Oracle® Process Manufacturing

Formula API User's Guide

Release 11*i*

October 2000

Part No. A87318-01

ORACLE

Oracle® Process Manufacturing Formula API User's Guide, Release 11*i*

Part No. A87318-01

Primary Author: Richard D. Persen

Major Contributors: Shyamsunder Sitaraman, Bill Stearns

Contents

Send Us Your Comments

Oracle® Process Manufacturing Formula API User's Guide Release 11*i*

Part No. A87318-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the chapter, section, and page number (if available). You can send comments to us in the following ways:

- FAX: 650-506-7200 Attn: Oracle Process Manufacturing
- Postal service:
  Oracle Corporation
  Oracle Process Manufacturing
  500 Oracle Parkway
  Redwood City, CA 94065
  U.S.A.
- Electronic mail message to appsdoc@us.oracle.com

If you would like a reply, please give your name, address, and telephone number below.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

Welcome to the *Oracle Process Manufacturing Formula API User's Guide*. This user's guide includes the information you need to work with the Oracle Process Manufacturing (OPM) application effectively.

This preface explains how this user's guide is organized and introduces other sources of information that can help you.

Intended Audience

This guide assumes that you have working knowledge of your business area's processes and tools. It also assumes that you are familiar with OPM Formula Management. If you have never used OPM, we suggest you attend one or more of the Oracle Process Manufacturing training classes available through Oracle World Wide Education.

This guide also assumes that you are familiar with the Oracle Applications graphical user interface. To learn more about Oracle Applications graphical user interface, read the *Oracle Applications User's Guide*.

About This Guide

This guide contains overviews as well as task and reference information. It includes the following:

Name	Description
Formula API Introduction	Describes the Application Program Interfaces (APIs) that support external interfaces to the OPM Formula Management tables including: packages supplied, Formula API bill of material, wrapper function, stored procedures, and validating the user name.
Formula API Business Objects	Provides the relationships between Formula API table structure and its entities. Discusses Formula API business objects, the entity relationship diagram, business object interface design, creating a new formula and importing formula data structures
Appendix A	Provides message handling, the interpretation of error conditions, and an understanding of error messages.

Information Sources

You can choose from many sources of information, including documentation, training, and support services to increase your knowledge and understanding.

Online Documentation

Oracle Applications documentation is available on CD-ROM, except for technical reference manuals. User's guides are available in HTML format and on paper. Technical reference manuals are available on paper only. Other documentation is available on paper and sometimes in PDF format.

The content of the documentation remains the same from format to format. Slight formatting differences could occur due to publication standards, but such differences do not affect content. For example, page numbers are included on paper, but are not included in HTML.

The HTML documentation is available from all Oracle Applications windows. Each window is programmed to start your web browser and open a specific, context-sensitive section. Once any section of the HTML documentation is open, you can navigate freely throughout all Oracle Applications documentation.

Related Documents

Oracle Process Manufacturing shares business and setup information with other Oracle products. You may find the following Oracle Applications user's guides useful:

- *Oracle Applications User's Guide*
- *Oracle Application's Flexfields Guide*
- *Oracle Workflow User Guide*
- *Oracle Applications System Administrator's Guide*
- *Oracle General Ledger User's Guide*
- *Oracle Payables User's Guide*
- *Oracle Receivables User's Guide*
- *Oracle Human Resources North American User's Guide*
- *Oracle Purchasing User's Guide*

Oracle Process Manufacturing Guides

The following is a list of documentation in each product group for OPM:

Financials

- *Oracle Process Manufacturing Accounting Setup User's Guide*
- *Oracle Process Manufacturing Cost Management User's Guide*
- *Oracle Process Manufacturing Manufacturing Accounting Controller User's Guide*
- *Oracle Process Manufacturing and Oracle Financials Integration User's Guide*

Inventory Control

- *Oracle Process Manufacturing Intrastat Reporting User's Guide*
- *Oracle Process Manufacturing Inventory Management User's Guide*
- *Oracle Process Manufacturing Physical Inventory User's Guide*

Logistics

- *Oracle Process Manufacturing Order Fulfillment User's Guide*
- *Oracle Process Manufacturing Purchase Management User's Guide*

Process Execution

- *Oracle Process Manufacturing Process Operation Control User's Guide*
- *Oracle Process Manufacturing Production Management User's Guide*

Process Planning

- *Oracle Process Manufacturing Capacity Planning User's Guide*
- *Oracle Process Manufacturing Integration with Advanced Planning and Scheduling User's Guide*
- *Oracle Process Manufacturing MPS/MRP and Forecasting User's Guide*

Product Development

- *Oracle Process Manufacturing Formula Management User's Guide*
- *Oracle Process Manufacturing Laboratory Management User's Guide*
- *Oracle Process Manufacturing Quality Management User's Guide*

Regulatory

- *Oracle Process Manufacturing Regulatory Management User's Guide*

System Administration and Technical Reference

- *Oracle Process Manufacturing Implementation Guide*
- *Oracle Process Manufacturing System Administration User's Guide*
- Oracle Process Manufacturing Technical Reference Manuals

Training

Oracle offers a complete set of formal training courses to help you master Oracle Process Manufacturing and reach full productivity quickly. We organize these courses into functional learning paths, so you take only those courses appropriate to your area of responsibility.

