文章
Louis Lu · 四月 15, 2021 阅读大约需 15 分钟

精华文章--访问IRIS数据平台的四种方式

IRIS 中支持的四种方式：

SQL、Objects、REST 和 GraphQL

卡济米尔·马列维奇，《运动员》(1932)

>

> “ 你当然无法理解！ 习惯了坐马车旅行的人怎么可能理解乘坐火车或者飞机旅行的人的感受和印象？ ”
>

> >

> 卡济米尔·马列维奇 (1916)
>

引言

我们已经讨论过为什么在主题领域建模使用对象类型优于使用 SQL。 当时得出的结论和总结的事实如今依然适用。
那么，我们为什么要退后到对象和类型之前的时代，讨论将对象的操作拖回到使用global的技术？
我们又为什么要鼓励面条式代码？难道是为了用它难以跟踪的错误考验开发者的技能熟练度？

目前有几种观点支持通过基于 SQL/REST/GraphQL 的 API 传输数据，而不是将其表示为类型/对象：

- 这些技术经过深入研究，相当易于部署。
- 知名度非常高，已在便捷的开源软件中广泛实现。

- 您通常别无选择，只能使用这些技术，尤其是在网络和数据库中。
- 最重要的是，API 仍然使用对象，因为它们提供了在代码中实现 API 的最适途径。

在讨论实现 API 之前，我们先来看一下底层的抽象层。
下图显示了数据在永久存储位置与处理并向应用程序用户呈现的位置之间的移动方式。

如今，数据存储在旋转硬盘驱动器 (HDD) 上，或者使用更现代的技术存储在 SSD 的闪存芯片中。 再使用由 HDD/SSD 上独立存储块组成的流完成数据的读取和写入。

分块并不是随机的， 而是由数据存储介质的物理学、力学和电子学特性决定。 在 HDD 中，这是旋转磁盘上的磁道/扇区。 在 SSD 中，这是可写硅芯片中的内存段。

它们本质上都是信息块，只有找到这些信息块并将其组合才能检索出需要的数据。
数据必须装配成将我们的数据模型/类型与查询时间对应值相匹配的结构。
数据的装配和检索过程由与操作系统中的文件子系统捆绑的 DBMS 负责。

我们可以直接寻址文件系统甚至 HDD/SSD，绕过 DBMS。 但这样一来就失去了两个极其重要的数据桥梁：存储块和文件流之间的桥梁，以及文件和数据库模型中的有序结构之间的桥梁。
换句话说，我们要负责开发所有处理块、文件和模型的代码，包括所有的优化、精细调试和长期可靠性测试。
DBMS 使用可理解的模型和表示形式，给我们提供了以高级语言处理数据的绝好机会。
这也是这些系统的一大优势。 DBMS 和数据平台（如 InterSystems IRIS ）提供了更多功能：能够以多种方式同时访问有序数据。 而在您的项目中，使用哪一种则取决于您自己。

利用 IRIS 提供的各种工具， 我们可以让代码更美观、更简洁。 我们将直接通过 ObjectScript 面向对象的语言来利用和开发 API。 也就是说，例如，我们会直接从 ObjectScript 软件内部调用 SQL 代码。
对于其他 API，我们将使用现成的库和内置的 ObjectScript 工具。

我们将以 SQLZoo Internet 项目为例，该项目提供了 SQL 的学习资源。 我们将在其他 API 示例中使用相同的数据。

> 如果您想了解 API 设计的各种方法并利用现成的解决方案，这里有一个有趣又实用的公共 API 集合，汇集到 GitHub 上的一个项目中。

SQL

最自然的方法莫过于从 SQL 开始。 这里没人对它不熟悉吧？

SQL 的教程和书籍浩如烟海。 我们选择的是 SQLZoo。 这是一门很好的 SQL 入门课程，其中有示例，有操作步骤详解，也有语言引用。

我们将一些任务从 SQLZoo 带到 IRIS 平台，并使用各种方法解决。

> 您可以用多快的速度在您的电脑上访问 InterSystems IRIS？ 最快的选择之一是在 Docker 中从现成的 InterSystems IRIS Community Edition 镜像部署容器。
>
> InterSystems IRIS Community Edition 是 InterSystems IRIS Data Platform 的免费开发者版本。
>
> 其他方式：在学习门户中访问 InterSystems IRIS Community Edition

