### クエリ構文

#### コマンド文の形式

クエリは1つ以上のコマンド文で構成されます。コマンド文を構成する基本単位は、コマンド、オプション、対象オブジェクト（object）です。

対象オブジェクトを持つコマンド文

オブジェクトには、ロガーやデータストリーム、ログパーサー、データが保存されているテーブル、テーブルインデックスなどが含まれます。また、サブクエリ文、関数、プロシージャなどで構成された式を呼び出し、その結果値を対象として実行することも可能です。このようなオブジェクトを対象とするコマンド文の構成形式は以下の通りです。

command-name [opt\_1=VALUE] [opt\_2=VALUE] ... OBJECT[, ...]

最も単純なコマンド文の例として、基本システムテーブルであるaraqne\_query\_logsからデータを照会するコマンド文は次のようになります。

table araqne\_query\_logs

対象オブジェクトを持たないコマンド文

オブジェクトを持たないコマンド文は、他のコマンド文からデータを受け取り処理するコマンドで主に使用されます。例えば、[decodedns](https://docs.logpresso.comnull)コマンドが該当します。構成形式は以下の通りです。

FORWADING\_STATEMENT | command-name [opt\_1=VALUE] [opt\_2=VALUE] [opt\_N=VALUE] ...

このようなコマンドは、オブジェクトを渡す先行コマンド文（FORWARDING\_STATEMENT）が返す出力をパイプ（|）を通じて入力として受け取り処理します。

#### パイプによる入力処理

ログプレッソ・ソナーでは、コマンド文の出力をパイプ（|）を利用して他のコマンド文の入力として渡すことができます。例えば、**araqne\_query\_log**テーブルから**login\_name**フィールドが「root」であるログのみを照会するには、次のようにクエリを記述します。

table araqne\_query\_logs | search login\_name == "root"

このクエリは、rootユーザーで実行されたクエリ文を表示します。root文字列を含む行が10分単位で何件発生したかを統計的に集計するには、次のようにクエリを記述します。

table araqne\_query\_logs| search login\_name == "root"| timechart span=10m count

このように、最初のコマンド文の出力を2番目のコマンド文の入力へ、2番目のコマンド文の出力を3番目のコマンド文の入力へと順に渡し、3番目のコマンド文の出力が全体クエリの結果として返されます。クエリ結果は、クライアントのリクエスト形式に応じてディスクに一時的に記録されたり、ネットワーク経由で即時ストリーミングされることもあります。

#### サブクエリ

一部のクエリコマンドは、コマンド文内にネストされたコマンド文（サブクエリ）を実行し、その結果を受け取って処理します。

サブクエリ文は、入力時に角括弧（[ ]）で囲んで表現します。サブクエリは、全体クエリ実行時に親クエリより先に実行されます。サブクエリが返すレコードをクエリコマンドが受け取り、処理を行います。

サブクエリがある場合、コマンド文の構造は以下の通りです。

command [ SUBCOMMAND\_STATEMENT ]

#### コマンド文のコメント処理

コメント処理コマンド「#」を利用して、コマンド行に説明を挿入したり、単一または連続したコマンド行をコメントアウトできます。クエリ入力欄では、コメントアウトされたコマンド行はグレーで表示されます。

単一コマンド行のコメント処理

コマンド文の先頭に「#」を挿入することでコメントアウトできます。

「#」の後ろには空白文字が必要です。空白がない場合、コメントとして認識されません。

# 直近1時間のsys\_cpu\_logに記録されたCPU使用率を照会| table duration=1h sys\_cpu\_logs| # eval total = kernel + user

上記の例では、「直近1時間のsys\_cpu\_logに記録されたCPU使用率を照会」と「eval total = kernel + user」がコメントアウトされ、「table duration=1h sys\_cpu\_logs」だけが実行されます。

複数行コマンドのコメント処理

「# 」と「」でコメントアウトしたいコマンド行を囲むことで、複数行をまとめてコメントアウトできます。

table duration=1h sys\_cpu\_logs| # [ eval total = kernel + user| search total > 10 ]| sort \_time

上記の例では、角括弧内のコマンド文がコメントアウトされ、実際に実行されるクエリは「table duration=1h sys\_cpu\_logs | sort \_time」となります。

