Streaming Processing with Azure Stream Analytics

Data Science Dojo

datasciencedojo
unleash the data scientist in you

IN 60 SECONDS...

1 **NEW** DEFINITION IS ADDED ON **urban** DICTIONARY

1,600+ **READS ON Scribd.**

13,000+HOURS **MUSIC** STREAMING ON **PANDORA**

12,000+ **NEW ADS POSTED ON** craigslist

New Craigslist Ads

370,000+ MINUTES **VOICE CALLS ON skype**

98,000+ **TWEETS**

20,000+ **NEW POSTS ON tumblr.**

THE **LARGEST** SOCIAL READING PUBLISHING COMPANY!!

320+ **NEW twitter ACCOUNTS**

100+ **NEW Linkedin ACCOUNTS**

13,000+ **iPhone APPLICATIONS DOWNLOADED**

1 **associatedcontent NEW ARTICLE IS PUBLISHED**

THE WORLD'S **LARGEST** COMMUNITY CREATED CONTENT!!

QUESTIONS ASKED ON THE INTERNET...

100+ Answers.com 40+ **YAHOO!** ANSWERS

6,600+ **NEW PICTURES ARE UPLOADED ON flickr**

600+ **NEW VIDEOS** You Tube

50+ **WordPress DOWNLOADS**

25+ HOURS **TOTAL DURATION**

70+ **DOMAINS REGISTERED**

60+ **NEW BLOGS**

168 MILLION **EMAILS ARE SENT**

694,445 **SEARCH QUERIES**

1,700+ **Firefox DOWNLOADS**

695,000+ **facebook STATUS UPDATES**

=125+ **PLUGIN DOWNLOADS**

1,500+ **BLOG POSTS**

Google

Google Search

79,364 **WALL POSTS**

510,040 **COMMENTS**

Introducing Big Data
Continued
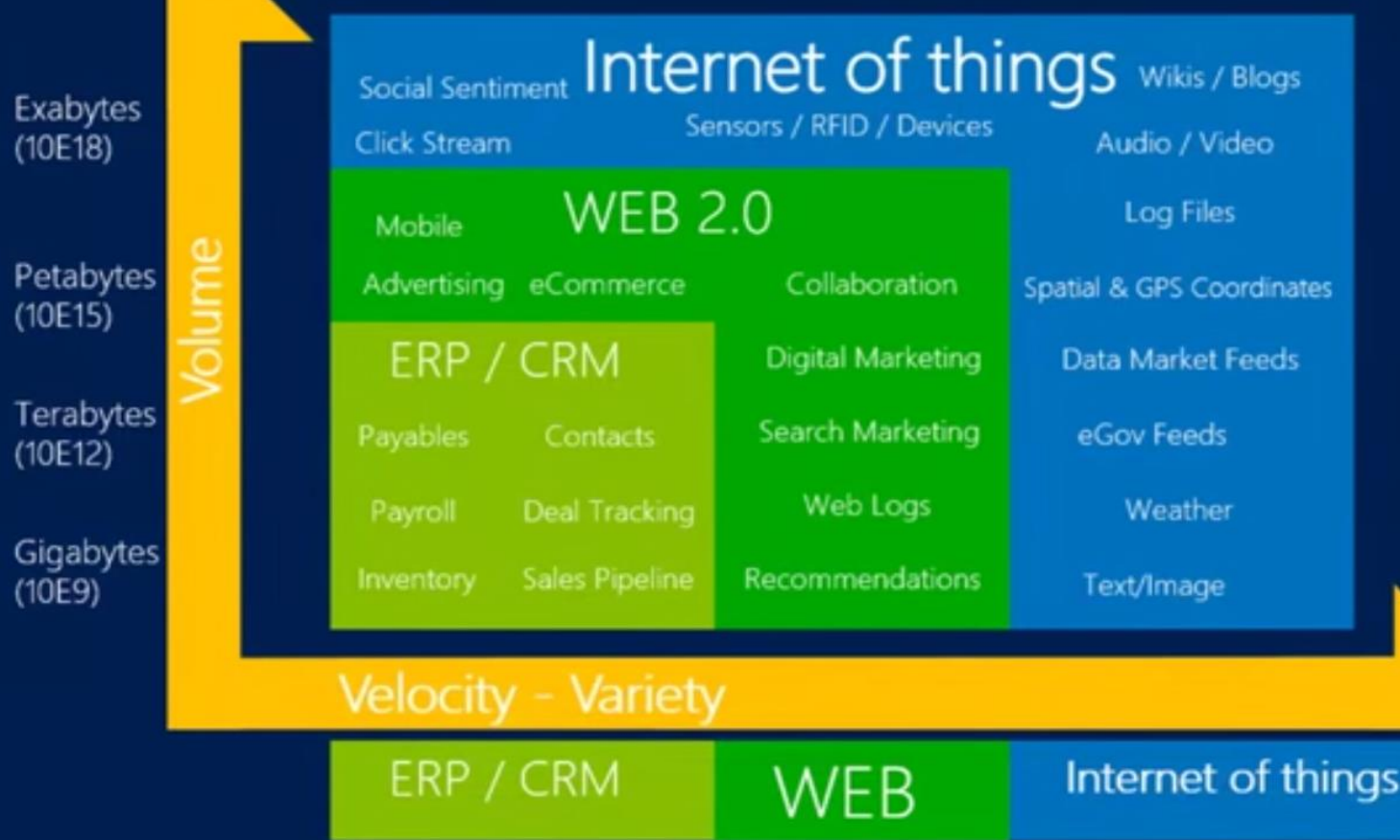
Defining Real-time