You have a choice of educational environments. You can attend courses offered by Oracle Education Services at any one of our many Education Centers, or you can arrange for our trainers to teach at your facility. In addition, Oracle Training professionals can tailor standard courses or develop custom courses to meet your needs. For example, you may want to use your organization's structure, terminology, and data as examples in a customized training session delivered at your own facility.

Conventions

The following conventions are used in this guide:

Bolded Text

Buttons, fields, keys, menus, and selections are bolded in procedures only. For example: To access the next window, click **OK**. Otherwise, references to these features appear in regular type.

Additional Menu Options

Only nonstandard menu options are discussed. Standard menu bar options (such as Save) are not discussed. These standard options are described in the *Oracle Applications User's Guide*. Only menu options unique to the use of the specific window are discussed.

Field References

References to fields within procedures are in bold type. References within the body of this guide appear in regular type.

Required Fields

The word Required appears as the last word in the field description of all required fields. When the field is required contingent on the entry in another field, or only in specific situations, "Required if..." is the last sentence of the field description.

Fields Reserved for Future Use

Fields with no current processing implications are referenced by the statement "This field is not currently used" or "Reserved for future use." Do not use these fields for your own reference data, because there are plans to link future functionality to these fields. Fields intended for informational purposes only are referenced by the statement "This field is for informational purposes only."

Pending/Completed Transactions

Discussions about processing transactions that use the words pending and completed refer to the status of a transaction. Pending and completed do not refer to the database tables that are updated as a result of transactions (for example, some completed transactions are stored in the Pending Transactions table).

Procedures

Most topics contain a procedure with numbered steps. Any actions which are subordinate to a step are assigned letters. You can customize your Oracle Application, therefore, all procedures are suggestive only. Navigate to windows

and between responsibilities in a way that works best for your particular setup. Also note that fields may appear in a different order than they are discussed.

Use of the Word Character

The word character means an alphanumeric character. Characters that are numeric or alphabetic only are referenced specifically. Depending on your system security profile, you many not have access to all of the windows and functions described in this guide. If you do not see a menu option described in this guide, and you want access to it, contact your System Administrator.

Do Not Use Database Tools to Modify Oracle Applications Data

Oracle Applications tables are interrelated. As a result, any change you make using Oracle Applications can update many tables at once. If you modify the Oracle Applications data using anything other than Oracle Applications, you could change a row in one table without making corresponding changes in related tables. If your tables are not synchronized with each other, you risk retrieving erroneous information and receiving unpredictable results throughout Oracle Applications.

When you use Oracle Applications to modify your data, Oracle Applications automatically checks that your changes are valid. Oracle Applications also track who changes information. If you enter information into database tables using database tools, you could store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

Consequently, we strongly recommend that you never use SQL*Plus or any other tool to modify Oracle Applications data unless otherwise instructed by Oracle Support Services.

About Oracle

Oracle Corporation develops and markets an integrated line of software products for database management, applications development, decision support, and office automation, as well as Oracle Applications, an integrated suite of more than 45 software modules for financial management, supply chain management, manufacturing, project systems, human resources, sales, and service management.

Oracle Products are available for mainframes, minicomputers, personal computers, network computers, and personal digital assistants, allowing organizations to integrate different computers, different operating systems, different networks, and

even different database management systems, into a single, unified computing, and information resource.

Oracle is the world's leading supplier of software for information management, and the world's second largest software company. Oracle offers its database, tools, and applications products, along with related consulting, education, and support services in over 140 countries around the world.

Thank You

Thank you for choosing Oracle Process Manufacturing and this user's guide.

We value your comments and feedback. At the beginning of this guide is a Reader's Comment Form that you can use to explain what you like or dislike about this user's guide. Mail your comments to the following address or call us directly at 650-506-7000.

Oracle Applications Documentation Manager
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Or, send an electronic mail message to appsdoc@us.oracle.com

1

Formula API Introduction

This document describes the Application Program Interfaces (APIs) that support external interfaces to the Oracle Process Manufacturing (OPM) Formula Management tables. The topics discussed in this topic are:

- Introducing the Formula API
- Oracle Applications Packages Supplied
- Formula API Bill of Materials
- Wrapper Function and Stored Procedures
- Validating the User Name

Introducing the Formula API

Formula APIs allow you to import Formula information from an existing formula management system into the Oracle Process Manufacturing Formula Management tables. When you import formulas you can now include all pertinent information using a friendly tool that does not have cryptic IDs and system specific information. Formula APIs can process virtually all formula types. The interface insures that your imported formulas contain the same detail as those you would enter manually on the OPM Formulas window. However, every formula should include at least one product and one ingredient.

What's In This Document?

This document describes the basic business needs, major features, architecture, and components for the Insert, Update and Delete features for the Formula APIs. Much of the application is divided into module-specific objects that allow you to link OPM functionality into your own programs. The interfaces can make use of the standard functionality and logic implemented in the Formula Management module.

