


## DeepSee: Databases, Namespaces, and Mappings - Part 1 of 5

Article

[Alessandro Marin](#) · Mar 27, 2018  3m read

## DeepSee: Databases, Namespaces, and Mappings - Part 1 of 5

I am planning to implement Business Intelligence based on the data in my instances. What is the best way to set up my databases and environment to use DeepSee?



This tutorial addresses this question by showing three examples of architecture for DeepSee. We will start from a basic architectural model and highlight its limitations. The subsequent model is recommended for Business Intelligence applications of intermediate complexity and should be sufficient for most use cases. We will end this tutorial by describing how to enhance the flexibility of the architecture for managing advanced implementations.

Each example in this tutorial introduces a new databases and global mappings, together with a discussion on why and when they should be set up. While building up the architecture, the benefits provided by the more flexible examples will be highlighted.

## Before starting

### Primary and Analytics servers

To make data highly available InterSystems generally recommends using mirroring or shadowing and base a DeepSee implementation on the mirror/shadow server. The machine hosting the original copy of the data is called the Primary server, whereas the machines hosting copies of the data and the Business Intelligence applications are often called the Analytics (or sometimes Reporting) servers.

Having Primary and Analytics servers is very important, the main reason being to avoid performance problems on either server. Please check the documentation about [Recommended Architecture](#).

### Data and application code

Storing source data and code in the same database usually works well only for small-scale applications. For more extensive applications it is recommended to store source data and code in two dedicated databases, which allows you to share code with all namespaces where DeepSee runs while keeping the data separate. The database for source data should be mirrored from the Production server. This database can be either Read-only or Read-Write. It is recommended to keep journaling enabled for this database.

Source classes and custom applications should be stored in a dedicated database on both the Production and Analytics servers. Note that these two databases for source code do not need to be in-synch or even run the same Caché version. Journaling is usually not needed, provided the code is regularly backed up elsewhere.

In this tutorial we will have the following configuration. The APP namespace on the Analytics server has the APP-DATA and the APP-CODE as default databases. The APP-DATA database has access to the data (the source table class and its facts) on the source data Database on the Primary. The APP-CODE database stores the Caché code (.cls and .INT files) and other custom code. This separation of data and code is a typical architecture and allows the user, for example, to efficiently deploy DeepSee code and custom application.

Namespaces

Create New Namespace
Refresh:  off  on  sec

### Current Namespaces and their default databases for globals and routines:

Filter:  Page size:  Max rows:  Results: 6 Page: [<](#) [<<](#) **1** [>>](#) [>](#) of 1

| Namespace               | Globals   | Routines  | Temp Storage |                                 |                                  |                                  |                        |
|-------------------------|-----------|-----------|--------------|---------------------------------|----------------------------------|----------------------------------|------------------------|
| %ALL                    | CACHETEMP | CACHETEMP | CACHETEMP    | <a href="#">Global Mappings</a> | <a href="#">Routine Mappings</a> | <a href="#">Package Mappings</a> | <a href="#">Delete</a> |
| %SYS                    | CACHESYS  | CACHESYS  | CACHETEMP    | <a href="#">Global Mappings</a> | <a href="#">Routine Mappings</a> | <a href="#">Package Mappings</a> | -                      |
| APP                     | APP-DATA  | APP-CODE  | CACHETEMP    | <a href="#">Global Mappings</a> | <a href="#">Routine Mappings</a> | <a href="#">Package Mappings</a> | <a href="#">Delete</a> |
| <a href="#">DOCBOOK</a> | DOCBOOK   | DOCBOOK   | CACHETEMP    | <a href="#">Global Mappings</a> | <a href="#">Routine Mappings</a> | <a href="#">Package Mappings</a> | <a href="#">Delete</a> |
| <a href="#">SAMPLES</a> | SAMPLES   | SAMPLES   | CACHETEMP    | <a href="#">Global Mappings</a> | <a href="#">Routine Mappings</a> | <a href="#">Package Mappings</a> | <a href="#">Delete</a> |
| <a href="#">USER</a>    | USER      | USER      | CACHETEMP    | <a href="#">Global Mappings</a> | <a href="#">Routine Mappings</a> | <a href="#">Package Mappings</a> | <a href="#">Delete</a> |

## Running DeepSee on different namespaces

Business Intelligence implementations using DeepSee often run from different namespaces. In this post we will show how to set up a single APP namespace but the same procedure applies to all namespaces where the business intelligence application runs.

## DeepSee: Databases, Namespaces, and Mappings - Part 1 of 5

Published on InterSystems Developer Community (<https://community.intersystems.com>)

---

### Documentation

It is recommended to get familiar with the documentation page [Perform the Initial Setup](#). This page includes setting up web applications, how to place DeepSee globals in separate databases, and a list of alternative mappings for DeepSee globals.

---

In the [second part](#) of this series we will show with the implementation of a basic architectural model

[#Analytics](#) [#Beginner](#) [#Best Practices](#) [#Databases](#) [#Deployment](#) [#Mapping](#) [#Tutorial](#) [#IRIS Analytics \(DeepSee\)](#)

30 5 1 4 705

Related posts

- [DeepSee: Databases, Namespaces, and Mappings - Part 1 of 5](#)
- [DeepSee: Databases, Namespaces, and Mappings - Part 2 of 5](#)
- [DeepSee: Databases, Namespaces, and Mappings - Part 3 of 5](#)

[Show all](#)

Log in or sign up to continue

Add reply

**Source URL:** <https://community.intersystems.com/post/deepsee-databases-namespaces-and-mappings-part-1-5>