

Willkommen beim #GWAB 2014!

# HDInsight on Azure

## Verteilte Datenanalyse mit Apache Hadoop

Hans-Peter Grahl, FH CAMPUS 02  
@hpgrahl on Twitter



Lokale Sponsoren:



# Einführung

## ■ Was ist HDInsight?

- 100% auf Apache Hadoop basierendes Azure Cloud Service
- entstanden aus enger Zusammenarbeit von Microsoft & Hortonworks
- Hortonworks HDP ist die on-premise Version für Windows Server Umgebungen
- Verfügbare HDInsight 3.0 Komponenten des Hadoop Ökosystems:

Apache Hadoop	2.2.0
Apache Hive	0.12.0
Apache Pig	0.12
Apache Sqoop	1.4.4
Apache Oozie	4.0.0
Ambari	API v1.0

# Was steckt hinter diesen Komponenten?

## Hadoop

- verteiltes System zur Daten-Speicherung & -Analyse
- typischerweise große unstrukturierte Datenmengen
  - Batch-Verarbeitung ruhender Daten
- **2 Hauptkomponenten => „Hadoop’s Kernel“**
  - **HDFS:** redundante verteilte Datenspeicherung  
Hadoop Distributed File System
  - **MapReduce:** fehlertolerantes skalierbares Programmier-Paradigma  
inkl. Ressourcen Verwaltung und Job Scheduling
- **Datenlokalität:** Berechnungen laufen auf jenen Knoten im Cluster wo Daten gespeichert sind (bzw. in maximaler Nähe dazu)



# Was steckt hinter diesen Komponenten?

## Hive

- ursprünglich von Facebook konzipiert & entwickelt
- SQL Abstraktionsschicht für MapReduce auf Hadoop
- gut geeignet für ad-hoc Analysen, sehr effiziente Entwicklung
- Performance ursprünglich mäßig, wird aber laufend besser
  - Stinger Initiative => Interactive SQL (Ziel: bis zu 100x Performance!)
- Unterstützung wichtigster SQL-Funktionalität bereits vorhanden
- Zielgruppe: Analysten mit Datenbankhintergrund



# Was steckt hinter diesen Komponenten?

## Hive

### Hive SQL Datatypes

INT
TINYINT/SMALLINT/BIGINT
BOOLEAN
FLOAT
DOUBLE
STRING
TIMESTAMP
BINARY
DECIMAL
ARRAY, MAP, STRUCT, UNION
DATE
VARCHAR
CHAR

### Hive SQL Semantics

SELECT, INSERT
GROUP BY, ORDER BY, SORT BY
JOIN on explicit join key
Inner, outer, cross and semi joins
Sub-queries in FROM clause
ROLLUP and CUBE
UNION
Windowing Functions (OVER, RANK, etc)
Custom Java UDFs
Standard Aggregation (SUM, AVG, etc.)
Advanced UDFs (ngram, Xpath, URL)
Sub-queries for IN/NOT IN, HAVING
INTERSECT / EXCEPT
Expanded JOIN Syntax

Available
Hive 0.12 (HDP 2.0)
Roadmap

Source: Hortonworks, Inc.

# Was steckt hinter diesen Komponenten?

## Pig

- ursprünglich von Yahoo konzipiert & entwickelt
- Pig als high-level Ansatz zur Abstraktion von Java MapReduce
- Pig Latin als expressive data-flow language
- simples Datenmodell: Atom, Tuple, Bag, Map
- viele an SQL angelehnte bzw. vergleichbare Sprachkonstrukte
- interaktive Konsole => Grunt Shell
- UDFs (user-defined functions)
  - Bibliotheken: piggybank, DataFu (LinkedIn), ...
- Zielgruppe: Entwickler, Datenanalysten mit Programmierkenntnissen



# Was steckt hinter diesen Komponenten?

## Sqoop

- effizientes (Bulk)Loading: Hadoop <-> strukturierten Speichersystemen
- häufigstes Anwendungsszenario: RDBMS I/O
- Cmd-Line Tool / Interface



## Oozie

- Workflow Management & Scheduling für Hadoop Job Pipelines
- unterstützt Java MapReduce & Hadoop Streaming, Hive, Pig & Sqoop

