Wave Analytics バインド開発者ガイド

Salesforce, Winter '17

@salesforcedocs

最終更新日: 2016/09/09

本書の英語版と翻訳版で相違がある場合は英語版を優先するものとします。

目次

Wave ダッシュボードでのバインド 1
選択バインド 2
結果バインド 2
Wave デザイナダッシュボードのバインド 3
Classic デザイナダッシュボードのバインド 54

Wave ダッシュボードでのバインド

バインドを使用すると、ダッシュボードの異なるコンポーネント間のインタラクションが可能になります。ステップを互いにバインドしてインタラクションを制御します。選択バインドと結果バインドの2種類のバインドがあります。1つのステップの選択内容または結果によってダッシュボードの他のステップが更新されます。

Waveダッシュボードデザイナまたは Classic デザイナで作成されたダッシュボードでバインドを設定できます。バインド構文は各デザイナで異なります。より多くのバインドの使用方法を提供する Wave ダッシュボードデザイナを使用することをお勧めします。

ヒント: バインドを作成してウィジェットでインタラクションが行えるようにする前に、ファセットについて考えます。ファセットは、ウィジェット間のインタラクションを指定するための最も簡単かつ一般的な方法です。ファセットを設定すると、1つのウィジェットで選択した内容によって、同じデータセットからのステップを使用して他のすべてのウィジェットが自動的に絞り込まれます。ファセットは設定が容易ですが、制限があります。ファセットで絞り込むことができるのは他のステップのみです。また、ファセットは同じデータセットからのステップに対してのみ機能します。この範囲を超えたインタラクションを作成するには、バインドを使用します。

ステップについての詳細は、「Widget Steps in a Wave Dashboard」を参照してください。ファセットの詳細は、「Making Widgets Interactive Using Facets and Bindings」を参照してください。

選択バインド

選択バインドはインタラクション駆動型です。ユーザがウィジェットの要素を選択するたびに評価されます。たとえば、次のダッシュボードがあるとします。

結果バインド

結果バインドは、ステップの結果に基づいています。これはインタラクション駆動型ではありません。

Wave デザイナダッシュボードのバインド

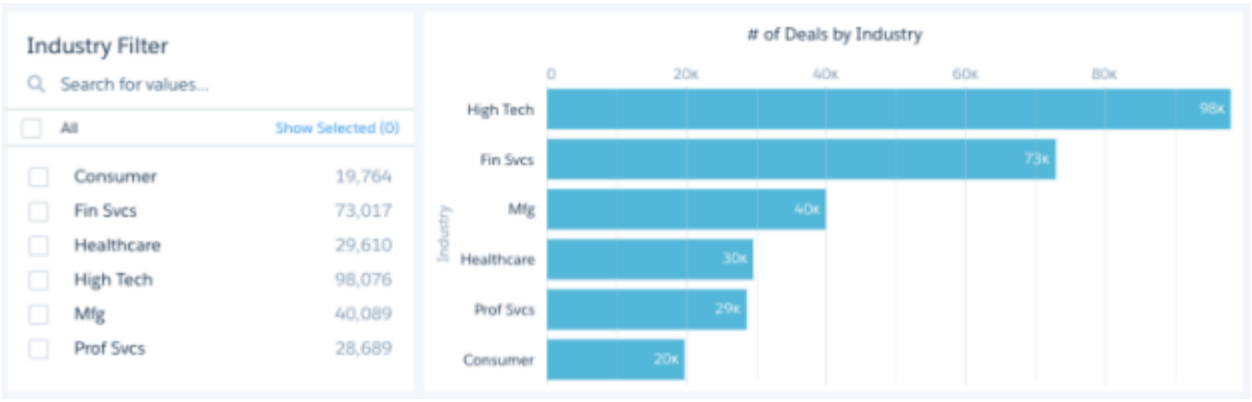
Waveダッシュボードデザイナでは、選択バインドと結果バインドを同様に処理します。どちらのバインド種別も表形式データを処理し、ウィジェットで使用されていない列でも完全な行を返します。Wave ダッシュボードデザイナは複数選択を表形式データとして扱い、単一選択を表形式データの単一行として処理します。デザイナは、表形式データの各行をオブジェクトの配列形式 (各オブジェクトのキーは列名) で表します。

Classic デザイナダッシュボードのバインド

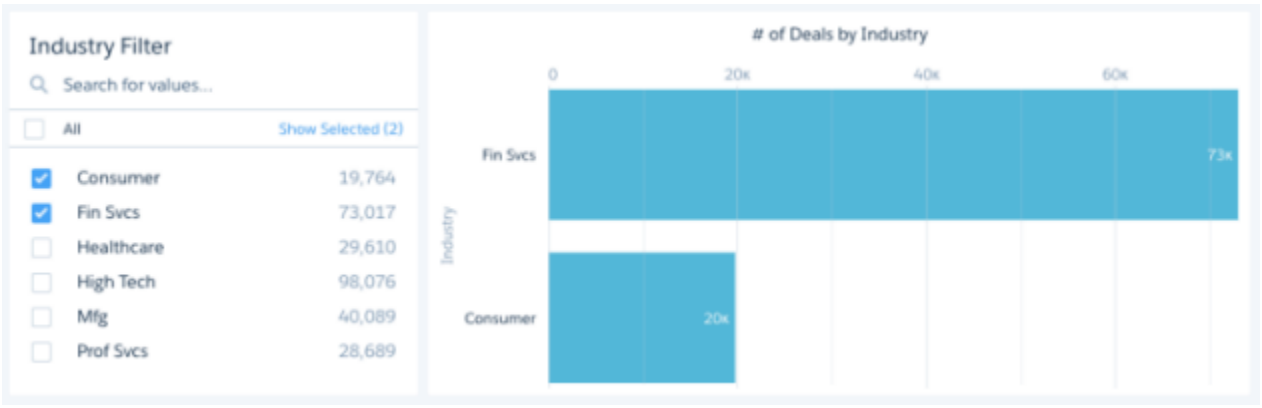
Wave ダッシュボードデザイナとは異なり、Classic デザイナでは選択バインドと結果バインドの扱いが異なります。Classicデザイナの場合、結果バインドでは表形式データが返され、選択バインドでは単一文字列が返されることを想定しています。Classicデザイナでは完全な行が生成されないため、グルーピング列に基づいてのみバインドを作成できます。

選択バインド

選択バインドはインタラクション駆動型です。ユーザがウィジェットの要素を選択するたびに評価されます。たとえば、次のダッシュボードがあるとします。



[Industry Filter (業種の絞り込み)] ウィジェットの選択内容によって、[# of Deals by Industry (業種ごとの商談数)] グラフの結果が絞り込まれます。



選択バインドの用途は次のとおりです。

- 別のデータセットのステップを使用するウィジェット間のインタラクションを指定する。
- 絞り込みに加えて、基準、グルーピング、およびステップクエリのその他の要素を指定する。
- 一部のウィジェットタイプのウィジェット表示プロパティを設定する(数値ウィジェットとグラフウィジェットのみ)。

結果バインド

結果バインドは、ステップの結果に基づいています。これはインタラクション駆動型ではありません。

結果バインドの主な用途は次のとおりです。

- 複雑な計算の中間結果を定義する。たとえば、上位5個の商品の合計商談金額を計算するには、あるステップを使用して上位 5 個の商品を計算します。次のこれらの結果を使用して、各商品の合計商談金額を計算する別のステップを絞り込みます。
- ステップの結果に基づいて動的にウィジェットの表示を変更する。たとえば、基準の値に基づいて異なる色を表示するように数値ウィジェットを設定できます (Wave ダッシュボードデザイナのみ)。

Wave デザイナダッシュボードのバインド

Wave ダッシュボードデザイナでは、選択バインドと結果バインドを同様に処理します。どちらのバインド種別も表形式データを処理し、ウィジェットで使用されていない列でも完全な行を返します。Wave ダッシュボードデザイナは複数選択を表形式データとして扱い、単一選択を表形式データの単一行として処理します。デザイナは、表形式データの各行をオブジェクトの配列形式 (各オブジェクトのキーは列名) で表します。

構文

Wave デザイナダッシュボードでバインドを作成する場合、適切な構文を指定する必要があります。構文は、各ダッシュボードデザイナで異なります。

バインド関数

Wave ダッシュボードデザイナには、ソースステップからデータを取得し、そのデータを操作し、逐次化してターゲットステップで使用できるようにする、さまざまなバインド関数が用意されています。

バインドエラー

Wave デザイナダッシュボードで無効なバインドを作成すると、そのバインドを使用するダッシュボードのウィジェットにエラーが表示されます。

使用事例

関数を使用してバインドを構築する方法をより深く理解できるように、さまざまなバインドの使用事例を見てみましょう。

構文

Wave デザイナダッシュボードでバインドを作成する場合、適切な構文を指定する必要があります。構文は、各ダッシュボードデザイナで異なります。

Wave ダッシュボードデザイナで選択バインドまたは結果バインドを指定するには、次の構文を使用します。

```
<stepName>.<result|selection>
```

次に例を示します。

```
mySourceStep.selection
```

バインド関数

Wave ダッシュボードデザイナには、ソースステップからデータを取得し、そのデータを操作し、逐次化してターゲットステップで使用できるようにする、さまざまなバインド関数が用意されています。

入力データに空の結果がある場合、null 値が返されます。指定されたデータの場所が存在しない場合、エラーが発生します。たとえば、バインドが行 3 からデータを取得するように定義されていて、行 3 が存在しない場合はエラーになります。エラーは、入力データの形式が関数の要件を満たしていない場合にも発生します。

バインド関数は、次のいずれかの形式の入力データで動作します。

- 0、"this is scalar"、null などのスカラー値
- ([1, 2, 3]) または (["one","two","three"]) などの一次元配列
- ([[1, 2], [3, 4]]) などの二次元配列 (配列の配列)

入力データに必要な形式は、関数によって異なります。データ処理後、関数はデータの形式を変更できます。

各バインドは、ネストされた関数で構成されます。各バインドには、1 つのデータ選択関数と 1 つのデータ逐次化関数が必要です。必要に応じて、バインドに複数のデータ操作関数を含めることができます(次のセクションでは、これらの関数について説明します)。次の例は、バインドのネストされた関数が連携して、定義されているターゲットステップで必要な結果を返す方法を示しています。

`mySourceStep` ステップには次の入力データがあります。

表示	グループ化
Regional Area	region
Country	country
State	state

厳密に言えば、このステップのデータは二次元配列として保存され、各行はキー - 値ペアの対応付けとして保存されます。

```
[
    ["display":"Regional Area", "grouping":"region"],
    ["display":"Country", "grouping":"country"],
    ["display":"State", "grouping":"state"]
]
```

実行時に、Wave はバインド関数を最も内側の関数から評価します。このロジックに従い、Wave はこの例のバインド関数を次の順序で評価します。

順序	関数	説明
1	`mySourceStep.selection`	Wave は、選択されたデータの各行をキー - 値ペアの対応付けとして返します。1 つの選択が行われた場合、1 つの行のみが返されます。複数の選択が行われた場合、複数の行が返されます。 `[ ["display":"Regional Area", "grouping":"region"], ["display":"Country", "grouping":"country"],`

順序	関数	説明
		`["display":"State", "grouping":"state"]]`
2	`cell(mySourceStep.selection, 0, \"grouping\")`	`cell` 関数は、`mySourceStep.selection` によって返される最初の行 (0 インデックスで示される) の `"grouping"` 列からスカラー値を返します。選択に基づき、戻り値は `"region"`、`"country"`、`"state"` のいずれかになります。選択が行われなかった場合、`null` が返されます。
3	`coalesce(cell(mySourceStep.selection, 0, \"grouping\"), \"state\")`	`coalesce` 関数は、選択が行われた場合は `cell` 関数の値を返します。選択が行われなかった場合、`coalesce` 関数は指定されたデフォルトの `"state"` 値を返します。 ヒント: 通常、`coalesce` 関数は null 値でエラーが発生することを防ぐために使用されます。
4	`coalesce(cell(mySourceStep.selection, 0, \"grouping\"), \"state\").asString()`	`.asString` 関数は、`coalesce` 関数の結果を SAQL 文字列として返します。

これらの関数がバインドで使用される方法については、「使用事例」を参照してください。

データ選択関数

データ選択関数は、ソースからデータを選択します。ソースには、ステップの選択または結果を使用できます。この関数はデータのテーブルを返します。各列には名前が含まれ、各行には 0 から始まるインデックスが含まれます。そのテーブルから、1 つ以上の行、1 つ以上の列、またはバインドに含めるセルを選択できます。

データ操作関数

データ操作関数は、データ逐次化関数 (次のセクションを参照) で必要な形式にデータを変更します。操作関数は、データ選択または他のデータ操作関数の結果に適用できます。入力データが null の場合、特に指定がなければ操作関数は null を返します。

データ逐次化関数

逐次化関数は、バインドが挿入されるステップで必要な形式にデータを変換します。たとえば、バインドがコンパクトフォームステップで使用される場合、`asObject()` 関数を使用してデータを一次元オブジェクトの形式に変換します。

データ選択関数

データ選択関数は、ソースからデータを選択します。ソースには、ステップの選択または結果を使用できます。この関数はデータのテーブルを返します。各列には名前が含まれ、各行には0から始まるインデックスが含まれます。そのテーブルから、1つ以上の行、1つ以上の列、またはバインドに含めるセルを選択できます。

データの複数の行または列を選択した場合、二次元配列が返されます。1 つの行または列を選択した場合、一次元配列が返されます。セルを選択した場合、スカラー値が返されます。ソースが空の場合、null が返されます。存在しないデータを選択しようとした場合、バインドエラーが発生します。たとえば、テーブルには2行のみあり、3 行目のデータを選択しようとするとエラーになります。

cell 関数
データの 1 つのセルを "This salesperson rocks"、2、`null` などのスカラーとして返します。`rowIndex` が整数でない場合、`columnName` が文字列でない場合、またはセルがテーブル内に存在しない場合、エラーが発生します。

column 関数
データの 1 つの列 (一次元配列として) またはデータの複数の列 (二次元配列として) を返します。

row 関数
1行のデータを一次元配列として、複数行のデータを二次元配列として返します。選択バインドの場合、一般的にこの関数を使用して最初の行またはすべての行を返します。結果バインドの場合、特定の行が必要なことがあります。行インデックスを決定するには、値テーブルのステップ結果を表示します。

cell 関数

データの 1 つのセルを "This salesperson rocks"、2、`null` などのスカラーとして返します。`rowIndex` が整数でない場合、`columnName` が文字列でない場合、またはセルがテーブル内に存在しない場合、エラーが発生します。

構文

```
cell(source), rowIndex, columnName)
```

引数

引数	説明
source	(必須) ステップの名前と `selection` または `result` を指定します。
rowIndex	(必須) 行をそのインデックスで指定します。行インデックスは 0 から始まります。
columnName	(必須) 列名を指定します。

