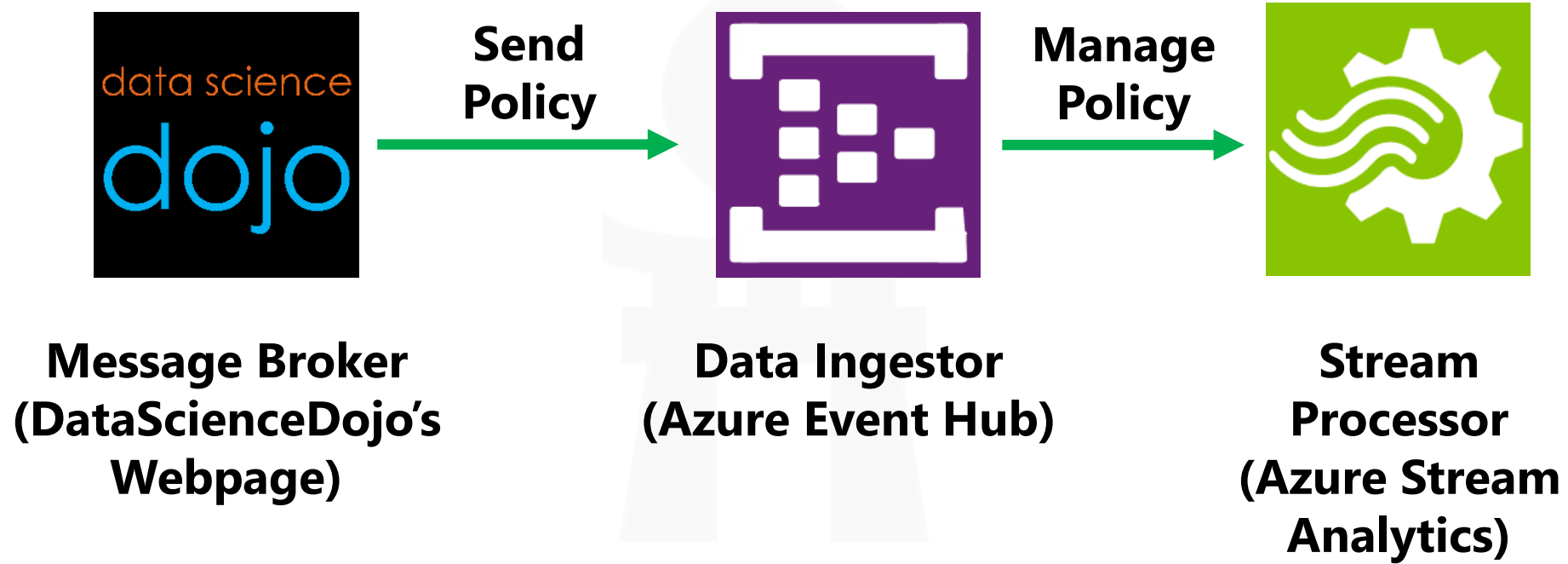
List 

Successfully loaded database. Ready to simulate data.

Inside the Event Hub



Setting Policies



QUESTIONS

A faint, light gray background illustration of a stylized human figure. The figure has a circular head and a rectangular body. It is holding a large, light gray question mark in front of its chest. The figure's legs are also rectangular, and it appears to be standing on a small base.



Stream Processing

Popular Up and Coming Event Processors



Google DataFlow



Azure Stream Analytics



Amazon Kinesis



DEMO

Credit Card Transactions (swipes)



- Credit card transactions are usually done in batch as an EOTD send.
- Stream process for insights now.
- US mainland transactions



Previously...



**Credit Card
Reader
(Synthetic)**

Swipes



**Message Broker
(DataScienceDojo's
Webpage)**



**Data Ingestor
(Azure Event Hub)**


The Streamer


- <http://demos.datasciencedojo.com/app/credit-card-streamer/>


Credit Card Streamer

This app will simulate the kind of data streams that banks would encounter, credit card swipe data. The app will generate synthetic data from a credit card transaction (swipe) and pushes it into a given Azure Event Hub as a JSON. The application logic for this app is written entirely in JavaScript so the speed and interval of the transactions is dependent on the processing power of the user device.

