HITACHI
Inspire the Next

ストリームデータ処理基盤

uCosminexus Stream Data Platform - Application Framework 解説

解説書

3020-3-V01-20

■ 対象製品

●適用 OS：Windows Server 2008 R2 Standard，Windows Server 2008 R2 Enterprise，Windows Server 2008 R2 Datacenter

P-2964-9B14 uCosminexus Stream Data Platform - Application Framework 01-05※

●適用 OS：Red Hat Enterprise Linux 5，Red Hat Enterprise Linux 6

P-9W64-9B11 uCosminexus Stream Data Platform - Application Framework 01-05

●適用 OS：Red Hat Enterprise Linux 6

P-9W64-9V11 uCosminexus Stream Data Platform - Application Framework 01-50

注※　この製品については，サポート時期をご確認ください。

これらのプログラムプロダクトのほかにもこのマニュアルをご利用になれる場合があります。詳細は「リリースノート」でご確認ください。

■ 輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。

なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

including various modifications by Spyglass Inc., Carnegie Mellon University, and Bell Communications Research, Inc (Bellcore).

uCosminexus Stream Data Platform - Application Framework は，米国 EMC コーポレーションの RSA BSAFE(R)ソフトウェアを搭載しています。

■ マイクロソフト製品の表記について

このマニュアルでは，マイクロソフト製品の名称を次のように表記しています。

表記		製品名
Windows Server 2008 または Windows	Windows Server 2008 R2 Standard	Microsoft(R) Windows Server(R) 2008 R2 Standard x64
	Windows Server 2008 R2 Enterprise	Microsoft(R) Windows Server(R) 2008 R2 Enterprise x64
	Windows Server 2008 R2 Datacenter	Microsoft(R) Windows Server(R) 2008 R2 Datacenter x64
Internet Explorer		Windows(R) Internet Explorer(R)

■ 発行

2014 年 6 月　3020-3-V01-20

■ 著作権

変更内容

変更内容(3020-3-V01-20)　uCosminexus Stream Data Platform - Application Framework　01-50

追加・変更内容	変更個所
バージョンの変更に伴い，01-50 に対応した記述を追加した。	－

単なる誤字・脱字などはお断りなく訂正しました。

はじめに

このマニュアルは，uCosminexus Stream Data Platform - Application Framework の概要や前提知識について説明したものです。uCosminexus Stream Data Platform - Application Framework の特長やシステム構成などの概要，およびシステムを構築・運用するために必要な前提知識の習得を目的としています。

■ 対象読者

uCosminexus Stream Data Platform - Application Framework を使用するすべてのユーザーを対象としています。

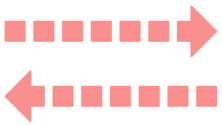
次の知識をお持ちの方を前提としています。

- OS の基礎的な知識

■ 図中で使用する記号

このマニュアルの図中で使用する記号を，次のように定義します。

●ストリームデータの流れ



目次

1 Stream Data Platform - AF とは

Stream Data Platform - AF は，リアルタイムに出力され続ける膨大なデータを分析できるストリームデータ処理を実現するための製品です。この章では，Stream Data Platform - AF の概要，特長，導入例，システム構成などについて説明します。

1.1　「今」を分析するデータ処理システム

社会インフラの変化によって，携帯電話，IC カード，家電製品など，身の回りにあるさまざまなものが膨大な情報を持つようになった現在，データ処理システムが扱う情報量は日々増え続けています。そして，これらの情報の中には，迅速に集計・分析することで新たな価値を生むものがあります。リアルタイムに出力され続ける膨大な「今」の情報の中から新たな価値を生み出すことは，データ処理システムへの要求の一つになっています。

Stream Data Platform - AF は，連続して発生する膨大な時系列順のデータ（**ストリームデータ**）に対して，発生時からのタイムラグなしでリアルタイムに集計・分析する**ストリームデータ処理**を実現することで，この要求にこたえます。

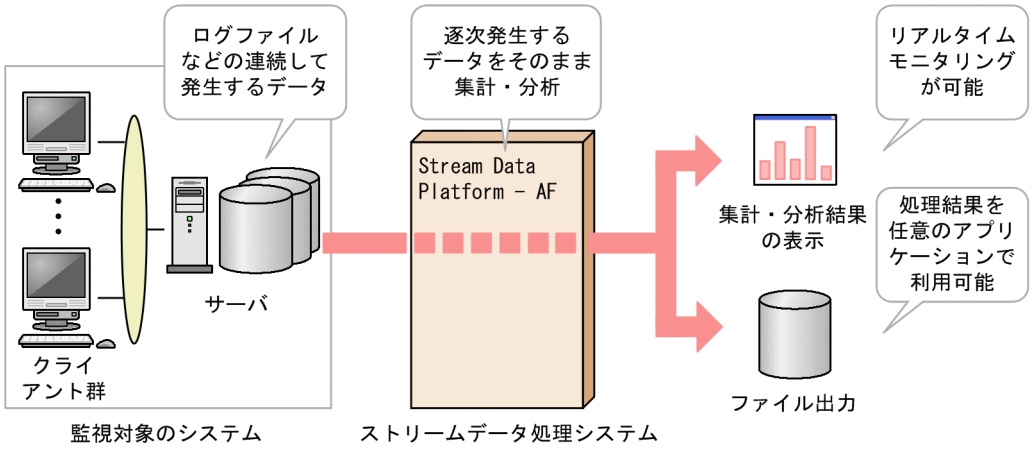
例えば，PC や携帯電話からアクセスできる検索サイトが持つ情報をリアルタイムに集計・分析することで，商品の販売好機がわかります。ある商品がコミュニティサイトなどで話題になり，需要の増加が見込まれる場合，検索サイトでは対象の商品名の検索回数が増加する傾向があります。ストリームデータ処理によって，検索回数をリアルタイムに集計・分析することでそのような商品を特定できれば，販売店側では迅速に商品を追加発注でき，メーカー側では商品の増産を迅速に決断できます。

また，IT システムでは，高効率での運用やコストの削減が求められ，仮想化やクラウドコンピューティングの採用が加速しています。この結果，システム構成の大規模化・複雑化が起き，IT システムの稼働状況は見えにくくなっています。そのため，障害の発生時に，障害の検知や対策に時間が掛かることがあります。しかし，ストリームデータ処理によって膨大なデータを集計・分析し続け，システムの稼働状況をリアルタイムにモニタリングすれば，障害発生時に迅速な対応ができます。また，システムの稼働情報の傾向分析や相関分析によってシステムの障害予兆を検知し，障害の発生防止に役立てることができます。

このように，ストリームデータ処理を実現する Stream Data Platform - AF のデータ処理システムへの導入は，膨大な情報を処理するためのアプローチとして適しています。

Stream Data Platform - AF を利用したストリームデータ処理の概要を次の図に示します。

図 1-1　Stream Data Platform - AF を利用したストリームデータ処理の概要



Stream Data Platform - AF を導入したストリームデータ処理システムでは，連続して発生するデータをそのまま集計・分析の対象にできます。

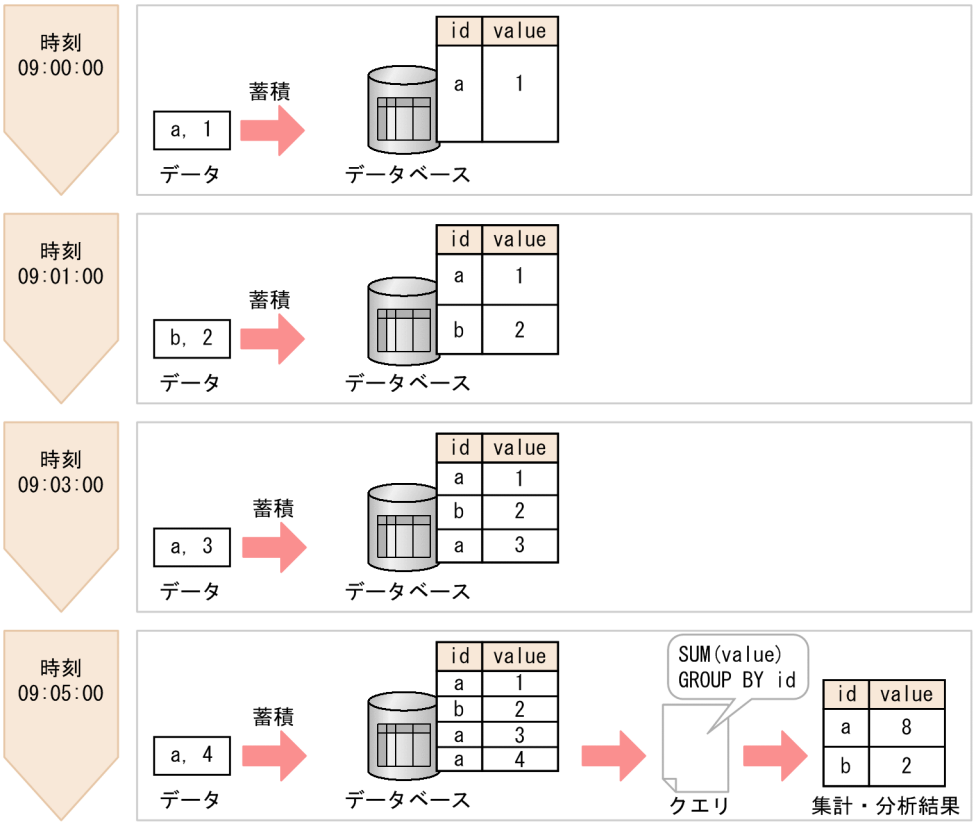
例えば，ストリームデータ処理システムでシステムの稼働監視をする場合，サーバが出力するログファイルや，ネットワーク上の HTTP パケットなどを集計・分析し，処理結果をダッシュボードで表示することで，システムの稼働状況をリアルタイムにモニタリングできます。これによって，システムの障害発生時に迅速

に対処することができ，運用効率や保守効率を向上できます。また，処理結果をファイルに出力すれば，任意のアプリケーションで利用できます。

ストリームデータ処理が，どのようにしてリアルタイムな処理を実現しているのか，従来のストック型データ処理と比較して説明します。

従来のストック型データ処理を次の図に示します。

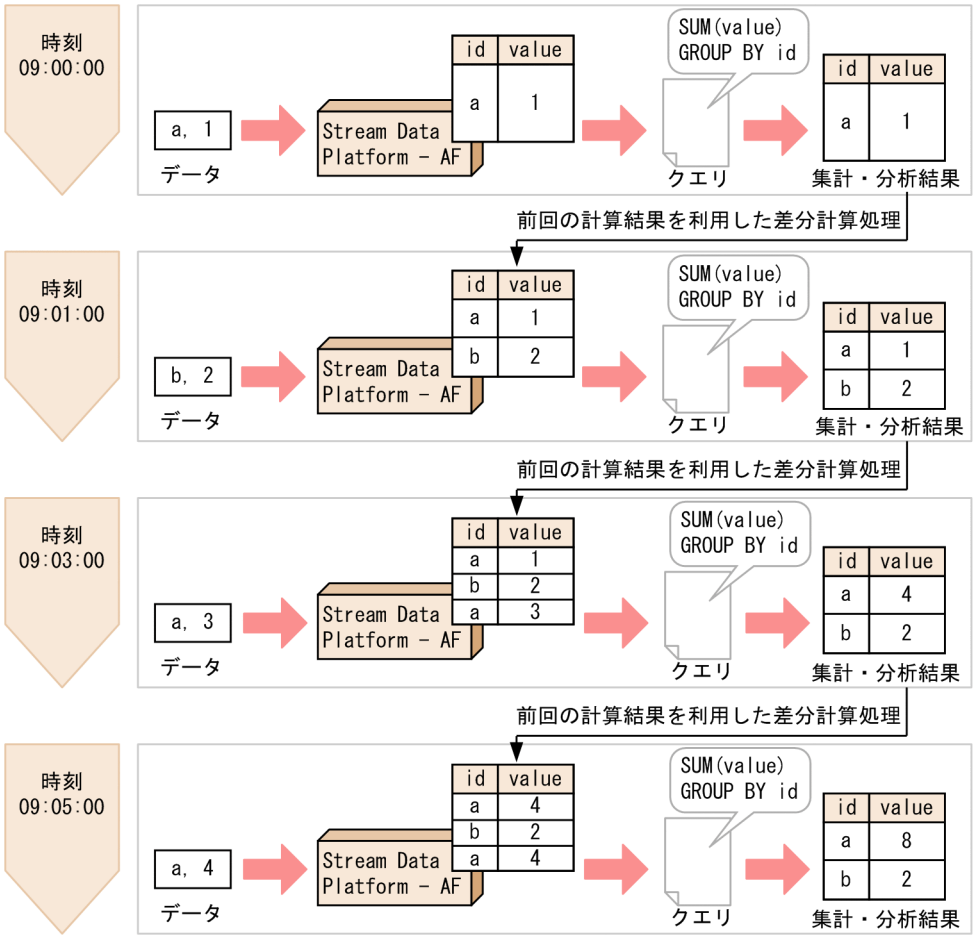
図 1-2　ストック型データ処理



ストック型データ処理を利用したデータ処理では，発生したデータは順次データベースなどに蓄積されます。蓄積されたデータはクエリの発行を契機に処理され，集計・分析結果を得ます。問い合わせを受けてから，あらかじめデータベースに格納されたデータを検索するため，得られたデータの集計・分析結果には，データ発生時からのタイムラグが生じます。図に示した処理では，09:00:00 に発生したデータが，09:05:00 のクエリの発行によって処理され，データ発生時からのタイムラグが生じていることがわかります。

これに対して，ストリームデータ処理を次の図に示します。

図 1-3　ストリームデータ処理



ストリームデータ処理では，クエリ（集計・分析シナリオ）をあらかじめ登録しておき，最小限の計算処理をする差分計算処理を実行します。この計算処理は，データの入力を契機に実行されるため，データの発生からのタイムラグのない，リアルタイムな集計・分析結果が得られます。データの入力を契機に処理が実行されるストリームデータ処理は，データが次々に発生する場合に効果的です。

このように，Stream Data Platform - AF を導入してストリームデータ処理を実現することで，リアルタイムな集計・分析を行えます。

1.2　Stream Data Platform - AF の特長

Stream Data Platform - AF には，次の特長があります。

- 膨大な時系列データの高速処理
- プログラミングを必要としない集計・分析シナリオの定義

それぞれの特長について説明します。

1.2.1　膨大な時系列データの高速処理

Stream Data Platform - AF では，**インメモリ処理**と**差分計算処理**を併用することで，膨大な時系列データを高速処理しています。

インメモリ処理

インメモリ処理では，ディスクドライブを使用しないで，メモリ上で処理を実行します。

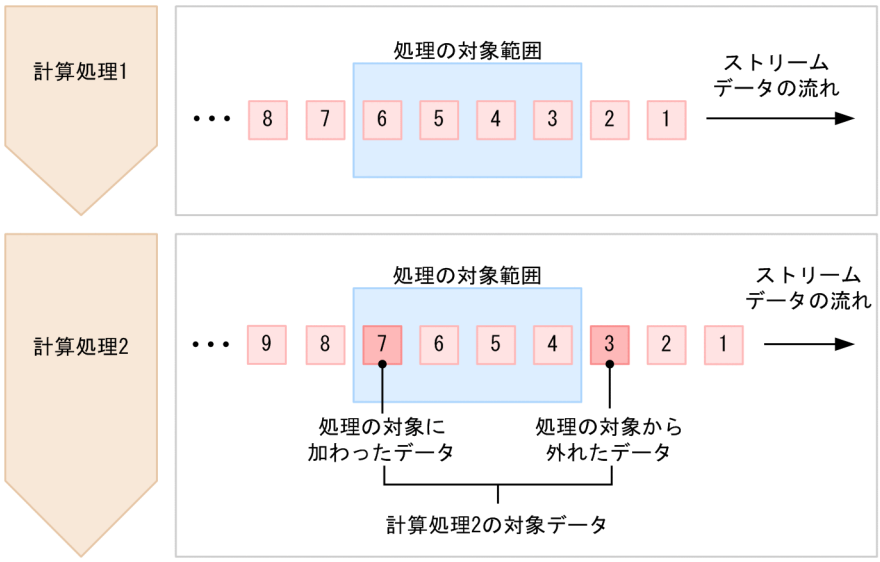
膨大なデータを処理する場合，ディスクドライブとの入出力処理に掛かる時間を無視できなくなります。Stream Data Platform - AF では，このディスクドライブとの入出力処理をなくし，メモリ上でデータを処理することで，データの処理速度を高速化しています。

差分計算処理

差分計算処理では，クエリをあらかじめ登録しておき，データの入力を契機に逐次処理して，処理結果を次の処理に利用します。したがって，次の処理では処理対象のデータ全体を計算する必要がなく，変化量の計算処理だけで済みます。

Stream Data Platform - AF でのストリームデータの差分計算を次の図に示します。

図 1-4　ストリームデータの差分計算



この図に示すとおり，Stream Data Platform - AF は，最初に計算処理 1 を実行します。次のストリームデータが到来したとき，計算処理 2 では，計算処理 1 の結果から，対象から外れた 3 番目のデータを引き，新たに対象となった 7 番目のデータを加える処理だけを実行します。このように最小限の計算処理を行うことで，データの処理速度を高速化しています。

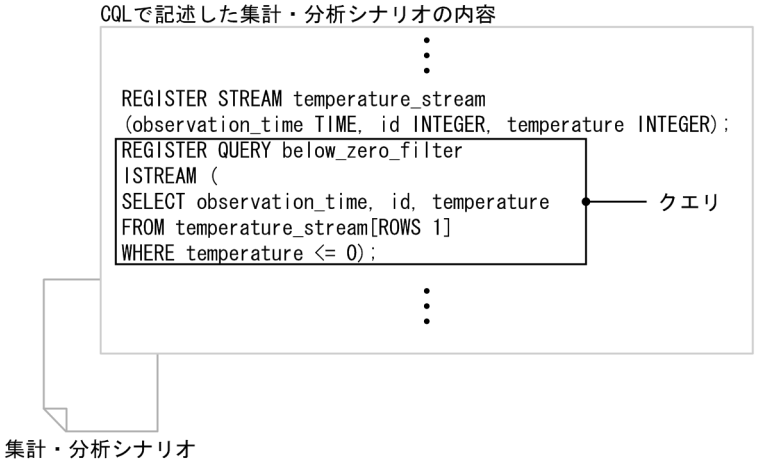
1.2.2　プログラミングを必要としない集計・分析シナリオの定義

ストリームデータ処理の内容は，**集計・分析シナリオ**に定義します。この集計・分析シナリオの定義は，データベースで標準的に扱われる SQL に類似した言語である **CQL** で記述します。そのため，集計・分析シナリオを定義するときに，分析専用アプリケーションの作成は必要ありません。集計・分析シナリオを変更する場合も，CQL で記述した定義ファイルの書き換えだけで済みます。

CQL で記述するストリームデータ処理の内容を**クエリ**といいます。集計・分析シナリオ内には，複数のクエリが定義できます。

例えば，それぞれ ID が与えられた複数の観測所を持つ気温観測システムで，観測結果が氷点下のデータを集計・分析の対象とする場合，集計・分析シナリオには，次の図に示すように CQL を記述します。

図 1-5　CQL を使用した集計・分析シナリオの記述例



CQL は，クエリ言語として汎用的な言語であり，いろいろな処理を記述できます。複数のクエリを組み合わせることで，さまざまな業務に対応する集計・分析シナリオを定義できます。

1.3　Stream Data Platform - AF の導入例

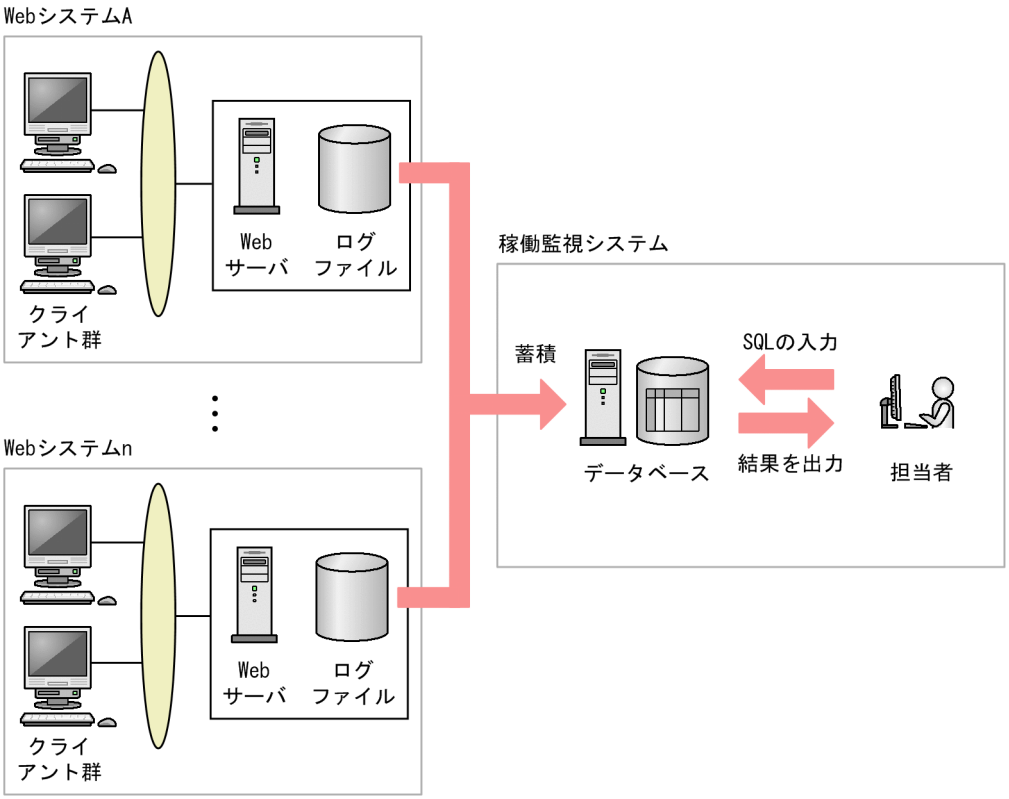
Stream Data Platform - AF は，さまざまなデータを集計・分析の対象にできます。ここでは，その一例として HTTP パケットを集計・分析の対象とすることで実現する Web システムの稼働監視について説明します。

参考

ログファイルなどを集計・分析の対象としてシステムの稼働監視を行うこともできます。

Web システムを構成するそれぞれの Web サーバのログファイルを確認するなどして，定期的に稼働監視をしていた場合，Stream Data Platform - AF を導入して HTTP パケットを利用すればリアルタイムな稼働監視を実現できます。ここで説明するシステムの構成を次の図に示します。