次の例は、myStep ソースステップに基づいています。myStep.selection がステップから次の行を取得すると仮定します。

```
[
    {stateName: 'CA', Amount:100},
    {stateName: 'TX', Amount:200},
    {stateName: 'OR'', Amount:300},
    {stateName: 'AL', Amount:400},
]
```

Wave ではこのデータはテーブルとして保存されませんが、この例をわかりやすくするためにデータをテーブル形式で次に示します。

(行インデックス)	stateName	Amount (金額)
0	CA	100
1	TX	200
2	OR	300
3	AL	400

例:

```
cell(myStep.selection, 1, "stateName")
```

出力:

```
"TX"
```

column 関数

データの 1 つの列 (一次元配列として) またはデータの複数の列 (二次元配列として) を返します。

構文

```
column(source), [columnNames...])
```

引数

引数	説明
source	(必須) ステップの名前と selection または result を指定します。
columnNames	(必須)列名の配列を指定します。リストされた列の順序は、出力で表示される列の順序に影響します。逐次化関数では順序が重要になります。たとえば asOrder

引数	説明
	関数では、最初の要素は項目名、2番目の要素は方向である必要があります。

次の例は、myStep ソースステップに基づいています。myStep.selection がステップから次の行を取得すると仮定します。

```
[
    {stateName: 'CA', Amount:100},
    {stateName: 'TX', Amount:200},
    {stateName: 'OR'', Amount:300},
    {stateName: 'AL', Amount:400},
]
```

Wave ではこのデータはテーブルとして保存されませんが、この例をわかりやすくするためにデータをテーブル形式で次に示します。

(行インデックス)	stateName	Amount (金額)
0	CA	100
1	TX	200
2	OR	300
3	AL	400

例:

```
column(myStep.selection, ["stateName"])
```

出力:

```
["CA", "TX", "OR", "AL"]
```

例:

```
column(myStep.selection, [])
```

出力:

```
[ ["CA", "TX", "OR", "AL"], ["100", "200″," "300", "400"] ]
```

row 関数

1行のデータを一次元配列として、複数行のデータを二次元配列として返します。選択バインドの場合、一般的にこの関数を使用して最初の行またはすべての行を返します。結果バインドの場合、特定の行が必要なことがあります。行インデックスを決定するには、値テーブルのステップ結果を表示します。

構文

```
row(source), [rowIndices...], [columnNames...])
```

引数

引数	説明
source	(必須) ステップの名前と `selection` または `result` を指定します。
rowIndices	(省略可能) インデックスを使用して 1 つ以上の行を指定します。行インデックスは0から始まります。すべての行を含めるには、`rowIndices` に空の配列を指定します。
columnNames	(省略可能)選択して並び替える列名の配列を指定します。指定しない場合は、すべての列が選択され、すべての行の列の順序が同じになります。ただし、別のクエリで同じ順序になるとは限りません。

次の例は、`myStep` ソースステップに基づいています。`myStep.selection` がステップから次の行を取得すると仮定します。

```
[
    {stateName: 'CA', Amount:100},
    {stateName: 'TX', Amount:200},
    {stateName: 'OR'', Amount:300},
    {stateName: 'AL', Amount:400},
]
```

Wave ではこのデータはテーブルとして保存されませんが、この例をわかりやすくするためにデータをテーブル形式で次に示します。

(行インデックス)	stateName	Amount (金額)
0	CA	100
1	TX	200
2	OR	300
3	AL	400

例:

```
row(myStep.selection, [0], ["Amount"])
```

出力:

```
["100"]
```

例:

```
row(myStep.selection, [0,2], [])
```

出力:

```
[ ["CA", "100"], ["OR", "300"] ]
```

例:

```
row(myStep.selection [], ["stateName"])
```

出力:

```
[ ["CA"], ["TX"], ["OR"], ["AL"] ]
```

例:

```
row(myStep.selection [0,2], ["stateName", "Amount"])
```

出力:

```
[ ["CA", "100"], ["OR", "300"] ]
```

データ操作関数

データ操作関数は、データ逐次化関数(次のセクションを参照)で必要な形式にデータを変更します。操作関数は、データ選択または他のデータ操作関数の結果に適用できます。入力データがnullの場合、特に指定がなければ操作関数はnullを返します。

メモ: データ操作が不要な場合、データ選択関数の結果にデータ逐次化関数を追加します。

coalesce 関数

ソースのリストから最初のnull以外のソースを返します。関数がnull値を返す場合のデフォルト値を指定するために役立ちます。

concat 関数

複数のソースからのストリームを一次元または二次元配列に結合します。nullのソースはスキップされます。

flatten 関数

二次元配列を一次元配列にフラット化します。

join 関数

指定したトークンを使用して要素を結合し、一次元配列または二次元配列を文字列に変換します。データが他の形式の場合、エラーが発生します。

slice 関数
開始位置と終了位置 (終了位置は省略可能) を指定して一次元配列から 1 つ以上の値を選択すると、一次元配列が返されます。開始値が終了値よりも大きい場合、エラーが発生します。負のインデックスがサポートされています。

valueAt 関数
指定されたインデックスの 1 つのスカラー値を返します。

coalesce 関数

ソースのリストから最初の null 以外のソースを返します。関数が null 値を返す場合のデフォルト値を指定するために役立ちます。

構文

```
coalesce(source1, source2,...)
```

引数

引数	説明
source	(必須) ソースにはデータ選択または他のデータ操作関数の結果を使用できます。ソースの形式に制限はありません。

例:

```
coalesce(cell(step1.selection, 0, "column1"), "green")
```

出力: `cell(step1.selection, 0, "column1")` によって返された結果が出力されます。ただし、`cell(step1.selection, 0, "column1")` が `null` を返した場合、出力は次のようになります。

```
"green"
```

バインド使用事例でのこの関数の適用については、「切り替えウィジェットに基づく地図タイプの変更」を参照してください。

concat 関数

複数のソースからのストリームを一次元または二次元配列に結合します。null のソースはスキップされます。

構文

```
concat(source1, source2,...)
```

引数

引数	説明
source	(必須)各ソースにはデータ選択または他のデータ操作関数の結果を使用できます。各ソースは、一次元または二次元配列である必要があります。形式が異なるソースのデータを連結しようとすると、エラーが発生します。たとえば、関数 `concat(["a", "b"], ["c", "d", "e"])` はエラーになります。

例:

```
concat(["a", "b"], ["c", "d"])
```

出力:

```
["a", "b", "c", "d"]
```

例:

```
concat([["a", "b"]], [["c", "d"]])
```

出力:

```
[["a", "b"], ["c", "d"]]
```

flatten 関数

二次元配列を一次元配列にフラット化します。

構文

```
flatten(source)
```

引数

引数	説明
source	(必須)ソースにはデータ選択または他のデータ操作関数の結果を使用できます。source は二次元配列にする必要があります。それ以外の場合は一次元配列またはスカラーをフラット化する理由がないため、エラーが発生します。

例:

```
flatten([["CDG", "SAN"], ["BLR", "HND"], ["SMF", "JFK"])
```

出力:

```
["CDG", "SAN", "BLR", "HND", "SMF", "JFK"]
```

join 関数

指定したトークンを使用して要素を結合し、一次元配列または二次元配列を文字列に変換します。データが他の形式の場合、エラーが発生します。

構文

```
join(source, token)
```

引数

引数	説明
source	(必須) ソースにはデータ選択または他のデータ操作関数の結果を使用できます。source は 2次元配列にする必要があります。それ以外の場合はエラーが発生します。
トークン	(必須) + や , などの任意の文字列値。

例:

```
join(["a", "b", "c"], "+")
```

出力:

```
["a+b+c"]
```

例:

```
join([["a", "b", "c"], [1, 2]], "~")
```

出力:

```
["a~b~c~1~2"]
```

slice 関数

開始位置と終了位置 (終了位置は省略可能) を指定して一次元配列から 1 つ以上の値を選択すると、一次元配列が返されます。開始値が終了値よりも大きい場合、エラーが発生します。負のインデックスがサポートされています。

構文

```
slice(source, start, end)
```

引数

引数	説明
source	(必須) ソースにはデータ選択または他のデータ操作関数の結果を使用できます。ソースの形式に制限はありません。
start	(必須) 配列の開始値を識別するインデックス。たとえば、0 は配列の最初の要素を表します。
end	(省略可能) 配列の終了値を識別するインデックス。

例:

```
slice(step.selection, -1, 0)
```

選択した最後の行を返します。

valueAt 関数

指定されたインデックスの 1 つのスカラー値を返します。

構文

```
valueAt(source, index)
```

引数

引数	説明
source	(必須) ソースにはデータ選択または他のデータ操作関数の結果を使用できます。ソースの形式に制限はありません。

引数	説明
index	(必須)負のインデックスがサポートされています。存在しないインデックスを指定すると、この関数は null を返します。

例:

```
valueAt(step.selection, -1)
```

最後に選択した値を返します。

データ逐次化関数

逐次化関数は、バインドが挿入されるステップで必要な形式にデータを変換します。たとえば、バインドがコンパクトフォームステップで使用される場合、`asObject()` 関数を使用してデータを一次元オブジェクトの形式に変換します。

asDateRange() 関数

日付範囲検索条件を SAQL クエリの文字列として返します。日付範囲は包括的です。日付に基づく検索条件の一部として文字列を使用します。

asEquality() 関数

等価または「in」検索条件を SAQL クエリの文字列として返します。入力データは、スカラー、一次元または二次元配列である必要があります。

asGrouping() 関数

グルーピングを SAQL クエリの文字列として返します。

asObject() 関数

逐次化なしでデータを渡します。データをオブジェクト (文字列の配列) として返します。

asOrder() 関数

並び替え順を SAQL クエリの文字列として返します。

asProjection() 関数

クエリ式と別名をステップでの項目の表示に使用できる文字列として返します。クエリ式により、項目の値が決まります。別名は項目表示ラベルになります。

asRange() 関数

範囲検索条件を SAQL クエリの文字列として返します。範囲は包括的です。

asString() 関数

スカラー、一次元配列、または二次元配列を文字列として逐次化します。文字列内の二重引用符はエスケープします。

asDateRange() 関数

日付範囲検索条件を SAQL クエリの文字列として返します。日付範囲は包括的です。日付に基づく検索条件の一部として文字列を使用します。

入力データは、一次元または二次元配列である必要があります。入力データが2つの要素を含む一次元配列の場合、この関数は最初の要素を最小、2 番目の要素を最大として使用します。null の場合は `fieldName in all` になり、検索条件は適用されません。

構文

```
<input data>.asDateRange(fieldName)
```

引数

引数	説明
fieldName	(必須) 日付項目の名前。

次の例は、`stepFoo` ソースステップに基づいています。`stepFoo.selection` がステップから次の行を取得すると仮定します。

```
[
    {min: 1016504910000, max: 1281655993000}
]
```

例:

```
row(stepFoo.selection, [0], ["min", "max"]).asDateRange("date(year, month, day)")
```

出力:

```
date(year, month, day) in [dateRange([2002,3,19], [2010,8,12])]
```

「日付範囲検索条件」も参照してください。

asEquality() 関数

等価または「in」検索条件をSAQLクエリの文字列として返します。入力データは、スカラー、一次元または二次元配列である必要があります。

1 つの項目名が指定された場合、返される文字列には一次元配列の `in` 演算子 (`fieldName in ["foo", "bar"]`) またはスカラーの等価演算子 (`fieldName == "foo"`) が含まれます。

複数の項目名が指定された場合、返される文字列には複合検索条件が含まれます。この場合、二次元配列が使用されます。各配列内の値の数は、指定された項目の数と一致する必要があります。

この関数への入力が null の場合、`<fieldName> by all` が返されます。この `<fieldName>` は最初の項目です。たとえば、`cell(step1.selection, 0, "column1")` が `null` と評価された場合、

`cell(step1.selection, 0, "column1").asEquality("field1")` は `'field1' by all` と評価され、検索条件は適用されません。

構文

```
<input data>.asEquality(fieldName)
```

引数

引数	説明
fieldName	(必須) 項目の名前。

次の例は、`myStep` ソースステップに基づいています。`myStep.selection` がステップから次の行を取得すると仮定します。

```
[
    {grouping: "first", measure: 19}
    {grouping: "second", measure: 32}
]
```

例:

```
cell(myStep.selection, 1, "measure").asEquality("bar")
```

出力:

```
bar == 32
```

例:

```
column(myStep.selection, ["grouping"]).asEquality("bar")
```

出力:

```
bar in ["first","second"]
```

「等価検索条件」も参照してください。

asGrouping() 関数

グルーピングを SAQL クエリの文字列として返します。

入力データは、グルーピングのスカラーまたは一次元配列である必要があります。null の場合は `group by all` になります。

構文

```
<input data>.asGrouping()
```

次の例は、stepFoo ソースステップに基づいています。stepFoo.selection がステップから次の行を取得すると仮定します。

```
[
    {grouping: "first", alias: "foo"}
    {grouping: "second", alias: "bar"}
]
```

例:

```
cell(stepFoo.selection, 1, "grouping").asGrouping()
```

出力:

```
'second'
```

例:

```
column(stepFoo.selection, ["grouping"]).asGrouping()
```

出力:

```
('first', 'second')
```

「グループのバインド」も参照してください。

asObject() 関数

逐次化なしでデータを渡します。データをオブジェクト (文字列の配列) として返します。

構文

```
<input data>.asObject()
```

例:

```
column(StaticMeasureNamesStep.selection, [\"value\"]).asObject()
```

バインド使用事例でのこの関数の適用については、「ステップクエリの一部のバインド」を参照してください。

例:

```
cell(static_1.selection, 0, \"value\").asObject()
```

asOrder() 関数

並び替え順を SAQL クエリの文字列として返します。

入力データは、スカラー、一次元または二次元配列である必要があります。二次元配列は、バインドのタプルとして処理されます。

構文

```
<input data>.asOrder()
```

次の例は、stepFoo ソースステップに基づいています。stepFoo.selection がステップから次の行を取得すると仮定します。

```
[
    {order: "first", direction: "desc"}
    {order: "second", direction: "asc"}
]
```

例:

```
cell(stepFoo.selection, 1, "order").asOrder()
```

出力:

```
'second'
```

例:

```
column(stepFoo.selection, ["order"]).asOrder()
```

出力:

```
('first', 'second')
```

例:

```
row(stepFoo.selection, [], ["order", "direction"]).asOrder()
```

出力:

```
('first' desc, 'second' asc)
```

「順序のバインド」も参照してください。

asProjection() 関数

クエリ式と別名をステップでの項目の表示に使用できる文字列として返します。クエリ式により、項目の値が決まります。別名は項目表示ラベルになります。

クエリ式と別名をステップでの項目の表示に使用できる文字列として返します。クエリ式により、項目の値が決まります。別名は項目表示ラベルになります。

構文

```
<input data>.asProjection()
```

次の例は、stepFoo ソースステップに基づいています。stepFoo.selection がステップから次の行を取得すると仮定します。

```
[
    {expression: "first", alias: "foo"}
    {expression: "second", alias: "bar"}
]
```

