



オンライン技術勉強会 Qlik Sense アプリ開発 ベーシックトレーニング

データ変換を実現するロードスクリプト

クリックテック・ジャパン株式会社

2025年 2月 12日

Agenda

- データ準備
- 環境準備
- スクリプトの編集とモデルの確認
- スクリプトの作成とデータフローの作成画面
- データソースへの接続とデータの取り込み
- ロードスクリプトの基本
- ロードスクリプトによる高度なデータ変換
- その他の便利なスクリプト
- スクリプト Tips
- Section Access
- 3 日目のまとめ
- Q&A
- その他の情報
- Appendix

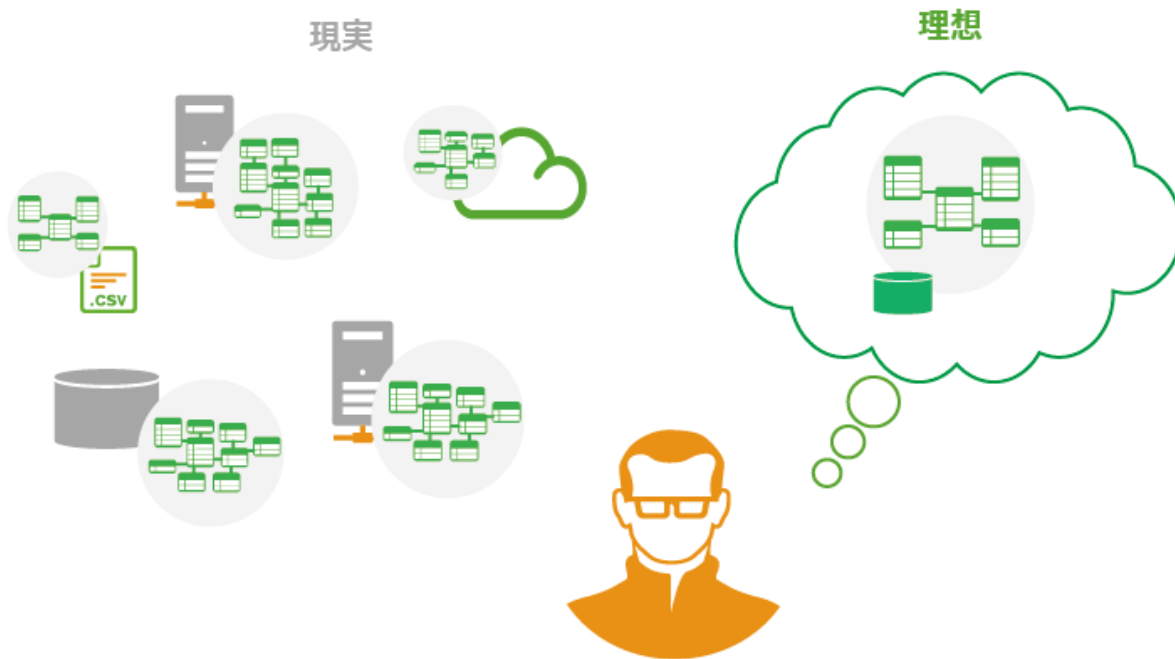
データ準備



データ準備とは

データ準備とは、データの集計・可視化・分析の直前に、必要に応じてデータを変換することです。

- 分析用に取得したデータが、必ずしもそのまま分析に必要な形になっているわけではない。
- その時の分析要件に合わせてデータを変換したい（データのラストワンマイル）
- 異なるソースからなる複数のデータを統合して分析したい。

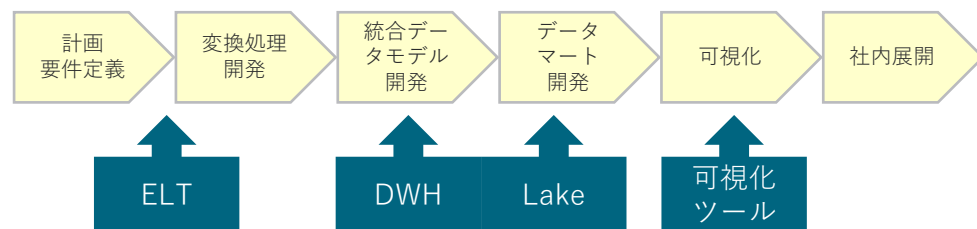


データ準備におけるQlik Senseの特徴

Qlik Senseは(一般的なBIと異なり)データ準備機能を備えているため、DWH/ETLに依存することなくデータ準備から可視化・分析までをワンツールで一気通貫に行えます。

一般的なBIツール

- データウェアハウスやデータレイクに分析できる状態にまで整備されたデータが格納されていることが前提
- 基幹システムからETL(Extract/Transform/Load)ツールでデータを抽出・変換
- 必要なサブセットのデータを取り出して集計を実施
- 新たなデータソースを追加した場合、これらすべてのシステムで変更が必要



Qlik Sense

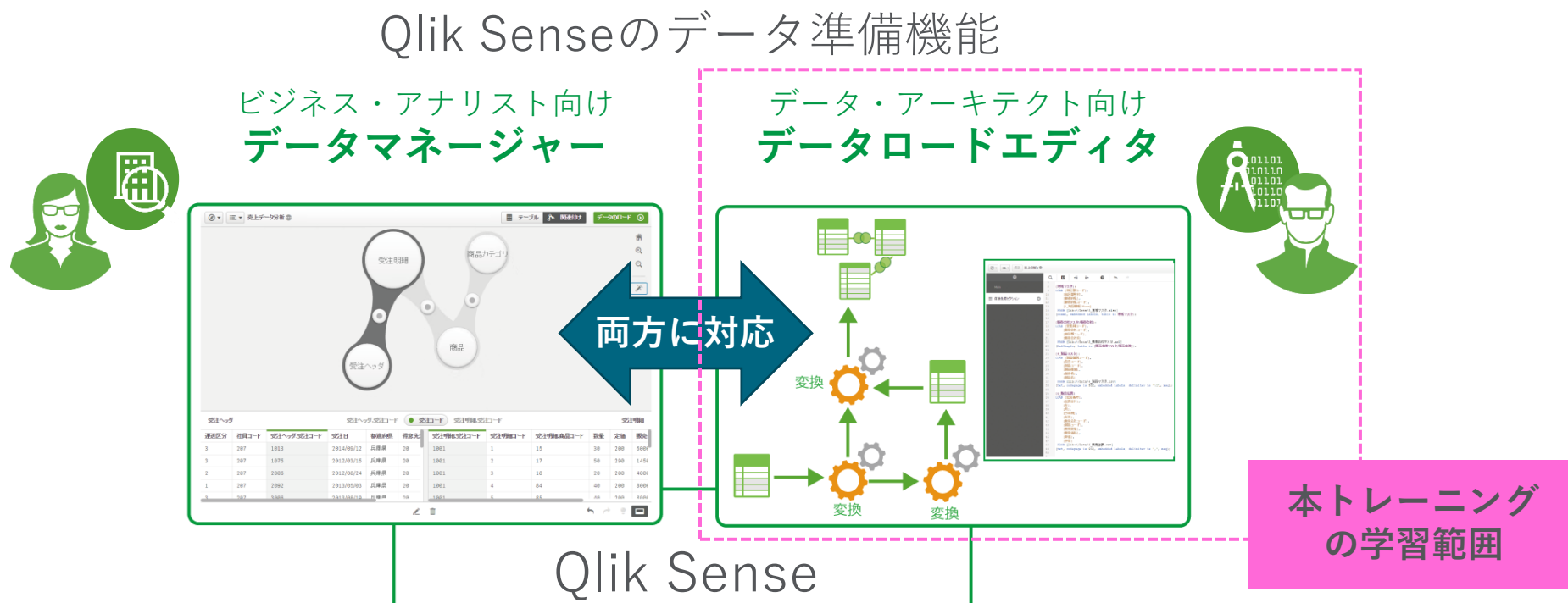
- データ準備機能を備えているため様々なシステムから直接データを抽出できる
- データ準備～ビジュアライゼーションまでを一気通貫で実施可能
- ワンツールで簡単な操作でのデータ準備と、複雑なデータ変換の両方に対応できる
- プロジェクト初期段階から実機上で検証可能



Qlik Senseのデータ準備機能

Qlik Senseには、ビジネスユーザー向けのセルフサービスデータ準備機能「データマネージャー」と、より複雑なデータ変換を行う「データロードエディタ」の2種類のデータ準備機能が用意されています。

本トレーニングでは複雑なデータ変換に対応するためのスキルを会得するため、データロードエディタの使い方を学習します。



環境準備



前提

- 本セッションでは、Cloud の Qlik Sense を利用します。
- 接続先のデータソースの設定の説明は致しません。ヘルプや過去のセミナーを参照してください。
- アプリケーションは、個人スペースに作成します。データファイルも個人スペースに格納されているものとしてします。
- アプリとデータは、事前に配布されたコンテンツを利用します。

アプリ： *Qlik Sense Basic Training Day 3.qvf*

データ： *SampleData.xlsx*

アプリのアップロード

個人用スペースに、Qlik Sense Basic Training Day 3.qvfをアップロードします。

The screenshot displays the Qlik Sense 'カタログ' (Catalog) page. A blue circle highlights the 'スペース' (Space) dropdown menu, which is currently set to '個人用' (Personal). A blue arrow points from the 'アップロード' (Upload) button in the right-hand menu to the 'アップロード' dialog box. The dialog box shows a file upload area with the text 'ファイルをここにドロップ' (Drop files here) and '対応するファイルタイプ: .qvw, .qvf'. A yellow callout box points to the file name 'Qlik Sense Basic Training Day 3.qvf'. Below the upload area, the 'スペース' (Space) dropdown is set to '個人用', and the 'タグ' (Tag) field is empty. The 'アップロード' button is at the bottom right of the dialog box. The background shows the 'カタログ' page with various filters and a list of items.

Qlik 分析

コンテンツを検索

Insight Advisor に質問する

+ 新規作成

ホーム

作成

お気に入り

コレクション

カタログ

可視化と分析

データを準備

予測

自動化

アラート

サブスクリプション

はじめに

カタログ

名前で絞り込む

スペース 個人用

タイプ

所有者

すべてのフィルター

個人用

アップロード

アプリ

スクリプト

ファイルをここにドロップ

対応するファイルタイプ: .qvw, .qvf

参照

スペース

個人用

タグ

検索

スペースに移動する

キャンセル

アップロード

アップロード

既存の app とスクリプトを追加

詳細を見る

スクリプト

データの結合と変換

スペース

コンテンツへのアクセスを共有および制御

データセット

データ ファイルをアップロードするか、外部データを登...

データ接続

新しいデータ接続を確立する

メモ

分析でコラボレーション

リンク

外部リソースへのリンクを追加

自動化

ビジネス ワークフローを作成

分析アプリ

データを追加して探索

田中 作

App Analyzer v6.1.8

更新済み 16時間前

test script

更新済み 2024年6月11日

Qlik

11

サンプルデータの準備

事前に配布済みのサンプルデータ(SampleData.xlsx)をDatafilesに取り込みます。

The screenshot displays the Qlik Sense web application interface. The top navigation bar includes the Qlik logo, '分析アプリ' (Analysis App), and a dropdown menu for 'データロードエ...' (Data Load Editor). The main toolbar contains icons for 'セクション' (Section), search, and various view options. The left sidebar shows a list of sections, with 'Main' selected. The central workspace is divided into two panes. The left pane, titled 'ファイルを選択' (Select File), shows a file browser with a list of files including 'Test1', 'Test2', and several 'app_analyzer' files. The right pane, titled 'データの追加' (Add Data), shows a list of data sources including 'Amazon_S3' and 'DataFiles'. A blue arrow points from the 'DataFiles' folder in the right pane to the 'SampleData.xlsx' file in the left pane. A yellow callout box labeled 'SampleData.xlsx' is positioned over the file in the left pane. A grey callout box at the bottom center contains the text '事前に配布されたデータファイル' (Data file distributed in advance).

Qlik 分析アプリ データロードエ...

セクション

Main

スクリプトの記載方法

*** データソースへの接続と

Data Fileからの取り込み

DBMSからの取り込み

Slackからの取り込み

*** ロードスクリプトの基本

Load

先行 Load

Resident Load

*** ロードスクリプトによるデータロード

ファイルを選択

パス

<DataFiles>

Test1

Test2

app_analyzer_AppRAM_3.0.1.qvd

app_analyzer_AppRAM_6.1.8.qvd

app_analyzer_AppReloadCPUMetadata_3.0.1.qvd

app_analyzer_AppReloadCPUMetadata_6.1.8.qvd

app_analyzer_Consumption_6.1.8.qvd

ファイルの種類

全てのテーブルファイル

ここにファイルをドロップするか、ファイルをクリックして選択します

SampleData.xlsx

キャンセル 選択

データの追加

+ データカタログから追加

データ接続

接続の新規作成

個人用

検索

Amazon_S3

Amazon S3

DataFiles

フォルダ

File (via Direct Access gateway)_CDataDemo Data

File (via Direct Access gateway)

事前に配布されたデータファイル

スクリプトの編集とモデルの確認

データロードエディタ

データモデルビューア

データロードエディタの構成

The screenshot shows the Qlik Data Load Editor interface. Key components are highlighted with blue boxes and labels:

- セクション (Section):** Located in the top-left toolbar.
- テキストエディタ (Text Editor):** The central area containing the load script code.
- データソース画面 (Data Source Screen):** Located in the top-right toolbar, containing options like 'データソース' (Data Source), 'データカタログから追加' (Add from Data Catalog), 'データ接続' (Data Connection), and 'データプレビュー' (Data Preview).
- プレビュー画面 (Preview Screen):** The bottom section displaying a table of data.

The load script in the text editor is as follows:

```
1 SET ThousandSep=',';
2 SET DecimalSep='.';
3 SET MoneyThousandSep=',';
4 SET MoneyDecimalSep='.';
5 SET MoneyFormat='¥#,##0;-¥#,##0';
6 SET TimeFormat='h:mm:ss';
7 SET DateFormat='YYYY/MM/DD';
8 SET TimestampFormat='YYYY/MM/DD h:mm:ss[.fff]';
9 SET FirstWeekDay=6;
10 SET BrokenWeeks=1;
11 SET ReferenceDay=0;
12 SET FirstMonthOfYear=1;
13 SET CollationLocale='ja-JP';
14 SET CreateSearchIndexOnReload=1;
15 SET MonthNames='1月;2月;3月;4月;5月;6月;7月;8月;9月;10月;11月;12月';
16 SET LongMonthNames='1月;2月;3月;4月;5月;6月;7月;8月;9月;10月;11月;12月';
17 SET DayNames='月;火;水;木;金;土;日';
18 SET LongDayNames='月曜日;火曜日;水曜日;木曜日;金曜日;土曜日;日曜日';
19 SET NumericalAbbreviation='3:千;6:百万;8:億;12:兆;16:京;20:垓;24:垺;28:京;32:垓;36:垺;40:京;44:垓;48:垺;52:京;56:垓;60:垺;64:京;68:垓;72:垺;76:京;80:垓;84:垺;88:京;92:垓;96:垺;100:京';
```

The preview table at the bottom contains the following data:

受注コード	売上日	年月度	商品コード	得意先コード	出荷先都道府県	出荷先住所	出荷先名	社員コード	納品書番号	数量	単価	割引	販売金額	原価	粗利	利益
10001	2022/08/09	202208	P15	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0002	30	200	2160	3840	2390	1450	
10002	2022/08/09	202208	P17	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0003	50	290	5510	8990	4570	4420	
10003	2022/08/09			T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0004	20	200	840	3160	2860	300	
10004	2022/08/09			T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0005	40	200	1360	6640	6120	520	
10005	2022/08/09	202208	P85	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0006	40	200	2960	5040	280	4760	
10006	2022/08/09	202208	P86	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0007	40	200	1360	6640	4430	2210	
10007	2022/08/07	202208	P20	T22	福岡県	8112206粕屋郡志免町御手洗51-X	食所あんど本店	E304	201808-IN0008	30	320	3552	6048	7500	-1452	

主な機能

機能	概要
セクション	スクリプトをセクションに分けて、読みやすく、管理しやすくします。セクションは上から下に実行されます。
履歴	以前のバージョンのロード スクリプトを表示および管理できます。
データソース	よく使用するデータ ソース (データベースまたはリモート ファイル) へのショートカットを保存できます。ここからロードするデータを選択できます。データ接続は、アクセス権のあるスペース別にソートされます。
テキストエディタ	スクリプト コードを記述、編集することができます。スクリプトの各行には番号が振られ、スクリプトは構文のコンポーネント別に色分けされます。
プレビュー	[データのプレビュー] または [データのロード] のいずれかを使用してスクリプトを実行すると、[プレビュー] でロードされたテーブルのプレビューを表示できます。テーブルビュー、リストビュー、タイルビューを表示できます。
デバッグ	ブレイクポイントを使用してスクリプトを実行できます。変数の値や出力を調べることができます。
データをロード	スクリプトを実行して、データをロードします。データロードの実行経過が、ウィンドウに表示されます。ロードが完了したら、編集を続けるボタンを押してください。

データをプレビュー

Qlik 分析アプリ データロードエ...

セクション

1 SET ThousandSep=',';

名前	ユニーク値	データの種類	NULL 値	サンプル値
受注コード	4,009	INTEGER	0	10,001, 10,002, 10,003, 10,004, 10,005
売上日	408			44,592, 44,620, 44,594, 44,601, 44,570
年月度	36	INTEGER	0	202,201, 202,208, 202,202, 202,209, 202,207
商品コード	123	STRING	0	P123, P97, P81, P10, P8
得意先コード	50	STRING	0	T9, T10, T26, T11, T50

リストビュー

出荷先住所
ユニーク値: 51

出荷先名
ユニーク値: 88

社員コード
ユニーク値: 9

タイルビュー

dammy 2,027
3810022長野市大豆島1798-X 68
2750016留志野市津田沼2-6-XX 64
3001203牛久市下根町1504-XX 57
その他 1,793

寿スプアー 110
高瀬亭 108
酒蔵でん 102
その他 3,545

E110 630
E107 614
E105 522
E305 448
その他 1,795

データのプレビュー

データのロードが完了しました。

経過時間: 00:00:03

データのロードを開始しました

受注
フェッチした行: 100

アプリが保存されました

正常に終了しました

の強制エラー

の合成キー

☐ 正常に終了した後、閉じる

編集を続ける

合計行数: 4,009

受注コード	売上日	年月度	商品コード	得意先コード	出荷先都道府県	出荷先住所	出荷先名	社員コード	納品書番号	数量	単価	割引	販売金額	原価	粗利	利益
10001	2022/08/09	202208	P15	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0002	30	200	2160	3840	2390	1450	
10002	2022/08/09	202208	P17	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0003	50	290	5510	8990	4570	4420	
10003	2022/08/09	202208	P18	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0004	20	200	840	3160	2860	300	
10004	2022/08/09	202208	P84	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0005	40	200	1360	6640	6120	520	
10005	2022/08/09	202208	P85	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0006	40	200	2960	5040	280	4760	
10006	2022/08/09	202208	P86	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0007	40	200	1360	6640	4430	2210	
10007	2022/08/07	202208	P20	T22	福岡県	8112206粕屋郡志免町御手洗51-X	食所あんど本店	E304	201808-IN0008	30	320	3552	6048	7500	-1452	

ページあたりの行数: 100 1-100 of 4009

ページ 1

履歴管理

履歴

変更は自動的に保存されます。リストを使用して、以前のバージョンを復元します。

本日

- SA 1月15日 19:25 Satoshi Abe
- SA 1月15日 19:21 Satoshi Abe

① 最近の50件のバージョンが保存されます。

バージョンの名前を変更する
復元
ダウンロード
削除

Qlik 分析アプリ データロードエ...

Basic Training Qlik Sense Basic Training (ロードスクリプト) ...

セクション

Binary

Main

スクリプトの記載方法

Data Fileからの取り込み

DBMSからの取り込み

Slackからの取り込み

Load

先行 Load

Resident Load

Concatenate

NoConcatenate

Group By

```
1 SET ThousandSep=',';
2 SET DecimalSep='.';
3 SET MoneyThousandSep=',';
4 SET MoneyDecimalSep='.';
5 SET MoneyFormat='¥#,##0;-¥#,##0';
6 SET TimeFormat='h:mm:ss';
7 SET DateFormat='YYYY/MM/DD';
8 SET TimestampFormat='YYYY/MM/DD h:mm:ss[.fff]';
9 SET FirstWeekDay=6;
10 SET BrokenWeeks=1;
11 SET ReferenceDay=0;
12 SET FirstMonthOfYear=1;
13 SET CollationLocale='ja-JP';
14 SET CreateSearchIndexOnReload=1;
15 SET MonthNames='1月;2月;3月;4月;5月;6月;7月;8月;9月;10月;11月;12月';
16 SET LongMonthNames='1月;2月;3月;4月;5月;6月;7月;8月;9月;10月;11月;12月';
17 SET DayNames='月;火;水;木;金;土;日';
18 SET LongDayNames='月曜日;火曜日;水曜日;木曜日;金曜日;土曜日;日曜日';
19 SET NumericalAbbreviation='3:千;6:百万;8:億;12:兆;16:京;20:垓;24:垓;24:Y;-3:m;-6:μ;-9:n;-12:p;-15:f;-18:a;-21:z;-24:y';
```

プレビュー 受注 出力

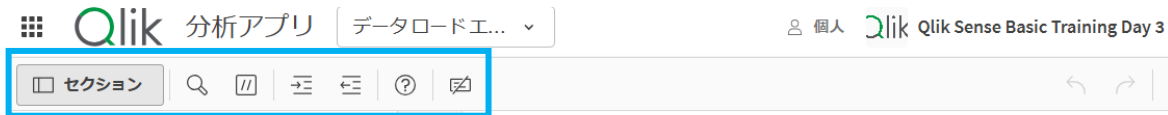
数量	受注コード	売上日	年月度	商品コード	得意先コード	出荷先都道府県	出荷先住所	出荷先名	社員コード	納品書番号	単価	割引	販売金額	原価	粗利	利益
30	10001	2022/08/09	202208	P15	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0002	200	2160	3840	2390	1450	
50	10002	2022/08/09	202208	P17	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0003	290	5510	8990	4570	4420	
20	10003	2022/08/09	202208	P18	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0004	200	840	3160	2860	300	
40	10004	2022/08/09	202208	P84	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0005	200	1360	6640	6120	520	
40	10005	2022/08/09	202208	P85	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0006	200	2960	5040	280	4760	
40	10006	2022/08/09	202208	P86	T26	愛知県	4860969春日井市味美白山町1-9-X	小料理ひろ春日井店	E210	201808-IN0007	200	1360	6640	4430	2210	
30	10007	2022/08/07	202208	P20	T22	福岡県	8112206粕屋郡志免町御手洗51-X	食所あんど本店	E304	201808-IN0008	320	3552	6048	7500	-1452	

合計行数: 4009

ページあたりの行数: 100 1-100 of 4009

ページ 1

ツールバー



セクション

Main

スクリプトの記載方法

*** データソースへの接続とデー...

Data Fileからの取り込み

DBMSからの取り込み

Slackからの取り込み

*** ロードスクリプトの基本 ***



オートコンプリートのオン/オフ

ヘルプモード

インデントを解除

インデントを設定

コメント設定/解除

検索と置換

ヘルプモードでは、予約語のリンクをクリックするとヘルプが表示されます。

```
1 SET ThousandSep=',';
2 SET DecimalSep='.';
3 SET MoneyThousandSep=',';
4 SET MoneyDecimalSep='.';
5 SET MoneyFormat='¥#,##0;-¥#,##0';
6 SET TimeFormat='h:mm:ss';
7 SET DateFormat='YYYY/MM/DD';
8 SET TimestampFormat='YYYY/MM/DD h:mm:ss[.fff]';
9 SET FirstWeekDay=6;
10 SET BrokenWeeks=1;
11 SET ReferenceDate=;
```


データモデルビューア

アプリに追加したデータの構造やメタデータを表示します。

The screenshot displays the Qlik Data Model Viewer interface. On the left is a navigation pane with categories: ホーム (Home), 分析 (Analysis), 話す (Talk), 準備 (Prepare), ビジネスロジック (Business Logic), and 語彙 (Vocabulary). The 'データモデルビューア' (Data Model Viewer) option is highlighted under the '話す' category. The main area shows a data model diagram with three tables: '受注' (Orders), '取引先' (Customers), and '商品' (Products). The '受注' table is connected to both '取引先' and '商品'. The '商品' table is connected to '商品区分' (Product Categories). A label 'モデルの表示' (Model Display) is present. Below the diagram is a '▼プレビュー' (Preview) section. It includes a '受注' table with columns: 行 (Row), 項目 (Item), キー (Key), タグ (Tag), and a 'サンプルデータの表示' (Sample Data Display) table with columns: 売上店舗ID (Sales Store ID), 取引先コード-得意先コード (Customer Code - Client Code), 商品コード (Product Code), 都道府県-出荷先都道府県 (Prefecture - Shipping Prefecture), 受注コード (Order Code), 売上日 (Sales Date), 年月度 (Year/Month/Day), 出荷先住所 (Shipping Address), 出荷先名 (Shipping Name), 社員コード (Employee Code), 納品書番号 (Invoice Number), 数量 (Quantity), and 単価 (Unit Price). A label 'メタデータの表示' (Metadata Display) is also present.

