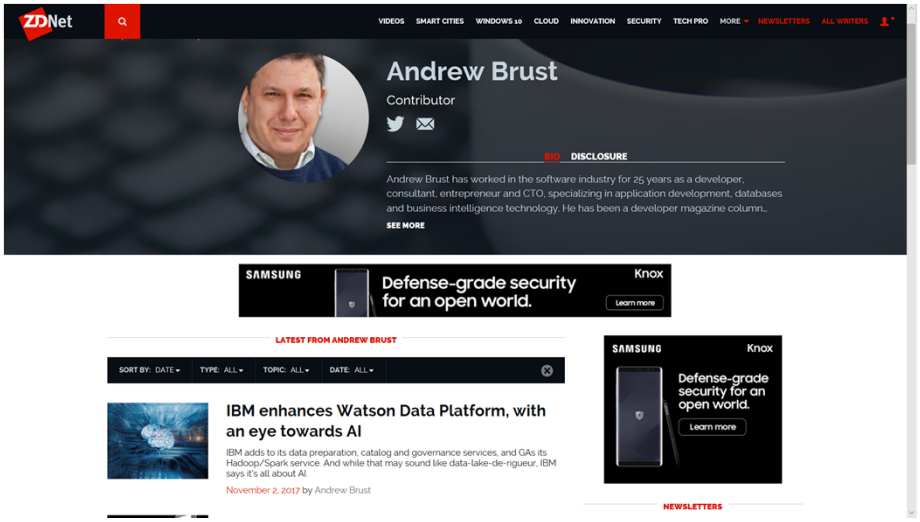

# Meet Andrew

- BLUE BADGE INSIGHTS Founder and CEO
- Big Data blogger for ZDNet
- Data and Analytics analyst for Gigaom
- Microsoft Regional Director, MVP
- Co-chair Visual Studio Live!
- Twitter: @andrewbrust

## Andrew's Blog (bit.ly/abrustzdnet)



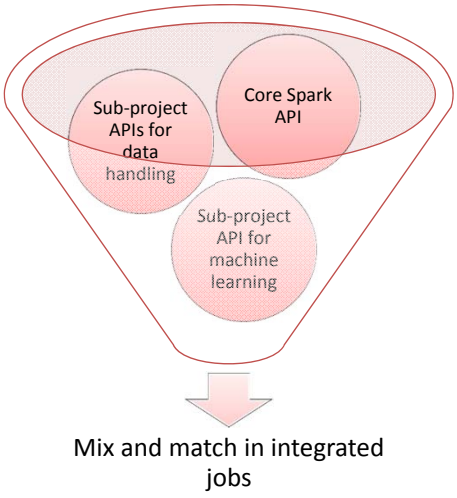

## Agenda

- Background on Apache Spark and Databricks
- Databricks workspace provisioning and core concepts
- Notebooks and the Spark programming model
- Demos
  - Blob storage, external database access
  - BI tool access
  - Machine learning

