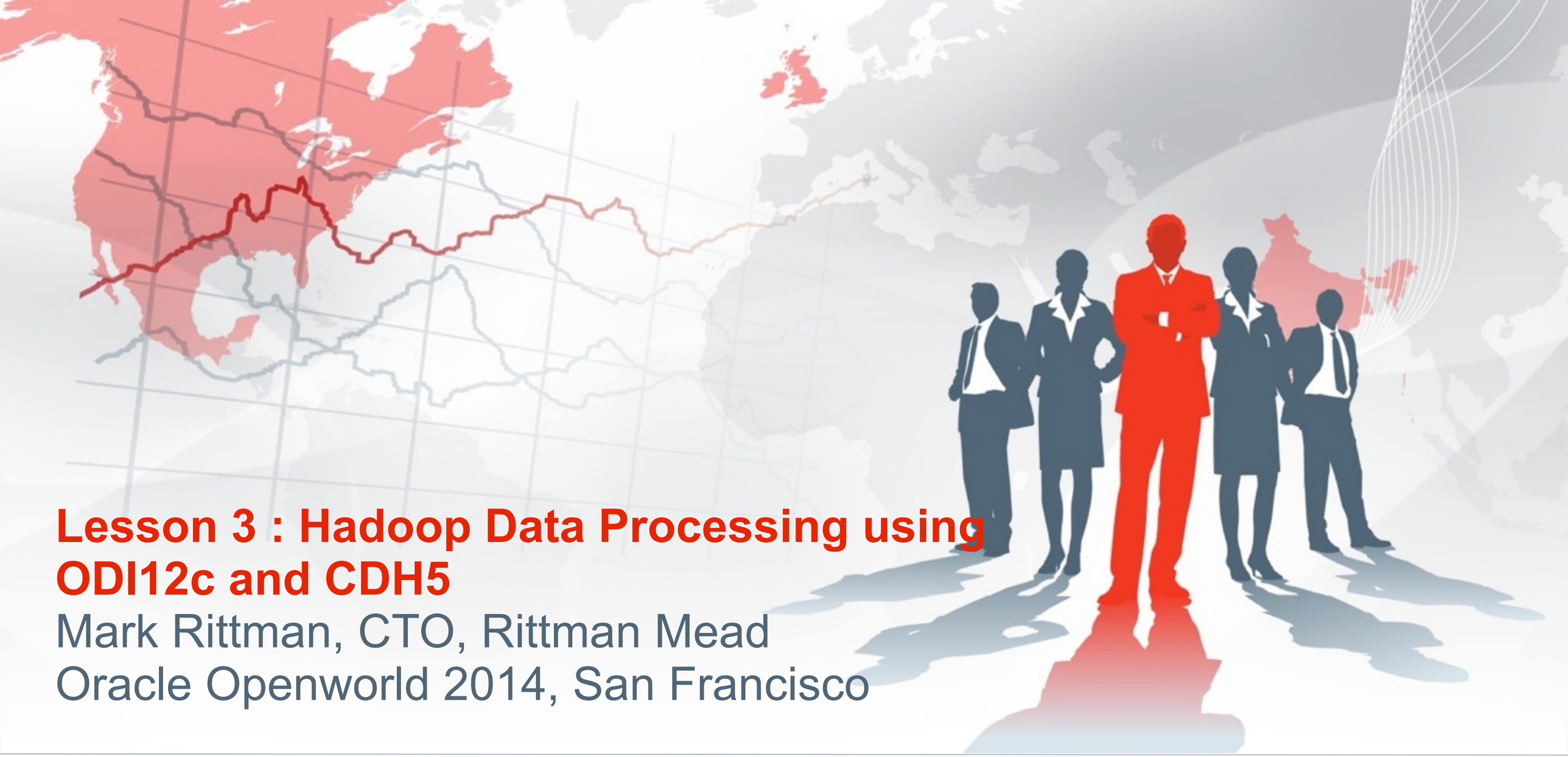


Lesson 3 : Hadoop Data Processing using ODI12c and CDH5

Mark Rittman, CTO, Rittman Mead

Oracle Openworld 2014, San Francisco

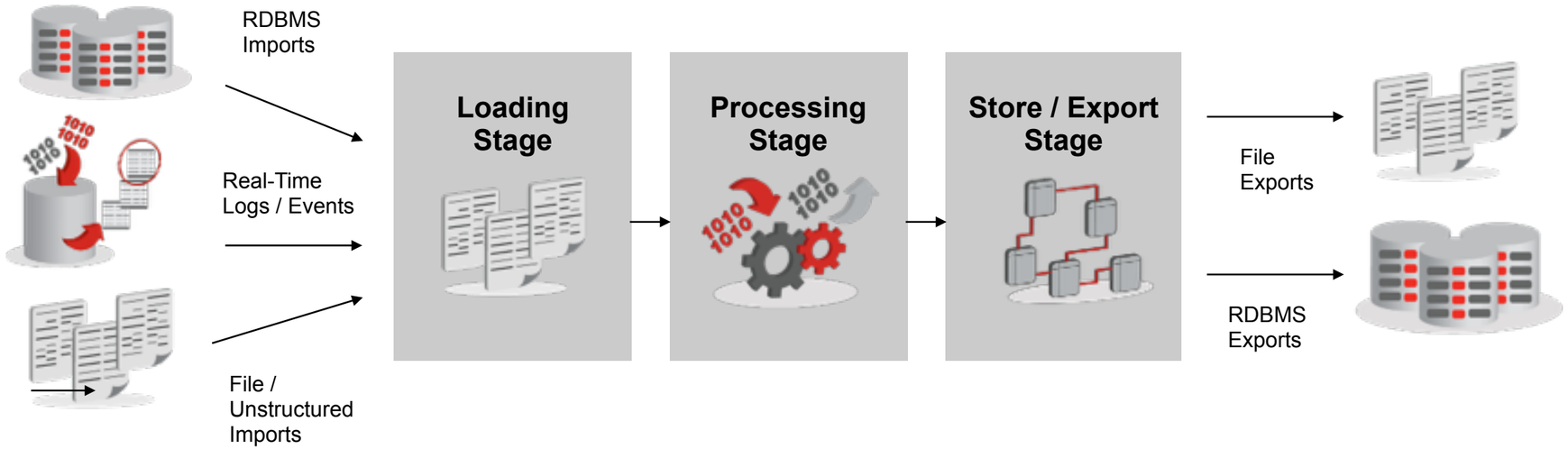
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Moving Data In, Around and Out of Hadoop

- Three stages to Hadoop data movement, with dedicated Apache / other tools
  - **Load** : receive files in batch, or in real-time (logs, events)
  - **Transform** : process & transform data to answer questions
  - **Store / Export** : store in structured form, or export to RDBMS using Sqoop



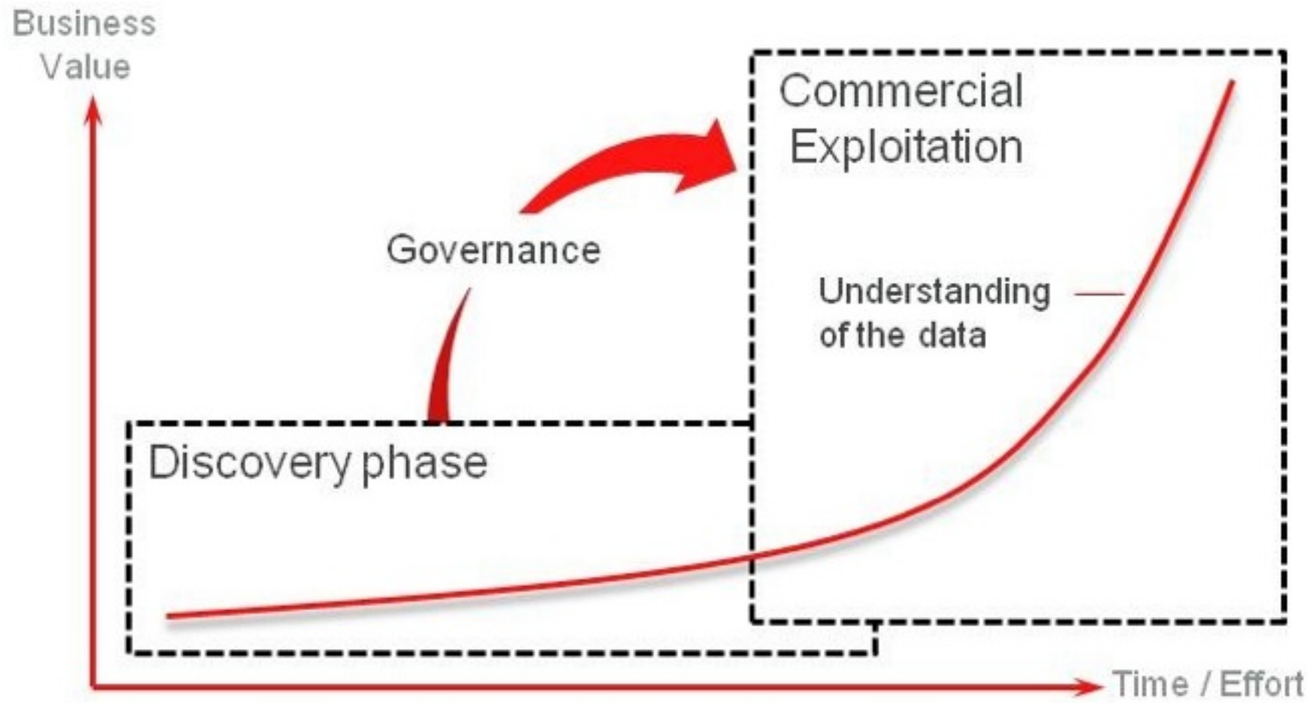
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Discovery vs. Exploitation Project Phases

- Discovery and monetising steps in Big Data projects have different requirements
- Discovery phase
  - Unbounded discovery
  - Self-Service sandbox
  - Wide toolset
- Promotion to Exploitation
  - Commercial exploitation
  - Narrower toolset
  - Integration to operations
  - Non-functional requirements
  - Code standardisation & gover

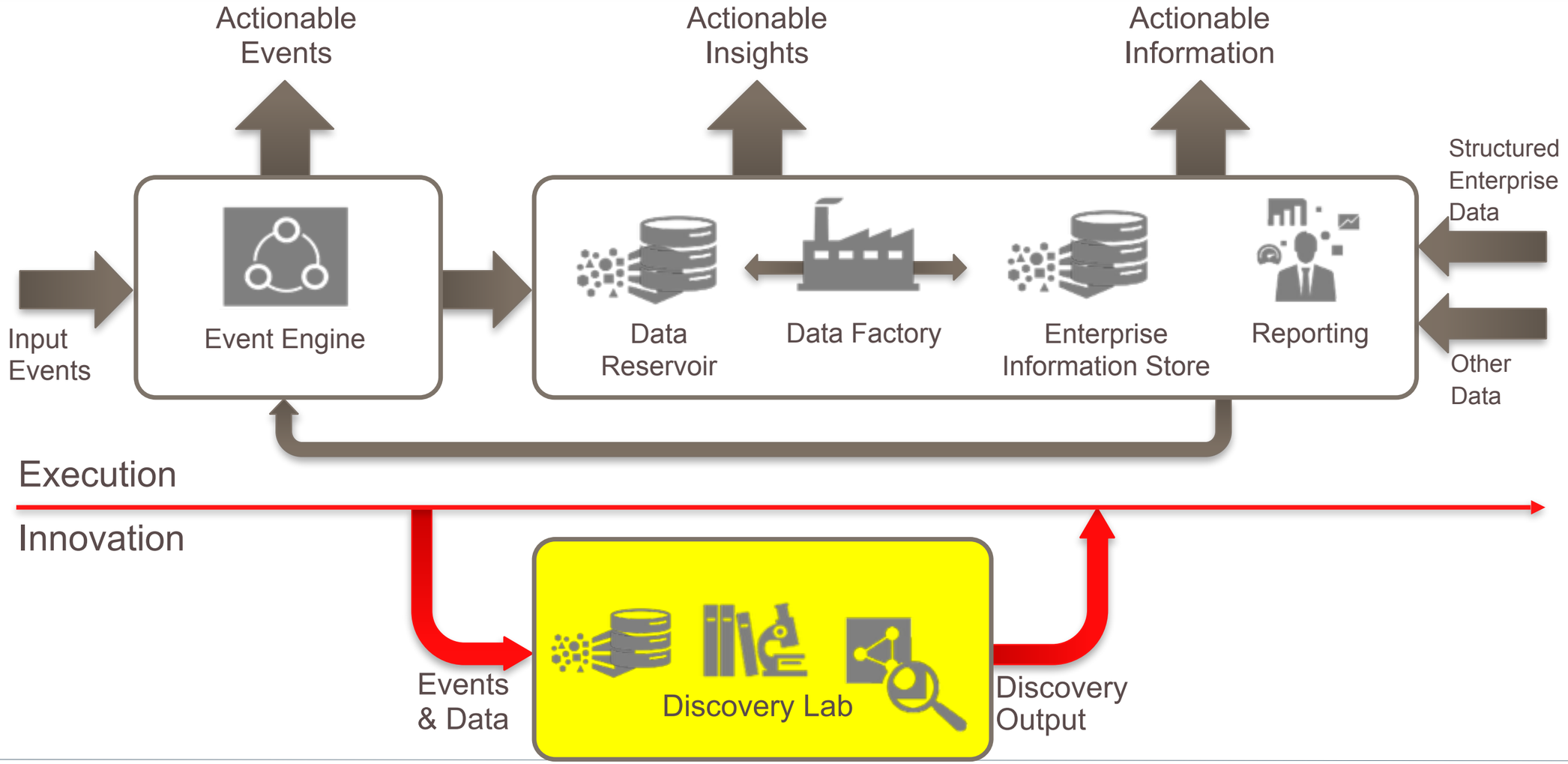


T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Introducing the “Discovery Lab”



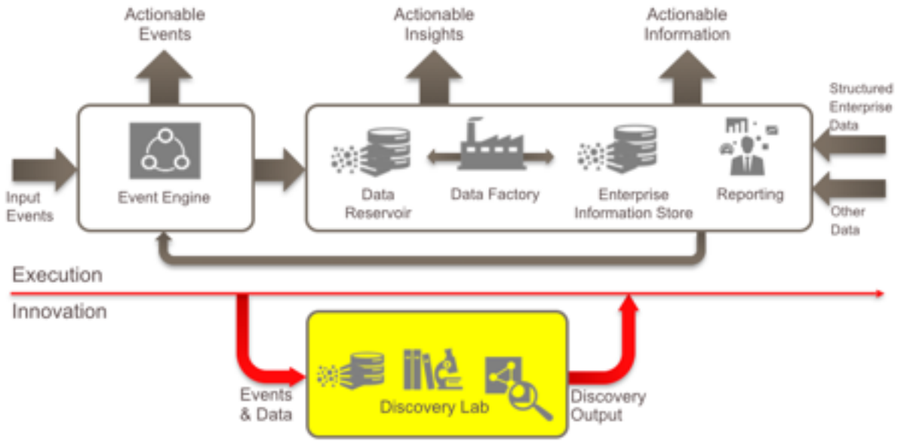
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Design Pattern : Discovery Lab

- Specific focus on identifying commercial value for exploitation
- Small group of highly skilled individuals (aka Data Scientists)
- Iterative development approach – data oriented NOT development oriented
- Wide range of tools and techniques applied
  - Searching and discovering unstructured data
  - Finding correlations and clusters
  - Filtering, aggregating, deriving and enhancing data
- Data provisioned through Data Factory or own ETL
- Typically separate infrastructure but could also be unified Reservoir if resource managed effectively



T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Lesson 3 : Hadoop Data Processing

Hadoop Data Processing / Analysis Fundamentals

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Core Apache Hadoop Tools

- Apache Hadoop, including MapReduce and HDFS
  - Scaleable, fault-tolerant file storage for Hadoop
  - Parallel programming framework for Hadoop
- Apache Hive
  - SQL abstraction layer over HDFS
  - Perform set-based ETL within Hadoop
- Apache Pig, Spark
  - Dataflow-type languages over HDFS, Hive etc
  - Extensible through UDFs, streaming etc
- Apache Flume, Apache Sqoop, Apache Kafka
  - Real-time and batch loading into HDFS
  - Modular, fault-tolerant, wide source/target coverage

