### join

入力として受け取ったデータのフィールドとサブクエリ結果のフィールドを比較し、結合（join）します。

joinはストリームクエリをサポートしていません。ストリームクエリで結合を適用する場合は、streamjoinコマンドを使用してください。

#### 構文

join [type={cross|full|inner|left|leftonly|right|rightonly}] KEY\_FIELD, ... [ SUBQUERY ]

**type={cross|full|inner|left|leftonly|right|rightonly}**

結合タイプを指定します（デフォルト: inner）。

1. cross: 「直積（カーティジアン積）」とも呼ばれます。入力データセット（M件）とサブクエリ結果セット（N件）を結合し、M×N件のレコードからなる集合を出力します。cross結合を使用する場合、KEY\_FIELDは指定しません。
2. full: キーが一致するレコードはマージし、一致しないレコードはそのまま出力します。データの和集合に類似しています。
3. キーが一致する場合、入力データレコードにサブクエリのフィールドを結合して出力します。
4. キーが一致しない場合、入力データとサブクエリデータをそれぞれそのまま出力します。
5. inner: キーが一致するレコードのみを結合して出力します。キーを含まないレコードは出力しません。データの積集合に類似しています。一般的に「join」とは「inner join」を指します。
6. left: キーが一致するレコードは結合して出力し、一致しないレコードは入力データのレコードのみを出力します。
7. leftonly: 入力レコードセットのうち、サブクエリ結果セットとキーが一致しないレコードのみを出力します。キーが一致するレコードは出力しません。
8. right: キーが一致するレコードは結合して出力し、一致しないレコードはサブクエリ結果のみを出力します。
9. rightonly: サブクエリ結果セットのうち、入力レコードセットとキーが一致しないレコードのみを出力します。キーが一致するレコードは出力しません。

**KEY\_FIELD, ...**

結合の基準となるキー・フィールド。フィールドの区切りにはカンマ（,）を使用します。type=crossの場合のみ、キー・フィールドの指定は不要です。

**[ SUBQUERY ]**

入力データと結合するデータを出力するクエリ文を角括弧（[ ]）内に記述します。

#### 使用例

**code**フィールドをキーにしたinner結合

# codeフィールドを持つjsonを入力データとして渡す | json "[{'code':1}, {'code':2}, {'code':3}]" | # code, nameフィールドを持つjsonを返すサブクエリ | # 入力データとサブクエリ結果をcodeフィールドでinner join | join code [ json "[ {'code':1, 'name':'foo'}, {'code':2, 'name':'bar'} ]" ]

**クエリ実行結果（inner join）**

|  |  |
| --- | --- |
| code | name |
| 1 | foo |
| 2 | bar |

サブクエリで取得された結果を除外して出力（leftonly結合）

json "[ {'field1': 'A'}, {'field1': 'B'}, {'field1': 'C'}, {'field1': 'D'}]" | join type=leftonly field1 [ json "[ {'field1': 'A', 'field2': 'Foo'}, {'field1': 'D', 'field2': 'Bar'} ]" ]

**クエリ実行結果（leftonly join）**

|  |
| --- |
| field1 |
| B |
| C |

上記のクエリは、次のクエリ（left結合の応用）と同じ結果になります。

json "[ {'field1': 'A'}, {'field1': 'B'}, {'field1': 'C'}, {'field1': 'D'}]" | join type=left field1 [ json "[ {'field1': 'A', 'field2': 'Foo'}, {'field1': 'D', 'field2': 'Bar'} ]" ] | search isnull(field2)

部署ごとの統計を出力し、統計値が存在しなくてもすべての部署を出力（right結合）

json "[{'id': 1, '件数': 1000}, {'id':2, '件数': 2000}]" | join type=right id [ json "[ {'id':1, '部署':'営業'}, {'id':2, '部署':'運用'}, {'id':3, '部署':'技術'} ]" ]

**クエリ実行結果（right join）**

|  |  |  |
| --- | --- | --- |
| id | 部署 | 件数 |
| 1 | 営業 | 1000 |
| 2 | 運用 | 2000 |
| 3 | 技術 |  |

文書セキュリティ違反とメディア制御違反ログをユーザー基準で結合し、一致しない場合は各ログを出力（full結合）

json "[ {'ユーザー':'bob', '文書セキュリティ違反': 1}, {'ユーザー':'alice', '文書セキュリティ違反': 5} ]" | join type=full ユーザー [ json "[ {'ユーザー':'alice', 'メディア制御違反': 8}, {'ユーザー':'clark', 'メディア制御違反': 3} ]" ]

**クエリ実行結果（full join）**

|  |  |  |
| --- | --- | --- |
| ユーザー | 文書セキュリティ違反 | メディア制御違反 |
| bob | 1 |  |
| alice | 5 | 8 |
| clark |  | 3 |

#### 互換性

typeのうち、crossおよびrightonlyはENT-3.0.2003.0バージョン以降で利用可能です。