例:

```
stepFoo.selection, [0], ["expression", "alias"]).asProjection()
```

出力:

```
first as 'foo'
```

例:

```
stepFoo.selection, [], ["expression", "alias"]).asProjection()
```

出力:

```
first as 'foo', second as 'bar'
```

「射影のバインド」も参照してください。

asRange() 関数

範囲検索条件を SAQL クエリの文字列として返します。範囲は包括的です。

入力データは、少なくとも 2 つの要素を含む一次元配列である必要があります。この関数は、最初の要素を最小、2 番目の要素を最大として使用します。null の場合は fieldName by all になり、検索条件は適用されません。

構文

```
<input data>.asRange(fieldName)
```

引数

引数	説明
fieldName	(必須) 項目の名前。

次の例は、myStep ソースステップに基づいています。myStep.selection がステップから次の行を取得すると仮定します。

```
[
    {grouping: "first", measure: 19}
    {grouping: "second", measure: 32}
]
```

例:

```
row(myStep.selection, [0], ["min", "max"]).asRange("bar")
```

出力:

```
bar >= 19 && bar <= 32
```

「範囲検索条件」も参照してください。

asString() 関数

スカラー、一次元配列、または二次元配列を文字列として逐次化します。文字列内の二重引用符はエスケープします。

構文

```
<input data>.asString()
```

例:

```
cell(stepOpportunity.selection, 1, "measure").asString()
```

例:

```
cell(color_1.result, 0, "color").asString()
```

バインド使用事例でのこの関数の適用については、「色分けを使用した値の強調表示」を参照してください。

バインドエラー

Waveデザイナダッシュボードで無効なバインドを作成すると、そのバインドを使用するダッシュボードのウィジェットにエラーが表示されます。

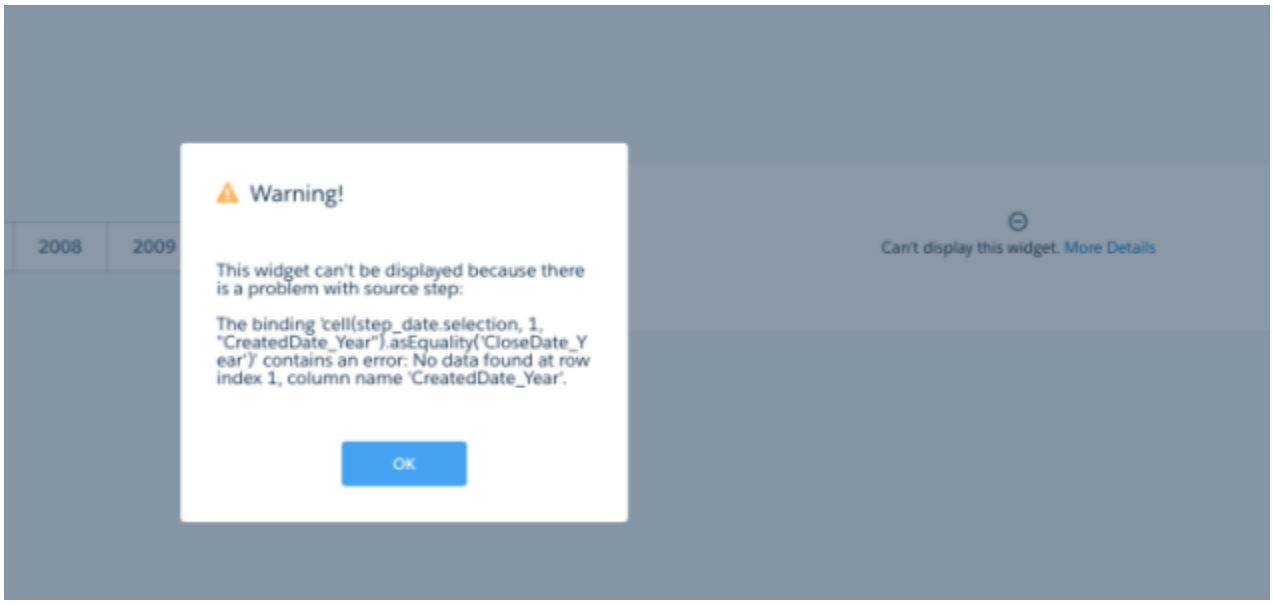
通常、エラーには次の 2 種類があります。

検証エラー
: このエラーは、バインドで誤った構文または無効な引数が使用されているためWaveがバインドを解析できない場合に発生します。別のよくある問題は、他の二重引用符の内側にある二重引用符をエスケープしていないことです。たとえば、次の例では内側の二重引用符セットがエスケープされています。

```
"numberColor": "{{cell(color_1.result, 0, \"color\").asString()}}"
```

実行エラー
: このエラーは、Waveがバインドを実行し、必要な列または行が見つからない場合、またはデータの形式が正しくない場合に発生します。たとえば、バインドでセルが必要なときに行を受け取った場合などがあります。

エラーメッセージで、バインドの問題の解決方法を確認してください。ダッシュボードでのバインドエラーの例を次に示します。



使用事例

関数を使用してバインドを構築する方法をより深く理解できるように、さまざまなバインドの使用事例を見てみましょう。

ステップクエリの一部のバインド

別のステップの選択内容または結果に基づいて、ステップクエリの一部を動的に設定できます。たとえば、グラフで選択したグループ化に基づいて、ステップ内のグループ化を設定できます。

異なるデータセットのステップのバインド

異なるデータセットのステップをバインドできます。たとえば、次のダッシュボードにはそれぞれ独自のデータセットに基づいている 2 つのグラフが含まれています。

静的ステップと他のステップのバインド

クエリから値を取得する代わりに、静的ステップを作成してステップの独自の値を指定できます。たとえば、切り替えウィジェットに [Top 5 Customer (上位 5 位 (顧客))] と [Bottom 5 Customers (下位 5 位 (顧客))] を表示する静的ステップを作成できます。静的ステップを作成したら、ダッシュボードの他のウィジェットとやりとりするために、静的ステップを他のウィジェットのステップに手動でバインドします。

ウィジェットプロパティのバインド

Wave デザイナダッシュボードでは、多くのグラフウィジェットのプロパティを動的に変更するバインドを実装できます。

ステップクエリの一部のバインド

別のステップの選択内容または結果に基づいて、ステップクエリの一部を動的に設定できます。たとえば、グラフで選択したグループ化に基づいて、ステップ内のグループ化を設定できます。

クエリのさまざまな部分をバインドする方法を説明する前に、クエリのさまざまな部分のバインドがどのようなものであるかを示す包括的な例を見てみましょう。次の例では、グラフがグループ化、基準、検索条件、順序、および制限の選択内容に基づいてバインドされます。いずれかの切り替えウィジェットでデータを選択すると、グラフが変化し、変更されたクエリの結果が視覚化されます。

次に、このダッシュボードを強化するステップのJSONを示します。Account_BillingCount_1ステップがグラフウィジェットの基盤となるステップです。このステップには、他のステップに基づく複数のバインドが含まれます。

```
"steps": {
            "Account_BillingCount_1": {
                "datasets": [
                    {
                        "id": "0FbB00000000oEkKAI",
                        "label": "Opportunities",
                        "name": "opportunity",
                        "url": "/services/data/v38.0/wave/datasets/0FbB00000000oEkKAI"
                    }
                ],
                "isFacet": true,
                "isGlobal": false,
                "query": {
                    "measures": "{{column(StaticMeasureNames.selection,
[\"value\"]).asObject()}}",
                  "limit": "{{column(StaticLimits.selection, [\"value\"]).asObject()}}",

                   "groups": "{{column(StaticGroupingNames.selection,
[\"value\"]).asObject()}}",
                "filters": "{{column(StaticFilters.selection, [\"value\"]).asObject()}}",

                 "order": "{{column(StaticOrdering.selection, [\"value\"]).asObject()}}"
```

```
        },
        "selectMode": "single",
        "type": "aggregateflex",
        "useGlobal": true,
        "visualizationParameters": {
            "visualizationType": "hbar",
            "options": {}
        }
    },
    "StaticGroupingNames": {
        "datasets": [],
        "dimensions": [],
        "isFacet": true,
        "isGlobal": false,
        "selectMode": "single",
        "start": {
            "display": [
                "Country"
            ]
        },
        "type": "staticflex",
        "useGlobal": true,
        "values": [
            {
                "display": "Country",
                "value": "Account.BillingCountry"
            },
            {
                "display": "Industry",
                "value": "Account.Industry"
            },
            {
                "display": "Product",
                "value": "Product.Product.Family"
            },
            {
                "display": "Source",
                "value": "Account.AccountSource"
            }
        ],
        "visualizationParameters": {
            "options": {}
        }
    },
    "StaticFilters": {
        "datasets": [],
        "dimensions": [],
        "isFacet": true,
        "isGlobal": false,
        "selectMode": "single",
        "start": {
            "display": "Ads Only"
        },
```

```
        "type": "staticflex",
        "useGlobal": true,
        "values": [
            {
                "display": "Ads Only",
                "value": [
                    "LeadSource",
                    [
                        "Advertisement"
                    ],
                    "in"
                ]
            },
            {
                "display": "Partners Only",
                "value": [
                    "Account.Type",
                    [
                        "Partner"
                    ],
                    "in"
                ]
            },
            {
                "display": "$1M+ Only",
                "value": [
                    "Amount",
                    [
                        [
                            1000000,
                            11921896
                        ]
                    ],
                    ">=<="
                ]
            }
        ],
        "visualizationParameters": {
            "options": {}
        }
    },
    "StaticOrdering": {
        "datasets": [],
        "dimensions": [],
        "isFacet": true,
        "isGlobal": false,
        "selectMode": "single",
        "start": {
            "display": "Ads Only"
        },
        "type": "staticflex",
        "useGlobal": true,
        "values": [
            {
```

```
                "display": "Ascending",
                "value": [
                    -1,
                    {
                        "ascending": true
                    }
                ]
            },
            {
                "display": "Descending",
                "value": [
                    -1,
                    {
                        "ascending": false
                    }
                ]
            }
        ],
        "visualizationParameters": {
            "options": {}
        }
    },
    "StaticLimits": {
        "datasets": [],
        "dimensions": [],
        "isFacet": true,
        "isGlobal": false,
        "selectMode": "single",
        "start": {
            "display": [
                "5"
            ]
        },
        "type": "staticflex",
        "useGlobal": true,
        "values": [
            {
                "display": "5",
                "value": 5
            },
            {
                "display": "10",
                "value": 10
            },
            {
                "display": "25",
                "value": 25
            }
        ],
        "visualizationParameters": {
            "options": {}
        }
    },
    "StaticMeasureNames": {
```

```
        "datasets": [],
        "dimensions": [],
        "isFacet": true,
        "isGlobal": false,
        "selectMode": "singlerequired",
        "start": {
            "display": [
                "Total Amount"
            ]
        },
        "type": "staticflex",
        "useGlobal": true,
        "values": [
            {
                "display": "Max Employees",
                "value": [
                    "max",
                    "Account.NumberOfEmployees"
                ]
            },
            {
                "display": "Total Amount",
                "value": [
                    "sum",
                    "Amount"
                ]
            },
            {
                "display": "Avg Amount",
                "value": [
                    "avg",
                    "Amount"
                ]
            }
        ],
        "visualizationParameters": {
            "options": {}
        }
    },
    "Account_AccountSourc_1": {
        "datasets": [
            {
                "id": "0FbB00000000oEkKAI",
                "label": "Opportunities",
                "name": "opportunity",
                "url": "/services/data/v38.0/wave/datasets/0FbB00000000oEkKAI"
            }
        ],
        "isFacet": true,
        "isGlobal": false,
        "query": {
            "measures": [
                [
                    "count",
```

```
                        "*"
                    ]
                ],
                "groups": [
                    "Account.AccountSource"
                ],
                "order": [
                    [
                        -1,
                        {
                            "ascending": false
                        }
                    ]
                ]
            },
            "type": "aggregateflex",
            "useGlobal": true,
            "visualizationParameters": {
                "visualizationType": "hbar",
                "options": {}
            }
        }
    },
    "widgetStyle": {
        "backgroundColor": "#FFFFFF",
        "borderColor": "#E6ECF2",
        "borderEdges": [],
        "borderRadius": 0,
        "borderWidth": 1
    }
```

基準のバインド

ウィジェットに表示する基準を選択できます。たとえば、切り替えウィジェットの選択内容に基づいて、ダッシュボードにさまざまな基準を表示できます。静的ステップ内の選択内容に基づいて、ステップクエリ内の基準を決定します。

条件のバインド

SAQL クエリでさまざまな条件種別を作成できます。次のセクションでは、さまざまなバインド種別を使用する条件の例について説明します。

射影のバインド

`asProjection()` 逐次化関数を使用して、SAQL クエリで項目の射影を指定します。

グループのバインド

`asGrouping()` 逐次化関数を使用して、SAQL クエリまたはコンパクトフォームクエリでグループ化をバインドします。

順序のバインド

`asOrder()` 逐次化関数を使用して、SAQL クエリの並び替え順序を指定します。

制限およびオフセットのバインド

SAQL クエリの制限とオフセットもバインドできます。このバインドにデータ逐次化関数は必要ありません。

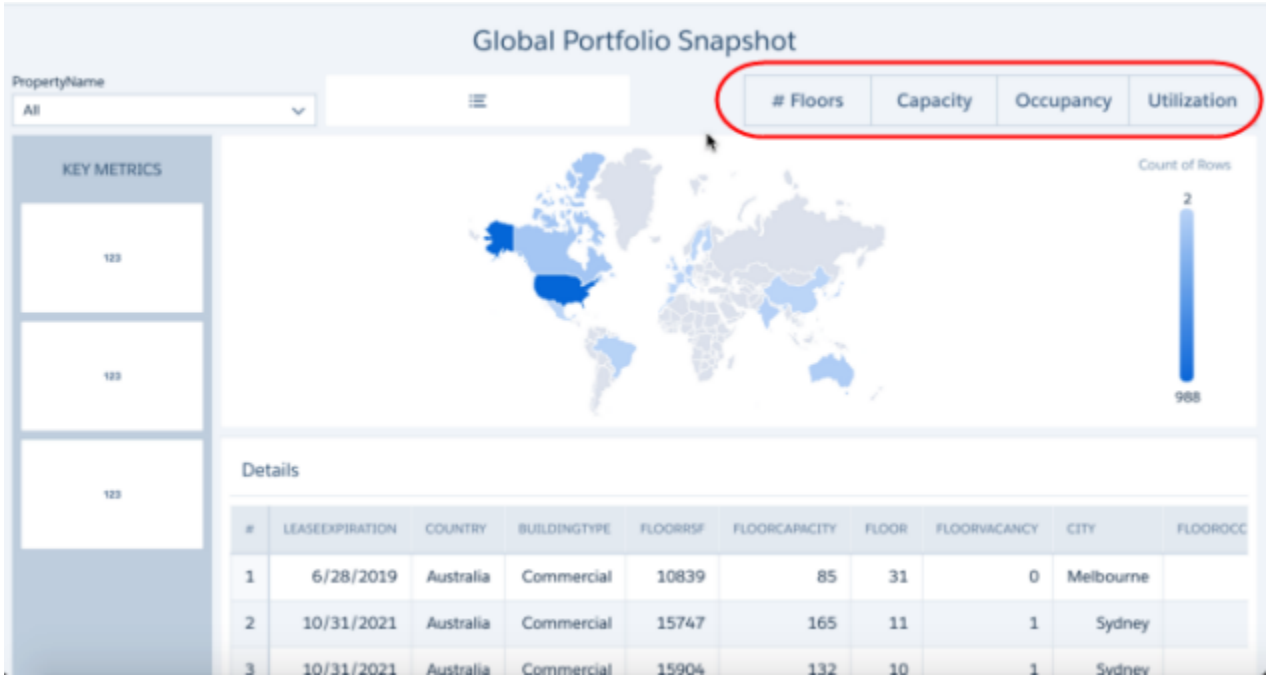
コンパクトフォームクエリと SAQL クエリのバインド

バインドはコンパクトフォームクエリと SAQL クエリの両方で使用できます。SAQL クエリステップの基準またはグループをバインドする場合、`measures` 項目と `groups` 項目の 2 つの場所、および `pigql` 項目でバインドを定義する必要があります。