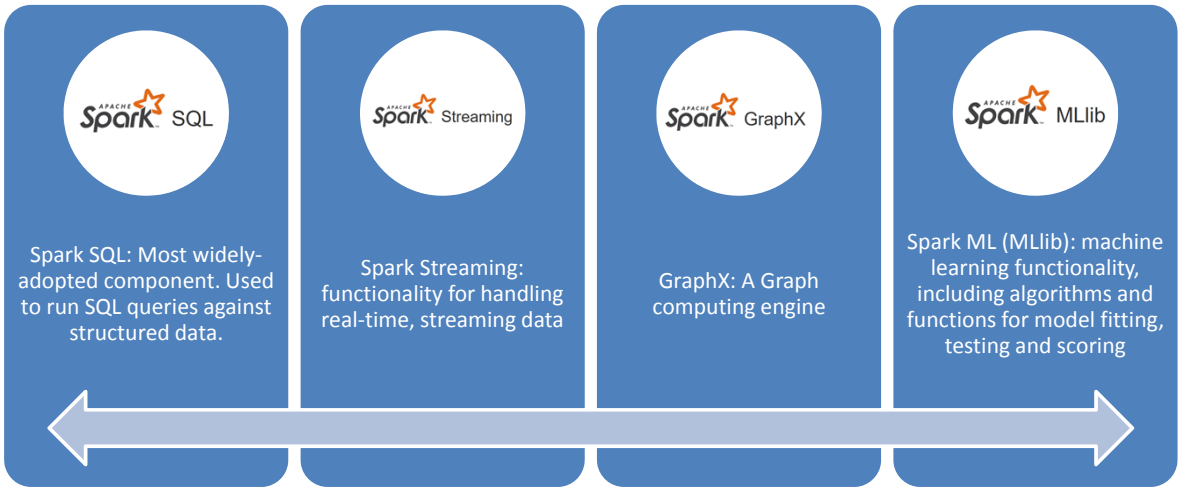
# Apache Spark

- Big Data framework, originally viewed as a competitor to Apache Hadoop
- Very memory-oriented. Compare to Hadoop's disk-heavy approach. Eliminates lots of latency.
- Great for all variety of data preparation/engineering and analytics work, but handles other workloads too
- Workloads can be comingled in a single job using very dev-friendly APIs

# Spark is More Than an In-Memory Hadoop



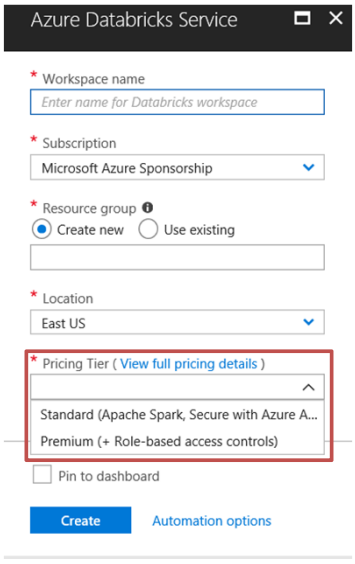
# The Sub-Projects



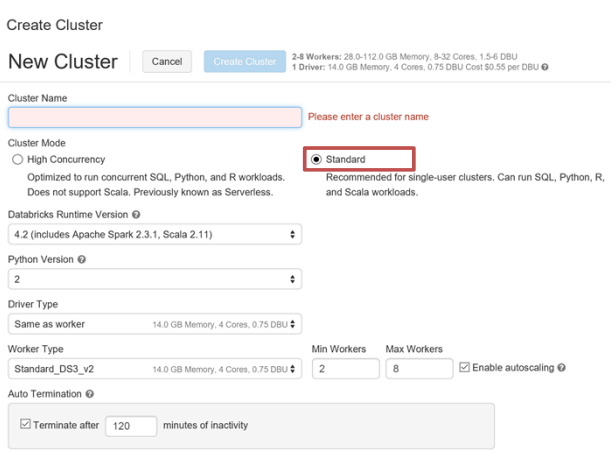

# Spark vs. Databricks

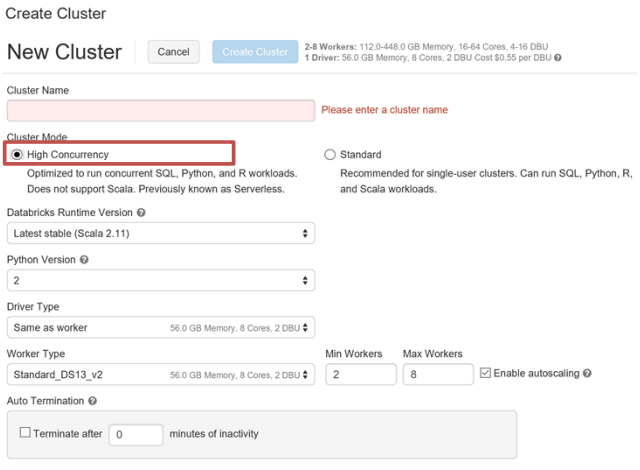
- Spark is just a framework and APIs.
  - You can integrate it with your own code, or use it from Jupyter or Zeppelin notebooks which are typically resident on the cluster
  - You set up the clusters that run it, either on-prem or in-cloud
  - The execution and/or scheduling of Spark jobs is your responsibility
- Databricks is a managed environment for Spark
  - Databricks runtime: 7x-8x faster than Apache Spark
  - You define clusters, but Databricks manages their lifecycles
  - Databricks has its own notebooks, which can be authored offline and feature a dashboard mode
  - Databricks has its own job scheduling facility, and it can be used to execute JAR file- or notebook-based jobs