Formula APIs are currently written in PL/SQL which can be called by your own programs. To make use of these APIs, you must code your wrapper function that passes the appropriate parameters to the APIs. Your program will also be responsible for connecting to a database before calling an API function, and disconnecting from the database upon return. You may also choose to write log files before calling and after returning from a function. If there is a problem during execution of a call, the APIs return one of the following status codes:

- S for success
- E for error
- U unknown/unexpected status.

Which API Functions are Included?

The following presents a list of Formula API functions, the business object name, and a brief explanation of each function:

Function	Business Object Name	Description
Formula Create	GMD_FORMULA_xxx	Creates formula header, detail and effectivity information. It can be used for modification of the formula header information.
Formula Detail	GMD_FORMULA_DETAIL_xxx	Creates or modifies the formula detail information.

Function	Business Object Name	Description
Formula Effectivity	GMD_FORMULA_EFFECTIVITY_xxx	Creates or modifies the formula effectivity information.
Sample Wrapper Function	create_formula (Note: This is a sample wrapper function. You will need to design your own.)	Parses the formulated information from a flat (text) file and structures the data into PL/SQL table format that is passed as a parameter to API.

Where xxx in the Business Object Name is PVT (private) or PUB (public).

Formula API Features

- Creating Updating and Deleting Information
- Proper Encapsulation
- Synchronous Processing Following the Business Hierarchy
- Detailed and Translatable Error Messages

Formula API Support Policy

Formula APIs are supported by Oracle. This means:

- Oracle will provide objects and libraries needed to use to the APIs and the documentation for their use.
- Oracle will ensure that the APIs function as designed.
- Oracle will not support customer generated programs that use the APIs.

Oracle Applications Packages Supplied

Formula APIs make use of the following standard Oracle Applications packages:

- FND_API - the standard Oracle Applications API version checking function. This is used by the stored procedure to check valid API version number and also contains constant variables such as TRUE and FALSE.
- FND_MESSAGE - the standard Oracle Applications messaging function. This is used by the stored procedure to report status and error handling.
- FND_PUB_MSG - the standard Oracle Applications message retrieval function, used to interrogate the procedure messages.

These packages are installed as part of the current release. Please refer to the *Oracle Applications Coding Standards* manual for additional details.

Formula API Bill of Materials

The following is a list of packages and files that are delivered with OPM Formula API. These must be on your system for your interface to compile and link properly.

Package Name	File Names	Description
GMD_FORMULA_PUB	GMDPFMHS.pls GMDPFMHB.pls	Public Formula Header package that the user defined function calls. The business API can be used for creating, modifying, or deleting a formula header. While creating a Formula header the API also creates Detail and Effectivity associated with this header.
GMD_FORMULA_DETAIL_PUB	GMDPFMDS.pls GMDPFMDB.pls	Public Formula Detail package that the wrapper or user defined function calls. The business API can be used for creating, modifying, or deleting a formula detail.
GMD_FORMULA_EFFECTIVITY_PUB	GMDPFMES.pls GMDPFMEB.pls	Public Formula Effectivity package that the wrapper or user defined function calls. The business API can be used for creating, modifying, or deleting a formula effectivity.
GMD_FORMULA_PVT	GMDVFMHS.pls GMDVFMHB.pls	Private Formula Header package that cannot be called directly by the user defined function. The private API is called by public formula header API after performing the necessary validations.

Package Name	File Names	Description
GMD_FORMULA_DETAIL_PVT	GMDVFMDS.pls GMDVFMDB.pls	Private Formula Detail package that cannot be called directly by the user defined function. The private API is called by public formula detail API after performing the necessary validations.
GMD_FORMULA_EFFECTIVITY_PVT	GMDVFMES.pls GMDVFMEB.pls	Private Formula Header package that cannot be called directly by the user defined function. The private API is called by the Public Formula Effectivity API after performing the necessary validations.
Public Package for Surrogates	GMDPSURS.pls GMDPSURB.pls	Public Formula Surrogate package used primarily for creating surrogate keys for formula header, detail lines, and effectivity.
Public Validation Packages	GMDPFMVB.pls GMDPFMVS.pls	Public Formula Validation package used by other procedures in the APIs to perform all business rules validations. For example, prior to the formula line insertion the procedure 'Detail_line_ val' (included in this package) is called to check if the same line exists in the OPM system. It also uses lookup tables for validations.
GMD_FORMULA_COMMON_PKG	GMDPFMCS.pls	Common formula package that defines common data type structure (PL/SQL table) for inserts or updates.

Wrapper Function and Stored Procedures

A wrapper function is used for collating the formulated data in a specific structure (PL/SQL table). This wrapper function calls the Formula API by passing this PL/SQL table as a parameter. Depending on the action (creation/inserts or modification/updates) and the entity (header/detail/effectivity) these wrapper functions can call the public procedures defined in the APIs. For example, to insert all Formula details you can call the procedure 'Insert_FormulaDetail' defined in the public package GMD_FORMULA_DETAIL_PUB. The stored procedures return one of these the status codes:

- S for success
- E for error
- U unknown/unexpected status.

