



JCC LOGMINER LOADER AND JAVA 8

Keith W. Hare

Tom Musson

Cheryl Jalbert

JCC Consulting, Inc.

October 29, 2019



Introduction

- Introduction to JCC LogMiner Loader
- Java 8 on OpenVMS
- Why support Java 8?
 - New JDBC Drivers
 - Kafka
- Summary

Introduction to JCC LogMiner Loader

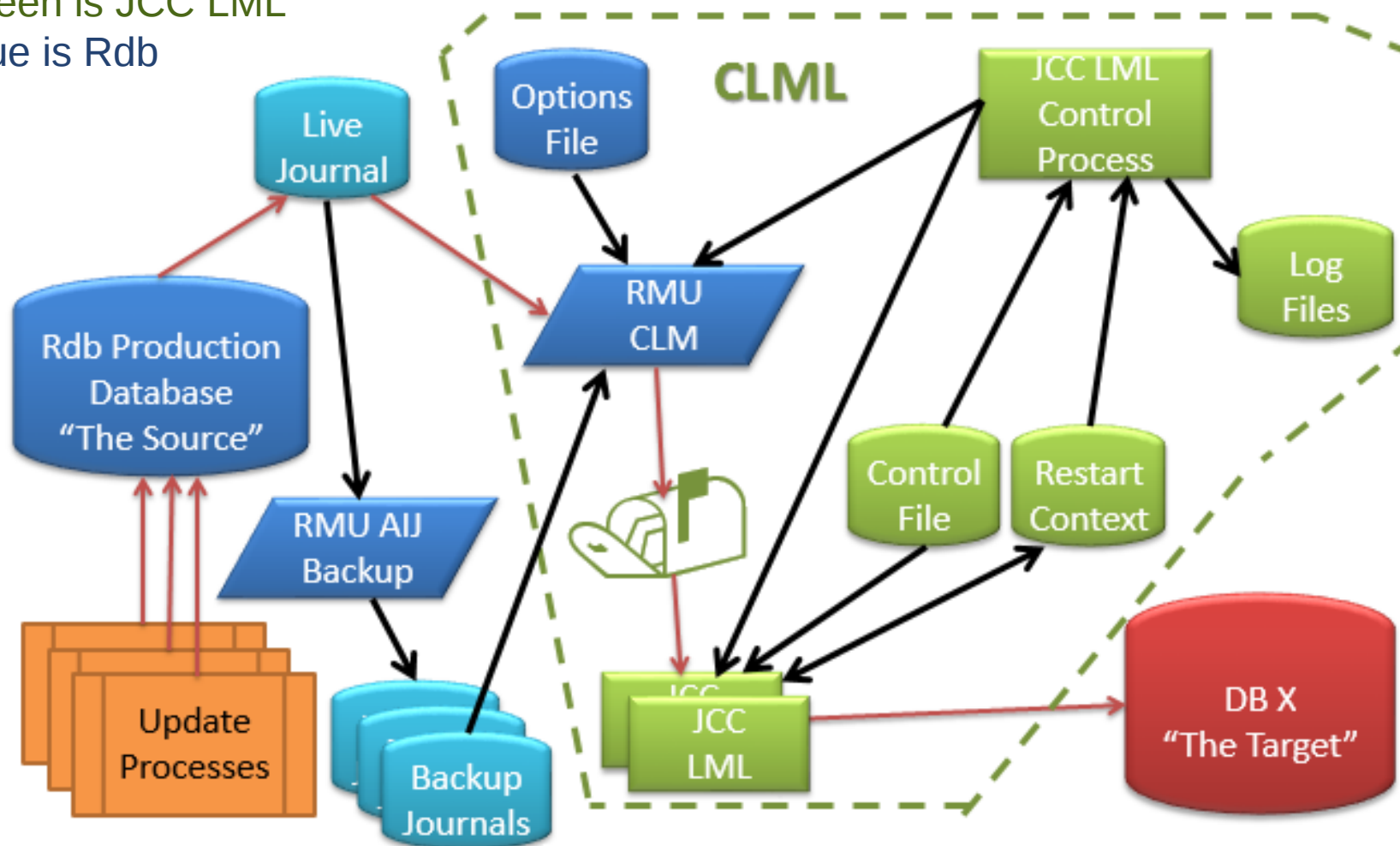
Replicates committed transactions from a source to a target

