OpenDataHub Query Language (ODHQL)

Inhalt

- UPPER(strings)
- XPATH(values, path)

Was ist ODHQL?

ODHQL ist eine an SQL angelehnte Abfrage- und Transformations-Sprache für OpenDataHub.

Syntax

Bestandteile einer Abfrage

Eine Abfrage besteht aus folgenden Teilen:

- Eine Liste von Feldern oder Ausdrücken, welche im Resultat erscheinen sollen
- Eine Liste von Datenquellen
- Optional eine Liste von Filter-Ausdrücken
- Optional eine Sortier-Klausel

Gross- und Kleinschreibung wird nicht beachtet.

Mehrere Abfragen können kombiniert werden mithilfe von Union. In diesem Fall ist nur eine Sortier-Klausel am Ende der kombinierten Abfrage erlaubt.

```
SELECT NULL AS userid,
           -- Null-Ausdruck
       SUBSTRING(NVL(EXTRACT(t.text, '\|([^|\.]+)'), 'no value'), 1, 100)
AS title,  -- Funktions-Aufruf
       EXTRACT(t.text, '\|([^|]+)') AS description,
       CAST(CAST(t."df", 'bigint'), 'datetime') AS trob_start,
       CAST(CAST(t."dd", 'bigint'), 'datetime') AS trob_end,
       NULL AS trob_interval,
       'both' AS direction,
           -- String-Ausdruck
       NULL AS diversion_advice,
       CASE WHEN t.p = 'Switzerland' THEN 'CH'
           -- Case-Ausdruck
            WHEN t.p = 'France' THEN 'FR'
            WHEN t.p = 'Austria' THEN 'AT'
       END AS country,
       t.text AS reason,
           -- Feld
       EXTRACT(t.text, '\|([^|,]+)') AS object_name,
       'street' AS object_type,
       CASE WHEN CONTAINS(t.text, 'closed', FALSE) THEN 'closed'
            WHEN (CONTAINS(t.text, 'maintenance', FALSE) OR CONTAINS(t.tex
t, 'maintenance', FALSE))
                THEN 'obstructed'
            ELSE 'other'
       END AS trob_type,
       ST_SETSRID(ST_GEOMFROMTEXT(CONCAT('POINT(', t.x, ' ', t.y, ')')), 3
395) AS trobdb_point
```

```
 FROM ODH11 AS t
          -- Datenquelle
ORDER BY 4
```

Felder und Ausdrücke

In der Feld-Liste sind folgende Ausdrücke erlaubt:

Feld

Bezieht sich direkt auf ein Feld einer Datenquelle. Der Name des Feldes muss mit dem Namen oder Alias der Datenquelle prefixed werden. Optional kann ein Alias angegeben werden.

```
prefix.feld AS alias
```

Wert

Ganzzahl (Integer), Fliesskommazahl (Float, Trennzeichen ist ein Punkt), Zeichenkette (String, in einfachen Anführungszeichen), Boolean (true, false) oder Null. Es muss ein Alias angegeben werden.

Einfache Anführungszeichen können innerhalb eines Strings mit "\'" erzeugt werden, Backslashes ("\") mit "\\".

Funktion

Besteht aus einem Namen und einer Liste von Argumenten. Es muss zwingend ein Alias angegeben werden.

```
SUBSTRING(NVL(EXTRACT(t.text, '\|([^|\.]+)'), 'no value'), 1, 100) AS titl
e
```

Fallunterscheidung (Case-Ausdruck)

Kann verwendet werden, um Werte zu übersetzen. Es muss mindestens eine Bedingung angegeben werden. Das Format ist 'when <Bedingung> then <Ausdruck>', wobei alle unten beschriebenen Bedingungs-Arten sowie hier beschriebenen Ausdrücke erlaubt sind.

```
CASE WHEN CONTAINS(t.text, 'closed', FALSE) THEN 'closed'
     WHEN (CONTAINS(t.text, 'maintenance', FALSE) OR CONTAINS(t.text, 'mai
ntenance', FALSE))
         THEN 'obstructed'
     ELSE 'other'
END
```

Datenquellen

Es muss mindestens eine Datenquelle angegeben werden. Falls mehrere Datenquellen verwendet werden sollen, muss eine Verknüpfungsbedingung angegeben werden.