Depending on the status returned, you can decide if the operation needs to be committed. Optionally, the defined calling function can make the API commit after performing the operation.

Validating the User Name

The Formula API validates the user name that has been passed to the API against the fnd_user table. If the user you enter does not have a valid ID, the API returns an error without performing any other functions or validations.

2

Formula API Business Objects

In order to use Formula API Business Objects it is important to understand the relationships of the table structure with all its entities. You should also understand the fundamental API business object interface design before you import formula data structures and manage any errors that arise during these operations.

The following topics are discussed:

- Using Formula API Business Objects
- Formula API Entity Diagram
- Formula API Business Object Interface Design
- Creating a New Formula
- Importing Formula Data Structures

Using Formula API Business Objects

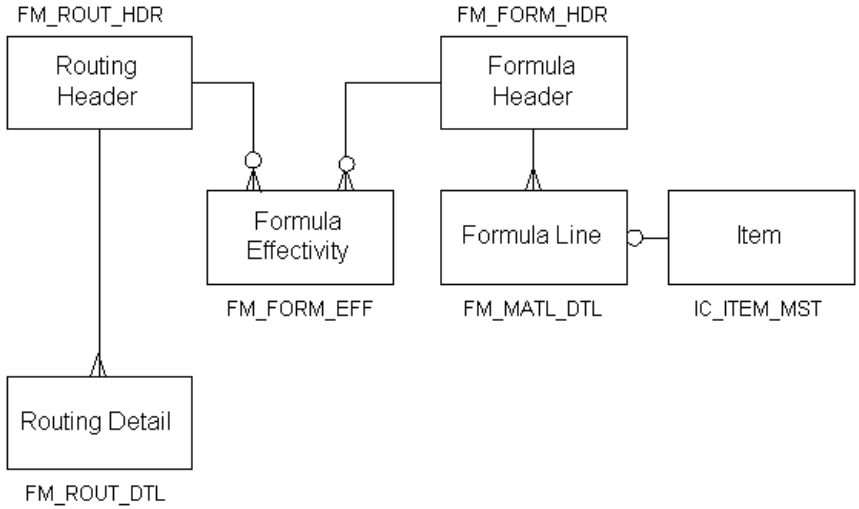
A formula is a method to prescribe ingredients and quantities that are required to manufacture a particular product or products. Formulas occasionally contain by-products and coproducts. The information included in a formula is used in various other Oracle Manufacturing modules, including Supply Chain Planning (Material Requirement Planning and Advanced Planning and Scheduling), Production Management and Costing.

Formula APIs are business objects that can create or change the formula information. Formula information within the OPM application is comprised of:

- A header that provides the formula number, version, description and status or validity of a formula (active / inactive).
- One or more detail lines for each header. Each detail line provides the product / coproduct / ingredient / by-product specification for a formula.
- One or more effectivity that represents the set of conditions under which a formula can be used. For example: date, quantity and customer specifics for a formula can define a unique effectivity. Effectivity also provides the link between a formula and routing (actual steps or operation carried out in a manufacturing setup).

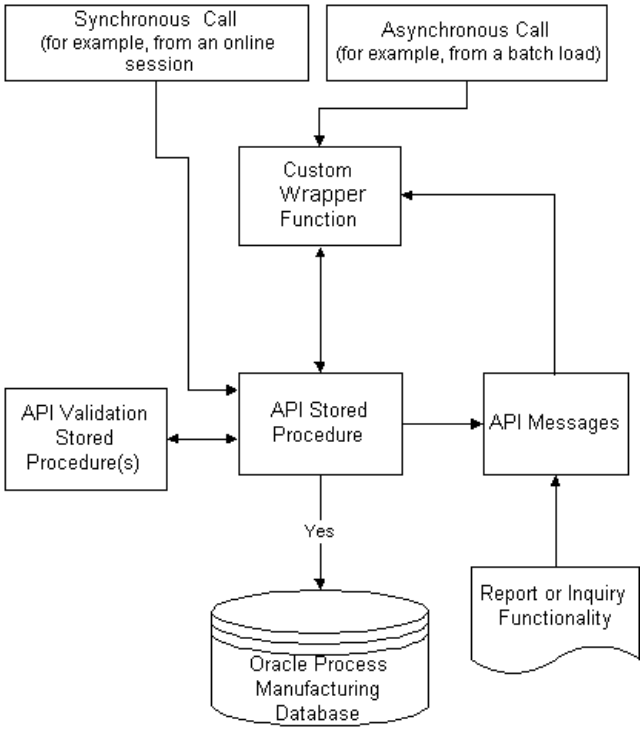
Formula API Entity Diagram

The following diagram shows the table structure for the Formula API business object along with all its entities:



Formula API Business Object Interface Design

The following is a schematic of the Oracle Process Manufacturing (OPM) stored procedure API mechanism. It shows the structure of the API and the mechanism needed to execute it. It also indicates the functionality required to both execute and audit results.



Creating a New Formula

In order to use the Formula business object APIs effectively, you must follow these steps:

Step 1: Collate formula data into a PL/SQL table format.