将数据从 SQLZoo 转移到我们自己的 IRIS 实例存储中。

需要执行以下操作：

- 打开管理门户（以我的为例，http://localhost:52773/csp/sys/UtilHome.csp）。
- 切换到 USER 命名空间 - 在 Namespace %SYS 中，点击 “ Switch ” 链接，选择 USER

- 转到 System > SQL - 依次打开 System Explorer、SQL，然后点击 “ Go ” 按钮。
- 右侧将打开 “ Execute query ” 标签页，带有 “ Execute ” 按钮，这就是我们需要的。

要详细了解如何通过管理门户网站使用 SQL，请参阅此文档。

在 Data 部分的描述中查看用于部署数据库和 SQLZoo 测试数据集的现成脚本。

这里有 world 表的几个直接链接：

- 用于创建 world 数据库的脚本
- 进入该表的数据

创建数据库的脚本可以在 IRIS 管理门户的 Query Executor 表单中执行。

```
CREATE TABLE world(
   name VARCHAR(50) NOT NULL
  ,continent VARCHAR(60)
  ,area DECIMAL(10)
  ,population DECIMAL(11)
  ,gdp DECIMAL(14)
  ,capital VARCHAR(60)
  ,tld VARCHAR(5)
  ,flag VARCHAR(255)
  ,PRIMARY KEY (name)
)
```

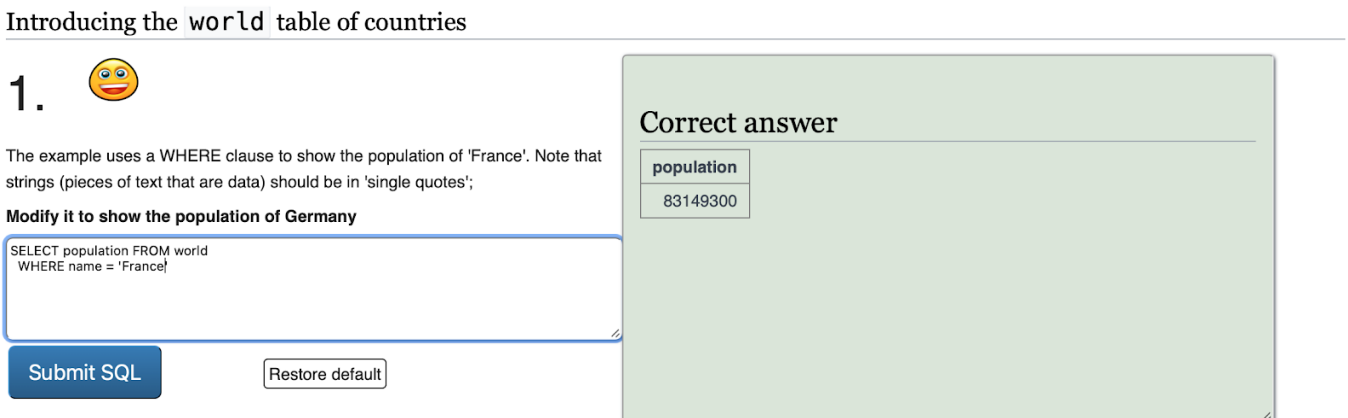
要加载 Query Executor 表单的测试集，首先转到 Wizards > Data Import 菜单。
请注意，在创建容器时，必须预先添加带有测试数据文件的目录，或者通过浏览器从计算机中加载。
该选项在数据导入向导的控制面板中可用。

在 Query Executor 表单中运行以下脚本，检查带有数据的表是否存在：

```
SELECT * FROM world
```

接下来可以从 SQLZoo 网站访问示例和任务。 以下所有示例均要求您在第一次赋值中实现 SQL 查询：

```
SELECT population    FROM world   WHERE name = 'France'
```



这样就可以将任务从 SQLZoo 转移到 IRIS 平台，保持与 API 的无缝协作。

请注意：我发现，SQLZoo 网站界面的数据与导出的数据不同。
至少在第一个示例中，法国和德国的人口值是不同的。 不必过度纠结。

使用[欧洲统计局数据](https://ec.europa.eu/eurostat/tgm/table.do?tab=table&language=en&pcode=t...)作为参考。

> 另一种在 IRIS 中获得数据库 SQL 访问的便捷途径是 Visual Studio 代码编辑器与 SQLTools 插件和 SQLTools Driver for InterSystems IRIS。 这种解决方案很受开发者的欢迎，不妨一试。