Within seconds...

      or...

Within minutes...

      of an event occurring

Up to 2 hours

Timeliness of Information

What was trending in the past 5 minutes?

CHILD ABDUCTION
BLUE HONDA CIVIC
CA LIC 5TIC261

Amber alert car detected!

A tornado will form in the next 30 minutes.

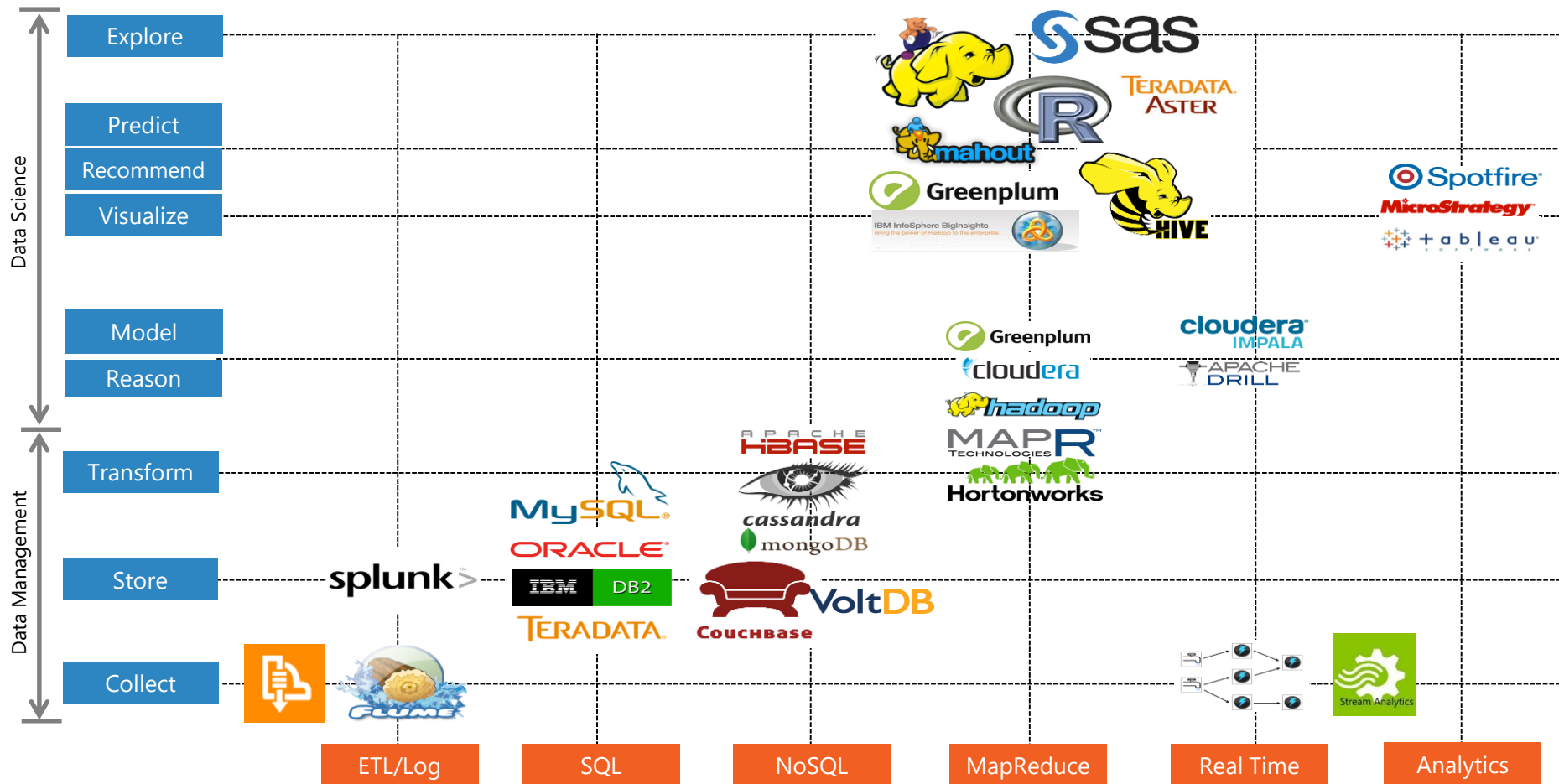datasciencedojo
unleash the data scientist in you

Timeliness of Information

A stock is going to crash in 20 minutes.

A fire is about to start in your house.
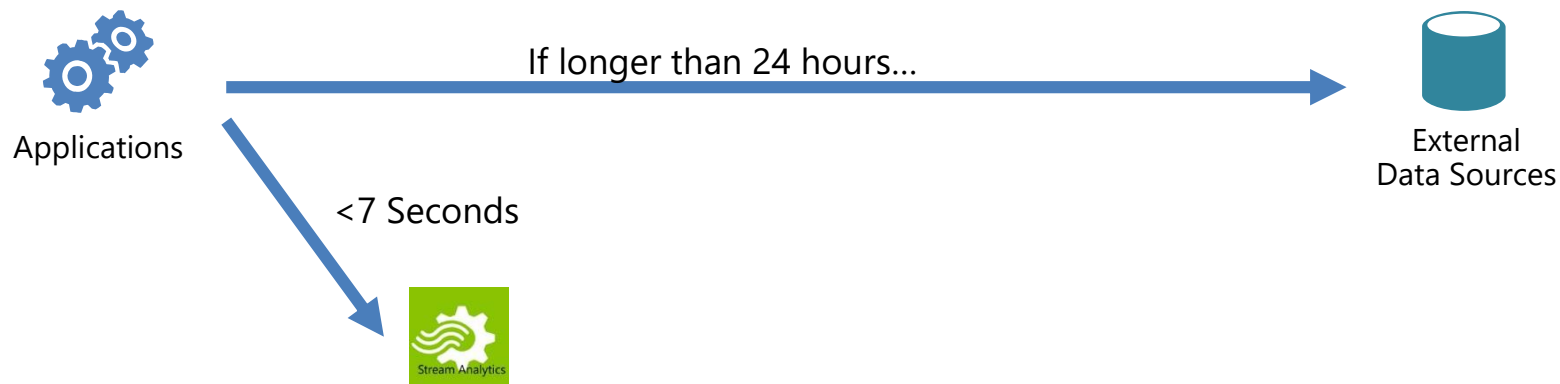
The power grid will overload in 2 minutes.

