



Willkommen beim #GAB2016!

# Stream Processing mit Azure

Hans-Peter Grahl  
Entwickler & Berater | Netconomy | FH CAMPUS 02

Twitter: @hpgrahl 



*16. April 2016*



# Inhalte



- Motivation / Einführung
- Microsoft's **Azure Stream Analytics + Demo**
- Fragen & Zusammenfassung

# Warum Datenstromanalyse in der Cloud?



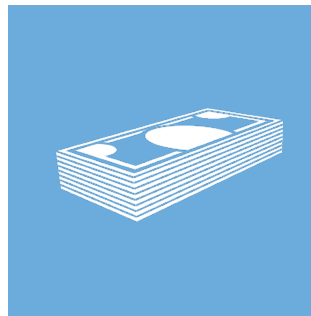
Eventbasierte Daten  
befinden sich  
oftmals bereits in  
der Cloud

Ein Großteil der Daten  
ist nicht mehr lokal

*“Bring the processing to the data,  
not the data to the processing!”*



Eventbasierte  
Daten sind immer  
häufiger global  
verteilt



Reduced TCO

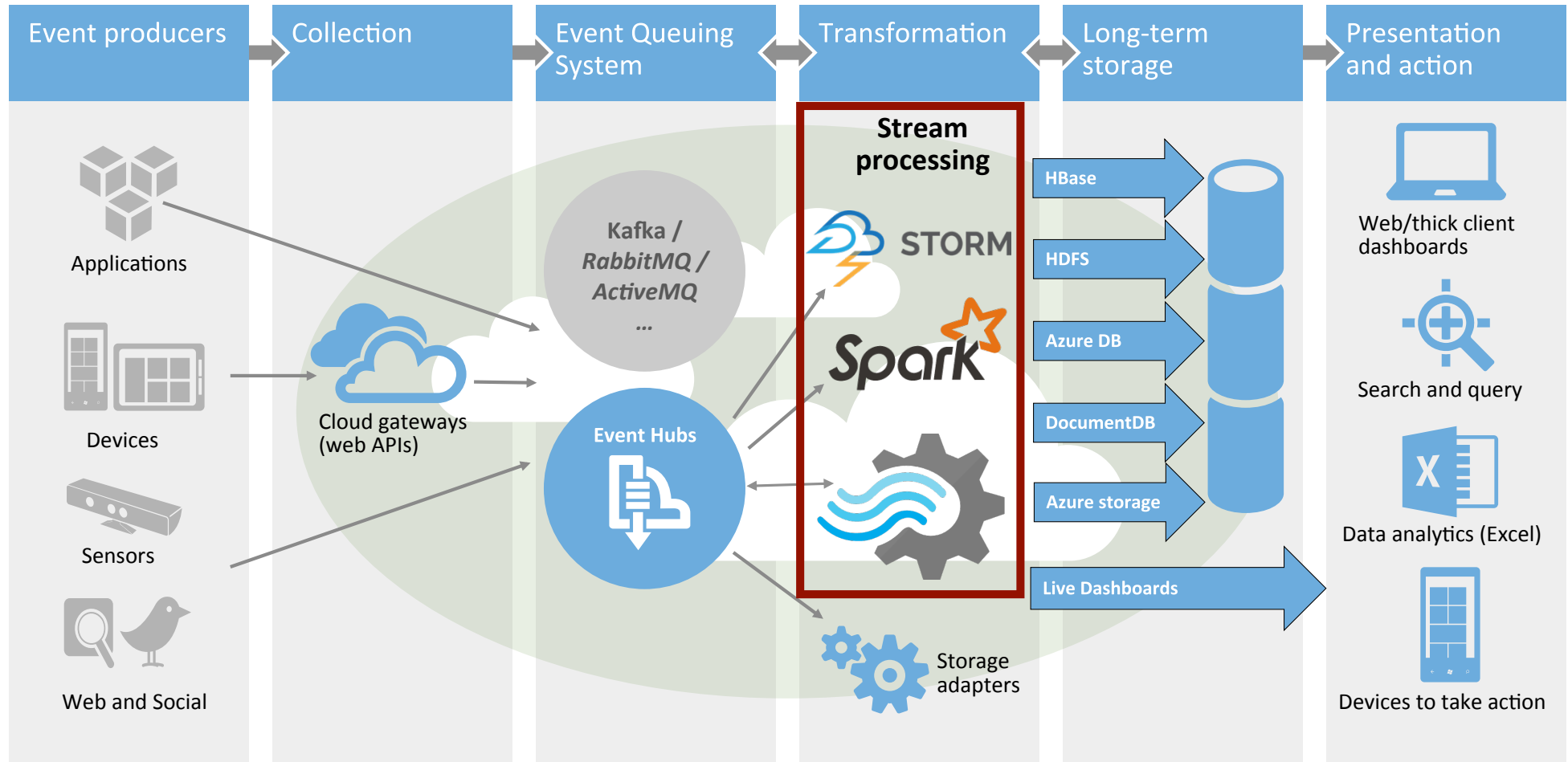


Elastic scale-out



Service,  
not infrastructure

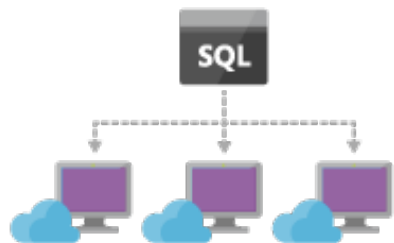