Unterstützt werden folgende Join-Typen:

Inner

Standard. Verlangt, dass beide Seiten vorhanden sind

```
FROM ODH12 AS employees
JOIN ODH13 AS employers ON employees.employer_id = employers.id
```

Left

Verwendet die Schlüssel der linken Seite für den Vergleich (rechte Seite kann null sein)

```
FROM ODH12 AS employees
LEFT JOIN ODH13 AS employers ON employees.employer_id = employers.id
```

Right
Verwendet die Schlüssel der rechten Seite für den Vergleich (linke Seite kann null sein)

```
FROM ODH12 AS employees
RIGHT JOIN ODH13 AS employers ON employees.employer_id = employers.id
```

FULL
Verwendet die Vereinigungsmenge der Schlüssel der beiden Seiten für den Vergleich (beide Seiten sind optional, es kann jedoch pro Zeile nur eine Seite null sein).

```
FROM ODH12 AS employees
FULL JOIN ODH13 AS employers ON employees.employer_id = employers.id
```

Filter

Folgende Filter-Ausdrücke sind möglich:
is null*, *is not null
Prüft ob ein Feld (nicht) null ist

in*, *not in
Prüft ob ein Ausdruck (nicht) in einer Liste enthalten ist. .. code:: sql

> country IN ('CH', 'DE', 'AT')

<, >, <=, >=, =, !=
Vergleicht zwei Ausdrücke miteinander.

like*, *not like
Prüft ob ein Ausdruck einem Muster entspricht. Verwendet wird dazu ein Regulärer Ausdruck mit Python Syntax.

Prädikat
Eine Funktion, welche ein boolsches Resultat liefert kann direkt als Bedingung verwendet werden.

Mehrere Bedingungen können mit 'and' und 'or' verknüpft und mit runden Klammern gruppiert werden.

```
WHERE t.a IS NOT NULL
  AND (t.b IN (1, 2, 3) OR t.b > 20)
```

Sortier-Klausel

Es kann sortiert werden nach Feld-Name, Alias oder Position in der Feld-Liste.

```
ORDER BY 1, ODH4.name DESC, surname ASC
```

Union

Die Resultate mehrerer Abfragen können mithilfe von Union kombiniert werden. Zu beachten sind folgende Punkte: - Union verhält sich wie UNION ALL in SQL, d.h. es wird keine Deduplizierung der Einträge vorgenommen - Die einzelnen Abfragen müssen kompatible Feldlisten liefern, d.h. gleiche Feld-Zahl und Feld-Typen. - Sollten Geometrie-Spalten vorhanden sein, so werden die Geometrien in das Referenzsystem der ersten Abfrage umgewandelt. Es ist somit nötig, dass für alle Geometrien ein Referenzsystem bekannt ist.

Als Feld-Namen werden im Resultat die Feld-Namen der ersten Abfrage verwendet.

Datentypen

ODHQL unterstützt zurzeit folgende Datentypen:

> **INTEGER**
> 32-Bit Ganzzahl-Wert
>
> **BIGINT**
> 64-Bit Ganzzahl-Wert
>
> **SMALLINT**
> 16-Bit Ganzzahl-Wert
>
> **FLOAT**
> 64-Bit Fliesskommazahl
>
> **DATETIME**
> Datum und Zeit
>
> **TEXT**
> Zeichenkette beliebiger Länge
>
> **GEOMETRY**
> Geometrie-Objekte. Nicht näher spezifiziert, d.h. es können Geometrien verschiedener Art in einer Spalte vorhanden sein. Der Experte ist selbst dafür Verantwortlich, dass gleiche Typen bei einer Transformation herauskommen, sodass diese für alle Zielformate kompatibel ist.