为了顺利进行下一步并获得对我们数据库的对象访问权，让我们绕一个小弯，从 “ 纯粹 ” 的 SQL 查询转到[在 ObjectScript
中嵌入应用程序代码](https://docs.intersystems.com/iris20203/csp/docbook/Doc.View.cls?KEY=GSQ...)的 SQL 查询，ObjectScript 是 IRIS 内置的一种面向对象的语言。

> 如何在 VSCode 中设置 IRIS 访问并使用 ObjectScript 进行开发。

```
Class User.worldquery {
  ClassMethod WhereName(name As %String)
  {
    &sql(
          SELECT population INTO :population
            FROM world
          WHERE name = :name
    )
    IF SQLCODE<0 {WRITE "SQLCODE error ",SQLCODE," ",%msg  QUIT}
    ELSEIF SQLCODE=100 {WRITE "Query returns no results"  QUIT}
    WRITE name, " ", population }
}
```

在终端中查看结果：

```
do ##class(User.worldquery).WhereName("France")
```

您应该会收到国家/地区名称和居民人数的响应。

对象 类型

接下来是 REST/GraphQL 的部分。 我们正在为 Web 协议实现一个 API。
大多数情况下，服务器端的后台源代码是以良好支持类型的语言开发的，甚至是完全面向对象的范式编写的。
包括：Java/Kotlin 中的 Spring、Python 中的 Django、Ruby on Rails、C# 中的 ASP.NET或者 TypeScript 中的 Angular。 当然，还有 IRIS 平台原生的 ObjectScript 中的对象。

为什么这很重要？ 因为在发送时，代码中的类型和对象将简化为数据结构。
您需要考虑如何在程序中简化模型，这类似于考虑关系模型中的损失。 您还需要确保在 API
的另一侧，模型得到了充分的还原，可以不失真地投入使用。
这就带来了额外的负担：您作为程序员需要承担额外的责任。
在代码之外，除去转换器、编译器和其他自动工具的帮助，您还需要不断确保模型得到正确转移。

如果从另一个角度看待上述问题，我们还没有发现任何技术和工具可以用来轻松地将类型/对象从一种语言的程序转移到另一种语言的程序。 还剩下什么？ 有 SQL/REST/GraphQL 的简化实现，以及大量用人类友好语言描述 API 的文档。 面向开发者的非正式（从计算机的角度）文档具体描述了应使用所有可用方法将哪些内容转换为正式代码，以便计算机处理。

程序员一直在开发不同的方法来解决上述问题。 其中一个成功的方法是 IRIS 平台对象 DBMS 中的跨语言范式。

下表应该可以帮助您理解 IRIS 中 OPP 与 SQL 模型之间的关系：

面向对象编程 (OOP)	结构化查询语言 (SQL)
包	Schema

类	表
属性	列
方法	存储过程
两个类之间的关系	外键约束，内置联接
对象（在内存中或磁盘上）	行（在磁盘上）

您可以从 [IRIS 文档]中详细了解如何显示对象和关系模型。

执行我们的 SQL 查询根据示例创建 world 表时，IRIS 将在名为 User.world 的类中自动生成对应对象的描述。

```
Class User.world Extends %Persistent [ ClassType = persistent, DdlAllowed, Final, Own
er = {_SYSTEM}, ProcedureBlock, SqlRowIdPrivate, SqlTableName = world ]
{
  Property name As %Library.String(MAXLEN = 50) [ Required, SqlColumnNumber = 2 ];
  Property continent As %Library.String(MAXLEN = 60) [ SqlColumnNumber = 3 ];
  Property area As %Library.Numeric(MAXVAL = 9999999999, MINVAL = -9999999999, SCALE
= 0) [ SqlColumnNumber = 4 ];
  Property population As %Library.Numeric(MAXVAL = 99999999999, MINVAL = -99999999999
, SCALE = 0) [ SqlColumnNumber = 5 ];
  Property gdp As %Library.Numeric(MAXVAL = 99999999999999, MINVAL = -99999999999999,
 SCALE = 0) [ SqlColumnNumber = 6 ];
  Property capital As %Library.String(MAXLEN = 60) [ SqlColumnNumber = 7 ];
  Property tld As %Library.String(MAXLEN = 5) [ SqlColumnNumber = 8 ];
  Property flag As %Library.String(MAXLEN = 255) [ SqlColumnNumber = 9 ];
  Parameter USEEXTENTSET = 1;
  /// Bitmap Extent Index auto-
generated by DDL CREATE TABLE statement.  ???????? SqlName ?
  Index DDLBEIndex [ Extent, SqlName = "%%DDLBEIndex", Type = bitmap ];
  /// DDL Primary Key Specification
  Index WORLDPKey2 On name [ PrimaryKey, Type = index, Unique ];
}
```

