

# Overview of Orixia System tables and Database Framework Standards

This technical document intends to describe the purpose of the different data-tables in an Orixia database, so you can understand how to use them as you structure your own Orixia App.

Orixia uses a set of standards for building the relational database that provides data for your App. The App then links together the data-tables following these standards to create your App. Standards which affect the App include constraints and foreign-keys, specialized descriptions and decorations added on schema elements. This results in considerable automation of system-building. Orixia allows skilled users to build a database, and almost immediately have a working system they can use, with a lot of powerful functionality.

In addition to the use of the database schema itself, an Orixia App also has a few data-tables in the database which are used to structure and generate the App.

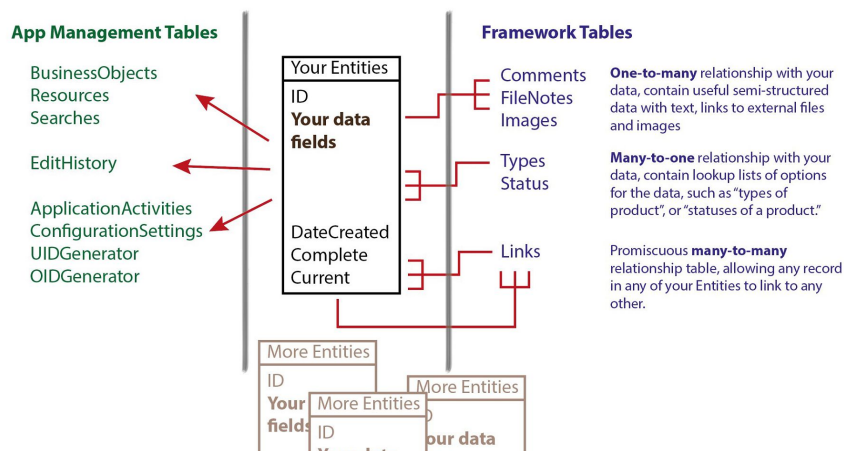
These fall into categories:

"Framework Tables" which Users add data to, holding things like Images, Pictures, Comments and Links to on-disk files.

"Linking and Listing Tables" which make it easy to set up pick-lists, and join data-records together.

"App Management Tables" which Developers and the App itself add data to. These hold data which control how your App behaves, and how it looks. For example the size and position of on-screen elements is saved by your App as you use it.

## The Framework data-tables



Overview of the Orixia Framework tables

## Your Entities

You add your own data-tables to the Orixia database as you need them. These will typically be tables for storing your business data, such as tables for "Customers", "Products" or "Services" or any other line-of-business data. These become "Entities" once you add records to the **BusinessObjects**, **Searches** and **Resources** framework data-tables so your App can use them. Orixia uses the term "Entities" to refer to anything from a single data-table with added App Management records, to sets of data-tables which are grouped to work together as "parent" "child" and "extensions" of each other.

The main rules that Orixia has for Your Entities are:

1. Every table must have an "ID" field containing unique **integer** (whole number) values. This ID is used to Link, Index and Search the data. This ID is generated using Orixia's internal "UID()" function, the developer should not touch this process.
2. Every table must have a "DateCreated" field which defaults to the Current\_Timestamp (the exact moment the record was created). This data is used by Orixia to manage your data.
3. Wherever possible an Entity should include a "Name" field, containing a string of characters. This is not required, but is helpful as Orixia has built-in code to create lists of a record, based on the presence of a field called "Name" or "FullName". Note that the "FullName" field can be **generated** from other fields in

3. the record or data elsewhere in the database.

## Framework Tables: "Child" tables which link to your entities on a one-to-many basis

### Comments & FileNotes

Any Entity with "LinkToComments" ticked in their BusinessObjects data-record will automatically allow Users to connect records to one or more "Comments" or "FileNotes" records, which can then be viewed, exported and added to resources and reports as needed.

A link to the **Comments** framework table can be incredibly useful for an Entity where a record has a series of actions undertaken on it, it allows Users to add "notes along the way". Comments are also a useful way of adding optional data, which does not need to be stored within the main data-columns of the record.

The **FileNotes** framework table allows the Users to create a permanent link between one record in an Orixa Entity and an on-disk file. For example, this can be useful where Orixa is managing data such as projects, and there is a wish to make it easy for Users to access external Project Reports, terms of reference or other external information.

Remember that Orixa stores the relative path of the linked file, therefore for FileNotes to be used by multiple users on a network the file-storage heirarchy must be network-based. A local file added by one User (ie "C:\MyDocument") will not be accesible to a second user. However a network file (ie "//Server/Specifications/MyDocument") will be universally available.