図 1-6　Web システムの稼働監視でのシステム構成例



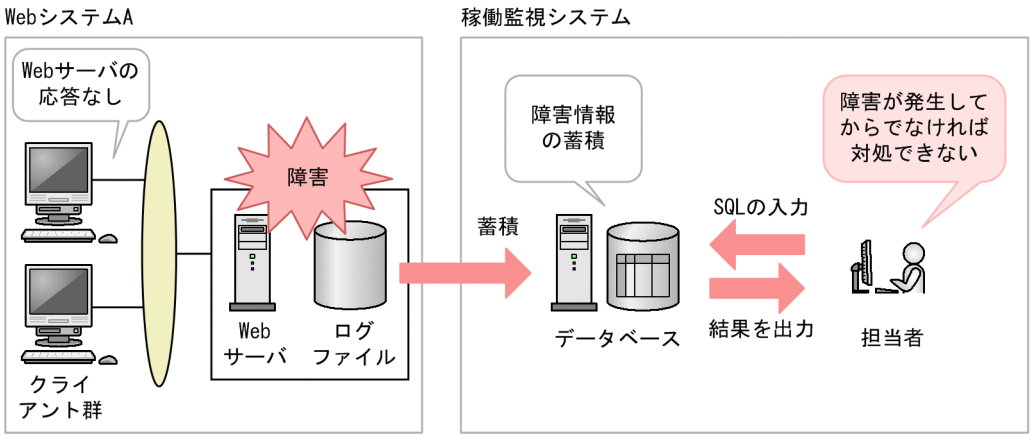
この図で示しているのは，Web サーバのログファイルをデータベースに蓄積したあと，稼働監視システムの担当者が SQL を入力して集計・分析を行うことで，定期的に Web システム全体を監視するシステムです。

図中の Web システム A で障害が発生した場合，Stream Data Platform - AF の導入前と導入後では稼働監視がどのように変化するのかを説明します。

Stream Data Platform - AF の導入前

Web サーバに異常が発生していても，担当者がログファイルを確認し，集計・分析しないかぎりは対処できません。障害をリアルタイムに検知できないため，対策が常に後手に回ってしまいます。この状態を次の図に示します。

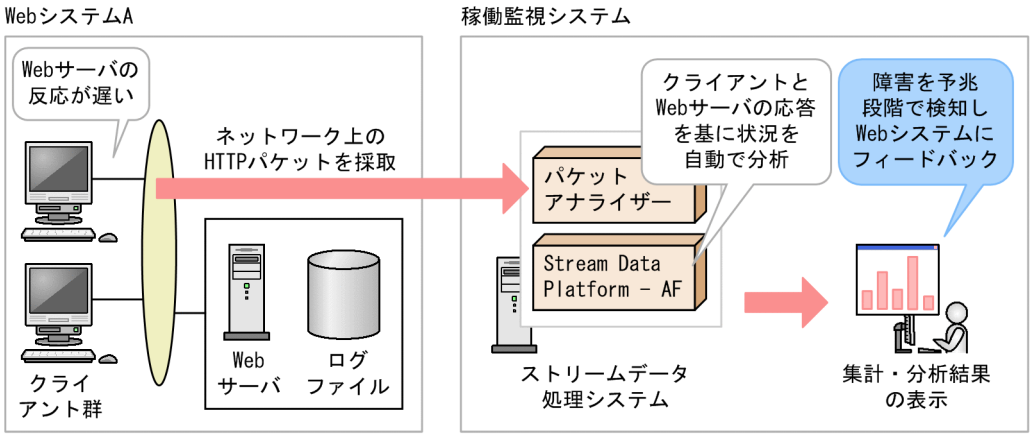
図 1-7　Web システムでの稼働監視での導入例（導入前）



Stream Data Platform - AF の導入後

Stream Data Platform - AF を導入すると，ネットワーク上のデータの通信量やその変化に関する情報をログ出力前に HTTP パケットから直接取得し，システムの稼働状況をリアルタイムに監視できます。これによって，障害の発生を予兆段階で検知できるようになり，保守効率を向上できます。なお，Stream Data Platform - AF では，HTTP パケットを集計・分析するために，パケットアナライザー（ネットワーク上の HTTP パケットを採取できるプログラム）を使用します。この状態を次の図に示します。

図 1-8　Web システムの稼働監視での導入例（導入後）



ネットワーク上の HTTP パケットを採取し，クライアントと Web サーバ間の応答から，Stream Data Platform - AF がリアルタイムに状況を分析します。例えば，クライアントからのリクエストに対する Web サーバのレスポンスの時間を分析することで，Web サーバの応答が遅れ始めているといった現象を迅速に把握できます。担当者は，この情報を Web システムにフィードバックすることで障害の発生を防止できます。

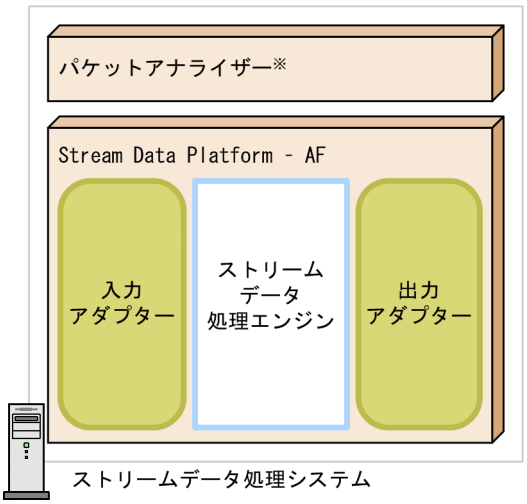
1.4　Stream Data Platform - AF のシステム構成と前提プログラム

ここでは，Stream Data Platform - AF の導入で実現するストリームデータ処理システムの構成と，目的に応じた前提プログラムについて説明します。

1.4.1　システム構成

Stream Data Platform - AF を導入したストリームデータ処理システムの構成を次の図に示します。

図 1-9　ストリームデータ処理システムの構成



注※
HTTPパケットを入力する場合に必要です。

図中で示した Stream Data Platform - AF を構成する要素について説明します。

- **入力アダプター**

  入力アダプターは，入力となるデータを，ストリームデータ処理エンジンで処理できる形式へ変換したり，データの絞り込み（フィルタリング）などを実行したりして，ストリームデータ処理エンジンに送信します。

  入力アダプターには，標準提供アダプターとカスタムアダプターの 2 種類があります。標準提供アダプターを使用する場合，入力できるデータは，分析対象となるシステムで出力されたファイルや HTTP パケットなどです。カスタムアダプターを使用する場合は，作成するアプリケーション次第でさまざまなデータを入力できます。

  入力アダプターについては，「3.　データの送受信」を参照してください。

- **ストリームデータ処理エンジン**

  ストリームデータ処理エンジンは，入力アダプターから送信されてきたデータを，あらかじめ登録されているクエリに従って処理します。ストリームデータ処理エンジンについては，「2.　ストリームデータ処理」を参照してください。

- **出力アダプター**

  出力アダプターは，ストリームデータ処理エンジンで処理されたデータを受信し，フィルタリングや指定された形式への変換を実行して出力先に出力します。

出力アダプターには，標準提供アダプターとカスタムアダプターの 2 種類があります。標準提供アダプターを使用する場合，処理結果は，ファイルやダッシュボードに出力できます。カスタムアダプターを使用する場合は，作成するアプリケーション次第でさまざまな出力先に出力できます。

出力アダプターについては，「3.　データの送受信」を参照してください。

1.4.2　前提プログラム

Stream Data Platform - AF には，データの入出力方法によって必要になる前提プログラムがあります。ここで説明するデータの入出力をしない場合は，前提プログラムは必要ありません。

(1)　HTTP パケットを入力する場合

ネットワーク上の HTTP パケットを入力して，Stream Data Platform - AF で集計・分析する場合，Pcap 形式でパケットを取得できる**パケットアナライザー**が必要です。

Stream Data Platform - AF では，次に示すパケットアナライザーに対応しています。

- WinDump（Windows の場合）
- tcpdump（Linux の場合）

(2)　ダッシュボードに出力する場合

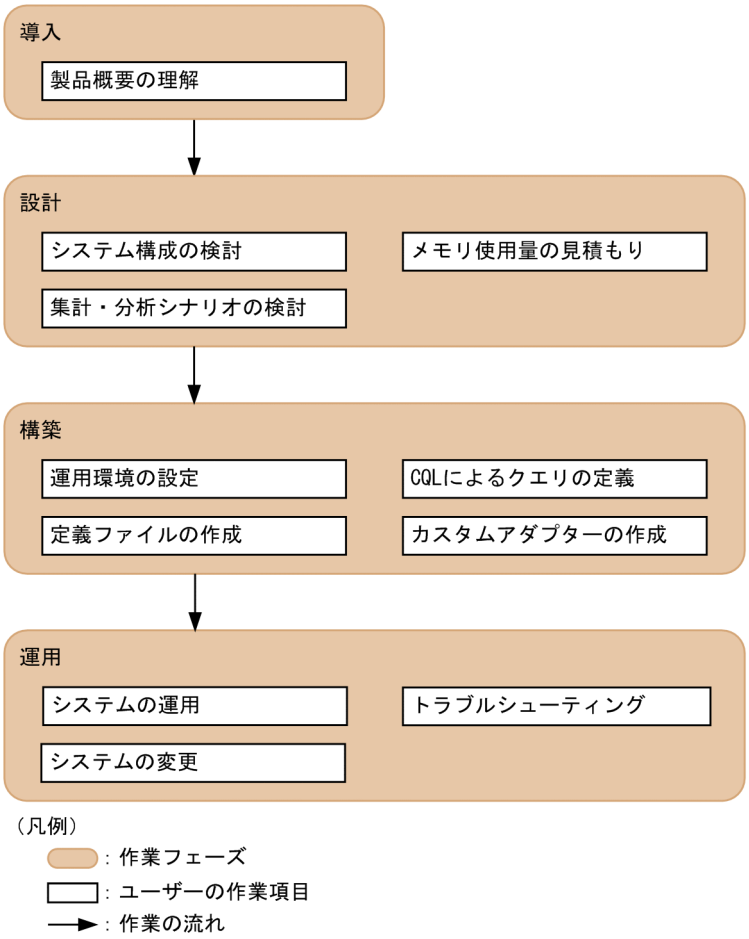
Stream Data Platform - AF でのストリームデータの集計・分析結果をダッシュボードに出力して表示する場合，次に示すプログラムが必要です。

- Internet Explorer
- Flash Player

1.5　Stream Data Platform - AF の導入以降の作業の流れ

Stream Data Platform - AF を導入した場合，作業の流れは，次の図のようになります。

図 1-10　導入以降の作業の流れ



図中で示した各作業フェーズの概要を説明します。

- **導入**

  導入フェーズでは，Stream Data Platform - AF を使用する前に，Stream Data Platform - AF の製品概要について理解します。

- **設計**

  設計フェーズでは，Stream Data Platform - AF を導入したシステムを設計するに当たり，システム構成や集計・分析シナリオの検討，およびメモリ使用量の見積もりを行います。

- **構築**

  構築フェーズでは，Stream Data Platform - AF を導入したシステムを構築するに当たり，運用環境の設定，および定義ファイルの作成を行います。また，CQL によるクエリの定義をします。

  なお，アダプターにカスタムアダプターを使用する場合は，Stream Data Platform - AF で提供する API を使用してカスタムアダプターを作成します。

- **運用**

運用フェーズでは，Stream Data Platform - AF を導入したシステムを運用し，必要に応じて変更します。また，障害の発生時には，トラブルシューティング（障害情報の採取や，トラブルへの対処など）も行います。

製品概要についてはこのマニュアルを，そのほかの作業フェーズについてはシリーズマニュアルをお読みください。

シリーズマニュアルと作業項目の対応については，「4.1　シリーズマニュアルとユーザーの作業項目の対応」を参照してください。

2 ストリームデータ処理

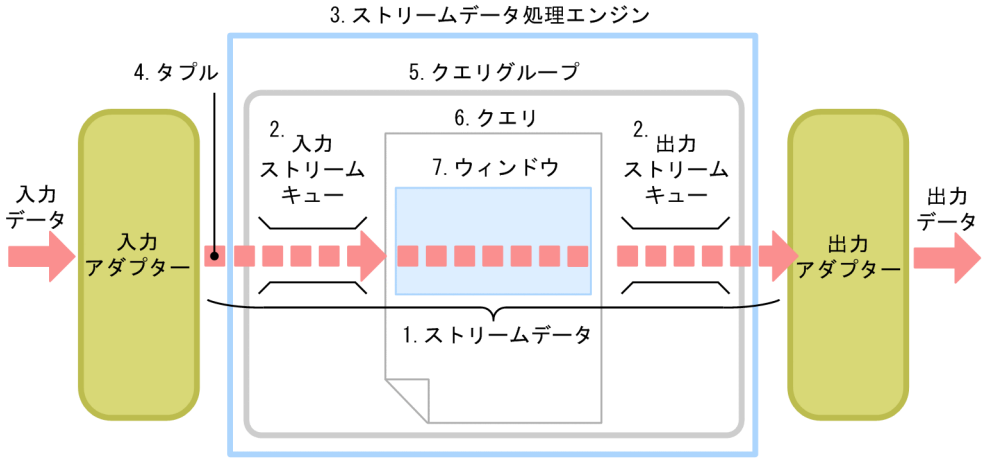
この章では，ストリームデータ処理で使用する要素や，集計・分析シナリオを定義するために使用するクエリ言語 CQL について説明します。

2.1　ストリームデータ処理で使用する要素

ここでは，ストリームデータ処理で使用する要素について説明します。

ストリームデータ処理で使用する要素を次の図に示します。

図 2-1　ストリームデータ処理で使用する要素



この章では，図中に示した要素のうち，次の要素について説明します。

1. **ストリームデータ**
   連続して発生する膨大な時系列順のデータです。
2. **入力ストリームキュー・出力ストリームキュー**
   ストリームデータの通路です。
3. **ストリームデータ処理エンジン**
   ストリームデータ処理システムで，実際にストリームデータ処理を行う部分です。
4. **タプル**
   ストリームデータを構成する，値と時刻を併せ持つデータです。
5. **クエリグループ**
   ストリームデータ処理での集計・分析シナリオです。業務の目的ごとに作成します。
6. **クエリ**
   ストリームデータ処理の内容です。CQL で記述します。
7. **ウィンドウ**
   ストリームデータ処理の対象範囲です。ストリームデータのうち，ウィンドウで囲まれた範囲が処理の対象となります。クエリ中に定義します。

なお，入力アダプター，出力アダプターについては，「3.　データの送受信」を参照してください。

2.1.1　ストリームデータ

ストリームデータは，連続して発生する膨大な時系列順のデータです。

ストリームデータは，CQL で定義するストリームデータの型（**ストリーム**）に従って流れ，**入力ストリームキュー**を通過して，クエリで処理されます。また，クエリでの処理結果は，ストリームデータに変換されたあと，**出力ストリームキュー**を通過して出力されます。

2.1.2　入力ストリームキュー・出力ストリームキュー

入力ストリームキューは，入力となるストリームデータが通過するための通路です。ストリームを定義するための CQL をクエリ中に記述することで作成されます。ストリームを定義するための CQL については，「2.2.1(1)　ストリームの定義（REGISTER STREAM 句）」を参照してください。

出力ストリームキューは，ストリームデータ処理エンジンでの処理結果（ストリームデータ）を出力するための通路です。出力ストリームキューは，クエリでの処理結果をストリームデータとして出力するための CQL をクエリ中に記述することで作成されます。クエリでの処理結果をストリームデータとして出力するための CQL については，「2.2.2(3)　ストリーム化演算（データの処理結果の出力）」を参照してください。

なお，入力ストリームキューを通過するストリームデータの型を**入力ストリーム**といい，出力ストリームキューを通過するストリームデータの型を**出力ストリーム**といいます。

2.1.3　ストリームデータ処理エンジン

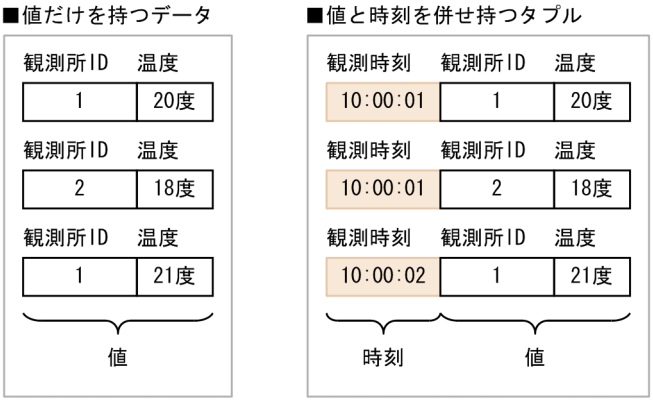
ストリームデータ処理エンジンは，ストリームデータ処理を実行する Stream Data Platform - AF の中心的な要素です。あらかじめ登録したクエリの定義内容に従い，入力アダプターから送信されてきたストリームデータをリアルタイムに処理します。処理結果は，出力アダプターに送られます。

2.1.4　タプル

タプルは，ストリームデータを構成する，値と時刻（**タイムスタンプ**）を併せ持ったデータです。

例として，観測所 1（観測所 ID：1）と観測所 2（観測所 ID：2）でそれぞれ気温を観測した場合について，値だけを持つデータと，値と時刻を併せ持つタプルを比較した図を次に示します。

図 2-2　値だけを持つデータと，値と時刻を併せ持つタプルの比較



この図に示すとおり，観測所ごとの温度情報だけを扱うのではなく，観測時刻というタイムスタンプをタプルに設定することで，ストリームデータ処理の対象にできます。

タプルのタイムスタンプの設定には，タプルがストリームデータ処理エンジンに到着した時刻のタイムスタンプを設定する**サーバモード**と，データの発生した時点の時刻でタイムスタンプを設定する**データソース**

モードがあります。ログ解析処理など，データソース上の時刻情報の時系列順にストリームデータ処理をしたい場合は，データソースモードを使用してください。

以降で，それぞれのモードについて説明します。

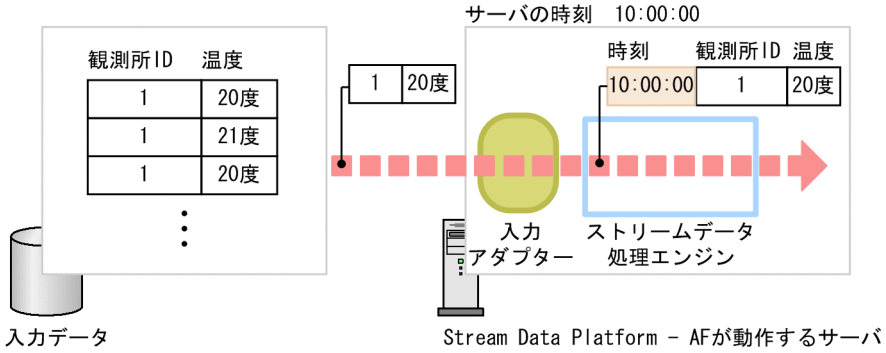
参考

ストリームデータ処理エンジンへの入力タプル，およびストリームデータ処理エンジンからの出力タプルは，それぞれログファイル（**タプルログ**）に出力されます。タプルログは，クエリの再実行や処理結果の確認に使用できます。タプルログについては，マニュアル「uCosminexus Stream Data Platform - Application Framework システム構築・運用ガイド」を参照してください。

(1)　サーバモードによるサーバのシステム時刻の設定

ストリームデータ処理エンジンにタプルが到着した時点で，Stream Data Platform - AF が動作するサーバのシステム時刻をタプルに設定するモードを**サーバモード**といいます。サーバモードによるタイムスタンプの設定を次の図に示します。

図 2-3　サーバモードによるタイムスタンプの設定

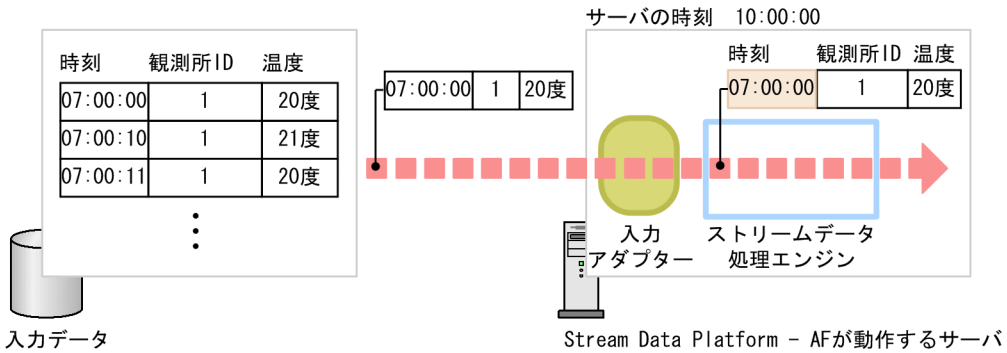


サーバモードでは，入力データがタイムスタンプを持っているかどうかに関係なく，Stream Data Platform - AF が動作するサーバの時刻をタプルに設定します。

(2)　データソースモードによるデータソースの時刻の設定

ログファイルなど，入力とするデータソース上に時刻情報がある場合に，その時刻情報をタプルに設定するモードを**データソースモード**といいます。データソースモードによるタイムスタンプの設定を次の図に示します。

図 2-4　データソースモードによるタイムスタンプの設定



データソースモードでは，入力データが持つタイムスタンプをタプルに設定します。

なお，データソースモードでストリームデータ処理を実行するには，ストリームデータをタイムスタンプの昇順に並べる必要があります。ストリームデータのタイムスタンプが誤差を含んでいるような場合に，タイムスタンプの昇順に並べる方法については，マニュアル「uCosminexus Stream Data Platform - Application Framework システム構築・運用ガイド」を参照してください。

2.1.5　クエリグループ

クエリグループは，あらかじめユーザーが作成するストリームデータの**集計・分析シナリオ**です。入力ストリームキュー（入力ストリーム），出力ストリームキュー（出力ストリーム），およびクエリで構成されます。

