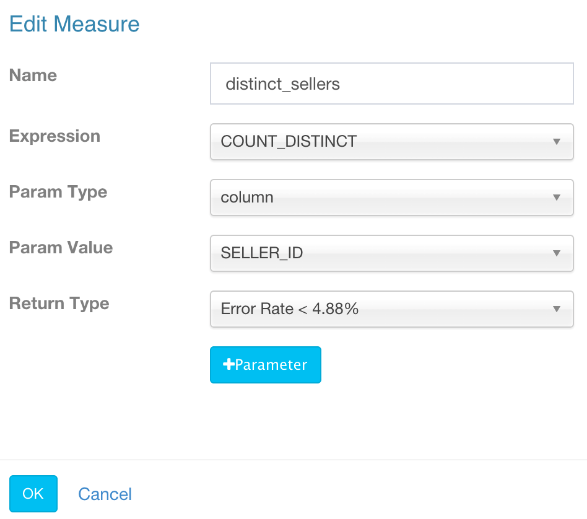
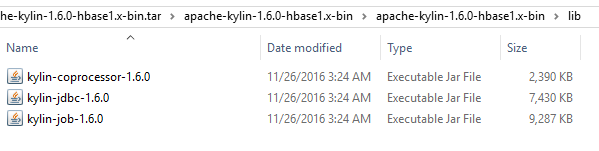
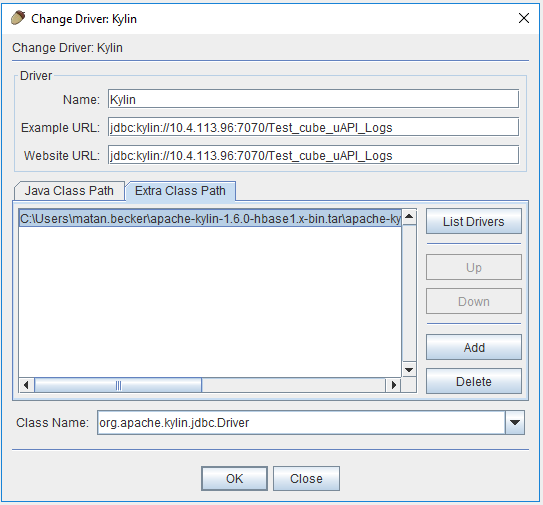
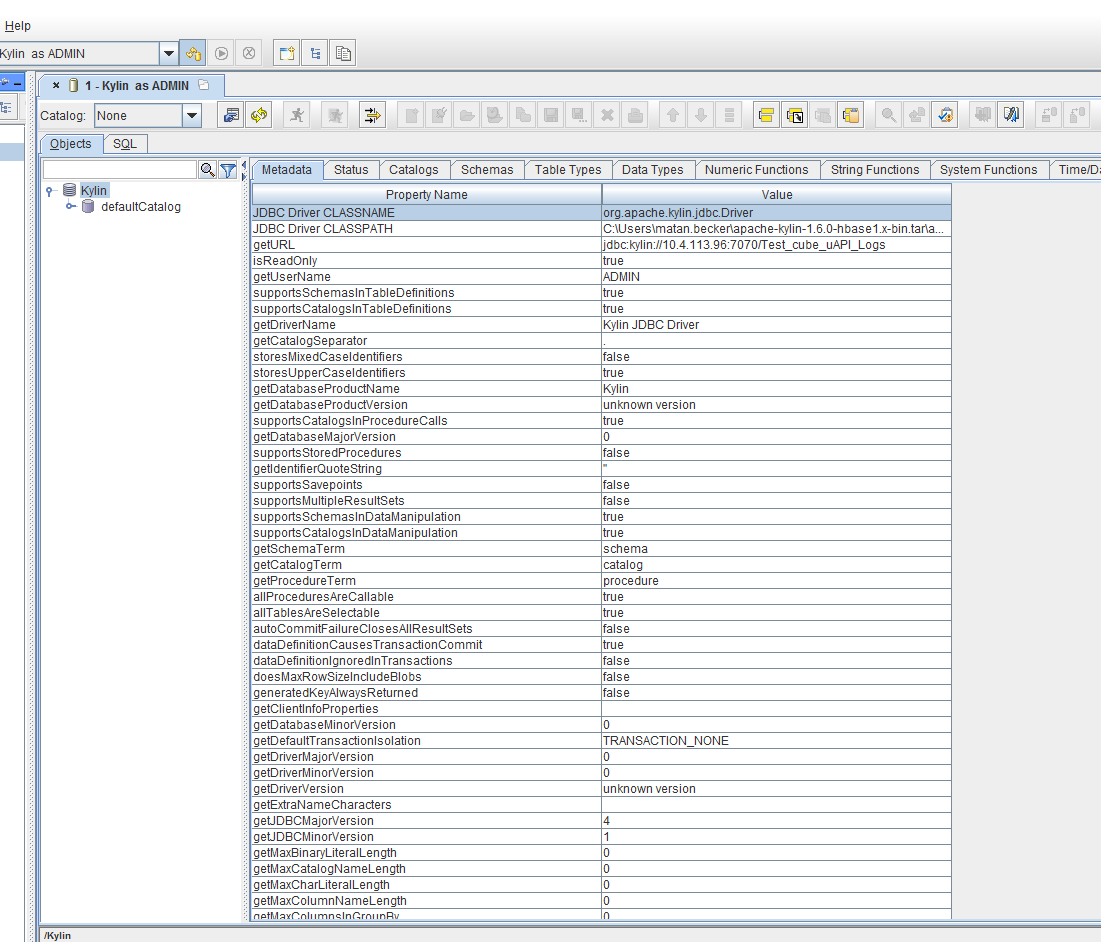
**Cube Design Instructions:**