- Source
 - Oracle Rdb (any version that supports the LogMiner)
- Target
 - Oracle Rdb (any version that supports multi-statement procedures)
 - Oracle (requires SQL*net client on the system running the Loader)
 - JDBC Class 4 driver
 - Tuxedo
 - XML (to your own API)
 - File
 - Kafka (JCC LogMiner Loader V3.6)

Replicating to a Database Target

Green is JCC LML

Blue is Rdb





Java 8

- Java 8 available for OpenVMS V8.4 on Itanium, not on Alpha
- 64 bit pointers
 - Java 6 uses 32 bit pointers
- Java 8 support added in JCC LogMiner Loader V3.5.1



JCC LogMiner Loader Changes for Java 8

Some changes were needed to support Java 8

- Modify code to be aware of pointer length
- Transition 32-bit pointers to 64-bit pointers where/when needed
- Add second build of same code with 64-bit pointers
- Add second link of 64-bit object into 64-bit image
- Modify environment to use 64-bit images when user requests Java 8



Why support Java 8?

- New JDBC drivers
- Kafka support



New JDBC Drivers for Java 8

Most (all?) database targets have JDBC Drivers for Java 8. For example

- MySQL – mysql-connector-java-8.0.11.jar
- SQL Server
 - jtds-1.3.1.jar
 - mssql-jdbc-6.2.0.jre8.jar
- Postgres – postgresql.42.2.2.jar
- Etc.