クエリグループは，業務の目的ごとに作成して登録します。また，複数のクエリグループの登録もできます。

2.1.6　クエリ

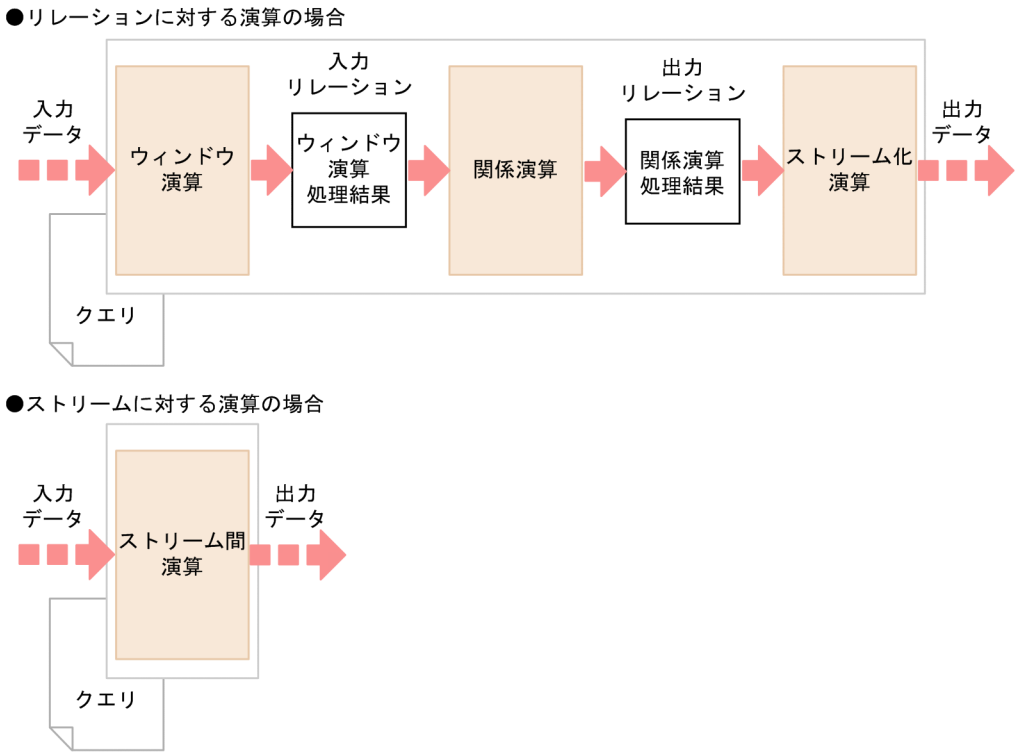
クエリは，ストリームデータに対してどのような処理を実行するかを定義したものです。クエリは，**クエリ定義ファイル**内に CQL で記述します。クエリ定義ファイルについては，マニュアル「uCosminexus Stream Data Platform - Application Framework システム構築・運用ガイド」を参照してください。

クエリでは，次の 4 種類の演算処理を定義します。

- ストリームデータから分析対象のデータを抽出する**ウィンドウ演算**
- 抽出したデータに処理を実行する**関係演算**
- 処理の結果をストリーム化して出力する**ストリーム化演算**
- ストリームデータを処理して別のストリームデータに変換する**ストリーム間演算**

これらの演算処理の関係を次の図に示します。

図 2-5　クエリでの処理の関係



ウィンドウ演算は，ストリームデータからウィンドウで一定の範囲内のデータを抽出するための演算です。このとき生成されるデータ（タプルの集合）を**入力リレーション**といいます。

関係演算は，ウィンドウ演算で抽出されたデータを処理するための演算です。このとき生成されるタプルの集合を**出力リレーション**といいます。

ストリーム化演算は，関係演算で処理されたデータをストリームデータに変換して出力するための演算です。

これに対して，ストリーム間演算は，リレーションを生成しないで直接ストリームデータに対して演算を実行し，別のストリームデータに変換します。ストリーム間演算では，入出力がストリームデータであること以外は特に規定がなく，入力されたストリームデータに対して実行する処理は任意です。ストリーム間演算関数の処理ロジックを，ユーザーが Java で記述して作成するクラスファイルにメソッドとして実装することで，任意の処理を実行できます。

このため，ウィンドウ演算，関係演算，およびストリーム化演算の組み合わせでは実現が難しかった区間集計（一定区間（時間）の発生データを定期的に集計する）などの処理が，ストリーム間演算でできるようになります。

なお，ストリーム間演算を使用する場合は，CQL でストリーム間演算関数を定義するほかに，外部定義関数の作成が必要です。外部定義関数の作成については，マニュアル「uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド」を参照してください。

それぞれの演算については，「2.2.2　操作系 CQL によるストリームデータの演算処理の指定」を参照してください。

ストリームデータ処理は，ストリームデータ処理エンジンに登録したクエリ定義ファイルの内容に従って実行されます。クエリ定義ファイルの内容については「2.2　CQL を使用したストリームデータ処理の実行」

を，クエリの記述例については「2.3　CQL を使用したストリームデータ処理の実行例」を参照してください。

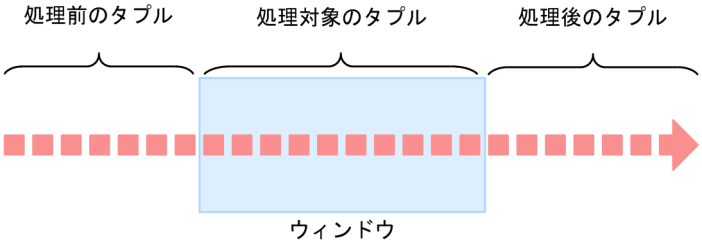
2.1.7　ウィンドウ

ウィンドウは，ストリームデータに対し，集計・分析を行うために設定する範囲です。クエリ中に定義します。

データを集計・分析するには，対象とする範囲を明確にする必要があります。ストリームデータの場合も，あらかじめ一定の範囲を決め，範囲内のデータを処理の対象にする必要があります。

ストリームデータとウィンドウの関係を次の図に示します。

図 2-6　ストリームデータとウィンドウの関係



この図に示すウィンドウで囲まれた範囲のストリームデータ（タプル）が，一時的にメモリ上に格納され，処理の対象になります。

ウィンドウでは，時間やタプルの個数などで，処理の対象となるストリームデータの範囲を指定できます。ウィンドウの指定については，「2.2.2　操作系 CQL によるストリームデータの演算処理の指定」を参照してください。

2.2　CQL を使用したストリームデータ処理の実行

ストリームデータ処理は，システムに登録したクエリ定義ファイルに従って実行されます。クエリ定義ファイルには，ストリームデータの型であるストリームおよびクエリを CQL で記述します。これらの CQL による命令を **CQL 文**といいます。

クエリ定義ファイルに記述するために使用する CQL には，次の 2 種類があります。

- **定義系 CQL**
  ストリームおよびクエリを定義するために使用する CQL
- **操作系 CQL**
  ストリームデータの演算処理で使用する CQL

ここでは，定義系 CQL によるストリームおよびクエリの定義方法，ならびに操作系 CQL によるストリームデータの演算処理の指定方法について説明します。

CQL については，マニュアル「uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド」を参照してください。

なお，CQL 文は，あらかじめ意味の与えられているキーワードと，キーワードに続けて指定する項目で構成されます。このキーワードと指定項目の組を**句**といいます。例えば，以降で説明する「REGISTER STREAM ストリームの名称」のように，「REGISTER STREAM」というキーワードと「ストリームの名称」という指定項目を合わせたものを REGISTER STREAM 句といいます。

2.2.1　定義系 CQL によるストリームおよびクエリの定義

ストリームおよびクエリの定義に使用する CQL を**定義系 CQL** といいます。定義系 CQL には，次の 2 種類があります。

- REGISTER STREAM 句
- REGISTER QUERY 句

以降で，それぞれの句の指定方法について説明します。

(1)　ストリームの定義（REGISTER STREAM 句）

ストリームの定義には，定義系 CQL の REGISTER STREAM 句を使用します。

REGISTER STREAM 句には，ストリームの名称（入力ストリームの名称），スキーマ指定文字列（ストリームに従って流れるストリームデータの形式）を指定します。REGISTER STREAM 句の指定方法を次に示します。

```
REGISTER STREAM ストリームの名称
(スキーマ指定文字列);
```

例として，気温分析システムのストリームを定義する CQL 文を次に示します。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
```

ストリームの名称には，「temperature_stream」を指定しています。また，「observation_time」（観測時刻），「id」（観測所の ID），および「temperature」（気温）のデータを，それぞれ「TIME 型」，「INTEGER 型」，および「INTEGER 型」のデータ型で指定しています。

(2)　クエリの定義（REGISTER QUERY 句）

クエリの定義には，定義系 CQL の REGISTER QUERY 句を使用します。REGISTER QUERY 句には，クエリの名称，ストリーム句，SELECT 句，FROM 句，WHERE 句の順序で記述します。なお，ここで指定するクエリの名称は，クエリによって処理されたストリームデータが流れるストリーム（出力ストリーム）の名称になります。

REGISTER QUERY 句中のストリーム句，SELECT 句，FROM 句，および WHERE 句の CQL 文の句とクエリの動作は，次の表のように対応します。

表 2-1　CQL 文の句とクエリの動作の対応

CQL 文の句	クエリの動作
ストリーム句	ストリーム化演算の動作を指定
SELECT 句，WHERE 句	関係演算の動作を指定
FROM 句	ウィンドウ演算の動作を指定

REGISTER QUERY 句の指定方法を次に示します。

```
REGISTER QUERY クエリの名称
ストリーム句（
SELECT句
FROM句
WHERE句);
```

REGISTER QUERY 句の指定内容について説明します。

ストリーム句

ストリームデータの出力方法を指定します。目的に応じて，ISTREAM 句，DSTREAM 句，RSTREAM 句のどれかを指定します。これらの句については，「2.2.2(3)　ストリーム化演算（データの処理結果の出力）」を参照してください。

SELECT 句

演算処理の対象となったタプルから抽出するデータを指定します。抽出するデータは，REGISTER STREAM 句のスキーマ指定文字列で指定したデータの名称で指定します。なお，演算処理の対象とするタプルは，WHERE 句で指定します。

FROM 句

クエリによる処理の対象となるストリームの名称と，ウィンドウを指定します。ウィンドウには，目的に対応した，ROWS ウィンドウ，RANGE ウィンドウ，NOW ウィンドウ，PARTITION BY ウィンドウの 4 種類があります。

ストリームデータは，REGISTER STREAM 句で指定した名称で指定します。ウィンドウは，ストリームの名称に続けて，半角角括弧（[ ]）で囲んで指定します。ウィンドウについては，「2.2.2(1)　ウィンドウ演算（分析対象データの抽出）」を参照してください。

クエリによる処理の対象がリレーションの場合は，リレーションの名称を指定します。

WHERE 句

WHERE 句には，ウィンドウで抽出されたタプルから，演算処理の対象を選択するための選択条件を指定します。

例として，「(1)　ストリームの定義（REGISTER STREAM 句）」で説明した気温分析システムに続けてクエリを定義します。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
REGISTER QUERY sensor_filter
ISTREAM (
SELECT id, temperature
FROM temperature_stream[ROWS 3]
WHERE id = 1);
```

REGISTER QUERY 句の指定内容について説明します。

- クエリの名称

  クエリの名称として，「sensor_filter」を指定しています。クエリによって処理されたストリームデータが流れるストリームの名称になります。

- ストリーム句

  ストリーム化演算として，「ISTREAM」を指定しています。

- SELECT 句

  WHERE 句で選択されたタプルから抽出するデータとして，「id」，および「temperature」を指定しています。これらは，REGISTER STREAM 句のスキーマ指定文字列で指定したデータの名称です。

- FROM 句

  クエリによる処理の対象となるストリームの名称，およびウィンドウとして，「temperature_stream[ROWS 3]」を指定しています。ストリームの名称は，REGISTER STREAM 句で指定した名称です。また，ウィンドウには，「[ROWS 3]」を指定しています。これは，指定したストリームを流れるストリームデータについて，最新のタプルから順番に 3 個のタプルを抽出することを示します。

- WHERE 句

  WHERE 句には，ウィンドウで抽出されたタプルから，演算処理の対象を選択するための選択条件として，「id ＝ 1」を指定しています。これは，ウィンドウで抽出されたタプルから，ID が 1 のタプルを選択することを示します。

なお，REGISTER QUERY 句には，ここで説明した以外の句も指定できます。REGISTER QUERY 句に指定できる句については，マニュアル「uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド」を参照してください。

2.2.2　操作系 CQL によるストリームデータの演算処理の指定

操作系 CQL には，次の 4 種類があります。

- ウィンドウ演算
- 関係演算
- ストリーム化演算
- ストリーム間演算

ここでは，ウィンドウ演算，関係演算，およびストリーム化演算の 3 種類の演算について，「2.2.1(1)　ストリームの定義（REGISTER STREAM 句)」で説明した気温分析システムを例に，以降で説明します。

ストリーム間演算については，マニュアル「uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド」を参照してください。

(1)　ウィンドウ演算（分析対象データの抽出）

ウィンドウ演算は，ストリームデータから分析対象のデータを抽出するために使用します。クエリ中では，FROM 句でのストリームの名称に続けてウィンドウを指定します。ウィンドウ演算には，次の 4 種類があります。

- ROWS ウィンドウ
- RANGE ウィンドウ
- NOW ウィンドウ
- PARTITION BY ウィンドウ

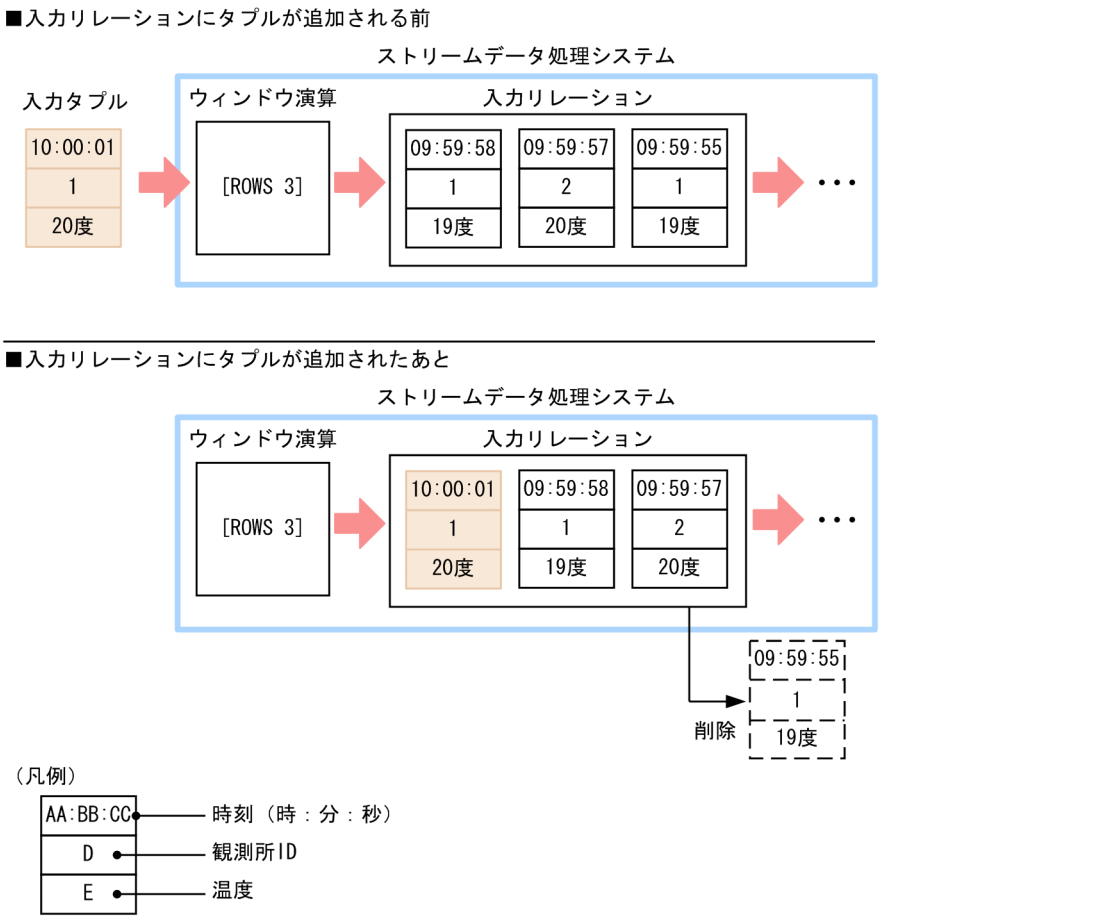
それぞれのウィンドウ演算について，以降に説明します。

ROWS ウィンドウ

タプルの個数でストリームデータから抽出するタプルを指定するウィンドウです。ROWS ウィンドウで生成される入力リレーションは，最新のタプルから指定の個数までさかのぼったタプルの集合です。ROWS ウィンドウでは，タプルを受信するごとに新たなタプルが入力リレーションに追加されます。また，不要になったタプルは，入力リレーションから削除されます。

ROWS ウィンドウでは，抽出するタプルの個数を指定します。例えば，「[ROWS 3]」と指定すると，最新のタプルから順番に 3 個のタプルが抽出され，入力リレーションとなります。ROWS ウィンドウによる分析対象データの抽出を次の図に示します。

図 2-7　ROWS ウィンドウによる分析対象データの抽出



この図に示したウィンドウ演算では，抽出するタプルを 3 個に指定しているため，入力リレーションには 3 個のタプルが保持されます。タプルの保持数の上限を超えないよう，入力タプルが入力リレーションに追加されたタイミングで，入力リレーション内の最も古いタプルが削除されます。

RANGE ウィンドウ

時間でストリームデータから抽出するタプルを指定するウィンドウです。RANGE ウィンドウで生成される入力リレーションは，最新のタプルから指定された時間までさかのぼったタプルの集合です。

RANGE ウィンドウでは，タプルを抽出する時間を指定します。例えば，「[RANGE 3 SECOND]」と指定すると，最新のタプルから 3 秒以内のタイムスタンプが設定されているすべてのタプルが抽出され，入力リレーションとなります。

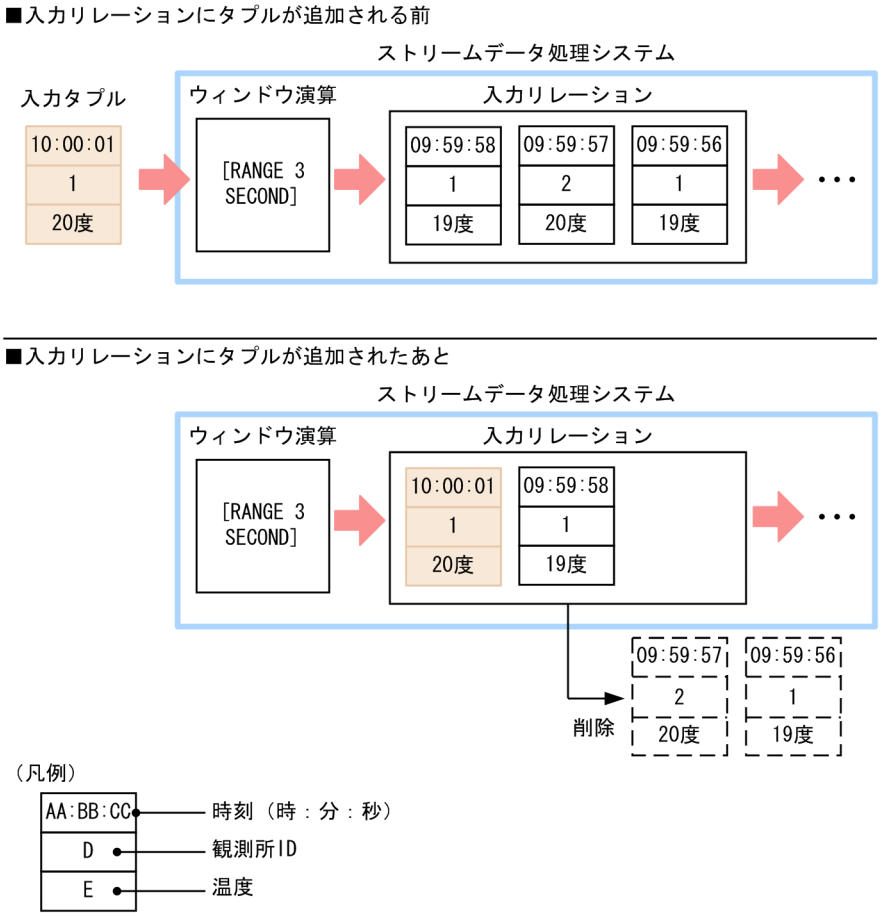
時間の指定には，次の表に示す単位を指定できます。

表 2–2　CQL 文に指定できる時間の単位

CQL 文の指定	単位
MILLISECOND	ミリ秒
SECOND	秒
MINUTE	分
HOUR	時
DAY	日

RANGE ウィンドウによる分析対象データの抽出を次の図に示します。

図 2-8　RANGE ウィンドウによる分析対象データの抽出



この図に示したウィンドウ演算では，最新のタプルから 3 秒以内（10:00:01～09:59:58）のタイムスタンプが設定されているすべてのタプルを抽出するよう指定しています。そのため，この条件に該当しないタプルは，入力タプルが入力リレーションに追加されたタイミングで，入力リレーションから削除されます。

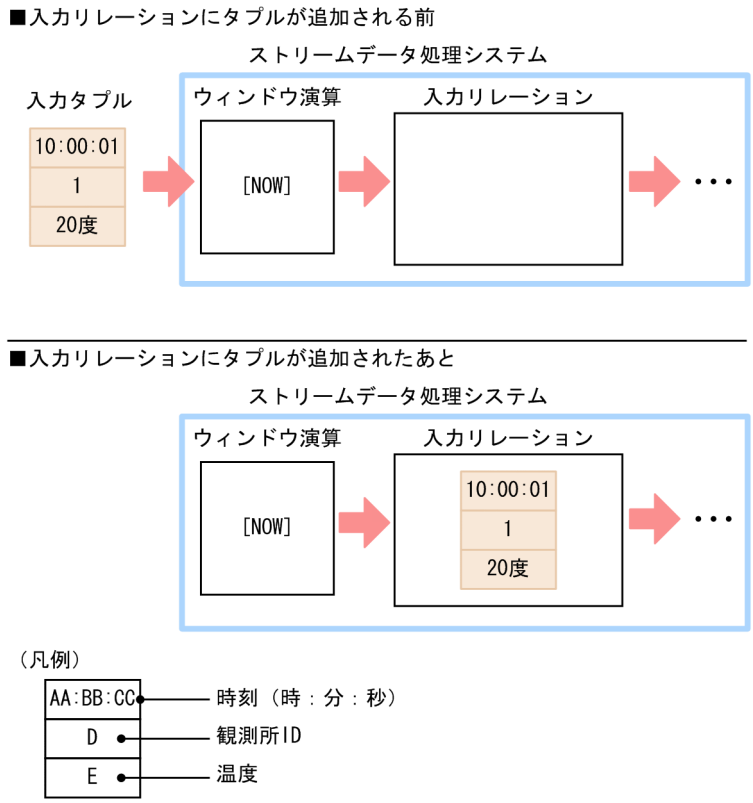
なお，RANGE ウィンドウを使用する場合，入力データによっては，Stream Data Platform - AF 上で扱うタプルの数が膨大になり，メモリ使用量を増大させることがあります。このとき，時刻解像度機能を使用すると，メモリ使用量の増加を防止できます。時刻解像度機能については，マニュアル「uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド」を参照してください。

