SAS® GLOBAL FORUM 2017

April 2 – 5 | Orlando, FL

SAS® In-Memory Analytics for Hadoop

USERS PROGRAM

Paper 1271 - 2017

SAS® In-Memory Analytics for Hadoop

Venkateswarlu Toluchuri, United Health Group (Optum) , Hyderabad, India

Paper 1271- 2017

Author: Venkat T., SAS Administrator/Developer
Presenter: James Chris, SAS Administrator/Manager
Company: Optum, a UnitedHealth Group business (www.optum.com)

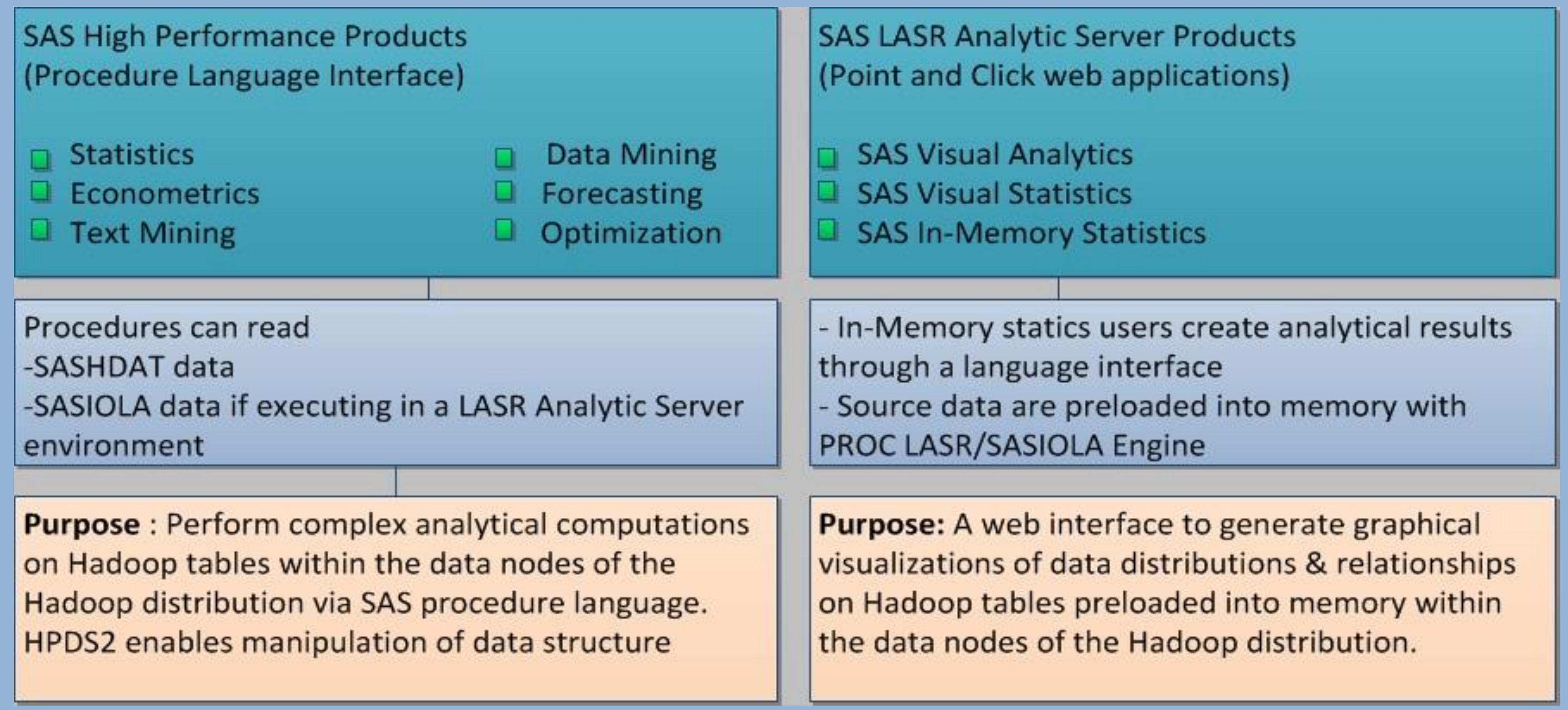
SAS Fraud Framework

The Optum solution uses SAS's Fraud Framework and Optum's deep health care expertise and extensive health care claims and fraud case datasets to identify and prevent instances of fraud, waste and abuse for payers. The solution delivers broad detection capabilities including rules, flags, predictive modeling, text mining and SAS Visual Analytics to identify possible instances of provider and consumer fraud, including multi-party fraud schemes and organized crime.

