**Tutorial 2**

* **Introduction - Set up Python, Pycharm and Spark on Windows 10**

**Tutorial 3**

* **Understanding the Pre- requisites**

**Tutorial 4**

* **Setup Python on Windows 10**

**Tutorial 5**

* **Configure environment variables for python on Windows 10**

**Tutorial 6**

* **Setup Pycharm on Windows 10**

**Tutorial 7**

* **Develop Python Program using Pycharm**

**Tutorial 8**

* **Download Spark - Compressed tar ball**

**Tutorial 9**

* **Install 7z to uncompress and untar on windows 10**

**Tutorial 10**

* **Setup JDK 1.8 on windows 10 and configure environment variables**

**Tutorial 11**

* **Configure environment variables for spark**

**Tutorial 12**

* **Setup winutils to integrate HDFS API'S with Windows**

**Tutorial 13**

* **Develop pyspark program using Pycharm on Windows 10**

**Tutorial 14**

* **Python fundamentals -Introduction (for Spark)**

**Tutorial 15**

* **Python fundamentals - Basic programming Constructs**

**Tutorial 16**

* **Python fundamentals - Functions and lambda functions**
  + Step 1 - defining a function with the help of previous example by creating sum of lower bound and upper bound functions.
  + Step 2 -\* Introduction to lambda function (passing functions as arguments to other functions).
  + Lambda means functions without name

**Tutorial 17**

* **Python fundamentals- Collections**
  + **List -** basic collection command in python, defined by l = []
  + Steps involved - printing  elements from right and left side, deleting elements using pop with in a given list
  + **Set-** diff b/w set and list is set does not allow duplicate values, defined by s = ()
  + Steps involved - printing elements using set functions - intersection, union, difference, converting set to list
  + **Dict -** Is a set of key value pairs, defined by d = {}, the values are separated by comma, with key colan ':' value
  + Steps involved - printing values and keys using d.values(), d.keys() & d.items()

**Tutorial 18**

* **Python fundamentals- Basic** **Map-reduce operations**
  + Map-reduce functions are used on collections only
  + Steps involved - using filter, map, reduce with lambda function. Showed the diff b/w conventional program and map reduce operations.

**Tutorial 19**

* **Scala or Python - Setup data set for basic I/O operations**
  + downloading the required data from git repository, setting the path

**Tutorial 20**

* **Python Fundamentals - IO operations and processing data from files**
  + Reading data from files
  + Steps involved - using map, reduce, filter operations on the files

**Tutorial 21**

* **CCA175 Introduction and curriculum**
  + **Data Ingest**
    - Sqoop
    - HDFS
    - Flume
    - Kafka
    - Spark Streaming
  + **Transform, Stage and Store**
    - Loading Data from HDFS
    - Write data back to HDFS
    - File Formats
    - Standard ETL
  + **Data Analysis**
    - Use Hive Tables
    - Fundamentals of Querying
    - Analytical or windowing Functions
  + **Configuration**

**Tutorial 22**

* **Getting Started - Setup Options**

**Tutorial 23**

* **Getting Started - Setup Cloudera Quickstart VM**

**Tutorial 24**

* **Spark - Getting started using ITversity labs**
  + login to Itversity big data lab console using the authorized username and password
  + Launch Spark shell on the cluster using: **spark-shell --master yarn --conf spark.ui.port=12245**  (port number can be anywhere b/w 10000-65535)
  + using command **sc** check spark is installed properly or not
  + using command **ls -ltr /data/** check the available data sets
  + using command **du -sh /data/file-name** check the size of the file
  + using command **hadoop fs -ls /directory path** to find the files in HDFS,
  + to know size **hadoop fs -du -s -h /name of directory or file**

**Tutorial 25**

* **Spark -Getting Started - Windows- Setup putty and WinScp**

**Tutorial 26**

* **Spark - Getting Started -Windows- Setup Cygwin**

**Tutorial 27**

* **Spark - Getting Started -HDFS quick preview**
  + The property files that controls HDFS environment are stored in **cd /etc/hadoop/conf**
  + Most important files that are need to be concentrated are **hdfs-site.xml, core-site.xml**
  + The cluster properties are stored in **core-site.xml**
  + **fs.dafaultFS** has the url address of the name node
  + the block size,replication properties are stored in **hdfs.site.xml**
  + the user space information can be viewed using the formula **hadoop fs -ls /user/username**
  + As horton work does not provide root user space so in order to create that the commands are used :

**sudo -u hdfs hadoop fs -mkdir /user/root; sudo -u hdfs hadoop fs -chown -R root /user/root**

* copying locally available files to HDFS **-  hadoop fs -copyFromLocal path of the file destination address**
* To know the blocksize and locations of the files use **hdfs fsck /user/username/path of the file -files -blocks -locations**

**Tutorial 28**

* **Spark - Getting Started - YARN Quick Preview**
  + Yarn file can be found in **cd /etc/hadoop/conf/yarn-site.xml**
  + In the Yarn file we can find resource manager webapp addres (url of resource manager) to access the web UI
  + Spark env file can be found in **cd /etc/spark/conf/spark-env.sh**
  + Each task takes at least one core.