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Other Tools Typically Used…

- Python, Scala, Java and other programming languages
  - For more complex and procedural transformations
- Shell scripts, sed, awk, regexes etc
- R and R-on-Hadoop
  - Typically at the “discovery” phase
- And down the line - ETL tools to automate the process
  - ODI, Pentaho Data Integrator etc

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

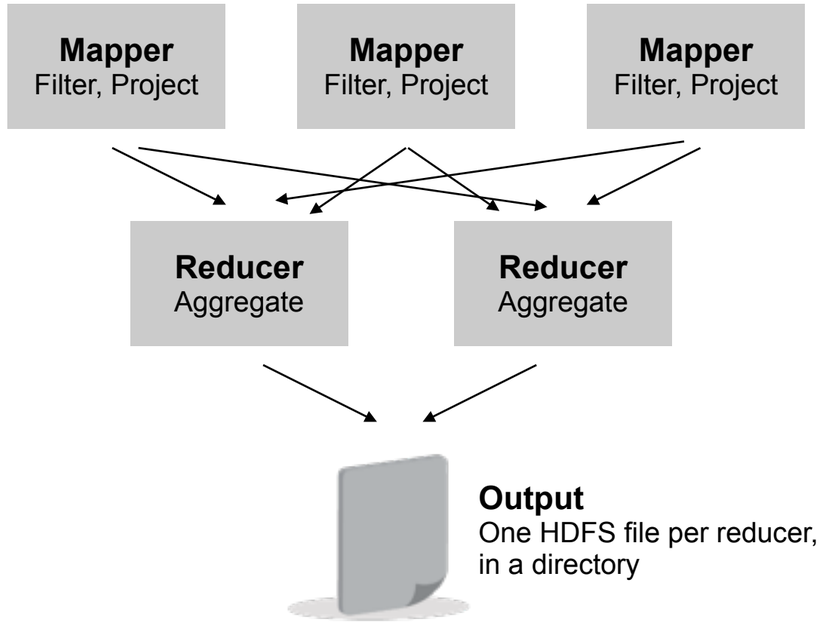
E : info@rittmanmead.com
W : www.rittmanmead.com

Techniques and Tools We'll Cover in this Seminar

- Apache Hive (and Cloudera Impala)
- Apache Pig
- Apache Spark
- R, and Oracle R Advanced Analytics for Hadoop
- Automating this processing using ODI12c

In the Beginning, There Was … MapReduce

- Programming model for processing large data sets in parallel on a cluster
- Not specific to a particular language, but usually written in Java
- Inspired by the **map** and **reduce** functions commonly used in functional programming
  - `Map()` performs filtering and sorting
  - `Reduce()` aggregates the output of mappers
  - and a `Shuffle()` step to redistribute output by keys
- Resolved several complications of distributed computing:
  - Allows unlimited computations on unlimited data
  - Map and reduce functions can be easily distributed
  - Combined with Hadoop, very network and rack aware, minimising network traffic and inherently fault-tolerant



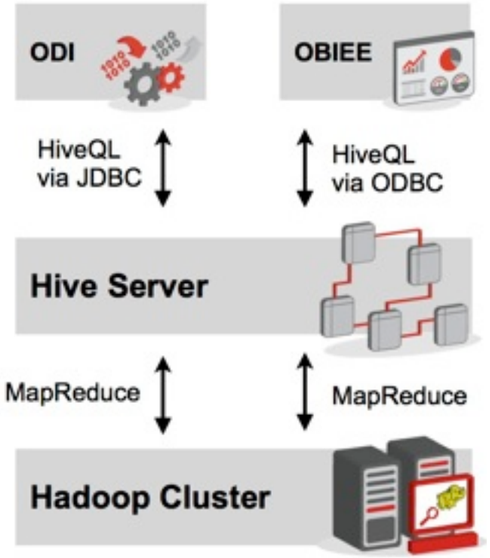
… But writing MapReduce Code is Hard

- Typically written in Java
- Requires programming skills (though Hadoop takes care of parallelism, fault tolerance)

```
package net.pascalalma.hadoop;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

import java.io.IOException;

public class AllTranslationsReducer extends Reducer<Text, Text, Text, Text> {

    private Text result = new Text();

    @Override
    protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
        String translations = "";

        for (Text val : values) {
            translations += "|" + val.toString();
        }

        result.set(translations);
        context.write(key, result);
    }
}
```

Hive as the Hadoop SQL Access Layer

- Hive can make generating MapReduce easier
- A query environment over Hadoop/MapReduce to support SQL-like queries
- Hive server accepts HiveQL queries via HiveODBC or HiveJDBC, automatically creates MapReduce jobs against data previously loaded into the Hive HDFS tables
- Approach used by ODI and OBIEE to gain access to Hadoop data
- Allows Hadoop data to be accessed just like any other data source (sort of...)



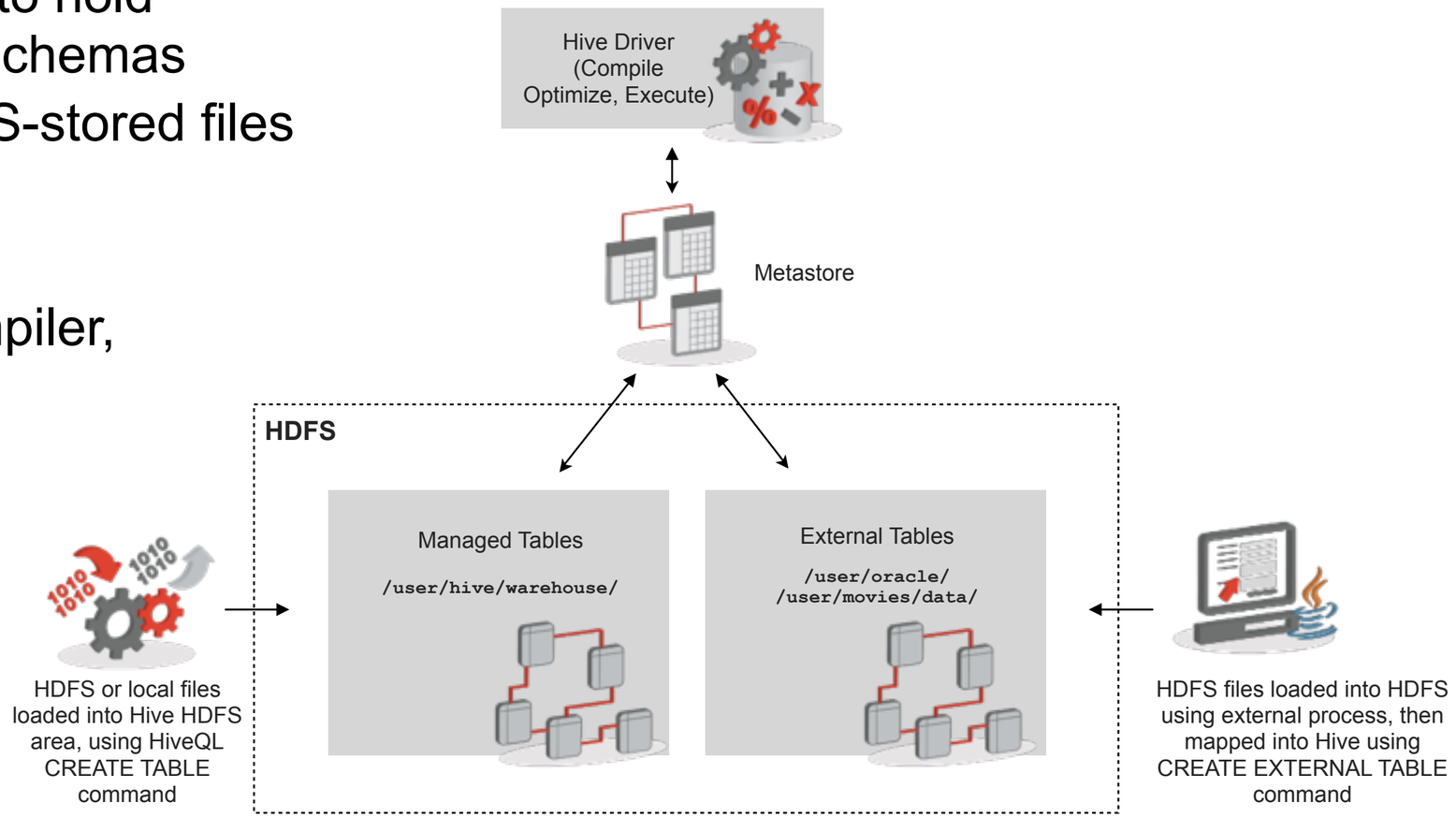
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

How Hive Provides SQL Access over Hadoop

- Hive uses a RBDMS metastore to hold table and column definitions in schemas
- Hive tables then map onto HDFS-stored files
  - Managed tables
  - External tables
- Oracle-like query optimizer, compiler, executor
- JDBC and OBDC drivers, plus CLI etc



T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Typical Hive Interactions

```
•CREATE TABLE test (
     product_id int,
     product_desc string);

•SHOW TABLES;

•CREATE TABLE test2
   AS SELECT * FROM test;

•SELECT SUM(sales)
 FROM sales_summary;

•LOAD DATA INPATH '/user/mrittman/logs' INTO TABLE log_entries;
```

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

An example Hive Query Session: Connect and Display Table List

```
[oracle@bigdatalite ~]$ hive
Hive history file=/tmp/oracle/hive_job_log_oracle_201304170403_1991392312.txt
hive> show tables;
OK
dwh_customer
dwh_customer_tmp
i_dwh_customer
ratings
src_customer
src_sales_person
weblog
weblog_preprocessed
weblog_sessionized
Time taken: 2.925 seconds
```

Hive Server lists out all "tables" that have been defined within the Hive environment

An example Hive Query Session: Display Table Row Count

```
hive> select count(*) from src_customer;


Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=
In order to set a constant number of reducers:
set mapred.reduce.tasks=
Starting Job = job_201303171815_0003, Tracking URL =
  http://localhost.localdomain:50030/jobdetails.jsp?jobid=job_201303171815_0003
Kill Command = /usr/lib/hadoop-0.20/bin/
 hadoop job -Dmapred.job.tracker=localhost.localdomain:8021 -kill job_201303171815_0003

2013-04-17 04:06:59,867 Stage-1 map = 0%, reduce = 0%
2013-04-17 04:07:03,926 Stage-1 map = 100%, reduce = 0%
2013-04-17 04:07:14,040 Stage-1 map = 100%, reduce = 33%
2013-04-17 04:07:15,049 Stage-1 map = 100%, reduce = 100%
Ended Job = job_201303171815_0003
OK


25
Time taken: 22.21 seconds
```

Request count(*) from table

Hive server generates MapReduce job to “map” table key/value pairs, and then reduce the results to table count

MapReduce job automatically run by Hive Server

Results returned to user

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Hive SerDes - Process Semi-Structured Data

- Plug-in technology to Hive that allows it to parse data, and access alternatives to HDFS for data storage
- Distributed as JAR file, gives Hive ability to parse semi-structured formats
- We can use the RegEx SerDe to parse the Apache CombinedLogFormat file into columns

```
CREATE EXTERNAL TABLE apachelog (
  host STRING,
  identity STRING,
  user STRING,
  time STRING,
  request STRING,
  status STRING,
  size STRING,
  referer STRING,
  agent STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
  "input.regex" = "([^ ]*) ([^ ]*) ([^ ]*) (-|\\[[^\\]]*\\]) ([^ \"]*|\"[^\"]*\") (-|[0-9]*) (-|[0-9]*)(?: ([^ \"]*|\"[^\"]*\")
([^ \"]*|\"[^\"]*\"))?",
  "output.format.string" = "%1$s %2$s %3$s %4$s %5$s %6$s %7$s %8$s %9$s"
)
STORED AS TEXTFILE
LOCATION '/user/root/logs';
```

Hive and HDFS File Storage

- Hive tables can either map to a single HDFS file, or a directory of files
  - Entire contents of directory becomes source for table
- Directories can have sub-directories, to provide “partitioning” feature for Hive
  - Only scan and process those subdirectories relevant to query
- Combined with SerDes, a useful way to process and parse lots of separate log files

```
[root@cdh51-node1 ~]# hadoop fs -ls /user/flume/rm_logs/apache_access_combined
Found 278 items
-rw-r--r--   3 root root     672480 2014-10-06 14:31 /user/flume/rm_logs/apache_access_combined/FlumeData.1412601698996
-rw-r--r--   3 root root     727711 2014-10-06 14:41 /user/flume/rm_logs/apache_access_combined/FlumeData.1412602299095
-rw-r--r--   3 root root     707441 2014-10-06 14:51 /user/flume/rm_logs/apache_access_combined/FlumeData.1412602915327
-rw-r--r--   3 root root     807375 2014-10-06 15:02 /user/flume/rm_logs/apache_access_combined/FlumeData.1412603531022
-rw-r--r--   3 root root     785963 2014-10-06 15:12 /user/flume/rm_logs/apache_access_combined/FlumeData.1412604138450
-rw-r--r--   3 root root     534005 2014-10-06 15:22 /user/flume/rm_logs/apache_access_combined/FlumeData.1412604744386
-rw-r--r--   3 root root     634051 2014-10-06 15:32 /user/flume/rm_logs/apache_access_combined/FlumeData.1412605344622
-rw-r--r--   3 root root     737031 2014-10-06 15:42 /user/flume/rm_logs/apache_access_combined/FlumeData.1412605968231
-rw-r--r--   3 root root     670881 2014-10-06 15:53 /user/flume/rm_logs/apache_access_combined/FlumeData.1412606584235
-rw-r--r--   3 root root     800607 2014-10-06 16:03 /user/flume/rm_logs/apache_access_combined/FlumeData.1412607185371
-rw-r--r--   3 root root     684562 2014-10-06 16:13 /user/flume/rm_logs/apache_access_combined/FlumeData.1412607794366
-rw-r--r--   3 root root     846410 2014-10-06 16:23 /user/flume/rm_logs/apache_access_combined/FlumeData.1412608398806
-rw-r--r--   3 root root     576884 2014-10-06 16:33 /user/flume/rm_logs/apache_access_combined/FlumeData.1412608999875
-rw-r--r--   3 root root     601540 2014-10-06 16:43 /user/flume/rm_logs/apache_access_combined/FlumeData.1412609607071
-rw-r--r--   3 root root     559014 2014-10-06 16:53 /user/flume/rm_logs/apache_access_combined/FlumeData.1412610215067
```

Hive Storage Handlers - Access NoSQL Databases

- MongoDB Hadoop connector allows MongoDB to be accessed via Hive tables

```
{
    "interactionId": "f35ace79b903eedfe5198f386d6fda0c",
    "subscriptionId": "ABC123456a891e3406789123216789",
    "hash": null,
    "hashType": null,
    "interaction": {
        "schema": {
            "version": 3
        },
        "source": "Twitter for Android",
        "type": "twitter",
        "created_at": "Thu, 09 May 2013 08:59:46 +0000",
        "content": "RT @Real_Liam_Payne: Thank you denmarkkkk :) love youuuu :)",
        "id": "f35ace79b903eedf75198f386d6fd40b",
        "author": {
            "hash_id": "c6add0a675830a785598a513d586203c"
        },
        "tags": [
            "type.share",
            "demo.tag",
            "another.one",
            "foobar"
        ]
    }
}
```

basic-interaction-meta.json hosted with ❤ by GitHub view raw

```
CREATE TABLE tweet_data(
  interactionId string,
  username string,
  content string,
  author_followers int)
ROW FORMAT SERDE
  'com.mongodb.hadoop.hive.BSONSerDe'
STORED BY
  'com.mongodb.hadoop.hive.MongoStorageHandler'
WITH SERDEPROPERTIES (
  'mongo.columns.mapping'='{"interactionId":"interactionId",
  "username":"interaction.interaction.author.username",
  "content":\"interaction.interaction.content",
  "author_followers_count":"interaction.twitter.user.followers_
count"}'
  )
TBLPROPERTIES (
  'mongo.uri'='mongodb://cdh51-node1:27017/
datasiftmongodb.rm_tweets'
  )
```