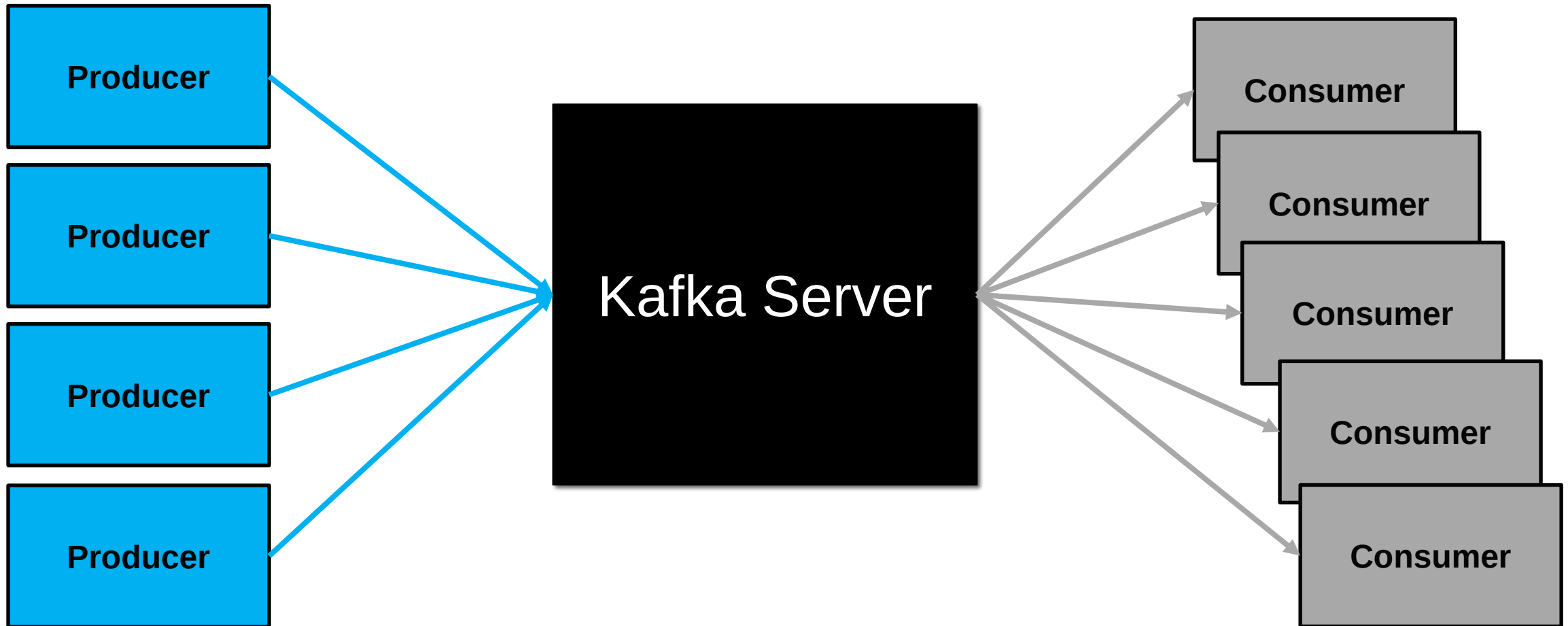
Kafka

- Kafka is a Reliable Messaging Service
- Similar function as
 - MQ Series
 - DEC Reliable Transaction Router
- Open Source
 - Apache Software Foundation – Apache 2 License
- Originally created by LinkedIn
- Ongoing development, new releases quarterly
- Written in Java – requires Java 7 or later

Kafka Pieces

- Kafka Server
 - Does the hard work
 - Can be clustered
 - Topics can be partitioned and replicated
- Message Producers
 - Publish messages to Topics on a Kafka Server
 - Multiple producers can publish to the same topic
 - Each producer can publish to multiple topics
- Message Consumers
 - Read Messages from Topics on a Kafka Server
 - Multiple consumers can read from the same topic
 - Each consumer can read from multiple topics
 - Parallelism – If multiple consumers specify the same group, they will get different data

The Big Picture





Why Use Kafka

Make information available to other applications

- Integrate data from multiple silos
- Downstream data analysis
- Populate a data lake
- Drive a dashboard
- Etc...

Kafka Producer and Consumer Example

Kafka Producer Console

```
$ java -cp /confluent_kafka_path/*:. -  
_$ kafka.tools.ConsoleProducer -  
_$ --broker-list confluent01.jcc.com:9093 -  
_$ --topic Test.Topic -  
_$ --producer.config KAFKA_AVRO.PROPERTIES  
>this is a test message  
>Second test message  
>Another arbitrary message  
>
```

Kafka Consumer Console

```
$ java -cp /confluent_kafka_path/*:. -  
_$ kafka.tools.ConsoleConsumer -  
_$ --bootstrap-server confluent01.jcc.com:9093 -  
_$ --topic Test.Topic -  
_$ --consumer.config KAFKA_AVRO.PROPERTIES  
this is a test message  
Second test message  
Another arbitrary message
```

