

ORACLE + KAFKA BUILD ETL PIPELINES WITH KAFKA CONNECTORS

geeksinsight

Learnings

- Graduate in Commerce
- Started with Career in
 Foxpro/Sybase/Access/Oracle
- Oracle certified professional in 2000, started with v7
- ITIL Certified, AWS Certified
- Trainer

Blog & Around the community

- Started blogging around 2012
- Made good friends around the globe
- Learnt a lot from fellow DBA's and Internet
- Happy to learn , keen to share ☺
- Technology Director, Independent Australia Oracle Group

Companies worked

18+ Years of Experience

- Prokarma Senior DBA
- Oracle Senior Technical Analyst
- HSBC- Manager/Consultant , Global
 Production Support
- IBM Database Architect
- Open Universities Australia, Lead DevOps
- Qualcomm, Senior Manager, Database Team

My Blog





ABOUT ME

AGENDA

Oracle Database – No need of introduction

Kafka – Introduction

Kafka – Use cases

Kafka – Connectors to get data in and out from external systems

• JDBC, HDFS, S3, Elasticsearch, File-source etc.

Kafka – Connectors options

Partitioners

Schema Evolution

Kafka – Test case – Scalable ETL solution with CDC using connectors

•Extract Data from Oracle using JDBC driver

•Register the schema in Hive – Using Hive schematool

- •Build Hadoop HDFS to store the extracted data in hdfs partitions Using Hadoop with Hive
- •Change Data Capture Using HDFS Sink Connector
- •Use KSQL to aggregate the topic data and submit to another topic Using KSQL streams topic
- •Send aggregated data to S3 Using S3 Sink Connector



STREAMING DATA PIPELINE -CHALLENGE

Real Time Processing for stream of data from various sources with having different structures of data.

The traditional message queues cannot process or transform the data on fly and storing back the data to multiple sources involved various development processes etc.







Kafka Connect is a framework for large scale, real-time stream data integration using Kafka. It abstracts away the common problems every connector to Kafka needs to solve: schema management, fault tolerance, partitioning, offset management and delivery semantics, operations, and monitoring.

GEEKSNSIGHT

Kafka Streams Platform offers two things:-

Data integration: A stream data platform captures streams of events or data changes and feeds them to other data systems such as relational databases, keyvalue stores, Hadoop, or the data warehouse.

Stream processing: It enables continuous, real-time processing and transformation of these exact same streams and makes the results available system-wide.



If a stream represents a database, then a stream partition would represent a table in the database. Likewise, if a stream represents an HBase cluster, for example, then a stream partition would represent a specific HBase region.

KAFKA DATA PIPELINES ARCHITECTURE

Architecture



Image Source: Internet

KAFKA CONNECT

Features

Framework for Connectors



•An operating-system process (<u>Java</u>-based) which executes connectors and their associated tasks in child threads, is what we call a **Kafka Connect worker**.

•Also, there is an object that defines parameters for one or more tasks which should actually do the work of importing or exporting data, is what we call a **connector**.

•To read from some arbitrary input and write to Kafka, a **source connector** generates tasks.

•In order to read from Kafka and write to some arbitrary output, a **sink** connector generates tasks.

KAFKA CONNECT



Image Souce: Confluent

KAFKA CONNECT



KAFKA CONNECT FOR STREAMING ETL



SOURCE & SINK cassandro afka Source JDBC Connectors for Databases Sink Connectors for target data sources

Source Connectors

For Example, The JDBC source connector allows you to import data from any relational database with a JDBC driver into Kafka topics. By using JDBC, this connector can support a wide variety of databases without requiring custom code for each one.

Data is loaded by periodically executing a SQL query and creating an output record for each row in the result set. By default, all tables in a database are copied, each to its own output topic. The database is monitored for new or deleted tables and adapts automatically. When copying data from a table, the connector can load only new or modified rows by specifying which columns should be used to detect new or modified data.

Sink Connectors

For example: The JDBC sink connector allows you to export data from Kafka topics to any relational database with a JDBC driver. By using JDBC, this connector can support a wide variety of databases without requiring a dedicated connector for each one. The connector polls data from Kafka to write to the database based on the topics subscription. It is possible to achieve idempotent writes with upserts. Auto-creation of tables, and limited autoevolution is also supported.