# Datenstromanalyse in der Cloud



# Azure Stream Analytics



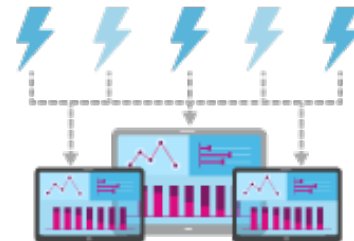
“...is a fully managed, cost effective real-time event processing engine that helps to unlock deep insights from data.”



**Rasche / Einfache  
Entwicklung**



**Zuverlässigkeit  
& Skalierbarkeit**

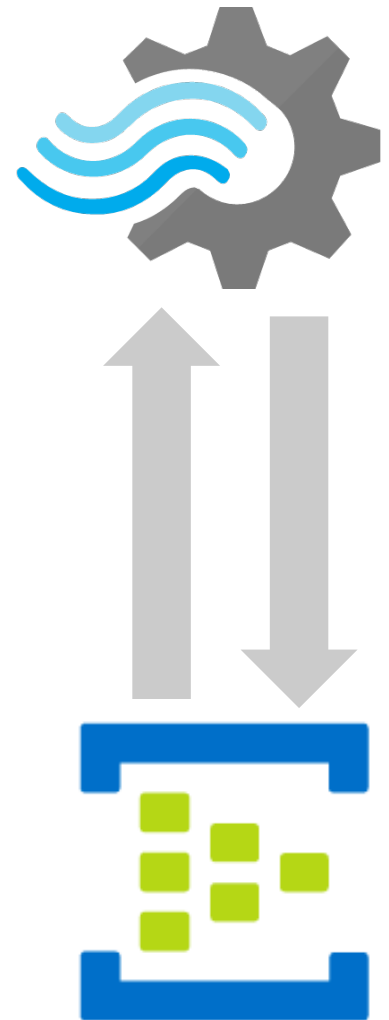


**Echtzeitnahe  
Verarbeitung**

# Azure Stream Analytics

## Charakteristiken

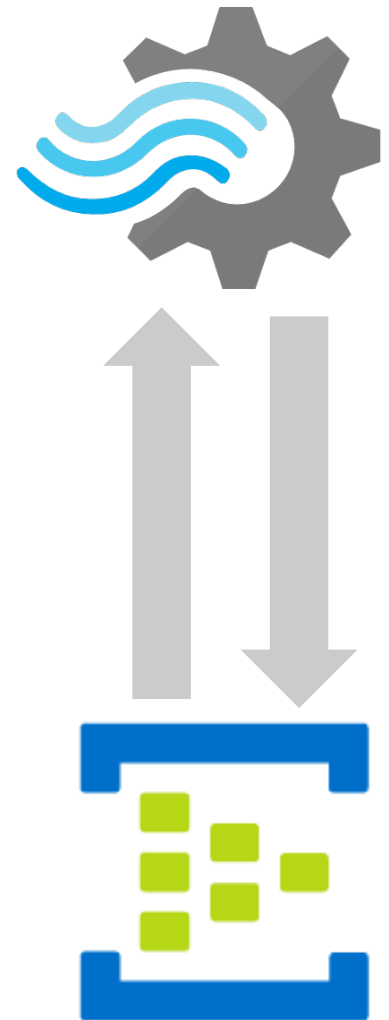
- sehr kurze Einarbeitungszeit durch T-SQL Syntax
- nahtlose Integration zu Azure Event Hubs
- Kombination von Streams und statischen Daten
- horizontale Skalierung
- niedrige Latenz unter hoher Last
- garantierte Verfügbarkeit (99.9%)
- defacto kein administrativer Aufwand