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Hive Extensibility through UDFs and UDAFs

- Extend Hive by adding new computation and aggregation capabilities
  - UDFs (row-based), UDAFs (aggregation) and UDTFs (table functions)

```
public class JsonSplitUDF extends GenericUDF {
    private StringObjectInspector stringInspector;

    @Override
    public Object evaluate(DeferredObject[] arguments)
     throws HiveException {
        try {
            String jsonString = this.stringInspector.
            getPrimitiveJavaObject(arguments[0].get());

            ObjectMapper om = new ObjectMapper();
            ArrayList<Object> root = (ArrayList<Object>)
             om.readValue(jsonString, ArrayList.class);
            ArrayList<Object[]> json = new ArrayList<Object[]>
             (root.size());
            for (int i=0; i<root.size(); i++){
                json.add(new Object[]{i,
                om.writeValueAsString(root.get(i))});
            }
            return json;}}
```

```
add jar target/JsonSplit-1.0-SNAPSHOT.jar;
create temporary function json_split
 as 'com.pythian.hive.udf.JsonSplitUDF';

create table json_example (json string);
load data local inpath 'split_example.json'
into table json_example;

SELECT ex.* FROM json_example
LATERAL VIEW explode(json_split(json_example.json)) ex;
```

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Hive Extensibility through Streaming

- TRANSFORM function streams query columns through arbitrary script
- Use Python, Java etc to transform Hive data when UDFs etc not sufficient

```
import sys
import datetime

for line in sys.stdin:
  line = line.strip()
  userid, movieid, rating, unixtime = line.split('\t')
  weekday = datetime.datetime.fromtimestamp(float(unixtime)).isoweekday()
  print '\t'.join([userid, movieid, rating, str(weekday)])
```

```
add FILE weekday_mapper.py;

INSERT OVERWRITE TABLE u_data_new
SELECT
  TRANSFORM (userid, movieid, rating, unixtime)
  USING 'python weekday_mapper.py'
  AS (userid, movieid, rating, weekday)
FROM u_data;
```

Distributing SerDe JAR Files for Hive across Cluster

- Hive SerDe and UDF functionality requires additional JARs to be made available to Hive
- Following steps must be performed across ALL Hadoop nodes:
  - Add JAR reference to HIVE_AUX_JARS_PATH in /usr/lib/hive/conf/hive.env.sh

```
export HIVE_AUX_JARS_PATH=/usr/lib/hive/lib/hive-contrib-0.12.0-cdh5.0.1.jar:$
(echo $HIVE_AUX_JARS_PATH…
```

  - Add JAR file to /usr/lib/hadoop

```
[root@bdanode1 hadoop]# ls /usr/lib/hadoop/hive-*

/usr/lib/hadoop/hive-contrib-0.12.0-cdh5.0.1.jar
```

  - Restart YARN / MR1 TaskTrackers across cluster

Hive Data Processing Example : Find Top Referers

- Return the top 5 website URLs linking to the Rittman Mead website
- Exclude links from our own website

```
select referer, count(*) as cnt
from   apachelog
where  substr(referer,1,28) <> '"http://www.rittmanmead.com/'
group by referer
order by cnt desc
limit 5
```

Recent queries | Query | Log | Columns | Results | Chart

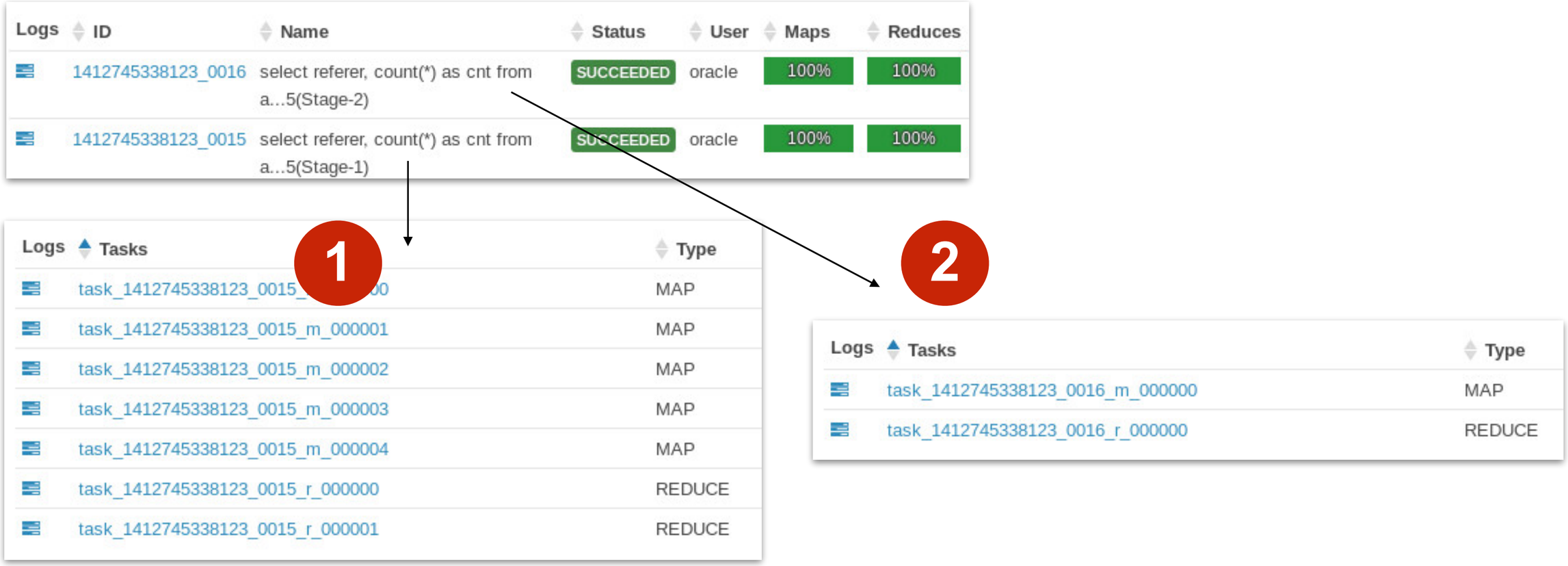
	referer	cnt
0	"-"	1575976
1	"http://obieetraining11.blogspot.in/search/label/OBIEE%2011g"	16639
2	"http://www.orafaq.com/aggregator/sources/14"	12777
3	"https://www.google.com/"	11717
4	"http://www.orafaq.com/aggregator/sources/14?page=1"	9946

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

How Hive Turns HiveQL into MapReduce + Hadoop Tasks

- Two step process; first step filters and groups the data, second sorts and returns top 5



T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

SQL Considerations : Using Hive vs. Regular Oracle SQL

- Not all join types are available in Hive - joins must be equality joins
- No sequences, no primary keys on tables
- Generally need to stage Oracle or other external data into Hive before joining to it
- Hive latency - not good for small microbatch-type work
  - But other alternatives exist - Spark, Impala etc
- Hive is INSERT / APPEND only - no updates, deletes etc
  - But HBase may be suitable for CRUD-type loading
- Don’t assume that HiveQL == Oracle SQL
  - Test assumptions before committing to platform

vs.

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Apache Pig : Set-Based Dataflow Language

- Alternative to Hive, defines data manipulation as dataflow steps (like an execution plan)
- Start with one or more data sources, add steps to apply filters, group, project columns
- Generates MapReduce to execute data flow, similar to Hive; extensible through UDFs

```
a = load '/user/oracle/pig_demo/marriott_wifi.txt';
b = foreach a generate flatten(TOKENIZE((chararray)$0)) as word;
c = group b by word;
d = foreach c generate COUNT(b), group;
store d into '/user/oracle/pig_demo/pig_wordcount';
```

1

2

Logs	Tasks	Type
	task_1412745338123_0018_m_000000	MAP
	task_1412745338123_0018_r_000000	REDUCE

```
[oracle@bigdatalite ~]$ hadoop fs -ls /user/oracle/pig_demo/pig_wordcount
Found 2 items
-rw-r--r--   1 oracle oracle          0 2014-10-11 11:48 /user/oracle/pig_demo/pig_wordcount/_SUCCESS
-rw-r--r--   1 oracle oracle       1965 2014-10-11 11:48 /user/oracle/pig_demo/pig_wordcount/part-r-00000
[oracle@bigdatalite ~]$ hadoop fs -cat /user/oracle/pig_demo/pig_wordcount/part-r-00000
2               .
1               I
6               a
...
```

3

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Apache Pig Characteristics vs. Hive

- Ability to load data into a defined schema, or use schema-less (access fields by position)
- Fields can contain nested fields (tuples)
- Grouping records on a key doesn't aggregate them, it creates a nested set of rows in column
- Uses "lazy execution" - only evaluates data flow once final output has been requests
- Makes Pig an excellent language for interactive data exploration

vs.

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Pig Data Processing Example : Count Page Request Totals

```
raw_logs =LOAD '/user/oracle/rm_logs/' USING TextLoader AS (line:chararray);
logs_base = FOREACH raw_logs
 GENERATE FLATTEN
    (
        REGEX_EXTRACT_ALL
        (
            line,
            '^(\\S+) (\\S+) (\\S+) \\[([\\w:/]+\\s[+\\-]\\d{4})\\] "(.+?)" (\\S+) (\\S+) "([^"]*)" "([^"]*)"'
        )
    )
    AS
    (
        remoteAddr: chararray, remoteLogname: chararray, user: chararray,
        time: chararray, request: chararray, status: chararray, bytes_string: chararray,
        referrer: chararray, browser: chararray
    );
page_requests = FOREACH logs_base
 GENERATE SUBSTRING(time,3,6) as month,
 FLATTEN(STRSPLIT(request,' ',5)) AS (method:chararray, request_page:chararray, protocol:chararray);
page_requests_short = FOREACH page_requests
 GENERATE $0,$2;
page_requests_short_filtered = FILTER page_requests_short BY (request_page is not null AND SUBSTRING(request_page,0,3) == '/20');
page_request_group = GROUP page_requests_short_filtered BY request_page;
page_request_group_count = FOREACH page_request_group GENERATE $0, COUNT(page_requests_short_filtered) as total_hits;
page_request_group_count_sorted = ORDER page_request_group_count BY $1 DESC;
page_request_group_count_limited = LIMIT page_request_group_count_sorted 10;
```

Demo

Running Pig using the Hue Pig Editor in CDH5

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Pig Data Processing Example : Join to Post Titles, Authors

- Pig allows aliases (datasets) to be joined to each other
- Example below adds details of post names, authors; outputs top pages dataset to file

```
raw_posts = LOAD '/user/oracle/pig_demo/posts_for_pig.csv' USING TextLoader AS (line:chararray);
posts_line = FOREACH raw_posts
 GENERATE FLATTEN
 (
  STRSPLIT(line,';',10)
 )
 AS
 (
  post_id: chararray, title: chararray, post_date: chararray,
  type: chararray, author: chararray, post_name: chararray,
  url_generated: chararray
 );
posts_and_authors = FOREACH posts_line
 GENERATE title,author,post_name,CONCAT(REPLACE(url_generated,'"',''),'/') AS (url_generated:chararray);
pages_and_authors_join = JOIN posts_and_authors BY url_generated, page_request_group_count_limited BY group;
pages_and_authors = FOREACH pages_and_authors_join GENERATE url_generated, post_name, author, total_hits;
top_pages_and_authors = ORDER pages_and_authors BY total_hits DESC;
STORE top_pages_and_authors into '/user/oracle/pig_demo/top-pages-and-authors.csv' USING PigStorage(`,');
```

Pig Extensibility through UDFs and Streaming

- Similar to Apache Hive, Pig can be programatically extended through UDFs
- Example below uses Function defined in Python script to geocode IP addresses

```
#!/usr/bin/python

import sys
sys.path.append('/usr/lib/python2.6/site-packages/')
import pygeoip

@outputSchema("country:chararray")
def getCountry(ip):
    gi = pygeoip.GeoIP('/home/nelio/GeoIP.dat')
    country = gi.country_name_by_addr(ip)
    return country
```



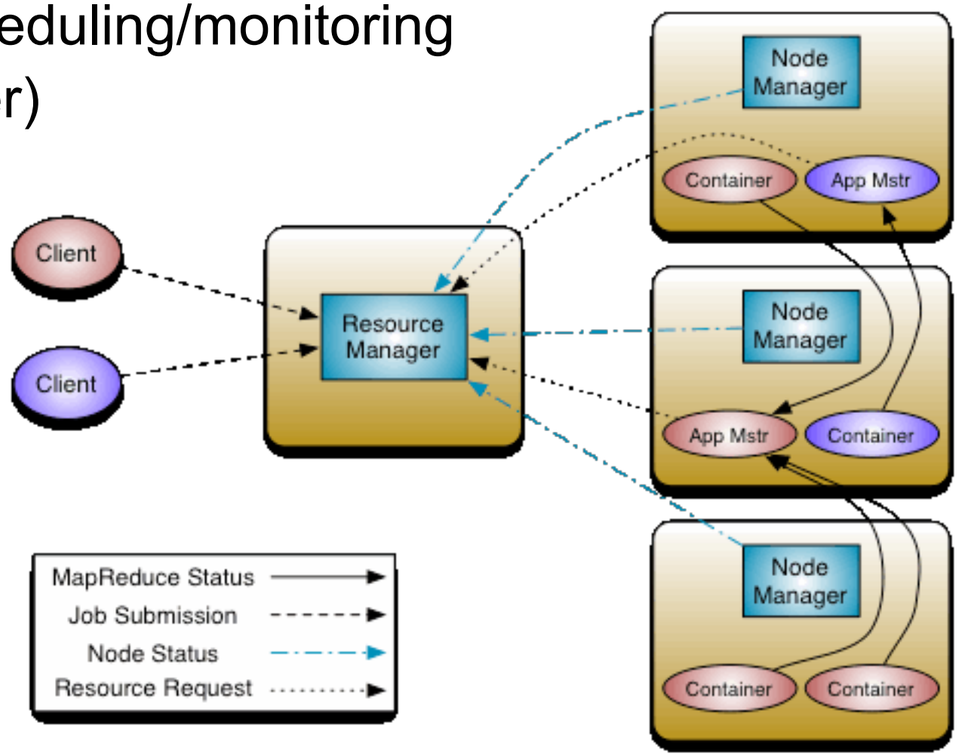
```
register 'python_geoip.py' using jython as pythonGeoIP;
raw_logs =LOAD '/user/root/logs/' USING TextLoader AS (line:chararray);
logs_base = FOREACH raw_logs
 GENERATE FLATTEN
   (
    REGEX_EXTRACT_ALL
     (
      line,
       '^(\\S+) (\\S+) (\\S+) \\[([\\w:/]+\\s[+\\-]\\d{4})\\]
"(.+?)" (\\S+) (\\S+) "([^"]*)" "([^"]*)"'
     )
   )
AS (
     remoteAddr: chararray, remoteLogname: chararray, user: chararray,
     time: chararray, request: chararray,
     status: int, bytes_string: chararray, referrer: chararray,
     browser: chararray
   );
ipaddress =  FOREACH logs_base GENERATE remoteAddr;
clean_ip  = FILTER ipaddress BY
(remoteAddr matches '^.*?((?:\\d{1,3}\\.){3}\\d{1,3}).*?$');
country_by_ip = FOREACH clean_ip
  GENERATE pythonGeoIP.getCountry(remoteAddr);
```

MapReduce and Hadoop 1.0 Limitations

- MapReduce, and the original Hadoop framework, worked well for batch-type analysis
  - Made it possible to load, process and analyse at scale, economically
- But MapReduce is not well suited to interactive, ad-hoc analysis type work
  - Each step in the process requires JVMs to be started, and data typically resides on disk
- Hadoop 1.0 was also tightly bound to MapReduce
  - API, framework and resource management all assumed MapReduce only
  - Limits usefulness as new processing types come into use

