Unica Journey
Version 12.1 Optimierungshandbuch

Contents

Chapter 1. Optimieren von Unica Journey für optimale Leistung

Eine Installation von Unica Journey besteht aus mehreren Komponenten, einschließlich Tools von anderen Anbietern (wie Webanwendungsserver, Datenbanken und Lastausgleichsmodule) und Komponenten wie Unica Platform. Die verschiedenen Eigenschaften, Funktionen und Einstellungen dieser Komponenten können Sie zur Optimierung der Leistung konfigurieren.

Mit den Konfigurationseigenschaften von Unica Journey selbst können Sie Ihre Installation so anpassen, dass die optimale Leistung erzielt wird.

Die Definition von "optimale Leistung" ist jedoch schwierig. Jede Umgebung, jede Implementierung stellt andere Anforderungen.

Die Laufzeitleistung von Unica Journey wird von verschiedenen Faktoren beeinflusst, darunter die Hardwarekonfiguration, Netzkonfiguration und die Konfiguration von Unica Journey. Die folgenden Richtlinien und Empfehlungen können sich daher unterschiedlich auf Ihre Umgebung auswirken.

Journey Hauptkonfigurationseigenschaften

Eine Auflistung der Engine-Eigenschaften für die Konfiguration von Journey Master

1. spring.kafka.retries.config = Anzahl der Male, die der Kafka Producer Client im Falle eines Scheiterns versuchen soll, Daten an den Broker zu übergeben
2. connections.max.idle.ms=-1, gibt die Leerlaufzeit an, um Produzenten und Verbraucher lebendig zu halten und beträgt standardmäßig 540 Sekunden, Änderung auf -1, damit die Verbindungen lebendig bleiben.
3. *.topic=DATA_CLEAN -- gibt den Name des Themas an, von dem aus die Daten vom Service gelesen werden (NICHT ÄNDERN)
4. *.topic.group.id=data-clean-service-consumer-group, Gruppe, zu der der Konsument dieses Service gehört, ist für jeden Service eindeutig (NICHT ÄNDERN).

5. *.topic.max.poll.records = Anzahl des Lesevorgänge des Mitteilungsdiensts, jedes Mal, wenn er mit dem Broker kommuniziert
6. *.topic.partitions = Zahl der Partitionen je Thema (empfohlen: 5)
7. *.topic.replications = Referenz: Konfiguration für die Replizierung von Kafka
8. *.topic.min.threshold = Werte, die für die Skalierung des Service verwendet werden, basierend auf der Anzahl der abgefragten Nachrichten, Anzahl der durchgeführten E/A-Operationen (Standard: 2), sollte immer kleiner als der Maximalwert sein
9. *.topic.max.threshold = Werte, die für die Skalierung des Service verwendet werden, basierend auf der Anzahl der abgefragten Nachrichten, Anzahl der durchgeführten E/A-Operationen (Standard: 10), sollte immer größer als der Minimalwert sein

   (Der Service-Schwellenwertping sollte zwischen der minimalen und maximalen Schwelle liegen, wenn der Ping unter dem Minimum liegt, wird der Service für die Skalierung berücksichtigt (Instanz wird gestoppt), wenn das Schwellenwert Ping mehr als das Maximum ist, dann wird der Service für die Größenausstufung berücksichtigt (Instanz wird erhöht)
10. *.topic.max.instance = (Anzahl der maximalen Instanzen, die dieser Service haben kann, wenn der Schwellenwert steigt, sollte der Wert dieses Elements immer dem Wert von topic.partitions entsprechen (empfohlen: 5)
11. *.topic.default.instance = Standardnummer der Instanz für diesen Service (empfohlen: 1)
12. *.topic.sleep.time = Schlafdauer (in Millisekunden) zwischen der Verarbeitung zweier Stapel (empfohlen: 500)
13. *.topic.max.isalive.threshold = Schwellenwert (in Millisekunden), um den Herzschlag jeder Serviceinstanz zu vergleichen, wenn der Schwellenwert diesen zugewiesenen Wert erhöht, wird die Instanz erneut gestartet (empfohlen: 200)
14. journey_control.topic=JOURNEY_CONTROL, wird für die Kommunikation zwischen Web/Engine/Platform verwendet (NICHT ÄNDERN)
15. journey.engine.monitoring.topic=JOURNEY_ENGINE_MONITORING, wird für Überwachungsservice verwendet (NICHT ÄNDERN)
16. journey.engine.errors.topic=JOURNEY_ENGINE_ERRORS - alle Übertragungsfehler werden in diesem Thema veröffentlicht (NICHT ÄNDERN)

17. hip.request.topic=OUTGOING_MESSAGES, Thema, das für die Kommunikation mit dem HIP-Service zum Senden (E-Mail, SMS) genutzt wird (NICHT ÄNDERN)