基準のバインド

ウィジェットに表示する基準を選択できます。たとえば、切り替えウィジェットの選択内容に基づいて、ダッシュボードにさまざまな基準を表示できます。静的ステップ内の選択内容に基づいて、ステップクエリ内の基準を決定します。

例を見てみましょう。



切り替えウィジェットでは、次のステップを使用します。

```
"static_1": {
 "datasets": [],
 "dimensions": [],
 "isFacet": false,
 "isGlobal": false,
 "selectMode": "single",
 "type": "staticflex",
 "useGlobal": false,
 "values": [{
  "display": "# Floors",
```

```
  "value": [
   "count",
   "*"
  ]
 }, {
  "display": "Capacity",
  "value": [
   "sum",
   "FloorCapacity"
  ]
 }, {
  "display": "Occupancy",
  "value": [
   "sum",
   "Occupied"
  ]
 }, {
  "display": "Utilization",
  "value": [
   "sum",
   "FloorVacancy"
  ]
 }],
 "visualizationParameters": {
  "options": {}
 }
}
```

COUNTRY_1 ステップの measures 属性が static_1 ステップにバインドされます。静的ステップ内の選択内容によって集計方法 (sum、count など) と基準項目 (Occupied、FloorCapacity など) が渡されます。

```
"COUNTRY_1": {
 "datasets": [{
  "id": "0FbB00000000qwyKAA",
  "label": "Serraview Floor Metrics",
  "name": "Serraview_Floor_Metrics",
  "url": "/services/data/v38.0/wave/datasets/0FbB00000000qwyKAA"
 }],
 "isFacet": true,
 "isGlobal": false,
 "query": {
  "measures": ["{{cell(static_1.selection, 0, \"value\").asObject()}}"],
  "groups": [
   "COUNTRY"
  ]
 },
 "selectMode": "single",
 "type": "aggregateflex",
 "useGlobal": true,
 "visualizationParameters": {
  "visualizationType": "hbar",
  "options": {}
 }
}
```

地図ウィジェットは COUNTRY_1 ステップに基づきます。

```
"chart_1": {
 "parameters": {
  "legend": {
   "showHeader": true,
   "show": true,
   "position": "right-top",
   "inside": false
  },
  "highColor": "#1674D9",
  "lowColor": "#C5DBF7",
  "visualizationType": "choropleth",
  "step": "COUNTRY_1",
  "theme": "wave",
  "exploreLink": true,
  "trellis": {
   "enable": false,
   "type": "x",
   "chartsPerLine": 4
  },
  "title": {
   "label": "",
   "align": "center",
   "subtitleLabel": ""
  },
  "map": "World Countries"
 },
 "type": "chart"
}
```

条件のバインド

SAQL クエリでさまざまな条件種別を作成できます。次のセクションでは、さまざまなバインド種別を使用する条件の例について説明します。

等価検索条件

asEquality() 逐次化関数を使用して、等価に基づく検索条件をバインドします。

不等価検索条件

不等価検索条件と SAQL 比較演算子を組み合わせて、より複雑なバインドに基づく検索条件を作成します。

範囲検索条件

asRange() 逐次化関数を使用して、数値範囲に基づいて検索条件をバインドします。

日付範囲検索条件

asDateRange() 逐次化関数を使用して、日付範囲に基づいて検索条件をバインドします。絶対日付範囲または相対日付範囲を使用して検索条件を作成できます。

等価検索条件

asEquality() 逐次化関数を使用して、等価に基づく検索条件をバインドします。

ソースステップから次の結果が返されたとします。

```
[
    {grouping: "first", measure: 19}
    {grouping: "second", measure: 32}
]
```

asEquality() バインド関数を使用して、検索条件をバインドできます。次の検索条件で、返された値が「bar」と等しいかどうかを判定します。

```
q = filter q by {{cell(stepFoo.selection, 1, "measure").asEquality("bar")}};
```

ソースステップから返されたデータに基づいてバインドを評価した後、Wave は次の検索条件を生成します。

```
q = filter q by bar == 32;
```

次の検索条件で、「grouping」列のいずれかの値が「bar」と等しいかどうかを判定します。

```
q = filter q by {{column(stepFoo.selection, ["grouping"]).asEquality("bar")}};
```

バインドを評価した後、検索条件は次のようになります。

```
q = filter q by bar in ["first","second"];
```

不等価検索条件

不等価検索条件と SAQL 比較演算子を組み合わせて、より複雑なバインドに基づく検索条件を作成します。

たとえば、ソースステップから次の結果が返されたとします。

```
[ {grouping: "first", measure: 19} {grouping: "second", measure: 32} ]
```

不等価演算子を使用して、検索条件のバインドを作成できます。

```
q = filter q by bar > {{cell(stepFoo.selection, 1, "measure").asString()}};
```

バインドを評価した後、検索条件は次のようになります。

```
q = filter q by bar > 32;
```

matches など、他の SAQL 比較演算子を使用することもできます。

```
q = filter q by bar matches "{{cell(stepFoo.selection, 1, "grouping").asString()}}";
```

バインドを評価した後、検索条件は次のようになります。

```
q = filter q by bar matches "second";
```

範囲検索条件

asRange() 逐次化関数を使用して、数値範囲に基づいて検索条件をバインドします。

包括的範囲を使用した例を見てみましょう。

バインドのソースステップによって、次の結果が生成されます。

```
[ {grouping: "first", measure: 19} {grouping: "second", measure: 32} ]
```

次の構文を使用して、検索条件をバインドできます。

```
q = filter q by {{row(stepFoo.selection, [0], ["min", "max"]).asRange("bar")}};
```

バインドを評価した後、Wave は次の範囲検索条件を生成します。

```
q = filter q by bar >= 19 && bar <= 32;
```

日付範囲検索条件

`asDateRange()` 逐次化関数を使用して、日付範囲に基づいて検索条件をバインドします。絶対日付範囲または相対日付範囲を使用して検索条件を作成できます。

入力データが、2 つの要素を含む一次元配列の場合、次のように処理されます。

- 両方の要素が数値の場合、Wave はその数値をエポック時間とみなします。`[1016504910000, 1016504910000]` は `fieldName in [dateRange([2002,3,19], [2010,8,12])]` になります。
- それ以外の場合、最初の要素を最小として使用し、2 番目の要素を最大として使用します。`["current day", "1 month ahead"]` は `fieldName in ["current day".."1 month ahead"]` になります。どちらかの要素が null の場合、無制限の日付範囲になります。`["1 month ago", null]` は `fieldName in ["1 month ago"..]` になります。

入力データが二次元配列であり、外側の配列に 2 つの要素がある場合、次のように処理されます。

- ネストされた両方の配列に 2 つの要素がある場合、Wave はデータを相対日付の配列形式とみなします。`[["year", -2], ["year", 1]]` は `fieldName in ["2 years ago".."1 year ahead"]` になります。ネストされた両方の配列に 3 つの要素がある場合、ネストされた配列は SAQL `dateRange()` 関数に渡されます。`[[2015, 2, 1], [2016, 2, 1]]` は `fieldName in [dateRange([2015,2,1], [2016,2,1])]` になります。

入力データが `null` の場合、`fieldName in all` になり、何も絞り込まれません。

日付検索条件ウィジェットへのバインド

たとえば、日付ウィジェットでデータを選択し、次の絶対日付範囲 (エポック形式) が返されたとします。

```
[ {min: 1016504910000, max: 1281655993000} ]
```

返された選択データを使用して検索条件を作成できます。

```
q = filter q by {{row(stepFoo.selection, [0], ["min", "max"]).asDateRange("date(year,
month, day)")}};
```

バインドを評価した後、Wave は次の日付範囲検索条件を生成します。

```
q = filter q by date(year, month, day) in [dateRange([2002,3,19], [2010,8,12])];
```

相対日付ではどうでしょうか? 選択内容に基づいて、日付ウィジェットが次の相対日付を返したとします。

```
[ {min: ["quarter", -2], max: ["quarter", 3]} ]
```

評価の後、次の日付範囲検索条件が生成されます。

```
q = filter q by date(year, month, day) in ["2 quarters ago".."3 quarters ahead"];
```

日付範囲のカスタムリストへのバインド

カスタムの日付範囲セットに基づいて絞り込みを行うことはよくあります。これを行うには、各カスタム日付範囲の行を持つ静的ステップを作成します。絶対または相対日付を使用して範囲を指定できます。

絶対範囲を使用してこれを行うには、静的ステップの結果で絶対日付を返す必要があります。

```
[
    {label: "8/30/15 - 8/30/16", range: [[2015, 8, 30], [2016, 8, 30]]}
    {label: "7/30/16 - 8/30/16", range: [[2016, 7, 30], [2016, 8, 30]]}
]
```

ソースステップの選択値に基づいて検索条件を作成できます。

```
q = filter q by {{cell(stepFoo.selection, 0, "range").asDateRange("date(year, month,
day)")}};
```

Wave **がバインドを評価した後、検索条件は次のようになります。**

```
q = filter q by date(year, month, day) in [dateRange([2015, 8, 30], [2016, 8, 30])];
```

相対範囲を使用してこれを行うには、ソースステップの結果が次のようになる必要があります。

```
[
    {"label": "YTD", "range": ["1 year ago", "current day"]}
    {"label": "MTD", "range": ["1 month ago", "current day"]}
    {"label": "Everything up to today", "range": [null, "current day"]}
]
```

次のバインドを使用して、ソースステップの選択値に基づく検索条件を作成できます。

```
q = filter q by {{cell(stepFoo.selection, 0, "range").asDateRange("date(year, month,
day)")}};
```

Wave **がバインドを評価した後、検索条件は次のようになります。**

```
q = filter q by date(year, month, day) in ["1 year ago".."current day"];
```

ソースステップで相対日付キーワードの 1 **つに** null **を指定することで、無制限の範囲の検索条件を作成することもできます。バインドした検索条件は次のようになります。**

```
q = filter q by {{cell(stepFoo.selection, 2, "range").asDateRange("date(year, month,
day)")}};
```

Wave **がバインドを評価した後、検索条件は次のようになります。**

```
q = filter q by date(year, month, day) in [.."current day"];
```

射影のバインド

`asProjection()` **逐次化関数を使用して、**SAQL **クエリで項目の射影を指定します。**

ソースステップから次のデータが提供されています。

```
[
    {expression: "first", alias: "foo"}
    {expression: "second", alias: "bar"}
]
```

対象のステップで項目の射影をバインドできます。

```
q = foreach q generate {{row(stepFoo.selection, [0], ["expression",
"alias"]).asProjection()}};
```

Wave がバインドを評価した後、射影は次のようになります。

```
q = foreach q generate first as 'foo';
```

バインドにすべての行を返すには、次の検索条件を作成します。

```
q = foreach q generate {{row(stepFoo.selection, [], ["expression",
"alias"]).asProjection()}};
```

Wave がバインドを評価した後、検索条件は次のようになります。

```
q = foreach q generate first as 'foo', second as 'bar';
```

グループのバインド

`asGrouping()` 逐次化関数を使用して、SAQL クエリまたはコンパクトフォームクエリでグループ化をバインドします。

切り替えウィジェットの選択内容によってステップクエリ内のグループ化を決定する例を見てみましょう。

ソースステップから次のデータが提供されています。

```
[ {grouping: "first", alias: "foo"} {grouping: "second", alias: "bar"} ]
```

対象のステップに次のグループ化を適用します。

```
q = group q by {{cell(stepFoo.selection, 1, "grouping").asGrouping()}};
```

Wave がバインドを評価した後、グループ化は次のようになります。

```
q = group q by 'second';
```

グループ化に対してバインドで複数の項目を返すには、次のグループ化ロジックを適用します。

```
q = group q by {{column(stepFoo.selection, ["grouping"]).asGrouping()}};
```

Wave がバインドを評価した後、グループ化は次のようになります。

```
q = group q by ('first', 'second');
```

順序のバインド

`asOrder()` 逐次化関数を使用して、SAQL クエリの並び替え順序を指定します。

切り替えウィジェットの選択内容によってステップのSAQLクエリの並び替え順序を決定する例を見てみましょう。

ソースステップから次のデータが提供されています。

```
[
    {order: "first", direction: "desc"}
    {order: "second", direction: "asc"}
]
```

1 つの項目で並び替えるには、次の順序ロジックを適用します。クエリで方向を指定しない場合、デフォルト値は昇順です。

```
q = order q by {{cell(stepFoo.selection, 1, "order").asOrder()}};
```

Wave がバインドを評価した後、グループ化は次のようになります。

```
q = order q by 'second';
```

複数の項目で並び替えるには、次のグループ化ロジックを使用します。

```
q = order q by {{column(stepFoo.selection, ["order"]).asOrder()}};
```

Wave がバインドを評価した後、グループ化は次のようになります。

```
q = order q by ('first', 'second');
```

順序と方向を指定するには、次のグループ化ロジックを使用します。

```
q = order q by {{row(stepFoo.selection, [], ["order", "direction"]).asOrder()}};
```

Wave がバインドを評価した後、グループ化は次のようになります。

```
q = order q by ('first' desc, 'second' asc);
```

