Retirement & Omni Induction

Version 1.0

**Change Log**

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Comments** |
| Dec 20th 2019 | Sohom Chatterjee | Initial version |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

[Retirement and Financial Services Overview 6](#_Toc11690938)

[2.1 Introduction 6](#_Toc11690939)

[2.2 Types of Plan 7](#_Toc11690940)

[2.3 Eligibility 8](#_Toc11690941)

[2.4 Contribution 10](#_Toc11690942)

[2.5 Vesting 12](#_Toc11690943)

[2.5.1 Overview 12](#_Toc11690944)

[2.5.2 Vesting Credit 12](#_Toc11690945)

[2.6 Forfeitures 14](#_Toc11690946)

[2.7 Account Activity 14](#_Toc11690947)

[2.8 Loans 16](#_Toc11690948)

[2.9 Withdrawal 19](#_Toc11690949)

[2.9.1 Taxation of In-Service Withdrawals 20](#_Toc11690950)

[2.9.2 Regular Withdrawal 21](#_Toc11690951)

[2.9.3 Hardship Withdrawal 21](#_Toc11690952)

[2.9.4 Age 59 ½ Withdrawal 22](#_Toc11690953)

[2.10 Rollover 23](#_Toc11690954)

[2.11 Distribution 25](#_Toc11690955)

[2.11.1 Final Distribution Requirements 26](#_Toc11690956)

[2.11.2 Minimum Required Distribution 28](#_Toc11690957)

[3.0 Omni Plus Application Overview 30](#_Toc11690958)

[3.1 Omni Plus Solution Suite 30](#_Toc11690959)

[3.1.1 FIS & Omni 30](#_Toc11690960)

[3.1.2 What Omni offers 31](#_Toc11690961)

[3.2 Omni Plus Architecture 36](#_Toc11690962)

[3.2.1 Overview 36](#_Toc11690963)

[3.2.2 OmniPlus – Terminologies 38](#_Toc11690964)

[3.3 Omni Plus Database 50](#_Toc11690965)

[3.3.1 Overview 50](#_Toc11690966)

[3.3.2 Omni plus Database Hierarchy 54](#_Toc11690967)

[3.3.3 Omni Plus Record Relationships 55](#_Toc11690968)

[3.3.4 Omni Plus Database Relationships 55](#_Toc11690969)

[3.3.5 Key Indexed Files 56](#_Toc11690970)

[3.3.6 Sequential files 68](#_Toc11690971)

[3.4 Omni Station 69](#_Toc11690972)

[3.4.1 Getting Started 70](#_Toc11690973)

[3.4.2 Working in the Windows Environment 72](#_Toc11690974)

[3.4.3 Components - Pull Down Menu 72](#_Toc11690975)

[3.4.4 Plan Component Menu 73](#_Toc11690976)

[3.4.5 Participant Component Menu 75](#_Toc11690977)

[3.4.1 Fund Component Menu 76](#_Toc11690978)

[3.4.2 Right Click Mouse Functions 77](#_Toc11690979)

[4.0 Omni Script 79](#_Toc11690980)

[4.2 Features 79](#_Toc11690981)

[4.2.1 Transaction-Level Processing Extensions 79](#_Toc11690982)

[4.2.2 Processing Overrides 80](#_Toc11690983)

[4.2.3 Customized Reports and Screens 80](#_Toc11690984)

[4.2.4 Data and XML/HTML Generation 81](#_Toc11690985)

[4.3 Basic Concepts 81](#_Toc11690986)

[4.3.1 Programming Structure 81](#_Toc11690987)

[4.3.2 Omni Script Data 88](#_Toc11690988)

[4.3.3 Intermediate Work Field 91](#_Toc11690989)

[4.3.4 Environment Fields 97](#_Toc11690990)

[Aliases 106](#_Toc11690991)

[4.4 Omni Functions 117](#_Toc11690992)

[4.4.1 Overview 117](#_Toc11690993)

[4.4.2 Object Access Functions 120](#_Toc11690994)

[4.5 Online Access 144](#_Toc11690995)

[4.5.1 OmniScript Menu 144](#_Toc11690996)

[4.5.2 Custom Windows (OCCSCR) 145](#_Toc11690997)

[4.5.3 Custom Report Testing (OCCS99) 147](#_Toc11690998)

[4.5.4 OmniScript Packet Testing (OCDDCALC) 149](#_Toc11690999)

[4.6 Report Writer 150](#_Toc11691000)

[4.6.1 Parm Section 151](#_Toc11691001)

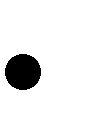
[4.6.2 Text Section 151](#_Toc11691002)

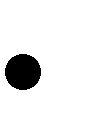
[4.6.3 CALC SECTION 152](#_Toc11691003)

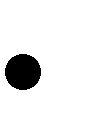
[5.0 Omni Plus Technical Overview 153](#_Toc11691004)

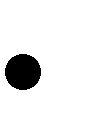
[5.1 System Architecture 153](#_Toc11691005)

[5.2 IO Modules 163](#_Toc11691006)

[ Indexed files I/O modules 164](#_Toc11691007)

[ Sequential Files I/O modules 165](#_Toc11691008)

[ Record Level I/O Modules 165](#_Toc11691009)

[ High Level I/O Modules 166](#_Toc11691010)

[5.3 FIS programming Standard 167](#_Toc11691011)

[Batch Program Naming convention 167](#_Toc11691012)

[Copybook Naming Convention 167](#_Toc11691013)

[5.4 User Exits 168](#_Toc11691014)

[5.4.1 Storage 168](#_Toc11691015)

[5.4.2 OmniScript Text Allowed in Transaction 171](#_Toc11691016)

[5.4.3 OmniScript Suffix Value 172](#_Toc11691017)

[5.5 Control Files 172](#_Toc11691018)

[6.0 Omni Plus Transactions & Records 174](#_Toc11691019)

[6.1 Major Transactions 174](#_Toc11691020)

[6.3 Transaction Trailers 184](#_Toc11691021)

[7.1 Batch Process Overview 189](#_Toc11691022)

[7.2 Unified Processing 195](#_Toc11691023)

# Retirement and Financial Services Overview

## 2.1 Introduction

Benefit programs help employers provide for the well being of their employees and their families. Some benefit plans, such as health insurance and dental insurance, provide for their immediate well being. Other benefit plans promise to provide financial security at retirement. There are a number of ways employers provide retirement benefits for their employees. Most employers will use defined benefit, or pension plans, and/or defined contribution plans to provide retirement benefits. The most common type of defined contribution plan is the 401(k) plan.

Employers have various methods of providing retirement benefits for their employees. Today, more and more employers are asking their employees to share in the responsibility for their retirement. One of the ways employers ask employees to share the cost is through a 401(k) retirement plan. Employees can save for their retirement and also provide an immediate tax savings for themselves and their employer by participating in a 401(k) plan. A 401(k) plan participant decides, within plan limits: how much to contribute, where to invest the contributions, and how long the contributions remain in the plan.

* Highly Compensated Employees

Under ERISA guidelines for plan qualification, a plan may not discriminate in favor of any particular group of employees, such as highly compensated employees (HOEs). Employees are considered highly compensated if:

They own 5% or more of the company, or



They have earned $90,000 or more during the previous year (effective for year 2003). For 2004, the amount will remain at $90,000.

* Qualified Plan

Qualified plans are plans that meet strict IRS requirements. 401(k) plans are a type of qualified plan

## 2.2 Types of Plan

* Overview

Employers who provide employees with a retirement benefit have a choice whether to use the defined benefit (DB) plan design, commonly called a pension plan or the defined contribution (DC) plan design. For many years the DB plan was the primary method used by employers to provide a retirement benefit. In recent years, employers have turned to a popular DC plan method known as a 401(k) plan.

Some employers offer both types of plans while others offer neither. However, most employers elect to offer at least one type of plan to receive the immediate tax & retirement benefit.

* Defined Benefit vs. Defined Contribution

Defined benefit pension plans and defined contribution savings plans are both designed to provide the employee with retirement income. Some of the common differences between the DB and DC plans are listed in the following chart.

|  |  |
| --- | --- |
| **Defined Benefit Plan** | **Defined Contribution Plan** |
| The employer funds it. | Employee and employer contributions are allocated to individual participants. The participant may have to contribute to receive a benefit. |
| The employer bears the investment risk. | The participant bears the investment risk. |
| It is generally difficult to communicate. | It is easier to communicate. |
| There is a definite benefit formula. | The final retirement benefit is unknown. |
| The benefit is expressed as an annuity payable at normal retirement age, usually age 65. | The benefit is expressed as an account balance. |

The following plans are designated as defined contribution plans:

401(k) plans



Profit sharing plans

Thrift plans

403(b) and 457 plans

Employee stock ownership plans (ESOPs)

The 401(k) plan is the most popular form of defined contribution plan used by employers today.

## 2.3 Eligibility

* Overview

When an employer decides to offer a 401(k) plan, it must determine, within ERISA guidelines who will be permitted to participate and how eligibility will be defined. Although ERISA dictates many of the parameters plans must follow it also gives employers some choices when setting eligibility requirements for 401(k) plan participation

Employers may choose to limit eligibility based on:

* + Age: Under ERISA, a plan may limit participation to employees age 21 and over. Plans may choose to allow participation before age 21.
  + Service: The maximum service requirement a 401(k) plan can impose for eligibility to participate, by law, is one year. A plan may choose to impose a service requirement as a cost savings measure, especially in industries where there is high turnover in the first year of employment.

**Note:** DC plans other than 401(k) plans may require two years of service for eligibility; however, employees entering the plan after two years must be 100% vested immediately upon entering the plan.

* + Employment classification: Employers may choose to only allow full-time and/or permanent employees to participate, thereby excluding part-time and/or temporary employees.
  + No eligibility requirements: Some plans may choose to have no eligibility restrictions. In this case, all employees would have the option to begin participating immediately upon hire.
  + Service: If an employer chooses to impose a service requirement, the plan must decide how it will calculate one year of service. ERISA allows a number of methods for calculating when one year of service has been completed including:

|  |  |
| --- | --- |
| 1,000 Hours Method | Elapsed Time Method |
| The most common method is to determine if the employee has completed at least 1,000 hours of service in the 12-month period following the date of hire. | This option does not count hours, only the specific length of service, such as one year from the date of hire. |

* Entry Dates

ERISA allows employers to establish specific times of the year when employees can enroll in 401(k) plans. These are called entry dates. The designated enrollment dates may be quarterly, monthly, or by payroll cycle. Entry dates relieve plans from the possible administrative task of enrolling new participants on a daily basis.

ERISA requires that the use of entry dates may not delay entry into the plan by more than six months after the employee has met the plan‟s eligibility requirements.

## 2.4 Contribution

401(k) plans offer employers opportunities for flexible plan design within the requirements established by ERISA, the Internal Revenue Code (IRC) and the Department of Labor (DOL). 401 (k) plans may be offered in several different ways with several different contribution options. Some 401(k) plans have employee contributions only and others have both employee and employer contributions. 401(k) plans may have only before-tax contributions or they may have both before- and after-tax contributions.

* Employee Contributions

Internal Revenue Code section 401 (a) (17) limits the annual amount of pay a qualified plan may consider in calculating tax-deferred contributions or benefits. This is known as the Maximum Compensation Limit, which limits the amount of pay that is eligible for all contributions. The limit for 2004 is $205,000.