Formula APIs can be called through different types of sources. For example, you can use an interface like Oracle forms, or optionally you can create a wrapper function that calls these APIs. However, you need to ensure that relevant formula information is structured in a PL/SQL table format *before* passing it as a parameter to these APIs.

A test function ('Create_formula') provided with the APIs describes a typical wrapper function that creates a PL/SQL table from a comma-delimited text file. The function parses through this flat file and creates the table structure that is passed as parameter to the 'Insert_Formula' procedure within the API 'GMD_ FORMULA_HEADER_PUB'. Since the test function is just a representation of a typical wrapper function, the actual wrapper function may be designed differently.

The APIs also require certain standard parameters such as API version and API name which need to be passed by the wrapper function. After performing the appropriate tasks the APIs return the status code. Depending on this status code, the calling function may decide to commit the work. If the return code is an error, the function could retrieve the error message text from the error stack.

Step 2: Call the Public API - Main validation performed:

a. The API checks for the existence of an appropriate User ID in the FND_USR table. If there is no valid user, the API puts error messages in an error stack, and prevents the creation of a formula.

b. The API checks for valid formula number and version. For all updates or changes to the Formula Header information, Formula ID, or (Formula Number and Formula Version) need to be provided. For changes to Formula Detail and Effectivity, the Formula Line ID and Formula Effectivity ID need to be provided.

c. The API checks every Formula Header to determine that there is at least one product and one ingredient associated with it.

d. After the formula header is created successfully, formula details and formula effectivity are created. If a header has more than one line (which is usual), the API creates a header only once (for the first line), and creates all Formula Details as separate lines.

e. No changes or updates are performed for any formulas that have cost rollups done on them.

Step 3: Review any error messages.

The API returns the status code as one of its parameters after it is executed. The status code could represent 'S' for success, 'E' for error or 'U' Unexpected or Unknown status. If an error 'E' has occurred, the calling function or wrapper function could analyze the errors from the message stack.

For more information on error messages please refer to Appendix A .

Step 4: Check if relevant formula information has been created.

After executing these APIs successfully, and committing the work, the Formula information should be tested using a different user interface. For example, after the formula (header, detail and effectivity) is created, commit the work, and run the appropriate Oracle application to test whether or not a new formula has been actually created. You should check to determine if at least one product and one ingredient is associated with the newly created formula.

Importing Formula Data Structures

The following topic demonstrates the mechanics of importing data structures for:

- Formula Header
- Formula Detail
- Formula Effectivity

A single general input structure is described in the following topic.

Understanding the General Input Structure

All stored procedure API will be called with PL/SQL parameters. Examination of both the x_return_status and x_msg_count will indicate the pass/fail status of the call. Standard parameters that are common to all API activities and their details are summarized in the following:

Parameter	Type	IN/OUT	Required	Validation
p_api_version	varchar2	IN	Y	Validates version compatibility. The version sent by the calling function is compared to the internal version of the API and an unexpected error (U) is generated if these do not match.
p_init_msg_list	varchar2	IN	N	Used to specify whether the message list should be initialized on entry to the API. It is an optional parameter, and if not supplied will default to FND_API.G_ FALSE which means that the API will not initialize the message list.
p_commit	varchar2	IN	N	Used to specify whether the API should commit its work before returning to the calling function. If not supplied it will default to FND_API.G_FALSE.
x_return_status	varchar2	OUT	-	Specifies whether the API was successful or failed: 'S' Successful 'E' failed due to expected error 'U' failed due to unexpected error
x_msg_count	number	OUT	-	Specifies number of messages added to message list.

Parameter	Type	IN/OUT	Required	Validation
x_msg_data	varchar2	OUT	-	Returns the messages in an encoded format. These messages can then be processed by the standard message functions as defined in Business Object API Coding Standards Document.

Using Input Parameters for Each Formula API

The Formula API requires data to passed in as PL/SQL tables. The wrapper functions can create these tables by referencing the table types in the API specifications. Although the table types are similar, the default values set for these table types are different depending on the DML operation.

If the wrapper function needs to insert or create new formula header detail, it needs to create a table of type:

- p_formula_insert_tbl_type

The specification for this is provided in this topic.

If the wrapper function needs to update or delete a formula header detail, it needs to create a table of type:

- p_formula_update_tbl_type

The table type provides default values for all optional parameters.

These parameters pass the item-specific data required to create an inventory item. However, parameters can have formula data related to formula detail and effectivity. Since the header data can be in multiple instances it is passed as a PL/SQL table.

Parameter Specification for p_formula_insert_tbl_type

The parameter specification for p_formula_insert_tbl_type is described below. This specification is used for inserts. You will note that no validation is applied to the descriptive flexfield segments.