您可以用这个模板来开发面向对象风格的应用程序。 您需要做的就是向 ObjectScript 中的类添加方法，ObjectScript 天生与数据库捆绑。 事实上，这个类的方法按 SQL 术语来说就是 “ 存储过程 ” 。

让我们用 SQL 实现之前完成的同一个示例。 将 WhereName 方法添加到 User.world 类中，充当输入的国家/地区名称的 “ Country information ” 对象设计器：

```
ClassMethod WhereName(name As %String) As User.world
{
  Set id = 1
  While ( ..%ExistsId(id) ) {
    Set countryInfo = ..%OpenId(id)
    if ( countryInfo.name = name ) { Return countryInfo }
    Set id = id + 1
  }
  Return countryInfo = ""
}
```

在终端中查看：

```
set countryInfo = ##class(User.world).WhereName("France")



write countryInfo.name
```

```
write countryInfo.population
```

从此示例可以看出，与 SQL
查询不同，为了通过国家/地区名称查找所需对象，我们需要手动对数据库记录进行逐一排序。 在最坏情况下，如果我们的对象位于列表末尾（或根本不在列表中），我们就不得不从头到尾将所有记录都整理一遍。 关于如何在 IRIS 中通过索引对象字段和自动生成类方法来加快搜索过程有一个单独的讨论。
有关更多详情，请参阅[文档]和[开发者社区门户]中的帖子。

例如，对于我们的类，了解 IRIS 从 WORLDPKey2
国家/地区名称生成的索引名称后，您可以使用单个快速查询从数据库中直接加载一个对象：

```
set countryInfo = ##class(User.world).WORLDPKey2Open("France")
```

同时检查：

```
write countryInfo.name

write countryInfo.population
```

此文档中包含一些指导原则，可以帮助您决定使用对象还是 SQL 访问存储的内容。

当然，您也要始终牢记，您只能将其中之一完全用于您的任务。

此外，由于 IRIS 中提供了现成的二进制捆绑包，支持常见的 OOP 语言，例如
Java、Python、C、C#(.Net)、JavaScript，甚至是正在迅速普及的 Julia（参见 GitHub 和 OpenExchange ），您总能找到最方便的语言开发工具。

接下来，让我们深入讨论 Web API 中的数据。

REST 或 RESTful Web API

跳出服务器和常见终端，转向一些更主流的接口：浏览器和类似应用程序。 这些应用程序依靠 HTTP
系列的超文本协议管理系统之间的交互。 IRIS 自带许多适合这个目的的工具，包括一个实实在在的数据库服务器和 Apache HTTP 服务器。

> 表述性状态转移 (REST) 是一种架构样式，用于设计分布式应用程序，尤其是 Web 应用程序。 REST 虽然流行，但它只是一套架构原则，而 SOAP 则是由 World Wide Web Consortium (W3C) 维护的标准协议，因此基于 SOAP 的技术具有标准支持。

REST 中的全局 ID 是 URL，当与数据库或后端应用程序交换时，由它定义每个连续的信息单元。 请参阅在 IRIS 中开发 REST 服务的文档。

在我们的示例中，基础标识符类似于 IRIS 服务器地址的基础 (http://localhost:52773)，以及指向我们数据的 /world/ 子目录路径。 特别是我们的 /world/France 国家/地区目录。

在 Docker 容器中类似于：

http://localhost:52773/world/France

如果要开发完整应用程序，务必查看 IRIS 文档推荐。 其中之一基于遵循 OpenAPI 2.0 规范的 REST API 描述。

让我们用简单的方法，手动实现 API。 在示例中，我们将创建最简单的 REST 解决方案，这个解决方案在 IRIS

中只需要两个步骤：

- 在 URL 内创建一个类路径监视器，它将继承 %CSP.REST 系统类
- 配置 IRIS web application时，添加对我们的监视器类的调用

第 1 步：创建类监视器

您应该清楚如何实现一个类。 按照文档中的说明手动创建REST。

```
Class User.worldrest Extends %CSP.REST
{
  Parameter UseSession As Integer = 1;
  Parameter CHARSET = "utf-8";
  XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
  { <Routes>
      <Route Url="/:name" Method="GET" Call="countryInfo" />
    </Routes>
  }
}
```