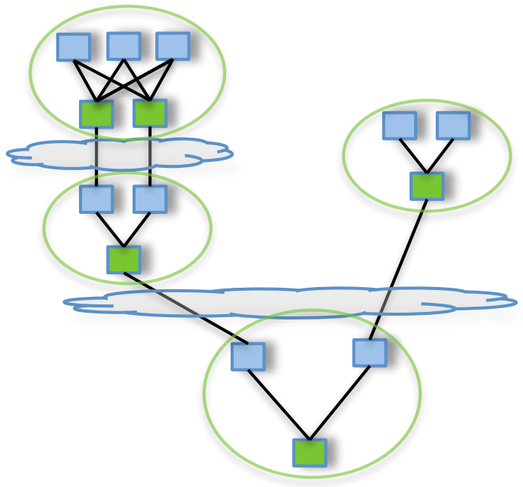
MR2 and YARN

- MapReduce 2 (MR2) splits the functionality of the JobTracker by separating resource management and job scheduling/monitoring
- Introduces YARN (Yet Another Resource Manager)
- Permits other processing frameworks to MR
  - For example, Apache Spark
- Maintains backwards compatibility with MR1
- Introduced with CDH5+

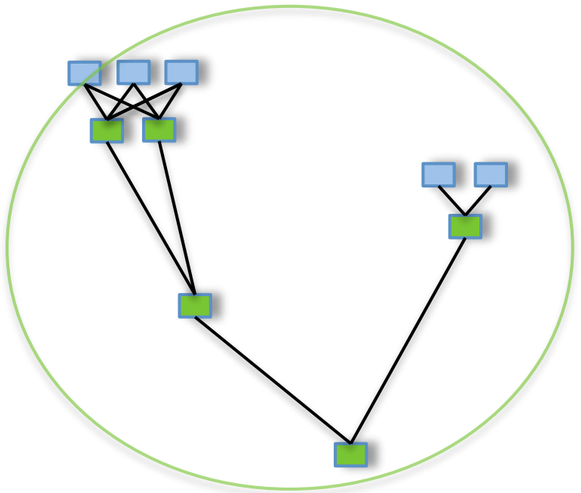


Apache Tez

- Runs on top of YARN, provides a faster execution engine than MapReduce for Hive, Pig etc
- Models processing as an entire data flow graph (DAG), rather than separate job steps
  - DAG (Directed Acyclic Graph) is a new programming style for distributed systems
  - Dataflow steps pass data between them as streams, rather than writing/reading from disk
- Supports in-memory computation, enables Hive on Tez (Stinger) and Pig on Tez
- Favoured In-memory / Hive v2 route by Hortonworks



Pig/Hive - MR



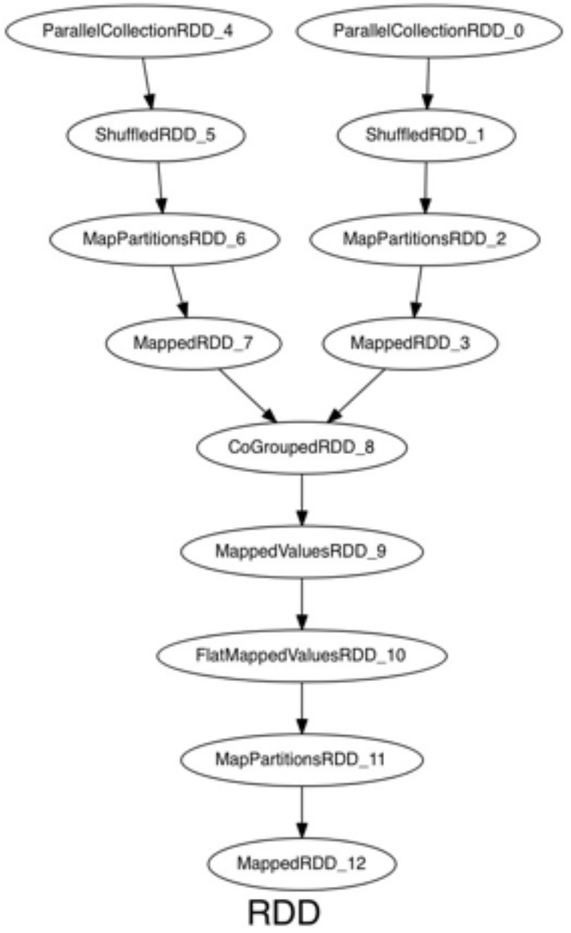
Pig/Hive - Tez

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Apache Spark

- Another DAG execution engine running on YARN
- More mature than TEZ, with richer API and more vendor support
- Uses concept of an RDD (Resilient Distributed Dataset)
  - RDDs like tables or Pig relations, but can be cached in-memory
  - Great for in-memory transformations, or iterative/cyclic processes
- Spark jobs comprise of a DAG of tasks operating on RDDs
- Access through Scala, Python or Java APIs
- Related projects include
  - Spark SQL
  - Spark Streaming



T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Apache Spark Example : Simple Log Analysis

- Load logfile into RDD, do row count

```
scala> val logfile = sc.textFile("logs/access_log")
14/05/12 21:18:59 INFO MemoryStore: ensureFreeSpace(77353) called with curMem=234759, maxMem=309225062
14/05/12 21:18:59 INFO MemoryStore: Block broadcast_2 stored as values to memory (estimated size 75.5 KB, free 294.6 MB)
logfile: org.apache.spark.rdd.RDD[String] = MappedRDD[31] at textFile at <console>:15

scala> logfile.count()
14/05/12 21:19:06 INFO FileInputFormat: Total input paths to process : 1
14/05/12 21:19:06 INFO SparkContext: Starting job: count at <console>:1
...
14/05/12 21:19:06 INFO SparkContext: Job finished: count at <console>:18, took 0.192536694 s
res7: Long = 154563
```

- Load logfile into RDD and cache it, create another RDD from it filtered on /biapps11g/

```
scala> val logfile = sc.textFile("logs/access_log").cache
scala> val biapps11g = logfile.filter(line => line.contains("/biapps11g/"))
biapps11g: org.apache.spark.rdd.RDD[String] = FilteredRDD[34] at filter at <console>:17
scala> biapps11g.count()
...
14/05/12 21:28:28 INFO SparkContext: Job finished: count at <console>:20, took 0.387960876 s
res9: Long = 403
```

Apache Spark Example : Simple Log Analysis

- Import a log parsing library, then use it to generate a list of URIs creating 404 errors

```
scala> import com.alvinalexander.accesslogparser._

val p = new AccessLogParser
def getStatusCode(line: Option[AccessLogRecord]) = {
  line match {
    case Some(l) => l.httpStatusCode
    case None => "0"
  }
}
def getRequest(rawAccessLogString: String): Option[String] = {
  val accessLogRecordOption = p.parseRecord(rawAccessLogString)
  accessLogRecordOption match {
    case Some(rec) => Some(rec.request)
    case None => None
  }
}
def extractUriFromRequest(requestField: String) = requestField.split(" ")(1)
log.filter(line => getStatusCode(p.parseRecord(line)) == "404").map(getRequest(_)).count
val recs = log.filter(line => getStatusCode(p.parseRecord(line)) == "404").map(getRequest(_))
val distinctRecs = log.filter(line => getStatusCode(p.parseRecord(line)) == "404")
                      .map(getRequest(_))
                      .collect { case Some(requestField) => requestField }
                      .map(extractUriFromRequest(_))
                      .distinct
distinctRecs.count
distinctRecs.foreach(println)
```

Lesson 3 : Hadoop Data Processing

Automating the Data Processing Step using ODI12c

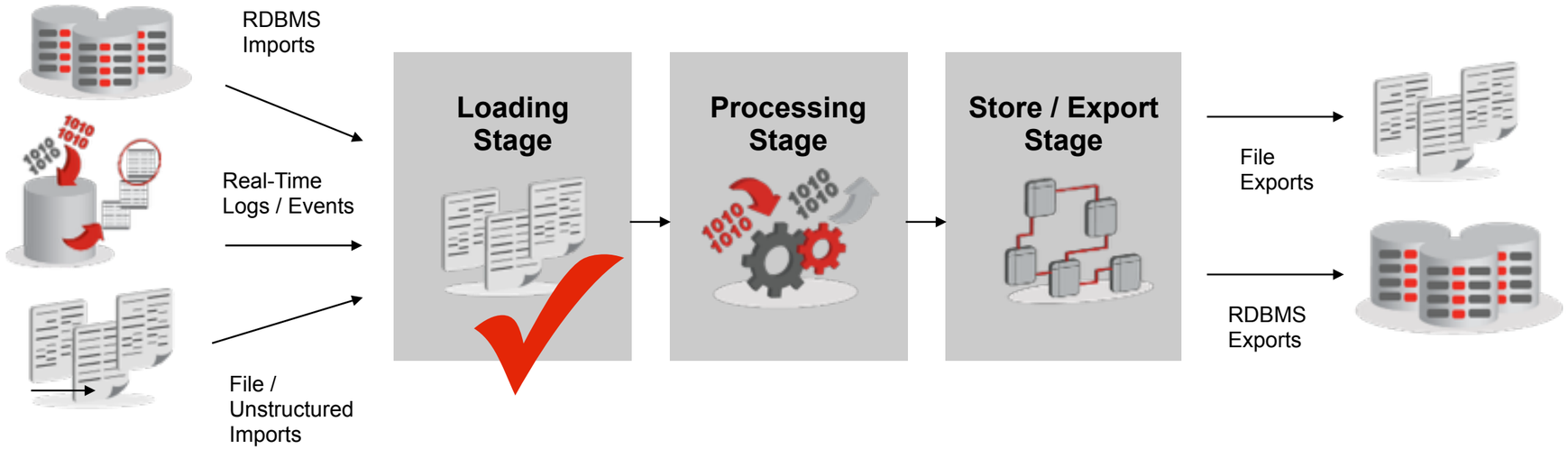
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

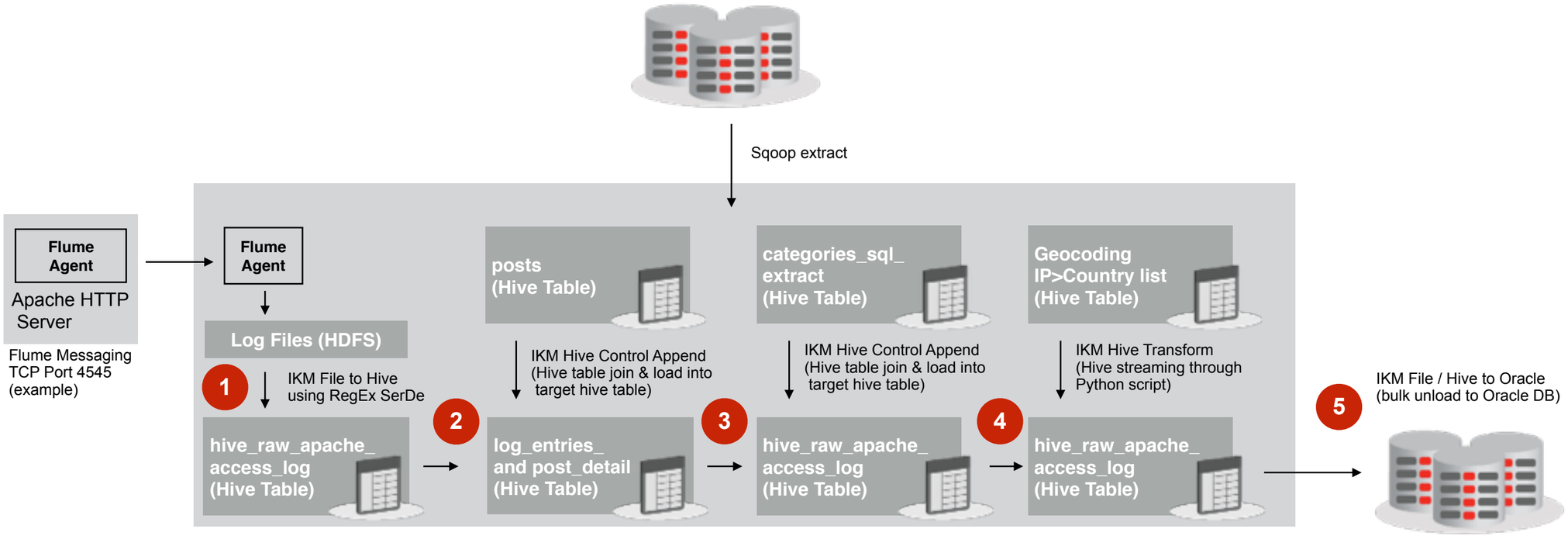
Current Status of Web Server Log Analysis Scenario

- We've now landed log activity from the Rittman Mead website into Hadoop, using Flume
- Data arrives as Apache Webserver log files, is then loaded into a Hive table and parsed
- Supplemented by social media activity (Twitter) accessed through a MongoDB database
- Now we can start processing, analysing, supplementing and working with the dataset…



Planned ETL & Data Flow through BDA System

- Five-step process to load, transform, aggregate and filter incoming log data
- Leverage ODI's capabilities where possible
- Make use of Hadoop power + scalability



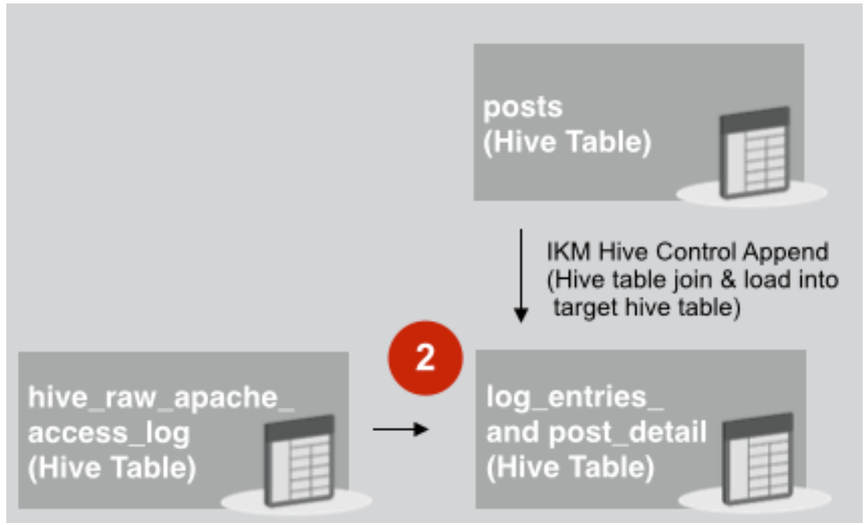
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead INTEGRATED ANALYTICS

2 Join to Additional Hive Tables, Transform using HiveQL

- IKM Hive to Hive Control Append can be used to perform Hive table joins, filtering, agg. etc.
- INSERT only, no DELETE, UPDATE etc
- Not all ODI12c mapping operators supported, but basic functionality works OK
- Use this KM to join to other Hive tables, adding more details on post, title etc
- Perform DISTINCT on join output, load into summary Hive table



