Big Analytics in the Cloud

Matt Winkler
PM, Big Data @ Microsoft
@mwinkle
Agenda

Introduction
3 use cases
Introduction to Azure Data Lake
Dive into the productive, intelligent cloud
JustGiving is a global online social platform for giving that lets you raise money for a cause you care about through your network of friends. Their goal is to become "Facebook of Giving".

They chose Azure HDInsight, a Data Lake service to give a way to recommend to fundraisers about a potential fundraising goal, recommended causes that they might be interested in and other people in the social graph to add to their fundraising initiative.
One of the leaders in the development and management of renewable energy, infrastructure, water and services needed to understand data coming from their wind turbines/wind farms in an Internet of Things (IoT) scenario.

They chose Azure Data Lake to work with SQL Server and Excel reports to generate analytics around power/consumption of each windmill turbine. They provide querying capabilities to their customers to understand consumption.
A government organization that handles finances, taxes, budget, income, and national debt for their country. Their tax department allows clients to uploads their digital documents (pay stubs, expenditure slips) and now have billions of documents uploaded.

They chose Azure HDInsight, SQL Server, to run queries and to process the electronic invoices to gain insights. HDInsight is able to handle a peak of 150+ million invoices uploaded / day. They can now do fraud detection by understanding what people are doing and flagging/detecting anomalies.
Introducing Azure Data Lake
Big Data Made Easy

Analytics on any data, any size
All users productive on day one
Ready for your enterprise
Microsoft Azure Data Lake

Analytics Service

HDInsight

U-SQL  Spark  STORM

YARN

HDFS

Store
Proven at Microsoft

Used at Microsoft for some of the largest big data projects across Office, XBox Live, Azure, Windows, Bing and Skype

Thousands of developers

Exabytes of data under management

Using same technology powering Microsoft
Be productive without worrying about infrastructure

Deploy big data projects within minutes/seconds
No hardware to install, tune, configure or deploy
No infrastructure or software to manage
Scale to tens to thousands of machines instantly

Up and running instantly
Demo: Deployment and Management
New HDInsight Cluster

- Cluster Name: mrtinkledemo454
  - azurehdinsight.net

Cluster Type:
- Hadoop
- HBase
- Storm
- Spark (PREVIEW)
  - Ubuntu 12.04 LTS

- Subscription:
  - BDHadoopTeamPMTesDemo2

- Create a new resource group:
  - awesomeStrataDemo

Select Existing

- Credentials
  - Configure required settings

- Data Source:
  - 414141413270 (East US 2)

- Node Pricing Tiers
  - Configure required settings

Optional Configuration
Reliable
Managed, monitored, and supported by Microsoft

Managed, monitored and supported by Microsoft
Enterprise-leading SLA: 99.9% uptime
No IT resources needed for upgrades/patching
Microsoft monitors your deployment so you don’t have to
Rich tools for moving from development to production

Peace of mind that you’ll run continuously
<table>
<thead>
<tr>
<th>USER</th>
<th>ROLE</th>
<th>ACCESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDI-SPI</td>
<td>Contributor</td>
<td>Assigned</td>
</tr>
<tr>
<td>Subscription admins</td>
<td>Owner</td>
<td>Inherited</td>
</tr>
</tbody>
</table>

