轻松学习 接口开发

GL 总账，AP 应付，OM 销售订单，工作流

Author: 王景远

Creation Date: January 16, 2015

Last Updated:

Document Ref:

Version: 1.0

目录

这篇文章将详细介绍总账接口，应付发票接口，销售订单 API。写文档的经验不多，可能比较乱或不周全，希望感兴趣的兄弟提出建议，补充周全，QQ：754496613。

数据导入有两种方法，接口表和 API。

一般导入流程：临时表->系统接口表->基表

临时表到接口表有两种：

一表对一表，比如日记账 gl_interface，一行临时表数据生成两行接口表数据。

一表对两表，比如应付发票 ap_invoices_interface，ap_invoice_lines_interface，一行临时表数据生成一行接口头表数据和一行接口行表数据。

对于有头有行的接口表，往往需要两次循环临时表，循环套循环 insert 数据到接口表，外层循环导入头数据到头表，内层循环导入属于这个头的所有行数据到行表。

临时表到接口表是我们工作的重点，在 insert 到接口表之前要经过一系列验证：

- 必输字段不能为空
- 同一单据的头信息要一致，如果 l_same_count=1，说明头信息一致，如果>=2，说明不一致

```
SELECT COUNT(*)
  INTO l_same_count
  FROM (SELECT COUNT(*)
         FROM cux_2_fnd_imports_temp pit
        WHERE pit.order_number = r_order_number.order_number
        GROUP BY attribute1,
                 attribute2,
                 attribute3,
                 attribute4,
                 attribute5,
                 attribute6,
                 attribute7,
                 attribute8,
                 attribute9)
```

- 转换验证，业务实体转换成org_id，订单类型转换成order_type_id等等

```
        SELECT hou.organization_id
           INTO r_importable_data.org_id
           FROM hr_operating_units hou
          WHERE hou.name = r_importable_data.operating_unit;
```

- 数字验证，to_number
- 日期验证，to_date，to_char

1、日记账接口导入

1.1、用日记账导入程序集成总账

用日记账导入程序集成总账应用和应付工资，应收账款，应付账款和固定资产等其它应用。在一个会计期内，可以导入源系统的会计数据到总账，然后检查，更新和过账这些日记账分录（总账凭证）。也可以导入之前会计系统的历史数据。
总账允许你从多个接口表导入数据。你可以根据特定的需求客制化接口表。每个特定的“来源/组 ID”组合仅从一个接口表中取数据。日记账导入程序每次从一个接口表中处理数据。

1.2、导入子帐薄和源系统数据到总账

1、定义帐薄，币种，会计科目，日记账来源，日记账分类，设置总账接受日记账导入数据。你也应该运行优化器程序，定义你的并发程序控制。
2、导出源系统数据到总账接口表 GL_interface
3、运行日记账导入并发程序
　　你必须有总账的读写权限，或接口表中的平衡段或管理段的读写段权限.
4、用日记账导入执行报表检查导入的日记账的状态，日记账导入执行报表为每一个日记账输入来源打印一行。
5、如果你偶然遇到少量的错误，你可以直接在 GL_interface 表中修改。
6、如果你遇到较多的错误，你应该删除 GL_interface 表中的数据，在源系统中修改，然后再次运行日记账导入。
7、在发布（post）导入的日记账之前，检查导入的日记账。
8，发布你的导入的日记账。为了发布日记账批，你必须拥有总账读写权限或日记账行的平衡段或管理段的读写权限。

1.3、导入日记账准备

在使用日记账导入之前，设置总账来确保日记账导入顺利运行。
1、定义所有的会计科目段值，这些段值在源系统中被使用。
2、用账户设置管理器（Accounting Setup Manager）定义你的帐簿，为了使用它的帐薄，账户必须有完整的状态。
3、定义或生效所有的币种。这些币种在源系统中被使用。
4、定义日记账来源，这个来源在源系统中被使用。你也可以用日记账来源指定总账是否存储来自源系统的日记账参考信息。
5、定义日记账分类，这个分类在源系统中被使用。
6、如果你想日记账导入程序自动分配序列号给你的导入的日记账，你可以生效序列号，指定文档标号生成方法为自动（Automatic）。
7、打开会计期，这个会计期在源系统中被使用。你只能导入日记账到总账中打开的（Open）或未来可输入（Future-Enterable）的会计期。

8、运行优化器程序（Optimizer program）给会计科目段创建索引。
9、设置磁盘空间和内存使用大小来定义并发程序控制，可以提高日记账导入性能。日记账导入程序大约使用 1.4M 的内存。你也可以指定每次运行日记账导 入程序时是否存储日记账导入数据，如果不归档运行的会更快。
10、失效动态插入。日记账导入程序运行的更快如果它不必动态的创建新的账户组合。导入账户有两种方式，导入科目段和导入 ccid，如果允许动态插入，导入科目段时会自动创建 ccid，并在 gl_code_combination 表中创建一条记录。
11、定义任何在总账中未定义但在源系统中使用的账户。

1.4、从源系统中导入数据

日记账导入程序从 GL_interface 表中接受数据。对于非 Oracle 应用，你必须从源系统中导入数据到 GL_interface 接口表。你的导入程序必须转换源系统的数据为日记账导入程序能读取的数据。这样，日记账导入程序可以把 GL_interface 的数据转换成日记账分录（总账凭证）。

1.5、GL_interface 总账接口表

日记账导入程序从 GL_interface 接口表中接收来自源系统的会计数据。当日记账导入程序接收这些数据时，验证并转换这些数据为总账应用的日记账分录。日记账分录（总账凭证）的来源信息被存储在接口表的 USER_JE_SOURCE_NAME 列。

1.5.1、账户字段

你有两种方法在 GL_interface 接口表中指定账户信息：账户段值 segment1-segment30，账户组合 ID code_combination_id。

segment1-segment30	日记账导入程序不允许空的段值。段值必须在总账中已经定义。01 和 1 是不同的，你可以在定义值集的时候指定最大长度（Maximum Size）和补零右端调整（Right-justify Zero-fill）。最大长度是 3，段值是 001，最大长度是 4，段值是 0001。
code_combination_id	GL_CODE_COMBINATIONS.CODE_COMBINATION_ID。你也可以在 GL_interface 表中对列 code_combination_id 赋值而不必对 segment1-segment30 赋值。如果你既对账户段 segment1-segment30 赋值又对 code_combination_id 赋值，日记账导入程序将取账户段值来创建日记账分录。 SELECT code_combination_id FROM gl_code_combinations;

如果你给账户段赋了一个无效的值，日记账导入执行报表将打印错误信息，无效的账户。如果你输入 ccid，并且暂记过账(suspense posting)失效，执行报表将打印错误信息，无

效的 code_combination_id。如果你输入 ccid，并且暂记过账(suspense posting)有效，执行报表将只打印段值分隔符。因此，如果输入 code_combination_id ,建议让暂记过账(suspense posting)失效。

1.5.2、必输字段

status	输入值‘NEW’。
ledger_id	gl_ledgers.ledger_id。在会计科目设置管理器(Accounting Setup Manager)中定义帐薄。 SELECT ledger_id, NAME FROM gl_ledgers;
user_je_source_name	gl_je_sources.user_je_source_name。在来源（Journal Sources）中定义日记账来源。如果有 Import Using Key 选项，也可以用 gl_je_sources.je_source_key 代替。 SELECT user_je_source_name FROM gl_je_sources_tl
user_je_category_name	gl_je_categories.user_je_category_name。在类别（Journal Categories）中定义日记账分类。如果有 Import Using Key 选项，也可以用 gl_je_categories.user_je_category_name 代替。 SELECT user_je_category_name FROM gl_je_categories_tl
accounting_date	总账将自动将日记账批赋给包含此会计日期的非调整会计期。
currency_code	fnd_currencies.currency_code。在币种（Currencies）中定义币种。 SELECT currency_code FROM fnd_currencies
date_created	创建日期
created_by	创建人
actual_flag	‘A’ 代表 actual amounts，实际数量， ‘B’代表 budget amounts，预算数量， ‘E’代表 encumbrance amounts 保留数量。
entered_dr	借方金额。你可以在一行上对 entered_dr 和 entered_cr 赋值。
entered_cr	贷方金额。

1.5.3、条件必输字段

encumbrance_type_id	gl_encumbrance_types.encumbrance_tupe_id。如果 actual_flag 的值是‘E’，则此字段是必输字段。在保留款类型（Encumbrance Types）中定义保留类型。 SELECT encumbrance_type_id, encumbrance_type FROM gl_encumbrance_types WHERE enabled_flag = 'Y';
budget_version_id	gl_budget_versions.budget_version_id。如果 actual_flag 的值是‘B’，则此字段是必输字段。在定义预算（Define Budget）中定义预算版本。 SELECT budget_version_id, budget_name FROM gl_budget_versions WHERE status IN ('C', 'O');
period_name	为了导入实际日记账和保留日记账，要指定期间名和此期间的一个会计日期，日记账导入程序会把日记账导入调整和非调整期间。为了导入预算日记账，要为预算事务输入一个期间名和会计日期，会计日期是日记账的生效日期；预算数量的会计日期会被忽略，用会计期间代替。导入预算数据时，期间名是必输字段，并且要与打开的预算财年相关。日记账导入程序会把预算日记账导入调整和非调整期间。

1.5.4、币种转换字段

有两种方式对实际输入币种数据（actual entered currency data）赋值。你可以指定输入数量（entered amount）和转换率类型（conversion rate type）和转换率日期（conversion date），总账应用自动为你计算转换数量（converted amount）。或者，你可以直接指定输入数量（entered amount）和转换数量（converted amount）而不指定转换率，转换率类型和转换率日期。

如果是保留款（encumbrance）和预算（budget）输入币种数据（entered currency data），不要对以下字段赋值。

系统计算转换数量：

user_currency_conversion_type	gl_daily_conversion_types.user_currency_conversion_type。实际输入币种事务（actual entered currency transaction）对此字段赋值。可以接受

