



Rapidly Build Data-Driven Apps

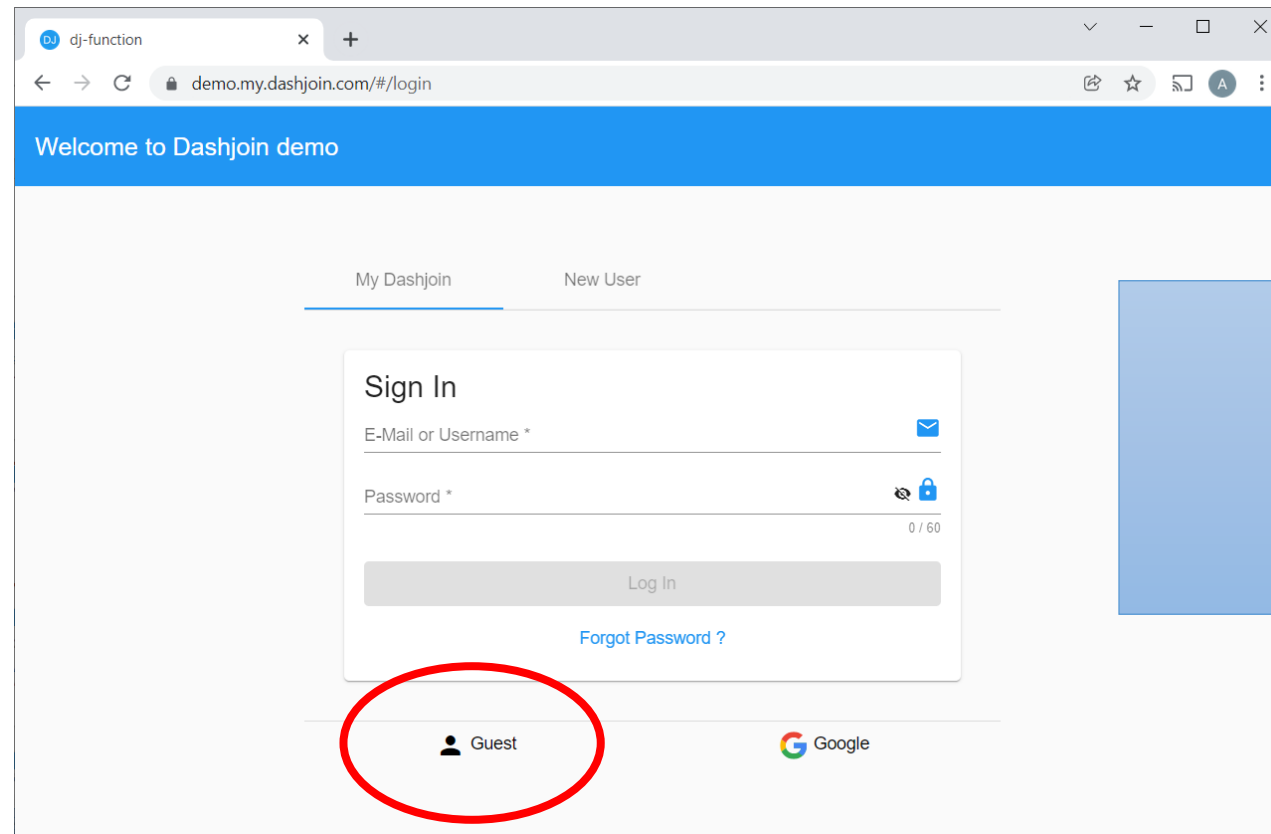
[dashjoin.com](https://dashjoin.com)

# Dashjoin Low Code Platform Training

- Covers all aspects of the platform
  - Accompanying resources are listed below
- Documentation: <https://github.com/dashjoin/platform>
  - Contains a detailed documentation and reference for all features
- Demo videos: [https://www.youtube.com/channel/UCcXpJB1GzQN\\_opSGqegCgoA](https://www.youtube.com/channel/UCcXpJB1GzQN_opSGqegCgoA)
  - Demo videos and tutorials
- Blog: <https://dashjoin.medium.com/>
  - Articles on selected topics
- Twitter: <https://twitter.com/dashjoin>
  - Follow for announcements and news
- Slack Channel: Follow the “join our slack” link here: <https://github.com/dashjoin/platform#contribute>
  - Ask questions in the community

# Access to the Demo System

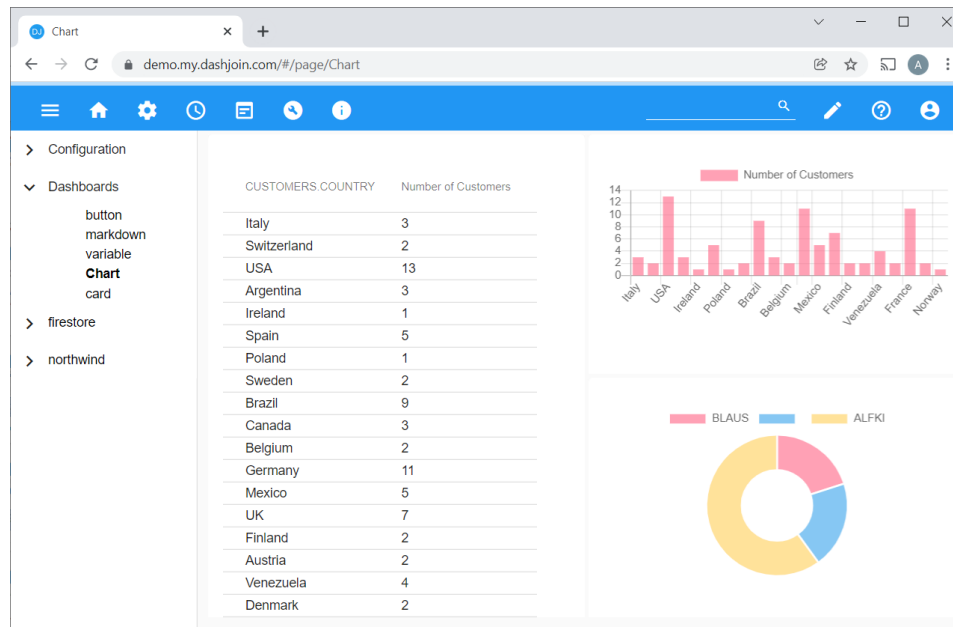
- This training contains some “try yourself” links which can be performed on the demo system
- <https://demo.my.dashjoin.com/>