This means that an employee who earns over this amount annually will be recognized as only earning $205,000 for 401(k) plan purposes. The limit is periodically adjusted by the IRS to reflect changes in the cost of living index.

* + - Employee Before-Tax Contributions

Deductions taken from the employee‟s paycheck on a before-tax basis are applied before taxes are taken out of the employee‟s gross income. Only Social Security and Medicare taxes are deducted prior to any before-tax contributions. Before-tax contributions reduce the participant‟s current taxable income. Before-tax contributions are also commonly referred to as pre-tax.

* + - Employee After - Tax Contributions

Deductions taken from the employee‟s paycheck on an after-tax basis are applied after all taxes have been taken out of the employee‟s gross income. After-tax contributions are also commonly referred to as post-tax.

* Employer Contributions

In addition to employees making contributions, some employers make contributions on behalf of the employees. Employers may choose to provide these contributions to encourage employee participation in the 401(k) plan and/or attract and retain skilled employees. Employer contributions are considered before-tax contributions and remain tax deferred until withdrawn. Some employers may choose to provide:

* + A matching contribution based on the contributions the participant makes ;
  + Contributions based on the annual profits of the company;
  + Both types of employer contributions; and  Neither type of employer contributions.
  + Employer Matching Contributions

Matching contributions are usually a percentage of the participant‟s compensation and are tied to the amount the participant contributes.

Employers frequently place a limit on the amount of the match. Only employees who have chosen to contribute to the 401(k) plan receive matching contributions.

* + Employer Discretionary Contributions

Discretionary contributions can be provided in addition to, or instead of, employer matching contributions. One example of a discretionary contribution is a profit sharing contribution.

Discretionary contributions may change each year and are generally tied to annual profits.

Employers may offer discretionary contributions one year and none the next.

Discretionary contributions are made on behalf of all eligible employees even if they choose not to contribute to the 401(k) plan

## 2.5 Vesting

### 2.5.1 Overview

Vesting refers to the ownership in all or a portion of the company‟s matching contributions or discretionary contributions and the earnings on these contributions. A participant is always 100% vested in employee and rollover contributions and the earnings on those contributions. Employer contributions such as matching or discretionary contributions may be subject to a vesting schedule.

### 2.5.2 Vesting Credit

A participant gains ownership of employer contributions and the earnings associated with those contributions according to a schedule referred to as Vesting Schedule, Under ERISA, an employer generally must give credit for a year of vesting for any year in which the employee worked 1,000 hours or more.

Employers may also determine vesting based on elapsed time. Elapsed time does not count hours. An employee receives vesting credit after completing a specific length of service.

ERISA provides minimum standards for two types of vesting schedules: cliff and graded.

However, 401(k) plans may choose to be more generous than the ERISA requirements.

* Cliff Vesting Schedule

In a cliff vesting schedule, the participant becomes fully vested in the employer contributions and associated earnings after a certain period of time. No partial vesting occurs. For employer contributions for plan years beginning n 2002, a participant must be fully vested after completing three years of vesting service.

|  |  |
| --- | --- |
| **ERISA MINIMUM** - **CLIFF SCHEDULE** | |
| *Years of Service* | *Vesting %* |
| Less than 3 years | 0% |
| 3 years or more | 100% |

* Graded Vesting Schedule

In a graded vesting schedule, the participant becomes vested in the employer contributions and associated earnings gradually over a period of time. For plan years beginning in 2002, ERISA requires that a participant must be fully vested after six years of vesting service. The participant must be at least 20% vested after two years, 40% after three years, 60% after four years, 80% vested after five years of vesting service, and 100% after six years of vesting service.

A plan may be more generous and offer a two-year cliff, for example.

|  |  |
| --- | --- |
| **ERISA MINIMUM - GRADED SCHEDULE** | |
| *Years of Service* | *Vesting %* |
| Less than 2 years | 0% |
| 2 years | 20% |
| 3 years | 40% |
| 4 years | 60% |
| 5 years | 80% |
| 6 years | 100% |

A plan may be more generous and offer a four-year graded schedule, for example, where the participant is fully vested after four years of vesting service.

## 2.6 Forfeitures

If a participant leaves the company before being fully vested, the participant may forfeit all or a portion of the employer contributions and associated earnings. The Exclusive Benefit Rule states that forfeited amounts cannot be returned to the employer, but must be used for the plan‟s participants. The employer can either apply these forfeited amounts to reduce future employer contributions or reallocate forfeitures among the remaining plan participants.

## 2.7 Account Activity

Plan representatives and administrators have the ability to routinely process transactions for participants. They have access to each participant‟s account information to process requisitions to:

* Change contribution rates;
* Change future investment directions;
* Move existing balances from one fund to another; and  Respond to information requests.

These transactions are processed based on the plan‟s valuation cycle. Participants generally complete these types of transactions through an automated touch-tone phone system or a web site and then receive a confirmation statement. Participants may also contact a plan representative to assist them with the transaction.

* Contribution Rate Changes

When participants first enroll in a 401(k) plan, they must elect the amount they want to contribute to the plan. This amount is expressed as a percentage of eligible compensation. If the 40 1(k) plan permits before-tax and after-tax contributions, the participant elects how much will be contributed on a before-tax and/or after-tax basis.

After the initial enrollment, participants may wish to change the percentage of compensation they contribute throughout the year. The plan may, however, limit the number of times a participant can make a change in any plan year. Contribution rate changes are typically processed. Only one or two times per month-usually based on the participant‟s payroll cycle.

Changes cannot exceed plan limits. For example, a 401(k) plan may permit a NHCE participant to contribute between 1% and 15% of eligible compensation. The plan may restrict HCE participants to contribution elections between 1% and 8% of eligible compensation.

* Investment Direction

An investment direction election determines where future contributions and loan repayments should be invested. Investment directions generally must be made in increments of 1% and must total 100%. A participant‟s investment strategy may change or the participant may want to respond to market changes, and therefore, redirect where the contributions are invested. The investment direction election has no effect on existing balances in the investment funds.

* Fund Transfers

Fund transfers move balances directly **from** one investment fund **to** another investment fund to another investment fund or funds. The participant determines the percentages to be transferred. Participants may make this election through the automated touch-tone system, web site or plan representative.

* Fund Reallocations

Fund reallocations (sometimes called balance reallocations) require the participant to indicate, in whole percentages, an amount for each of the investment choices. The participant may elect a zero percentage for some of the funds. Theoretically, it is as though the participant takes all the money in all the funds and holds it in her hands. She then has to decide, in whole percentages, where the money is to be invested. The total must equal 100%.

## 2.8 Loans

 Overview

If the plan has a loan provision, participants may borrow money from their 401(k) plan account. A plan loan is an agreement that allows participants to borrow from their account and repay the principal, with interest, back into the account, generally repaid through payroll deductions. If the loan is repaid on schedule, no tax liability is incurred; however, failure to repay will result in default of the loan and tax consequences to the participant.

The Department of Labor (DOL) & Internal Revenue Service (IRS) imposes several plan loan restrictions to ensure that loan provisions are designed in a nondiscriminatory manner with regard to:

Availability;



Interest rates;

The security of the loan and Administrative procedures.

Loan amounts;

The duration of the loan; and Taxes.

Although 401(k) plans are not required to offer loans, an employer may choose to offer one or both of the following loan:

* General purpose loan (also referred to as a general loan)
* Home loan (also referred to as a primary residence loan)

ERISA imposes restrictions on the maximum duration of a loan. Typically, a general purpose loan must be repaid within 5 years. A home loan may be extended beyond the five years period (that is, 5, 10, 15 or 20 years or other period). If participant is refinancing a home loan, the new loan will need to be paid off within the original term of the loan.

The IRS restricts maximum loan amounts. If the plan permits, a participant may have more than one loan outstanding. However, the maximum amount a participant may borrow at any one time is $50,000 of his vested account balance (VAB). Under ERISA, the VAB includes outstanding loan balances (OLBs).

The amount a participant may borrow is determined by using the following formula:

The lesser of:

1/2 (Vested Account Balance + Outstanding Loan Balance)-Outstanding Loan Balance or

$50,000 minus the highest Outstanding Loan Balance in the past 12 months (The highest OLB can come from loans currently outstanding or loans that were paid in full during the past 12 months.)

For example, if a plan sets a minimum loan amount of $1 000, the participant‟s vested account balance must be at least $2000 with no outstanding loans.

The plan may restrict a particular type of contribution (for example, employer match) from being borrowed, yet allow those contributions to be included in the account balance recognized in the maximum loan amount calculation.

The amount available for a participant to borrow is based on the vested value of the account on the day he requests for the loan, since the loan is actually paid from the participant‟s account on the date it is processed. It is likely that the participant‟s vested account balance has changed. However, it is important to note that the 50% rule pertains to the vested account balance at the time of the request, not at the time the loan is processed.

Loan money comes from the before-tax money type, which includes:

* Employee before-tax contributions;
* Employer match;
* Previous rollover monies; and
* The earnings from these monies.

In some cases, after-tax contributions are also included. The plan determines the order of the money types used to provide loan. For example, some plans may require loan amounts to come from rollover money first, then employee before-tax contributions, then employer contributions, and last employee after-tax contributions.

* Loan Repayments

Participants must repay loans according to the terms agreed to under the plan, but no less frequently than quarterly. Generally, loan repayments are made through payroll deductions each pay period with after tax dollars. Exceptions may occur, however. A participant can be on unpaid leave, for example, and make payments from a personal check or electronic fund transfer, if the plan allows. An amortization schedule shows how the portion of each repayment applies to the outstanding interest and principal.

The money is usually repaid according to the current investment direction elected by the participant and is repaid to its original money type even though it is deducted from the participants paycheck after taxes are taken out. Therefore the investment funds the loan is paid from will often not be the same investment funds to which repayment is made. However the repayment is made into the contribution type the loan money was taken from.

* Loan Prepayments

Prepayment of loans, in full, is generally permitted at any time. Some plans may permit partial prepayment. Procedure for prepayment varies from plan to plan.

* Deemed Distribution

Most loans are repaid by deductions from the employee‟s paycheck. Generally, when a participant with an outstanding loan goes on unpaid leave, the loan can be repaid manually by check or electronic fund transfer. If an active participant does not make the required payments, the loan will be considered a deemed distribution after a fixed period of time, usually 90 days.

Participants who have a deemed distribution will receive an IRS Form 1099-R in January of the year following the year in which the deemed distribution occurred. Deemed distributions are not eligible for rollover to a qualified plan or IRA.

If a participant has a loan that is considered to be a deemed distribution, and the loan has not been repaid, he will generally not be allowed to request a new loan until the deemed loan is paid back. As of January 1, 2004, some plans may allow their participants with an unpaid loan considered as a deemed distribution to request a new loan if the participant enters into a payroll withholding agreement or provides the plan with adequate security for the loan. This policy may not apply to all our clients‟ plans. In other words, this policy may vary from client to client.

Suspension of loan repayments during an active military leave of absence will NOT cause the loan to be deemed distributed, even if the leave exceeds a year. The loan must be fully repaid by the end of the original term plus the period of the active military service. .

* Separating from Service with an Outstanding Loan Balance

If the participant separates from service with an outstanding loan balance, most employers require that the loan be repaid, in full, within a specified period of time after separation. If the loan is not repaid within the specified time, it is defaulted and reported as a taxable distribution for the year in which it is defaulted.