	的值是，‘User’，‘Spot’，‘Corporate’，或者在折换率类型（Conversion Rate Types）中定义的其他类型。如果转换类型是‘User’，币种转换率 currency_conversion_rate 必输。如果是其它的转换类型，币种转换时间 currency_conversion_date 必输。
currency_conversion_date	实际输入币种事务（actual entered currency transaction）对此字段赋值。如果转换类型不是‘User’，此字段必输。如果转换类型是‘User’，此字段的默认值是会计日期（accounting date）。
currency_conversion_rate	实际输入币种事务（actual entered currency transaction）对此字段赋值。如果转换类型是‘User’，此字段必输。不过不是‘User’，不要对此字段赋值。

用户输入转换数量：

accounted_dr	为实际输入币种事务（actual entered currency transaction）输入一个转换的借记数量。可以在一行上对 accounted_dr 和 account_cr 赋值。如果你对 accounted_dr 赋值，你必须首先对 entered_dr 赋值。
accounted_cr	

1.5.5、可选字段

你可以给 GL_interface 表中的很多可选字段赋值，这些字段可以最大限度的控制日记账导入程序分组（group）日记账分录行。如果不对这些字段赋值，日记账导入将自动赋缺省值。

REFERENCE1	Batch Name 批名 默认值：Take the first 85 characters of:(Source) (Balance type- A, B, or E) (Group ID) then add a space followed by the concurrent request ID
REFERENCE2	Batch Description 批描述 默认值：Journal Import (Source) (Request Id)
REFERENCE4	Journal entry name 日记账分录名 默认值：(Category Name) (Currency) (Encumbrance Type ID, if applicable) (Currency Conversion Rate, if applicable) (Currency Conversion Date, if applicable) (Originating

	Balancing Segment Value), chopped to the first 100 characters
REFERENCE5	Journal entry description 日记账分录描述 默认值：Journal Import - Concurrent Request ID
REFERENCE6	Journal entry reference 日记账分录参考 默认值：Journal Import Created
REFERENCE7	Journal entry reversal flag 冲销标记 默认值：NO
REFERENCE8	Journal entry reversal period 冲销期间 如果 reference7 的值是 YES，此字段必输
REFERENCE9	Journal reversal method 冲销方法 Enter Yes to use the **change sign** method, No to use the **switch debit/credit** method
REFERENCE10	Journal entry line description 日记账分录行描述 默认值：Journal Import Created
REFERENCE21 through REFERENCE30	map into columns REFERENCE_1 through REFERENCE_10,respectively, of the GL_JE_LINES table
GROUP_ID	输入唯一的组号来区分导入数据，为每一个请求指定一个唯一的组号可以并行运行日记账导入程序
STAT_AMOUNT	与日记账分录行数据相关的统计数量。在 Statistal Units of Measure 定义统计单位
USSGL_TRANSACTION_CODE	
ATTRIBUTE1 through ATTRIBUTE 10	‘Journals - Journal Entry Line 日记帐 -日记帐分录行’说明性弹性域
ATTRIBUTE11 through ATTRIBUTE 20	‘Journals - Captured Information 日记帐 -捕获信息’说明性弹性域
CONTEXT	描述性弹性域‘Journals - Journal Entry Line 日记帐 -日记帐分录行’的上下文结构列
CONTEXT2	输入‘YES’来标识增值税 Value Added Tax 描述性弹性域结构列。输入‘NO’表明日记账分录行是一个非税项，4 个增值税说明性弹性域字段必须为 NULL。增值税项必输字段

CONTEXT3	为"日记帐 -捕获信息"说明性弹性域输入上下文结构列（自然帐户）。
INVOICE_DATE	发票日期。增值税项必输字段
INVOICE_AMOUNT	发票金额。增值税项必输字段
TAX_CODE	税码。增值税项必输字段
REFERENCE_DATE	
JGZZ_RECON_REF	
AVERAGE_JOURNAL_FLAG	
ORIGINATING_BAL_SEG_VAL	
INVOICE_IDENTIFIER	发票标识符。增值税项必输字段
LEDGER_ID	输入合适的 ledger id 来导入 11i 之前版本使用 set of books ID 的应用的数据。 SELECT sob.set_of_books_id FROM gl_sets_of_books sob

1.5.6、留空字段

GL_interface 中的一些字段必须设置为 null，这些字段在内部处理中被使用或在当前版本中还未启用。

REFERENCE3	
REFERENCE11 through REFERENCE20	
TRANSACTION_DATE	
JE_BATCH_ID	
JE_HEADER_ID	
JE_LINE_NUM	
CHART_OF_ACCOUNTS_ID	
FUNCTIONAL_CURRENCY_CODE	
DATE_CREATED_IN_GL	
WARNING_CODE	
STATUS_DESCRIPTION	
DESC_FLEX_ERROR_MESSAGE	
REQUEST_ID	
SUBLEDGER_DOC_SEQUENCE_ID	
SUBLEDGER_DOC_SEQUENCE_VALUE	
GL_SL_LINK_ID	
GL_SL_LINK_ID	
GL_SL_LINK_TABLE	
BALANCING_SEGMENT_VALUE	
MANAGEMENT_SEGMENT_VALUE	
FUNDS_RESERVED_FLAG	

1.6、最简例子

```
DECLARE
  l_iface_rec       gl_interface%ROWTYPE;
  l_request_id      NUMBER;
  l_interface_run_id NUMBER;
  l_phase           VARCHAR2(80);
  l_status          VARCHAR2(80);
  l_dev_phase       VARCHAR2(80);
  l_dev_status      VARCHAR2(80);
  l_message         VARCHAR2(2000);
  l_wait_status     BOOLEAN;
BEGIN
  --初始化
  fnd_global.apps_initialize(user_id     => 2175,
                         resp_id     => 50699,
                         resp_appl_id => 101);
  --账户信息
  l_iface_rec.code_combination_id := 1002;
  --必输字段
  l_iface_rec.status              := 'NEW';
  l_iface_rec.ledger_id           := 2021; --R12帐套
  l_iface_rec.set_of_books_id      := 2021; --11i帐套
  l_iface_rec.user_je_source_name   := '人工';
  l_iface_rec.user_je_category_name := '记账凭证';
  l_iface_rec.accounting_date      := SYSDATE;
  l_iface_rec.currency_code        := 'CNY';
  l_iface_rec.date_created         := SYSDATE;
  l_iface_rec.created_by           := fnd_global.user_id;
  l_iface_rec.actual_flag          := 'A';
  l_iface_rec.entered_dr           := 1000;
  l_iface_rec.entered_cr           := 1000;
  INSERT INTO gl_interface VALUES l_iface_rec;
  --接口控制
  SELECT gl_interface_control_s.nextval INTO l_interface_run_id FROM
dual;
  INSERT INTO gl_interface_control
    (je_source_name,
     status,
     interface_run_id,
     group_id,
     set_of_books_id,
     packet_id,
     interface_table_name,
```

```
    processed_table_code,
    request_id)
  VALUES
    ('Manual', 'S', l_interface_run_id, NULL, 2021, NULL, NULL, NULL,
NULL);
  --提交请求
  l_request_id := fnd_request.submit_request(application => 'SQLGL',
                                     program     => 'GLLEZL',
                                     description => '',
                                     start_time  => SYSDATE,
                                     sub_request => FALSE,
                                     argument1   =>
l_interface_run_id, --p_interface_run_id
                                     argument2   => 2021, --
p_set_of_books_id
                                     argument3   => 'N',
                                     argument4   => NULL,
                                     argument5   => NULL,
                                     argument6   => 'N',
                                     argument7   => 'O');
  dbms_output.put_line('l_request_id:' || l_request_id);
  COMMIT;
  IF l_request_id > 0 THEN
    --请求等待
    l_wait_status := fnd_concurrent.wait_for_request(request_id =>
l_request_id,
                                            INTERVAL   => 10,
                                            max_wait   => 0,
                                            phase      => l_phase,
                                            status     => l_status,
                                            dev_phase  =>
l_dev_phase,
                                            dev_status =>
l_dev_status,
                                            message    => l_message);
    dbms_output.put_line('l_status:' || l_status);
  END IF;
END;
```

1.7、日记账多表导入

略。

2、应付发票接口导入

2.1、 应付接口表

应付接口表存储发票和供应商信息，导入到应付基表。
应付发票接口导入程序基于应付接口表的发票记录创建应付发票。导入程序创建发票之后，可以在发票工作台查看，修改和验证这些发票。
发票数据包括来自供应商的 EDI 和 XML 发票，在快速发票窗口输入的发票记录，sql*loader 导入的发票，来自财产管理员的租赁发票，贷记卡交易数据，供应商通过在线供应商门户输入并提交的发票。
供应商接口导入，供应商地点接口导入和供应商地点联系人接口导入处理供应商信息并装载到相应表中。

2.2、理解发票接口表

Oracle 电子商务门户，贷记卡发票接口汇总，财产管理员，资产出口租赁支付到应付流程，Oracle XML 门户和 SQL*Loader 通过快速发票窗口装载发票信息到接口表 ap_invoices_interface 和 ap_invoice_lines_interface 中。

应付接口程序会验证每条记录，如果记录是有效的，接口程序会创建一条应付发票，这条发票包含分配信息和基于发票头和行信息的计划支付信息。
应付发票接口行表 ap_invoice_lines_interface 中的每条记录可以创建一条或多条分配。注意，是一行可以创建多行分配。例如，如果你在这个表中输入一行税行，并且按比例分配在三个商品行上（prorate it across three Item lines），接口导入程序处理时，系统会基于这一行创建三行税发票分配。
接口表的一些字段用来分类和存储指定的发票信息。例如，发票来源信息被存储在表 ap_invoices_interface 的 source 字段上。