Important!  
Login using “Guest”

# Access to the Demo System

- Once you are authenticated, please leave the window open
- Copy and paste the “try yourself” links listed on these slides into the address bar of this browser window



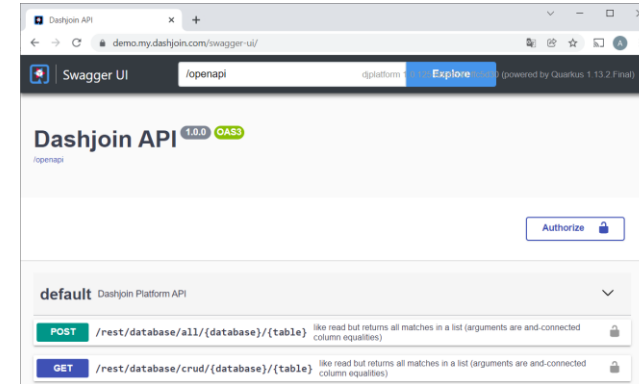
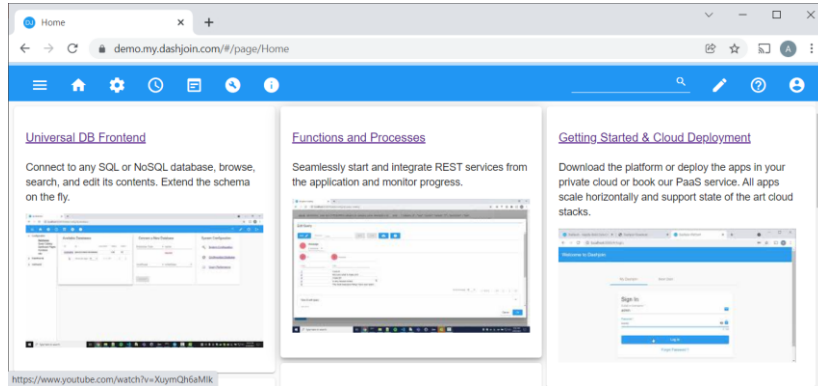
<https://demo.my.dashjoin.com/#/page/Chart>

The try yourself links and some instructions can be found at the bottom of the slide

# Dashjoin is an Open Source & Cloud Native Low Code Development Platform

- Low Code
  - A platform that allows you to build an application by expressing the essential business model and logic in a concise way with little code
  - The platform handles the nitty gritty details
- Open Source
  - You can submit ideas / issues and contribute on [GitHub](#)
- Cloud Native
  - The platform is built to scale and run in any cloud environment
  - Also supports local development setup

# Dashjoin Architecture



## Dashjoin Platform



Config

Dashjoin Container

Dashjoin Container

Dashjoin Container

OpenID

**Relational**  
Oracle  
Postgres

**Document**  
Firebase  
MongoDB

**Graph**  
ArangoDB  
RDF

{ **REST:API** }

# Registering Databases

The screenshot shows the Dashjoin web interface in a browser window. The address bar displays the URL: `demo.my.dashjoin.com/#/resource/config/dj-database/dj%2Fnorthwind`. The interface has a blue header bar with navigation icons. A red circle highlights the gear icon (Settings) in the header. On the left, a sidebar menu is visible with a red circle around the 'northwind' database entry, which is expanded to show a list of tables: CATEGORIES, CITY, CUSTOMERS, CUSTOMER\_CL, CUSTOMER\_DE, EMPLOYEES, EMPLOYEE\_TE, ORDERS, ORDER\_DETAIL, PRODUCTS, REGION, and SHIPPERS. The main content area is titled 'Connection Information' and contains fields for 'Database Type' (SQLDatabase), 'name' (northwind), 'readRoles' (authenticated), 'writeRoles' (writeRoles), 'url' (jdbc:h2:mem:northwind), 'username', and 'password'. At the bottom of this section, there are 'Update' and 'Delete...' buttons, with the 'Update' button circled in red. On the right, a 'Database Tables' section shows a table named 'dj-database'. A blue text box is overlaid on the right side of the interface.

If a new database is registered and every time the database is updated, Dashjoin connects to the database and collects the metadata

<https://demo.my.dashjoin.com/#/table/config/dj-database>

# Data Coordinates

- Dashjoin
  - ID of the Dashjoin installation that accesses the database
- Database
  - Unique name of the database containing the record
- Table
  - table name (unique within the database) of the table containing the record
- Record key(s)
  - Unique ID of the record within its table. This might be a list of keys if we are dealing with composite keys, for instance in a relational database

# Record Coordinates and URL

The screenshot shows a web browser displaying a record in the Dashjoin application. The URL in the address bar is `demo.my.dashjoin.com/#/resource/northwind/CUSTOMERS/ALFKI`. Red circles highlight the components of the URL: the domain `demo.my.dashjoin.com`, the resource path `/resource/northwind`, the table name `CUSTOMERS`, and the record ID `ALFKI`. Below the browser, a navigation bar contains icons for home, settings, and other functions. A sidebar on the left shows a tree view with 'northwind' expanded, listing 'CUSTOMERS'. The main content area displays a record for 'ALFKI' with fields: CUSTOMER\_ID, COMPANY\_NAME, CONTACT\_NAME, CONTACT\_TITLE, ADDRESS, CITY, REGION, POSTAL\_CODE, COUNTRY, PHONE, and FAX.

