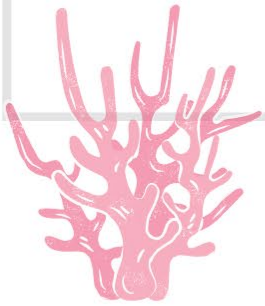
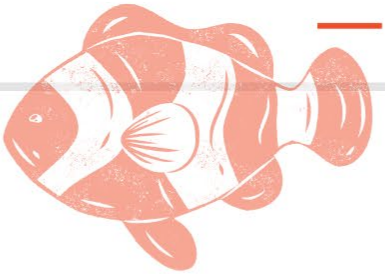
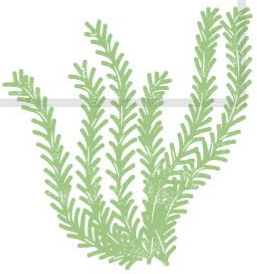
NetSuite 知识会汇编

—— 管理员篇 & 顾问篇 2023 ——







德之匠信息技术

序 言

本月初，开学之际，我们发布了《NetSuite 知识会汇编-用户篇 2023》，这次发布《NetSuite 知识会汇编-管理员篇&顾问篇 2023》。本篇挑选了近两年 NetSuite 知识会中的一些文章，涉及开发、权限、系统管理等较深的内容，共 19 篇。阅读对象是 NetSuite 的系统管理员与实施顾问。

中国的 NetSuite 系统管理员和顾问群体加起来不足千人，实在是个小众群体。人少、资源少，这是 NetSuite 在中国的现状。水浅王八多，我们在折腾。如果知识会能够在 NetSuite 生态中翻出些小水花，大家能会心一笑，也不枉我们的文字。

我们碰巧在早些年接触到这个产品，作为“云 ERP”的活化石，NetSuite 中有些东西值得我们学习。它“开发平台+应用模块”的模式跟 Salesforce 一脉相承，花开两朵，时间证明是正确的。不能方便客制的 ERP 软件就是耍流氓，不能客制的就根本是在谋财害命。即使是 SaaS 模式的 ERP 软件，也需要支持客制。我们说 NetSuite 的价值一半在 SuiteCloud，就是这个原因。系统管理员和顾问群体作为专业用户，需要在 SuiteCloud 上投入精力，呈现大家的专业价值。

基本上，NetSuite 的产品发行策略就是“产品骨干功能+开发平台+SuiteApp”，把毛细功能的开发留给了服务商和专业用户。在这个 AI 初纪元，这给了“NetSuite 超级顾问”以一个巨大的机会。所谓超级顾问是指熟悉 NetSuite 功能和开发，同时在甲方工作或者熟悉甲方业务，TA 们可以借助 AI 完成大部分的代码，实现单兵作战。TA 们可以最高效率的完成这些毛细功能的实现。TA 们的时代到来了！

NetSuite 知识会有志于成为团结超级顾问的平台，希望与各位有缘人一起做些有趣的东西。在 NetSuite 领域，遥遥领先！

愿你我做一个始终追求卓越的行者、匠人。

Rick

2023-9-13

目 录

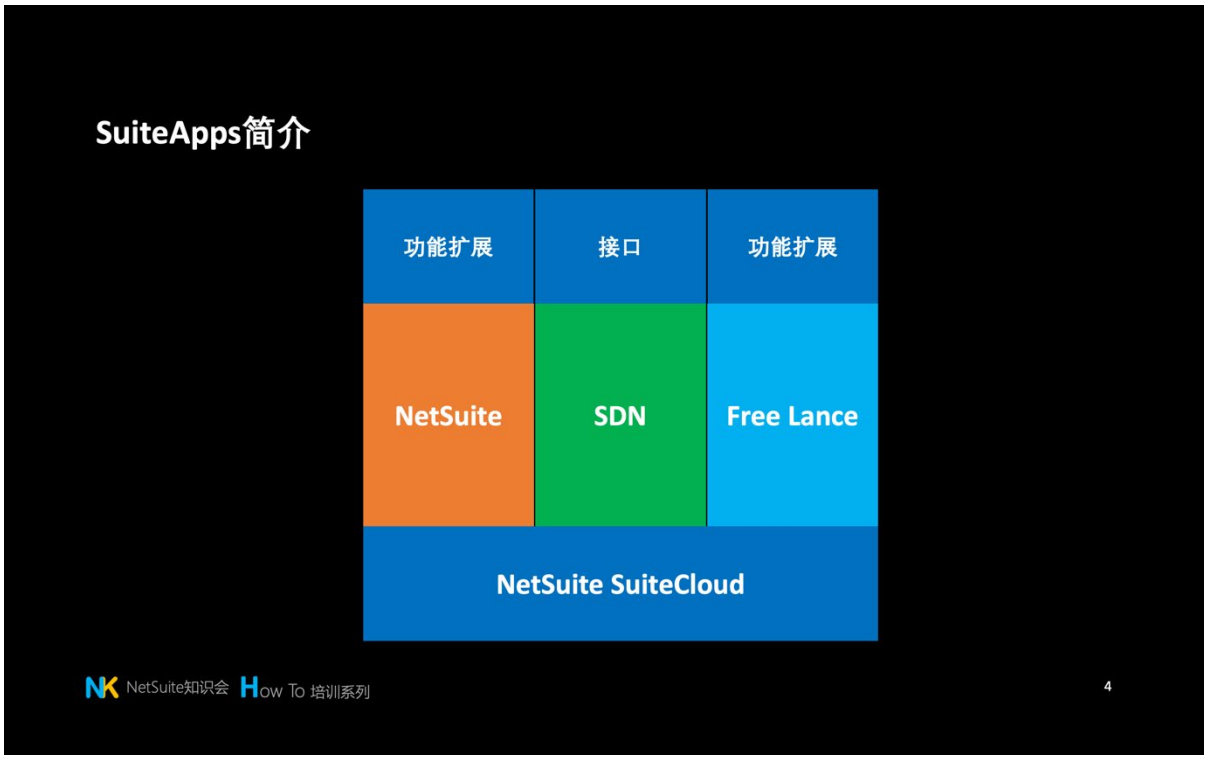
第一部分　管理员篇

1. 功能

SuiteApps—隐秘的功能宝藏

作者：Rick　2022.10.29

在近期的项目之中，我们发现大家对于 SuiteApps 的很多内容不是特别了解，所以也想和大家借 NetSuite 知识会第六讲的时间，通过认识十个常用的 SuiteApps 来揭露一个你所不知道的 NetSuite。



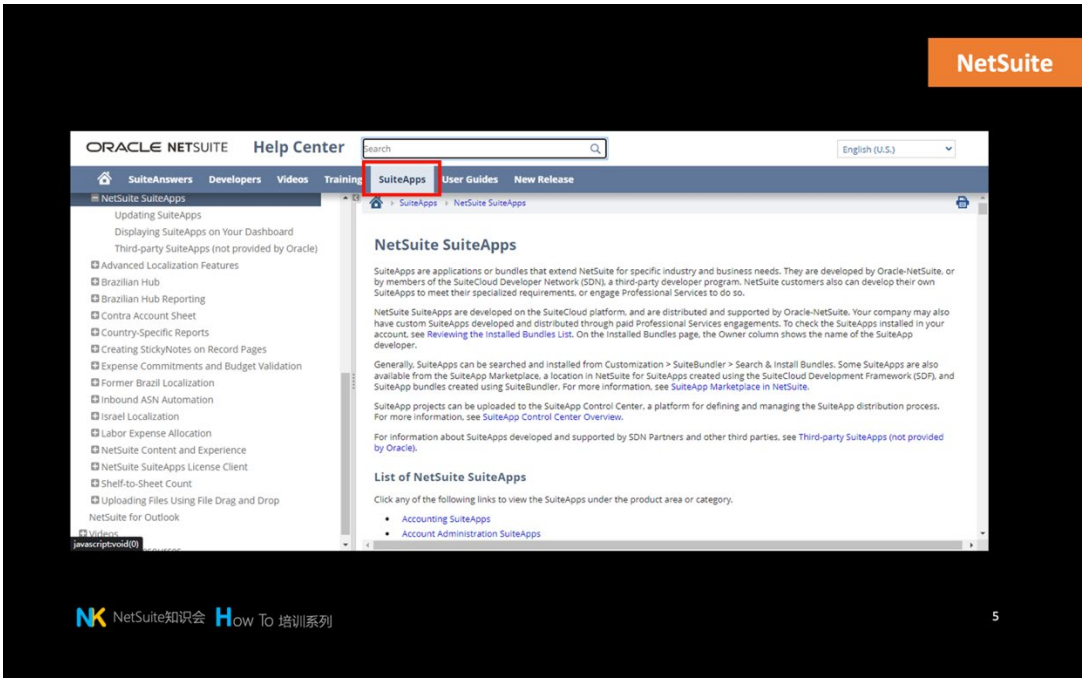
所谓 SuiteApps，其实是基于 NetSuite 的技术框架来做的——以 SuiteCloud 为扩展平台，利用其开发组件，做出自己的应用最终发布。

SuiteApps 的主要来源涉及以下三个部分：

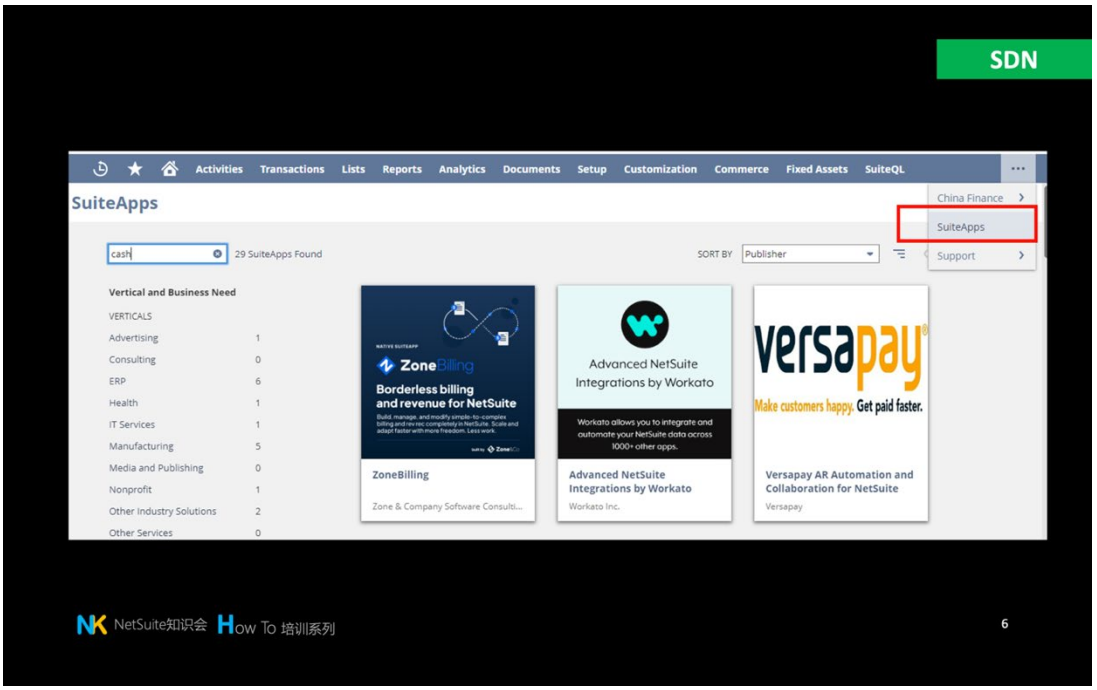
- **NetSuite**：原厂是大部分内容的直接贡献者，他们可以自己开发或者收购，比如固定资产模块，高级生产模块等；
- **SDN**：SuiteCloud Development Network，是第三方的厂商在 NetSuite 平台上与 NetSuite 做结合后产生的产品，更注重产品之间的联系，可以称第三方厂商为“合作伙伴”，比如易快报等；

- **Free Lance**：是逐渐兴起的“游击队”，所做的应用其中不乏一些“高质量”的产品，比如我后面会提到的 SuiteQL。

总结下来，SuiteApps 就是在以 NetSuite 原厂为主要贡献者的基础上，通过几方努力将形成的基于 NetSuite 核心的扩展包以 Bundle 的形式给到大家，这其中有免费的也有付费的。

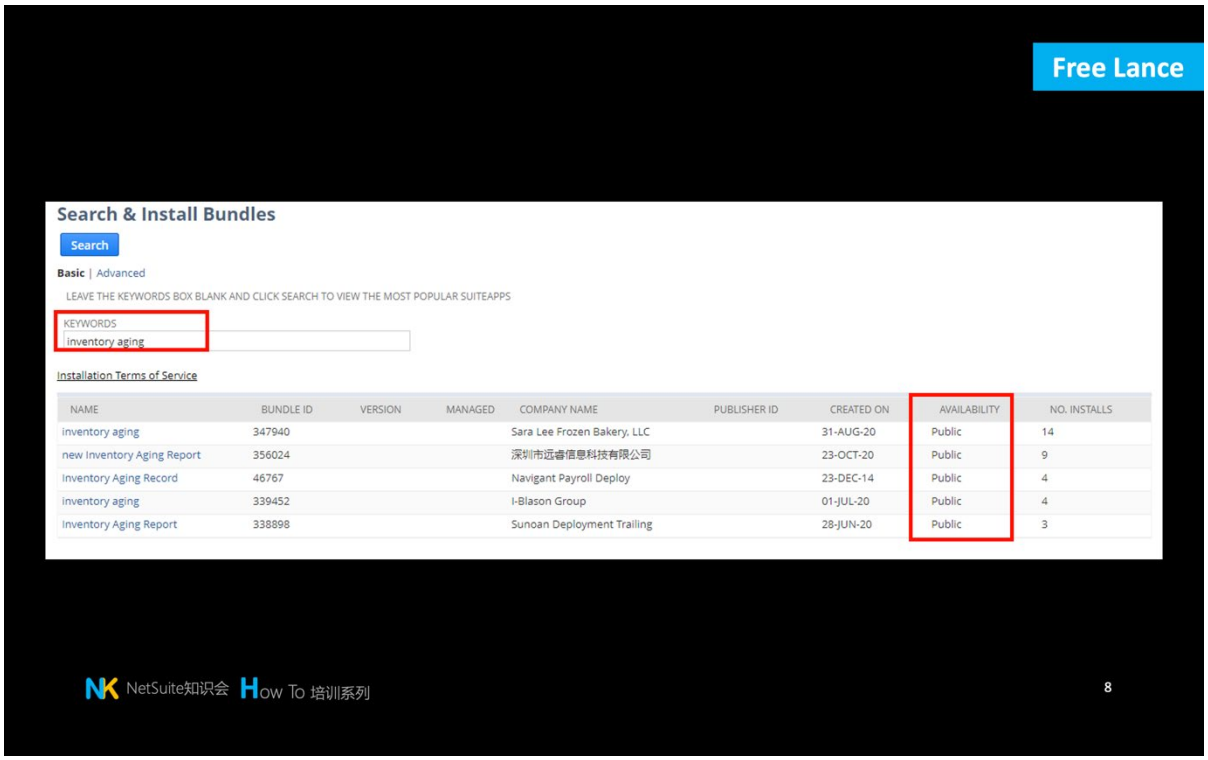


从“帮助”中进入，是官方版 NetSuite SuiteApps 的进入路径，可看到相关内容。



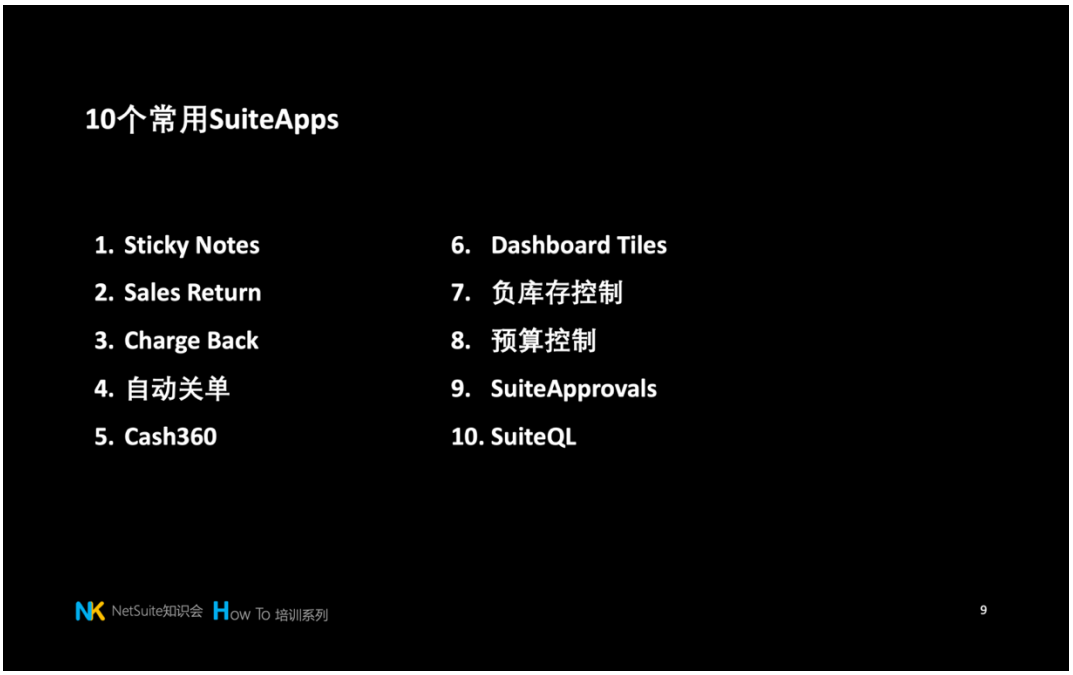
不同的第三方厂商所做的 App，在经过审核后才能被我们看到并使用，具体路径从菜单栏中进入“SuiteApps”，现在被称为应用市场 SuiteApps Marketplace。

图上展示的就是以 NetSuite 为核心的 Apps 生态圈，更侧重于接口部分，其本质上是相互之间的资源合作。



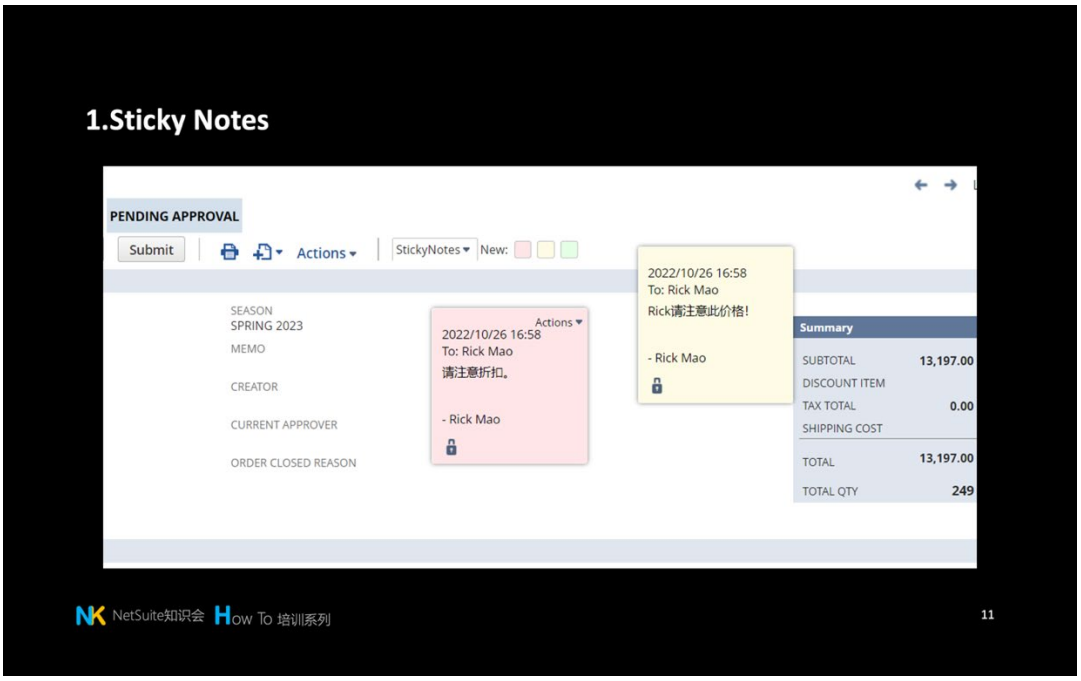
关于 Free Lance 的一些产品，可以在 Bundle 中直接进行搜索，比如虽然系统中没有 Inventory Aging 的报告，但是在 Bundle 中可以搜到相关 SuiteApps，就不需要顾问们重新再去从头做，可以根据自己的需求选择后进行加工使用。

以上三种就是 SuiteApps 的主要来源，帮助中心的内容可多了解。



以上是我个人认为比较有价值的十个 SuiteApps，有一些可能在功能上并不完美，也会有一些限制，但是对我们的使用带来了影响和意义，所以我们一个一个来谈。

关于 **Sticky Notes**，中文是“便利贴”的意思，比较典型的应用场景是在跨部门沟通时的应用，可以看到双方的留言过程及内容，在沟通方式上做出了改善。但是需要注意的是，它只针对 Transaction 进行应用，比如销售订单，采购订单等等，而不能够在报表上进行“留言”。

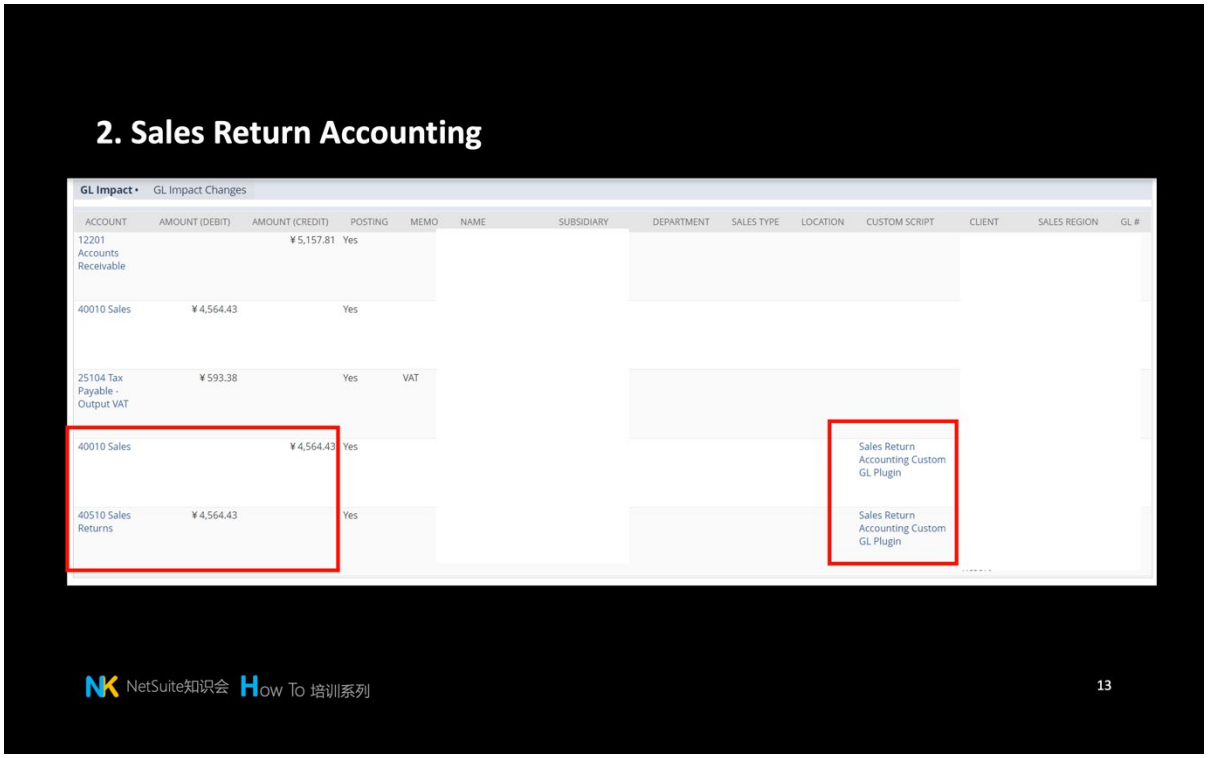


我们可以将描述或者备注信息发给对方，也能以邮件形式发出，对方也可以回复。

在合适的场景中，的确可以发挥其沟通的价值，对于我们的启发之一也是这个功能是 NetSuite 另一种 UI 的形式体现，非常美观。

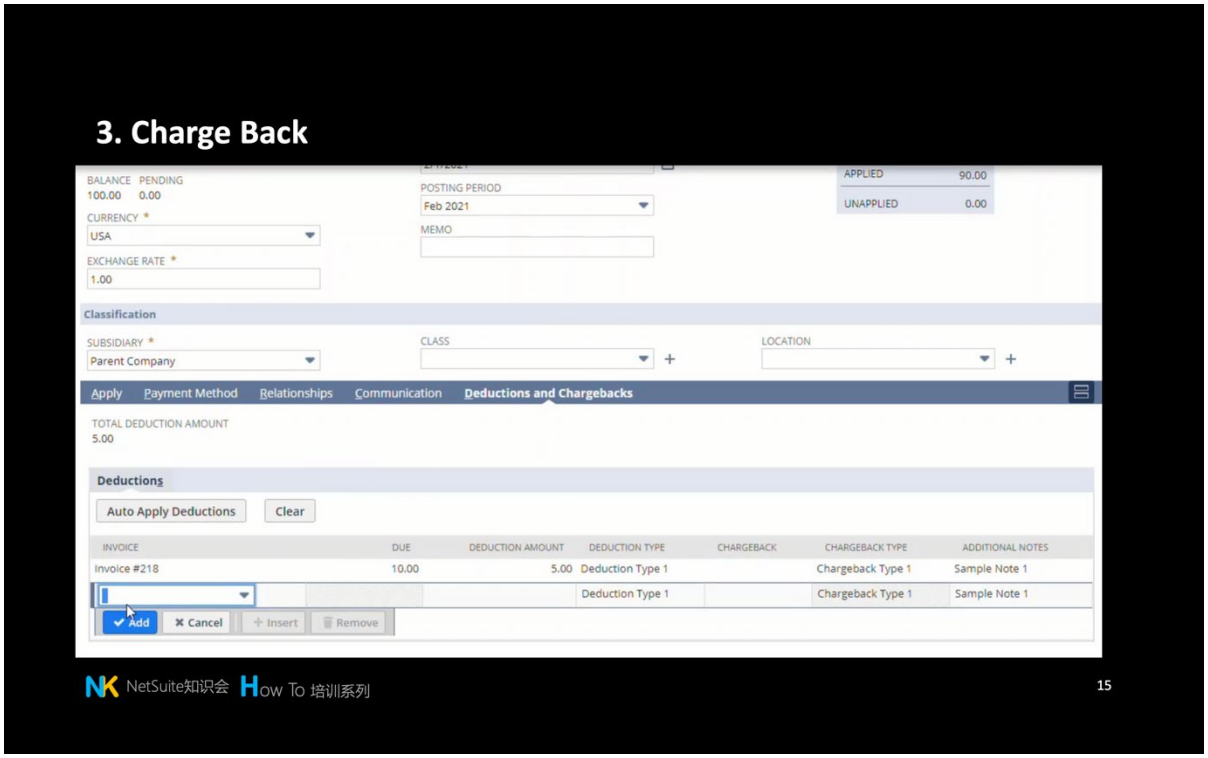
第二个是 **Sales Return Accounting**，基于系统的逻辑，当做 Credit Memo 时，系统的总账处理是冲减销售收入，但很多企业在记录销售收入时需要第二个科目来记，并不是以冲减销售收入的方式。所以，财务人员需要通过销售收入的借贷来判断是收入还是退货，假如财务人员不愿意以这样的方式来做，你的解决方案是什么？