接口表的字段可能包含以下属性：

非空字段（not null）：	必须赋值才能保存，否则数据不能被保存到接口表。
必输字段（required）：	必须输入一个有效值，否则这条记录会被拒绝。
置空字段（null）：	这些字段必须留空，否则导入程序会失败。例如，status 和 request_id 字　段，在导入时，程序会更新这些字段的值，对这些字段应付接口不支持任何导入值。
条件必输字段（conditionally required）：	条件必输字段依赖其他字段，当所依赖的字段有值时，此字段必输。
可选字段（optional）：	顾名思义，可选赋值。
内部 ID 字段（internal ID）：	这些列的值，应付只在内部使用，用户无法看到，只能通过表查询到。这些列与接口表中的其它列相关，当相关列有值时，这些字段不需要赋值，例如，terms_id，当相关字段 terms_name 有值时，程序会在执行时给 terms_id 赋值。如果 terms_id 和 terms_name 都赋值，但不匹配，这条记录会被程序拒绝。直接赋 id，处理速度快。

2.3、应付接口表描述

2.3.1、AP_INVOICES_INTERFACE

invoice_id	必输字段，主键。值来自序列 AP_INVOICES_INTERFACE_S 。 发票的唯一标识符，要给属于这张发票的行接口表 ap_invoice_lines_interface 赋同样的值用来表示属于同一张发票。

invoice_num	条件必输字段。发票编号。如果导入的供应商有多张发票，则是必输字段。如果没有赋值，应付会用导入时的系统日期作为默认值。 对一个供应商，发票编号必须唯一，如果赋了一个重复的值，接口导入程序将不会创建发票。
invoice_type_lookup_code	可选字段。发票类型代码。 发票类型：贷记（Credit）或标准（Standard）。如果没有赋值，系统会根据发票金额 Invoice_amount 来赋值。如果发票金额小于 0，发票类型的默认值是贷记（Credit），如果发票金额大于等于 0，发票类型的默认值是标准（Standard）。使用服务采购（Service Procurement）时，可以导入预付款。 值必须是 Credit 或 Standard，发票类型必须与发票金额匹配。 目标字段：ap_invoices_all.invoice_type_lookup_code。
invoice_date	可选字段。发票日期。必须在税码的有效期内，不然报错：ZX_TAX_RATE_ID_CODE_MISSING 请填写 tax_rate_id 或 tax_rate_code 以创建有效税行。 如果没有赋值，系统使用接口导入程序提交时的日期作为发票日期。根据系统设置，应付有可能用发票日期作为付款条件日期（terms date）和总账日期（GL date）。如果总账日期基准（GL Date Basis）是发票日期，发票日期必须在打开的或未来的会计期内。 目标字段：ap_invoices_all.invoice_date
po_number	可选字段。采购订单编号。验证 po_headers.segment1。匹配发票的采购订单编号。如果对此字段赋值，系统将在发票分配表 ap_invoice_distributions_all 中产生分配信息。如果不使用快速开票（Quick Invoices）并且不在 ap_invoices_interface 指定供应商，此字段将被用来产生 ap_invoices_all.vendor_id；如果不指定供应商地点，此字段将被用来产生 ap_invoices_all.vendor_site_id。 匹配发票到采购订单，如果在发票行级别 ap_invoice_lines_interface.po_number 赋了值，不必在发票头级别对此字段赋值。 验证：必须是有效的，批准的，打开的采购订单，还必须是同一个供应商。此采购订单必须没有被最终必配（final matched）。如果发票来源是电子商务网关（e-Commerce Gateway），应付接口程序将只导入满足数量和价格允差（tolerance）的发票，数量允差和价格允差在发票允差（Invoices Tolerances）界面设置。其

	它来源的发票，允差将在发票验证（Invoice Validation）时 check。
vendor_id	条件必输字段。验证 po_vendors.vendor_id。 vendor_id，vendor_num，vendor_name 必输一个，除非匹配一个 PO 采购订单。 通过对 vendor_id，vendor_num，vendor_site_id，po_number 赋值，可以识别一个供应商。 目标字段：ap_invoices_all.vendor_id。
vendor_num	供应商编号。验证 po_vendors.segment1。 目标字段：没有，将转换成 ap_invoices_all.vendor_id。
vendor_name	供应商名称。验证 po_vendors.vendor_name 目标字段：没有，将转换成 ap_invoices_all.vendor_id。
vendor_site_id	必输字段。验证 po_vendor_sites.vendor_site_id。 如果没有提供 vendor_site_code 或 vendor_site_id 来标识付款地点（pay site），导入程序将按以下顺序查找有效的供应商付款地点。 1、供应商的主付款地点（primary pay site） 2、供应商唯一存在的（single existing pay site）付款地点 3、发票头匹配的 PO 编号派生的付款地点。 导入程序将拒绝导入无法识别有效供应商的发票。 地点必须是一个付款地点（pay site）。 目标字段：ap_invoices_all.vendor_site_id。
vendor_site_code	供应商地点代码。验证 po_vendors.vendor_site_code 目标字段：没有，将转换成 ap_invoices_all.vendor_site_id。
invoice_amount	必输字段。发票金额。不要超过发票币种的精确度，一般四舍五入，保留两位小数，不然会报错。 必须等于行接口表 ap_invoice_lines_interface 中属于同一张发票的的金额之和。金额必须与发票类型相匹配，标准类型的发票，金额必须大于等于 0。贷记类型的发票，金额必须小于 0。 目标字段：ap_invoices_all.invoice_amount。
invoice_currency_code	发票币种。验证 fnd_currencies.currency_code。 如果没有赋值，导入时，将取供应商地点的币种。 如果发票币种和支付币种关联到固定汇率货币，payment_cross_rate_type，payment_cross_rate 和 payment_cross_rate_date 将不被导入到 ap_invoices_all。 目标字段：ap_invoices_all.invoice_currency_code

exchange_rate	条件必输字段。外币发票的汇率。验证 gl_daily_conversion_types.conversion_type。 如果应付选项中的需要外币输入（Require Exchange Rate Entry）参数被选中，必须提供汇率或足够的能获取汇率的信息。 如果汇率类型（exchange_rate_type）是用户（User），汇率字段（exchange_rate）必输；或者应付选项参数计算用户汇率（Calculate User Exchange Rate）被选中，给 no_xrate_base_amount 来代替。 如果汇率类型即期汇率（spot）或公司汇率（Corporate），导入程序将自动提供汇率，此字段应该留空。 如果发票币种有一个对功能货币的固定汇率，导入程序将用赋的值覆盖这个固定汇率。 目标字段：ap_invoices_all.exchange_rate
exchange_rate_type	汇率类型。条件必输字段。
exchange_date	汇率日期。条件必输字段。
terms_id	必输字段。支付条件。验证 ap_terms.terms_id。 terms_id 或 terms_name 必输一个。 支付条件（payment terms）在 Payment Terms 界面中维护。 导入程序将按以下顺序查找支付条件按： 1、发票头（terms_id or terms_name）。 2、发票头中的采购订单的支付条件。 3、发票行一行或多行直接匹配的采购订单的支付条件，或通过接受单间接匹配的采购订单的支付条件（如果没有多于一个支付条件）。 4、供应商地点。 如果还是没有找到值，导入程序将拒绝这个发票记录。 目标字段：ap_invoices_all.terms_id。
terms_name	目标字段：没有。将转换成 ap_invoices_all.terms_id。
description	可选字段。如果匹配采购订单，此字段留空，系统将会把采购订单行的物料描述赋给此字段。 基表字段：ap_invoices_all.description
awt_group_id	内部 ID 字段。验证 ap_awt_groups.awt_group_id。 automatic withholding tax，自动代扣税。
awt_group_name	可选字段
last_update_date	可选字段。最后更新日期。

last_updated_by	可选字段。最后更新人。
last_update_login	可选字段。最后登录用户
creation_date	可选字段。创建日期。
created_by	可选字段。创建人。
attribute_category	可选字段。
attribute[1-15]	可选字段。
global_attribute_category	可选字段。
global_attribute_category	可选字段。
global_attribute[1-20]	可选字段。
status	N/A。留空字段。
source	必输字段。发票数据来源，可以在应付查询码（lookup）中定义。 基表字段：ap_invoices_all.source。
group_id	可选字段。可以并行处理同一个 group 的数据。
request_d	N/A
payment_cross_rate_type	N/A。交叉汇率类型。
payment_cross_rate_date	N/A。交叉汇率日期。
payment_cross_rate	N/A。交叉汇率。
payment_currency_code	N/A.支付币种
workflow_flag	N/A.
doc_category_code	如果序列号配置文件（sequential numbering）的值是部分 partial 或 always
voucher_num	凭证编号。N/A。
payment_method_lookup_code	支付方法查询码。验证 ap_lookup_codes
pay_group_lookup_code	支付组查询码。
goods_received_date	货物接受日期
invoice_received_date	发票接受日期
gl_date	总帐日期

accts_pay_code_combination_id	负债账户。导入程序将把此字段作为发票分配的默认账户。如果没有赋值，系统将取供应商地点账户。
ussgl_transation_code	
org_id	业务实体
amount_applicable_to_discount	可选字段。可折扣数量
prepay_num	验证 ap_invoices_all.invoice_num。预付款编号
prepay_dist_num	预付款分配编号
prepay_apply_amount	
prepay_gl_date	
invoice_includes_prepay_flag	
no_xrate_base_amout	仅当应付选项‘Calculate User Exchange Rate’生效时使用。
vendor_email_address	拒绝 XML 发票时的供应商邮件地址
terms_date	付款条件日期
requester_id	

