Feedback for credit Card Management System case study for Data engineering

We have reviewed the Credit Card Management System case study for Data engineering. We suggested that the below topics will be covered in the Credit Card Management System. Below, we have proposed steps to implement these topics and the overall process to execute.

Proposed Steps:

1: Create modules/ Business logic on java using eclipse. User can only use console for input and output because Servlets are not under the scope of the course

2: Use RDBMS Mysql for Database. Initially, Data will be stored in Mysql. We will use Sqoop for pullout (import) the data from RDBMS into HADOOP or HIVE. Then will use Oozie for the schedule the Jobs.

Topics to be cover in Case Study:

- **SDLC / Agile** : Students will use SDLC and Agile methodology. They will be able to identify each of the phases of the SDLC (Requirements Analysis, Design, Development, Testing, Implementation, and Maintenance). Will use for information gathering and determine functional and non-functional requirements. Students will be able to prioritize the requirements to meet customer needs.

- **Core Java OOPs concept:** Student will apply/use OOPs concept here. They will develop business logic like Presentation layer, Business logic layer, and data access Layer for the Customer Details module, Bank Details module, and the Apply/Cancel for Credit Card module. We will also use JDBC, ODBC drivers for the connection of RDBMS.

- **Database(Mysql, Oracle):** Students will apply /use RDBMS concept with Structured Query Language (SQL). They will create Database for the credit card management system and they will perform multiple actions in a database such as create, delete, select, insert, update, modify, drop, and commit. They will then connect to the Database with java code/business logic which is defined above using any particular driver

- **Data modeling:** Students will use Data modeling concepts for optimization and Normalization in order to eliminate redundant information from the tables and organize the data.

- **Hadoop / HDFS:** Students will apply/use the Hadoop framework. They will set up Hadoop using Ambari. They will work on Architecture [Namenode(NN), DataNode(DN)]. Accessing, analyzing, and manipulating data in Hadoop.

- **Ambari**: Students will use Hortonworks sandbox for the credit card management case study. Ambari makes Hadoop management simpler by providing a consistent, secure platform for operational control.

- **Hive and Partitioning**: Students will use/apply Hive and partitioning in hive for the better Performance. Also will create external and internal tables. They will use hive to design batch queries on Hadoop by providing a declarative abstraction layer. They will perform multiple actions in a database such as create, delete, select, insert, update, modify, drop.

- **Sqoop**: Student will use Sqoop to import data from a relational source to HDFS and vice versa.

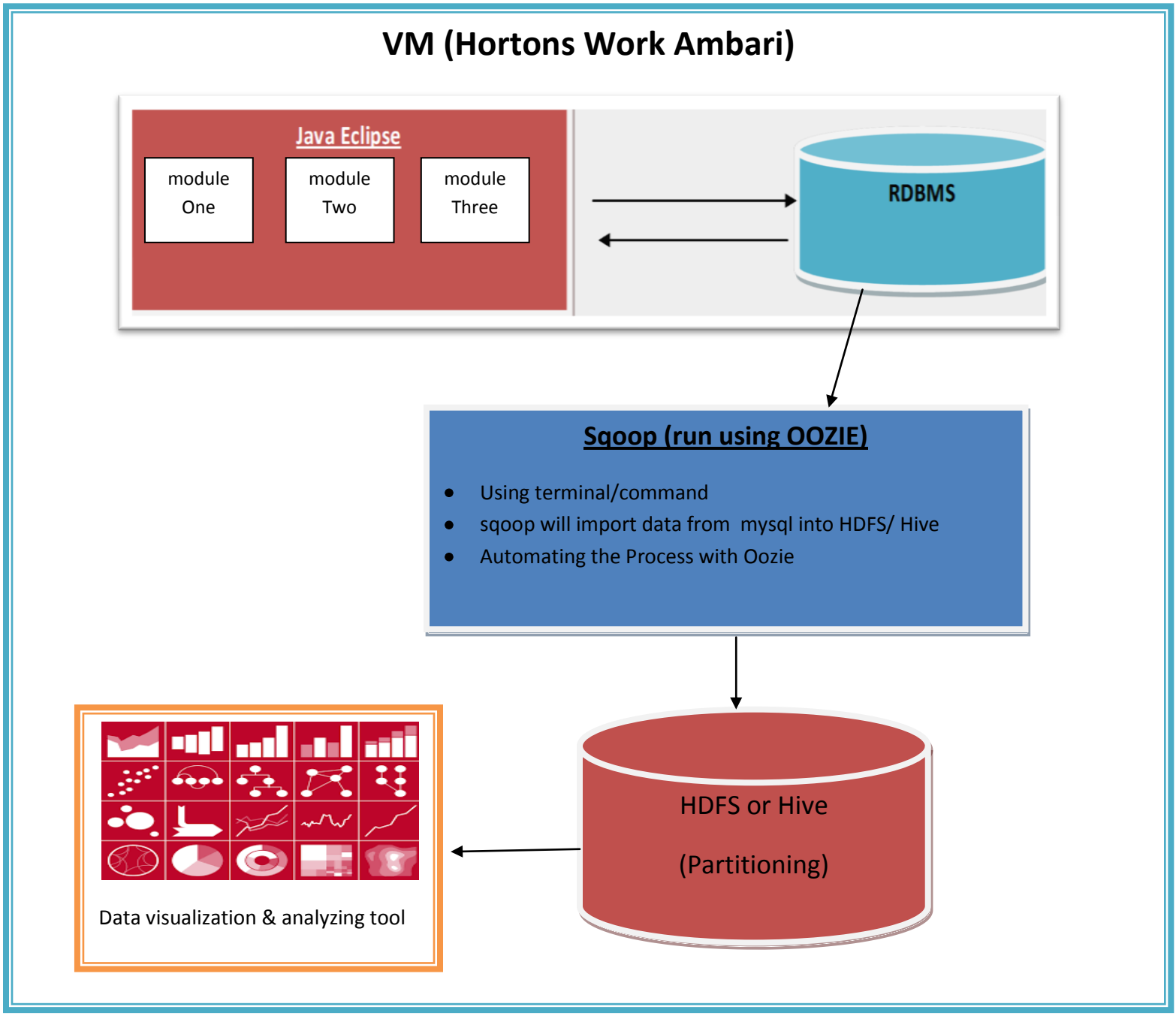
- **Oozie:** Student will use/ apply Oozie. They will create a Scheduler system to run and manage Hadoop jobs. Oozie is tightly integrated with the Hadoop stack supporting various Hadoop jobs like Hive, Pig, Sqoop, as well as system specific solutions like Java

