


Praktikum Teknologi Perekayasa Data

Tanggal Praktikum : Monday, June 5, 2023

1. Spark Shell

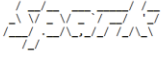
```
at org.apache.hadoop.ipc.Server$Handler.run(Server.java:2682)

at org.apache.hadoop.ipc.Client.getResponse(Client.java:1497)
at org.apache.hadoop.ipc.Client.call(Client.java:1443)
at org.apache.hadoop.ipc.Client.call(Client.java:1353)
at org.apache.hadoop.ipc.ProtobufRpcEngine$Invoker.invoke(ProtobufRpcEngine.java:228)
at org.apache.hadoop.ipc.ProtobufRpcEngine$Invoker.invoke(ProtobufRpcEngine.java:116)
at com.sun.proxy.$Proxy13.mkdirs(Unknown Source)
at org.apache.hadoop.hdfs.protocolPB.ClientNamenodeProtocolTranslatorPB.mkdirs(ClientNamenodeProtocolTranslatorPB.java:653)
at sun.reflect.NativeMethodAccessorImpl.invoke(Native Method)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:62)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.apache.hadoop.io.retry.RetryInvocationHandler.invokeMethod(RetryInvocationHandler.java:422)
at org.apache.hadoop.io.retry.RetryInvocationHandler$Call.invokeMethod(RetryInvocationHandler.java:165)
at org.apache.hadoop.io.retry.RetryInvocationHandler$Call.invoke(RetryInvocationHandler.java:157)
at org.apache.hadoop.io.retry.RetryInvocationHandler$Call.invokeOnce(RetryInvocationHandler.java:95)
at org.apache.hadoop.io.retry.RetryInvocationHandler.invoke(RetryInvocationHandler.java:359)
at com.sun.proxy.$Proxy14.mkdirs(Unknown Source)
at org.apache.hadoop.hdfs.DFSClient.primitiveMkdir(DFSClient.java:2409)
... 75 more
<console>:14: error: not found: value spark
import spark.implicits._
^
<console>:14: error: not found: value spark
import spark.sql
^

Welcome to
 version 2.3.1.3.0.1.0-187

Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.8.0_191)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

```
scala> [root@sandbox-hdp ~]# sudo -u hdfs hdfs dfs -mkdir /user/root
mkdir: /user/root: File exists
[root@sandbox-hdp ~]# sudo -u hdfs hdfs -chown root:root /user/root
sudo: hdfs: command not found
[root@sandbox-hdp ~]# sudo -u hdfs -chown root:root /user/root
usage: sudo -h | -K | -k | -V
usage: sudo -v [-AknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -i [-AknS] [-g group] [-h host] [-p prompt] [-u user] [command]
usage: sudo [-ABEknPS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p prompt] [-u user] [VAR=value] [-i|-s] [command]
usage: sudo -e [-AknS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p prompt] [-u user] file ...
[root@sandbox-hdp ~]# sudo -u hdfs hdfs dfs -chown root:root /user/root
sudo: hdfs: command not found
[root@sandbox-hdp ~]# spark-shell
SPARK_MAJOR_VERSION is set to 2, using Spark2
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://sandbox-hdp.hortonworks.com:4040
Spark context available as 'sc' (master = yarn, app id = application_1685889842736_0002).
Spark session available as 'spark'.
Welcome to
 version 2.3.1.3.0.1.0-187

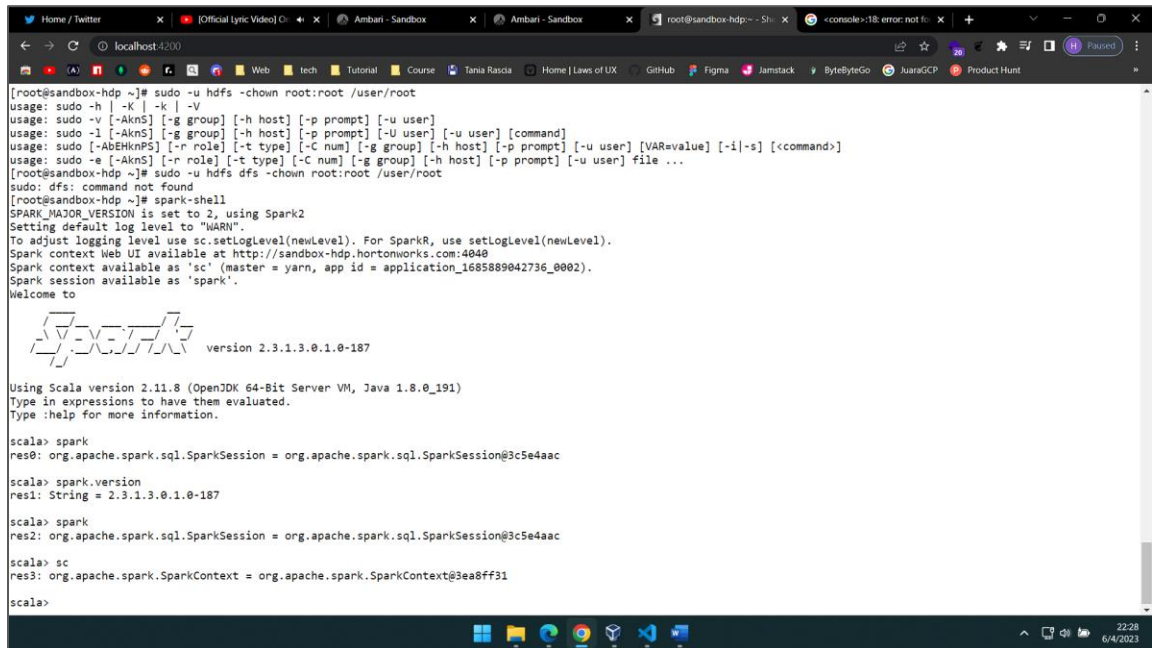
Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.8.0_191)
Type in expressions to have them evaluated.
Type :help for more information.