# Azure Stream Analytics

## Skalierungskonzept

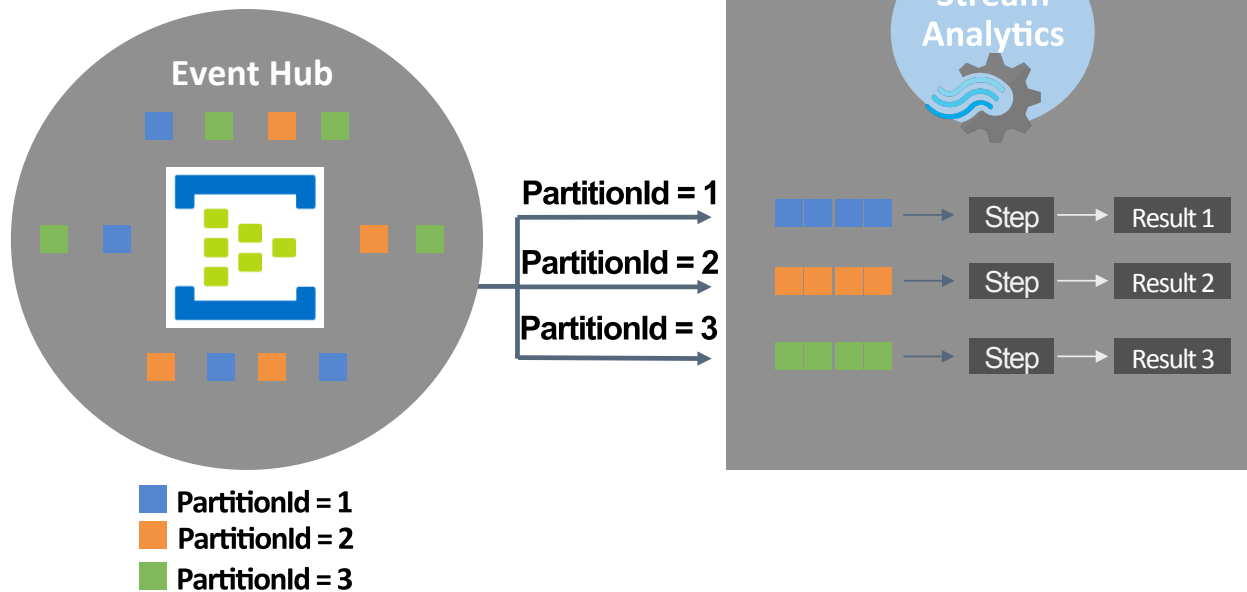
- durch mehrere sog. Streaming Units (SU)
    - kombiniertes Maß an CPU, RAM und I/O
  - per default 1 SU/Job
  - 1 SU bietet Durchsatz bis ~1MB/Sek.
  - im Standard Account gesamt max. 50 SUs
- Einsatz mehrerer SUs abhängig von entwickelter Abfrage sowie Partitionskonfiguration der involvierten Datenquellen



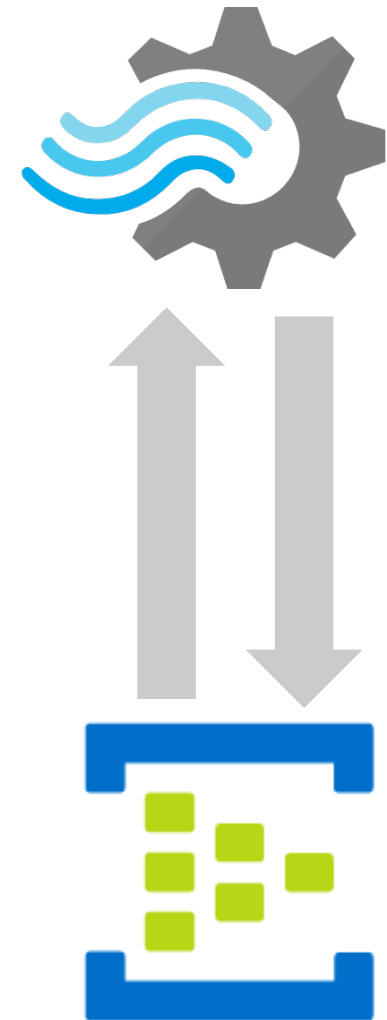
# Azure Stream Analytics

## Skalierungskonzept

*Bsp. 3 Partitions\**



\*Partition == geordnete Sequenz von Events





# Azure Stream Analytics

## Skalierungskonzept

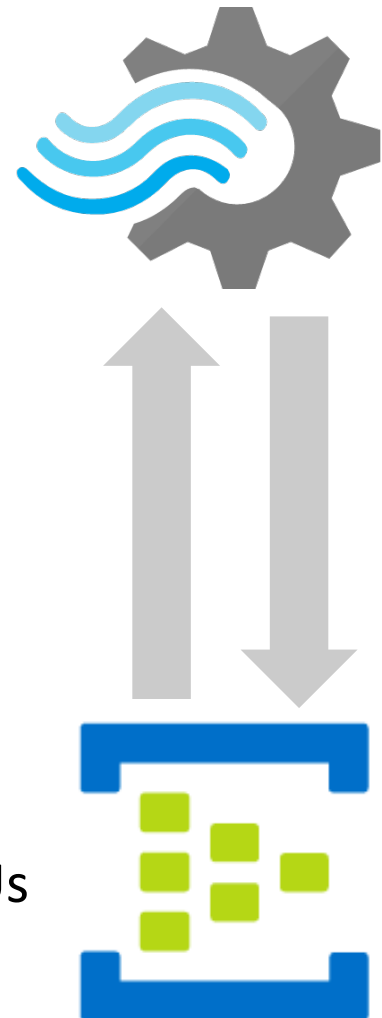
→ hochgradig paralleler Job sofern:

- #Input Partitionen == #Output Partitionen
- partitionierte Abfrage: **Partition By PartitionId**
- falls mehrstufige Abfrage alle mit gleicher Partitionierung

pro Partitionsabfrage sind bis zu 6 SUs verwendbar

alle nicht partitionierten Abfrageschritte gemeinsam max. 6 SUs

<https://azure.microsoft.com/en-us/documentation/articles/stream-analytics-scale-jobs/>



# Azure Stream Analytics

## 5 Schritte:

- 1) Stream Analytics Job erzeugen
- 2) Inputquelle(n) definieren
- 3) Abfrage mit T-SQL Syntax entwickeln
- 4) Outputquelle(n) definieren
- 5) Job starten & überwachen



# Azure Stream Analytics

The screenshot displays the Microsoft Azure portal interface for creating a new Stream Analytics job. The top navigation bar shows the breadcrumb path: Microsoft Azure > Everything > Stream Analytics job > New Stream Analytics Job. A search bar is also present.

**Left Sidebar:** Contains navigation options such as 'New', 'Resource groups', 'All resources', 'Recent', 'App Services', 'SQL databases', 'Virtual machines (classic)', 'Virtual machines', 'Cloud services (classic)', 'Subscriptions', and 'Storage accounts (classi...'. A 'Browse >' link is at the bottom.

**Main Content Area:** Features a 'Stream Analytics job' header with the Microsoft logo. Below it, a descriptive paragraph states: "Azure Stream Analytics is a fully managed, cost effective real-time event processing engine that helps to unlock deep insights from data. Stream Analytics makes it easy to set up real-time analytic computations on data streaming from devices, sensors, web sites, social media, applications, infrastructure systems, and more." A second paragraph explains: "With a few clicks in the Azure portal, you can author a Stream Analytics job specifying the input source of the streaming data, the output sink for the results of your job, and a data transformation expressed in a SQL-like language. You can monitor and adjust the scale/speed of your job in the Azure portal to scale from a few kilobytes to a gigabyte or more of events processed per second." A third paragraph notes: "Stream Analytics leverages years of Microsoft Research work in developing highly tuned streaming engines for time-sensitive processing, as well as language integrations for intuitive specifications of such." Below the text are social media icons for Twitter, Facebook, LinkedIn, YouTube, Google+, and Email.

**Diagram:** A flow diagram showing data flow from 'IoT Hub' to 'Stream Analytics', which then branches out to 'Dashboard', 'Alerts', and 'Storage'.

**Right Panel: 'New Stream Analytics Job'** Configuration fields:

- \* Job name:
- \* Subscription:
- \* Resource group:  with a sub-field for "New resource group name"
- \* Location:

A 'Pin to dashboard' checkbox is at the bottom left of the panel, and a blue 'Create' button is at the bottom right.

**Bottom:** A grey 'Create' button is located at the bottom left of the main content area.

# Azure Stream Analytics



## Inputquellen:

\* Input alias

\* Source Type ⓘ

\* Source ⓘ

\* Service bus namespace ⓘ

\* Event hub name ⓘ

\* Event hub policy name ⓘ

\* Event hub policy key ⓘ

Event hub consumer group ⓘ

\* Event serialization format ⓘ

Encoding ⓘ

- **Data Streams**  
=> EventHub, IoT Hub, BlobStorage
- **Reference Data**  
=> BlobStorage
- **Formate:** JSON, CSV, Avro
- **Encoding:** UTF-8

