Exam 2: SAS Big Data Programming and Loading

SAS and Hadoop - 30%

Describe the baseline requirements for interacting with Hadoop

- SAS_HADOOP_JAR_PATH and SAS_HADOOP_CONFIG_PATH environment variables
- JAR file requirements
- Hadoop XML configuration file contents
- JAR files and XML files must be accessible to the SAS Server
- Understand the precedence of settings for Hadoop XML configuration files
- Identify the components of a SAS and Hadoop solution
- List the communication paths between the components of a SAS and Hadoop solution

Use the HADOOP procedure and the Hadoop FILENAME statement to interact with Hadoop from a SAS session

- Know which HDFS commands are available through the Hadoop procedure
- Submit HDFS file system commands (DELETE, MKDIR, RENAME, CHMOD, LS, CAT)
- Copy files between SAS and Hadoop via COPYFROMLOCAL and COPYTOLOCAL statement
- Submit MapReduce programs with the MAPREDUCE statement
- Understand best practice considerations when using the FILENAME statement
- use the FILENAME statement to read data from and write data to the Hadoop file system in a SAS DATA step
- Execute Pig code with the PIG statement in the HADOOP procedure

Query and manage Hive tables stored in Hadoop using explicit SQL pass-through

- Manage connections to Hive with the CONNECT/DISCONNECT statements (schema, server, username, password, etc)
- Access Hive metadata via SHOW and DESCRIBE statements
- Select data from tables with HiveQL (select, from, where clauses)
- Join tables with HiveQL
- Use both HiveQL and SAS SQL features (ORDER BY, functions, labels) in the same SQL procedure SELECT statement
- Create SAS data sets and views from Hive results
- String dates vs. SAS dates
- Using the CAST function to control data type in explicit queries (32k string lengths)

- Create Hive table definitions
- Load data into Hive table defitions from local data
- Load data into Hive table definitions from HDFS data
- Control length of character variables created in Hive tables
- Work with Hive string types in SAS
- Control Hive table properties with TBLPROPERTIES statement (SASFMT or with data set option DBSASTYPE= option)
- Compare managed and external Hive tables
- Compare different Hive file types (textfile, sequencefile). Use of SERDEs
- Use data set options to define specific HDFS file types

Work with Hadoop files using the SAS/ACCESS LIBNAME statement

- Write a LIBNAME statement to access Hive tables
- Access Hive metadata using LIBNAME statement and the CONTENTS procedure
- Understand that the SAS/ACCESS engine writes database-specific SQL code when using implicit pass-through
- Maximize the use of HiveQL by optimzing implicit pass-through (summarization, subsetting, joins)
- Use system options to determine where processing occurs (SASTRACE, NOSTSUFFIX, SASTRACELOC)
- Evaluate SAS logs to determine the amount of implicit pass-through performed by a SAS program
- Identify SAS data set options that can be implicitly passed to Hive
- Identify SAS functions that can be passed to Hive
- Convert date formats with the SASDATEFMT= data set option
- Embed LIBNAME statements in SQL View defintions
- Identify best practices when combining tables to maximize Hive usage
- Methods to combine/join tables
- Copy data sets to Hive using the COPY procedure
- Identify advantages of using SAS/ACCESS LIBNAME method
- Identify disadvantages of using SAS/ACCESS LIBNAME method
- Maximize performance when using the LIBNAME statement
- Efficient methods for BY GROUP processing with in-database procedures
- Managing data types for computed columns.
- Partition and cluster Hive tables
- Create Hive external tables

SAS DS2 Programming - 30%

Write DS2 programs

- Utilize run group processing
- Use DATA, ENDDATA, and RUN statements properly
- Use system methods, INIT(), RUN(), TERM()
- Build user defined methods
- Pass arguments to user defined methods
- Explain the use of the INIT(), RUN(), TERM() system methods
- Use the OVERWRITE option
- Understand how DS2 handles reserved keywords
- Recognize components of traditional DATA Step programming that are or are not supported in DS2