* Taxation of Defaulted Loans

Failure to repay a loan after separating from service, according to the terms of the plan results in default of the loan. If the loan is defaulted, the outstanding loan balance is considered a taxable distribution and is thus subject to federal (and state, if applicable) income tax withholding for the year of the distribution. The participant receives an IRS Form 1099-R for the outstanding loan balance in January following the year of default. If the participant defaults on the loan after separating from service, the outstanding loan balance is reported as taxable income.

## 2.9 Withdrawal

Although the primary purpose of a 401(k) plan is to help employees save for retirement, many plans allow participants to withdraw money from their 401(k) plan (MS accounts while actively employed. Employers are not required to make withdrawals from 401(k) plans available to their participants. Therefore, the plans of different employers are structured differently, with some being more restrictive in their withdrawal provisions than others. If in-service withdrawals are permitted from the plan, the money withdrawn from the account cannot be paid back and is subject to taxation if not rolled over.

Actively employed participants can make these three main types of withdrawals from their plan accounts:

 Regular or ordinary withdrawal;  Hardship withdrawal; and  Age 59½ withdrawal.

Each type of withdrawal has specific Internal Revenue Code (IRC) and plan restrictions controlling the money types available for withdrawal and the order of their withdrawal.

### 2.9.1 Taxation of In-Service Withdrawals

Under current federal tax law, participants are not required to pay federal, and in most cases, state and local income tax on contributions and their earnings in a 401(k) plan as long as they remain in the plan. Generally, whenever money is withdrawn or distributed from a 401(k) plan it becomes subject to taxation. Taxation varies depending on the money type withdrawn and the reason for the withdrawal in general

* The withdrawal is subject to a 20% mandatory federal income tax withholding, if not rolled over. The amount of the mandatory withholding may be more or less than the amount the participant owes in federal income tax. A participants actual tax liability for the withdrawal will depend on the participants own tax situation for example exemptions household income deductions etc Withholding for state taxes may also be required.

* Amounts obtained from in-service withdrawals may be subject to a 10% penalty tax if the participant is under age 59 ½. The plan does not withhold for the 10% penalty tax at the time of distribution; participants pay the penalty when they file their taxes.

* An IRS Form 1099-R is mailed to the participant in January of the year following the year in which the withdrawal is taken. For example, for a withdrawal taken in March of 2003, the participant receives a 1099-R in January 2004. This form shows the total amount of the withdrawal, the amount subject to tax, and the amount of federal and state tax withheld and forwarded to the IRS and state agencies.

### 2.9.2 Regular Withdrawal

 Overview

A regular or ordinary withdrawal has the same basic characteristics of all in- service withdrawals:

* + They are generally available only to actively employed participants;
  + The plan sponsor chooses the money or contribution types available; and establishes a withdrawal hierarchy.

 Contribution Types Available

The types of contributions and associated earnings available for a regular withdrawal are chosen by the plan sponsor and may include:

* After-tax;
* Rollover; and
* Employer contributions.

Regular withdrawals do not include participant before-tax contributions or the associated earnings. ESOP, PAYSOP and TRASOP money is usually available for in-service withdrawals.

### 2.9.3 Hardship Withdrawal

* Overview

Participants with immediate and heavy financial need and no reasonable access to money from elsewhere can withdraw certain contributions from their 401(k) plans. The plan administrator determines whether the participant fulfills Internal Revenue Service (IRS) and plan-imposed hardship withdrawal requirements. He reviews the withdrawal request and associated documents and makes the determination based either on the “facts and circumstances” of each case or adoption of the Internal Revenue Code “safe harbor” guidelines or a combination of both.

* IRS Requirements

IRS regulations provide a narrow definition of hardship: The hardship must be caused by the immediate and heavy financial needs of the participant for which other resources are not reasonably available.

To qualify for a hardship withdrawal, the participant‟s request must pass a two-part test.

1. the qualifying need exists; and.
2. no other resources are reasonably available to meet the need.

To determine the “existence of need” employers may use one of two methods:

1. lRS Safe Harbors
2. Facts and Circumstances

### 2.9.4 Age 59 ½ Withdrawal

* Overview

Active participants age 59 ½ and older may take in-service withdrawals from their 401(k) plans. At age 59 ½, 401(k) plan participants may have access to more of their account.

* Contribution Types Available

If the 401(k) plan allows, the participant may withdraw vested amounts from all types of contributions within individual plan limits. All participant contributions and associated earnings, including before tax monies, are made available for withdrawal at age 59 ½ Depending on plan provisions, employer contributions may be restricted.

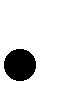
## 2.10 Rollover

* In-Bound Rollovers

Employees may also contribute to a 401(k) plan by rolling over contributions. In-bound rollovers from another qualified plan or conduit IRA. Employees who have contributions in another qualified plan, usually from another employer, may be allowed to roll over these contributions to the current employer‟s plan to continue deferring the contributions from taxes. Beginning with plan years after December 31, 2001, taxable money in an IRA can be rolled to a 401(k) plan without using a conduit IRA.

In-bound rollovers into a qualified plan may only contain taxable contributions and earnings. Beginning in 2002, after-tax contributions can be rolled over. Prior to 2002, after-tax contributions could not be rolled over, though the earnings on after-tax contributions were eligible for rollover since they had never been taxed. If a qualified plan accepts rollover of after-tax contributions, it must agree to account for the after-tax contributions separately.

In-Bound Rollover Procedures



There are no regulations stating that employers must accept in-bound rollover contributions into their qualified plans. As a result, employers may or may not allow in-bound rollovers. Plans are required, however, to offer rollovers-out to other plans.

If rollovers-in are permitted in the plan, the following general procedures may apply:

* + Employers may require employees to be able to participate in the 401(k) plan before rollovers-in will be accepted.
  + Employees may need to complete an in-bound rollover-in request form. The form will ask for the desired investment direction for the amount rolled over.
  + Employees must attach the check from the previous plan or IRA.
  + Employees may need to provide documentation verifying that the in-bound rollover came from a qualified plan or a conduit IRA. Examples of this documentation include: letter of determination or a distribution statement.

* Out - Bound Rollovers

At the time of distribution, the participant has the option to continue to defer taxation by rolling over all or a portion of the eligible account balance into an Individual Retirement Account or another qualified plan A direct rollover allows the participant to move the eligible account balance directly from one qualified plan to another or to an IRA Eligible account balances rolled into an IRA can subsequently be rolled into a qualified plan if it has been held in a conduit IRA (def An IRA that contains only taxable money from a qualified employer retirement plan It is not commingled with other money).

Beginning in 2002 taxable money in an IRA can be rolled to a 401(k) plan without being held in a conduit IRA .Amounts that are eligible to be rolled over include employee before tax contributions and earnings, employer contributions and earnings, rollovers and associated earnings and aftertax contributions and associated earnings. Because of EGTRRA, after tax contributions are now eligible for a rollover. If the entire eligible account balance is rolled over, there are no tax consequences. The account balance is simply moving from one tax-deferred vehicle to another.

If at the time of distribution, participants elect to have their account balance paid directly to them, 20% of the taxable distribution will be withheld for federal income tax withholding. Withholding for state taxes may be required. A rollover option still exists however. Participants may choose to open their own IRA or make a rollover contribution to their new employer‟s qualified plan within60 days of the distribution. Participants may add their own money to make up all or a portion of the entire 20% that was withheld.

The withholding tax paid to the IRS will be combined with all other taxes paid by the participant for the year. Participants may be due a refund when they file their tax return at year-end.

## 2.11 Distribution

 Overview

Shareholders may take final distributions, upon termination, from a stock fund in the form of shares or cash. In certain types of plans, the employer may be required to offer the distribution in the form of shares Upon distribution it is up to the participant to decide whether to take the shares or receive the value of the shares as cash instead. When a participant chooses a share distribution, fractional shares in a participant‟s account are paid in cash.

For example, a participant elects to take a distribution in shares. At the time of the distribution, she has a dollar value in her account of $1,520. The shares are valued at $50 a share. She would receive 30 shares of stock (30 X $50 = $1,500) and $20 in cash.

Participants qualify for final distribution of their vested account balances when they:

* Separate from service (retirement or otherwise);
* Qualify as disabled under the specific plan guidelines;
* Die, in which case the account balance can be distributed to their designated beneficiary.

Final distributions can be paid in several ways. Some forms of payment are required while others are not. Forms of payment include:

* A lump sum payment;
* Deferred payment;
* Installments; and
* An annuity, although these are rare.

When a participant dies, the beneficiary is entitled to payment of the account balance. If the participant is married at the time of death, the beneficiary would automatically be the spouse. If the spouse has signed a valid consent/waiver, the participant may name someone else as beneficiary to receive amounts in the 401(k) plan after the participant‟s death. Spousal beneficiaries have the option to defer distribution until the participant would have been 70Y2. Non-spousal beneficiaries must receive a distribution no later than five years after the participant‟s death. The employer decides which payment options, besides those legally required, are offered by the plan.

### 2.11.1 Final Distribution Requirements

When a participant retires, leaves the company, or becomes disabled the participant can elect to:

* Receive a distribution, or
* Leave the money in the plan until a later date (defer the payment) provided the vested account balance is greater than the plan‟s cash out level (see below).

If the participant leaves the account balance in the plan:

* Taxes will continue to be deferred until the participant receives the money.
* The balance of the account that remains in the plan will share in gains or losses of the investment funds.
* Investment changes may be made according to the rules of the individual plan.
* The participant can request a final distribution at any time in the future.

The forms of payment for final distribution must include a lump sum payment option; however, the plan sponsor does have some discretion. For example, the plan sponsor may or may not offer installment and/or annuity payments.

* Cash out Amounts

When participants request a final distribution, they receive a distribution package, which explains payment options as well as the rollover option. For administrative convenience, ERISA permits 401(k) plan balances at or below a certain level, referred to as the cash out amount, to be paid out after separation from service of employment without the participant‟s consent. The current threshold is a vested account balance of $5,000 or less. Plan sponsors have the choice to implement this option.

If the vested account balance is greater than $5,000, IRC regulations require the participant‟s consent before making a lump sum distribution. Without the participant‟s consent, the distribution will be deferred.

If the vested account balance is $5,000 or less, then the lump sum payment may be distributed without the participant‟s consent. The participant must be given an opportunity to elect a direct rollover of the eligible account balance.

* Payment Options

If participants choose to receive a distribution from the plan, they may be able to choose from various payment options, which may include:

**Lump sum paid to the participant** - the participant‟s vested account balance is distributed to the participant. The plan administrator automatically deducts the mandatory 20% federal income tax withholding from the distribution. The distribution may also be subject to the 10% penalty tax if the participant is under the age of 59 ½ and not otherwise exempt. Withholding for state taxes may also be required.

**Lump sum rolled over to another qualified plan or IRA** - the participant‟s vested account balance is distributed to the qualified plan or IRA. There is no federal income tax withholding.

**Installment** - a specific number of payments of approximately equal value. The payment amount is calculated based on the number of payments requested. An example of an installment payment option would be a provision, which allows participants to elect up to 20 annual installments. Installment payments of ten years or less are eligible to be rolled over.

**Annuity** - a series of regular payments, guaranteed to continue for a specific time, usually the participant‟s lifetime. The payment amount is calculated based on the life expectancy of the participant or the participant and the joint beneficiary (using IRS actuarial tables). Annuities are rare in 401(k) plans.