Event Hub Credentials


Event Hub Name (Need help? PDF Guide) 

Service Bus Namespace (Need help? PDF Guide) 

Shared Access Policy Name (Need help? PDF Guide) 

Output Preview

Display Format (Data is still sent as a JSON):

JSON </>
List 


```
Successfully loaded database. Ready to simulate data.
```

The Data

```
{  
  "swipe_date":"2015-05-22T20:16:27.122Z",  
  "transaction_id":3127484,  
  "card_type":"VISA",  
  "card_number":"4913419738164560",  
  "expiration_month":"02",  
  "expiration_year":"18",  
  "cvv_code":"520",  
  "user_id":"972288",  
  "user_gender":"male",  
  "user_first_name":"Alexander",  
  "user_last_name":"Hamilton",  
  "merchant":"McDonald's",  
  "transaction_amount":13.64,  
  "balance":336.48,  
  "merchant_fee":.5,  
  "swipe_city":"New York",  
  "swipe_state":"New York",  
  "swip_city_state":"New York, NY",  
  "InstanceNo":1  
}
```

Data vs Events

```
{  
  "swipe_date":"2015-05-22T20:16:27.122Z",  
  "transaction_id":3127484,  
  "card_type":"VISA",  
  "card_number":"4913419738164560",  
  "expiration_month":"02",  
  "expiration_year":"18",  
  "cvv_code":"520",  
  "user_id":"972288",  
  "user_gender":"male",  
  "user_first_name":"Alexander",  
  "user_last_name":"Hamilton",  
  "merchant":"McDonald's",  
  "transaction_amount":13.64,  
  "balance":336.48,  
  "merchant_fee":.5,  
  "swipe_city":"New York",  
  "swipe_state":"New York",  
  "swipe_city_state":"New York, NY",  
  "InstanceNo":1  
}
```