Read data using DS2

- read data with a SET statement
- write FedSQL code within SET statements to read data
- Use FedSQL SELECT statements to extract specific variables from input data sets
- Use FedSQL Join statements to merge data from multiple input data sets
- Use FedSQL WHERE statements to extract specific observations from input data sets
- Use the MERGE statement to join data.
- Subset data using subsetting IF statements
- Read table data with a BY statement, without pre-sorting the data
- Use a FedSQL query with an ORDER BY clause to provide sorted data to the SET statement for BY group processing

Work with variables, arrays, and ANSI SQL data types

- Define and use local and global variables (understand scope, what goes into PDV, output data sets)
- Declare variables with the DCL statement
- Use fractional, integer, character and Date & Time ANSI SQL data types
- Use CHAR, NCHAR, VARCHAR, NVARCHAR character data types
- Use DECIMAL, DOUBLE, FLOAT, REAL fractional numeric data types
- Use BIGINT, INTEGER, SMALLINT, TINYINT interger numeric data types
- Use BINARY, VARBINARY binary data types
- Use DATE, TIME, TIMESTAMP date and time data types
- Identify coercible and non-coercible data types
- Understand autoconversion of DS2 data types when DS2 variables are output to SAS data sets
- Select variables with KEEP and DROP statements and KEEP= and DROP= options
- Understand how SAS will perform automatic type conversions
- DS2SCOND option statement

- Use ANSI quoting standards in variable assignment statements
- Use macro variables within ANSI quoted variable assignment statements (%TSLIT macro)
- set variable attributes (Length, format, informat) within variable declaration statements
- Use the VARARRAY statement to declare arrays
- Use the DCL staement to declare temporary arrays
- Assign values to array variables
- Understand the difference between SAS MISSING and ANSI NULL data values
- Invoke ANSI data processing mode for NULL values with the ANSIMODE option

Use expressions and functions in DS2 programs

- Use the DS2 IF expression in place of IF/THEN conditional statements
- Use the DS2 LIKE expression to compare character values to specific patterns
- Convert SAS datetime variables to DS2 ANSI TIMESTAMP variables with the TO_TIMESTAMP function
- Convert SAS date variables to DS2 ANSI DATE variables with the TO_DATE function
- Convert SAS time variables to DS2 ANSI TIME variables with the TO_TIME function
- Convert DS2 ANSI DATE, TIME, and TIMESTAMP variables to SAS date, time and datetime variables with the TO_DOUBLE function
- Increment date and time values with the INTDT and INTTS functions
- Execute FedSQL statements with the SQLEXEC function

Work with Methods, Packages, and Threads

- Create methods that modify parameters at the site by using IN_OUT variables
- Create methods that return a value using a RETURN statement
- Overload methods by creating methods with multiple signatures
- Create user defined packages with the PACKAGE statement
- Understand the capabilities of predefined DS2 packages (such as FCMP, SQLSTMT, HASH, JSON)
- Intantiate DS2 packages with the DECLARE statement
- Use threading to alleviate CPU bound processes
- Create threads using the THREAD statement
- Declare instances of threads in a DS2 program
- Call threads using a SET FROM statement
- Specify the number of threads using a THREADS= option
- How to run threads inside parallel databases
- DS2ACCEL = YES option
- Requirements to execute DS2 code in-database

Hadoop Programming - 15%

Describe the Hadoop architecture

- Identify Hadoop elements such as Name Nodes, Data Nodes, Job Trackers, Task Trackers, YARN
- Explain Hadoop concepts such as distributed storage & processing, splits, replication, MapReduce
- Describe components of the Hadoop ecosystem (Hive, Pig, Sqoop)
- Describe attributes of big data
- Identify use cases for Hadoop

Manipulate and load data files using command line tools

- Use Linux shell commands (ls -ltr/pwd/mkdir/cd)
- Load and manipulate data into Hadoop using Linux commands (hdfs dfs -mkdir/put/copyFromLocal/ls/cat)
- Use Sqoop to move data from a RDBMS into Hadoop