确保包括处理程序方法。 它执行的功能应该与先前示例中终端的调用完全相同：

```
ClassMethod countryInfo(name As %String) As %Status
{
  set countryInfo = ##class(User.world).WhereName(name)
  write "Country: ", countryInfo.name
  write "<br>"
  write "Population: ", countryInfo.population
  return $$$OK
}
```

如您所见，以冒号开头的参数 :name 表示从传入的 REST
查询将监视器中被调用的处理程序方法的名称传递给参数。

第 2 步：配置 IRIS web application

在 System Administration > Security > Applications > Web Applications 下， 添加一个新的 web application，其 URL 入口地址为 /world，并添加以下处理程序：our worldrest monitor class。

配置完成后，转到 http://localhost:52773/world/France 时，web application应立即响应。
请记住，数据库区分大小写，因此在向方法参数传输请求数据时必须使用正确的字母大小写。

小诀窍：

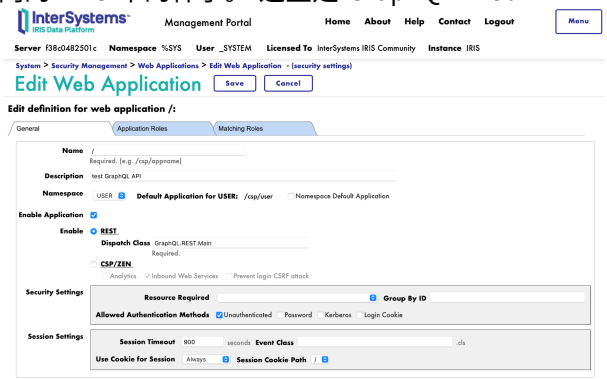
- 根据需要使用调试工具。
  您可以在这篇由两部分组成的文章中找到很好的描述（评论也值得一看）。 >
- 如果出现错误“ 401 Unauthorized ”，并且您确定监视器类在服务器上，链接中也没有错误，请尝试在 Web 应用程序设置的 Application Roles 标签页添加 %All 角色。 这不是一种完全安全的方法，您需要了解允许访问所有角色可能带来的影响，但是这对于本地安装是可以接受的。

GraphQL

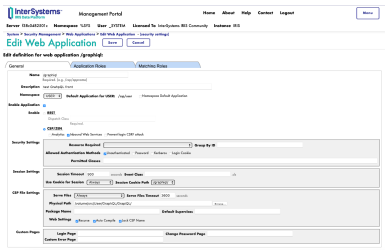
这是一个全新的领域，因为您在当前的 IRIS 文档中找不到关于使用 GraphQL 的 API 的任何内容。
然而，我们不能就这样停止使用这个出色的工具。

> 距 GraphQL 公开推出不过五年。 GraphQL 由 Linux Foundation 开发，是 API 的查询语言。
> 可以肯定地说，这是 REST 架构和各种 Web API 改进所带来的最好的技术。
> 这是是一篇简短介绍，供初学者参考。 并且，在 InterSystems 爱好者和工程师的努力下，IRIS 从 2018 年开始提供 GraphQL 支持。 > 这是一篇相关文章《为 InterSystems 平台实现 GraphQL》。

这里是 GraphQL 理解、解释和实现。 GraphQL 应用程序由两个模块组成：IRIS 侧的应用程序后端和在浏览器中运行的前端部分。 换句话说，您需要根据 GraphQL 和 GraphiQL Web 应用程序的说明进行配置。 例如，这是我的应用程序配置在 Docker 容器内的 IRIS 中的样子。 这些是 GraphQL Web 应用程序（充当 REST 监视器和数据库架构处理程序）的设置：



第二个 GraphQL 应用程序是浏览器的用户界面，使用 HTML 和 JavaScript 编写：



可前往 http://localhost:52773/graphiql/index.html 运行。
如果没有额外限制性设置，应用程序将立即拾取在安装区域中能找到的所有数据库schema。
这意味着我们的示例将立即开始工作。 另外，前端还提供了来自可用对象的一系列良好清晰的提示。