CUSTOMER_ID	COMPANY_NAME	CONTACT_NAME	CONTACT_TITLE
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative

ADDRESS	CITY	REGION	POSTAL_CODE
Obere Str. 57	Berlin		12209

COUNTRY	PHONE	FAX
Germany	030-0074321	030-0076545

<https://demo.my.dashjoin.com/#/resource/northwind/CUSTOMERS/ALFKI>

# Table Coordinates and URL

The screenshot shows the Dashjoin web application interface. The browser tab is labeled "CUSTOMERS". The URL in the address bar is "demo.my.dashjoin.com/#/table/northwind/CUSTOMERS". The left sidebar shows a navigation menu with "Configuration", "Dashboards", "Firestore", and "northwind". The "northwind" section is expanded, showing "CUSTOMERS" as the selected table. The table displays columns: CUSTOMER\_ID, COMPANY\_NAME, CONTACT\_NAME, CONTACT\_TITLE, and ADDRESS. The first row of data is highlighted, with the CUSTOMER\_ID "ALFKI" circled in red. A blue box with text explains that any time a key is shown in a table result, Dashjoin automatically adds a hyperlink to the record URL shown on the previous slide.

CUSTOMER_ID	COMPANY_NAME	CONTACT_NAME	CONTACT_TITLE	ADDRESS
<a href="#">ALFKI</a>			Sales Representative	Obere Str. 57
<a href="#">ANATR</a>			Owner	Avda. de la Constituc
<a href="#">ANTON</a>			Owner	Mataderos 2312
<a href="#">AROUT</a>			Sales Representative	120 Hanover Sq.
<a href="#">BERGS</a>			Order Administrator	Berguvsvägen 8
<a href="#">BLAUS</a>	Blauer See Delikatessen	Hanna Möss	Sales Representative	Forsterstr. 57

<https://demo.my.dashjoin.com/#/table/northwind/CUSTOMERS>

# Relationships & Navigation

Table  
[EMPLOYEES](#)

← [EMPLOYEES.REPORTS\\_TO](#)  
[Davolio](#) [Leverling](#) [Peacock](#)  
[Buchanan](#) [Callahan](#)

← [EMPLOYEE\\_TERRITORIES.EMPLOYEE\\_ID](#)  
[2/01581](#) [2/01730](#) [2/01833](#)  
[2/02116](#) [2/02139](#) [2/02184](#)  
[2/40222](#)

← [ORDERS.EMPLOYEE\\_ID](#)  
[10265](#) [10277](#) [10280](#)  
[10295](#) [10300](#) [10307](#)  
[10312](#) [10313](#) [10327](#)  
[10339](#)

Link labels can be customized to show fields other than the record ID. This example uses the employee's last name instead of the employee's ID

Dashjoin detects relationships between records and displays navigation links

<https://demo.my.dashjoin.com/#/resource/northwind/EMPLOYEES/2>

# Editing Schema

If your user is in the admin role, you can edit the schema

Column Metadata

Table Metadata

	HOME_PHONE	EXTENSION	PHOTO
Configuration			
Dashboards			
Firestore			
northwind			
CATEGORIES			
CITY			
CUSTOMERS			
CUSTOMER_CONTACTS			
CUSTOMER_DEPARTMENTS			
EMPLOYEES			
EMPLOYEE_TERMINATIONS			
ORDERS			
ORDER_DETAILS			
PRODUCTS			
REGION			
SHIPPERS			

Configuration

Dashboards

Firestore

northwind

REQUESTS

CATEGORIES

CITY

CUSTOMERS

CUSTOMER\_CONTACTS

CUSTOMER\_DEPARTMENTS

EMPLOYEES

EMPLOYEE\_TERMINATIONS

ORDERS

ORDER\_DETAILS

PRODUCTS

REGION

HOME\_PHONE

EXTENSION

PHOTO

REPORTS\_TO

PHOTO\_PATH

EMPLOYEE\_ID

Edit

Delete...

Name of the column to create

NewColumn

Datatype

string

Create Column

Table Metadata

Not accessible in the demo because the guest user has no permissions

# Editing Schema

- Database
  - Upload data to new tables
  - Create, rename, delete table
- Table
  - [Edit dj-label](#) (defines which field to use as link labels)
  - [Define triggers](#)
  - Create, rename, delete column
  - Change column datatype
  - Edit table relationship (in case it is not defined in the DB)