如果是在不知道该 Bundle 的情况下，就需要通过 JE 来进行调整。



但是现在的这个 Bundle 可以弥补这一点，它通过 GL Plugin 干预过账的科目走向，把原来借方的 Sales 作为贷方，原来的冲减值转入 Sales Return 科目。

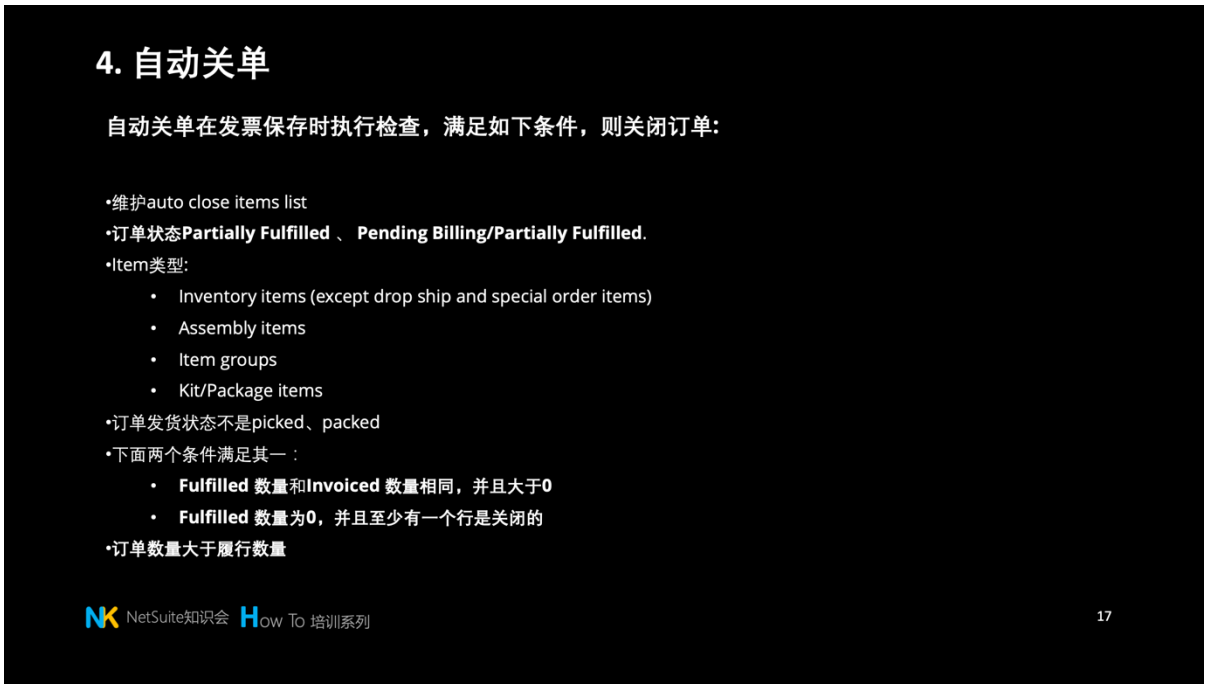
关于**Charge Back**，在接受客户付款时我们收到的钱不够，但也不是销售折扣，这个时候是什么？这里有三种情形：Deduction 冲减，Charge Back 待定冲减，Write-offs 小金额的销账。



可以看到，安装后在收款时增加了一个页签进行维护，这是一个实际的功能。

这个功能无需 JE 介入，由往来人员直接负责，这也意味着 NetSuite 中 AR 功能的丰富化。

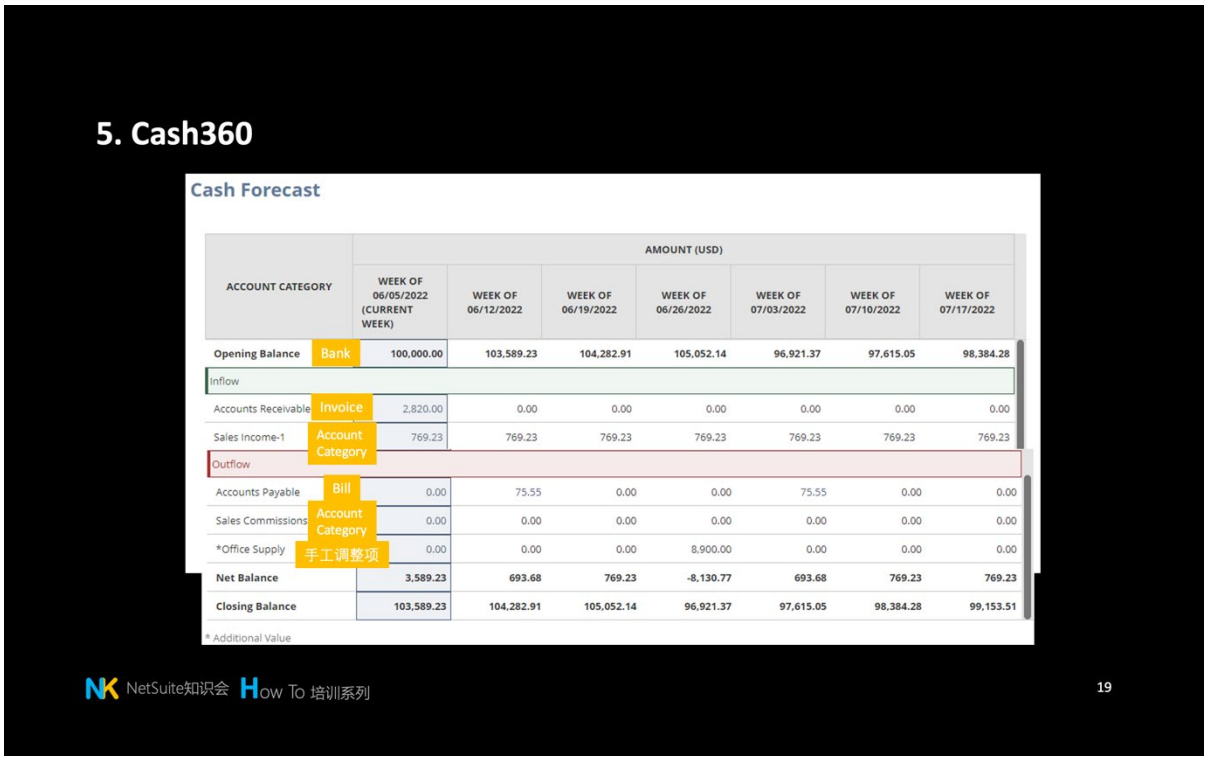
关于**自动关单**，我们知道在系统中销售订单无法自动关闭，只有通过手动或者通过脚本控制，但是如果不关闭在做报告时会受到影响。



为了弥补这一点所以推出了这个 Bundle，在发票保存时执行检查，当满足一定条件时关闭订单。

但是有一点限制是说，假如发货数量多，此时此刻要关的话就无法满足，因为该功能针对的是“未执行完的订单”（Back Orders），而对于已经执行完毕的销售订单仍旧无法关闭。

Cash360，不再是 Bundle，需要在应用市场中找到安装。



整个功能可以帮助我们展望未来的现金情况，数据来源包括 Bank，Invoice，Bill，专门设定某一些科目的发生额以及手工的调整项，基本上是满足我们的需求的。

但是目前有两个缺陷，第一点是 OneWorld 账户不能多个公司合并看，所以从集团角度看数字不可行；第二点是我们期望直接看到的是跟预算相关的数据，但是还需要再设置科目来抓取数据。

下一个是**磁贴**，主要用于导航，用起来非常方便，大家可以尝试，之前第五讲 Dashboard 中 Lisa 也分享了相关内容，感兴趣的话可以找来看看。

6. Dashboard Tiles

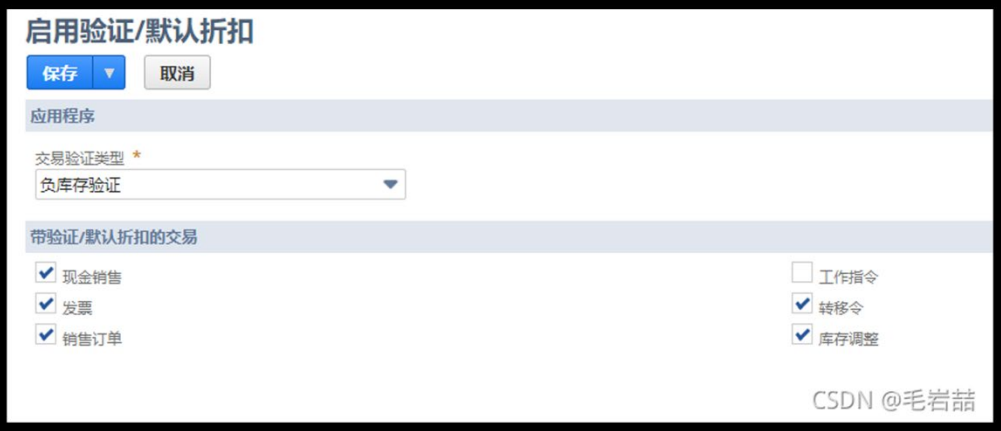
NetSuite知识会 How To 培训系列 21

7.负库存控制

发布者	NetSuite
Bundle名称	Enhanced Validations and Defaulting
Bundle编号	213294
是否收费	否
功能描述	此App的功能之一是负库存控制，尽管不完备。只能控制若干场景。

NetSuite知识会 How To 培训系列 22

7.负库存控制



NetSuite知识会 How To 培训系列 23

关于**负库存控制**，它的控制点并不完备。目前只能控制 6 种场景，Standalone 现金销售，Standalone 发票，销售订单，这些都是在建立的时间点进行控制。但其实有很多的订单在建立时候并没有货品，工单同样，所以检查的时间点会有一些奇怪。目前 TO 上可以在转移时检查发出仓数量是否足够，库存调整也是没有问题的。

但是，缺少了 IT 库存转移的场景，所以在项目上还需要代码控制来进行补充。

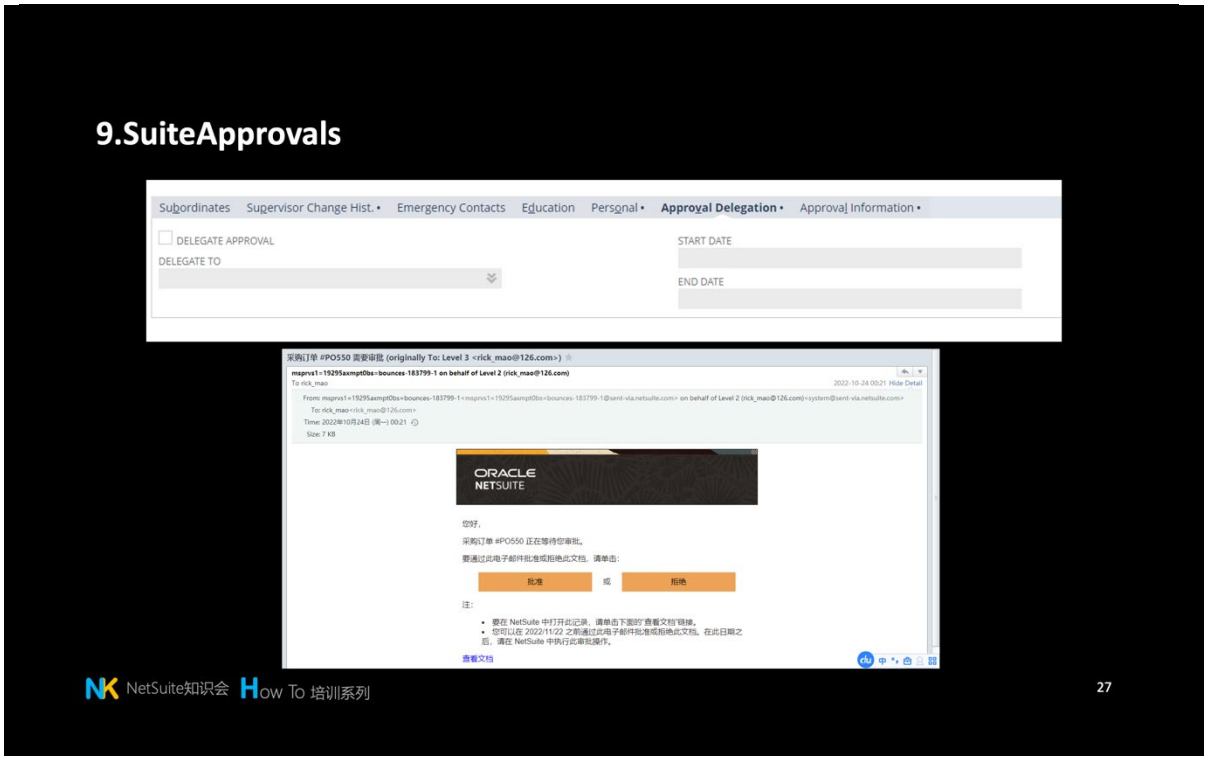
8.预算控制

发布者	NetSuite
Bundle名称	Expense Commitments and Budget Validation
Bundle编号	SuiteApp Marketplace
是否收费	否
功能描述	能够为特定的帐户、核算维度和期间组合创建预算。在PO, Purchase Request和Vendor Bill环节进行控制。

NetSuite知识会 How To 培训系列 24

预算控制，来自应用市场，但是它的先决条件是要有高级财务模块，其逻辑是针对科目设置预算，在 PO，PR 和 Vendor Bill 的环节进行检查，控制。

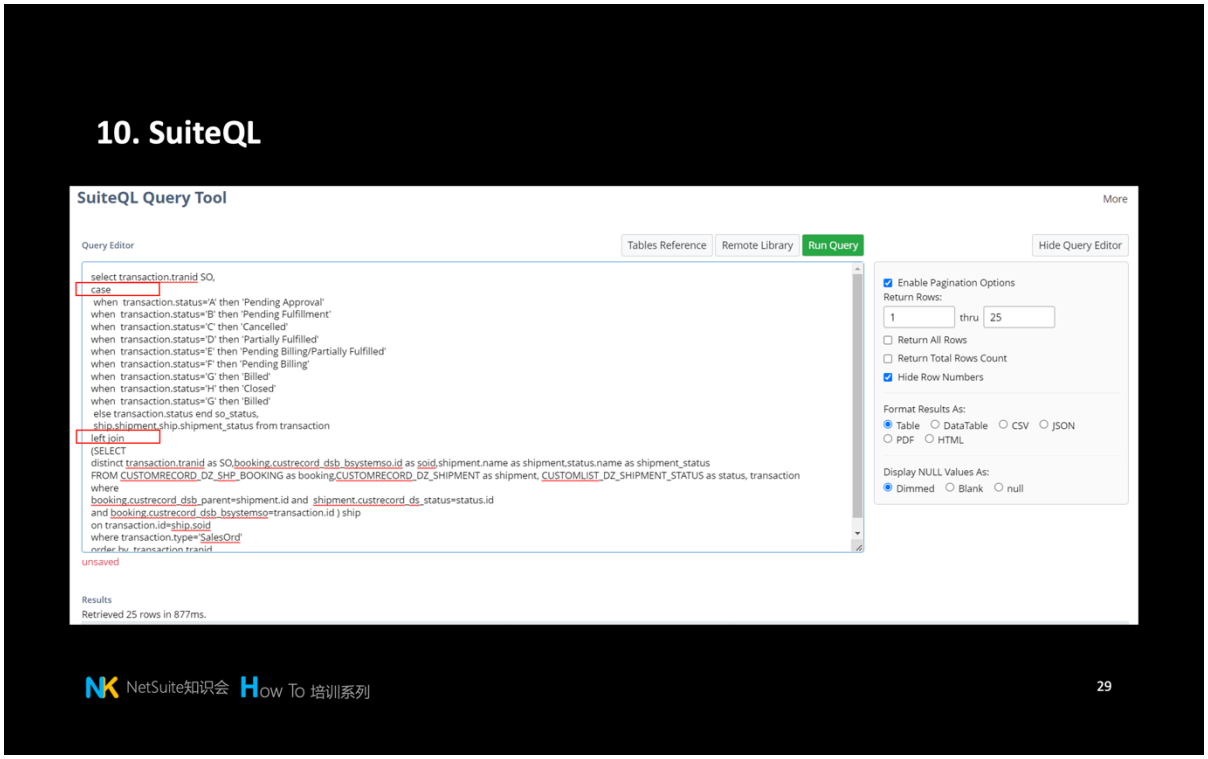
下一个是 **SuiteApprovals**，我认为它是一个非常有借鉴意义的工作流包，它针对 JE、工程变更、采购订单、Vendor Bill 做了封装包。



它对于我们有两点启发：

- 通过邮件审批，发出去的内容对方可以在邮件中直接回复审批“批准”或“拒绝”，然后再返回 NetSuite 直接进行同步，这个对于我们自己做工作流时可以借鉴；
- 审批代理，将审批权限临时给到他人，并且你可以设定开始和结束时间，比如休假的场景可以适用。

最后一个是我个人最喜欢的功能——**SuiteQL**，是 Free Lance 的作品，它可以支持 SQL 语句来帮助我们进行数据查询。



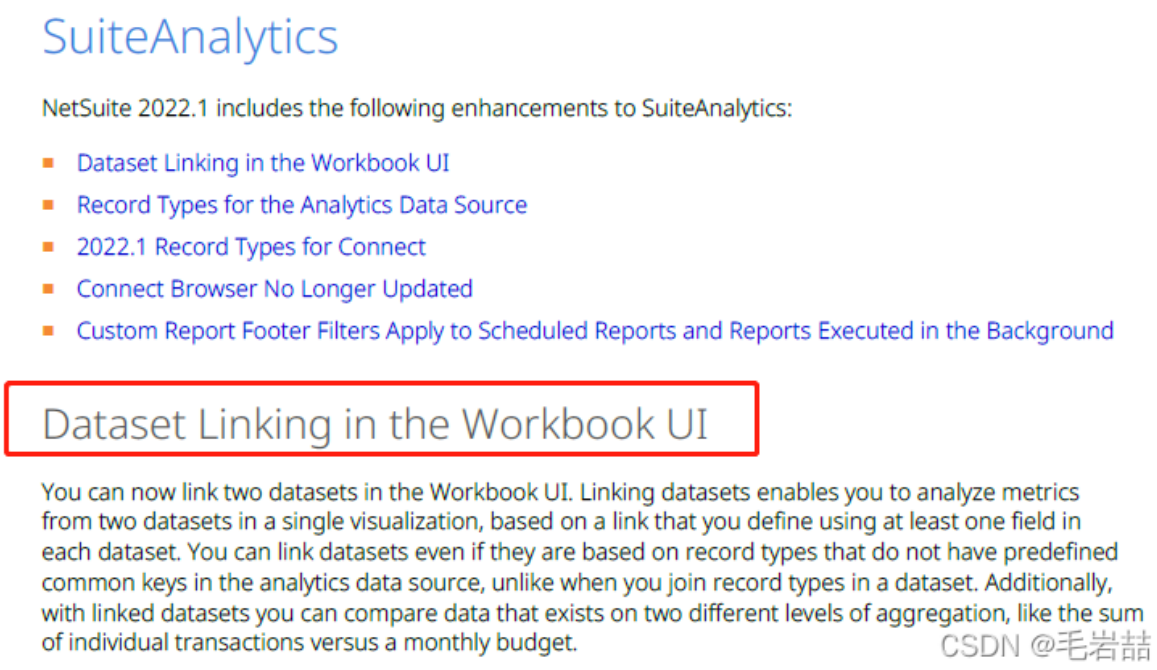
以上就是我们和大家分享的关于 SuiteApps 的“隐秘的功能宝藏”，其中的具体使用还需要大家后续进行探索。

NetSuite 知识会第 6 讲链接：SuiteApps——隐秘的功能宝藏

数据集连接-2022.1 新功能

作者：Rick 2022.02.06

2022.1 发版在即，今天先在预览环境测试了一下“数据集连接（Dataset Linking）”的功能。



首先来说下这个功能是咋回事。在之前的作文中，我们提到过 SuiteAnalytics Workbook 将作为 New King 扮演越来越重要的角色，特别是 Saved Search 无法完成的功能。例如，跨表查询。跨表查询（Join）的功能在 22.1 之前已经能够通过 SuiteScript 实现，但是对于没有编程技能的用户来说则没有任何意义。在 22.1，可以通过 UI 实现跨表查询了。

今朝，我们就做个讲解。

首先，说一下演示场景。我们想做一个科目预算与实际发生额的对比表，类似于标准报表中的“预算报表”。这在 Saved Search 中是无法实现的，因为需要以“科目”为键值，这在 Saved Search 中由于系统限制是无法做到的，但在 22.1 的版本中可以这样做：

1、基于“预算”、“科目发生额”分别建立数据集。

数据集的建立跟过去没啥不同，“预算数据集”选择需要显示的“科目、会计期、预算金额”等信息，在本例中名字为“预算 ByR”。“科目发生额数据集”选择需要显示的“科目、会计期间、发生额”等信息，在本例中名字为“实际 ByR”。