If the participant defers payment, the participant may come back at any time and request a distribution. The ability of a participant to leave money in the 401(k) plan account depends on whether it is above the cash out level ($5,000). If the balance of the account is above the cash out level, a participant cannot be forced to take a distribution. Ievel, a participant cannot be forced to take a distribution.

 Taxation of Final Distributions

As with withdrawals, final distributions are subject to the mandatory 20% federal income tax withholding, and the 10% penalty tax if the participant is under age 59¼ and not otherwise exempt, unless the distribution is directly rolled over Withholding of state taxes may also be required.

The 10% penalty tax does not apply if the final distribution is made

* After age 59 ½;
* Because of separation from service at or after age 55 (separation includes leaving the company for any reason or retirement);
* Payable in the form of periodic payment;
* On account of a QDRO; or
* For certain medical expenses (those that are deductible for income tax purpose; unreimbursed medical expenses must exceed 7.5% of adjusted gross income.

### 2.11.2 Minimum Required Distribution

* Overview

Age 70 ½ minimum required distributions, more commonly known as, Minimum Required Distributions (MRDs), mandate certain 401(k) plan participants receive certain minimum payment from their account balances beginning at age 70 ½. MRDs apply to all qualified retirement plans, including 401(k) plans and pension plans. MRDs also apply to IRAs.

* Requirements

The IRC requires participants to begin receiving MRDs from a 401(k) plan no later than April 1 following the year in which the participant reaches age 70 ½.

MRDs are paid annually. The first payment must begin by April 1 of the year following the year in which the participant turns 70 ½ Subsequent distributions must be paid by December 31 of each year. Consequently, there may be two payments received in the first year.

If the participant fails to receive an annual minimum required distribution during the required time frame, the individual may incur a tax penalty equal to 50% of the amount that should have been distributed.

Beginning in 1997, active participants in 401(k) plans who are over age 70 ½ can delay commencement of MRDs until the later of retirement or separation from service.

Note, this does not apply to IRA participants or 401(k) plan participants who own 5% or more of the company. The IRS requires them to begin receiving MRDs at age 70 ½.

The MRD calculation is based on the participant‟s life expectancy, or the joint life expectancies of the participant and designated beneficiary based on actuarial tables set forth by the IRS.

 Taxation of 70 ½ Minimum Required Distributions

The mandatory 20% federal income tax withholding does not apply to MRDs. Instead, the recipient is required to make a federal income tax withholding election on IRS Form W-4P. In some cases, if the participant fails to make a withholding election before the annual payment is made, the plan withholds federal income tax based on a default election. An MRD is not eligible for rollover. Participants can no longer defer paying taxes on this money once the MRD has been paid to them.

# Omni Plus Application Overview

## 3.1 Omni Plus Solution Suite

### 3.1.1 FIS & Omni

FIS is one of the world‟s leading software and technology services companies. FIS has more than 20,000 employees and serves more than 25,000 customers in more than 70 countries. FIS provides software and processing solutions for financial services, higher education and the public sector. FIS also provides disaster recovery services, managed IT services, information availability consulting services and business continuity management software. With annual revenue of about $5 billion, FIS is ranked 434 on the Fortune 500 and is the largest privately held business software and IT services company.

Headquartered in Wayne, Pennsylvania, FIS is comprised of four businesses - Availability Services, Financial Systems, Higher Education and Public Sector - that provide technology services and infrastructure, and software and processing solutions.

FIS offers a Benefit Administration product to help simplify your workload and amplify the value you provide. Recordkeeping, tax reporting, payment processing and trade management are just a few of the tasks measurably improved by FIS programs.

Some of FIS‟s achievements in the Benefit Administration industry include:

* Processing over 30 million retirement accounts on FIS systems
* Managing over 3.5 million bond holders with $10 billion in assets
* Serving the software needs of 9 of the 10 top life insurance carriers in the U.S.

FIS‟s retirement plan systems support many plan types and aspects of benefit administration ranging from record keeping and processing of contributions and payments, to tax reporting and trade management. OmniPlus is an integrated suite of products providing solutions for retirement plan administration.

Business areas supported by OmniPlus:

* **Benefits Administration** – DC & DB Plans Administration ranging from recordkeeping and processing of contributions and payments.
* **Insurance -** Insurance solutions are used by customers in life insurance, property and casualty, and reinsurance for activities ranging from front-office sales to back-office accounting.
* **Banking & Capital Markets**

### 3.1.2 What Omni offers

Omni provides a set of tools to help your business manage pension recordkeeping and administration functionality in the global landscape. By offering global, Web-based access to plan participants and sponsors, a high level of automation in trading transactions, benefit disbursement, and a commitment to self-improvement, Omni helps your business capitalize on the trend towards increased asset growth in pension investing worldwide.

* Available as a licensed installation, on a software-as-service/ASP basis or through full-service outsourcing
* Customizable to meet your exact needs with Scripting language features
* Automatically reconciles trading transactions and benefit disbursements

The most popular solution packages that Omni provides to customers are as follows:

 Omni Defined Benefit

Omni Defined Benefit offers a multi-tool approach to benefits administration including annuity calculating and tracking, Web account access, retirement forecasting based on user-provided variables, and tax maintenance. Defined Benefit is available in tandem with Defined Contribution. At present, only FIS offers a defined benefit participant recordkeeping solution that can be integrated with its defined contribution solution.

**Features**

* Integrates with Defined Contribution operations to simplify data processing
* Presents choice of platform and delivery method
* Customizes easily to help achieve your business requirements
* Develops participant- or management-level reporting using Scripting function

**Benefits**

* Reduces processing costs when combined with Defined Contribution
* Calculates tax maintenance and retirement forecast values with included applications  Saves time and money when participants self-direct to accounts online

 Omni Defined Contribution

Omni Defined Contribution offers a multi-tooled approach to retirement plan administration and recordkeeping solutions. The system helps provide your organization with global Web-based access to records alongside of the abilities to setup and administer annuities, and reconcile deposits. Every screen within the system is customizable according to your user preferences.

**Features**

* Offers powerful processing capability and connectivity
* Provides immediate access to administrative functions
* Maintains compliance automatically with all applicable taxes and regulations
* Provides automated call-center processing for plan participants
* Provides comprehensive security subsystem for meeting e-commerce requirements

**Benefits**

* Positions your business to take advantage of growth opportunities and markets
* Grants access from disparate locations to help enable work from anywhere in the world  Streamlines operations while helping you to improve customer service

 Omni Payments ( Omni Pay )

Omni Payments is a solution for managing your check writing, pension disbursement and taxreporting activities. It tracks and reconciles issued checks and supports maintaining compliance with applicable taxes and regulations. Access to the system is easily achieved; both plan sponsors and participants can view their account information via the Web from different portals.

**Features**

* Automates check disbursement and reconciliation
* Manages deductions according to your preferences and business needs
* Supports multiple delivery methods including checks, wire and EFT
* Provides sponsor-limited Web access to account information for plan participants
* Provides comprehensive market-tested online security
* Accesses all of your accounts in real-time through an internet connection

**Benefits**

* Eliminates live phone support expenses through online accessibility for plan participants
* Reduces costs with remote accessibility via Web of shared IP address
* Frees up time and expense due to cost-saving automation and Web access
* Manages different types of payments regardless of processing volume

 Omni Trade and Reconciliation

Reduce your trading costs and operational risk while maintaining control of your order flow and transaction history with a straight-through processing enterprise solution. Omni Trade and Reconciliation sends orders to trading systems for execution and settlement through the SGN interface link. SGN then returns transaction details to Omni Trade, which you can later access for historical data and reporting.

**Features**

* Automates trades, manages investment balances and records trading activity
* Self-enforces quality control by identifying and correcting transaction errors
* Trades at a global or plan level with Omnibus trading capabilities
* Reconciles trades with Share Control tool
* Maintains compliance with Rule 22c-2

**Benefits**

* Reduces your trading costs and operational risk through automated trading transactions
* Helps ensure trading integrity with overlapping levels of fail-safes  Generate reports and records for swift billing and monitoring

 Omni Connect

OmniConnect is an Application Programming Interface (API). It provides an easy to program interface to the OmniPlus server application. OmniConnect/EJB can help simplify construction and maintenance of enterprise applications by basing them on standardized, (not proprietary), modular components

**Features**

OmniConnect is a collection of stateless session EJB's that communicates with your OmniPlus server to allow your application:

* obtain participant balance information, including as of balances
* update participant indicative information
* add, change, view, delete, trial post or post OmniPlus transactions
* execute OmniScript "calculator" text files
* Inquire or update OmniPlus master records (PH, FC, PL, etc.)
* obtain lists of OmniPlus master records (PH, FC, PL, etc.), applying various server side

filters

* Work with OmniPlus text files, including "Custom Screen" text files

**Benefits**

* Build Web, Voice, Rapid Data Entry, or Customer Service applications which use OmniPlus data and business objects
* Build Servlet java.sun.com/products/servlet or Java Server Pages java.sun.com/products/jsp applications which use OmniPlus data and business objects
* Build a web service applications java.sun.com/web services which use OmniPlus data and business objects
* Integrate OmniPlus data and business objects into existing j2ee applications

 Omni Trust

Omni Trust comprises a comprehensive system environment, providing a series of interrelated and interdependent functions and capabilities. It controls the databases and Master files for all accounts, their holdings and history, as well as centralized name and address list. The security database is cusip-driven, which is currently the industry standard. Security pricing, broker information and tax information are kept current through downloading of vendor files. Omni Trust provides the online entry of transactions, online enquiry into those transactions and immediate report generation. Centralization allows for consolidate Tax reporting, the delivery of Private Banking products such as Performance Measurement, Client Statements and Tax reporting to all clients, and customization of Client Statements for individual recipients.

Omni Trust supports following kind of accounts:

* Trust Accounts
* Custody Accounts
* Advisory Accounts
* Investment Management Accounts
* Brokerage Accounts
* Reporting Accounts

Other products which comes under Omni solution suite comprises of:

**eReports -** Real-time interactive reporting **Omni Annuity** - Annuity administration.



**OmniCash**  - Match and balance plan deposits to participant transactions. **Omni Forecaster -** Projects future account balances and future contributions.

