

"Relational Data" how Orixo models data

This is an introductory article which tries to explain how relational databases (RDBMS) work. For anyone with technical knowledge of these issues, this article will be too introductory to be useful.

Many businesses store data. The starting point for this is usually a spread-sheet program like Excel.

A list of customers might be added to one sheet, with a list of orders on another sheet. Or one sheet might be given an area containing headings for a customer, and rows laid out below this for their orders.

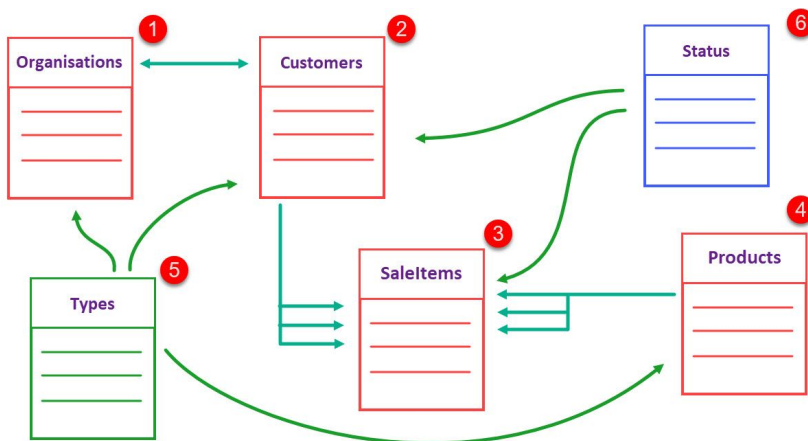
It is hard for data stored in this way to be managed at scale. This has been understood since the 1960s (or earlier) and the data-base work of Edgar F. Codd.

Codd developed the idea of relational data. This has been built on for decades in many database products. Orixo has taken the fundamental elements of relational database development and applied a standard to ease extensibility and rigour.

Relational Database theory contains the idea that data-tables should contain "atoms" of data, with links to allow multiple atoms from one data-table to connect to another.

Orixo has standardized this with a simple terminology for its different data elements

BusinessObject all data-tables which a user can see and edit are referred to as "BusinessObjects", these come in different types, which are described below. To turn any data-table in your database into a BusinessObject all that you need to do is add a new record to the BusinessObjects table, using the name of your data-table as the name in the "Name" field of the BusinessObjects record. When your App opens Orixo will see this record and use its contents to create items that users can interact with on-screen.



A simplified Data model diagram

BusinessObjects come in different flavours

1. **Master** (shown in the image above as "Organisations"). Any BusinessObjects record with a "BizObjType" = "M" will be treated as a "Master" BusinessObject. This means it can have **extensions**. Master BusinessObjects are usually big, important data-tables in your App.
2. **Extension** (shown in the image above as "Customers"). "BizObjType" = "E". An extension BusinessObject adds extra data-columns in a new data-table, but these are always added as "extras" onto some master record. An extension BusinessObject never exists by itself. It always has a "master." Extensions are useful as they allow data to be separated into compartments. Extension's are often useful for heavily used data-tables which might otherwise have a very large number of empty columns.
Examples of master / extension data-tables are:
Organisations/ Customers / Suppliers
People / Staff / Contacts / Farmers.
Products / RawMaterials / SalesProducts.
3. **Child** (shown in the image above as "SaleItems"). "BizObjType" = "C". A child BusinessObject adds multiple rows of data to a system, where each row can be linked to one or more "parent" BusinessObject. Any BusinessObject can be a Parent (including BusinessObjects which are themselves children). Orixo supports multiple levels of parenting, or "grand-parenting", great-grand-parenting etc., and supports "multiple parenting" (for example where two or more tables both link to a shared child).
Examples of child data-tables are:
Any table that contains "items" under a "main" table will normally be a "child", such as SalesItems (under

3. Customers), WorkItems (under Staff), Ingredients (under Products).

Once a BusinessObject is set as a "Child" it will automatically be created "alongside" its parent in the System Entities Screen. In most other ways it will behave in exactly the same way as a "normal" BusinessObject.

"Child" datatables are extremely useful as a way of correctly **atomizing** data so that it fully follows Codd's rules of databases. Orixa strongly encourages proper atomization of data in this way.

4. **Normal** (Shown in the image above as "Products"). "BizObjType" = "N". This is the default type of BusinessObject. You can always change a Business Object's type, so it is usually best to start with it as "Normal" and change it once you know how your relational model is going to be organized.
5. **System or Framework BusinessObjects** (shown in the image above as "Types" and "Status") You do not create these data-tables. They are integral parts of the framework and used to help build lists and features of your App.