

Report & Formular

- [Definition](#)
- [Konfiguration](#)
 - [Beispiele für Bedingungen in Datenquellen](#)
 - [Beispiele für Oracle Funktionen in der Spaltendefinition](#)
 - [Beispiel für Oracle Datum is null Bedingung](#)
 - [Parameterdefinition](#)
 - [SQL-Statement bearbeiten](#)
 - [Syntax prüfen](#)
 - [Ergebnis Vorschau](#)

Definition

Menüaufruf: (Konfiguration) - (Datenquellen) - (Report & Formular)

Datenquellen bilden die Grundlage für Reports und Formulare.

Eine Datenquelle legt fest, welche Businessobjekte betrachtet werden sollen. Mit Hilfe von Bedingungen und Tabellenverknüpfungen können die zu betrachtenden Daten eingeschränkt bzw. um weitere Attribute leicht ergänzt werden.

Der Einsatz von Parametern erlaubt die Wiederverwendung von gespeicherten Datenquellen, die abhängig von den ausgefüllten Parameterwerten eine veränderte Sicht auf die Daten liefern.

Konfiguration

[blocked URL](#)

Die Pflegemaske für die Datenquellen gliedert sich in 5 Abschnitte:

Im oberen Teil werden der Name und die Beschreibung der Datenquelle eingegeben.

Die *Tabellenauswahl* enthält alle Businessobjekte, die systemseitig zur Verfügung gestellt werden.

- Im Abschnitt *Systemobjekte* sind vornehmlich Bewegungs- und Stammdaten enthalten. Welche Tabellen zur Auswahl stehen kann über den Systemparameter „DataSourceEditor Systemtables Statement“ konfiguriert werden.
- Der Abschnitt *Dynamische Objekte* enthält alle relevanten Businessobjekttabellen (im Beispiel: Bewerber, Mitarbeiter, Vorfallmeldung, usw.).
- Der Abschnitt *Gespeicherte Abfragen* enthält alle bereits definierten Abfragen, die als Unterabfragen verwendet werden können.

Aus der Tabellenauswahl können per Drag&Drop Businessobjekte in die *Tabellenansicht* gezogen werden.

Die *Parameterdefinition* listet alle für eine Datenquelle definierten Parameter auf. Neben dem Namen muss hier auch der Datentyp eines Parameters bekannt gegeben werden. Der Parameterbeschreibung wird der Text hinzugefügt, der dem Benutzer als Eingabeaufforderung angezeigt wird. Datenquellen für Formulare müssen immer mindestens den Parameter *intid* für das Feld *intid* der Businessobjekttabelle definieren, für die das Formular ausgeführt werden soll.

Die *Spaltenansicht* dient schließlich zur Auswahl der Spalten, die in einem Bericht oder Formular sichtbar sein sollen. Hierzu können die Spalten einfach in der Tabellenansicht per Drag&Drop in die gewünschte Position gezogen werden oder mittels Doppelklick übernommen werden. Die Reihenfolge der Spalten kann durch Ziehen der Spaltenköpfe verändert werden. Hierbei wird ein Aliasname vom System vorbelegt, der vom Benutzer überschrieben werden kann. Der Aliasname wird später für die Feldnamen in Reports und Formularen verwendet. In der Spaltenansicht werden auch die Sortierung der Daten festgelegt und Bedingungen an Spalten geknüpft (siehe nachfolgendes Beispiel).

[blocked URL](#)

Das Flag *Sichtbar* gibt an, ob eine Spalte in dem resultierenden Abfrageergebnis erscheint.

Nach dem Speichern der Datenquelle stehen zusätzlich noch die Funktionen *SQL* und *Vorschau* zur Verfügung, die eine Voransicht des Abfrageergebnisses ermöglichen.

Wird eine Abfrage mit Parametern ausgeführt, wird dem Benutzer ein Dialog angezeigt, in dem er die Werte zu den definierten Parametern eingeben kann:

Bei Parametern des Datentyps *Text* besteht die Möglichkeit zu steuern, ob der Benutzer Wildcards in diesem Parameter verwenden darf, muss oder nicht darf. Der Administrator der Datenquelle hat dazu mehrere Möglichkeiten einen Parameter vom Typ *Text* in der Bedingung der Spalte zu hinterlegen:

= '\$parametername' Der Benutzer hat keine Möglichkeit Wildcards zu verwenden. Es wird eine „*=*“ Bedingung im der *SQL* Abfrage erzeugt. Die vom Benutzer eingegebene Zeichenkette wird so wie eingegeben zum Vergleich herangezogen. Dadurch besteht die Möglichkeit nach Zeichenketten zu suchen, die Zeichen wie „*%*“ und „*_*“ enthalten.

WIE '\$parametername' Der Benutzer hat die Möglichkeit Wildcards zu verwenden. Es wird eine *LIKE* Bedingung in der *SQL* Abfrage erzeugt. Es besteht nicht die Möglichkeit explizit nach den Zeichen „*%*“ und „*_*“ zu suchen.