NOW ウィンドウ

その時点で到着したタプルだけを処理する場合に指定するウィンドウです。タイムスタンプが同一の複数のタプルが同時に到着した場合は，それらのすべてのタプルが処理の対象となります。NOW ウィンドウで演算処理されたタプルは削除されるため，NOW ウィンドウで生成される入力リレーションには，最新のタプルだけが存在します。

NOW ウィンドウは，「[NOW]」と指定します。NOW ウィンドウによる分析対象データの抽出を次の図に示します。

図 2-9　NOW ウィンドウによる分析対象データの抽出



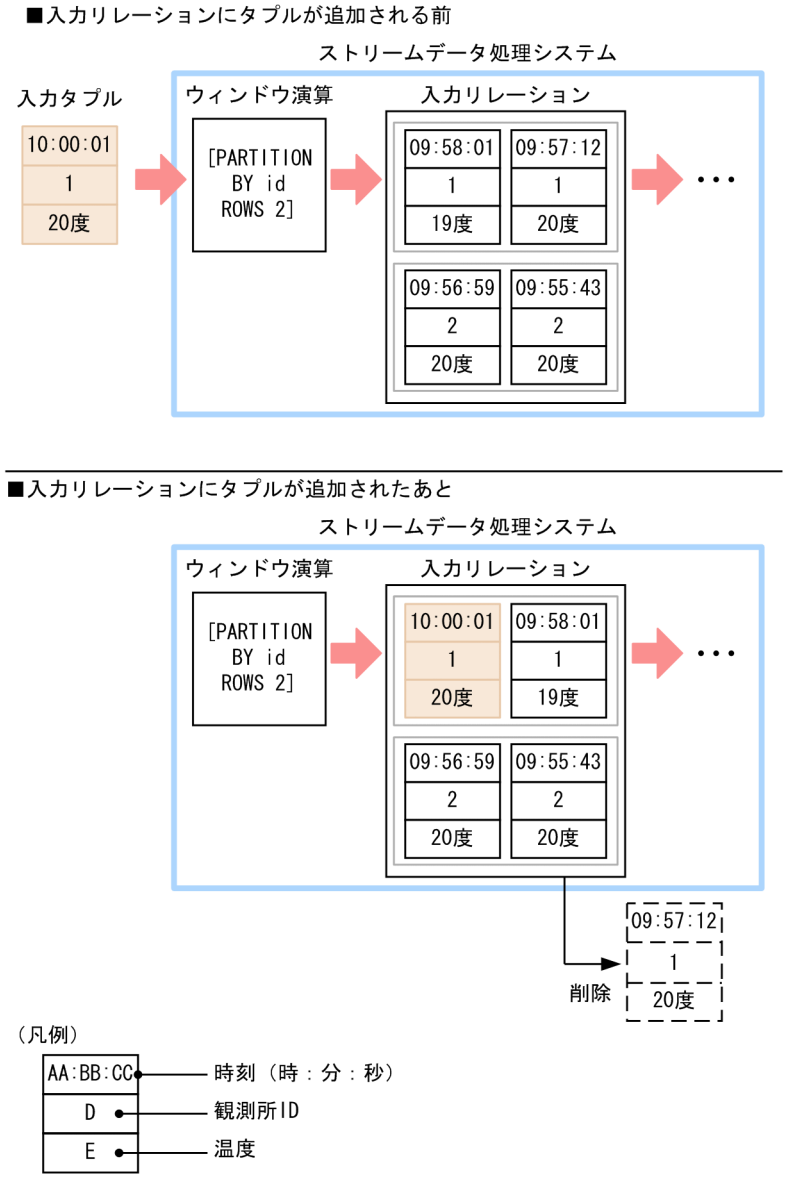
この図に示したウィンドウ演算では，その時点で到着したタプルだけを処理の対象とします。入力リレーション内のタプルは，演算処理後に削除されるため，タプルが追加されるタイミングでは，入力リレーション内にタプルはありません。

PARTITION BY ウィンドウ

データの値ごとにストリームデータから抽出するタプルを指定するウィンドウです。ROWS ウィンドウと組み合わせて使用します。PARTITION BY ウィンドウで生成される入力リレーションは，指定したデータの最新のタプルから，指定の個数までさかのぼったタプルの集合です。

PARTITION BY ウィンドウでは，グルーピングに使用するデータの名称を指定します。さらに，グループごとに抽出するタプルの個数を「ROWS」に続けて指定します。例えば，「[PARTITION BY id ROWS 2]」と指定すると，ID ごとに，最新のタプルから順番に 2 個のタプルが抽出され，入力リレーションとなります。PARTITION BY ウィンドウによる分析対象データの抽出を次の図に示します。

図 2-10　PARTITION BY ウィンドウによる分析対象データの抽出



この図に示したウィンドウ演算では，タプルの中段に指定された ID ごとに，2 個のタプルを抽出するように指定しています。そのため，入力タプルが入力リレーションに追加されたタイミングで，入力タプルと同じ ID で，かつ最も古いタプルが入力リレーションから削除されます。

(2)　関係演算（抽出したデータの処理）

関係演算は，ウィンドウ演算で抽出されたデータを処理するために使用します。次に示すような処理ができます。

- 条件を満たすデータの抽出
- データの計算
- データの集計
- データの分類後の集計
- ストリームデータの結合

- クエリの連結

これらの処理は，SELECT 句や WHERE 句に，四則演算子，比較演算子，論理演算子，および集合関数を使用して指定します。

指定できる比較演算子および集合関数を次の表に示します。

表 2-3　CQL 文に指定できる比較演算子

比較演算子	比較演算子の使用例	使用例の意味
<=	A <= B	A は B 以下
>=	A >= B	A は B 以上
<	A < B	A は B より小さい
>	A > B	A は B より大きい
=	A = B	A は B と等しい
!=	A != B	A は B と等しくない

表 2-4　CQL 文に指定できる集合関数

関数名※	説明
AVG	平均値の計算
COUNT	個数の計算
MAX	最大値の計算
MIN	最小値の計算
SUM	合計値の計算

注※

集合関数には，この表に示す関数のほかに，統計関数を提供する組み込み集合関数もあります。

なお，CQL 文で指定できる論理演算子は，句ごとに異なります。

CQL 文で指定できる演算子や関数の詳細については，マニュアル「uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド」を参照してください。

関係演算の実行例については，「2.3　CQL を使用したストリームデータ処理の実行例」を参照してください。

(3)　ストリーム化演算（データの処理結果の出力）

ストリーム化演算は，関係演算での処理結果をストリームデータに変換して出力するために使用します。REGISTER QUERY 句の直後に指定するストリーム句で指定します。ストリーム化演算には，次の 3 種類があります。

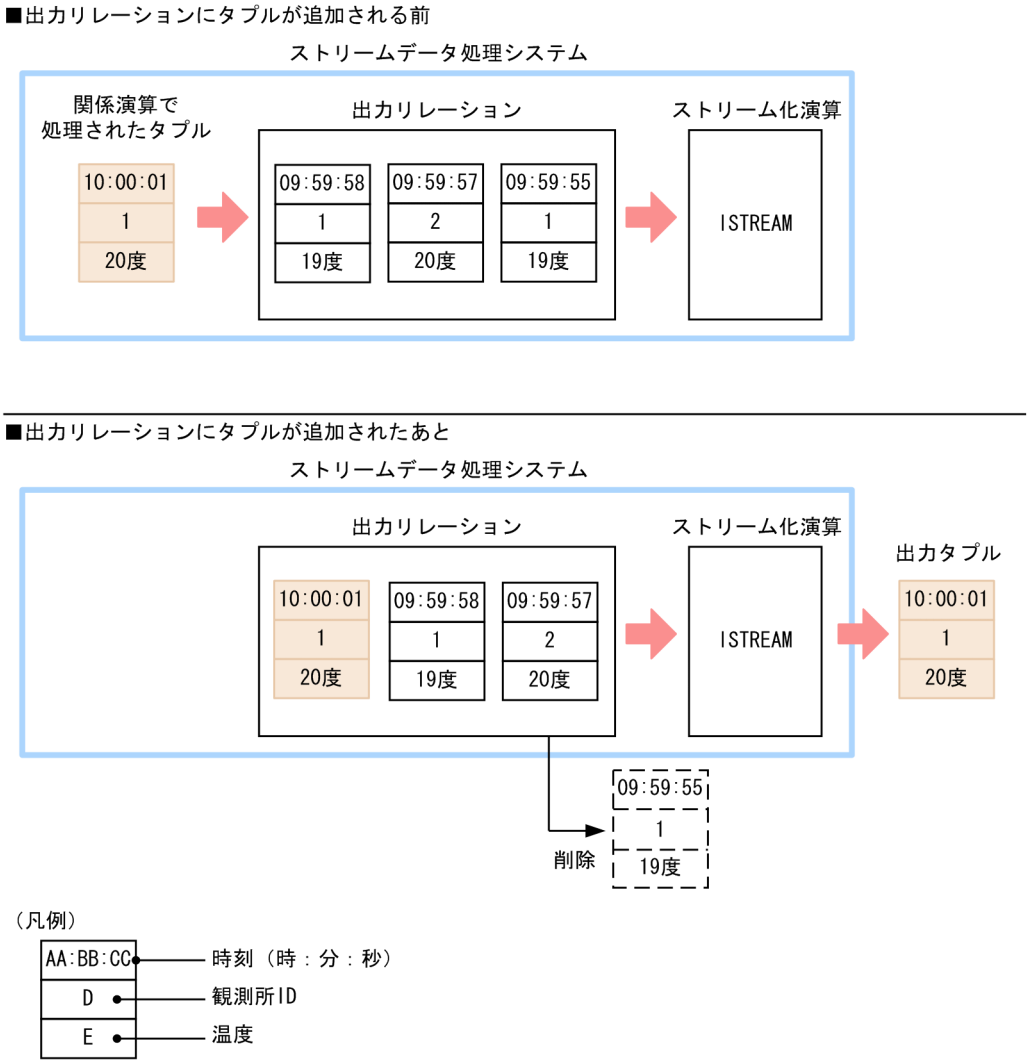
- ISTREAM 句
- DSTREAM 句
- RSTREAM 句

それぞれのストリーム化演算について，ウィンドウ演算で[ROWS 3]を指定した場合を例に，以降に説明します。

ISTREAM 句

新たに出力リレーションに追加されたタプルを出力するストリーム化演算です。ISTREAM 句は，出力リレーションが変化するごとに，変化前・変化後の出力リレーションを比較して新たに追加されたタプルを出力します。ISTREAM 句による処理結果の出力を次の図に示します。

図 2-11　ISTREAM 句による処理結果の出力

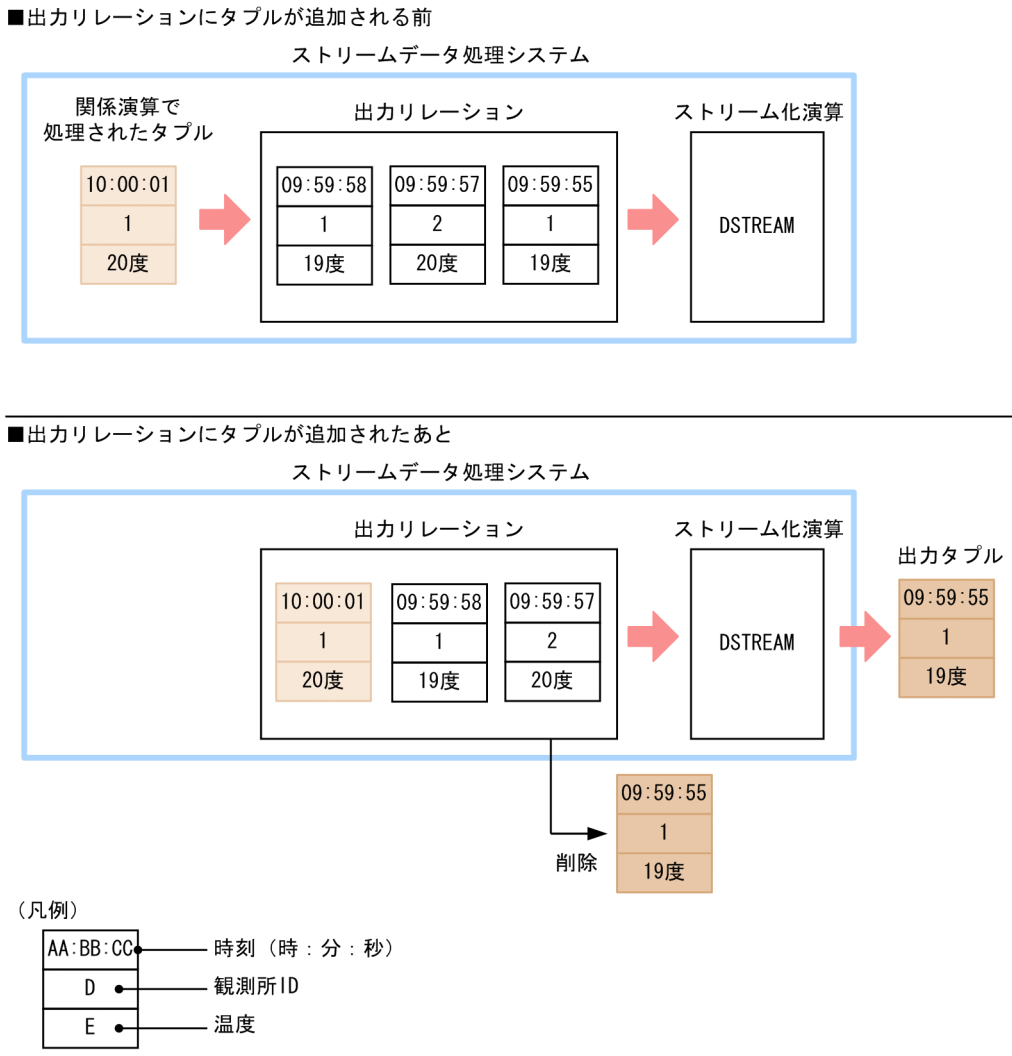


出力リレーションに，関係演算で処理されたタプルが追加されたため，出力リレーション内の最も古いタプルが削除されます。このとき，ストリーム化演算では「ISTREAM」を指定しているため，出力リレーションに追加されたタプルが出力タプルとなります。

DSTREAM 句

出力リレーションから削除されたタプルを出力するストリーム化演算です。DSTREAM 句は，出力リレーションが変化するごとに変化前・変化後の出力リレーションを比較し，削除されたタプルを出力します。DSTREAM 句による処理結果の出力を次の図に示します。

図 2-12　DSTREAM 句による処理結果の出力

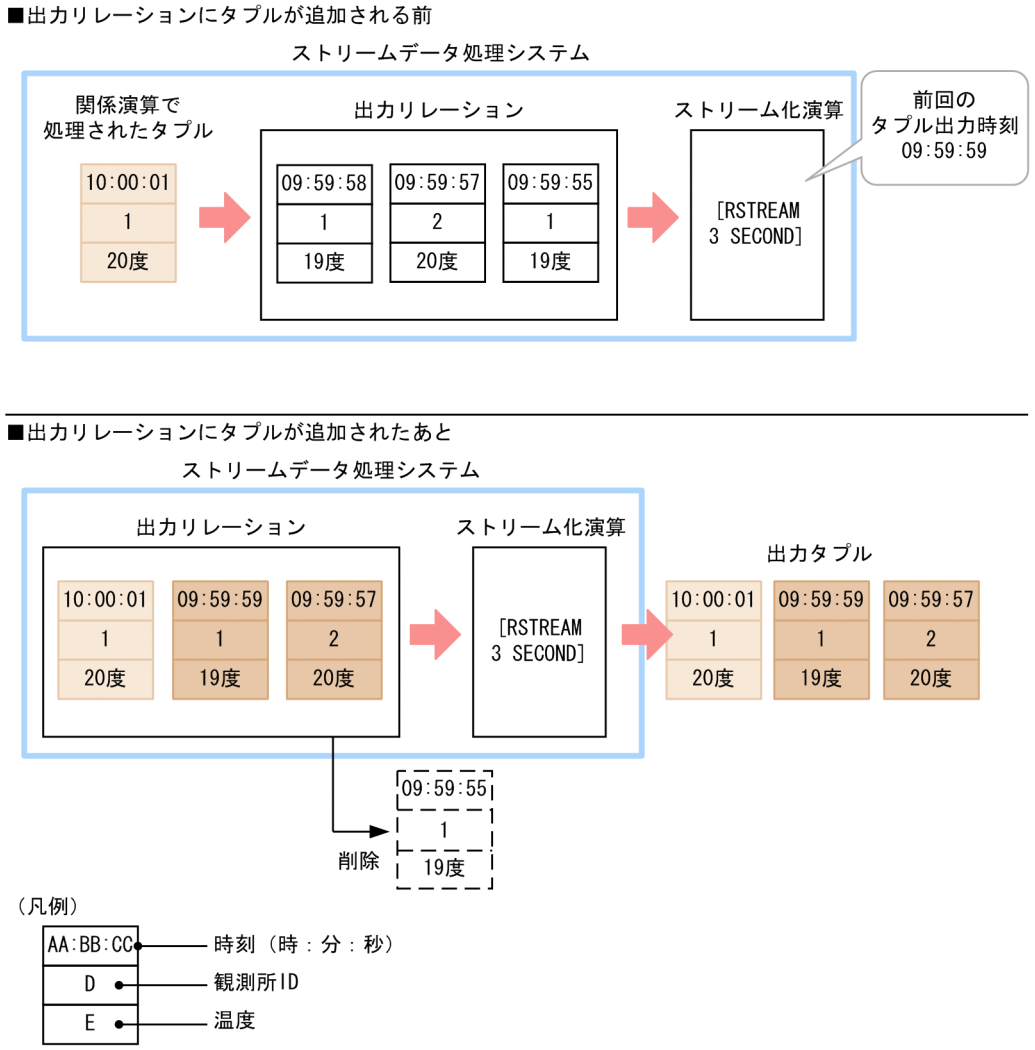


出力リレーションに，関係演算で処理されたタプルが追加されたため，出力リレーション内の最も古いタプルが削除されます。このとき，ストリーム化演算では「DSTREAM」を指定しているため，出力リレーションから削除されたタプルが出力タプルとなります。

RSTREAM 句

一定時間ごとに，出力リレーション内のすべてのタプルを出力するストリーム化演算です。RSTREAM 句を指定する場合は，ストリームデータを出力する時間間隔を半角角括弧（[ ]）で囲みます。例えば，「[RSTREAM 1 MINUTE]」や「[RSTREAM 3 SECOND]」と指定します。時間間隔には，「表 2-2 CQL 文に指定できる時間の単位」に示す単位を使用できます。RSTREAM 句による処理結果の出力を次の図に示します。

図 2-13　RSTREAM 句による処理結果の出力



ストリーム化演算で「RSTREAM 3 SECOND」を指定しているため，システム時刻を基準に，3 秒ごとの出力リレーション内のすべてタプルが出力タプルとなります。

2.3　CQL を使用したストリームデータ処理の実行例

ここでは，「2.2　CQL を使用したストリームデータ処理の実行」で説明した内容の実行例を説明します。

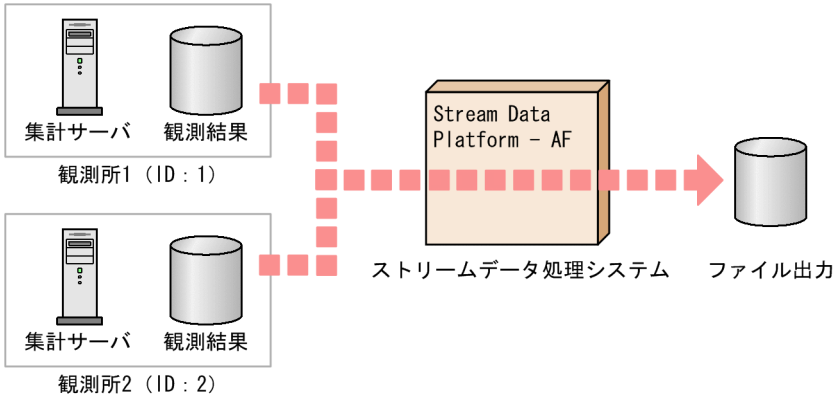
気温分析システムを例に，クエリの内容によってどのようにストリームデータ処理が実行されるかを説明します。この例で実行するのは，次の処理です。

- 条件を満たすデータの抽出
- データの計算
- データの集計
- データの分類と集計
- ストリームデータの結合
- クエリの連結

2.3.1　想定するシステム

実行例で想定するシステムを次の図に示します。

図 2-14　ストリームデータ処理の実行例で想定する気温分析システム



この気温分析システムでは，観測所 1（ID：1），観測所 2（ID：2）でそれぞれ気温を観測し，観測結果（ファイル）を逐次ストリームデータ処理システムに送信します。ストリームデータ処理システムは，送信されてきたデータをクエリに従い集計・分析し，その結果をファイルに出力します。

2.3.2　条件を満たすデータの抽出

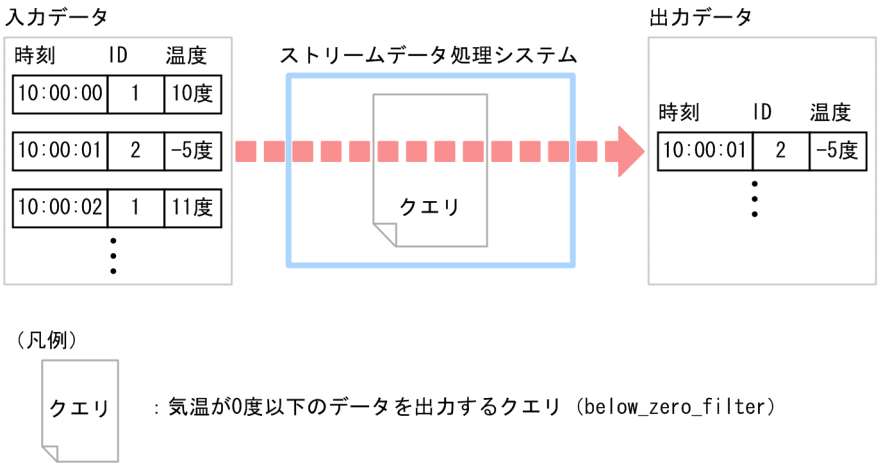
条件を満たすデータの抽出処理の実行例を次の二つのクエリで説明します。

- 気温が 0 度以下のデータを出力するクエリ
- 気温が-10 度以上，かつ 0 度以下のデータを出力するクエリ

(1)　気温が 0 度以下のデータを出力するクエリ

「気温が 0 度以下のデータを出力する」という，一つの条件を満たすデータを出力するクエリについて説明します。このクエリを実行した場合の，入力データと出力データを次の図に示します。