1. Select a role: Owner

2. Add users: None selected

- User: salvjg
  - Picture: Saikumar Singh Gill
  - Email: salvjg@microsoft.com

You haven't selected any users.
New U-SQL Job

```u-sql
@allRequests = EXTRACT UTCDate string,
   ActivityId string,
   Account string,
   operation string,
   Httpstatus string,
   Latency long
FROM @users/mwinkle/output/telemetryExplore/jobRequests3.csv
USING
Extractors.csv;

@allRequests = SELECT *, Microsoft.Analytics.Internal.Telemetry.Helpers.Dates.GetBeginningOfWeek(UTCDate).ToShortDateString() AS BusinessWeek FROM @allRequests;

@summary = SELECT DISTINCT BusinessWeek,
   operation,
   PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY BusinessWeek, operation) AS [25thPercentileLatency],
   PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY BusinessWeek, operation) AS [50thPercentileLatency],
   PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY BusinessWeek, operation) AS [75thPercentileLatency],
   PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY BusinessWeek, operation) AS [95thPercentileLatency],
   PERCENTILE_CONT(0.99) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY UTCDate, operation) AS [99thPercentileLatency]
FROM @allRequests;

@summaryCountMinMaxAvg = SELECT BusinessWeek,
   operation,
   COUNT(DISTINCT Account) AS Accounts,
   COUNT(DISTINCT ActivityId) AS Operations,
   MIN(Latency) AS MinLatency,
   MAX(Latency) AS MaxLatency,
   AVG(Latency) AS AvgLatency
FROM @allRequests;
```

Job Management

Jobs

<table>
<thead>
<tr>
<th>Date</th>
<th>Successes</th>
<th>Failed</th>
<th>Canceled</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/8/2015</td>
<td>109</td>
<td>184</td>
<td>14</td>
</tr>
<tr>
<td>9/10/2015</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/13/2015</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/23/2015</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Usage

Compute Hours

<table>
<thead>
<tr>
<th>Hours</th>
<th>Successes</th>
<th>Failed</th>
<th>Canceled</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

http://mssql.com/
Lower Total Cost of Ownership

No hardware licenses or service-specific support agreement

Pay only for what you use, when you need it, not more than you need

Independently scale storage and compute

No need to hire specialized operations team to do big data

63% lower total cost of ownership than on-premises*

*Pending IDC study found on a per TB basis, Microsoft customers using cloud-based Hadoop in Data Lake have a 63% lower TCO than on-premises
Focus on Insight, Not Limits

No limits on file sizes
Analytics scale on demand
No code rewrites as you increase size of data stored
Optimized for massive throughput
Optimized for IOT with high volume of small writes

Useful for very large data or for real-time
Demo: Large files
Be productive the tools you choose

HDInsight offers fully managed Hadoop, Spark, Storm and HBase clusters, running on Linux or Windows

Easily extend to install new tools (eg, Hue, Drill)

For ultimate control, deploy to VM’s and operate on the same data for HDFS compatible apps

Use the right tool for the job
Demo: Spark Notebooks, Hue
SELECT count(*) FROM hivesamplotable;
### SQL Query 1
```
SQL describe ny311
```

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CrossStreet1</td>
<td>string</td>
</tr>
<tr>
<td>CrossStreet2</td>
<td>string</td>
</tr>
<tr>
<td>IntersectionStreet1</td>
<td>string</td>
</tr>
<tr>
<td>IntersectionStreet2</td>
<td>string</td>
</tr>
<tr>
<td>AddressType</td>
<td>string</td>
</tr>
<tr>
<td>City</td>
<td>string</td>
</tr>
<tr>
<td>Landmark</td>
<td>string</td>
</tr>
<tr>
<td>FacilityType</td>
<td>string</td>
</tr>
<tr>
<td>Status</td>
<td>string</td>
</tr>
<tr>
<td>DueDate</td>
<td>string</td>
</tr>
</tbody>
</table>

**Note:**
- Field names and types are from the `ny311` table.

### SQL Query 2
```
SQL select complaintType, count(*) as total from ny311 group by complaintType order by total desc limit 8
```

**Graph:**
- The graph visualizes the distribution of complaints by type.
- The x-axis represents different complaint types, and the y-axis shows the count of complaints.
- The complaint types include: **Street Condition**, **GENERAL CONSTRUCTION**, **Blocked Driveway**, **PLUMBING**, **Water System**, **Street Light Condition**, **PAINT - PLASTER**, and **Illegal Parking**.
- The y-axis ranges from 0 to 456,487,000.
- The bars are grouped, and the total count is displayed for each category.
Be productive with a robust development environment

Deep integration to Visual Studio

Easy for novices to write simple queries

Robust environment for experts to also be productive

Integrated with U-SQL, Hive, and Storm

Playback that visualizes performance to identify bottlenecks and areas for optimization