2、基于“预算数据集”建立“工作簿”，进行“连接数据集”。

3、选择“科目发生额”数据集。

在本例中选择“实际 ByR”。

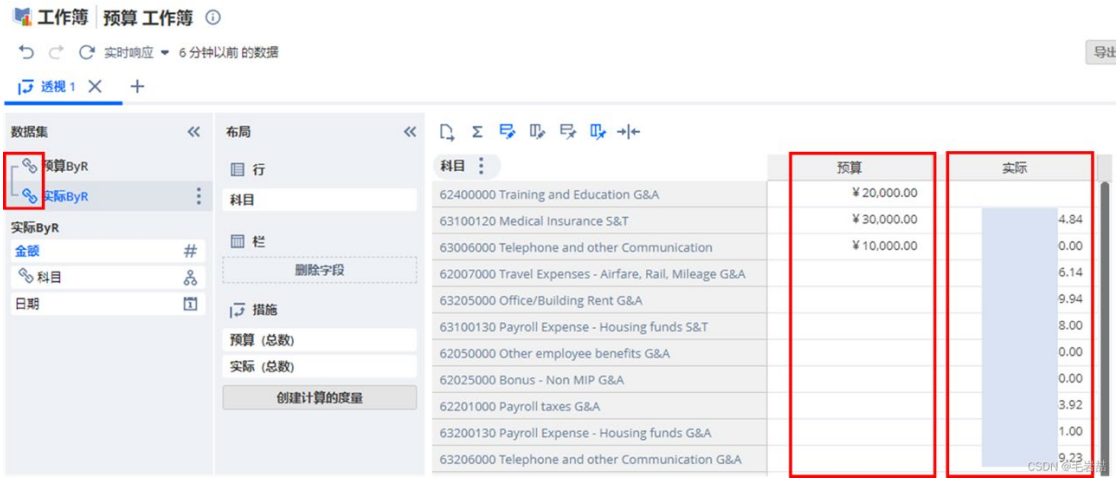
4、建立“数据集连接（Dataset Linking）”。

点击其中一个数据集的右边的菜单入口，选择建立数据集链接。

针对两个数据集选择其公用键（Common Key）建立链接。

5、针对链接的数据集建立透视表（Pivot），实现预实对比。

如下图所示，两个链接的数据集已经有了连接标志，跟过去建立透视表一样，选择相应的“行、数值”进行结果展示。



以上就是新功能的主要用法，详细的帮助还没有官方发布，但是可以通过预览视频了解大概，链接如下：

SuiteCloud: SuiteAnalytics Workbooks (custhelp.com)

目前已知的限制包括：

1、数据集链接只能发生在两个数据集；

2、连接而成的数据集不能作为新的数据集进行保存；

3、只有透视表、图表可以用来展现链接的数据集，表格是不行的；

4、链接类型只能是 Full Outer Join（这条是我们的推测，因为目前无法实现 Left、Right Join）。

以上就是此功能的预览小结！Enjoy your time！

阅读原文：数据集连接-2022.1 新功能

翻译机制

作者：Rick 2021.10.17

这个题目来自上周我们在做一个 Workshop 时，客户的反馈：“NetSuite 的翻译实在太那啥了！哈哈哈”。比如下面是销售发货界面，这个按钮在英文状态下是“Bill”，倒是还说得过去。但是在中文界面上居然出现了“采购收票”的按钮。这确实有点莫名其妙。因此我们觉得关于这个“哈哈哈”的翻译问题，以及 NetSuite 的翻译机制，有必要做个说明。

NetSuite 的翻译机制有如下几个事实，大家需要知道：

- **系统内建翻译无法由用户改变。**

除了“重命名记录”对事务处理和实体对象进行翻译、“批量更新转换”对货品主数据进行翻译外，没有其他的途径可以针对系统内建字串进行翻译。如果需要翻译，需要通过 NetSuite Support 部门传递到产品部门进行更新，更新频度为半年为单位的“版本”。

- **自定义对象都可以进行翻译。**

自定义对象，包括记录类型，列表等，都可以进行多语言的翻译维护。

- **Translation Management 自从 19.1 引入 Beta 版本，目前为 GA。**

其目的不是改变现有字符串，而是为了在代码中，维护多重翻译。例如，在不同 Locale 下的警告类信息。

在下图中可以看到，在 N/translation Module 下，可以按照 Locale 对 translation collection 进行选择性显示。

```
4  * @NModuleScope SameAccount
5  */
6  define(['N/translation'],
7
8  function(translation) {
9      function render(params) {
10         var portlet = params.portlet;
11
12         var locales = [translation.Locale.COMPANY_DEFAULT, translation.Locale.cs_CZ, translation.Locale.sk_SK];
13
14         var handle = translation.load({
15             collections: [
16                 {collection: 'custcollection_demo', alias: 'demo', keys: ['HEADING', 'WELCOME_MESSAGE']},
17                 {collection: 'custcollection_taglines', alias: 'taglines', keys: ['MAKE_FEEL_LOCAL']}
18             ],
19             locales: locales
20         });
21
22         portlet.title = handle.demo.HEADING();
23
24         locales.forEach(function (locale) {
25             var localeSpecific = translation.selectLocale({handle: handle, locale: locale});
26
27             portlet.addLine({
28                 text: locale + ':'
29             });
30
31             var conferenceName = 'SuiteWorld 2019';
32             portlet.addLine({
33                 text: localeSpecific.demo.WELCOME_MESSAGE({
34                     params: [conferenceName]
35                 }),
36                 url: 'http://www.netsuitesuiteworld.com/'
37             });
38
39             portlet.addLine({
40                 text: localeSecific.taglines.MAKE_FEEL_LOCAL()
41             });
```

- **翻译质量正在提升。**

在未来的版本中，将会对翻译质量进行改进。详细情况参考如下：

在 NetSuite 2021.2 和 2022.1 中，将逐步对大部分语言的翻译质量进行重大改进。此改进工作与集成到 Oracle 翻译流程密切相关，它将通过改进术语检查的质量控制和功能来显著改进翻译质量。 将在 NetSuite 2021.2 中发布术语或样式的新翻译，并应用于 NetSuite 2021.2 中新发布或更改的所有功能。我们将在 NetSuite 2022.1 中继续逐步改进所有其他功能（尤其是 UI 字符串）的术语翻译，使其符合新的翻译标准。请注意，因为 NetSuite 2022.1 中添加的新字符串都将遵循新翻译，所以在 NetSuite 2021.2 和 NetSuite 2022.1 之间，同一术语的翻译可 能会存在差异，即新翻译和现有翻译之间可能会存在差异。在短期内，这些差异可能会带来不便；但是从长期来看，这是显著提高翻 译质量的重要一步。 因为术语翻译的改进工作是逐步进行的，所以在该改进工作完成之前，您可能会遇到翻译不一致的情形。

可登录 NetSuite 查看如下链接：

Major Translation Quality Improvement Project 2021.2 and 2022.1

- **界面的翻译问题是可以靠我们自己解决的。**

虽然 NetSuite 将翻译质量问题作为未来的改进目标，但是对用户来说与其等改变，不如靠自己。我们可以通过定制 Form 处理，来解决本文开头的“哈哈哈”问题。

处理方式如下：

建立自定义 Form，在“操作”中找到出问题的按钮，将其字串进行改正。

可以看到字串已从“采购收票”修改为“销售开票”，问题解决！

阅读原文：翻译机制

2. 系统管理

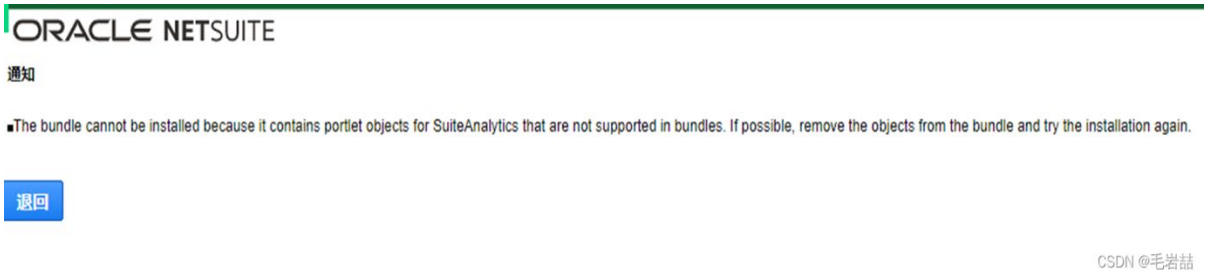
SuiteBundle 禁忌

作者：Rick 2021.12.05

SuiteBundle 作为环境迁移工具，目前已经停止更新了，这将引起一些不良后果，本周我们就遇到了一例，问题呈现给大家。

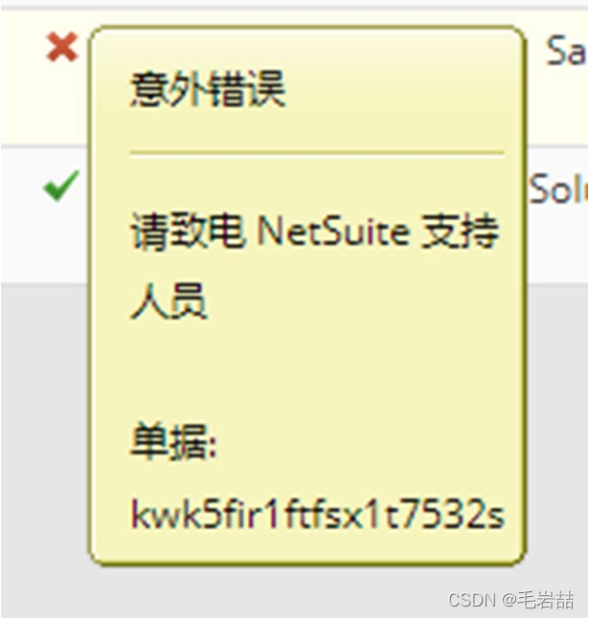
问题一：SuiteAnalytics Portlet Object 不能被 Bundle 迁移。

在 Bundle 选项中选择“Bundle All”进行打包，一切正常。但是在目标环境进行安装时，会出现“bundle cannot be installed”的错误提示，安装失败。



问题二，安装结束后，提示“意外错误”。

在剔除了疑似 SuiteAnalytics Portlet 错误源后，继续安装。但是安装结束后，出现如下意外错误。



这两个问题，百思不得其解，最后致电 NetSuite Support，得到解决。

解决方案：

由于 SuiteBundle 停更了，所以近期的一些新的功能，“禁止”通过 Bundle 进行迁移。目前有两个功能领域：一个是 Workbook，另一个是 Translation String。假如 Bundle 中包含上述两个定制内容，就会导致出错。

结合我们的案例，第一个错误提示的原因是，我们建立过一些 Dashboard，其中引用了 Workbook。第二个错误的原因是，在我们的环境中存在一些 Translation String，在 Bundle 时，因为选择的是 Bundle All，结果未将其在 Bundle 中删除掉，引发了安装时的异常保护，导致 Error。

那么，从这个案例我们得到的经验就是“混合迁移工具”，未来在 NetSuite 的环境迁移中，基本的形式就是：**SuiteBundle+Copy To Account。**利用 SuiteBundle 进行“传统、大量”定制内容的迁移，利用 Copy To Account 进行单点内容的迁移。

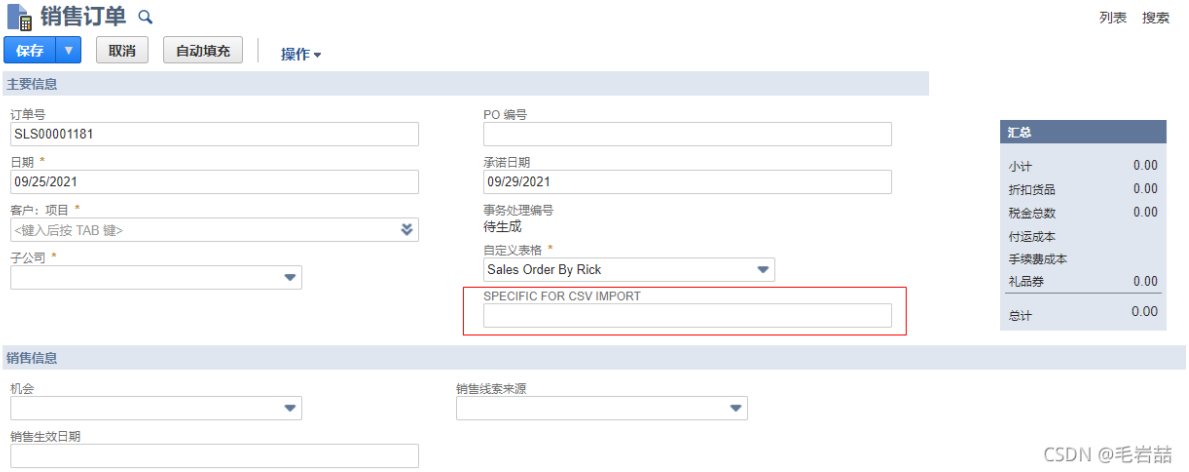
NetSuite 是个不断发展的系统，内容越来越多，功能越来越强大，坑也越来越多。我们把掉坑，爬坑当成了一种乐趣。掉坑懊丧，出坑豁然开朗，技术之路大致如此吧！

阅读原文：SuiteBundle 禁忌

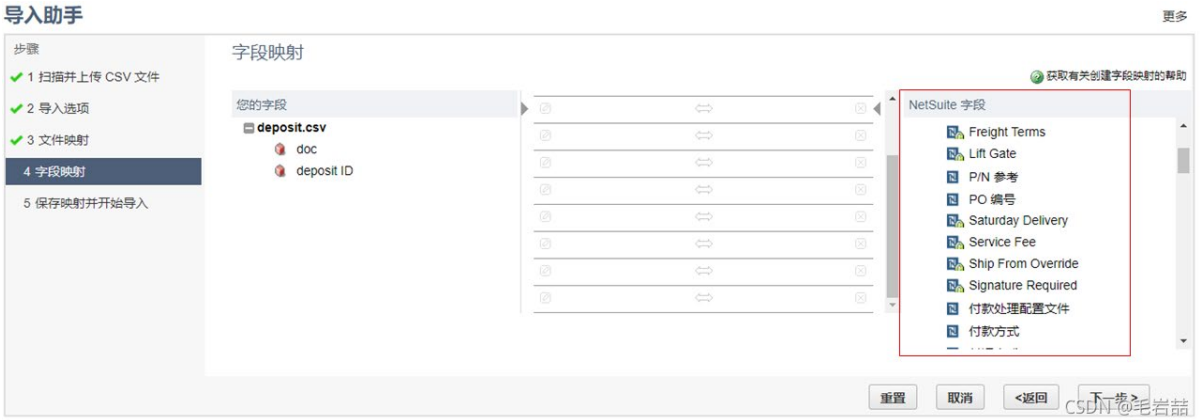
消失的字段

作者：Rick 2021.09.25

这周我们所遇到的问题是：在销售订单上增加了一个字段叫做“Special For CSV Import”，目的是想通过 CSV 导入的方式，给所有的销售订单赋值。



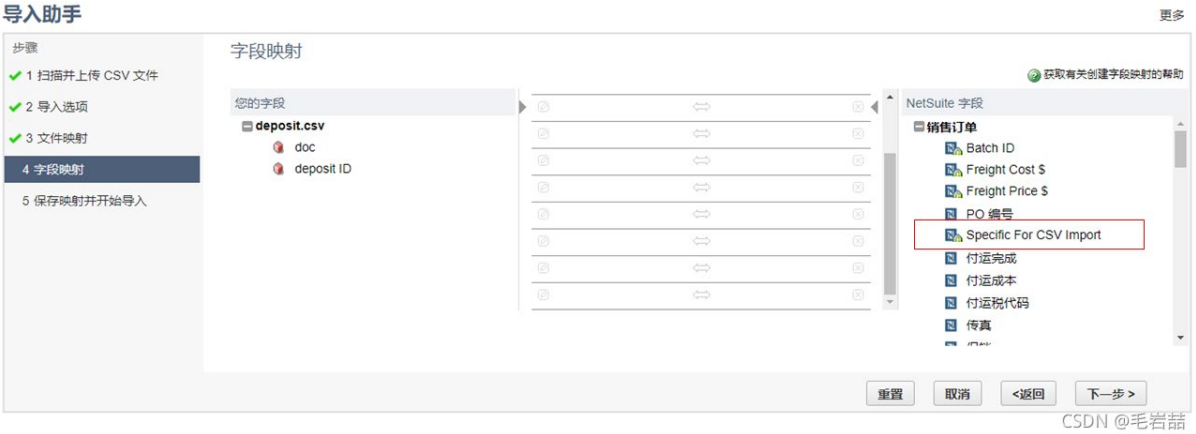
但是在建立字段映射关系时，怎么也找不到这个自定义字段。



分析之后，这个问题的解决方法比较简单。

在CSV导入向导中，“高级选项”下有一个参数，叫做“表单（Form）”。这里默认是“Primary Sales Order Form”，但是我们增加的那个自定义字段是在另外一个叫做“Sales Order By Rick”的Form中。由于那个自定义字段在“Primary Sales Order Form”中没有被激活，所以在做字段映射时就无法呈现了。

我们选择了正确的Form后，那个字段也就出现了。



CSDN @毛岩喆

通过解决这个问题，我们有个疑问，“为什么要这样设计？”经过长达300秒的思考后，有了感悟。

其实这样设计的主要意图还是“权限控制”。可以说CSV导入是另一种用户访问数据的途径，和在录入界面访问数据本质上是一样的。Form是NetSuite处理访问权限的手法之一，有其独特的使用场景。例如，系统原生字段是不受“Field Permission”控制的，譬如一些个人、财务敏感数据字段。用户不能针对这些字段进行权限控制。控制的主要方法就是用Form来做隔离，做个特别的Form把给看的字段显示出来。然后将角色与这个Form强制关联，从而保证用户看到的字段是“受限”的。因此，在CSV导入时，也是按照通过Form权限来限制字段显示的，没有选择到对的Form，字段当然就出不来了。

CSDN @毛岩喆

阅读原文：消失的字段

巧做打印模版

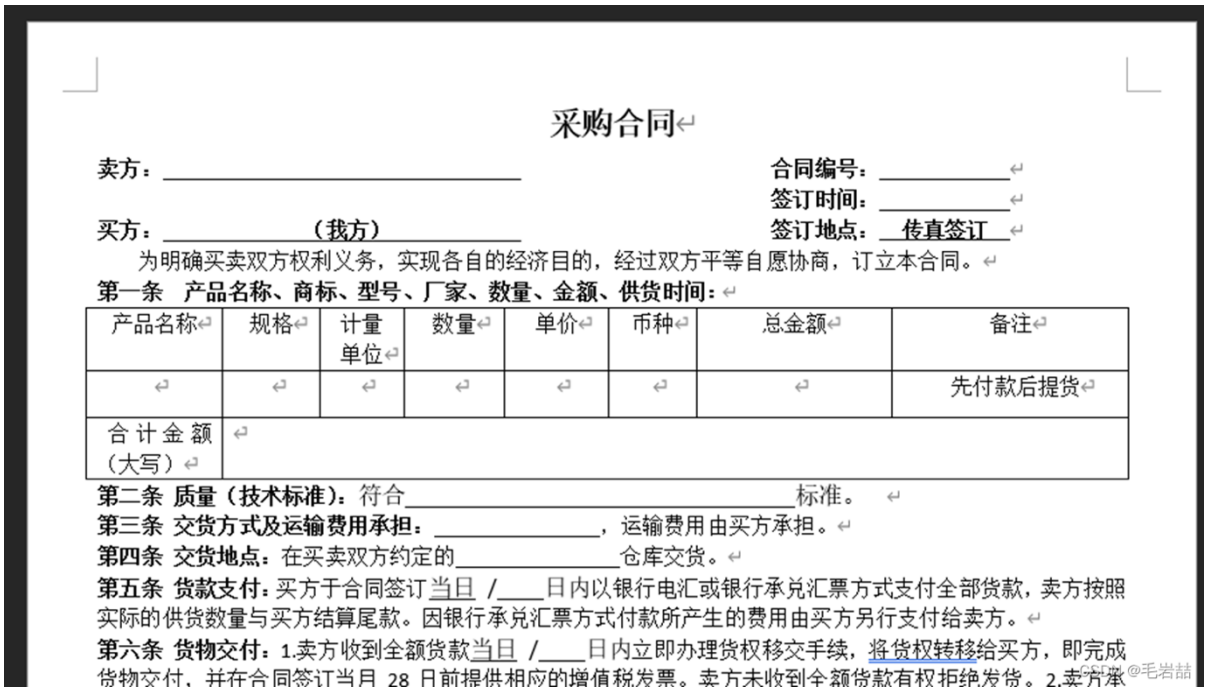
作者：Rick 2021.12.20

我们在项目上经常遇到打印模板移植的需求，也就是用户希望能够将目前正在使用的“购销合同”的打印模板能够在NetSuite上原样呈现。怎么处理这个需求，我们摸索出一个快速的处理方式，大家也可以一试！

基本原理是：将购销合同的 Word 文档通过在线工具转为 Html，然后直接复制 Html 内容到 NetSuite 打印模板中去进行二次加工。这样就能比较好地继承 Word 中的段落和表格形式，避免在 PDF 模板中从 0 写起。

Have a demo!

1. 准备好 Word 格式的购销合同样本。



采购合同

卖方：_____ 合同编号：_____

签订时间：_____

买方：_____（我方）_____ 签订地点：__传真签订__

为明确买卖双方权利义务，实现各自的经济目的，经过双方平等自愿协商，订立本合同。

第一条 产品名称、商标、型号、厂家、数量、金额、供货时间：

产品名称	规格	计量单位	数量	单价	币种	总金额	备注
							先付款后提货
合计金额（大写）							

第二条 质量（技术标准）：符合_____标准。

第三条 交货方式及运输费用承担：_____，运输费用由买方承担。

第四条 交货地点：在买卖双方约定的_____仓库交货。

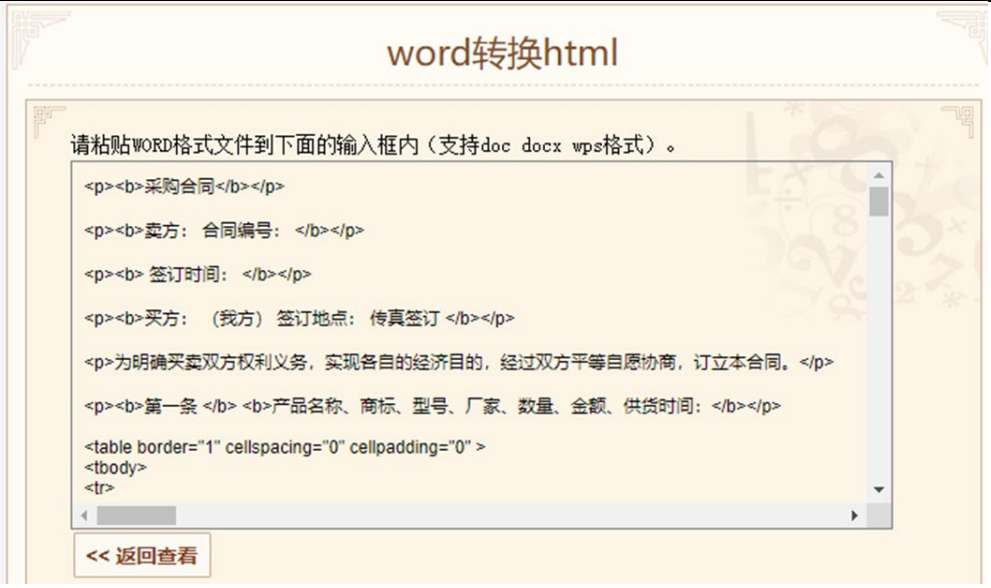
第五条 货款支付：买方于合同签订当日 /____日内以银行电汇或银行承兑汇票方式支付全部货款，卖方按照实际的供货数量与买方结算尾款。因银行承兑汇票方式付款所产生的费用由买方另行支付给卖方。

第六条 货物交付：1.卖方收到全额货款当日 /____日内立即办理货权移交手续，将货权转移给买方，即完成货物交付，并在合同签订当月 28 日前提供相应的增值税发票。卖方未收到全额货款有权拒绝发货。2.卖方承

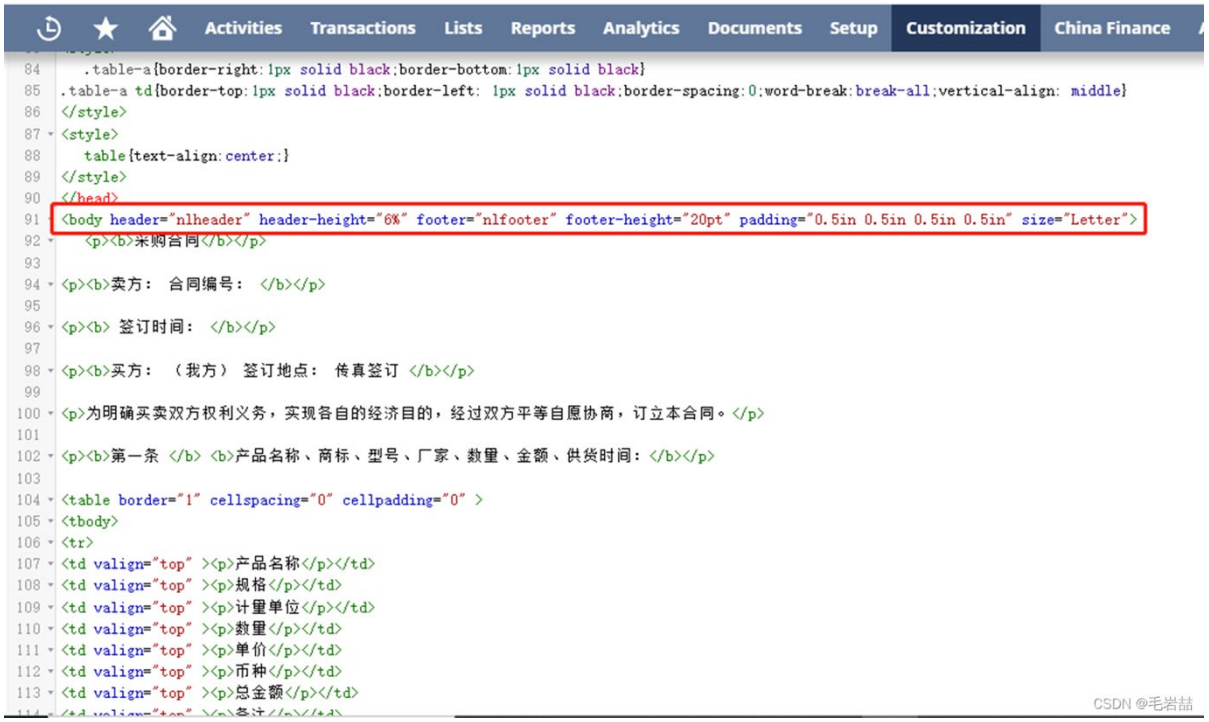
2.找到如下的 Word 转 Html 在线工具。

word 转换 html (99cankao.com)

