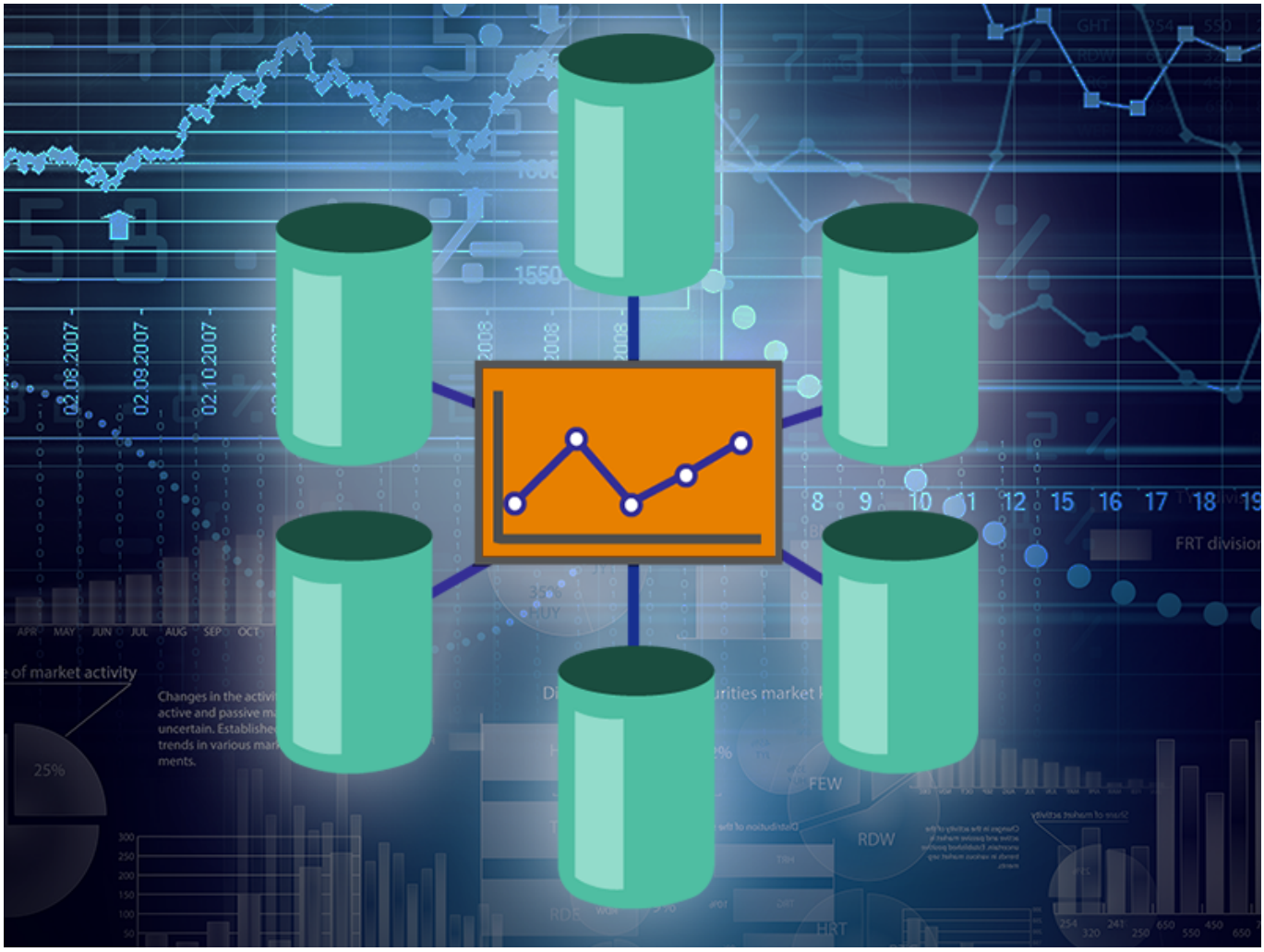
文章
Nicky Zhu · 一月 8, 2021 阅读大约需 3 分钟

DeepSee：数据库、命名空间和映射 - 第1部分，共 5 部分

我打算基于实例中的数据实现业务智能。 怎样才是设置数据库和环境来使用 DeepSee 的最佳方法呢？



本教程通过 3 个 DeepSee 架构示例来解决此问题。 首先，我们从基本架构模型开始，并重点说明其局限性。
对于复杂程度中等的业务智能应用，建议使用下一个模型，对于大多数用例而言，该模型应该足矣。
在本教程的最后，我们将说明如何增强架构的灵活性以管理高级实现。

本教程中的每个示例都介绍了新的数据库和全局映射，并讨论了为何以及何时设置它们。
在构建架构时，则重点说明更灵活的示例提供的好处。

开始前

主服务器和分析服务器

为了使数据高度可用，InterSystems 通常建议使用镜像或映射，并将 DeepSee 实现基于镜像/映射服务器。 承载数据原始副本的机器称为 “ 主服务器 ”，而承载数据副本和业务智能应用程序的计算机通常称为 “ 分析服务器 ”（有时称为 “ 报告服务器 ”）。

拥有主服务器和分析服务器至关重要，主要原因是避免任一台服务器出现性能问题。 请查阅有关推荐架构的文档。

数据和应用程序代码

通常，将源数据和代码存储在同一数据库中仅对小型应用程序有效。
对于更大型的应用程序，建议将源数据和代码存储在两个专用数据库中，这样您就可以与运行 DeepSee 的所有命名空间共享代码，同时保持数据分离。 源数据的数据库应从生产服务器镜像。
该数据库可以为只读，也可为读写。 建议为此数据库持续启用日志功能。

源类和自定义应用程序应存储在生产和分析服务器上的专用数据库中。
请注意，这两个用于源代码的数据库不需要同步，甚至不需要运行相同的 Caché 版本。
只要定期将代码备份到其他地方，通常就不需要日志。

在本教程中，我们采用以下配置。 分析服务器上的 APP 命名空间有 APP-DATA 和 APP-CODE 作为默认数据库。
APP-DATA 数据库可以访问主服务器上的源数据数据库中的数据（源表类及其事实数据）。 APP-CODE 数据库存储 Caché 代码（.cls 和 .INT 文件）以及其他自定义代码。
数据和代码的这种分离是一种典型的架构，这允许用户，例如，有效地部署 DeepSee 代码和自定义应用程序。

在不同的命名空间上运行 DeepSee

使用 DeepSee 的业务智能实现通常在不同的命名空间中运行。 在本文中，我们将说明如何设置单个的 APP 命名空间，但是相同的过程适用于运行业务智能应用程序的所有名称空间。

文档

建议熟悉文档页面执行初始设置。 该页面的内容包括：设置 Web 应用程序，如何将 DeepSee 全局变量放置在单独的数据库中，以及 Deepeep 全局变量的替代映射列表。

在本系列的第二部分中，我们将阐述基本架构模型的实现

#映射 #分析 #教程 #数据库 #新手 #部署 #InterSystems IRIS BI (DeepSee)

源 URL:
https://cn.community.intersystems.com/post/deepsee%EF%BC%9A%E6%95%B0%E6%8D%AE%E5%BA%93%E3%80%81%E5%91%BD%E5%90%8D%E7%A9%BA%E9%97%B4%E5%92%8C%E6%98%A0%E5%B0%84-%E7%AC%AC1%E9%83%A8%E5%88%86%EF%BC%8C%E5%85%B1-5-%E9%83%A8%E5%88%86