ホーム

- 概要
- 分析
 - シート
- 話す
 - ストーリーテリング
 - レポート
- 準備
 - データマネージャー
 - データロードエディタ
 - データモデルビューア**
- ビジネスロジック
 - 論理モデル
- 語彙

データモデルビューア

個人 ユーザー向けアプリ

モデルの表示

受注

- 売上店舗ID
- 取引先コード-得意先コード
- 商品コード
- 都道府県-出荷先都道府県
- 受注コード
- 売上日
- 年月度
- 出荷先住所
- 出荷先名
- 社員コード
- 納品書番号
- 数量
- 単価
- 割引
- 販売金額
- 原価
- 粗利
- 利益率
- 運送区分
- 気象データ

取引先

- 取引先コード-得意先コード
- フリガナ
- 取引先名
- 担当部署
- 得意先種別

商品

- 商品コード
- 商品区分コード
- 商品カナ
- 商品名
- 仕入先コード
- 梱包単位
- 商品単価
- 在庫
- 発注済
- 発注点

商品区分

- 商品区分コード
- 商品区分名
- 説明

▼プレビュー

受注

行	項目	キー	タグ
4009	20	4	Snumeric Sinteger Skey Sascii Stext Stimestamp Sdate

メタデータの表示

サンプルデータの表示

売上店舗ID	取引先コード-得意先コード	商品コード	都道府県-出荷先都道府県	受注コード	売上日	年月度	出荷先住所	出荷先名	社員コード	納品書番号	数量	単価
2	T26	P15	愛知県	1001-1	2022/08/09	202208	4860969 春日井市味美白山町 1-9-X	小料理ひろ 春日井店	E210	201808-IN0002	30	
2	T26	P17	愛知県	1001-2	2022/08/09	202208	4860969 春日井市味美白山町 1-9-X	小料理ひろ 春日井店	E210	201808-IN0003	50	
2	T26	P18	愛知県	1001-3	2022/08/09	202208	4860969 春日井市味美白山町 1-9-X	小料理ひろ 春日井店	E210	201808-IN0004	20	
2	T26	P19	愛知県	1001-4	2022/08/09	202208	4860969 春日井市味美白山町 1-9-X	小料理ひろ 春日井店	E210	201808-IN0005	40	
2	T26	P85	愛知県	1001-5	2022/08/09	202208	4860969 春日井市味美白山町 1-9-X	小料理ひろ 春日井店	E210	201808-IN0006	40	
2	T26	P86	愛知県	1001-6	2022/08/09	202208	4860969 春日井市味美白山町 1-9-X	小料理ひろ 春日井店	E210	201808-IN0007	40	
2	T22	P20	福岡県	1002-7	2022/08/07	202208	8112206 粕屋郡志免町御手洗 51-X	食所あんど 本店	E304	201808-IN0008	30	

スクリプトの作成とデータフロー の作成画面

スクリプト
データフロー

作成 – スクリプトとデータフロー

The image displays the Qlik Sense '作成' (Create) interface. The main view shows a grid of content types for creation, including 'スクリプト' (Script) and 'データフロー' (Data Flow), both of which are highlighted with blue boxes. Arrows from these boxes point to an inset window showing the 'カタログ' (Catalog) view. In the catalog, 'Script Sample' and 'Data Flow Sample' are also highlighted with blue boxes, indicating the next steps in the workflow.

Qlik 分析

コンテンツを検索

Insight Advisor に質問する

推奨を表示

作成

コンテンツとコラボレーション

- MLの実験**
履歴データを自動機械学習の実験にロードします。
詳細を見る
- スクリプト**
データをロード、変換、エクスポートするスクリプトを構築します。
詳細を見る
- スペース**
スペースを使用して、コンテンツへのアクセスを共有および制御します。
詳細を見る
- データフロー**
データを準備するためのデータフローを作成します。
詳細を見る
- メモ**
個人分析または共有分析を作成します。
詳細を見る
- リンク**
役立つコンテンツへのリンクを追加します。
詳細を見る

自動化
コード不要のビジュアルインターフェイスを使用して、自動化されたタスクを実行します。
詳細を見る

分析アプリ
データを視覚化して表示するアプリケーション。
詳細を見る

用語集
用語と定義を含むビジネス用語集。
詳細を見る

アップロード
既存の資産を Qlik Cloud にアップロードします。
詳細を見る

データとアップロード

カタログ

名前で絞り込む

スペース: **すべて** | タイプ: | 所有者: | 全てのフィルター

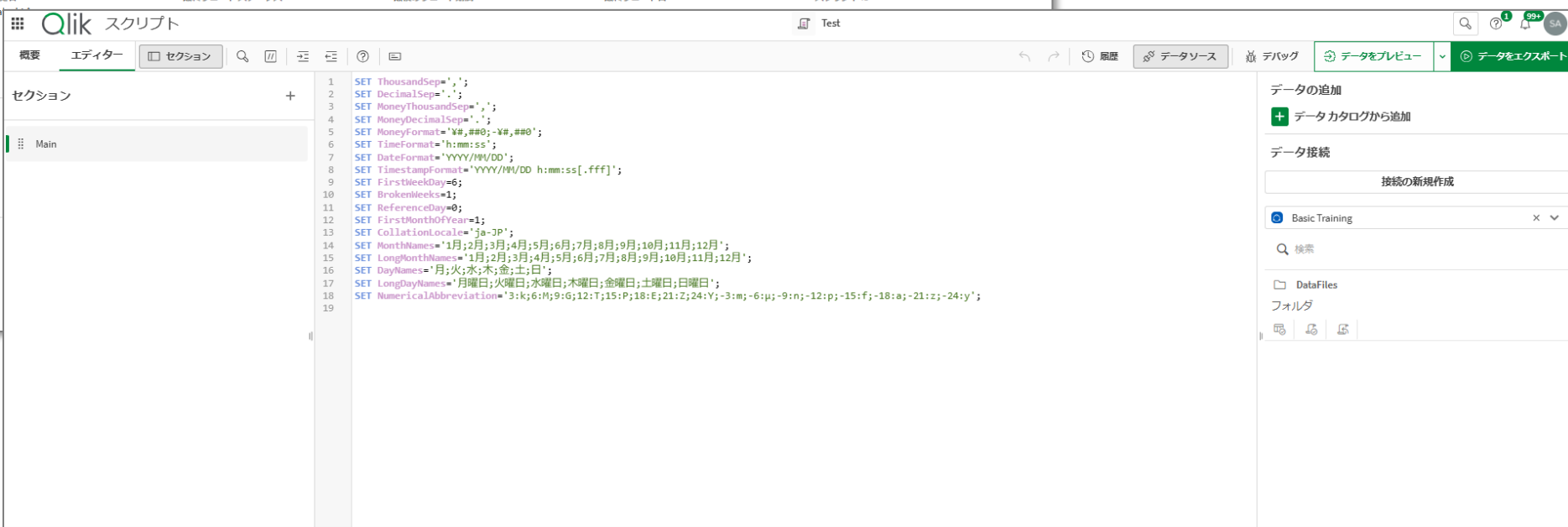
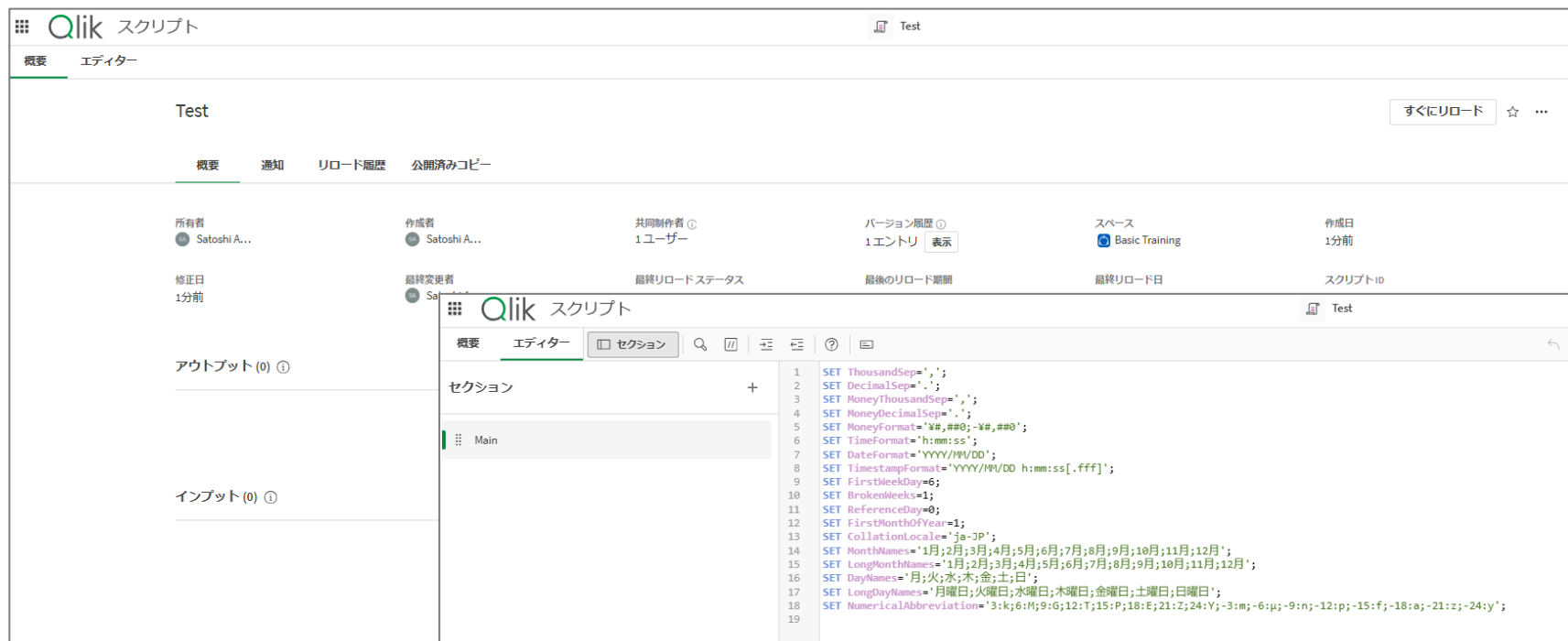
- Script Sample**
更新済み 数秒前
- Data Flow Sample**
更新済み 数秒前
- Qlik Sense Basic Training Day 3**
更新済み 2日前

Qlik

スクリプト

データをロード、変換、エクスポートするスクリプトを構築します。データの分析や視覚化には使用できません。

スクリプトを使用すると、分析アプリを必要とせずにデータをロードして変換できます。変換を再利用して、複数のアプリにデータを提供できます。



データフロー

データセットを視覚的に組み合わせて、直感的なドラッグアンドドロップインターフェイスを提供します。

The screenshot displays the Qlik Data Flow editor. On the left, a sidebar lists processors such as 'フィルター' (Filter), '項目を選択' (Select items), '結合' (Join), 'フォーク' (Fork), '集計' (Aggregate), 'ソート' (Sort), and '項目を削除' (Delete item). The main workspace, labeled 'ビジュアルエディタ' (Visual Editor), shows a data flow: 'Transaction-Japan データセットソース' (Transaction-Japan Data Set Source) → 'ハッシュ 1' (Hash 1) → 'データファイル...' (Data File...). The 'ハッシュ 1' node is configured with 'email' as the hashing item. At the bottom, the 'データプレビュー' (Data Preview) table shows data for the 'ハッシュ 1' node.

伝票番号	日付	会社名	製品ID	製品名	合計金額	粗利	通貨	担当営業	email	緯度	経度
1001	2022/07/05	Photobean	52125-212	Clams - Canned	97,906,630	791,600	JPY	Katharina Red	PW);N900C(5=C^B!#7"EVp"<LT\X1C6V,CQE57129"	25.890008	119.58
1002	2022/09/08	Trudoo	59676-566	Extract - Vanilla,artificial	97,808,770	311,500	JPY	Jarib Sulland	H+Y*YFRP"&XA#` `QLQ9EG\,ZFG*#+[]&x76)//7%L6	11.2220458	12.023
1003	2022/09/02	Quaxo	68788-9764	Syrup - Monin - Granny Smith	76,488,370	291,640	JPY	Skippy Fuxman	O,8!-B<6>7.-0+X&]]15UZ/<`B#(VX025)*>FN6HOE,	48.9848042	16.402
1004	2022/02/21	Skippe	0004 7822	Bread - Raisin	25,261,700	670,180	IDV	Moira Mackalle	KV-0418**A H4731-TADA#B/NI 0%6EH 06/11/11 05:57 E7	25.7222420	100.267

豊富なプロセッサパレットを使用すると、文字列、数値、日付、クレンジング関数などのデータ操作操作や、データの結合、集計、フィルタリングなどの再構築機能など、データのクレンジングと整形を行うことができます。

データソースへの接続と データの取り込み



Qlik Senseにはデータ接続のための様々なコネクタが標準装備されています。



ファイルの取り込み

Qlik Sense では、さまざまなファイル形式からデータを読み込むことができます。
ここでは、Excelファイルの取り込み方法を解説します。

1 データの追加
データカタログから追加
データ接続
接続の新規作成
Basic Training
検索
Amazon_S3_V2 - qlikfordemo
Amazon S3 V2
DataFiles
フォルダ

2 ファイルを選択
パス
<DataFiles>
実績データ.csv
受注.csv
売上データ.xlsx
売上明細.xlsx
ファイルの種類
全てのテーブルファイル

3 売上データ.xlsx からデータを選択する
テーブル → ファイル形式
Excel (XLSX)
項目名
埋め込まれた項目名
ヘッダー サイズ
- 0 +
読み込むテーブルを指定し、内容を確認します。

顧客マスタ 7
売上データ 3

受注日
2019/02/01
2019/02/01
2019/02/02
2019/02/02
2019/02/03
2019/02/03
2019/02/04
2019/02/04
2019/02/05
2019/02/05
2019/02/06
2019/02/06
2019/02/07
2019/02/07
2019/02/08
2019/02/08
2019/02/09
2019/02/09

データプレビュー

```
1 // // アップロードされたファイルからデータを取り込む
2
3 LOAD
4     顧客ID,
5     姓,
6     名,
7     性別,
8     血液型,
9     生年月日,
10    年齢
11 FROM [lib://Basic Training:DataFiles/売上データ.xlsx]
12 (ooxml, embedded labels, table is 顧客マスタ);
13
14 LOAD
15     受注日,
16     顧客コード,
17     購入金額
18 FROM [lib://Basic Training:DataFiles/売上データ.xlsx]
19 (ooxml, embedded labels, table is 売上データ);
20
21
```

キャンセル スクリプトを挿入

4 「スクリプトを挿入」で、テキストエディタにロードスクリプトが挿入されます。

RDBMSからのデータの取り込み

SQL Server や MySQL等のDBやODBC経由で様々なデータソースからデータを取り込むことができます。ここでは、MySQLからのデータの取り込み方法を解説します。

データの追加

+ データカタログから追加

データ接続

接続の新規作成

Basic Training

検索

Amazon S3 V2 - qlikfordemo

Amazon S3 V2

DataFiles

フォルダ

MySQL Enterprise Edition (via Direct Access gateway)

1

Select data to load (MySQL_Enterprise_Edition_(via_Direct_Access_gateway)_172.22.16.4)

Database: sampledb

Filter data: 読み込むデータベースを選択します。

Tables: 受注 (20)

2

読み込むテーブルを選択します。

Fields: Preview Metadata

受注コード	売上日	年月度	商品コード	得意先コード	出荷先
10001	8/9/2022	202208	P15	T26	愛知県
10002	8/9/2022	202208	P17	T26	愛知県
10003	8/9/2022	202208	P18	T26	愛知県
10004	8/9/2022	202208	P84	T26	愛知県
10007	8/7/2022	202208	P20	T22	福岡県
10008	8/7/2022	202208	P30	T22	福岡県
10009	8/7/2022	202208	P40	T22	福岡県
10010	8/7/2022	202208	P91	T22	福岡県

LOAD [受注コード],
[売上日],
[年月度],
[商品コード],
[得意先コード],
[出荷先都道府県],
[出荷先住所],
[出荷先名],
[社員コード],
[納品書番号],
[数量],
[単価],
[割引],
[販売金額],
[原価],
[粗利],
[利益率];

[受注]:
SELECT 受注コード,
売上日,
年月度,
商品コード,
得意先コード,
出荷先都道府県,
出荷先住所,
出荷先名,
社員コード,
納品書番号,
数量,
単価,
割引,
販売金額,
原価,
粗利,
利益率
FROM sampledb.`受注`;

Include LOAD statement

キャンセル スクリプトを挿入

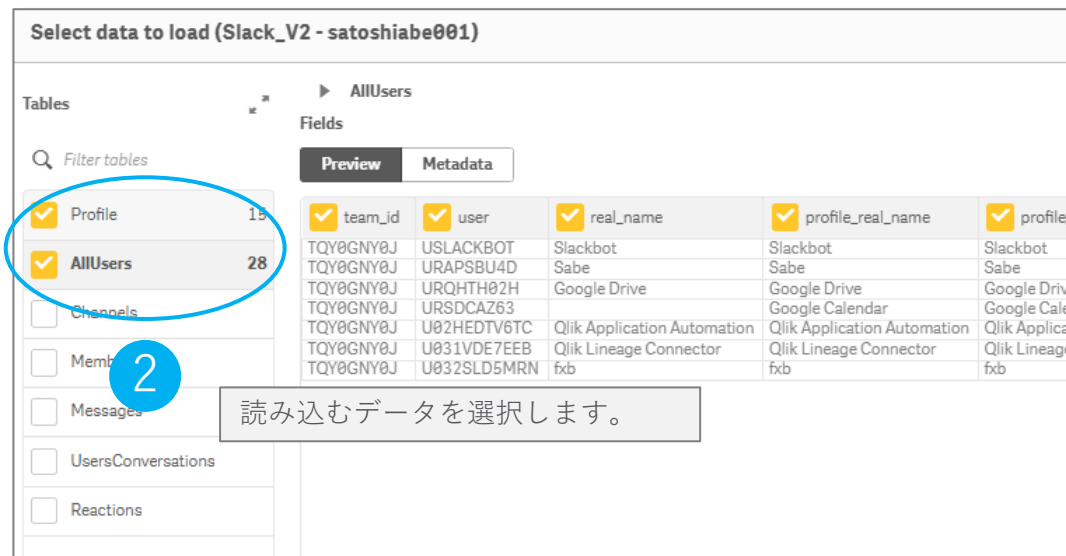
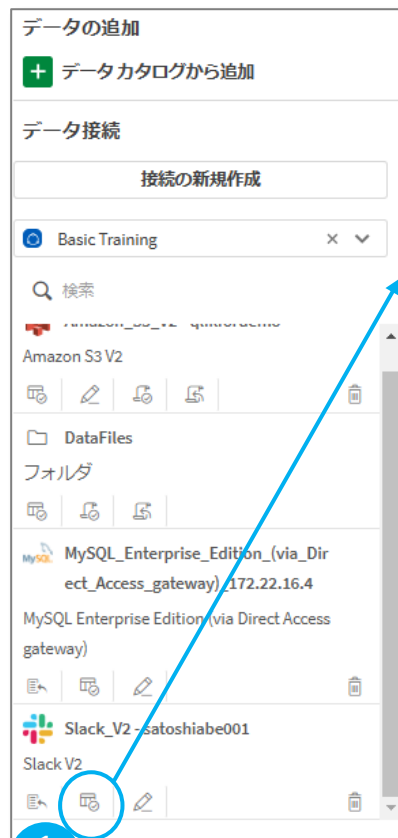
3

「スクリプトを挿入」で、テキストエディタにロードスクリプトが挿入されます。

アプリケーションからのデータの取り込み

Webアプリケーションから直接データを取り込むことができます。

ここでは、Slackからのデータの取り込み方法を解説します。



読み込むデータを選択します。

キャンセル

スクリプトを挿入

「スクリプトを挿入」で、
テキストエディタにロード
スクリプトが挿入されます。

```
1 // // Slack へ接続する
2
3 LIB CONNECT TO 'Basic Training:Slack_V2 - satoshiabe001';
4
5 LOAD user_id as [Profile.user_id],
6     title as [Profile.title],
7     phone as [Profile.phone],
8     real_name as [Profile.real_name],
9     real_name_normalized as [Profile.real_name_normalized],
10    display_name as [Profile.display_name],
11    display_name_normalized as [Profile.display_name_normalized],
12    status_text as [Profile.status_text],
13    status_emoji as [Profile.status_emoji],
14    status_expiration as [Profile.status_expiration],
15    image_48 as [Profile.image_48],
16    is_custom_image as [Profile.is_custom_image],
17    email as [Profile.email],
18    first_name as [Profile.first_name],
19    last_name as [Profile.last_name];
20
21 SELECT user_id,
22     title,
23     phone,
24     real_name,
25     real_name_normalized,
26     display_name,
27     display_name_normalized,
28     status_text,
29     status_emoji,
30     status_expiration,
31     image_48,
32     is_custom_image,
33     email,
34     first_name,
35     last_name
36 FROM Profile;
37
38 LOAD team_id as [AllUsers.team_id],
39     user as [AllUsers.user],
40     real_name as [AllUsers.real_name],
41     profile_real_name as [AllUsers.profile_real_name],
42     profile_real_name_normalized as [AllUsers.profile_real_name_normalized];
```