# Create, Read, Update, Delete

Configuration

Dashboards

firestore

northwind

CATEGORIES

CITY

CUSTOMERS

CUSTOMER\_CUS

CUSTOMER\_DEM

**EMPLOYEES**

EMPLOYEE\_TERF

ORDERS

ORDER\_DETAILS

PRODUCTS

REGION

SHIPPERS

SUPPLIERS

TERRITORIES

US\_STATES

New Record

EMPLOYEE\_ID

LAST\_NAME

FIRST\_NAME

TITLE

required

required

required

TITLE\_OF\_COURTESY

BIRTH\_DATE

HIRE\_DATE

ADDRESS

CITY

REGION

EXTENSION

REPORTS\_TO

PHOTO\_PATH

Davolio

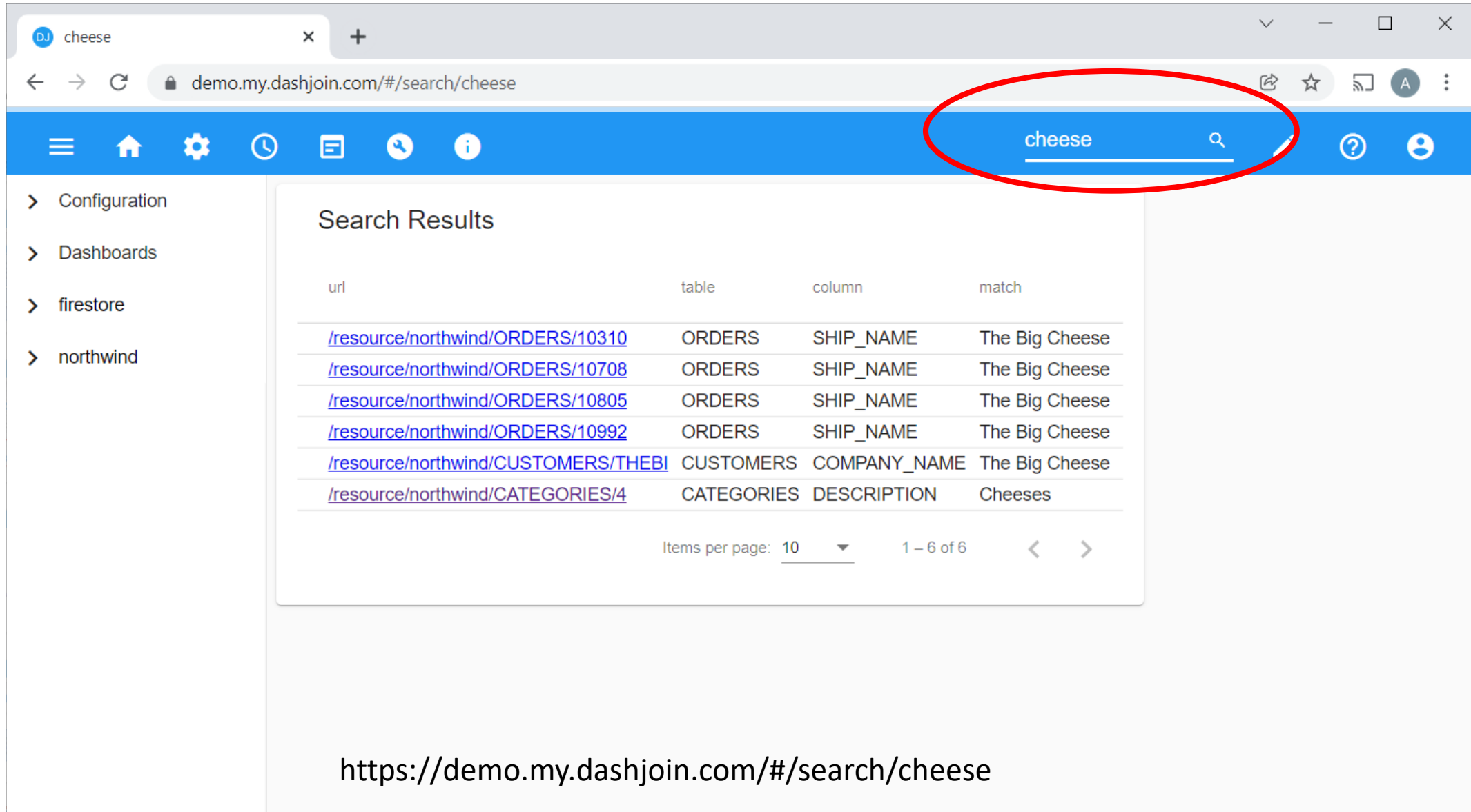
Dodsworth

Dashjoin automatically creates forms to create, read, update, and delete any record. The metadata is used to determine required fields or suggest values

<https://demo.my.dashjoin.com/#/table/northwind/EMPLOYEES>

If you “create”, you will get a permission error

# Search Across all Databases



The screenshot shows a web browser window with the address bar displaying `demo.my.dashjoin.com/#/search/cheese`. The application's search bar, located in the top right of the blue header, contains the text "cheese" and is circled in red. On the left, a sidebar menu lists "Configuration", "Dashboards", "firestore", and "northwind". The main content area, titled "Search Results", displays a table of results. The table has four columns: "url", "table", "column", and "match". It lists six results, all of which match the search term "cheese". The first five results are from the "ORDERS" table, and the last one is from the "CATEGORIES" table. At the bottom of the table, there is a pagination control showing "Items per page: 10" and "1 - 6 of 6".

url	table	column	match
<a href="/resource/northwind/ORDERS/10310">/resource/northwind/ORDERS/10310</a>	ORDERS	SHIP_NAME	The Big Cheese
<a href="/resource/northwind/ORDERS/10708">/resource/northwind/ORDERS/10708</a>	ORDERS	SHIP_NAME	The Big Cheese
<a href="/resource/northwind/ORDERS/10805">/resource/northwind/ORDERS/10805</a>	ORDERS	SHIP_NAME	The Big Cheese
<a href="/resource/northwind/ORDERS/10992">/resource/northwind/ORDERS/10992</a>	ORDERS	SHIP_NAME	The Big Cheese
<a href="/resource/northwind/CUSTOMERS/THEBI">/resource/northwind/CUSTOMERS/THEBI</a>	CUSTOMERS	COMPANY_NAME	The Big Cheese
<a href="/resource/northwind/CATEGORIES/4">/resource/northwind/CATEGORIES/4</a>	CATEGORIES	DESCRIPTION	Cheeses

Items per page: 10 1 - 6 of 6

<https://demo.my.dashjoin.com/#/search/cheese>

# Query Catalog

The screenshot displays the Dashjoin Query Catalog interface. The browser address bar shows the URL: [demo.my.dashjoin.com/#/table/config/dj-query-catalog](https://demo.my.dashjoin.com/#/table/config/dj-query-catalog). The left sidebar contains a 'Configuration' section with the following items: Databases, **Query Catalog**, Dashboard Pages, Functions, and Info. The main content area is titled 'Available Queries' and contains a table with the following data:

ID	database
<a href="#">firestore</a>	dj/firestore
<a href="#">submit</a>	dj/northwind
<a href="#">orgchart</a>	dj/northwind
<a href="#">list</a>	dj/northwind
<a href="#">group</a>	dj/northwind

A blue text box overlay states: "Dashjoin allows you to create and manage queries across all databases. These queries can be used in conjunction with widgets as will be shown shortly". Below the table is a 'New Query' form with the following fields:

ID	type	database	roles

The 'ID' field is marked as 'required'.

