**Solution**

**1.    Data Model:**

a.       Please find below is the data model required to support the requirement. This solution requires   two tables, Recipe and Ingredient.

b.       Recipe table is created with below mentioned columns

Recipe\_Id

Recipe\_name

Recipe\_description

Updated\_date

Created\_date

c.       Ingredient table is created with Ingredient\_name, Recipe\_id and Active columns.

d.       The relation between Recipe table and Ingredient table is many to many relation.

e.       These tables are created in Impala to query by Frontend users. Impala is used for faster performance.

f.        Recipe table helps the frontend developer to show list of all the recipes available.

When user clicks the recipe to see the ingredients, Ingredient table and Recipe table joined on Recipe Id and Active field from Ingredient table as true to list all active ingredients from Ingredient table for that recipe.

When user clicks Ingredient name, all distinct recipe ids associated with ingredient name lists the recipes from Recipe table by joining Ingredient table and Recipe Table on Recipe Id and active field from Ingredient table as true.

|  |
| --- |
| Recipe Table |
| Recipe\_Id  Recipe\_name  Recipe\_description  Updated\_date  Created\_date |

|  |
| --- |
| Ingredient Table |
| Ingredient\_name  Recipe\_id   Active |

**2.    Spark Job:**

This receives data from HDFS to Spark and process the data using below job. My assumption here is hdfs data is available at hdfs://nameservice1/user/cloudera/.

**package** com.techevaluation

**import** org.apache.spark.SparkContext

**import** org.apache.spark.SparkContext.\_

**import** org.apache.spark.SparkConf

**import** org.apache.spark.sql.SQLContext

**import** org.apache.spark.sql.\_

**import** org.apache.spark.sql.*DataFrame*

**object** RecipeSolution {

**def** main(args: Array[*String*]) {

**val** conf = **new** SparkConf().setAppName("Data Load").setMaster("local")

**val** sc= **new** SparkContext(conf)

**val** sqlCtx = **new** SQLContext(sc)

**import** sqlCtx.implicits.\_

/\* I am using partioning of 3 to fit my data into memory. I am assuming that 3 partitions are enough to fit my data into memory \*/

**val** recipeRDD = sc.textFile("hdfs://nameservice1/user/cloudera/recipedata.csv",3)

**val** datasplit = recipeRDD.map(data => data.split(",").map(\_.trim))

**case** **class** **recipe**(recipe\_id: Int, recipe\_name: *String*, recipe\_desc: *String*, Ingrediant\_name: *String*, active: Boolean, updated\_date: *String*, create\_date: *String*)

**val** recipers  = datasplit.map(x => recipe(x(0).toInt, x(1),x(2),x(3), x(4).toBoolean, x(5),x(6)))

**val** recipeDF = recipers.toDF

recipeDF.registerTempTable("recipes")

**val** recipetbl = sqlCtx.sql("""select recipe\_id, recipe\_name, recipe\_desc, updated\_date, create\_date from recipes""")

recipetbl.rdd.saveAsTextFile("hdfs://nameservice1/user/cloudera/recipe")

**val** ingredienttbl = sqlCtx.sql("""select Ingrediant\_name, recipe\_id, active from recipes""")

ingredienttbl.rdd.saveAsTextFile("hdfs://nameservice1/user/cloudera/ingredient")

 }

}

**Impala Tables:**

Impala tables can be created using above data.

**Recipe Table creation in Impala:**

create external table recipe(recipe\_id Int, recipe\_name *String*, recipe\_desc *String*, updated\_date *String*, create\_date *String)*

LOCATION ‘hdfs://nameservice1/user/cloudera/recipe’

Stored as textfile;

**Ingredient Table creation in Impala:**

create external table ingredient(Ingrediant\_name *String*, recipe\_id Int, active Boolean*)*

LOCATION ‘hdfs://nameservice1/user/cloudera/ingredient’

Stored as textfile;

**Queries:**

1. **Average Number of recipes updated for an hour.**

select recipe\_name,

       DATE(CAST(updated\_date AS timestamp)) AS date,

       HOUR(CAST(updated\_date AS timestamp)) AS hour,

       count(\*)

from recipe

group by recipe\_name, DATE(CAST(updated\_date AS timestamp)), HOUR(CAST(updated\_date AS timestamp));

**B. Number of recipes which got updated at 10:00 clock in the entire year.**

I am assuming that it would be for particular year. I am executing below query is for year 2018.

select recipe\_name,

      updated\_date,

       count(\*)

from recipe

group by recipe\_name, updated\_date

having YEAR(CAST(updated\_date AS timestamp)) = '2018'

and HOUR(CAST(updated\_date AS timestamp)) = '10'

and MINUTE(CAST(updated\_date AS timestamp)) = '00';