ロードスクリプトの基本



ロードスクリプトによる高度なデータ変換

Qlik Senseでは複雑な変換処理などを実装できる強力なロードスクリプトが提供されています。

ロードスクリプトでできること

- さまざまなファイル形式のロード
- 特定の項目の選択、項目名の変更および項目の計算
- 特定のレコードの選択
- ファイル上にないデータ、自動生成されたデータのロード
- 先行してロードされているテーブルからのデータのロード
- 後続のステートメントからのデータのロード
- データのグループ化
- テーブルの結合や関連付け
- 制御ステートメントや各種関数の利用

etc.

```
1  /*
2  Qlik Senseにはインライン データ項目一式をロードするテスト スクリプトが事前に設定されています。
3  このテスト スクリプトを使用すると、テスト目的のデータ セットを迅速に作成できます。
4
5  次の手順を実行します。
6
7      Ctrl + 00 を押します。
8
9  スクリプトにテスト スクリプト コードが挿入されます。
10 */
11
12 Characters:
13 Load Chr(RecNo()+Ord('A')-1) as Alpha, RecNo() as Num autogenerate 26;
14
15 ASCII:
16 Load
17   if(RecNo()>=85 and RecNo()<=90,RecNo()-84) as Num,
18   Chr(RecNo()) as AsciiAlpha,
19   RecNo() as AsciiNum
20 autogenerate 255
21   Where (RecNo())>=32 and RecNo()<=126) or RecNo()>=160 ;
22
23 Transactions:
24 Load
25   TransLineID,
26   TransID,
27   mod(TransID,26)+1 as Num,
28   Pick(Ceil(3*Rand()),'A','B','C') as Dim1,
29   Pick(Ceil(6*Rand()),'a','b','c','d','e','f') as Dim2,
30   Pick(Ceil(3*Rand()),'X','Y','Z') as Dim3,
31   Round(1000*Rand()*Rand()*Rand1) as Expression1,
32   Round( 10*Rand()*Rand()*Rand1) as Expression2,
33   Round(Rand()*Rand1,0.00001) as Expression3;
34 Load
35   Rand() as Rand1,
36   IterNo() as TransLineID,
37   RecNo() as TransID
38 Autogenerate 1000
39   While Rand()<=0.5 or IterNo()=1;
40
41   Comment Field Dim1 With "This is a field comment";
```

ロードスクリプトのサンプル

Load

LOAD ステートメントは、ファイル、スクリプトで定義されたデータ、事前にロードされたテーブル、Web ページ、後続の SELECT ステートメントの結果、または自動生成されたデータから項目をロードします。

- 解説
 - Load文に オプションを追記することで様々な形式のロードを実行することができます。
 - 主なオプション
 - ・ inline スクリプト内でデータを入力
 - ・ autogenerate データを自動生成
 - ・ where レコード選択の条件設定
 - ・ group by データを集計
 - ・ order by レコードのソート etc.
- 使い方
 - 基本的な使い方は、Load <field> From <file>;
 - 例では、データ接続先 DataFiles配下の「実績データ.csv」ファイルのすべての項目(*)をロードしています。

ロードスクリプト記述

```
1 LOAD * From 'lib://DataFiles/実績データ.csv';  
2
```

実行結果

実績データ

年月	部門	製品名	金額
2020年3月	部門A	製品A	50
2020年3月	部門A	製品B	70
2020年4月	部門B	製品A	90
2020年4月	部門B	製品B	60

その他の例

```
LOAD Quantity, Price, Quantity*Price as Value from data1.csv;  
//Quantity、Price、およびValue (QuantityとPriceの積) をロードします。  
  
LOAD ArtNo, round(Sum(TransAmount),0.05) as ArtNoTotal from table.csv group by ArtNo;  
//ArtNoによってグループ化 (集計) された項目をロードします。  
  
LOAD A, B, if(C>0,'positive','negative') as X, weekday(D) as Y; SELECT A,B,C,D from Table1;  
//後続の SELECT ステートメントにロードされている Table1 から、A、B、および計算された項目 X と Y をロードします。
```

スクリプトの挿入方法

ファイルを選択

パス

<DataFiles>

openops_ml_予測_SHAP...

openops_ml_予測_エラ...

openops_ml_予測.qvd

Order.json

SampleData.xlsx

受注-Org.csv

商品マ...

ファイルの種類

全てのテーブルファイル

SampleData.xlsx からデータを選択する

テーブル

→ ファイル形式

Excel (XLSX)

項目名

埋め込まれた項目名

読み込むデータを選択します。

☒ 売上データ

☐ 営業活動データ

☐ 顧客マスタ

☐ 国マスタ

☐ 商品マスタ

☐ 受注

☐ 増分ロード

☐ 都道府県データ

☐ 売上データ(2023...

☐ 売上データ(2023...

☐ 売上データ(2023...

☐ 売上データ(Cross t...

☐ 売上実績-2023年1月

☐ 売上実績-2023年2月

☐ 売上実績-2023年3月

☐ 売上実績-2023年4月

項目

☒ 受注日

☒ 顧客コード

☒ 購入金額

2024/02/01	1001	100
2024/02/01	1002	200
2024/02/02	1003	300
2024/02/02	1004	100
2024/02/03	1005	200
2024/02/03	1006	300
2024/02/04	1007	100
2024/02/04	1008	200
2024/02/05	1009	
2024/02/05	1010	
2024/02/06	1001	
2024/02/06	1002	

LOAD
受注日,
顧客コード,
購入金額
FROM [lib://DataFiles/SampleData.xlsx]
(ooxml, embedded labels, table is 売上データ);

スクリプトが生成されます。

スクリプトを隠す

項目を絞り込む

1 LOAD

2 受注日,

3 顧客コード,

4 購入金額

5 FROM [lib://DataFiles/SampleData.xlsx]

6 (ooxml, embedded labels, table is 売上データ);

7

8

スクリプトがエディタに挿入されます。

キャンセル

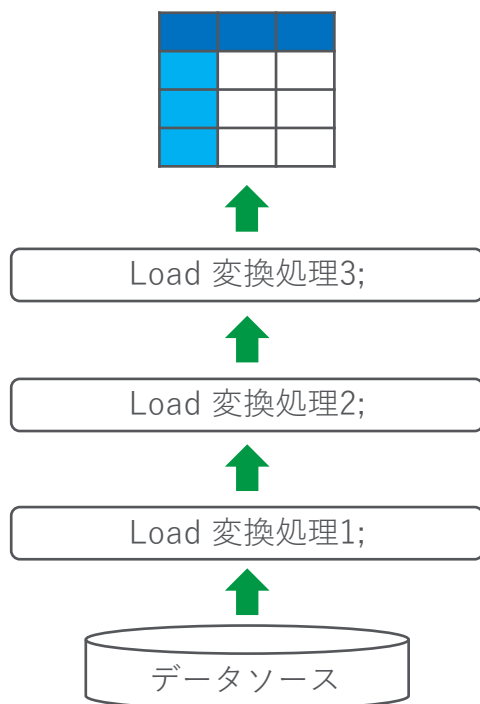
スクリプトを挿入

スクリプトの評価順

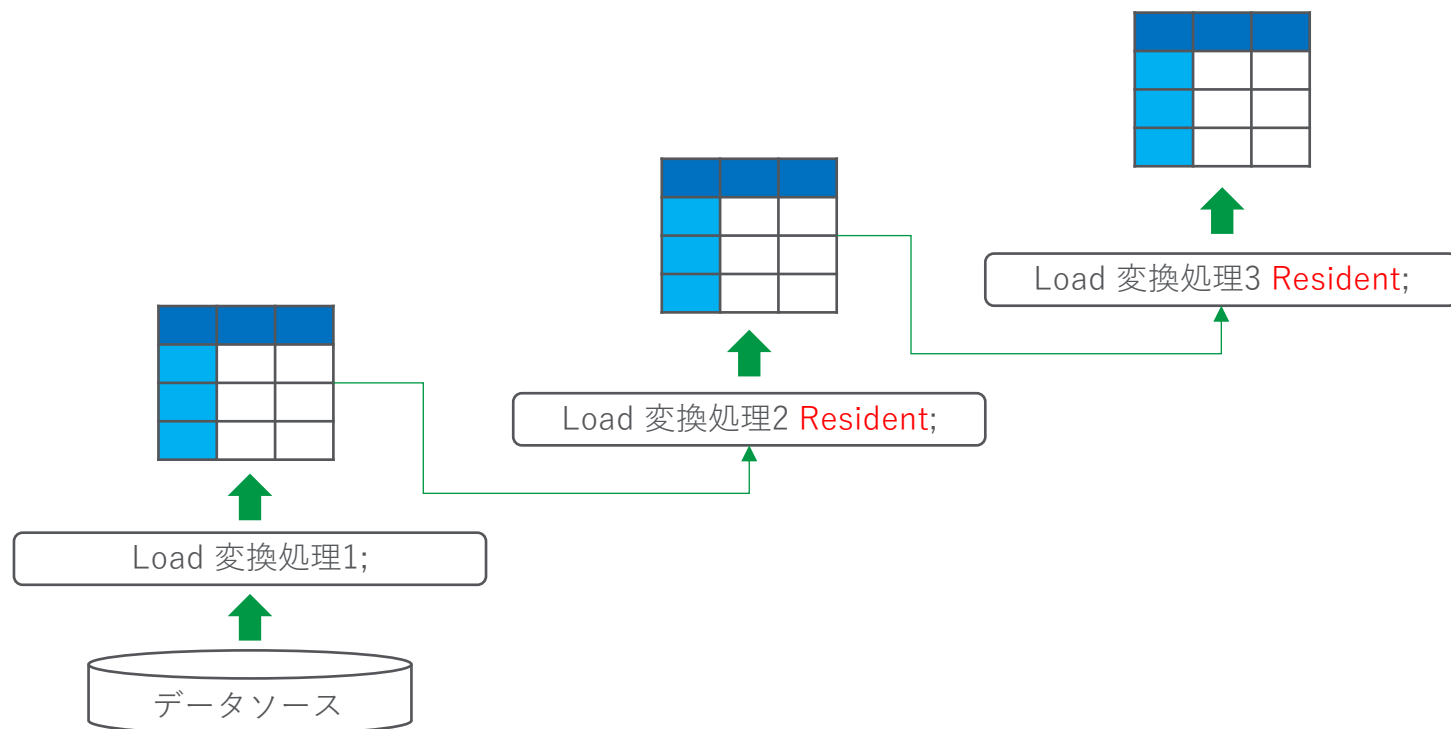
ある変換処理の結果として出力されたデータを入力データとして、別のデータ変換をつづける一連の変換処理を行うことができます。

先行する Load」と「Resident Load」の2つがあります。

① 先行する Load



② Resident Load



先行ロード

ある変換処理の結果として出力されたデータを次の変換処理に渡して、順に実施することができます。

- 解説
 - Load文の出力を次のLoad文に渡して、複数の変換処理を順に実施します。
 - 最下部のLoad文の出力が上のLoad文に順に渡されます。
 - 一般的に、先行ロードの方が後述のResident Loadよりも高速に処理を実行します。
- 使い方
 - Load文を多段階に記載します。
 - 例では、最下部のLoad文で「Date(Date#(発売日, 'YYYYMMDD'))」により整形した「発売日」を保存します。
 - 次のLoad文で「発売日」と現在日付を比較し、「経過日」や「保守状況」を保存します。
 - これにより、「Date(Date#(発売日, 'YYYYMMDD'))」を複数回記述することなく、「発売日」という項目として2つ目(上段)のLoad文でシンプルかつ効率よく記述ができています。

ロードスクリプト記述

```
1 LOAD
2   *,
3   Today() - 発売日 as 経過日,
4   If(AddYears(発売日, 3) > Today(), 'Yes', 'No') as 保守状況;
5 LOAD
6   製品名,
7   Date(Date#(発売日, 'YYYYMMDD')) as 発売日
8 FROM [lib://DataFiles/SampleData.xlsx]
9 (ooxml, embedded labels, table is 保守データ);
10
```

実行結果

プレビュー 保守データ ▼ 出力			
製品名	発売日	経過日	保守状況
製品A	2023/03/17	673	Yes
製品B	2021/03/17	1403	No
製品C	2018/03/17	2499	No

Lib Connect To

データソース（DB, ファイル等）を参照する際、データ接続を使用します。

- DataFiles接続からファイルをロードする
(例) 個人スペースからorders.csvをロードする場合
Load * FROM 'lib://DataFiles/orders.csv';
- 特定のスペースにあるDataFiles接続からファイルをロードする
(例) Basic Trainingスペースからorders.csvをロードする場合
Load * FROM 'lib://Basic Training:DataFiles/orders.csv';
- 特定のスペースにあるデータベースからテーブルをロードする
(例) Basic TrainingスペースにあるMySQL接続からSampleDBデータベースのSalesDataをロードする場合
LIB CONNECT TO 'Basic Training:MySQL';
Load * ; SQL SELECT * FROM 'SampleDB.SalesData';

Rem

スクリプト内にコメントを挿入します。

- 解説
 - スクリプト内に備考やコメントを挿入するため、またスクリプトを削除することなく一時的に無効にするために使用します。
- 使い方
 - `Rem string;`
 - `Rem`と次のセミコロン「`;`」の間に含まれるすべてがコメントと見なされます。

ロードスクリプト記述

```
1  Rem ** This is a comment **;  
2  
3  /* This is also a comment */  
4  
5  // This is a comment as well|
```

SET / LET

スクリプト変数を定義し、文字列やパスや数値などの代入に使用することができます。

ロードスクリプト記述

- 解説
 - SETステートメントはスクリプト変数を定義する際に使用します。
 - LETステートメントは、SETと同様にスクリプト変数を定義しますが、変数に代入する前に「=」の右側の数式が評価されます。
- 使い方
 - SET variablename = string;
 - LET variablename = expression;

```
1 SET FileToUse = Data1.csv;
2 LET T = now();
3
4 SET x=3+4; // $(x) は、'3+4'と評価されます。
5 LET y=3+4; // $(y) は、'7'と評価されます。
6
```

ロードスクリプトによる 高度なデータ変換



Apply Map

スクリプトの実行中に項目値または項目名を置き換える際にマッピングテーブルを使用できます。

- 解説
 - 「キー」と「値」の2列を含んだMappingテーブルを事前に作成し、「キー」を持つ他のテーブルデータを取り込む際に、このMappingテーブルを参照して「値」を取得できます。
 - 一般的に、この処理はルックアップや名寄せで同一種類と識別された情報へのマッチングなどを行うために利用します。
- 使い方
 - Load文の先頭にMappingを付加することでMappingテーブルを作成できます。
 - 例では、「国ID」と「国名」をキーと値としたMappingテーブルを作成し、「顧客マスタ」テーブルのロード時に、「国名」を取得しています。
 - マッチするキーがなかった場合、「Null」を返すように指定しています。

ロードスクリプト記述

```
1 Map_Country:
2 Mapping LOAD
3   国ID,
4   国名
5 FROM [lib://DataFiles/SampleData.xlsx]
6 (ooxml, embedded labels, table is 国マスタ);
7
8
9
10 顧客マスタ:
11 LOAD
12   顧客名,
13   ApplyMap('Map_Country', 国ID, Null()) as 国名
14 FROM [lib://DataFiles/SampleData.xlsx]
15 (ooxml, embedded labels, table is 顧客マスタ);
```

実行結果

プレビュー	顧客マスタ	出力
顧客名	国名	
顧客A	日本	
顧客B	アメリカ合衆国	
顧客C	中国	
顧客D	オーストラリア	

AutoGenerate

AutoGenerateを利用すると、生成するデータ件数を指定することができます。

ロードスクリプト記述

- 解説

- Autogenerateで指定した行数分のデータが生成されます。
- どのようなデータを入力するのか、項目定義を行うことも可能です。

```
1 GeneratedTbl:
2 LOAD
3     RowNo() as No,
4     Chr(64 + Ceil(Rand() * 5)) AS Region, // ランダムな名称
5     Round(Rand() * 10000) AS Sales, // 0~10000の売上
6     Date(Today() - Ceil(Rand() * 365)) AS Date // 過去1年の日付
7 AUTOGENERATE 20; // 20レコードの生成
```

- 使い方

- Load文でAutoGenerate(size)を指定します。
sizeは、生成するレコード数を示す整数
- 例では、RowNo関数によって作成される連番と、日付と時間をランダムな値で生成しています。

実行結果

プレビュー				GeneratedTbl	出力
No	Region	Sales	Date		
1	E	7519	2024/03/23		
2	A	6990	2024/11/23		
3	D	9526	2024/06/14		
4	A	4730	2024/04/26		
5	C	6237	2024/08/25		
6	B	4967	2025/01/11		
7	D	4806	2025/01/09		

AutoNumber

スクリプト実行中に発生するexpressionの評価値について、一意の整数値を設定することができます。

- 解説

- ある項目の値を元にして、一意の連番を作成します。
- 複合キーや桁数の多いコード等を一意のコードに置き換えてデータサイズを小さくすることができます。
- AutoNumberキーは、テーブルが読み込まれた順番で生成されるため、同じデータロードで生成された場合のみ結合できます。

- 使い方

- AutoNumber()関数に、値を置き換える必要がある項目のリストを渡します。
- 例では、「Region」と「Year」と「Month」の複合キーに一意の整数値を設定しています。

```
1 RegionSales:
2 Load *,
3     Autonumber(Region & Year & Month) as AutoNumKey;
4
5 Load * Inline [
6     Region, Year, Month, Sales
7     US, 2020, Jan, 100
8     US, 2020, Feb, 152
9     UK, 2020, Jun, 236
10    JP, 2020, May, 645
11 ];
12
```

実行結果

RegionSales				
Region	Year	Month	Sales	AutoNumKey
US	2020	Jan	100	1
US	2020	Feb	152	2
UK	2020	Jun	236	3
JP	2020	May	645	4

Concat

文字列値を組み合わせるために使用します。

- 解説
 - 文字列を組み合わせるために使用します。
- 使い方
 - Concat(“処理される文字列”, delimiter, weight)
 - 各値は、delimiter（この例では、'-'）によって結合されます。
 - 連結の順序は、weightの値によって決定されます。この例では、weightが省略されているため、アルファベット順にソートされています。

ロードスクリプト記述

```
1 TempData:
2 Load * inline [
3     id, Team
4     10, Alpha
5     20, Beta
6     30, Gamma
7     40, Zeta
8     50, Delta
9     60, Epsilon
10    70, Eta
11 ];
12
13 TeamName:
14 Load Concat(Team, '-') as TeamName // Weightがないため、アルファベット順にソート
15 Resident TempData;
16
17 Drop Table TempData;
```

実行結果

TeamName
Alpha-Beta-Delta-Epsilon-Eta-Gamma-Zeta

Concatenate Load

Qlik Senseでは、2つのテーブルが同じ構造を持たない場合でも、Load文の先頭にConcatenateを追加することで、2つのテーブルを強制的に結合することができます。

- 解説

- 連結される2つのテーブルに異なる項目セットが存在する場合、Concatenateプレフィックスを使用すると2つのテーブルを強制的に連結できます。このステートメントは、指定の既存テーブルまたは最後に作成された論理テーブルとの連結を強制します。
- 片方のテーブルにしか存在しない項目について、もう片方のテーブルではその値はNULLとなります。

- 使い方

- 強制的に結合するテーブルのLoadの前に、Concatenateを指定します。
- 例では、「売上データ_2020年3月.xlsx」と「売上データ_2020年4月.xlsx」が同じ構造でありそのテーブルに、異なる構造の「売上データ_2020年5月.xlsx」のデータを強制的に結合しています。

ロードスクリプト記述

```
1 売上データ:
2  LOAD
3     受注日,
4     商品名,
5     販売価格
6  FROM [lib://DataFiles/SampleData.xlsx]
7  (ooxml, embedded labels, table is [売上データ(2023年3月)]);
8
9  LOAD
10     受注日,
11     商品名,
12     販売価格
13 FROM [lib://DataFiles/SampleData.xlsx]
14 (ooxml, embedded labels, table is [売上データ(2023年4月)]);
15
16 // 売上データ 2023年5月は、フォーマットが異なるが、売上データに格納する
17 Concatenate LOAD
18     受注日,
19     商品名,
20     販売価格,
21     顧客名
22 FROM [lib://DataFiles/SampleData.xlsx]
23 (ooxml, embedded labels, table is [売上データ(2023年5月)]);
24
```

実行結果

売上データ			
受注日	商品名	販売価格	顧客名
3/1/2020	商品A	100	-
3/2/2020	商品B	200	-
4/1/2020	商品A	300	-
4/2/2020	商品B	400	-
5/1/2020	商品A	500	顧客A
5/2/2020	商品B	600	顧客B

CrossTableプレフィックス

Crosstableを利用して、クロス テーブルをストレート テーブルに変換します。

- 解説

- クロステーブルは、ピボットテーブルや多次元テーブルのように列と行で構成された双方向のテーブルです。
- 多くの列のある横長のテーブルは、列の見出しが単一の属性列を持つ縦長のテーブル(ストレートテーブル)に変換されます。

- 使い方

- LOAD または SELECT文の先頭に crosstable プレフィックスを追加します。
- crosstable (<属性値項目>, <データ値項目>, <修飾子項目の数 (デフォルト1)>
- 例では、行見出し列の数(修飾子項目の数)を crosstable プレフィックスの 3 番目のパラメータとして指定しています。

ロードスクリプト記述

Salesman	Year	Jan	Feb	Mar	Apr	May	Jun	Month
A	2008	45	65	78	12	78	22	
A	2009	11	23	22	22	45	85	
A	2010	65	56	22	79	12	56	
A	2011	45	24	32	78	55	15	
A	2012	45	56	35	78	68	82	
B	2008	57	77	90	24	90	34	
B	2009	23	35	34	34	57	97	
B	2010	77	68	24	61	24	68	
B	2011	57	68	24	61	24	68	
B	2012	57	68	24	61	24	68	

```
1 Crosstable (Month, Sales, 2)  
2 LOAD *  
3 FROM [lib://DataFiles/SampleData.xlsx]  
4 (ooxml, embedded labels, table is [売上データ(Cross table)]);  
5
```