# Azure Stream Analytics



## Outputquellen:

\* Output alias

\* Sink ⓘ  
Event hub ▼

\* Service bus namespace ⓘ

\* Event hub name ⓘ

\* Event hub policy name ⓘ

\* Event hub policy key ⓘ

Partition key column ⓘ

\* Event serialization format ⓘ  
JSON ▼

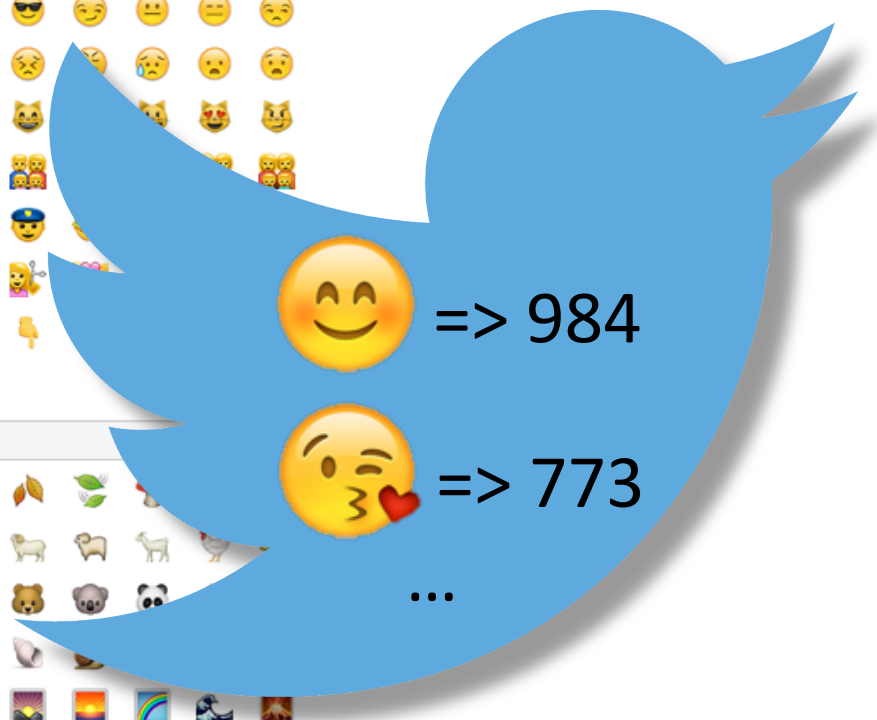
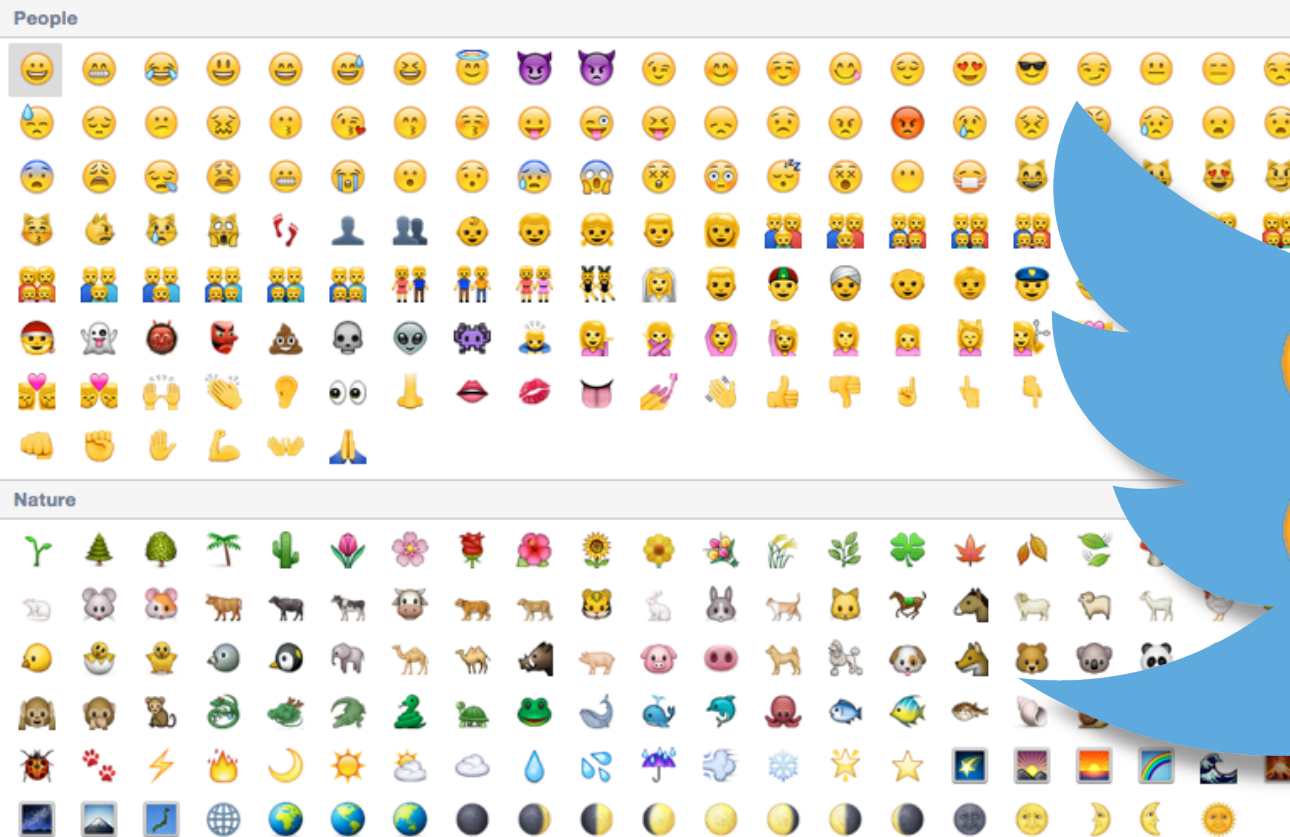
Encoding ⓘ  
UTF-8 ▼

Format ⓘ  
Line separated ▼

- Azure SQL DB, BlobStorage, EventHub,
- Table Storage, Service Bus (Queues & Topics),
- DocumentDB, Power BI

# Demo-Anwendung

=> Tracking Emojis in public Tweets



# Azure Stream Analytics

## Abfrage Editor



Need help with your query? Check out some of the most common Stream Analytics query patterns [here](#).

```
1 SELECT
2     *
3 INTO
4     [YourOutputAlias]
5 FROM
6     [YourInputAlias]
```

# Azure Stream Analytics

## Abfrage Sprache

- Subset von standard T-SQL Syntax

### DML Statements

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING
- CASE
- JOIN
- UNION

### Date and Time Functions

- DATENAME
- DATEPART
- DAY
- MONTH
- YEAR
- DATETIMEFROMPARTS
- DATEDIFF
- DATADD

### String Functions

- LEN
- CONCAT
- CHARINDEX
- SUBSTRING
- PATINDEX

### Scaling Functions

- WITH
- PARTITION BY

### Aggregate Functions

- SUM
- COUNT
- AVG
- MIN
- MAX

### Array Functions

...