Hierbei ist wichtig anzumerken, dass diese Typen keine Keywords der Sprache sind, sondern beispielsweise bei der Verwendung der CAST() Funktion als Zeichenkette (z.B. 'INTEGER') übergeben werden müssen.

Funktionen

CAST(values, datatype)

Konvertiert den Datentyp einer Spalte oder eines einzelnen Wertes.

Parameter

- *values*: Spalte oder Wert der konvertiert werden soll.
- *datatype*: Gültiger ODHQL Datentyp

Beispiel

```
CAST(ODH42.age, 'INTEGER') AS age
```

CONCAT(a, b, ...args)

Konkateniert eine Liste von TEXT-Spalten oder Werten.

Parameter

- *args*: Liste von TEXT-Spalten oder -Werten

Beispiel

```
CONCAT(ODH5.given_name, ' ', ODH5.surname) AS name
```

CONTAINS(strings, pattern, match_case=True)

Prüft ob eine Zeichenkette in einem Text enthalten ist.

Kann als Bedingung verwendet werden.

Parameter

- *strings*: Spalte oder Wert in welchem gesucht werden soll
- *pattern*: Spalte oder Wert welcher gesucht werden soll
- *match_case*: Optional. Gibt an ob Gross- und Kleinschreibung beachtet werden soll. Default: True.

Beispiel

```
CONTAINS(ODH9.haltestelle, 'Hauptbahnhof') AS ist_hb
```

COUNT(strings, pattern)

Zählt die Anzahl Vorkommnisse des Musters im Text.

Parameter

- *strings*: Spalte oder Wert
- *pattern*: Regulärer Ausdruck. Siehe Regex-Dokumentation.

Beispiel

```
COUNT(ODH30.description, '\d') AS numbers
```

ENDSWITH(strings, end)

Prüft ob ein Text mit einer angegebenen Zeichenkette endet.

Kann als Bedingung verwendet werden.

Parameter

- *strings*: TEXT-Spalte oder -Wert
- *end*: TEXT-Spalte oder -Wert

Beispiel

```
ENDSWITH(ODH9.haltestelle, 'Flughafen') AS ist_flughafen
```

EXTRACT(strings, pattern, group=1)

Wendet einen Regulären Ausdruck (Regex) auf die angegebene Spalte oder den Wert an und liefert das Resultat der angegebenen Gruppe zurück.

Da Backslashes ('')

Parameter

- *strings*: TEXT-Spalte oder -Wert
- *pattern*: Regulärer Ausdruck. Siehe Regex-Dokumentation.
- *group*: Optional. Gruppe, welche ausgewertet werden soll. Default: 1.

Beispiel

```
EXTRACT(t.text, '\|([^|\.]+)') AS title
```

GET(strings, index)

Liefert das Zeichen an der angegebenen Stelle im Text (0-basiert).

Parameter

- *strings*: Spalte oder -Wert
- *index*: Spalte oder Wert.

Beispiel

```
GET(ODH12.country, 1) AS c
```

LEN(strings)

Liefert die Länge des TEXTs.

Parameter

- *strings*: TEXT-Spalte oder -Wert

Beispiel

```
LEN(ODH7.description) AS description_length
```

LOWER(strings)

Konvertiert alle Buchstaben in Kleinbuchstaben.

Parameter

- *strings*: TEXT-Spalte oder -Wert

Beispiel

```
LOWER(ODH7.ort) AS ort
```

LTRIM(strings)

Entfernt White Space vom Beginn der Spalte oder des Wertes.

Parameter

- *strings*: TEXT-Spalte oder -Wert

Beispiel

```
LTRIM(ODH7.strasse) AS strasse
```

NVL(a, b)

Gibt das zweite Argument zurück, falls das erste NULL ist.

Parameter

- *a*: Spalte oder Wert der auf NULL geprüft werden soll
- *b*: Spalte oder Wert der als Ersatz verwendet werden soll

Beispiel

```
NVL(ODH12.title, '') as title
```

PAD(strings, width, side)

Füllt die Zeichenkette auf die angegebene Länge mit Leerzeichen auf.