コメント処理コマンド「#」は、後続の文字列だけでなく、サブクエリを囲む角括弧（[ ]）内のパイプ（|）も無視します。サブクエリ内に改行があっても無視されます。つまり、サブクエリ文全体を無視し、角括弧外にパイプが現れるまでコメントアウトします。

table sys\_cpu\_logs| # union [ table sys\_cpu\_logs | limit 30 ]| eval total = kernel + user

上記の例では、サブクエリ「union table sys\_cpu\_logs | limit 30」がすべてコメントアウトされ、実際に実行されるクエリは「table sys\_cpu\_logs | eval total = kernel + user」となります。

#### クエリパラメータ

クエリパラメータに値を割り当て、必要な時に呼び出して利用できます。定数の代わりに関数などを用いた式を利用することで、動的に値を割り当ててクエリを実行する際に便利です。例えば、予約クエリを実行する際に現在の日付を基準に1週間分のデータを照会したり、プロシージャ実行時にユーザーが入力したパラメータ値を利用してクエリを実行する場合などにクエリパラメータを使用します。

パラメータの宣言

パラメータは、[set](https://docs.logpresso.comnull)または[setq](https://docs.logpresso.comnull)コマンドを利用して宣言します。

パラメータの参照

パラメータに割り当てられた値は、パラメータ参照関数[$()](https://docs.logpresso.comnull)で参照します。

#### 関数

クエリ文内で関数を使用できます。関数は、式を利用できる場所であればどこでも使用可能です。

#### プロシージャ

ログプレッソ・ソナーは、事前定義されたクエリ文を関数のように呼び出せるプロシージャ機能を提供します。DBMSのプロシージャと類似した機能であり、以下のような利点があります。

**再利用性および保守性の向上**

プロシージャを通じて特定機能を提供するクエリをモジュール化することで、再利用性を高めることができます。ユーザーにはプロシージャ名と利用するパラメータのみを伝えればよく、繰り返しクエリを書く必要がなくなり、クエリ文が簡潔になり保守も容易になります。

**セキュリティの向上**

[dbquery](https://docs.logpresso.comnull)、[ftp](https://docs.logpresso.comnull)、[sftp](https://docs.logpresso.comnull)など外部システムへ接続するコマンドはプロファイル権限が必要です。プロファイル権限をユーザーに直接付与すると、外部システムで任意の操作が可能となり安全ではありません。ユーザーが特定の管理者権限が必要なコマンドを実行したり、ローカル／リモートホストで任意の操作ができるコマンドをプロシージャとして構成し、ユーザー権限を管理することで、システム全体の管理者権限を付与せずに必要な操作をユーザーアカウントで実行できるようにします。元データの一部のみを照会するよう制限したり、元データをマスキングして表示する方式にも応用できます。

**ログプレッソ・ソナーの設定情報へのアクセス**

ログプレッソ・ソナーのシステムテーブルは管理者権限が必要です。システム設定情報へユーザーがアクセスする必要がある場合、プロシージャを通じてアクセスできるようにします。

プロシージャの定義

Webコンソールでプロシージャを定義・管理できます。プロシージャ管理機能は以下のパスで利用可能です。

* （ENT, STD）**クエリ > プロシージャ**
* （SNR）**分析 > プロシージャ**

プロシージャとして利用するクエリ文には、プロシージャ呼び出し時に利用するパラメータやユーザー定義フィールドを含めることができます。

プロシージャ画面で定義するクエリは、[$()](https://docs.logpresso.comnull)関数を利用して、ユーザーがプロシージャ呼び出し時に渡したパラメータを参照できます。例は以下の通りです。

table duration=1d sys\_cpu\_logs | search kernel + user >= $("threshold")

この例では、thresholdがパラメータとなります。

プロシージャ作成時によくある誤りは、[$()](https://docs.logpresso.comnull)関数の参照がマクロのように置換されると考えてクエリを記述することです。[$()](https://docs.logpresso.comnull)関数は、クエリコマンドで式を入力できる位置でのみ指定可能です。例えば、以下のプロシージャは[dbquery](https://docs.logpresso.comnull)が任意のSQL文入力を式としてサポートしていないため、正しいクエリではありません。

dbquery USERDB $("sql")

プロシージャの呼び出し

プロシージャを呼び出して実行するコマンドとしては、[proc](https://docs.logpresso.comnull)があります。呼び出し方法はコマンドの説明を参照してください。