制限およびオフセットのバインド

SAQL クエリの制限とオフセットもバインドできます。このバインドにデータ逐次化関数は必要ありません。

次のデータを提供するソースステップを考えます。

```
[ {limit: 100, offset: 10} ]
```

制限とオフセットをバインドするには、次のロジックを作成します。

```
q = limit q {{cell(stepFoo.selection, 0, "limit").asString()}};
q = offset q {{cell(stepFoo.selection, 0, "offset").asString()}};
```

Wave がバインドを評価した後、制限とオフセットは次のようになります。

```
q = limit q 100; q = offset q 10;
```

制限とオフセットについての詳細は、 *『Wave Analytics SAQL リファレンス』* を参照してください。

コンパクトフォームクエリと SAQL クエリのバインド

バインドはコンパクトフォームクエリと SAQL クエリの両方で使用できます。SAQL クエリステップの基準またはグループをバインドする場合、`measures` 項目と `groups` 項目の 2 つの場所、および `pigql` 項目でバインドを定義する必要があります。

次に、SAQL クエリの一部が他のステップにバインドされている例を示します。measures と groups の 2 つの場所でバインドが定義されています。

```
"steps": {
    "StaticSAQLGroupingNames": {
        "datasets": [],
        "dimensions": [],
```

```
        "isFacet": true,
        "isGlobal": false,
        "selectMode": "multirequired",
        "start": {
            "display": [ "Country" ]
        },
        "type": "staticflex",
        "useGlobal": true,
        "values": [
            {
                "display": "Country",
                "value": "Account.BillingCountry",
                "expression": "'Account.BillingCountry'",
                "alias": "Account.BillingCountry"
            },
            {
                "display": "Industry",
                "value": "Account.Industry",
                "expression": "'Account.Industry'",
                "alias": "Account.Industry"
            },
            {
                "display": "Product",
                "value": "Product.Product.Family",
                "expression": "'Product.Product.Family'",
                "alias": "Product.Product.Family"
            },
            {
                "display": "Source",
                "value": "Account.AccountSource",
                "expression": "'Account.AccountSource'",
                "alias": "Account.AccountSource"
            }
        ],
        "visualizationParameters": {
            "options": {}
        }
    },
    "StaticSAQLMeasureNames": {
        "datasets": [],
        "dimensions": [],
        "isFacet": true,
        "isGlobal": false,
        "selectMode": "singlerequired",
        "start": {
            "display": [ "Total Amount" ]
        },
        "type": "staticflex",
        "useGlobal": true,
        "values": [
            {
                "display": "Max Employees",
                "cf": [
                    "max",
```

```
                    "Account.NumberOfEmployees"
                ],
                "expression": "max('Account.NumberOfEmployees')",
                "alias": "max_Account.NumberOfEmployees"
            },
            {
                "display": "Total Amount",
                "cf": [
                    "sum",
                    "Amount"
                ],
                "expression": "sum('Amount')",
                "alias": "sum_Amount"
            },
            {
                "display": "Avg Amount",
                "cf": [
                    "avg",
                    "Amount"
                ],
                "expression": "avg('Amount')",
                "alias": "avg_Amount"
            }
        ],
        "visualizationParameters": {
            "options": {}
        }
    },
    "Account_BillingCount_2": {
        "datasets": [
            {
                "id": "0FbB00000000oEkKAI",
                "label": "Opportunities",
                "name": "opportunity",
                "url": "/services/data/v38.0/wave/datasets/0FbB00000000oEkKAI"
            }
        ],
        "isFacet": true,
        "isGlobal": false,
        "query": {
            "pigql": "
                q = load \"opportunity\";\nq = filter
                q by 'Account.AccountSource' == \"Advertisement\";\n
                q = group q by {{column(StaticSAQLGroupingNames.selection,
[\"value\"]).asGrouping()}};\n
                q = foreach q generate
                    {{row(StaticSAQLGroupingNames.selection, [], [\"expression\",
\"alias\"]).asProjection()}},
                    {{row(StaticSAQLMeasureNames.selection, [], [\"expression\",
\"alias\"]).asProjection()}};\n
                q = filter q by {{column(StaticSAQLFilters.selection,
[\"value\"]).asEquality(\"Account.BillingCountry\")}};\n
                q = filter q by {{row(StaticSAQLMinRanges.selection, [0], [\"min\",
\"max\"]).asRange(\"sum_Amount\")}}; \n
```

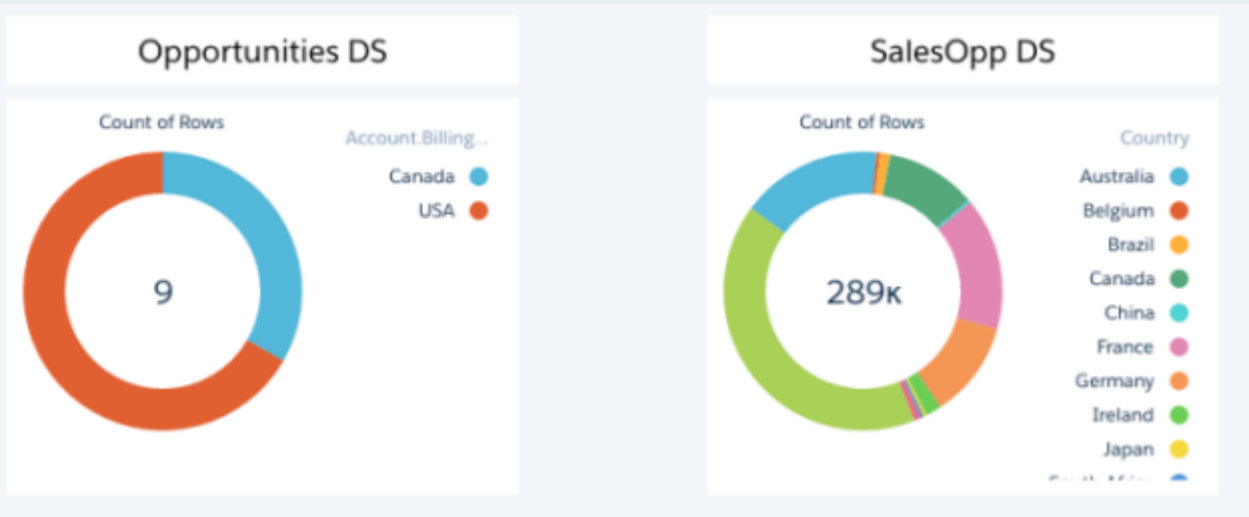
```
                q = order q by {{row(StaticSAQLOrdering.selection, [], [\"order\",
\"direction\"]).asOrder()}};",
            "measures": "{{column(StaticSAQLMeasureNames.selection, [\"cf\"]).asObject()}}",

              "measuresMap": {},
            "groups": "{{column(StaticSAQLGroupingNames.selection, [\"value\"]).asObject()}}"

        },
        "type": "aggregateflex",
        "useGlobal": true,
        "visualizationParameters": {
            "options": {}
        }
}
```

異なるデータセットのステップのバインド

異なるデータセットのステップをバインドできます。たとえば、次のダッシュボードにはそれぞれ独自のデータセットに基づいている 2 つのグラフが含まれています。



[Opportunities DS (商談 DS)] グラフで選択が行われると、選択バインドによって [SalesOpp DS (営業商談 DS)] グラフも絞り込まれます。`Country_1` ステップの `filters` 属性には、`Account_BillingCount_1` ステップに基づく選択バインドが含まれます。

次に、ステップおよびグラフウィジェットのダッシュボード JSON を示します。

```
{
    "label": "Cross-Dataset Bindings",
    "state": {
        "gridLayouts": [...],
        "layouts": [],
        "steps": {
            "Account_BillingCount_1": {
                "datasets": [
                    {
```

```
                        "id": "0Fbx000000000LzCAI",
                        "label": "Opportunities",
                        "name": "opportunity1",
                        "url": "/services/data/v38.0/wave/datasets/0Fbx000000000LzCAI"
                    }
                ],
                "isFacet": true,
                "isGlobal": false,
                "query": {
                    "measures": [
                        [
                            "count",
                            "*"
                        ]
                    ],
                    "groups": [
                        "Account.BillingCountry"
                    ]
                },
                "type": "aggregateflex",
                "useGlobal": true,
                "visualizationParameters": {
                    "visualizationType": "hbar",
                    "options": {}
                }
            },
            "Country_1": {
                "datasets": [
                    {
                        "id": "0Fbx000000000NACAY",
                        "label": "SalesOpps",
                        "name": "SalesOpps",
                        "url": "/services/data/v38.0/wave/datasets/0Fbx000000000NACAY"
                    }
                ],
                "isFacet": true,
                "isGlobal": false,
                "query": {
                    "measures": [
                        [
                            "count",
                            "*"
                        ]
                    ],
                    "groups": [
                        "Country"
                    ],
                    "filters": [
                        [
                            "Country",
                            "{{column(Account_BillingCount_1.selection,
[\"Account.BillingCountry\"]).asObject()}}"
                        ]
                    ]
```

```
            },
            "type": "aggregateflex",
            "useGlobal": true,
            "visualizationParameters": {
                "visualizationType": "hbar",
                "options": {}
            }
        }
    },
    "widgetStyle": {...},
    "widgets": {
        "text_1": {
            "parameters": {
                "fontSize": 20,
                "text": "Opportunities DS",
                "textAlignment": "center",
                "textColor": "#000000"
            },
            "type": "text"
        },
        "text_2": {
            "parameters": {
                "fontSize": 20,
                "text": "SalesOpp DS",
                "textAlignment": "center",
                "textColor": "#000000"
            },
            "type": "text"
        },
        "chart_2": {
            "parameters": {
                "legend": {
                    "showHeader": true,
                    "show": true,
                    "position": "right-top",
                    "inside": false
                },
                "showMeasureTitle": true,
                "showTotal": true,
                "visualizationType": "pie",
                "step": "Country_1",
                "exploreLink": true,
                "inner": 70,
                "title": {
                    "label": "",
                    "subtitleLabel": "",
                    "align": "center"
                },
                "theme": "wave",
                "trellis": {
                    "enable": false,
                    "type": "x",
                    "chartsPerLine": 4
                }
```

```
                },
                "type": "chart"
            },
            "chart_1": {
                "parameters": {
                    "legend": {
                        "showHeader": true,
                        "show": true,
                        "position": "right-top",
                        "inside": false
                    },
                    "showMeasureTitle": true,
                    "showTotal": true,
                    "visualizationType": "pie",
                    "step": "Account_BillingCount_1",
                    "exploreLink": true,
                    "inner": 70,
                    "title": {
                        "label": "",
                        "subtitleLabel": "",
                        "align": "center"
                    },
                    "theme": "wave",
                    "trellis": {
                        "enable": false,
                        "type": "x",
                        "chartsPerLine": 4
                    }
                },
                "type": "chart"
            }
        }
    },
    "datasets": [
        {
            "id": "0Fbx000000000LzCAI",
            "label": "Opportunities",
            "name": "opportunity1",
            "url": "/services/data/v38.0/wave/datasets/0Fbx000000000LzCAI"
        },
        {
            "id": "0Fbx000000000NACAY",
            "label": "SalesOpps",
            "name": "SalesOpps",
            "url": "/services/data/v38.0/wave/datasets/0Fbx000000000NACAY"
        }
    ]
}
```

静的ステップと他のステップのバインド

クエリから値を取得する代わりに、静的ステップを作成してステップの独自の値を指定できます。たとえば、切り替えウィジェットに [Top 5 Customer (上位 5位 (顧客))] と [Bottom 5 Customers (下位 5 位 (顧客))] を表示する静的ステップを作成できます。静的ステップを作成したら、ダッシュボードの他のウィジェットとやりとりするために、静的ステップを他のウィジェットのステップに手動でバインドします。

「基準のバインド」の例を参照してください。

ウィジェットプロパティのバインド

Wave デザイナダッシュボードでは、多くのグラフウィジェットのプロパティを動的に変更するバインドを実装できます。

色分けを使用した値の強調表示

選択内容または他のステップの結果に基づいてウィジェットのコンテンツを強調表示できます。たとえば、しきい値に基づいて数値ウィジェットの値を色コード化して、高い数値や低い数値が注目されるようにします。

切り替えウィジェットに基づく地図タイプの変更

切り替えウィジェットの選択内容に基づいて地図タイプを動的に変更できます。たとえば、2つの異なる地図タイプを切り替えるトグルを作成できます。

基準線と表示ラベルの動的な設定

ステップの基準の基づいて基準線とその表示ラベルを動的に設定できます。たとえば、販売目標を表す基準線を設定して、成立商談と比較できます。

色分けを使用した値の強調表示

選択内容または他のステップの結果に基づいてウィジェットのコンテンツを強調表示できます。たとえば、しきい値に基づいて数値ウィジェットの値を色コード化して、高い数値や低い数値が注目されるようにします。

数値が高 (緑色)、中 (黄色)、低 (赤色) のいずれになるかに基づいて、3つの数値ウィジェットで基準の色を変更するとします。

ダッシュボードJSONで、各ステップの測定値に基づいて色を計算します。次に各数値ウィジェットのnumberColor項目に計算された色を適用します。

```
{
  "label": "Sales Overview",
  "state": {
    "gridLayouts": [...],
    "layouts": [],
    "steps": {
      "color_1": {
        "type": "aggregateflex",
        "visualizationParameters": {
          "options": {}
        },
        "query": {
          "pigql": "q = load \"Opportunity_Dataset\";\n
                    q = filter q by 'Region' == \"US\";\n
                    q = group q by all;\n
                    q = foreach q generate count() as 'count',
                        (case when count() < 25000 then \"#EE0A50\"
                              when count() < 50000 then \"#F8CE00\"
                              else \"#0FD178\" end) as 'color';\n
                    q = limit q 2000;",
          "measures": [ [
            "count",
            "*",
            "count" ] ],
          "groups": [ "color" ],
          "measuresMap": {}
        },
  "isFacet": true,
  "useGlobal": true,
  "isGlobal": false,
  "datasets": [{
    "name": "Opportunity_Dataset",
     "url": "/services/data/v38.0/wave/datasets/0Fbx000000000KLCAY",
     "id": "0Fbx000000000KLCAY"
        }]
      },
      "color_2": {
        "type": "aggregateflex",
        "visualizationParameters": {
          "options": {}
        },
        "query": {
          "pigql": "q = load \"Opportunity_Dataset\";\n
                    q = filter q by 'Region' == \"AP\";\n
                    q = group q by all;\n
                    q = foreach q generate count() as 'count',
                        (case when count() < 25000 then \"#EE0A50\"
                              when count() < 50000 then \"#F8CE00\"
                              else \"#0FD178\" end) as 'color';\n
                    q = limit q 2000;",
          "measures": [ [
```