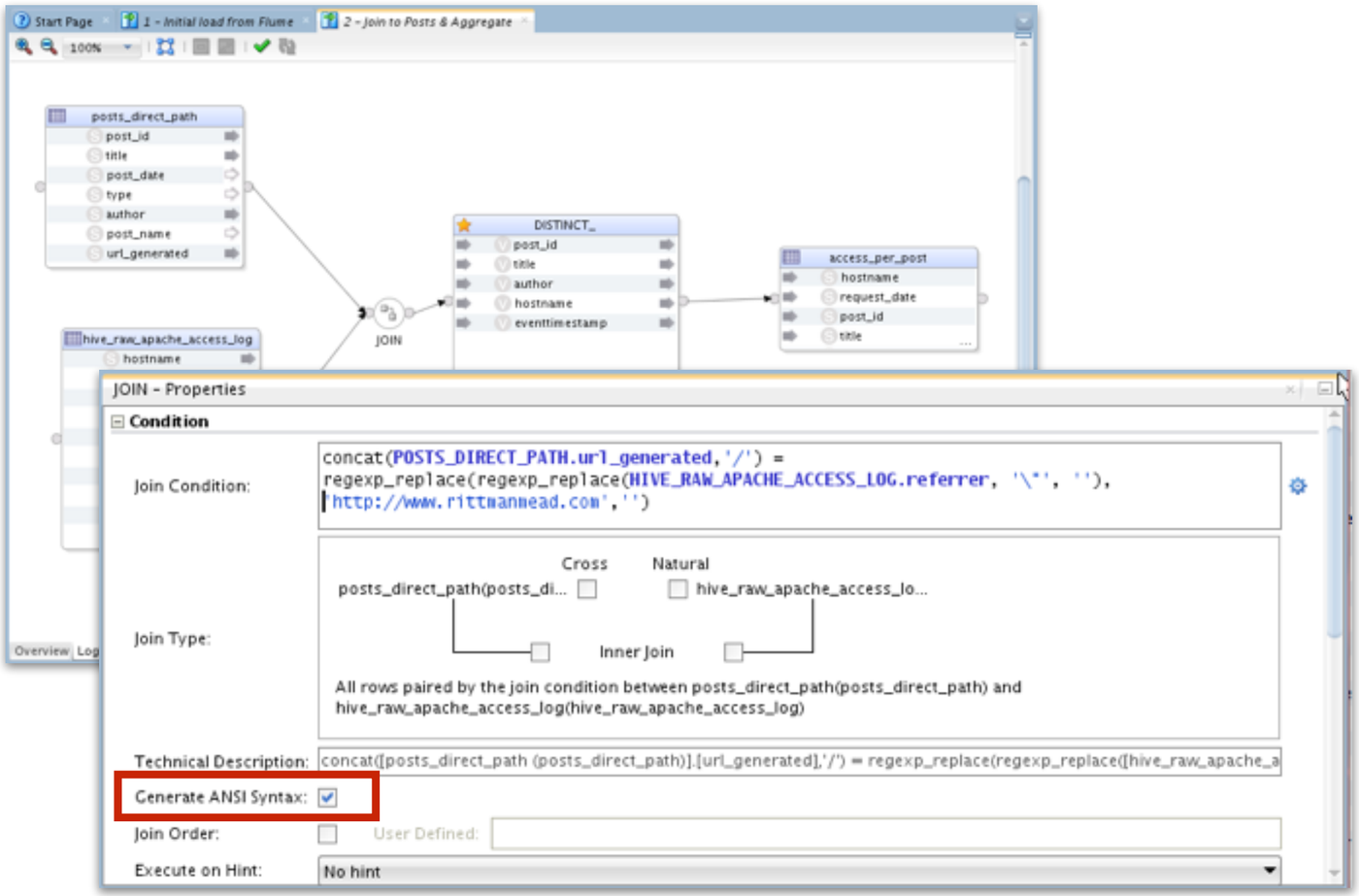
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Joining Hive Tables

- Only equi-joins supported
- Must use ANSI syntax
- More complex joins may not produce valid HiveQL (subqueries etc)



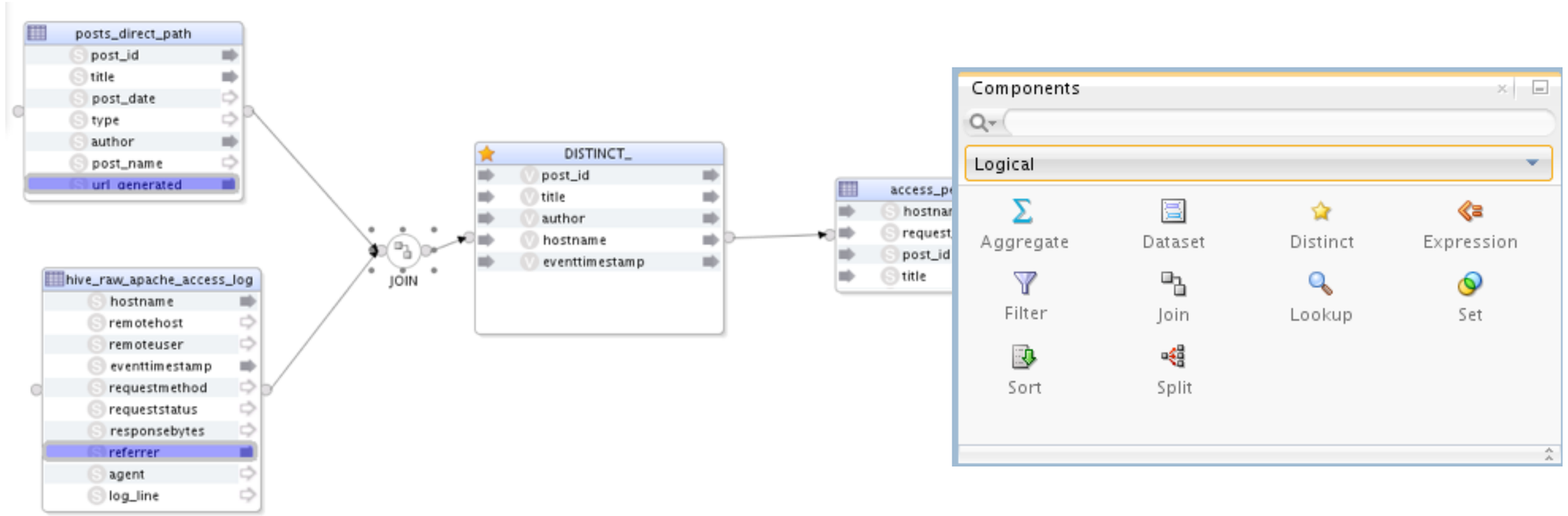
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

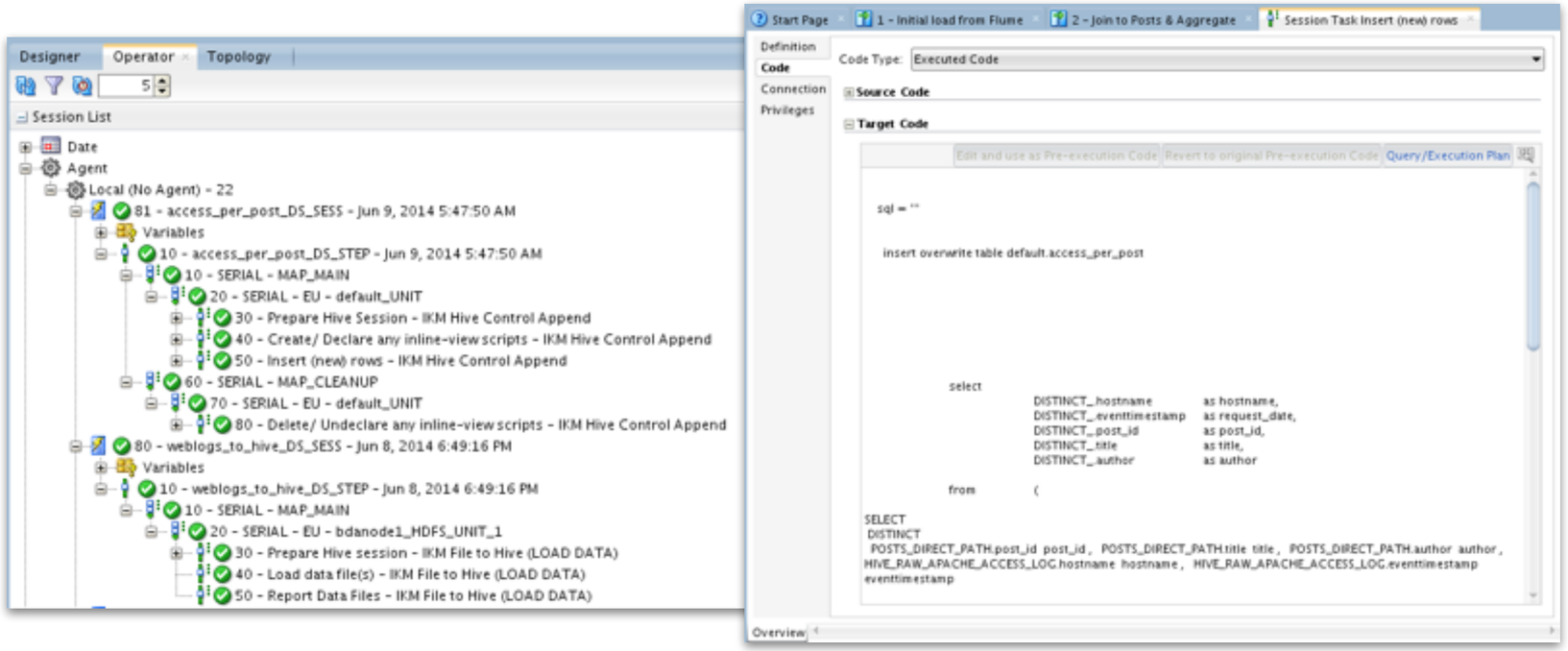
Filtering, Aggregating and Transforming Within Hive

- Aggregate (GROUP BY), DISTINCT, FILTER, EXPRESSION, JOIN, SORT etc mapping operators can be added to mapping to manipulate data
- Generates HiveQL functions, clauses etc



Executing Second Mapping

- ODI IKM Hive to Hive Control Append generates HiveQL to perform data loading
- In the background, Hive on BDA creates MapReduce job(s) to load and transform HDFS data
  - Automatically runs across the cluster, in parallel and with fault tolerance, HA



T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Demo

Joining Datasets in Hive using ODI12c

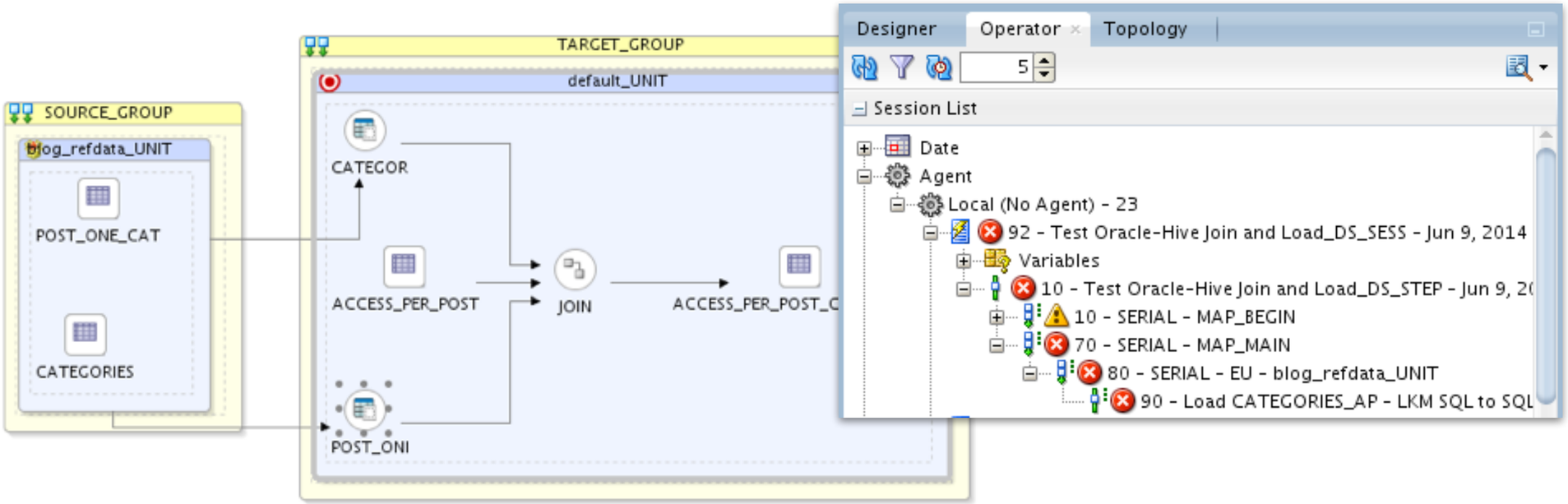
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

3 Bring in Reference Data from Oracle Database

- In this third step, additional reference data from Oracle Database needs to be added
- In theory, should be able to add Oracle-sourced datastores to mapping and join as usual
- But … Oracle / JDBC-generic LKMs don't get work with Hive

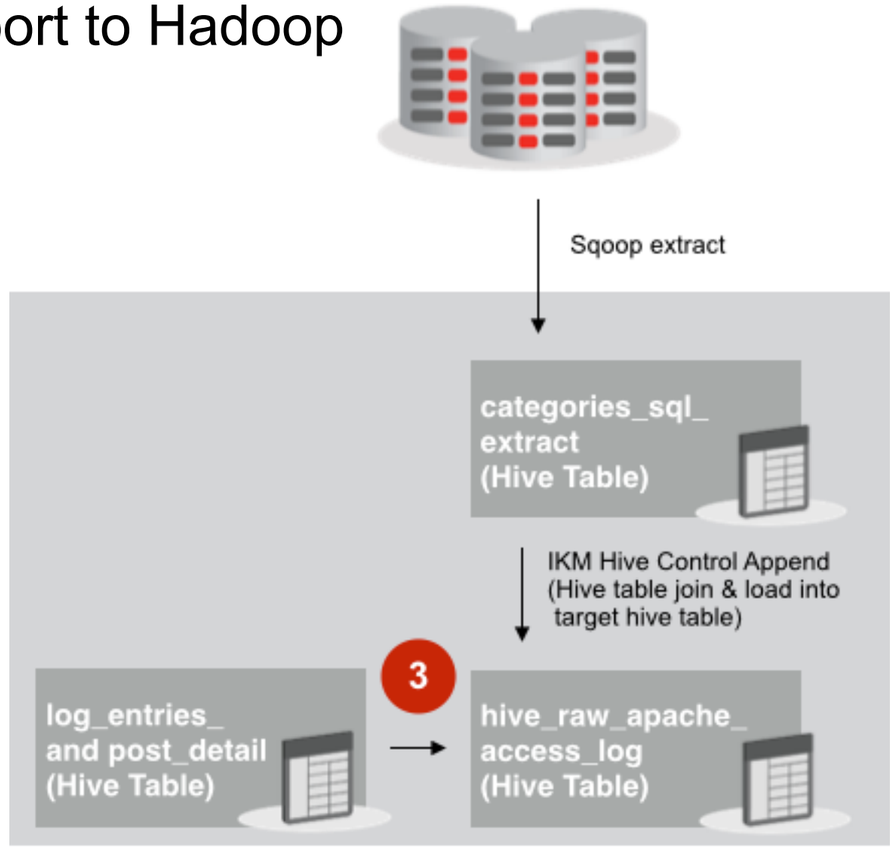


T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Options for Importing Oracle / RDBMS Data into Hadoop

- Could export RBDMS data to file, and load using IKM File to Hive
- Oracle Big Data Connectors only export to Oracle, not import to Hadoop
- Best option is to use Apache Sqoop, and new IKM SQL to Hive-HBase-File knowledge module
  - Hadoop-native, automatically runs in parallel
  - Uses native JDBC drivers, or OraOop (for example)
  - Bi-directional in-and-out of Hadoop to RDBMS
  - Run from OS command-line



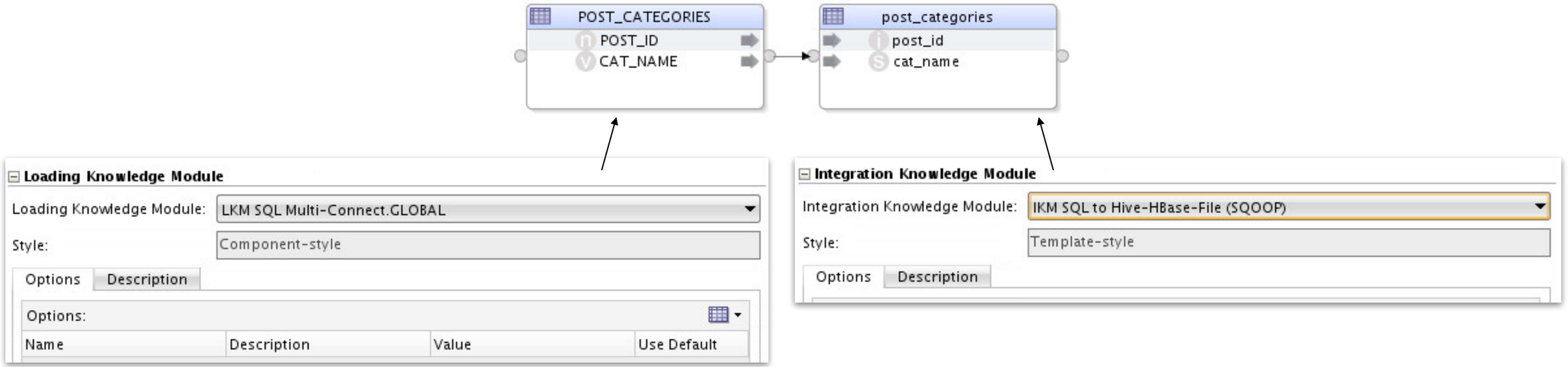
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