An event is just data
with a timestamp



Inside the Event Hub



**Credit Card
Reader
(Synthetic)**

Swipes



**Message Broker
(DataScienceDojo's
Webpage)**

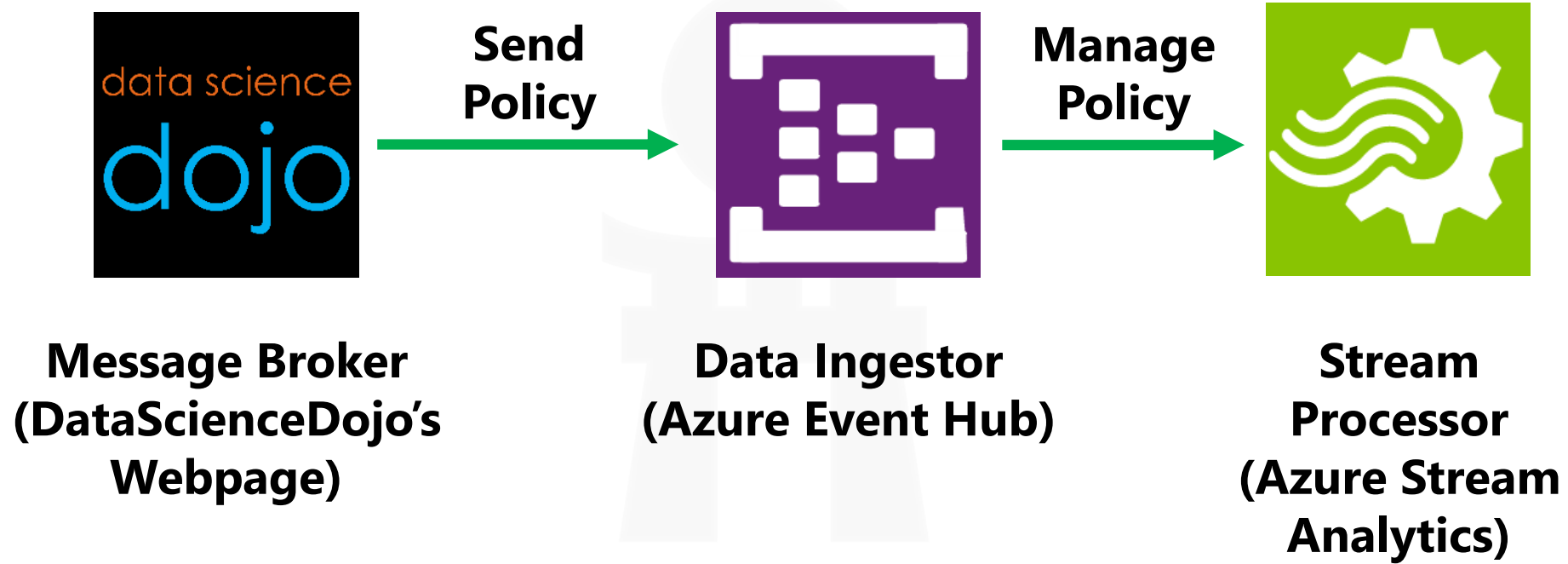


Service Bus Namespace:
YourNameSpace

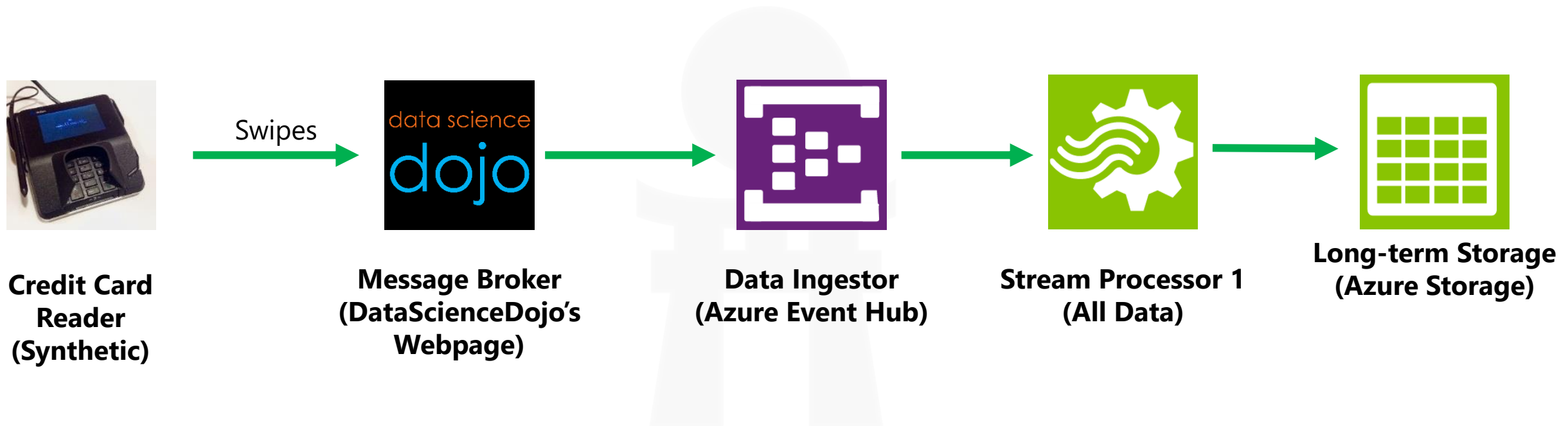
Event Hub 1:

- Credit Card Swipes

Setting Policies



With Stream Processor



More Processors

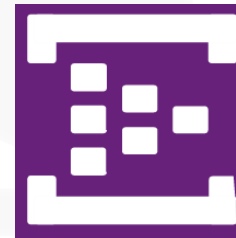


**Credit Card Reader
(Synthetic)**

Swipes
→



**Message Broker
(DataScienceDojo's
Webpage)**



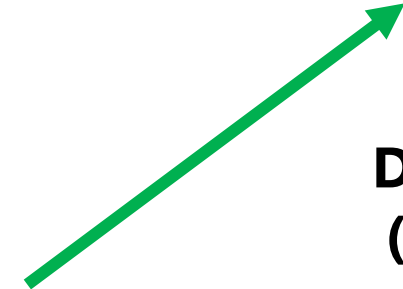
**Data Ingestor
(Azure Event Hub)**



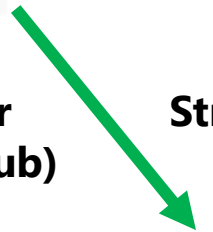
**Stream Processor 1
(All Data)**



**Long-term Storage
(Azure Storage)**



**Dashboard
(PowerBI)**



**Stream Processor 2
(Filter/Aggregate)**



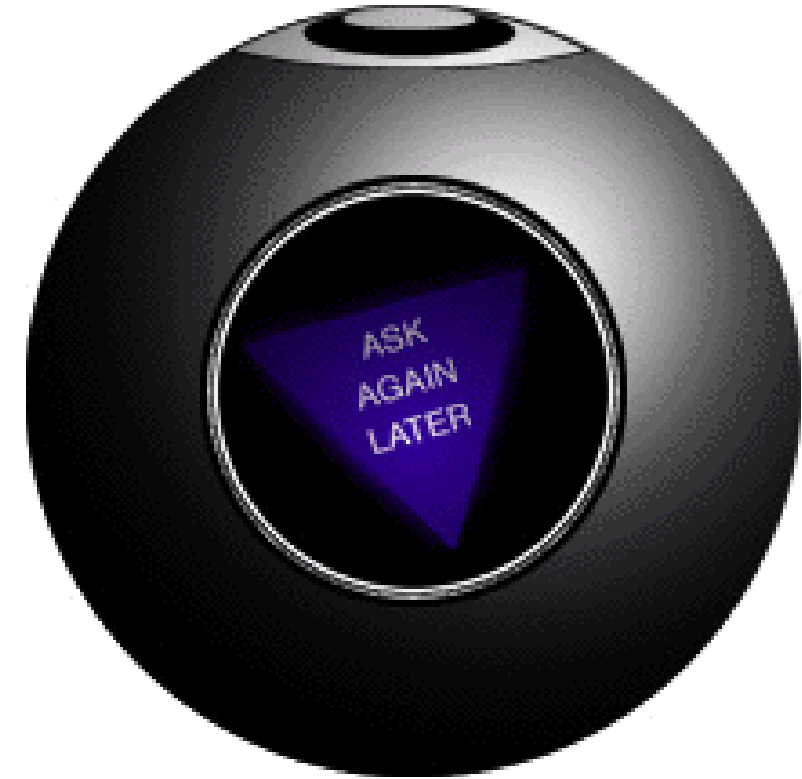
**Anomaly Detection
Event Hub**

SQL with Data at Rest

- **Question** "Show me VISA transactions from last month."
- **Answering with a relational database**
No problem! Here you go!
- **SELECT ***
FROM credit_db
WHERE card_type **like** 'VISA'

SQL Data in Motion

- **Different Question** "Show me VISA transactions in the past 2 minutes."
- **Answering with a relational database**
I'm not ready yet... Ask again later... Or tomorrow (after batch)...
- **Not a great solution...**



Temporal System

- Every event is a point in time, and thus must come with a timestamp
 - Remember how relational DBs need a PK? Temporal systems need a timestamp as its unique identifier.
 - Temporal integrity and referential integrity
- Stream Analytics can append your events with a timestamp (bad practice if standalone)
 - The default timestamp will be when the event enters Stream Analytics
 - Can be skewed by network and hardware latency, or legacy processing
- Users can define application time stamps with the `TIMESTAMP BY` clause

Which Timestamp?

```
{
  "swipe_date": "2015-05-21T22:47:55.0770000Z",
  "transaction_id": 222301082,
  "card_type": "VISA",
  "card_number": "40265691066025560",
  "expiration_month": "06",
  "expiration_year": "22",
  "cvv_code": "3310",
  "user_id": "690548",
  "user_gender": "male",
  "user_first_name": "Caden",
  "user_last_name": "Hatton",
  "merchant": "Macy's",
  "transaction_amount": 4.98,
  "balance": 7223.9,
  "merchant_fee": 0.5,
  "swipe_city": "New York",
  "swipe_state": "New York",
  "swipe_city_state": "New York, NY",
  "InstanceNo": 1,
  "EventProcessedUtcTime": "2015-05-21T22:47:50.0879821Z",
  "PartitionId": 3,
  "EventEnqueuedUtcTime": "2015-05-21T22:47:49.9850000Z"
}
```

Time of event

Time processed by
stream processor

Time entered broker

Same Event...

```
{  
  "swipe_date":"2015-05-21T22:47:55.0770000Z",  
  "EventProcessedUtcTime":"2015-05-21T22:47:50.0879821Z",  
  "EventEnqueuedUtcTime":"2015-05-21T22:47:49.9850000Z"  
}
```

According to these timestamps, the event happened 5 seconds AFTER the event was processed and queued.

- How can that be?
- The event was not confined to the physical laws of space and time.

The clock on your device matters.

Azure Stream Query Language

- Show me transactions as they happen. Write it to a blob AND powerBI.

```
SELECT *  
INTO MyBlob  
FROM SwipeStream TIMESTAMP BY swipe_date;  
SELECT *  
INTO PowerBI  
FROM SwipeStream TIMESTAMP BY swipe_date;
```

StreamQL: Calculations

- What was our commission on each transaction?

SELECT

transaction_id,
merchant_fee / transaction_amount **AS** Commision

FROM SwipeStream

TIMESTAMP BY swipe_date

StreamQL: Filter Queries

- Show me only VISA transactions that made over \$5 revenue.