Challenges

- ➢ **Understand LASR Server architecture**
- ➢ **Data flow in to LASR Server**
- ➢ **Different techniques of loading data in to SAS LASR**
- ➢ **Understand the analytics life cycle process in SAS In-Memory**
- ➢ **Different statements in PROC IMSTAT**

Types of SAS® In-Memory Analytics Products



SAS High Performance Products
(Procedure Language Interface)

- 🟩 Statistics
- 🟩 Econometrics
- 🟩 Text Mining
- 🟩 Data Mining
- 🟩 Forecasting
- 🟩 Optimization

Procedures can read
-SASHDAT data
-SASIOLA data if executing in a LASR Analytic Server environment

Purpose : Perform complex analytical computations on Hadoop tables within the data nodes of the Hadoop distribution via SAS procedure language. HPDS2 enables manipulation of data structure

SAS LASR Analytic Server Products
(Point and Click web applications)

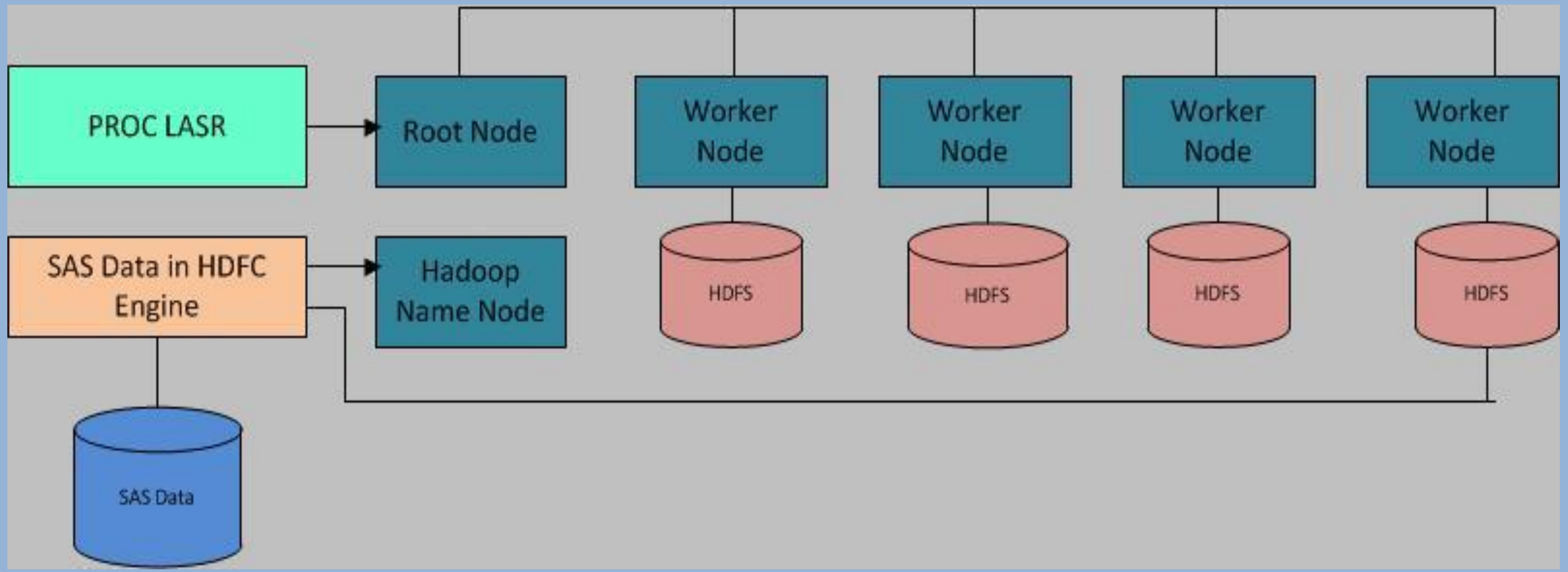
- 🟩 SAS Visual Analytics
- 🟩 SAS Visual Statistics
- 🟩 SAS In-Memory Statistics

- In-Memory statics users create analytical results through a language interface
- Source data are preloaded into memory with PROC LASR/SASIOLA Engine