# Workspace Provisioning

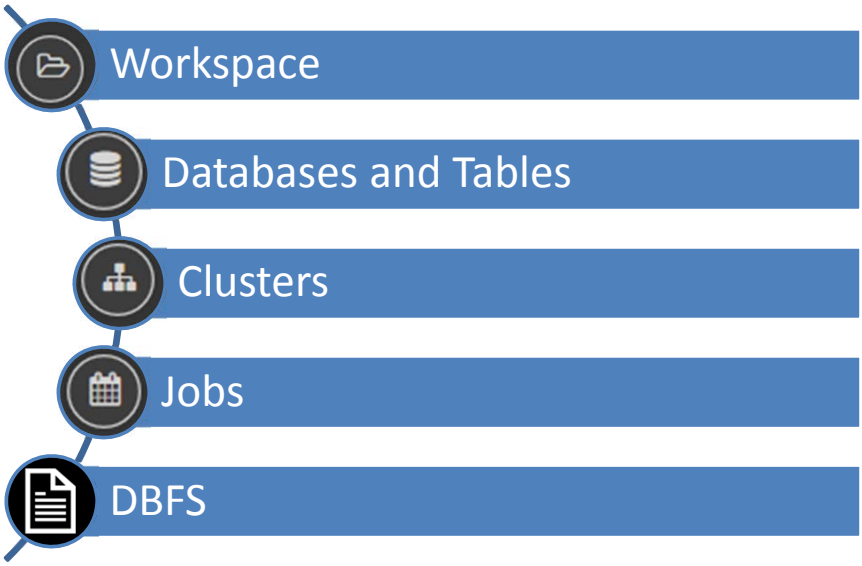



# Cluster Creation



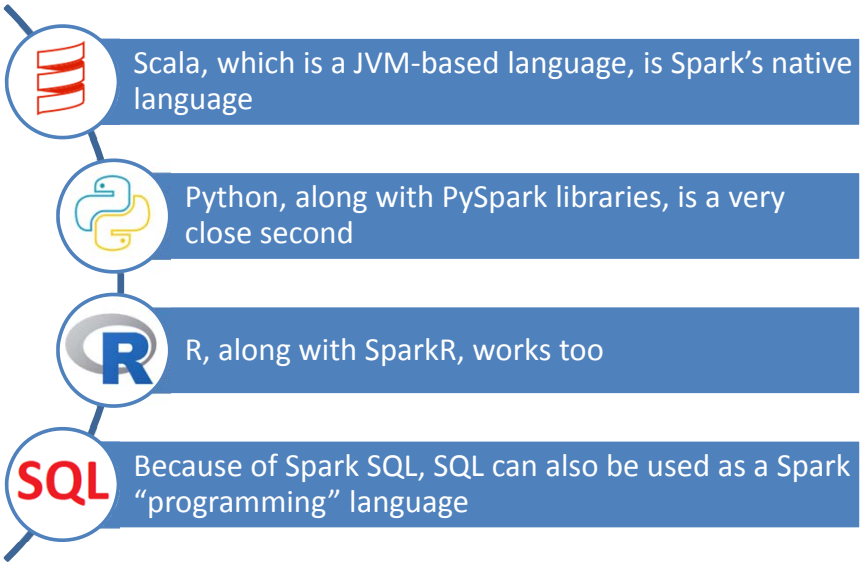
# Databricks Constructs




# Notebooks

- Think of them as Wikis with (executable) code regions and textual, tabular or data viz output
- Spark is very often paired with Jupyter or Apache Zeppelin notebooks
- But Databricks has its own notebook format and tooling
  - Can import/export from/to Jupyter notebooks
- Notebooks have a default language but accommodate a mix through use of "magics"
  - Just prefix code cell with `%python`, `%sql`, etc.
  - Can perform DBFS command with `%fs`