* Alle Eigenschaften sind Teil der Datei journey_master_config.properties

Table 1. Konfiguration für die Replizierung von Kafka

Zahl des Kafka-Brokers	Replikationsfaktor
1	1
2	1
3	2
5	3
7	5

Journey Engine-Anwendungseigenschaften

Eine Auflistung der Engine-Anwendungseigenschaften

spring.ignite.ipFinder.List - Diese Eigenschaft wird verwendet, um vorkonfigurierte, standardmäßig angegebene IP-Adressen festzulegen. Dieser IP-Finder wird standardmäßig wird nicht geteilt. Dies bedeutet, dass alle Rasterknoten mit der gleichen Webadresse konfiguriert werden müssen. Beispielsweise (Server A und Server B befinden sich im Cluster, dann für:

1. Server A – (ServerA_IP:port,ServerB_IP:Port)
2. Server B – (ServerB_IP:port,ServerA_IP:Port)

spring.ignite.defaultDataRegion.max.size - Diese Eigenschaft wird für die Speicherzuordnung für Ignite eingesetzt. Die Gesamtgröße darf nicht kleiner als 10 MB sein (empfohlene 2 GB auf 16GB RAM in Linux)/(1 GB auf 16 GB RAM in Windows).

journey.audience.next.state.on.data.error - Diese Eigenschaft entscheidet, ob die Zielgruppe zu ' kein Durchlauf ' oder ' Fehlerzustand ' verschoben wird. Werte können ' Fehler ' oder ' Nein ' sein

engine.logging.cron - Diese Eigenschaft wird verwendet, um die Ausführungszeit des Jobs für die Protokollierung von Interaktionsprogrammen festzulegen

journey_configuration.topic - Gibt den Namen des Topics an, von dem aus Daten vom Service eingelesen werden (NICHT ÄNDERN)

journey_configuration.topic.group.id Gruppe, zu der der Konsument dieses Service gehört, ist für jeden Service eindeutig (NICHT ÄNDERN)

journey_configuration.short_sleep Ruhezeit (in Millisekunden) zwischen der Verarbeitung zweier Stapel

Journey Konfiguration des Engine Service Thread

Journey Die Engine verwendet alle Thread-Poolkonfigurationen aus service_config.properties

1. **Synchrone Thread-Poolkonfiguration** - Dieser Threadpool verwaltet die gesamten Journey Engine Service Threads.
    a. **sync.thread.pool.max.size=80** (Maximale Poolgröße von ThreadPoolExecutor).
    b. **sync.thread.pool.core.size=60** (Kernpoolgröße von ThreadPoolExecutor).
    c. **sync.thread.pool.queue.capacity=500** (Kapazität der BlockingQueue von ThreadPoolExecutor).
2. **Asynchrone Thread-Poolkonfiguration:** Dieser Threadpool verwaltet alle Vorgänge, die asynchron ausgeführt werden können, z. B.: Speichern von ausrangierten Daten & Berichtsdaten in Datenbank, Verzögern der Verarbeitung eines TouchPoint.
    a. **async.thread.pool.max.size=30** (Maximale Poolgröße von ThreadPoolExecutor).
    b. **async.thread.pool.core.size=20** (Kernpoolgröße von ThreadPoolExecutor).
    c. **async.thread.pool.queue.capacity=100000** (Kapazität von ThreadPoolExecutor).
3. **Datenstapelung:** Große Zielgruppendaten werden in einem Datenblock verarbeitet (Stapel). Die Gesamtgröße des Stapels kann wie unten konfiguriert werden:
    a. **implicit.service.databatch.timeout = 15** (Wartezeit in Sekunden für Stapel. Wenn die Stapelgröße innerhalb von 15 Sekunden nicht voll ist, läuft das Zeitlimit für die Stapelung aus und die zu verarbeitenden Datensätze im Stapel werden verarbeitet).
    b. **implicit.service.databatch.batchSize = 100** (maximale Stapelgröße).

4. **journey.audience.batch.size** = 500 (die Zielgruppenzahl wurde aus Datenbank abgerufen und in 500er-Stapeln an den ersten Knoten der Journey zur Verarbeitung übergeben).
5. **batch.data.import.batch.size=5** (Reihe von Zielgruppendatensätzen, die in 5er-Stapeln übergeben werden, wenn entrysource vom Dateityp ist). Der Maximalwert für diese Eigenschaft ist 5.
6. **engine.monitoring.time=900000** (Zeitraum in ms, nach der die Engime die Komponenten überwacht, die für die Engine erforderlich sind, d. h. Datenbank, Kafka und Ignite).
7. **hip.batch.size = 100** (die Zielgruppe wird zur Verarbeitung in Batzen von 2 an die Hüfte übergeben).