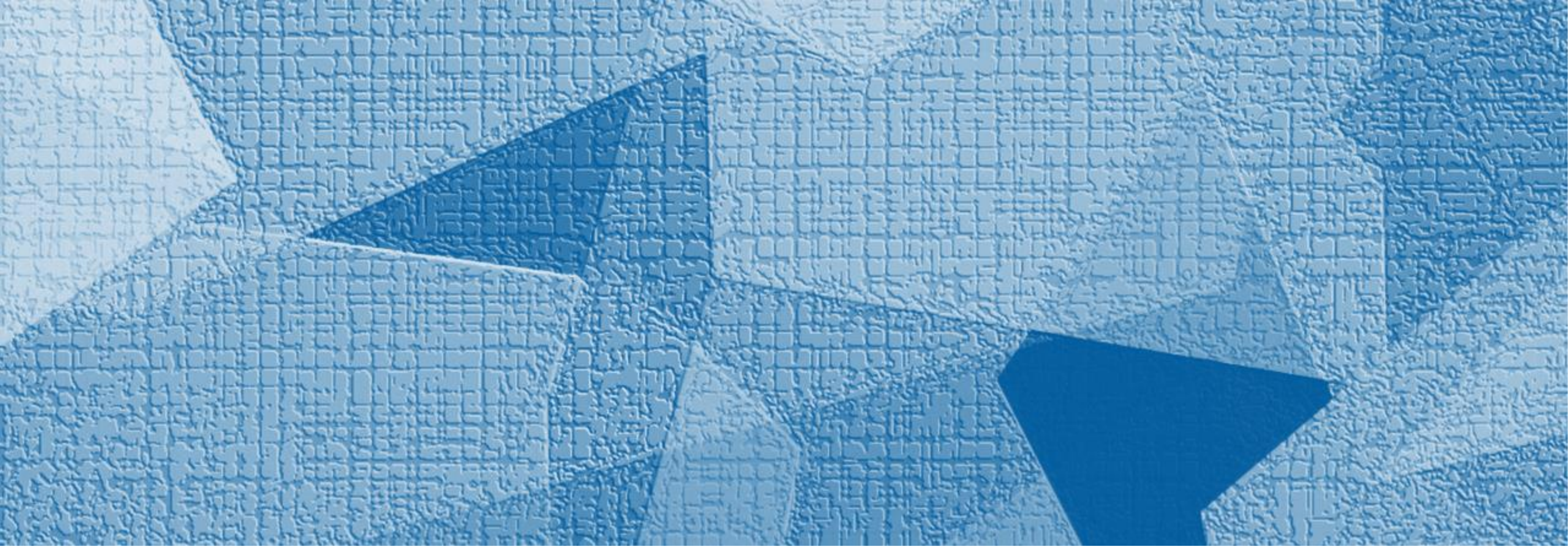