**OmniScript -** Tool for developing custom rules and logic.



**OmniSDA**  - Participant level trading.

**OmniVoice -** Automatic, interactive telephonic access to participant account information.

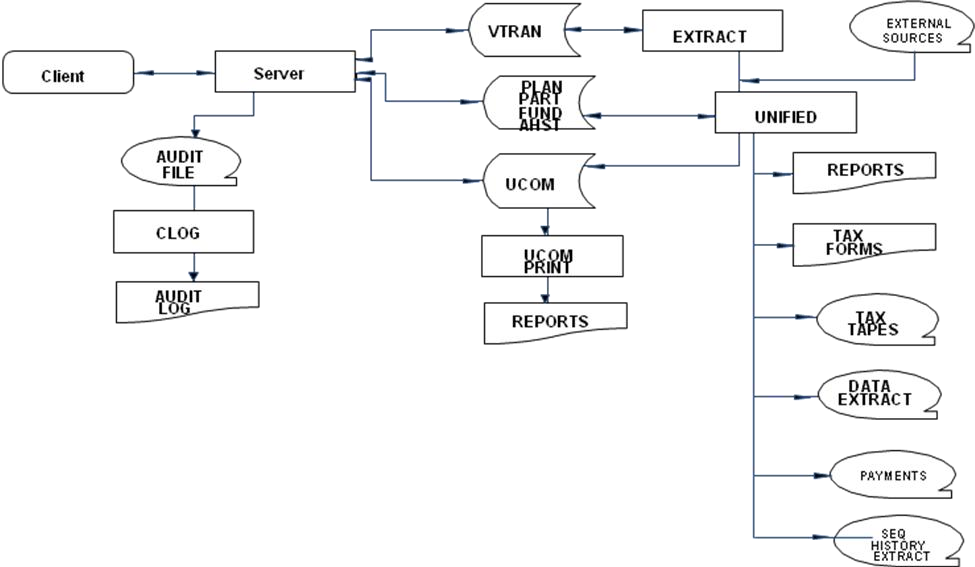
## 3.2 Omni Plus Architecture

### 3.2.1 Overview

OmniPlus is FIS EBS‟s COBOL / C++ client server based software package for participant recordkeeping primarily for Employee Benefit retirement Plans.

 Features

* It is a true Client/Server application.
* Presentation of information is via a GUI workstation.
* Server processing occurs on the Application Server (Mainframe, UNIX, Windows NT, etc.).
* The system is transaction driven.
* The system has batch and online components.
* Most update activity occurs in batch.
* The online component is used for inquiry, transaction maintenance, limited updating, and financial transaction posting.
* It is a multi-tiered & scalable application which allows high volume processing of almost any type of contributory pension plan around the world.
  + Input to Omni system will include :
* Payroll extract data defining dates and deduction amounts
* Transfer, withdrawal, termination, etc. transactions reflecting requests by participants
* Earnings information from trusts
* Conversion data/new plan setup
* Report requests



* + Output from Omni will include :
* Audit logs showing effect of transactions
* Requested reports showing master file status, etc.
* Trust reconciliation reports
* Participant statements
* Year-end tax tapes (W2P, 1099, etc.)
* Interface to check writing systems

### 3.2.2 OmniPlus – Terminologies

The various terminologies used for the OmniPlus components are given as follows:

 Plan

The term “Plan” stands for various “Retirement Plan Models”. Plan models includes, but are not limited to:

1. Traditional qualified employer-sponsored plans for:

* + Corporate
  + Governmental
  + Non-profit organization

1. Non-qualified plans such as:

* + Executive pay
  + ESOPs
  + Stock options

1. Individual-based retail plans
2. International pension plans
3. Many hybrid structures
4. Provident funds

**Note:** Every Plan has a “Plan Sponsor”: Generally these are the “Employers” or “Employers forming a Group”

Functionally OmniPlus categorizes “Plans” into two types:

1. **Local Plan:** “Local Plans” are nothing but the “Individual Retirement Plans” replicating a Retirement Plan Model in Omni.

1. **Global Plan:** Global Plan” stores information that all or most plans within that fileset share in common.

For E.g. the “Global plan” contains the Daily prices, Control files & OmniScript for all related plans. “Global Plans” are traditionally Plan#000001, although other plan numbers may be used.

* + Fileset

“Sets of Files” forming “physical directories” used to contain plan related data. OmniPlus File sets are the physical directories and files forming the OmniPlus database and are used to contain similar set of plan related information.

* + Master Files

Set of “VSAM files” forming System Database - contains all the information required to process System resident plan components.

“8” Master Files that make up the Omni Database.

* + - * VPLAN
      * VFUND
      * VPART
      * VAHST
      * VUCOM
      * VTRAN
      * VMUPD
      * VPASS

* + Data Elements

Data Elements refers to a named “Field” of information of a “Master File Record”. It is actually the variable or field in the Omni master VSAM file.

Data Elements are identified by:

* + 1. “Data Element Name”
    2. “Picture” – (Type and Size Definition)

1. **Data Element Name**: Comprises of “2” Parts:

* + - * Data Element Number: A “3” digit numeric identifier for a Data Element on a specific master File.
      * Data Element Prefix : A “2” Letter Alphabetic prefixed to the Data Element number indicating the applicable File/Record type.

For E.g. PL011 is the Plan Number where “PL” is the Data Element Prefix & 011 is the Data Element number.

1. **Data Element Picture:** The “COBOL Picture clause” that defines the type and number of characters those are stored in the data element field. e.g. S9 (9) V99 Comp-3

**Data Element Dictionary:** List of all Omni data elements which it reserves for specific use across its book of business. It contains the Data Element Descriptions.

**Data Element Description:** Describes the “Purpose” and “Use” of the specific Data Element in question.

E.g. PF110 - “Net Contribution Value – Current”

Data Name: PF-NET-CNTRBS

Picture : S9 (9) V99 Comp-3

 Folder

Folder is much like the “Windows Folders” it is a “Container” containing a group of “Similar” or “Related” Transactions.

**Features**

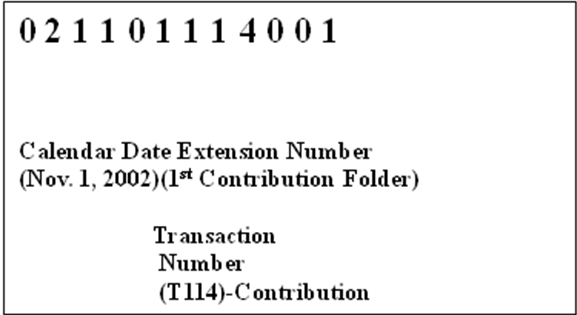


* Used to organize, maintain, and manage transactions.  Created with a name (naming is very important)  Detail transactions are added to it.
* Maintains critical system audit information - the UserID that created the folder, the last time it was updated, the status, etc.
* Maintained as a Record in the VTRAN File with pointers to the Transactions it contains.  Has a system defined lifetime.

**Folder Naming Conventions**



* Generally 12 numeric characters long. Can be 19 characters for “Protected” folders.  12 Characters based on the format: YYMMDDTTTNNN
* Protected folders has the word “PROTECT” in the first “7” positions.(System does not purge Protected folders)



 Transaction

OmniPlus is a transaction-based system. The efficient handling of transaction data is important to daily administrative processes. The OmniPlus Transaction Management System (VTRAN) provides extensive functionality to efficiently manage the daily flow of transactions. Transactions flow into the OmniPlus System from numerous sources at different times of the day and with different volumes of activity. Transactions in Omni are basically Instructions to Omni for Plan processing in the form of a predefined Template.

* Transactions are “Stored in” and “Processed from” the VTRAN Master File.

* Each transaction processed will affect one or more “DATA ELEMENTS” that are contained within the database.

* The primary sources Transactions are:
  + - 1. OmniStation
      2. Voice Response System
      3. Internet
      4. External File (created externally and batch loaded to the transaction system.)

**Transaction Naming Conventions in Omni**  Transactions names comprise of a “3” digit number Prefixed by the Letter “T”.



**e.g. : T010.**

**Transaction Types**



Transaction can be classified functionally into “4” Basic categories.

* **Accessory :** Provides additional information to the system regarding other transactions  being processed simultaneously.

* **File Maintenance:** These transactions update specific data elements with specific values

* **Financial :** These transactions are the Credits and Debits to the Master File and the most important type of transaction.

* **Reports:** The reporting transactions only report what is currently on the Master file.

* + - * + **Transaction Status Codes**

Transaction status codes essentially identify the transaction as active or inactive. Only “Active” transactions are ready to be updated (posted) through GET ACTIVE processing, whereas transactions.

“Active” is the Default status after the folder/transaction is newly created. Below is the list of Transaction Status & what they mean.

|  |  |
| --- | --- |
| **Transaction Status Code** | **Meaning** |
| **A** | Transaction is **A**ctive |
| **D** | Transaction is **D**eferred  [Any transaction processed with a future trade date will assume a Deferred status code. Only transactions with a trade date less than or equal to the current date will be eligible for current processing.] |
| **E** | **E**rror Transaction  [When submitted to an update/post through GET ACTIVE processing, any transaction which yields one or more errors will reject. The rejected transactions assume an Error status code.] |
| **H** | Transaction is on **H**old  [Users can purposely place a transaction on hold status for a variety of reasons. One such scenario is to build a transaction before it is  intended to be processed and to place the transaction on hold: (a) so  it does not inadvertently get picked up in a system-wide batch job  and, (b) until the individual is ready to use it.] |
| **U** | Price **U**navailable for a particular fund.  [When the prices for a fund is not available the corresponding folder is marked as U.] |

* + - * + **Transaction Layouts**

Each and every transaction being a Template has a independent, specific Layout. These are described in the System Cross Reference Manual supplied with the Product Documentation from FIS. Each OmniPlus transactions can be generated or viewed as 80 column positional records.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columns** | **Length** | **Picture** | **Field Name (Screen Prompt)** | **Req/Opt** |
| 1-3 | 3 | X(3) | Tran-Code | Req |
| 4-5 | 2 | X(2) | Seq-Code | Req |
| 6-8 | 3 |  | Not Used | N/A |
| 9-18 | 10 | X(10) | Type-Process (Process Type Option) | Req |
| 19-26 | 8 | 9(8) | Date (Run Date)1 | Opt |
| 27 | 1 |  | Not Used | N/A |
| 28-31 | 4 | X(4) | Location (Division/Subdivision) | Opt |
| 32-41 | 10 | 9(7)V9(3) | Amount (Total Contribution) | Opt |
| 42-61 | 20 | X(20) | Comment (Notes) | Opt |
| 62-80 | 19 |  | Not Used | N/A |

* + - * + **Transaction Header Layout:**

Following is the format for the standard header preceding each transaction‟s layout cards.

**The $TIHDR Card:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columns** | **Length** | **Picture** | **Field Name** | **Req/Opt** |
| 1-3 | 3 | X(3) | Tran-Code | Req |
| 4-5 | 2 | X(2) | Seq-Code | Req |
| 6-8 | 3 |  | Not Used | N/A |
| 9-15 | 7 | X(7) | Record Type ($TIHDR) | Req |
| 16-21 | 6 | X(6) | Plan Number | Req |
| 22-38 | 17 | X(17) | Participant ID | Opt |
| 39-41 | 3 | X(3) | Fund ID | Opt |
| 42-49 | 8 | 9(8) | Trade Date | Opt |
| 50-55 | 6 | 9(6) | Trade Time | Opt |
| 56-60 | 5 | X(5) | Currency Code | Opt |
| 61-80 | 20 |  | Not Used | N/A |

**T002:**

A Special, Accessory transaction automatically generated by the Online Transaction System. It provides information to the system to perform various utility on the folder, entered in the „File ID‟ field. Utility functions performed are as entered in the “Process Type” field.

For e.g.:

* **ADD :** Adds the transactions following this transaction to the file entered in the „File ID‟ field.
* **DEFINE** : Creates a new Folder and stores the transactions following this transaction in the Folder entered in the „File ID‟ field.

**T002:** “3” Properties/Fields.

* **Process System:** This field contains “Installation dependent” name of the “Process System” on which the folder, entered in the „File ID‟ field, is to be processed.

