

# Configuration Tables in the database and the Database Catalog

Orixa contains a set of mechanisms to allow the Developer to refer to its database schema structures. This is useful as it allows Orixa Systems to be self-describing.

It is important to emphasize that the contents of the Configuration and Catalog data-tables are highly critical to the operation of any Orixa App, and SQL targeting these tables should be written with care. It is NOT possible to update or change any of these system tables, but it is possible to use them to DELETE, DROP and ALTER parts of your App, so SQL statements should be drafted with due care.

## The Configuration Database

```
SELECT * FROM Information.Tables
ORDER BY Name
```

Name	Description
Backups	NULL
Collations	NULL
DatabasePrivileges	NULL
Databases	NULL
DataTypes	NULL
FileIOStatistics	NULL
Files	NULL
Jobs	NULL
LogEvents	NULL
LoggedStatements	NULL
MigratorFunctions	NULL
MigratorParams	NULL
MigratorProcedures	NULL
Migrators	NULL
MigratorTables	NULL
MigratorViews	NULL
Modules	NULL
Roles	NULL
ServerSessionLocks	NULL
ServerSessions	NULL
ServerSessionStatistics	NULL
SessionStatistics	NULL
StorePrivileges	NULL
Stores	NULL
TextFilters	NULL
Updates	NULL
UserRoles	NULL
Users	NULL
WordGenerators	NULL

Configuration Database

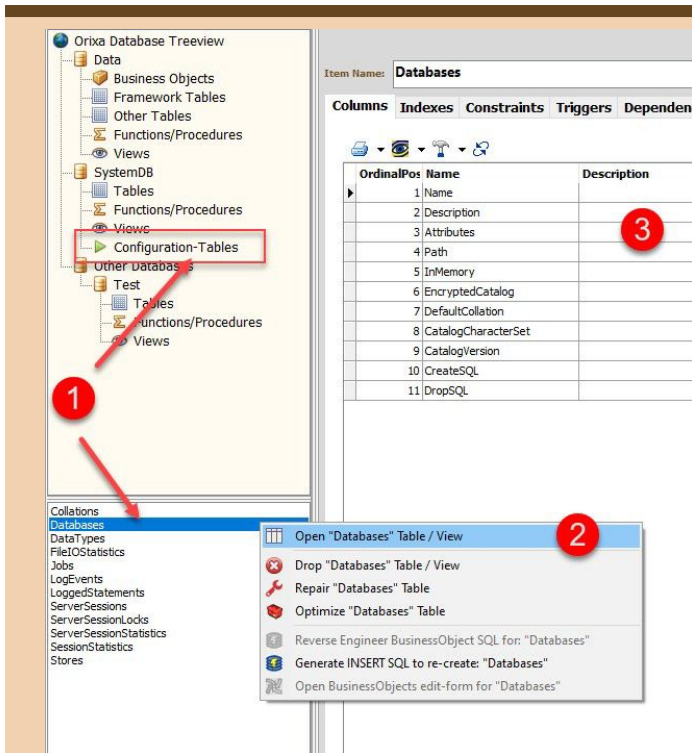
Each Configuration table contains data relating to your individual App's systems. A full list of the Configuration tables is:

**DataTypes**  
**Collations**  
**Modules**  
**TextFilters**  
**WordGenerators**  
**Migrators**  
**MigratorParams**  
**MigratorTables**  
**MigratorViews**  
**MigratorFunctions**  
**MigratorProcedures**  
**FileIOStatistics**  
**SessionStatistics**  
**LogEvents**  
**LoggedStatements**  
**Backups**  
**ServerSessions**  
**ServerSessionLocks**  
**ServerSessionStatistics**  
**Updates**  
**Users**  
**Roles**  
**UserRoles**  
**Databases**  
**DatabasePrivileges**  
**Jobs**  
**Stores**  
**Files**  
**StorePrivileges**

Only a few of these are of regular use for a developer. For example the "Users" configuration table can be queried to return a list of names of the Computers currently connected to the App.

A few of the configuration data-tables are made easily available via the Database Management Utility in your App.

1. Click on the "Configuration-Tables" item in the Database Treeview, and a list of tables will appear.
2. Select one table from the list and right-click to open a context menu from which you can Open the table to view it's records.



Viewing some of the Configuration data-tables

- Once any configuration data-table is selected all its details will display, including Columns, Indexes and Definition. Review the contents of these Columns to understand what the data-table is for and how you can use it.

## The Database Catalog Schema

As well as the Configuration database, Your database also contains a set of Catalog meta-data tables.