図 2-15　一つの条件を満たすデータを出力するクエリを実行した場合の入力データと出力データ



クエリの内容

気温が 0 度以下という条件を満たすデータを出力する場合，比較演算子を使用して WHERE 句にデータの選択条件を指定します。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
REGISTER QUERY below_zero_filter
ISTREAM (
SELECT observation_time, id, temperature
FROM temperature_stream[ROWS 1]
WHERE temperature <= 0);
```

クエリの解説

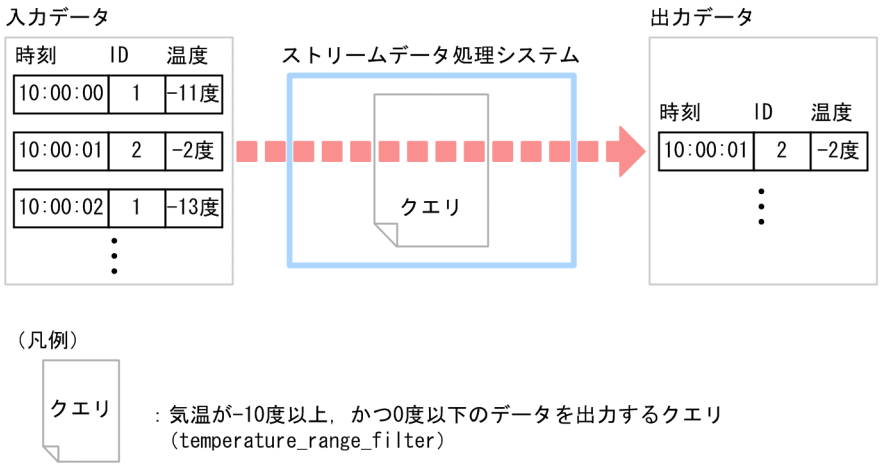
このクエリの処理対象は，最新の一つのタプルとします。FROM 句には，タプルの個数でストリームデータからタプルを抽出する ROWS ウィンドウを指定します。

出力するデータは，気温が 0 度以下のデータです。WHERE 句には，「temperature <= 0」を指定します。これは，temperature の値と 0 を比較し，temperature の値が 0 以下のタプルを選択することを示します。

(2)　気温が-10 度以上，かつ 0 度以下のデータを出力するクエリ

「気温が-10 度以上，かつ 0 度以下のデータを出力する」という，二つの条件を満たすデータを出力するクエリについて説明します。このクエリを実行した場合の，入力データと出力データを次の図に示します。

図 2-16　二つの条件を満たすデータを出力するクエリを実行した場合の入力データと出力データ



クエリの内容

気温が-10 度以上，かつ 0 度以下という条件を満たすデータを出力する場合，比較演算子と論理演算子 AND を使用して，WHERE 句にデータの選択条件を指定します。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
REGISTER QUERY temperature_range_filter
ISTREAM (
SELECT observation_time, id, temperature
FROM temperature_stream[ROWS 1]
WHERE -10 <= temperature AND temperature <= 0);
```

クエリの解説

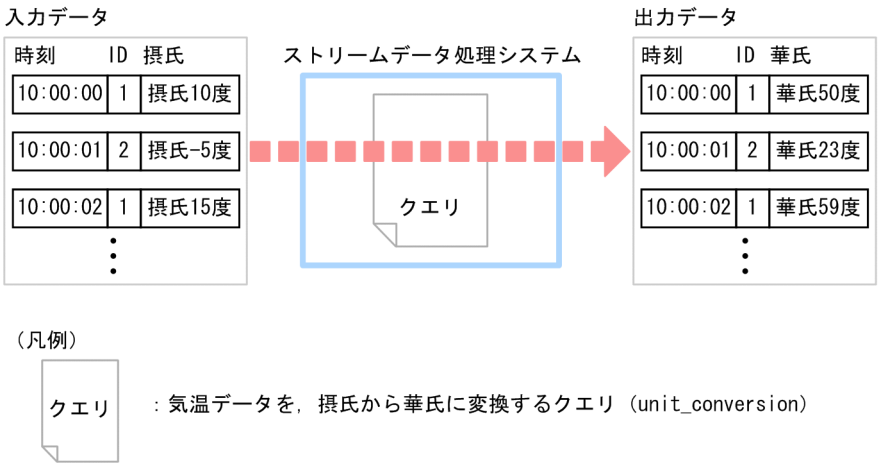
このクエリの処理対象は，最新の一つのタプルとします。FROM 句には，タプルの個数でストリームデータからタプルを抽出する ROWS ウィンドウを指定します。

出力するデータは，気温が-10 度以上，0 度以下のデータです。WHERE 句には，「-10 <= temperature」と「temperature <= 0」の条件を連結して指定します。これは，temperature の値と-10 を比較し，temperature の値が-10 以上であり，かつ temperature の値と 0 を比較し，temperature の値が 0 以下のタプルを選択することを示します。

2.3.3　データの計算

データを計算するクエリについて，摂氏で観測した気温データを華氏に変換する場合を例に説明します。摂氏から華氏へ変換するには，クエリ中に計算式を記述する必要があります。このクエリを実行した場合の，入力データと出力データを次の図に示します。

図 2-17　データを計算するクエリを実行した場合の入力データと出力データ



クエリの内容

気温を摂氏から華氏に変換するには，四則演算を使用して，SELECT 句に計算式を指定します。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
REGISTER QUERY unit_conversion
ISTREAM (
SELECT observation_time, id,
temperature*9/5+32 AS fahrenheit_temperature
FROM temperature_stream[ROWS 1]);
```

クエリの解説

このクエリの処理対象は，最新の一つのタプルとします。FROM 句には，タプルの個数でストリームデータからタプルを抽出する ROWS ウィンドウを指定します。

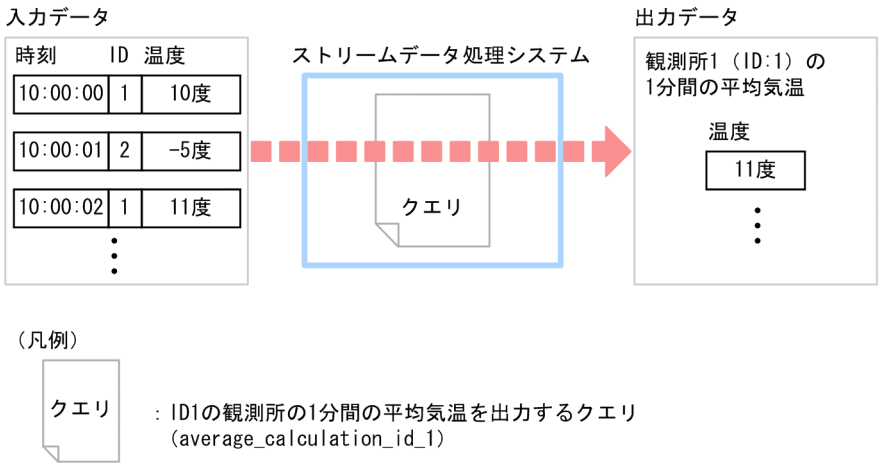
出力データは，華氏で表現された気温データです。SELECT 句には，摂氏から華氏に変換するための計算式として「temperature*9/5+32」を指定しています。

計算したデータを出力する場合，計算結果のデータに名称を与える必要があります。計算結果のデータの名称は，「AS」で指定します。CQL 中では，華氏の計算結果のデータの名称として，「fahrenheit_temperature」を指定しています。

2.3.4　データの集計

データを集計するクエリについて，ID1 の観測所の 1 分間の平均気温を出力する場合を例に説明します。この場合，まず ID1 の 1 分間の気温データを集計し，その集計結果から平均を求めます。このクエリを実行した場合の，入力データと出力データを次の図に示します。

図 2-18　データを集計するクエリを実行した場合の入力データと出力データ



クエリの内容

データを集計するには，集合関数を使用して，SELECT 句に集計方法を指定します。また，特定のデータを集計の対象とするため，比較演算子を使用して WHERE 句にデータの選択条件を指定します。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
REGISTER QUERY average_calculation_id_1
ISTREAM (
SELECT AVG(temperature) AS average_temperature
FROM temperature_stream[RANGE 1 MINUTE]
WHERE id = 1);
```

クエリの解説

このクエリの処理対象は，過去 1 分間のタプルです。FROM 句には，ストリームデータから時間でタプルを抽出する RANGE ウィンドウを指定します。

出力データは，ID1 の観測所の 1 分間の平均気温です。SELECT 句には，「temperature」を対象として平均値を計算する集合関数「AVG」と，計算結果のデータの名称として「average_temperature」を指定しています。また，この計算処理の対象として，WHERE 句には，「id ＝ 1」を指定しています。

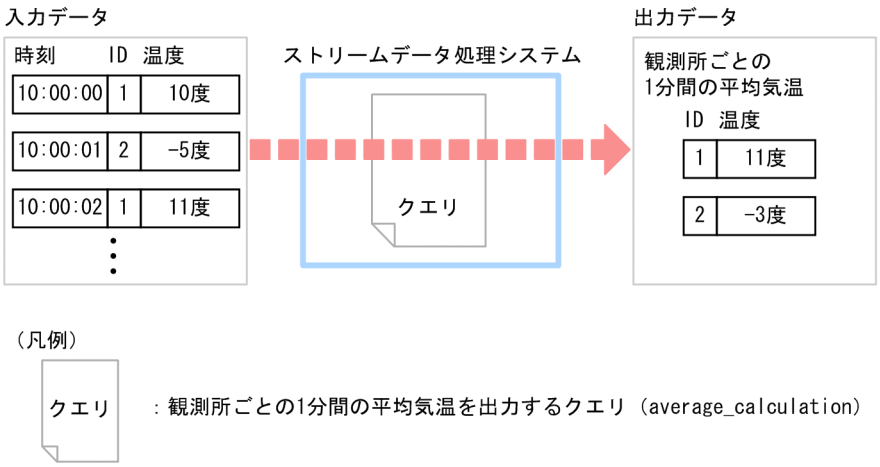
このように，SELECT 句と WHERE 句の両方で処理を実行した場合，WHERE 句の処理が最初に実行され，その結果に対して SELECT 句による処理が実行されます。

なお，先に処理が実行される WHERE 句で選択されたタプルの集合を**中間リレーション**といいます。このクエリでは，ID1 の観測所の過去 1 分間のタプルの集合が該当します。

2.3.5　データの分類と集計

複数あるデータの中から，データを分類したあとに集計をするクエリについて，観測所ごとの 1 分間の平均気温を出力する場合を例に説明します。このクエリを実行した場合の，入力データと出力データを次の図に示します。

図 2-19　データを分類したあとに集計をするクエリを実行した場合の入力データと出力データ



クエリの内容

データを分類するには，GROUP BY 句を使用して分類方法を指定します。GROUP BY 句は，WHERE 句の直後（WHERE 句を指定しない場合は FROM 句の直後）に指定します。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
REGISTER QUERY average_calculation
ISTREAM (
SELECT id, AVG(temperature) AS average_temperature
FROM temperature_stream[RANGE 1 MINUTE]
GROUP BY id);
```

クエリの解説

このクエリの処理対象は，過去 1 分間のタプルです。FROM 句には，ストリームデータから時間でタプルを抽出する RANGE ウィンドウを指定します。

出力データは，観測所ごとの 1 分間の平均気温です。SELECT 句には，「temperature」を対象として平均値を計算する集合関数「AVG」と，計算結果のデータの名称として「average_temperature」を指定しています。また，観測所ごとのデータを求めるため，GROUP BY 句に「id」を指定しています。

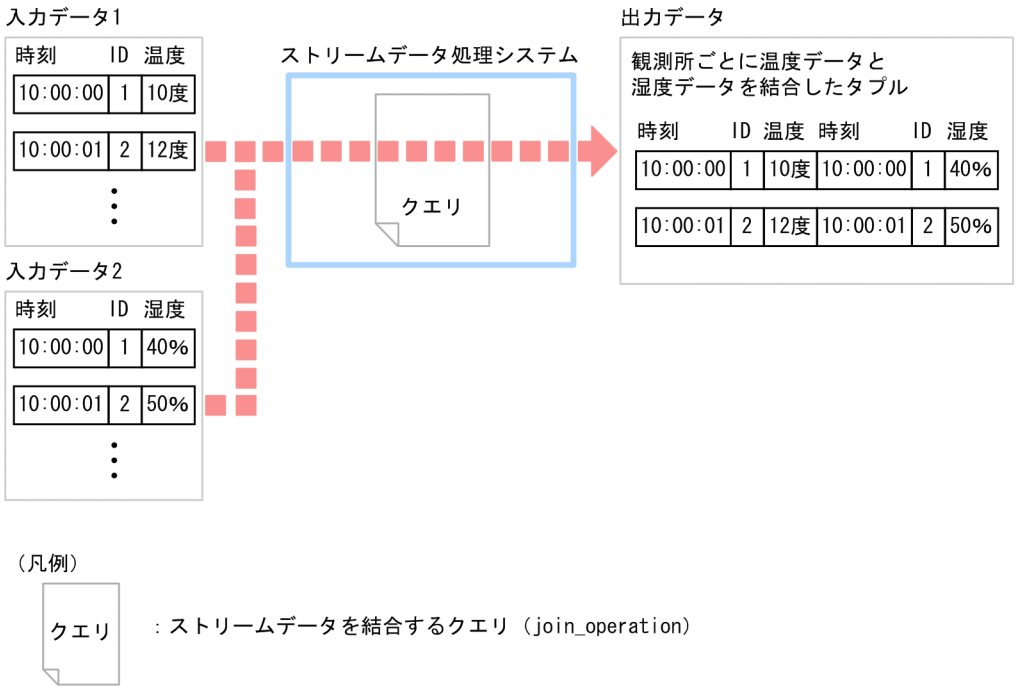
このように，SELECT 句と GROUP BY 句の両方で処理を実行した場合，GROUP BY 句で指定した分類に従って，SELECT 句による処理が実行されます。

2.3.6　ストリームデータの結合

複数のストリームデータからタプルを選択して，一つのタプルにまとめる演算処理を**ストリームデータの結合**といいます。

ストリームデータを結合するクエリについて，気温と湿度の 2 種類のストリームデータを結合し，さらに観測所ごとにタプルを結合する場合を例に説明します。このクエリを実行した場合の，入力データと出力データを次の図に示します。

図 2-20　ストリームデータを結合するクエリを実行した場合の入力データと出力データ



（凡例）

：ストリームデータを結合するクエリ（join_operation）

クエリの内容

複数のストリームデータを入力とする場合，FROM 句にコンマ（,）で区切ってストリームデータを指定します。この場合，ストリームデータごとにウィンドウ演算を指定する必要があります。また，データを結合するには，WHERE 句に結合条件を指定します。

湿度のストリームの名称は「humidity_stream」，データの名称は「humidity」とします。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
REGISTER STREAM humidity_stream
(observation_time TIME, id INTEGER, humidity INTEGER);
REGISTER QUERY join_operation
ISTREAM (
SELECT temperature_stream.observation_time AS temperature_stream_time,
temperature_stream.id AS temperature_stream_id,
temperature_stream.temperature,
humidity_stream.observation_time AS humidity_stream_time,
humidity_stream.id AS humidity_stream_id,
humidity_stream.humidity
FROM temperature_stream[PARTITION BY id ROWS 1],
humidity_stream[PARTITION BY id ROWS 1]
WHERE temperature_stream.id = humidity_stream.id);
```

クエリの解説

このクエリの処理対象は，観測所ごとの最新の一つのタプルです。CQL 文では観測所ごとに最新の一つのタプルを抽出する PARTITION BY ウィンドウを指定しています。

出力データは，気温と湿度の 2 種類のストリームデータを結合し，さらに観測所ごとにタプルを結合したデータです。WHERE 句には「temperature_stream.id = humidity_stream.id」と指定し，ID が同じタプルを結合する条件を指定しています。

また，SELECT 句には，出力するデータの名称として，結合したタプルのすべてのデータを指定しています。複数のストリームデータを入力とする場合，異なるストリームデータでデータの名称が重複していることがあります。そのときは，ストリームの名称とデータの名称をピリオド（.）で区切って，どのストリームデータのデータであるのかを明示します。また，データの重複を避けるため，「AS」でデータの名称を指定しています。

2.3.7　クエリの連結

目的の処理の実行が，一つのクエリでは実現できないことがあります。この場合，一つ目（前段）のクエリの処理結果を二つ目（後段）のクエリの処理対象とします。これを**クエリの連結**といいます。クエリの連結には，次の 2 種類があります。

- ストリームデータによるクエリの連結
- リレーションによるクエリの連結

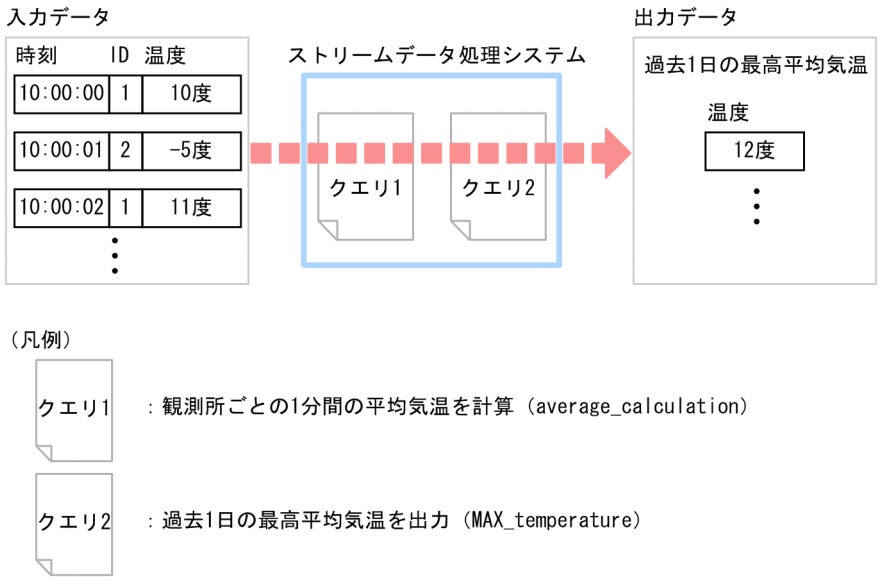
クエリを連結する場合に，前段のクエリと後段のクエリで処理対象のデータが異なることがあります。例えば，前段のクエリで 1 分間のデータを分析対象にし，後段のクエリで 1 時間分のデータを分析対象にするときは，ストリームデータによるクエリの連結が適しています。一方，前段のクエリと後段のクエリで，分析対象のデータが同じ場合は，リレーションによるクエリの連結が適しています。

以降で，それぞれのクエリの連結について説明します。

(1)　ストリームデータによるクエリの連結

ストリームデータによるクエリの連結について，観測所ごとの 1 分間の平均気温を計算し，過去 1 日の最高平均気温を出力する場合を例に説明します。このクエリを実行した場合の，入力データと出力データを次の図に示します。

図 2-21　ストリームデータによるクエリの連結を実行した場合の入力データと出力データ



前段のクエリ（クエリ 1）では観測所ごとの 1 分間の平均気温を計算し，後段のクエリ（クエリ 2）では過去 1 日の最高平均気温を計算します。

クエリの内容

クエリを連結する場合，後段のクエリの FROM 句に，前段のクエリの名称を指定します。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
REGISTER QUERY average_calculation
ISTREAM (
SELECT id, AVG(temperature) AS average_temperature
FROM temperature_stream[RANGE 1 MINUTE]
GROUP BY id);
REGISTER QUERY MAX_temperature
```

```
ISTREAM (
SELECT MAX(average_temperature)
FROM average_calculation[RANGE 1 DAY] );
```

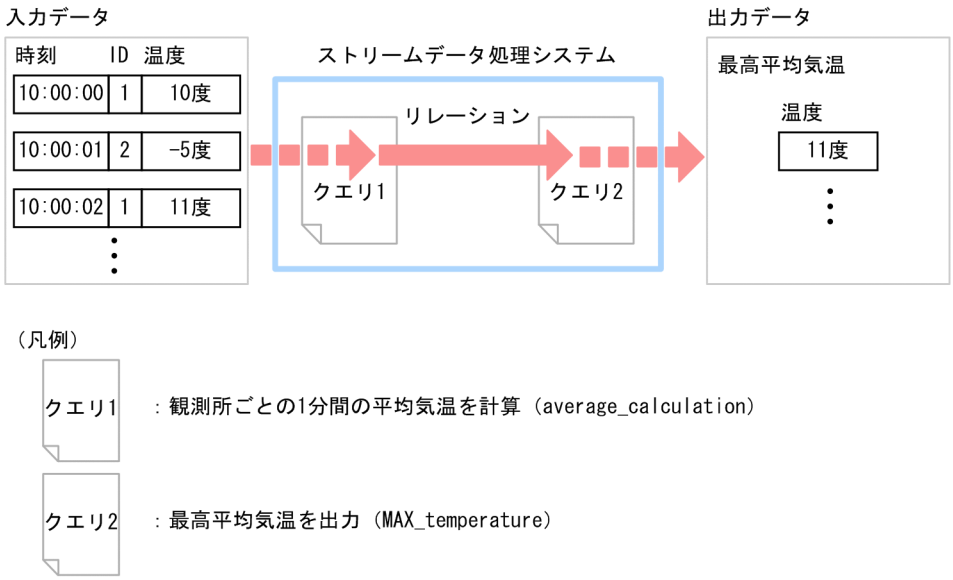
クエリの解説

前段のクエリ（average_calculation）は，「2.3.5　データの分類と集計」で説明したクエリと同じです。一方，後段のクエリ（MAX_temperature）は，前段のクエリが出力したストリームデータを入力とし，ウィンドウ演算で分析対象のタプルを抽出します。後段のクエリの分析対象は，過去 1 日のタプルです。そのため，後段のクエリの FROM 句には，「[RANGE 1 DAY]」を指定しています。

(2)　リレーションによるクエリの連結

リレーションによるクエリの連結について，観測所ごとの平均気温を計算し，現時点で最も高い平均気温（最高平均気温）を出力する場合を例に説明します。このクエリを実行した場合の，入力データと出力データを次の図に示します。

図 2-22　リレーションによるクエリの連結を実行した場合の入力データと出力データ



この図に示すとおり，前段のクエリ（クエリ 1）では観測所ごとの 1 分間の平均気温を計算し，後段のクエリ（クエリ 2）では最高平均気温を計算します。

クエリの内容

リレーションによるクエリの連結では，前段のクエリでのストリーム化演算，および後段のクエリでのウィンドウ演算を使用しません。後段のクエリの FROM 句には，前段のクエリの名称を指定します。

なお，ストリーム句を使用しない場合，SELECT 句以降を半角丸括弧（( )）で囲む必要はありません。