Parameter

- *strings*: Spalte oder Wert
- *width*: Gewünschte Länge
- *side*: Seite. Kann 'left', 'right' oder 'both' sein.

Beispiel

```
PAD(ODH4.name, 20, 'right') AS name
```

RANGE(start=1, step=1)

Liefert eine Sequenz von Ganzzahlen. Geeignet um beispielsweise künstliche IDs zu erstellen.

Parameter

- *start*: Erster Wert der Sequenz.
- *step*: Abstand zwischen den Ganzzahlen.

Beispiel

```
RANGE() AS id
```

REPEAT(strings, times)

Wiederholt die Zeichenkette eine bestimme Anzahl mal.

Parameter

- *strings*: Spalte oder Wert welcher wiederholt werden solle
- *times*: Anzahl Wiederholungen

Beispiel

```
REPEAT('Lorem ipsum dolor... ', 20) AS filler
```

REPLACE(strings, pattern, replace, match_case=True)

Ersetzt die angegebene Zeichenkette im Text.

Parameter

- *strings*: Spalte oder -Wert in welchem gesucht werden soll
- *pattern*: Spalte oder -Wert welcher gesucht werden soll
- *replace*: Ersatz-Spalte oder -Wert
- *match_case*: Optional. Gibt an ob Gross- und Kleinschreibung beachtet werden soll. Default: True.

Beispiel

```
REPLACE(ODH12.strasse, 'str.', 'strasse') AS strasse
```

ROUND(col, decimals)

Rundet auf die angegebene Anzahl Nachkommastellen.

Parameter

- *col*: Spalte oder Wert der gerundet werden soll. Muss vom Datentyp FLOAT sein.
- *decimals*: Anzahl Nachkommastellen, auf die gerundet werden soll.

Beispiel

```
ROUND(ODH20.fraction, 4) AS fraction
```

RTRIM(strings)

Entfernt White Space vom Ende der Spalte oder des Wertes.

Parameter

- *strings*: TEXT-Spalte oder -Wert

Beispiel

```
RTRIM(ODH7.strasse) AS strasse
```

ST_Area(geoms)

Berechnet die Fläche der Geometrien in der Einheit des Koordinatensystems.

Parameter

- *geoms*: Spalte mit Geometrien.

Siehe auch: PostGIS ST_Area

Beispiel

```
ST_X(ODH12.geometry) AS area
```

ST_AsText(geoms)

Liefert die Geometrien als Well-Known-Text (WKT) Zeichenkette,

Parameter

- *geoms*: Spalte mit Geometrien.

Siehe auch: PostGIS ST_AsText

Beispiel

```
ST_AsText(ODH12.geometry) AS wkt
```

ST_Centroid(geoms)

Berechnet den Mittelpunkt der Geometrien.

Parameter

- *geoms*: Spalte mit Geometrien.

Siehe auch: PostGIS ST_Centroid

Beispiel

```
ST_Centroid(ODH12.geometry) AS point
```

ST_GeomFromText(wkts, srid=None)

Erstellt eine Spalte mit Geometrie-Objekten aus einem Well-Known-Text (WKT).

Parameter

- *wkts*: Spalte oder Wert mit WKTs.
- *srid*: Optional die SRID der Geometrien.

Siehe auch: PostGIS ST_GeomFromText

Beispiel

```
ST_GeomFromText('POINT(7.2234283 48.8183157)', 4326) AS hsr
```

ST_SetSRID(geoms, srid)

Setzt das Koordinatensystem (Spatial Reference Identifier, kurz SRID) einer Geometrie-Spalte.

Parameter

- *geoms*: Spalte mit Geometrien.
- *srid*: SRID der Geometrien.

Siehe auch: PostGIS ST_SetSRID

Beispiel

```
ST_SetSRID(ODH12.latlng, 4326) AS geometry
```

ST_SRID(geoms)

Liefert das Koordinatensystem einer Geometrie-Spalte. Sollte keine Identifikationsnummer (SRID) bekannt sein, so wird eine Zeichenkette mit den Projektionsparametern im PROJ.4 <http://trac.osgeo.org/proj/>`_ Format zurückgegeben.