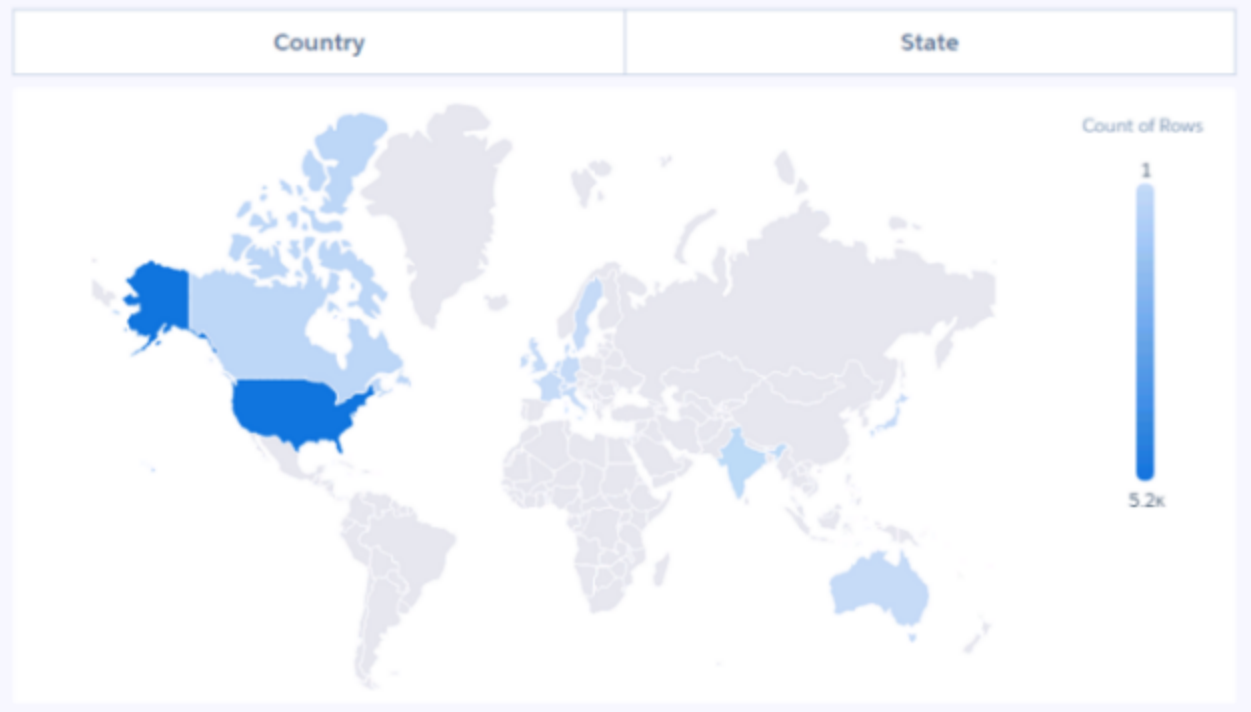
```
      "count",
      "*",
      "count"
          ] ],
          "groups": [ "color" ],
          "measuresMap": {}
        },
        "isFacet": true,
        "useGlobal": true,
        "isGlobal": false,
        "datasets": [{
          "name": "Opportunity_Dataset",
          "url": "/services/data/v38.0/wave/datasets/0Fbx000000000KLCAY",
          "id": "0Fbx000000000KLCAY"
        }]
      },
      "color_3": {
        "type": "aggregateflex",
        "visualizationParameters": {
          "options": {}
        },
        "query": {
          "pigql": "q = load \"Opportunity_Dataset\";\n
                    q = filter q by 'Region' == \"EU\";\n
                    q = group q by all;\n
                    q = foreach q generate count() as 'count',
                        (case when count() < 25000 then \"#EE0A50\"
                              when count() < 50000 then \"#F8CE00\"
                              else \"#0FD178\" end) as 'color';\n
                    q = limit q 2000;",
          "measures": [ [
            "count",
            "*",
            "count"
          ] ],
          "groups": [ "color" ],
          "measuresMap": {}
        },
        "isFacet": true,
        "useGlobal": true,
        "isGlobal": false,
        "datasets": [{
          "name": "Opportunity_Dataset",
          "url": "/services/data/v38.0/wave/datasets/0Fbx000000000KLCAY",
          "id": "0Fbx000000000KLCAY"
        }]
      }
    },
    "widgetStyle": {...},
    "widgets": {
      "number_5": {
        "type": "number",
        "parameters": {
          "step": "color_1",
```

```
          "measureField": "count",
          "textAlignment": "right",
          "compact": false,
          "exploreLink": true,
          "titleColor": "#335779",
          "titleSize": 14,
          "numberColor": "{{cell(color_1.result, 0, \"color\").asString()}}",
          "numberSize": 32,
          "title": "Opp Count (United States)"
        }
      },
      "number_6": {
        "type": "number",
        "parameters": {
          "step": "color_2",
          "measureField": "count",
          "textAlignment": "right",
          "compact": false,
          "exploreLink": true,
          "titleColor": "#335779",
          "titleSize": 14,
          "numberColor": "{{cell(color_2.result, 0, \"color\").asString()}}",
          "numberSize": 32,
          "title": "Opp Count (Asia Pacific)"
        }
      },
      "number_7": {
        "type": "number",
        "parameters": {
          "step": "color_3",
          "measureField": "count",
          "textAlignment": "right",
          "compact": false,
          "exploreLink": true,
          "titleColor": "#335779",
          "titleSize": 14,
          "numberColor": "{{cell(color_3.result, 0, \"color\").asString()}}",
          "numberSize": 32,
          "title": "Opp Count (Europe)"
        }
      }
    }
  },
  "datasets": [...]
}
```

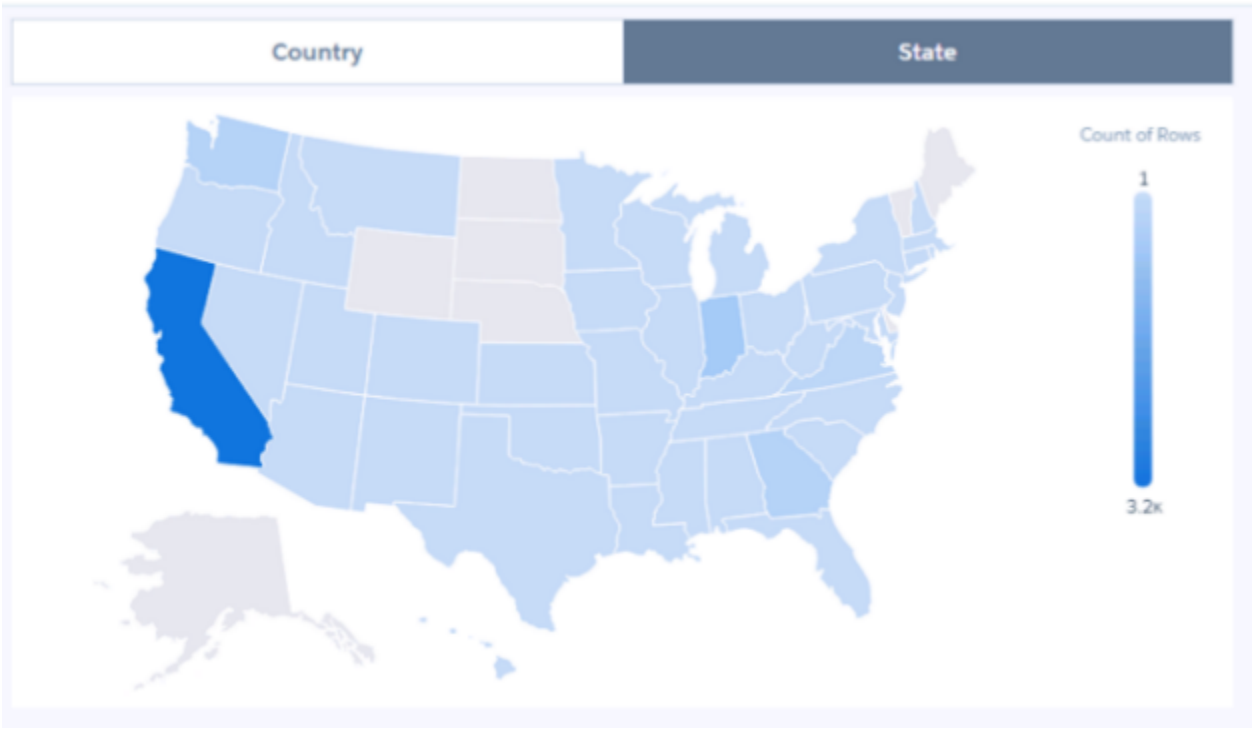
切り替えウィジェットに基づく地図タイプの変更

切り替えウィジェットの選択内容に基づいて地図タイプを動的に変更できます。たとえば、2つの異なる地図タイプを切り替えるトグルを作成できます。

世界と米国内の会社の状況を分析するとします。この作業を行うには、世界の国の地図と米国の州の地図の表示を切り替えることができるトグルを追加します。トグルで何も選択しないと、デフォルトで世界地図が表示されます。



[State (州)] 切り替えオプションをクリックすると、ダッシュボードには各州の結果が表示されます。



静的ステップ (static_1) により、切り替えウィジェットに表示される「Country (国)」と「State (州)」の値が指定されます。グラフウィジェットには、表示する地図タイプを選択できるようにする地図タイププロパティがあります。静的ステップ (static_1) で選択内容に基づいて地図タイプを動的に設定するには、グラフウィジェットのステップ (State__c_1) で地図タイププロパティを静的ステップ (static_1) にバインドします。地図についての詳細は、「地図」を参照してください。

次にダッシュボード JSON を示します。

```
{
    "label": "Choropleth Binding",
    "state": {
        "gridLayouts": [...],
        "layouts": [],
        "steps": {
            "static_1": {
                "datasets": [],
                "dimensions": [],
                "isFacet": false,
                "isGlobal": false,
                "selectMode": "single",
                "type": "staticflex",
                "useGlobal": false,
                "values": [
                    {
                        "display": "Country",
                        "grouping": "Country__c",
                        "mapType": "World Countries"
                    },
                    {
```

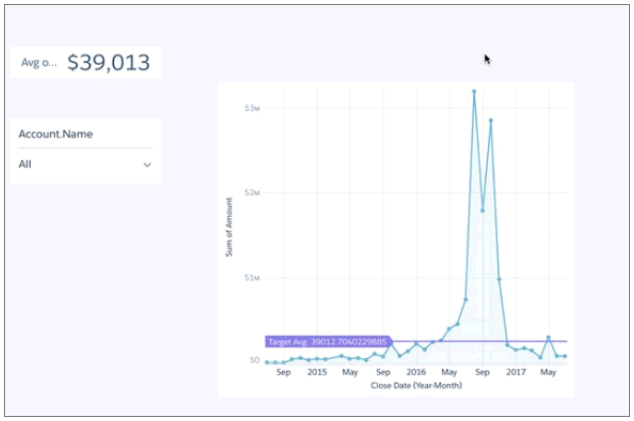
```
                        "display": "State",
                        "grouping": "State__c",
                        "mapType": "US States"
                    }
                ],
                "visualizationParameters": {
                    "options": {}
                }
            },
            "State__c_1": {
                "datasets": [
                    {
                        "id": "0FbB000000001uGKAQ",
                        "label": "GUS Roster",
                        "name": "Roster",
                        "url": "/services/data/v38.0/wave/datasets/0FbB000000001uGKAQ"
                    }
                ],
                "isFacet": true,
                "isGlobal": false,
                "query": {
                    "measures": [
                        [
                            "count",
                            "*"
                        ]
                    ],
                    "groups": [
                        "{{coalesce(cell(static_1.selection, 0, \"grouping\"),
cell(static_1.result, 0, \"grouping\")).asString()}}"
                    ]
                },
                "type": "aggregateflex",
                "useGlobal": true,
                "visualizationParameters": {
                    "visualizationType": "hbar",
                    "options": {}
                }
            }
        },
        "widgetStyle": {...},
        "widgets": {
            "pillbox_1": {
                "parameters": {
                    "compact": false,
                    "exploreLink": false,
                    "step": "static_1"
                },
                "type": "pillbox"
            },
            "chart_1": {
                "parameters": {
                    "legend": {
                        "showHeader": true,
```

```
                        "show": true,
                        "position": "right-top",
                        "inside": false
                    },
                    "highColor": "#1674D9",
                    "lowColor": "#C5DBF7",
                    "visualizationType": "choropleth",
                    "step": "State__c_1",
                    "theme": "wave",
                    "exploreLink": true,
                    "title": {
                        "label": "",
                        "align": "center",
                        "subtitleLabel": ""
                    },
                    "trellis": {
                        "enable": false,
                        "type": "x",
                        "chartsPerLine": 4
                    },
                    "map": "{{coalesce(cell(static_1.selection, 0, \"mapType\"),
cell(static_1.result, 0, \"mapType\")).asString()}}"
                },
                "type": "chart"
            }
        }
    },
    "datasets": [...]
}
```

基準線と表示ラベルの動的な設定

ステップの基準の基づいて基準線とその表示ラベルを動的に設定できます。たとえば、販売目標を表す基準線を設定して、成立商談と比較できます。

次の例では、経時的な合計商談金額を表示するタイムライングラフがあります。ダッシュボードには、特定の取引先の合計金額を表示できるようにするリストセレクタも含まれています。各取引先の合計をすべての取引先の平均に対して比較するには、すべての取引先の平均商談金額に基づいて基準線を設定します。