### Tables

**TablePrivileges**

**TableColumns**

**TemporaryTables**

**Constraints**

**ConstraintColumns**

**Indexes**

**IndexColumns**

**Triggers**

**TriggerColumns**

**Views**

**ViewPrivileges**

**ViewColumns**

**TemporaryViews**

**Procedures**

**ProcedurePrivileges**

**ProcedureParams**

**Functions**

**FunctionPrivileges**

**FunctionParams**

**Dependencies**

**SchemaObjects**

**SchemaDifference**

All of these tables can be accessed with SQL in the form:

```
SELECT * FROM Information.[catalog-tablename]
```

If you know the names of columns in these tables you can use these in the SELECT statement just like any ordinary SQL Statement.

## Examples of Catalog meta-data SQL Queries

System Information SQL Editor [+]

```

1 SELECT * FROM Information.TableColumns
2 WHERE TableName LIKE 'Subscription%'

```

Close

TableName	Name	Type
SubscriptionItems	Complete	Boolean
SubscriptionItems	Current	Boolean
SubscriptionItems	DateCreated	Timestamp
SubscriptionItems	DateEnd	Date
SubscriptionItems	DateStart	Date
SubscriptionItems	Description	CLOB
SubscriptionItems	ID	Integer
SubscriptionItems	Quantity	Float
SubscriptionItems	StatusID	Integer
SubscriptionItems	SubscriptionItemsTypeID	Integer
SubscriptionItems	SubscriptionsID	Integer
SubscriptionItems	Value	Decimal
Subscriptions	ContractsID	Integer
Subscriptions	Current	Boolean
Subscriptions	CustomersID	Integer
Subscriptions	DateCreated	Timestamp
Subscriptions	DateStart	Date
Subscriptions	Description	CLOB
Subscriptions	ID	Integer
Subscriptions	Key	VarChar
Subscriptions	ProductsID	Integer
Subscriptions	SubscriptionsTypeID	Integer
SubscriptionsLog	DateCreated	Timestamp
SubscriptionsLog	ID	Integer

1. A SQL Statement has been written using the Syntax Information.TableColumns. Note that a WHERE clause has also been added.
2. Data is returned that matches the criteria of the WHERE statement.

**SELECT \* FROM Information.TableColumns**

System Information SQL Editor [+]

```

1 SELECT CreateSQL, DropSQL FROM Information.Procedures
2

```

Close

CreateSQL	DropSQL
CREATE PROCEDURE "LoadUpdateAndRename" (IN "aFileName"	DROP PROCEDURE "LoadUpdateAndRename"
CREATE PROCEDURE "MergePeople" (IN "aOldID" INTEGER, IN	DROP PROCEDURE "MergePeople"
CREATE PROCEDURE "Monitoring_ListIDRanges" (IN "aTableList"	DROP PROCEDURE "Monitoring_ListIDRanges"
CREATE PROCEDURE "Monitoring_MaxIDs" ()	DROP PROCEDURE "Monitoring_MaxIDs"
CREATE PROCEDURE "Updates_SaveToTable" (IN "aFileName"	DROP PROCEDURE "Updates_SaveToTable"
CREATE PROCEDURE "Updates_SaveWholeStoreToTable" (IN	DROP PROCEDURE
CREATE PROCEDURE "Updates_Diagnosis" (IN "aStoreName"	DROP PROCEDURE "Updates_Diagnosis"
CREATE PROCEDURE "Maintenance_LoadSingleUpdate" (IN	DROP PROCEDURE
CREATE PROCEDURE "Maintenance_MergeStatusField" (IN	DROP PROCEDURE
CREATE PROCEDURE "Maintenance_MergeTypesField" (IN	DROP PROCEDURE
CREATE PROCEDURE "Maintenance_OptimizeAll" ()	DROP PROCEDURE "Maintenance_OptimizeAll"
CREATE PROCEDURE "Maintenance_RemoveDisconSessions" ()	DROP PROCEDURE
CREATE PROCEDURE "Maintenance_RepairAll" (OUT "RepairDone"	DROP PROCEDURE "Maintenance_RepairAll"
CREATE PROCEDURE "Maintenance_SetChildIDFieldNULL" (IN	DROP PROCEDURE
CREATE PROCEDURE "Updates_DeleteDuplicateFiles" (IN	DROP PROCEDURE
CREATE PROCEDURE "Updates_Execute" (IN "aUserID"	DROP PROCEDURE "Updates_Execute"
CREATE PROCEDURE "Updates_ExecuteOnServer" ()	DROP PROCEDURE "Updates_ExecuteOnServer"

1. A SQL Statement querying the Procedures catalog meta-data table,
2. The data that has been returned contains the SQL needed to either CREATE or DROP each Procedure.

**Information.Procedures catalog-table**