这是我们数据库的示例 GraphQL 查询：

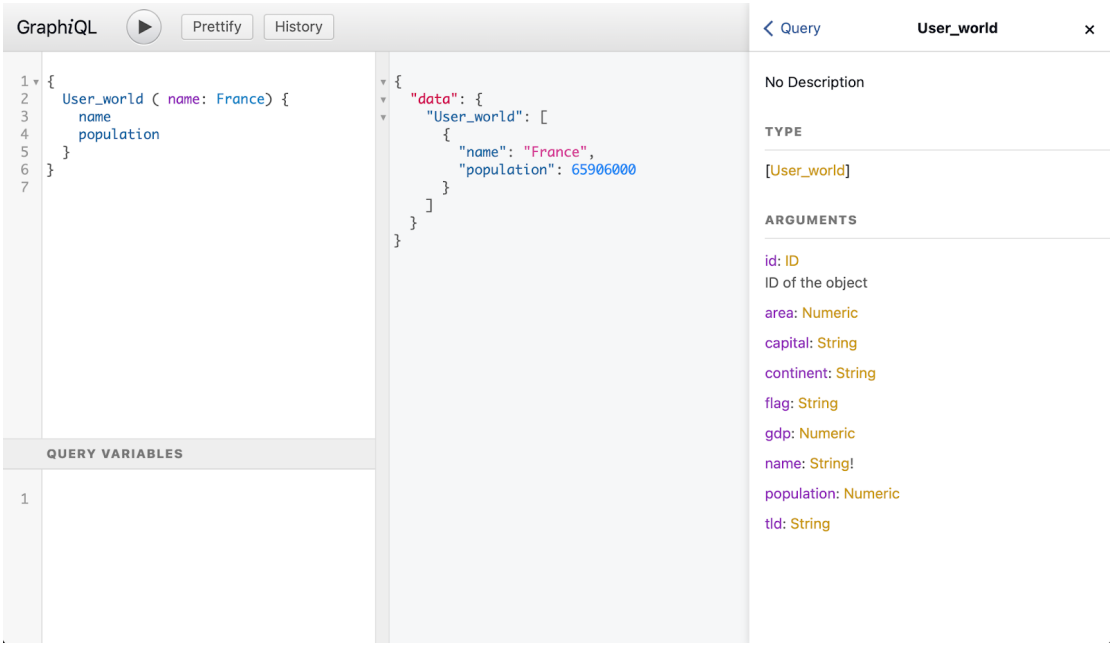
```
{
  User_world ( name: France ) {
    name
    population
  }
 }
```

这是匹配的响应：

```
{
  "data": {
     "User_world": [
       {
         "name": "France",
         "population": 65906000
```

```
        }
       ]
  }
}
```

在浏览器中是这样：



总结

技术名称	技术历史	查询示例
SQL	50 年 Edgar F. Codd	`SELECT population FROM world WHERE name = 'France'`
OOP	40 年 Alan Kay 和 Dan Ingalls	`set countryInfo = ##class(User.world).WhereName("France")`
REST	20 年 Roy Thomas Fielding	http://localhost:52773/world/France
GraphQL	5 年 Lee Byron	`{ User_world ( name: France ) { name population } }`

- SQL、REST，可能还有 GraphQL 等技术均已获得大量投入。 它们的背后也都有着丰富的历史。 在 IRIS 平台内，这些技术都可以良好配合，创建数据处理程序。
- 虽然本文未提及，但 IRIS 也支持其他已充分实现的基于 XML (SOAP) 和 JSON 的 API。
- 除非您会特别注意，比如整理对象，否则，请记住通过 API
  交换的数据仍然代表不完整的、精简的对象传输版本。
  作为开发者，您（而不是代码）有责任确保正确地传输对象的数据类型信息。

尊敬的读者，请您提供反馈

本文的目的不仅仅是与现代 API 进行比较，更不是要回顾 IRIS 的基本功能。 它是为了帮助您了解，访问数据库时在 API 之间切换是多么容易。向 IRIS 迈出第一步，马上就能从任务中快速获得结果。 因此，我很想知道您的想法：

这种方式能否帮助您上手使用软件？

哪些流程步骤使您难以掌握用于在 IRIS 中使用 API 的工具？

您能说出一个您先前无法预见的障碍吗？

如果您认识仍在学习如何使用 IRIS 的用户，请让这些用户在下方发表评论。
这可以引起让所有参与者都能受益的讨论。

#API #对象数据模型 #数据模型 #新手 #InterSystems IRIS

源 URL: https://cn.community.intersystems.com/post/%E7%B2%BE%E5%8D%8E%E6%96%87%E7%AB%A0-%E8%AE%BF%E9%97%AEiris%E6%95%B0%E6%8D%AE%E5%B9%B3%E5%8F%B0%E7%9A%84%E5%9B%9B%E7%A7%8D%E6%96%B9%E5%BC%8F