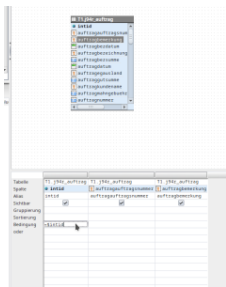
= '\$parametername%' Es wird immer mit Wildcards gesucht. Es wird eine „*LIKE*“ Bedingung in der *SQL* Abfrage erzeugt, der die dem Parameternamen folgende Wildcard angehängt wird. Gibt der Benutzer bei dem obigen Beispiel keinen Wert für den Parameter ein, werden alle Datensätze gefunden (so als gäbe es diese Bedingung nicht). Gibt der Benutzer „*Huber*“ ein wird nach „*Huber%*“ gesucht usw.

Der Parameter mit dem Namen *intid* hat eine besondere Bedeutung in den Datenquellen. Wird eine Datenquelle erstellt, die in einem Formular genutzt werden soll, wird über diesen Parameter die Referenz auf das zu druckende Objekt hergestellt. D.h. Datenquellen, die einem Formular zugrunde liegen, müssen immer den Parameter „*intid*“ für die Spalte *intid* der auszudruckenden Businessobjekt definieren. Nuclos füllt diesen Parameter beim Drucken aus den Businessobjektenmasken automatisch, so dass dem Benutzer kein entsprechender Dialog angezeigt wird.

Wird ein *Valuelistprovider* angegeben, so erscheint dem Benutzer eine Auswahlliste in einem Dropdown, mit den vom *VLP* gelieferten Werten. Über den angezeigten Textwert kann ganz Normal per '*\$parametername*' zugegriffen werden. Die *ID* des ausgewählten Wertes erhält man mittels '*\$parameternameId*'.

Interne Parameter Typen: **Text Liste, Integer Liste und SQL**

Parameter diesen Typs werden dem Benutzer nicht zur Eingabe angeboten, sie dienen der internen Verwendung in Regeln.



Parametername	Datentyp	Pflichtparam...	Anzeigename	Valuelist Provider
pListOfInteger	Intern: Integer Liste	☐		Kein ValueList Prov... ...
pListOfText	Intern: Text Liste	☐		Kein ValueList Prov... ...
pSql	Intern: SQL	☐		SQL Whitelist.ds ...

Beispiel Datenquelle

```
SELECT
  T1.INTID "<primaryKey>",
  T1.INTARTICLENUMBER "articleNumber",
  T1.STRNAME "name"
FROM
  V594_ARTICLE T1
WHERE
  $pSql
  OR
  T1.INTARTICLENUMBER in $pListOfInteger
  OR
  T1.STRNAME in $pListOfText
```

Beispiel Regel

```
List<Number> listOfInteger = new
ArrayList<>();
listOfInteger.add(4711L); // Test with Long
List<String> listOfStrings = new ArrayList<>();
listOfStrings.add("4711");

Map<String, Object> params = new HashMap<>();
params.put("pListOfInteger", listOfInteger);
params.put("pListOfText", listOfStrings);
params.put("pSql", "T1.INTARTICLENUMBER > 4000");

DatasourceProvider.run(TestDS.class, params);
```

Besonderheit des SQL Typs

Parameter vom Typ SQL müssen einen VLP definieren, der die erlaubten SQL "Erweiterungen" zurückliefert (Whitelist). Der Nuclos Server wird einen Wert gegen diese Liste prüfen und bei einer Verletzung einen Fehler werfen. Zur Validierung der Datenquelle setzt Nuclos für solch einen Parameter einfach den Wert "false" ein. Ist das für die Verwendung ungünstig, kann über einen kleinen Trick ein eigener Validierungsausdruck hinterlegt werden: Geben Sie im Anzeigenamen den gewünschten Ausdruck ein. Da diese Art von Parametern einem Benutzer nicht zur Eingabe angeboten werden, kann ein beliebiger Ausdruck verwendet werden.

Beispiele für Bedingungen in Datenquellen

In der Bedingung zu einer Spalte können sowohl einfache Vergleiche, als auch Ausdrücke eingegeben werden, die an Oracle weitergereicht werden. Im Allgemeinen gilt, Zeichenketten die in einfachen Anführungszeichen stehen werden von Nuclos ausgewertet. Alle anderen Zeichenketten werden weitergereicht und von Oracle interpretiert.

Alle eingegebenen Ausdrücke werden in den *WHERE* Teil der erzeugten Abfrage eingebaut.

Vergleichsoperatoren (=, >, <, >=, <=, !=, WIE)
z.B. >'100', ='Hans Mustermann', !='Düsseldorf', >='\$EndeDatum',
='1082%', WIE '1082%'

Wird der Operator '=' in einer Spalte mit dem Datentyp *Text* verwendet, wird automatisch eine LIKE Bedingung erzeugt, wenn die Bedingung eines der Zeichen '%' oder ' ' enthält. Mit dem Operator WIE kann man erzwingen, dass eine LIKE Bedingung erzeugt wird, auch ohne Wildcards zu verwenden.