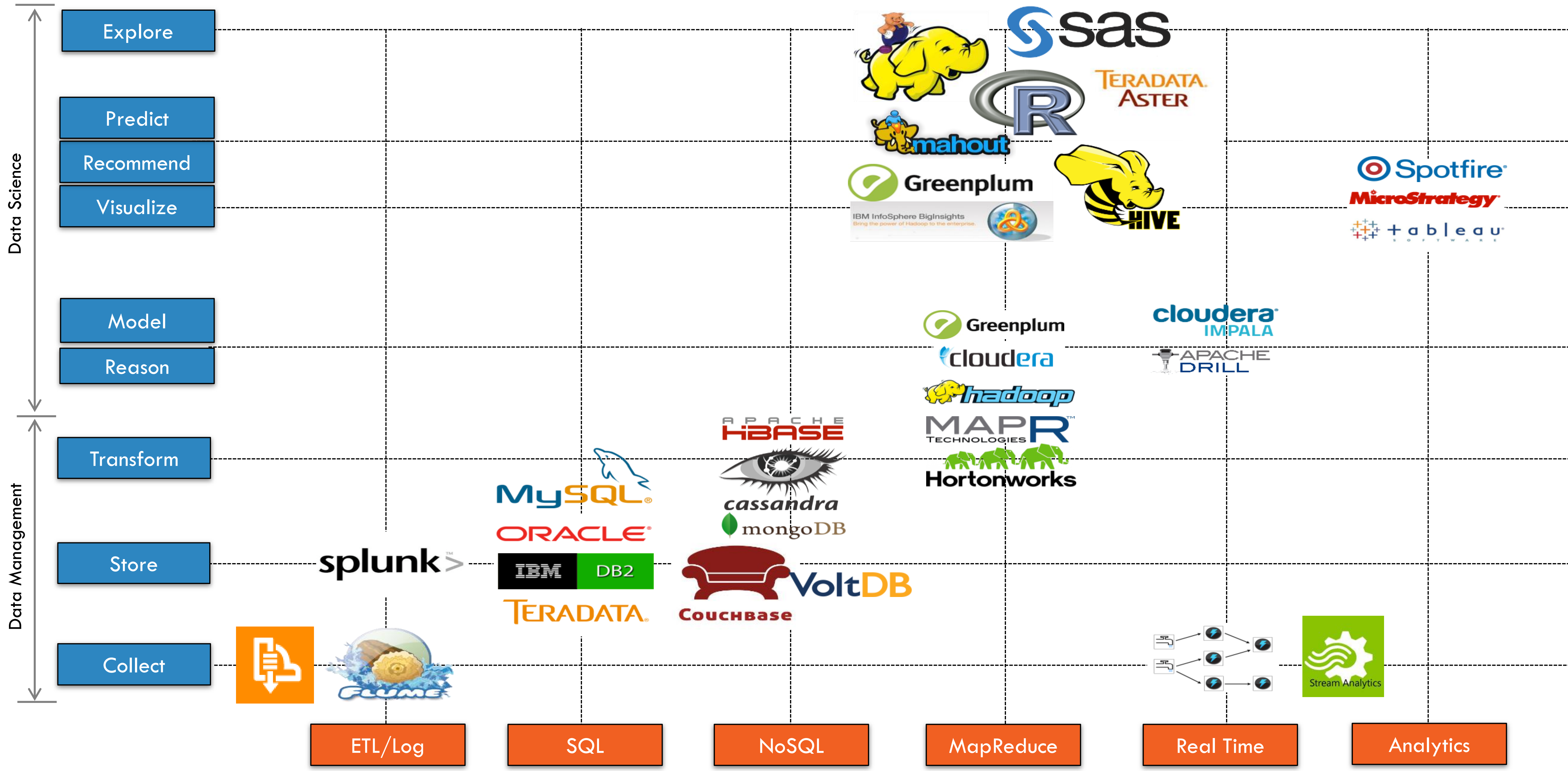