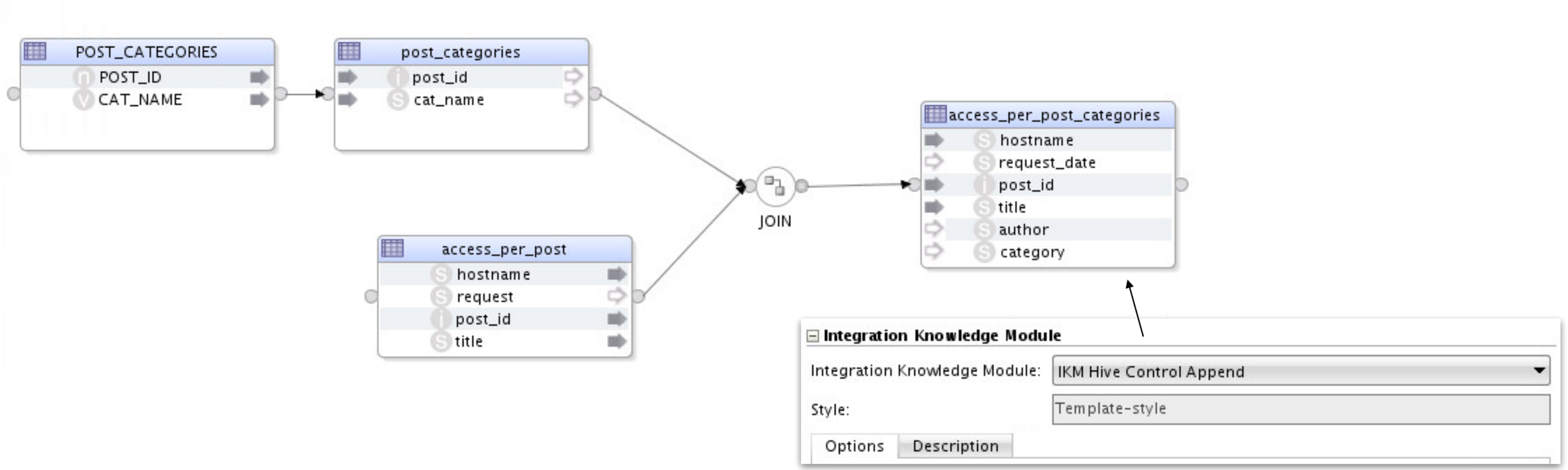
Loading RDBMS Data into Hive using Sqoop

- First step is to stage Oracle data into equivalent Hive table
- Use special LKM SQL Multi-Connect Global load knowledge module for Oracle source
  - Passes responsibility for load (extract) to following IKM
- Then use IKM SQL to Hive-HBase-File (Sqoop) to load the Hive table



Join Oracle-Sourced Hive Table to Existing Hive Table

- Oracle-sourced reference data in Hive can then be joined to existing Hive table as normal
- Filters, aggregation operators etc can be added to mapping if required
- Use IKM Hive Control Append as integration KM



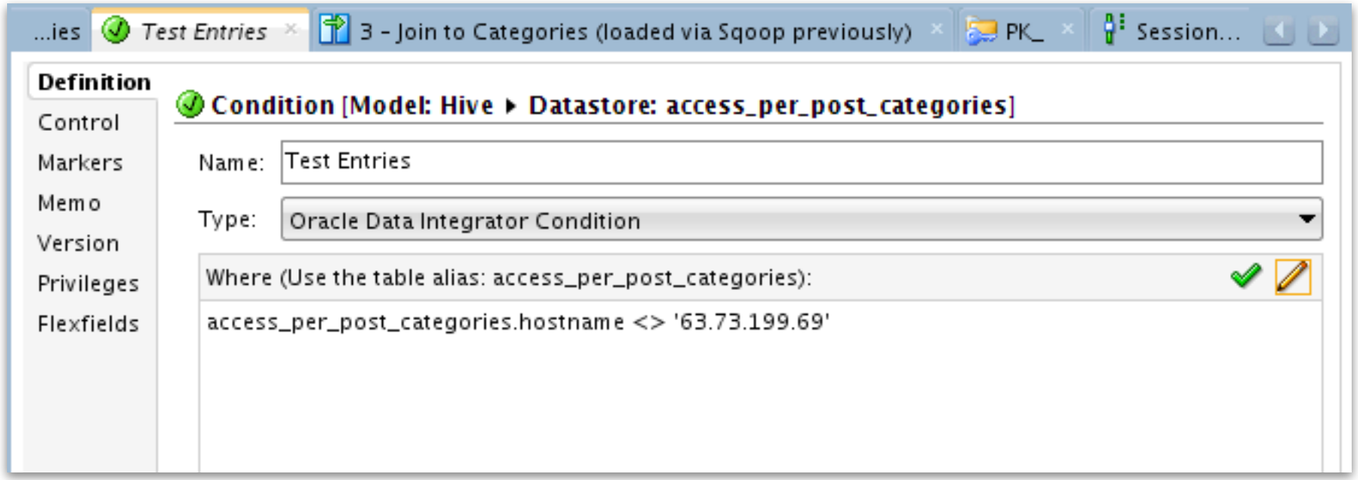
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

ODI Static and Flow Control : Data Quality and Error Handling

- CKM Hive can be used with IKM Hive to Hive Control Append to filter out erroneous data
- Static controls can be used to create “data firewalls”
- Flow control used in Physical mapping view to handle errors, exceptions
- **Example**: Filter out rows where IP address is from a test harness

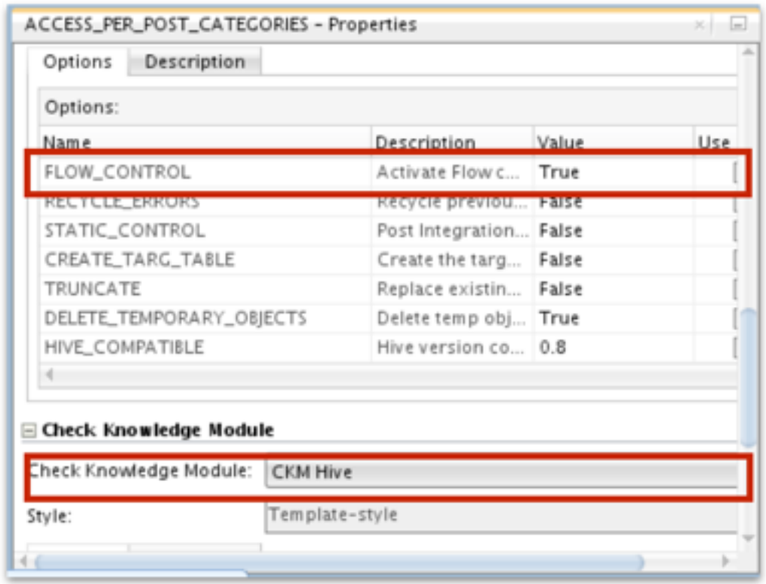


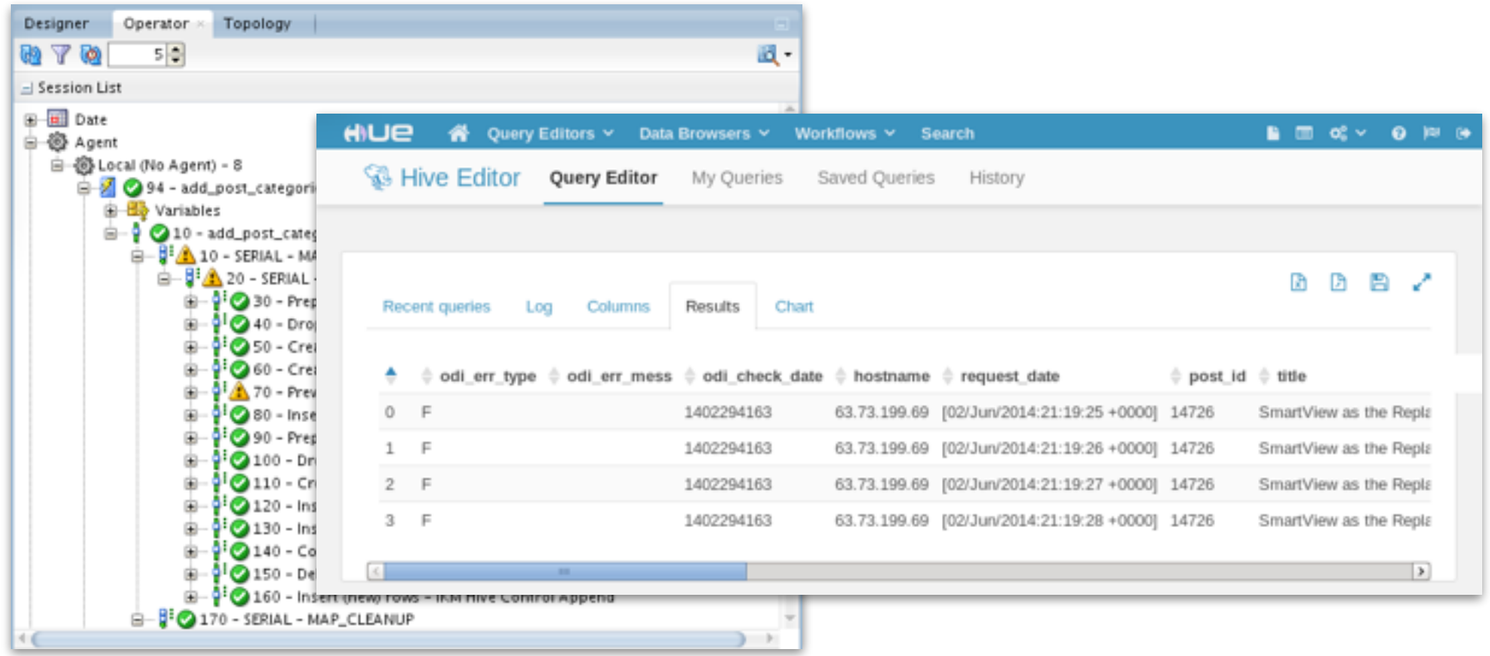
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Enabling Flow Control in IKM Hive to Hive Control Append

- Check the ENABLE_FLOW_CONTROL option in KM settings
- Select CKM Hive as the check knowledge module
- Erroneous rows will get moved to E_ table in Hive, not loaded into target Hive table





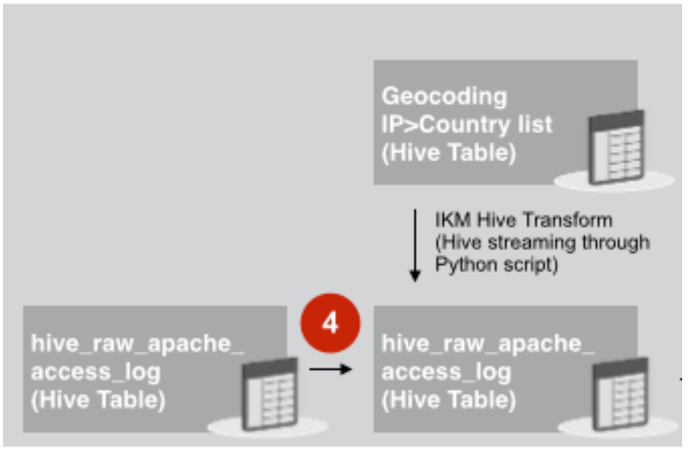
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

4 Using Hive Streaming and Python for Geocoding Data

- Another requirement we have is to “geocode” the webserver log entries
- Allows us to aggregate page views by country
- Based on the fact that IP ranges can usually be attributed to specific countries
- Not functionality normally found in Hive etc, but can be done with add-on APIs



T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

How GeoIP Geocoding Works

- Uses free Geocoding API and database from Maxmind
  - Convert IP address to an integer
  - Find which integer range our IP address sits within
  - But Hive can’t use BETWEEN in a join…

```
address = '174.36.207.186'

( o1, o2, o3, o4 ) = address.split('.')

integer_ip =   ( 16777216 * o1 )
             + (    65536 * o2 )
             + (      256 * o3 )
             +              o4
```

```
SELECT ip_country
FROM geoip
WHERE
2921648058 BETWEEN begin_ip_num AND end_ip_num
LIMIT 1
```

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

Solution : IKM Hive Transform

- IKM Hive Transform can pass the output of a Hive SELECT statement through a perl, python, shell etc script to transform content
- Uses Hive TRANSFORM … USING … AS functionality

```
hive> add file file:///tmp/add_countries.py;

Added resource: file:///tmp/add_countries.py

hive> select transform (hostname,request_date,post_id,title,author,category)

    > using 'add_countries.py'

    > as (hostname,request_date,post_id,title,author,category,country)

    > from access_per_post_categories;
```

Creating the Python Script for Hive Streaming

- Solution requires a Python API to be installed on all Hadoop nodes, along with geocode DB

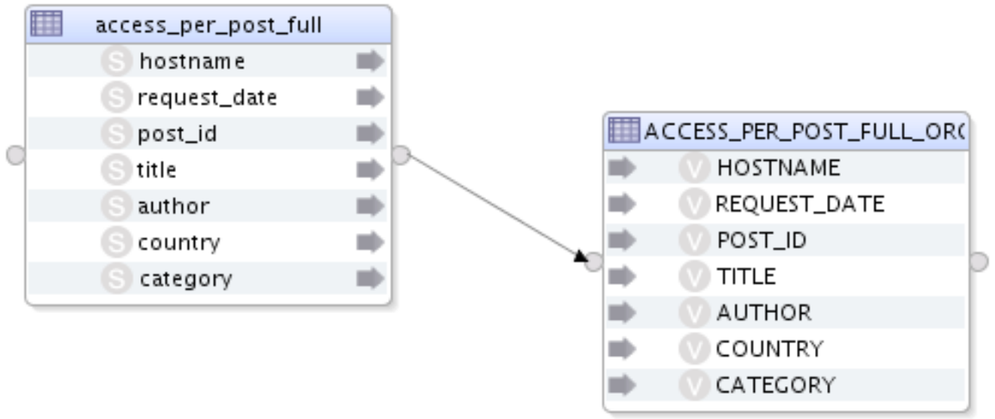
```
wget https://raw.github.com/pypa/pip/master/contrib/get-pip.py
python get-pip.py pip
install pygeoip
```

- Python script then parses incoming stdin lines using tab-separation of fields, outputs same (but with extra field for the country)

```
#!/usr/bin/python
import sys
sys.path.append('/usr/lib/python2.6/site-packages/')
import pygeoip
gi = pygeoip.GeoIP('/tmp/GeoIP.dat')
for line in sys.stdin:
  line = line.rstrip()
  hostname,request_date,post_id,title,author,category = line.split('\t')
  country = gi.country_name_by_addr(hostname)
  print hostname+'\t'+request_date+'\t'+post_id+'\t'+title+'\t'+author
+'\t'+country+'\t'+category
```