SELECT

swipe_date,
card_type,
merchant_fee **AS** revenue

FROM SwipeStream

TIMESTAMP BY swipe_date

WHERE card_type **LIKE** 'VISA'

AND merchant_fee **<** 5

SWIPE_DATE	CARD_TYPE	REVENUE
2015-05-21T2...	VISA	6.2
2015-05-21T2...	VISA	10.31
2015-05-21T2...	VISA	11.72
2015-05-21T2...	VISA	7.82
2015-05-21T2...	VISA	9.91
2015-05-21T2...	VISA	7.62
2015-05-21T2...	VISA	5.25

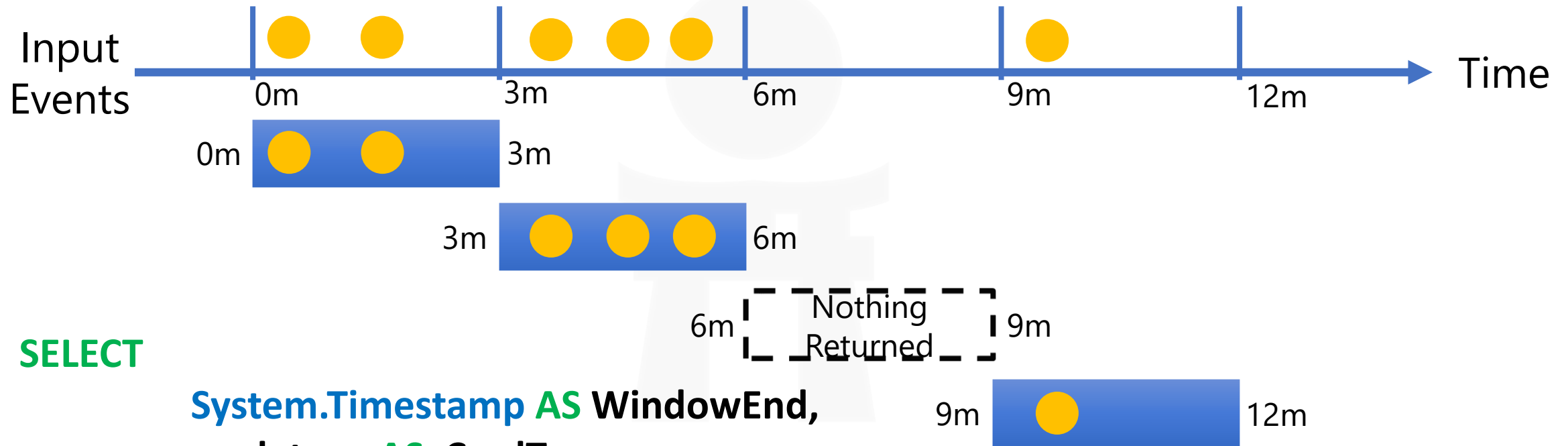
Temporal Questions

Count the number of transactions....

- When should the counting of transactions begin?
- When should the counting of transactions end?
- How long should the transactions be counted for?
- How often do transactions need to be counted?

Tumbling Window

How many transactions were made for each card type every 3 minute?