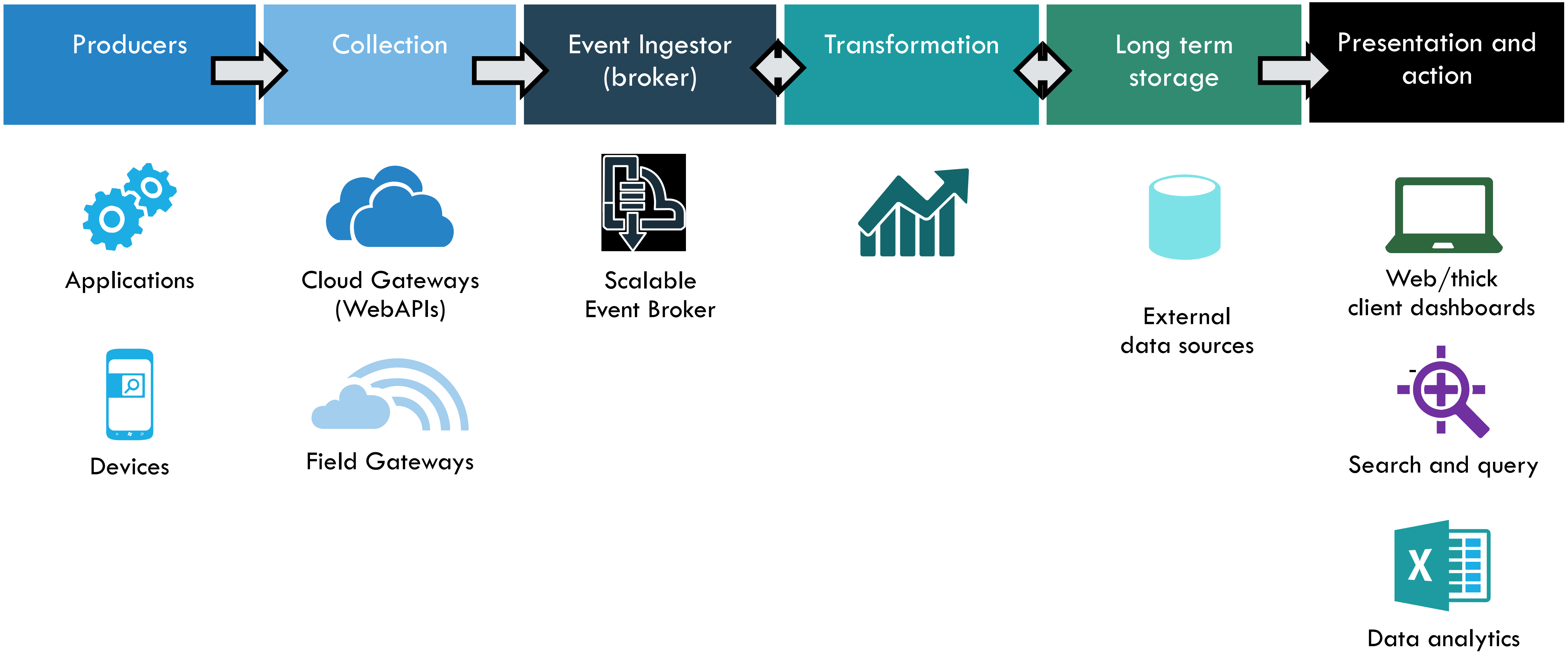
# EVENT PROCESSING WITH STREAM ANALYTICS