scala> spark
res0: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@3c5e4aac


scala> spark.version
res1: String = 2.3.1.3.0.1.0-187

scala>
```

2. Spark (SparkSession) and SparkContext



```
[root@sandbox-hdp ~]# sudo -u hdfs -chown root:root /user/root
usage: sudo -h | -K | -k | -V
usage: sudo -v [-AknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -l [-AknS] [-g group] [-h host] [-p prompt] [-U user] [-u user] [command]
usage: sudo [-AbEHknPS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p prompt] [-u user] [VAR=value] [-i|-s] [<command>]
usage: sudo -e [-AknS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p prompt] [-u user] file ...
[root@sandbox-hdp ~]# sudo -u hdfs dfs -chown root:root /user/root
sudo: dfs: command not found
[root@sandbox-hdp ~]# spark-shell
SPARK_MAJOR_VERSION is set to 2, using Spark2
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://sandbox-hdp.hortonworks.com:4040
Spark context available as 'sc' (master = yarn, app id = application_1685889042736_0002).
Spark session available as 'spark'.
Welcome to

 version 2.3.1.3.0.1.0-187

Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.8.0_191)
Type in expressions to have them evaluated.
Type :help for more information.

scala> spark
res0: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@3c5e4aac

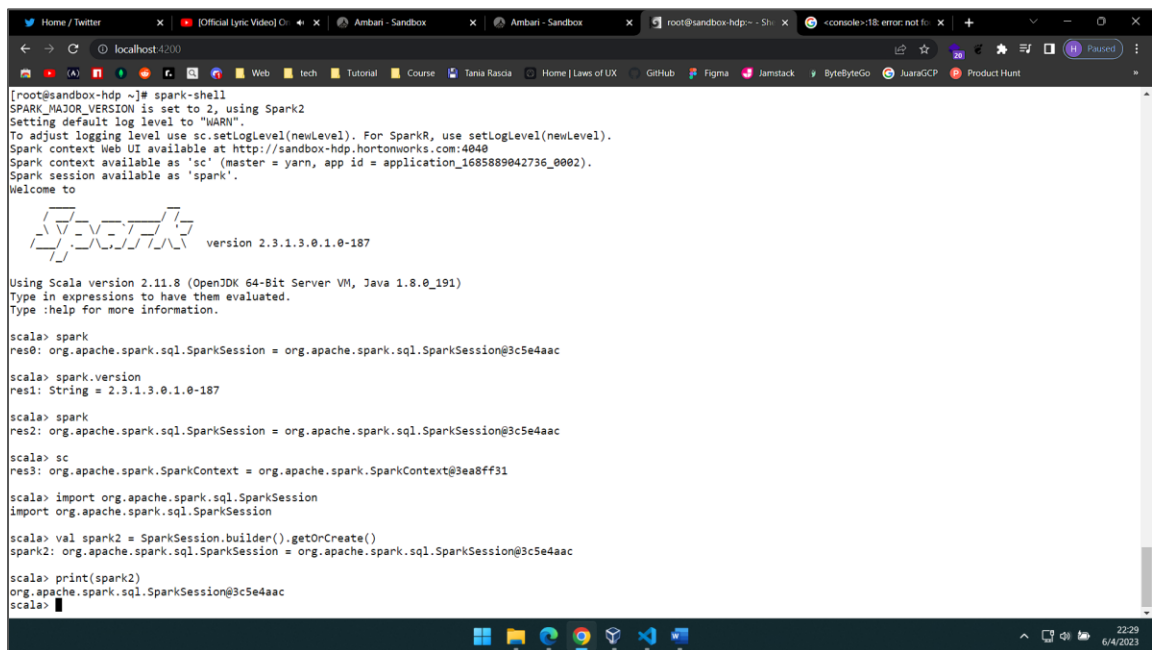
scala> spark.version
res1: String = 2.3.1.3.0.1.0-187

scala> spark
res2: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@3c5e4aac

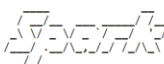
scala> sc
res3: org.apache.spark.SparkContext = org.apache.spark.SparkContext@3ea8ff31

scala>
```

3. Cek current spark session



```
[root@sandbox-hdp ~]# spark-shell
SPARK_MAJOR_VERSION is set to 2, using Spark2
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://sandbox-hdp.hortonworks.com:4040
Spark context available as 'sc' (master = yarn, app id = application_1685889042736_0002).
Spark session available as 'spark'.
Welcome to

 version 2.3.1.3.0.1.0-187

Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.8.0_191)
Type in expressions to have them evaluated.
Type :help for more information.

scala> spark
res0: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@3c5e4aac

scala> spark.version
res1: String = 2.3.1.3.0.1.0-187

scala> spark
res2: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@3c5e4aac

scala> sc
res3: org.apache.spark.SparkContext = org.apache.spark.SparkContext@3ea8ff31

scala> import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.SparkSession

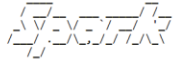
scala> val spark2 = SparkSession.builder().getOrCreate()
spark2: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@3c5e4aac

scala> print(spark2)
org.apache.spark.sql.SparkSession@3c5e4aac

scala>
```

4. Create RDD

```
Untitled Report x 352 [P] TEKNOLOGI PEREKAWA x 352 [T] TEKNOLOGI PEREKAWA x root@sandbox-hdp:~ - Shell In x tpd12.docx - Google Docs x +
localhost:4200
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://sandbox-hdp.hortonworks.com:4040
Spark context available as 'sc' (master = yarn, app id = application_1685937894203_0001).
Spark session available as 'spark'.
Welcome to

 version 2.3.1.3.0.1.0-187

Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.8.0_191)
Type in expressions to have them evaluated.
Type :help for more information.