* **Process Type:** This field indicates the type of process applicable to the processing of the transaction.
  1. g.: ADD, GET ACTIVE, EXTRACT, SUBMIT etc.
* **Process Step:** This field indicates the Processing Step of the “Unified Job” in which the Folder, entered in the „File ID‟ field, is to be processed.
  1. g.: PF, PT, M1, M2. Valid only if the „Process Type‟ field is “SUBMIT”, “UNSUBMIT”, or “GETACTIVE”.

**Transaction Trailers**



Each transaction in OmniPlus can stand alone as a business task. OmniPlus also allows functionality from other transactions to be merged into related transactions to form business events.  [Transaction Trailers](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/SystemCrossReferences/TransactionTrailers.htm) are used to merge functionality from multiple transactions into a base transaction to perform an entire business event. The base transaction becomes the business event, and all history generated from the base transaction and Transaction Trailers are stored together.

[Transaction Trailers](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/SystemCrossReferences/TransactionTrailers.htm) are appended at the end of the base transaction, and identified as “Screen ID”. The Screen ID is a 4 character acronym that identifies a Transaction Trailer, and it is preceded with a $ (Ex. $INST is the Installment Trailer). The following is an example of an [InterParticipant Transfer (T366)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/Inter-ParticipantTransfer(T366)/Inter-ParticipantTransfer(T366).htm) where the [Participant Enrollment Trailer (PTEN)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/TransactionTrailers/PARTICIPANTENROLLMENTTRAILERCARD(PTEN)/PARTICIPANTENROLLMENTTRAILERCARD(PTEN).htm) has been used to establish a new participant to transfer to.

00200 $TIHDR XFT366

00201 DEFINE TRIAL.T366

1. $TIHDR XFT366000000016 20050419
2. 10 20050401

36602

1. 111223333 \*\*\* 2 10000
2. $PTEN0
3. $PTEN1 011Name Examples
4. $PTEN1 0301
5. $PTEN1 289101 Test Street
6. $PTEN1 293Test City
7. $PTEN1 294AL
8. $PTEN1 29535244
9. $PTEN1 05019500101>

**Propagation of Transactions**



An OmniPlus system function that permits transactions to be copied for re-use in regular, ongoing processing it can be classified in 2 ways:

a) **Basic Propagation:** A Transaction or Folder created in the Global Plan will be copied to some or all plans in the same fileset.

**Steps:**

* + User creates a folder in the Global Plan and codes the transactions to be propagated.
  + Within the Global Plan, a VTRAN Command “Post” is executed to propagate the folder/transaction to designated local plans in that fileset depending on the settings of PL

Data Elements “PL941, PL947, PL948 and PL949”.

* + The original folder (as coded in the Global Plan) is copied into the local plans with all transactions intact.

b) **Periodic Propagation:** A Transaction or Folder will be regenerated over time so it can be processed on a regular, on-going schedule.

**Steps:**

* + The user creates a Template Folder and codes the transactions in it in the “Local Plan”.
  + The Properties Tab of each transaction is coded to set up VTRAN Detail Data Elements and the Propagation Tab of the Folder Detail (VTFHDT) screen is coded to set up VTRAN Header Data Elements.
  + A T002 transaction runs to extract transactions from all Template Folders.
  + The Template Folder and its transactions are replicated. The copy version is immediately ready for processing.

**GET ACTIVE Processing**



A “Process Type” to “extract” all transactions identified by the “File ID Pattern”, that have an Active (/A) status and are not in error (/E), on hold(/H),or deferred(/D) status.

 Textfiles

Textfiles are used to store many different kinds of OmniPlus system information, and can also be used to store user defined information. Textfiles are powerful tools that add flexibility to processing, reporting, and control of the OmniPlus system. They are used to store OmniScript, which extends the functionality of OmniPlus without the need for traditional code. In addition, text files are used to maintain user-defined fields and system control files.

OmniPlus text files are “containers” used to support the storing and access of multiple lines of text. Each line in a textfile can be up to 200 characters. Up to 9999 lines per textfile is supported.  **Naming Convention**



* + A 2 byte Prefix, identifying the file type (e.g., 'CT' – Control information)
  + A 1-25 byte File Name (Predefined for OmniPlus system files, variable for user defined

files)

**Use of Textfiles**



Textfiles are used for:

* + Storing of OmniScript used with certain transactions to meet plan specific rules
  + Storing of print parameters and special logic for customized participant reports, statements and output
  + Storing of system-control files
  + Creation and maintenance of user-defined fields (e.g. DED.PL)
  + Storing of JSUBs (Online Job Submit Feature of OmniPlus)

**Text Files : OmniScript**



OmniScript is a programming language used to customize OmniPlus in order to meet specific processing and reporting requirements.

OmniScript allows:

* + Customization of the OmniPlus system to meet plan- specific rules of transaction eligibility, processing, and amounts available.
  + Processing of custom edits and validations at the plan or participant level.
  + Creation of custom reports, statements, and data extracts to meet the needs of the record keeper, client, trust, or other recipient.
  + Adjustments and maintenance of database records and fields.
  + Generation of formatted XML or HTML web pages.

 UCOM : User Controlled Output Manager

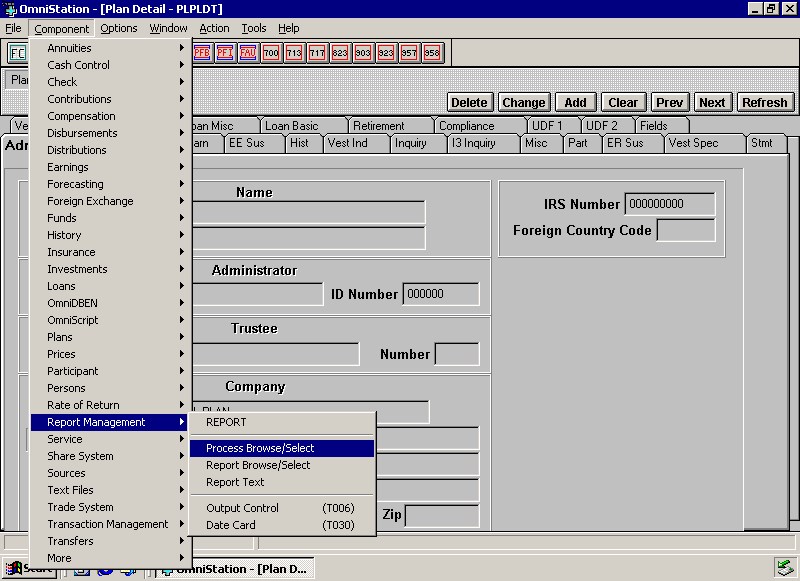
**Features**



* + UCOM serves as output storage and print manager facility.
  + Can be accessed from OmniStation using the Report Management facility.
  + Stores reports online so which makes it possible to inspect the reports before printing.
  + Reduces print charges by printing only the selected reports.
  + Allows to specify the number of copies for each report.
  + Reprints reports immediately (no reprocessing).
  + Provides an online overview of jobs processed and reports each job generated.  Provides password access to reports on a plan by plan basis  Allows control of report retention via the Archive Date facility.
  + Reduces excess reports and paper storage problems.

**UCOM Access Screen in OmniStation**





## 3.3 Omni Plus Database

### 3.3.1 Overview

The database of the OmniPlus comprises of VSAM files called as Master files. Each master file contains groups of logically related data elements identified by a record type. The 8 VSAM files which constitute the Omni database are:

1. **PLAN –** store all plan related data & refers to type of DC retirement plans
2. **FUND –** store all fund related data, fund wrt OMNI can be defined as combination of source of money and the investment where the money is allowed to grow
3. **PART** – store details of participants (Participant Header, Participant Fund & Participant Source) and a participant means any employee who is enrolled with any retirement plan
4. **AHST** – store history of all the activities i.e. financial or non-financial, contains primarily base records, investment control records, source control records, plan locator headers and loan fund records
5. **TRAN** – store all financial and non-financial transactions which are nothing but interactions with the Omni system
6. **MUPD –** Multiple User update files contain all the rules written in OmniScript which are stored as text files
7. **UCOM** – store the data related to reports generated
8. **VEXTK** – store all the file maintenance records

 Segmentation

* All Master Files are segmented into eight parts for reducing the access time (except the SYS Master File).
* Segmentation also helps to reduce the time span of Batch Cycle Process.
* Global Plan or Cross Plan resides in segment 01 and contains common attributes of all plans.

* + Plan Master File ( PLAN )
* Plan Master File contains plan related information.
* Every plan is uniquely identified by a 6 digit alphanumeric plan Number.
* Segment 01 of Plan master file contains cross plan (PLAN NO: 000001).Information in the Cross Plan is common to all the plans.

* + Participant Master File (PART )
* PART Master File contains Participant‟s data.  PART Master File holds two record types :

* + - **Participant Header (PH):** It holds participant personal information (like SSN, Name, Address, Contact no etc).
    - **Participant Fund (PF):** It holds Participant level FUND information.

* + Transaction Master File ( VTRAN )
* TRAN master File contains Transaction Folders.
* Transaction Folders have 2 main types of records

**VTH –** VTRAN Transaction Header. **VTD –** VTRAN Transaction Detail.



* + Fund Master File ( FUND )
* Fund master file holds information about all the funds at plan level.
* Each fund is identified by a 3 digit alphanumeric FUND ID which is unique at plan level.
* FUND ID composed of INVESTMENT ID (1st two digit of FUND ID ) and SOURCE

ID(last digit of FUND ID).

**INVESTMENT ID** identifies investments to a particular fund. **SOURCE ID** identifies the source of money.



* + History Master File ( AHST )
* Activity History is the history master file which holds the records of successfully posted transactions.
* Segment 01 of history file holds the price records.
* There are various record types in AHST. Some of them are BR, BT, SC, IC, LH etc which will be discussed later in detail.

* + MUPD Master File ( MUPD )
* MUPD contains text files which contains the business rules and logics specific to the Plan. MUPD indicates **M**ultiple **Upd**ates to the VSAM file.

* + BRAR Master File ( BRAR )
* BRAR stores archived history for BR records.
* The archived data can be restored in AHST from BRAR as per need.

* + EXTK Master File ( EXTK )
* EXTK contains file maintenance records(Modification of data elements will be reflected

* + UCOM Master File (User Controlled Output Manager)
* UCOM Master File is used to store various reports.
* It is used to print and review reports.

* + SYS Master File
* This Master file is used to locate in which File set and Segment a plan resides.
* It holds the PLLC (Plan locator) record.
* PLLC record holds the Fileset no. and the Segment no. for a specific Plan.