Setting up the Mapping

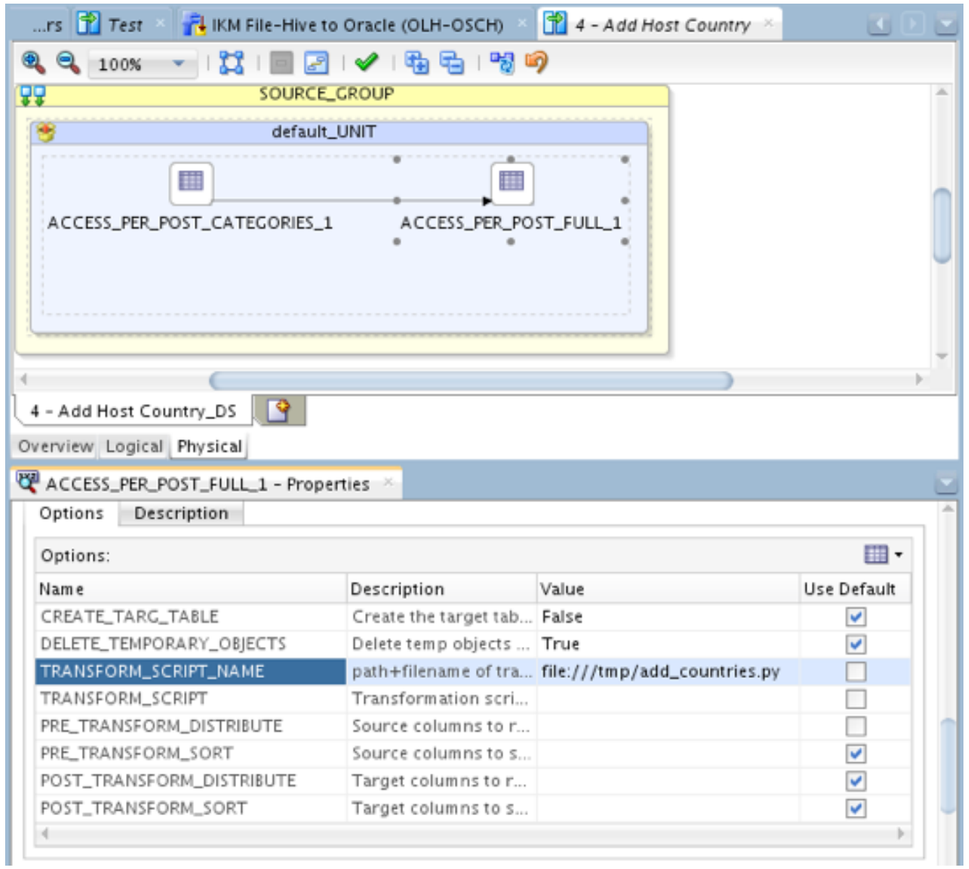
- Map source Hive table to target, which includes column for extra “country” column



- Copy script + GeoIP.dat file to every node’s /tmp directory
- Ensure all Python APIs and libraries are installed on each Hadoop node

Configuring IKM Hive Transform

- TRANSFORM_SCRIPT_NAME specifies name of script, and path to script
  - TRANSFORM_SCRIPT has issues with parsing; do not use, leave blank and KM will use existing one
- Optional ability to specify sort and distribution columns (can be compound)
- Leave other options at default



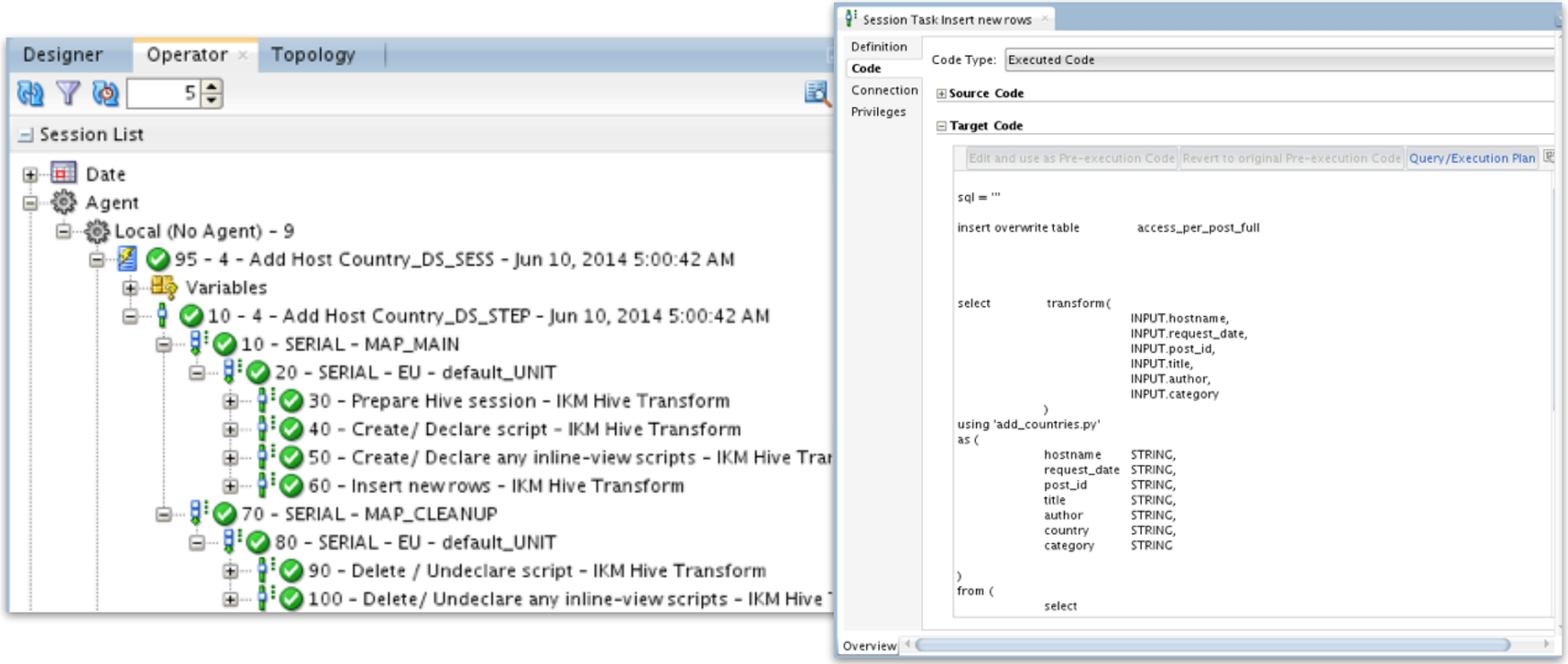
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Executing the Mapping

- KM automatically registers the script with Hive (which caches it on all nodes)
- HiveQL output then runs the contents of the first Hive table through the script, outputting results to target table



T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Demo

Using Hive Streaming in ODI12c

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

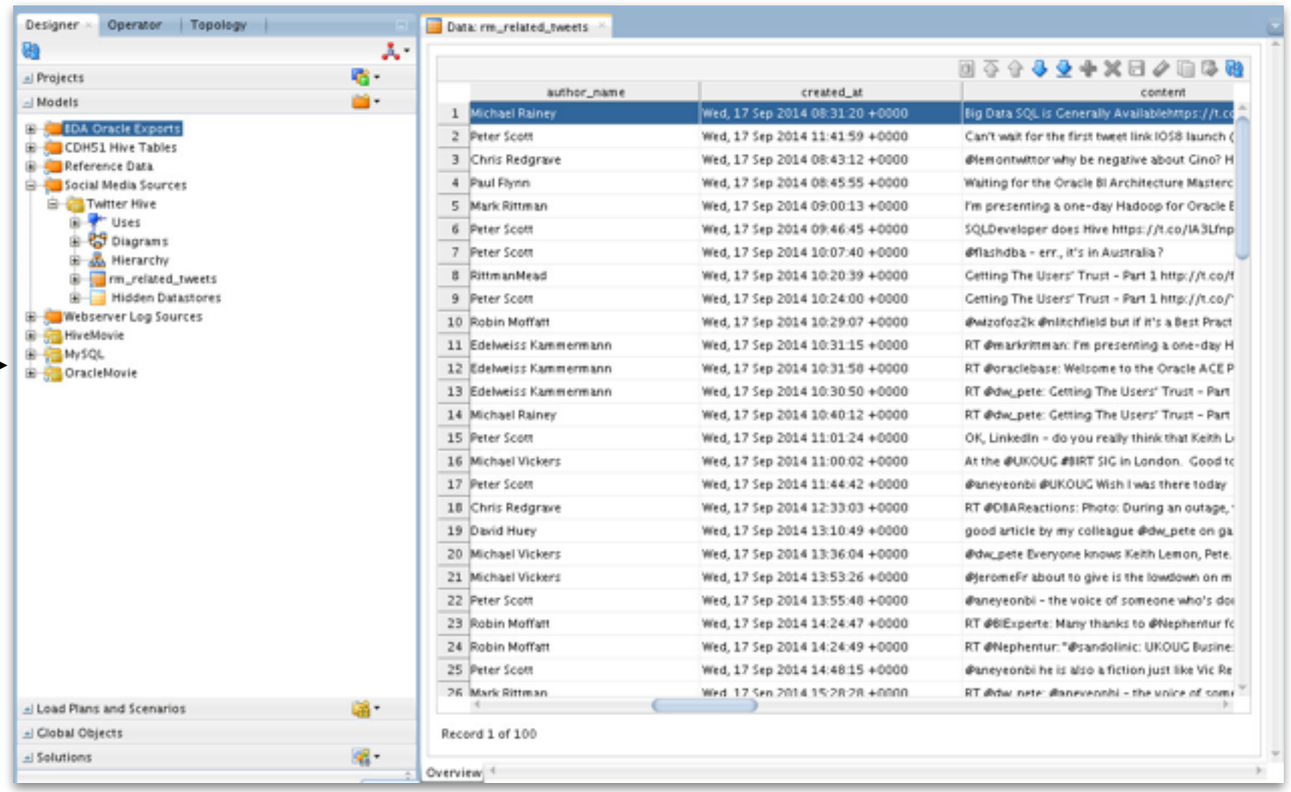
E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Adding the Twitter Data from MongoDB

- Previous steps exposed the Twitter data, in MongoDB, through a Hive table
- RM_RELATED_TWEETS Hive table now included in ODI Topology + Model

```
hive> describe rm_related_tweets;
OK
id                                  string        from deserializer
interactionid                       string        from deserializer
username                            string        from deserializer
author_name                         string        from deserializer
created_at                          string        from deserializer
content                             string        from deserializer
twitter_link                        string        from deserializer
language                            string        from deserializer
sentiment                           int           from deserializer
author_tweet_count                  int           from deserializer
author_followers_co                 int           from deserializer
author_profile_img_u                string        from deserializer
author_timezone                     string        from deserializer
normalized_url                      string        from deserializer
mentions                            string        from deserializer
hashtags                            string        from deserializer
Time taken: 0.109 seconds, Fetched: 16 row(s)
```



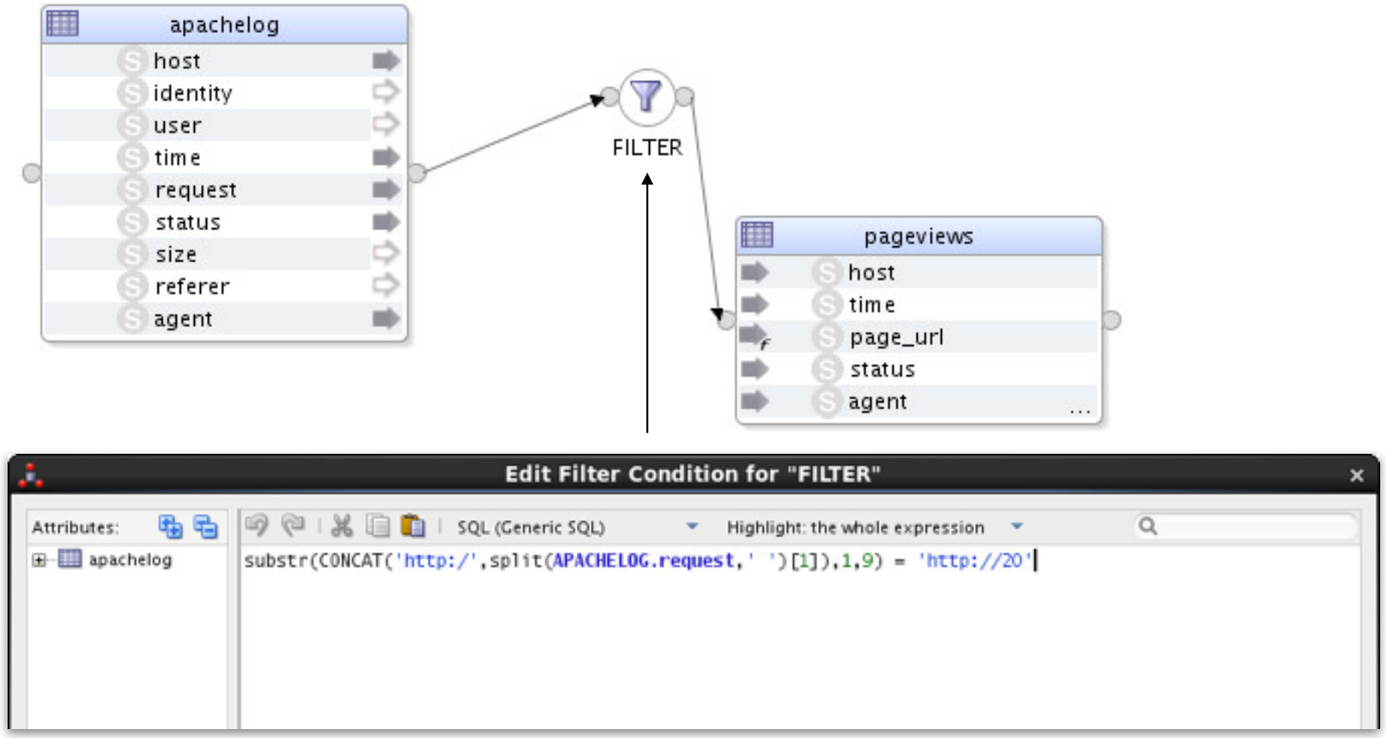
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Filter Log Entries to Only Leave Blog Post Views

- We're only interested in Twitter activity around blog posts
- Create an additional Hive table to contain just blog post views, to then join to tweets



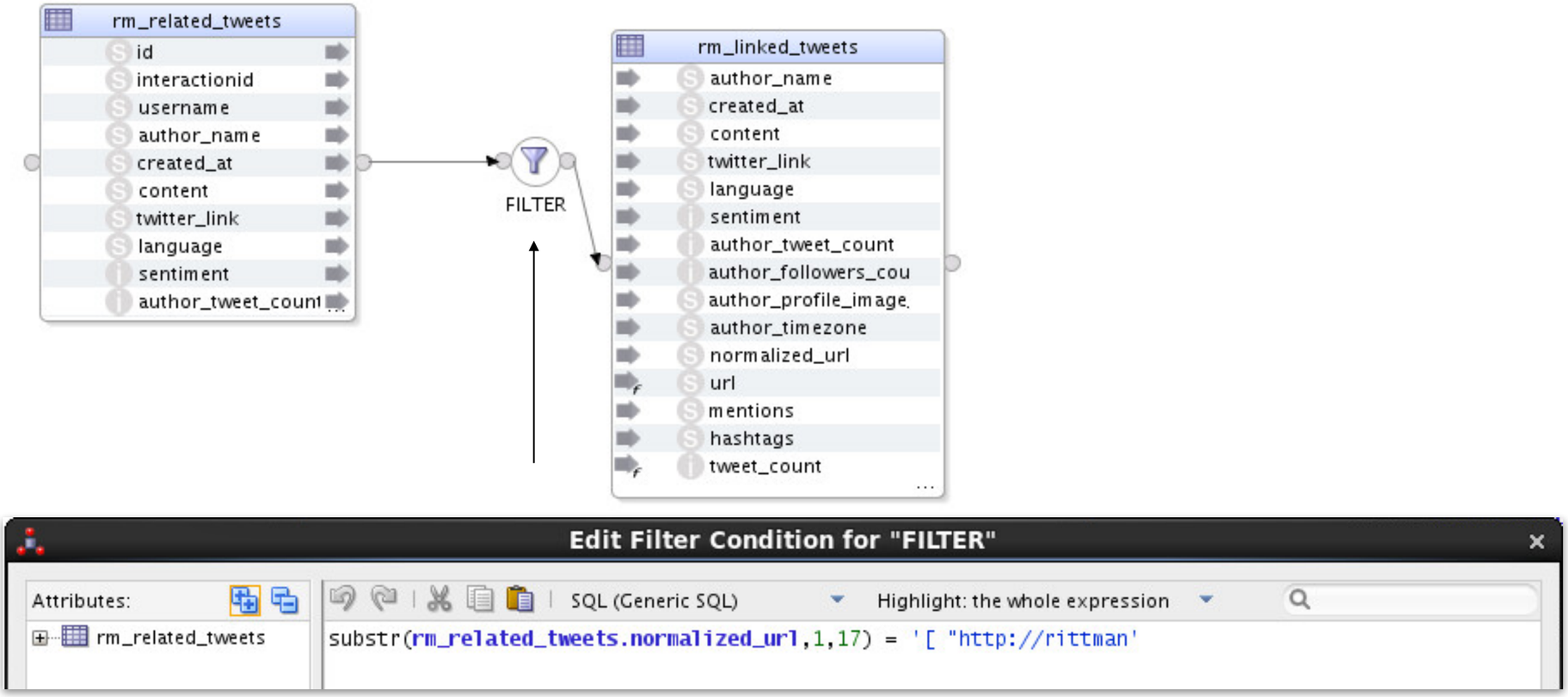
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