BUILT IN TRANSFORMATIONS

Example Transformation for File Sink Connector

name=local-file-source connector.class=FileStreamSource tasks.max=1

file=test.txt

topic=connect-test

transforms=**MakeMap,InsertSource** transforms.**MakeMap.type=org.apache.kafka.connect.transforms.HoistField\$Value**

transforms.MakeMap.field=line

transforms.InsertSource.type=org.apache.kafka.connect.transforms.InsertField\$Value transforms.InsertSource.static.field=data_source transforms.InsertSource.static.value=test-file-source InsertField – Add a field using either static data or record metadata

• ReplaceField – Filter or rename fields

- MaskField Replace field with valid null value for the type (0, empty string, etc)
- ValueToKey Set the key to one of the value's fields
- HoistField Wrap the entire event as a single field inside a Struct or a Map

• ExtractField – Extract a specific field from Struct and Map and include only this field in results • SetSchemaMetadata – modify the schema name or version

• TimestampRouter – Modify the topic of a record based on original topic and timestamp. Useful when using a sink that needs to write to different tables or indexes based on timestamps

• RegexpRouter – modify the topic of a record based on original topic, replacement string and a regular expression



KAFKA STREAMS FOR COMPLEX AGGREGATIONS OR EVENT MAPPINGS ON TOP OF TOPICS

Kafka Streams

- Complex transformations including
- Aggregations
- Windowing
- Joins
- Transformed data stored back in Kafka, enabling reuse
- Write, deploy, and monitor a Java application

TRANSFORM SINGLE MESSAGE TRANSFORMATIONS

Modify events before storing in Kafka:

- Mask sensitive information
- Add identifiers
- Tag events
- Store lineage
- Remove unnecessary columns

Modify events going out of Kafka:

- Route high priority events to faster data stores
- Direct events to different Elasticsearch indexes
- Cast data types to match destination
- Remove unnecessary columns

BATCH ETL TO UPDATE A MASK A COLUMN









KAFKA CONNECTORS FEATURES

Exactly Once Delivery: The connector uses a write ahead log to ensure each record exports to HDFS exactly once. Also, the connector manages offsets commit by encoding the Kafka offset information into the file so that the we can start from the last committed offsets in case of failures and task restarts.

Extensible Data Format: Out of the box, the connector supports writing data to HDFS in Avro and Parquet format. Also, you can write other formats to HDFS by extending the Format class.

Hive Integration: The connector supports Hive integration out of the box, and when it is enabled, the connector automatically creates a Hive external partitioned table for each topic exported to HDFS.

Schema Evolution: The connector supports schema evolution and different schema compatibility levels. When the connector observes a schema change, it projects to the proper schema according to the schema.compatibility configuration. Hive integration is supported if BACKWARD, FORWARD and FULL is specified for schema.compatibility and Hive tables have the table schema that are able to query the whole data under a topic written with different schemas.

Secure HDFS and Hive Metastore Support: The connector supports Kerberos authentication and thus works with secure HDFS and Hive metastore.

Pluggable Partitioner: The connector supports default partitioner, field partitioner, and time based partitioner including daily and hourly partitioner out of the box. You can implement your own partitioner by extending the Partitioner class. Plus, you can customize time based partitioner by extending the TimeBasedPartitioner class.

Modes : Incremental, Timestamp, Custom query, bulk to load data from source for capturing data changes



DEMO

Source : Oracle Database

Table: Users

Connector: JDBC

Mode: Incrementing+timestamp

Hive Integration : True Hive Metastore : True (in oracle schema as hive)

Kafka Topic: test_jdbc_users

Integrated with JDBC Source connector and to avro format and to HDFS Sink Connector

HDFS : Topic data created as hive data and to partition in hdfs /topic/test_jdbc_users