- **Linux/ Shell:** Students will use/apply Linux and shell commands. Because Hadoop/ Ambari is running on Linux (specifically, RHEL/CentOS). Student need use the Shell to perform any task including the setup of the tools they will be using for this case study. Students will make use of bash scripting to automate the extraction, transportation, and loading of data.

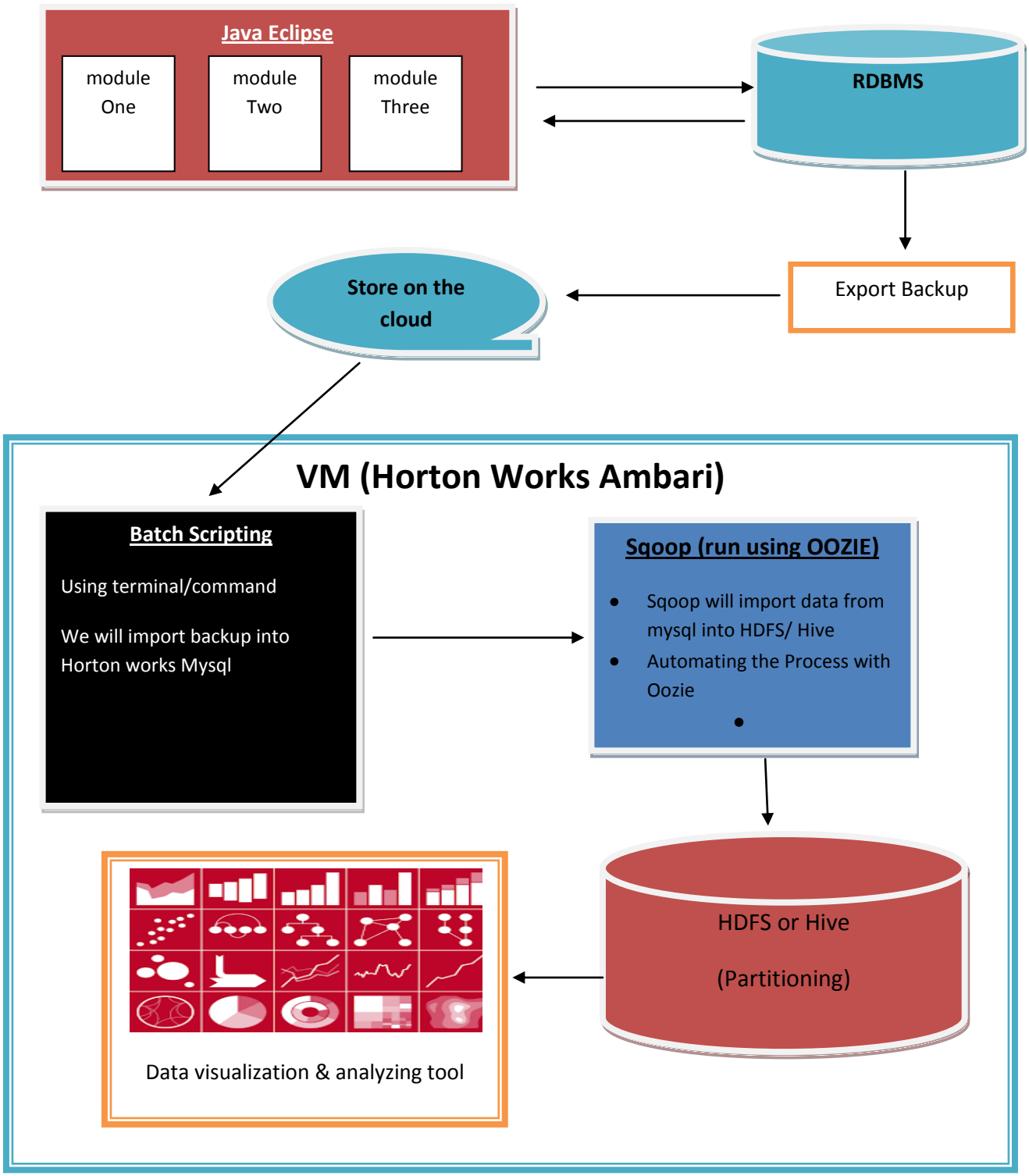
- **Data Visualization:** Students will use/apply Visualization tool for data exploration and visualization. This would help to complete the whole analysis. This approach presents the opportunity to act immediately, based on what's happening at a moment in time.