datasciencedojo
unleash the data scientist in you

Typical Event Processing

Producers → Collection → Event Ingestor (broker) ↔ Transformation ↔ Long Term Storage → Presentation And Action

Applications

Devices

Cloud Gateways (WebAPIs)

Field Gateways

Scalable Event Broker

Event Hub

Real-Time Analytics

External Data Sources

Web/Thick Client Dashboards

Search And Query

Data Analytics

ETL Timespan
(Extract, Transform, Load)

datasciencedojo
unleash the data scientist in you

Typical Event Processing

Producers	→	Collection	→	Event Ingestor (broker)	◆	Transformation	◆	Long Term Storage	→	Presentation And Action

Applications

Cloud Gateways (WebAPIs)

Scalable Event Broker

Real-time Analytics

External Data Sources

Web/Thick Client Dashboards

Devices

Field Gateways

Search And Query

Stream Analytics

Data Analytics

datasciencedojo
unleash the data scientist in you

When to use Stream Processors

Applications

If longer than 24 hours...

External
Data Sources

<7 Seconds

Stream Analytics

datasciencedojo
unleash the data scientist in you

ETL Should Still Happen

Stream Processing

ETL

- Stream Processing is only icing on the cake.
- It can, but should not replace a company's normal ETL cycle.

datasciencedojo
unleash the data scientist in you

Popular Up and Coming Event Processors

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) 🖼

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.

datasciencedojo
unleash the data scientist in you

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

An event is just data
with a timestamp

```
{
    "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,
```

Inside the Event Hub

Credit Card Reader (Synthetic)

Swipes

Message Broker (DataScienceDojo's Webpage)

Service Bus Namespace: YourNameSpace

Event Hub 1:

- Credit Card Swipes

datasciencedojo
unleash the data scientist in you

Setting Policies

Message Broker (DataScienceDojo's Webpage) — Send Policy → **Data Ingestor (Azure Event Hub)** — Manage Policy → **Stream Processor (Azure Stream Analytics)**
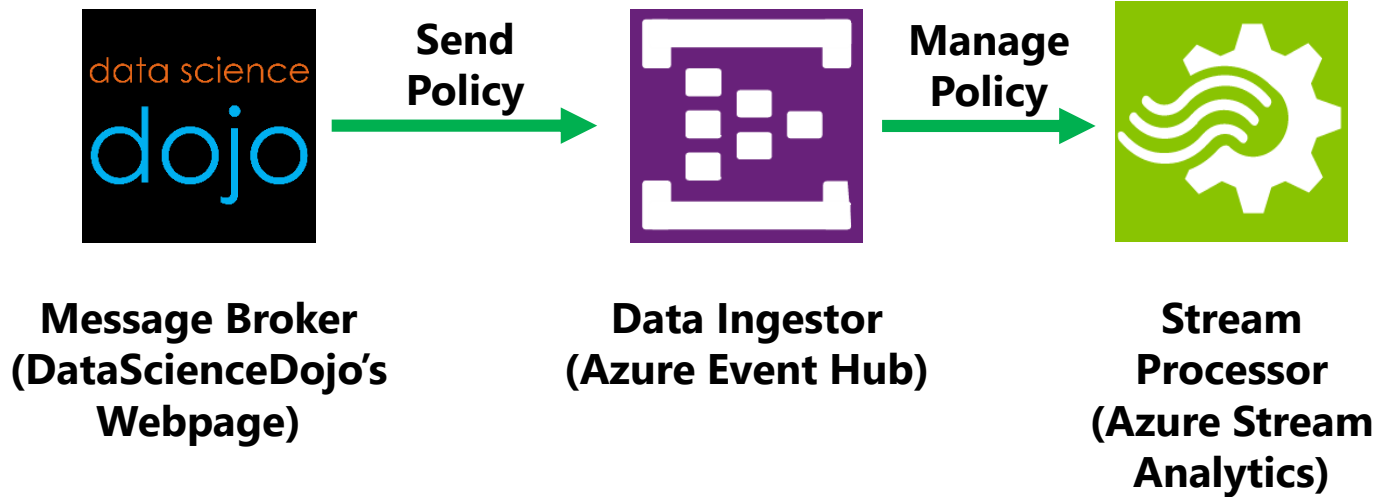
With Stream Processor