2.3.2、AP_INVOICE_LINES_INTERFACE

invoice_id	必输字段。验证 ap_invoices_interface
invoice_line_id	必输字段。值来自序列 ap_invoice_lines_interface_s。主键。
line_number	必输字段。行号
line_type_lookup_code	必输字段。行类型查询码。
line_group_number	可选字段。行组号。如果要按比例分配费用给指定的若干行，给指定的若干物料行赋一个行组号，费用行也赋同值，则费用行的费用会按比例非配给指定的物料行。例如，如果要按比例分配税行给两个物料行，给税行和两个物料行赋一个相同的行组号。
amount	必输字段。行发票金额。
accounting_date	会计日期
description	可选字段。

prorate_across_flag	可选字段。行类型不能是物料行。如果值设为‘Y’，并且为非物料行（税行 tax，杂项 Miscellaneous，运费 Freight），应付会按比例分配给所有的拥有同一个行组号的物料行。 如果不分配，发票分配界面打不开。
tax_code	税码。验证 ap_tax_codes.name
final_match_flag	N/A。
po_header_id	验证 po_headers.po_header_id
po_number	可选字段。采购字段编号。验证 po_headers.segment1
po_line_id	验证 po_lines.po_line_id。
po_line_number	可选字段。采购订单行号。验证 po_lines.po_line_num。
po_line_location_id	验证 po_line_locations.line_location_id。
po_shipment_num	可选字段。采购订单发运号。验证 po_line_locations.shipment_num。
po_distribution_id	验证 po_distributions.po_distribution_id。
po_distribution_num	可选字段。采购订单分配号。验证 po_distributions.po_distribution_num。
inventory_item_id	可选字段。验证 mtl_system_items.inventory_item_id。
item_descrition	可选字段。
quantity_invoiced	可选字段。
ship_to_location_code	N/A
unit_price	可选字段。
distribution_set_id	验证 ap_distribution_sets.distribution_set_id
distribution_set_name	可选字段
dist_code_concatenated	可选字段
dis_code_combination_id	验证 gl_code_combinations.code_combination_id
awt_group_id	验证 ap_awt_groups.group_id
awt_group_name	可选字段
last_updated_by	可选字段
last_update_date	可选字段

last_update_login	可选字段
created_by	可选字段
creation_date	可选字段
attribute_category	
attribute[1-15]	
global_attribute_category	
global_attribute[1-20]	
po_release_id	可选字段。po_releases_all
account_segment	可选字段。科目段
balancing_segment	可选字段。平衡段
cost_center_segment	可选字段。成本中心段
project_id	项目。pa_projects
task_id	任务。pa_tasks
expenditure_type	支出类型。pa_expenditure_types
expenditure_item_date	支出项日期。pa_expenditure_types
expenditure_organization_id	支出项组织。 per_organization_units.expenditure_organization_id
pa_addition_flag	Projects
pa_quantity	Projects
ussgl_transaction_code	General Ledger
stat_amount	N/A
type_1099	N/A
income_tax_region	N/A
assets_tracking_flag	N/A
tax_code_id	N/A
price_correction_flag	N/A
org_id	业务实体
receipt_number	接收单号。rcv_shipment_headers. receipt_num

receipt_line_num	接受单行号。N/A
match_option	N/A
packing_slip	N/A
rcv_transation_id	接受事务处理。N/A
pa_cc_ar_invoice_id	相关的公司间应收发票
reference_1	Projects
reference_2	Projects
pa_cc_processed_code	N/A
credit_card_trx_id	信用卡事务处理
award_id	grants requirement to store award
vendor_item_num	可选字段。po_lines_all.vendor_product_num。采购订单行上的供应商的产品号。
taxable_flag	如果值为‘Y’，表示这行应该纳税。
price_correct_inv_num	N/A
external_doc_line_ref	
serial_number	物料的序列号
manufacturer	制造商名称
model_number	
warranty_number	
deferred_acctg_flag	
def_acctg_start_date	
def_acctg_end_date	
def_acctg_number_of_periods	
def_acctg_period_type	
unit_of_meas_lookup_code	mtl_units_of_measure.units_of_measure
price_correct_inv_line_num	
asset_book_type_code	
asset_category_id	

requester_id	
requester_first_name	
requester_last_name	
requester_employee_num	
application_id	应用标识符
product_table	产品来源表
reference_key[1-5]	
purchasing_category	物料分类组合字段
purchasing_category_id	物料分类唯一 ID
cost_factor_id	
cost_factor_name	
control_amount	
assessable_values	
default_dist_ccid	TAX Driver
primary_intended_use	
ship_to_location_id	
product_type	
product_category	
product_fisc_classification	
user_defined_fisc_class	
trx_business_category	
tax_regime_code	
tax	
tax_jurisdiction_code	
tax_status_code	
tax_rate_id	税率标识符，发票日期要在税码的有效期内
tax_rate_code	在不同的日期范围内，一个税码可能有不同的税率，所以日期限制特别重要。

tax_rate	
incl_in_taxable_line_flag	
source_application_id	
source_entity_code	
source_event_class_code	
source_trx_id	
source_line_id	
source_trx_level_type	
tax_classification_code	

2.4、最简例子

```
DECLARE
  rec_ap_invoice      ap_invoices_interface%ROWTYPE;
  rec_ap_invoice_line ap_invoice_lines_interface%ROWTYPE;
  rec_tax             ap_invoice_lines_interface%ROWTYPE;
  l_request_id        NUMBER;
  l_phase             VARCHAR2(100);
  l_status            VARCHAR2(100);
  l_dev_phase         VARCHAR2(100);
  l_dev_status        VARCHAR2(100);
  l_message           VARCHAR2(500);
BEGIN
  fnd_global.apps_initialize(user_id      => 2175,
                             resp_id      => 50707,
                             resp_appl_id => 200);
  --头
  SELECT ap_invoices_interface_s.nextval
    INTO rec_ap_invoice.invoice_id
    FROM dual;
  rec_ap_invoice.org_id              := 82;
  rec_ap_invoice.vendor_id           := 6157;
  rec_ap_invoice.vendor_site_id      := 5300;
  rec_ap_invoice.gl_date             := SYSDATE;
  rec_ap_invoice.invoice_amount      := 12.34;
  rec_ap_invoice.invoice_currency_code := 'CNY';
  rec_ap_invoice.payment_method_code   := 'JZ_CHECK';
```

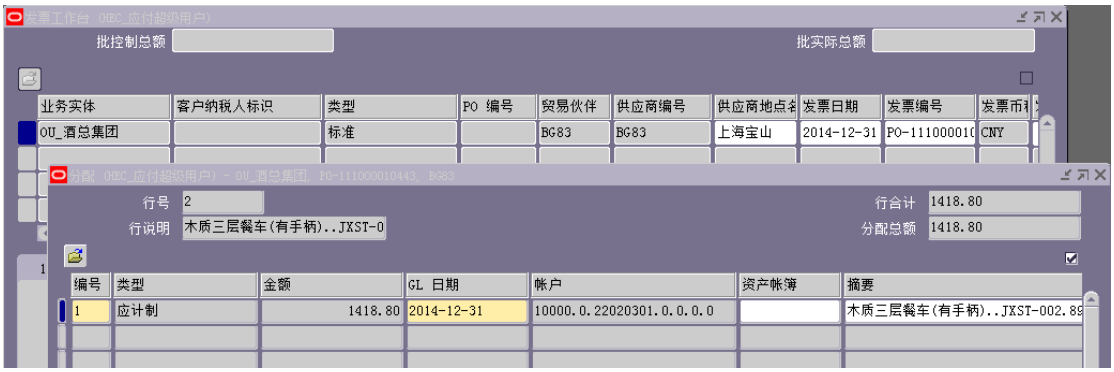
```
  rec_ap_invoice.terms_id            := 10002;
  rec_ap_invoice.terms_date          := SYSDATE;
  rec_ap_invoice.source              := 'MANUAL INVOICE ENTRY';
  rec_ap_invoice.group_id            := 'AP_wangjingyuan';
  rec_ap_invoice.invoice_num         := to_char(SYSDATE,
                                        'yyyymmddhh24miss');
  INSERT INTO ap_invoices_interface VALUES rec_ap_invoice;
  --税
  SELECT ap_invoice_lines_interface_s.nextval
    INTO rec_tax.invoice_line_id
    FROM dual;
  rec_tax.invoice_id          := rec_ap_invoice.invoice_id;
  rec_tax.org_id              := rec_ap_invoice.org_id;
  rec_tax.line_type_lookup_code := 'TAX';
  rec_tax.line_number         := 1;
  rec_tax.amount              := 0;
  rec_tax.tax_rate_code       := 'CN_VAT_0';
  rec_tax.prorate_across_flag   := 'Y'; --按比例
  INSERT INTO ap_invoice_lines_interface VALUES rec_tax;
  --行
  SELECT ap_invoice_lines_interface_s.nextval
    INTO rec_ap_invoice_line.invoice_line_id
    FROM dual;
  rec_ap_invoice_line.org_id              := 82;
  rec_ap_invoice_line.invoice_id          :=
rec_ap_invoice.invoice_id;
  rec_ap_invoice_line.accounting_date       := SYSDATE;
  rec_ap_invoice_line.line_number           := 2;
  rec_ap_invoice_line.line_type_lookup_code  := 'ITEM';
  rec_ap_invoice_line.amount                := 12.45;
  rec_ap_invoice_line.dist_code_concatenated :=
'10000.0.66012901.0.0.0.0';
  /*rec_ap_invoice_line.po_number            := 111000000284;
  rec_ap_invoice_line.po_line_number        := 1;
  rec_ap_invoice_line.po_shipment_num       := 1;
  rec_ap_invoice_line.po_distribution_num    := 1;
  rec_ap_invoice_line.receipt_number        := 271000000184;
  rec_ap_invoice_line.receipt_line_number    := 1;*/
  INSERT INTO ap_invoice_lines_interface VALUES rec_ap_invoice_line;

  l_request_id := fnd_request.submit_request(application => 'SQLAP',
--应付帐款
                                        program     => 'APXIIMPT', --应付
款管理系统开放接口导入
```

