package com.twitter.search.earlybird.config;

import java.util.Date;

import com.google.common.annotations.VisibleForTesting;

import com.google.common.base.Preconditions;

import com.twitter.common.util.Clock;

/\*\*

\* Properties of a single tier.

\*/

public class TierInfo implements ServingRange {

// What I'm seeing historically is that this has been used when adding a new tier. First you

// add it and send dark traffic to it, then possibly grey and then you launch it by turning on

// light traffic.

public static enum RequestReadType {

// Light read: send request, wait for results, and results are returned

LIGHT,

// Dark read: send request, do not wait for results, and results are discarded

DARK,

// Grey read: send request, wait for results, but discard after results come back.

// Same results as dark read; similar latency as light read.

GREY,

}

private final String tierName;

private final Date dataStartDate;

private final Date dataEndDate;

private final int numPartitions;

private final int maxTimeslices;

private final TierServingBoundaryEndPoint servingRangeSince;

private final TierServingBoundaryEndPoint servingRangeMax;

private final TierServingBoundaryEndPoint servingRangeSinceOverride;

private final TierServingBoundaryEndPoint servingRangeMaxOverride;

// These two properties are only used by clients of Earlybird (E.g. roots),

// but not by Earlybirds.

private final boolean enabled;

private final RequestReadType readType;

private final RequestReadType readTypeOverride;

public TierInfo(String tierName,

Date dataStartDate,

Date dataEndDate,

int numPartitions,

int maxTimeslices,

boolean enabled,

String sinceIdString,

String maxIdString,

Date servingStartDateOverride,

Date servingEndDateOverride,

RequestReadType readType,

RequestReadType readTypeOverride,

Clock clock) {

Preconditions.checkArgument(numPartitions > 0);

Preconditions.checkArgument(maxTimeslices > 0);

this.tierName = tierName;

this.dataStartDate = dataStartDate;

this.dataEndDate = dataEndDate;

this.numPartitions = numPartitions;

this.maxTimeslices = maxTimeslices;

this.enabled = enabled;

this.readType = readType;

this.readTypeOverride = readTypeOverride;

this.servingRangeSince = TierServingBoundaryEndPoint

.newTierServingBoundaryEndPoint(sinceIdString, dataStartDate, clock);

this.servingRangeMax = TierServingBoundaryEndPoint

.newTierServingBoundaryEndPoint(maxIdString, dataEndDate, clock);

if (servingStartDateOverride != null) {

this.servingRangeSinceOverride = TierServingBoundaryEndPoint.newTierServingBoundaryEndPoint(

TierServingBoundaryEndPoint.INFERRED\_FROM\_DATA\_RANGE, servingStartDateOverride, clock);

} else {

this.servingRangeSinceOverride = servingRangeSince;

}

if (servingEndDateOverride != null) {

this.servingRangeMaxOverride = TierServingBoundaryEndPoint.newTierServingBoundaryEndPoint(

TierServingBoundaryEndPoint.INFERRED\_FROM\_DATA\_RANGE, servingEndDateOverride, clock);

} else {

this.servingRangeMaxOverride = servingRangeMax;

}

}

@VisibleForTesting

public TierInfo(String tierName,

Date dataStartDate,

Date dataEndDate,

int numPartitions,

int maxTimeslices,

boolean enabled,

String sinceIdString,

String maxIdString,

RequestReadType readType,

Clock clock) {

// No overrides:

// servingRangeSinceOverride == servingRangeSince

// servingRangeMaxOverride == servingRangeMax

// readTypeOverride == readType

this(tierName, dataStartDate, dataEndDate, numPartitions, maxTimeslices, enabled, sinceIdString,

maxIdString, null, null, readType, readType, clock);

}

@Override

public String toString() {

return tierName;

}

public String getTierName() {

return tierName;

}

public Date getDataStartDate() {

return dataStartDate;

}

public Date getDataEndDate() {

return dataEndDate;

}

public int getNumPartitions() {

return numPartitions;

}

public int getMaxTimeslices() {

return maxTimeslices;

}

public TierConfig.ConfigSource getSource() {

return TierConfig.getTierConfigSource();

}

public boolean isEnabled() {

return enabled;

}

public boolean isDarkRead() {

return readType == RequestReadType.DARK;

}

public RequestReadType getReadType() {

return readType;

}

public RequestReadType getReadTypeOverride() {

return readTypeOverride;

}

public long getServingRangeSinceId() {

return servingRangeSince.getBoundaryTweetId();

}

public long getServingRangeMaxId() {

return servingRangeMax.getBoundaryTweetId();

}

long getServingRangeOverrideSinceId() {

return servingRangeSinceOverride.getBoundaryTweetId();

}

long getServingRangeOverrideMaxId() {

return servingRangeMaxOverride.getBoundaryTweetId();

}

public long getServingRangeSinceTimeSecondsFromEpoch() {

return servingRangeSince.getBoundaryTimeSecondsFromEpoch();

}

public long getServingRangeUntilTimeSecondsFromEpoch() {

return servingRangeMax.getBoundaryTimeSecondsFromEpoch();

}

long getServingRangeOverrideSinceTimeSecondsFromEpoch() {

return servingRangeSinceOverride.getBoundaryTimeSecondsFromEpoch();

}

long getServingRangeOverrideUntilTimeSecondsFromEpoch() {

return servingRangeMaxOverride.getBoundaryTimeSecondsFromEpoch();

}

}