scala> import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.SparkSession

scala> val dataSeq = Seq(("Java", 20000), ("Python", 100000), ("Scala", 3000))
dataSeq: Seq[(String, Int)] = List((Java,20000), (Python,100000), (Scala,3000))

scala> val rdd = spark.sparkContext.parallelize(dataSeq)
rdd: org.apache.spark.rdd.RDD[(String, Int)] = ParallelCollectionRDD[0] at parallelize at <console>:26

scala> val data = rdd.collect()
data: Array[(String, Int)] = Array((Java,20000), (Python,100000), (Scala,3000))

scala> data.foreach(println)
(Java,20000)
(Python,100000)
(Scala,3000)

scala> data.foreach(f => println(f._1)+" "+println(f._2))
Java
20000
Python
100000
Scala
3000

scala>
```

5. RDD Operation (Transformation)

```
Untitled Report x 352 [P] TEKNOLOGI PEREKAWA x 352 [T] TEKNOLOGI PEREKAWA x root@sandbox-hdp:~ - Shell In x tpd12.docx - Google Docs x +
localhost:4200

scala> data.foreach(f => println(f._1)+" "+println(f._2))
Java
20000
Python
100000
Scala
3000

scala> val filter = dataSeq.filter(a => a._1.startsWith("J"))
<console>:1: error: illegal start of simple pattern
val filter = dataSeq.filter(a => a._1.startsWith("J"))
^

scala> val filter = dataSeq.filter(a => a._1.startsWith("J"))
filter: Seq[(String, Int)] = List((Java,20000))

scala> val filter = dataSeq.filter(a => a._1.startsWith("P"))
filter: Seq[(String, Int)] = List((Python,100000))

scala> val filter = dataSeq.filter(a => a._1.contains("P"))
filter: Seq[(String, Int)] = List((Python,100000))

scala> val filter = dataSeq.filter(a => a._1.contains("a"))
filter: Seq[(String, Int)] = List((Java,20000), (Scala,3000))

scala> val filter = dataSeq.filter(a => a._1.matches("a"))
filter: Seq[(String, Int)] = List()

scala> val filter = dataSeq.filter(a => a._1.matches(".*a.*"))
filter: Seq[(String, Int)] = List((Java,20000), (Scala,3000))

scala> val filter = dataSeq.filter(a => a._1.length > 5)
filter: Seq[(String, Int)] = List((Python,100000))

scala> val filter = dataSeq.filter(a => a._2 > 3000)
filter: Seq[(String, Int)] = List((Java,20000), (Python,100000))

scala> val sort = dataSeq.sortBy(a => a._2)
sort: Seq[(String, Int)] = List((Scala,3000), (Java,20000), (Python,100000))

scala>
```

6. Create Dataframe

```
Gogo on Twitter: 'Se...' x JKT48 - Seventi... x Ambari - Sandbox x Ambari - Sandbox x root@sandbox-hdp: x root@sandbox-hdp: x HDFS User Read anu x +
localhost:4200
scala> val data = rdd.collect()
<console>:26: error: not found: value rdd
    val data = rdd.collect()
                  ^

scala> val data = rdd.collect()
<console>:26: error: not found: value rdd
    val data = rdd.collect()
                  ^

scala> val columns = Seq("programming_language", "total_user")
columns: Seq[String] = List(programming_language, total_user)

scala> val df = spark.createDataFrame(dataSeq).toDF(columns:_)
<console>:1: error: illegal start of simple pattern
val df = spark.createDataFrame(dataSeq).toDF(columns:_)
              ^

scala> val df = spark.createDataFrame(dataSeq).toDF(columns:_)
df: org.apache.spark.sql.DataFrame = [programming_language: string, total_user: int]

scala> df.show()
+-----+-----+
|programming_language|total_user|
+-----+-----+
|                    |          |
|          Python    |    100000|
|          Scala     |     30000|
+-----+-----+

scala> df.printSchema()
root
 |-- programming_language: string (nullable = true)
 |-- total_user: integer (nullable = false)

scala>
```

7. Dataframe Operation

```
Gogo on Twitter: 'Se...' x JKT48 - Aitakatti... x Ambari - Sandbox x Ambari - Sandbox x root@sandbox-hdp: x root@sandbox-hdp: x HDFS User Read and x +
localhost:4200
-- total_user: integer (nullable = false)

scala> df.select("programming_language").show()
+-----+
|programming_language|
+-----+
|          Python    |
|          Scala     |
+-----+

scala> val df2 = df.withColumn("total_user", df("total_user") + 100)
df2: org.apache.spark.sql.DataFrame = [programming_language: string, total_user: int]

scala>

scala> df2.show()
+-----+-----+
|programming_language|total_user|
+-----+-----+
|          Python    |    100100|
|          Scala     |     30100|
+-----+-----+

scala> df.groupBy("programming_language").count().show()
+-----+-----+
|programming_language|count|
+-----+-----+
|          Scala     |     1|
|          Python    |     1|
|          Java      |     1|
+-----+-----+

scala>
```

8. Save to table

```
Gogo on Twitter: X Junus Rahasi: X Ambari - Sandb: X Ambari - Sandb: X root@sandbox: X root@sandbox: X HDFS User Read X localhost / 127.0.0.1 X +
localhost:4200
scala> val df2 = df.withColumn("total_user", df("total_user") + 100)
df2: org.apache.spark.sql.DataFrame = [programming_language: string, total_user: int]

scala>

scala> df2.show()
+-----+
|programming_language|total_user|
+-----+
|Java|20100|
|Python|100100|
|Scala|3100|
+-----+

scala> df.groupBy("programming_language").count.show()
+-----+
|programming_language|count|
+-----+
|Scala|1|
|Python|1|
|Java|1|
+-----+

scala> val data_hdfs = spark.read.csv("/user/root/emp.csv")
data_hdfs: org.apache.spark.sql.DataFrame = [_c0: string, _c1: string ... 3 more fields]

scala> data_hdfs.show()
+-----+
|_c0|_c1|_c2|_c3|_c4|
+-----+
|Id|Name|Salary|Department|Designation|
|1|John|5000|IT|Developer|
|2|Chris|6000|Sales|Manager|
|3|Jamie|7000|Support|Director|
|4|Henning|400|IT|Developer|
+-----+
```