```
REGISTER STREAM temperature_stream
(observation_time TIME, id INTEGER, temperature INTEGER);
REGISTER QUERY average_calculation
SELECT id, AVG(temperature) AS average_temperature
FROM temperature_stream[RANGE 1 MINUTE]
GROUP BY id;
REGISTER QUERY MAX_temperature
ISTREAM (
SELECT MAX(average_temperature) AS MAX_temperature
FROM average_calculation);
```

クエリの解説

リレーションによるクエリの連結では，前段のクエリ（average_calculation）の出力リレーション（観測所ごとの平均気温）と，後段のクエリ（MAX_temperature）の入力リレーション（観測所ごとの平均気温）は同じ内容です。

3 データの送受信

この章では, 入出力データとストリームデータ処理エンジン間でのデータの送受信について説明します。なお, データの送受信に使用するアダプターについては，Stream Data Platform - AF が提供する標準提供アダプターを中心に説明します。

3.1　データの送受信に使用するアダプターの種類

Stream Data Platform - AF では，入出力データとストリームデータ処理エンジン間でのデータの送受信に，**アダプター**を使用します。アダプターでは，入出力データの形式の変換や，データの絞り込み（フィルタリング）などを実行できます。

アダプターは，ストリームデータ処理エンジンと連携させて使用します。アダプターとストリームデータ処理エンジンとの連携方法には，次の 2 種類があります。

- **インプロセス連携**
  アダプターとストリームデータ処理エンジンが同一のプロセスで動作する連携方式です。
- **RMI 連携**
  アダプターとストリームデータ処理エンジンが別々のプロセスで動作し，Java の RMI を使用して連携する連携方式です。

これらの連携については，マニュアル「uCosminexus Stream Data Platform - Application Framework システム構築・運用ガイド」を参照してください。

また，アダプターには，**標準提供アダプター**と，ユーザーが Java でプログラミングして作成する**カスタムアダプター**があります。ここでは，それぞれのアダプターについて説明します。

なお，それぞれのアダプターで，入力データとストリームデータ処理エンジン間の送受信で使用するアダプターを**入力アダプター**といい，ストリームデータ処理エンジンと出力データ間の送受信で使用するアダプターを**出力アダプター**といいます。

3.1.1　標準提供アダプター

Stream Data Platform - AF が提供するアダプターを**標準提供アダプター**といいます。標準提供アダプターでは，アダプターとして必要な機能をアダプター用定義ファイルの定義で実現できます。アダプター用定義ファイルについては，マニュアル「uCosminexus Stream Data Platform - Application Framework システム構築・運用ガイド」を参照してください。

標準提供アダプターを使用する場合，ストリームデータ処理エンジンには，次のデータを入力できます。

- ファイル
- HTTP パケット

また，標準提供アダプターでは，ストリームデータ処理エンジンでの処理結果を次の出力先に出力できます。

- ファイル
- ダッシュボード

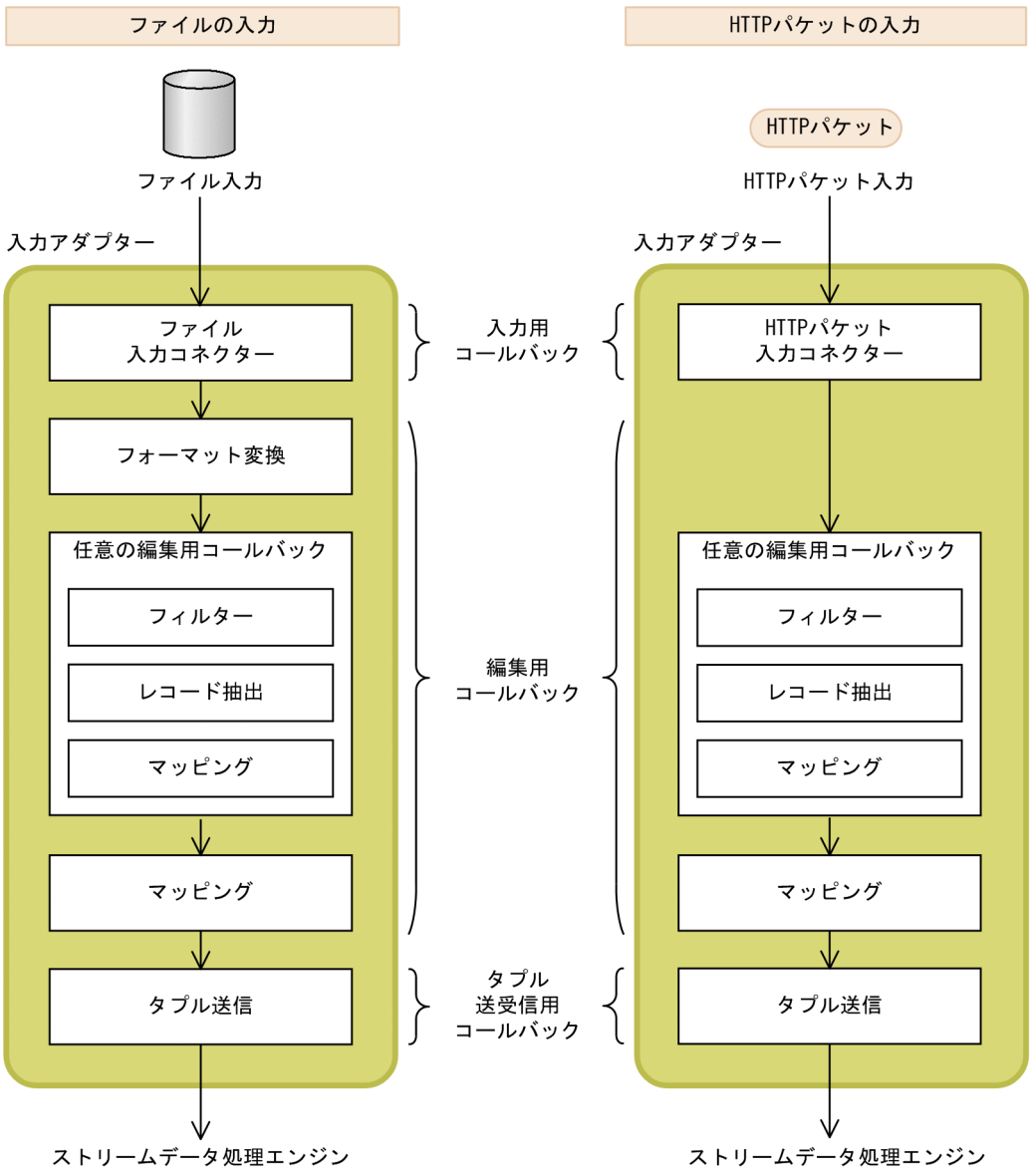
上記以外のデータを入力したり，別の出力先に処理結果を出力したりしたい場合は，カスタムアダプターを使用してください。カスタムアダプターについては「3.1.2　カスタムアダプター」を参照してください。

標準提供アダプターの機能を使用した処理の単位を**コールバック**といいます。アダプター用定義ファイルには，このコールバック単位に機能を定義します。

また，標準提供アダプターでのコールバックは，入力となるデータの 1 行ごと（HTTP パケットではリクエストメッセージ，またはレスポンスメッセージごと）に対して実行されます。この 1 行の単位を**レコード**といいます。

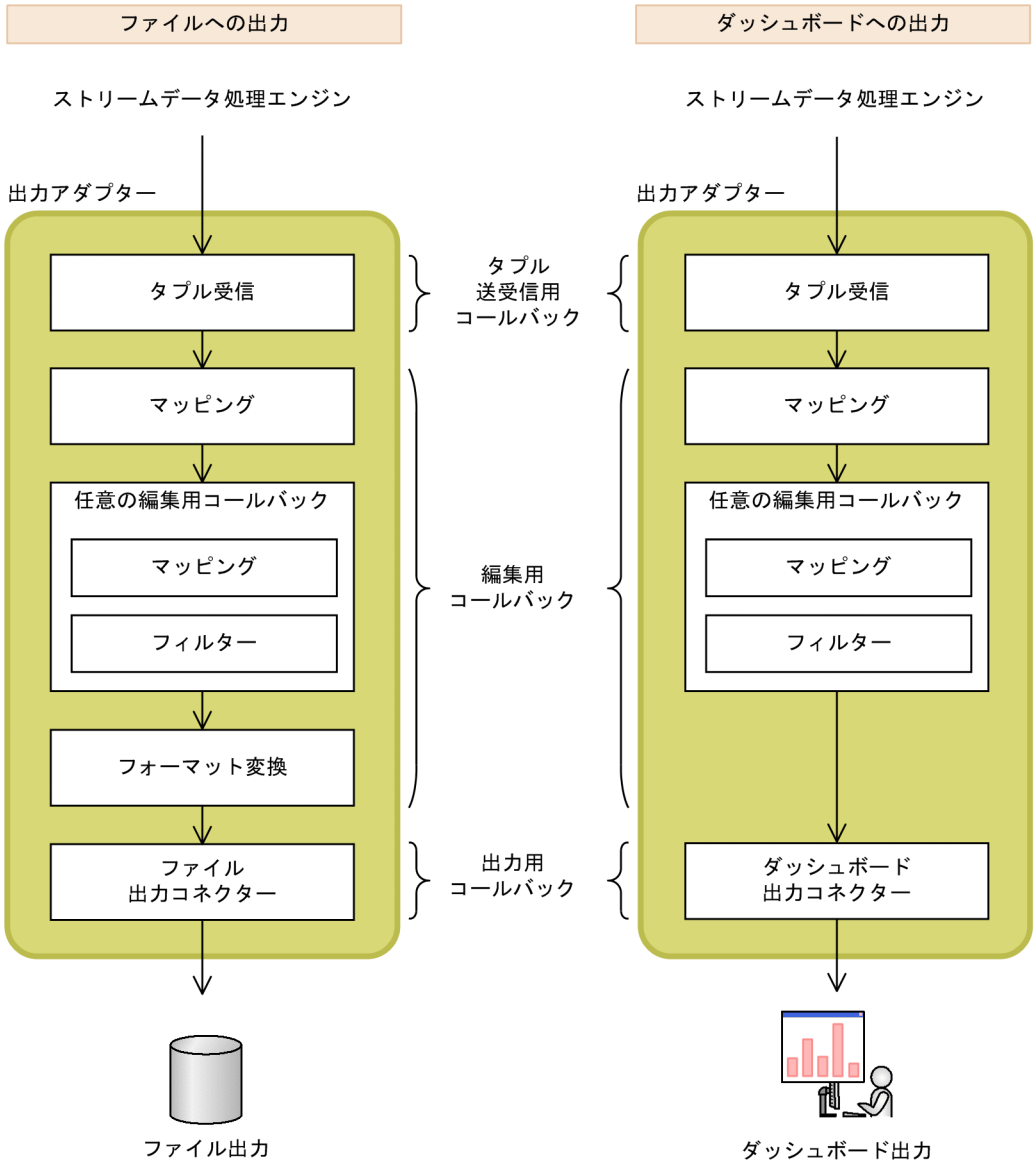
入力アダプターでのコールバックの構成を次の図に示します。

図 3-1　入力アダプターでのコールバックの構成



出力アダプターでのコールバックの構成を次の図に示します。

図 3-2　出力アダプターでのコールバックの構成



標準提供アダプターの機能の一覧を次の表に示します。

表 3-1　標準提供アダプターの機能の一覧

項番	コールバック	分類	目的	説明
1	ファイル入力コネクター	入力用コールバック	ファイルの入力	ファイルを入力とする場合に使用します。
2	HTTP パケット入力コネクター		HTTP パケットの入力	ネットワーク上の HTTP パケットを入力とする場合に使用します。
3	フォーマット変換	編集用コールバック	ファイル入力時または出力時のレコード形式の変換	ファイル入力コネクターまたはファイル出力コネクターを使用する場合にだけ指定します。入力アダプター用と出力アダプター用の 2 種類があります。

項番	コールバック	分類	目的	説明
3	フォーマット変換	編集用コールバック	ファイル入力時または出力時のレコード形式の変換	入力アダプター用 入力データから取得したレコード（**入力形式レコード**）をストリームデータ処理エンジンで処理できる形式（**共通形式レコード**）に変換します。 出力アダプター用 ストリームデータ処理エンジンで処理を終えた共通形式レコードを**出力形式レコード**に変換します。
4	フィルター		レコードのフィルタリング	フィルター対象のレコードを取捨選択できます。 フィルターを使用すると，「2.3.2　条件を満たすデータの抽出」で説明したような CQL で実行する処理をアダプターで実行できます。処理効率向上のため，アダプターで実行することをお勧めします。
5	レコード抽出		レコードの抽出	入力アダプターでだけ使用できます。 レコードを取捨選択して，特定のレコードを取得したあと結合し，新たなレコードを生成できます。 レコード抽出を使用すると，「2.3.6　ストリームデータの結合」で説明したような CQL で実行する処理をアダプターで実行できます。処理効率向上のため，アダプターで実行することをお勧めします。
6	マッピング		レコードのマッピング	入力アダプター用と出力アダプター用の 2 種類があります。 入力アダプター用 共通形式レコードを入力ストリームの形式に従った共通形式レコードに変換します。また，必要に応じて，前もって共通形式レコードを次のコールバックでの処理に適した共通形式レコードに変換できます。 出力アダプター用 出力ストリームの形式に従った共通形式レコードを次のコールバックでの処理に適する共通形式レコードに変換します。また，必要に応じて，共通形式レコードを次のコールバックでの処理に適した共通形式レコードに変換できます。
7	タプル送信	タプル送受信用コールバック	ストリームデータ処理エンジンへの送信	入力アダプターからタプルをストリームデータ処理エンジンに送信するために使用します。
8	タプル受信		ストリームデータ処理エンジンでの処理結果の受信	ストリームデータ処理エンジンで処理されたタプルを出力アダプターが受信するために使用します。
9	ファイル出力コネクター	出力用コールバック	ファイルへの出力	ストリームデータ処理エンジンでの処理結果をファイルに出力するために使用します。

項番	コールバック	分類	目的	説明
10	ダッシュボード出力コネクター	出力用コールバック	ダッシュボードへの出力	ストリームデータ処理エンジンでの処理結果をダッシュボードに出力して表示するために使用します。

コールバックの詳細については，マニュアル「uCosminexus Stream Data Platform - Application Framework システム構築・運用ガイド」を参照してください。

なお，このマニュアルでは，「3.2　ファイルの入力」以降で次のコールバックの概要について説明します。

- ファイル入力コネクター
- HTTP パケット入力コネクター
- フィルター
- レコード抽出
- ファイル出力コネクター
- ダッシュボード出力コネクター

表で示したコールバックは，組み合わせて使用します。コールバックの構成例について説明します。

コールバックの構成例

複数の観測所で気温を観測している場合に，特定の観測所での観測結果だけを集計・分析し，処理結果をファイルに出力する場合を想定します。この処理は，次の条件を満たすものとします。

- 観測所 1（ID：1）から送られてくるデータだけを集計・分析の対象とする。
- 処理結果は，ファイルに出力する。

このときのアダプターのコールバックの構成を，入力アダプターと出力アダプターに分けて，次の二つの図に示します。

図 3-3　入力アダプターのコールバックの構成例と処理の内容

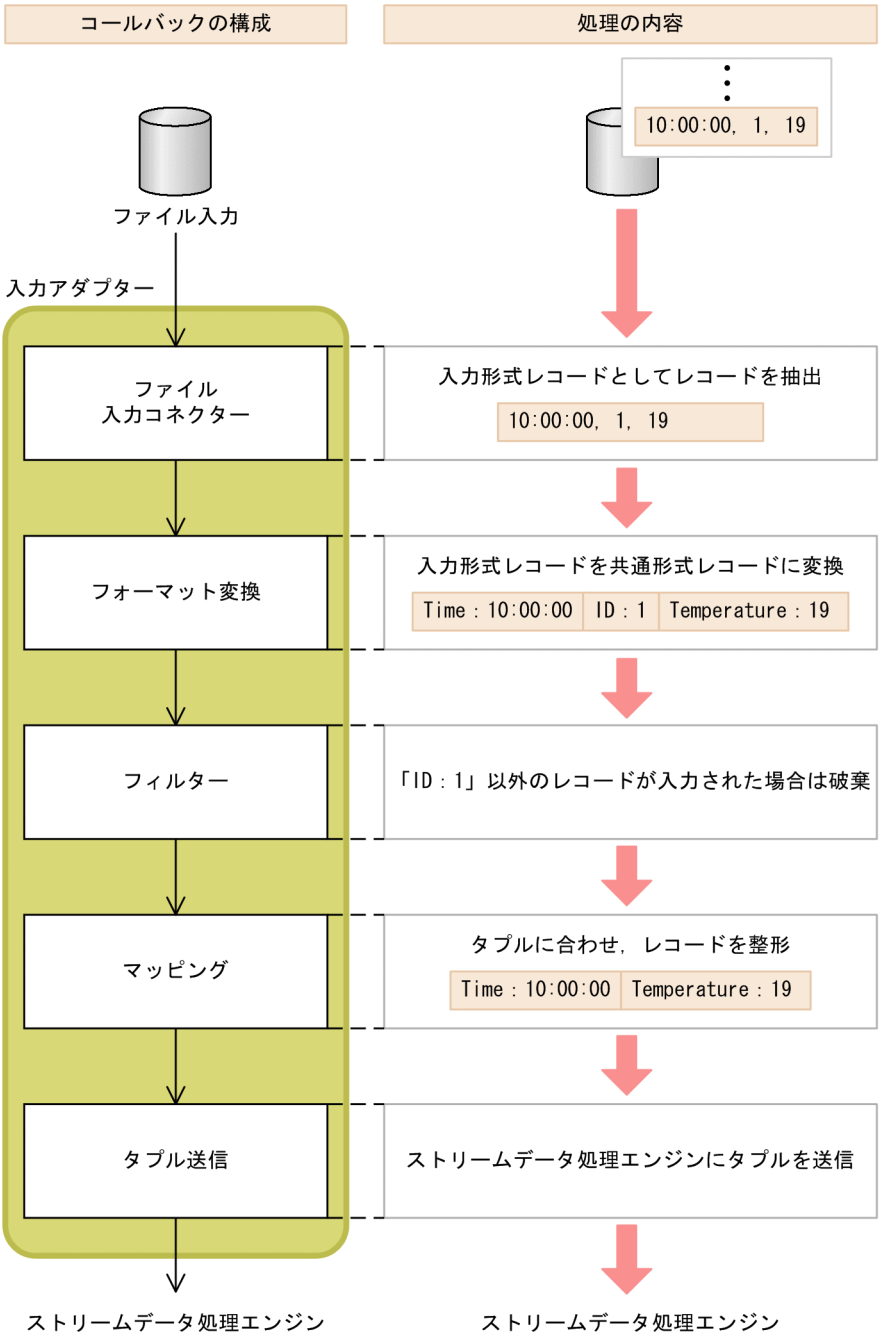
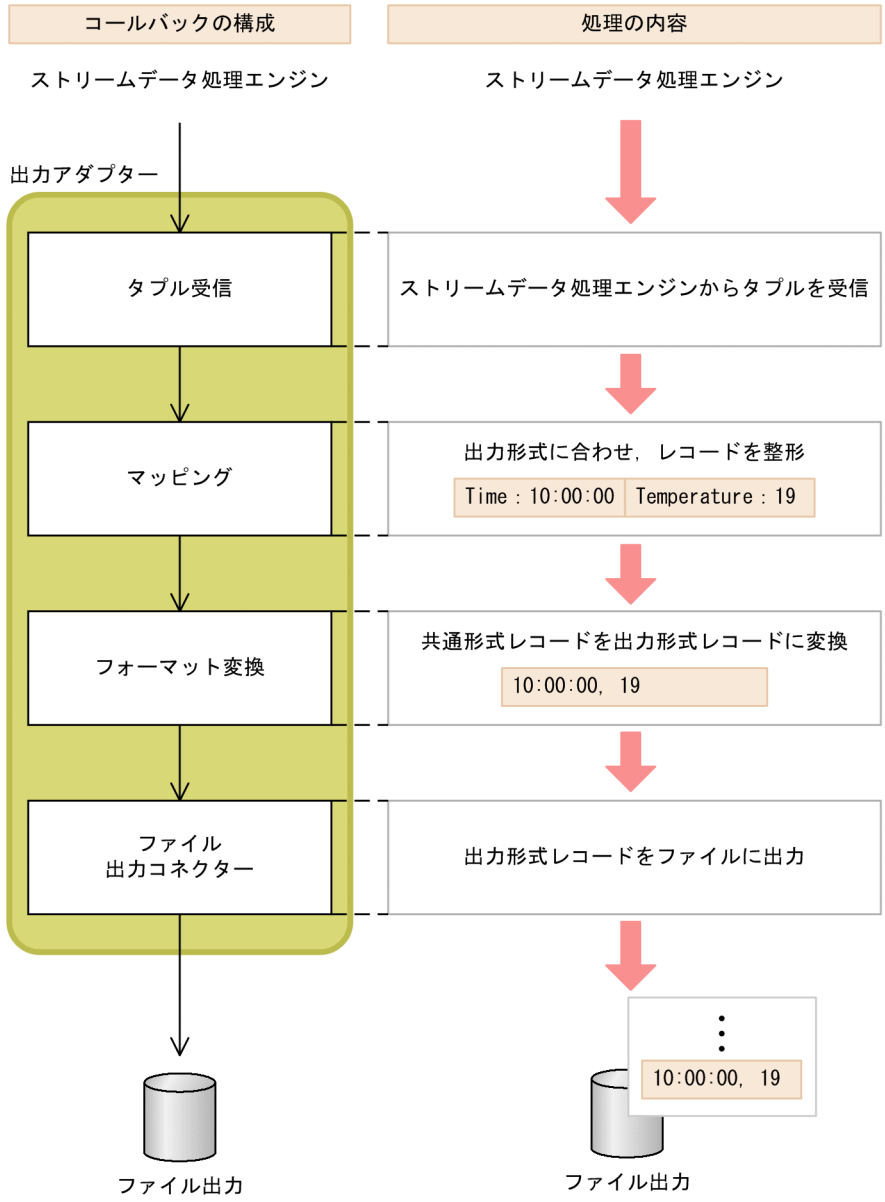


図 3-4　出力アダプターのコールバックの構成例と処理の内容



入力アダプターでは，観測所 1（ID：1）から送られてくるデータだけを集計・分析の対象とするため，フィルターを使用します。また，出力アダプターでは，処理結果をファイルに出力するため，ファイル出力コネクターを使用します。

3.1.2　カスタムアダプター

カスタムアダプターは，Stream Data Platform - AF が提供する Java の API を使用して，ユーザーがアプリケーションとして作成します。プログラミング次第で，さまざまな機能を実現できます。

カスタムアダプターを作成する場合は，最低限，次に示す機能を実装する必要があります。

- 外部データの入出力
- 入出力データとストリームデータの相互変換
- 入力ストリームへのタプル送信

- 出力ストリームからのタプル受信

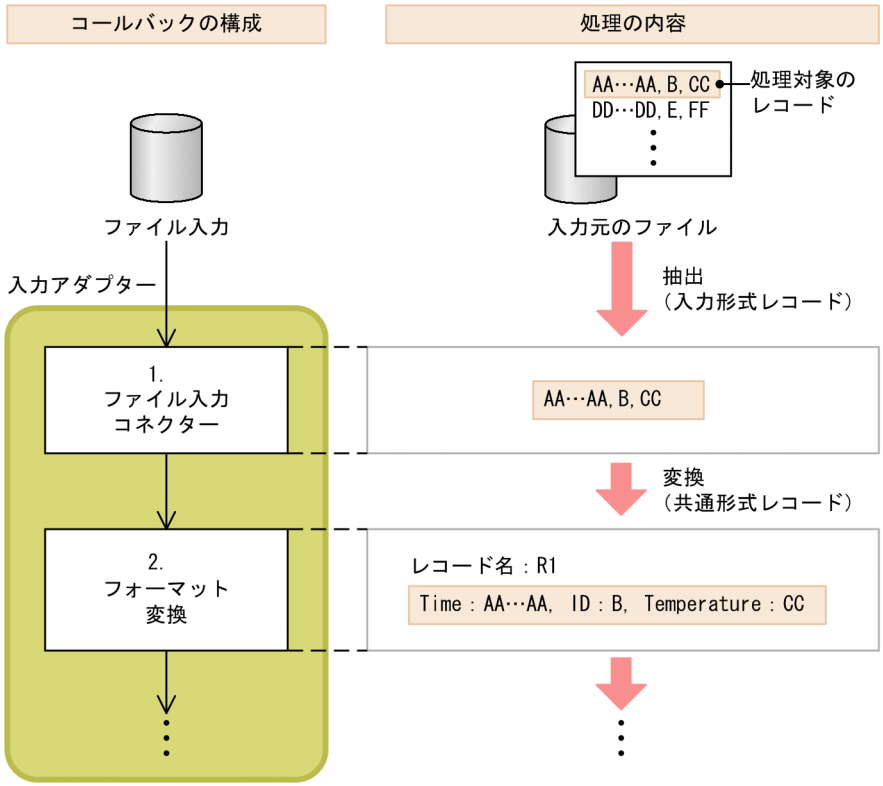
カスタムアダプターについては，マニュアル「uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド」を参照してください。

3.2　ファイルの入力

ストリームデータ処理の対象が，ログファイルなどファイル形式のデータである場合，入力用コールバックとして，**ファイル入力コネクター**を使用します。

ファイル入力コネクターは，入力元のファイルから，処理の対象となるレコードを抽出します。このレコードは，入力形式レコードとして抽出されるため，フォーマット変換で共通形式レコードに変換し，ストリームデータ処理エンジンで扱える形式にする必要があります。ファイル入力でのコールバックの構成と処理の内容を次の図に示します。

図 3-5　ファイルの入力でのコールバックの構成と処理の内容



1. ファイル入力コネクターで，入力されたファイルの 1 行目（レコード）を抽出します。このとき，抽出されるレコードは，入力形式レコードです。
2. フォーマット変換をして，入力形式レコードを共通形式レコードに変換します。

参考

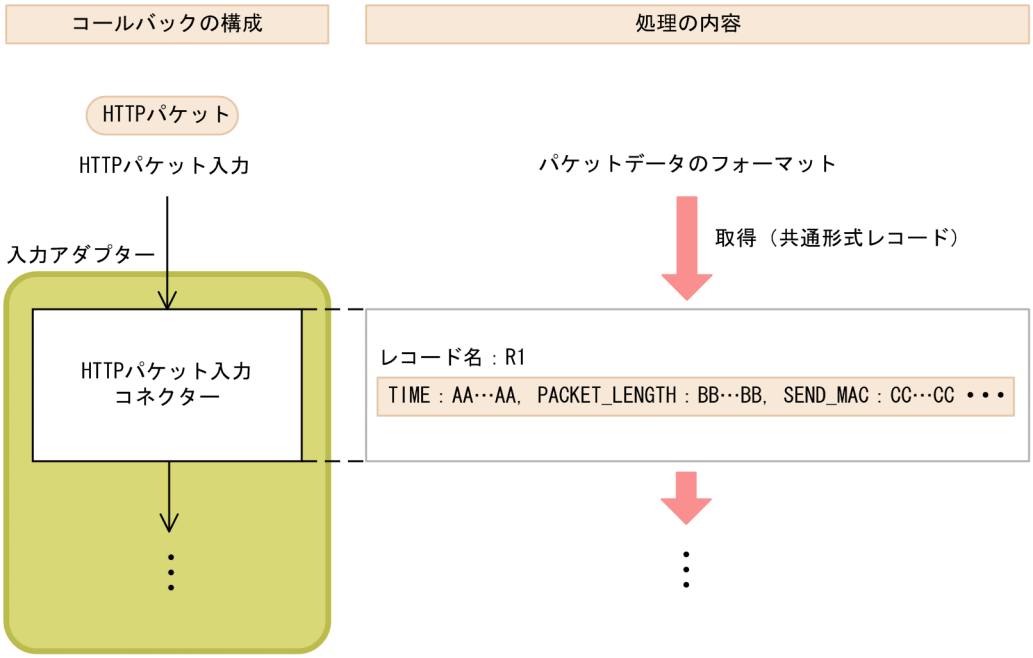
入力元のファイルからは，処理の対象となるレコードを一度に複数行抽出することもできます。

3.3　HTTP パケットの入力

ストリームデータ処理の対象が，ネットワーク上の HTTP パケットである場合，入力用コールバックとして，**HTTP パケット入力コネクター**を使用します。

パケット入力コネクターでは，パケットアナライザーから標準出力された HTTP パケットを取得します。HTTP パケットの入力でのコールバックの構成と処理の内容を次の図に示します。

図 3-6　HTTP パケットの入力でのコールバックの構成と処理の内容



この図に示すとおり，パケット入力コネクターが取得した HTTP パケットは，パケット入力コネクター内で共通形式レコードに変換され，ストリームデータ処理システムで扱えるデータ形式になります。

3.4　レコードのフィルタリング

特定のレコードだけをストリームデータ処理の対象にしたい場合，編集用コールバックとして，**フィルター**を使用します。

参考