### Images

Any Entity with "LinkToImages" ticked in their BusinessObjects data-record will automatically allow Users to add one or more "FileNotes" records, which can then be viewed, exported and added to resources and reports as needed.

The **Images** framework table is a simple way to link pictures and graphic content to a record without the need for complex programming.

## Linking and Listing Tables: Tables which allow your entities to contain lists and ease linkages between different parts of a database.

### Types

This data-table is used to hold simple lists of possible values, such as "Categories of Product" or "Types of Service" provided by the business. Orixa creates a mechanism to pick from the internally generated list, and for Users (with suitable security clearance) to extend and edit the list. The main record holds the ID of the record in the "Types" table, rather than the longer text, meaning that data storage and querying are very efficient.

### Status

This data-table is used to hold simple lists of possible values, but unlike the "Types" framework-table, "Status" records include an "Ordering" field which allows records with a "StatusID" field to be organised. Orixa creates a mechanism to pick from the internally generated list, and for Users (with suitable security clearance) to extend and edit the list. Note that "Status" records can also include a colour field, and this colour is pulled through into the colouring of the user-interface of the App in some situations.

It is normal for both the Status and Types data-tables to have a **security-level** to restrict data-entry. This means that lists cannot be accidentally updated, but can be changed if this is needed.

### Links

In order to enable Links for any table, just tick the "AcceptsLinks" and / or "CreatesLinks" check-box in that Entity's Business Object Record.

The Links Framework table is a special case. It is a table which is used to create linkages between **any** record an Entity and any other record in any other Entity. It is a mechanism for creating "promiscuous linkages".

## App Management Tables

### BusinessObjects

Each record in this table holds the information Orixa needs about one Entity, including SQL scripts needed in certain parts of the App and master-settings. Orixa matches the name of the record in the BusinessObjects table to the name of the data-table, so a record with the name "Products" in the BusinessObjects table will create an Entity that displays the data in the "Products" table.

### Resources

Each record in this table holds the information Orixa needs to generate some part of the App. Reports and Charts for example are created by the App based on the prescence of records in this data-table. The "ComponentsID" field shows the names of the different parts of the App which can be added / created by adding Resources.

### Searches

The most general way of finding data in Orixia is via the View-Grid. Users can see many different grids. Their contents are controlled by the records in this table.

## EditHistory

Orixia keeps a basic log of all actions by all Users by adding records to this data-table. When a user Inserts, Edits or Deletes a record or executes a Resource data is added to the EditHistory. These records only log the record that was changed or accessed, they do not keep track of the details of the change that were made. In technical terms the EditHistory table is intended as a log **not** a "Journal" in the sense of a journal being a store of all changes made during the edit process.

Orixia includes mechanisms for "Journalling" (ie storing a full list of the changes made to a data record over time), but this requires the addition of customization to the basic Orixia App.

## Additional App Management Tables

In addition to the above, important framework tables, Orixia includes several "service" tables in a small "SystemDB" database. Users and Developers rarely need to know the purpose of these tables, as most of their operation is related to actions behind the scenes in the App. As well as the tables listed below your App may contain other SystemDB data-tables which have been added for additional purposes specific to your App.

## UIDGenerator

This table is used to generate the ID values used by all records in the App. It contains a column called "UID" which will be used for the next ID and then incremented, and a column called "NextMaxUID". NextMaxUID is used in multi-server environments. Once the UID has been incremented to a point where it is equal to NextMaxUID no further records will be added.

This is done because values above NextMaxUID have been allocated to another server, and cannot be used. All servers sharing one set of database across multiple instances of the database must work with their own, separate **ranges** of UID Values. Server one might use 1,000,000 - 10,000,000, Server two might use 10,000,001 to 20,000,000 etc.

If you have a multi-server App it will include two Functions one called "FewRecordsRemain" and a second called "NextMaxUIDFromCloud" these work together to update the UID and NextMaxUID when a Server needs a new range of UID values.

## ApplicationActivities

This table contains a list of Users' activities and populates various "RecentLists" in your App. For example recently edited records, recently viewed Resources etc.

## ConfigurationSettings

This table contains saved settings for users. For example the size and position of Windows on a User's screen, or other settings which need to be stored between sessions.

## Summary Conclusions

The above gives a very brief overview intended to make the basic shape of the Orixia Framework as clear as possible. Generally, the framework tables just work in an App, and Users do not need to really know about them or understand them.

However if you wish to take greater control of your Orixia App it is sensible to have at least a basic understanding of what its components are and how they work together.

Hopefully this article and others in this section of the documentation will make that possible.