Parameter

- *geoms*: Spalte mit Geometrien.

Siehe auch: PostGIS ST_SRID

Beispiel

```
ST_SRID(ODH12.latlng) AS srid
```

ST_Transform(geoms, srid)

Transformiert die Geometrien in ein anderes Koordinatensystem. Auf den Geometrien muss bereits eine SRID gesetzt sein.

Parameter

- *geoms*: Spalte mit Geometrien.
- *srid*: SRID der Geometrien.

Siehe auch: PostGIS ST_Transform

Beispiel

```
ST_Transform(ODH12.mercator, 4326) AS latlng
```

ST_X(geoms)

Liefert die kleinste X-Komponente der Geometrien. **Achtung:** Funktioniert aus diesem Grund anders wie in

PostGIS nicht nur mit Punkten.

Parameter

- *geoms*: Spalte mit Geometrien.

Siehe auch: PostGIS ST_X

Beispiel

```
ST_X(ODH12.latlng) AS x
```

ST_Y(geoms)

Liefert die kleinste Y-Komponente der Geometrien. **Achtung:** Funktioniert aus diesem Grund anders wie in PostGIS nicht nur mit Punkten.

Parameter

- *geoms*: Spalte mit Geometrien.

Siehe auch: PostGIS ST_Y

Beispiel

```
ST_Y(ODH12.latlng) AS y
```

STARTSWITH(strings, start)

Prüft ob ein Text mit einer angegebenen Zeichenkette anfängt.

Kann als Bedingung verwendet werden.

Parameter

- *strings*: TEXT-Spalte oder -Wert
- *start*: TEXT-Spalte oder -Wert

Beispiel

```
STARTSWITH(ODH7.ort, 'Zürich') AS in_zurich
```

SUBSTRING(strings, start, length=None)

Liefert den angegebenen Teil des Texts.

Parameter

- *strings*: Spalte oder Wert
- *start*: Startposition (startet bei 1)
- *length*: Gewünschte Länge. Ohne Längenangabe wird der Rest des Texts zurückgegeben.

Beispiel

```
SUBSTRING(ODH30.description, 1, 100) AS title
```

TO_CHAR(values, format=None)

Konvertiert eine Spalte oder einen Werte zu TEXT. Für DATETIME-Spalten/Werte kann ein Format angegeben werden.

Parameter

- *values*: Spalte oder Wert
- *format*: Falls *values* vom Typ DATETIME ist. Siehe Dokumentation.

Beispiel

```
TO_CHAR(TO_DATE(ODH30.baubeginn, '%d%m%Y'), '%Y-%m-%d') AS baubeginn
```

TO_DATE(values, format=None)

Konvertiert ein Datum in TEXT-Form zu DATETIME.

Parameter

- *values*: Spalte oder Wert der konvertiert werden soll.
- *format*: Format-Angabe. Siehe Dokumentation.

Beispiel

```
TO_DATE(ODH5.baubeginn, '%d%m%Y') AS baubeginn
```

TRIM(strings)

Entfernt White Space vom Beginn und Ende der Spalte oder des Wertes.

Parameter

- *strings*: TEXT-Spalte oder -Wert

Beispiel

```
TRIM(ODH7.strasse) AS strasse
```

UPPER(strings)

Konvertiert alle Buchstaben in Grossbuchstaben.

Parameter

- *strings*: TEXT-Spalte oder -Wert

Beispiel

```
UPPER(ODH7.ort) AS ort
```

XPATH(values, path)

Wendet den XPath-Ausdruck auf die Spalte oder den Wert an.

Parameter

- *values*: Spalte oder Wert mit gültigem XML in TEXT-Form.
- *path*: XPath-Ausdruck. Zu beachten: Der Ausdruck darf genau ein Resultat pro Zeile in *values* liefern.

Beispiel

```
XPATH(t.description, '//tr[1]/td[2]/text()') AS abschnitt
```