```
Home / Twitter X Only Today, Rapsodi, Karens: X Ambari - Sandbox X root@sandbox hdp: - Shell In / X adhiy zara jk148 - Penelusuran: X +
localhost:4200
at scala.collection.immutable.List.flatMap(List.scala:344)
at org.apache.spark.sql.execution.datasources.DataSource.resolveRelation(DataSource.scala:388)
at org.apache.spark.sql.DataFrameReader.loadV1Source(DataFrameReader.scala:239)
at org.apache.spark.sql.DataFrameReader.load(DataFrameReader.scala:227)
at org.apache.spark.sql.DataFrameReader.csv(DataFrameReader.scala:596)
at org.apache.spark.sql.DataFrameReader.csv(DataFrameReader.scala:473)
... 49 elided

scala> val data_hdfs = spark.read.csv("/user/root/emp.csv")
data_hdfs: org.apache.spark.sql.DataFrame = [_c0: string, _c1: string ... 3 more fields]

scala> data_hdfs.createOrReplaceTempView("sample_table")

scala> val data_hdfs2 = spark.sql("select * from sample_table")
data_hdfs2: org.apache.spark.sql.DataFrame = [_c0: string, _c1: string ... 3 more fields]

scala> data_hdfs2.show()
+-----+
|_c0|_c1|_c2|_c3|_c4|
+-----+
|Id|Name|Salary|Department|Designation|
|1|John|5000|IT|Developer|
|2|Chris|6000|Sales|Manager|
|3|Jamie|7000|Support|Director|
|4|Henning|400|IT|Developer|
+-----+

scala> val salary_more_than_5000 = spark.sql("select * from sample_table where salary > 5000")
org.apache.spark.sql.AnalysisException: cannot resolve 'salary' given input columns: [sample_table._c0, sample_table._c2, sample_table._c4, sample_table._c1, sample_t
able._c3]; line 1 pos 33;
'Project [1]
+- 'Filter ('salary > 5000)
+- SubqueryAlias sample_table
+- Relation[_c0#10,_c1#11,_c2#12,_c3#13,_c4#14] csv

at org.apache.spark.sql.catalyst.analysis.package$AnalysisErrorAt.failAnalysis(package.scala:42)
at org.apache.spark.sql.catalyst.analysis.CheckAnalysis$$anonfun$checkAnalysis$1$$anonfun$apply$2.applyOrElse(CheckAnalysis.scala:92)
at org.apache.spark.sql.catalyst.analysis.CheckAnalysis$$anonfun$checkAnalysis$1$$anonfun$apply$2.applyOrElse(CheckAnalysis.scala:89)
at org.apache.spark.sql.catalyst.triggers.TriggerNode$$anonfun$transform$in$1.apply(TriggerNode.scala:789)
```

```
scala> val salary_more_than_5000 = spark.sql("select * from sample_table where sample_table._c2 > 5000")
salary_more_than_5000: org.apache.spark.sql.DataFrame = [_c0: string, _c1: string ... 3 more fields]

scala> salary_more_than_5000.show()
+----+----+----+----+----+
|_c0|_c1|_c2|_c3|_c4|
+----+----+----+----+----+
| 2|Chris|6000|Sales|Manager|
| 3|Jamie|7000|Support|Director|
+----+----+----+----+----+

scala> spark.table("sample_table").write.saveAsTable("sample_hive_table")
23/06/04 16:16:05 WARN SessionState: METASTORE_FILTER_HOOK will be ignored, since hive.security.authorization.manager is set to instance of HiveAuthorizerFactory.

scala> read_salary2 = spark.sql("SELECT * FROM sample_hive_table")
<console>:25: error: not found: value read_salary2
val $ires6 = read_salary2
               ^

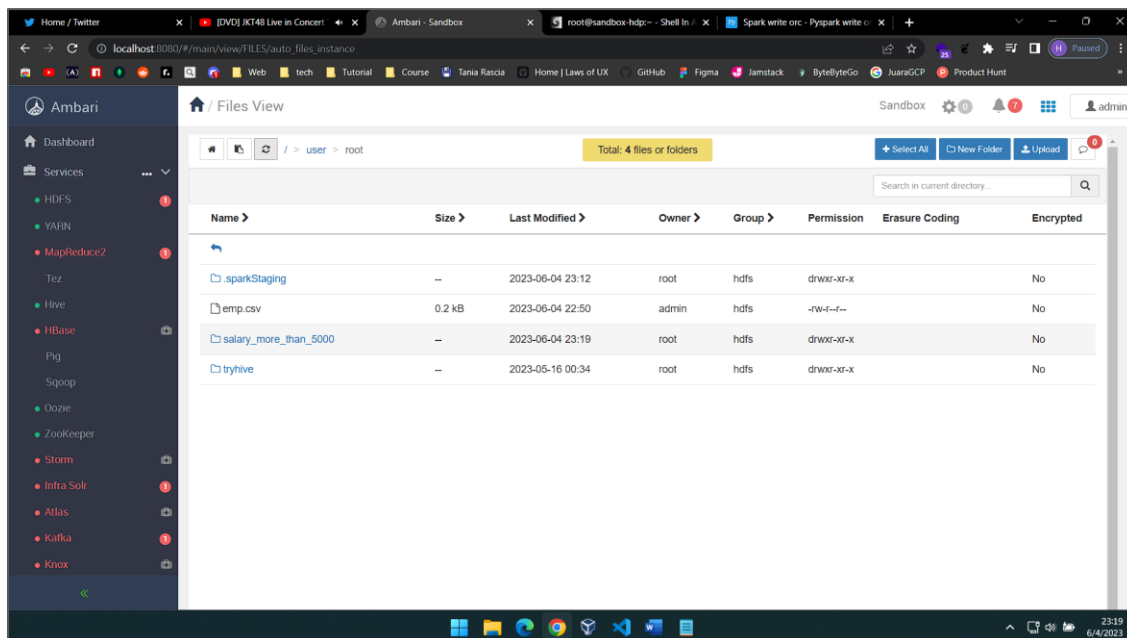
<console>:23: error: not found: value read_salary2
    read_salary2 = spark.sql("SELECT * FROM sample_hive_table")
    ^