将 Word 文档 Ctrl+V 到其中，转换之，得到 Html。



3.将得到的 Html Ctrl+V 到打印模板 PDF 的源码编辑中去。



记得放在<body></body>之间。关于 Html 的基本编辑技术，大家另行学习。

4.预览 PDF，调整格式。

上述的 Html 基本上会把 Word 的布局带过来，但是对于“居中”、“相对宽度”之类的 Style 是没有处理的，需要手工调整之。

上述就是我们在工作中的一点技巧，减少 Donkey Work，节省时间胜造七级浮屠！

阅读原文：巧做打印模版

3. 权限

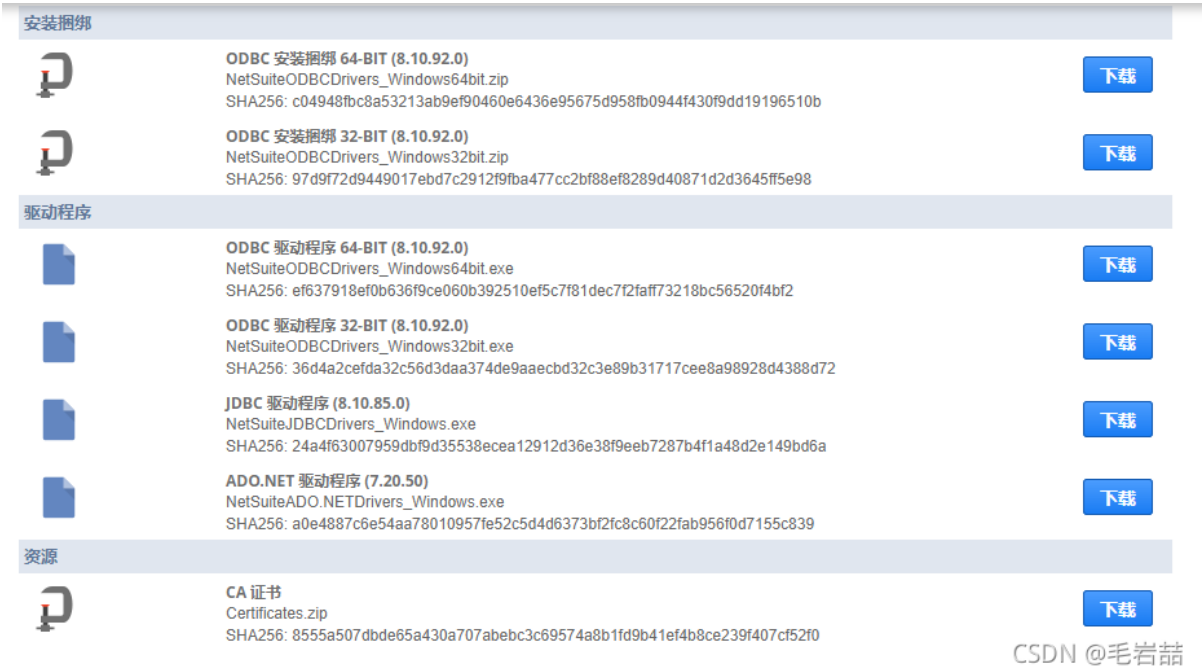
ODBC 2FA 角色问题

作者：Rick 2021.11.21

SuiteAnalytics Connect 是枚银弹，用于将数据从云端拉到本地做数仓分析的来源。这个功能一般用户用得比较少，基本上都是有数据仓库应用需求的用户才会选择，因为这是要花钱的 NetSuite 组件。

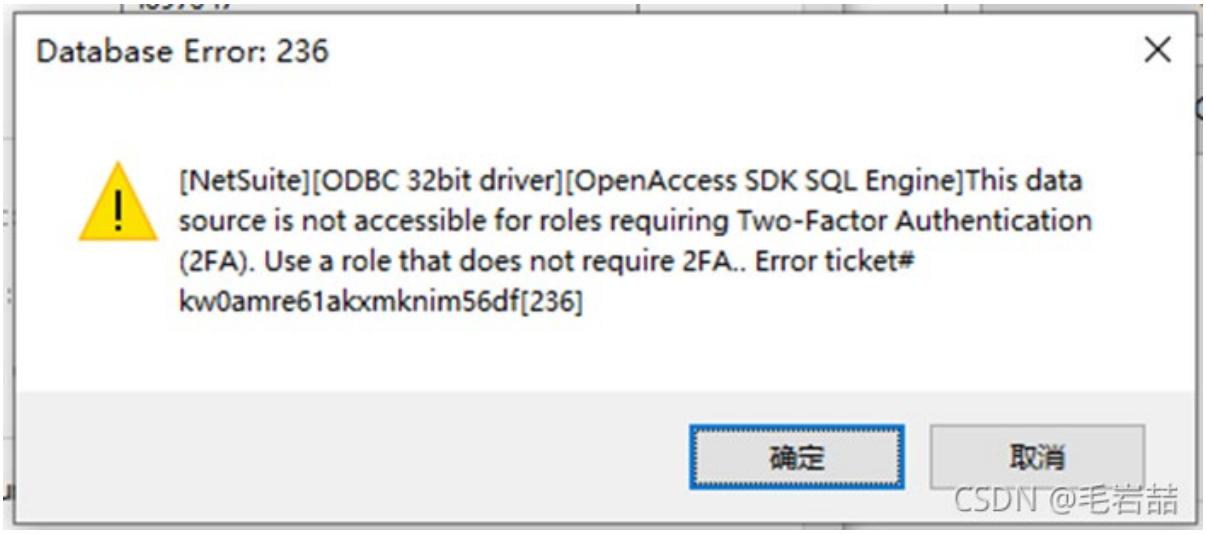
当购买了此模块后，登录后在“设置”中就多了个选项。

可以选择 ODBC、JDBC、ADO.Net 三种接口驱动程序。



这周，因为我需要将数据拉到本地的数据库做分析，所以我选择了 ODBC。在配置 ODBC 过程中碰到一个问题，于此分享之，避免大家入坑。

在 ODBC 配置过程中，出现了 **2FA** 的问题。



这个报错的意思是，我不能用带 2FA 的角色访问。OK！我就换个角色呗，所以就建立一个不带 2FA 的角色进行连接。

问题来了！我试了无数次，始终是无法解决此问题。即使我多次检查角色、密码、大小写。。。。。。

经过长达 3000 多秒的反复折腾，最后，发现了问题所在。NetSuite 有个 2FA 角色的总览工具，可以统一管理各个角色是否开启 2FA。

原来，即使你把角色设置为“不需要双因素验证”，也不耽误系统给你一个“强制 2FA”的约束。如果你不幸选择了一个“强制 2FA”的角色，另存为你的 ODBC 连接角色。那你就中奖了！

发现这个问题后，解决就简单了，做一个不强制 2FA 认证的角色就可以了。

这个“强制 2FA 认证”背后的**逻辑**是这样：NetSuite 认为某些权限（Permission）的危害性太大，所以当你的角色涉及到这些权限时，就被自动地打上了“强制 2FA”的约束。管理员就不用说了，肯定因为 2FA 限制，不能访问 ODBC。其他有“接口”类权限的角色也是被强制 2FA 的。

具体情况线上看帮助：
Permissions Requiring Two-Factor Authentication (2FA) (custhelp.com)

分享爬坑经历，节省时间胜造七级浮屠！

阅读原文：ODBC 2FA 角色问题

4. 开发

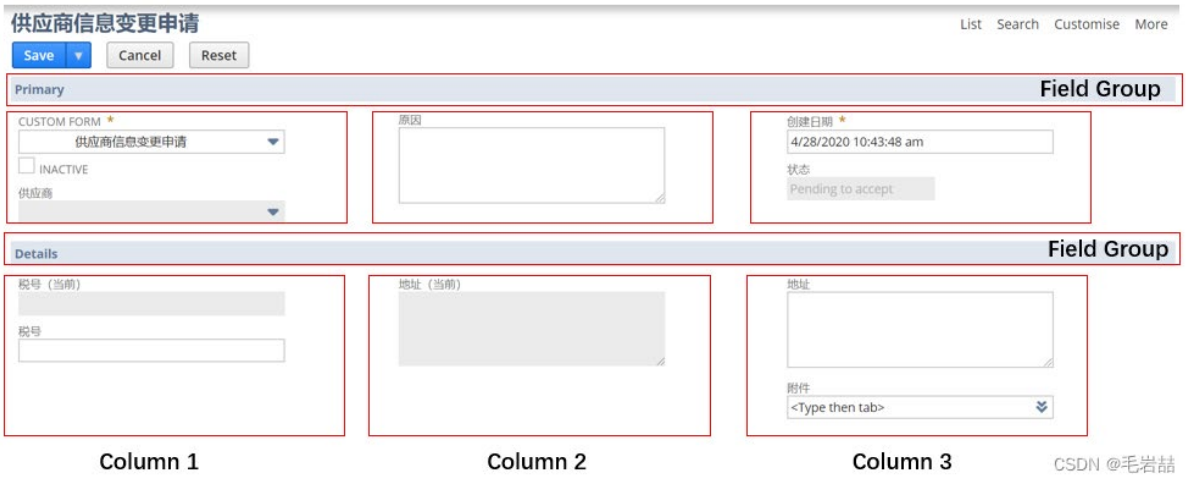
定制表格布局的要点

作者：Rick 2023.04.28

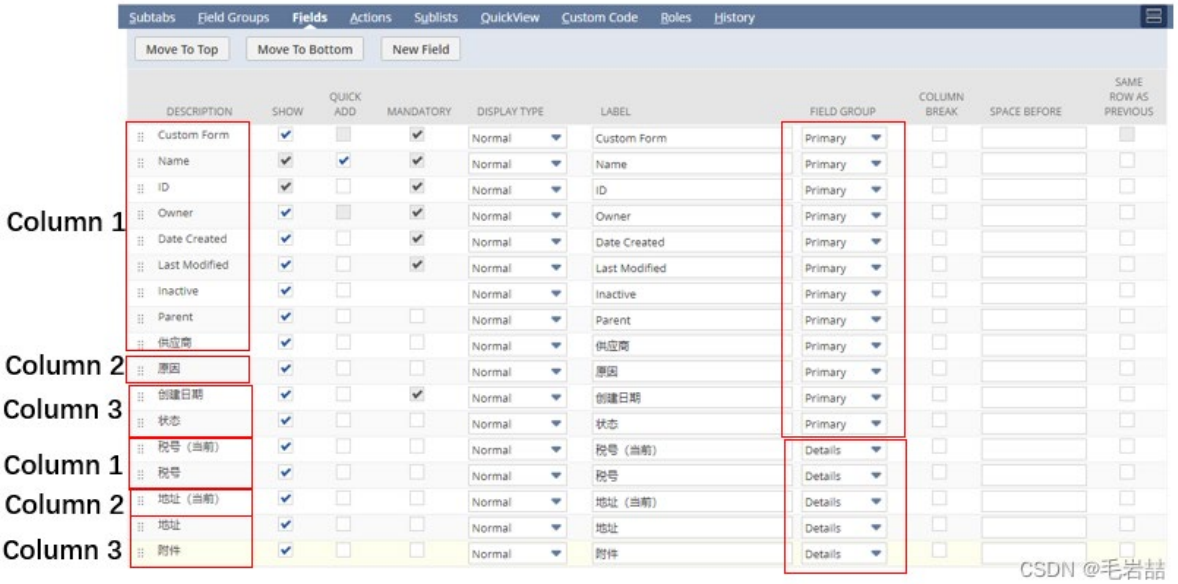
Form Layout （表格布局）是直接呈现给用户的程序界面，其美观与否跟顾问的态度有关，是追求细节的匠心体现。今朝分享一下常被忽略的几个 UI 参数，大家可以实践之。

首先，我们谈一下 NetSuite 布局的基本概念 - Field Group 和 Column。

NS 的字段布局原则，是按照 Field Group 来分区，然后依照 Column 从左到右，从上到下排列字段的。如下图所示，这个 Form 包含两个 Field Group，每个 Field Group 分为 3 个 Column。



一个 Field Group 中的多个 Fields，按顺序平均地分配到 Columns 中。



如果想在上面的样式上做些改变。在目前 UI 框架下，我们能做得不多，只能通过如下的 4 个参数进行调配：

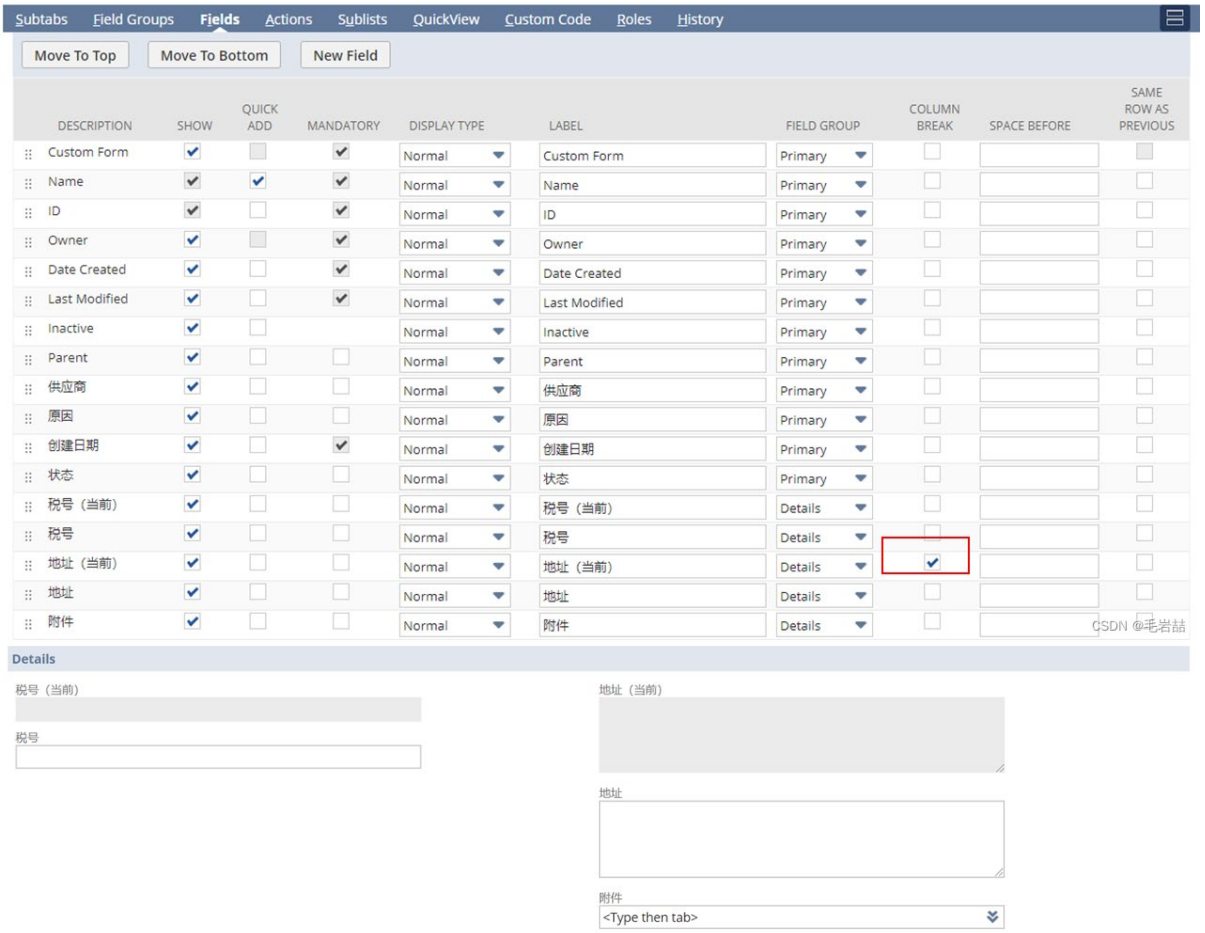
- Column Break
- Same Row as Previous
- Space Before
- Single Column

我们下面逐个用图例来演示这些参数的用途。

- **Column Break**

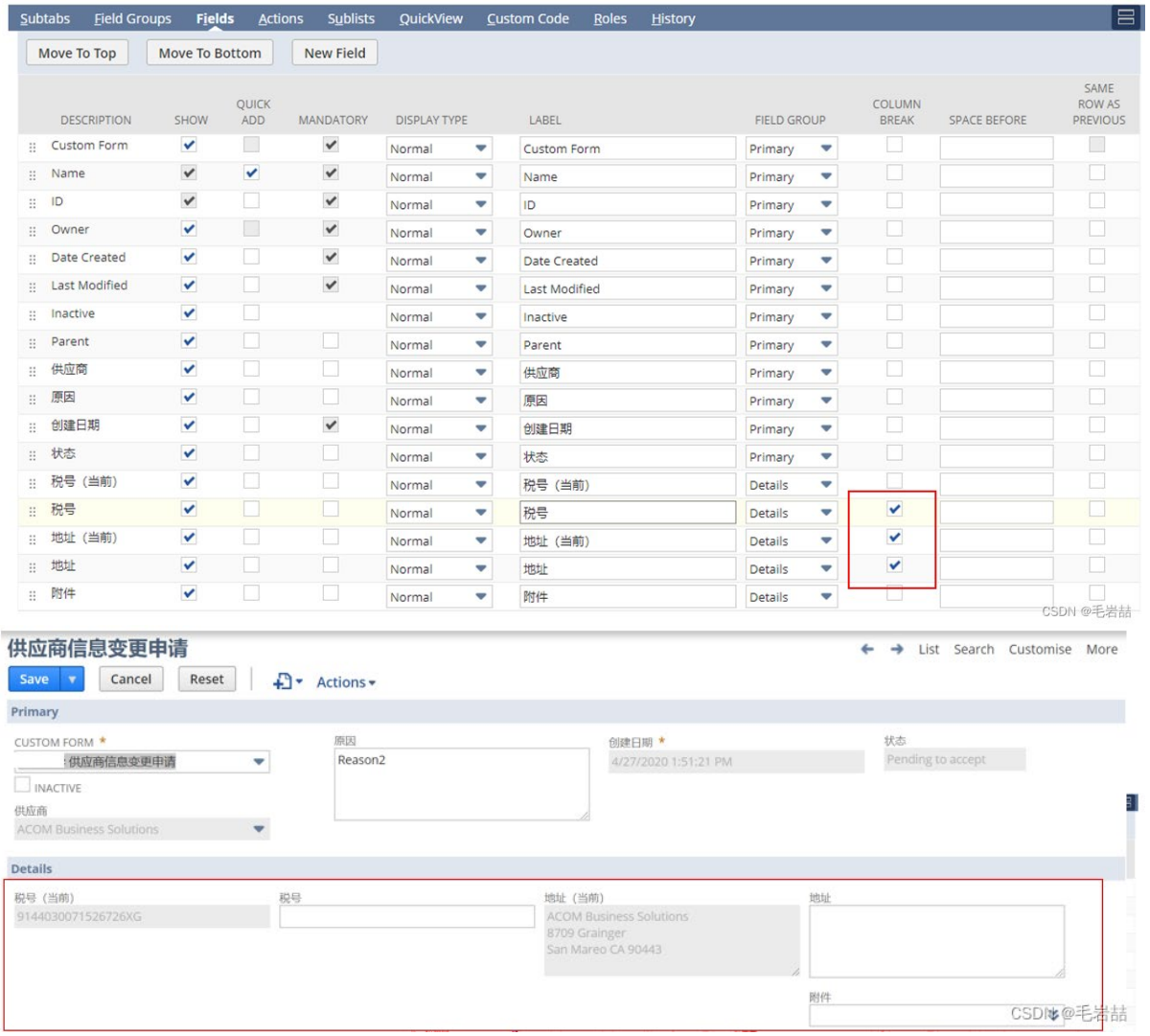
由于字段是按照从上到下顺序分配排列的，Column Break 的含义是结束系统默认的列，开启新的列。

以下图为例，地址（当前）字段如果勾选了 Column Break，则意味着从这个字段开始变为一个新 Column。



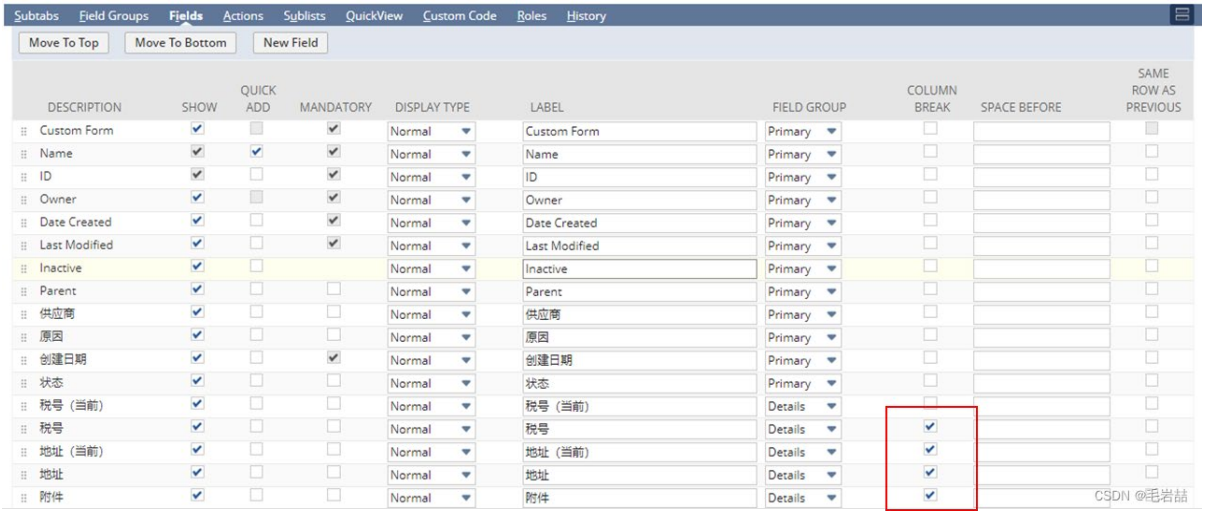
通过 Column Break，强制分列，突破原始的 3 列布局，从而建立多 Column 布局。

4 列样式：


Subtabs Field Groups Fields Actions Sublists QuickView Custom Code Roles History
Move To Top
Move To Bottom
New Field
DESCRIPTION SHOW QUICK ADD MANDATORY DISPLAY TYPE LABEL FIELD GROUP COLUMN BREAK SPACE BEFORE SAME ROW AS PREVIOUS
Custom Form Normal Custom Form Primary
Name Normal Name Primary
ID Normal ID Primary
Owner Normal Owner Primary
Date Created Normal Date Created Primary
Last Modified Normal Last Modified Primary
Inactive Normal Inactive Primary
Parent Normal Parent Primary
供应商 Normal 供应商 Primary
原因 Normal 原因 Primary
创建日期 Normal 创建日期 Primary
状态 Normal 状态 Primary
税号（当前） Normal 税号（当前） Details
税号 Normal 税号 Details
地址（当前） Normal 地址（当前） Details
地址 Normal 地址 Details
附件 Normal 附件 Details
CSDN @毛岩喆