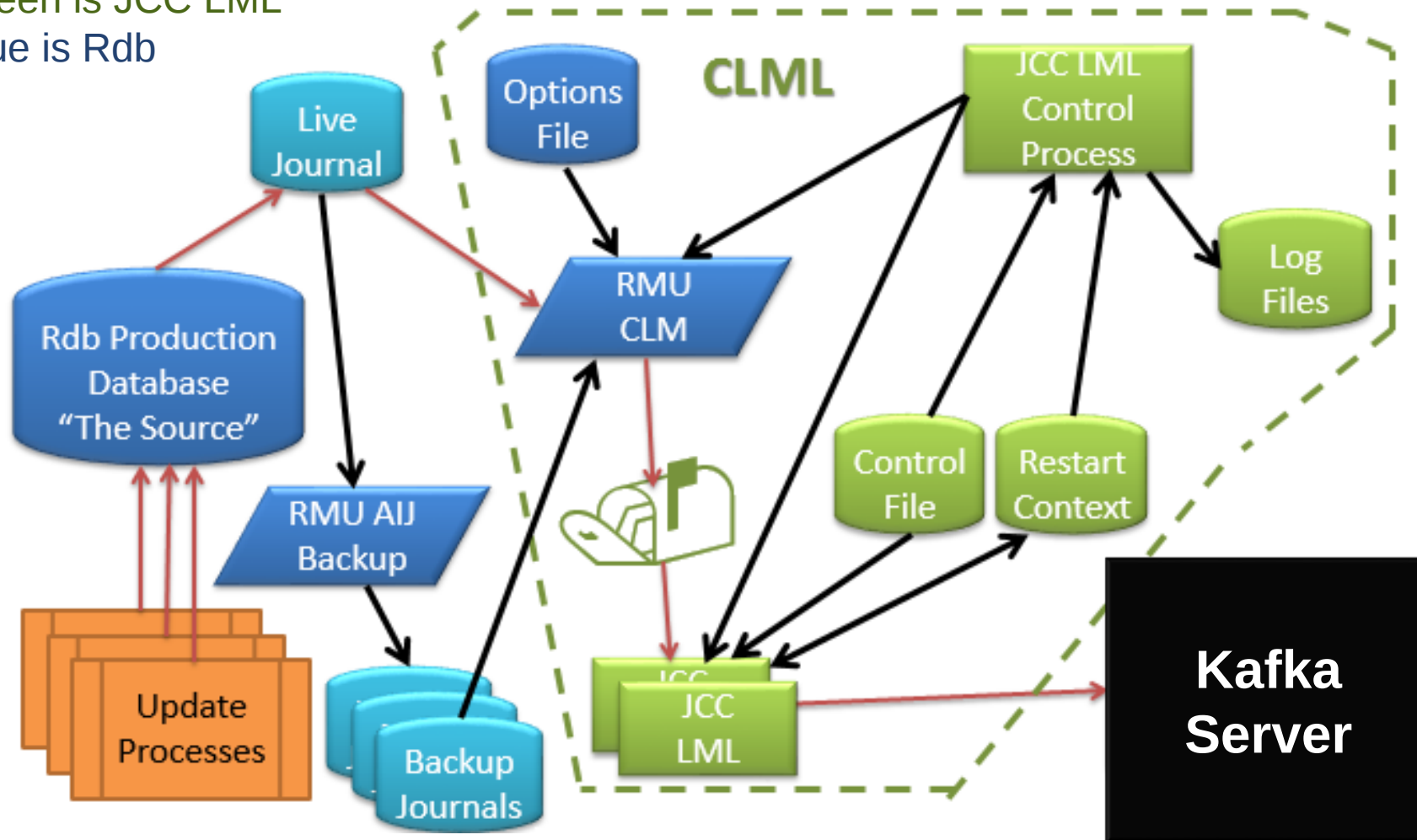
Kafka Distributions

- Apache Kafka Project
 - <https://kafka.apache.org/downloads>
- Confluent
 - <https://www.confluent.io/download/plans>
 - Supported version
 - Community version
- These distributions are available for Linux or Windows

Replicating to Kafka

Green is JCC LML

Blue is Rdb



Replicating to Kafka

- JCC LogMiner Loader Kafka Option publishes committed transactions to a Kafka Server
- Message Formats
 - XML
 - JSON
 - Avro with schema registration
- Some new configuration options specific to Kafka

Prerequisite: Java 8

- JCC LogMiner Loader Kafka Option uses Kafka Libraries
 - Copy to OpenVMS and reset file characteristics
- Kafka libraries require Java 7 or later
- Java 7 is not available on OpenVMS
- Java 8 is available on OpenVMS IPF (not available on Alpha)
- Java can be the installed JDK or a separate JRE



Kafka Example – Avro

```
!  
! Define the Kafka Avro Specific stuff  
!  
output~kafka~synch~connect~record~Avro  
kafka~connect~cnflnt4-04:9093,cnflnt4-03:9093,cnflnt4-02:9093  
kafka~avro~SchemaRegistry~http://cnflnt4-02:8081  
kafka~avro~namespace~MFP.avro  
!  
! routine to expand wildcards does not work on unix-style names.  
!  
kafka~classpath~disk$static:[kafka.CONFLUENT-4-1-1.kafka-serde-tools]*.jar  
kafka~classpath~disk$static:[kafka.CONFLUENT-4-1-1.confluent-common]*.jar  
kafka~classpath~disk$static:[kafka.CONFLUENT-4-1-1.kafka]*.jar  
!  
kafka~Topic~'Personnel.',Table_Name  
kafka~Model~ExactlyOnce  
!  
! External Properties File  
!  
java~properties~/control_files/kafka_avro.properties  
!  
! The remainder of this file is standard Loader "stuff"  
!
```



Kafka Example – Output

- Output target is Kafka with a record format of Avro
output~kafka~synch~connect~record~Avro
- List of Kafka nodes and ports
kafka~connect~cnf1nt4-04:9093,cnf1nt4-03:9093,cnf1nt4-02:9093
 - Three nodes in this Kafka cluster – supports failover
 - Port 9093 is SSL port
 - Kafka nodes and port numbers can be configured in Kafka server