1. Navigate to Kylin in the lab:
   * <http://10.4.113.96:7070/kylin/>
   * Login: ADMIN
   * Password: KYLIN
2. If you have already built a cube and want to use the same data model, skip to number 7
3. From Home page, click on “Manage Projects” 
4. Click on “+Project”  to start new cube set up
   * Project Name: Enter a name for the project
   * Project Description: Not necessary
   * Click “Submit”
   * If successful, the new project will show in the table
5. Sync a table from Hive:
   * Click “Model” in the top menu bar and click “Data Source”
   * Click “Load Hive Table from Tree” 
     + Select database🡪select table
     + Click “Sync”
     + A success message should appear
       - Can now click on the database/table and see fields
6. Create Data Model: before creating a cube, a data model needs to created that defines the star schema:
   * Click “Model” in the top menu bar-->select the “Models” tab (above +New dropdown)🡪click “+New” drop down🡪select “New Model”
   * Model Info:
     + Model Name: Enter a name for the model
     + Description: Optional
     + Click “Next”
   * Data Model:
     + Fact Table: select Fact Table from dropdown
     + [Optional] -Click Add Lookup Table button to add a lookup table. Select the table name and join type (inner or left).
     + Click “Next”
   * Dimensions:
     + Select dimension columns to use in the cube
     + Click “Next”
   * Measures:
     + Select measure columns to use in the cube
     + Click “Next
   * Settings:
     + From Kylin website: If data in the fact table increases by day, select the corresponding date column in the Partition Date Column, and select the date format, otherwise leave it as blank.
     + [Optional] Select Cube Size, which is an indicator of the scale of the cube - by default is set to MEDIUM
     + [Optional] If some records want to excluded from the cube, you can input the condition in Filter
     + Click “save”🡪”Yes” to save the data model
     + Model should now be shown in the list
7. Create the Cube:
   * Click “Model” in the top menu bar-->click “Models”🡪click “+New” drop down🡪select “New Cube”
   * Cube Info:
     + Model Name: Select model from drop-down
     + Cube Name: You can use letters, numbers, and ‘\_’ to name your cube (blank spaces not allowed)
     + Notification List: you can list email addresses which be notified on cube job success/failure
     + Description: optional
     + Click “Next”
   * Dimensions:
     + Click “Add Dimensions”
     + Select all or select the dimensions you want to use
     + Click “Ok”
     + Click “Next”
   * Measures:
     + Click “+Measure” to add a new measure
     + There are multiple types of measures according to its expression: SUM, MAX, MIN, COUNT, COUNT\_DISTINCT, TOP\_N, RAW, EXTENDED\_COLUMN, PERCENTILE.
       - Properly select the return type for COUNT\_DISTINCT and TOP\_N, as it will impact on the cube size
       - i.e. 
       - Please note: distinct count is a very heavy data type, it is slower to build and query comparing to other measures.
   * Refresh Setting:
     + This step is designed for incremental cube build
     + Auto Merge Time Ranges (days): merge the small segments into medium and large segments automatically
       - If you don’t want to auto merge, remove the default two ranges
     + Retention Range (days): only keep the segment whose data is in past given days in cube
       - The old segment will be automatically dropped from head
       - 0 means not enable this feature
     + Partition Start Date: the start date of this cube
   * Advanced Settings:
     + Aggregation Groups: by default, Kylin puts all dimensions into one aggregation group
       - You can create multiple aggregation groups by knowing your query patterns well.
       - For the concepts of “Mandatory Dimensions”, “Hierarchy Dimensions” and “Joint Dimensions”, read this blog: [New Aggregation Group](http://kylin.apache.org/blog/2016/02/18/new-aggregation-group/)
     + Rowkeys: the rowkeys are composed by the dimension encoded values. “Dictionary” is the default encoding method;
       - If a dimension is not fit with dictionary (e.g., cardinality > 10 million), select “false” and then enter the fixed length for that dimension
         * Usually that is the max. length of that column
         * If a value is longer than that size it will be truncated.
         * Please note, without dictionary encoding, the cube size might be much bigger.
     + You can drag & drop a dimension column to adjust its position in rowkey;
       - Put the mandatory dimension at the beginning, then followed the dimensions that heavily involved in filters (where condition).
       - Put high cardinality dimensions ahead of low cardinality dimensions.
     + Click “Next”
   * Configuration Overwrites:
     + Click “Next”
   * Overview:
     + If everything looks good, click “Save” to finish setting up the cube
8. Start Cube Build:
   * Click “Model” in the top menu bar-->click “Models”
   * In the row with your cube go to “Actions” column and click the “Action” drop-down
   * Click “Build”
   * Are you sure to start the build: click “Yes”
   * Go to the “Monitor” tab in the top menu bar to monitor the progress of the cube build
9. You now have a built your cube and can connect to SQL clients/BI applications through the JDBC or ODBC drivers
10. You can also query your data directly on the Kylin UI: Select the “Insight” tab and write your query in the SQL editor box