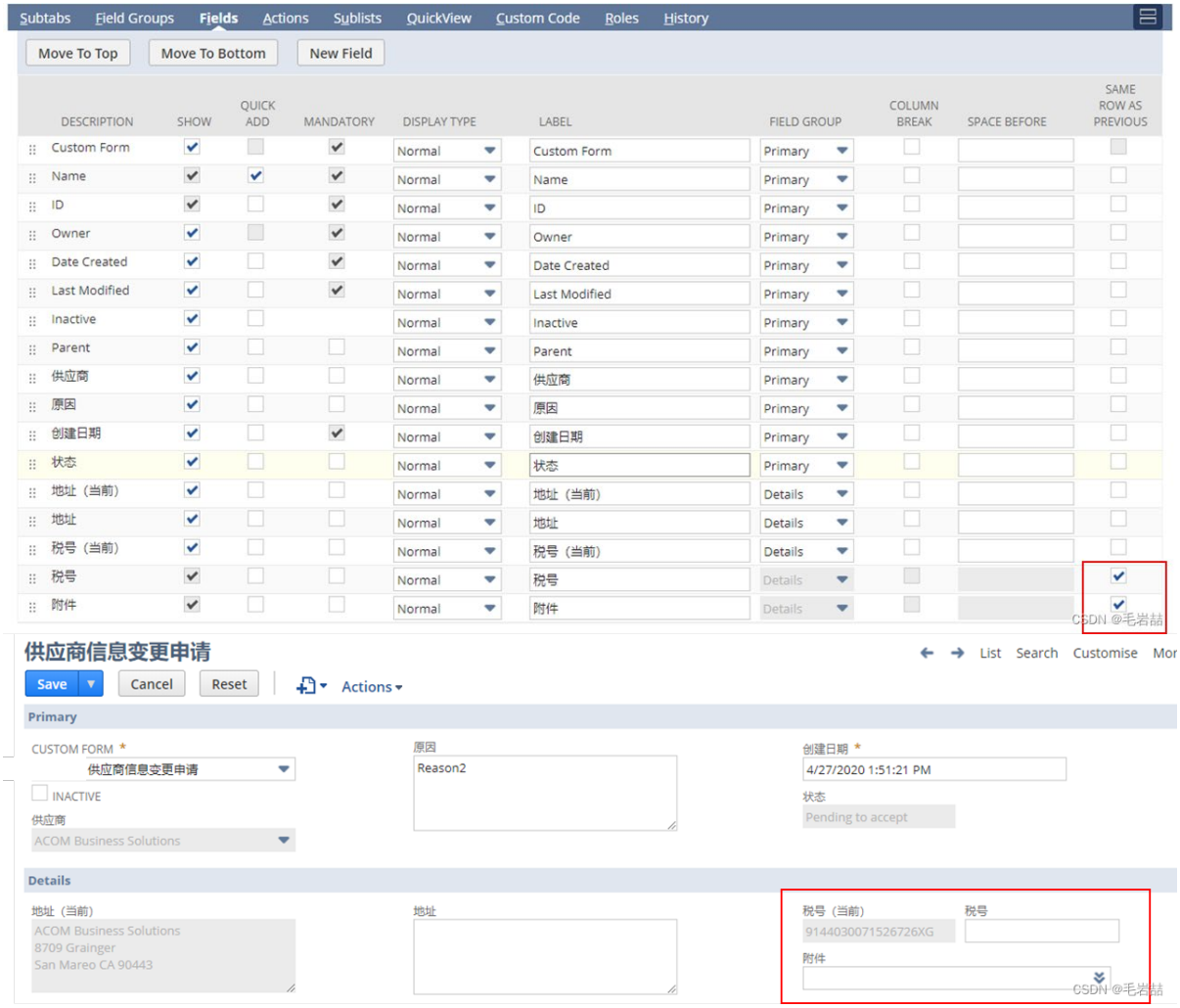
供应商信息变更申请
List Search Customise More
Save Cancel Reset Actions
Primary
CUSTOM FORM *
供应商信息变更申请
INACTIVE
供应商
ACOM Business Solutions
原因
Reason2
创建日期 *
4/27/2020 1:51:21 PM
状态
Pending to accept
Details
税号（当前）
9144030071526726XG
税号
地址（当前）
ACOM Business Solutions
8709 Grainger
San Mareo CA 90443
地址
附件
CSDN @毛岩喆

5 列样式：


Subtabs Field Groups Fields Actions Sublists QuickView Custom Code Roles History
Move To Top
Move To Bottom
New Field
DESCRIPTION SHOW QUICK ADD MANDATORY DISPLAY TYPE LABEL FIELD GROUP COLUMN BREAK SPACE BEFORE SAME ROW AS PREVIOUS
Custom Form Normal Custom Form Primary
Name Normal Name Primary
ID Normal ID Primary
Owner Normal Owner Primary
Date Created Normal Date Created Primary
Last Modified Normal Last Modified Primary
Inactive Normal Inactive Primary
Parent Normal Parent Primary
供应商 Normal 供应商 Primary
原因 Normal 原因 Primary
创建日期 Normal 创建日期 Primary
状态 Normal 状态 Primary
税号（当前） Normal 税号（当前） Details
税号 Normal 税号 Details
地址（当前） Normal 地址（当前） Details
地址 Normal 地址 Details
附件 Normal 附件 Details
CSDN @毛岩喆

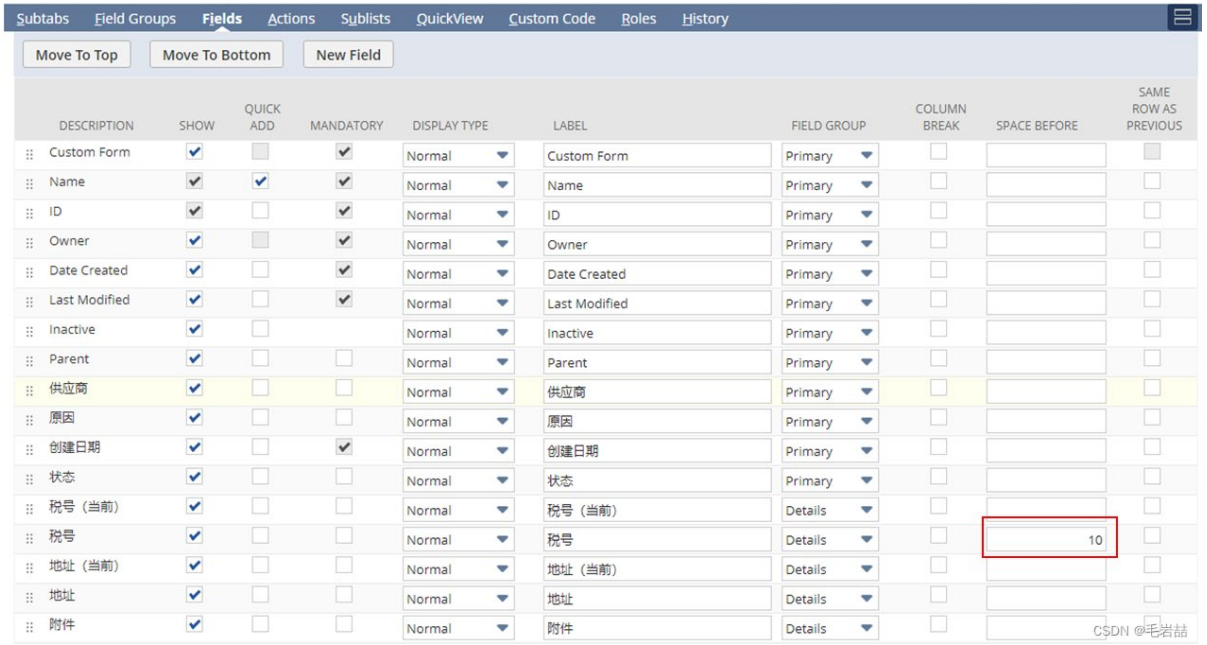
- **Same Row as Previous**

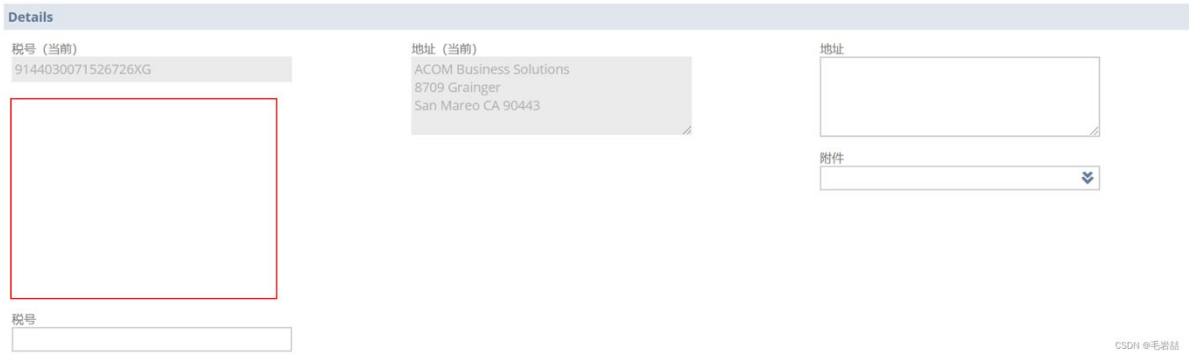
顾名思义，强制排在一行上，牺牲字段控件的宽度。



- **Space Before**

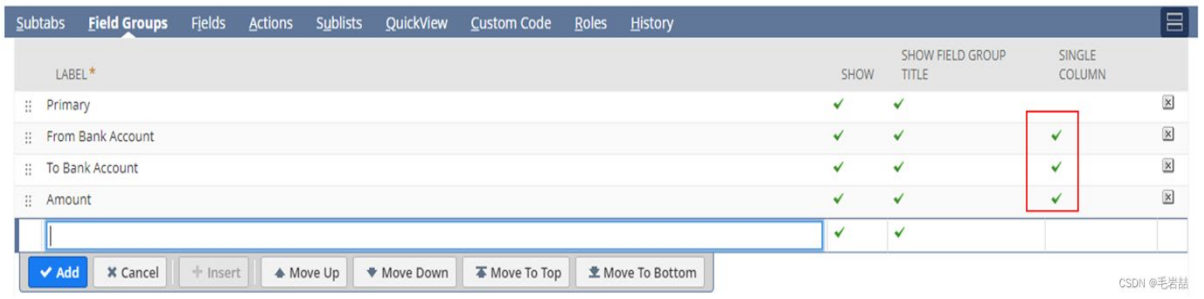
在同一 Column 中，两个 Field 之间的行距。以下图为例，两个字段间被隔开了 10 行。

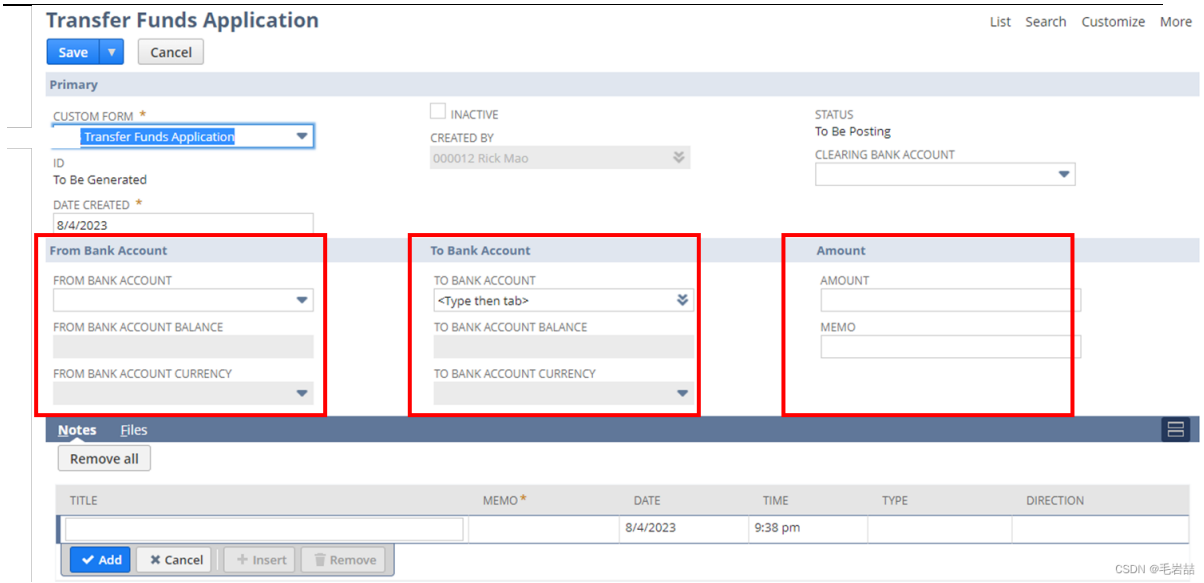




- **Single Column**

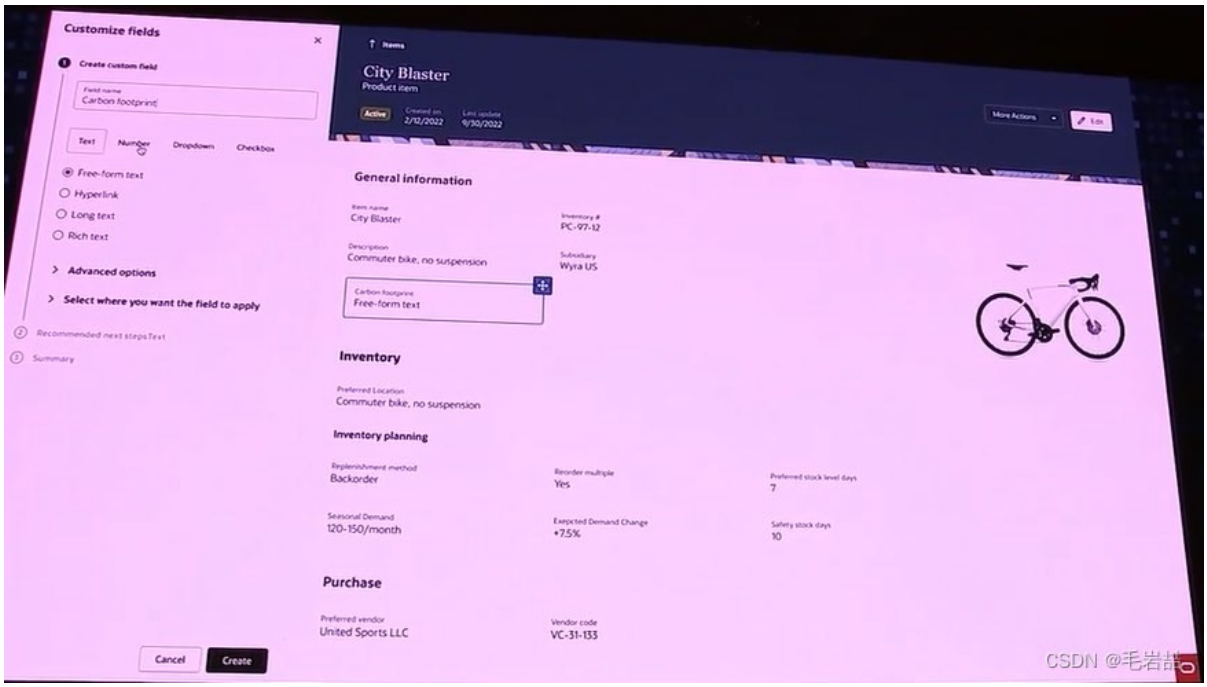
  多个 Field Groups 水平分布。





截止到目前的 23.1 版本，Form 的布局还是由以上的几个参数来支配的。但是，根据 22 年 9 月底 SuiteWorld 2022 的未来展望来看，UI 会被重构，布局逻辑完全改变。

我们拭目以待吧！



阅读原文：[定制表格布局的要点]

定制内容本地保存

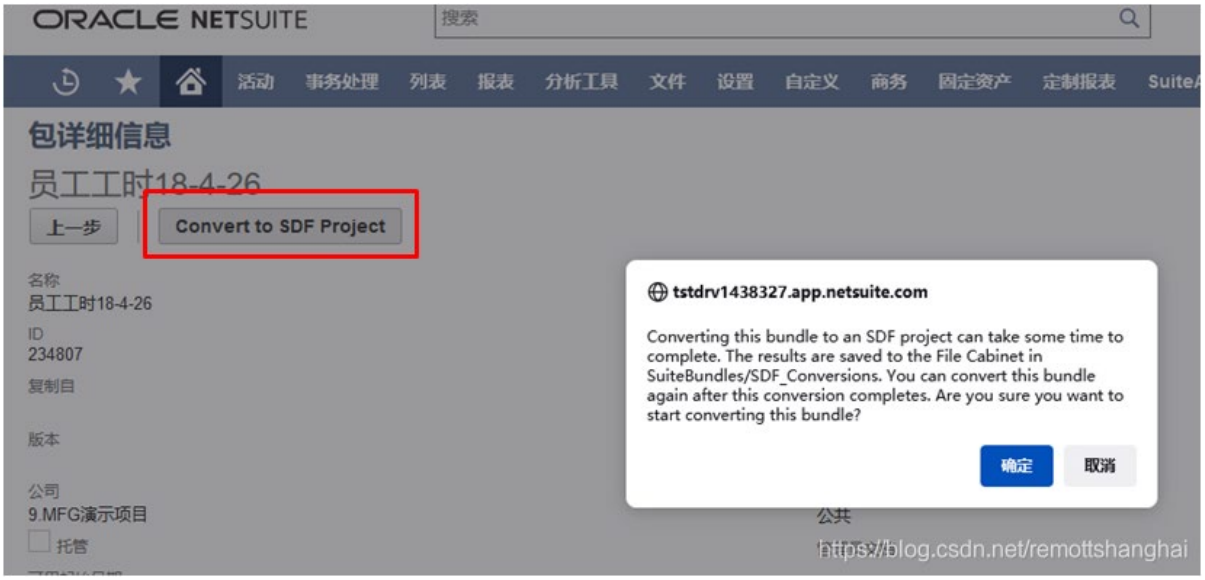
作者：Rick 2021.08.16

由于 Bundle 在 Repository 的暂存功能在今年 1 月 18 日被废止了，所以如果我们想把开发的内容保存在 NetSuite 环境以外的话，目前有个办法——**Convert to SDF Project**，此功能发版于 19.2。

原理是：

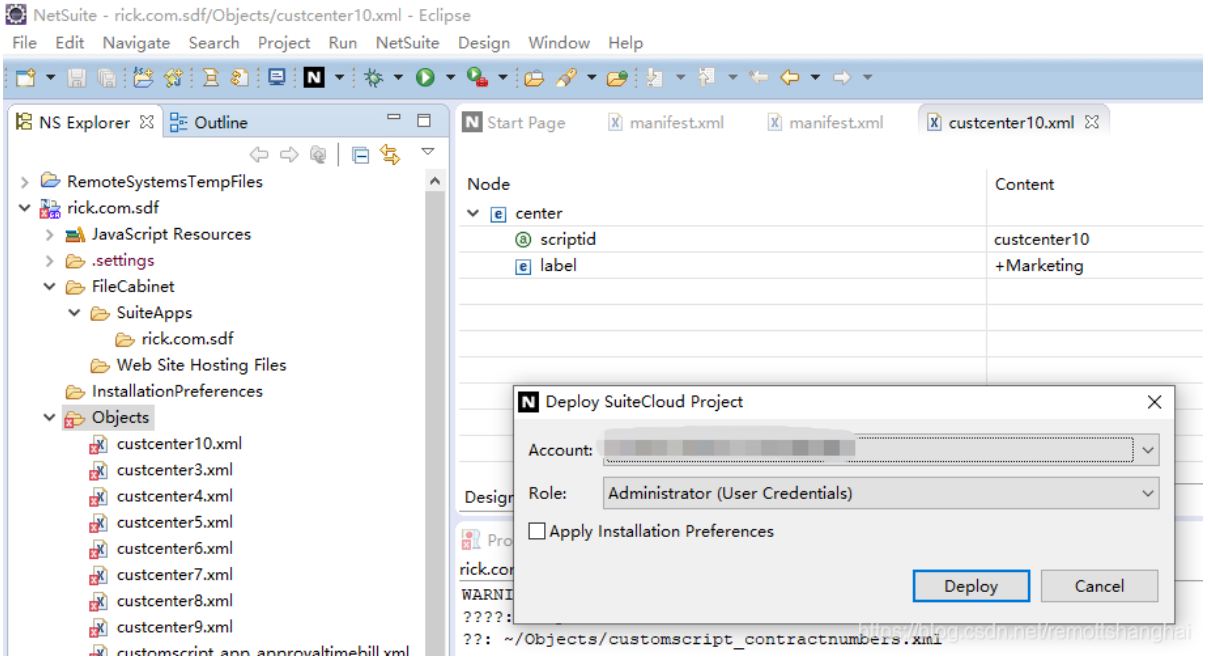
1. 把你想保存的内容先做个 Bundle；
2. 利用 Convert to SDF Project 功能导出为 Zip 文件，保存在本地；
3. 需要恢复到目标环境时，利用 SDF IDE 来部署到目标环境。

第一步：将需要打包的定制内容打包。在打包完成，或者是查看包时，都能看到 Convert to SDF Project 的按钮。确认后，系统会将定制内容存储于文件柜中。



第二步：在文件柜的“SuiteBundles-SDF_Conversions”文件夹中，可以下载上述生成的 zip 文件。

第三步：需要恢复到目标环境时，可以将 zip 中的文件导入到 IDE 中，利用 IDE 的部署功能，发布到目标环境。



具体操作可以观看视频演示，链接如下：

https://videohub.oracle.com/media/SuiteCloud+Development+Framework+Converting+SuiteBundles+to+Account+Customization+Projects/1_juro0wri?ed=383

阅读原文：定制内容本地保存

第二部分　顾问篇

1. 工具贴

Sublist 解释

作者：Rick　　2023.04.22

今朝汇编一下 Sublist 主题的知识点以备忘。

2 个数据源类型

- Related Record - 以 Saved Search 建立的关联记录；
- Child Record - 父子表；

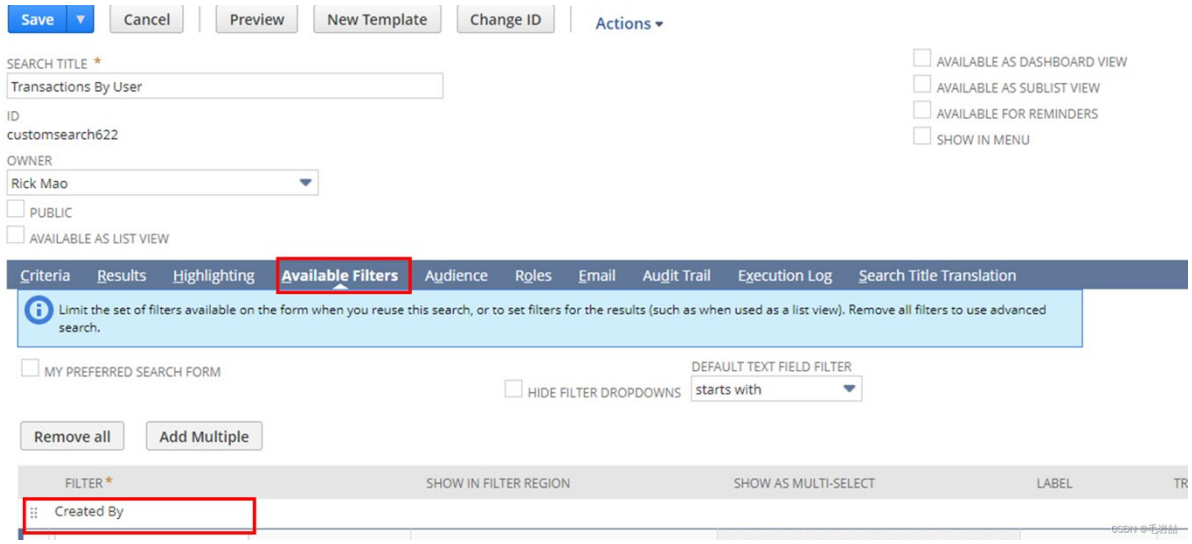
我们详细来说：

- **Related Record**
  - **Saved Search 关键点**

这种形式的 Sublist 是利用 Saved Search 作为 Sublist 的数据源，将某个字段作为 Key 来 Join 两个表。建立 Saved Search 时，要点有二：

