QueryProvider

Language: Deutsch · **English**

QueryProvider

> **Typed data queries in rules – create/where/and/orderBy/execute, getByState, getByProcess, subqueries.**

REFERENZ | DEVELOPER | UPDATED: JUL 2026 | APPLIES TO NUCLOS 4.2026.X

On this page

What is the QueryProvider for?

The `QueryProvider` loads data from Nuclos and provides it as a business object or list in rules. Instead of SQL it uses a **typed, abstract query language**.

> **Logically deleted entries**
>
> Without `NuclosLogicalDeleted`, logically deleted entries are **not** considered. Only `NuclosLogicalDeleted.eq(Boolean.TRUE)` returns deleted entries.

create(type)

Creates a typed `Query` object for a business object.

```
public static <T extends BusinessObject> Query<T> create(Class<T> type);
```

execute(query)

Runs the query and returns a typed, never-`null` list (empty if no hit).

```
public static <T extends BusinessObject> List<T> execute(Query<T> query);
```

get(id)

Reads a specific entry by type and id; `null` if nothing is found.

getByState / getByProcess

Search entries by **state** or by assigned **action** (process). At least one state/action is required.

Conditions & combining

Fields offer matching comparison operators: `eq`, `neq`, `isNull`, `notNull` (all types) plus `Gt`, `Gte`, `Lt`, `Lte` (numeric). `where()`/`and()`/`orderBy()` return the query and can be chained. `or()` exists only at the *SearchExpression* level, allowing nested blocks.

```
boolean sortAscending = true;

Query<Auftrag> queryAuftrag = QueryProvider.create(Auftrag.class);

queryAuftrag.where(Auftrag.Auftragsnr.notNull())
            .and(Auftrag.Bestellwert.Gt(BigDecimal.ZERO))
            .orderBy(Auftrag.Auftragsnr, sortAscending);
```

Execute & iterate

```
List<Auftrag> results = QueryProvider.execute(queryAuftrag);
for (Auftrag a :results) {
        BigDecimal bestellwert = a.getBestellwert();
}
```

Subqueries (exist)

With `exist(subQuery, …)` subqueries can be embedded via (foreign) keys; the subquery itself is not executed separately.

```
// Alle Kunden, fuer die Zahlungsbedingungen hinterlegt wurden
Query<Kunde> qryKunden = QueryProvider.create(Kunde.class);
qryKunden.where(Kunde.Zahlungsbedingung.notNull());

// Nun brauchen wir alle Bestellungen dieser Kunden
Query<Bestellung> queryBestellungen = QueryProvider.create(Bestellung.class);
queryBestellungen.where(Bestellung.Bestellnr.notNull())
                 .exist(qryKunden, Bestellung.KundeId)
                 .orderBy(Bestellung.Bestellnr, true);
List<Bestellung> results = QueryProvider.execute(queryBestellungen);
```

Related pages

BusinessObjectProvider

BOs schreiben.

Open

DatasourceProvider

run data sources.

Open

GenerationProvider

Arbeitsschritte.

Open