Field/Column	Type	Length	Default	Required	Validation
formula_no	varchar2	32		Y	Must be non-space
formula_vers	number	4		Y	Must be > = 0 The formula_no / formula_vers combination must NOT exist on fm_form_mst Duplicates are not allowed.
formula_type	number	5	0	N	0 - standard formula
scale_type	number	5		Y	0 - formula cannot be scaled 1 - formula can be scaled
formula_desc1	varchar2	70		Y	
formula_desc2	varchar2	70	null	N	
formula_class	varchar2	32	null	N	Must exist on fm_form_cls with delete_mark=0
fmcontrol_class	varchar2	32	null	N	Not currently used; must be null
inactive_ind	number	5	0	N	0 = active 1 = inactive
formula_id	number				
formulaline_id	number				
line_type	number	5		Y	-1 is an ingredient 1 is a product 2 is a by-product Line-type of 1 (product) must be in place against a particular formula before line types of -1 and 2 can be accepted.
line_no	number	5		Y	Must be > 0 Duplicates are not allowed within line_type

Field/Column	Type	Length	Default	Required	Validation
item_no	varchar2	32		Y	Must exist in ic_item_mst with experimental_ind=0, inactive_ind=0, delete_mark=0
qty	number			Y	Must be > = 0
item_um	varchar2	4	ic_item_mst.item_um	N	Must exist on sy_uoms_mst Conversion to ic_item_mst.item_um (primary UOM) must be defined.
release_type	number	5		Y	0 = automatic release 1 = partial release
scrap_factor	number		0	N	Must be >= 0 Must be <= 10,000
scale_type	number	5	1	N	0 - fixed 1 - linear scaling 2 - step scaling
cost_alloc	number		0	N	Must be in the range 0 to 1 inclusive
phantom_type	number	5	0	N	0 - not a phantom 1 - automatic phantom replacement 2 - manual phantom
rework_type	number	5	0	N	0 - no rework 1 - default qty as stated
fmeff_id	number				
orgn_code	varchar2	4	null	N	If supplied, must exist on sy_orgn_mst with delete_mark=0
item_no	varchar2	32		Y	Must exist on ic_item_mst with delete_mark=0 & inactive_ind=0 Must be a product associated with the formula nominated above. Check fm_matl_dtl for line_type=1 where formula_id and item_id match the supplied criteria

Field/Column	Type	Length	Default	Required	Validation
formula_use	number	5		Y	0 - production 1 - planning 2 - costing 3 - MSDS
start_date	date		system date	N	Must be a valid date. Must be >= SY$MIN_DATE Must be <. SY$MAX_DATE
end_date	date		SY$MAX_DATE	N	Must be a valid date. Must be > start_date Must be <= SY$MAX_DATE
min_qty	number	5	0	N	Must be >= 0
max_qty	number	5	999999999	N	Must be > 0
std_qty	number	5		Y	Must be > 0
item_um	varchar2	4	ic_item_mst.item_um	N	Must be valid UOM from sy_uoms_mst where delete_mark=0 If different from the item_master primary unit of measure, a conversion must exist
preference	number	5	1	N	Must be >= 0
routing_id	number	10		N*	
routing_no	varchar2	32		N*	Must be non-space.
routing_vers	number	5		N*	Must be > = 0 The routing_no / routing_vers combination must exist on fm_rout_hdr where delete_mark=0.
customer_no	varchar2	32	0	N	Must exist on op_cust_mst with delete_mark=0 and co_code=SY$DEFAULT_CO
attribute1	varchar2	240	null	N	
attribute2	varchar2	240	null	N	
attribute3	varchar2	240	null	N	
attribute4	varchar2	240	null	N	

Field/Column	Type	Length	Default	Required	Validation
attribute5	varchar2	240	null	N	
attribute6	varchar2	240	null	N	
attribute7	varchar2	240	null	N	
attribute8	varchar2	240	null	N	
attribute9	varchar2	240	null	N	
attribute10	varchar2	240	null	N	
attribute11	varchar2	240	null	N	
attribute12	varchar2	240	null	N	
attribute13	varchar2	240	null	N	
attribute14	varchar2	240	null	N	
attribute15	varchar2	240	null	N	
attribute16	varchar2	240	null	N	
attribute17	varchar2	240	null	N	
attribute17	varchar2	240	null	N	
attribute18	varchar2	240	null	N	
attribute19	varchar2	240	null	N	
attribute20	varchar2	240	null	N	
attribute21	varchar2	240	null	N	
attribute22	varchar2	240	null	N	
attribute23	varchar2	240	null	N	
attribute24	varchar2	240	null	N	
attribute25	varchar2	240	null	N	
attribute26	varchar2	240	null	N	
attribute27	varchar2	240	null	N	
attribute28	varchar2	240	null	N	
attribute29	varchar2	240	null	N	
attribute30	varchar2	240	null	N	
attribute_ category	varchar2	30	null	N	
user_name	varchar2	100		Y	Must exist on fnd_user

Parameter Specification for p_formula_update_tbl_type

The parameter specification for p_formula_update_tbl_type is described below. This specification is used for updates. You will note that no validation is applied to descriptive flexfield segments.