**Tutorial 29**

* **Spark - Getting Started - Setup Data Sets**
  + **wc -l path of the file** shows the records of the following file

**Tutorial  30**

* **Apache Sqoop - Introduction and objectives**
  + **Importing data**
  + **Importing data - Hive**
  + **Exporting data**
  + **Troubleshooting**
* **External systems and my cluster**
  + **Import data from MySQL database into HDFS using Sqoop**
  + **Export data to MySQL database from HDFS using Sqoop**
  + **Change delimiter and file format of data during import using sqoop**
  + **Ingest real time and near real time streaming data into HDFS**
  + **Process streaming data as it loaded into cluster**
  + **Load data into and out of HDFS using Hadoop file system commands**

**Tutorial 31**

* **Apache Sqoop - Accessing Documentation**
  + <https://sqoop.apache.org/docs/1.4.6/SqoopDevGuide.html>

**Tutorial 32**

* **Apache Sqoop -Validating Mysql and Environment**
* **Preview of mysql**
  + **List of database:**
    - **retail\_db**
    - **hr\_db**
    - **nyse\_db**
  + **List of users:**
    - **retail\_user**
    - **hr\_user**
    - **nyse\_user**
  + **password:**
    - **itversity**
  + validating sqoop by **sqoop version** command
  + validating Mysql by **mysql -u username -h hostname -p password(#use the above user name and password to access the databases)**
  + **Eg: mysql - u retail\_user -h ms.itversity.com -p password**
  + To list the databases use command **show databases;**
  + To go to the db use command **use dbname;**
  + **Eg: use retail\_db;**
  + To view the following tables in the db use command **show tables;**
  + To view the data from particular table use command **select \* from table name with required limit no**
  + **Eg: select \* from categories limit 10;**

**Tutorial 33**

* **Apache Sqoop - listing tables and databases**
  + Connect String -(Make sure to know the location of sqoop lib) Example: cd /usr/hdp/current/sqoop-client/lib
  + To list mysql databases in sqoop:
* Using sqoop help for searching commands:
  + sqoop help list-databases (This will give us the information about list-databases)

**sqoop-list-databases \**

**--connect jdbc:**[**mysql://**](mysql://ms.itversity.com:3306/)**hostname \**

**--username  \**

**--password**

**Eg: sqoop list-databases \**

**--connect jdbc:mysql://ms.itversity.com:3306 \**

**--username retail\_user \**

**--password itversity**

**sqoop**

* To list tables from the Mysql db in sqoop
* **sqoop-list-tables \**

**--connect jdbc:**[**mysql://**](mysql://ms.itversity.com:3306/retail_db)**hostname/db name \**

**--username \**

**--password**

**Eg: sqoop list-tables** \

               --connect jdbc:mysql://ms.itversity.com:3306/retail\_db \

               --username retail\_user \

               --password itversity

**Tutorial 34**

* **Apache Sqoop - Querying from Databases using eval**
  + Usage of eval command: to retrieve data from selected table

**sqoop eval \**

**--connect jdbc:**[**mysql://**](mysql://ms.itversity.com:3306/retail_db)**hostname \ db name \**

**--username  \**

**--password  \**

**--query "SELECT \* FROM tablename"**

* To insert data to a particular table using eval

**sqoop eval \**

**--connect jdbc:**[**mysql://**](mysql://ms.itversity.com:3306/retail_db)**Hostname \ db name \**

**--username  \**

**--password  \**

**--query "INSERT INTO tablename values**

* Create Table using eval

**sqoop eval \**

**--connect jdbc:**[**mysql://**](mysql://ms.itversity.com:3306/retail_export)**host name \ db name \**

**--username  \**

**--password  \**

**--query "CREATE TABLE table name "**

**Tutorial 35**

* **Apache Sqoop - Sqoop Import - Run simple import**
  + list db followed up in the previous tutorial
  + Make sure that u can able to access the data with in the db
  + Import data

**sqoop import \**

**--connect jdbc:mysql://host name \ db name \**

**--username  \**

**--password  \**

**--table (source table name) \**

**--warehouse-dir / destination address**

* Validating the imported data

**hadoop fs -ls /destination address**

* to view the data after importing to the destination

**hadoop fs -tail /destination address**

**Tutorial 36**

* **Apache Sqoop - Sqoop Import - Execution Life Cycle**
  + to control no of mappers

**sqoop import \**

**--connect jdbc:mysql://host name \ db name \**

**--username  \**

**--password  \**

**--table (source table name) \**

**--warehouse-dir / destination address**

**-- num-mappers-1 (declare no of mappers that are required)**

**Tutorial 37**

* **Apache Sqoop - Sqoop Import – Managing Directories**
  + target  dir copies data directly into the particular dir
  + warehouse dir creates the sub dir and then copies the data into it
  + deleting data using delete and append command

**sqoop import \**

**--connect jdbc:mysql://host name \ db name \**

**--username  \**

**--password  \**

**--table (source table name) \**

**--warehouse-dir / destination address**

**-- num-mappers-1 (declare no of mappers that are required)**

**-- append**

**(or)**

**sqoop import \**

**--connect jdbc:mysql://host name \ db name \**

**--username  \**

**--password  \**

**--table (source table name) \**

**--warehouse-dir / destination address**

**-- num-mappers-1 (declare no of mappers that are required)**

**--delete-target-dir**

# Tutorial 38

* **Apache Sqoop - Sqoop Import - using split by**
  + Things to remember while working with split-by includes
    - Column should be indexed
    - Values in the field should be sparse
    - Also often it should be sequence generated or evenly incremented
    - it should not have null values
  + Using split-by to import data

**sqoop import \**

**--connect jdbc:mysql://host name \ db name \**

**--username  \**

**--password  \**

**--table (source table name) \**

**--warehouse-dir / destination address**

**--split-by column names**

**Tutorial 39**

* **Apache Sqoop - Sqoop Import - auto reset to one mapper**