Example PLLC0666666 02

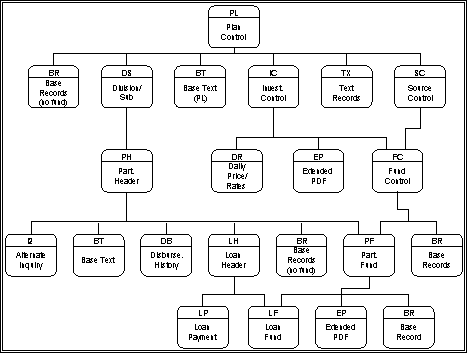
Where

Position 5 **Value 0** - File Set

Position 6 – 11 **Value 666666** - Plan Number

Position 12-13 **Value 02** - Segment Number

### 3.3.2 Omni plus Database Hierarchy

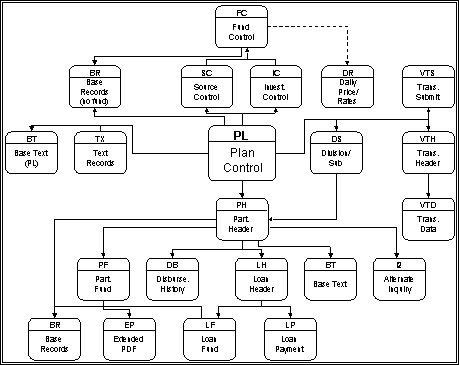


### 3.3.3 Omni Plus Record Relationships

|  |  |
| --- | --- |
| **One** | **to Many (Direct)** |
| **One** | **Many** |
| PL | IC.SC,TXT,DS,BR,VTH,BT |
| PH | PF,LH,DB,BR,BT,I2 |
| IC | FC,DR,EP |
| SC | FC |
| FC | PF,BR |
| LH | LF,LP |
| DS | PH |
| PF | LF,BR,EP |
| VTH | VTD,VTS |

|  |  |
| --- | --- |
| **Summary Relationships** | |
| FC | PF |
| BR (Fund Level) | BR (Participant Fund Level) |
| PF (Loan Fund) | LF |

### 3.3.4 Omni Plus Database Relationships



### 3.3.5 Key Indexed Files

Omni File contains records. Each record is built in the following way:

Omni Plus Record Key Fields



Omni Plus Record Data Fields

Omni Plus UDF/EUDF - additional fields on the record, but not used directly by the system, shown on various screens/reports. Such User Defined Fields (UDF‟s) are useable by users for additional options or data.

All files are single key, keyed-sequential files (MVS - VSAM, AS400- DB2, all other platforms - Micro focus keyed indexed). Refer to the OmniPlus Data Element Dictionary for specific record layout/use.



* Indexed Files

|  |  |  |  |
| --- | --- | --- | --- |
| **Record ID** | **Record Name** | **Key Length** | **Record Length** |
| AAAA | Alternate Address | 46 | 860 |
| ANAM | Annuity Master | 46 | 1129 |
| BABT | Base text | 46 | 368 |
| BAEU | Extended Udf | 46 | 368 |
| BAFM | File Maintenance | 100 | 363 |
| BAUD | User Defined | 46 | 368 |
| CACA | Cash Control Account | 46 | 388 |
| CACH | Cash Control History | 46 | 324 |
| CKCD | Check Detail | 46 | 153 |
| CKCK | Check Header | 46 | 163 |
| CMCM | Compensation Record | 46 | 238 |
| CMSL | Salary Record | 46 | 174 |
| DBDB | Disbursement | 46 | 4505 |
| DBDG | Deduction Group | 46 | 2528 |
| DBDH | Deduction Header | 46 | 385 |
| DIMD | Minimum Distribution | 46 | 368 |
| DIRO | Rollover | 46 | 368 |
| DISG | Source Group | 46 | 368 |
| DNBE | Dben Beneficiary | 46 | 499 |
| DNCR | Dben Service | 46 | 865 |
| DNPB | Dben Prior Benefit | 46 | 442 |
| DNPN | Dben Participant | 46 | 722 |
| DSDS | Division/Subsidiary | 46 | 389 |

|  |  |  |  |
| --- | --- | --- | --- |
| FNFC | Fund Control | 9 | 700 |
| FRFA | Forecasting Annuity | 46 | 320 |
| FRFB | Forecasting Beneficiary | 46 | 111 |
| FRFP | Forecasting Projection | 46 | 204 |
| FRFR | Forecasting Header | 46 | 491 |
| HIBR | History Base Record | 46 | 157 |
| HIFX | Foreign Exchange History | 46 | 89 |
| ININ | Installments | 46 | 246 |
| IVIA | Investment Action | 46 | 160 |
| IVIC | Investment Control | 46 | 490 |
| BAI2 | Alternate Index | 46 | 86 |
| LNLF | Loan Fund | 46 | 129 |
| LNLH | Loan Header | 46 | 500 |
| LNLP | Loan Payments | 46 | 122 |
| MSMS | Message Record | 32 | 2360 |
| NTND | Note Detail | 46 | 121 |
| NTNH | Note Header | 46 | 275 |
| PEPE | Person Record | 46 | 1300 |
| PIEV | Insurance Event | 46 | 404 |
| PIPD | Insurance Detail | 46 | 579 |
| PIPI | Insurance Partic Level Source | 46 | 530 |
| PLCS | Plan Compliance Summary | 46 | 179 |
| PLCO | Plan Compliance | 46 | 166 |
| PLLC | Plan Locator Detail | 32 | 118 |
| PLLH | Plan Locator Header | 32 | 87 |
| PLPL | Plan Record | 6 | 2450 |
| PMPM | Product Master | 46 | 898 |
| PRDR | Price Daily Rate | 46 | 160 |
| PRPR | Price Header | 46 | 337 |
| PTAF | Participant Annual Financial | 46 | 387 |
| PTAI | Associated Individual | 46 | 745 |
| PTPF | Participant Fund | 26 | 700 |
| PTPH | Participant Header | 26 | 1100 |
| PTPS | Participant Source | 46 | 262 |
| PTVR | Voice Response | 46 | 434 |
| RPRH | Report Header Record | 21 | 175 |
| RPRP | Report Detail Record | 21 | 2037 |
| RRRR | Rate of Return | 46 | 3473 |
| SAGI | Gics | 46 | 368 |
| SAIL | Investor Lots | 46 | 2013 |
| SOSC | Source Control | 46 | 233 |
| SPSP | Sub Plan | 46 | 1284 |
| SSSA | Share Account | 46 | 670 |
| SSSH | Share History Control | 46 | 338 |
| SSSR | Share Request | 46 | 502 |
| SVST | Service Record | 46 | 136 |
| SVSV | Service | 46 | 151 |
| TALM | Tax Limit Records | 32 | 182 |
| TRTH | Trade History | 46 | 289 |
| TRTO | Trade Order | 46 | 578 |
| TXTF | Text File Header | 46 | 589 |
| TXTX | Text File Detail | 46 | 246 |
| VTTL | Transaction Log Record | 64 | 458 |
| VTTS | Transaction Submit Record | 64 | 131 |
| VTTD | Transaction Detail Record | 64 | 9000 |
| VTTP | Transaction Detail Pointer | 64 | 192 |
| XFEQ | Equity wash | 46 | 253 |
| XRXR | Foreign Exchange Rate Record | 46 | 96 |

* Omni Plus Record Key Fields

|  |  |  |  |
| --- | --- | --- | --- |
| **Record** | **Name** | **Copybook** | **Effective Keys** |
| BR | BASE ACTIVITY | AHBR | PLAN,PARTIC,TDATE, RUN-D-T,SEQ,FUND |
| BT | BASE TEXT | AHBT | PLAN,PARTIC,BTTYPE,KEYDATA |
| BTAD | ADDRESSES | AHBTAD | PLAN,PARTIC,'AD',ADDR-NUM |
| BTAPER | ASSOCIATED PERSON | AHBTAPER | PLAN,PARTIC,'APER',BTSEQ |
| BTLOG | PROCESSING LOG | AHBTLOG | PLAN,PARTIC,'LOG',VOUCHER,BTSEQ |
| DB | DISBURSEMENT | AHDB | PLAN,PARTIC,TDATE,RUN-D-T |
| DR | DAILY PRICE/RATE | AHDR | PLAN,INV,TDATE |
| DS | DIVISION/SUBSIDIARY | AHDS | PLAN,LOCATION |
| EP | EXTENDED PDF'S | AHEP | PLAN,PARTIC,FUND |
| FC | FUND CONTROL | MSTRFC | PLAN,FUND |
| IC | INVESTMENT CONTROL | AHIC | PLAN,INV |
| I2 | ALTERNATE INQUIRY | AHI2 | PLAN,VARIABLE,PARTIC |
| LF | LOAN FUND | AHLF | PLAN,PARTIC,LOAN#,FUND |
| LH | LOAN HEADER | AHLH | PLAN,PARTIC,LOAN# |
| LP | LOAN PAYMENT | AHLP | PLAN,PARTIC,LOAN#,TDATE |
| PF | PARTICIPANT FUND | MSTRPF | PLAN,PARTIC,FUND |
| PH | PARTICIPANT HEADER | MSTRPH | PLAN,PARTIC |
| PL | PLAN CONTROL | MSTRPL | PLAN |
| SC | SOURCE CONTROL | AHSC | PLAN,SRC |
| TXT | TEXT | AHTXT | PLAN,FILE-NAME,LINE# |
| VTD | TRANS DATA | MSTRVTD | PLAN,FILEID,TRAN,PARTIC,FUND,SEQ |
| VTH | TRANS HEADER | VMSTRVTH | PLAN,FILEID |
| VTS | TRANS SUBMIT | MSTRVTS | PLAN,FILEID,SYSTEM,PROCES |

* MSTR.PLAN – Plan Control
  1. record type, 6 byte key, approx 2200 byte total length. (MSTRPL) 1 record per plan.



Low use.

Referenced by DDNAME: 'VPLAN' or 'dd\_VPLAN'.

Backed up by periodic backup, restored from user request.

Reorganized by Month End Processing

* MSTR.FUND - Fund Control/Totals
  1. record type, 9 byte key, 700 byte total length. (MSTRFC) 1 to 400 Records per plan.



Low use.

Referenced by DDNAME: 'VFUND' or 'dd\_VFUND'.

Backed up by periodic backup, restored from user request. Reorganized by Month End Processing.

* MSTR.PART - Part Header/Funds

2 record types, variable length, and approx. 880 bytes maximum.



2 to 401 records per participant.

High use file, many fetches and rewrites.

Adequate index buffering is important.

Referenced by DDNAME: 'VPART' or 'dd\_VPART'.

Backed up by periodic backup, restored from user request. Reorganized by Month End Processing.

* MSTR.PART - Part Header

PH record type has 26 byte key Approx 880 bytes total length. (MSTRPH) 1 PH record per participant per sub plan.



PH record contains low-values in the PH-FUND-ID field.

* MSTR.PART - Part Fund Record

PF record type has 26 byte key



Approx 550 bytes total length. (MSTRPF)

2 to 401 PF records per participant per sub plan.

PF record contains the fund-id in the PF-FUND-ID field.

* MSTR.AHST - Misc/History

Multiple record types, variable length, 5000 bytes maximum. Several different types of records are stored on this file, including Loan, Disbursement, Investment Control, etc.



Size depends on activity and history retention.



High use file, many adds and fetches.

Adequate index buffering is important.

Referenced by DDNAME: 'VAHST' or 'dd\_VAHST'.

Backed up by periodic backup, restored from user request.

Reorganized by Month End Processing.

History is purged from AHST via the DPRBRET Purge/Condense process.