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)
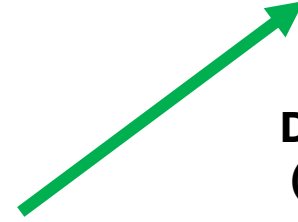
More Processors

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'

datasciencedojo
unleash the data scientist in you

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…**

datasciencedojo
unleash the data scientist in you

Azure Stream Query Language

- Queries through time
- Simple SQL dialect
  - Familiar – learning curve reduction
  - High-Level – expression of intent, not implementation
  - Maintainable – focus on the essentials of the problem

- Extended in natural ways
  to express temporal concepts
  - WINDOW – multiple kinds
    - Tumbling, hopping, sliding
  - TIMESTAMP BY, BETWEEN
  - DATEDIFF in joins
  - PARTITION BY for scale-out

```sql
WITH agg AS
(
    SELECT Avg(reading), Building
    FROM Temperature
    GROUP BY TumblingWindow(minute, 1), building
)
SELECT A1.Avg AS Old, A2.Avg AS New, A1.Building
FROM Agg A1 JOIN Agg A2
ON A1.Building = A2.Building
AND DATEDIFF(minute,A1,A2) BETWEEN 4.5 AND 5.5
WHERE
    (a1.avg < a2.avg - 10) OR (a1.avg > a2.avg+10)
```

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",
  "swip_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

datasciencedojo
unleash the data scientist in you

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.

datasciencedojo
unleash the data scientist in you

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

datasciencedojo
unleash the data scientist in you

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?

