PassQueen
a.k.a dumpsqueen.com

Oracle Hyperion Financial Management 11 Essentials

Oracle 1z0-532

Version Demo

Total Demo Questions: 10

Total Premium Questions: 103

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, Financial Management Architecture	10
Topic 2, Application Creation and Maintenance	13
Topic 3, Metadata	26
Topic 4, Data Management	11
Topic 5, Business Rules	15
Topic 6, Consolidation	8
Topic 7, Journals	5
Topic 8, Process Management	5
Topic 9, Security	6
Topic 10, Reporting	4
Total	103

QUESTION NO: 1

Identify the three true statements about accounts and account types. (Choose three.)

A. Financial Management provides predefined account types such as Asset, Revenue, Flow, Balance, and Dynamic.

B. Account types tell FM how the child account should aggregate to the parent.

C. Account types tell FM how the account balances should accumulate over time.

D. Base level accounts do not automatically roll up to parents; business rules are required.

E. Children cannot inherit the account type of their parent unless specifically defined by the administrator.

ANSWER: A B C

Explanation:

In Oracle Hyperion Financial Management, every account is assigned an account type. Financial Management provides predefined account types, including Asset, Liability, Revenue, Expense, Flow, Balance, and Dynamic, so “Financial Management provides predefined account types such as Asset, Revenue, Flow, Balance, and Dynamic” is true. The account type is essential application metadata: it defines the behavior used when child accounts aggregate into parent accounts. Consequently, “Account types tell FM how the child account should aggregate to the parent” is also true.

Account types additionally control the way account data accumulates across periods. Balance-sheet-oriented types such as Asset and Liability represent balances at a point in time, whereas income-statement-oriented types such as Revenue and Expense accumulate activity over time for period-to-date, quarter-to-date, and year-to-date reporting. Therefore, “Account types tell FM how the account balances should accumulate over time” accurately describes their purpose. These settings allow HFM to apply consistent consolidation and time-balance behavior throughout the account hierarchy without requiring separate calculations for standard rollup behavior. Oracle’s Financial Management administration documentation describes account type as the property that determines both child-to-parent aggregation and the accumulation of account balances over time. See the Oracle Hyperion Financial Management Administrator’s Guide and the Oracle Hyperion Financial Management product information.

QUESTION NO: 2

Assuming you are using HFM Version 11.1.1.3, identify the three true statements about Calculation Manager. (Choose three.)

A. FMRulesMigrator.exe will migrate VB scripts into business rules in Calculation Manager.

B. There is no utility to migrate VB script rules to Calculation Manager.

C. In Calculation Manager, you can toggle back and forth between the graphical and script mode.

D. In Calculation Manager, you can create one script that can execute against both a Financial Management application and Planning application.

E. To use a business rule, you must validate and deploy the business rule.

ANSWER: A C D

Explanation:

In Financial Management 11.1.1.3, Calculation Manager provides the framework for creating, maintaining, and deploying calculation logic, including support for Financial Management rule migration and both graphical and script-based rule development. **FMRulesMigrator.exe will migrate VB scripts into business rules in Calculation Manager.** This utility supports moving existing Financial Management VBScript calculation rules into the Calculation Manager environment, helping administrators transition legacy rule content into the newer rule-management interface.

In Calculation Manager, you can toggle back and forth between the graphical and script mode. Calculation Manager exposes rule logic through graphical and script-oriented design capabilities, allowing developers to work with the representation most appropriate for the task while maintaining the rule in the Calculation Manager repository.

In Calculation Manager, you can create one script that can execute against both a Financial Management application and Planning application. Calculation Manager is an EPM Foundation component designed to centralize calculation-rule development across supported EPM products, including Financial Management and Planning. This shared framework enables reusable, cross-product rule-development approaches where the script is applicable to both target application types. Oracle's EPM documentation library is available at Oracle Enterprise Performance Management documentation; product and lifecycle information is available from Oracle Enterprise Performance Management.

QUESTION NO: 3

A Classic FM application profile contains which three components? (Choose three.)

A. Frequency

B. Calendar

C. Dimension Associations

D. Language(s)

E. Scenario

ANSWER: A B D

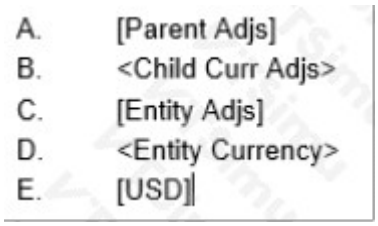
Explanation:

A Classic Financial Management application profile establishes core application characteristics used when the application is created. **Frequency** is part of this profile because it defines the reporting cadence, such as monthly, quarterly, or yearly, and therefore determines the application's period structure. **Calendar** is also a profile component because it establishes the accounting calendar framework used to define the periods and years available for data entry, consolidation, and reporting. **Language(s)** belongs in the application profile because Financial Management supports application-language settings that control the languages available for application labels and translated metadata display.