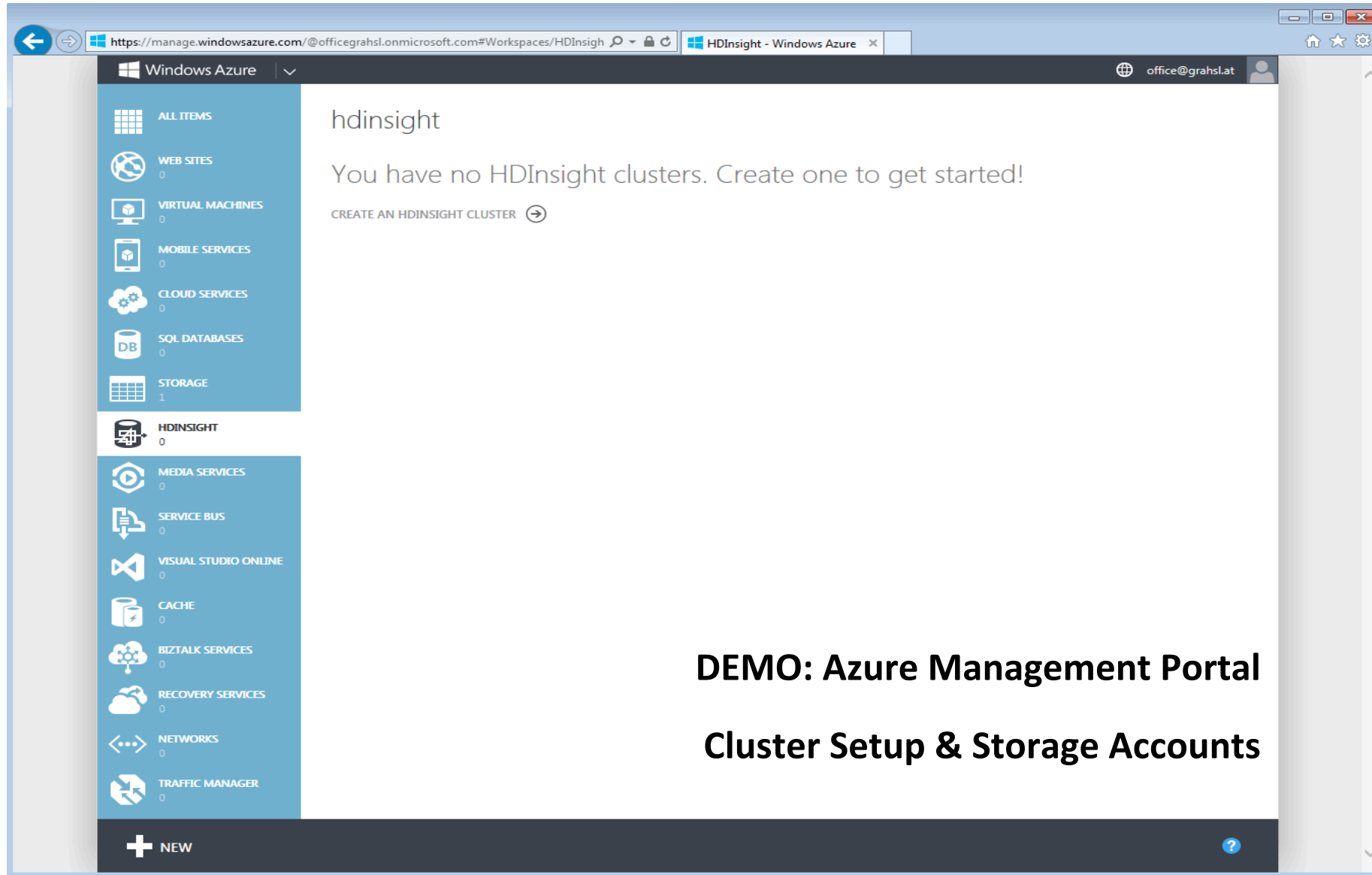


## Ambari

- Provisionierung, Management & Monitoring von Hadoop Clustern
- webUI + RESTful API