Write Hive programs to create, join, and query data tables

- Create databases and tables in Hive
- Understand the difference between external and internal tables
- Work with Hive variable types
- Load data into Hive table definitions with LOAD and INSERT statements
- Recognize challenges when importing data (embedded delimiter characters, header values)
- Limit values returned by a Hive query with the SELECT.....LIMIT keyword
- Sort Hive query results with the SELECT...ORDER BY keyword
- Group Hive query results with the GROUP BY keyword
- Choose which values to select from a data table using the SELECT WHERE keyword
- Retrieve unique values with the SELECT DISTINCT
- Join tables (Inner, Outer, Left, Right)
- Use functions in Hive queries (sum, count, avg, max, min, round, floor, ceil, rand, concat, substr, upper, ucase, lower, lcase, trim)
- Use relational and arithmetic operators in Hive queries

Write Pig programs to perform ETL tasks and to analyze large data sets

- Identify Pig data types
- Build Pig programs with LOAD, FOREACH/GENERATE, FILTER, SPLT, LIMIT, UNION, DISTINCT, ORDER, GROUP, STORE, DUMP keywords
- Use name and positional references in Pig programs
- Identify valid identifiers (start with letter, then letters, digits, underscores)
- Use Arithmetic, String, and Boolean Expressions

- Use the CAST operator to change variable types
- Increase parallel processing with the PARALLEL keyword
- Combine data from multiple tables with INNER, LEFT, RIGHT, and OUTER JOIN keywords
- Combine data using special join types: REPLICATED, SKEWED, MERGE
- Use parameters in a Pig program
- Use Diagnostic operators: DESCRIBE, EXPLAIN, DUMP, ILLUSTRATE
- Use functions in Pig programs
- EVAL Functions: AVG, SUM, CONCAT, COUNT, COUNT_STAR, IsEmpty, MIN, MAX, SIZE, SUBTRACT, TOKENIZE
- DATE Functions: CurrentTime, DaysBetween, HoursBEtween, GetDay, GetHour (etc.), AddDuration, ToUnixTime, ToDate, ToMilliSeconds, ToString
- String Functions: STARTSWITH, ENDSWITH, REGEX_EXTRACT, REPLACE, TRIM, LTRIM, RTIM, INDEXOF, LAST_INDEX_OF, LOWER, UPPER, LCFIRST, UCFIRST SUBSTRING, EqualsIgnoreCase
- Math Functions: ABS, ACOS, ATAN (etc) SQRT, CBRT, Exp, CEIL, FLOOR, LOG, LOG10, RANDOM, ROUND
- Tuple, Bag, Map functions: TOTuPLE, TOBAG, TOMAP, TOP
- Register and use User Defined Functions

Data manipulation with the IMSTAT procedure - 25%

Execute IMSTAT procedures

- Define a SASIOLA library to access in-memory data in a LASR Analytic Server
- Describe the key functionality of the IMSTAT procedure
- Perform one-dimensional numerical exploration with IMSTAT procedure statements SUMMARY and FREQUENCY
- Perform two-dimensional numerical exploration using the CROSSTAB or GROUPBY=option

Perform actions required to produce graphs with PROC IMSTAT

- Use PROC IMSTAT statements and options that calculate summary statistics for graphing
- Transfer the summary statistics tables to the SAS server

Manipulate In-Memory Data

- Define WHERE clauses to explore subsets of an in-memory table
- Create permanent columns using the COMPUTE statement
- Create temporary columns using temporary expressions of computed columns
- Work with SAS formats in the IMSTAT procedures
- Use the Fetch statement to retrieve data from an in-memory table
- Join in-memory tables

Use High-Performance procedures with the SAS LASR Analytic Server

- Compare the SAS High-Performance procedures and SAS IN-Memory Statistics
- Use the HPIMPUTE procedure to add imputed columns to an in-memory table

---

Note: All 17 main objectives will be tested on every exam. The 165 expanded objectives are provided for additional explanation and define the entire domain that could be tested.