These settings are foundational configuration choices rather than ordinary member metadata. They are selected as part of defining the application's basic profile and provide the time and localization context needed by the application after it is deployed. Oracle's Financial Management administration documentation describes application creation and the profile settings used to establish calendar, frequency, and language behavior. See the Oracle Hyperion Financial Management Administrator's Guide and Oracle's Financial Management administration documentation.

QUESTION NO: 4

Into which Value dimension member can a journal be entered?



A.

B. Option B

C. Option C

D. Option D

E. Option E

ANSWER: A

Explanation:

<Entity Curr Adjs> is the Value dimension member used for an entity-currency journal adjustment in Oracle Hyperion Financial Management. HFM separates reported entity-currency data from adjustments so that users can review and report the original loaded or entered balances independently of journal activity. When a journal is posted at the entity-currency level, its debit and credit lines are stored as entity-currency adjustments and are included in the appropriate totals during consolidation and reporting.

This design also provides a controlled audit trail: the journal retains its header, line details, preparer, approval status, and supporting documentation, while the resulting data remains identifiable as an adjustment rather than becoming indistinguishable from base entity-currency balances. The adjustment member is therefore the appropriate destination for a journal that affects an entity's local/entity-currency results. Oracle's HFM documentation describes the Value dimension and the use of adjustment members in journal processing and consolidation. See the Oracle Enterprise Performance Management System 11.1.2.4 documentation library and Oracle's Hyperion Financial Management product information.

QUESTION NO: 5

Your source data file has values for 12 months across the columns. What is the best way to load this file into FM with native load functionality?

A. Split the file into 12 different files, each with a single month. Set the POV for the Period dimension = specific month.

B. Define a range for periods in the POV with the following syntax: !Period = January...December.

C. Make sure the column header has the month label; FM will load the column to the correct month.

D. List the months in the POV with the following syntax: ! Period = January-December.

E. List the months in the POV with the following syntax: ! Period = January,December.

ANSWER: B

Explanation:

Define a range for periods in the POV with the following syntax: !Period = January...December. This is the appropriate native Financial Management data-load approach when a source record contains a series of values spread horizontally across columns, one value for each month. The period range declared in the POV establishes the ordered set of Period members that correspond to those consecutive data-value columns. Financial Management can therefore interpret the first value column as January, the next as February, and continue sequentially through December without requiring the source to be separated into individual monthly files.

The three-dot range notation is important because it identifies an inclusive sequence of Period members, allowing the loader to map multiple columns based on their position in the input row. This keeps the file compact and makes the load definition explicit and repeatable. The Period members must exist in the application metadata and be specified in their intended chronological order. Oracle's Financial Management documentation describes the structure and POV directives used by native data load files; see the Oracle Enterprise Performance Management 11.2 documentation library and Oracle Hyperion Financial Management product information.

QUESTION NO: 6

Given the following syntax in a business rule:

```
Sub Calculate()
Hs.Exp "A#DiffCash = " & Difference("A#Cash", "YTD")
End Sub
FUNCTION DIFFERENCE(strPOV,strVIEW)
IF strVIEW = "" THEN
strVIEW = HS.SCENARIO.DEFAULTVIEW ("")
END IF
strPOV = UCASE(strPOV)
strVIEW = UCASE(strVIEW)
IF strVIEW = "PERIODIC" THEN
DIFFERENCE = "("&strPOV &"-"& strPOV & ".P#PRIOR" &")"
ELSEIF strVIEW = "YTD" THEN
DIFFERENCE = "("&strPOV &"-"& strPOV & ".Y#PRIOR.P#LAST" &")"
ELSE
EXIT FUNCTION
END IF
END FUNCTION
```

Identify the subroutine.

A. Calculate ()

B. strView

C. Hs.Exp

D. DIFFERENCE

E. Hs.SCENARIO.DEFAULTVIEW(" ")

ANSWER: A

Explanation:

Calculate () identifies the business-rule subroutine. In Hyperion Financial Management rules, a calculation routine is declared using Visual Basic Script-style syntax, beginning with `Sub Calculate()` and ending with `End Sub`. The code placed within this block is the logic that HFM executes when it performs the Calculate consolidation process. The name `Calculate` is therefore the named subroutine entry point, while the parentheses form part of its declaration syntax.

HFM rules are organized into predefined subroutines that are invoked at specific points in processing. The Calculate subroutine is the standard location for calculation logic, such as assigning values through expressions and applying calculation functions. Identifying the declaration is important because it establishes the execution context for the statements contained in the rule; the calculation code is not merely an independent expression, but code enclosed within the Calculate routine.

Oracle's HFM Administrator documentation describes the rules language structure and the use of rule subroutines in calculation logic. See the Oracle Hyperion Financial Management Administrator's Guide and the HFM documentation library.

QUESTION NO: 7

To which dimension do custom dimensions provide additional detail?