1.关联字段类型必须是 List/Record 类型的；

2.Key 字段必须放在 Available Filter 的第一个；

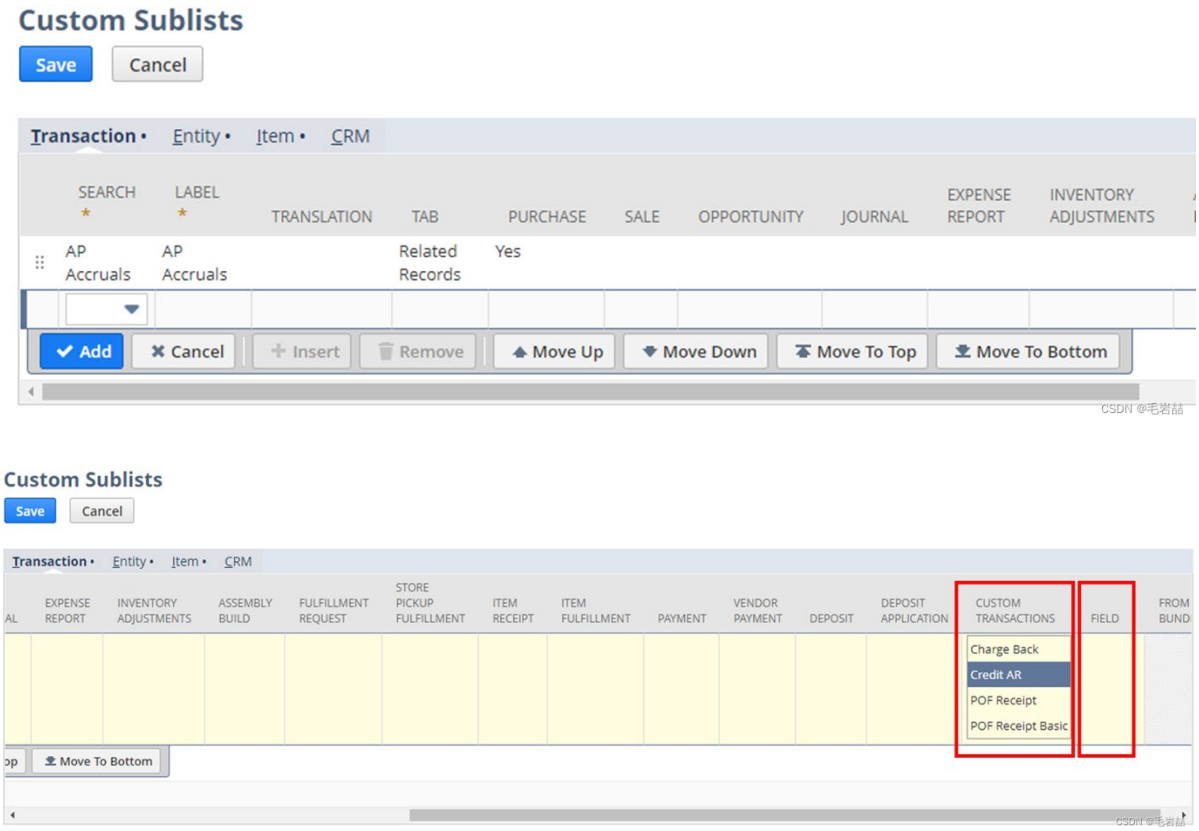


- **Sublist 的添加路径**

由于记录类型的不同，在添加 Sublist 时的路径有所不同。按照类型分为两类：

A：Standard Record Types (Including Custom Transaction Type)

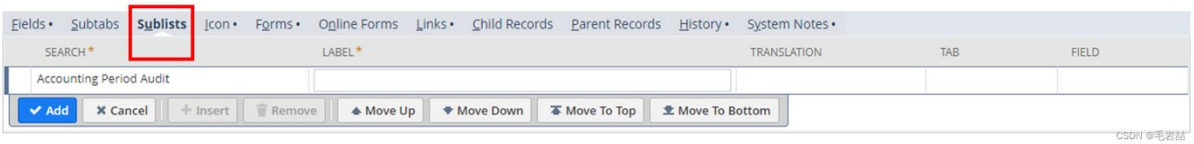
路径为：Customization > Forms > Sublists



上图中的 Field 字段，就是 Saved Search 中的 Field 字段。

B：Custom Record Type

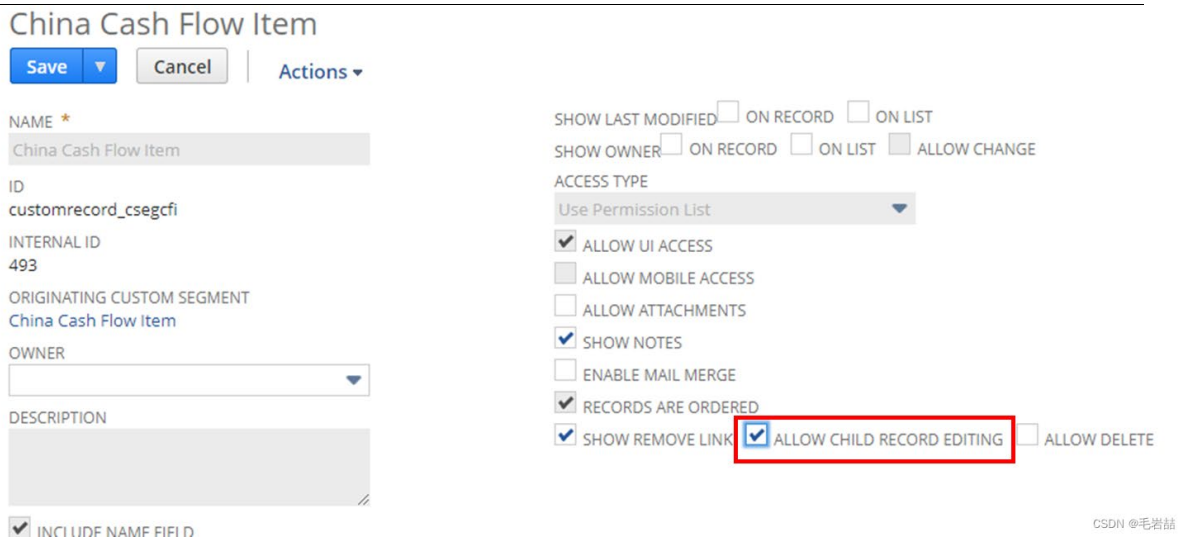
在 Custom Record Type 的定义界面进行添加。



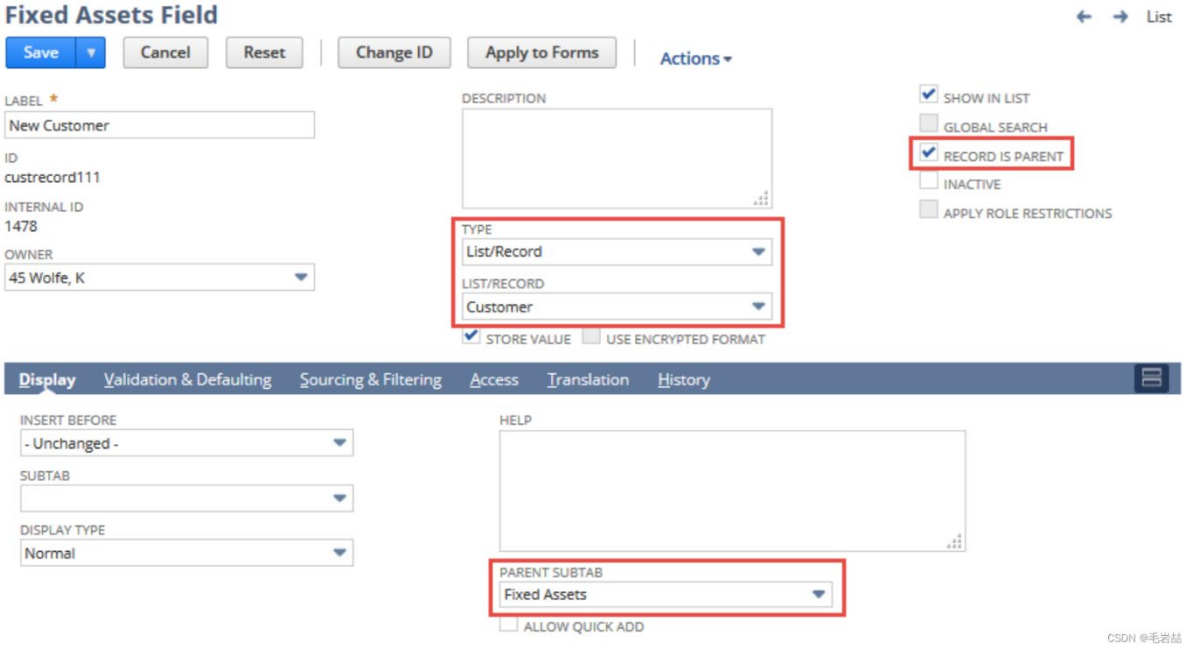
- **Child Record**

不同于上面用 Saved Search 来建立关联（Join）的方式，Child Record 顾名思义，就是在表的设计上采用了“父子表”的形式，是一种天然的 Sublist 关系，有两点需要注意：

1.在父表定义时，需要勾选“Allow Child Record Edit”参数，否则不能在编辑时显示子表（Sublist）。

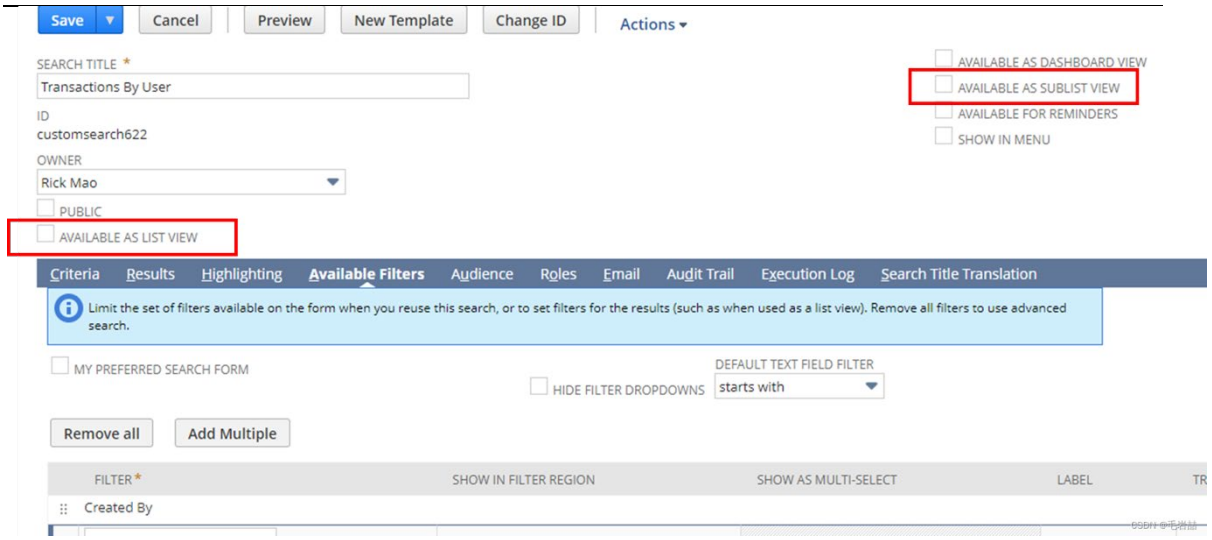


2.在建立子表时，需要用一个 List/Record 类型的字段建立父子关系（Record Is Parent）。

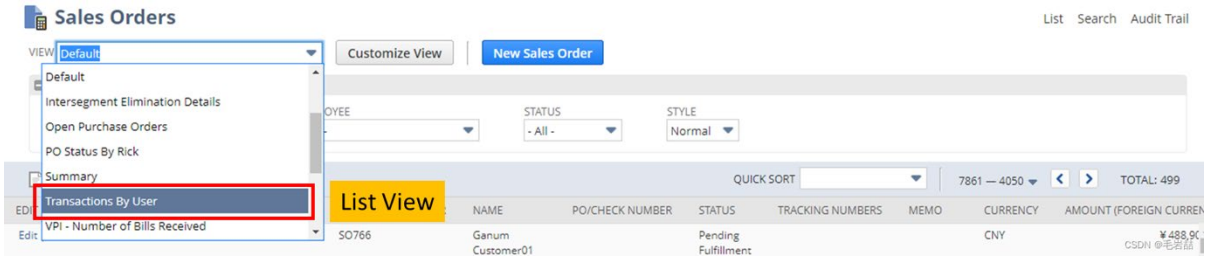


- **Saved Search 中有关 Sublist 的参数**

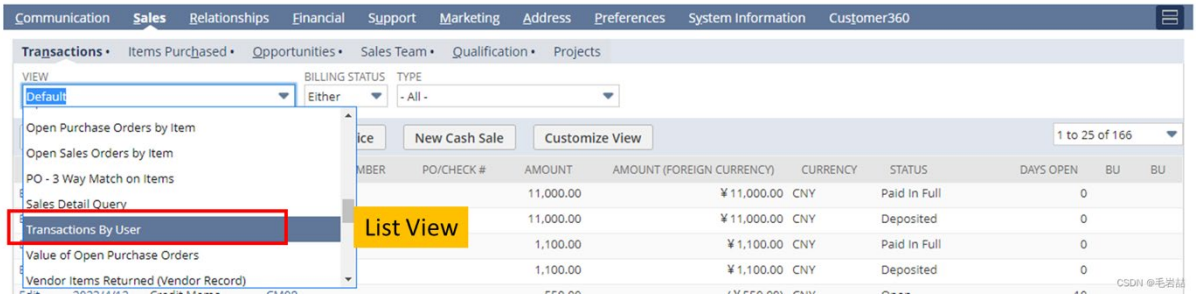
在 Saved Search 的定义中，有两个参数，一个是 Available As List View， 另一个是 Available As Sublist View。



其中前者的作用是决定是否显示在 List 的 View 选择中。



后者的作用是决定是否显示在 Sublist 的 View 选择中。



无论 Available As Sublist View 这个参数勾选与否，都不影响本文前面所说的 Related Record 类型 Sublists 的选择。

阅读原文： Sublist 解释

Decode 函数

作者：Rick 2022.12.25

Decode 的意思是 De-Code，就是解码。它是一个来自 Oracle 数据库的特别函数，它设计初衷是实现类似于解码的用途。例如：在查询结果中发现

A，那就意味着 Apple，返回 Apple

B，那就意味着 Banana，返回 Banana

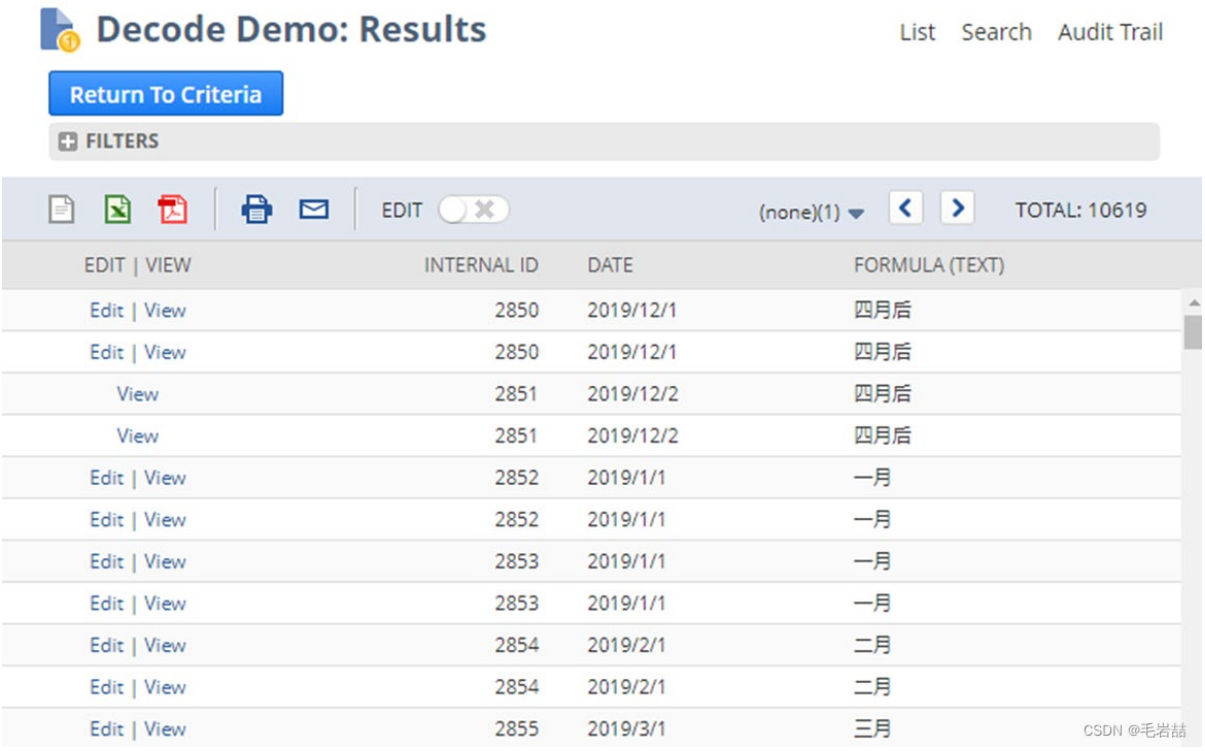
C，那就意味着 Charlie，返回 Charlie

就是这样一个东西。其语法如下所示：

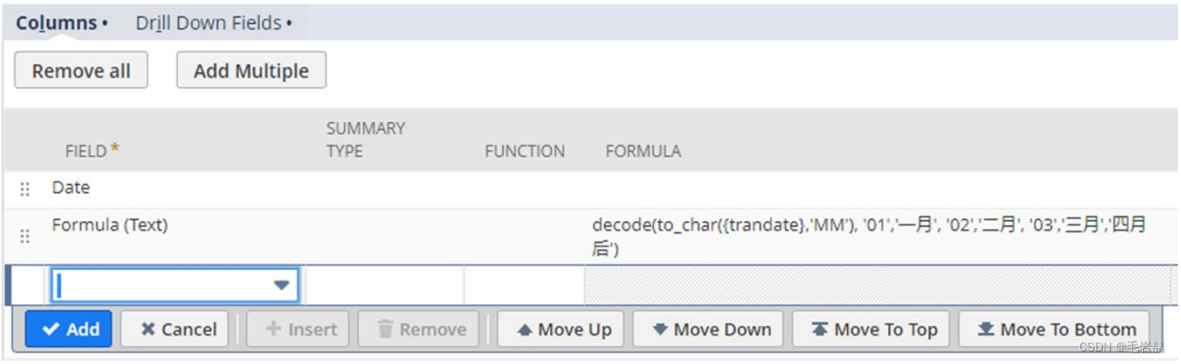
Decode

Function	Syntax	Short Descriptions	Example
DECODE	DECODE(expr, search, result [, search, result]... [, default])	Compares expr to each search value one by one. If expr is equal to a search, the corresponding result is returned. If no match is found, default is returned.	DECODE({systemnotes.name}, {assigned},'T','F')

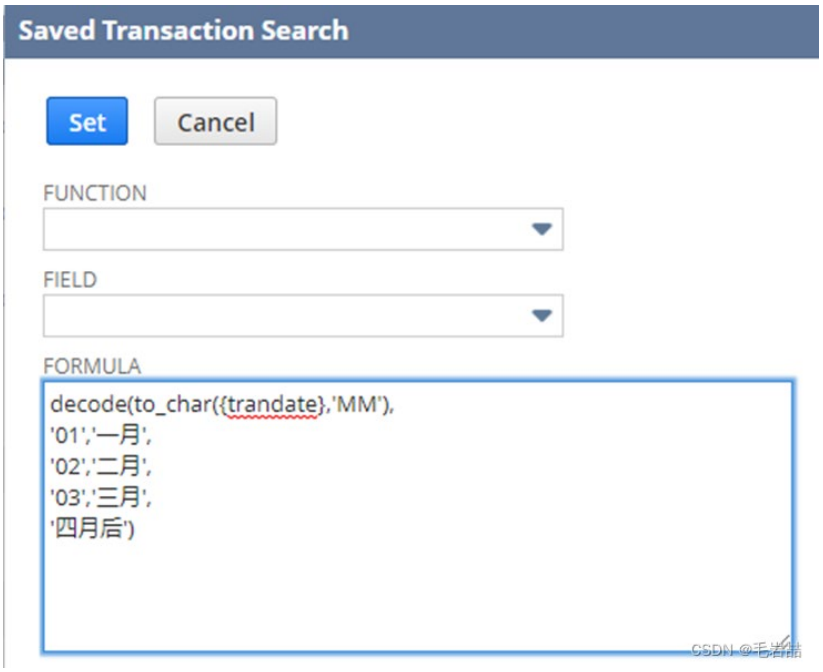
基本上看不太懂哈，给个例子吧。例如，我们想在 Transaction 查询时，把碰到的 Date 字段用月份表示一下，这样统计报表好看些，实现这样的效果：



实现方式如下：



编辑时，为了看起来舒服，可以这样排版：



就着这个例子，我们解释一下参数：

decode(A=你想比较的字段或值，B=跟 A 那个值去比较，C=如果 A 和 B 两者相同就返回这个值，重复 B，重复 C...，Z=跟谁都不相同就返回这个值)

```
decode(to_char({trandate},'MM'), '01','一月', '02','二月', '03','三月','四月后')
```

你大概会说，这就是简化版的 Case When 嘛，确实是，但是简洁真好！

阅读原文：Decode 函数

Item Type ID

作者：Rick 2022.11.28

Item Type ID 除了在脚本中用，在 Saved Search 环境中是无法使用的。于此，只作为 SuiteScript 时的参数列表。

Item Type	Item Type ID
Assembly	Assembly
Description	Description
Discount	Discount
Down Load Item	DwnLdItem
Item Group	EndGroup
GiftCertification	GiftCert
Group	Group
Inventory Item	InvtPart
Kit	Kit
Markup	Markup
Non Inventory Item	NonInvtPart
Other Charge	OthCharge
Payment	Payment
Service	Service
ShipItem	ShipItem
Subtotal	Subtotal
TaxGroup	TaxGroup
TaxItem	TaxItem

阅读原文： Item Type ID

Transaction 表中的通用与专用字段

作者：Rick 2022.05.29

由于 NetSuite 对企业事务处的理解是，将各类事务类型抽象化之后，合并在“Transaction”这个统一表中去。这是 NetSuite 构架师的选择，我个人推测此种设计的好处在于“简洁”，可以复用一些字段。例如，采购订单和销售订单，都会有“交易对象”，“交易行 Item”，“本外币金额合计”，“币别”等等相同的属性字段，如果能够将之合并，就能够降低数据表中字段设计上的冗余。

但是这样设计的不好之处，也非常明显：

- **一个 Transaction 表承载太多类型的业务信息。**

虽然这个表对于整体系统来说，保持了简洁，但是导致这个表相对于某个类型的业务，有太多的冗余字段。对于用户来说，在处理某一类型的事务处理时，例如“销售订单”时，会发现表中有大量无关的，非“销售”类的字段。对于用户来说，从一大堆字段中选择某几个，是有难度的。这就是我们写这篇文章的原因。

- **未分表导致的交易量雪崩。**

如果做了分表设计，例如销售类交易与采购类交易分表设计的话，那么如果“销售类交易”量激增的话，不会拖慢“采购类交易”的数据库操作性能。但是由于 NetSuite 是将各种事务处理类型混合在一个表中。那么出现“单一业务类型“激增带来整体事务处理的性能下降的情况。这就是我们把 NetSuite 的应用定位于中型企业的原因，用户 1000 人是个坎。

基于 Transaction“统一表”设计的基本事实，我们需要将 Transaction 表中众多字段，按照“通用”与“专用”进行分类，进而开展下一步工作。

因此，在 Transaction 字典中我们将对字段加入 Tag（通用/专用）。

通用字段

- 交易实体（Entity）
- 行属性、标志
- 金额
- 交易日期
- 创建信息

专用字段

事务处理类型特色的专有字段。例如，销售订单涉及的人员，工单涉及的完工信息等等。

具体有哪些事务处理类型，可以查询系统帮助：

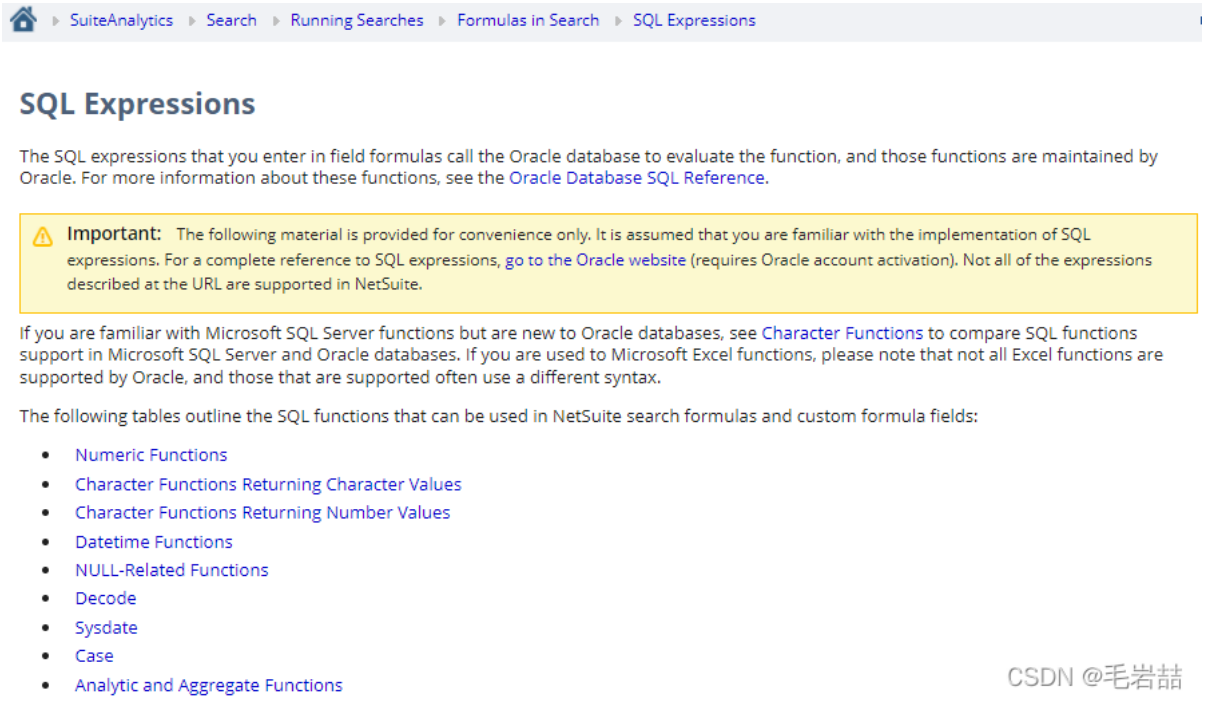
NetSuite Applications Suite - Transaction Types Included in Monthly Transaction Lines Metric (oracle.com)