Compatibility: backward

Partitioner for HDFS Partition : department column

BUILD VIRTUAL BOX AND INSTALL Oracle 11g USING VAGRANT+ANSIBLE

DEMO

Command Transf. August 18	
>> oraclekafkaet1: Running 'pre-boot' VM customizations	
oraclekafkaetl: Naiting for machine to boot. This may take a few minutes oraclekafkaetl: SSH address: 127.0.0.1:2222 oraclekafkaetl: SSH username: vagrant oraclekafkaetl: SSH username: vagrant	
> oraclekafkaet1: Machine booted and ready!	
<pre>oraclekafkaetl; GuestAdditions 5.2.16 running OK. ers oraclekafkaetl; Checking for guest additions in VM</pre>	
++> uraclekafkaetl: Setting hostname	
<pre>> oraclekafkaetl: Mounting shared folders</pre>	
<pre>oraclekafkaet1: /vagrant *> Cf/Users/gandhi/Downloads/oraclekafkaet1 => oraclekafkaet1: Machine already provisioned. Run `vagrant provision` or use the `provision` => oraclekafkaet1: flag to force provisioning. Provisioners marked to run always will still run.</pre>	
C:\Users\gandhi\Downloads\oraclekafkaetl>vagrant ssh Last login: Tue Mar 12 08:12:28 2019 from 10.0.2.2 [vagrant@oraclekafkaetl -]\$ cd /vagrant/ [vagrant@oraclekafkaetl vagrant]\$ ansible-playbook oracle12.yml	
[vagrant@oraclekafkaet1 vagrant]\$ ansible-playbook pracle12c.yml [washizho]: wost file mot cound: /etc/ansible/hosts	
[WARREND]: provided houts list is empty, only locathoat is weatlable	
PLAy [localhost]	****
TASK [Gathering Facts]	

BUILD HADOOP, HIVE, KAFKA USING ANSIBLE



10. 10. Per	t par bridge press that has been bet been down a	
10.63.45	5.111. 关注: 14. 单位: 单位: 14.111. 24.11. 单位: 14.11.11.11.11.11.11.11.11.11.11.11.11.1	
Conception in a	A REPAIR OF A CONTRACT AND A REPAIR AND A CONTRACT OF A REPAIR OF	
43.10	- hase; marries and they the holders and have	
781.04	rillede	
2461	nd, inge Januar Gan Konstantan	
144.5	为我此一个想到后期,这年前的封闭上来。我,我一个知道上课外。	
1000	141-1504 August 141-111 August 141-111 August 141-111	
289.7	aralge sentitioned size-s.1.2-2.12.4AP.	
240	La 's ('opt/insele/oraleta/apade/is/apade/is/i/opt/integ	
148.2	EX. 14. (appl:staticalize/inexacts / helicop/2.0.5.) (sprithedos)	
1995	Let - p (uptionality/peoplets/conflicts/field)	
100	an feminant postile feine Company, feine	
	North Control of the second seco	
18.1	HINWY NATI AND ADDRESS OF THE STATUS	
1225	um standarding and standard and stand	
100	THE PERSON AND AND ADDRESS (MARK)	
100	The second of the second s	
	perse	
1221	Charach, VB: VecCrAAL (VecCrAAL) State 1 and 4	
1991	andra -p (end/arts/news	
1441	white ye very with the set	
1201	chore	
1222	and the second sec	
	seems -R regrant vegrant (but/dir/lata	
	de traditario algorgi den contributed base	
100	re is Canadaman buyedat-commuted-lana a p.71 lan tabutrate translati committer-lans lan	
	is is that again they add at a summarize they are the source and they are any safet comments that the the	
121	The restriction of the restricti	
100	Per La Catal Manuel Report (Dama 2014, 1944, 1944) - 1914 - 1914	
CT 1	surgers years and prolls	
2211	discuss of the second	
0221	N-COMBILETAR	
	PENDE CONT. CLAR	
100	The second	
	and a second second in the	
	Received a financial of the second se	
	Statistics France	
1000		_