すべての取引先の平均に基づいて基準線の表示ラベルとその値を作成するには、次のようにタイムライングラフのウィジェットプロパティでバインドを追加します。

```
{
 "label": "Total Opportunity Amount Per Rep Vs. Average",
 "state": {
  "gridLayouts": [...],
  "layouts": [],
  "steps": {
   "Account_Name_1": {
    "datasets": [{
     "id": "0FbB00000000pNNKAY",
     "label": "Opportunities",
     "name": "opportunity",
     "url": "/services/data/v38.0/wave/datasets/0FbB00000000pNNKAY"
    }],
    "isFacet": true,
    "isGlobal": false,
    "query": {
     "measures": [
      [
       "sum",
       "Amount"
      ]
     ],
     "groups": [
      "Account.Name"
     ]
    },
    "type": "aggregateflex",
    "useGlobal": true,
    "visualizationParameters": {
     "visualizationType": "hbar",
     "options": {}
    }
   },
   "CloseDate_Year_Close_1": {
    "datasets": [{
     "id": "0FbB00000000pNNKAY",
     "label": "Opportunities",
     "name": "opportunity",
     "url": "/services/data/v38.0/wave/datasets/0FbB00000000pNNKAY"
    }],
    "isFacet": true,
    "isGlobal": false,
    "query": {
     "measures": [
      [
       "sum",
       "Amount"
      ]
     ],
     "groups": [
```

```
      [
       "CloseDate_Year",
       "CloseDate_Month"
      ]
     ]
    },
    "type": "aggregateflex",
    "useGlobal": true,
    "visualizationParameters": {
     "visualizationType": "time",
     "options": {}
    }
   },
   "Amount_1": {
    "datasets": [{
     "id": "0FbB00000000pNNKAY",
     "label": "Opportunities",
     "name": "opportunity",
     "url": "/services/data/v38.0/wave/datasets/0FbB00000000pNNKAY"
    }],
    "isFacet": true,
    "isGlobal": false,
    "query": {
     "measures": [
      [
       "avg",
       "Amount"
      ]
     ]
    },
    "type": "aggregateflex",
    "useGlobal": true,
    "visualizationParameters": {
     "visualizationType": "hbar",
     "options": {}
    }
   }
  },
  "widgetStyle": {
   "backgroundColor": "#FFFFFF",
   "borderColor": "#E6ECF2",
   "borderEdges": [],
   "borderRadius": 0,
   "borderWidth": 1
  },
  "widgets": {
   "number_1": {
    "parameters": {
     "compact": false,
     "exploreLink": true,
     "measureField": "avg_Amount",
     "numberColor": "#335779",
     "numberSize": 32,
     "step": "Amount_1",
```

```
    "textAlignment": "right",
    "titleColor": "#335779",
    "titleSize": 16
   },
   "type": "number"
  },
  "listselector_1": {
   "parameters": {
    "compact": false,
    "expanded": true,
    "exploreLink": false,
    "instant": true,
    "measureField": "sum_Amount",
    "step": "Account_Name_1",
    "title": "Account.Name"
   },
   "type": "listselector"
  },
  "chart_3": {
   "parameters": {
    "fillArea": true,
    "showPoints": true,
    "legend": {
     "showHeader": true,
     "show": true,
     "position": "right-top",
     "inside": false
    },
    "measureAxis1": {
     "showZero": true,
     "showTitle": true,
     "referenceLine": {
      "color": "#9271E8",
      "label": "Target Avg: {{cell(Amount_1.result, 0, \"avg_Amount\").asString()}}",
      "value": "{{cell(Amount_1.result, 0, \"avg_Amount\").asString()}}"
     },
     "showAxis": true,
     "title": ""
    },
    "visualizationType": "time",
    "missingValue": "connect",
    "theme": "wave",
    "step": "CloseDate_Year_Close_1",
    "timeAxis": {
     "showTitle": true,
     "showAxis": true,
     "title": ""
    },
    "exploreLink": true,
    "title": {
     "label": "",
     "align": "center",
     "subtitleLabel": ""
    },
```

```
      "trellis": {
       "enable": false,
       "type": "x",
       "chartsPerLine": 4
      }
     },
     "type": "chart"
    }
   }
  },
  "datasets": [{
   "id": "0FbB00000000pNNKAY",
   "label": "Opportunities",
   "name": "opportunity",
   "url": "/services/data/v38.0/wave/datasets/0FbB00000000pNNKAY"
  }]
}
```

Classic デザイナダッシュボードのバインド

Wave ダッシュボードデザイナとは異なり、Classic デザイナでは選択バインドと結果バインドの扱いが異なります。Classic デザイナの場合、結果バインドでは表形式データが返され、選択バインドでは単一文字列が返されることを想定しています。Classic デザイナでは完全な行が生成されないため、グルーピング列に基づいてのみバインドを作成できます。

メモ: Classic デザイナダッシュボードの場合、静的ステップを範囲ウィジェットにバインドして特定の基準 (金額の範囲など) を表示することはできません。

静的ステップでの選択バインド

ステップのほぼすべての部分で、前のクエリ結果に対する選択バインドを含めることができます。

静的検索条件およびグループセレクタのクエリへのバインド

静的検索条件またはグループセレクタは、SAQL で記述されたクエリにバインドできます。

日付ピッカーおよび静的日付のバインド

選択バインドを使用して、日付ピッカーレンズまたは静的な絶対/相対日付ステップから日付のレンズを絞り込むことができます。

バインド操作

いくつかの追加操作で結果バインドと選択バインドを使用して、正しい結果を抽出できます。

静的ステップでの選択バインド

ステップのほぼすべての部分で、前のクエリ結果に対する選択バインドを含めることができます。

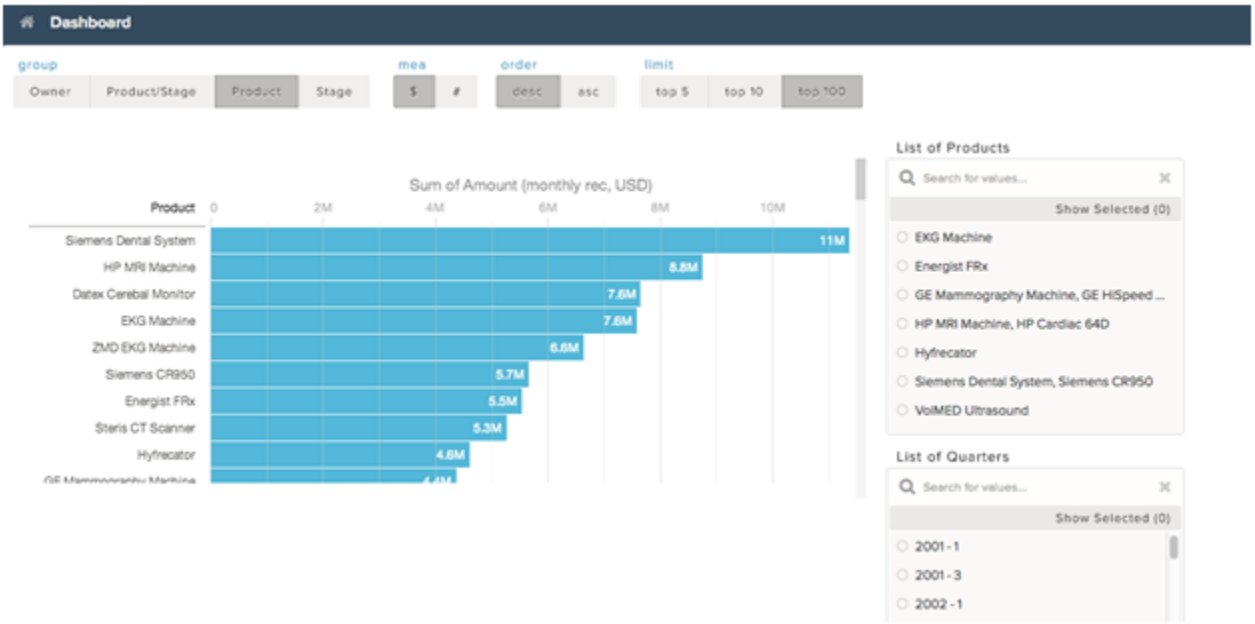
集計クエリでは、選択バインドに次の項目を含めることができます。

- グループ
- 基準

- 検索条件
- 並び替え
- 制限

クエリの任意の部分をバインドする静的ステップの使用

次の例は、クエリの複数の部分に静的ステップと選択バインドが使用されるダッシュボードを示しています。



この例では次の処理が行われます。

- 静的ステップ `step_filter_dim` によって、`"List of Products"` リストセレクタが設定されます。このセレクタには、複数の値があるオプションが含まれます。
- 静的ステップ `step_group` によって、グループ切り替えセレクタが設定されます。`start` 値が `"Product"` であるため、ダッシュボードが初期化されるときのデフォルト値は `"Product"` です。`display` 値によって、ユーザインターフェースの表示名が変更されます。
- 静的ステップ `step_measure` によって、基準切り替えセレクタが設定されます。
- 静的ステップ `step_order` によって、順序切り替えセレクタが設定されます。
- 静的ステップ `step_limit` によって、制限切り替えセレクタが設定されます。
- 集計ステップクエリ `step_quarterly_bookings` は、完了予定日の年と四半期でグループ化されます。
- 集計ステップクエリ `step_top_10` では、静的 `step_group` の選択オプションに応じてグループ化が行われます。`start` 値は、(`step_group` に基づく) `"Product"` グループ化です。

```
{
  "steps": {
    "step_filter_dim": {
      "type": "static",
      "dimensions": [ "Product" ],
```

```
    "datasets":[{"name":"opp"}],
    "selectMode": "single",
    "values": [
      {
        "value": ["EKG Machine"]
      }, {
        "value": ["Energist FRx"]
      }, {
        "value": ["GE Mammography Machine", "GE HiSpeed DXi", "GE Stress System"]
      }, {
        "value": ["HP MRI Machine", "HP Cardiac 64D"]
      }, {
        "value": ["Hyfrecator"]
      }, {
        "value": ["Siemens Dental System", "Siemens CR950"]
      }, {
        "value": ["VolMED Ultrasound"]
      }
    ],
    "isFacet": true
  },
  "step_group": {
    "type": "static",
    "values": [
      {
        "display": "Owner",
        "value": ["Owner-Name"]
      }, {
        "display": "Product/Stage",
        "value": ["Product", "StageName"]
      }, {
        "display": "Product",
        "value": ["Product"]
      }, {
        "display": "Stage",
        "value": ["StageName"]
      }
    ],
    "start": [["Product"]],
    "selectMode": "single"
  },
  "step_measure": {
    "type": "static",
    "values": [
      {
        "display": "$",
        "value": [["sum", "Amount"]]
      }, {
        "display": "#",
        "value": [["count", "*"]]
      }
    ],
    "start": [["sum", "Amount"]]],
    "selectMode": "singlerequired"
```

```
  },
  "step_order": {
    "type": "static",
    "values": [
      {
        "display": "desc",
        "value": false
      }, {
        "display": "asc",
        "value": true
      }
    ],
    "selectMode": "singlerequired"
  },
  "step_limit": {
    "type": "static",
    "values": [
      {
        "display": "top 5",
        "value": 5
      }, {
        "display": "top 10",
        "value": 10
      }, {
        "display": "top 100",
        "value": 100
      }
    ],
    "start": [100],
    "selectMode": "singlerequired"
  },
  "step_quarterly_bookings": {
    "type": "aggregate",
    "datasets":[{"name":"opp"}],
    "query": {
      "groups": [["CloseDate_Year", "CloseDate_Quarter"]],
      "measures": [["sum", "Amount"]]
    },
    "isFacet": true,
    "useGlobal": true
  },
  "step_top_10": {
    "type": "aggregate",
    "datasets":[{"name":"opp"}],
    "query": {
      "groups": "{{ selection(step_group) }}",
      "measures": "{{ selection(step_measure) }}",
      "order": [
        [
          -1, {
            "ascending": "{{ value(selection(step_order)) }}"
          }
        ]
      ],
```

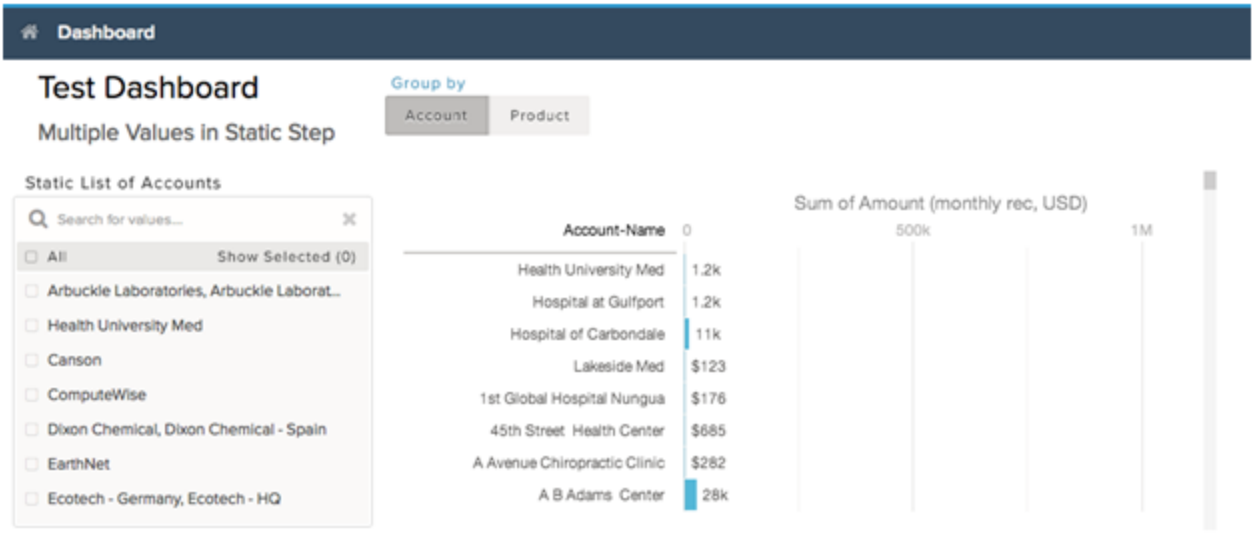
```
        "limit": "{{ value(selection(step_limit)) }}"
      },
      "isFacet": true
    }
  },
  "widgets": {
    "sel_list_filter_dim": {
      "type": "listselector",
      "position": {
        "x": 860,
        "y": 90,
        "w": "290",
        "h": "288"
      },
      "parameters": {
        "step": "step_filter_dim",
        "title": "List of Products",
        "expanded": true,
        "instant": true
      }
    },
    "sel_list_filter_compound_dim": {
      "type": "listselector",
      "position": {
        "x": 860,
        "y": 390,
        "w": "290",
        "h": "288"
      },
      "parameters": {
        "step": "step_quarterly_bookings",
        "title": "List of Quarters",
        "expanded": true,
        "instant": true
      }
    },
    "sel_group": {
      "type": "pillbox",
      "position": {
        "x": 10,
        "y": 10
      },
      "parameters": {
        "title": "group",
        "step": "step_group"
      }
    },
    "sel_measure": {
      "type": "pillbox",
      "position": {
        "x": 380,
        "y": 10
      },
      "parameters": {
```

```
        "title": "mea",
        "step": "step_measure"
      }
    },
    "sel_order": {
      "type": "pillbox",
      "position": {
        "x": 480,
        "y": 10
      },
      "parameters": {
        "title": "order",
        "step": "step_order",
        "start": true
      }
    },
    "sel_limit": {
      "type": "pillbox",
      "position": {
        "x": 620,
        "y": 10
      },
      "parameters": {
        "title": "limit",
        "step": "step_limit"
      }
    },
    "widget1": {
      "type": "chart",
      "position": {
        "x": 10,
        "y": 110,
        "w": "830",
        "h": "330"
      },
      "parameters": {
        "visualizationType": "hbar",
        "step": "step_top_10"
      }
    }
  }
}
```

静的検索条件およびグループセレクタのクエリへのバインド

静的検索条件またはグループセレクタは、SAQL で記述されたクエリにバインドできます。

テンプレートは二重中括弧 ({{ }}) に埋め込まれた式であり、関連付けられているステップの現在の状態で置き換えられます。



たとえば、このダッシュボードには取引先のリストが含まれる静的検索条件ウィジェットがあります。ユーザが取引先と商品のどちらでグループ化するかを指定できる、グループセレクタウィジェットもあります。ユーザの選択に応じて、グラフが更新されます。検索条件はクエリの次の部分で制御します。

```
q = filter q by 'Account-Name' in {{ selection(step_Account_Owner_Name_2) }};
```

step_Account_Owner_Name_2 という名前のステップは選択バインドとして設定されているため、現在の選択状態を取得します。二重中括弧内にあるため、その選択値が代入されてクエリで使用されます。