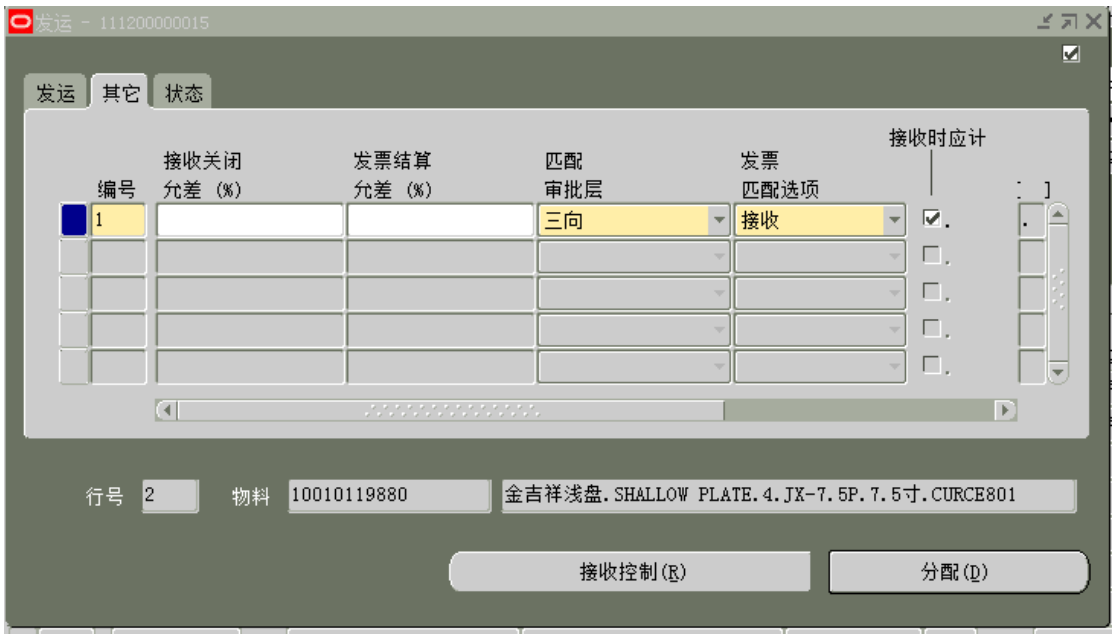
```
                                        description => '',
                                        start_time  => '',
                                        sub_request => FALSE,
                                        argument1   => 82, --业务实体
                                        argument2   => 'MANUAL INVOICE
ENTRY', --来源 必输
                                        argument3   =>
rec_ap_invoice.group_id, --组
                                        argument4   => 'N/A', --批名 必输
                                        argument5   => '', --暂挂名
                                        argument6   => '', --暂挂原因
                                        argument7   => '', --GL 日期
                                        argument8   => 'Y', --清除
                                        argument9   => 'N', --跟踪开关
                                        argument10  => 'N', --调试开关
                                        argument11  => 'N', --汇总报告以审
核报表
                                        argument12  => 1000, --提交批大小
                                        argument13  => 2175, --用户标识
                                        argument14  => 1642144 --登录标识
                                        );
  COMMIT;
  dbms_output.put_line('l_request_id:' || l_request_id);
  IF l_request_id > 0 THEN
    IF fnd_concurrent.wait_for_request(request_id => l_request_id,
                                       INTERVAL   => 3,
                                       max_wait   => 0,
                                       phase      => l_phase,
                                       status     => l_status,
                                       dev_phase  => l_dev_phase,
                                       dev_status => l_dev_status,
                                       message    => l_message) THEN
      dbms_output.put_line('l_dev_status:' || l_dev_status);
    END IF;
  END IF;
END;
```

2.5、易出错误

- 行组号,税行按比例分配，行组号作为关联,此字段如果不为空，必须有税行否则报错:INSUFFICIENT PA INFO
- expenditure_item_date 不为空，expenditure_type，expenditure_organization_id 为空，也会报错：PA 信息不足，强制项目列为空
- 在非税行上找到纳税数据，不能在非税行上提供与纳税相关的信息
- 标准请求参数 来源 不能随意赋值，要赋正确的值，报错： 'cf_1formula': 发生了严重的 PL/SQL 错误
- 税行一般只有一个，经过测试即便给了多行税行，最后还是只显示最后一行，有待学习这部分内容
- 发票日期要在税码的有效期之内，不然会报错：ZX_TAX_RATE_ID_CODE_MISSING 请填写 tax_rate_id 或 tax_rate_code 以创建有效税行。
- 在匹配 PO 时，报一下错误：接收时应计-行与标记为“接收时应计”的发运相匹配，并且已指定了重叠段。原因是采购选项的费用 AP 应计账户与导入数据中给的分配账户不一致，导致冲突。与采购/设置/组织/采购选项的接受会计有关。

解决方法：导入数据中的分配账户去掉，取接受会计中默认的费用 AP 应计账户。





- 自动生成 ccid

```
DECLARE
  l_bool BOOLEAN;
BEGIN
  l_bool := fnd_flex_keyval.validate_segs(operation          => 'CREATE_COMBINATION',
                                          appl_short_name    => 'SQLGL',
                                          key_flex_code      => 'GL#',
                                          structure_number => 50348,
                                          concat_segments    =>
'10000.0.19990107.0.0.0.0');
  IF l_bool THEN
    dbms_output.put_line(fnd_flex_keyval.combination_id);
  END IF;
END;
```

- 调用其它功能

```
app_navigate.execute('FUNCTION_NAME'
                                        ,'Y'
                                        ,'Y'
                                        ,'P_ORG_ID = "' || :LINE.SHIP_FROM_ORG_ID || '" ' || '
P_INVENTORY_ITEM_ID= "' || :LINE.INVENTORY_ITEM_ID || '" ');
```

3、销售订单 API 导入

3.1、订单导入请求

如果要优化接口数据处理，在提交订单导入并发程序之前，要先手动提交订单导入统计数据并发程序。订单导入和 Oracle 的其它应用一样，用基于成本的优化器来优化查询。订单导入统计数据并发程序将收集优化器所需要的统计信息。
订单导入程序是订单管理的一个开放接口，由开放接口表和一套 API 组成。订单导入可以导入新订单，修改和完成销售订单或退货单，或来自其它系统的订单。也可以导入取消的，关闭的和已登记的订单，这些订单可以是任何来源，比如电子商务网关所处理的 EDI 事务处理或内部请求生成的内部采购订单。
订单导入有以下特点：验证，缺省值，处理约束检查，申请和释放订单暂挂，计划发运，最终插入，更新，删除订单到基表。订单管理系统将检查所有的数据，转换到基表中的行，保留，支付，价格调整，销售信贷。
如果没有成功导入，可以在更正窗口修改接口数据，然后再次带入，直至成功。

3.1.1、订单管理系统参数

下面的系统参数会影响订单导入，
项目验证组织（Item Validation Organization）：决定用哪个库存组织验证物料和物料清单结构。翻译成物料验证组织才对。因为一个业务实体下会有多个库存组织，取行物料时及物料描述时，要从这个库存组织取。

```
SELECT parameter_value,
                org_id,
                oe_sys_parameters_util.get_value(spd.value_set_id, sp.parameter_value)
  FROM oe_sys_parameters_all sp, oe_sys_parameter_def_vl spd
 WHERE sp.parameter_code = spd.parameter_code
   AND spd.parameter_code = 'MASTER_ORGANIZATION_ID';
```

保留时间栏（Reservation Time Fence）：
在计划时自动保留。时间（天数）。如果计划日期处于保留时间栏内，则系统将在计划时将其保留。

计划处于暂挂状态的行（Schedule Line on Hold）：
此参数确定 Oracle Order Management 的计划程序是否应计划处于暂挂状态的行。

3.1.2、配置文件

OM：Apply Automatic Attachments	决定是否应用基于规则的不需要用户干涉的附件。
OM：AutoSchedule	决定订单行是否自动计划。
OM：Authotized to Overridde ATP	向用户提供覆盖计划结果的权限。
OM：Import Multiple Shipments	决定是否通过订单导入导入多次发运。如果值是 yes，orig_sys_shipment_ref 将和 order_source_id 和 orig_sys_document_ref 和 orig_sys_line_ref 联合决定行记录的唯一性。
OM：Unique Order Source，Orig Sys Document Ref Combination for Each Customer	决定客户信息是否和参考信息一起辨别销售订单。

3.2、处理订单 API

销售订单已经模式化一个订单管理系统拥有的业务对象。销售订单业务对象有以下实体组成：

头	Header
头销售信贷	Header Sales Credits
头价格调整	Header Price Adjustments
头价格属性	Header Pricing Attributes
头调整属性	Header Adjustment Attributes
头调整关联	Header Adjustment Associations
行	Lines
行销售信贷	Line Sales Credits
行价格调整	Line Price Adjustments
行价格属性	Line Pricing Attributes
行调整属性	Line Adjustment Attributes
行调整关联	Line Adjustment Association

行批次序列号	and Line Lot Serial Numbers

订单处理 API 被设计成一个机制，所有的数据增删改操作都以一致的方式执行在销售订单业务对象实体和它们的属性上。其它的活动请求，比如申请暂挂，附件，预订等，也可以通过 API 来处理。API 中的逻辑不仅考虑更新属性也根据属性值的改变调用其它的功能。使用订单处理 API 来操作销售订单业务对象的重要性是非常重要的。必须理解，使用订单处理 API 不仅可以避免重复的业务逻辑也能向前推进业务。

3.2.1、Process Orders Entities and Associated Tables

Entity Name	Table Name
Order Header	OE_ORDER_HEADERS_ALL
Order Price Adjustments	OE_PRICE_ADJUSTMENTS
Order Sales Credits	OE_SALES_CREDITS
Order Line	OE_ORDER_LINES_ALL
Order Pricing Attributes	OE_ORDER_PRICE_ATTRIBS
Order Adjustment Attributes	OE_PRICE_ADJ_ATTRIBS
Order Adjustment Associations	OE_PRICE_ADJ_ASSOCS
Line Sales Credits	OE_SALES_CREDITS
Line Price Adjustments	OE_PRICE_ADJUSTMENTS
Line Pricing Attributes	OE_ORDER_PRICE_ATTRIBS
Line Adjustment Attributes	OE_PRICE_ADJ_ATTRIBS
Line Adjustment Associations	OE_PRICE_ADJ_ASSOCS
Lot Serial Numbers	OE_LOT_SERIAL_NUMBERS