阅读原文：Transaction表中的通用与专用字段

常用字符串函数

作者：Rick 2022.01.27

近期写了一些 Saved Search，其中涉及一些字符串公式。在帮助中也有这些帮助的粗略解释，但是有些看不明白。只有实践后才有感觉。



有感于这些字符串函数可能会被多次复用，所以计划将这些函数和应用场景总结一下。今后也会不断补充，将此贴作为字典档以备查。

- **截取一个字符串中的某一段字串。**

  例如：想截取 “2022” 这个字符串中 “22” 字串。语法如下:

```
SUBSTR({字段名},3,2)
```

  这里的 3 指是从第三个字符开始，2 指取两个。

- **截取一个字符串中的某个字符之后的一个字串。**

  例如：想截取 “XYZ-ABC” 这个字符串中 “-” 之后 “ABC” 这个字串。语法如下:

```
SUBSTR({字段名},INSTR({字段名},'-')+1,99)
```

其中 Instr（）的用途是返回待查找字符串的位置。结合 Substr 的用法，就能做到对字符串中特殊字符的定位和截取。上面例子中的 99，是指从“-”之后取 99 个字符，在上例中设为 3 也是可以的。

- **数字月份转换英文月份。**

例如：字段采用数字表示月份，但是在 Saved Search 结果中希望将数字转为英文月份。语法如下:

```
CASE
WHEN {字段名}='1' THEN 'JAN'
WHEN  {字段名}='2' THEN 'FEB'
WHEN  {字段名}='3' THEN 'MAR'
WHEN  {字段名}='4' THEN 'APR'
WHEN  {字段名}='5' THEN 'MAY'
WHEN  {字段名}='6' THEN 'JUN'
WHEN  {字段名}='7' THEN 'JUL'
WHEN  {字段名}='8' THEN 'AUG'
WHEN  {字段名}='9' THEN 'SEP'
WHEN  {字段名}='10' THEN 'OCT'
WHEN  {字段名}='11' THEN 'NOV'
WHEN  {字段名}='12' THEN 'DEC'
END
```

- **字符串合并**

例如，希望将两个字符型字段的值进行合并，显示在一个字段上。可以使用 CONCAT 函数。语法如下:

```
CONCAT({字段1},{字段2})
```

上面语句的返回结果为“字段 1 字段 2”。

如果希望将更多的字符型字段值进行合并，显示在一个字段上。可以使用管道符“||”，语法如下：

```
{字段1}||{字段2}||{字段3}
```

上面语句的返回结果为“字段 1 字段 2 字段 3”。

- **字符串替代**

例如，供应商编号为“VEN1009”，希望能够将“VEN”替代为“供应商”。可以使用 REPLACE 函数，语法是 REPLACE(待处理字段值，查找字符串，替代字符串)，示例如下：

```
REPLACE({供应商编号字段},'VEN','供应商')
```

上面语句的返回结果为“供应商 1009”。

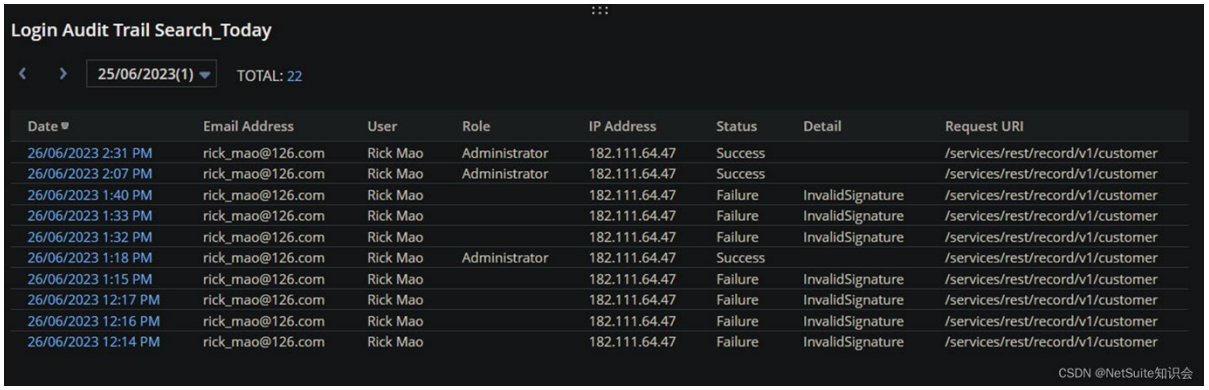
阅读原文：常用字符串函数

2. 开发

OAuth1.0 中 InvalidSignature 问题

作者：Rick 2023.06.26

本周闭关写代码，用Java通过TBA方式访问NetSuite REST Webservices。由于手生，卡在InvalidSignature报错上，在这个问题上被卡了一整天。



Login Audit Trail Search_Today

25/06/2023(1) TOTAL: 22

Date	Email Address	User	Role	IP Address	Status	Detail	Request URI
26/06/2023 2:31 PM	rick_mao@126.com	Rick Mao	Administrator	182.111.64.47	Success		/services/rest/record/v1/customer
26/06/2023 2:07 PM	rick_mao@126.com	Rick Mao	Administrator	182.111.64.47	Success		/services/rest/record/v1/customer
26/06/2023 1:40 PM	rick_mao@126.com	Rick Mao		182.111.64.47	Failure	InvalidSignature	/services/rest/record/v1/customer
26/06/2023 1:33 PM	rick_mao@126.com	Rick Mao		182.111.64.47	Failure	InvalidSignature	/services/rest/record/v1/customer
26/06/2023 1:32 PM	rick_mao@126.com	Rick Mao		182.111.64.47	Failure	InvalidSignature	/services/rest/record/v1/customer
26/06/2023 1:18 PM	rick_mao@126.com	Rick Mao	Administrator	182.111.64.47	Success		/services/rest/record/v1/customer
26/06/2023 1:15 PM	rick_mao@126.com	Rick Mao		182.111.64.47	Failure	InvalidSignature	/services/rest/record/v1/customer
26/06/2023 12:17 PM	rick_mao@126.com	Rick Mao		182.111.64.47	Failure	InvalidSignature	/services/rest/record/v1/customer
26/06/2023 12:16 PM	rick_mao@126.com	Rick Mao		182.111.64.47	Failure	InvalidSignature	/services/rest/record/v1/customer
26/06/2023 12:14 PM	rick_mao@126.com	Rick Mao		182.111.64.47	Failure	InvalidSignature	/services/rest/record/v1/customer

CSDN @NetSuite知识会

直到终于到来的Aha时刻。

在NetSuite中的样例代码是PHP的， 我平移到Java后，代码逻辑丝毫未变。反复核对代码后，发现代码没有任何的问题。按照NetSuite系统的指引，流程如下：

1. 构建Base String。这时，请注意字符串的格式，有三种格式。我采用的是REST Webservices，所以按照相应指引进行了构建。

请参考：

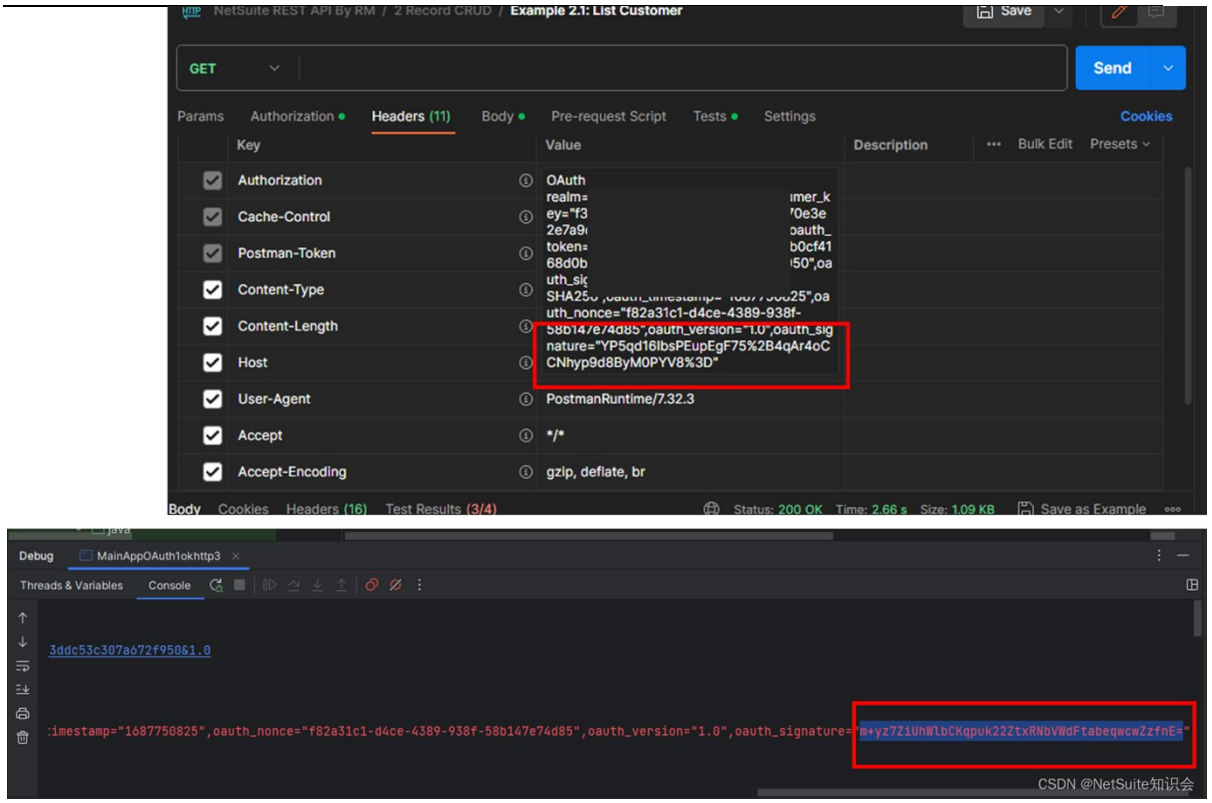
NetSuite Applications Suite - The Signature for Web Services and RESTlets

2. 构建Secret。

3. 用Secret对Base String进行加密，形成Signature，加密方法为HMAC-SHA256。

4. 把Signature结合其他字串，形成Authorization字串，赋给Header。

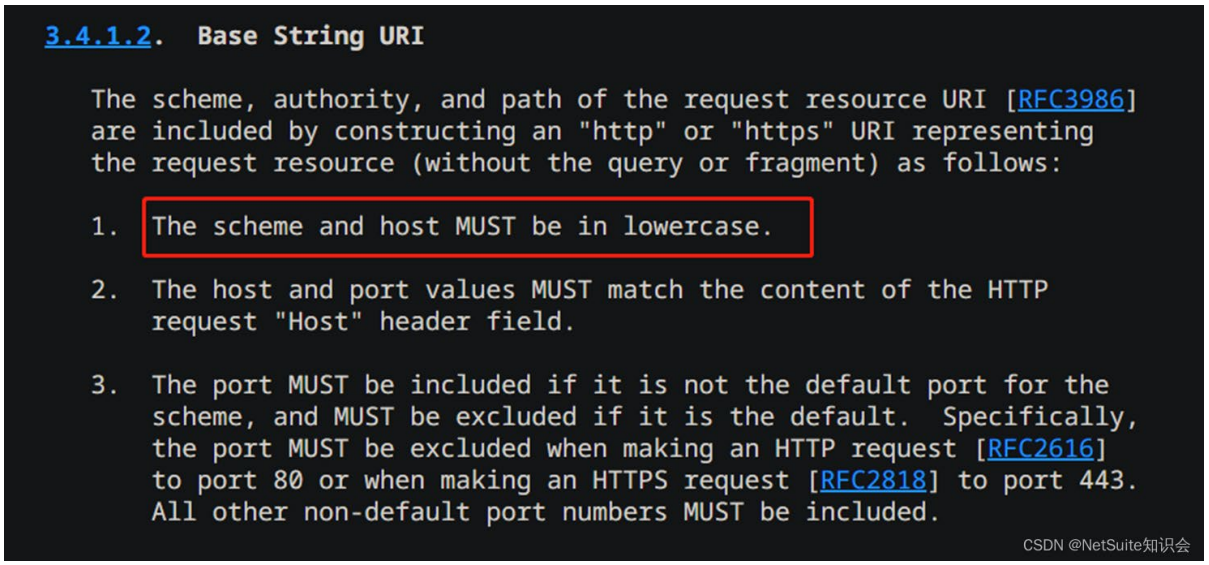
反复检查上述逻辑，但是一直报InvalidSignature错。相同的参数，在Postman中是毫无问题的。我们在对比了Header中的Authorization字符串后，发现只有最后的oauth_signature的值是有差别的。如下所示：



Signature的差别，只会有两种可能性，一是加密方法出错，二是Base String出错。在确定了HmacSHA256没有问题后，我们把问题聚焦在了Base String。

最后，虫子找到了！

原来在OAuth1.0的规范中，host必须是小写的。例如，123456-SB1必须格式化为123456-sb1。



但是，在构建的Header中，Host是要大写的，这就是大坑所在。

```java
//访问NetSuite REST Webservices，请注意Base String中Host的小写格式。
//Rick Mao 2023-6-26


import okhttp3.HttpUrl;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;
import okhttp3.logging.HttpLoggingInterceptor;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.nio.charset.StandardCharsets;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.net.URLEncoder;
import java.util.*;

public class MainAppOAuth1okhttp3 {
    private static final String NETSUITE_ACCOUNT = "你的账户"; //对字母大写
    private static final String NETSUITE_CONSUMER_KEY = "你的consumer key";
    private static final String NETSUITE_CONSUMER_SECRET = "你的consumer secret";
    private static final String NETSUITE_TOKEN_ID = "你的token id";
    private static final String NETSUITE_TOKEN_SECRET = "你的token secret";
    // Generate the timestamp and nonce
    private static final String timestamp = Long.toString(System.currentTimeMillis()
    private static final String nonce = UUID.randomUUID().toString();

    public static void main(String[] args) throws Exception {
        // Create OkHttpClient with logging interceptor for debugging
        HttpLoggingInterceptor loggingInterceptor = new HttpLoggingInterceptor();
        loggingInterceptor.setLevel(HttpLoggingInterceptor.Level.BODY);
        OkHttpClient httpClient = new OkHttpClient.Builder()
                .addInterceptor(loggingInterceptor)
                .build();

        // Create the request URL
        HttpUrl requestUrl = HttpUrl.parse("https://" + NETSUITE_ACCOUNT + ".suiteta

        // Generate the Authorization header value
        String authorizationHeader = generateAuthorizationHeader(requestUrl.toString
```

以上代码已调试通过，但因为代码较长，上图只是其中一部分，如需参考，请移步NetSuite知识会原文中参考复制。

阅读原文：NetSuite OAuth1.0中InvalidSignature问题

NetSuite .id 的用法

作者：Rick 2023.05.06

我们必须认清一个事实，NetSuite Saved Search是一个被封装化的SQL查询工具。在NetSuite的早期版本中，可以利用Formula字段做很多SQL语句上的灰色应用。但是慢慢地，灰色应用范围被压缩了，目前只剩下一个“.id”的应用了。

今朝我们就谈谈.id的用法，以及其背后的逻辑。在我们早前的一篇文章中，提到过某些字段会基于不同的语言环境在Search结果中返回不同的值。这样的话，会造成错误。文章链接如下：

NetSuite 交易记录类型中文环境下的失效_毛岩喆的博客-CSDN博客

近期，我们发现.id能够更加彻底解决这个语言环境导致的意外问题。

举个例子，在一个Transaction Search中，我们想在结果字段中用Account Type来作为条件，格式化结果。类似于：

```
case when {accounttype}='Other Current Liability' then '正确' else '错误' end
```

但是发现问题来了。因为Account Type在不同的语言环境下的字符串，例如在中文环境下字符串是“其他流动负债”。那么，我们的公式将不得不改为：

```
case when {accounttype}='Other Current Liability' or {accounttype}='其他流动负债'
then '正确' else '错误' end
```

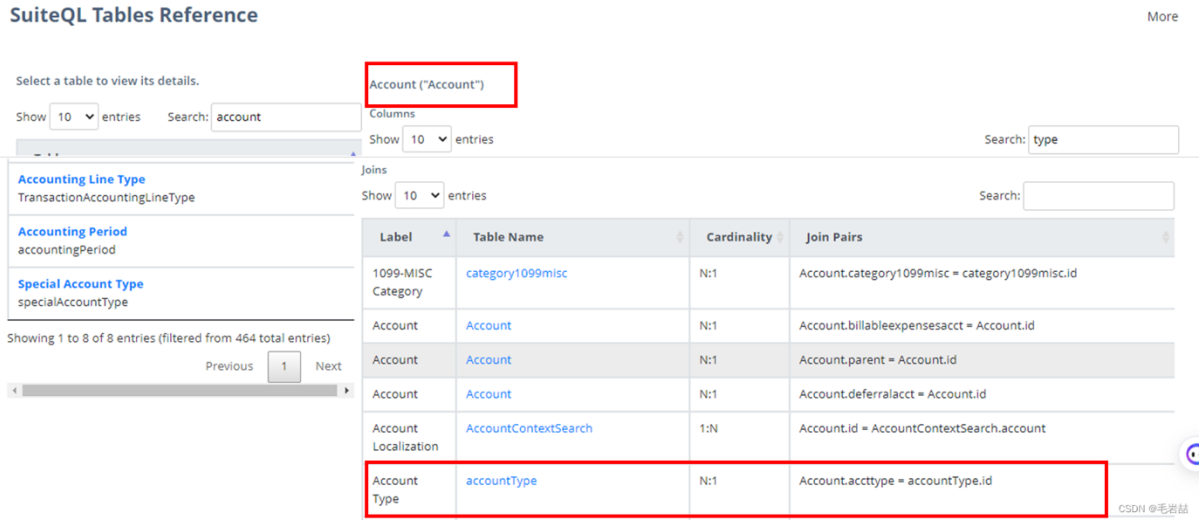
但是，如果有法文用户呢，德文用户呢？所以，这不是个办法。

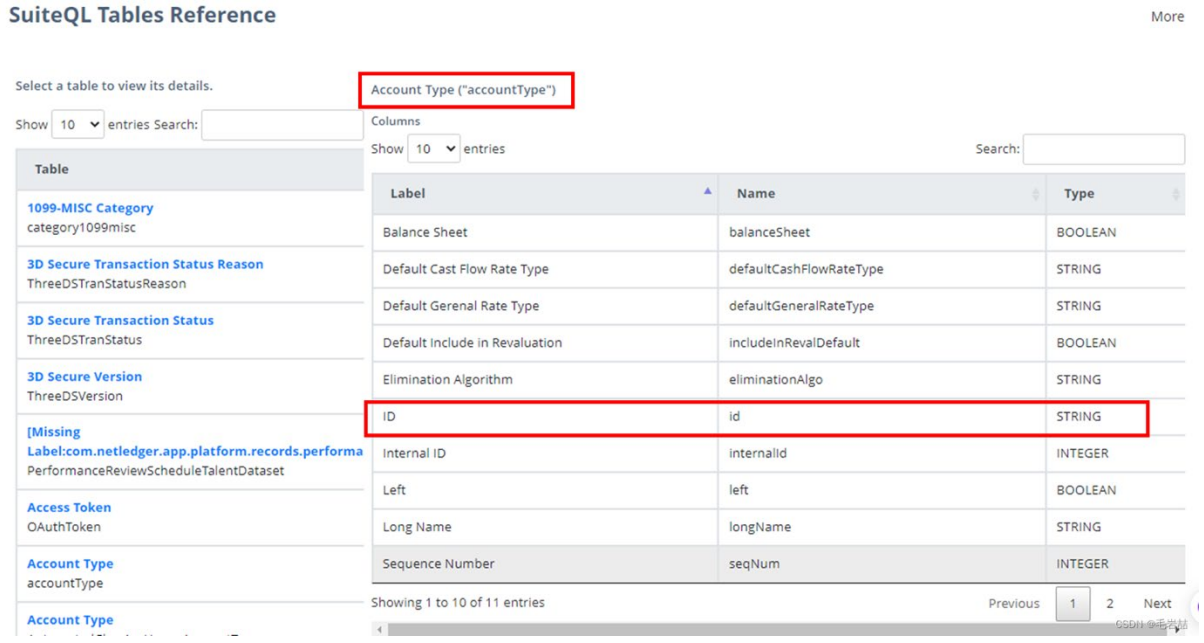
NetSuite留的.id就发挥作用了。

作为Formula中的灰色应用，用户可以在公式中的基本上任何字段中增加.id作为结果输出的加强，这时输出的就是这个字段的内部ID值。这个ID值通常为整型或者原生英文的字段串，不会因为语言环境而改变，从而解决我们上面面临的问题。

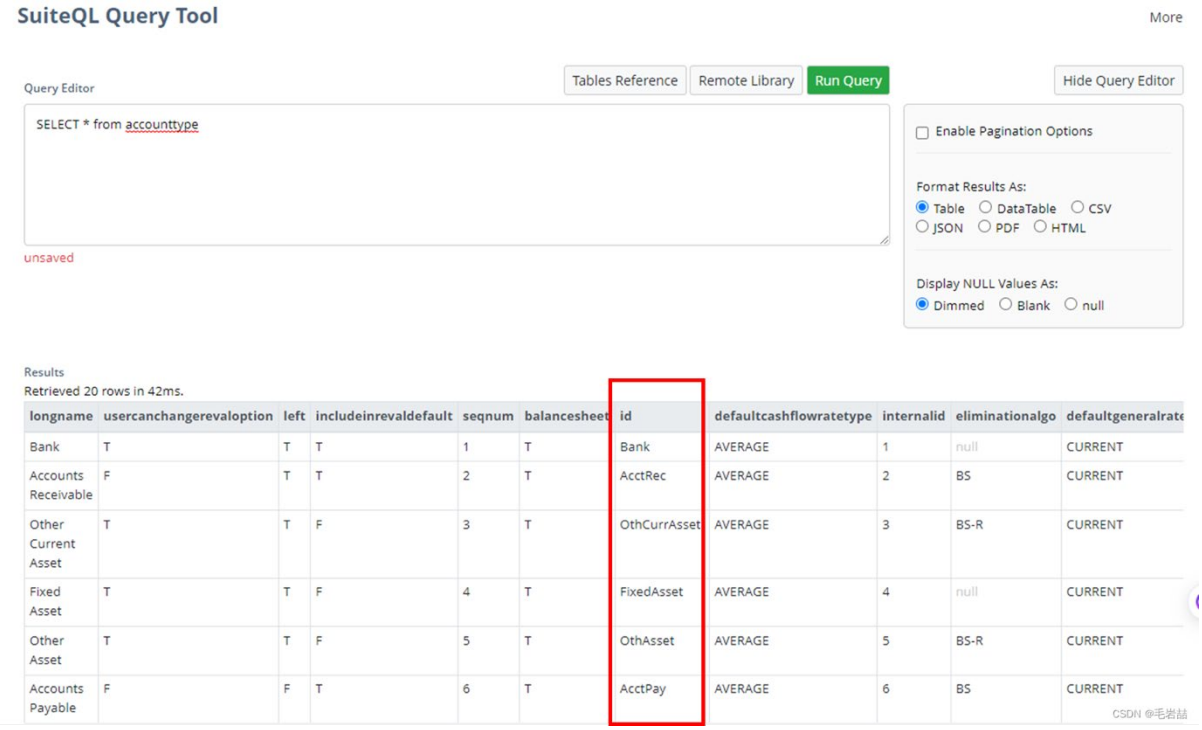
这个ID值其实是在数据库中某个字段所关联（Join）表中的id字段的值，举个例子：

Account会Join一个表叫做Account Type，在这个表中是有个字段真的叫做ID的。





这个Account Type表的ID字段为String类型，我们拉取其中的值看一下。



longname	usercanchangerevaloption	left	includeinrevaldefault	seqnum	balancesheet	id	defaultcashflowratetype	internalid	eliminationalgo	defaultgeneralrate
Bank	T	T	T	1	T	Bank	AVERAGE	1	null	CURRENT
Accounts Receivable	F	T	T	2	T	AcctRec	AVERAGE	2	BS	CURRENT
Other Current Asset	T	T	F	3	T	OthCurrAsset	AVERAGE	3	BS-R	CURRENT
Fixed Asset	T	T	F	4	T	FixedAsset	AVERAGE	4	null	CURRENT
Other Asset	T	T	F	5	T	OthAsset	AVERAGE	5	BS-R	CURRENT
Accounts Payable	F	F	T	6	T	AcctPay	AVERAGE	6	BS	CURRENT

其值为原生的英文字符串，这些字符串是我们真正需要的值。

因此，上面的公式改为如下内容，行不行?

```
case when {accounttype.id}='OthCurrLiab' then '正确' else '错误' end
```

答案是：不行!