修飾子項目

実行結果

売上データ

Salesman	Year	Month	Sales
A	2008	Jan	45
A	2008	Feb	65
A	2008	Mar	78
A	2008	Apr	12
A	2008	May	78
A	2008	Jun	22
A	2009	Jan	11
A	2009	Feb	23

Exists

特定の項目値がデータ ロード スクリプトの項目にすでにロードされているかどうかを決定します。

ロードスクリプト記述

```
1 Employee:
2 LOAD * inline [
3 Employee|ID|Salary
4 Bill|001|20000
5 John|002|30000
6 Steve|003|35000
7 ] (delimiter is '|');
8
9 Citizens:
10 Load * inline [
11 Employee|Address
12 Bill|New York
13 Mary|London
14 Steve|Chicago
15 Lucy|Madrid
16 Lucy|Paris
17 John|Miami
18 ] (delimiter is '|') where Exists (Employee);
```

実行結果

Employee

Employee	ID	Salary
Bill	001	20000
John	002	30000
Steve	003	35000

Citizens

Employee	Address
Bill	New York
Steve	Chicago
John	Miami

- 解説
 - 項目値がすでにロード済か TRUE または FALSE で返します。
 - Load ステートメント、またはIFステートメントのWhere句で使用できます。
- 使い方
 - Exists(“フィールド名”, expr)
 - 値を検索する項目の名前。明示的な項目名を引用符なしで使用できます。
 - exprは、存在するかどうか確認したい値。省略した場合、現在のレコード内に“フィールド名”の値がすでに存在しているかどうかを確認されます。

FieldValue

フィールドの指定位置にある値を返します。

- 解説
 - フィールドの指定位置にある値を返します。
 - 指定される位置は、ロード順の位置となります。
- 使い方
 - FieldValue(field, no)
 - 最初のパラメータにフィールドを指定し、2番目のパラメータに項目の位置の番号を指定します。
 - fieldは、文字列値でなければならないため、単一引用符で囲む必要があります。

ロードスクリプト記述

```
1 Names:
2 LOAD * inline [
3 First name|Last name|Initials|Has cellphone
4 John|Anderson|JA|Yes
5 Sue|Brown|SB|Yes
6 Mark|Carr|MC|No
7 Peter|Devonshire|PD|No
8 Jane|Elliot|JE|Yes
9 Peter|Franc|PF|Yes ] (delimiter is '|');
10
11
12 LET vPosition1 = FieldValue('First name', 1);
13 LET vPosition7 = FieldValue('First name', 7);
14
```

実行結果

vPosition7	<NULL>
vPosition1	"John"

FirstValue

対象となるデータが含まれている数式、またはフィールドの最初にロードされた値を返します。

- 解説
 - 数式で定義され、group byでソートされたレコードから、最初にロードされた値を返します。
- 使い方
 - FirstValue(“数式、またはフィールド”)
 - この例では、group byによってソートされたTeamフィールドの最初の値を返します。

ロードスクリプト記述

```
1 TeamData:
2 LOAD * inline [
3 SalesGroup|Team|Date|Amount
4 East|Gamma|01/05/2013|20000
5 East|Gamma|02/05/2013|20000
6 West|Zeta|01/06/2013|19000
7 East|Alpha|01/07/2013|25000
8 East|Delta|01/08/2013|14000
9 West|Epsilon|01/09/2013|17000
10 West|Eta|01/10/2013|14000
11 East|Beta|01/11/2013|20000
12 West|Theta|01/12/2013|23000
13 ] (delimiter is '|');
14
15
16 FirstValue1:
17 LOAD SalesGroup,FirstValue(Team) as FirstTeamLoaded
18 Resident TeamData Group By SalesGroup;
```

実行結果

FirstValue1

SalesGroup	FirstTeamLoaded
East	Gamma
West	Zeta

FirstSortedValue

指定したフィールドから、ソート順に対応する値を取得します。

- 解説
 - valueで指定したフィールドからソート結果に対応した値を返します。
- 使い方
 - FirstSortedValue(“フィールド”, weight, rank)
 - weightはソート対象となるデータ。weightが最小値となるフィールド値が返されます。
 - weightにマイナス記号を付けると、weightが最大となるフィールド値が返されます。
 - rankにnを設定すると n番目のソート値を取得できます。rankは省略可能です。

ロードスクリプト記述

```
1 Temp:
2 LOAD * inline [
3 Customer|Product|OrderNumber|UnitSales|CustomerID
4 Astrida|AA|1|10|1
5 Astrida|AA|7|18|1
6 Astrida|BB|4|9|1
7 Astrida|CC|6|2|1
8 Betacab|AA|5|4|2
9 Betacab|BB|2|5|2
10 Betacab|DD|12|25|2
11 Canutility|AA|3|8|3
12 Canutility|CC|13|19|3
13 Divadip|AA|9|16|4
14 Divadip|AA|10|16|4
15 Divadip|DD|11|10|4
16 ] (delimiter is '|');
17
18
19 FirstSortedValue:
20 LOAD Customer, FirstSortedValue(Product, UnitSales) as MyProductWithSmallestOrderByCustomer
21 Resident Temp Group By Customer;
```

実行結果

FirstSortedValue	
Customer	MyProductWithSmallestOrderByCustomer
Astrida	CC
Betacab	AA
Canutility	AA
Divadip	DD

Force

項目値を大文字のみ、小文字のみ、常に先頭を大文字化、またはそのまま（混合）として強制的に解釈します。

- 解説
 - 後続のLOADおよびSELECTステートメントの項目値を大文字、小文字、先頭を大文字、混在に設定できます。
- 使い方
 - Force Capitalization で先頭を大文字にします。
 - Force Case Upper ですべて大文字にします。
 - Force Case Lower ですべて小文字にします。
 - Force Case Mixed で大文字・小文字混在にします。

```
1 FORCE Capitalization;  
2 Capitalization:  
3 LOAD * Inline [  
4 Item1  
5 ab  
6 Cd  
7 eF  
8 GH  
9 ];  
10  
11 FORCE Case Upper;  
12 CaseUpper:  
13 LOAD * Inline [  
14 Item2  
15 ab  
16 Cd  
17 eF  
18 GH  
19 ];  
20  
21 FORCE Case Lower;  
22 CaseLower:  
23 LOAD * Inline [  
24 Item3  
25 ab  
26 Cd  
27 eF  
28 GH  
29 ];  
30  
31 FORCE Case Mixed;  
32 CaseMixed:  
33 LOAD * Inline [  
34 Item4  
35 ab  
36 Cd  
37 eF  
38 GH  
39 ];  
40
```

Capitalization

Item1
Ab
Cd
Ef
Gh

CaseUpper

ITEM2
AB
CD
EF
GH

CaseLower

item3
ab
cd
ef
gh

CaseMixed

Item4
ab
Cd
eF
GH

Group By

データを集計して取り込みたい場合に、集計関数とGroup Byを組み合わせて集計処理をします。

ロードスクリプト記述

- 解説
 - Loadスクリプトに「Goup By グループ化する項目」を追加することで、その項目でデータをグループ化することができます。
 - グループ化する項目以外の集計対象となる項目に対して、Sum, Count, Min, Max, Avgなどの集計関数を利用して集計を行います。
- 使い方
 - Load文の最終行に Group Byを追加します。
 - 例では、「従業員ID」と「年月」ごとにグループ化し、「売上実績」と「顧客訪問数」をSum関数で集計しています。

```
1 // 営業活動データ:
2 LOAD
3     従業員ID,
4     年月,
5     Sum(売上実績) as 売上実績,
6     Sum(顧客訪問数) as 顧客訪問数
7 FROM [lib://DataFiles/SampleData.xlsx]
8 (ooxml, embedded labels, table is 営業活動データ)
9 Group By 従業員ID, 年月;
10
```

実行結果

プレビュー 営業活動データ 出力				
従業員ID	年月	売上実績	顧客訪問数	
4	2024年4月	12391	87	
5	2024年2月	7428	51	
5	2024年3月	6027	66	
3	2024年3月	7198	54	
1	2024年1月	9597	78	
2	2024年3月	4982	48	
3	2024年4月	6585	65	

Intervalmatch

不連続数値を 1 つ以上の数値間隔に一致させるテーブルを作成します。

ロードスクリプト記述

- 解説
 - ロードされた間隔テーブルとキーを変換して追加列 (不連続数値データ点) を含むテーブルを生成します。
 - そしてオプションとして 1 つ以上の追加キーの値を一致させるテーブルを作成できます。
- 使い方
 - 間隔をロードするLoadまたはSELECTの前に配置します。
 - `IntervalMatch(matchfield, keyfield)`
Load/Select-statement;
 - `matchfield`は、不連続の数値が含まれた項目
 - `keyfield(Optional)`は、変換で一致させる追加属性が含まれた項目
 - Load/Select-statementには、最初の項目には各間隔の下限、2 つ目の項目には各間隔の上限を指定します。

```
1  EventLog:
2  LOAD * Inline [
3  Time, Event, Comment
4  00:00, 0, Start of shift 1
5  01:18, 1, Line stop
6  02:23, 2, Line restart 50%
7  04:15, 3, Line speed 100%
8  08:00, 4, Start of shift 2
9  11:43, 5, End of production
10 ];
11
12 OrderLog:
13 LOAD * INLINE [
14 Start, End, Order
15 01:00, 03:35, A
16 02:30, 07:58, B
17 03:04, 10:27, C
18 07:23, 11:43, D
19 ];
20
21 //Link the field Time to the time intervals defined by the fields Start and End.
22 Inner Join IntervalMatch ( Time )
23 LOAD Start, End
24 Resident OrderLog;
```

実行結果

OrderLog			
Time	Start	End	Order
01:18	01:00	03:35	A
02:23	01:00	03:35	A
04:15	02:30	07:58	B
04:15	03:04	10:27	C
08:00	03:04	10:27	C
08:00	07:23	11:43	D
11:43	07:23	11:43	D

Join

Joinを使うと、2つのテーブルを結合できます。

- 解説
 - Join プレフィックスは、ロード済みのテーブルを名前が付いた既存テーブルとjoinの後に処理するテーブルを結合します。既存テーブルの指定が無い場合は、直前に作成されたデータテーブルと結合します。
 - Outer Join, Inner Join, Left Join, Right Joinを指定することができます。指定しない場合は**Outer Join**とみなされます。
- 使い方
 - Loadステートメントの先頭にJoinを追加し、「Join (結合先テーブル名)」で結合先テーブル名を指定します。
 - 例では、結合先のテーブル名を省略し、1つ前のテーブルと結合しています。

ロードスクリプト記述

```
1 LOAD
2   顧客名,
3   国ID
4 FROM [lib://DataFiles/SampleData.xlsx]
5 (ooxml, embedded labels, table is 顧客マスタ);
6
7 Join LOAD
8   国ID,
9   国名
10 FROM [lib://DataFiles/SampleData.xlsx]
11 (ooxml, embedded labels, table is 国マスタ);
12
```

実行結果

顧客マスタ

顧客名	国ID	国名
顧客A	1	日本
顧客B	3	中国
顧客C	6	-
顧客D	2	アメリカ合衆国
-	4	イギリス

Joinの補足

Inner, Left, Right, Outer Join

/ ---- Inner Join ---- */*

```
Table1:
LOAD * inline [
  A,B
  1,aa
  2,bb
  3,cc
];

Table2:
Inner Join(Table1) LOAD * Inline [
  A,C
  1,xx
  2,yy
  4,zz
];
```

Table1

A	B	C
1	aa	xx
2	bb	yy

/ ---- Left Join ---- */*

```
Table3:
LOAD * inline [
  A,B
  1,aa
  2,bb
  3,cc
];

Table4:
Left Join(Table3) LOAD * Inline [
  A,C
  1,xx
  2,yy
  4,zz
];
```

Table3

A	B	C
1	aa	xx
2	bb	yy
3	cc	-

/ ---- Right Join ---- */*

```
Table5:
LOAD * inline [
  A,B
  1,aa
  2,bb
  3,cc
];

Table6:
Right Join(Table5) LOAD * Inline [
  A,C
  1,xx
  2,yy
  4,zz
];
```

Table5

A	B	C
1	aa	xx
2	bb	yy
4	-	zz

/ ---- Outer Join ---- */*

```
Table7:
LOAD * inline [
  A,B
  1,aa
  2,bb
  3,cc
];

Table8:
Outer Join(Table7) LOAD * Inline [
  A,C
  1,xx
  2,yy
  4,zz
];
```

Table7

A	B	C
1	aa	xx
2	bb	yy
3	cc	-
4	-	zz

LastValue

対象となるデータが含まれている数式、またはフィールドの最後にロードされた値を返します。

- 解説

- 数式で定義され、group byでソートされたレコードから、最後にロードされた値を返します。

- 使い方

- LastValue(“数式、またはフィールド”)
- この例では、group byによってソートされたTeamフィールドの最後の値を返します。

ロードスクリプト記述

```
1 TeamData:
2 LOAD * inline [
3 SalesGroup|Team|Date|Amount
4 East|Gamma|01/05/2013|20000
5 East|Gamma|02/05/2013|20000
6 West|Zeta|01/06/2013|19000
7 East|Alpha|01/07/2013|25000
8 East|Delta|01/08/2013|14000
9 West|Epsilon|01/09/2013|17000
10 West|Eta|01/10/2013|14000
11 East|Beta|01/11/2013|20000
12 West|Theta|01/12/2013|23000
13 ] (delimiter is '|');
14
15
16 LastValue1:
17 LOAD SalesGroup, LastValue(Team) as LastTeamLoaded
18 Resident TeamData Group By SalesGroup;
```

実行結果

SalesGroup	LastTeamLoaded
East	Beta
West	Theta

Map …using

特定のマッピング テーブルの値に、特定の項目値または数式をマップするために使用されます。

- 解説

- スクリプトのこの場所からマッピングされる項目のコンマ区切りリスト。 項目リストに * を使用すると、すべての項目が対象となります。

- 使い方

- Map *filedlist* Using *mapname*
- *filedlist*は、 マッピングされる項目、*mapname*は、読み取られたマッピングテーブル名。
- この例では、*国ID*を*Map_Country*でマッピングされた*国名*に置き換えています。

ロードスクリプト記述

```
1 Map_Country:
2 Mapping LOAD
3   国ID,
4   国名
5 FROM [lib://DataFiles/SampleData.xlsx]
6 (ooxml, embedded labels, table is 国マスタ);
7
8 Map [国ID] using Map_Country;
9 Load [顧客名], [国ID]
10 FROM [lib://DataFiles/SampleData.xlsx]
11 (ooxml, embedded labels, table is 顧客マスタ);
12
```

実行結果

	A	B
1	顧客名	国ID
2	顧客A	1
3	顧客B	3
4	顧客C	4
5	顧客D	2

	A	B
1	国ID	国名
2		1 日本
3		2 オーストラリア
4		3 アメリカ合衆国
5		4 中国
6		5 イギリス

Map_Country

顧客名	国ID
顧客A	日本
顧客B	アメリカ合衆国
顧客C	中国
顧客D	オーストラリア

No Concatenate Load

NoConcatenateを利用すると、完全に合致する複数のテーブルの自動連結を回避することができます。

※ Qlik Senseでは、ロードされた複数のファイルの項目名と項目数が完全に合致する場合、これらのテーブルを連結します。（自動連結）

- 解説

- 強制連結とは逆に、NoConcatenateをLoad文の先頭に追加すると、項目名と項目数が完全に合致する複数のテーブルの自動連結を回避することができます。

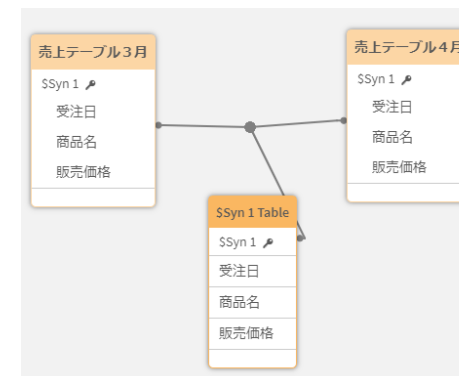
- 使い方

- 自動的に連結させたくないテーブルのLoad文にNoConcatenateを指定します。
- 例の場合、「売上データ_2020年3月.xlsx」のテーブルと「売上データ_2020年4月.xlsx」のテーブルは強制的に別個の内部テーブルとして扱います。NoConcatenateが指定されていない場合、2つのテーブルは自動的に連結されます。

ロードスクリプト記述

```
1 売上テーブル3月:  
2  LOAD  
3     受注日,  
4     商品名,  
5     販売価格  
6  FROM [lib://DataFiles/SampleData.xlsx]  
7  (ooxml, embedded labels, table is [売上データ(2023年3月)]);  
8  
9  売上テーブル4月:  
10 NoConcatenate LOAD  
11     受注日,  
12     商品名,  
13     販売価格  
14 FROM [lib://DataFiles/SampleData.xlsx]  
15 (ooxml, embedded labels, table is [売上データ(2023年4月)]);  
16
```

実行結果



「売上テーブル3月」と「売上テーブル4月」は同じ構造ですが、強制的に別の内部テーブルとしてロードされます。

Only

集計データに絞込結果が 1 つしか存在しない場合に値を返します。

- 解説
 - レコードに値が 1 つしか含まれていない場合はその値を、値が 2 つ以上含まれている場合は NULL を返します。
 - group by 句を使用して複数のレコードを評価します。Only() 関数は数値およびテキスト値を返すこともあります。
- 使い方
 - Only(expr)
 - expr は、メジャーの対象となるデータが含まれている数式または項目。

ロードスクリプト記述

```
1 Temp:
2 LOAD * inline [
3     Customer,Product,OrderNumber,UnitSales,CustomerID
4     Astrida,AA,1,10,1
5     Astrida,AA,7,18,1
6     Astrida,BB,4,9,1
7     Astrida,CC,6,2,1
8     Betacab,AA,5,4,2
9     Betacab,BB,2,5,2
10    Betacab,DD,,
11    Canutility,DD,3,8,3
12    Canutility,CC,,8
13 ];
14
15 Only:
16 LOAD Customer, Only(CustomerID) as MyUniqIDCheck Resident Temp Group By Customer;
```

実行結果

Only	
Customer	MyUniqIDCheck
Astrida	1
Betacab	-
Canutility	-

Order By

取り込まれたデータをOrder Byを利用してソートすることができます。

ロードスクリプト記述

```
1 営業活動データ:  
2 Load * FROM [lib://DataFiles/SampleData.xlsx]  
3 (ooxml, embedded labels, table is 営業活動データ);  
4  
5 営業活動データ_ソート済:  
6 NoConcatenate LOAD * Resident 営業活動データ  
7 Order by 従業員ID, 年月;  
8  
9 Drop table 営業活動データ;  
10
```

実行結果

プレビュー 営業活動データ_ソート済 出力				
従業員ID	年月	売上実績	顧客訪問数	
1	2024年1月	1272	9	
1	2024年1月	125	2	
1	2024年1月	673	3	
1	2024年1月	990	8	
1	2024年1月	344	4	
1	2024年1月	1305	8	
1	2024年1月	1489	7	

- 解説

- Order Byは、外部からデータを取り込む際のLoad文では行えず、すでにQlik Senseに取り込まれたテーブルに対してのみ実行できます。
- ソートを行う場合、昇順には asc, 降順には descを指定します。(ascは省略可)

- 使い方

- Load文の最終行に Order Byを追加します。
- 例では、「営業活動データ」テーブルから Resident Loadで全項目を読み込み、「従業員ID」と「年月」で昇順にソートし、「営業活動データ_ソート済」テーブルに格納しています。
- なお、ソートしたデータを「営業活動データ」に自動連結しないように NoConcatenateを指定しています。

Partial Load

データ モデル内のすべてのテーブルを保持し、[Add]、[Merge]、または [Replace] プレフィックスが前に付いた [Load] ステートメントと [Select] ステートメントのみを実行します。

- 解説

- フルリロードは常に既存のデータモデルのすべてを削除し、ロードスクリプトを実行しますが、Partial Loadは、これを行いません。
- [Add],[Merge],[Replace]の付いたロードを実行します。
- 引数Onlyは、フルリロード中には無視されます。

- 使い方

- **Add/Replace/Merge** Load <Loadステートメント>
- **Add/Replace/Merge only** Load <Loadステートメント>
- IFステートメントを利用することもできます。
IF Not **IsPartialReload()** THEN (A) ENDIF.
(A)は、Partial Loadの場合、実行されません。

ロードスクリプト記述

```
16 //
17 // ***** Partial Load (Updateの実施) *****
18 //
19 Merge only on No Concatenate(Table1)
20 LOAD * inline [
21     Operation, No, Date, Customer, Sales
22     Update, 1000, 2024/12/01, CompanyA, 1000
23     Update, 2000, 2024/11/02, CompanyB, 1500
24 ];
```

実行結果

No	Q	Date	Q	Custo...	Q	Sales	Q
1000		2024/11/01		CompanyA		1000	
2000		2024/11/02		CompanyB		2000	
3000		2024/11/03		CompanyC		3000	
4000		2024/11/04		CompanyD		4000	
5000		2024/11/05		CompanyE		5000	
6000		2024/11/06		CompanyF		6000	
7000		2024/11/07		CompanyG		7000	



No	Q	Date	Q	Custo...	Q	Sales	Q
1000		2024/12/01		CompanyA		1000	
2000		2024/11/02		CompanyB		1500	
3000		2024/11/03		CompanyC		3000	
4000		2024/11/04		CompanyD		4000	
5000		2024/11/05		CompanyE		5000	
6000		2024/11/06		CompanyF		6000	
7000		2024/11/07		CompanyG		7000	

Partial Load 実装例

スクリプト例	内容
<pre>Tab1: LOAD Name, Number FROM Persons.csv; Add Only LOAD Name, Number FROM NewPersons.csv where not exists(Name);</pre>	<p>通常のリロードでは、データは Persons.csv からロードされ、Qlik Sense テーブル Tab1 に保存されます。NewPersons.csv をロードするステートメントは無視されます。</p> <p>パーシャル リロードを実行している場合、データは NewPersons.csv からロードされ、Qlik Sense テーブル Tab1 に追加されます。</p>
<pre>Tab1: LOAD a,b,c from File1.csv; Replace only LOAD a,b,c from File2.csv;</pre>	<p>通常のリロードでは、データは File1.csv からロードされ、Qlik Sense テーブル Tab1 に保存されます。その際、File2.csv は無視されます。</p> <p>パーシャル リロードでは、まず Qlik Sense テーブル Tab1 が削除されます。次に File2.csv からロードされた新しいデータに置き換えられます。その結果、File1.csv のデータはすべて失われます。</p>
<pre>Merge only on ユーザ名 Concatenate(Position) LOAD * inline [Operation, ユーザ名, 年収, 総資産, 借入 Update, 相談者, \$(vSalary), \$(vAssets), \$(vBorrow)];</pre>	<p>パーシャル リロードの際、ユーザ名が「相談者」の年収、総資産、借入のフィールドを、vSalary, vAssets, vBorrowで Updateします。結果は、Positionテーブルへ反映されます。</p> <p>Operationには、Insert, Update, Deleteが指定できます。</p>

