SYBASE®
An SAP® Company

新機能ガイド

Sybase Event Stream Processor 5.0

ドキュメント ID： DC01739-01-0500-01
改訂： 2011 年 8 月

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

目次

Aleri Streaming Platform 3.2 を継承する Sybase Event Stream Processor 5.0

Sybase® Event Stream Processor 5.0 は次世代の Aleri Streaming Platform です。

バージョン 5.0 には、Sybase CEP R4 の多くの機能も含まれています。

本書では、新しい機能と変更された機能について簡単に説明します。Aleri Streaming Platform からの移行方法の詳細については、『Migration Guide』を参照してください。

CCL

Sybase Event Stream Processor の主要なイベント処理言語として、AleriML に代わって CCL (Continuous Computation Language) が使用されるようになりました。

CCL は SQL (Structured Query Language) をベースとしますが、イベント・ストリーム処理に合わせて変更されています。機能的には AleriML に似ていますが、次の点が異なります。

- XML タグではなく、SQL に似た構文を使用します。
- AleriML のプリミティブ型に限定されない複雑なクエリを表現できます。

Sybase Event Stream Processor 5.0 の CCL は、単純化と合理化が実現された次世代の言語です。CCL は Sybase CEP R4 から採用されていますが、Sybase CEP R4 の構文とセマンティックのすべてがサポートされているわけではありません。CCL の構文と使用方法の詳細については、『CCL Programmers Guide』を参照してください。

CCL は SPLASH コードを埋め込む機能を継承しています。SPLASH を使用して、Flex 演算子 (Aleri では Flex ストリーム) と呼ばれるカスタムの関数と演算子を定義できます。SPLASH 言語は変更されていません。SPLASH の詳細については、『SPLASH Programmers Reference』と『SPLASH Tutorial』を参照してください。

イベント・ストリームの機能
CCL のイベント・ストリームは、AleriML よりも柔軟性に優れています。

- ストリームとウィンドウは異なります。AleriML ではイベント・ストリームはすべてストリームと呼ばれていましたが、状態があったことから、ストリームよりもウィンドウに近いものでした。CCL では、ストリームとウィンドウの両方ともイベントを処理します。ウィンドウには状態があります。つまり、ウィ

ンドウではデータの保持と保管が可能ですが、ストリームはステートレスのため、データの保持も保管もできません。
- ストリームにはプライマリ・キーがありません。
- ストリームは挿入のみをサポートします。ウィンドウは、そのタイプに応じて、挿入、更新、削除、アップサートをサポートします。
- デルタ・ストリームは、通常のストリームと同様にステートレスですが、プライマリ・キーを設定できるため更新と削除を処理できます。更新と削除は保存する必要があるが、状態は保持する必要がないウィンドウからのダウンストリームなど、一部のユース・ケースでは最適化のためにデルタ・ストリームを使用できます。
- ストリームとウィンドウには、入力、ローカル、出力などがあります。ローカル・ストリームとローカル・ウィンドウは、Event Stream Processor プロジェクト内部でのみ表示できます。外部で表示できるのは出力ストリームと出力ウィンドウのみのため、ダウンストリーム・アプリケーションを実装している場合は選択が簡単になります。
- Flex 演算子では、ストリームまたはウィンドウのどちらでも生成できます。

柔軟性の高い継続クエリ
CCL では、継続クエリを含むストリームまたはウィンドウを定義できます。クエリは「型指定されません」。つまり、Aleri の場合のように、ジョイン、集約、またはその他のストリーム・タイプとして指定する必要はありません。便宜上、標準のクエリ・タイプも使用できますが、汎用のストリームとウィンドウの各オブジェクトにはどのような CCL クエリでも自由に入れられます。

CCL は、内部ジョインだけでなく、AleriML でサポートされる他のジョイン・タイプもサポートしています。

注意： ジョインの点では、Aleri コンパイラより Event Stream Processor コンパイラの方が厳密です。Event Stream Processor Server で移行できる場合もありますが、コンパイルはできません。