Field/Column	Type	Length	Default	Required	Validation
formula_no	varchar2	32		Y	Must be non-space
formula_vers	number	4		Y	Must be > = 0 formula_no / formula_vers combination must NOT exist on fm_form_mst Duplicates are not allowed.
formula_vers	number	4		Y	Must be > = 0 formula_no / formula_vers combination must NOT exist on fm_form_mst Duplicates are not allowed.
formula_type	number	5	0	N	0 - standard formula
scale_type	number	5		Y	0 - formula cannot be scaled 1 - formula can be scaled
formula_desc1	varchar2	70		Y	
formula_desc2	varchar2	70	FND_API. G_MISS_CHAR	N	
formula_class	varchar2	32	FND_API. G_MISS_CHAR	N	Must exist on fm_form_cls with delete_mark=0
fmcontrol_class	varchar2	32	FND_API. G_MISS_CHAR	N	Not currently used; must be null
inactive_ind	number	5	0	N	0 = active 1 = inactive
formula_id	number				

Field/Column	Type	Length	Default	Required	Validation
formulaline_ id	number				
line_type	number	5		Y	-1 - ingredient 1 - product 2 - by-product Line-type of 1 (product) must be in place against a particular formula before line types of -1 and 2 can be accepted.
line_no	number	5		Y	Must be > 0 Duplicates are not allowed within line_ type
item_no	varchar2	32		Y	Must exist in ic_item_ mst with experimental_ind=0, inactive_ind=0, delete_ mark=0
qty	number			Y	Must be > = 0
item_um	varchar2	4	ic_item_mst.item_ um	N	Must exist on sy_ uoms_mst. Conversion to ic_item_ mst.item_um (primary UOM) must be defined.
release_type	number	5		Y	0 = automatic release 1 = partial release
scrap_factor	number		0	N	Must be >= 0 Must be <= 10,000
scale_type	number	5	1	N	0 - fixed 1 - linear scaling 2 - step scaling
cost_alloc	number		0	N	Must be in the range 0 to 1 inclusive
phantom_type	number	5	0	N	0 - not a phantom 1 - automatic phantom replacement 2 - manual phantom

Field/Column	Type	Length	Default	Required	Validation
rework_type	number	5	0	N	0 - no rework 1 - default qty as stated
fmeff_id	number				
orgn_code	varchar2	4	FND_API. G_MISS_CHAR	N	If supplied, must exist on sy_orgn_mst with delete_mark=0
item_no	varchar2	32		Y	Must exist on ic_item_mst with delete_mark=0 & inactive_ind=0 Must be a product associated with the formula nominated above. Check fm_matl_dtl for line_type=1 where formula_id and item_id match supplied criteria
formula_use	number	5		Y	0 - production 1 - planning 2 - costing 3 - MSDS
start_date	date		system date	N	Must be a valid date. Must be >= SY$MIN_DATE Must be <. SY$MAX_DATE
end_date	date		SY$MAX_DATE	N	Must be a valid date. Must be > start_date Must be <= SY$MAX_DATE
min_qty	number	5	0	N	Must be >= 0
max_qty	number	5	999999999	N	Must be > 0
std_qty	number	5		Y	Must be > 0

Field/Column	Type	Length	Default	Required	Validation
item_um	varchar2	4	ic_item_mst.item_um	N	Must be valid UOM from sy_uoms_mst where delete_mark=0 If different from the item_master primary unit of measure, a conversion must exist
preference	number	5	1	N	Must be >= 0
routing_id	number	10		N*	
routing_no	varchar2	32		N*	Must be non-space.
routing_vers	number	5		N*	Must be > = 0 routing_no / routing_vers combination must exist on fm_rout_hdr where delete_mark=0.
customer_no	varchar2	32	0	N	Must exist on op_cust_mst with delete_mark=0 and co_code=SY$DEFAULT_CO
attribute1	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute2	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute3	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute4	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute5	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute6	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute7	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute8	varchar2	240	FND_API. G_MISS_CHAR	N	

Field/Column	Type	Length	Default	Required	Validation
attribute9	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute10	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute11	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute12	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute13	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute14	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute15	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute16	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute17	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute17	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute18	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute19	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute20	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute21	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute22	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute23	varchar2	240	FND_API. G_MISS_CHAR	N	

Field/Column	Type	Length	Default	Required	Validation
attribute24	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute25	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute26	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute27	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute28	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute29	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute30	varchar2	240	FND_API. G_MISS_CHAR	N	
attribute_ category	varchar2	30	FND_API. G_MISS_CHAR	N	
user_name	varchar2	100		Y	Must exist on fnd_user

A

Messages and Errors

This appendix covers the following topics:

- Handling Messages
- Interpreting Error Conditions
- Understanding Error Messages

Handling Messages

APIs put result messages into a message list. Programs calling APIs can then get the messages from this list, and process them appropriately. This may be by issuing them to the user if calling the API from an interactive process, or writing them to database tables or log files if calling the API from a batch process.

Messages are stored in an encode format to enable API callers to find out message names by using the standard functions provided by the message dictionary.

The structure of the message list is not public. Neither API developers nor API callers can access this list except through calling the API message utility routines mentioned below.

The following utility functions are defined in the FND_MSG_PUB package, in the file AFASMSGS.pls:

- Initialize - Initializes the API message list
- Add - Adds a message to the API message list
- Get - Gets a message from the API message list
- Count_Msg - Returns the number of messages in the API message list
- Delete - Deletes one or more messages from the API message list
- Reset - Resets the index used in getting messages
- Count_And_Get - Returns the number of messages in the API message list. If this number is one, it also returns the message data.