グルーピングはクエリの次の部分で制御します。

```
q = group q by {{ single_quote(value(selection(step_StageName_3))) }};
q = foreach q generate {{ single_quote(value(selection(step_StageName_3))) }} as {{
value(selection(step_StageName_3)) }}, sum('Amount') as 'sum_Amount', count() as 'count'";
```

ユーザがグループセレクタウィジェットで[商品]を選択すると、クエリエンジンに渡される実際のクエリは次のようになります。

```
q = group q by 'Product';
q = foreach q generate 'Product' as "Product", sum('Amount') as 'sum_Amount', count() as
'count';
```

メモ: グラフの更新に使用されるクエリを表示するには、ブラウザの JavaScript コンソールを開いて *edge.log.query=true* と入力します。ダッシュボードで、別のグループを選択します。クエリがキャッシュされていない限り、新しいクエリがコンソールに表示されます。

```
  "steps": {
    "step_Account_Name_1": {
      "isFacet": false,
      "query": {
        "pigql": "q = load \"opp\";\nq = filter q by 'Account-Name' in {{
selection(step_Account_Owner_Name_2) }};\nq = group q by {{
single_quote(value(selection(step_StageName_3))) }};\nq = foreach q generate {{
single_quote(value(selection(step_StageName_3))) }} as {{ value(selection(step_StageName_3))
 }}, sum('Amount') as 'sum_Amount', count() as 'count'",
```

```
        "groups": "{{ selection(step_StageName_3) }}",
        "measures": [["sum", "Amount"]]
      },
      "visualizationParameters": {
        "visualizationType": "hbar"
      },
      "selectMode": "none",
      "useGlobal": true,
      "datasets":[{"name":"opp"}],
      "type": "aggregate",
      "isGlobal": false
    },
    "step_Account_Owner_Name_2": {
      "dimensions": [ "Account-Name" ],
      "isFacet": false,
      "values": [
        {
          "value": ["Lakeside Med", "Hospital at Gulfport", "Hospital at Carbondale"],
          "display": "Arbuckle Laboratories, Arbuckle Laboratories - Austria, Arbuckle
Laboratories - France"
        }, {
          "value": ["Health University Med"],
          "display": "Health University Med"
        }, {
          "value": ["Canson"],
          "display": "Canson"
        }, {
          "value": ["ComputeWise"],
          "display": "ComputeWise"
        }, {
          "value": ["Dixon Chemical", "Dixon Chemical - Spain"],
          "display": "Dixon Chemical, Dixon Chemical - Spain"
        }, {
          "value": ["EarthNet"],
          "display": "EarthNet"
        }, {
          "value": ["Ecotech - Germany", "Ecotech - HQ"],
          "display": "Ecotech - Germany, Ecotech - HQ"
        }
      ],
      "selectMode": "multi",
      "useGlobal": true,
      "datasets":[{"name":"opp"}],
      "type": "static",
      "isGlobal": false
    },
    "step_StageName_3": {
      "isFacet": false,
      "values": [
        {
          "value": ["Account-Name"],
          "display": "Account"
        }, {
          "value": ["Product"],
```

```
          "display": "Product"
        }
      ],
      "useGlobal": true,
      "datasets":[{"name":"opp"}],
      "type": "static",
      "selectMode": "singlerequired",
      "isGlobal": false
    }
  }
```

日付ピッカーおよび静的日付のバインド

選択バインドを使用して、日付ピッカーレンズまたは静的な絶対/相対日付ステップから日付のレンズを絞り込むことができます。

次の例に、別のクエリおよび静的な相対日付ステップを絞り込む日付ピッカーレンズを別のクエリにバインドする方法を示します。

コンパクトおよび SAQL クエリへの日付ピッカーのバインド

この例では、日付ピッカーレンズで selection() **バインドを使用して時間グラフレンズを絞り込みます。日付ピッカーのレンズは、次のとおりです。**

```
"step_for_datePicker": {
      "type": "aggregate",
      "datasets":[{"name":"opp"}],
      "query": {
       "groups": [
        [
         "CloseDate_Year",
         "CloseDate_Month"
        ]
       ],
       "measures": [
        [
         "count",
         "*"
        ]
       ],
       "limit": 50
      },
      "start": [
       [
        [
         "year",
         -3
        ],
        [
         "year",
         1
        ]
```

```
        ]
       ]
      },
```

日付ピッカーの選択内容によって別のレンズを絞り込むには、次のコードをコンパクトまたは SAQL ステップに追加します。

```
{{selection(step_for_datePicker)}}
```

コンパクトフォームは次のようになります。

```
"step_compact_filtered_by_date_saql": {
      "type": "aggregate",
      "datasets":[{"name":"OpportunityWithAccount"}],
      "query": {
       "groups": [
        [
         "CloseDate_Year",
         "CloseDate_Month"
        ]
       ],
       "measures": [
        [
         "count",
         "*"
        ]
       ],
       "filters": [
        [
         "CloseDate",
         "{{ selection(step_for_datePicker) }}"
        ]
       ],
       "limit": 50
      }
     }
```

SAQL は次のようになります。

```
"step_date_saql_binding": {
      "type": "aggregate",
      "query": {
      "pigql": "q = load \"OpportunityWithAccount\";\nq = filter q by date('CloseDate_Year',
 'CloseDate_Month', 'CloseDate_Day') in {{selection(step_for_datePicker)}};\nq = group q
by ('CloseDate_Year', 'CloseDate_Month');\nq = foreach q generate 'CloseDate_Year' + \"~~~\"
 + 'CloseDate_Month' as 'CloseDate_Year~~~CloseDate_Month', count() as 'count';\nq = limit
 q 2000;",
       "groups": [
        [
         "CloseDate_Year",
         "CloseDate_Month"
        ]
       ],
       "measures": [
        [
```

```
          "count",
          "*"
         ]
        ]
       },
       "isFacet": false,
       "useGlobal": true
      }
     }
```

メモ: 選択内容で絞り込まれる日付ディメンション (この例では "CloseDate") は、日付ピッカーレンズの "groups" で使用されるディメンションと同じ名前である必要があります。

他のコンパクトまたは SAQL レンズを絞り込むための静的日付リストセレクタのバインド

この例では、定義済みの日付範囲のリストまたは切り替えレンズの選択内容によって、ダッシュボードで別のレンズを絞り込みます。次のサンプルでは、コンパクトフォーム ("compact_step_faceted_by_static") または SAQL ("saql_step_faceted_by_static") で静的切り替えボタンレンズ ("step_date_static_with_start") から棒グラフレンズにバインドする selection() を示します。各値は、5 年前 ("year", -5)、今年 ("year", 0) までなど、相対的な日付範囲です。

```
"step_date_static_with_start": {
      "type": "static",
      "values": [
       {
        "display": "-6 years",
        "value": [
         [
          [
           "year",
           -6
          ],
          [
           "year",
           0
          ]
         ]
        ]
       },
       {
        "display": "-5 years",
        "value": [
         [
          [
           "year",
           -5
          ],
          [
           "year",
           0
          ]
```

```
          ]
         ]
        },
        {
         "display": "-4 years",
         "value": [
          [
           [
            "year",
            -4
           ],
           [
            "year",
            0
           ]
          ]
         ]
        }
       ],
       "selectMode": "singlerequired",
       "start": [
        [
         [
          "year",
          -5
         ],
         [
          "year",
          0
         ]
        ]
       ]
      }
```

次に、上記のサンプルで selection() **バインドを使用して、選択内容で別のコンパクトまたは** SAQL **ステップを絞り込むことができます。**

```
{{selection(step_date_static_with_start)}}
```

コンパクトフォームは次のようになります。

```
"compact_step_faceted_by_static": {
      "type": "aggregate",
      "datasets":[{"name":"opp"}],
      "query": {
       "groups": [
        "Product"
       ],
       "filters": [
        [
         "CreatedDate",
         "{{selection(step_date_static_with_start)}}"
        ]
```

```
        ],
        "measures": [
         [
          "sum",
          "Amount"
         ]
        ],
        "limit": 2000
       },
       "isFacet": false
      }
```

SAQL 選択バインドは次のようになります。

```
"saql_step_faceted_by_static": {
       "type": "aggregate",
       "query": {
        "pigql": "q = load \"opp\";\nq = filter q by date('CreatedDate_Year',
'CreatedDate_Month', 'CreatedDate_Day') in {{selection(step_date_static_with_start)}};\nq
 = group q by 'Product';\nq = foreach q generate 'Product' as 'Product', sum('Amount') as
 'sum_Amount', count() as 'count';\nq = limit q 2000;",
        "groups": [
         "Product"
        ],
        "measures": [
         [
          "sum",
          "Amount"
         ]
        ]
       },
       "isFacet": false,
       "useGlobal": true
      },
```

バインド操作

いくつかの追加操作で結果バインドと選択バインドを使用して、正しい結果を抽出できます。

value()

`value()` 操作は、セレクタ配列値を取得して単一値に変換するために使用されます。セレクタ配列値が空白の場合、操作はすべての値を返します。value() 操作はセレクタ配列値が空白の場合に複数の値を返す可能性があるため、次の例のように、`==` ではなく `in` を使用します。

```
q = filter q by 'Owner Name' in {{ value(selection(step_StageName_3))}}
```

single_quote()

`single_quote()` 操作は通常、クエリの `"group"` および `"foreach generate"` 行が正しい形式になるように SAQL ステップの選択バインドで使用されます。`single_quote()` 操作は、値の配列を取得し、二重引用符を

単一引用符、角括弧を丸括弧に変換します。たとえば、`"Owner-Name"` は `'Owner-Name'` に変換され、`["Owner-Name", "Owner-Region"]` は `('Owner-Name', 'Owner-Region')` に変換されます。

配列値 `["Account-Name"]` および `["Product"]` を使用する次の静的セレクタがあるとします。

```
{
   "step_StageName_3": {
       "isFacet": false,
       "values": [
           {
               "value": [
                   "Account-Name"
               ],
               "display": "Account"
           },
           {
               "value": [
                   "Product"
               ],
               "display": "Product"
           }
       ],
       "useGlobal": true,
       "datasets":[{"name":"opp"}],
       "type": "static",
       "selectMode": "singlerequired",
       "isGlobal": false
   }
}
```

次の例は、`"group by"` 値と `"foreach generate"` 値で単一引用符を使用する必要がある SAQL クエリに配列値をバインドしています。したがって、`single_quote()` によって `["Account-Name"]` が `'Account-Name'` に変換されます。

```
{
   "step_Account_Name_1": {
       "isFacet": false,
       "query": {
           "pigql": "q = load \"opp\";\nq = group q by
               {{ single_quote(value(selection(step_StageName_3))) }};\nq =
               foreach q generate {{ single_quote(value(selection(step_StageName_3)))
               }} as {{ single_quote(value(selection(step_StageName_3)) }},
               sum('Amount') as 'sum_Amount', count() as 'count'",
           "groups": "{{ selection(step_StageName_3) }}",
           "measures": [
               [
                   "sum",
                   "Amount"
               ]
           ]
       },
       "visualizationParameters": {
           "visualizationType": "hbar"
       },
       "selectMode": "none",
```

```
        "useGlobal": true,
        "datasets":[{"name":"opp"}],
        "type": "aggregate",
        "isGlobal": false
    }
}
```

結果のクエリは次のとおりです。

```
q = load "opp";\nq = group q by 'Account-Name';\nq =
                foreach q generate 'Account-Name' as 'Account-Name', sum('Amount') as
                'sum_Amount', count() as 'count'
```

no_quote()

no_quote() 操作は通常、クエリの "order" 行が正しい形式になるように SAQL ステップの選択バインドで使用されます。no_quote() 操作は、値の配列を取得し、二重引用符と角括弧を引用符なしに変換します。たとえば、["desc"] は desc に変換されます。

["desc"] 配列値と ["asc"] 配列値が指定された次の静的ステップがあるとします。

```
{
    "step_order": {
        "type": "static",
        "values": [
            {
                "display": "desc",
                "value": [
                    "desc"
                ]
            },
            {
                "display": "asc",
                "value": [
                    "asc"
                ]
            }
        ],
        "selectMode": "singlerequired"
    }
}
```

次の例は、配列値を SAQL ステップにバインドしています。

```
q = order q by 'Amount' {{ no_quote(value(selection(step_order))) }}
```

引用符のない desc 値または asc 値が挿入されます。

```
q = order q by 'Amount' desc
```

field()

field() 操作は、配列内の各オブジェクトに対して項目を作成します。

次の静的ステップ(step_measure)では、"$" オプションと "#" オプションに "compact"、"alias"、および "proj" の3つの項目値が割り当てられています。

```
{
    "step_measure": {
        "type": "static",
        "values": [
            {
                "display": "$",
                "value": [
                    {
                        "compact": [["sum", "Amount"]],
                        "alias": "sum_Amount",
                        "proj": "sum('Amount')"
                    }
                ],
                "display": "#",
                "value": [
                    {
                        "compact": [["count", "*"]],
                        "alias": "count",
                        "proj": "count()"
                    }
                ]
            }
        ],
        "selectMode": "singlerequired"
    }
}
```

割り当て後、field() 操作を使用して他のステップ選択バインドで各項目値を参照できます。

たとえば、step_measure を使用する切り替えセレクタでダッシュボードユーザが [#] をクリックすると、この集計ステップ(step_top_10)の SAQL クエリは、"proj" 項目を参照して count() 関数を挿入し、"alias" 項目を参照して "count" を文字列として挿入し、"compact" 項目を参照して [["count", "*"]] を挿入します。

```
{
  "step_top_10": {
    "type": "aggregate",
    "datasets":[{"name":"opp"}],
    "query": {
      "pigql":
       "q = load 'edgemarts/Opportunity/OpportunityEM';
       q = group q by 'Account_Name';
       q = foreach q generate
         'Account_Name' as 'Account_Name',
         {{ no_quote(value(field(selection(step_measure),'proj'))) }}
           as {{ single_quote(value(field(selection(step_measure), 'alias'))) }};
       q = order q by {{ single_quote(value(field(selection(step_measure), 'alias'))) }}
         {{ no_quote(value(field(selection(step_order), 'pigql'))) }};
       q = limit q {{ value(selection(step_limit)) }};",
      "groups": ["Account_Name"],
      "measures": "{{ value(field(selection(step_measure), 'compact')) }}",
      "order":
```

```
        [[ -1, { "ascending": "{{ value(field(selection(step_order), 'compact')) }}" } ]]
    },
  "isFacet": true
  }
}
```