コンパイル
CCL クエリは、CCL コンパイラによって実行可能な形式に変換されます。一般的に、コンパイルは Event Stream Processor Studio 内で実行されますが、コマンド・ラインから CCL コンパイラを呼び出しても実行できます。

モジュール性

新しいモジュール性に関する機能によって、再利用とチーム開発が容易になり、大規模で複雑なプロジェクトを簡単に保守できるようになりました。

モジュール化された CCL コードを作成してロードする場合の特徴は、次のとおりです。

- **CREATE MODULE** 文を使用してモジュールを定義します。
- モジュールにはストリームとウィンドウを入れることができ、ストリームとウィンドウには継続クエリを入れることができます。
- モジュールは、宣言済みのパラメータを参照できます。このパラメータは、モジュールがロードされると設定されます。
- **IMPORT** ステートメントを使用して、外部の C/C++ 関数と Java 関数の宣言、モジュール、ライブラリや、別個の `.ccl` ファイルで定義されているその他の要素を参照できます。**IMPORT** は C/C++ の `#include` に似ています。
- **LOAD MODULE** 文は、モジュールを呼び出して、パラメータを設定し、ストリームとウィンドウへのバインディングを定義することで、データをモジュールに提供し、モジュールからデータを受け取ります。
- 1 つのプロジェクトで、同じモジュールを複数回ロードできます。

プロジェクト

Aleri データ・モデルに代わって、ESP プロジェクトが使用されるようになりました。

Aleri モデルと同様、ESP プロジェクトにはストリーム定義とウィンドウ定義だけでなく、SPLASH コードも格納されます。ただし、柔軟性の点では、サポートするイベント処理オプションや実行時の設定可能性などで ESP プロジェクトの方がはるかに優れています。

ESP プロジェクトに入っているファイルは、次のとおりです。

- 1 つ以上の `.ccl` ファイル (メイン・ファイルと、メイン・ファイルにインポートされるその他のファイルを含みます)
- `.cclnotation` ファイル (CCL をグラフィックで表現する図が入っています)
- 1 つ以上の `.ccr` ファイル (ランタイム・オプションを指定するプロジェクト設定が入っています)
- コンパイル済みプロジェクト用の `.ccx` 実行可能ファイル。

ESP Studio

ESP Studio では、Aleri Studio の編集機能とテスト機能が拡張されました。

ビジュアルとテキストでの作成／編集

ESP Studio でのプロジェクトの編集には、ビジュアル・エディタまたは CCL エディタを使用します。

ビジュアル・エディタでは、CCL 構文の知識がなくてもプロジェクトの作成と編集を実行できます。Aleri Studio のエディタと同様、新しい機能のサポートが追加されています。追加されたサポートには、新しいストリーム・タイプとウィンドウ・タイプ、Aleri Stream のタイプに対応するシンプルなクエリ・オブジェクト、モジュール性などの高度な機能などが含まれます。

CCL エディタでは、CCL ファイルをテキストとして編集できます。また、構文完了オプション、構文チェック、エラー検証も指定できます。プログラマとして経験がある方は、CCL エディタのみで作業したり、CCL エディタをビジュアル・エディタの補助ツールとして使用したりできます。

ビジュアル・エディタと CCL エディタは、いつでも簡単に切り替えられます。一方のエディタで加えた変更は、もう一方のエディタに反映されます。

実行・テスト環境

Studio の実行・テスト環境が見直されて、ナビゲーション・ツールが追加されました。このツールを使用して、複数の ESP Server 上でプロジェクトの開始と停止を実行したり、任意のライブ・プロジェクトに対してテスト・ツールを実行したりできるようになりました。

クラスタ・アーキテクチャ

新しい、高度なクラスタ・アーキテクチャでは、より柔軟性の高い配備がサポートされるようになりました。物理リソースを管理せずに、クラスタ内でプロジェクトの開始と停止を実行できるようになりました。