3.2.3、必输字段

Order Header

Order_Number	订单编号
Invoice_to_Org_Id	客户
Price_list	价目表
Tax_Exempt_Flag	免税标志

Salesrep_Id	销售员
Ordered_Date	订单日期
Ship_to_Org	仓库，发货库存组织。regular 或 mixed 订单必输，return 不是必输。
Payment_Term_Id	支付条件。regular 或 mixed 订单必输，return 不是必输。
Conversion_Rate	汇率。如果 Conversion_Type_Code = User，必输。
Conversion_Rate_Date	汇率日期。如果 Conversion_Type_Code = User，必输。
Payment_Amount	支付金额。如果 Payment_Type_Code 不为空，必输。
Check_Number	支票编号。如果 payment_type=check，必输。

Line

Order_Number	订单编号
Sold_to_Org	客户
Invoice_to_Org_Id	发票组织
Price_list	价目表
Tax_Exempt_Flag	免税标志
Salesrep_Id	销售员
Ordered_Date	订单日期
Inventory_Item_Id	物料
Ordered_Quantity	数量
Ordered_Quantity_UOM	数量单位
Ship_to_Org	仓库。发货库存组织。regular 或 mixed 订单必输，return 不是必输。
Payment_Term_Id	支付条件。regular 或 mixed 订单必输，return 不是必输。
Warehouse	仓库。退货单 return 必输。
Schedule Date	计划日期。退货单 return 必输。
Tax_Date	退货单 return 必输。
Tax_Code	税码。如果 Tax_Exempt_Flag = R (Required)，必输。
Service_Start_Date	如果 Service_Coterminate_Flag = Y (yes) and Service_Reference_Type_Code is

	Customer Product，必输。
Service_End_Date	如果 Service_Coterminate_Flag = Y (yes) and Service_Reference_Type_Code is Customer Product，必输。
Service_Duration	如果 Service_Coterminate_Flag = Y (yes) and Service_Reference_Type_Code is Customer Product，必输。
Service_Period	如果 Service_Coterminate_Flag = Y (yes) and Service_Reference_Type_Code is Customer Product，必输。

3.1、创建销售订单

```
DECLARE
   l_api_version_number CONSTANT NUMBER := 1.0;
   l_control_rec                  oe_globals.control_rec_type;
   l_return_status                VARCHAR2(1);
   l_msg_count                      NUMBER;
   l_line_con                     NUMBER := 0;
   x_msg_data                       VARCHAR2(1000);
   l_source_id                    NUMBER := 0;
   l_header_rec                                                    oe_order_pub.header_rec_type    :=
oe_order_pub.g_miss_header_rec; --缺省值重要
   o_header_rec                     oe_order_pub.header_rec_type;
   l_header_val_rec               oe_order_pub.header_val_rec_type;
   l_header_adj_tbl               oe_order_pub.header_adj_tbl_type;
   l_header_adj_val_tbl         oe_order_pub.header_adj_val_tbl_type;
   l_header_price_att_tbl     oe_order_pub.header_price_att_tbl_type;
   l_header_adj_att_tbl         oe_order_pub.header_adj_att_tbl_type;
   l_header_adj_assoc_tbl      oe_order_pub.header_adj_assoc_tbl_type;
   l_header_scredit_tbl         oe_order_pub.header_scredit_tbl_type;
   l_header_scredit_val_tbl oe_order_pub.header_scredit_val_tbl_type;
   l_line_tbl                      oe_order_pub.line_tbl_type := oe_order_pub.g_miss_line_tbl; --缺
省值重要
   o_line_tbl                       oe_order_pub.line_tbl_type;
   l_line_adj_tbl                 oe_order_pub.line_adj_tbl_type;
   l_line_price_att_tbl         oe_order_pub.line_price_att_tbl_type;
   l_line_adj_att_tbl           oe_order_pub.line_adj_att_tbl_type;
   l_line_adj_assoc_tbl         oe_order_pub.line_adj_assoc_tbl_type;
   l_line_scredit_tbl           oe_order_pub.line_scredit_tbl_type;
```

```
  l_lot_serial_tbl               oe_order_pub.lot_serial_tbl_type;
  l_line_val_tbl                  oe_order_pub.line_val_tbl_type;
  l_line_adj_val_tbl             oe_order_pub.line_adj_val_tbl_type;
  l_line_scredit_val_tbl      oe_order_pub.line_scredit_val_tbl_type;
  l_lot_serial_val_tbl oe_order_pub.lot_serial_val_tbl_type;
  l_action_qeq_tbl          oe_order_pub.request_tbl_type;
  x_action_qeq_tbl          oe_order_pub.request_tbl_type;
  v_count                         NUMBER;
  l_customer_id                NUMBER;
  l_cust_acct_site_id      NUMBER;
  l_bill_site_use_id        NUMBER;
  l_ship_site_use_id        NUMBER;
  l_ship_from_org_id        NUMBER;
  l_order_type_id             NUMBER;
  l_price_list_id             NUMBER;
  l_payment_term_id          NUMBER;
  l_primary_salesrep_id NUMBER;
  l_salesrep_id                NUMBER;
  l_inventory_item_id      NUMBER;
  l_po_tax_price               NUMBER;
  l_price_list_price        NUMBER;
  l_msg_data                     VARCHAR2(2000);
  i                                    NUMBER;
  g_org_id                       NUMBER := fnd_profile.value('ORG_ID'); --职责或应用设置的默认
profile 值
BEGIN
  fnd_global.apps_initialize(user_id          => 2190,
                                       resp_id          => 21623, --销售订单所在职责
                                       resp_appl_id => 660); --销售订单所在应用
  mo_global.init('ONT');
  mo_global.set_policy_context('S', 106); --订单头上的业务实体，而不是配置文件 profile 中的
org_id
  oe_msg_pub.initialize;
  oe_debug_pub.initialize;
  --    fnd_request.set_org_id(88);
  l_header_rec.order_source_id               := NULL;
  l_header_rec.orig_sys_document_ref     := NULL;
  l_header_rec.sold_to_org_id                := 5040; --
  l_header_rec.ship_from_org_id              := 106; --
  l_header_rec.salesrep_id                     := 100006040; --
  l_header_rec.booked_flag                     := 'Y'; --
  l_header_rec.flow_status_code              := 'BOOKED'; --
  l_header_rec.booked_date                     := SYSDATE; --
  l_header_rec.pricing_date                   := SYSDATE; --
```

```
  l_header_rec.transactional_curr_code := 'CNY'; --
  l_header_rec.operation                := oe_globals.g_opr_create; --
  l_header_rec.cancelled_flag           := 'N'; --
  l_header_rec.cust_po_number           := NULL;
  l_header_rec.ship_to_org_id           := 1541; --参数  --
  l_header_rec.invoice_to_org_id        := 1540; --参数  --
  l_header_rec.order_category_code      := 'MIXED';
  l_header_rec.ordered_date             := SYSDATE; --
  l_header_rec.order_type_id            := 1180; --参数--
  l_header_rec.org_id                   := 106; --
  l_header_rec.price_list_id            := 85012; --参数  --
  l_header_rec.payment_term_id          := 1000; --参数--
  l_header_rec.shipping_method_code     := '000001_DHL_R_D2D'; --参数
  l_line_tbl.delete;
  i := 1;
  l_line_tbl(i) := oe_order_pub.g_miss_line_rec; --缺省值重要
  l_line_tbl(i).operation := oe_globals.g_opr_create; --
  l_line_tbl(i).booked_flag := 'Y'; --
  l_line_tbl(i).cancelled_flag := 'N'; --
  l_line_tbl(i).inventory_item_id := 1001; --
  l_line_tbl(i).ship_from_org_id := 106;
  l_line_tbl(i).line_category_code := 'ORDER';
  l_line_tbl(i).line_number := i; --
  --l_line_tbl(i).schedule_ship_date := SYSDATE;--计划发运日期
  l_line_tbl(i).order_quantity_uom := 'EA'; --
  l_line_tbl(i).unit_selling_price := 200; --
  l_line_tbl(i).ordered_quantity := 2; --
  l_line_tbl(i).org_id := 106; --
  l_line_tbl(i).line_type_id := 1172;
  l_line_tbl(i).source_type_code := 'EXTERNAL'; --来源类型为内部，行状态为等待发运，来源类型为外部，行状态为已登记
  --l_line_tbl(i).schedule_arrival_date := SYSDATE;
  l_line_tbl(i).calculate_price_flag := 'N'; --加上之后价格不会变化，照本宣科
  l_line_tbl(i).subinventory := NULL;
  --登记销售订单
  l_action_qeq_tbl(1).entity_code := oe_globals.g_entity_header;
  l_action_qeq_tbl(1).request_type := oe_globals.g_book_order;
  --调用  api  生成销售订单
  oe_order_pub.process_order(p_org_id                    => 106, --订单头上的业务实体，而不是配置文件 profile 中的 org_id
                             p_api_version_number        => l_api_version_number,
                             p_init_msg_list             => fnd_api.g_true,
                             p_return_values             => fnd_api.g_false,
                             p_action_commit             => fnd_api.g_false,
```

```
                                    x_return_status                => l_return_status,
                                    x_msg_count                    => l_msg_count,
                                    x_msg_data                     => x_msg_data,
                                    p_header_rec                   => l_header_rec,
                                    p_line_tbl                     => l_line_tbl,
                                    p_line_adj_tbl                 => l_line_adj_tbl,
                                    p_action_request_tbl           => l_action_qeq_tbl,
                                    x_header_rec                   => o_header_rec,
                                    x_header_val_rec               => l_header_val_rec,
                                    x_header_adj_tbl               => l_header_adj_tbl,
                                    x_header_adj_val_tbl           => l_header_adj_val_tbl,
                                    x_header_price_att_tbl         => l_header_price_att_tbl,
                                    x_header_adj_att_tbl           => l_header_adj_att_tbl,
                                    x_header_adj_assoc_tbl         => l_header_adj_assoc_tbl,
                                    x_header_scredit_tbl           => l_header_scredit_tbl,
                                    x_header_scredit_val_tbl       => l_header_scredit_val_tbl,
                                    x_line_tbl                     => o_line_tbl,
                                    x_line_val_tbl                 => l_line_val_tbl,
                                    x_line_adj_tbl                 => l_line_adj_tbl,
                                    x_line_adj_val_tbl             => l_line_adj_val_tbl,
                                    x_line_price_att_tbl           => l_line_price_att_tbl,
                                    x_line_adj_att_tbl             => l_line_adj_att_tbl,
                                    x_line_adj_assoc_tbl           => l_line_adj_assoc_tbl,
                                    x_line_scredit_tbl             => l_line_scredit_tbl,
                                    x_line_scredit_val_tbl         => l_line_scredit_val_tbl,
                                    x_lot_serial_tbl               => l_lot_serial_tbl,
                                    x_lot_serial_val_tbl           => l_lot_serial_val_tbl,
                                    x_action_request_tbl           => x_action_qeq_tbl);
  --返回错误信息
  dbms_output.put_line('l_return_status:' || l_return_status);
  dbms_output.put_line('x_msg_data:' || x_msg_data);

  IF l_return_status <> fnd_api.g_ret_sts_success THEN
    dbms_output.put_line('未导入成功');
    x_msg_data := oe_msg_pub.get(p_msg_index => l_msg_count,
                                 p_encoded     => 'F');
    dbms_output.put_line(x_msg_data);
  ELSE
    dbms_output.put_line('S');
    dbms_output.put_line(x_action_qeq_tbl(1).return_status);
    dbms_output.put_line('订单编号：' || o_header_rec.order_number);
  END IF;
END;
```

