package com.twitter.search.earlybird.partition;

import java.util.Date;

import com.google.common.annotations.VisibleForTesting;

import com.google.common.base.Preconditions;

import org.apache.commons.lang3.builder.ToStringBuilder;

import com.twitter.search.common.config.Config;

import com.twitter.search.earlybird.common.config.EarlybirdConfig;

import com.twitter.search.earlybird.config.TierConfig;

public class PartitionConfig {

// Which sub-cluster this host belongs to

private final String tierName;

// Which cluster this host belongs to

private final String clusterName;

public static final String DEFAULT\_TIER\_NAME = "all";

// the date range of the timeslices this tier will load. The start date is inclusive, while

// the end date is exclusive.

private final Date tierStartDate;

private final Date tierEndDate;

private final int indexingHashPartitionID; // Hash Partition ID assigned for this EB

private final int maxEnabledLocalSegments; // Number of segments to keep

// The position of this host in the ordered list of hosts serving this hash partition

private final int hostPositionWithinHashPartition;

private volatile int numReplicasInHashPartition;

private final int numPartitions; // Total number of partitions in the current cluster

public PartitionConfig(

int indexingHashPartitionID,

int maxEnabledLocalSegments,

int hostPositionWithinHashPartition,

int numReplicasInHashPartition,

int numPartitions) {

this(DEFAULT\_TIER\_NAME,

TierConfig.DEFAULT\_TIER\_START\_DATE,

TierConfig.DEFAULT\_TIER\_END\_DATE,

indexingHashPartitionID,

maxEnabledLocalSegments,

hostPositionWithinHashPartition,

numReplicasInHashPartition,

numPartitions);

}

public PartitionConfig(String tierName,

Date tierStartDate,

Date tierEndDate,

int indexingHashPartitionID,

int maxEnabledLocalSegments,

int hostPositionWithinHashPartition,

int numReplicasInHashPartition,

int numPartitions) {

this(tierName, tierStartDate, tierEndDate, indexingHashPartitionID, maxEnabledLocalSegments,

hostPositionWithinHashPartition, numReplicasInHashPartition, Config.getEnvironment(),

numPartitions);

}

public PartitionConfig(String tierName,

Date tierStartDate,

Date tierEndDate,

int indexingHashPartitionID,

int maxEnabledLocalSegments,

int hostPositionWithinHashPartition,

int numReplicasInHashPartition,

String clusterName,

int numPartitions) {

this.tierName = Preconditions.checkNotNull(tierName);

this.clusterName = Preconditions.checkNotNull(clusterName);

this.tierStartDate = Preconditions.checkNotNull(tierStartDate);

this.tierEndDate = Preconditions.checkNotNull(tierEndDate);

this.indexingHashPartitionID = indexingHashPartitionID;

this.maxEnabledLocalSegments = maxEnabledLocalSegments;

this.hostPositionWithinHashPartition = hostPositionWithinHashPartition;

this.numReplicasInHashPartition = numReplicasInHashPartition;

this.numPartitions = numPartitions;

}

public String getTierName() {

return tierName;

}

public String getClusterName() {

return clusterName;

}

public Date getTierStartDate() {

return tierStartDate;

}

public Date getTierEndDate() {

return tierEndDate;

}

public int getIndexingHashPartitionID() {

return indexingHashPartitionID;

}

public int getMaxEnabledLocalSegments() {

return maxEnabledLocalSegments;

}

public int getHostPositionWithinHashPartition() {

return hostPositionWithinHashPartition;

}

public int getNumReplicasInHashPartition() {

return numReplicasInHashPartition;

}

/\*\*

\* The number of ways the Tweet and/or user data is partitioned (or sharded) in this Earlybird, in

\* this tier.

\*/

public int getNumPartitions() {

return numPartitions;

}

public String getPartitionConfigDescription() {

return ToStringBuilder.reflectionToString(this);

}

public void setNumReplicasInHashPartition(int numReplicas) {

numReplicasInHashPartition = numReplicas;

}

public static final int DEFAULT\_NUM\_SERVING\_TIMESLICES\_FOR\_TEST = 18;

public static PartitionConfig getPartitionConfigForTests() {

return getPartitionConfigForTests(

TierConfig.DEFAULT\_TIER\_START\_DATE,

TierConfig.DEFAULT\_TIER\_END\_DATE);

}

public static PartitionConfig getPartitionConfigForTests(Date tierStartDate, Date tierEndDate) {

return getPartitionConfigForTests(

DEFAULT\_NUM\_SERVING\_TIMESLICES\_FOR\_TEST, tierStartDate, tierEndDate, 1);

}

/\*\*

\* Returns a PartitionConfig instance configured for tests.

\*

\* @param numServingTimeslices The number of timeslices that should be served.

\* @param tierStartDate The tier's start date. Used only in the full archive earlybirds.

\* @param tierEndDate The tier's end date. Used only by in the full archive earlybirds.

\* @param numReplicasInHashPartition The number of replicas for each partition.

\* @return A PartitionConfig instance configured for tests.

\*/

@VisibleForTesting

public static PartitionConfig getPartitionConfigForTests(

int numServingTimeslices,

Date tierStartDate,

Date tierEndDate,

int numReplicasInHashPartition) {

return new PartitionConfig(

EarlybirdConfig.getString("sub\_tiers\_for\_tests", "test"),

tierStartDate,

tierEndDate,

EarlybirdConfig.getInt("hash\_partition\_for\_tests", -1),

numServingTimeslices,

0, // hostPositionWithinHashPartition

numReplicasInHashPartition,

EarlybirdConfig.getInt("num\_partitions\_for\_tests", -1)

);

}

}