# HDI Cluster & Storage Management



The screenshot shows the Azure Management Portal interface. The browser address bar displays the URL: <https://manage.windowsazure.com/@officegrahsl.onmicrosoft.com/#Workspaces/HDInsigh>. The page title is "hdinsight". The main content area contains the text: "You have no HDInsight clusters. Create one to get started!" and a button labeled "CREATE AN HDINSIGHT CLUSTER" with a right-pointing arrow icon. The left sidebar lists various Azure services, including ALL ITEMS, WEB SITES, VIRTUAL MACHINES, MOBILE SERVICES, CLOUD SERVICES, SQL DATABASES, STORAGE, HDINSIGHT, MEDIA SERVICES, SERVICE BUS, VISUAL STUDIO ONLINE, CACHE, BIZTALK SERVICES, RECOVERY SERVICES, NETWORKS, and TRAFFIC MANAGER. At the bottom left of the sidebar is a "+ NEW" button, and at the bottom right is a help icon (question mark).

**DEMO: Azure Management Portal**

**Cluster Setup & Storage Accounts**



# HDI Cluster & Storage Management

- **Provisionierungsmöglichkeiten**
  - Web-Portal, PowerShell-Module, .NET Client Bibliotheken, X-platform CLI
- **Storage Account**
  - Regionen: Southeast Asia, North Europe, West Europe, East US, West US
- **HDInsight Cluster**
  - #1 Head-Node (fix) + #N Data-Nodes + #1 Secure-Gateway-Node (gratis)
  - Preismodell je nach Verwendung (pay-as-you-go od. monthly-plan)
  - Versionen: 1.6 & 2.1 (Hadoop 1.x) 3.0 (Hadoop 2.x)
- **Remote Access?**
  - muss explizit freigeschalten werden => RDP Connection auf Head-Node

# Head-Node Instanz

## DEMO: RDP => Head-Node Überblick auf Server 2008 R2 Instanz

Hadoop NameNode namenodehost:9000 - Windows Internet Explorer

http://namenodehost:50070/dfshealth.jsp

### NameNode 'namenodehost:9000'

Started: Mon Mar 24 15:28:10 GMT 2014  
Version: 1.2.0.1.3.6.0-0862, rdc78a9b2823375078cd3bbdba47d9d3cead67122  
Compiled: Thu Feb 20 17:40:40 Pacific Standard Time 2014 by jenkins  
Upgrades: There are no upgrades in progress.

[Browse the filesystem](#)  
[Namenode Logs](#)

#### Cluster Summary

7 files and directories, 14 blocks = 21 total. Heap Size is 270.69 MB / 3.56 GB (7%)

Configured Capacity	: 1000 GB
DFS Used	: 1.38 KB
Non DFS Used	: 8.03 GB
DFS Remaining	: 991.97 GB
DFS Used%	: 0 %
DFS Remaining%	: 99.2 %
<a href="#">Live Nodes</a>	: 1
<a href="#">Dead Nodes</a>	: 0
<a href="#">Decommissioning Nodes</a>	: 0
Number of Under-Replicated Blocks	: 1

#### NameNode Storage:

Storage Directory	Type	State
c:\hdfs\mn	IMAGE_AND_EDITS	Active

```
Hadoop Command Line
Usage: hadoop [--config confdir] COMMAND
where COMMAND is one of:
namenode -format      format the DFS filesystem
secondarynamenode   run the DFS secondary namenode
namenode             run the DFS namenode
datanode             run a DFS datanode
dfsadmin            run a DFS admin client
mradmin            run a Map-Reduce admin client
fsck                run a DFS filesystem checking utility
fs                  run a generic filesystem user client
balancer            run a cluster balancing utility
snapshotDiff        diff two snapshots of a directory or diff the
                    current directory contents with a snapshot
lsSnapshottableDir  list all snapshottable dirs owned by the current user
oiv                 apply the offline fsimage viewer to an fsimage
fetchdt             fetch a delegation token from the NameNode
jobtracker          run the MapReduce job Tracker node
pipes               run a Pipes Job
tasktracker         run a MapReduce task Tracker node
historyserver       run job history servers as a standalone daemon
job                 manipulate MapReduce jobs
queue               get information regarding JobQueues
version             print the version
jar <jar>           run a jar file

distcp <srcurl> <desturl> copy file or directories recursively
distcp2 <srcurl> <desturl> DistCp version 2
archive -archiveName NAME <src> <dest> create a hadoop archive
daemonlog           get/set the log level for each daemon
or
CLASSNAME           run the class named CLASSNAME
Most commands print help when invoked w/o parameters.
C:\apps\dist\hadoop-1.2.0.1.3.6.0-0862>
```

# Entwicklung für HDInsight

## ▪ low-level Optionen:

- MapReduce API: Java
- Hadoop Streaming: Java, **C#**, Python, ...
- **.NET SDK for Hadoop**: Wrapper API für Hadoop Streaming
  - JSON-basierte Serialisierung für den gesamten I/O Datenfluss in MapReduce möglich
  - CodePlex Incubator Project => experimentell!