scala> val read_salary2 = spark.sql("SELECT * FROM sample_hive_table")
read_salary2: org.apache.spark.sql.DataFrame = [_c0: string, _c1: string ... 3 more fields]

scala> read_salary2.show()
+----+----+----+----+----+
|_c0|_c1|_c2|_c3|_c4|
+----+----+----+----+----+
|Id|Name|Salary|Department|Designation|
| 1|John|5000|IT|Developer|
| 2|Chris|6000|Sales|Manager|
| 3|Jamie|7000|Support|Director|
| 4|Henning|400|IT|Developer|
+----+----+----+----+----+

scala>
```

9. Simpan ke format data lain (orc)



```
Home / Twitter x [DND] KT48 Live in Concert x Ambani - Sandbox x root@sandbox-hdp:~ - Shell In x Spark write orc - PySpark write o x +
localhost:4200
1 | John | 5000 | IT | Developer |
2 | Chris | 6000 | Sales | Manager |
3 | Jamie | 7000 | Support | Director |
4 | Henning | 400 | IT | Developer |

scala> salary_more_than_5000.write.orc.save("/user/root//salary_more_than_5000/")
<console>:26: error: missing argument list for method orc in class DataFrameWriter
Unapplied methods are only converted to functions when a function type is expected.
You can make this conversion explicit by writing 'orc _' or 'orc(_)' instead of 'orc'.
    salary_more_than_5000.write.orc.save("/user/root//salary_more_than_5000/")
                                ^

scala> salary_more_than_5000.write.orc_.save("/user/root//salary_more_than_5000/")
<console>:26: error: value orc_ is not a member of org.apache.spark.sql.DataFrameWriter[org.apache.spark.sql.Row]
    salary_more_than_5000.write.orc_.save("/user/root//salary_more_than_5000/")
                                ^

scala> salary_more_than_5000.write().format("orc").save("/user/root//salary_more_than_5000/")
<console>:26: error: org.apache.spark.sql.DataFrameWriter[org.apache.spark.sql.Row] does not take parameters
    salary_more_than_5000.write().format("orc").save("/user/root//salary_more_than_5000/")
                                ^

scala> salary_more_than_5000.write.format("orc").save("/user/root//salary_more_than_5000/")

scala> val read_salary_orc = spark.read.orc("/user/root/salary_more_than_5000/")
read_salary_orc: org.apache.spark.sql.DataFrame = [_c0: string, _c1: string ... 3 more fields]

scala> read_salary_orc.show()
+-----+
|_c0|_c1|_c2|_c3|_c4|
+-----+
| 2|Chris|6000|Sales|Manager|
| 3|Jamie|7000|Support|Director|
+-----+

scala>
```


Penugasan

1. Lakukan input data dalam bentuk RDD.

```
scala> Seq[String, Int] = List((Java,20000), (Scala,3000))

scala> val filter = dataSeq.filter(a => a._1.matches("a"))
filter: Seq[(String, Int)] = List()

scala> val filter = dataSeq.filter(a => a._1.matches(".*a.*"))
filter: Seq[(String, Int)] = List((Java,20000), (Scala,3000))

scala> val filter = dataSeq.filter(a => a._1.length > 5)
filter: Seq[(String, Int)] = List((Python,100000))

scala> val filter = dataSeq.filter(a => a._2 > 3000)
filter: Seq[(String, Int)] = List((Java,20000), (Python,100000))

scala> val sort = dataSeq.sortBy(a => a._2)
sort: Seq[(String, Int)] = List((Scala,3000), (Java,20000), (Python,100000))

scala> val tpd12_rdd = Seq(("James", "", "Smith", "1991-04-01", "M", 3000), ("Michael", "Rose", "", "2000-05-19", "M", 4000), ("Robert", "", "Williams", "1978-09-05", "M", 4000), ("Maria", "Anne", "Jones", "1967-12-01", "F", 4000), ("Jen", "Mary", "Brown", "1980-02-17", "F", 10000))
tpd12_rdd: Seq[(String, String, String, String, Int)] = List((James,"",Smith,1991-04-01,M,3000), (Michael,Rose,"",2000-05-19,M,4000), (Robert,"",Williams,1978-09-05,M,4000), (Maria,Anne,Jones,1967-12-01,F,4000), (Jen,Mary,Brown,1980-02-17,F,10000))

scala> tpd12_rdd.foreach(println)
(James,,Smith,1991-04-01,M,3000)
(Michael,Rose,,2000-05-19,M,4000)
(Robert,,Williams,1978-09-05,M,4000)
(Maria,Anne,Jones,1967-12-01,F,4000)
(Jen,Mary,Brown,1980-02-17,F,10000)