# Programming Languages




# Spark Fundamental Data Structures

- RDD – resilient distributed data set
- DataFrame – versatile and similar in character to namesake in Python/Pandas and R
- Dataset – newest structure and most "endorsed." Still, DataFrames seem to be most adopted, by far.

# Transformations and Actions

- Data operations in Spark are “lazy”
- Transformations not applied until an action is taken
- Examples of transformations: `select()`, `filter()`
- Examples of actions: `take()`, `first()`, `show()`
- **You won’t see errors on your transformations until you attempt an action!**

# More on Spark SQL

- Compatible with Apache Hive tables, and uses Hive’s dialect of SQL: HiveQL, aka HQL
- Has JDBC driver and can work well with BI tools
- Tight relationship between Spark SQL and DataFrame APIs
- To cache data in memory, use:
  - API: `spark.table("tablename").cache()`
  - SQL: `CACHE TABLE tablename`

# Databricks Integrations

- **Blob storage**
  - Gain access via DBFS APIs
- ADLS
  - https://docs.microsoft.com/en-us/azure/azure-databricks/databricks-extract-load-sql-data-warehouse
- **ADLS Gen 2**
  - https://docs.databricks.com/spark/latest/data-sources/azure/azure-datalake-gen2.html
- SQL Data Warehouse
  - https://docs.azuredatabricks.net/spark/latest/data-sources/azure/sql-data-warehouse.html
- **Power BI**
  - https://docs.azuredatabricks.net/user-guide/bi/power-bi.html
- Cosmos DB
  - https://docs.databricks.com/spark/latest/data-sources/azure/cosmosdb-connector.html
- Event Hubs

## Blob Storage Interop and ADLS Gen 2 Driver

# DEMO

# External Database Access

- Code in notebooks can leverage JDBC to gain access to external data
- This works well for services like Azure SQL Database
- As with ODBC, you'll need:
  - Driver name, server hostname, schema and database name, user name, password and a SQL query
- Once the data is acquired, you can manipulate with DataFrame API or SQL. You can also persist it as Databricks/Spark SQL table


Database Access (Azure SQL DB)

# DEMO

# BI Tool Access

- BI tools can connect to data in Spark SQL tables, via a JDBC connector *into* Databricks
- This only works with *Premium* workspaces
- Use Spark UI for connection string (and get ready to modify it)
- Use the Access Tokens tab of the User Settings screen for a personal access token
  - Token id will be password; user name will be “token”
- Fully supported from Power BI, including DirectQuery


Power BI and Azure Databricks

# DEMO

# Machine Learning Basics

- Classification, regression, clustering, recommendation
- Algorithms, parameters
- Features and label
- Training (fitting) and testing
- Scoring (inference)

# Machine Learning with Databricks

- randomSplit
- org.apache.spark.ml. (…)
  - VectorAssembler
    - setInputCols(featureCols), setOutputCol("features")
  - regression.LinearRegression (etc.)
    - setFeaturesCol("features"), setLabelCol("column"), .setPredictionCol("column")
    - fit(training), transform(test)

Data Science/ML

# DEMO

# Resources

- Spark home page
  - http://spark.apache.org
- Azure Databricks documentation
  - https://docs.azuredatabricks.net/user-guide/getting-started.html
- Databricks Unified Analytics Platform homepage
  - https://databricks.com/product/unified-analytics-platform
- Databricks blog
  - https://databricks.com/blog

# Thank You!

- Email
  - andrew.brust@bluebadgeinsights.com
- Twitter
  - @andrewbrust on twitter