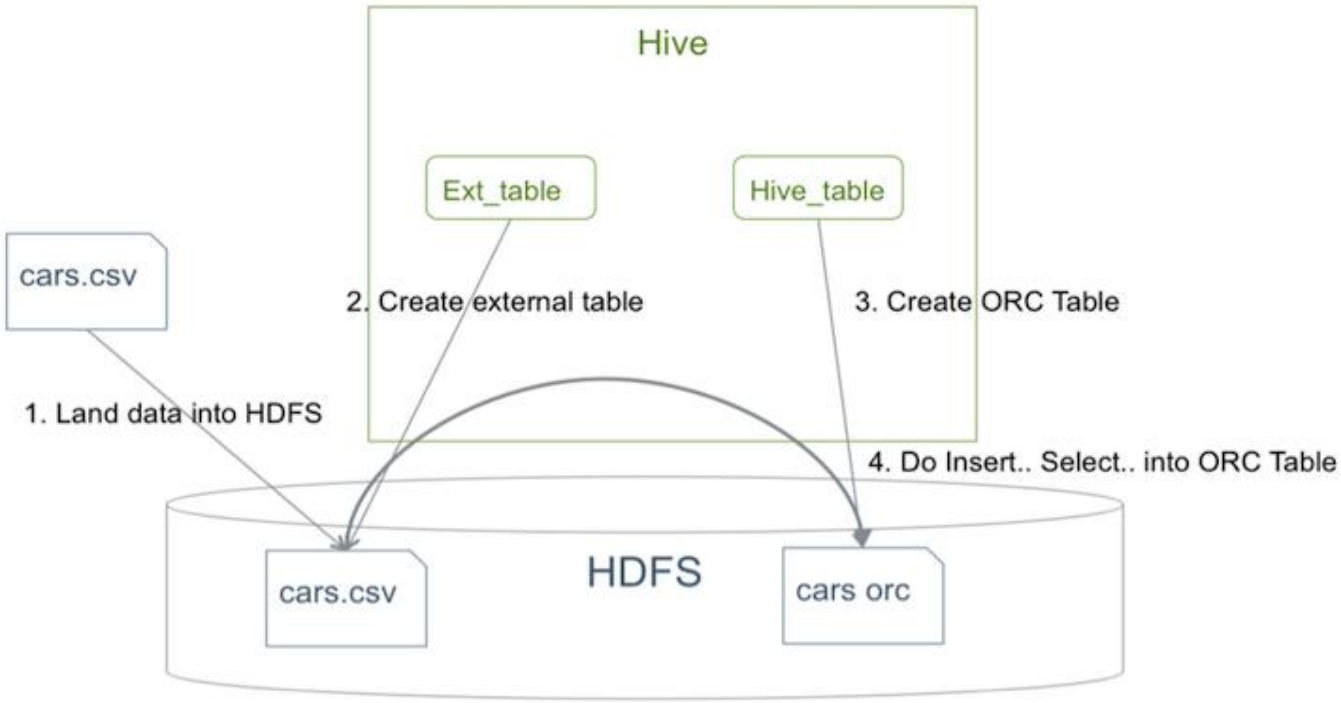
- **Pig (optional):** Student can perform all the data manipulation operations in Hadoop using Pig.

Proposed Workflow



Alternate Work Flow




Hive
Ext_table
Hive_table
cars.csv
2. Create external table
3. Create ORC Table
1. Land data into HDFS
4. Do Insert.. Select.. into ORC Table
HDFS
cars.csv
cars orc