Named Query Parameters

This feature is new in Ignition version **7.9.4**
Click here to check out the other new features

Parameters allow you to make Named Queries dynamic. They act as placeholders you can pass values into when requesting the query to execute. Other resources in Ignition can then pass arguments into the parameters. The exact implementation depends on what resource is requesting the Named Query, such as a Named Query binding, a Named Query datasource, or the system.db.runNamedQuery function. Check out the Named Query Example page for a complete example on passing parameters into a named query.

On this page ...

Types of Named Query Parameters

There are three types of Parameters in Named queries. Each varies in usage.

Value

The Value type should be used whenever a Named Query needs a dynamic WHERE clause. These act like values passed to a prepared statement, meaning they can never be used to parameterize column or table names. However, they are resilient to SQL injection attacks.

SQL Query - Using a Parameter

```
SELECT * FROM mytable
WHERE name = :myParam
```

QueryString

QueryStrings are more flexible than the Value type in that they can be used to parameterize column and table names. However, their values are never sanitized, which causes them to be more susceptible to SQL injection attacks. When using QueryStrings, it is best to avoid situations where the user can manually type in the value that will be passed to the Name Query. Additionally, if you are using a QueryString for a string in the where clause, you would need to provide quotation marks.

SQL Query - A Using QueryString

```
SELECT {myColumnName} FROM mytable
WHERE name = '{myName}'
```

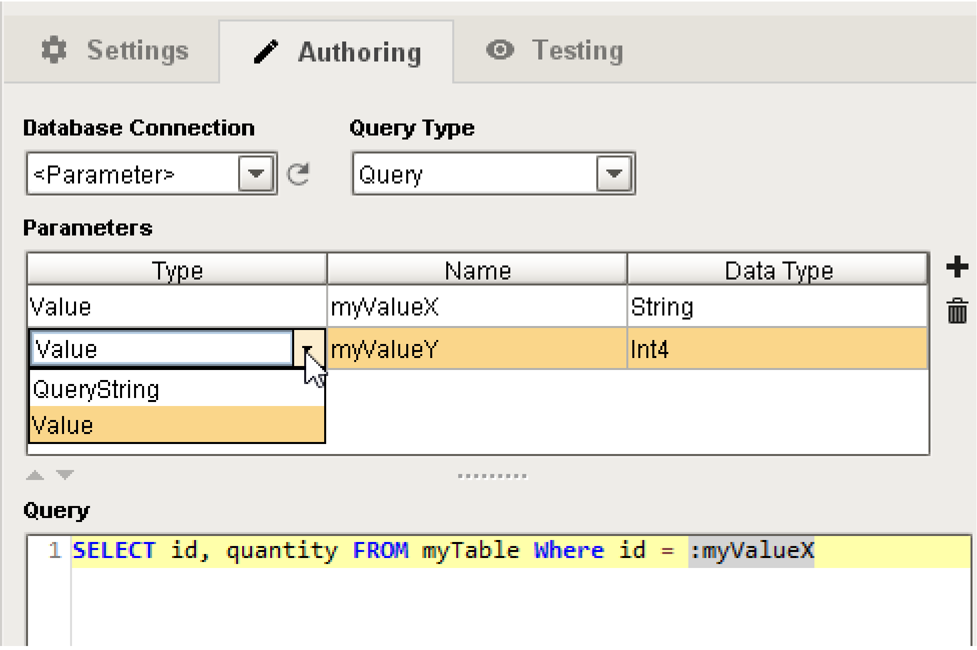
Database

Database type parameters cannot be created manually. Instead, it is automatically created when the **Database Connection** dropdown on the **Authoring** section is set to **<Parameter>**. Additionally, this parameter is not used in the body of the query. This type allows you to parameterize the database connection when the Named Query is called. This way the Named Query can run against multiple database connections specified by the resource that made the request.

Parameters while Authoring a Named Query

Creating Parameters

New parameters can be created in the Authoring section of a Named Query by clicking the add button (✚) next to the Parameters table.



Database Connection

<Parameter>

Query Type

Query

Parameters

Type	Name	Data Type
Value	myValueX	String
Value	myValueY	Int4

Query

```
SELECT id, quantity FROM myTable Where id = :myValueX
```

In Named Queries, Parameters are referenced by their name, so renaming the Parameter will require you to update it on any other resources that are using it.

Each parameter has three properties represented by different columns in the table. These may be edited by double-clicking on the cell you wish to modify:

- **Type**: Changes the type between Value and QueryString.
- **Name**: Determines the name of the parameter, and how it will appear in the query. Names are **not** case-sensitive and must be unique. Additionally, they may only use letters, numbers, dashes and underscores.
- **Data Type**: Specifies the datatype of the parameter. The Type of the parameter determines which data types are available. Note, that QueryStrings may only be configured as strings, where as Value-type parameters have more types available.

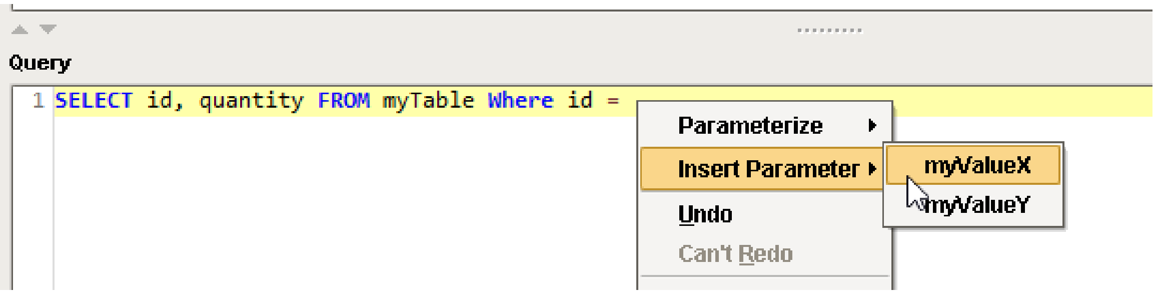
Byte Arrays

This feature is new in Ignition version **7.9.9**
Click here to check out the other new features

As of Ignition 7.9.9, parameters in a named query may now be configured with a byte array datatype, which can in turn be used to pass files into a named query.

Using Parameters in the Query

Once created, parameters can be inserted into the Query field by **drag-and-drop** from the Parameter table onto the Query field, or by using the **right-click menu** in the Query field.



Query

```
SELECT id, quantity FROM myTable Where id =
```

Additionally, the parameters may be typed in manually, but the correct syntax must be used.

Related Topics ...

- Named Query Binding
- system.db.runNamedQuery
- Named Query Conversions
- Using Named Queries - Example