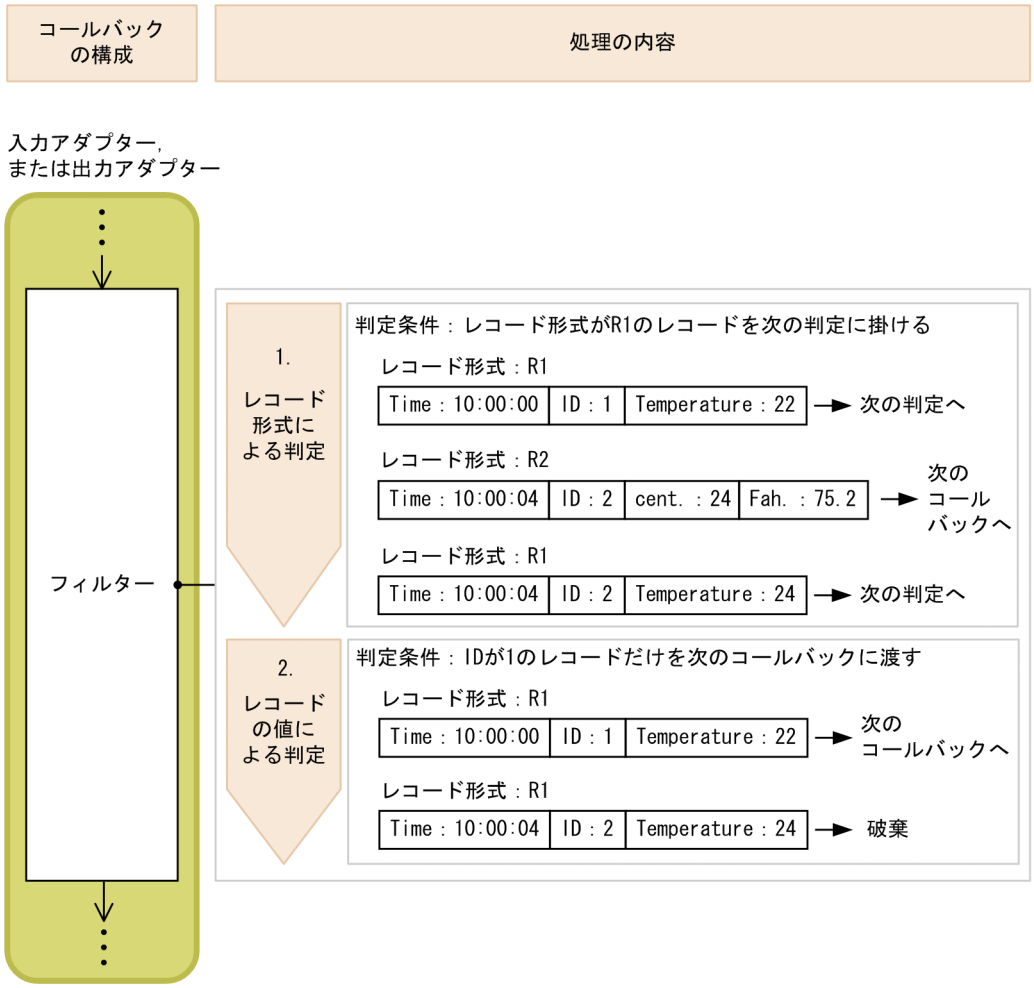
「2.3.2　条件を満たすデータの抽出」で説明したように，CQL でもフィルター処理を実行できますが，処理効率向上のため，アダプターで実行することをお勧めします。

例えば，複数の観測所で気温を観測している場合に，特定の観測所の気温だけを集計・分析の対象としたいときは，観測所に与えられている ID でフィルタリングを実行してください。

フィルタリングの対象は，共通形式レコードです。入力元がファイルの場合，ファイル入力コネクターで入力形式レコードを抽出したあと，フォーマット変換で共通形式レコードに変換してからフィルターを使用してください。

フィルタリングの判定条件には，レコード形式，およびレコードに設定されている値を指定できます。レコードのフィルタリングでのコールバックの構成と処理の内容を次の図に示します。

図 3-7　レコードのフィルタリングでのコールバックの構成と処理の内容



1. フィルターに入力されたレコードは，最初にレコード形式でフィルタリングされます。ここでは，レコード形式が R1 のレコードだけを次の判定に掛けるように条件を指定しているため，この条件を満たすレコードが，次の判定の対象となります。条件を満たさないレコードは，次のコールバックに渡されます。
2. レコード形式でフィルタリングしたあとは，レコードの値でフィルタリングします。ここでは，レコードの値に設定されている ID が 1 のレコードだけを次のコールバックに渡すように条件を指定しています。そのため，この条件を満たすレコードが，次のコールバックでの処理の対象となります。条件を満たさないレコードは，破棄されます。

3.5　レコードの抽出

レコードを取捨選択して，特定のレコードを取得したあと，それらのレコードを意味のある一つのレコードにする場合，**レコード抽出**を使用します。

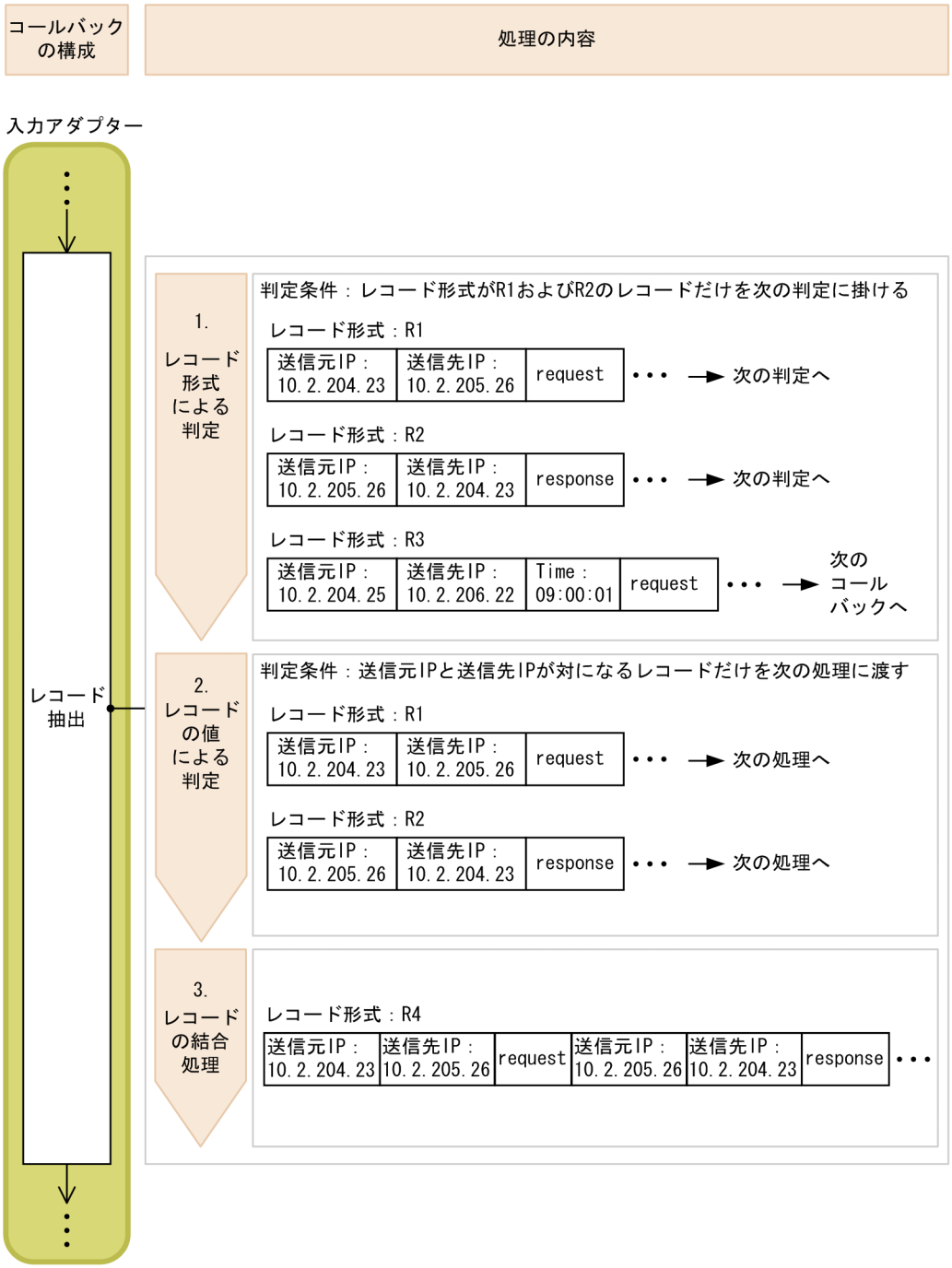
参考

「2.3.6　ストリームデータの結合」で説明したように，CQL でもレコード抽出処理を実行できますが，処理効率向上のため，アダプターで実行することをお勧めします。

例えば，クライアントとサーバ間の応答状況を集計・分析の対象としたい場合，入力用コールバックに HTTP パケット入力コネクターを使用した上で，編集用コールバックとしてレコード抽出を使用します。ネットワーク上のリクエストとレスポンスのそれぞれの HTTP パケットを，レコード抽出によって送信元 IP と送信先 IP で結び付けた一つのレコードにすれば，応答時間などが明確になり，集計・分析がしやすくなります。

レコード抽出では，レコード形式およびレコードに設定されている値で目的のレコードをフィルタリングしたあと，それらのレコードを結合して新たなレコードを生成します。レコードの抽出でのコールバックの構成と処理の内容を次の図に示します。

図 3-8　レコードの抽出でのコールバックの構成と処理の内容



1. レコード抽出に入力されたレコードは，最初にレコード形式でフィルタリングされます。ここでは，レコード形式が R1 および R2 のレコードだけを次の判定に掛けるように条件を指定しているため，この条件を満たすレコードが，次の判定の対象となります。条件を満たさないレコードは，次のコールバックに渡されます。
2. レコード形式でフィルタリングしたあとは，レコードの値でフィルタリングします。ここでは，リクエストの送信元 IP および送信先 IP，ならびにレスポンスの送信元 IP および送信先 IP が，それぞれ対になるレコードだけを次の処理に渡すように条件を指定しています。そのため，この条件を満たすレコードが，次の処理での対象となります。

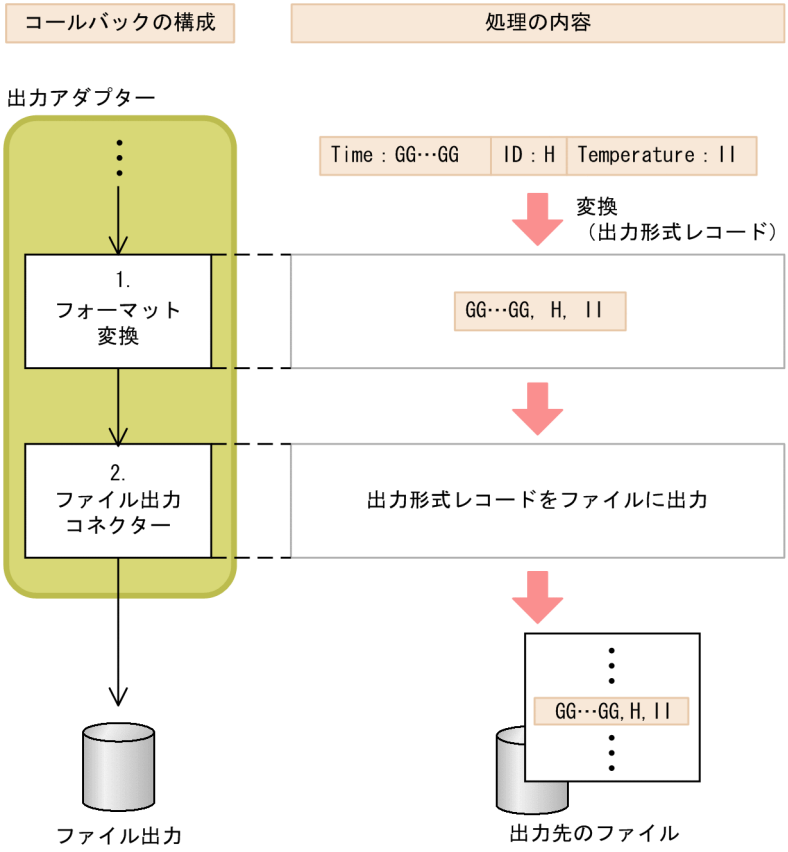
3. レコード形式，およびレコードに設定されている値でフィルタリングされたレコードを結合し，一つのレコードとします。ここで結合されたレコードが，次のコールバックでの処理の対象となります。

3.6　ファイルへの出力

ストリームデータ処理の結果をファイルに出力する場合，出力用コールバックとして**ファイル出力コネクター**を使用します。

ファイル出力コネクターでは，出力形式レコードをファイルに出力します。そのため，ファイル出力コネクターで処理する前に，フォーマット変換で共通形式レコードを出力形式レコードに変換しておく必要があります。ファイルへの出力でのコールバックの構成と処理の内容を次の図に示します。

図 3-9　ファイルへの出力でのコールバックの構成と処理の内容



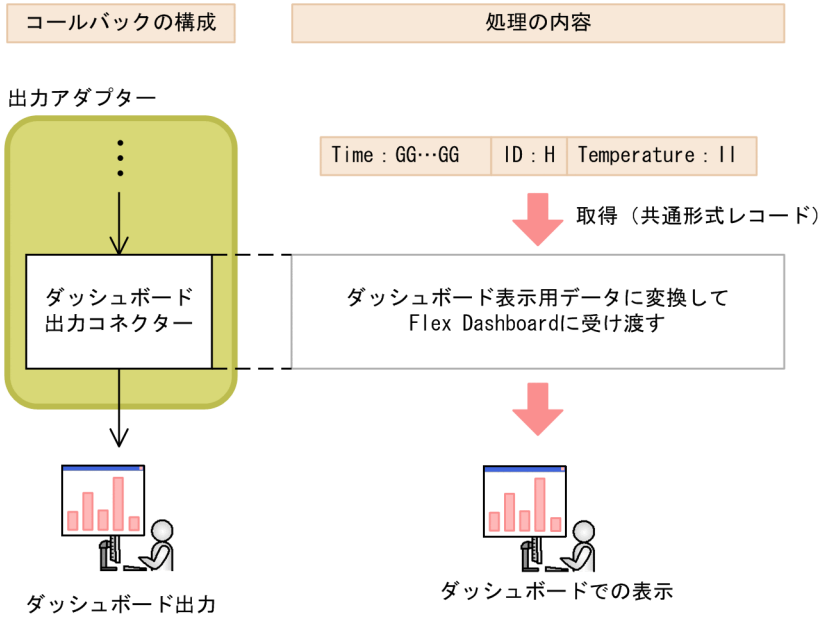
1. ストリームデータ処理の結果をファイルに出力するために，フォーマット変換でレコード形式を出力形式レコードに変換します。
2. ファイル出力コネクターで出力形式レコードをファイルに出力します。

3.7　ダッシュボードへの出力

ストリームデータ処理の結果をダッシュボードに出力して表示する場合，出力用コールバックとして**ダッシュボード出力コネクター**を指定します。ダッシュボードに出力されたデータは，折れ線グラフや棒グラフで表示できます。

ダッシュボード出力コネクターでは，直前のコールバックから共通形式レコードを取得します。そのあと，ダッシュボード表示用データに変換し，Flex Dashboard に受け渡します。ダッシュボードへの出力でのコールバックの構成と処理の内容を次の図に示します。

図 3-10　ダッシュボードへの出力でのコールバックの構成と処理の内容



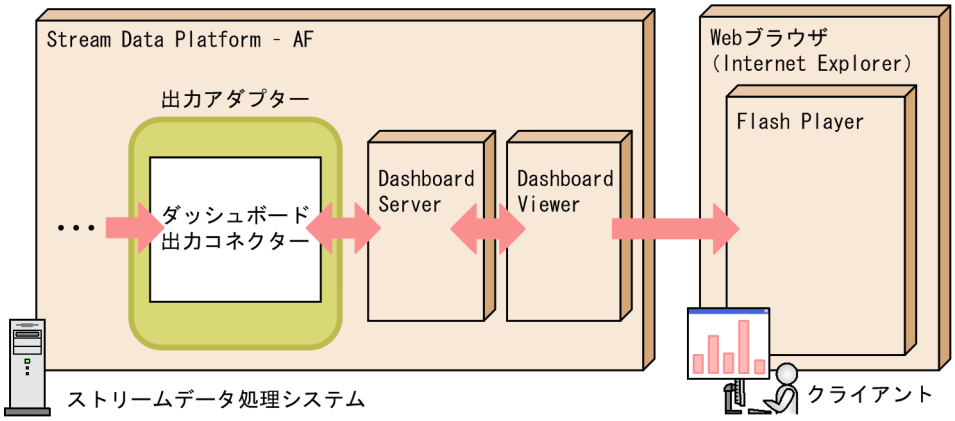
3.7.1　Flex Dashboard

Stream Data Platform - AF では，ダッシュボードとして Flex Dashboard を使用します。Flex Dashboard は，次に示す 2 種類のアプリケーションで構成されます。

- Dashboard Server
  ダッシュボード出力コネクターと連携し，処理結果を取得するアプリケーションです。Stream Data Platform - AF が動作するサーバでコマンドを実行して登録します。
- Dashboard Viewer
  ストリームデータ処理の結果を Web 上で表示するための GUI を提供するアプリケーションです。クライアントの Web ブラウザにダウンロードされて実行されます。Dashboard Viewer での表示内容は，Dashboard Viewer の画面編集用ファイルの定義内容に従います。

Flex Dashboard の構成を次の図に示します。

図 3-11　Flex Dashboard の構成



出力アダプターに送られてきたデータは，ダッシュボード出力コネクターで処理され，Dashboard Server に受け渡されます。Dashboard Server は，データを Dashboard Viewer に受け渡し，閲覧者がストリームデータ処理システムでの処理結果をダッシュボードで表示できるようにします。

3.7.2　表示例

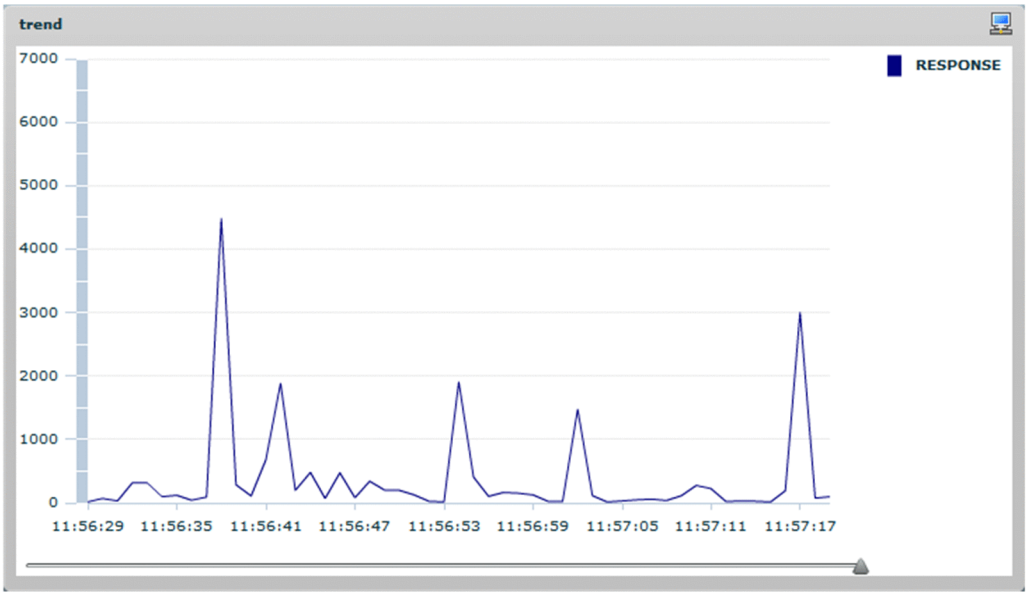
ダッシュボードでの表示例について，HTTP パケットを利用して次の集計・分析結果を表示させる場合を例に説明します。

- リクエストからレスポンスまでの平均応答時間（ミリ秒）
- レスポンスのステータスコードごとの割合

(1)　線グラフによる表示

HTTP パケットを集計・分析し，リクエストからレスポンスまでの平均応答時間（ミリ秒）を線グラフで表示させた場合の表示例を次の図に示します。

図 3-12　リクエストからレスポンスまでの平均応答時間（ミリ秒）を線グラフで表示させた場合の表示例



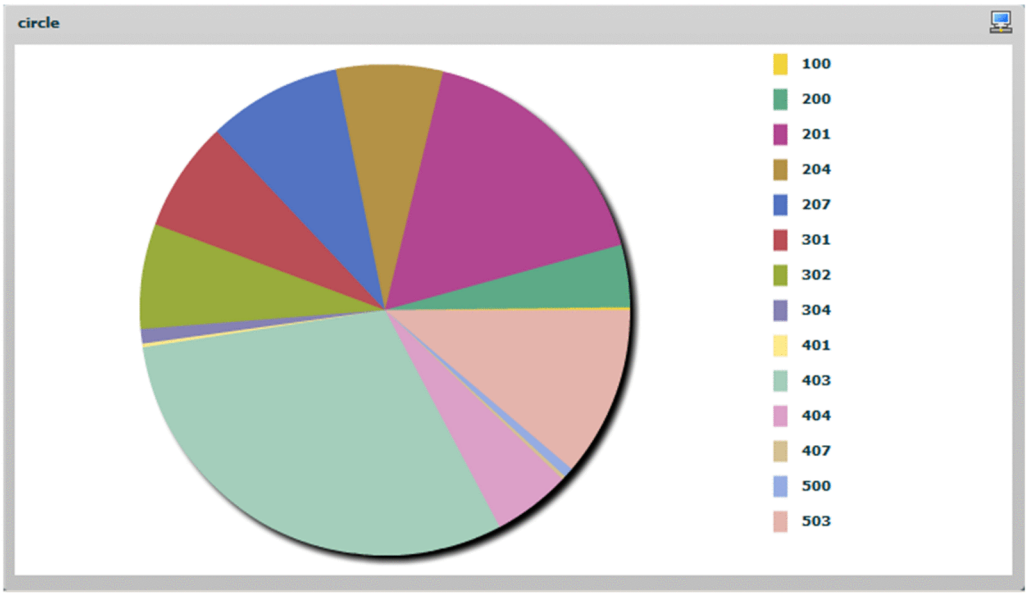
縦軸が平均応答時間（ミリ秒）を，横軸が時刻を示しています。

このように表示させることで，例えば時刻 11:57:17 でのリクエストからレスポンスまでの平均応答時間が 3,000 ミリ秒（3 秒）であることがわかります。

(2)　円グラフによる表示

HTTP パケットを集計・分析し，レスポンスのステータスコードごとの割合を円グラフで表示させた場合の表示例を次の図に示します。

図 3-13　レスポンスのステータスコードごとの割合を円グラフで表示させた場合の表示例



円グラフ内の色は，ステータスコードごとに塗り分けられています。

このように表示させることで，例えばレスポンス全体のうち，ステータスコード 403 が 1/3 程度を占めていることがわかります。

4 シリーズマニュアルの紹介

この章では，Stream Data Platform - AF の導入，設計，構築，および運用の各フェーズでの作業とシリーズマニュアルの対応, およびシリーズマニュアルの概要について説明します。

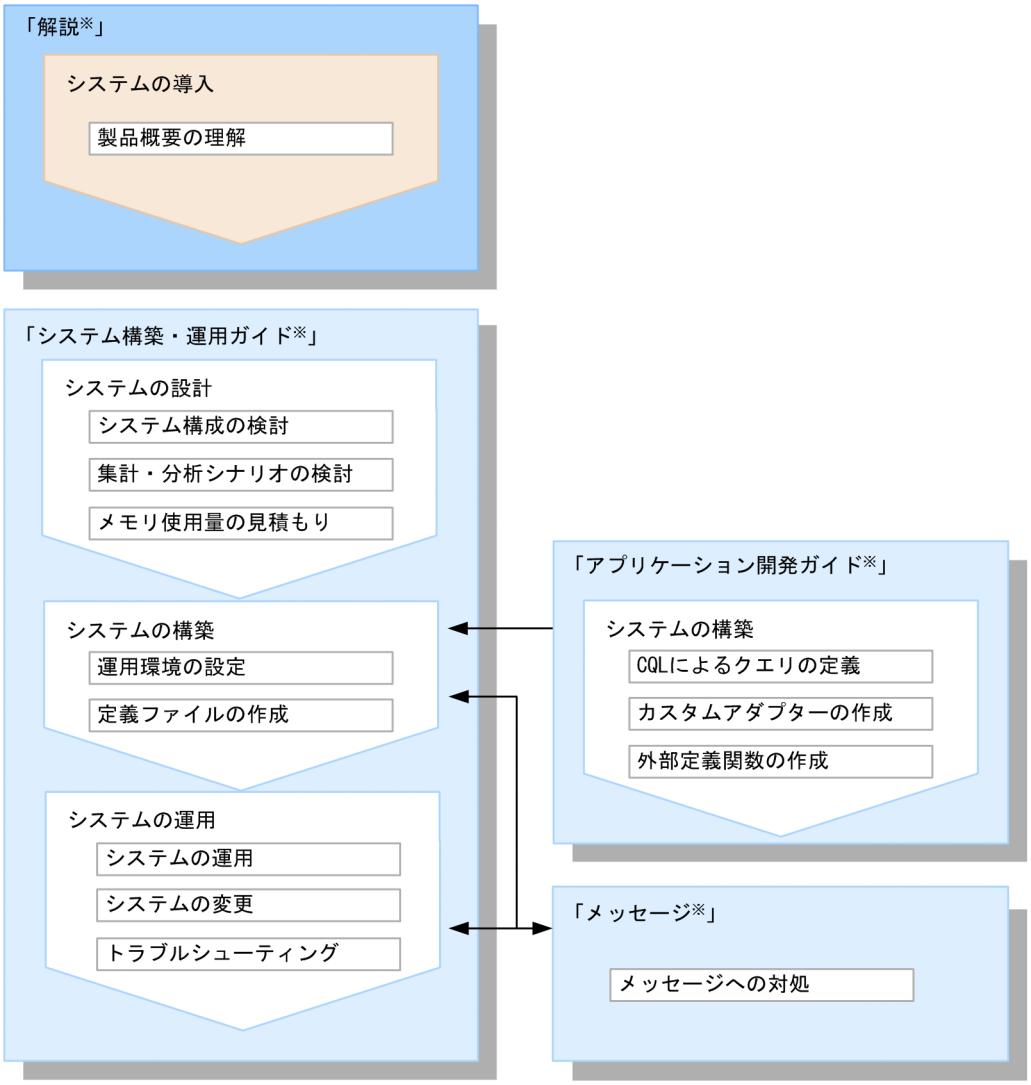
4.1　シリーズマニュアルとユーザーの作業項目の対応

Stream Data Platform - AF のシリーズマニュアルには，次の 4 冊があります。

- uCosminexus Stream Data Platform - Application Framework 解説
- uCosminexus Stream Data Platform - Application Framework システム構築・運用ガイド
- uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド
- uCosminexus Stream Data Platform - Application Framework メッセージ

これらのマニュアルと，Stream Data Platform - AF の導入，設計，構築，および運用の各フェーズでの作業項目がどのように対応しているのかを次の図に示します。

図 4-1　導入，設計，構築，および運用の各フェーズでの作業項目とシリーズマニュアルの対応



注※　マニュアル名称のうち，「uCosminexus Stream Data Platform - Application Framework」は省略して表記しています。

それぞれのマニュアルの概要については，「4.2　シリーズマニュアルの概要」を参照してください。

4.2　シリーズマニュアルの概要

Stream Data Platform - AF のシリーズマニュアルの記載内容を紹介します。

- uCosminexus Stream Data Platform - Application Framework 解説

  このマニュアルです。

  Stream Data Platform - AF の導入時にお読みください。Stream Data Platform - AF の機能概要のうち，基本的な使い方について記載しています。

- uCosminexus Stream Data Platform - Application Framework システム構築・運用ガイド

  マニュアル「uCosminexus Stream Data Platform - Application Framework 解説」を読んでいただいた上で，お読みください。Stream Data Platform - AF の設計・構築・運用方法，および構築時に設定できる機能の詳細について説明しています。

  Stream Data Platform - AF のリファレンスや，トラブルシューティングを確認する場合も，このマニュアルをお読みください。

- uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド

  マニュアル「uCosminexus Stream Data Platform - Application Framework 解説」を読んでいただいた上で，CQL によるクエリの定義と，カスタムアダプターなどのアプリケーションを作成する場合にお読みください。

- uCosminexus Stream Data Platform - Application Framework メッセージ

  Stream Data Platform - AF の構築，および運用で，Stream Data Platform - AF のメッセージが出力された場合にお読みください。

付録

付録 A　各バージョンの変更内容

表 A-1　変更内容(3020-3-V01-10)　uCosminexus Stream Data Platform - Application Framework　01-05

追加・変更内容