## ▪ high-level Optionen:

- Pig Latin Scripts
- Hive Queries

# Ausführung von Jobs auf HDInsight

## ▪ Job Submission mit PowerShell cmdlets in 3 Schritten

### 1. Job Definition

```
New-AzureHDInsightMapReduceJobDefinition  
New-AzureHDInsightStreamingMapReduceJobDefinition  
New-AzureHDInsightHiveJobDefinition  
New-AzureHDInsightPigJobDefinition  
New-AzureHDInsightSqoopJobDefinition
```

### 2. Job Submission

```
Start-AzureHDInsightJob
```

### 3. Wait for Completion & Results

```
Wait-AzureHDInsightJob  
Get-AzureHDInsightJobOutput
```

# Ausführung von Jobs auf HDInsight

## ▪ Job Submission mit CLI am Head-Node (via RDP)

### 1. Hadoop Shell

```
hadoop jar your.jar your.package.to.JobClass arg1 ...
```

### 2. Pig Grunt Shell

```
pig -x local runs interactive grunt shell
```

```
pig (-x local) your_pig_script.pig
```

### 3. Hive Shell

```
hive runs interactive query shell
```

```
hive -f your_hive_query_file.sql
```

```
hive -e `your_inline_query_here`
```

# Ausführung von Jobs auf HDInsight

## ▪ Job Submission mit .NET Client Libraries (explizit)

- eigene API zur Verwaltung von Jobs
- sehr ähnlich zu Möglichkeiten der PowerShell cmdlets

## ▪ Job Submission mit .NET SDK for Hadoop (implizit)

- MapReduce Streaming Job Code
- kein eigener Code zur Job Verwaltung nötig  
*=> autom. Packaging, Deployment und Ausführung bei <Start> aus Visual Studio*

# C# Hadoop Streaming on HDInsight

## ▪ DEMO => C# Beispiel mit Twitter JSON Feed

- Sammlung öffentlicher Tweets via Twitter API
- JSON-basierter MapReduce Datenfluss mit .NET SDK for Hadoop
- Ausführung direkt aus Visual Studio
- Ergebnis mittels Storage Explorer ansehen / herunterladen

TASK: Für Tweets im JSON-Format sollen nach Sprachen  
gruppiert einfache Statistiken berechnet werden.

**Quick Code-Walkthrough + Job Execution + Results Checking**

# C# Hadoop Streaming on HDInsight

## DEMO RESULTS:

```
ar      {"num_tweets":506,"avg_num_chars":119,"perc_containing_urls":0.77865612648221338,"at_least_three_hashtags":176}
bg      {"num_tweets":168,"avg_num_chars":90,"perc_containing_urls":0.77976190476190477,"at_least_three_hashtags":67}
bn      {"num_tweets":3,"avg_num_chars":87,"perc_containing_urls":1.0,"at_least_three_hashtags":0}
da      {"num_tweets":368,"avg_num_chars":92,"perc_containing_urls":0.391304347826087,"at_least_three_hashtags":15}
de      {"num_tweets":4534,"avg_num_chars":112,"perc_containing_urls":0.68372298191442438,"at_least_three_hashtags":1027}
el      {"num_tweets":42,"avg_num_chars":103,"perc_containing_urls":0.69047619047619047,"at_least_three_hashtags":5}
en      {"num_tweets":135578,"avg_num_chars":114,"perc_containing_urls":0.59829028308427623,"at_least_three_hashtags":14421}
es      {"num_tweets":20482,"avg_num_chars":116,"perc_containing_urls":0.52978224782736061,"at_least_three_hashtags":667}
et      {"num_tweets":333,"avg_num_chars":74,"perc_containing_urls":0.63063063063063063,"at_least_three_hashtags":9}
fa      {"num_tweets":37,"avg_num_chars":107,"perc_containing_urls":0.7567567567567568,"at_least_three_hashtags":8}
fi      {"num_tweets":157,"avg_num_chars":83,"perc_containing_urls":0.52229299363057324,"at_least_three_hashtags":21}
...
```