Peek

Peekを利用して、テーブル内の指定行の値を取得することができます。

- 解説
 - すでにロードされている行または内部メモリに存在する行に対して、テーブルの項目の値を取得します。
 - 最初のレコードは 0, 2 番目のレコードは 1 となります。 -1 は、読み取られた最後のレコードを示します。
 - 行番号が指定されていない場合は、-1 として処理されます。
 - テーブル名が指定されていない場合は、現在のテーブルとして処理されます。
 - 項目名は、文字列として指定する必要があります。
- 使い方
 - Peek (フィールド名, 行番号, テーブル名)と記載します。
 - 例では、「箱根駅伝」テーブルから10行目の大学を取得しています。

ロードスクリプト記述

```
1 箱根駅伝:  
2 Load * inline [  
3     順位, 大学  
4     1, 駒沢大学  
5     2, 創価大学  
6     3, 東洋大学  
7     4, 青山学院大学  
8     5, 東海大学  
9     6, 早稲田大学  
10    7, 順天堂大学  
11    8, 帝京大学  
12    9, 國學院大學  
13    10, 東京国際大学  
14    11, 明治大学  
15    12, 中央大学  
16    13, 神奈川大学  
17    14, 日本体育大学  
18    15, 拓殖大学  
19    16, 城西大学  
20    17, 法政大学  
21    18, 国土館大学  
22    19, 山梨学院大学  
23    20, 専修大学  
24 ];  
25  
26 LET vRank10 = Peek('大学', 9, '箱根駅伝');
```

実行結果

 vRank10	東京国際大学
---	--------

Pick

リストの指定位置の数式を取得します。

- 解説
 - リストのn番目の数式を返します。
- 使い方
 - $\text{Pick}(n, \text{expr})$
 - n は 1～Nの間の整数です。

ロードスクリプト記述

```
vResult = Pick(5, 'January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October')
```

実行結果

 vResult

May

参考

Pick(Position, Value1, Value2,...) Positionの位置の値が返る
Match(Value, Position1, Position2, ...) Valueのある位置が返る

Previous

Previousを利用して、テーブル内の指定行の値を取得することができます。

ロードスクリプト記述

- 解説
 - 先行する入力レコードのデータを使用して、expr数式を算出します。
 - 入力されるデータソースから取得されたデータを利用するため、Qlik Senseにロードされていない項目を参照することも可能です。
 - 内部テーブルの最初のレコードの場合は、NULLを返します。
- 使い方
 - Previous (expr数式)と記載します。
 - 例では、Previous() 関数を Load文で使用することで、Sales の現在の値を先行する値と比較でき、3 番目の項目 Increase で使用できます。

```
1 Sales2020:  
2 Load *,  
3 (Sales - Previous(Sales)) as Increase  
4 Inline [  
5     Month, Sales  
6     Q1, 12  
7     Q2, 13  
8     Q3, 15  
9     Q4, 17  
10 ];
```

実行結果

Sales2020		
Month	Sales	Increase
Q1	12	-
Q2	13	1
Q3	15	2
Q4	17	2

Previous vs Peek

```
1 ExmpPeek:
2 load *,
3   if (OrderID = Previous(OrderID), peek(Total)+Sales, Sales) as Total
4 inline [
5   OrderID, Sales
6   1, 3
7   1, 4
8   1, 6
9   2, 1
10  2, 3
11  3, 8
12  3, 5
13  4, 1
14  5, 8
15 ];
16
17 ExmpPrevious:
18 load *,
19   if (OrderID = Previous(OrderID), Previous(Sales)+Sales, Sales) as Total
20 inline [
21   OrderID, Sales
22   1, 3
23   1, 4
24   1, 6
25   2, 1
26   2, 3
27   3, 8
28   3, 5
29   4, 1
30   5, 8
31 ];
```

Totalを利用可
Peekは、読み込んだレコードのデータを使用

Totalは利用不可
Previousは、先行する入力レコードのデータを使用

ExmpPeek

OrderID	Sales	Total
1	3	3
1	4	7
1	6	13
2	1	1
2	3	4
3	8	8
3	5	13
4	1	1

ExmpPrevious

OrderID	Sales	Total
1	3	3
1	4	7
1	6	10
2	1	1
2	3	4
3	8	8
3	5	13
4	1	1

Replace


指定された文字列内に含まれる指定されたサブストリングすべてを別のサブストリングで置き換えた文字列を返します。

- 解説
 - 入力文字列に 1 回以上現れる文字列を指定し、入力文字列に現れた文字を指定した文字列に置き換えます。
- 使い方
 - Replace(“入力文字”, “現れる文字列”, “置き換える文字列”)

ロードスクリプト記述

```
1 LET vRep = Replace('abcdefghijklmn', 'cdefg', 'xyz');  
2
```

実行結果

 vRep	abxyzhijklmn
--	--------------

Resident Load

ある変換処理の結果として出力されたデータを入力データとし、そちらに別のデータ変換を行うといった一連の変換処理を行うことができます。

- 解説

- Load文の出力をインメモリ上のテーブルに保存し、そのテーブルのデータをResident Loadで読み込んで、複数の変換処理を順に実施することができます。
- 一時的にメモリに保存されるため、スクリプトの別の箇所でテーブルを繰り返し呼び出して利用する場合に便利な方法です。
- Order by, Crosstable, Join(→後述)などは先行ロードで使用できないため、こちらのResident Loadを使用する必要があります

- 使い方

- Load文の入力先にResidentでテーブルを指定します。
- 例では、先にロードした「一時テーブル」を「保守データ」テーブルの入力先として、「President一時テーブル;」と指定しています。
- また、不要となった「一時テーブル」はDrop Tableで削除しています。

ロードスクリプト記述

```
1 一時テーブル:
2  LOAD
3      製品名,
4      Date(Date#(発売日, 'YYYYMMDD')) as 発売日
5  FROM [lib://DataFiles/SampleData.xlsx]
6  (ooxml, embedded labels, table is 保守データ);
7
8
9
10 保守データ:
11  LOAD
12      *,
13      Today() - 発売日 as 経過日,
14      If(AddYears(発売日, 3) > Today(), 'Yes', 'No') as 保守状況
15  Resident 一時テーブル;
16
17  Drop Table 一時テーブル;
18
```

実行結果

プレビュー

保守データ

▼

出力

製品名	発売日	経過日	保守状況
製品A	2023/03/17	673	Yes
製品B	2021/03/17	1403	No
製品C	2018/03/17	2499	No

SQL Select

SQL ステートメントを使用すると、任意の SQL コマンドを送信できます。

- 解説

- 以下の構文を使用できます。
*SQL SELECT * from tab1;*
- SELECT ステートメントには、この構文が推奨されます (一貫性のため)。ただし、SQL プレフィックスは SELECT ステートメントではオプションです。

- 使い方

- SQL *sql_command*
- *sql_command*には、有効なSQL コマンドを指定します。
- この例では、データ接続で指定したSnowflakeのデータウェアハウス内のcustomerデータベースに接続しています。

ロードスクリプト記述

```
1 LIB CONNECT TO 'MySQL_172.22.16.4';
2
3 SQL SELECT * FROM sampledب.`店舗`;
```

実行結果

プレビュー QueryResult 出力			
売上店舗ID	店舗名	開店日	退店日
1	渋谷店	2007/03/07	2100/01/01
2	新宿店	2007/04/01	2100/01/01
3	恵比寿店	2007/10/10	2100/01/01
4	大宮店	2010/02/01	2100/01/01
5	川崎店	2009/01/01	2011/03/31

Where

Where句を使って条件を指定して、ソースデータから取り込むデータの絞り込みを行うことができます。

ロードスクリプト記述

- 解説
 - Where句に、データ絞り込みの条件を指定してデータをロードします。
 - Where句では、次の演算子や関数が利用可能です。
=, >, >=, <, <=, AND, OR, Match(), MixMatch(), WildMatch()
 - すでに取り込まれているデータに合致するデータに絞り混む場合には、Exist()を利用します。

```
1 LOAD
2   国ID,
3   国名
4 FROM [lib://DataFiles/SampleData.xlsx]
5 (ooxml, embedded labels, table is 国マスタ)
6 Where 国ID <= 2;
7
```

実行結果

プレビュー	国マスタ	出力
国ID	国名	
1	日本	
2	オーストラリア	

Wildmatch

ワイルドカード(*,?)を使用してパラメータの比較を行います。ワイルドカードを使わずに比較する場合には、match, mixmatchを使用します。

- 解説
 - 最初のパラメータとそれに続くすべてのパラメータを比較し、一致した場合には、一致した数式の番号を返します。
 - 一致しない場合には 0 を返します。
 - * は、任意の文字列、? は、任意の 1 文字と一致します。
- 使い方
 - wildmatch(str, ex1, ex2, ...)
 - str は、比較元の文字列、以降は比較する文字列。

ロードスクリプト記述

```
1 ColorTable:
2 Load * inline [
3     ID, Colorname
4     1, Red1
5     2, Red2
6     3, NewRed
7     4, OldRed
8     5, Orange
9     6, Black
10    7, Blue
11    8, White
12 ];
13
14 ColorSelected:
15 Load
16     Colorname
17 Resident ColorTable
18 Where WildMatch(Colorname, 'B*', 'Red?') > 0;
19
```

実行結果

ColorSelected

Colorname
Red1
Red2
Black
Blue

その他の便利なスクリプト

Auto Calendar

データマネージャを利用して日付データを取り込むと、自動生成セクションにautoCalendarが生成されます。

セクション

+

Main

*** Configuration ***

自動生成セクション

「データマネージャー」でデータが追加された場合には、それに対応するスクリプトが自動的に自動生成セクションに生成されます。

```
1 [autoCalendar]:
2 DECLARE FIELD DEFINITION Tagged ('$date')
3 FIELDS
4 Dual(Year($1), YearStart($1)) AS [Year] Tagged ('$axis', '$year'),
5 Dual('Q'&Num(Ceil(Num(Month($1))/3)),Num(Ceil(Num(Month($1))/3),00)) AS [Quarter] Tagged ('$quarter', '$cyclic'),
6 Dual(Year($1)&'-'&Q'&Num(Ceil(Num(Month($1))/3)),QuarterStart($1)) AS [YearQuarter] Tagged ('$yearquarter', '$qualified'),
7 Dual('Q'&Num(Ceil(Num(Month($1))/3)),QuarterStart($1)) AS [_YearQuarter] Tagged ('$yearquarter', '$hidden', '$simplified'),
8 Month($1) AS [Month] Tagged ('$month', '$cyclic'),
9 Dual(Year($1)&'-'&Month($1), monthstart($1)) AS [YearMonth] Tagged ('$axis', '$yearmonth', '$qualified'),
10 Dual(Month($1), monthstart($1)) AS [_YearMonth] Tagged ('$axis', '$yearmonth', '$simplified', '$hidden'),
11 Dual('W'&Num(Week($1),00), Num(Week($1),00)) AS [Week] Tagged ('$weeknumber', '$cyclic'),
12 Date(Floor($1)) AS [Date] Tagged ('$axis', '$date', '$qualified'),
13 Date(Floor($1), 'D') AS [_Date] Tagged ('$axis', '$date', '$hidden', '$simplified'),
14 If (DayNumberOfYear($1) <= DayNumberOfYear(Today()), 1, 0) AS [InYTD] ,
15 Year(Today())-Year($1) AS [YearsAgo] ,
16 If (DayNumberOfQuarter($1) <= DayNumberOfQuarter(Today()),1,0) AS [InQTD] ,
17 4*Year(Today())&Ceil(Month(Today())/3)-4*Year($1)-Ceil(Month($1)/3) AS [QuartersAgo] ,
18 Ceil(Month(Today())/3)-Ceil(Month($1)/3) AS [QuarterRelNo] ,
19 If(Day($1)<=Day(Today()),1,0) AS [InMTD] ,
20 12*Year(Today())&Month(Today())-12*Year($1)-Month($1) AS [MonthsAgo] ,
21 Month(Today())-Month($1) AS [MonthRelNo] ,
22 If(WeekDay($1)<=WeekDay(Today()),1,0) AS [InWTD] ,
23 (WeekStart(Today())-WeekStart($1))/7 AS [WeeksAgo] ,
24 Week(Today())-Week($1) AS [WeekRelNo] ;
25
26 DERIVE FIELDS FROM FIELDS [売上日] USING [autoCalendar] ;
```

データの追加

+ データカタログから追加

データ接続

接続の新規作成

Basic Training

検索

DataFiles

フォルダ

- 日付から年や月、四半期などを取得
- 年>月などのドリルダウンが可能
- autoCalendarを編集してカレンダーをカスタマイズ可能

Binary Load

他のアプリからデータモデルを取り込むことができます。

ロードスクリプト記述

- 解説

- 別の Qlik Sense アプリや QlikView ドキュメントからデータをロードする際に使用します。
- シート、ストーリー、ビジュアライゼーション、マスター アイテム、変数といった、アプリのその他の要素は含まれません。
- スクリプト内では、1つのバイナリ文しか使用できず、スクリプトの最初の文である必要があります。 (SET文よりも前である必要があります)

- 使い方

- binary [path] filename 構文をスクリプトの先頭に記載します。
- 例では、Qsdemo_selfservice.dvfのデータを取り込んでいます。

```
1 Binary [lib://Google_Drive/root/Qsdemo_selfservice.qvf];  
2
```

この例では、Google Drive のルートフォルダに格納された Qsdemo_selfservice.qvf ファイルをバイナリロードしています。

Qsdemo_selfservice.qvfのデータモデルが取り込まれますが、シート、ストーリー、ビジュアライゼーション、マスター アイテム、変数といった、アプリのその他の要素は含まれません。

For Loop

For .. next制御ステートメントでカウンタ付きの反復スクリプトを実行できます。

ロードスクリプト記述

- 解説

- for と next で囲まれたループ内のステートメントは、カウンタ変数の初期値と最終値で指定された回数分実行されます。

```
1 For counter=1 to 3
2
3     SET vSheetName = 売上実績-2023年$(counter)月;
4
5     LOAD *,
6         '$(vSheetName)' as シート名
7     FROM [lib://DataFiles/SampleData.xlsx]
8         (ooxml, embedded labels, table is [$(vSheetName)]);
9 Next
```

- 使い方

- For counter = expr1 to expr2 [ステートメント] next*
- 例では、ファイル名が「売上実績-2021年1月」から「売上実績-2021年3月」までのファイルをロードしています。
- データのロードと合わせて、取り込んだファイル名をフィールドに追加しています。

実行結果

販売日	商品名	販売価格	シート名
2023/01/01	商品A	100	売上実績-2023年1月
2023/01/02	商品B	200	売上実績-2023年1月
2023/02/01	商品C	300	売上実績-2023年2月
2023/02/02	商品D	400	売上実績-2023年2月
2023/03/01	商品E	500	売上実績-2023年3月
2023/03/02	商品F	600	売上実績-2023年3月

Hide プレフィクス

項目リストに項目名を表示させたくない場合、HidePrefixを利用します。

ロードスクリプト記述

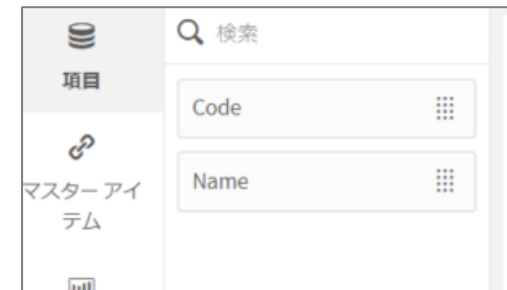
- 解説
 - このテキスト文字列で始まる項目名はすべて、システム項目と同様に非表示になります。
- 使い方
 - set HidePrefix='_';
 - このステートメントを使用すると、システム項目が非表示の場合、アンダースコア () から始まる項目名が項目名リストに表示されません。

```
1 set HidePrefix='_';
2
3 HideList:
4 Load * inline [
5     Code, _Hideitem, Name
6     1, 'xxxxx', Abe
7     2, 'yyyyy', Hamano
8     3, 'wwwwww', Kawahata
9 ];
10
```

実行結果

HideList

Code	_Hideitem	Name
1	xxxxx	Abe
2	yyyyy	Hamano
3	wwwwww	Kawahata



Include

テキストが格納されたファイルを、スクリプト内でインクルードして利用することができます。

- 解説

- Include変数は、スクリプトコードとして評価する必要があるテキストが格納された外部ファイルを指定します。指定した外部ファイル内のテキストはロードスクリプトとして評価されます。
- スクリプトを外部ファイル化することで、複数のアプリで再利用することが可能となります。

- 使い方

- `$(Include=filename)` と記載します。filenameには、テキストが格納されたファイル名を指定します。ファイルが見つからない場合、何もしません。
- `$(Must_Include=filename)` とした場合、ファイルが見つからなかった場合にエラーを生成します。

ロードスクリプト記述

1	<code>\$(Include=lib://DataFiles/SampleScript.txt);</code>
2	

Load Inline

Load...Inlineを使用してインラインテーブルを作成できます。

- 解説

- 検証目的などのためにテーブル形式のデータを直接入力してデータを作成します。
- 少ない件数のデータを手早く入力してテーブルを作成してチャートを試しに作ってみるといった場合に便利な方法です。

- 使い方

- スクリプトエディタで「Load * Inline []」と記述し、[]の中に作成するデータを入力します。
- []内の1行目に項目名、2行目以降に実際のデータをカンマ区切りで入力します。
- 例では、「国, 売上, 売上原価」が項目名、それ以下の「US, 100, 50...」がデータの行にあたります。

ロードスクリプト記述

```
1 売上データ:  
2 LOAD * Inline [  
3 国, 売上, 売上原価  
4 US, 100, 50  
5 UK, 200, 180  
6 JP, 300, 250  
7 ];
```

実行結果

売上データ		
国	売上	売上原価
US	100	50
UK	200	180
JP	300	250

QVD/CSV ファイル保存

Qlik Sense内部に取り込んだデータをQVDやCSV形式のファイルに保存できます。

ロードスクリプト記述

- 解説

- QVDファイル(次頁参照)やCSV形式のファイルへテーブルデータを保存できます。
- テーブルの指定した列のみをエクスポートすることも可能です。

例) Store 列1, 列2 from テーブル名 Into [lib://データ接続名/ファイル名];

- 使い方

- Store関数を利用し、保存先は、[lib://データ接続名/ファイル名] の形式で入力します。
- 例では、[lib://DataFile]のデータ接続のフォルダに「契約データ.qvd」というファイル名で保存します。
- CSVで保存する場合は、拡張子を .csvとして、Store文の末尾に (txt)を付加します。

```
1  契約データ:  
2  LOAD  
3      契約番号,  
4      契約日,  
5      顧客コード,  
6      契約金額  
7  FROM [lib://DataFiles/SampleData.xlsx]  
8  (ooxml, embedded labels, table is 契約データ);  
9  
10 Store 契約データ Into [lib://DataFiles/契約データ.qvd];  
11  
12 Store 契約データ Into [lib://DataFiles/契約データ.csv](txt);  
13
```

QVDとは

QVD (QlikView Data) ファイルは、Qlik Senseが内部的に保持するデータ形式のままエクスポートされたテーブルデータです。

前出の方法でQVDファイルにデータを保存しそれを複数のアプリから参照することで、DBなどのデータソースへ繰り返しアクセスすることなく効率よくデータを利用できます。

QVDの特徴

- データが圧縮されておりサイズが小さい
- Qlik Senseから高速に読み込み、書き込み可能
- 複数のアプリで、共通化されたQVDファイルの再利用が可能
- 1つのデータ テーブルを保持し、ヘッダー、シンボルテーブル、データ テーブルで構成される

Qualify / Unqualify

自動的な紐付けや連結の無効化／有効化を指定することができます。

- 解説
 - Qualifyを利用すると、テーブル名を項目名に自動的に付加し、同じ項目名のキーとして自動的に紐付けられることを回避することができます。
 - 例えば複数のテーブルにキーとして紐付けたくない同名の項目が多く存在するような場合には、このような処理が必要になります。
- 使い方
 - Load文の先頭に「Qualify」を付加します。
 - 例では、「年月」、「部門」、「金額」の項目を共通で持つため、「Qualify *;」を追加し、すべての項目を対象にテーブル名を自動的に付加しています。
 - 最後尾に「Unqualify *;」を追加して、Qualifyの設定を無効化しています。

ロードスクリプト記述

```
1 Qualify *;  
2  
3 LOAD  
4     年月,  
5     部門,  
6     製品名,  
7     金額  
8 FROM [lib://DataFiles/SampleData.xlsx]  
9 (ooxml, embedded labels, table is 実績データ);  
10  
11 LOAD  
12     年月,  
13     部門,  
14     金額  
15 FROM [lib://DataFiles/SampleData.xlsx]  
16 (ooxml, embedded labels, table is 予算データ);  
17  
18 Unqualify *;  
19
```

実行結果

実績データ	予算データ
実績データ.年月	予算データ.年月
実績データ.部門	予算データ.部門
実績データ.製品名	予算データ.金額
実績データ.金額	

Trace

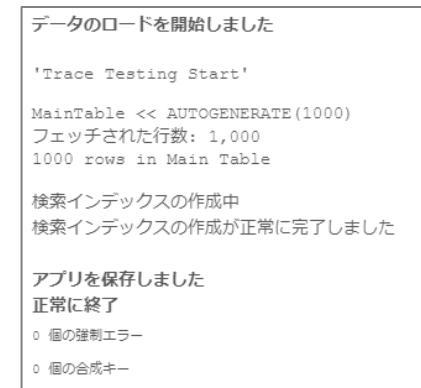
任意の文字列を出力させることができます。

- 解説
 - ロードスクリプトの進捗ウィンドウとスクリプトのログファイルにデバッグ用のメッセージを出力させることができます。
- 使い方
 - *Trace string*
 - *string*に出力させたいメッセージを記載します。
 - この例では、'*Trace Testing Start*'と'*xxxx rows in Main Table*'が進捗ウィンドウやリロードログに出力されます。

ロードスクリプト記述

```
1 Trace 'Trace Testing Start';
2
3 MainTable:
4 Load
5     RowNo() as rowNo,
6     1 as num
7 AutoGenerate 1000;
8
9
10 Let MyMessage = NoOfRows('MainTable') & ' rows in Main Table';
11 Trace $(MyMessage);
12
```

実行結果



データロードエディタ
のデータロード



アプリのリロード
(詳細⇒ロードの履歴)

スクリプトのTips



CreateSearchIndexOnReload

データのリロードの間に検索インデックス ファイルが作成されるかどうかを定義します。

例 1 : データのリロードの間に、検索インデックス項目を作成します。

Set CreateSearchIndexOnReload = 1;

例 2 : 最初の検索リクエストの後で、検索インデックス項目を作成します。

Set CreateSearchIndexOnReload = 0;