Bereichsoperator

z.B. zwischen '1082900' und '1082910'

Klammerebenen

z.B. (= 'München' OR = 'Düsseldorf'), (< '\$BeginDatum' AND
> '\$EndeDatum') OR = null
(Parameter beginnen immer mit dem \$-Zeichen)

Vergleich mit Wert aus anderer Spalte
= T2."basisschlüssel"

Beispiele für Oracle Funktionen in der Spaltendefinition

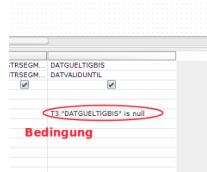
In der Spaltendefinition besteht die Möglichkeit Oracle Ausdrücke und Aggregatfunktionen zu verwenden. Dort eingegebene Ausdrücke werden in den SELECT-Teil der erzeugten Abfrage übernommen.

Differenz von zwei Datumswerten in Tagen
(T2."DATVALIDUNTIL" - T2."DATVALIDFROM")

Durchschnittliche Mietzeit eines Objektes in Wochen
AVG((T1."termin_entgeltspflicht_ende" - T1."termin_entgeltspflicht_beginn") / 7)

Beispiel für Oracle Datum is null Bedingung

Soll ein Datum auf Null geprüft werden, so muss in der Bedingung folgendes eingetragen werden:
"Tabelle"."Attribut" is null



Parameterdefinition

In den Bedingungen der Spalten einer Abfrage können Parameter verwendet werden, deren Werte bei jeder Ausführung der Abfrage vom Benutzer bestimmt werden können. In der Bedingung einer Spalte wird ein Parameter in der Form

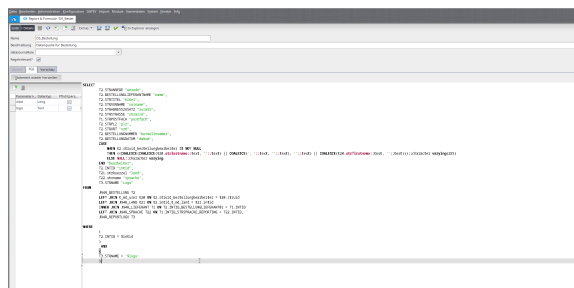
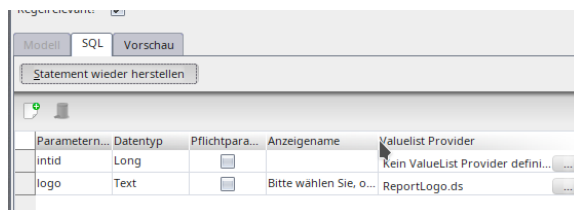
= '\$parametername'

definiert. Wichtig ist dabei, dass dem Namen des Parameters in der Bedingung immer ein \$-Zeichen vorangestellt wird.

Zusätzlich muss jeder Parameter noch in dem Unterformular definiert werden. Hier werden der Datentyp der Spalte sowie ein dem Benutzer anzuzeigender Text zu jedem Parameter definiert:

SQL-Statement bearbeiten

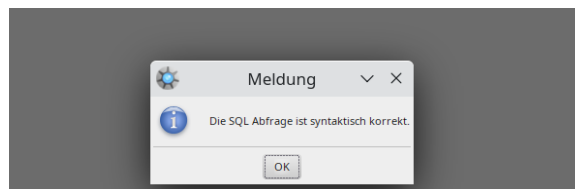
Sollten bei komplexen Abfragen die Möglichkeiten der grafischen Modellierung von Datenquellen nicht ausreichen, gibt es die Möglichkeit, beliebige SELECT-Abfragen in Form von SQL-Statements manuell in einem Editor zu erfassen. Dazu steht im Datenquelleneditor der Reiter SQL zur Verfügung.



Aktiviert man diesen Reiter, wird zunächst die aus dem aktuellen Modell generierte Abfrage angezeigt. Über die Schaltfläche *Statement bearbeiten* wird die Abfrage veränderbar. Es besteht anschließend die Möglichkeit, eine geänderte Abfrage aus dem Modell wiederherzustellen. Hat sich der Benutzer entschlossen, die Abfrage manuell zu erstellen, ist das Modell zunächst gesperrt, bis die Schaltfläche *Statement wieder herstellen* betätigt wurde.

Syntax prüfen

Um festzustellen, ob eine modellierte Abfrage ausführbar ist, besteht die Möglichkeit, die Syntax des erzeugten SQL Statements zu prüfen. Ist die Abfrage korrekt, erscheint die Meldung:



Enthält die Abfrage Fehler, erscheint eine Dialogbox mit der Oracle Fehlermeldung:



Ergebnis Vorschau

Der Reiter Vorschau dient der Ansicht des Ergebnisses, das die Datenquelle liefert.

Um die Laufzeit der Abfrage einzuschränken ist es möglich, die Anzahl der Ergebniszeilen zu beschränken.

Zusätzlich besteht die Möglichkeit, das angezeigte Ergebnis der Abfrage ins Excel-, PDF- oder CSV- Format zu exportieren. Dabei werden immer die angezeigten Zeilen exportiert, die Abfrage wird nicht erneut ausgeführt.