START KAFKA/HADOOP/HIVE AND KAFKA CONNECT IN STANDALONE MODE

DEMO

· · · · · · · · · · · · · · · · · · ·	
[oraclegorecletafkaati atc]5 splplus test/manager	- [root@oraclekafka@t1 @tz]#
505.921us: Helwaye 12.1.0.2.0 Production on Time Nam 12 18:00157 2019	
Copyright (c) 1962, 2014, Crecis. All rights reserved.	
Last Successful login time: Two Her 12 2019 09:56:42 +80:00	
Converted to:	
<pre>re for test_jdbc_users+0 (is.confluent.connect.hdfs.TopicPertitionerit [2019-03-13 10:02135.037] 1000 Counties hdfs://localhost:9000/topics/ 000_aveo for test_jdbc_users+0 (is.confluent.connect.hdfs.TopicPertitioner) [2019-03-12 10:02135.037] 1000 Opening record writer for: hdfs://local 4046-0631-d6128e590200_tmp.aveo (is.confluent.connect.hdfs.aveo.Aveo0 [2019-03-12 10:02135.057] 1000 Opening record writer for: hdfs://local au0f-0600035f46030_tmp.aveo (is.confluent.connect.hdfs.aveo.Aveo0 [2013-03-12 10:02135.057] 1000 Starting connect.hdfs.aveo.Aveo0ecorde [2013-03-12 10:02135.058] 1000 Starting connect.hdfs.aveo.Aveo0ecorde [2013-03-12 10:02135.058] 1000 Starting connect.hdfs.aveo.Aveo0ecorde [2013-03-12 10:02135.058] 1000 Committed hdfs://localhost:0000/topics/ no for test_jdbc_users+0 (is.confluent.connect.hdfs.TopicPertitionarity [2019-03-12 10:02135.059] 1000 Committed hdfs://localhost:0000/topics/ no for test_jdbc_users+0 (is.confluent.connect.hdfs.TopicPertitionarity [2019-03-12 10:02135.051] 1000 starting connect.hdfs.TopicPertitionarity [2019-03-12 10:02155.051] 1000 Committed hdfs://localhost:0000/tupics/ no for test_jdbc_users+0 (is.confluent.connect.hdfs.TopicPertitionarity [2019-03-12 10:02155.051] 1000 Committed hdfs://localhost:0000/tupics/ no for test_jdbc_users+0 (is.confluent.connect.hdfs.TopicPertitionarity [2019-03-12 10:02155.051] 1000 Committed hdfs://localhost:0000/tupics/ no for test_jdbc_users+0 (is.confluent.connect.hdfs.repicPertitionarity [2019-03-12 10:02153.052] 1000 Committed hdfs://localhost:for000/tupics/ no for test_jdbc_users+0 (is.confluent.connect</pre>	<pre>tri746) test_jdtr_users/departmentiongineering/test_jdtr_users+0+000000000;+0000000 everiteri746) hest:0000/tepics//+tmp/test_jdtr_users/department-sales/fa040001-01aa-40of- itarFrowider:05) partition test_jdtr_users-0 with start offsets (department-sales/fa040000000; an er:740; test_jdtr_users/department-sales/fa010000000;+0000000; an er:740; test_jdtr_users/department-sales/fa010000000;+0000000; an er:740; test_jdtr_users/department-sales/fa010000000;+0000000; an er:740; test_jdtr_users/department-sales/fa010000000;+0000000; an er:740; test_jdtr_users/department-sales/fa01000000;+0000000; an er:740; bast:0000/tepics//+tmp/test_jdtr_users/department-test/10010623-101c-Aedi-t SterFrowider:05) partition test_jduc_users-0 with start offsets (department-test/10010623-101c-Aedi-t SterFrowider:05) test_jdtc_users/department-test/test_iduc_useriere:endepode0000000000000000000000000000000000</pre>

CHANGE DATA CAPTURE USING JDBC SOURCE CONNECTOR & HDFS SINK CONNECTOR (COMPATIBILITY - BACKWARD)

DEMO

Stay Tuned – Introducing dbms_kafka package for pub/sub to topic (12.2)

BEGIN

dbms_kafka.register_cluster ('SENS2', '<Zookeeper URL>:2181', '<Kafka broker URL>:9092' ,'DBMSKAFKA DEFAULT DIR', 'DBMSKAFKA_LOCATION DIR' 'Testing DBMS KAFKA'); END; DECLARE
rows_loaded number;
BEGIN
 dbms_kafka.load_table
 ('SENS2', 'LOADAPP',
 'sensor2', 'sensormessages_shape_table', rows_loaded
);
 dbms_output.put_1ine ('rows loaded: '|| rows loaded);
END;

THANK YOU FOR JOINING

Q&A

Videos Embedded in this presentation are available at

https://www.youtube.com/playlist?list=PLR6rN4cTV4BTxVQS-uL7NE6htS-pyLsRV

You can reach me via

Email:- db@geeksinsight.com

Linkedin: https://www.linkedin.com/in/suresh-gandhip/

Blog: <u>http://db.geeksinsight.com</u>

Twitter: <u>https://twitter.com/geeksinsights</u> @geeksinsights