scala> val tpd12_col = Seq("firstname", "middlename", "lastname", "dob", "gender", "salary")
tpd12_col: Seq[String] = List(firstname, middlename, lastname, dob, gender, salary)

scala>

scala> val tpd12_df = spark.createDataFrame(tpd12_rdd).toDF(tpd12_col:_)
tpd12_df: org.apache.spark.sql.DataFrame = [firstname: string, middlename: string ... 4 more fields]

scala>

scala> tpd12_df.show()
+-----+-----+-----+-----+-----+-----+
|firstname|middlename|lastname|dob|gender|salary|
+-----+-----+-----+-----+-----+-----+
|James| |Smith|1991-04-01|M|3000|
|Michael|Rose| |2000-05-19|M|4000|
|Robert| |Williams|1978-09-05|M|4000|
|Maria|Anne|Jones|1967-12-01|F|4000|
|Jen|Mary|Brown|1980-02-17|F|10000|
+-----+-----+-----+-----+-----+-----+

```

2. Mapping dengan skema yang telah disiapkan dan simpan dalam bentuk dataframe.

```
(James,,Smith,1991-04-01,M,3000)
(Michael,Rose,,2000-05-19,M,4000)
(Robert,,Williams,1978-09-05,M,4000)
(Maria,Anne,Jones,1967-12-01,F,4000)
(Jen,Mary,Brown,1980-02-17,F,10000)

scala> val tpd12_col = Seq("firstname", "middlename", "lastname", "dob", "gender", "salary")
tpd12_col: Seq[String] = List(firstname, middlename, lastname, dob, gender, salary)

scala>

scala> val tpd12_df = spark.createDataFrame(tpd12_rdd).toDF(tpd12_col:_)
tpd12_df: org.apache.spark.sql.DataFrame = [firstname: string, middlename: string ... 4 more fields]

scala>

scala> tpd12_df.show()
+-----+-----+-----+-----+-----+-----+
|firstname|middlename|lastname|dob|gender|salary|
+-----+-----+-----+-----+-----+-----+
|James| |Smith|1991-04-01|M|3000|
|Michael|Rose| |2000-05-19|M|4000|
|Robert| |Williams|1978-09-05|M|4000|
|Maria|Anne|Jones|1967-12-01|F|4000|
|Jen|Mary|Brown|1980-02-17|F|10000|
+-----+-----+-----+-----+-----+-----+

scala>

scala> tpd12_df.printSchema()
root
 |-- firstname: string (nullable = true)
 |-- middlename: string (nullable = true)
 |-- lastname: string (nullable = true)
 |-- dob: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- salary: integer (nullable = false)

scala>

```


3. Lakukan operasi terhadap dataframe yaitu:

a. Filter record dengan tahun lahir 2000

```
scala> tpd12_df.printSchema()
root
 |-- firstname: string (nullable = true)
 |-- middlename: string (nullable = true)
 |-- lastname: string (nullable = true)
 |-- dob: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- salary: integer (nullable = false)

scala> tpd12_df.filter(substring($"dob",0,4) > 2000).show()
+-----+-----+-----+-----+-----+
|firstname|middlename|lastname|dob|gender|salary|
+-----+-----+-----+-----+-----+
|James|Smith|1991-04-01|M|3000| |
|Robert|Williams|1978-09-05|M|4000|
|Maria|Anne|Jones|1967-12-01|F|4000|
|Jen|Mary|Brown|1980-02-17|F|10000|

scala> tpd12_df.filter(substring($"dob",0,4) < 2000).show()
+-----+-----+-----+-----+-----+
|firstname|middlename|lastname|dob|gender|salary|
+-----+-----+-----+-----+-----+
|James|Smith|1991-04-01|M|3000| |
|Robert|Williams|1978-09-05|M|4000|
|Maria|Anne|Jones|1967-12-01|F|4000|
|Jen|Mary|Brown|1980-02-17|F|10000|

scala>
```

b. Group by Gender

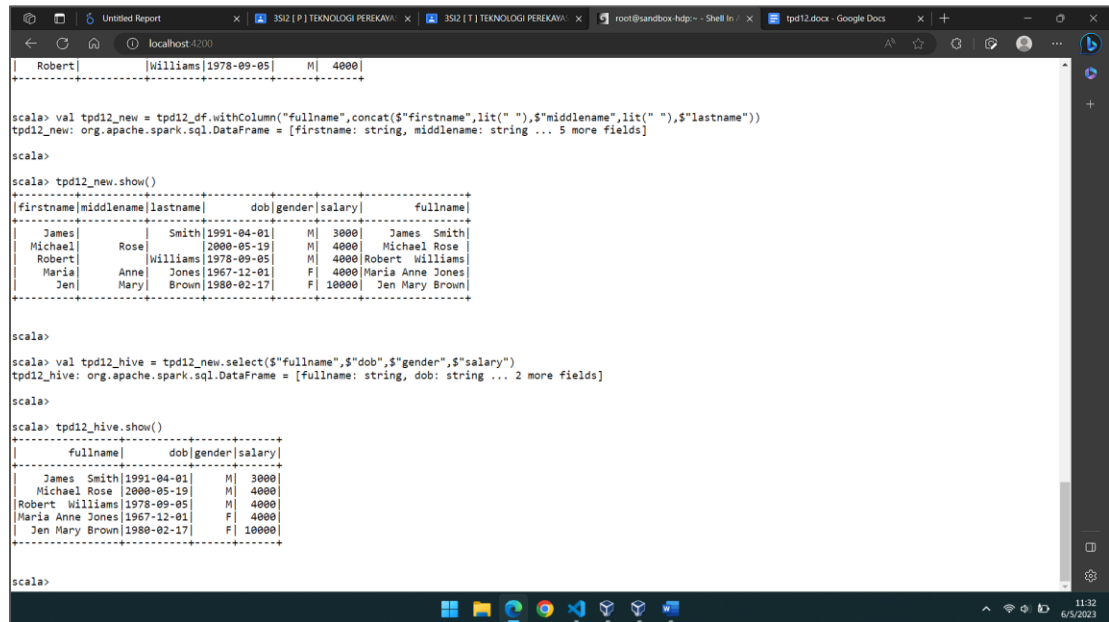
```
scala> tpd12_df.groupBy("gender").count().show()
+-----+-----+
|gender|count|
+-----+-----+
|F|2|
|M|3|
+-----+-----+