**Connecting Cube to SquirrelSQL:**

1. Download and install SquirrelSQL: <http://www.squirrelsql.org/#installation>
2. If you don’t already have the JDBC .jar file, navigate here and download the correct JDBC .jar file depending on version of HBase (version 1 in this case): <http://kylin.apache.org/download/>
   * 
   * Under the version number, under “Binary Download”, click the file name depending on the HBase version
   * This will take you to another page with many URL’s, click the first one and wait for the file to finish downloading
   * Unpack/Unzip the file after it is done downloading. You can use the program 7-zip (just google and download it).
     1. Once 7-zip is installed, you can right click zipped files and there will be an option for 7-zip to extract all files
3. Add Kylin JDBC Driver:
   * Open SquirrelSQL and navigate to the “Driver” tab on the left side menu
   * Click the blue plus sign to add a new driver 
     1. Name: Kylin
     2. Example URL: jdbc:kylin://10.4.113.96:7070/[INSERT CUBE NAME HERE]
     3. Website URL: jdbc:kylin://10.4.113.96:7070/[INSERT CUBE NAME HERE]
     4. Class Name: org.apache.kylin.jdbc.Driver
     5. Click “Extra Class Path Tab”🡪”Add”
     6. Locate the JDBC .jar file and click “Ok”
        1. 
     7. Click “Ok”
4. Add Alias:
   * On the left side menu, click “Aliases” and then click the blue plus sign 
     1. Name: Kylin
     2. Driver: Select driver that you just made
     3. URL: jdbc:kylin://10.4.113.96:7070/[INSERT CUBE NAME HERE]
     4. User Name: ADMIN
     5. Password: KYLIN
     6. Click “Test Connection”
     7. Pop-up message should appear that says “Connection Successful”
        1. If connection is unsuccessful, it’s probably because you are using the wrong JDBC file
     8. Click “Ok”
5. Connect to SQL environment:
   * Click the “Aliases” tab on the left side menu and double click “Kylin” or whatever alias you just made
   * Connection should start and new environment populate. See screenshot below:



1. Queries ran in SquirrelSQL:
   * **\*Note**: both queries exclude APP\_SERV\_HOST\_NM and HOST\_LOCATOR\_CD – cube build with all dimensions took over 1 day
   * Hourly Aggregates Query:
     1. “SELECT LOG\_DATE, REQUEST\_TYPE\_ID, SUCCESS\_IND, AGENCY\_ID, PCC\_CD, BRANCH\_CD, SOAP\_FAULT\_TEXT, SCHEMA\_VERSION\_NM, PROVIDER\_CD\_LIST, ERROR\_TEXT, SUPPLIER\_CD\_LIST, ERROR\_TYPE\_CD, HOTEL\_CODE, sum(BOOKED\_SEG\_QTY) as BOOKED\_SEG\_QTY, sum(TOTAL\_MILLISECOND\_QTY) as TOTAL\_MILLISECOND\_QTY, sum(GAL\_RT\_MILLISEC\_QTY) as GAL\_RT\_MILLISEC\_QTY, sum(APO\_RT\_MILLISEC\_QTY) as APO\_RT\_MILLISEC\_QTY, sum(WSP\_RT\_MILLISEC\_QTY) as WSP\_RT\_MILLISEC\_QTY, sum(JAL\_RT\_MILLISEC\_QTY) as JAL\_RT\_MILLISEC\_QTY, sum(ACH\_RT\_MILLISEC\_QTY) as ACH\_RT\_MILLISEC\_QTY, sum(RCH\_RT\_MILLISEC\_QTY) as RCH\_RT\_MILLISEC\_QTY, sum(MCH\_RT\_MILLISEC\_QTY) as MCH\_RT\_MILLISEC\_QTY, sum(TRM\_RT\_MILLISEC\_QTY) as TRM\_RT\_MILLISEC\_QTY, sum(SVCS\_RT\_MILLISEC\_QTY) as SVCS\_RT\_MILLISEC\_QTY, sum(RESPONSE\_BYTES\_QTY) as RESPONSE\_BYTES\_QTY, sum(OPTIONAL\_SVC\_QTY) as OPTIONAL\_SVC\_QTY, count(\*) as DETAILED\_TRANSACTION\_COUNT FROM RAW\_ULOGS Where LOG\_DATE >= '2016-08-01' and LOG\_DATE <= '2016-08-10' Group by LOG\_DATE, REQUEST\_TYPE\_ID, SUCCESS\_IND, AGENCY\_ID, PCC\_CD, BRANCH\_CD, SOAP\_FAULT\_TEXT, SCHEMA\_VERSION\_NM, PROVIDER\_CD\_LIST, ERROR\_TEXT, SUPPLIER\_CD\_LIST, ERROR\_TYPE\_CD, HOTEL\_CODE limit 3000;”
   * Daily Aggregates Query:
     1. “SELECT LOG\_DATE, REQUEST\_TYPE\_ID, SUCCESS\_IND, AGENCY\_ID, PCC\_CD, BRANCH\_CD, SOAP\_FAULT\_TEXT, SCHEMA\_VERSION\_NM, PROVIDER\_CD\_LIST, ERROR\_TEXT, SUPPLIER\_CD\_LIST, ERROR\_TYPE\_CD, HOTEL\_CODE, sum(BOOKED\_SEG\_QTY) as BOOKED\_SEG\_QTY, sum(TOTAL\_MILLISECOND\_QTY) as TOTAL\_MILLISECOND\_QTY, sum(GAL\_RT\_MILLISEC\_QTY) as GAL\_RT\_MILLISEC\_QTY, sum(APO\_RT\_MILLISEC\_QTY) as APO\_RT\_MILLISEC\_QTY, sum(WSP\_RT\_MILLISEC\_QTY) as WSP\_RT\_MILLISEC\_QTY, sum(JAL\_RT\_MILLISEC\_QTY) as JAL\_RT\_MILLISEC\_QTY, sum(ACH\_RT\_MILLISEC\_QTY) as ACH\_RT\_MILLISEC\_QTY, sum(RCH\_RT\_MILLISEC\_QTY) as RCH\_RT\_MILLISEC\_QTY, sum(MCH\_RT\_MILLISEC\_QTY) as MCH\_RT\_MILLISEC\_QTY, sum(TRM\_RT\_MILLISEC\_QTY) as TRM\_RT\_MILLISEC\_QTY, sum(SVCS\_RT\_MILLISEC\_QTY) as SVCS\_RT\_MILLISEC\_QTY, sum(RESPONSE\_BYTES\_QTY) as RESPONSE\_BYTES\_QTY, sum(OPTIONAL\_SVC\_QTY) as OPTIONAL\_SVC\_QTY, count(\*) as DETAILED\_TRANSACTION\_COUNT FROM raw\_ulogs Where LOG\_DATE >= '2016-08-01' and LOG\_DATE < '2016-08-10' Group by LOG\_DATE, REQUEST\_TYPE\_ID, SUCCESS\_IND, AGENCY\_ID, PCC\_CD, BRANCH\_CD, SOAP\_FAULT\_TEXT, SCHEMA\_VERSION\_NM, PROVIDER\_CD\_LIST, ERROR\_TEXT, SUPPLIER\_CD\_LIST, ERROR\_TYPE\_CD, HOTEL\_CODE limit 3000;”

**Query data in Hadoop using Spark -DataFrames/Read-ORC-File:**

SparkSQL Documentation: <https://spark.apache.org/docs/latest/sql-programming-guide.html>

Pyspark API Documentation: <http://spark.apache.org/docs/2.1.0/api/python/pyspark.sql.html>

* For this process, we will be using Pyspark, the Spark Python API, however you can also write scripts in Scala (native Spark language) and R