SELECT

System.Timestamp AS WindowEnd,

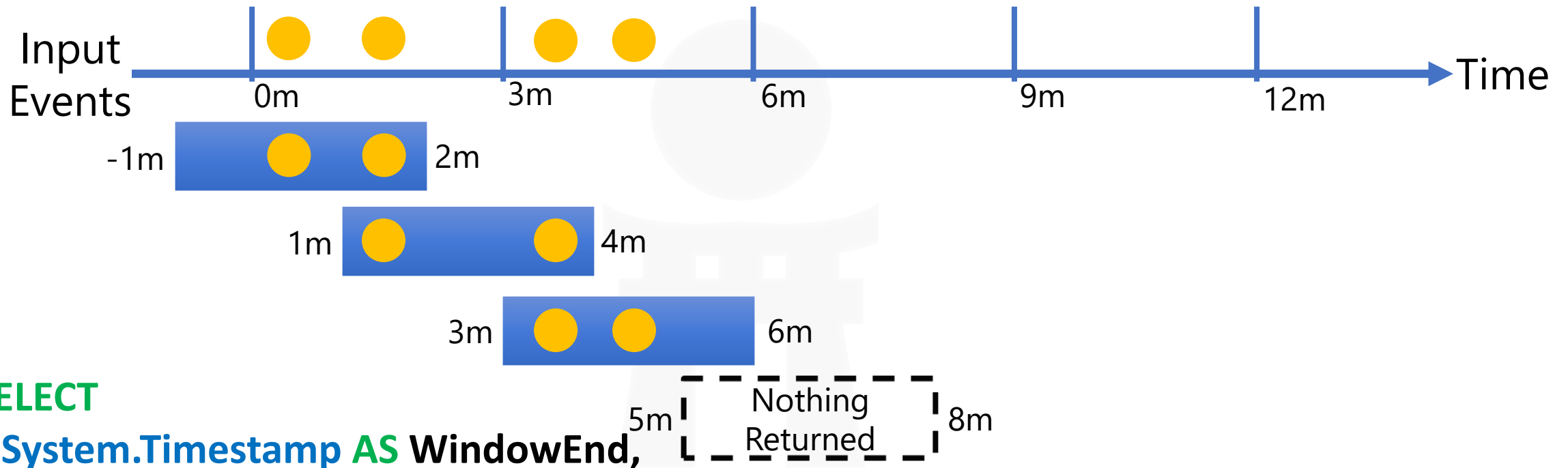
card_type AS CardType,

Count(*) AS Frequency

FROM SwipeStream **TIMESTAMP BY** swipe_date

GROUP BY TUMBLINGWINDOW(minute, 3), card_type

Hopping Window



SELECT

System.Timestamp **AS** WindowEnd,

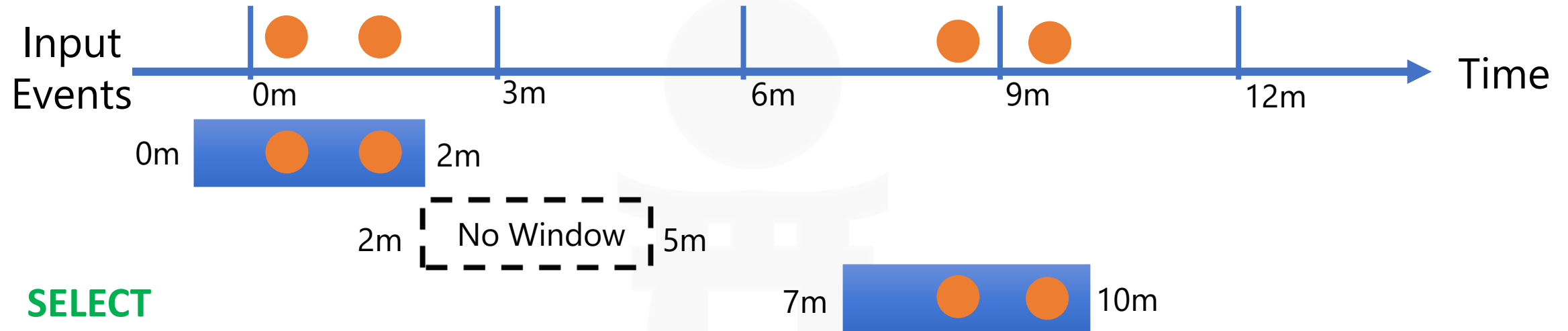
card_type **AS** CardType,

Count(*) **AS** Frequency

FROM SwipeStream **TIMESTAMP BY** swipe_date

GROUP BY **HoppingWindow**(minute, 3, 2), card_type

Sliding Window

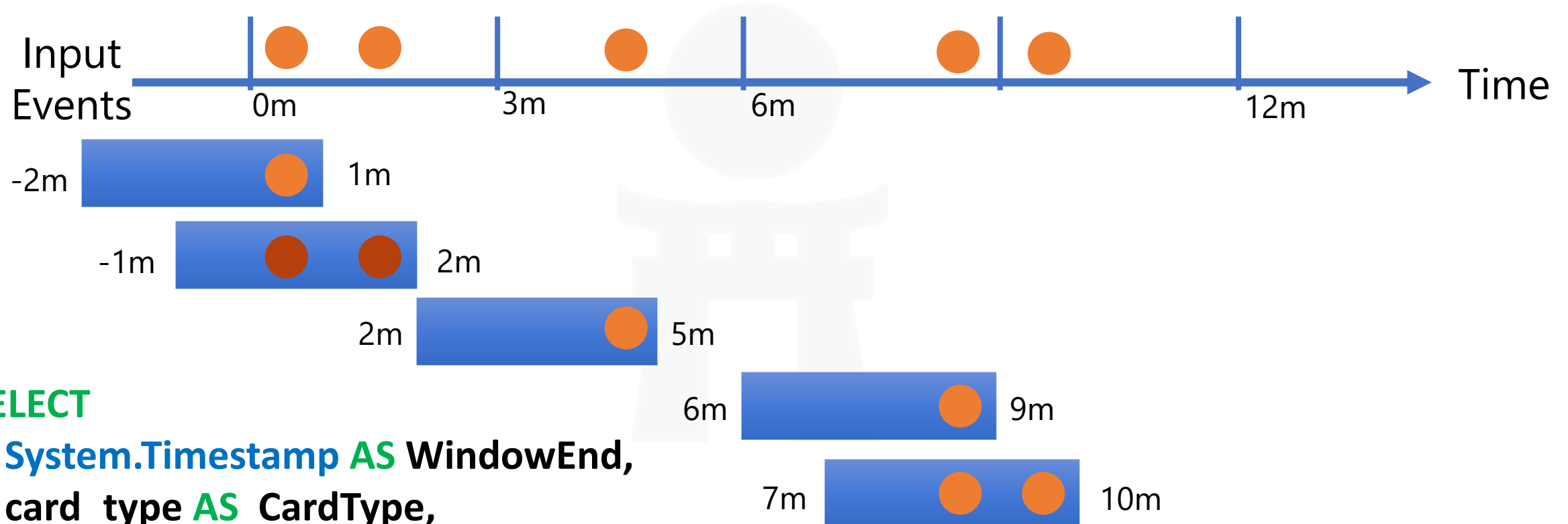


SELECT

System.Timestamp **AS** WindowEnd,
card_type **AS** CardType,
Count(*) **AS** Frequency

FROM SwipeStream **TIMESTAMP BY** swipe_date
GROUP BY **SlidingWindow**(minute, 3), card_type
HAVING Frequency > 2

Sliding Window: Without 'Having' Clause



SELECT

System.Timestamp **AS** WindowEnd,

card_type **AS** CardType,

Count(*) **AS** Frequency

FROM SwipeStream **TIMESTAMP BY** swipe_date

GROUP BY **SlidingWindow**(minute, 3), card_type

Sum Aggregation

- How much revenue is being accumulated from merchants every 3 minutes?

SELECT

System.Timestamp AS WindowEnd,

Sum(merchant_fee) AS IntervalRevenue

FROM SwipeStream TIMESTAMP BY swipe_date

GROUP BY TUMBLINGWINDOW(minute, 3), WindowEnd

Sum Aggregation: With Filtering

- Which 3-minute time interval made more than \$10?

```
SELECT
  System.Timestamp AS WindowEnd,
  Sum(merchant_fee) AS IntervalRevenue
FROM SwipeStream TIMESTAMP BY swipe_date
GROUP BY TUMBLINGWINDOW(minute, 3), WindowEnd
Having IntervalRevenue > 10
```


Descriptive Statistics

- Generate descriptive statistics for revenue every 3 minutes (car count, min, max, average, standard deviation, and total revenue).

SELECT

```
System.Timestamp AS WindowEnd,  