### Analytic Functions

...



<https://msdn.microsoft.com/en-us/library/azure/dn835030.aspx>



# Azure Stream Analytics

## Abfrage Sprache

- Windowing Erweiterungen

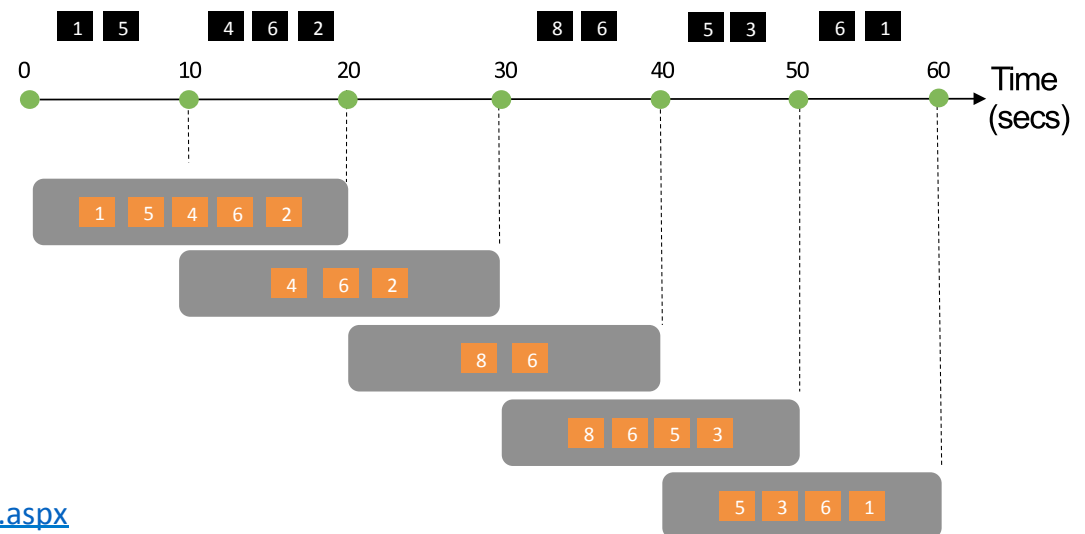
### Hopping Windows (HW)

- wiederholend
- Überlappung möglich
- Versatz um fixe Zeitspanne

<https://msdn.microsoft.com/en-us/library/azure/dn835041.aspx>



A 20-second Hopping Window with a 10 second "Hop"