# Graphically Compose Queries on any Database

orgchart

demo.my.dashjoin.com/#/resource/config/dj-query-catalog/orgchart

### Edit Query

northwind Distinct Limit

**CUSTOMERS** 8 columns   
**CUSTOMER\_CUSTOMER\_DEMO** 2 columns   
**ORDERS**

CUSTOMER\_ID COMPANY\_NAME ADDRESS

Filter Filter Filter

<a href="#">ALFKI</a>	Alfreds Futterkiste	Obere Str. 57
<a href="#">ANATR</a>	Ana Trujillo Emparedados y helados	Avda. de la Constitució
<a href="#">ANTON</a>	Antonio Moreno Taquería	Mataderos 2312
<a href="#">AROUT</a>	Around the Horn	120 Hanover Sq.
<a href="#">BERGS</a>	Berglunds snabbköp	Berguvsvägen 8

ORDER\_ID = 10248  
CUSTOMER\_ID = VINET  
EMPLOYEE\_ID = 5  
ORDER\_DATE = 1996-07-04  
REQUIRED\_DATE = 1996-08-01  
SHIPPED\_DATE = 1996-07-16

Cancel Ok

<https://demo.my.dashjoin.com/#/table/config/dj-query-catalog> , scroll down, click “Editor” and then the pen symbol

# Query Parameters

The screenshot shows the 'Edit Record' page for a query catalog entry in the DashJoin application. The interface includes a sidebar with navigation options like 'Configuration', 'Databases', 'Query Catalog', 'Dashboard Pages', 'Functions', and 'Info'. The main content area displays the 'Edit Record' form with the following fields:

ID	type	database	roles
orgchart	read	dj/northwind	authenticated

arguments key	type	sample
node	integer	2

Below the form, there is a '+' icon and a SQL query snippet:

```
SELECT  
  "EMPLOYEES"."EMPLOYEE_ID"  
FROM  
  "EMPLOYEES"  
WHERE  
  EMPLOYEES.REPORTS_TO = ${node}
```

A blue callout box on the right states: "Some queries require a parameter in order to run".

# Functions

The screenshot shows the DashJoin web interface. The browser address bar displays `demo.my.dashjoin.com/#/table/config/dj-function`. The left sidebar contains a navigation menu with the following items: Configuration (expanded), Databases, Query Catalog, Dashboard Pages, **Functions**, and Info. Under 'Configuration', there are sub-items: Databases, Query Catalog, Dashboard Pages, **Functions**, and Info. The main content area is titled 'Available Functions' and contains a table with the following data:

ID	class	database	type	roles	status	start	end
<a href="#">invoke</a>	Invoke		read	authenticated			
<a href="#">misp</a>	ETL	firestore	write				
<a href="#">email</a>	Email		write	admin			

Below the table, there is a pagination control showing 'Items per page: 10' and '1 - 3 of 3'. Below the table is a 'New Function' form with a 'Function Type' dropdown menu and an 'ID' input field. The 'ID' field is marked as 'required' in red text.

Functions are used to interact with external services (APIs, Email, ...), load data into a database, or simply express and store a reusable algorithm

<https://demo.my.dashjoin.com/#/table/config/dj-function>

Select function type "Invoke" and enter the expression  $4*9$  (more on expressions shortly...)