datasciencedojo
unleash the data scientist in you

Tumbling Window

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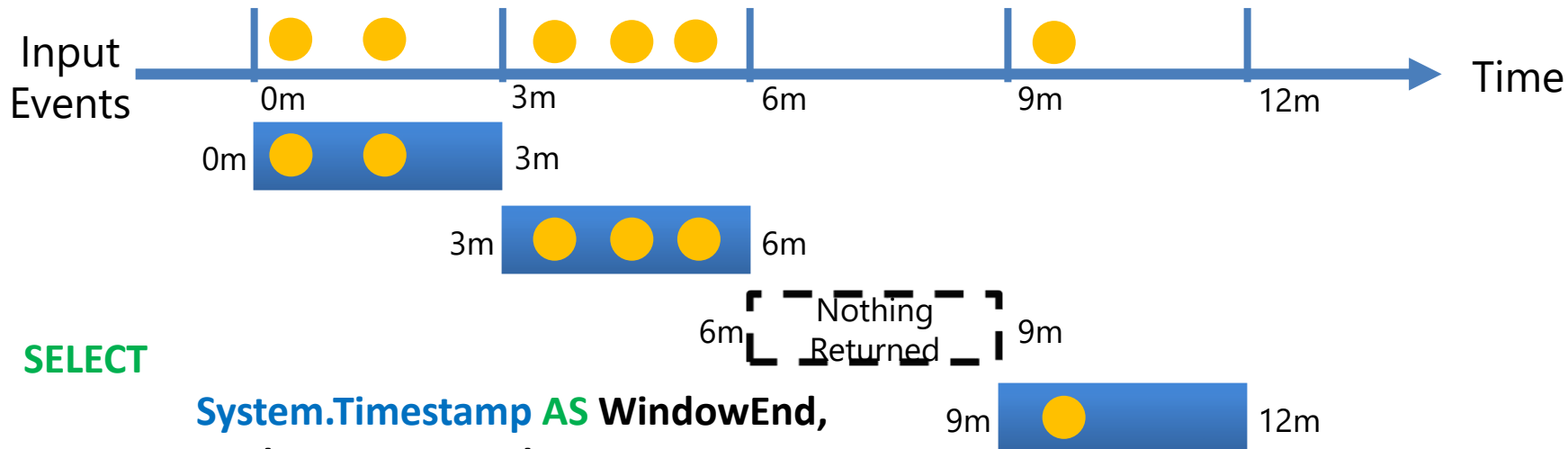

```
SELECT
        System.Timestamp AS WindowEnd,
        card_type AS CardType,
        Count(*) AS Frequency