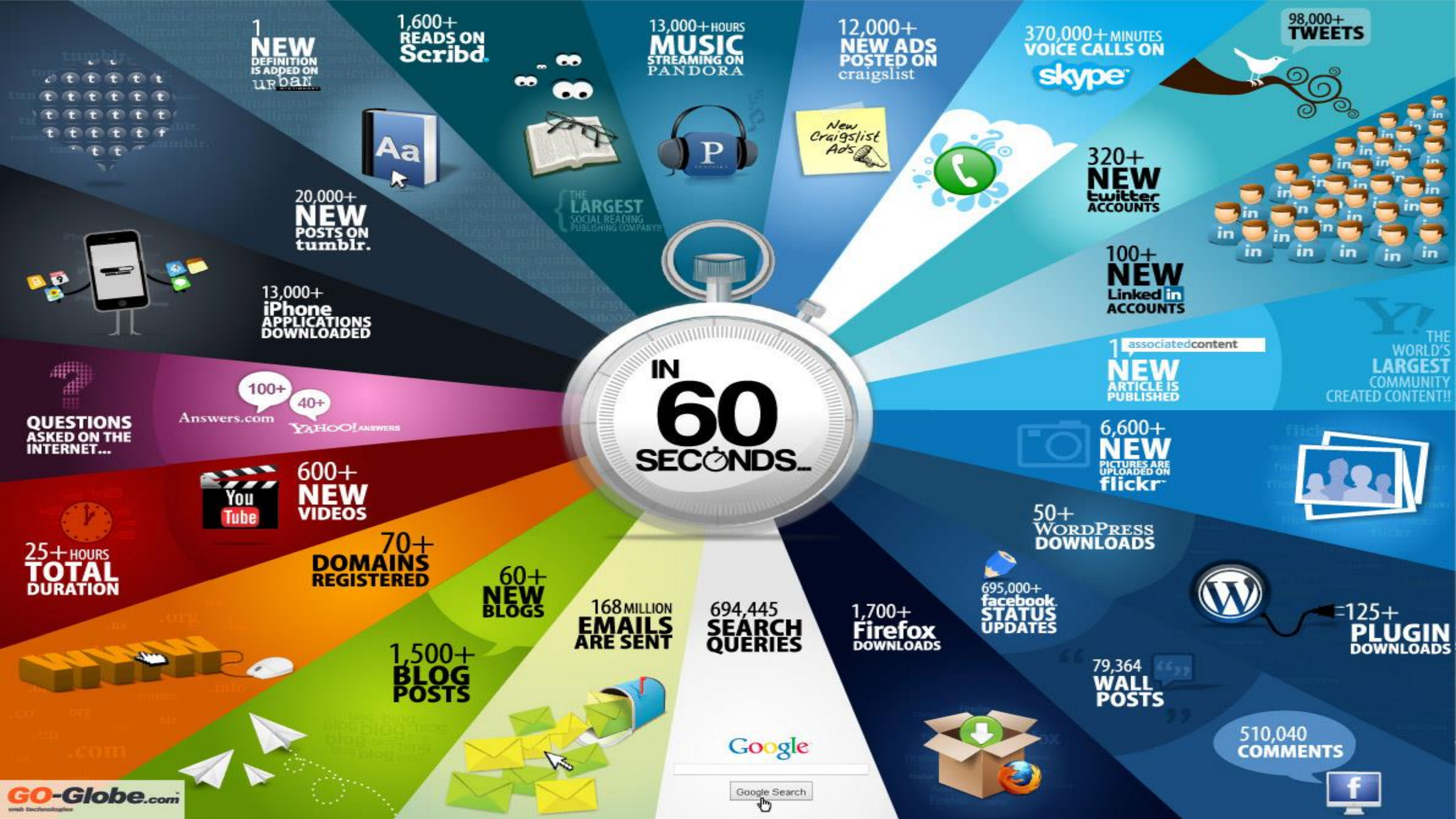
datasciencedojo
unleash the data scientist in you



