package com.twitter.product\_mixer.core.functional\_component.common.alert

import com.twitter.product\_mixer.core.functional\_component.common.alert.predicate.Predicate

import com.twitter.strato.catalog.OpTag

/\*\*

\* triggers when the a Strato column's is outside of the predicate set by the provided [[Alert]]

\*

\* @note the [[Alert]] passed into a [[StratoColumnAlert]]

\* can not be a [[StratoColumnAlert]]

\*/

case class StratoColumnAlert(column: String, op: OpTag, alert: Alert with IsObservableFromStrato)

extends Alert {

override val source: Source = Strato(column, op.tag)

override val notificationGroup: NotificationGroup = alert.notificationGroup

override val warnPredicate: Predicate = alert.warnPredicate

override val criticalPredicate: Predicate = alert.criticalPredicate

override val runbookLink: Option[String] = alert.runbookLink

override val alertType: AlertType = alert.alertType

override val metricSuffix: Option[String] = alert.metricSuffix

}

object StratoColumnAlerts {

/\*\* Make a seq of Alerts for the provided Strato column \*/

def apply(

column: String,

op: OpTag,

alerts: Seq[Alert with IsObservableFromStrato]

): Seq[Alert] = {

alerts.map(StratoColumnAlert(column, op, \_))

}

}