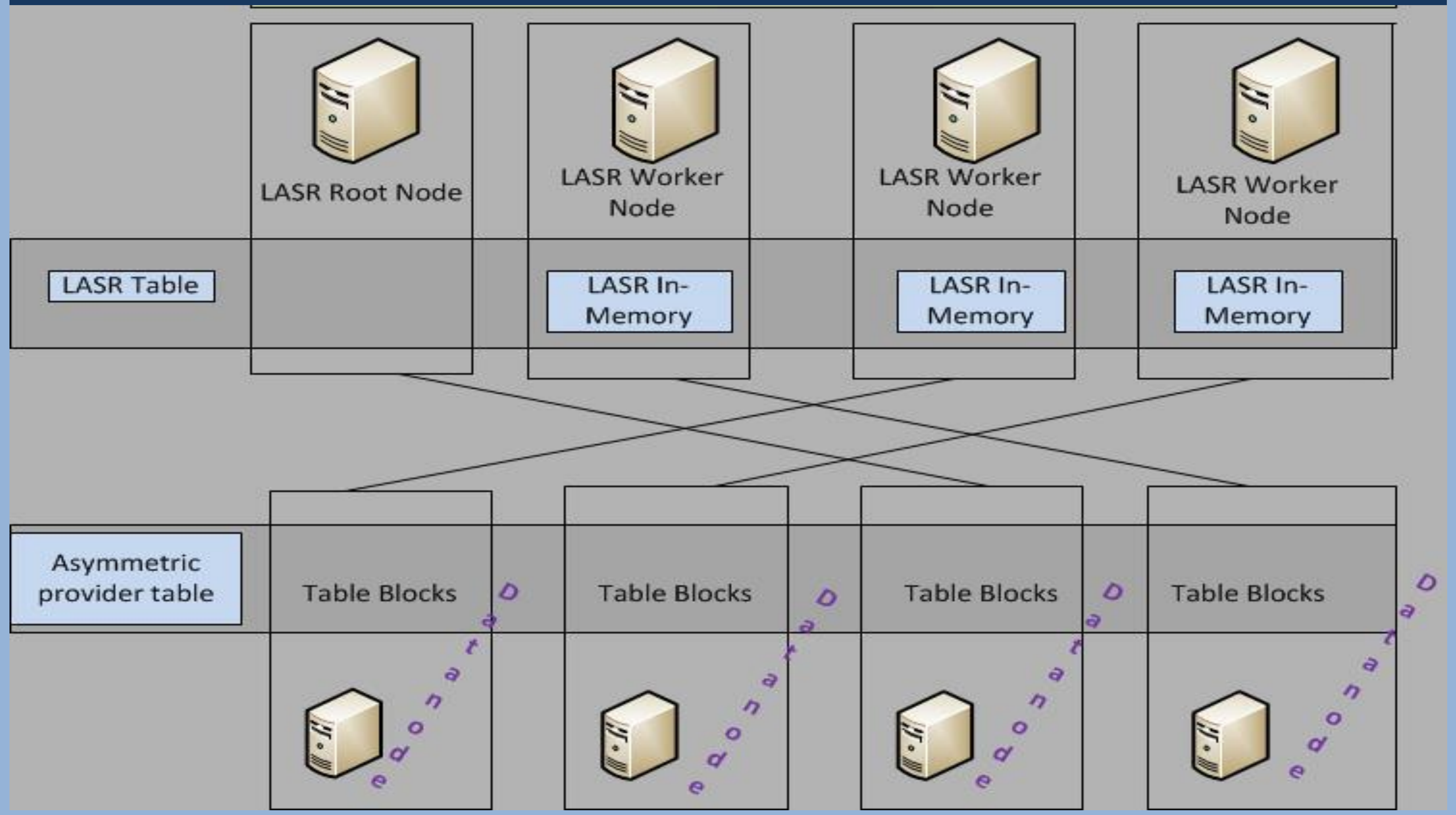
Purpose: A web interface to generate graphical visualizations of data distributions & relationships on Hadoop tables preloaded into memory within the data nodes of the Hadoop distribution.

LASR and Hadoop

The LASR Analytic Server integrates with Hadoop by reading and writing SAS data in SASHDAT format in the Hadoop Distributed File System (HDFS).