Big Data – Technology, Platforms & Products

# TYPICAL EVENT PROCESSING



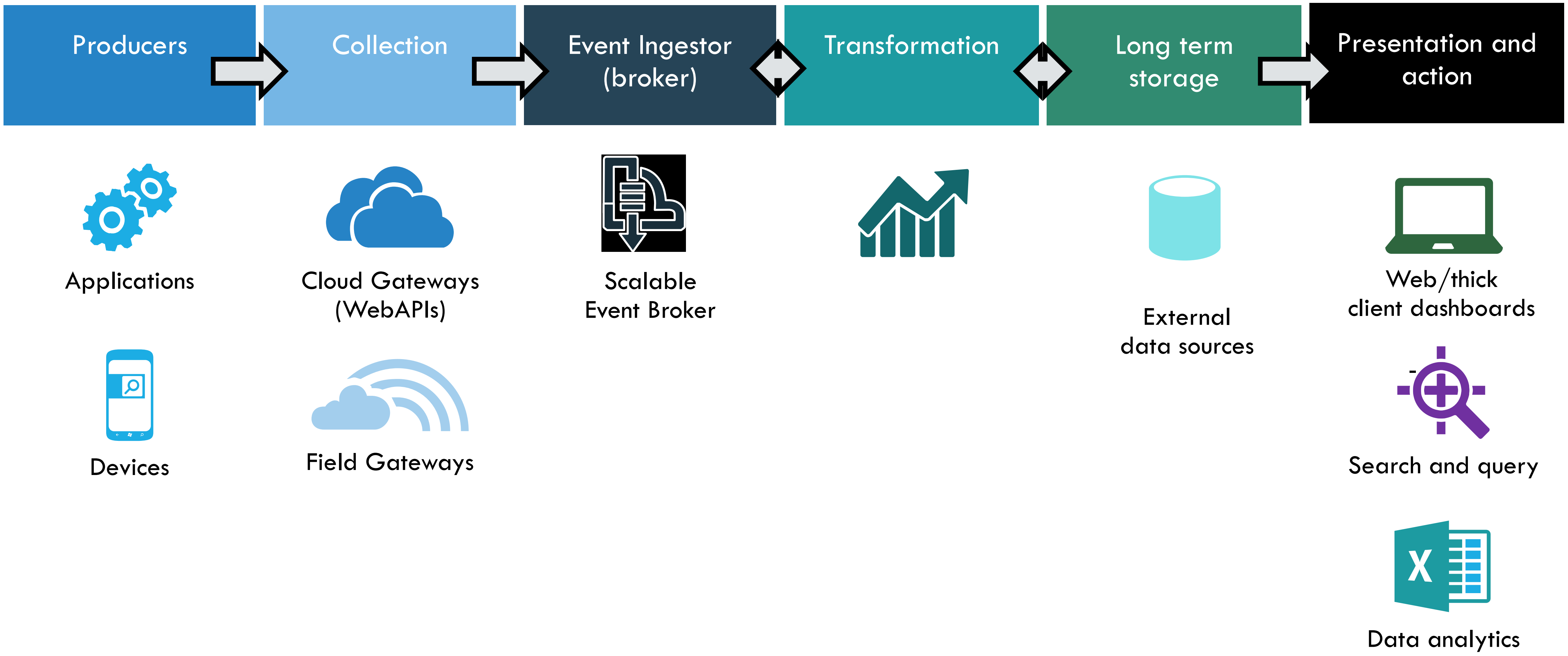

IN 60 SECONDS...
1 NEW DEFINITION IS ADDED ON urban dictionary
1,600+ READS ON Scribd.
THE LARGEST SOCIAL READING PUBLISHING COMPANY!!
13,000+ HOURS MUSIC STREAMING ON PANDORA
12,000+ NEW ADS POSTED ON craigslist
New Craigslist Ads
370,000+ MINUTES VOICE CALLS ON skype
98,000+ TWEETS
320+ NEW twitter ACCOUNTS
100+ NEW LinkedIn ACCOUNTS
1 associatedcontent NEW ARTICLE IS PUBLISHED
THE WORLD'S LARGEST COMMUNITY CREATED CONTENT!!
6,600+ NEW PICTURES ARE UPLOADED ON flickr
50+ WORDPRESS DOWNLOADS
=125+ PLUGIN DOWNLOADS
695,000+ facebook STATUS UPDATES
79,364 WALL POSTS
510,040 COMMENTS
1,700+ Firefox DOWNLOADS
694,445 SEARCH QUERIES
Google
Google Search
168 MILLION EMAILS ARE SENT
60+ NEW BLOGS
1,500+ BLOG POSTS
70+ DOMAINS REGISTERED
600+ NEW VIDEOS
You Tube
25+ HOURS TOTAL DURATION
QUESTIONS ASKED ON THE INTERNET...
100+ Answers.com
40+ YAHOO! ANSWERS
13,000+ iPhone APPLICATIONS DOWNLOADED
20,000+ NEW POSTS ON tumblr.
GO-Globe.com web technologies

## TIMELINESS OF INFORMATION

What's trending in the past 5 minutes?

Your high school friend is also in Vegas NOW.

# TYPICAL EVENT PROCESSING



# DATA AT REST

Question

"How many red cars are in the parking lot?"

Answering with a relational database

Walk out to the parking lot

Count vehicles that are: Red, Car

```
SELECT count(*) FROM ParkingLot
WHERE type = 'Auto'
   AND color = 'Red'
```

## DATA IN MOTION

Different Question

"How many red cars have passed exit 18A on A-10 in the last hour?"