# Pig on HDInsight mit PowerShell

- **Vergleich:** dasselbe Beispiel mit Pig & PowerShell...  
...nur wesentlich kompakter ;-)

PigLatin + PowerShell Source < **30(!) LOC**

# Java on HDInsight using PowerShell

## ▪ DEMO => Java Beispiel mit Twitter JSON Feed

- Sammlung öffentlicher Tweets via Twitter API
- MapReduce Datenfluss über Hadoop Writable Serialization
- Ausführung mit PowerShell
- Ergebnis mittels Storage Explorer ansehen / herunterladen

**TASK:** Für Tweets im JSON-Format soll die Verwendung der enthaltenen HashTags analysiert werden => Ziel: primitiver „HashTag-Recommendender“ basierend auf HashTag Co-Occurrences

**Quick Code-Walkthrough + Job Execution + Results Checking**

# Java on HDInsight using PowerShell

HashTag #Co-Tags

all co-occurring HashTags with frequency in DESC order

<b>Ukraine</b>	<b>2668</b>	Crimea	3974	Russia	3283	Putin	946	Россия	906	Крым	901 ...
<b>Russia</b>	<b>2406</b>	Ukraine	3283	Crimea	2434	Putin	748	G8	474	G7	317 ...
<b>Obama</b>	<b>1926</b>	WARONDRUGS	939	drugwar	939	endthedrugwar	939	endit	939	NSS2014	460 ...
<b>Crimea</b>	<b>1886</b>	Ukraine	3974	Russia	2434	Putin	974	Крым	699	Россия	594 ...
<b>Putin</b>	<b>1292</b>	Crimea	974	Ukraine	946	Russia	748	Obama	320	Russian	154 ...
<b>russia</b>	<b>1001</b>	ukraine	399	putin	247	россия	211	crimea	184	Ukraine	159 ...
<b>ukraine</b>	<b>753</b>	russia	399	crimea	285	putin	231	usa	127	украина	105 ...
<b>obama</b>	<b>618</b>	NSS2014	176	tcot	153	obamacare	114	TeaParty	95	sgp	93 ...
<b>tcot</b>	<b>533</b>	Obama	422	tgdn	313	p2	304	tlot	288	teaparty	250 ...
<b>USA</b>	<b>485</b>	Ukraine	156	Russia	156	Crimea	95	Putin	73	Obama	64 ...
<b>crimea</b>	<b>476</b>	ukraine	285	russia	184	Ukraine	150	putin	98	euromaidan	97 ...
<b>US</b>	<b>403</b>	Russia	300	Ukraine	285	EU	213	Crimea	155	Putin	84 ...
<b>EU</b>	<b>389</b>	Ukraine	366	Russia	250	US	213	Crimea	196	Putin	96 ...
<b>news</b>	<b>351</b>	p2	107	breaking	99	TFB	82	usa	74	Obama	73 ...
<b>putin</b>	<b>337</b>	russia	247	ukraine	231	war	112	путин	108	usa	108 ...

...

# Zusammenfassung

## ▪ Big Data Analysis mit HDInsight

- gängige Analyseszenarien werden sehr gut unterstützt: **MapReduce /Streaming, Pig & Hive**
- **umfassende Dokumentation & anschauliche Tutorials** von Microsoft und aus der Community
- Entwicklern & Administratoren bieten sich verschiedenste Zugänge:  
**.NET Client Libraries, Azure PowerShell Scripts/Modules, Management Portal, X-platform CLI**

## ▪ Empfehlungen für die lokale Entwicklung

- reale od. virtualisierte HDP 2.0 als on-premise Variante auf Server 2008  
*=> alternativ HDP 2.0 turnkey Sandbox basierend auf RHEL CentOS*
- HDInsight Emulator als lokale Entwicklungs-/Testumgebung auf WIN 7

2014

Global  Windows Azure  
**BOOTCAMP**

Big Data analysis has never  
been more flexible!

On Windows Azure it's easy and fun, so

**START TODAY!**