※ データのリロードの間に検索インデックス ファイルを作成する利点は、ユーザーによる最初の検索リクエストの待ち時間を回避できることです。これは、検索インデックスの作成によりデータのリロード時間が長くなるという点を考慮する必要があります。

ErrorMode

スクリプトの実行中にエラーが発生したときに、Qlik Sense によって実行されるアクションを定義します。

- **Set ErrorMode = 1;**
 - ✓ デフォルト設定。スクリプトの実行が中止され、ユーザーのアクションが要求されます。
(バッチモード以外)
- **Set ErrorMode = 0;**
 - ✓ Qlik Sense は、エラーを無視し、スクリプトの次のステートメントから、スクリプトの実行を継続します。
- **Set ErrorMode = 2;**
 - ✓ Qlik Sense は、エラーの発生直後に “Execution of script failed…”(スクリプトの実行失敗) と言う
エラーメッセージを表示します。ユーザーに要求されるアクションはありません。

Incremental Load (増分ロード)

継続的に更新されるデータソースから大量のデータがアプリに取り込まれる場合、データセット全体をリロードするのではなく、新しいレコードまたは変更されたレコードのみをロードすることができます。

ロードスクリプト記述

```
1 LIB CONNECT TO 'MySQL_172.22.16.4';
2
3 IDTable:
4 LOAD
5     伝票番号
6 FROM 'lib://DataFiles/増分ロード.qvd'(qvd);
7
8 Let MaxID = peek('伝票番号', 0, IDTable);
9
10 TransactionData:
11 Load *;
12 SQL SELECT * from sampled.`transaction_japan`
13 Where '伝票番号' > '${MaxID}';
14
15 if count(TransactionData) > 0 then
16     Store TransactionData into 'lib://DataFiles/増分ロード.qvd'(qvd);
17 Endif
```

(例)

- 例では、「挿入のみ(更新または削除なし)」を想定しています。
- QVDファイルからIDの最大値を取得します。
- データソースにID最大値よりも大きなIDがあった場合にはデータをロードします。
- 新しいデータセットをQVDへ出力します。

- 増分ロードのパターン
 - 追加のみ (通常はログファイルに使用)
 - 挿入のみ (更新または削除なし)
 - 挿入および更新 (削除なし)
 - 挿入、更新、および削除
- 基本手順
 1. データソースから新規または更新されたデータをロード
 2. QVDファイルから、アプリですでに利用可能なデータをロード
 3. 新しいQVDファイルを作成 (これが、次回増分ロードで使用するファイルとなります)
 4. ロードされるテーブルごとに手順を繰り返す

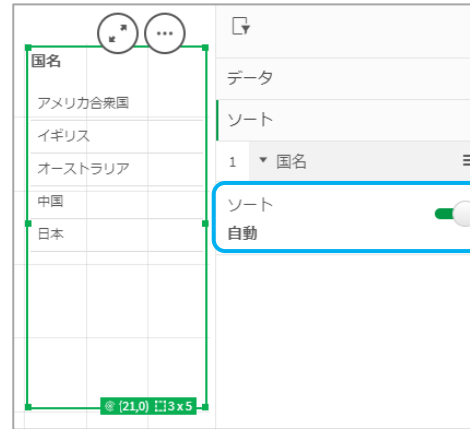
Load Inline – Custom Sort Order

ロード順でソートしたい場合に Inline を利用することができます。

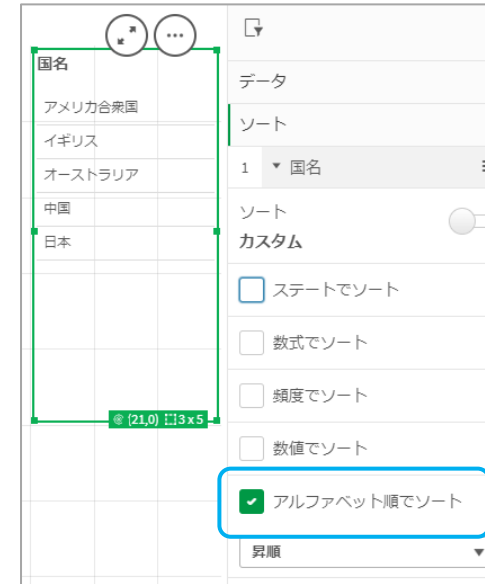
```
1 // ** 国マスタをGDP順に並べ替え
2 GDPOrder:
3 Load * Inline [
4   国名
5   アメリカ合衆国
6   中国
7   日本
8   イギリス
9   オーストラリア
10 ];
11
12 国マスタ:
13 LOAD
14   国ID,
15   国名
16 FROM [lib://DataFiles/国マスタ.xlsx]
17 (ooxml, embedded labels, table is 国マスタ);
18
19 Drop Table GDPOrder;
```

国マスタ

国名	国ID
日本	1
オーストラリア	2
アメリカ合衆国	3
中国	4
イギリス	5



自動ソート



アルファベット順でソート



ロード順でソート

この例の場合、GDP順でソートされることになります。

ScriptError

最後に実行されたスクリプト ステートメントのエラー コードを返します。

この変数は、各スクリプト ステートメントが正常に実行されるたびに、0 にリセットされます。エラーが発生すると、変数は内部 Qlik Sense エラー コードに設定されます。エラー コードは、数値とテキスト値のデュアル値です。以下のようなエラー コードがあります。

エラー コード	説明
0	エラーなし
1	一般エラー
2	構文エラー
3	一般的な ODBC エラー
4	一般的な OLE DB エラー
5	一般的なカスタム データベース エラー
6	一般的な XML エラー
7	一般的な HTML エラー
8	ファイルが見つかりません
9	データベースが見つかりません
10	テーブルが見つかりません
11	項目が見つかりません
12	ファイル形式が正しくありません
13	BIFF エラー
14	暗号化された BIFF エラー
15	サポートされていないバージョンの BIFF エラー
16	セマンティック エラー

記述例

```
1 Set ErrorMode=0;  
2 LET vScriptErr = 0;  
3  
4 LOAD * from abc.qvf;  
5 if ScriptError=8 then  
6     exit script;  
7     //no file;  
8 end if  
9
```

エラー内容

以下のエラーが発生しました:

No qualified path for file: ***

ここでエラーが発生しました:

LOAD * from abc.qvf

アプリを保存しました

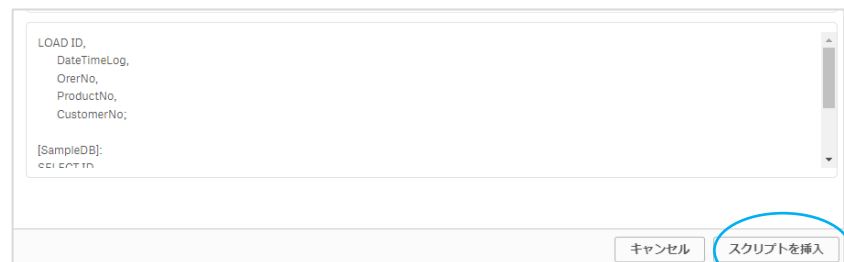
終了時にエラー/警告あり

1 個の強制エラー

0 個の合成キー

SQL Select の記載パターン

下記パターンは、すべて有効です。



```
LOAD ID,
    DateTimeLog,
    OrerNo,
    ProductNo,
    CustomerNo;

[SampleDB]:
SELECT ID,
```

// 「スクリプトの挿入」を選択した場合

```
LOAD ID,
    DateTimeLog,
    OrerNo,
    ProductNo,
    CustomerNo;

[SampleDB]:
SELECT ID,
    DateTimeLog,
    OrerNo,
    ProductNo,
    CustomerNo
FROM Demodb.SampleDB;
```

// テーブルを指定しない場合、'Query Result' 名のテーブルが作成される

```
LOAD ID,
    DateTimeLog,
    OrerNo,
    ProductNo,
    CustomerNo;

SELECT ID,
    DateTimeLog,
    OrerNo,
    ProductNo,
    CustomerNo
FROM Demodb.SampleDB;
```

QueryResult	
ID	
DateTimeL...	
OrerNo	
ProductNo	
CustomerNo	

// Load ではなく SQL を利用した場合

```
[SampleDB]:
SQL SELECT
    ID,
    DateTimeLog,
    OrerNo,
    ProductNo,
    CustomerNo
FROM Demodb.SampleDB;
```

// SQL の記載はオプション

```
[SampleDB]:
SELECT
    ID,
    DateTimeLog,
    OrerNo,
    ProductNo,
    CustomerNo
FROM Demodb.SampleDB;
```

バケット(フォルダ)内の複数ファイルの一括ロード

- Amazon S3 Connector はワイルドカードに対応しないため、特定バケットから複数ファイルを取得したい場合には、For each文で取得できます。
- Filelistの引数にS3接続の定義を渡せるため、これによってバケット内のファイルリストを取得します。
- リストからIF+WildMatch関数で指定した条件(ここにワイルドカード*を使用)のファイル名のデータを取得します。

```
for each file in filelist('lib://Amazon_S3/')  
  IF WildMatch('${file}', '*.csv') THEN  
    LOAD  
      *  
    FROM [$(file)](txt);  
  END IF  
next file
```

Section Access



Section Access とは

データセキュリティの管理

Section Accessは、アプリケーションのセキュリティを制御するために使用されます。セキュリティテーブルを設定して、誰がどのフィールドを表示するかを定義し、ユーザーがアプリケーションを開いた際に、権限で抑制されたデータを非表示とすることができます。

目的

- ユーザー毎に、**行レベル**のアクセス制御を行いたい
- ユーザー毎に、**列レベル**のアクセス制御を行いたい
- ユーザー毎に、**アプリ**へのアクセス制御を行いたい

ロードスクリプトの記述方法

```
1 // Section Access を利用した、行・列へのアクセス制御
2
3 Section Access;
4
5 AuthorizationTable:
6 Load
7     ACCESS,
8     USER-EMAIL,
9     REDUCTION,
10    Upper(OMIT) as OMIT
11 Inline [
12     ACCESS, USER.EMAIL, REDUCTION, OMIT
13     ADMIN, satoshiabe002@gmail.com, ,
14     USER, qlikuser74@gmail.com, 第一営業部,
15     USER, qlikuser84@gmail.com, 第一営業部,
16     USER, usera.qlik@gmail.com, 第二営業部,
17     USER, userb.qlik@gmail.com, 第二営業部, Rating
18 ];
19
20 Section Application;
21
22 RatingTable:
23 Load
24     [Sales Email],
25     [RATING]
26 inline [
27     Sales Email, RATING
28     qlikuser74@gmail.com, 5
29     qlikuser84@gmail.com, 4
30     usera.qlik@gmail.com, 3
31     userb.qlik@gmail.com, 4
32 ];
33
34 SalesData:
35 Load
36     [ID],
37     [Sales Person],
38     [Sales Email],
39     [Department],
40     [Sales Target],
41     [Sales Manager],
42     [Department] as REDUCTION
43 inline [
44     ID, Sales Person, Sales Email, Department, Sales Target, Sales Manager
45     1, 小林, qlikuser74@gmail.com, 第一営業部, 100, 高瀬
46     2, 大田, qlikuser84@gmail.com, 第一営業部, 200, 高瀬
47     3, 柳, usera.qlik@gmail.com, 第二営業部, 150, 平野
48     4, 安西, userb.qlik@gmail.com, 第二営業部, 150, 平野
49 ];
50
51
```

Section Accessテーブル
(セキュリティのテーブル)を
定義するスクリプト

アプリのデータロードを
定義するスクリプト

Section Access;

Section Application;

ACCESS	USER	REDUCTION	OMIT
--------	------	-----------	------

Sales Email	RATING
-------------	--------

ID	Sales Person	Department	Sales Email	Sales Target	Sales Manager	REDUCTION
----	--------------	------------	-------------	--------------	---------------	-----------

ACCESS

対応するユーザーに与えられるアクセス権を定義します。
ADMIN(アプリ内のすべてのデータにアクセス可能)、または**USER**(セキュリティテーブルで指定されたデータのみアクセス可能)を設定します。

USER

Qlik Sense ユーザーに対応する「USERID」を指定します。

OMIT

ユーザもしくはグループに対して利用不可とする列名を指定します。
ワイルドカードを使用することができます。空にすることもできます。

※ ここに記載したフィールド値の行のみ表示できます。

※ ここに記載したフィールド値は表示できません。

※ 同一名称
例えば、どちらも
Departmentとすることが可能

行レベルのアクセス制御

REDUCTIONに設定されたフィールドに関連づけられた行のみ表示

ADMINが設定されているため、すべての項目が表示されます。

ID	Sales Person	Department	Sales Email	Sales Target	Sales Manager	REDUCTION
1	小林	第一営業部	qlikuser74@xxx	100	高瀬	第一営業部
2	大田	第一営業部	qlikuser84@xxx	200	高瀬	第一営業部
3	柳	第二営業部	usera.qlik@xxx	150	平野	第二営業部
4	安西	第二営業部	userb.qlik@xxx	150	平野	第二営業部

REDUCTIONに「第一営業部」が設定されているため、第一営業部の行のみ表示されます。

ID	Sales Person	Department	Sales Email	Sales Target	Sales Manager	REDUCTION
1	小林	第一営業部	qlikuser74@xxx	100	高瀬	第一営業部
2	大田	第一営業部	qlikuser84@xxx	200	高瀬	第一営業部

REDUCTIONに「第二営業部」が設定されているため、第二営業部の行のみ表示されます。

ID	Sales Person	Department	Sales Email	Sales Target	Sales Manager	REDUCTION
3	柳	第二営業部	usera.qlik@xxx	150	平野	第二営業部
4	安西	第二営業部	userb.qlik@xxx	150	平野	第二営業部

```
3 Section Access;
4
5 AuthorizationTable:
6 Load
7     ACCESS,
8     USER.EMAIL,
9     REDUCTION,
10    Upper(OMIT) as OMIT
11 Inline [
12     ACCESS, USER.EMAIL, REDUCTION, OMIT
13     ADMIN, satoshiabe002@gmail.com, ,
14     USER, qlikuser74@gmail.com, 第一営業部, Rating
15     USER, qlikuser84@gmail.com, 第一営業部, Rating
16     USER, usera.qlik@gmail.com, 第二営業部, Rating
17     USER, userb.qlik@gmail.com, 第二営業部, Rating
18 ];
```

列レベルのアクセス制御

OMITに設定されたフィールドを非表示

ADMINが設定されているため、すべての項目が表示されます。

ID	Sales Person	Department	Sales Email	Sales Target	Sales Manager	Rating
1	小林	第一営業部	qlikuser74@xxx	100	高瀬	5
2	大田	第一営業部	qlikuser84@xxx	200	高瀬	4
3	柳	第二営業部	usera.qlik@xxx	150	平野	3
4	安西	第二営業部	userb.qlik@xxx	150	平野	4

REDUCTIONに「第一営業部」が設定されていて、OMITに何も設定されていないので、第一営業部のすべて列が表示されます。

ID	Sales Person	Department	Sales Email	Sales Target	Sales Manager	Rating
1	小林	第一営業部	qlikuser74@xxx	100	高瀬	5
2	大田	第一営業部	qlikuser84@xxx	200	高瀬	4

REDUCTIONに「第二営業部」が設定されていて、OMITに「Rating」が設定されているので、第二営業部のRatingを除いた列のみ表示されます。

ID	Sales Person	Department	Sales Email	Sales Target	Sales Manager	Rating
3	柳	第二営業部	usera.qlik@xxx	150	平野	
4	安西	第二営業部	userb.qlik@xxx	150	平野	

```
3 Section Access;
4
5 AuthorizationTable:
6 Load
7     ACCESS,
8     USER.EMAIL,
9     REDUCTION,
10    Upper(OMIT) as OMIT
11 Inline [
12     ACCESS, USER.EMAIL, REDUCTION, OMIT
13     ADMIN, satoshiabe002@gmail.com, ,
14     USER, qlikuser74@gmail.com, 第一営業部,
15     USER, qlikuser84@gmail.com, 第一営業部,
16     USER, usera.qlik@gmail.com, 第二営業部, Rating
17     USER, userb.qlik@gmail.com, 第二営業部, Rating
18 ];
```

アプリへのアクセス制御

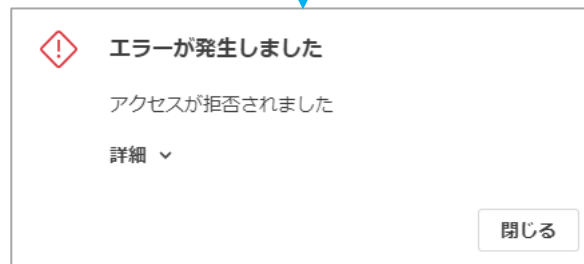
リストされたユーザーのみアクセス可能

```
1 Section Access;  
2 Load * inline [  
3     ACCESS, USER.EMAIL  
4     ADMIN, satoshiabe002@gmail.com  
5     USER, qlikuser74@gmail.com  
6     USER, qlikuser84@gmail.com  
7 ];
```

USER.EMAILにリストされていないユーザは、アプリを開くことができません。

例えば、下記ユーザがアプリを開こうとするとエラーが発生します。

usera.qlik@gmail.com
userb.qlik@gmail.com



営業担当者情報

ID	Sales Person	Department	Sales Email	Sales Target	Sales Manager
1	小林	第一営業部	qlikuser74@gmail.com	100	高瀬
2	大田	第一営業部	qlikuser84@gmail.com	200	高瀬
3	柳	第二営業部	usera.qlik@gmail.com	150	平野
4	安西	第二営業部	userb.qlik@gmail.com	150	平野

Section Accessに登録されたユーザはアプリを開くことができます。

営業別 評価一覧

ID	Sales Person	Sales Email	RATING
1	小林	qlikuser74@gmail.com	5
2	大田	qlikuser84@gmail.com	4
3	柳	usera.qlik@gmail.com	3
4	安西	userb.qlik@gmail.com	4

3 日目のまとめ

3 日目に学習した内容

- データロードエディタの使い方
- ファイルやDBMSからデータをロードする方法
- ロードスクリプトの基本
- ロードスクリプトによる高度なデータ変換
- 便利なスクリプトの使用方法和Tips
- Section Access の利用方法



ロードスクリプトの使い方と頻出構文を学習しました。

複雑なデータ変換も、Qlik Sense のみでデータ準備からモデル作成まで行うことができます。

Q&A

ご質問と回答一覧 (1/2)

No.	ご質問	回答
1	「スクリプト」アプリは、複数のアプリから取り込めるとの事ですが 1. 共通処理を「スクリプト」アプリに実装して、視覚化用のアプリでそれを取り込み、個別処理（スクリプト）をそれぞれの視覚化用のアプリの中に追加で組むといった事も可能でしょうか 2. 「スクリプト」アプリを、別の「スクリプト」アプリから取り込むことも可能でしょうか	スクリプトは視覚化用のアプリで使用することはできません。データを直接準備するための機能で、QVDやCSVを作成するために使用します。共通処理のためのスクリプトを複数アプリで使用したい場合はQVSファイルをご利用ください。 https://help.qlik.com/ja-JP/cloud-services/Subsystems/Hub/Content/Sense_Hub/LoadData/QVS/create-common-scripting-qvs.htm
2	【CROSS TABLE】 Monthはなぜ正しく認識されますか？ 左軸が2列なので、3列目以降の一行目がMonthとして認識されるということでしょうか？	Crosstable(Month, Sales,2)の場合、最初の2項目はキーで、その次が属性となり行見出しが値として使用されるので、Monthの値が使用されます。
3	Partial Load部分はスケジュールでリロード組むことはできないですか？	はい可能です。スケジュールを設定するときに、[部分的なリロード] をオンにして設定してください。
4	inlineで取り込むデータを見ると、カンマ(,)と縦棒()の両ケースがあるように見受けられます。 使い分けはありますか？ 或いはどちらを使っても問題ないですか？	区切り文字をカンマから変更する場合はDelimiter isで指定します。下記をご参照ください。 https://help.qlik.com/ja-JP/sense/November2024/Subsystems/Hub/Content/Sense_Hub/Scripting/inline-loads-qs.htm#anchor-6
5	本日を含む3日間のトレーニングの録画は「マーケティングサイト」に公開予定とのことですが、可能であればURLをご教示いただけますでしょうか。	まだ準備中ですので公開できましたらご連絡させていただきます。

ご質問と回答一覧 (2/2)

No.	ご質問	回答
6	SectionAccessについて、例ではインラインロードで定義していますが、例えばユーザー情報とリダクションとする項目のデータを持っているExcelをデータソースとしてSection Accessテーブルを作るようにすれば、スクリプトをいじることなくExcelの編集でアクセス権制御ができるようになる、という理解で合ってますか？	はい、ご認識の通りです。インラインロードである必要はありません。
7	ロードスクリプトにおいて、あらかじめ用意された関数ではなく、独自関数を作成、利用することは可能でしょうか？	独自の関数を利用することはできません。あらかじめ用意された関数をご利用いただく必要があります。
8	スクリプトは、次第に複雑化してしまいそうです。 一方で、「スクリプト」アプリや Include、QVSファイル、QVDファイル等、使い分ければ便利そうな機能も沢山ありそうに感じました。 作り方のプラクティスや、使い分けなど解説があればありがたいのですが、そういったものはございますでしょうか。	弊社の実施している Webセミナー Qlik TECH TALKやQlik Tipsのアーカイブをご利用ください。YoutubeとSlideshareにあります。 https://www.youtube.com/user/QTJMarketing