count(merchant_fee) AS CarCount,  
min(merchant_fee) AS MinRev,  
max(merchant_fee) AS MaxRev,  
avg(merchant_fee) AS AvgRev,  
stdev(merchant_fee) AS VarRev,  
sum(merchant_fee) AS TotalRev
```

FROM SwipeStream **TIMESTAMP BY** swipe_date

GROUP BY TUMBLINGWINDOW(minute, 3)

DateDiff and Time

- What is the duration between the first transaction in the window and the last transaction in the window? What was the duration between the first transaction in the window and the end of the window?

SELECT

System.Timestamp AS WindowEnd,

count(*) AS Frequency,

datediff(second, min(swipe_date), max(swipe_date)) AS FirstLastDuration,

datediff(second, min(swipe_date), System.Timestamp) AS FirstEndDuration

FROM SwipeStream**TIMESTAMP BY** swipe_date

GROUP BY TUMBLINGWINDOW(minute, 3)

Joining Stream with Reference Data

- Say we had a list of stolen credit card numbers. Let's run each transaction against this list and get the locations.

SELECT

```
SwipeStream.swipe_date as SwipeTime,  
SwipeStream.card_number as CardNumber,  
SwipeStream.merchant as Store,  
SwipeStream.swipe_city_state as Location,  
StolenList.Stolen as Stolen
```

FROM SwipeStream **TIMESTAMP BY** swipe_date

JOIN StolenList

ON SwipeStream.card_number = StolenList.card_number

WHERE StolenList.Stolen = '1'

Joining Streams, Temporally

- How long did it take for each transaction to get approval from the bank?
 - Joining on events through time
 - JOIN operator requires specifying a temporal wiggle room describing an acceptable time difference between the joined events
 - If two transactions occurred within the same join interval, then consider them the same event.

Joining Streams

- How long did it take for each transaction to get approval from the bank?