Create Pyspark script to run in Spark shell:

1. Use “vim” command to create a .py file 🡪 i.e. vim test\_spark\_file.py
   1. Once your new file has initialized use “I” (insert) to start writing your script
   2. Useful commands once inside script:
      1. To get out of insert mode🡪 :
      2. Quit without saving 🡪 :q!
      3. Quit with saving 🡪 :wq!
2. The entry point into all functionality in Spark is the “SparkSession” class. To create a basic SparkSession, use SparkSession.builder:
3. from pyspark.sql import SparkSession

spark = SparkSession \

.builder \

.appName("[ENTER APPLICATION NAME]") \ #names your application

.enableHiveSupport() \ #allows access to Hive database and tables

.getOrCreate #Uses existing SparkContext/creates new one

1. Create DataFrame from hive table:
   1. df = spark.read.orc("hdfs://DEVCLUSTER/apps/hive/warehouse/uapi.db/raw\_ulogs/log\_date=\*")
2. Now that DataFrame is created – you can code the query aggregations, groupBys, orderBys using Python language
3. In order to run the script in Spark in the cluster – you must use “spark-submit” shell
   1. Spark Submit Documentation - <https://spark.apache.org/docs/latest/submitting-applications.html>
      1. The link above goes into depth about Spark-Submit’s parameters and customization options
4. When in linux – type this in the command line:
   1. “spark-submit --master yarn --conf spark.hadoop.yarn.timeline-service.enabled=false --num-executors 12 --driver-memory 50G --executor-memory 40G --executor-cores 10 [NAME OF FILE]”
      1. You can change the configuration settings in the script as needed (i.e. num-executors, driver-memory, executor-memory, cores)

**Query data in Hadoop using Spark.sql():**

1. Navigate to the 10.4.113.106 server (\*servers 10.4.113.106 and 10.4.113.95 have different versions of Spark)
2. Create a .py file:
   1. Follow step 1 from the process above (“Query data in Hadoop using Spark -DataFrames/Read-ORC-File”)
3. Use the following to set up your script:
   1. from pyspark.sql import \*

from pyspark import SparkConf, SparkContext, SQLContext

from pyspark.sql import HiveContext

conf = (SparkConf()

.setAppName("[NAME YOUR APP]")

.set("spark.serializer", "org.apache.spark.serializer.KryoSerializer")

.set("spark.sql.inMemoryColumnarStorage.compressed", "true"))

sc = SparkContext(conf = conf)

sqlContext = HiveContext(sc)

sqlContext.sql("use [ENTER DATABASE NAME]")

df = sqlContext.table("uapi.raw\_ulogs")

df.registerTempTable("raw\_ulogs\_temp")

sqlContext.sql("[TYPE YOUR SQL QUERY HERE]").show()

* 1. You can write your query using SQL syntax in the brackets of the sqlContext.sql() function

1. Run script/ job using “Spark-Submit” shell – repeat step 5 in above process

**Connecting Kylin Cube to Qlikview:**

1. First you must download and install the Kylin ODBC driver and add it to the Windows System DSN as a data source:
   1. Download and install the Kylin ODBC driver here: [Kylin ODBC Driver v1.6](http://kylin.apache.org/download/KylinODBCDriver-1.6.zip)
   2. Once the driver is installed navigate to “ODBC Data Sources” in Windows – you can search this in the taskbar
   3. Click the “System DSN” tab
   4. Click “Add”
   5. Click “KylinODBCDriver” 🡪 click “Finish”
2. Download and install Qlikview: <http://www.qlik.com/us/try-or-buy>
   1. Once installed, open Qlikview
   2. Click the “Edit Script” icon in the icon bar: 
   3. In the “Data” tab, click the “Database” dropdown and select “ODBC”
   4. Click “connect” and highlight the Kylin data source
      1. May need to select the “Force 32 bit” box if the Kylin data source is not showing
   5. Type in username and password:
      1. User: ADMIN
      2. Password: KYLIN
   6. Click “Test Connection” and make sure the connection is successful
   7. Click ok
3. Query cube data:
   1. While still in the “Edit Script” window, click “Select” (below Connect)
   2. Select dimensions and measures that you want to use
   3. Click “Ok”
   4. Click the “Reload Script” icon in the icon bar: 
4. You can now create a sheet or chart using the data:
   1. Create Sheet:
      1. Go to “Layout”🡪”Add Sheet”
      2. Go to “Layout”🡪”Select Fields..”
         1. Select the fields you want in the sheet
   2. Create Chart:
      1. Click “Quick Chart Wizard”: 
      2. Follow the prompts to customize your chart based on the queried data