scala>
scala> tpd12_df.filter($"gender"=="F").show()
+-----+-----+-----+-----+-----+
|firstname|middlename|lastname|dob|gender|salary|
+-----+-----+-----+-----+-----+
|Maria|Anne|Jones|1967-12-01|F|4000|
|Jen|Mary|Brown|1980-02-17|F|10000|

scala>
scala> tpd12_df.filter($"gender"=="M").show()
+-----+-----+-----+-----+-----+
|firstname|middlename|lastname|dob|gender|salary|
+-----+-----+-----+-----+-----+
|James|Smith|1991-04-01|M|3000|
|Michael|Rose|2000-05-19|M|4000|
|Robert|Williams|1978-09-05|M|4000|

scala>
```

- c. Bentuk kolom baru yang berisi gabungan antara first name, middle name, dan last name



```
scala> val tpd12_new = tpd12_df.withColumn("fullname",concat($"firstname",lit(" "),$"middlename",lit(" "),$"lastname"))
tpd12_new: org.apache.spark.sql.DataFrame = [firstname: string, middlename: string ... 5 more fields]

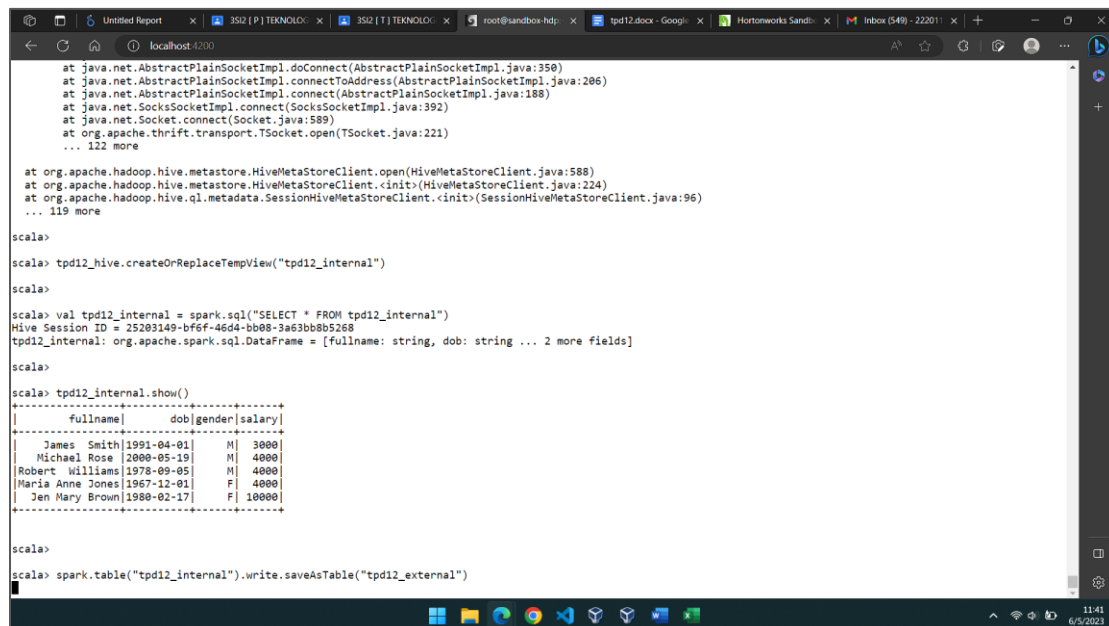
scala>
scala> tpd12_new.show()
+-----+-----+-----+-----+-----+-----+
|firstname|middlename|lastname|dob|gender|salary|fullname|
+-----+-----+-----+-----+-----+-----+
|James|Smith|1991-04-01|M|3000|James Smith| |
|Michael|Rose|2000-05-19|M|4000|Michael Rose|
|Robert|Williams|1978-09-05|M|4000|Robert Williams|
|Maria|Anne|Jones|1967-12-01|F|4000|Maria Anne Jones|
|Jen|Mary|Brown|1988-02-17|F|10000|Jen Mary Brown|
+-----+-----+-----+-----+-----+-----+

scala>
scala> val tpd12_hive = tpd12_new.select($"fullname",$"dob",$"gender",$"salary")
tpd12_hive: org.apache.spark.sql.DataFrame = [fullname: string, dob: string ... 2 more fields]

scala>
scala> tpd12_hive.show()
+-----+-----+-----+-----+
|fullname|dob|gender|salary|
+-----+-----+-----+-----+
|James Smith|1991-04-01|M|3000|
|Michael Rose|2000-05-19|M|4000|
|Robert Williams|1978-09-05|M|4000|
|Maria Anne Jones|1967-12-01|F|4000|
|Jen Mary Brown|1988-02-17|F|10000|
+-----+-----+-----+-----+
```

4. Penggunaan SQL spark

- a. Simpan kolom baru yang berisi gabungan antara first name, middle name, dan last name, dob, gender, dan salary ke dalam tabel hive.