Aleri ではサーバあたりのライブ・プロジェクト数が 1 つに限られていましたが、Event Stream Processor 5.0 のクラスタ・アーキテクチャでは次の機能が提供されるようになりました。

- ESP Server は、複数の物理マシンにまたがる仮想サーバ (*クラスタ*) として使用できるようになりました。
- ESP Server は、使用可能なハードウェア・リソースの容量に応じて、複数のプロジェクトを同時に実行できるようになりました。
- プロジェクトは、特定のマシン上ではなく、ESP Server 上で開始します。この ESP Server がマシンにプロジェクトを割り当てます。
- プロジェクト内のストリームまたはウィンドウへの接続は、`server.workspace.project.element` で構成される URL を介して確立されます。クラスタがこの URL を解決します。
- プロジェクトは作業領域で実行され、ネームスペースを提供します。
- プロジェクトの開始と停止は、Server 上で実行されている他のすべてのプロジェクトから切り離して実行できます
- プロジェクトはウォーム・スタンバイ対応に設定できます。この場合は、プロジェクトが実行されているマシンで障害が発生しても、プロジェクトはクラスタ内の別のマシンで自動的に再開されます。
- ホット・スタンバイも使用可能です。この場合は、マシンはライブ・バックアップ・インスタンスにフェールオーバするため、再開による遅延が発生しません。
- クラスタ・マネージャは、完全な冗長性を備えています。複数のクラスタ・マネージャがピアとして実行され、シングル・ポイント障害を防ぎます。

動的な継続クエリ

新しいクラスタ・アーキテクチャは、非常に動的な環境をサポートしています。

実行中の Server に、新しいプロジェクトの形式で新しい継続クエリを随時追加できます。不要になったプロジェクトは、他のプロジェクト (ダウンストリーム・プロジェクトを除く) に影響を与えることなく、いつでも停止できます。

たとえば、ライブ・プロジェクト内の既存の出力ストリームで新しい集約を実行するために、既存の出力ストリームをサブスクライブして必要な集約 (さらにフィルタリングやその他の計算) を適用する新しいプロジェクトを作成できます。その後、このプロジェクトをサーバに追加します。これはただちにライブ・プロジェクトになり、既存のプロジェクトをサブスクライブし、停止されるまで実行されます。

データ型

Event Stream Processor では、新しいデータ型がいくつか追加されました。

新しいデータ型と名前が変更されたデータ型

- `bigdatetime` - マイクロ秒の粒度を持つタイムスタンプです。
- `interval` - マイクロ秒の粒度を持ちます。
- `binary` - ロー・バイナリ・バッファを表します。値の最大長はプラットフォームによって異なります。絶対値としての上限はありません。NULL 文字を使用できます。
- `boolean` - true または false です。変換される値は次のとおりです (大文字と小文字は区別されません)。

値	変換先
1、On/ON、Yes/YES/Y、True/TRUE/T	true
0、Off/OFF、No/NO/N、False/FALSE/F	false

サポートされるデータ型の詳細については、『CCL Programmers Guide』を参照してください。具体的なアダプタのデータ型マッピングについては、『アダプタ・ガイド』を参照してください。

注意： 既存のデータ型の名前が、一部変更されました。詳細については、『Migration Guide』を参照してください。

money データ型の機能強化

`money (n)` フォーマットを使用して、グローバルではなく個別に `money` データ型の精度を設定できるようになりました (*n* には 1 から 15 までの値を指定できます)。サポートされる値の範囲は、指定する精度によって異なります。次に例を示します。

- `money(1)`：-922337203685477580.8 から 922337203685477580.7 まで
- `money(15)`：-9223.372036854775808 から 9223.372036854775807 まで

money 定数の精度を明示的に指定するには、**Dn** 構文を使用します (*n* は精度を表します)。たとえば、100.1234567D7、100.12345D5 というように指定します。