Parallel (Asymmetric) Data Load



- ▪ **The data is not co-located.**
- ▪ **LASR table blocks exist on dedicated hardware while the asymmetric provider table blocks exist on separate hardware.**
- ▪ **The blocks are pushed from the data provider into LASR just like with co-located data except that, they travel across a dedicated network.**
- ▪ **The number of provider nodes does not have to be equal to the number of LASR nodes (thus the term, asymmetric)**
- ▪ **Data does not pass through the LASR Head node for distribution. The blocks are pushed straight from the provider into the LASR worker nodes.**
- ▪ **The mapping algorithm, that maps blocks to worker nodes, is extremely simple and tries to distribute the blocks as evenly as possible.**
- ▪ **The SAS Embedded Process (EP) must be installed on the parallel data provider.**

OPTUM™

Paper 1271 - 2017

SAS® In-Memory Analytics for Hadoop

Venkateswarlu Toluchuri, United Health Group (Optum) , Hyderabad, India

Data Load Engine Techniques

SASIOLA	All data sources that a SAS procedure or a DATA step can be loaded into a LASR Analytic Server by this technique. The Libname could point to SAS Tables, Hadoop, Oracle, DB2 etc. Data loading via SASIOLA does not happen in parallel. It is therefor not an efficient way to load big data. However data loading via SASIOLA you can: Append Partition Join
SASHDAT	Data stored permanently in HDFS in the distributed data nodes. Data read in to memory in parallel by SAS High Performance procedures and released from memory when completed execution. SASHDAT files use a proprietary SAS format that High Performance Procedures and PROC LASR can read. To understand SASHDAT files, let's consider a SAS dataset broken up into blocks and distributed across a multi-machine Hadoop file system. Each block contains its subset of the data in along with header and metadata information that allows the LASR Analytic Server to read it very efficiently. The SASHDAT Libname engine deals with SASHDAT files and CSV files. The engine is uni-directional for data; it does not bring data from a SASHDAT file back to the SAS session. Data moves only from the SAS session into the SASHDAT format.
HDFS	Data stored permanently in HDFS in the distributed data nodes. Read and write Hive tables exactly like SAS Datasets. Load BASE SAS table in to LASR Server: libname lasrlib sasiola host="xxxxxxx" port=10090; data lasrlib.sashelp.class; set sashelp.class; run;

Partitioned Table – In Memory

SAS in-Memory tables can be partitioned. A partition is a set of data that shares the same key. In the LASR Analytic Server a partition implies that all observations in the partition reside on the same node.

Partitions are constructed based on the formatted values of the partition variables. If you partition using multiple variables the server constructs a single character key based on the variables.

How to Create a Partitioned Table:

Partitioned tables can be created using PARTITION= data step option

By loading a SASHDAT file that has been previously partitioned

IMSTAT statements that create temporary partitioned tables which can be promoted to memory

IMSTAT statements with the GROUPBY= option

Pros – Better performance when analysis is conducted by the partition keys

Cons – Can result in slower performance as a result of unbalanced workloads across worker nodes.

PROC IMSTAT

The IMSTAT procedure is used to perform in-memory analytics on tables that are loaded in a SAS LASR Analytic Server instance. When you use the IMSTAT procedure the:
LASR Analytic Server performs the analytic processing
Results of the analytic operations can be returned to the SAS session and some results can be saved to a temporary table in memory.

PROC IMSTAT Allows a Base SAS session to interact with a LASR Server via the SASIOLA Libname.

Manipulate Data
Compute statistics
Create new tables in-memory
Return results/data to Base SAS.