その他の情報

更なる学習のために

過去のWebセミナーの情報

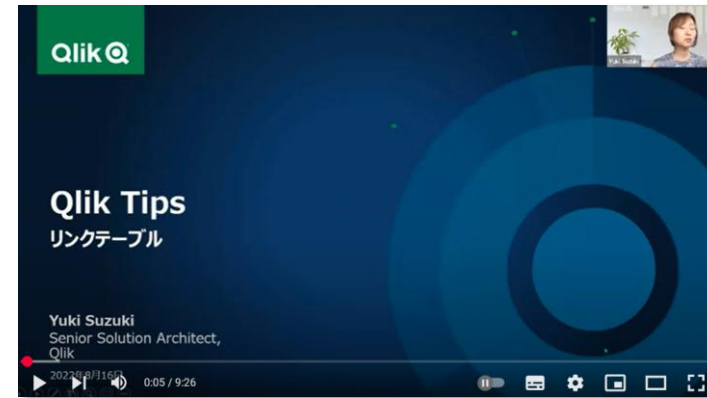
Section Access

<https://youtu.be/01nyfJs6naQ>



リンクテーブル

<https://youtu.be/LBm8adQpBT4>



Qlik Data Gateway – Direct Access

<https://youtu.be/xUjsXnwWhRg>



AutoCalendar

<https://youtu.be/ouYhoYsxHd8>



Qlik Tips プレイリスト

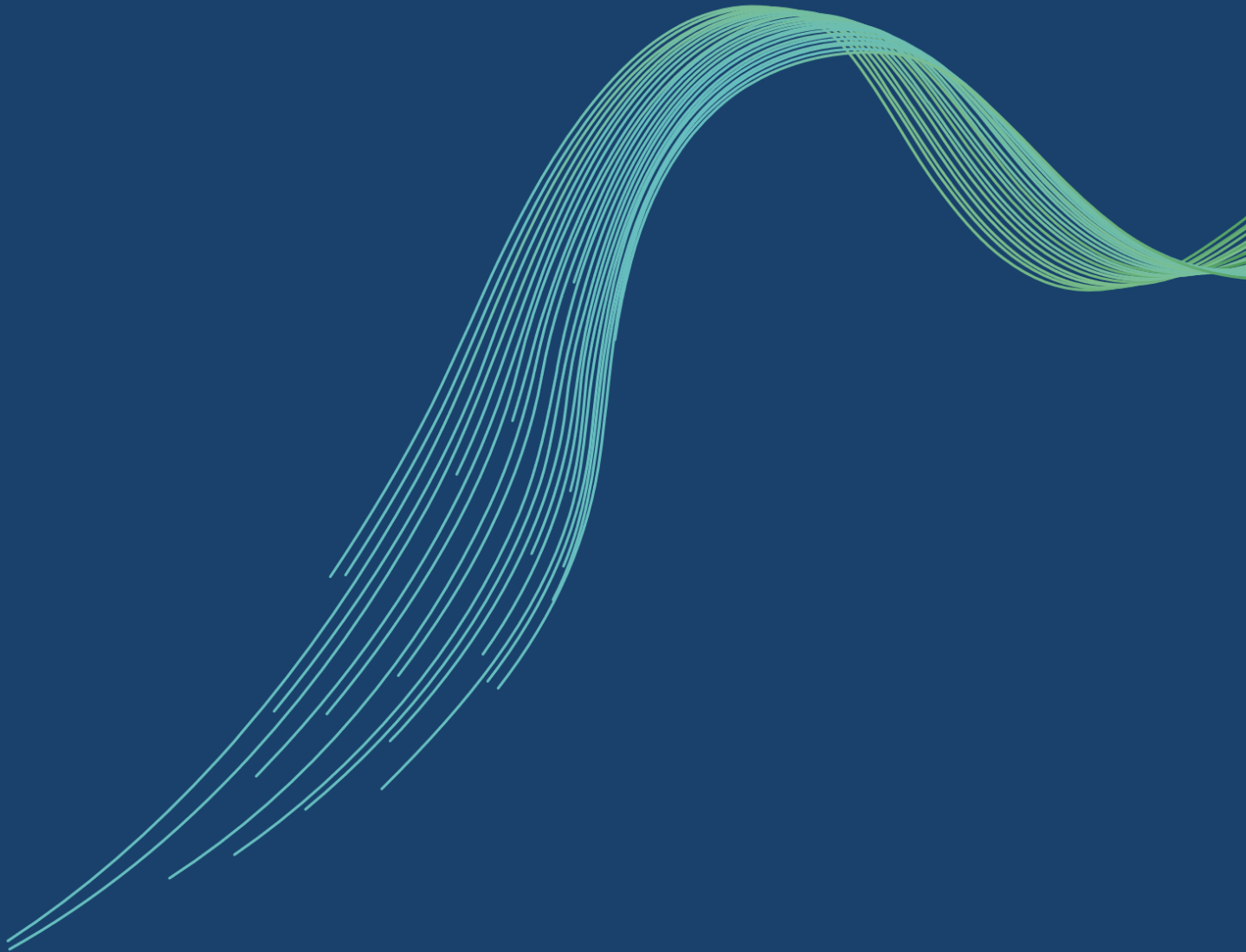
Qlik 使用のちょっとした技術やコツを短い動画でご紹介しています。



<https://www.youtube.com/playlist?list=PLTGfcjh8Hh5a6vjjOlBa7mAtL2H0hw7K>

	Qlik Tips: アプリの自動化の新テンプレート Qlik Japan	Qlik Tips: チャートデザイン~アドバンスドスタイリング Qlik Japan	Qlik Tips: チャートデザイン Qlik Japan	Qlik Tips: K-means関数 Qlik Japan
	Qlik Tips: Entitlement Analyzerのご紹介 Qlik Japan	Qlik Tips: Qlik特有のデータ型「Dual値」の詳細 Qlik Japan	Qlik Tips: クラウドストレージからのデータロード Qlik Japan	Qlik Tips: 複数言語対応 Qlik Japan
	Qlik Tips: スクリプトにおけるエラー処理について Qlik Japan	Qlik Tips: Null値の課題と対策 Qlik Japan	Qlik Tips: スクリプト実行可能なSSEの実装例とその応用 Node.js編 Qlik Japan	Qlik Tips: 画面からの入力機能 Qlik Japan
	Qlik Tips: 理解を促すビジュアライゼーション Qlik Japan	Qlik Tips: チャートデザイン~効果的な色の使用例 Qlik Japan	Qlik Tips: IntervalMatchとSlowly Changing Dimensions Qlik Japan	Qlik Tips: 構内図 Qlik Japan
	Qlik Tips: Mapsubstring関数を利用したデータの全角半角変換 および文字列のアスキーコード変換 Qlik Japan	Qlik Tips: 日付データの取り扱い Qlik Japan	Qlik Tips: 階層データの分析 Qlik Japan	Qlik Tips: 書式・変換関数と日付関数 Qlik Japan
	Qlik Tips: Qlik Senseで無制限精度の数値演算を実現する - Windows版Qlikエンジンのサーバーサイド... Qlik Japan	Qlik Tips: Qlik Sense Enterprise SaaS のイベントの取得とレビュー Qlik Japan	Qlik Tips: コンテナ Qlik Japan	Qlik Tips: テーブルの機能 Qlik Japan
	Qlik Tips: Aggr関数の使いどころ Qlik Japan	Qlik Tips: Qlik Sense Search Cheat Sheetの紹介 Qlik Japan	Qlik Tips: 書式指定アイテム Qlik Japan	Qlik Tips: AutoCalendarのロジック解説と会計期への対応 Qlik Japan
	Qlik Tips: Mobile アプリで、場所を問わずデータを活用しませんか？ Qlik Japan	Qlik Tips: Qlik Senseでサンプルデータを作成する方法 Qlik Japan	Qlik Tips: IntervalMatch Qlik Japan	Qlik Tips: スクリプトエディターのデバッグ機能 Qlik Japan
	Qlik Tips: Qlik Sense SaaSでソフトウェア開発ライフサイクルプロセスを活用 Qlik Japan	Qlik Tips: リロードタスクのログデータの取得 Qlik Japan	Qlik Tips: 並列ステート Qlik Japan	Qlik Tips: 地図~ラインレイヤー、背景レイヤー Qlik Japan

Appendix

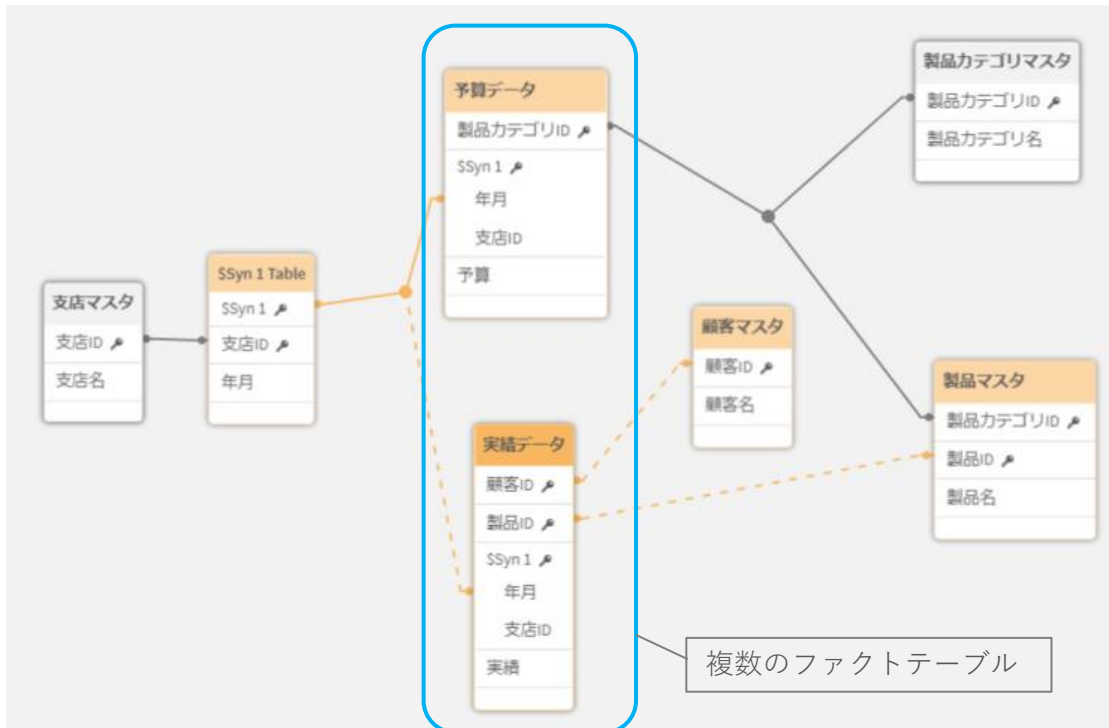


Link Tableと循環参照

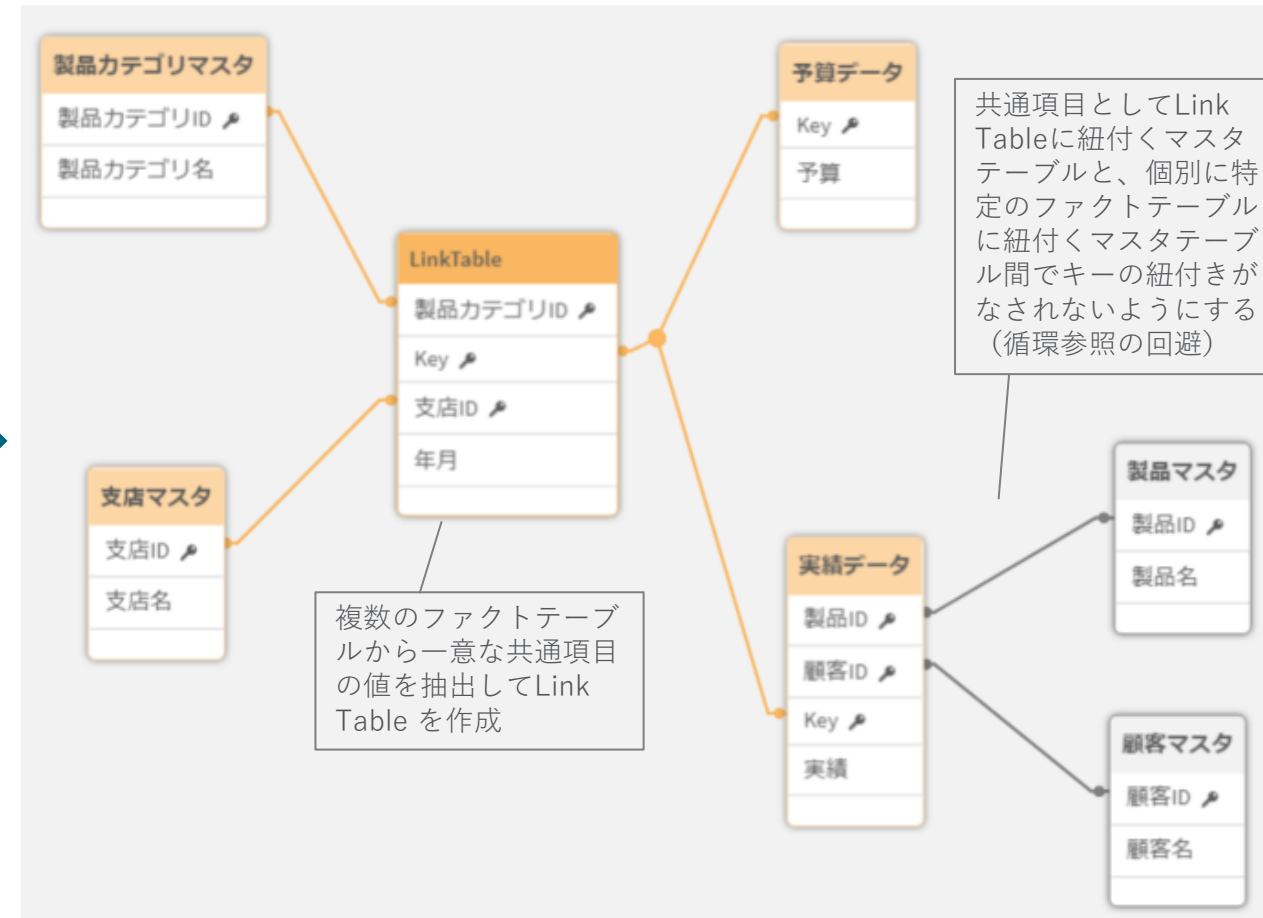
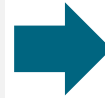


Link Table

- 集計の対象となるファクトテーブルが複数存在する（マルチファクト）場合、リンクテーブルを使用します



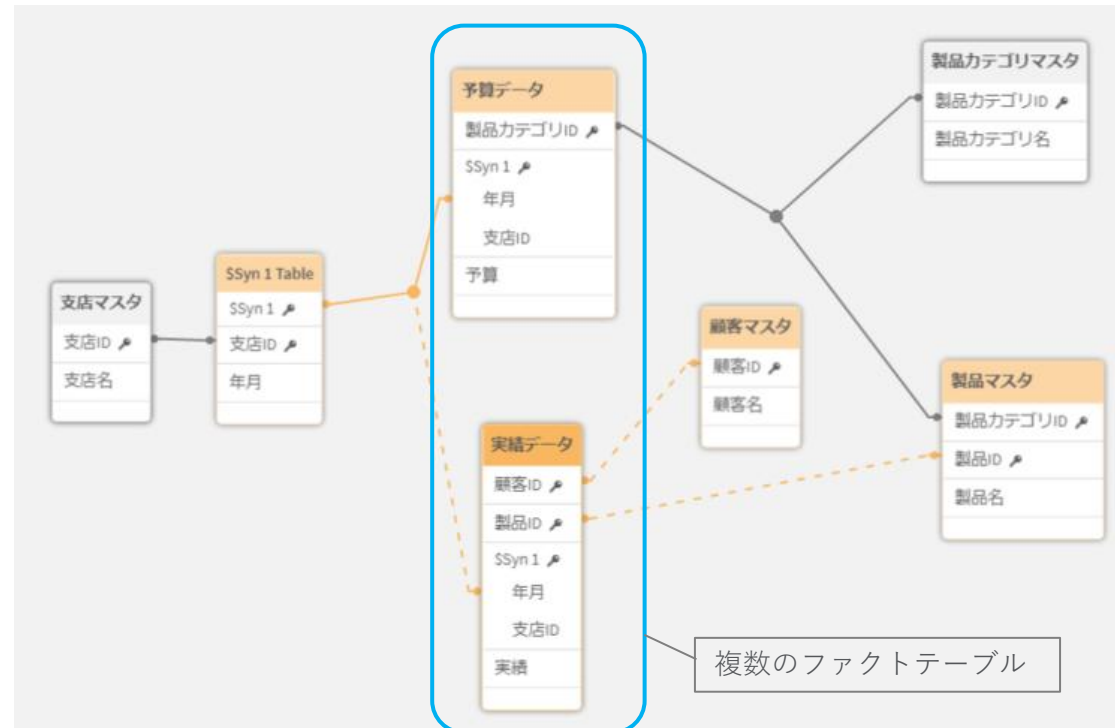
予算では「年月」「支店」「製品カテゴリ」でデータを持っているが
実績では「年月」「営業担当」「顧客」「製品」でデータを持っている。



Link Table は、複数のファクトテーブルを紐付けるすべてのキー項目を格納したテーブル。

Link Tableの利用 (1/3)

予算と実績を組み合わせるなど、集計の対象となるファクトテーブルが複数存在するデータモデルを一般的に「マルチファクト」と呼びます。実際のケースではデータモデルが複雑となってしまう分析が難しくなるケースがあります。



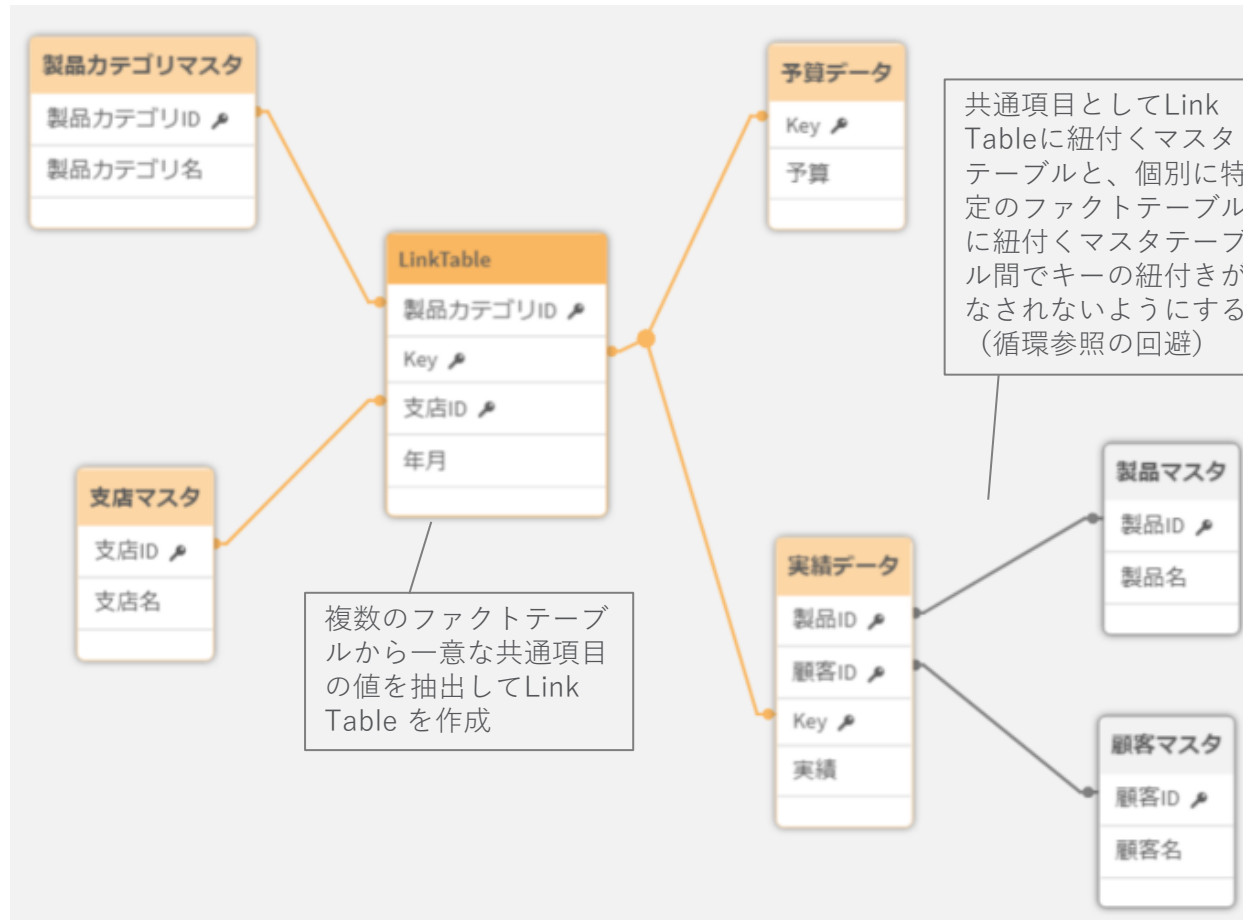
LinkTable(before).txt

予算では「年月」「支店」「製品カテゴリ」でデータを持っているが、実績では「年月」「営業担当」「顧客」「製品」でデータを持っている。

さらに予算、実績、速報、見込み等といった形でファクトテーブルの数が増えたり、マスタテーブルが多くなったりするとデータモデルが複雑となり、なかなか求める形のものが作成できないケースも起こります。

Link Tableの利用 (2/3)

「マルチファクト」の場合の対処方法としてLink Table を利用します。



Link Table は複数のファクトテーブルを紐付ける全てのキー項目を格納したテーブルで、このLink Tableを中心としてファクトテーブルとディメンションテーブル(マスタテーブル)を紐付ける役割を担います。

LinkTable(after).txt

Link Table は、複数のファクトテーブルを紐付けるすべてのキー項目を格納したテーブル。

Link Tableの利用 (3/3)

Link Tableの作成ステップ

ステップ	詳細	スクリプトファイル
共通の項目を含んだ複数のファクトテーブルをロード	<ul style="list-style-type: none">データの粒度が異なる場合にはその粒度も合わせます。先ほどの例では「予算データ」は「製品カテゴリID」、「実績データ」は「製品ID」とそれぞれ異なる階層の項目で集計されていたため、ここではより上位階層の「製品カテゴリID」に粒度を揃えます。複数ファクトテーブル間を紐付ける共通のキー項目を複合キーとして一つの項目に纏めます。	予算データ.txt Mapping.txt 実績データ.txt
複数のファクトテーブルから一意な共通項目の値を抽出してLink Tableを作成	<ul style="list-style-type: none">共通項目を1つの項目に纏めて作成した複合キー（ここでの例では「Key」の項目）とします。全ての共通の項目（ここでの例では「年月」「支店ID」「製品カテゴリID」）を格納します。	LinkTable.txt
ディメンションテーブル(マスタデータ)をロード	<ul style="list-style-type: none">「支店マスタ」、「製品マスタ」、「製品カテゴリマスタ」、「顧客マスタ」をロードします。循環参照を回避するため、「製品マスタ」から「製品カテゴリID」を削除します。	各マスタ.txt
複合キーへのAutoNumberの追加	<ul style="list-style-type: none">「年月」、「支店ID」、「製品カテゴリID」の3つの項目をまとめて、「Key」という名称の項目を作成します。	

Link Table の利用については、Qlik Japan 公認ブログに記載されています。

<https://www.qlikspace.net/%E8%A4%87%E6%95%B0%E3%81%AE%E3%83%95%E3%82%A1%E3%82%AF%E3%83%88%E3%83%86%E3%83%BC%E3%83%96%E3%83%AB%E3%81%AE%E3%83%AA%E3%83%B3%E3%82%AF-link-table/>

