DNNrocket Index System

# Introduction

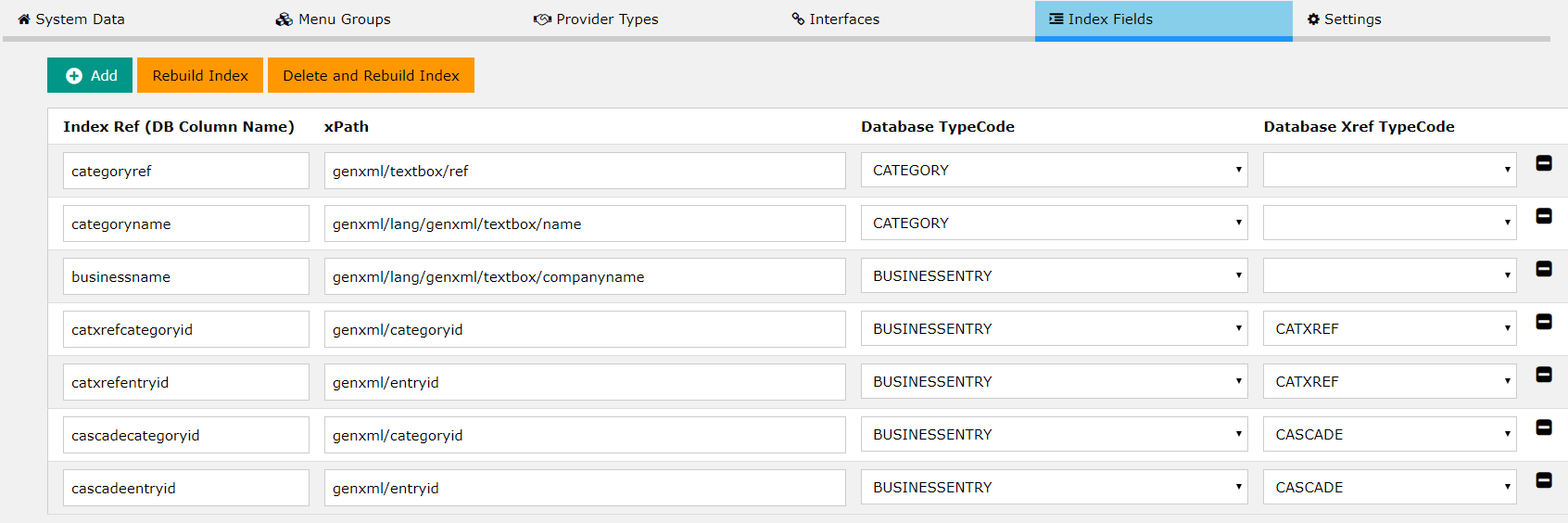
To make data selection quicker DNNrocket uses a SQL indexing system. This is controlled by the DNNrocket administration panel. “/Desktopmodules/dnnrocket/adminsystem.html”.

# Admin Panel

The admin panel is an admiration system across all systems on the install. It can be found on the url of:

*<domain>*/Desktopmodules/dnnrocket/adminsystem.html

The index fields section controls the indexes created on the database and in turn this creates the DB table join which is part of the SPROC. “DNNrocket\_GetList”, “DNNrocket\_GetListCount”.



**Index Ref**: This is the database joined table name that will be used in code and by any SPROC call.

Example:

filter += " and companyname.GuidKey like '%" + searchtext + "%'";

*NOTE*: The index field always uses the “GuidKey” to store data and hence it is this DB column that will need to be used.

**xPath**: This is the xpath of the data field. It identifies what data should be put into the index.

**Database TypeCode**: The interface Entity TypeCode, this is the main data and any Xref TypeCode will be linked to this.

**Xref TypeCode**: Join cross reference data, like CATXREF TypeCode for linking categories to the Database TypeCode selected.

**Rebuild Index**: The index fields in the database can be re-indexed by using the button.

# SYSTEMLINK records

When the index fields are saved they create a set of DB records which start with “SYSTEMLINK%”, these records tell the indexing system what needs doing.

# Indexing Code

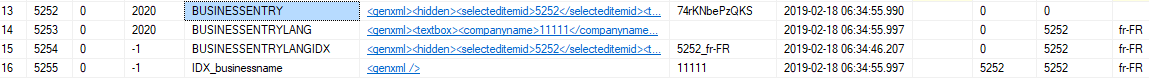
The code to create the indexes is in c#, not DB triggers. It can be found in the “DNNrocketController” class. With 2 functions “RebuildLangIndex(SimplisityRecord objInfo, int itemid)” and “RebuildIndex(SimplisityRecord objInfo, int itemid)”.

These functions are call when a record is updated. They are also called by the re-index operation.

# Language Index

To make selection quicker, we have a language record index. This is a XML merge between the language data fields and the non-language data fields. It uses a TypeCode of “<EntityTypeCode>LANGIDX”

# Example BUSINESSENTITY record



The creation of 1 record with 1 language will result in 3 record, the last one being the Language index. The 4th record is the field index record, the value to be indexed will populate the GUIDKey column.

# When to create an index

Index records are designed to make the database selection and sorting faster, but keep in mind that, like any database, creating indexes does have a performance affect.

Selection based on XML values is already quick and on a small database may not be required. The slow thing for XML is when the records are sorted, so think about adding an index if you know a sort of a data field will happen.

All columns apart from the XML are already index by SQL server, hence the reason we use the GUIDKey field.

# Things to Know

* Only data with a language record will have an index created. If the data record has no need for a language, but you require an index you must create a dummy language record.
* The database record MUST have a valid systemId field, this is required for the IDX record to be created.