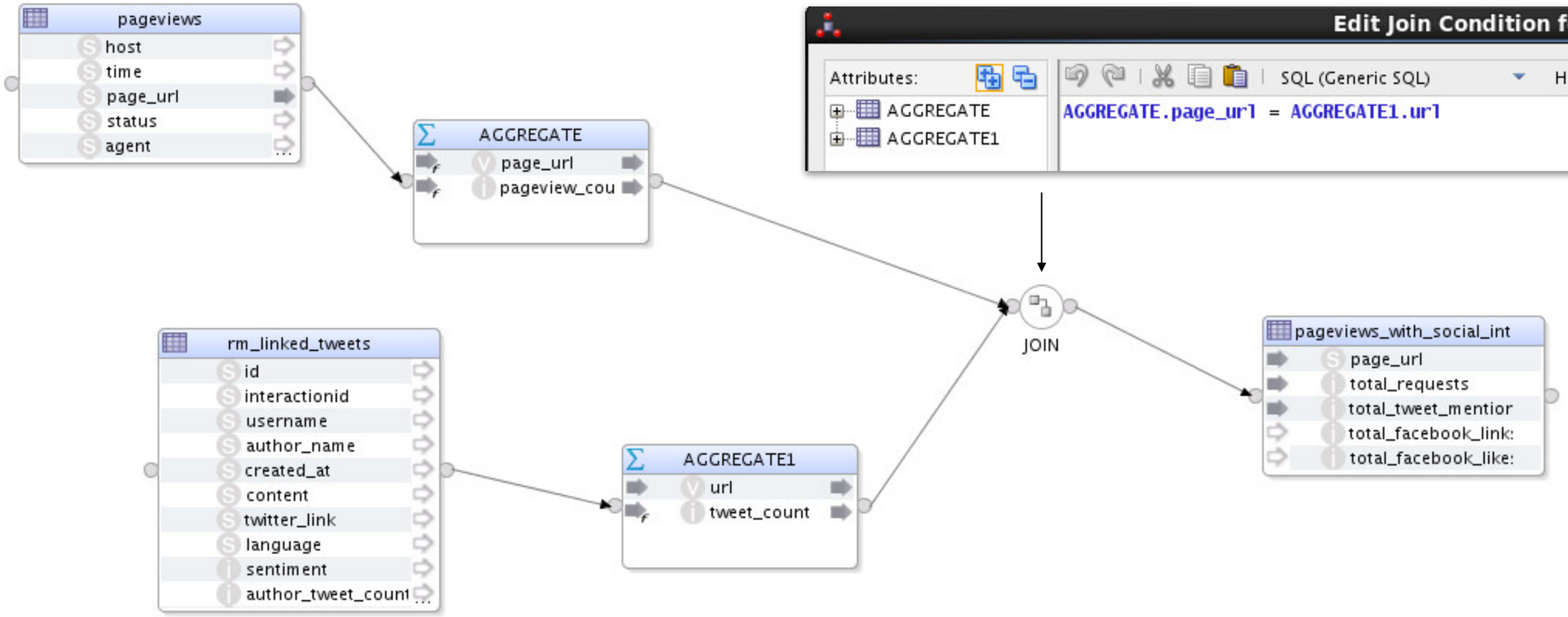
Filter Tweets Down to Just Those Linking to RM Website

- Filter the list of tweets down to just those that link to RM blog



Join Twitter Data to Page View Data

- Create summary table showing twitter activity per page on the blog



T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Demo

Joining Twitter Data to Log Data in ODI12c

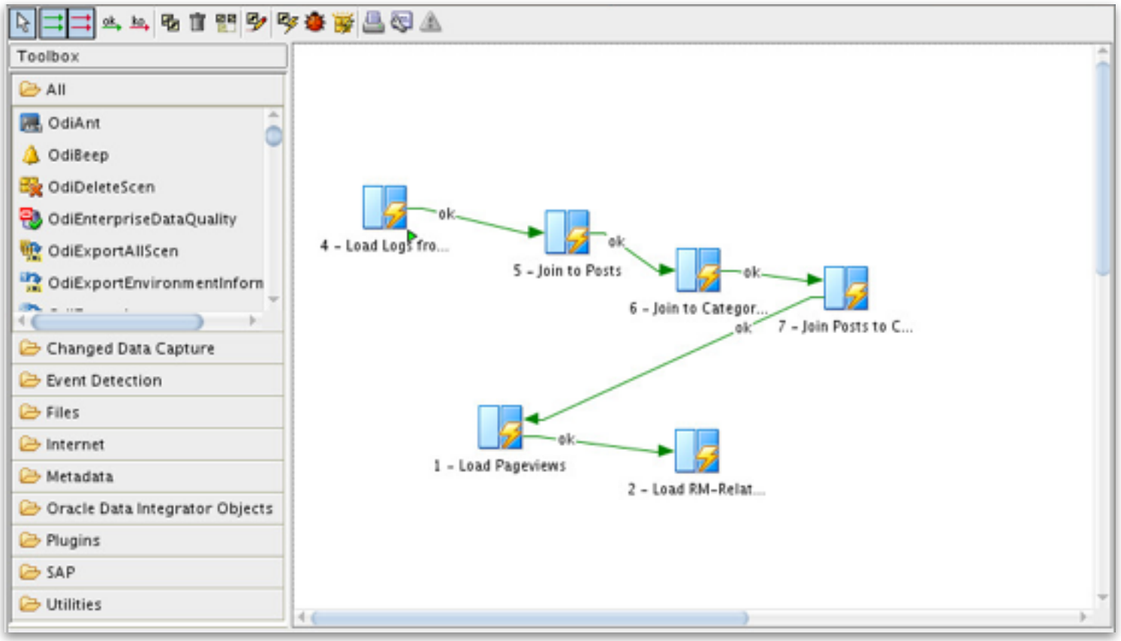
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

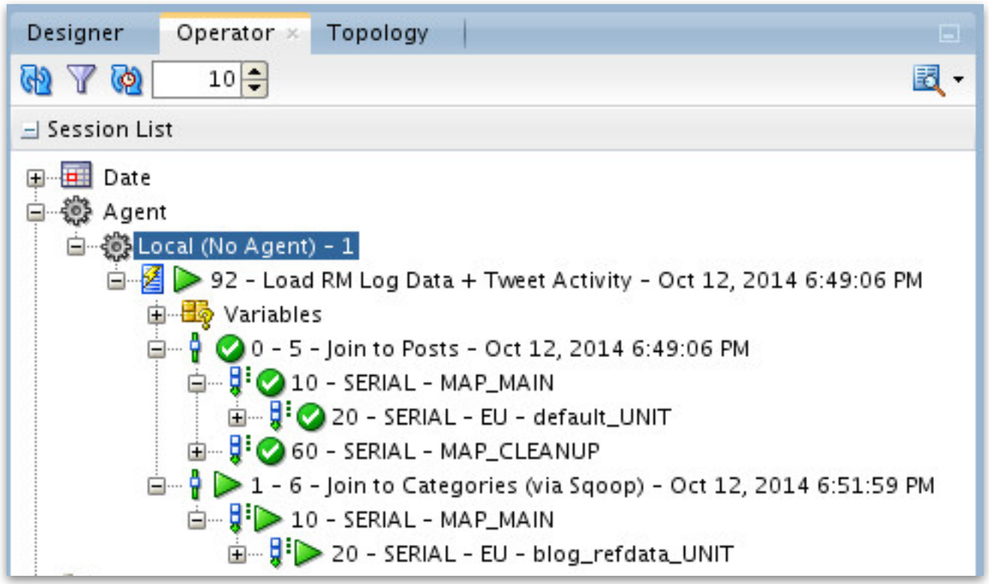
E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Create ODI Package for Processing Steps, and Execute

- Create ODI Package or Load Plan to run steps in sequence
  - With load plan, can also add exceptions and recoverability
- Execute package to load data into final Hive tables





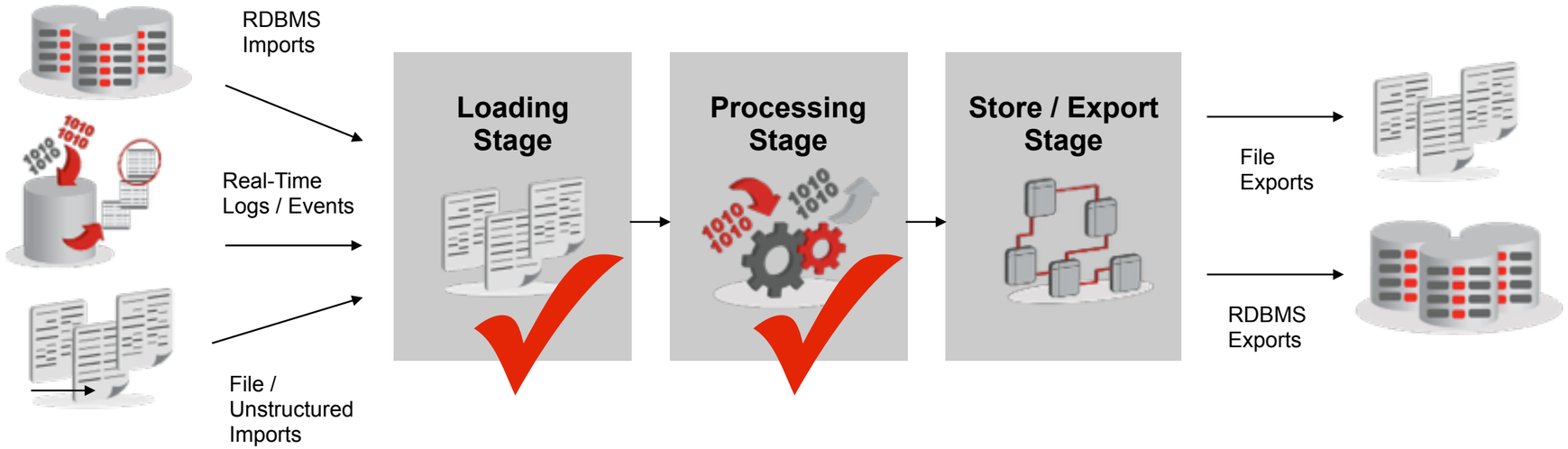
T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS

Summary : Data Processing Phase

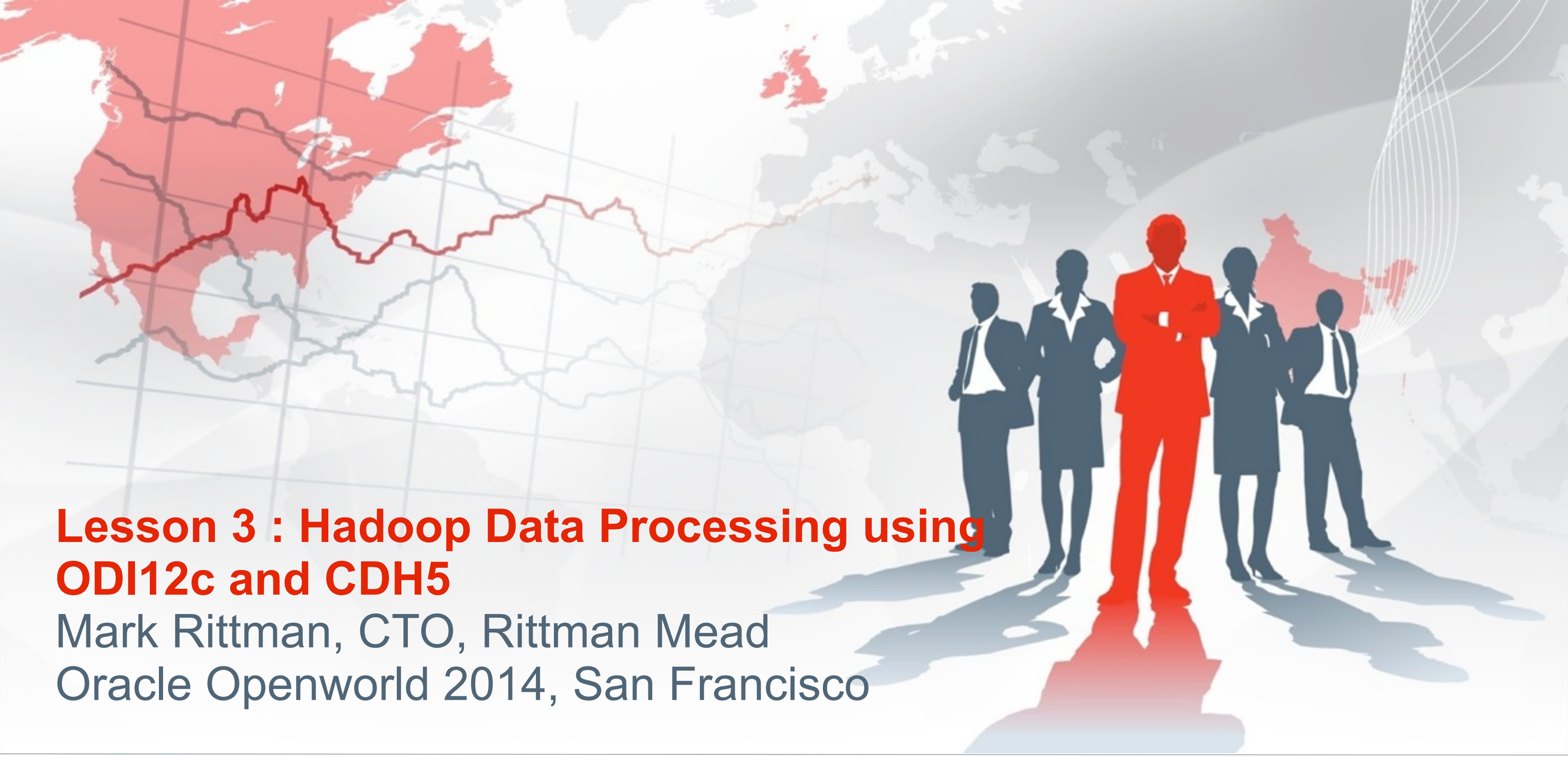
- We've now processed the incoming data, filtering it and transforming to required state
- Joined ("mashed-up") datasets from website activity, and social media mentions
- Discovery phase and the load/processing stages are now complete
- Now we want to make the Hadoop output available to a wider, non-technical audience…



T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead INTEGRATED ANALYTICS



Lesson 3 : Hadoop Data Processing using ODI12c and CDH5

Mark Rittman, CTO, Rittman Mead

Oracle Openworld 2014, San Francisco

T : +44 (0) 1273 911 268 (UK) **or** (888) 631-1410 (USA) **or**
+61 3 9596 7186 (Australia & New Zealand) **or** +91 997 256 7970 (India)

E : info@rittmanmead.com
W : www.rittmanmead.com

rittmanmead
INTEGRATED ANALYTICS