3.2、更新销售订单行

```
DECLARE
   p_init_msg_list VARCHAR2 := fnd_api.g_false;
   p_commit              VARCHAR2 := fnd_api.g_false;
   x_return_status VARCHAR2;
   x_msg_count         NUMBER;
   x_msg_data          VARCHAR2;
   l_line_tbl          oe_order_pub.line_tbl_type;
   x_line_tbl          oe_order_pub.line_tbl_type;
   l_line_num          NUMBER;
   l_org_id            NUMBER;
   p_oe_header_id    NUMBER;
   p_schedule_date DATE;
   CURSOR c_lins IS
      SELECT ool.header_id, ool.line_id, ool.org_id, ool.request_date
        FROM oe_order_lines_all ool
       WHERE ool.header_id = p_oe_header_id
         AND oe_line_status_pub.get_line_status(p_line_id                  => ool.line_id,
                                                p_flow_status_code                         =>
ool.flow_status_code) NOT IN
                ('已发运', '已关闭', '已挑库', '部分挑库');
BEGIN
   l_line_tbl.delete;
   l_line_num := 0;
   FOR l_rec IN c_lins LOOP
      l_line_num := l_line_num + 1;
      l_org_id := l_rec.org_id;
      l_line_tbl(l_line_num) := oe_order_pub.g_miss_line_rec;
      l_line_tbl(l_line_num).line_id := l_rec.line_id;
      l_line_tbl(l_line_num).header_id := l_rec.header_id;
      l_line_tbl(l_line_num).schedule_ship_date := nvl(p_schedule_date,
                                                       l_rec.request_date);
      l_line_tbl(l_line_num).operation := oe_globals.g_opr_update;
   END LOOP;
   oe_order_pub.process_line(p_line_tbl             => l_line_tbl,
                             p_org_id               => l_org_id,
                             p_operating_unit => NULL,
                             x_line_out_tbl      => x_line_tbl,
                             x_return_status    => x_return_status,
                             x_msg_count          => x_msg_count,
                             x_msg_data           => x_msg_data);
   dbms_output.put_line(x_return_status);
```

```
END;
```

3.3、更新销售订单

```
DECLARE
   x_return_status                VARCHAR2(100);
   x_msg_count                     NUMBER;
   x_msg_data                      VARCHAR2(100);
   p_org_id                        NUMBER := 90;
   p_header_id                     NUMBER := 944004;
   p_flow_status_code              VARCHAR2(100) := 'PENDING_CUSTOMER_ACCEPTANCE';
   l_index                         NUMBER;
   l_header_rec                 oe_order_pub.header_rec_type := oe_order_pub.g_miss_header_rec;
   o_header_rec                    oe_order_pub.header_rec_type;
   l_header_val_rec               oe_order_pub.header_val_rec_type;
   l_header_adj_tbl               oe_order_pub.header_adj_tbl_type;
   l_header_adj_val_tbl         oe_order_pub.header_adj_val_tbl_type;
   l_header_price_att_tbl      oe_order_pub.header_price_att_tbl_type;
   l_header_adj_att_tbl         oe_order_pub.header_adj_att_tbl_type;
   l_header_adj_assoc_tbl      oe_order_pub.header_adj_assoc_tbl_type;
   l_header_scredit_tbl         oe_order_pub.header_scredit_tbl_type;
   l_header_scredit_val_tbl oe_order_pub.header_scredit_val_tbl_type;
   l_line_tbl                      oe_order_pub.line_tbl_type := oe_order_pub.g_miss_line_tbl;
   o_line_tbl                      oe_order_pub.line_tbl_type;
   l_line_adj_tbl                 oe_order_pub.line_adj_tbl_type;
   l_line_price_att_tbl         oe_order_pub.line_price_att_tbl_type;
   l_line_adj_att_tbl            oe_order_pub.line_adj_att_tbl_type;
   l_line_adj_assoc_tbl         oe_order_pub.line_adj_assoc_tbl_type;
   l_line_scredit_tbl            oe_order_pub.line_scredit_tbl_type;
   l_lot_serial_tbl               oe_order_pub.lot_serial_tbl_type;
   l_line_val_tbl                  oe_order_pub.line_val_tbl_type;
   l_line_adj_val_tbl            oe_order_pub.line_adj_val_tbl_type;
   l_line_scredit_val_tbl      oe_order_pub.line_scredit_val_tbl_type;
   l_lot_serial_val_tbl         oe_order_pub.lot_serial_val_tbl_type;
   l_action_qeq_tbl               oe_order_pub.request_tbl_type;
   x_action_qeq_tbl               oe_order_pub.request_tbl_type;
BEGIN
   mo_global.init('ONT');
   mo_global.set_policy_context('S', p_org_id);
   fnd_global.apps_initialize(user_id           => 2175,
                                        resp_id           => 50703,
                                        resp_appl_id => 20004);
   oe_msg_pub.initialize;
   oe_debug_pub.initialize;
```

```
l_index := 0;
FOR header IN (SELECT oeh.order_type_id
                 FROM oe_order_headers_all oeh
                WHERE oeh.header_id = p_header_id) LOOP
   l_header_rec.header_id                := p_header_id;
   l_header_rec.flow_status_code    := p_flow_status_code;
   l_header_rec.operation                := oe_globals.g_opr_update;
   l_header_rec.last_updated_by     := fnd_global.user_id;
   l_header_rec.last_update_date    := SYSDATE;
   l_header_rec.last_update_login := fnd_global.login_id;
   l_header_rec.order_type_id         := header.order_type_id;
   l_line_tbl.delete;
   FOR lines IN (SELECT oel.line_id,
                        oel.line_type_id,
                        oel.line_category_code,
                        oel.item_type_code,
                        oel.org_id
                   FROM oe_order_lines_all oel
                  WHERE oel.header_id = p_header_id) LOOP
      l_index := l_index + 1;
      l_line_tbl(l_index) := oe_order_pub.g_miss_line_rec;
      l_line_tbl(l_index).org_id := lines.org_id;
      l_line_tbl(l_index).header_id := p_header_id;
      l_line_tbl(l_index).line_id := lines.line_id;
      l_line_tbl(l_index).flow_status_code := p_flow_status_code;
      l_line_tbl(l_index).operation := oe_globals.g_opr_update;
      l_line_tbl(l_index).last_updated_by := fnd_global.user_id;
      l_line_tbl(l_index).last_update_date := SYSDATE;
      l_line_tbl(l_index).last_update_login := fnd_global.login_id;
   END LOOP;
END LOOP;
oe_order_pub.process_order(p_api_version_number         => 1.0,
                           p_org_id                     => p_org_id,
                           p_init_msg_list              => fnd_api.g_false,
                           p_return_values              => fnd_api.g_false,
                           p_action_commit              => fnd_api.g_false,
                           x_return_status              => x_return_status,
                           x_msg_count                  => x_msg_count,
                           x_msg_data                   => x_msg_data,
                           p_header_rec                 => l_header_rec,
                           p_line_tbl                   => l_line_tbl,
                           p_line_adj_tbl               => l_line_adj_tbl,
                           p_action_request_tbl         => l_action_qeq_tbl,
                           x_header_rec                 => o_header_rec,
```

```
                                x_header_val_rec              => l_header_val_rec,
                                x_header_adj_tbl              => l_header_adj_tbl,
                                x_header_adj_val_tbl          => l_header_adj_val_tbl,
                                x_header_price_att_tbl        => l_header_price_att_tbl,
                                x_header_adj_att_tbl          => l_header_adj_att_tbl,
                                x_header_adj_assoc_tbl        => l_header_adj_assoc_tbl,
                                x_header_scredit_tbl          => l_header_scredit_tbl,
                                x_header_scredit_val_tbl => l_header_scredit_val_tbl,
                                x_line_tbl                    => o_line_tbl,
                                x_line_val_tbl                => l_line_val_tbl,
                                x_line_adj_tbl                => l_line_adj_tbl,
                                x_line_adj_val_tbl            => l_line_adj_val_tbl,
                                x_line_price_att_tbl          => l_line_price_att_tbl,
                                x_line_adj_att_tbl            => l_line_adj_att_tbl,
                                x_line_adj_assoc_tbl          => l_line_adj_assoc_tbl,
                                x_line_scredit_tbl            => l_line_scredit_tbl,
                                x_line_scredit_val_tbl        => l_line_scredit_val_tbl,
                                x_lot_serial_tbl              => l_lot_serial_tbl,
                                x_lot_serial_val_tbl          => l_lot_serial_val_tbl,
                                x_action_request_tbl          => x_action_qeq_tbl);
   dbms_output.put_line(x_return_status);
END;
```