SELECT

swipe.transaction_id
swipe.swipe_date,
bank.approval_time,

DATEDIFF (**second**, swipe.swipe_date, bank. approval_time) **AS** DurationInSeconds

FROM SwipeStream **AS** swipe **TIMESTAMP BY** swipe_date

JOIN BankStream **AS** bank **TIMESTAMP BY** approval_time

ON (swipe.transaction_id = bank.transaction_id)

AND DATEDIFF (**minute**, swipe, bank) **BETWEEN** 0 AND 15

Joining Streams, by Window

- What was the average time that it took for transactions to get approved every 3 minutes?

SELECT

System.Timestamp AS WindowEnd,

avg(DATEDIFF (second, swipe.swipe_date, bank.approval_time)) AS ApprovalTime

FROM SwipeStream AS swipe TIMESTAMP BY swipe_date

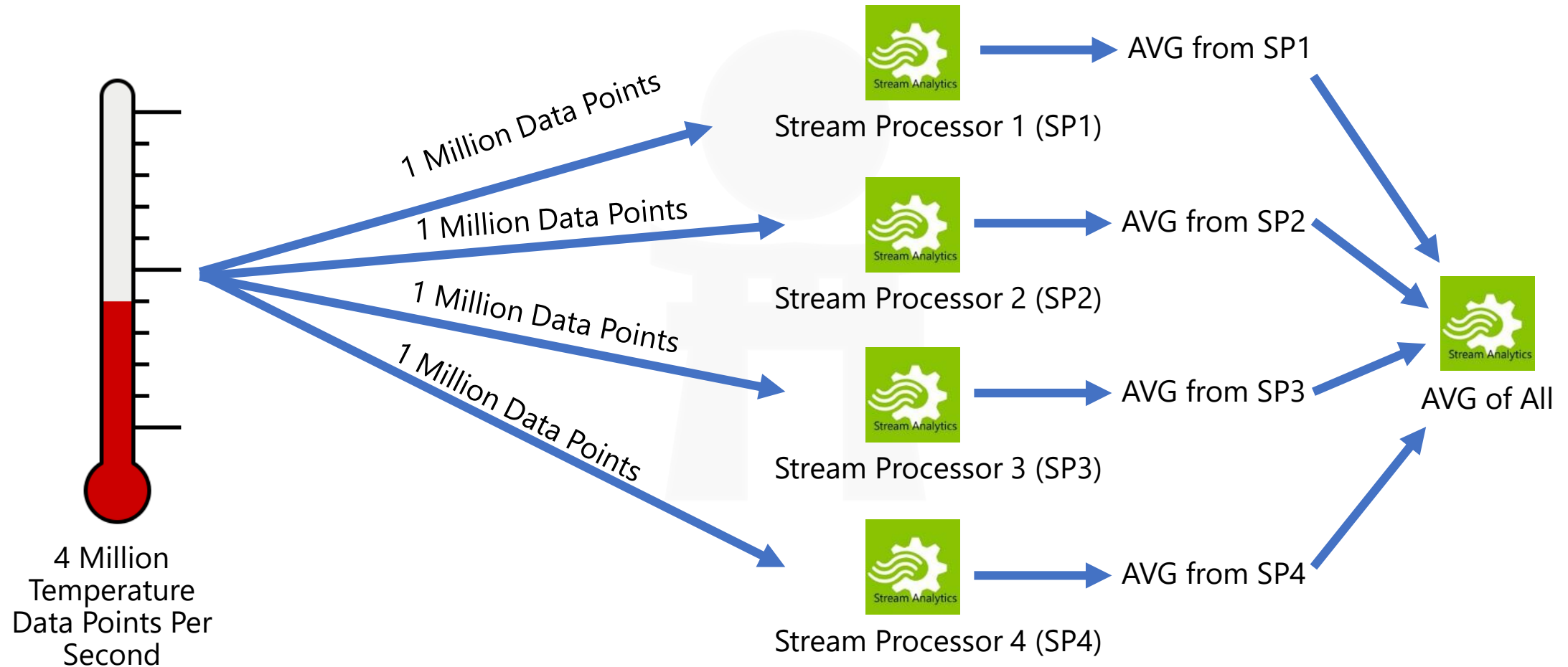
JOIN BankStream AS bank TIMESTAMP BY approval_time

ON (swipe.transaction_id = bank.transaction_id)

AND DATEDIFF (minute, swipe, bank) BETWEEN 0 AND 15

Group by TumblingWindow(minute, 3)

Average of Average Approximations



Built-In Functions And Supported Types

Aggregate functions

Count, Min, Max, Avg, Sum

Scalar functions

Cast

Date and time

**Datetime, Datetimepart, Day, Month, Year,
Datediff, Dateadd**

String

Len, Concat, Charindex, Substring, Patindex



QUESTIONS