Kafka Example – Schema Registry

- Schema Registry Service for Avro

kafka~avro~SchemaRegistry~http://cnflnt4-02:8081

- Registry traffic is not encrypted – HTTP
- Port number is 8081
- HTTP or HTTPS and port number can be configured in Schema Registry service

- Namespace used for Schema Registry

kafka~avro~namespace~MFP.avro

- Namespace is an arbitrary string
- Namespace should be unique across publishers
- Negotiate with Kafka consumer team

Kafka Example – Topic and Model

- Topic Naming

kafka~Topic~'Personnel.', Table_Name

- Table_Name – wild card for source table names
- Kafka Topics will be Personnel.EMPLOYEES, Personnel.SALARY_HISTORY, etc.

- Kafka Model

kafka~Model~ExactlyOnce

- JCC LogMiner Loader Kafka Option supports:
 - Flush – all rows in a Loader checkpoint flushed and verified as received before the Loader thread updates it's high-water checkpoint data.
 - Transaction
 - Default
 - Use Kafka start/commit/rollback transactions
 - ExactlyOnce – Use Kafka ExactlyOnce semantics

Kafka Example – External Properties File

- Properties file contents passed to Java Engine
`java~properties~/control_files/kafka_avro.properties`
- Control_files is an OpenVMS logical name
`$ define control_files JCC_ROOT:[KEITH.SQL_CLASS.LML_KAFKA]`
- kafka_avro.properties contains:
`# kafka_avro.properties
security.protocol=ssl
ssl.truststore.location=/mfp_ssl/kafka_truststore.jks
ssl.truststore.password=<password>
ssl.keystore.location=/mfp_ssl/kafka_keystore.jks
ssl.keystore.password=<password>
ssl.key.password=<password>`
- Mfp_ssl is an OpenVMS logical name
`$ define mfp_ssl JCC_ROOT:[KEITH.SQL_CLASS.LML_KAFKA.ssl]`

Kafka Example – Rest of the Configuration File

- Remainder of the JCC LogMiner Loader configuration file identical to any other target
 - Table
 - MapTable
 - Etc.

Avro and Schema Registration

- Record descriptions registered with Confluent Schema Registration Service
- Messages
 - Compressed (serialized) when published
 - Decompressed (de-serialized) when consumed
- Revised record description gets new registration
 - Consumer gets new de-serialized record
 - Kafka consumers can programmatically handle metadata changes

Kafka Partitions

- Kafka supports partitioning topics for performance
- JCC LogMiner Loader is aware of partitions
- Distributes data across partitions based on hash of DBKey
 - All Messages for a particular DBKey will be written to the same partition
 - Updates of the same row in consecutive transactions will be processed in the correct order
 - Rows from the same source transaction might be written to different partitions

JCC LogMiner Loader Kafka Option Summary

- LML configuration for Kafka has some new syntax
 - Output Record Avro, JSON, or XML
 - Connect syntax can contain multiple nodes in a cluster
 - Schema Registry server
 - Need classpath to Kafka jar files
 - Kafka Topic naming
 - Kafka Model
- Most of configuration identical to other targets
- Operation identical to other targets

Kafka Configuration Summary

- Kafka is fairly easy to configure and manage
 - Once you find the correct white papers and blogs
 - Once you figure out the parts and terminology
- JCC LogMiner Loader Kafka Option includes only a producer
 - Does not include Kafka Consumer
 - Existing consumers – e.g. Oracle
 - `ORA_KAFKA.load_table(view_on_kafka_topic, database_table)`
 - We can help evaluate and design Kafka consumer and architecture



Summary – JCC LogMiner Loader and Java 8

- New JDBC Drivers
- Kafka
 - Kafka provides a reliable message queuing service
 - JCC LogMiner Loader Kafka Option supports publishing committed Rdb transactions to a Kafka server
 - Data published to Kafka is available to variety of downstream tools
 - Data can be published in formats:
 - XML
 - JSON
 - Avro

Learning More

- The product is The JCC LogMiner Loader.
 - Read about it at <http://www.jcc.com/lml>
 - Check out the blogs <http://www.jcc.com/lml-blog>
 - Ask for a temporary license
- Contact us at Info@JCC.com



Join the Conversation

Join the worldwide Rdb community. Send mail to
OracleRdb-request@JCC.com.

Include “SUBSCRIBE” in
the body of the message.