Answering with a relational database

Pull over, park all vehicles in a lot, keep them there for an hour

Count vehicles in the lot

Not a great solution…

## AZURE STREAM QUERY LANGUAGE

### 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

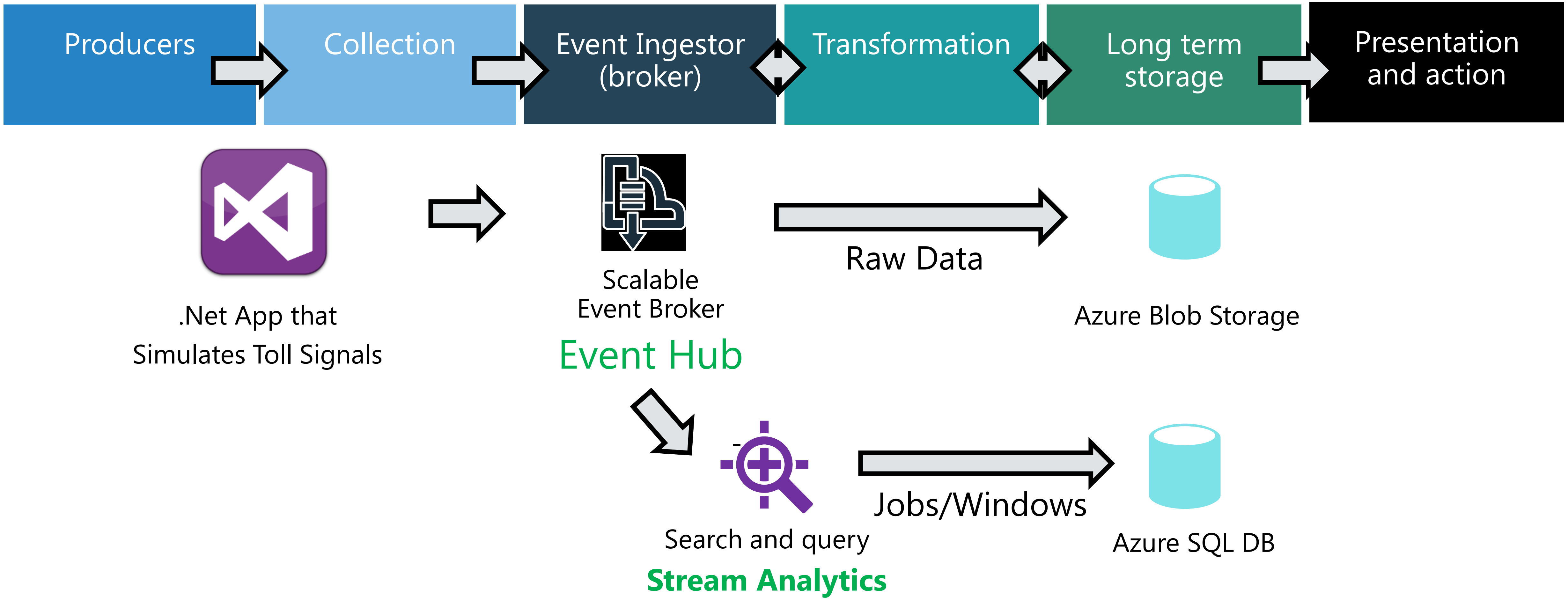
DATEDIFF in joins

PARTITION BY for scale-out

```
WITH agg AS
(
     SELECT Avg(reading), Building
     FROM Temperature
     GROUP BY TumblingWindow(second, 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)
```

## DEMO

# TYPICAL EVENT PROCESSING



## STRENGTHS

- Analyze millions of events per SECOND
- Fault tolerant
- SQL spoken here
- Fully managed service by Azure

## 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`

## A 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)
- Stream Analytics can append your events with a timestamp. (bad practice if standalone)
  - Can be skewed by network and hardware latency.
- Users can define application time stamps with the **TIMESTAMP BY** clause.
- Aggregations have timestamps at the end of the window.

## TRADITIONAL SQL

How many vehicles passed through each toll booth yesterday?

- Why can’t we ask how many cars have gone through so far today?

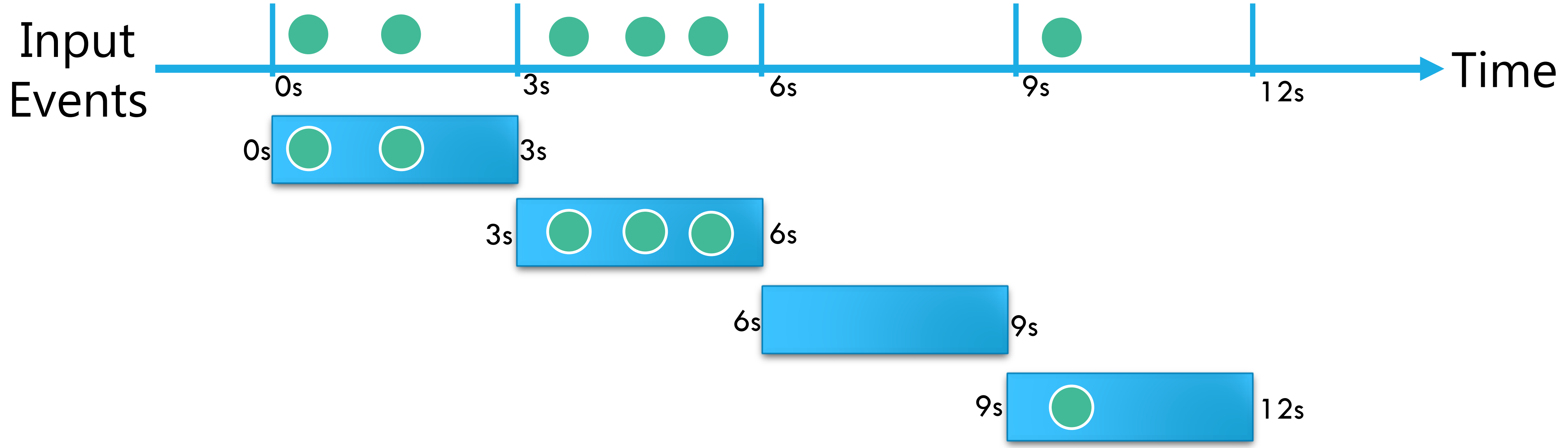
```
SELECT TollID, Count(*) AS Count
FROM EntryTable
WHERE date = ‘yesterday’
GROUP BY TollID
```

## AZURE STREAM QUERY LANGUAGE

How many vehicles pass through each toll booth every 3 seconds?