```
scala>
scala> tpd12_hive.createOrReplaceTempView("tpd12_internal")

scala>
scala> val tpd12_internal = spark.sql("SELECT * FROM tpd12_internal")
Hive Session ID = 25203149-bf6f-46d4-bb08-3a63bb8b5268
tpd12_internal: org.apache.spark.sql.DataFrame = [fullname: string, dob: string ... 2 more fields]

scala>
scala> tpd12_internal.show()
+-----+-----+-----+-----+
|fullname|dob|gender|salary|
+-----+-----+-----+-----+
|James Smith|1991-04-01|M|3000|
|Michael Rose|2000-05-19|M|4000|
|Robert Williams|1978-09-05|M|4000|
|Maria Anne Jones|1967-12-01|F|4000|
|Jen Mary Brown|1988-02-17|F|10000|
+-----+-----+-----+-----+

scala>
scala> spark.table("tpd12_internal").write.saveAsTable("tpd12_external")
```

```
tpd12_internal: org.apache.spark.sql.DataFrame = [fullname: string, dob: string ... 2 more fields]

scala>

scala> tpd12_internal.show()
-----+-----+-----+-----+
|  fullname|      dob|gender|salary|
-----+-----+-----+-----+
|   James Smith|1991-04-01|    M|   3000|
| Michael Rose |2000-05-19|    M|   4000|
| Robert Williams|1978-09-05|    M|   4000|
| Maria Anne Jones|1967-12-01|    F|   4000|
| Jen Mary Brown|1980-02-17|    F|  10000|
-----+-----+-----+-----+

scala>

scala> spark.table("tpd12_internal").write.saveAsTable("tpd12_external")
23/06/05 04:42:05 WARN SessionState: METASTORE_FILTER_HOOK will be ignored, since hive.security.authorization.manager is set to instance of HiveAuthorizerFactory.

scala>

scala> val tpd12_external = spark.sql("SELECT * FROM tpd12_external")
tpd12_external: org.apache.spark.sql.DataFrame = [fullname: string, dob: string ... 2 more fields]

scala>

scala> tpd12_external.show()
-----+-----+-----+-----+
|  fullname|      dob|gender|salary|
-----+-----+-----+-----+
| Robert Williams|1978-09-05|    M|   4000|
| Maria Anne Jones|1967-12-01|    F|   4000|
| Jen Mary Brown|1980-02-17|    F|  10000|
|   James Smith|1991-04-01|    M|   3000|
| Michael Rose |2000-05-19|    M|   4000|
-----+-----+-----+-----+

scala>
```

- b. Lakukan transformasi dengan menambah kolom umur yang berisi hasil pengurangan tahun 2023 dengan tahun lahir
- c. Tampilkan hasil transformasi

```
scala> spark.table("tpd12_internal").write.saveAsTable("tpd12_external")
23/06/05 04:42:05 WARN SessionState: METASTORE_FILTER_HOOK will be ignored, since hive.security.authorization.manager is set to instance of HiveAuthorizerFactory.

scala>

scala> val tpd12_external = spark.sql("SELECT * FROM tpd12_external")
tpd12_external: org.apache.spark.sql.DataFrame = [fullname: string, dob: string ... 2 more fields]

scala>

scala> tpd12_external.show()
-----+-----+-----+-----+
|  fullname|      dob|gender|salary|
-----+-----+-----+-----+
| Robert Williams|1978-09-05|    M|   4000|
| Maria Anne Jones|1967-12-01|    F|   4000|
| Jen Mary Brown|1980-02-17|    F|  10000|
|   James Smith|1991-04-01|    M|   3000|
| Michael Rose |2000-05-19|    M|   4000|
-----+-----+-----+-----+

scala> val tpd12_transform = spark.sql("SELECT *, int(2023-substring(dob,0,4)) as age FROM tpd12_external")
tpd12_transform: org.apache.spark.sql.DataFrame = [fullname: string, dob: string ... 3 more fields]

scala>

scala> tpd12_transform.show()
-----+-----+-----+-----+
|  fullname|      dob|gender|salary|age|
-----+-----+-----+-----+
| Robert Williams|1978-09-05|    M|   4000|  45|
| Maria Anne Jones|1967-12-01|    F|   4000|  56|
| Jen Mary Brown|1980-02-17|    F|  10000|  43|
|   James Smith|1991-04-01|    M|   3000|  32|
| Michael Rose |2000-05-19|    M|   4000|  23|
-----+-----+-----+-----+

scala>
```

5. Simpan hasil transformasi pada nomor 4 dalam format orc di hdfs

```
scala> val tpd12_transform = spark.sql("SELECT *, int(2023-substring(dob,0,4)) as age FROM tpd12_external")
tpd12_transform: org.apache.spark.sql.DataFrame = [fullname: string, dob: string ... 3 more fields]

scala>

scala> tpd12_transform.show()
+-----+
|      fullname|      dob|gender|salary|age|
+-----+
|Robert Williams|1978-09-05|M|4000|45|
|Maria Anne Jones|1967-12-01|F|4000|56|
|Jen Mary Brown|1980-02-17|F|10000|43|
|James Smith|1991-04-01|M|3000|32|
|Michael Rose|2000-05-19|M|4000|23|
+-----+

scala> tpd12_transform.write.orc("/user/root/tpd12/transform_result/")

scala>

scala> val tpd12_read_orc = spark.read.orc("/user/root/tpd12/transform_result/")
tpd12_read_orc: org.apache.spark.sql.DataFrame = [fullname: string, dob: string ... 3 more fields]

scala>

scala> tpd12_read_orc.show()
+-----+
|      fullname|      dob|gender|salary|age|
+-----+
|Robert Williams|1978-09-05|M|4000|45|
|Maria Anne Jones|1967-12-01|F|4000|56|
|Jen Mary Brown|1980-02-17|F|10000|43|
|James Smith|1991-04-01|M|3000|32|
|Michael Rose|2000-05-19|M|4000|23|
+-----+

scala> 
```