Productive for novices and experts
Demo: Ibiza, Visual Studio Tooling
Be productive with U-SQL, a simple and powerful language

U-SQL

Simple and familiar, easily extensible
Unifies declarative nature of SQL with expressive power of C#
Distributed query support for rapid experimentation over all different shape and sources of data
Familiar syntax to millions of .NET developers

Empower SQL/.NET developers with big data
gt = EXTRACT date string
    , time string
    , author string
    , tweet string
    FROM "/input/MyTwitterHistory.csv"
    USING Extractors.Csv();

@m = SELECT new SQL.ARRAY<string>((
    tweet.Split(' ').Where(x => x.StartsWith("@"))) AS refs
    FROM @t;

@t = SELECT author, "authored" AS category
    FROM @t
    UNION ALL
    SELECT r.Substring(1) AS r, "mentioned" AS category
    FROM @m CROSS APPLY EXPLODE(refs) AS Refs(r);

@res = SELECT author
    , category
    , COUNT(*) AS tweetcount
    FROM @t
    GROUP BY author, category;

OUTPUT @res TO "/output/MyTwitterAnalysis.csv"
ORDER BY tweetcount DESC
USING Outputters.Csv();
;

@allRequests = EXTRACT
    , UTCDate string
    , ActivityId string
    , Account string
    , operation string
    , HttpStatus string
    , Latency long
FROM @"users/mwinkle/output/telemetryExplore/jobRequests3.csv" USING Extractors.Csv();

@allRequests = SELECT *, Microsoft.Analytics.Internal.Telemetry.Helpers.Dates.GetBeginningOfWeek(UTCDate).ToShortDateString() AS BusinessWeek FROM @allRequests;

@summary = SELECT DISTINCT BusinessWeek,
    , operation
    , PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY BusinessWeek, operation) AS [25thPercentileLatency]
    , PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY BusinessWeek, operation) AS MedianLatency
    , PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY BusinessWeek, operation) AS [75thPercentileLatency]
    , PERCENTILE_CONT(0.90) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY BusinessWeek, operation) AS [90thPercentileLatency]
    , PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY BusinessWeek, operation) AS [95thPercentileLatency]
    , PERCENTILE_CONT(0.99) WITHIN GROUP (ORDER BY Latency) OVER (PARTITION BY UTCDate, operation) AS [99thPercentileLatency]
FROM @allRequests;

@summaryCountMinMaxAvg = SELECT BusinessWeek,
    , operation
    , COUNT(DISTINCT Account) AS Accounts
    , COUNT(DISTINCT ActivityId) AS Operations
    , MIN(Latency) AS MinLatency
    , MAX(Latency) AS MaxLatency
    , AVG(Latency) AS AvgLatency
FROM @allRequests GROUP BY BusinessWeek, operation;

@totalSummary = SELECT s.BusinessWeek, s.operation,
    , scm.Acounts,
    , scm.Operations,
    , scm.MinLatency,
    , scm.MaxLatency,
    , scm.AvgLatency
FROM Table1 s INNER JOIN Table2 scm ON s.BusinessWeek = scm.Account
CREATE DATA SOURCE mwinklesqldw FROM SQLAZURE WITH
(PROVIDER_STRING = "Database=mwinklesqldw;Trusted_Connection=False;Encrypt=True", CREDENTIAL = Mwinkle_Credential,
REMOVABLE_TYPES = (bool, byte, sbyte, short, ushort, int, uint, long, ulong, decimal, float, double, string, DateTime));

@B = SELECT COUNT(*) as count FROM EXTERNAL mwinklesqldw LOCATION "SalesLT.SalesOrderDetail";
OUTPUT @B TO "users/mwinkle/output/federation/part2.txt" USING Outputters.Csv();
Demo: U-SQL
Be productive with a powerful app ecosystem

Partnering with the leading big data applications

Enabling simple, one-click deployment of apps

Integrated with your data and our services

Let your users use the best tools
Wrapping Up

Get started today

Sign up for the upcoming preview of Azure Data Lake Store and Analytics services

Visit us in the booth!