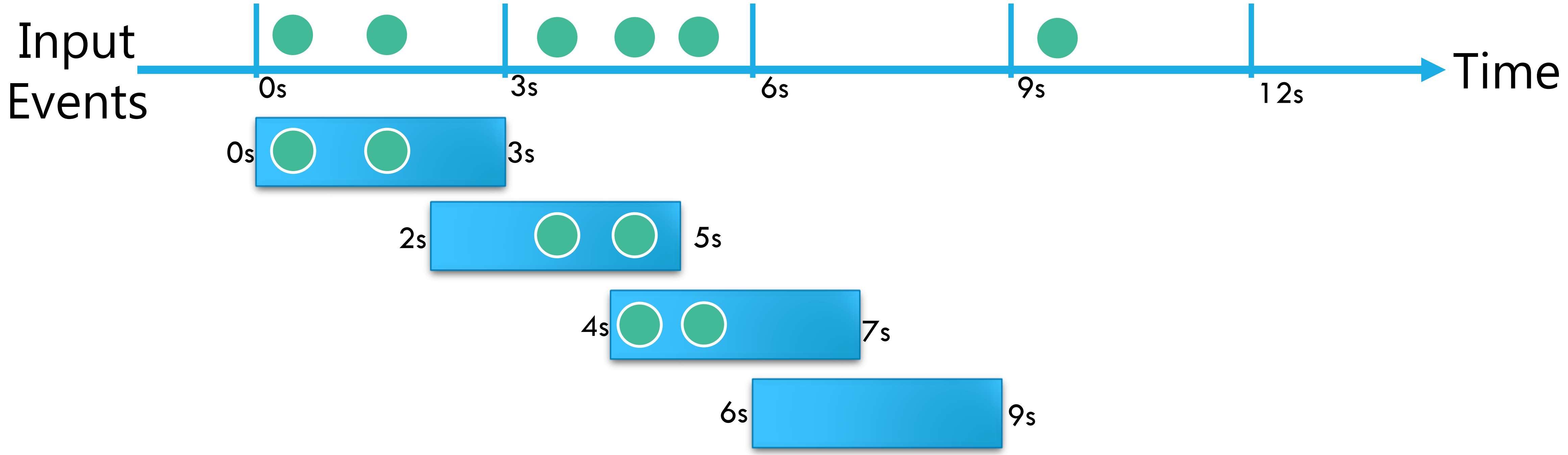
```
SELECT TollID, System.Timestamp AS WindowEnd, Count(*) AS Count
FROM EntryStream TIMESTAMP BY EntryTime
GROUP BY TUMBLINGWINDOW(second, 3), TollID
```

# TUMBLING WINDOW



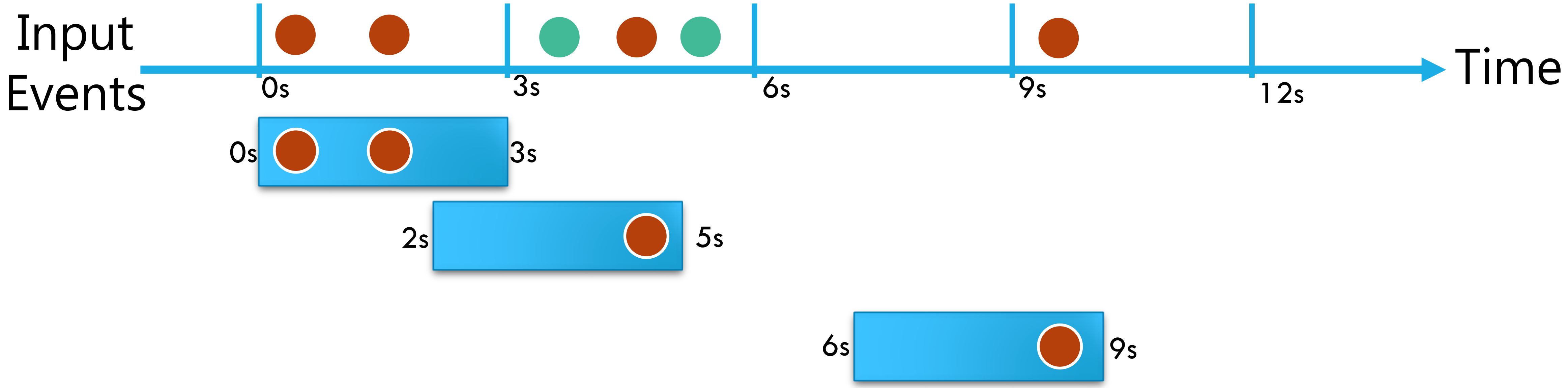
```
SELECT TollID, System.Timestamp AS WindowEnd, Count(*) AS Count
FROM EntryStream TIMESTAMP BY EntryTime
GROUP BY TUMBLINGWINDOW(second, 3), TollID
```

# HOPPING WINDOW



```
SELECT TollID, System.Timestamp AS WindowEnd, Count(*) AS Count
FROM EntryStream TIMESTAMP BY EntryTime
GROUP BY HOPPINGWINDOW(second, 2, 3), TollID
```