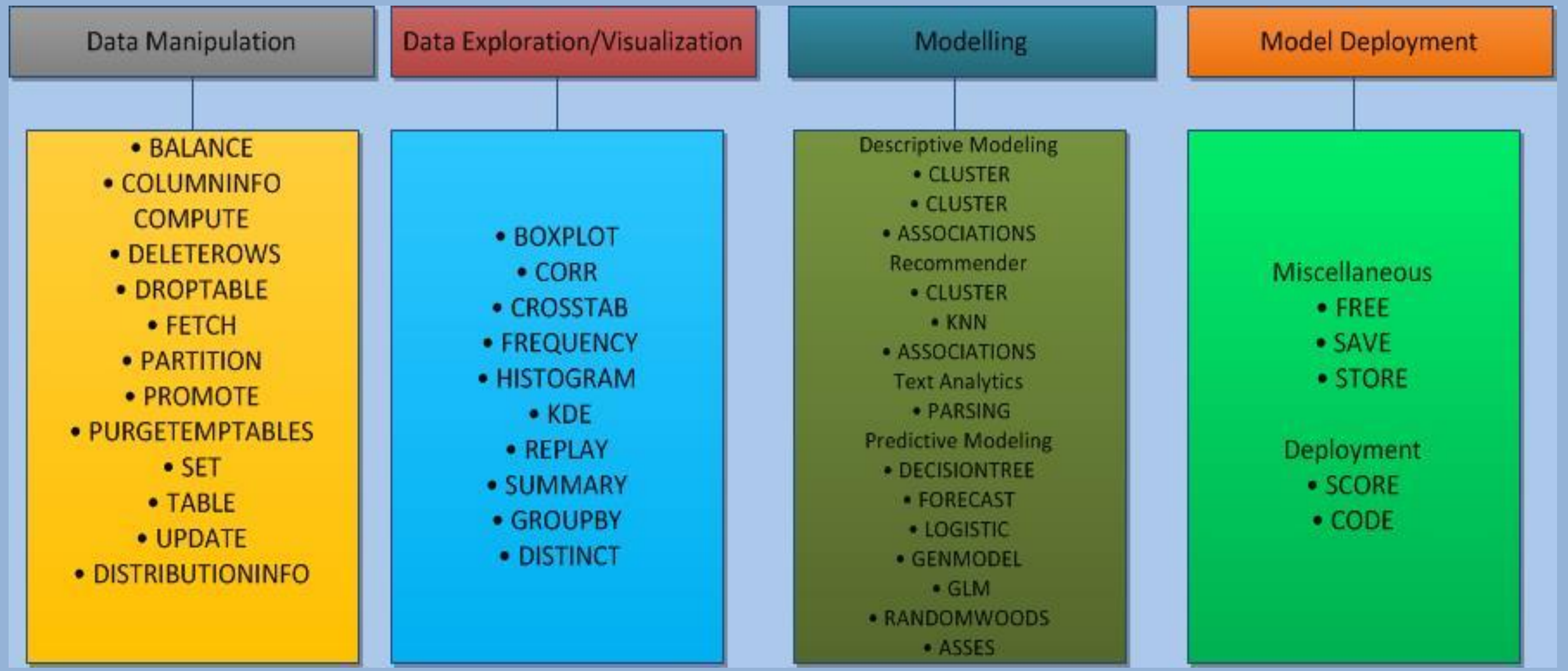
Advanced LASR Loading Techniques Comparison

COMPARISION	BASE SAS	PROC LASR	PROC HPDS2
Partition	YES	NO	NO
Append	YES	NO	NO
HDAT Source	NO	YES	YES
Filter, Transform HDAT	NO	NO	YES
Table Name and Tag	YES	NO	YES
Join, Merge	YES	NO	NO
Distributed Processing	NO	YES	YES
HDAT Block to In-Memory Load	NO	YES	NO
HDAT Memory Mapping	NO	YES	NO

Analytics Life Cycle support SAS In-Memory

Prepare Data	Explore Data	Develop Models	Scoring
• Access structured, unstructured data • Data filtering • Join tables, promote tables, compute columns • Group filtering, partitioning, data ordering within partitions	• Access structured, unstructured data • Data filtering • Join tables, promote tables, compute columns • Group filtering, partitioning, data ordering within partitions	• Explore multiple modeling approaches using advanced analytical and machine learning algorithms • Combine structured and unstructured data for predictive analytics • Classification, predictions	• Integration with SAS Model Manager for model management and model performance monitoring • Generation of SAS data step code to execute model on new data

Proc IMSTAT Statements



Data Manipulation
- BALANCE
- COLUMNINFO
COMPUTE
- DELETEROWS
- DROPTABLE
- FETCH
- PARTITION
- PROMOTE
- PURGETEMPTABLES
- SET
- TABLE
- UPDATE
- DISTRIBUTIONINFO

Data Exploration/Visualization
- BOXPLOT
- CORR
- CROSSTAB
- FREQUENCY
- HISTOGRAM
- KDE
- REPLAY
- SUMMARY
- GROUPBY
- DISTINCT

Modelling

Descriptive Modeling
- CLUSTER
- CLUSTER
- ASSOCIATIONS

Recommender
- CLUSTER
- KNN
- ASSOCIATIONS

Text Analytics
- PARSING

Predictive Modeling
- DECISIONTREE
- FORECAST
- LOGISTIC
- GENMODEL
- GLM
- RANDOMWOODS
- ASSES

Model Deployment

Miscellaneous
- FREE
- SAVE
- STORE

Deployment
- SCORE
- CODE

SAS® GLOBAL FORUM 2017

April 2 – 5 | Orlando, FL