<サンプルスクリプト>

LinkTable(before).txt

```
1  【予算データ】:
2  Load * Inline [
3  年月,支店ID,製品カテゴリID,予算
4  201501,1,1,200
5  201501,1,2,120
6  201501,2,1,220
7  201501,2,2,230
8  201502,1,1,120
9  201502,1,2,130
10 201502,2,1,200
11 201502,2,2,80
12 201503,1,1,120
13 201503,1,2,90
14 201503,2,1,150
15 201503,2,2,170
16 ];
17
18 【実績データ】:
19 Load * Inline [
20 年月,支店ID,製品ID,顧客ID,実績
21 201501,1,1,1,120
22 201501,1,1,2,120
23 201501,1,2,1,100
24 201501,2,1,1,100
25 201501,2,1,2,90
26 201501,2,2,2,260
27 201502,1,1,1,90
28 201502,1,2,2,140
29 201502,2,1,1,100
30 201502,2,1,2,130
31 201502,2,2,1,30
32 201502,2,2,2,60
33 201503,1,1,2,130
34 201503,1,2,2,80
35 201503,2,1,1,80
36 201503,2,1,2,90
37 201503,2,2,1,90
38 201503,2,2,2,100
39 ];
40
```

```
41 【支店マスタ】:
42 Load * Inline [
43 支店ID, 支店名
44 1,関東支店
45 2,関西支店
46 ];
47
48 【製品マスタ】:
49 Load * Inline [
50 製品ID, 製品名, 製品カテゴリID
51 1,製品A,1
52 2,製品B,2
53 ];
54
55 【製品カテゴリマスタ】:
56 Load * Inline [
57 製品カテゴリID,製品カテゴリ名
58 1,製品カテゴリA
59 2,製品カテゴリB
60 ];
61
62 【顧客マスタ】:
63 Load * Inline [
64 顧客ID,顧客名
65 1,顧客A
66 2,顧客B
67 ];
```

<サンプルスクリプト> *LinkTable(after).txt*

1	//「年月」「支店ID」「製品カテゴリID」の複合キーを作成して予算データをロード	51	201502,2,1,1,100	100	201501,2,2,2,260
2	[予算データ]:	52	201502,2,1,2,130	101	201502,1,1,1,90
3	Load	53	201502,2,2,1,30	102	201502,1,2,2,140
4	年月 & ' ' & 支店ID & ' ' & 製品カテゴリID AS Key,	54	201502,2,2,2,60	103	201502,2,1,1,100
5	予算	55	201503,1,1,2,130	104	201502,2,1,2,130
6	Inline [56	201503,1,2,2,80	105	201502,2,2,1,30
7	年月,支店ID,製品カテゴリID,予算	57	201503,2,1,1,80	106	201502,2,2,2,80
8	201501,1,1,200	58	201503,2,1,2,90	107	201503,1,1,2,130
9	201501,1,2,120	59	201503,2,2,1,90	108	201503,1,2,2,80
10	201501,2,1,220	60	201503,2,2,2,100	109	201503,2,1,1,80
11	201501,2,2,230	61];	110	201503,2,1,2,90
12	201502,1,1,120	62		111	201503,2,2,1,90
13	201502,1,2,130	63	//予算データの一意な共通項目の値をLink Tableに格納	112	201503,2,2,2,100
14	201502,2,1,200	64	LinkTable:	113];
15	201502,2,2,80]	65	Load Distinct	114	
16	201503,1,1,120	66	年月 & ' ' & 支店ID & ' ' & 製品カテゴリID AS Key,	115	// [支店マスタ]:
17	201503,1,2,90	67	年月,	116	Load * Inline [
18	201503,2,1,150	68	支店ID,	117	支店ID, 支店名
19	201503,2,2,170	69	製品カテゴリID	118	1,関東支店
20];	70	Inline [119	2,関西支店
21		71	年月,支店ID,製品カテゴリID,予算	120];
22	//「製品ID」を「製品カテゴリID」に変換するためのMapping Tableを作成	72	201501,1,1,200	121	
23	Mapping Product:	73	201501,1,2,120	122	// [製品マスタ]:
24	Mapping Load	74	201501,2,1,220	123	Load
25	製品ID,	75	201501,2,2,230	124	製品ID,
26	製品カテゴリID	76	201502,1,1,120	125	製品名
27	Inline [77	201502,1,2,130	126	Inline [
28	製品ID, 製品名, 製品カテゴリID	78	201502,2,1,200	127	製品ID, 製品名, 製品カテゴリID
29	1,製品A,1	79	201502,2,2,80	128	1,製品A,1
30	2,製品B,2	80	201503,1,1,120	129	2,製品B,2
31];	81	201503,1,2,90	130];
32		82	201503,2,1,150	131	
33	//「製品カテゴリID」はApplyMap関数を使ってMapping Tableより取得	83	201503,2,2,170	132	// [製品カテゴリマスタ]:
34	「年月」「支店ID」「製品カテゴリID」の複合キーを作成して実績データをロード	84];	133	Load * Inline [
35	[実績データ]:	85		134	製品カテゴリID,製品カテゴリ名
36	Load	86	// //実績データの一意な共通項目の値をLink Tableに格納	135	1,製品カテゴリA
37	年月 & ' ' & 支店ID & ' ' & ApplyMap('Mapping_Product',製品ID) AS Key,	87	LinkTable:	136	2,製品カテゴリB
38	製品ID,	88	Load Distinct	137];
39	顧客ID,	89	年月 & ' ' & 支店ID & ' ' & ApplyMap('Mapping_Product',製品ID) AS Key,	138	
40	実績	90	年月,	139	// [顧客マスタ]:
41	Inline [91	支店ID,	140	Load * Inline [
42	年月,支店ID,製品ID,顧客ID,実績	92	ApplyMap('Mapping_Product',製品ID) AS 製品カテゴリID	141	顧客ID,顧客名
43	201501,1,1,1,120	93	Inline [142	1,顧客A
44	201501,1,1,2,120	94	年月,支店ID,製品ID,顧客ID,実績	143	2,顧客B
45	201501,1,2,1,100	95	201501,1,1,1,120	144];
46	201501,2,1,1,100	96	201501,1,1,2,120	145	
47	201501,2,1,2,90	97	201501,1,2,1,100	146	
48	201501,2,2,2,260	98	201501,2,1,1,100	147	AutoNumber (年月 & ' ' & 支店ID & ' ' & 製品カテゴリID)
49	201502,1,1,1,90	99	201501,2,1,2,90		
50	201502,1,2,2,140	100	201501,2,2,2,260		

Circular References (循環参照)

Qlik Senseではデータをロードする際、複数のテーブル間に存在する同名の項目をキーとして、自動的にテーブルを関連付けます。しかし、データモデルに問題がある場合、「循環参照」と呼ばれる警告が表示されることがあります。

テーブル間の関連付けがループしている状態となります。

終了時にエラー/警告あり

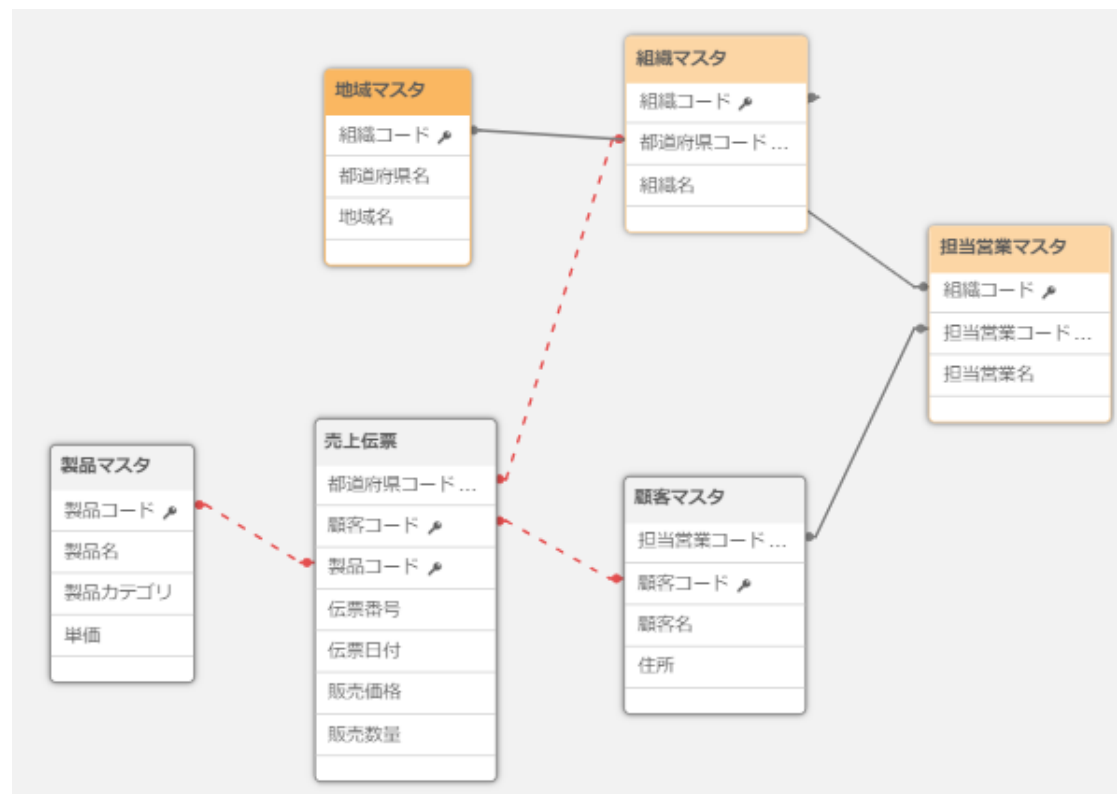
循環参照:

One or more loops have been detected in your database structure. Loops may cause ambiguous results and should therefore be avoided. Loop(s) will be cut by setting one or more tables as loosely coupled.

0 個の強制エラー

0 個の合成キー

データモデルの関連付けがループすると正しく処理されないため、循環参照が発生した場合には、データモデルを修正する必要があります。

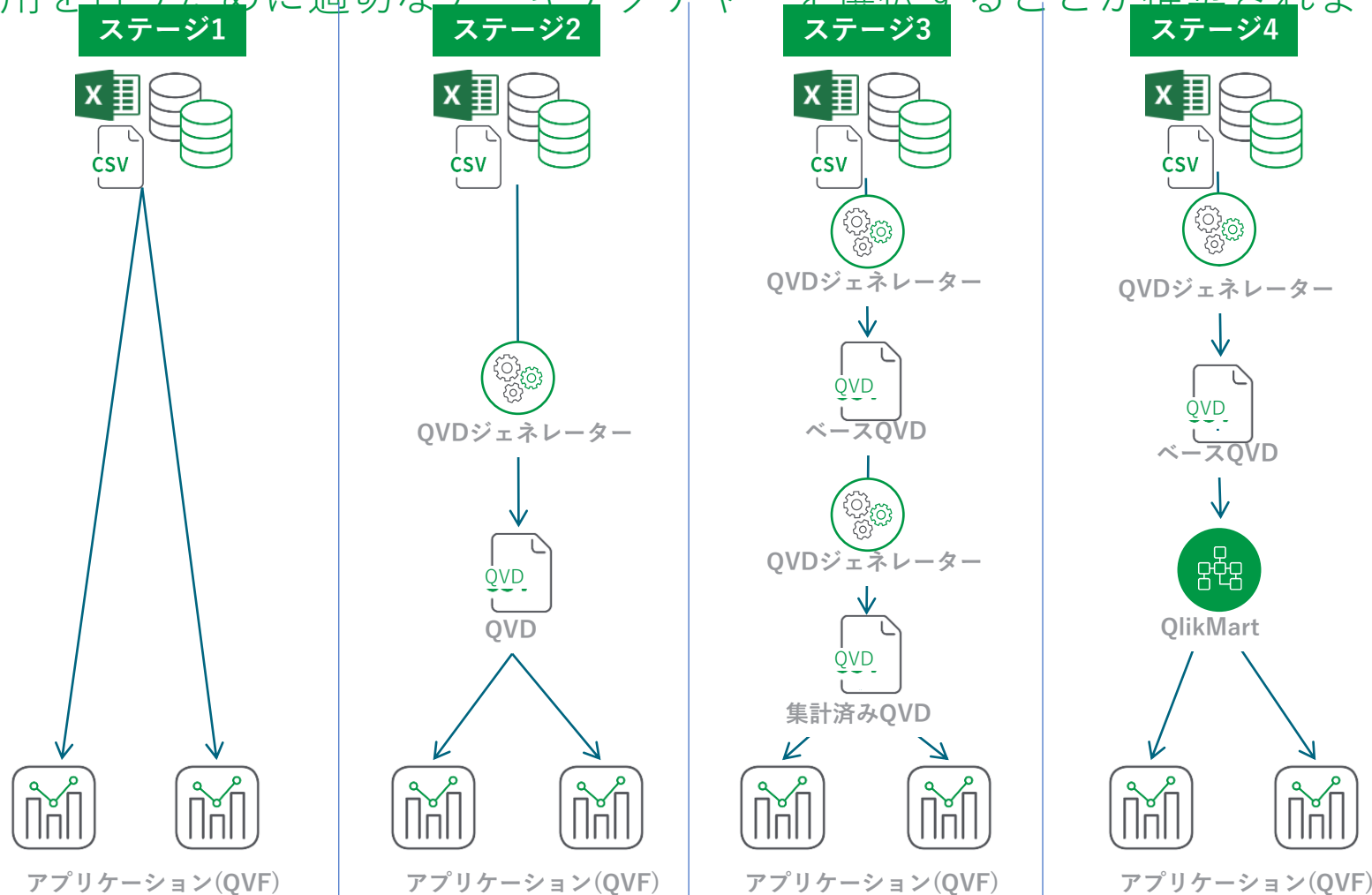


QVDの活用



QVDを活用したデータアーキテクチャ

- QVDを活用することで複数ステップに分けてデータ変換、QVDへの保存、分析での利用を行うことができます。
- より最適な運用を行うために適切なアーキテクチャを選択することが推奨されます。



データアーキテクチャ(1)

① ステージ1 アーキテクチャ

- プレゼンテーション層のアプリケーションがデータソースに対する接続情報を保持し、データロード及び集計処理を実施
- 以下の理由により、開発・運用の効率が一般的に劣る：
 - ✓ データソースに対して複数アプリから同様なクエリが発行され、余計な負荷が発生するケースが生じる
 - ✓ 同様なクエリや変換処理が複数のアプリケーションのスクリプトで実装される
- 最もシンプルな構成のアーキテクチャだが、エンタープライズ環境での実装には一般的に推奨されない

ソース層

DBなどのデータソース



プレゼンテーション層

エンドユーザーにダッシュボード画面を提供するアプリケーション



アプリケーション(QVF)

データアーキテクチャ(2)

② ステージ2アーキテクチャ

- 第2階層にQVDファイルを利用
- プレゼンテーション層のアプリケーションは複数のQVDファイルからデータモデルを作成
- プレゼンテーション層の開発者が直接データソースにアクセスすることは不要で、共通化されたQVDファイルの再利用が可能
- プレゼンテーション層でのデータモデリングやスク립ティングが必要となるが、データソースへのクエリに関する知識は不要
- IT担当とビジネスユーザーの役割分割が可能

ソース層

DBなどのデータソース



抽出層

ソース層のデータソースからデータを抽出しQVDファイルを作成



QVDジェネレーター

QVD層

抽出層での処理により生成されたQVDで、プレゼンテーション層のアプリケーションはこれらのQVDのデータを取り込んで作成



QVD

プレゼンテーション層

エンドユーザーにダッシュボード画面を提供するアプリケーション

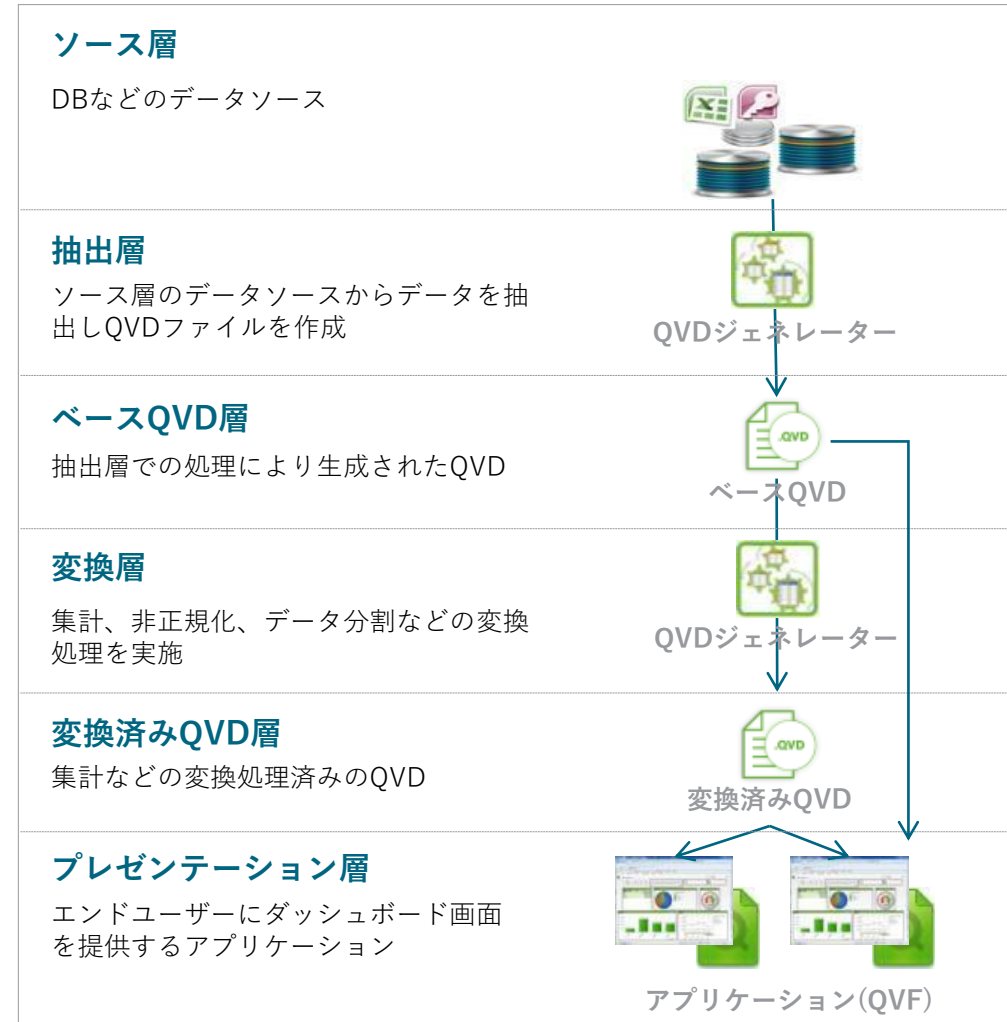


アプリケーション(QVF)

データアーキテクチャ(3)

③ ステージ3アーキテクチャ

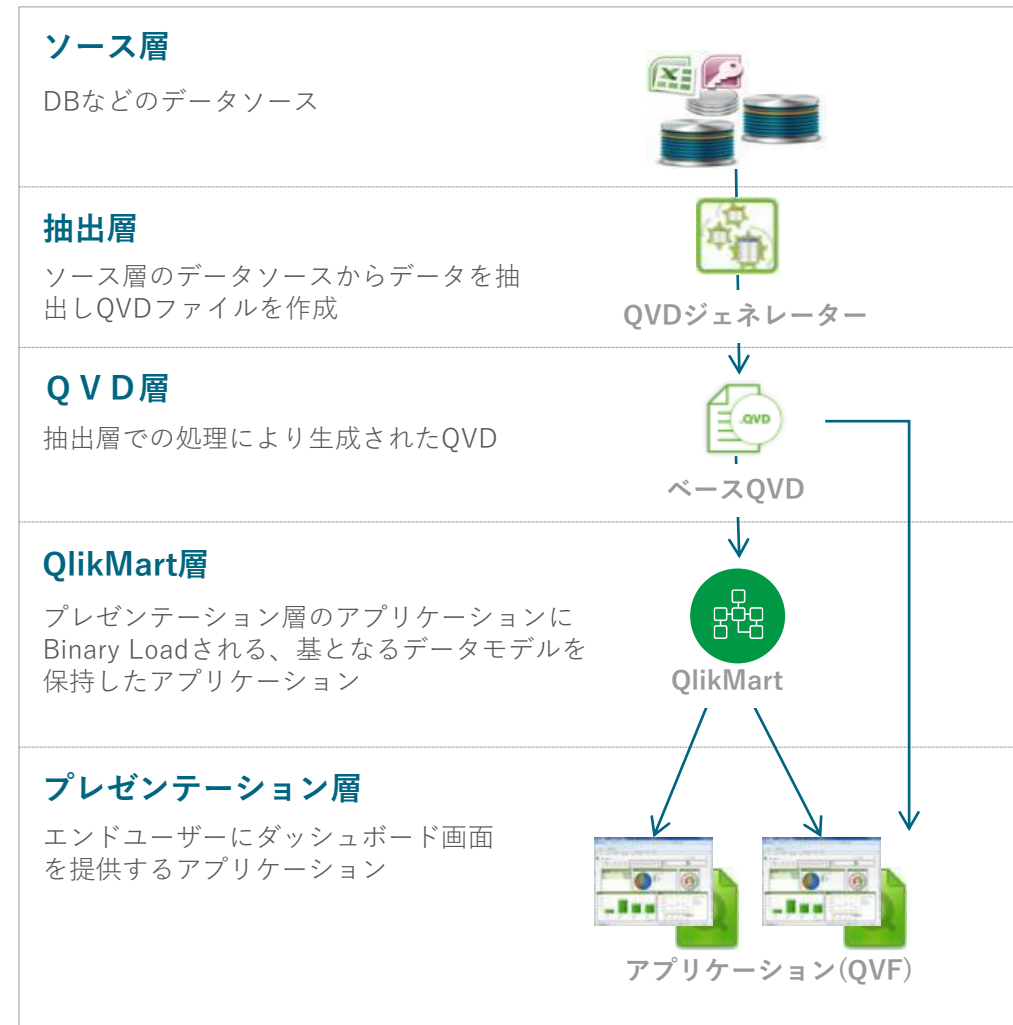
- 2層目と3層目にQVDを配置：
 - ✓ 2層目のQVDはソースシステム上の正規化されたDBテーブル、もしくは非正規化されたDBビューのレプリカ
 - ✓ 3層目のQVDは集計、非正規化、データ分割などの処理済みのデータ
- 変換処理・集計処理をプレゼンテーション層から切り離し、QVDの再利用性をさらに向上
- プレゼンテーション層から変換・集計・パフォーマンス最適化済みデータを利用可能で、より高い再利用性を実現
- IT担当とビジネスユーザーの役割分割が可能



データアーキテクチャ(4)

④ ステージ4アーキテクチャ

- QlikMartのアプリケーションを含むアーキテクチャ。(QlikMartはユーザーインターフェースは含まない、完成されたデータモデルを含むアプリケーション)
- プレゼンテーション層のアプリケーションからQlikMartをbinary loadで取り込み、データモデルを再利用 (binary loadは後述)
- セルフサービスBIを実現するには最適なアーキテクチャ。ダッシュボード間で利用されるデータモデルの一貫性、ダッシュボードのパフォーマンスがIT担当によってより厳密に管理が可能
- IT担当とビジネスユーザーの役割分割が可能 (プレゼンテーション層では最低限のデータロードのみの知識が必要)





Thank You