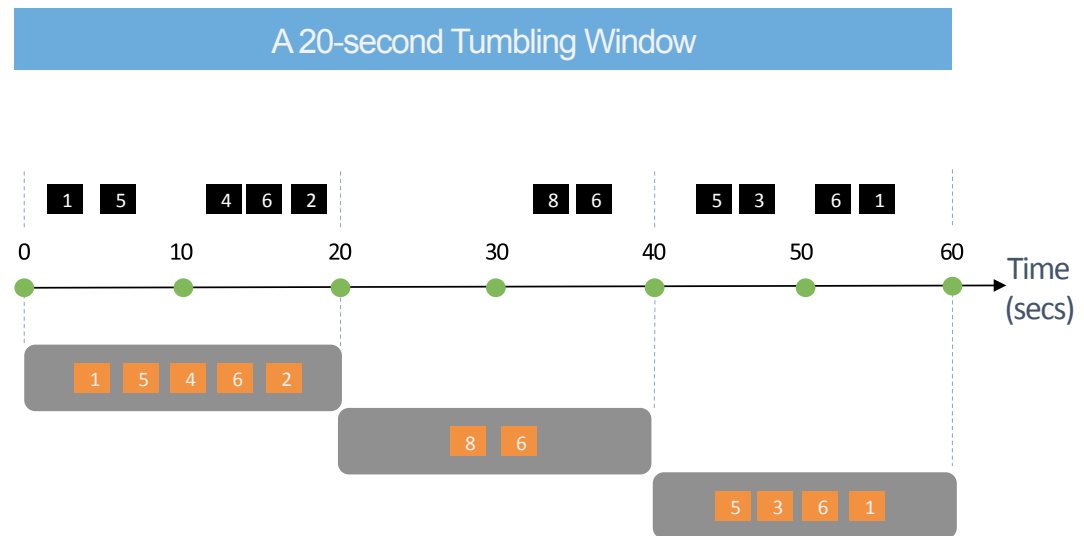
# Azure Stream Analytics

## Abfrage Sprache

- Windowing Erweiterungen

## Tumbling Windows (TW)

- wiederholend
- nicht überlappend



<https://msdn.microsoft.com/en-us/library/azure/dn835055.aspx>

# Azure Stream Analytics

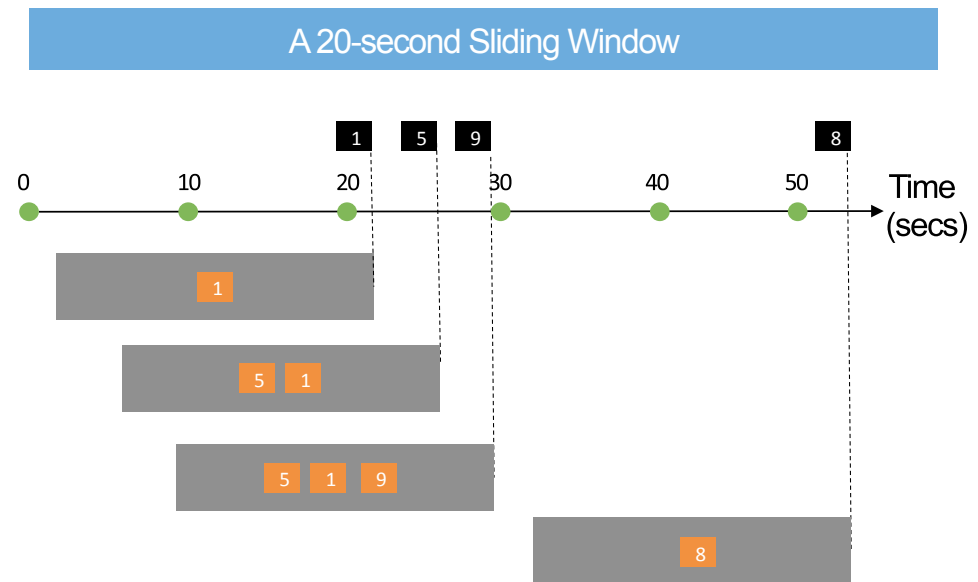
## Abfrage Sprache

- Windowing Erweiterungen

## Sliding Windows (SW)

- „kontinuierliche“ Verschiebung
- Outputs nur bei Änderungen der Daten im Window  
→ „Delta-Triggerring“
- mind. 1 Event / SW

<https://msdn.microsoft.com/en-us/library/azure/dn835051.aspx>



# Azure Stream Analytics

## Zeitversatz Regeln?



→ „Late arrival“ (Events treffen verspätet ein)

- a) Zeitstempel werden ggf. automatisch korrigiert
- b) od. Events verworfen

→ „Out of order“ (Events treffen nicht chronologisch ein)

- a) Events werden ggf. neu-/umsortiert
- b) od. Events verworfen



<https://www.flickr.com/photos/smemon/5281453002/>

# Azure Stream Analytics



## Abfrage Sprache

- GROUP BYs bzw. JOINS für Datenströme brauchen zwingend Angabe von Zeitpunkt bzw. Zeitspanne

```
GROUP BY [fieldname], [Hopping|Tumbling|Sliding]Window(...)
```