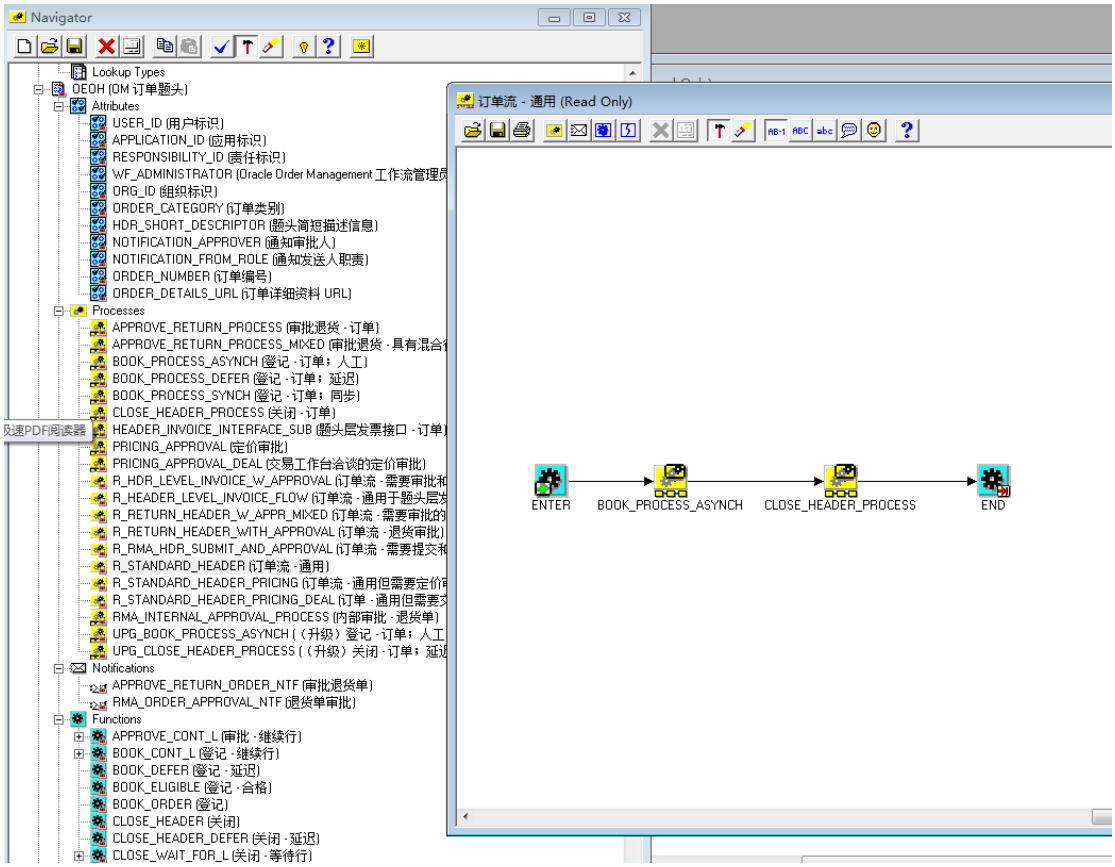
3.4、易出错误

- 已登记状态的销售订单，头部不允许修改。
- 不对头记录，行索引表，行记录初始化会报如下错误：
- 程序包 OE_Delayed_Requests_PVT 过程 LOGREQUEST 中出现错误 ORA-06502: PL/SQL: 数字或值错误： NULL 索引表键值
- OU 用的是头表里的 org_id,而不是职责或应用设置的 profile
- 在 cursor 里面用 commit 会报错：ORA-01002: 读取违反顺序。可以用自治事务解决。
- FND_MESSAGE 设置消息，FND_MSG_PUB 获取消息
- 计划发运日期：系统会根据日历控制，如果日期在休息日，会改到工作日。通过个性化也可以控制计划发运日期。
- 来源类型是内部，行状态是等待发运，外部是已预订。
- 如果是退货，行数量是负数，行类型的类别必须是 return 退货。
- 如过是退货，退货原因 return_reason_code 是必输字段
- 获取行状态不能仅从行基表字段取，要用包，

```
oe_line_status_pub.get_line_status(p_line_id=> ool.line_id,
                        p_flow_status_code=> ool.flow_status_code)
```

4、工作流

4.1、工作流概念



工作流由活动组成，活动可以是流程，通知，功能，事件。流程是活动的集合，包括流程，通知，功能，事件。流程本身可以作为其它流程的一个元素。

与形式语言很类似，前一个节点的输出决定转向后一个节点。

Item_Type：工作流模板，类似 Java 的类。

Item：具体的工作流实例，类似 Java 中的对象。项目关键字（item_key）也即是这个工作流实例的 ID，一般用事务处理单据的主键作为项目关键字（叫工作流实例名更准确），在创建工作流实例时指定。在销售订单中是 header_id。

属性和活动可以看作是 Java 中的属性，结构体和方法。

```
--活动的状态
SELECT wpa.process_version,
       wpa.activity_name,
       wias.activity_status,
       wias.activity_result_code,
       wias.*
  FROM wf_item_activity_statuses wias, wf_process_activities wpa
 WHERE item_key = '976002'
```

```
    AND wias.process_activity = wpa.instance_id
    AND wpa.process_name = 'INTERNAL_APPROVAL_PROCESS';
--流程上的活动
SELECT *
  FROM wf_process_activities wpa
 WHERE wpa.process_name = 'INTERNAL_APPROVAL_PROCESS'
    AND wpa.process_version = 2;
--活动定义
SELECT wa.result_type, wa.*
  FROM wf_activities_vl wa
 WHERE wa.name = 'NEGOTIATION_APPROVAL_NTF';
---------------
--工作流模板定义
SELECT wit.*
  FROM wf_item_types_vl wit
 WHERE wit.name = 'OENH';
--工作流实例定义
SELECT wi.*   FROM wf_items wi
 WHERE wi.item_type = 'OENH'
    AND wi.item_key = '964020';
---------------
工作流 lookup，每个节点的返回值在此取值，不是 fnd_lookup 的视图
SELECT * FROM wf_lookup_types w WHERE w.lookup_type = 'WFSTD_APPROVAL';
SELECT * FROM wf_lookups WHERE lookup_type = 'WFSTD_APPROVAL';
```

4.2、Workflow Engine

工作流的状况由工作流流程上的所有的活动的状况定义。引擎应答 API 调用来更新活动的状况。
基于之前活动的结果，引擎将直接执行下一个活动。
活动可能的状态：
Active：活动正在运行
Complete：活动正常完成
Waiting：活动等待运行
Notified：通知活动已经交付并打开
Deferred：活动延迟
Error：活动错误
Suspended：活动暂停

工作流引擎通过在功能活动之前设置保存点来捕捉由功能活动产生的错误。如果一个活动产生一个不能处理的异常，引擎会回滚到保存点，并设置活动的状态为‘ERROR’。所以，一定不要在功能活动的程序中有 commit 提交语句。工作流引擎从来不会有 commit 提交语句，因为 commit 提交是职责的责任。

数据库触发器和分布式事务处理不允许保存点，工作流引擎自动捕捉‘不允许保存点（Savepoint not allowed）’错误并且延迟活动的执行给后台引擎。
工作流组件异步执行，比如后台引擎和通知系统，代表调用程序适时 commit 提交。

Oracle 数据库支持自治事务，在定义存储过程时前缀 autonomous_transaction，可以独立于主事务进行提交回滚。Oracle 视之为一个独立的会话，所以不能够访问主会话做出的还没有提交的更改。因此，在自治事务中更新工作流数据是受到限制的，比如，不能设置工作流（item）的属性，因为工作流（item）本身还没有提交，否则会会主会话造成锁冲突。

工作不支持在它直接调用的任何存储过程中自治提交。如果需要提交，可以把你的 sql 语句嵌套在一个子程序中，声明其为一个自治块，子程序必须能够重复运行。工作流会通过回滚整个存储过程和设置状态为'ERROR'来处理错误。自治事务提交的更新不能够被回滚，所以要写额外的逻辑来处理错误。

4.3、CompleteActivity

通知工作流引擎某个工作流实例的指定的活动已经完成。
指明一个已经完成的活动并给出一个可选的结果：告诉工作流引擎一个异步活动已经完成。这个程序要求活动当前有一个 Notified 的状态。也可以传递一个可选的活动完成结果。这个结果决定了流程的下一步转换。
参数：
activity：完成的活动节点名。提供活动节点的标签名。如果活动节点标签名不能唯一标识子流程，可以在标签名之前缀上父流程的内部名，活动节点必须被标记为‘开始’活动。
result：可选的活动完成结果。可能的值由流程活动的‘Result Type’决定，或引擎标准结果。

```
--处理状态为‘noticed’的通知 不需要初始化
BEGIN
  /*fnd_global.apps_initialize(user_id          => 1801,
  resp_id          => 50847,
  resp_appl_id => 660);*/
  wf_engine.completeactivity(itemtype => 'OENH',
                             itemkey    => '986001',
                             activity => 'NEGOTIATION_APPROVAL_NTF',
                             RESULT     => 'REJECTED'); --REJECTED APPROVED
  COMMIT;
END;
```

4.4、Background

后台引擎通过指定的参数可以处理延迟的（deferred）活动，过时的（timeout）活动和停滞的（stuck）的流程。后台引擎被激活时，会执行所有的满足参数的活动。后台引擎不会长时

间运行，所以必须周期性的重启这个存储过程。在当前后台引擎开始后新延迟的活动，新过时的活动和新停滞的过程都会被下一个激活的后台引擎处理。
可以使用并发管理器设置后台引擎周期性的运行，也可以使用工作流管理器提交后台引擎并发程序。

```
--处理状态为‘defered'等的活动
BEGIN
   fnd_global.apps_initialize(user_id           => 1801,
                              resp_id           => 50847,
                              resp_appl_id => 660);
   wf_engine.background(itemtype => 'OENH');
   COMMIT;
END;
```