精度をグローバルに適用する `money` データ型もサポートされています。

money データ型の使用方法と変換の詳細については、『CCL Programmers Guide』を参照してください。

Pub/Sub API

Event Stream Processor には、新しい Pub/Sub API があります。

Aleri 3.x の Pub/Sub API を使用しているカスタムの統合は、すべて新しい API を使用するように更新してください。

新しい API には、次のような特徴があります。

- パフォーマンスが向上しました。
- 単純化 - プロジェクトを設定してパブリッシュまたはサブスクライブを開始する手順が削減されました。
- C++ API が C API に代わったことでコンパイラへの従属性がなくなり、希望するコンパイラを使用できるようになりました。
- イベントを受け取るためのプログラミング・モデルを 直接モード、コールバック・モード、選択モードの中から選択できるようになりました。
- スレッド・オプションを指定できるようになりました。

詳細については、Event Stream Processor SDK のマニュアルを参照してください。

アダプタ

Event Stream Processor では、複数のアダプタの機能強化が行われています。

- データベース・アダプタが、ODBC で接続されている幅広いデータベースをサポートするようになりました。
- JMS アダプタが、あらゆる JMS プロバイダをサポートするようになりました。
- ほとんどのアダプタが、サポートされているすべてのプラットフォームで使用可能になりました。

分析関数

Event Stream Processor の組み込み関数ライブラリに、70 を超える新しい関数が追加されています。

- **数値関数** – `atan2`、`compare`、`cosd`、`cosh`、`distance`、`distancesquared`、`log2`、`nextval`、`pi`、`power`、`random`、`sign`、`sind`、`sinh`、`tand`、`tanh`
- **ビット処理関数** – `bitand`、 `bitclear`、`bitflag`、`bitmask`、`bitnot`、`bitor`、`bitset`、`bitshiftleft`、`bitshiftright`、`bittest`、`bittoggle`、`bitxor`
- **set 関数** – `avgof`、`coalesce`、`maxof`、`minof`
- **集合関数** – `corr`、`covar_pop`、`covar_samp`、`exp_weighted_avg`、`meandeviation`、`median`、`regr_avgx`、`regr_avgy`、`regr_count`、`regr_intercept`、`regr_r2`、`regr_slope`、`regr_sxx`、`regr_sxy`、`regr_syy`、`var_pop`、`var_samp`、`vwap`
- **文字列関数** – `ascii`、`char`、`left`、`regexp_firstsearch`、`regexp_replace`、`regexp_search`、`trim`、`upper`
- **日付関数と時間関数** – `dateceiling`、`datefloor`、`dateround`、`dayofmonth`、`dayofweek`、`dayofyear`、`timeToUsec`、`hour`、`microsecond`、`minute`、`month`、`second`、`usecToTime`、`to_bigdatetime`、`to_date`、`to_timestamp`、`to_interval`、`year`
- **バイナリ関数** – `extract`、`concat`、`length`、`to_binary`、`hex_binary`、`base64_binary`、`to_string`、`string`、`hex_string`、`base64_string`、`tonetbinary`、`fromnetbinary`
- **カラム・アクセス関数** – `get*columnbyname(record, colname)`、`get*columnbyindex(record, colindex)`、`isInsert`、`isDelete`

プロジェクトの設定

プロジェクトの設定データはビジネス論理から切り離されています。

ホスト名などの物理的な配備設定要素はすべて、ビジネス論理を定義する CCL とは別のプロジェクト設定ファイルに定義されます。このため、テスト環境から運

用環境へのプロジェクトの移動や、複数ロケーションへのプロジェクトの配備が簡単に行えます。

インストールとライセンス

Event Stream Processor では、30 日の猶予期間を含む SySAM のライセンス・メカニズムが使用されています。

サポートされるプラットフォームとオペレーティング・システムの詳細については、『インストール・ガイド』を参照してください。