FROM SwipeStream TIMESTAMP BY swipe_date
GROUP BY TUMBLINGWINDOW(minute, 3), card_type
```
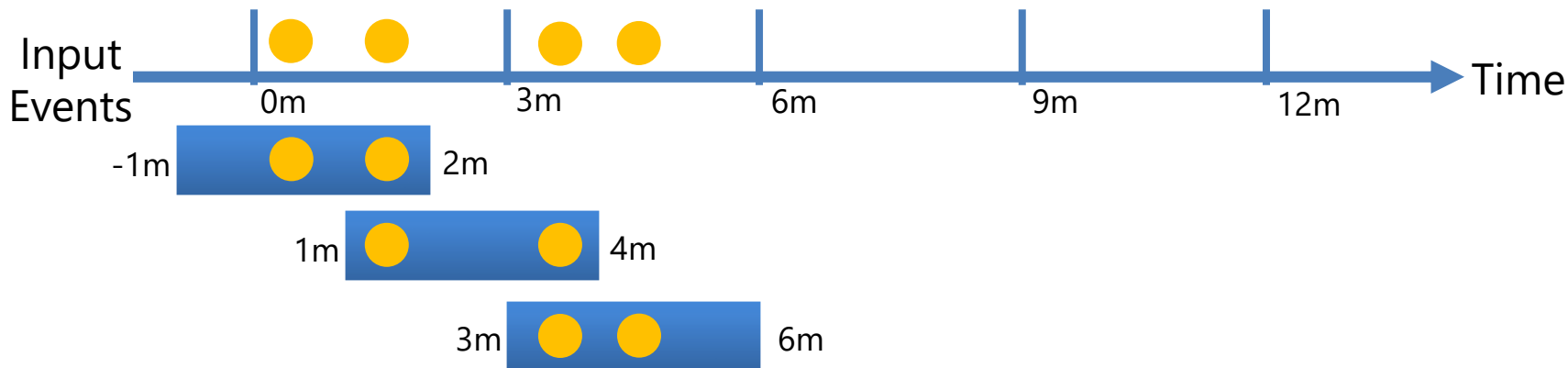
Hopping Window


```
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
```
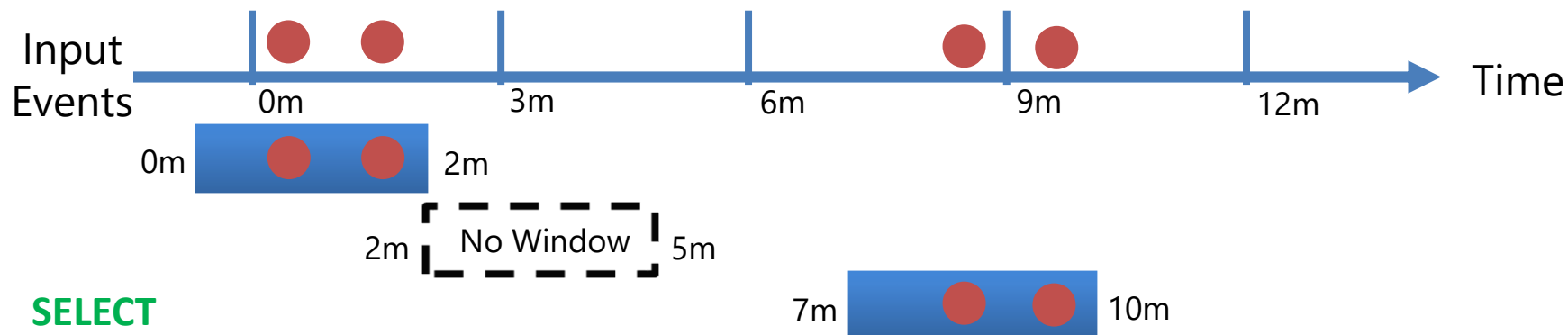
Sliding Window