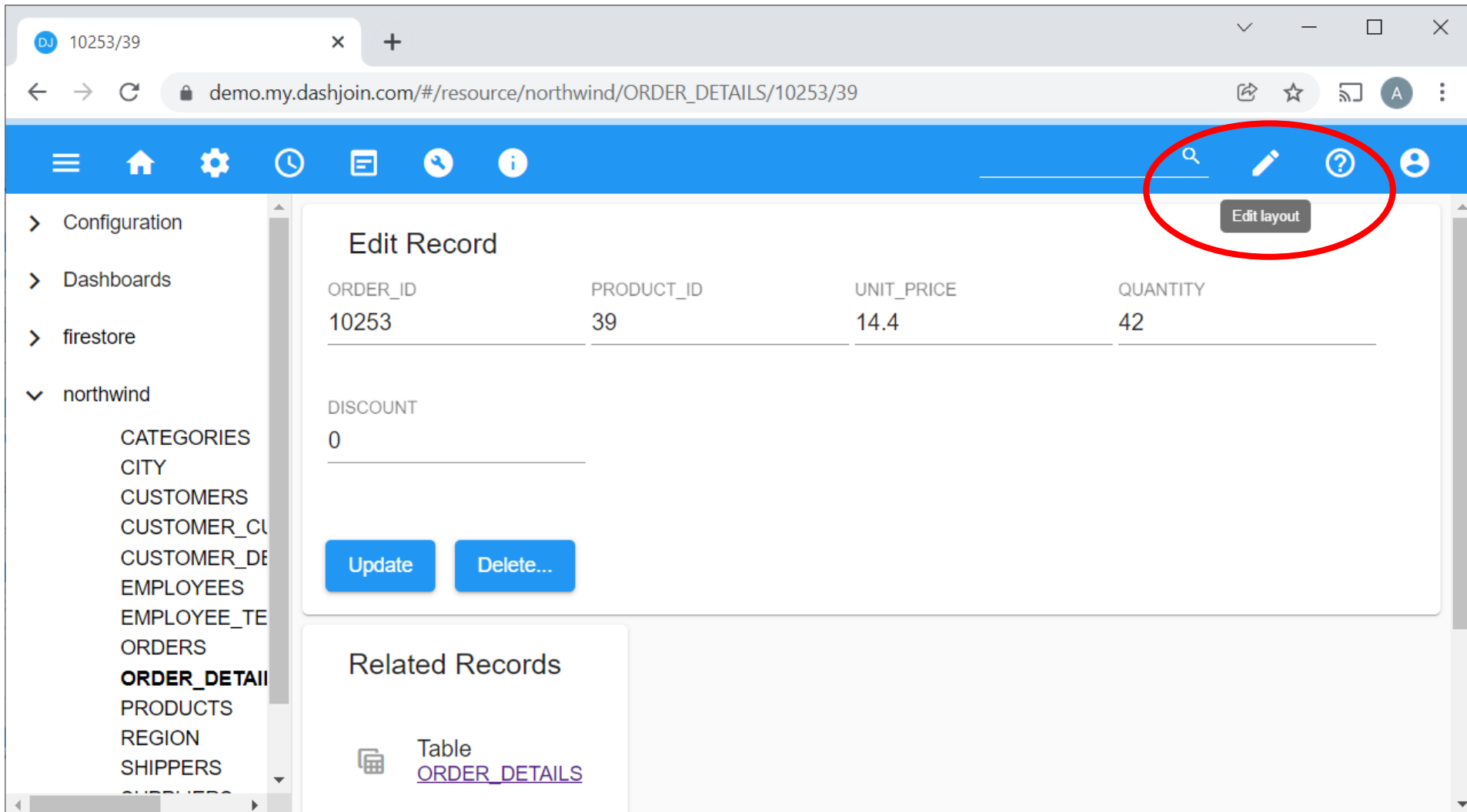
# Dashjoin Page Types

- In Dashjoin, there are three types of user interface pages
- Table UI: visualizes a table (type / collection) in a database
  - <https://demo.my.dashjoin.com/#/table/northwind/EMPLOYEES>
- Record UI: visualizes a record of a table
  - <https://demo.my.dashjoin.com/#/resource/northwind/EMPLOYEES/2>
- Dashboard: a page that is not associated with a database element
  - <https://demo.my.dashjoin.com/#/page/Info>

# Page Defaults

- Unless it is customized by the user, every page type has some defaults
- Table UI default
  - Table with all entries
  - Form to create a new record
- Record UI default
  - Form to update / delete the record
  - Widget showing links to all related records
- Dashboard default
  - The system comes with the home and info pages
  - The links to these are included in the toolbar

# Customizing Pages



[https://demo.my.dashjoin.com/#/resource/northwind/ORDER\\_DETAILS/10253/39](https://demo.my.dashjoin.com/#/resource/northwind/ORDER_DETAILS/10253/39)

# Customizing Pages

The screenshot shows a web application interface with a browser window at the top. The address bar displays the URL: `demo.my.dashjoin.com/#/resource/northwind/ORDER_DETAILS/10253/39`. The application has a sidebar on the left with a menu structure:

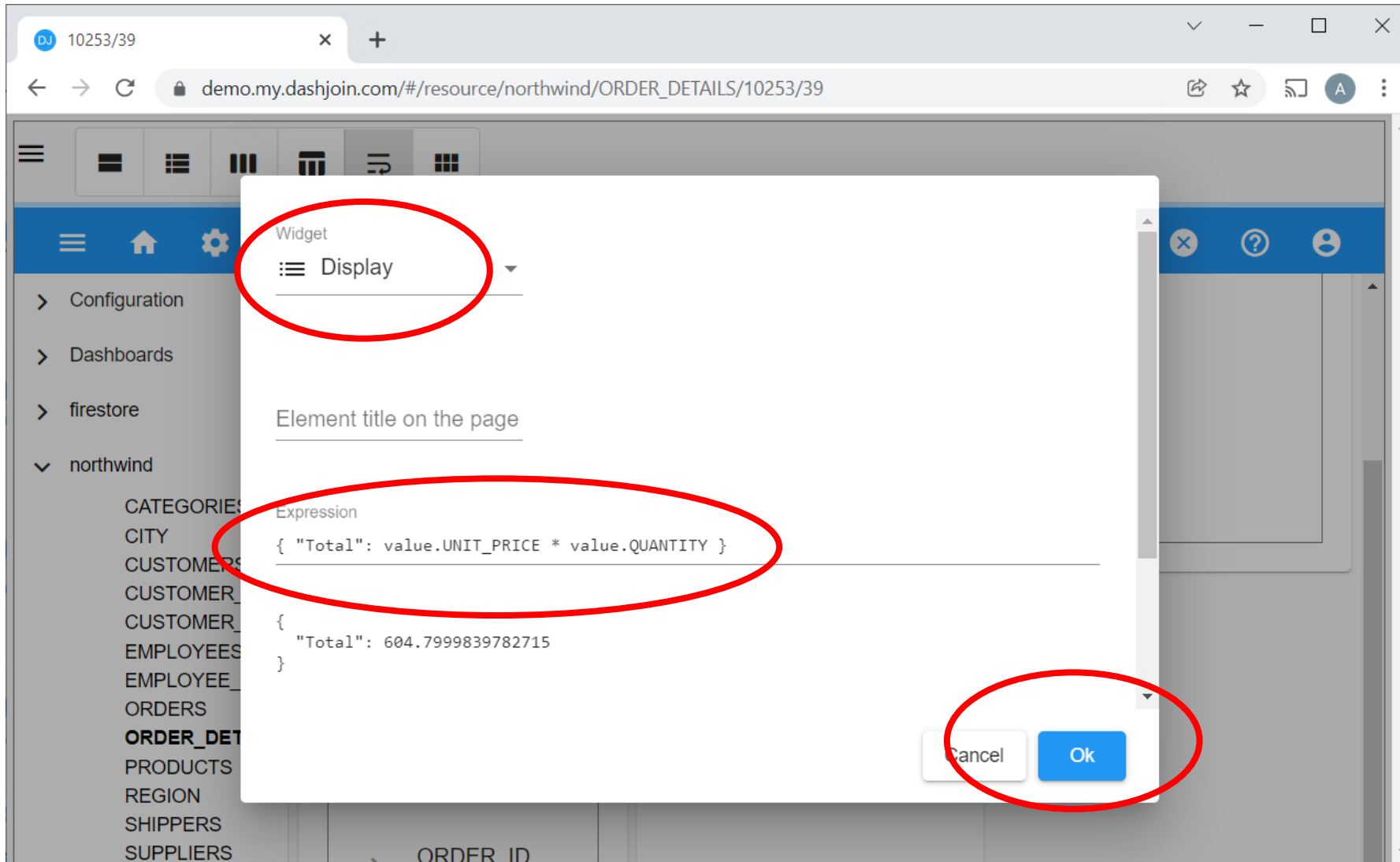
- > Configuration
- > Dashboards
- > firestore
- ✓ northwind
  - CATEGORIES
  - CITY
  - CUSTOMERS
  - CUSTOMER\_CU
  - CUSTOMER\_DE
  - EMPLOYEES
  - EMPLOYEE\_TE
  - ORDERS
  - ORDER\_DETAIL**
  - PRODUCTS