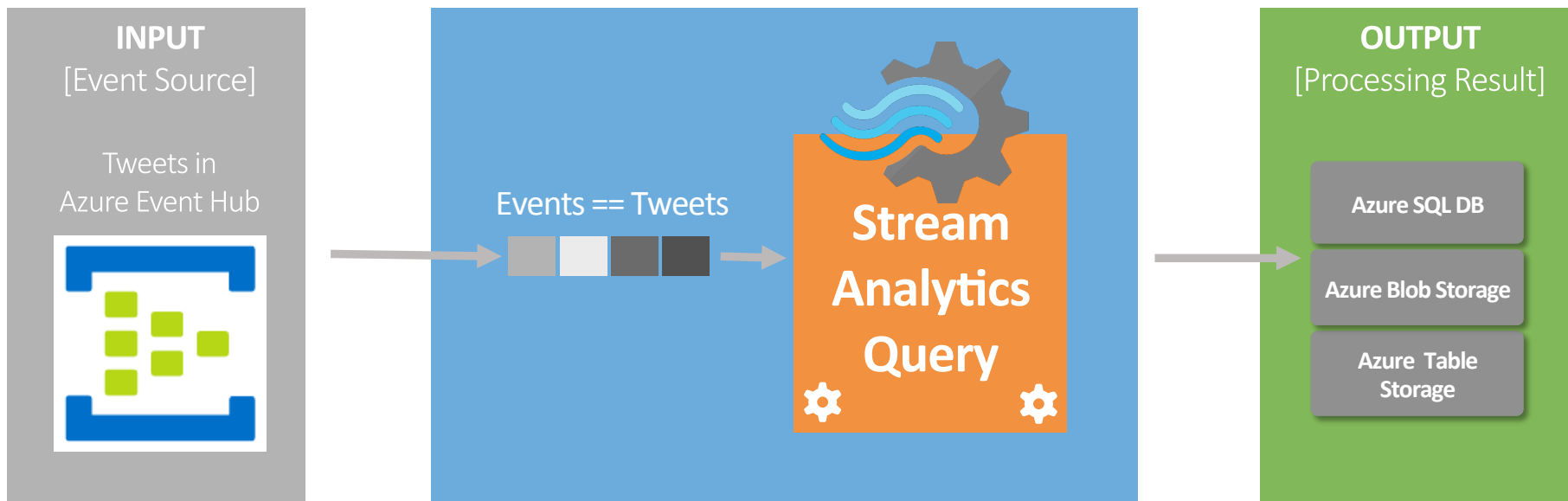
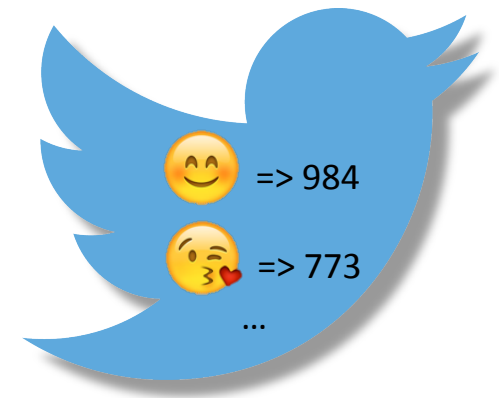
```
JOIN ... ON ... AND DATEDIFF(...) BETWEEN 0 AND N
```

- WITH für „mehrstufige“ Abfragen bzw. zur Erzeugung von temp. Result Sets

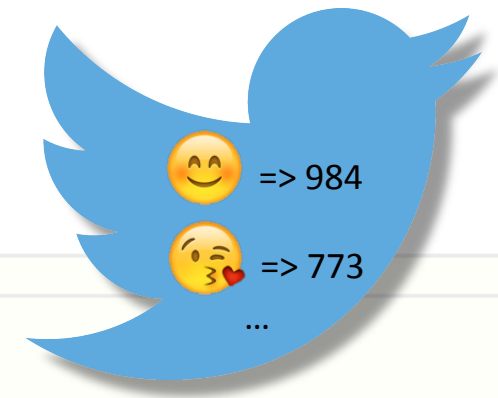
```
WITH temp1 AS (SELECT ... FROM input),  
    WITH temp2 AS ( SELECT ... FROM temp1) [,...]  
SELECT ... FROM temp2
```

# Azure Stream Analytics Demo

=> Tracking Emojis in public Tweets



# Azure Stream Analytics Demo



```
1 WITH emojis AS (  
2     SELECT  
3         emoji.ArrayValue as emj  
4     FROM  
5         PublicTweetStream as tweets  
6     CROSS APPLY GetArrayElements(tweets.emojis) AS emoji  
7 )  
8 -- store windowed calculations to Azure SQL DB every X secs  
9 SELECT emj AS ecode,COUNT(*) AS counter INTO winCountDB FROM emojis  
10     GROUP BY emj,TumblingWindow(second,5)  
11  
12 -- store a projection of raw data to Azure Blob Storage continuously as well  
13 SELECT tweets.id,tweets.text AS msg INTO rawEventsBS FROM PublicTweetStream as tweets  
14  
15 -- store windowed calculations to Azure Table Storage every X secs  
16 SELECT CONCAT('emojis_',DATENAME(yyyy,System.Timestamp),'_',DATENAME(mm,System.Timestamp),  
17 '_ ',DATENAME(dd,System.Timestamp),'_',DATENAME(hh,System.Timestamp),'_',DATENAME(mi,System.Timestamp),  
18 '_ ',DATENAME(ss,System.Timestamp)) AS emojiwindow, emj AS ecode,COUNT(*) AS counter INTO winCountTS FROM emojis  
19     GROUP BY emj,TumblingWindow(second,30)
```

# Azure Stream Analytics

## Kurzresumée

- ++ sehr zugänglich für Einsteiger
- ++ ausdrucksstarke T-SQL Query-Language
- ++ flexibles Windowing Konzept für div. Analyseaufgaben
- + viele Azure I/O Quellen out-of-the-box (Event Hub/BlobStorage,SqlDB,...)
- + nahtlose Anbindung an weitere Services z.B. Azure ML
- ~ Skalierbarkeit: Überlegungen / Konfiguration von Beginn an nötig
- leider (noch?) keine User Defined Functions (abgesehen von Azure ML)





# Kontakt

Hans-Peter Grahsl

hanspeter@grahsl.at

+43 650 217 17 04



@hpgrahsl



[https://www.xing.com/profile/HansPeter\\_Grahsl](https://www.xing.com/profile/HansPeter_Grahsl)



hans\_peter\_g