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

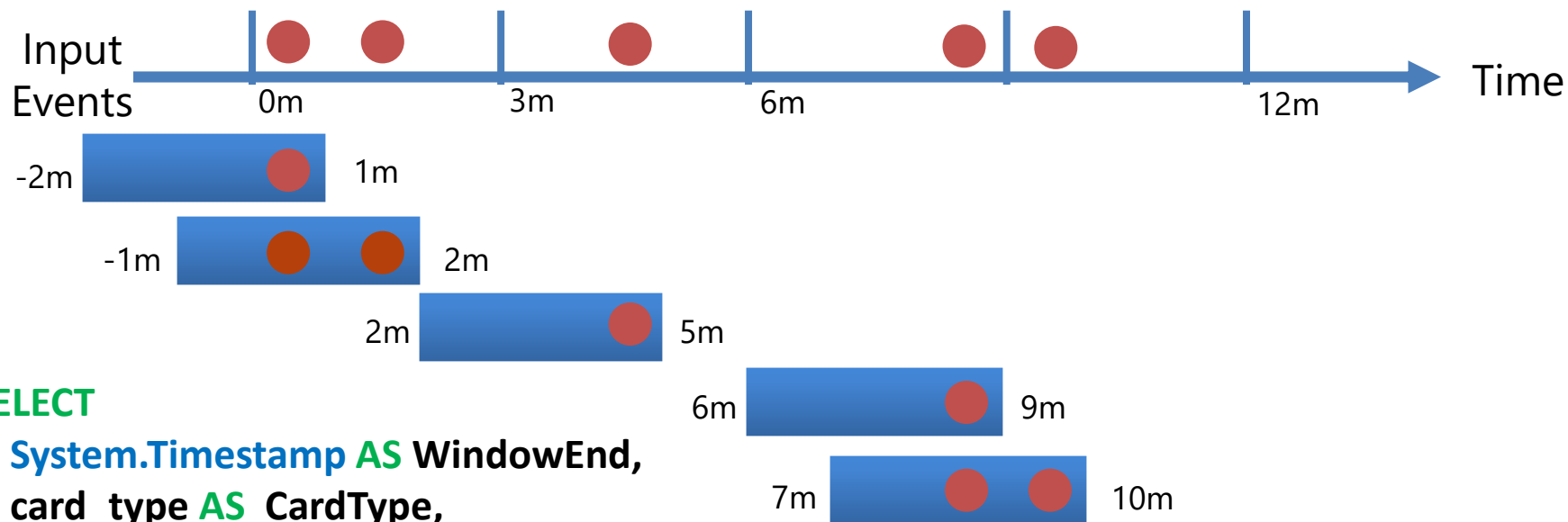# Sliding Window: Without 'Having' Clause


```sql
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
```

datasciencedojo
unleash the data scientist in you

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).

```sql
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?