The main content area features a 'Related Records' section. It includes a table with the following data:

Table	ORDER_ID	PRODUCT_ID
<a href="#">ORDER_DETAILS</a>	<a href="#">10253</a>	

A red circle highlights a button labeled 'custom instance layout' in the top right corner of the 'Related Records' section. A tooltip labeled '"ORDER\_DETAILS" record' is visible over the button.

# Customizing Pages



# Customizing Pages

The screenshot displays a web application interface for customizing pages. The browser address bar shows the URL: `demo.my.dashjoin.com/#/resource/northwind/ORDER_DETAILS/10253/39`. The interface includes a sidebar with a menu, a top navigation bar with icons, and a main content area.

Two red circles highlight specific features:

- The first circle highlights a set of icons in the top navigation bar, including a home icon, a gear icon, a clock icon, a document icon, a mail icon, an information icon, a search icon, a save icon, a refresh icon, a question mark icon, and a user profile icon.
- The second circle highlights a 'Total' value in the 'Related Records' section, which is `604.7999839782715`.

The 'Related Records' section shows a table with the following data:

Table	ORDER_ID	PRODUCT_ID
<a href="#">ORDER_DETAILS</a>	<a href="#">10253</a>	

The 'Total' value is `604.7999839782715`.

# Customizing Pages

The screenshot displays a web application interface with a sidebar on the left and a main content area. The sidebar contains a menu with various icons and a 'Delete custom layout' option at the bottom. The main content area shows a form with fields and buttons, and a 'Related Records' section below it. Two blue callout boxes provide instructions on how to customize the page layout.

Choose from different layouts like 1 column, floating, grid, etc.

Delete custom layout reverts back to the default page

# Expressions

- Expressions are a very central concept in Dashjoin
- Expressions are used to define
  - What to visualize on a page
  - How to react to user interface events (e.g., button click)
  - How to react to database trigger events (e.g., row inserted)
- Expressions are also used for extract transform load (ETL) processes
  - How to extract data from sources
  - How to transform the data

# JSONata

- Dashjoin uses the query and transformation language [JSONata](https://jsonata.org/)
- `expression(input) = output`

The screenshot shows the JSONata Exerciser interface. On the left, the input JSON is displayed: `{ "FirstName": "Fred", "Surname": "Smith", "Age": 28, "Address": { "Street": "Hursley Park", "City": "Winchester", "Postcode": "SO21 3JN" } }`. A dropdown menu shows "Address" selected. In the center, the JSONata expression `*.number` is entered. On the right, the output is shown as an array: `[ "0203 544 1234", "01962 001234", "01962 001235", "077 7700 1234" ]`.

The input: in Dashjoin, this is

```
{
  "value": the current DB record,
  "user": name of the user logged in,
  "form": this widget's form values,
  "variable": page variables
}
```

JSONata expression (e.g., the one entered in the display widget)

Output (e.g., the data shown in the display widget)