我前面提过“某个字段所关联（Join）表中的id字段的值”。如果{accounttype}字段没有关联任何表，所以你是取不出id字段的值的。虽然系统不报错，但是你就是取不出。我们需要做个变通，将公式改为{account.type.id}，也就是取Account这个关联表的Type关联表的ID字段的值。有点绕哈：）

```
case when {account.type.id}='OthCurrLiab' then '正确' else '错误' end
```

这样就可以了么？答案再次是：不行!

按照这种方式取出的值，有可能会有些异常，以上面为例，在Search结果中的字段上会出现“表达式错误”的话，记得格式化一下字符串（如果ID是String类型的话）。这是很诡异的情况，原因不明，反正这样做对了。

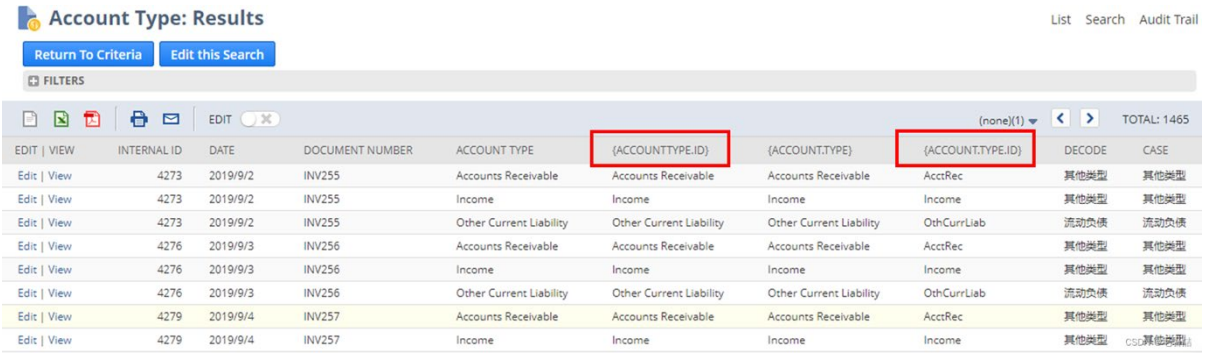
```
case when to_char({account.type.id})='OthCurrLiab' then '正确' else '错误' end
```

下面是实验数据，自己操练一下吧。


```
DECODE({account.type.id},'OthCurrLiab','流动负债','其他类型')

case when to_char({account.type.id})='OthCurrLiab' then '流动负债' else '其他类型' end
```

请注意case when中 to_char的使用。不是多余，你可以去掉试试。但是用decode就运行正常，奇了个怪。



阅读原文： NetSuite .id的用法

NetSuite nlapiRunReport 隐秘的 API

作者：Rick 2023.03.25

作为中国现金流量表剖析的后续篇，今朝我们谈一个隐秘函数。在几周之前，当用户问我：“SuiteScript能调用标准财务报表的运行结果么？”我的答案是“No！”，斩钉截铁。但是直到我们从代码堆里发现了这个API - nlapiRunReport。终于又让我们对NetSuite的理解又进了一层，再一次印证了我的认知局限，学艺不精、惭愧不已。

Google、GitHub了一圈，只有一篇有价值的文章，亦来自C站。

NetSuite 记录通过脚本从报表中读取数据的过程_吴铁柱儿的博客-CSDN博客

Chat GPT的答案其实也是来自网络，只说了是隐藏，并且最好别用。因为会被进化掉，只是概率较小。

基于上面提供的细节，在咨询了NetSuite开发大神后，我们渐渐扒出了这个API的全貌。

nlapiRunReport 是一个隐藏的API，遵从SuiteScript 1.0规范，其用途是对Financial Report的代码调用并获取其运行结果中的数据。在2.0被进化掉了，但是由于在某些官方本土化Bundle中仍然使用，所以只能逐步Phase Out，个人判断永远不会。因为，作为一个平台产品是不能不考虑既有的用户自己做的那些客制功能的，那是客户的资产了，如果弃用这个API，影响太大。想想各类End Of Available的那些基础API，你就知道退出成本有多高了。

> API定义：nlapiRunReport(reportId, reportSettings)
>
> - 参数reportId是财务报表的Internal ID；
> - 参数reportSettings是财务报表的搜索参数；

除了这个主体API外，还有几个附属函数，基本上可以顾名思义。

```
nlobjReportSettings(paramsObj.periodFrom, paramsObj.periodTo)
getColumnHierarchy()
getVisibleChildren()
getRowHierarchy()
getSummaryLine()
```

由于没有任何的官方文档可以参考，我们通过分析现有代码，做了Demo出来，实证有效，搞开发的同学参考之。

以下是Restlet脚本，Get Function = cn.rlv1.fin.standard.reports.run

由于环境有差异，请将Line 40处的参数调整为自己的。

```
var params = {
    subsidiary: '3', // Sub 3
    fiscalCalendar:'1',//Calendar 1
    periodFrom: '258', // Feb 2023
    periodTo: '259', // Feb 2023
    };
```

以下图中的代码是调试通过的，但由于代码较长，图中只是其中一部分，如需参考，请移步NetSuite知识会原文进行参考复制。

Enjoy your coding!

阅读原文：NetSuite nlapiRunReport隐秘的API

```
 *      subsidiary      - mandatory for One-World account, will throw error if it is
 *      fiscalCalendar  - mandatory if it is One-World account and Multi-Calendar fea
 *      periodFrom      - mandatory
 *      periodTo        - mandatory
 *      location        - optional
 *      department      - optional
 *      classification  - optional
 *
 * Return
 * String representative of below json object.
 * {
        startBalanceCurrent: 0,
        endBalanceCurrent: 0,
   }
 */
if (!cn) {
    var cn = {};
}
cn.rlv1 = cn.rlv1 || {};
cn.rlv1.fin = cn.rlv1.fin || {};
cn.rlv1.fin.standard = cn.rlv1.fin.standard || {};
cn.rlv1.fin.standard.reports = cn.rlv1.fin.standard.reports || {};

cn.rlv1.fin.standard.reports = function() {
    var isOW;
    var reportByPeriod;
    var generalLedgerColumnName;
    var unrealizedGainAndLossColumnName;

    this.doGet = function(params) {
        var params = {
            subsidiary: '3', // Sub 3
            fiscalCalendar:'1',//Calendar 1
            periodFrom: '258', // Feb 2023
            periodTo: '259', // Feb 2023
            };

        isOW = isOW();
        var paramsObj = validateParams(params);
```

3. 报表开发

SuiteQL 内建函数

作者：Rick 2023.05.19

之前写过一篇文章介绍SutieQL Query Tool，今天继续挖掘一下SuiteQL的价值。以前的文章链接如下：

NetSuite SuiteQL Query Tool_netsuite好用吗_毛岩喆的博客-CSDN博客

SuiteQL中有几个内建函数（Built-In），可以帮助我们在编写SQL语句时更加方便，并不详细的介绍可以参见在线帮助：

NetSuite Applications Suite - SuiteQL Supported Built-in Functions

今朝我们用一个SQL语句来综合实践一下：

```
SELECT
    Transaction.TranID,
    Transaction.trandate,
    Transaction.postingPeriod,
    TransactionAccountingLine.Account,
    BUILTIN.DF( TransactionAccountingLine.Account ) AS func_df,
    TransactionAccountingLine.Debit,
    TransactionAccountingLine.Credit,
    BUILTIN.CONSOLIDATE(TransactionAccountingLine.Credit, 'INCOME', 'DEFAULT', 'DEFAU
    BUILTIN.CURRENCY(TransactionAccountingLine.Credit) as func_currency,
    BUILTIN.CURRENCY_CONVERT(TransactionAccountingLine.Credit, 1) as func_currency_co
    BUILTIN.DF( TransactionLine.Subsidiary ) AS Subsidiary,
    BUILTIN.HIERARCHY(TransactionLine.Subsidiary, 'DISPLAY_JOINED')  AS func_hierarch
FROM
    Transaction
    INNER JOIN TransactionAccountingLine ON
        ( TransactionAccountingLine.Transaction = Transaction.ID )
    LEFT OUTER JOIN TransactionLine ON
        ( TransactionLine.Transaction = TransactionAccountingLine.Transaction )
        AND ( TransactionLine.LineSequenceNumber = TransactionAccountingLine.Transact
WHERE
    --Transaction.trandate >= BUILTIN.RELATIVE_RANGES('TY', 'START')
    Transaction.postingPeriod  in BUILTIN.PERIOD('LFY', 'START', 'ALL', '>')
    AND ( TransactionAccountingLine.Account IS NOT NULL )
ORDER BY
    Transaction.trandate
```

在这段语句中，我们涉及了如下几个函数：

BUILTIN.DF (TransactionAccountingLine.Account)

BUILTIN.CONSOLIDATE (TransactionAccountingLine.Credit, 'INCOME', 'DEFAULT', 'DEFAULT', 3, 263, 'DEFAULT')

BUILTIN.CURRENCY (TransactionAccountingLine.Credit)

BUILTIN.CURRENCY_CONVERT (TransactionAccountingLine.Credit, 1)

BUILTIN.HIERARCHY (TransactionLine.Subsidiary, 'DISPLAY_JOINED')

BUILTIN.RELATIVE_RANGES ('TY', 'START')

BUILTIN.PERIOD ('LFY', 'START', 'ALL', '>')

运行结果如下：

tranid	trandate	postingperiod	account	func_df	debit	credit	func_consolidate	func_currency	func_currency_convert	subsidiary	func_hierarchy
CR10	2022/2/1	225	1158	未实现的收益/损失	3000			5		Subsidiary 1	Parent Company : Subsidiary 1
CR10	2022/2/1	225	1162	100207 招商银行美元账户		3000	3000	5	3000	Subsidiary 1	Parent Company : Subsidiary 1

由于严重缺乏支持材料，还有两个函数，我们不清楚如何使用。

- CF
- NAMED_GROUP

我们在提了Support Case后，得到如下答复：

*For **BUILTIN.CF**, This function has only one parameter - UMD field where UMD means Unified Metadata. Intention of this function is to get CRITERIA version of field value. In some special cases, there is different SQL fragment defined for UMD field whether it is used as return value or criteria field. (e.g. UMD field which is used as simple join key on UMD level, but it is implemented by two or more keys in DB - composite key). This function will allow us to return criteria version as return value in case that we would like to filter data outside of DB (e.g. in JAVA code or on customer side).*

Examples of SuiteQL queries for BUILTIN.CF:

1. select transaction.status, BUILTIN.CF(transaction.status) from transaction

2. / does not return anything because "where transaction.status" and "select transaction.status" are different SQL expressions */*

*select * from transaction where transaction.status IN (select transaction.status from transaction)*

3. / returns everything because "where transaction.status" and "select BUILTIN.CF(transaction.status)" are same SQL expressions */*

*select * from transaction where transaction.status IN (select BUILTIN.CF(transaction.status) from transaction)*

*For **BUILTIN.NAMED_GROUP**, we know only following:*
parameters are: Builtin.Named_Group(record_name, group_name) where record_name is UMD name of record and group_name is name of group defined on that record. On each record there can be different set of group_names - so the question should be directed to owner of the specified record.

The function returns filter options for workgroups for some record types (filter by: My Location, My Subsidiary, My team, etc). Named groups (label: my) currently for records:

- *Department*
- *Class*
- *Location*
- *Subsidiary*
- *Entity*

Also, every key field targeting above records (such as Entity -> Location) should support filtering by the {Me} group.

Example of SuiteQL queries for BUILTIN.NAMED_GROUP:

1. *SELECT id FROM Transaction WHERE entity IN (BUILTIN.NAMED_GROUP('employee', 'me'))*
2. *SELECT id FROM Transaction WHERE entity IN (BUILTIN.NAMED_GROUP('employee', 'me'), BUILTIN.NAMED_GROUP('employee', 'me'), 5, 6, 7, 8)*

请注意：文中提到的SQL语句内容较长，上图只是其中一部分，如需参考，请移步NetSuite知识会中参考复制。

阅读原文：NetSuite SuiteQL 内建函数

Saved Search 中 When Ordered By Field 与 Keep Dense_Rank 辨析

作者：Rick 2023.02.26

今朝的题目是一个隐藏的宝藏话题，Saved Search中我们极少用的一个功能——When Ordered By Field和Keep Dense_Rank。

假如你碰到一个需求，要求是：“在销售历史中按照客户类别，取最早交易日期的销售金额，以识别VIP客户”。这是一个朴素的需求吧？你的直觉是一个Saved Search。

不要往下看，你想想该怎么做？

。

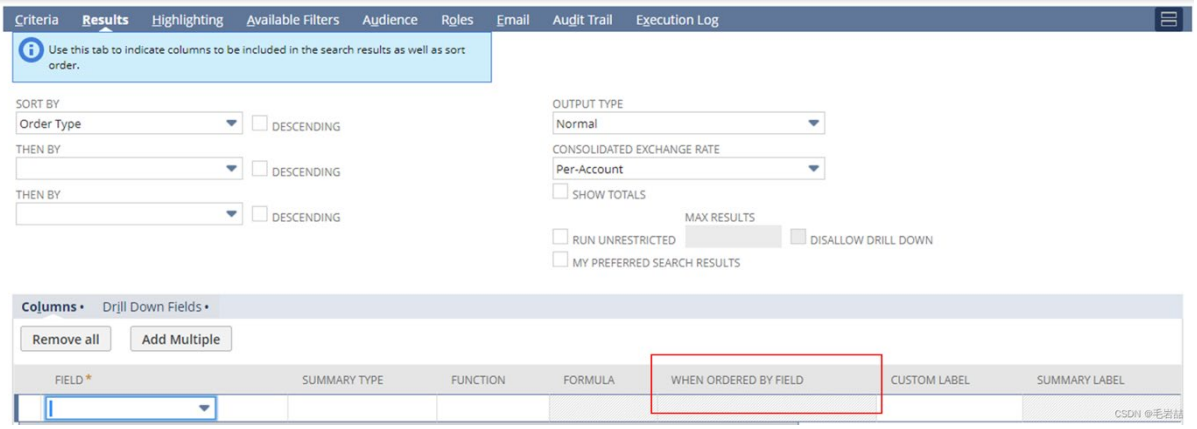
。 。

。 。 。

。 。 。 。

。 。 。 。 。

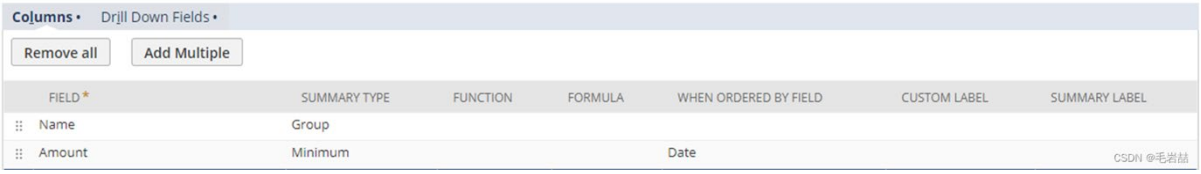
。 。 。 。 。 。



关键之处就在这里，When Order By Field。这个参数的作用是“按照某个字段进行排序”，然后取另外一个字段（Field）的最大或最小值（Summary Type）。

结合上面的需求，这几个参数这样用：

Field	Summary Type	When Ordered By
Name	Group	--
Amount	Minimum	Date



上面的含义是：

Group By Customer，Minimum Amount When Ordered By Minimum Date，就是指：按照客户分类，将最早的日期的交易金额列示出来。

这看起来是不错的，对吧？但是~~，如果最早的日期中，有多笔交易，金额不同，但是我们想取金额大的那个。怎么办？上面的功能只支持我们取“金额低“的记录，不支持“金额高”的。因为，Summary Type只有一个，要么时间和金额都取Max的，要么都取Min的，没有可能时间取远（Min）的，金额取大（Max）的。

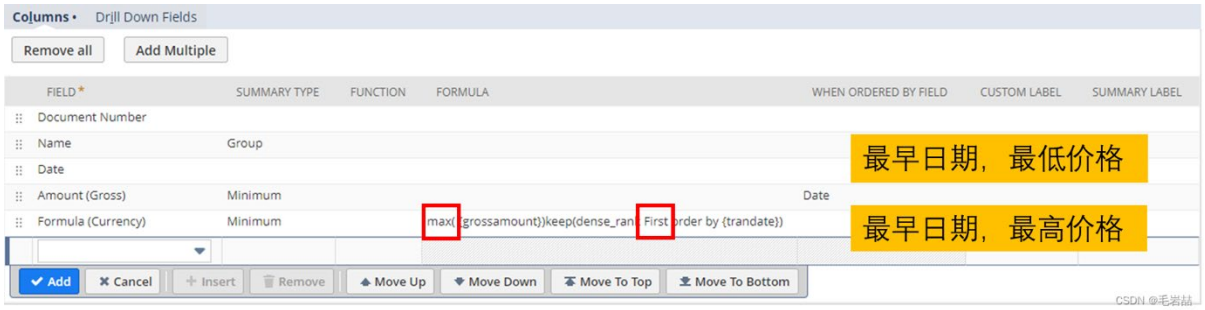
要解决这个同一日期有多个金额的场景需求，需要引入另一个主题“Keep Dense_Rank”。这是个Oracle的分析函数，其功能用于补足上述的缺陷。

Analytic and Aggregate Functions

Function	Syntax	Short Description	Example
DENSE_RANK	DENSE_RANK(expr [, expr]...) WITHIN GROUP (ORDER BY expr [DESC | ASC] [NULLS { FIRST | LAST }] [,expr [DESC | ASC] [NULLS { FIRST | LAST }]]...) or DENSE_RANK() OVER (PARTITION by expr [,expr...] ORDER BY expr [DESC | ASC] [NULLS { FIRST | LAST }] [,expr [DESC | ASC] [NULLS { FIRST | LAST }]]...)	Computes the rank of a value with respect to other values and returns the rank as a NUMBER. Always results in consecutive rankings.	DENSE_RANK ({amount}) WITHIN GROUP (ORDER BY {AMOUNT}) or DENSE_RANK() OVER (PARTITION BY {name} ORDER BY {trandate} DESC)
KEEP()	KEEP(DENSE_RANK { FIRST | LAST } ORDER BY expr [NULLS { FIRST | LAST }])	Qualifies an aggregate function indicating that only the FIRST or LAST values of the function are returned.	KEEP(DENSE_RANK LAST ORDER BY {internalid})
RANK	RANK(expr [, expr]...) WITHIN GROUP (ORDER BY expr [DESC | ASC] [NULLS { FIRST | LAST }] [, expr [DESC | ASC] [NULLS { FIRST | LAST }]]...) or RANK() OVER (PARTITION by expr [,expr...] ORDER BY expr [DESC | ASC] [NULLS { FIRST | LAST }] [,expr [DESC | ASC] [NULLS { FIRST | LAST }]]...)	Computes the rank of a value in a group of values. Can result in non-consecutive rankings if values are the same.	RANK() OVER (PARTITION by {tranid} ORDER BY {line} DESC) or RANK ({amount}) WITHIN GROUP (ORDER BY {amount})

CSDN @毛岩喆

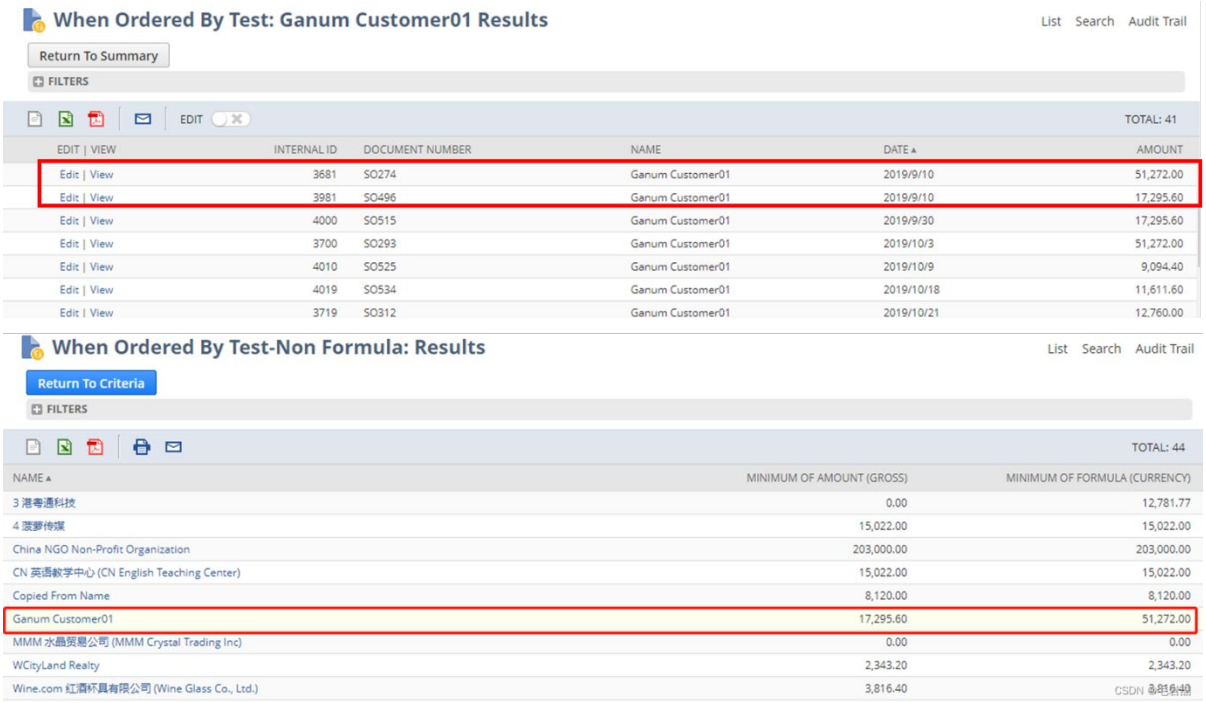
看上面的解释，大家基本懵圈，请参考下面的系统实例：



max ({grossamount})keep(dense_rank first order by {trandate})

这个公式的意思是：用Keep函数取First Transaction Date的日期，然后在那个日期里，取Max的交易金额。请注意，公式提供了First日期和Max金额组合，从而解决在一个日期里多笔交易，多个金额的问题。

下图是上面两种方法在取数上的差别。请观察一下两者的差异。



When Ordered By Test: Ganum Customer01 Results

List Search Audit Trail

Return To Summary

FILTERS

EDIT TOTAL: 41

EDIT | VIEW	INTERNAL ID	DOCUMENT NUMBER	NAME	DATE	AMOUNT
Edit | View	3681	SO274	Ganum Customer01	2019/9/10	51,272.00
Edit | View	3981	SO496	Ganum Customer01	2019/9/10	17,295.60
Edit | View	4000	SO515	Ganum Customer01	2019/9/30	17,295.60
Edit | View	3700	SO293	Ganum Customer01	2019/10/3	51,272.00
Edit | View	4010	SO525	Ganum Customer01	2019/10/9	9,094.40
Edit | View	4019	SO534	Ganum Customer01	2019/10/18	11,611.60
Edit | View	3719	SO312	Ganum Customer01	2019/10/21	12,760.00

When Ordered By Test-Non Formula: Results

List Search Audit Trail

Return To Criteria

FILTERS

TOTAL: 44

NAME	MINIMUM OF AMOUNT (GROSS)	MINIMUM OF FORMULA (CURRENCY)
3 港粤通科技	0.00	12,781.77
4 菠萝传媒	15,022.00	15,022.00
China NGO Non-Profit Organization	203,000.00	203,000.00
CN 英语教学中心 (CN English Teaching Center)	15,022.00	15,022.00
Copied From Name	8,120.00	8,120.00
Ganum Customer01	17,295.60	51,272.00
MMM 水晶贸易公司 (MMM Crystal Trading Inc)	0.00	0.00
WCityLand Realty	2,343.20	2,343.20
Wine.com 红酒杯具有限公司 (Wine Glass Co., Ltd.)	3,816.40	3,816.40

CSDN

希望本文能够启发我们的思考，NetSuite的功能还有多少你不知道，有多少隐藏的功能宝藏等你来发掘。

阅读原文：Saved Search中When Ordered By Field 与 Keep Dense_Rank辨析

NetSuite 知识会博客一直保持更新，敬请关注!