You can refer to *Oracle Applications Coding Standards* for complete documentation of these functions and procedures for usage information.

To add a message to the API message list, developers should use the regular message dictionary procedures FND_MESSAGE.SET_NAME and FND_MESSAGE.SET_TOKEN to set the message name and tokens on the message dictionary stack. They should then call FND_MSG_PUB.Add to fetch the messages off the message dictionary stack and add it to the API message list.

To get a message from the API message list, API callers should use the procedure FND_MSG_PUB.Get. This procedure operates in 5 different modes :

- First - Gets the first message in the API message list
- Next - Gets the next message in the API message list
- Last - Gets the last message in the API message list

- Previous - Gets the previous message in the API message list
- Specific - Gets a specific message from the API message list

Interpreting Error Conditions

The parameter x_return_status indicates whether the API was successful or failed. The values are as follows:

- S - Successful
- E - Expected error
- U - Unexpected error

Understanding Error Messages

These error messages will be output to the stored procedure message file, and can be monitored through the return x_msg_count. In conjunction with the x_return_ status, this can be used to monitor the success or failure of the procedure call.

Displaying Errors in Languages Other than English

Language translation of error messages is determined by the environment variable NLS_LANGUAGE. If the message is not found in the required language, then the message will be retrieved in US English.

The following is a complete listing of Formula API Error Messages:

Error Messages (Listed Alphabetically)	**Message Code**
Duplicate line number for item &ITEM_NO in formula/version &FORMULA_NO &FORMULA_VERS	FM _DUPLICATE_LINE_NO
Experimental item &ITEM_NO in formula/version &FORMULA_NO &FORMULA_VERS	FM _EXPERIMENTAL_ITEM
Formula number/version &FORMULA_NO &FORMULA_VERS already exists	FM _FORMULA_ALREADY_EXISTS
Formula number/version &FORMULA_NO &FORMULA_VERS cannot be located	FM _INVALID_FORMULA

Error Messages (Listed Alphabetically)	Message Code
Ingredient / by-product lines cannot be loaded until product is established for formula / version &FORMULA_NO &FORMULA_VERS	FM _MISSING_PRODUCT
Invalid cost allocation for item &ITEM_NO in formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_COST_ALLOC
Invalid customer &CUST_NO	FM_INVALID_CUSTOMER
Invalid end_date against effectivity for formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_END_DATE
Invalid formula class for formula / version &FORMULA_NO &FORMULA_VERS	FM _INVALID_FORMULA_CLASS
Invalid formula type for formula / version &FORMULA_NO &FORMULA_VERS	FM _INVALID_FORMULA_TYPE
Invalid formula_use for formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_FORMULA_USE
Invalid item &ITEM_NO in formula / version &FORMULA_NO &FORMULA_VERS	FM _INVALID_ITEM
Invalid item number &ITEM_NO	FM_INVALID_ITEM_NO
Invalid line number for item &ITEM_NO in formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_LINE_NO
Invalid line type for item &ITEM_NO in formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_LINE_TYPE
Invalid maximum quantity against effectivity for formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_MAX_QTY
Invalid minimum quantity against effectivity for formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_MIN_QTY
Invalid phantom type for item &ITEM_NO in formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_PHANTOM_TYPE

Error Messages (Listed Alphabetically)	Message Code
Invalid preference against effectivity for formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_PREFERENCE
Invalid product &ITEM_NO for formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_FORMULA_PRODUCT
Invalid release type for item &ITEM_NO in formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_RELEASE_TYPE
Invalid rework type for item &ITEM_NO in formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_REWORK_TYPE
Invalid routing number / version &ROUTING_ NO &ROUTING_VERS	FM_INVALID_ROUTING
Invalid scale type for formula / version &FORMULA_NO &FORMULA_VERS	FM _INVALID_SCALE_TYPE
Invalid scrap factor for item &ITEM_NO in formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_SCRAP_FACTOR
Invalid standard quantity against effectivity for formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_STD_QTY
Invalid start_date against effectivity for formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_START_DATE
Invalid UOM &ITEM_UM against effectivity for formula / version &FORMULA_NO &FORMULA_VERS	FM_INVALID_UOM
Invalid user name &USER_NAME	FM_INVALID_USER
Missing formula number	FM _MISSING_FORMULA_NO
Missing formula version	FM _MISSING_FORMULA_VERS
Negative quantity not permitted see item &ITEM_NO in formula / version &FORMULA_ NO &FORMULA_VERS	FM _NEGATIVE_QTY

Error Messages (Listed Alphabetically)	Message Code
No conversion defined from &ITEM_UM to primary UOM for item &ITEM_NO in formula/version &FORMULA_NO &FORMULA_VERS	FM_UOM_CONVERSION
No conversion to standard UOM for item &ITEM_NO	FM_MISSING_UOM_CONVERSION
Supplied Organization code &ORGN_CODE is invalid	FM_INVALID_ORGN_CODE

Index

G

H

I

L

V

W

X