```sql
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)
```

datasciencedojo
unleash the data scientist in you

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,
        StoleList.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
```

datasciencedojo
unleash the data scientist in you

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)
```
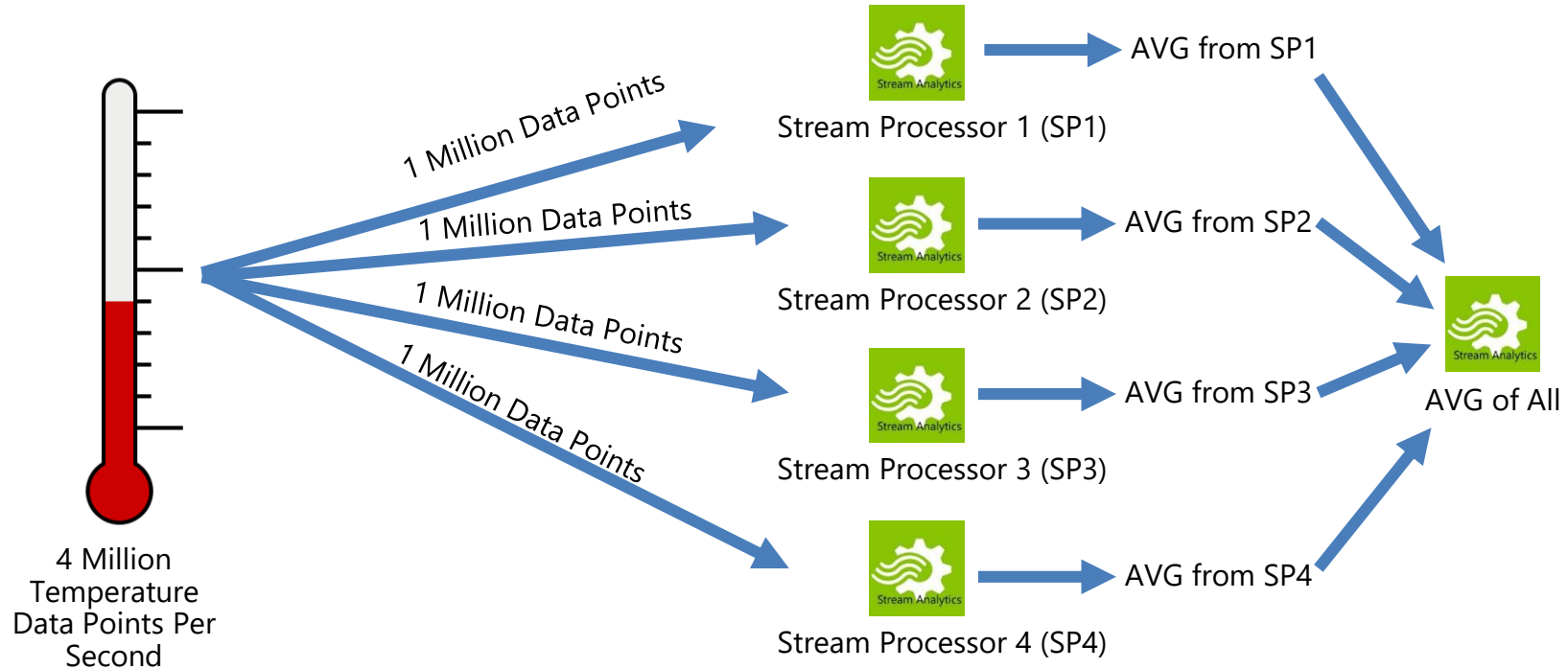
Average of Average Approximations

Built-In Functions And Supported Types

Aggregate functions
`Count, Min, Max, Avg, Sum`

Scalar functions
`Cast`

Date and time
`Datename, Datepart, Day, Month, Year, Datediff, Dateadd`

String
`Len, Concat, Charindex, Substring, Patindex`

datasciencedojo
unleash the data scientist in you

QUESTIONS