リレーションを生成しないで直接ストリームデータに対して演算（ストリーム間演算）を実行できるようにした。 また，CQL で，ユーザーが Java でプログラミングした外部定義関数を使用できるようにした。 ストリームデータに対する演算機能の追加に伴い，操作系 CQL に次の関数を追加した。 • ストリーム間演算関数 • 外部定義ストリーム間演算関数 また，次の定義ファイルを追加した。 • 外部定義関数定義ファイル
操作系 CQL に次の関数を追加した。 • 組み込み集合関数 • スカラ関数 • 組み込みスカラ関数

表 A-2　uCosminexus Stream Data Platform - Application Framework　01-01

追加・変更内容
適用 OS に Linux を追加した。

付録 B　このマニュアルの参考情報

このマニュアルを読むに当たっての参考情報を示します。

付録 B.1　関連マニュアル

関連マニュアルを次に示します。必要に応じてお読みください。

- **ストリームデータ処理基盤 uCosminexus Stream Data Platform - Application Framework システム構築・運用ガイド（3020-3-V02）**

  Stream Data Platform - AF の設計・構築・運用方法，および構築時に設定できる機能の詳細について説明しています。

  Stream Data Platform - AF のシステムを設計・構築・運用して，ストリームデータの分析を実施する場合に参照してください。

- **ストリームデータ処理基盤 uCosminexus Stream Data Platform - Application Framework アプリケーション開発ガイド（3020-3-V03）**

  Stream Data Platform - AF でデータを分析するために使用する CQL の記述方法，カスタムアダプターなどのアプリケーションを作成する方法について説明しています。

  分析目的に合わせた CQL を記述したり，API でカスタムアダプターなどのアプリケーションを作成したりする場合に参照してください。

- **ストリームデータ処理基盤 uCosminexus Stream Data Platform - Application Framework メッセージ（3020-3-V04）**

  Stream Data Platform - AF が出力するメッセージについて説明しています。

  メッセージが出力された際に，必要に応じて参照してください。

なお，このマニュアルでは，関連マニュアルについて，「ストリームデータ処理基盤」を省略して表記しています。

付録 B.2　このマニュアルでの表記

このマニュアルでは，製品名および Java 関連用語を次のように表記しています。

表記		製品名または Java 関連用語
Java		Java™
JavaVM		Java™ Virtual Machine
Linux		Linux(R)
Linux	Red Hat Enterprise Linux 5	Red Hat Enterprise Linux(R) 5 Advanced Platform（AMD/Intel 64）
		Red Hat Enterprise Linux(R) 5（AMD/Intel 64）
	Red Hat Enterprise Linux 6	Red Hat Enterprise Linux(R) Server 6（64-bit x86_64）
Stream Data Platform - AF		uCosminexus Stream Data Platform - Application Framework

付録 B.3　英略語

このマニュアルで使用する英略語を次に示します。

英略語	英語での表記
AP	Application Program
API	Application Programming Interface
CQL	Continuous Query Language
GUI	Graphical User Interface
HTTP	Hyper Text Transfer Protocol
IC	Integrated Circuit
IP	Internet Protocol
RMI	Remote Method Invocation

付録 B.4　KB（キロバイト）などの単位表記について

1KB（キロバイト），1MB（メガバイト），1GB（ギガバイト），1TB（テラバイト）はそれぞれ 1,024 バイト，$1{,}024^2$ バイト，$1{,}024^3$ バイト，$1{,}024^4$ バイトです。

付録 C　用語解説

(英字)

CQL（Continuous Query Language）

クエリを記述するためのクエリ言語です。

RMI 連携（Remote Method Invocation）

アダプターと SDP サーバの連携形態の一つです。アダプターと SDP サーバが，別のプロセスで動作し，Java の RMI を使用してデータを送受信します。

SDP サーバ

ストリームデータ処理エンジン上で動作し，ストリームデータを処理するサーバプロセスです。

SDP サーバ用定義ファイル

SDP サーバの動作を設定するファイルです。SDP サーバやアダプターを実行する JavaVM の起動オプション，SDP サーバのポート番号や取得する API トレースとタプルログの詳細などを設定します。

(ア行)

アダプター

入出力データとストリームデータ処理エンジン間でのデータの送受信に必要なプログラムです。

アダプターには，製品が提供する標準提供アダプターと，ユーザーが Java でプログラミングして作成するカスタムアダプターがあります。

それぞれのアダプターで，入力データとストリームデータ処理エンジン間で使用するアダプターを入力アダプターといい，出力データとストリームデータ処理エンジン間で使用するアダプターを出力アダプターといいます。

アダプターグループ

入出力アダプターの集合です。標準提供アダプターでは，アダプターグループ単位でアダプターを運用します。

インプロセス連携をするアダプターグループのことをインプロセスグループ，RMI 連携をするアダプターグループのことを RMI グループといいます。

アダプタートレース

トラブルシュート情報として，アダプター内の処理状況を追跡する情報を取得するための機能です。

アダプター用定義ファイル

標準提供アダプターの動作を設定するファイルです。RMI 連携用アダプターが使用するポート番号，アダプターグループの構成や，アダプターが使用する入出力コネクターの詳細などを設定します。

インプロセス連携

アダプターと SDP サーバの連携形態の一つです。アダプターと SDP サーバは，同一プロセスで動作してデータを送受信します。

ウィンドウ

ストリームデータに対し，集計・分析を行うために設定する範囲です。クエリ中に定義します。

ウィンドウ演算

ウィンドウを指定するための演算です。CQL でクエリ中に記述します。

オペレーター

ストリームデータ処理の最小処理単位です。一つのクエリは，一つ以上のオペレーターで構成されます。

（カ行）

外部定義関数

ユーザーが Java の API などを使用して作成する関数です。外部定義関数の処理ロジックを，ユーザーが Java で記述して作成するクラスファイルにメソッドとして実装することで，任意の処理を実行できます。

外部定義関数定義ファイル

外部定義関数の処理を実装したクラス，およびメソッドについて定義するファイルです。

カスタムアダプター

Stream Data Platform - AF が提供する Java の API を使用してユーザーが作成するアダプターです。

関係演算

ウィンドウ演算によって抽出されたデータの処理を指定する演算です。計算，集計，結合などを指定できます。

共通形式レコード

ストリームデータ処理システムで処理を行うためのレコードの形式です。

クエリ

ストリームデータに対してどのような処理を実行するかを定義したものです。CQL で記述します。

クエリグループ

あらかじめユーザーが作成するストリームデータの集計・分析シナリオです。入力ストリームキュー（入力ストリーム），出力ストリームキュー（出力ストリーム），およびクエリで構成されます。

組み込み関数

Stream Data Platform - AF が提供する関数です。統計関数を提供する組み込み集合関数や，数学関数や文字列関数を提供する組み込みスカラ関数があります。

コールバック

標準提供アダプターの機能を使用した処理の単位です。

コネクター

標準提供アダプター内に定義する Stream Data Platform - AF と外部を接続するインタフェースです。

Stream Data Platform - AF への入力に使用するコネクターには，ファイル入力コネクター，および HTTP パケット入力コネクターがあります。また，Stream Data Platform - AF からの出力に使用するコネクターには，ファイル出力コネクター，およびダッシュボード出力コネクターがあります。

（サ行）

サーバモード

タプルのタイムスタンプの設定モードの一つです。ストリームデータ処理エンジンにタプルが到着した時点で，Stream Data Platform - AF が動作するサーバのシステム時刻をタプルに設定します。

時刻解像度機能

RANGE ウィンドウを任意の単位時間に分割（メッシング）し，分割した時間ごとに演算処理を行う機能です。

出力形式レコード

ストリームデータの処理結果をファイルに出力する場合のレコード形式です。

出力リレーション

関係演算によって抽出されたタプルの集合です。このタプルの集合が，ストリーム化演算の対象となります。

ストリーム

ストリームデータの型です。入力ストリームキューを通過するストリームデータの型を入力ストリームといい，出力ストリームキューを通過するストリームデータの型を出力ストリームといいます。

ストリーム化演算

出力リレーションのデータの出力方法を指定する演算です。

ストリーム間演算

リレーションを生成しないで直接ストリームデータに対して演算を実行し，別のストリームデータに変換する演算です。

ストリーム間演算を使用する場合は，CQL でストリーム間演算関数を定義するほかに，外部定義関数の作成が必要です。

ストリームキュー

ストリームデータの入出力で使用される通路です。ストリームデータ処理エンジンへの入力で使用されるストリームキューを入力ストリームキューといい，ストリームデータ処理エンジンからの出力で使用されるストリームキューを出力ストリームキューといいます。

ストリームデータ

連続して発生する膨大な時系列順のデータです。

ストリームデータ処理エンジン

クエリに従い，ストリームデータ処理を実際に行う部分です。

（タ行）

タイムスタンプ

タプルに設定されたデータの時刻です。

タプル

ストリームデータを構成する，値と時刻（タイムスタンプ）を併せ持ったデータです。

タプルログ

ストリームデータ処理エンジンへの入力タプル，およびストリームデータ処理エンジンからの出力タプルが出力されるログファイルです。

中間リレーション

関係演算の処理中に，WHERE 句で抽出されたタプルの集合です。

データ受信 AP

クライアント AP であり，SDP サーバから出力されるストリームデータに対してイベント処理するアプリケーションです。

データ送信 AP

クライアント AP であり，ストリームデータを SDP サーバへ送信するデータ送信アプリケーションです。

データソースモード

タプルのタイムスタンプの設定モードの一つです。ログファイルなど，入力とするデータソース上に時刻情報がある場合に，その時刻情報をタプルに設定します。

（ナ行）

入力形式レコード

入力元がファイルの場合に取得したレコードです。

入力リレーション

ウィンドウ演算によって取り出されたタプルの集合です。このタプルの集合が，関係演算の対象となります。

（ハ行）

標準提供アダプター

Stream Data Platform - AF が提供するアダプターです。ファイルまたは HTTP パケットを入力データとして扱えます。処理結果は，ファイルに出力，またはダッシュボードに表示できます。

フィールド

レコードをある意味に切り出した単位です。

（ラ行）

リレーション

生存期間を持つレコード群です。CQL でのウィンドウ指定によって，ストリームデータからウィンドウで指定された生存期間を持つリレーションに変換されます。

レコード

ストリームデータ処理で扱われるデータの 1 行の単位です。

レコード構成

複数のフィールド（フィールド名とそれに対応するフィールド値）の組み合わせを表した構成です。

索引

く

け

こ

さ

し

す

せ

そ

た

ち

て

と

に

は

ひ

ふ

へ

り

れ