A. Accounts

B. Entity

C. Value

D. ICP

E. Custom dimensions are independent of other dimensions.

ANSWER: A

Explanation:

In Oracle Hyperion Financial Management, custom dimensions provide additional analytical detail for the **Accounts** dimension. An account identifies the financial measure being reported, such as Revenue, Salaries, Cash, or Inventory. Custom dimensions let an application further qualify the account balance by meaningful business attributes, for example product line, customer, channel, region, project, or cost center. This makes it possible to collect, consolidate, and report a balance at a more detailed level without creating an impractically large account hierarchy.

For example, rather than maintaining separate revenue accounts for every product, the application can use a Revenue account together with a custom-dimension member for each product. The resulting intersections retain the standard HFM account structure while supporting multidimensional analysis of the account data. HFM supports up to four custom dimensions, and administrators define their members, hierarchies, and account associations as part of application metadata design. Oracle's HFM documentation describes custom dimensions as dimensions used to provide additional detail for accounts and explains their role in application metadata. See the Oracle Hyperion Financial Management Administrator's Guide and Oracle's Enterprise Performance Management 11.2 documentation library.

QUESTION NO: 8

What are the three characteristics of sub EnumMemberLists ()? (Choose three.)

A. Specifies which dimensions have member lists

B. Names each member list in each dimension

C. Specifies the valid members for a member list

D. Specifies the number of lists for each dimension

E. Is used to create both static and dynamic lists

ANSWER: A B D

Explanation:

The `EnumMemberLists` subroutine is used to enumerate the member-list metadata available in an HFM application. Its returned information identifies the dimensions that contain member lists, so a rule or routine can determine where list definitions are available. It also returns the names of the member lists associated with each applicable dimension. Those names let the calling logic identify and work with the individual lists that have been defined for a dimension. In addition, the routine provides the number of member lists for each dimension. That count is important when processing the returned list information because it establishes how many list entries are present for a given dimension and can therefore be iterated.

These characteristics describe enumeration of existing member-list definitions and their organization by dimension. Oracle's Financial Management documentation covers the Rules language and the metadata structures used by HFM applications, including member lists and their use in application logic. Consult the Oracle Enterprise Performance Management 11.2 documentation library and the Financial Management documentation for the applicable Rules reference and administration guidance.

QUESTION NO: 9

Before you can define an alias for a dimension or member in EPMA, you must _____.

A. Do nothing; the Alias property is created by default

B. Create an Alias dimension and associate it

C. Create an Alias attribute

D. Create an Alias Property

E. Create an Alias Table

ANSWER: D

Explanation:

Create an Alias Property is correct because Enterprise Performance Management Architect stores alternate member and dimension names through alias properties. An alias property defines the metadata field that holds an alternate, user-friendly, or language-specific name. Once the property exists, it can be associated with the relevant dimension or members and populated with the appropriate alias values during metadata maintenance or import. This setup provides a defined destination for aliases in the application metadata model, allowing those alternate labels to be deployed and used consistently by the target application.

In EPMA, properties are foundational metadata definitions: they establish the attributes that can be assigned to objects in the Dimension Library and application views. Creating the alias property first therefore ensures that the alias is recognized as a valid metadata property rather than being treated merely as free-form information. Oracle's Enterprise Performance Management documentation describes the role of the Dimension Library and properties in defining and managing shared metadata. See the Oracle Enterprise Performance Management Architect Administrator's Guide and Oracle's Enterprise Performance Management documentation library.

QUESTION NO: 10

Identify the three true statements about Financial Reporting. (Choose three.)

A. Financial Reporting provides highly formatted, printable reports for FM applications.

B. Financial Reporting provides ad hoc reporting capabilities for FM applications.

C. Financial Reporting document types include reports, books, snapshots, and snapshot books.

D. You can create a single P&L report and run that report for any member in the Entity dimension

ANSWER: A C D

Explanation:

Financial Reporting is Oracle Hyperion's production reporting tool for Financial Management applications. It is designed to create highly formatted, presentation-quality, printable output, including report layouts with grids, text, charts, formulas, headers, footers, and formatting suitable for management and statutory-style reporting. Its repository-based document model supports reports as well as books, which organize reports into distributable reporting packages. It also supports snapshots and snapshot books, which preserve a rendered, point-in-time version of report output for later review or distribution without rerunning the underlying report.

A single profit-and-loss report can be designed as a reusable report by keeping the Entity dimension context dynamic rather than hard-coding one entity member in the report design. At run time, users can select an Entity member through the report's point of view, prompts, or a related member-selection mechanism. This permits the same report definition to display the appropriate financial data for different entities while retaining a consistent layout and calculations. Oracle documents Financial Reporting as a formatted reporting component and describes report objects, books, and output management in its Financial Reporting documentation. See the Oracle Hyperion Financial Reporting User's Guide and the Oracle Enterprise Performance Management 11.2 documentation library.