# SLIDING WINDOW



```
SELECT TollID, System.Timestamp AS WindowEnd, Count(*) AS Count
FROM EntryStream TIMESTAMP BY EntryTime
GROUP BY SLIDING(second, 2, 3), TollID
HAVING Color = RED
```

## SUM AGGREGATION

How much toll revenue is being accumulated every 3 minutes?

```
SELECT System.Timestamp AS WindowEnd, Sum(TollAmount) AS IntervalRevenue
FROM EntryStream TIMESTAMP BY EntryTime
GROUP BY TUMBLINGWINDOW(minute, 3), WindowEnd
```

## SUM AGGREGATION

Which 3 minute time interval made more than $10?

```
SELECT System.Timestamp AS WindowEnd, Sum(TollAmount) AS IntervalRevenue
FROM EntryStream TIMESTAMP BY EntryTime
GROUP BY TUMBLINGWINDOW(minute, 3), WindowEnd
Having IntervalRevenue > 10
```

## DATEDIFF

How long does it take for each car to pass through the toll zone?

```
SELECT
    EntryStream.LicensePlate,
    DATEDIFF (second, EntryStream.EntryTime, Exitstream.ExitTime) AS DurationInSeconds
FROM EntryStream timestamp by EntryTime
JOIN Exitstream timestamp by ExitTime
ON Exitstream.LicensePlate = ExitStream.LicensePlate
AND DATEDIFF(second, EntryStream, ExitStream) BETWEEN 0 AND 1800
```

## DATEDIFF

How long does it take for each car to pass through the toll zone?

```
SELECT
   EntryStream.LicensePlate,
   DATEDIFF(second, EntryStream.EntryTime, Exitstream.ExitTime) AS DurationInSeconds
FROM EntryStream timestamp by EntryTime
JOIN Exitstream timestamp by ExitTime
ON Exitstream.LicensePlate = ExitStream.LicensePlate
AND DATEDIFF(second, EntryStream, ExitStream) BETWEEN 0 AND 1800
```

## DATEDIFF, INTEGER ONLY

How long (in HOURS) does it take for each car to pass through the toll zone?

- Decimal floats cut off, returns only 0s.

```
SELECT
   EntryStream.LicensePlate,
   DATEDIFF(hour, EntryStream.EntryTime, Exitstream.ExitTime) AS DurationInSeconds
FROM EntryStream timestamp by EntryTime
JOIN Exitstream timestamp by ExitTime
ON Exitstream.LicensePlate = ExitStream.LicensePlate
AND DATEDIFF(hour, EntryStream, ExitStream) BETWEEN 0 AND 1800
```

## CALCULATIONS

How fast (mph) was each car traveling through the toll zone? Assuming the toll zone was 1.5 miles long.

```
SELECT
   EntryStream.LicensePlate,
   1.5 / (DATEDIFF(second, (second, EntryStream.EntryTime, Exitstream.ExitTime) / 60 / 60) AS MilesPerHour
FROM EntryStream timestamp by EntryTime
JOIN Exitstream timestamp by ExitTime
ON Exitstream.LicensePlate = ExitStream.LicensePlate
AND DATEDIFF(second, EntryStream, ExitStream) BETWEEN 0 AND 3600
```

# AZURE STREAM-QL QUIRKS

Who was speeding through the toll zone?

- Simple question… but the query below will break.

```
SELECT
   EntryStream.LicensePlate,
   1.5 / (DATEDIFF(second, (second, EntryStream.EntryTime, Exitstream.ExitTime) / 60 / 60) AS MilesPerHour
FROM EntryStream timestamp by EntryTime
JOIN Exitstream timestamp by ExitTime
ON Exitstream.LicensePlate = ExitStream.LicensePlate
AND DATEDIFF(second, EntryStream, ExitStream) BETWEEN 0 AND 3600
~~WHERE MilesPerHour > 62~~
```

## AZURE STREAM-QL QUIRKS

Who was speeding through the toll zone?

- Works… but ugly solution.

```
SELECT
   EntryStream.LicensePlate,
   1.5 / (DATEDIFF(second, (second, EntryStream.EntryTime, Exitstream.ExitTime) / 60 / 60) AS MilesPerHour
FROM EntryStream timestamp by EntryTime
JOIN Exitstream timestamp by ExitTime
ON Exitstream.LicensePlate = ExitStream.LicensePlate
AND DATEDIFF(second, EntryStream, ExitStream) BETWEEN 0 AND 3600
WHERE 1.5 / (DATEDIFF(second, (second, EntryStream.EntryTime, Exitstream.ExitTime) / 60 / 60) > 62
```