# Dashjoin JSONata Extensions: \$query

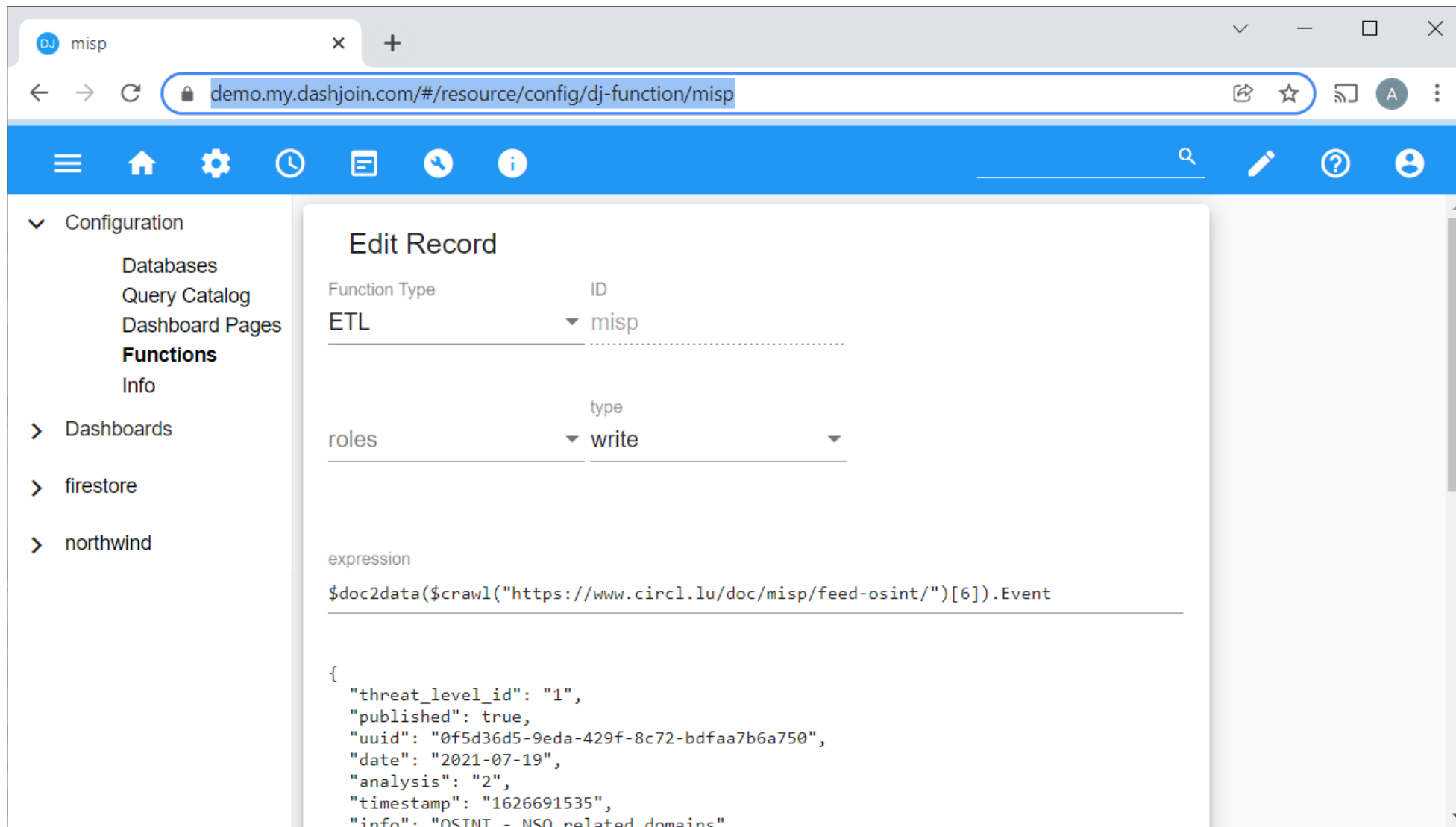
Query parameter: Replace \$ with a number like 2

```
$query("northwind", "list", {"limit": 2}).$echo$. "CATEGORIES.DESRIPTION"
```

```
[  
  "Soft drinks, coffees, teas, beers, and ales",  
  "Sweet and savory sauces, relishes, spreads, and seasonings"  
]
```

<https://demo.my.dashjoin.com/#/resource/config/dj-function/invoke>

# Dashjoin JSONata Extensions: \$scrawl, \$doc2data



\$scrawl and \$doc2data are not allowed as guest user

# App Development using GitHub

- As a Dashjoin developer, you can make changes to the following items
  - dj-database
    - Connection info
    - Custom table and instance layouts
    - Additional schema metadata like dj-label
  - dj-function: functions used in the app
  - dj-query-catalog: queries used in the app
  - dj-role: defines which roles are available
  - page: additional dashboard pages
  - widget: reusable layout blocks
- All of these are stored in the config database and are live as soon as you save them

# Collaborative Development Workflow

- Dashjoin allows teams to adopt a full-fledged development workflow including issue tracking, branching, code reviews, unit testing, and controlled production releases



# Setup

- Any Dashjoin instance can relate to a GitHub repository via environment variables
  - DASHJOIN\_APPURL: repository URL like <https://github.com/dashjoin/dashjoin-demo>
  - DASHJOIN\_HOME: folder where the app is installed
- If the folder is empty, a git clone is performed
- A git pull is performed otherwise

# Using Docker

- Run the container and mount the dashjoin-demo folder to the host
- Open this folder using [VSCode](#)

```
C:\tmp> docker run
  -d
  -p 8080:8080
  -e DJ_ADMIN_PASS=djdjdj
  -e DASHJOIN_HOME=dashjoin-demo
  -e DASHJOIN_APPURL=https://github.com/dashjoin/dashjoin-demo
  -v c:/tmp/dashjoin-demo:/deployments/dashjoin-demo dashjoin/platform
```

```
C:\tmp> code dashjoin-demo
```

# Make a Change

The screenshot shows a web browser window with the address bar displaying `localhost:8080/#/table/config/dj-function`. The page title is "dj-function". The interface has a blue header bar with navigation icons. On the left, a sidebar lists the following items: Configuration (expanded), Databases, Query Catalog, Dashboard Pages, **Functions**, Info, > Dashboards, > northwind, rdf4j, and sqlite. The main content area is titled "New Function" and contains the following fields:

- Function Type**: A dropdown menu with "Invoke" selected.
- ID**: A text input field containing "test".
- roles**: A dropdown menu with "type" selected.
- expression**: A text area containing the text "1+1".

At the bottom left of the form, there is a blue button labeled "Create", which is circled in red.

<http://localhost:8080/#/table/config/dj-function>

# View, Stage, Commit, and Push Change