* Activity History File
  + **Base Records (AHBR)** (Participant Activity History)

**Key -** Plan number, +200 (record sequence), participant ID, trade date, run date, run time, sequence number (to maintain unique key), “BR”

* + **Base Text Records (AHBT)**

**Key -** Plan number, +100, participant ID, type (address, alternate payee, additional person, log, note, mdst, sgrp, etc.), key data, “BT”

(ADnn, APAY, APER, LOG, NOTE, MDST, SGRP)

* + **Disbursement History Records (AHDB)**

**Key -** plan number, +400, participant ID, trade date, run date, run time, sequence number (to maintain unique key), “DB”

* + **Daily Price/Rate Records (AHDR)**

**Key -**  plan number, +500, price id , trade date, contract id, “DR”

**Data** - Stock-Price, Unit2Shr-Conv-Factor, Daily-Fee-Factor, Purchase-Price, SellPrice, Distribution-Price

**Price ID** - Cusip number or investment id (first two bytes of fund-id + „\*‟)

* + **Investment Control Records (AHIC)**

**Key -** plan number, +550, investment id, “IC”

* + **Source Control Records (AHSC) Key -** plan number, +575, source id (last byte of fund-id), “SC”  **Loan Header Records (AHLH)**



**Key** - plan number, +384, participant ID, loan number, fund-id (low-values), “LH”

* + **Loan Fund Records (AHLF)**

**Key** - plan number, +384, participant ID, loan number, fund-id (corresponds to a

Fund Control fund-id for a “regular” investment, not the loan investment), “LF”

* + **Loan Payment Records - (AHLP)**

**Key -** plan number, +385, participant ID, actual payment date, loan number, sequence number (to maintain unique key), payment type (xx), “LP”

Payment types: LS = lump sum

PD = payroll deduction

* + **Division/Subsidiary Records (AHDS)**

**Key** - plan number, +800, div/sub identifier (x(4)), “DS”

**Data** - Primary Division/Subsidiary Name, Secondary Division/Subsidiary Name, Address.

* + **Text Records (AHTXT)**

**Key -** plan number, +950, prefix, text file name (x(25)), sequence number (9(4)v9(9)),

“TX”

* + **Alternate Inquiry Records (I2’s) (AHI2)**

**Key** - plan number, +700, view type (plan or part), view position, key text (x(15)), participant ID, specific plan number, “I2”

* + **Extended PDF Records (AHEP)**

**Key** - plan number, +600, participant ID, usage (xx), key data (x(6)), fund-id, “EP”

(usage codes indicate BP - Bond Price, BB-Bond Bal., GB - GIC, UD - User Defined)

* + **Annuity Master Record (ANXAM)**

**Key** - plan number/annuity group (x(6)), +061, annuity control (x(4)), participant number/annuitant id, sequence number, “AM”

* + **Annuity History Record (ANXAS)**

**Key -** plan number/annuity group (x(6)), +062, annuity control (x(4)), participant number/annuitant id, sequence number, transaction-id, activity code, run-date, runtime, sequence number, “AS”

* + **Forecasting Header Record (FRXFR)**

**Key -** plan number (x(6)), +544, participant number, forecast date, forecast time, forecast plan, sequence number, “FR”

* + **Forecasting Annuity Record (FRXFA)**

**Key** - plan number (x(6)), +541, participant number, forecast date, forecast time, forecast plan, sequence number, beneficiary SSN, annuity sequence number, forecasting sequence number, “FA”

* + **Forecasting Beneficiary Record (FRXFB)**

**Key** - plan number (x(6)), +542, participant number, forecast date, forecast time, beneficiary SSN, forecast sequence number, “FB”

* + **Forecasting Pointer Record (FRXFP)**

**Key -** plan number (x(6)), +543, participant number, forecast date, forecast time, forecast plan, projection sequence number, forecasting sequence number, “FP”

* + **Person Master Record (PEXPE)**

**Key -** plan number/group number (x(6)), +075, prefix (x(02)), person ID (x(13)), sequence number, “PE”

####  MSTR.TRAN - Transactions File

Multiple record types, variable length, approx. 8952 bytes maximum Records per participant depends on retention.



Contains transactions entered via online or loaded from batch via the VTUTIL job.

Referenced by DDNAME: 'VTRAN' or 'dd\_VTRAN'.

Backed up by periodic backup, restored from user request. Reorganized periodically

####  MSTR.TRAN - Record Types

|  |  |  |
| --- | --- | --- |
| **Record** | **Defintion** | **Screen ID** |
| VTH | Batch Headers | FHDR |
| VTD | Transaction Detail | T\*\*\* |
| VTS | Submit Records | TSUB |
| VTP | Pointers to VTD’s |  |
| VTL | Log Records for Online/Batch Processing |  |

####  MSTR.UCOM - Report Storage

Contains online report file records: 1 header record (MSTRRHR) and multiple page segment records (MSTRRPR) per report.



Referenced by DDNAME:'VUCOM' or 'dd\_VUCOM'.



Reports are added to 'VUCOM' via unified processing or UCOM utility. Backup/Restore of full or partial files by user request.

####  MSTR.PASS - Password Control

Key encrypted.



One record per password in use. One file per application region.

Referenced by DDNAME: 'DPRSPASS' or „dd\_DPRSPASS'.

####  MSTR.BALK - Lock Control File

One record per lock file.



One record read/written prior to files being opened for output.

Record contains a lock flag for each set of master files. (VTRAN, MSTR, UCOM) High Use.

Referenced by DDNAME: VBALK or 'dd\_BALK'.

### 3.3.6 Sequential files

Other than the Indexed files, the sequential files are as follows:

&LIBS.CTLLIB - Control Cards



Keyed indexed Define Cards

Sort Control Cards

Standard Default Controls

Environment Control Card (ENVRMNT)

Override run date

Override run time

VTRAN integration - On/Off

Global run mode - Edit/Post

Sequential Activity History - On/Off

Price lookback/when - Edit/Post

UCOM report routing - On/Off

####  Sequential Datasets

Input Transactions (TI)



Processing Transactions (TC)

Report Strings (RS)

Payment Interface Deferred Work

## 3.4 Omni Station

Omni Station is a Microsoft® Windows™ based PC system that interfaces with the Omni Plus Employee Benefits Mainframe System. It is intended to provide an efficient, user-friendly environment in which to process Omni Plus.

The following are the features of Omni Station:

Omni Station runs in a normal Windows™ environment.



Omni Station takes advantage of many standard Windows™ features such as “cut and paste” capabilities.

Omni Plus has multi-tiered client server architecture.

The architecture is open and non-proprietary.

Omni Station is the client or front-end presentation layer. Omni Plus is the server.

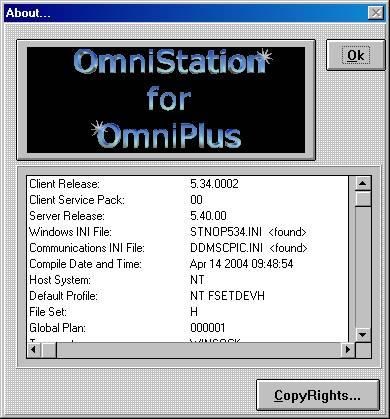


* Omni Station which is a thick client written in VB has Sockets programming involved which directly connects to the CICS transaction listener (Read ZINCIC\*) that runs on the mainframe.

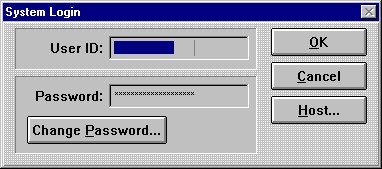
* The CICS listener fires a Transaction on behalf of Station for after getting the data from Station and returns the results back to station to be displayed for the user.

### 3.4.1 Getting Started

* When Omni Plus is first accessed, the „About‟ dialog box appears which contains information such as the release number, the compilation date of the pgm, etc.



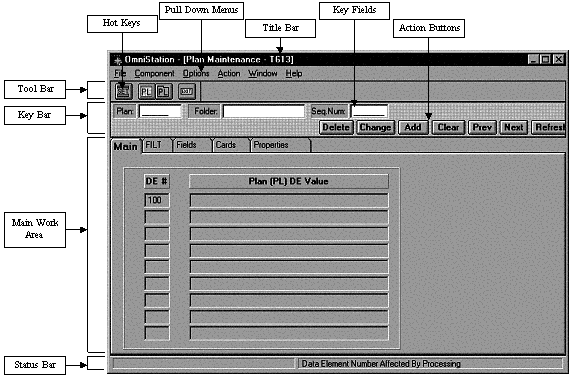
* From this box, you may exit the application by clicking on the QUIT button, or continue with the application by clicking the OK button. If you click on the OK button, the System Login box appears.



* This box allows you to log on using your User ID and Password. Enter these two fields and click OK.

### 3.4.2 Working in the Windows Environment

Every Omni Station window contains standard components, which are illustrated below:



### 3.4.3 Components - Pull Down Menu

Every Omni Station window contains standard components, which are illustrated below:

Annuities



Cash

Check

Contributions

Compensation

Disbursement

Distribution

Earning

Forecasting

Foreign Exchange

Funds History

Insurance

Investments

Loans

Omni Ben



Omni Script

Plans

Prices

Participant

Persons

RateOfReturn

Report Mngmt

Share

Sources

Text Files

Trade System

Transaction Mngmt

Transfers

Addr/ Adjustm

Base Text Fees

Div/Subsidiary

Installments

Notes

Participant

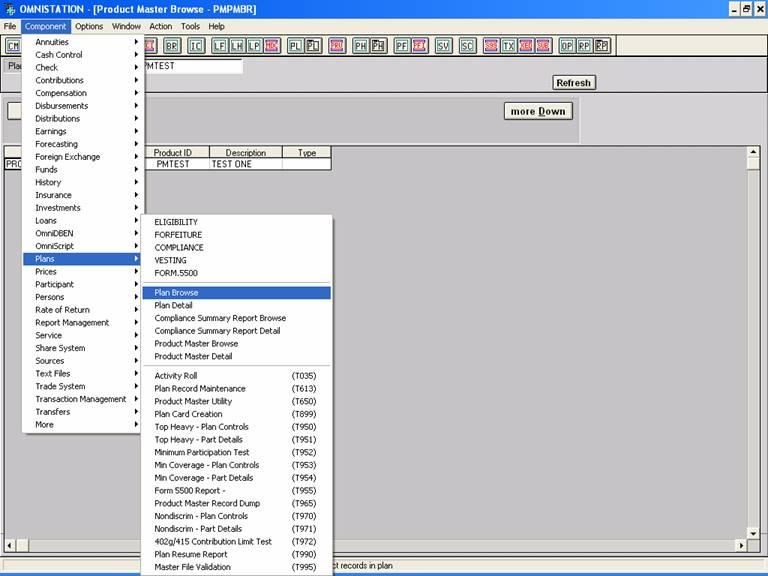
Sub Accounting

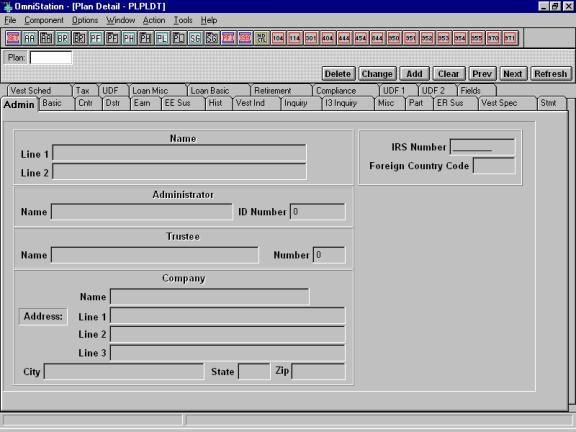
Sub Plan

Utilities

### 3.4.4 Plan Component Menu

The Plan options on the Component menu of Omni Plus provide the user with the ability to access a listing of plans on a file set (Plan Browse), specific information about a particular plan (Plan Detail), and a series of transactions used for plan record maintenance, validation and compliance testing.

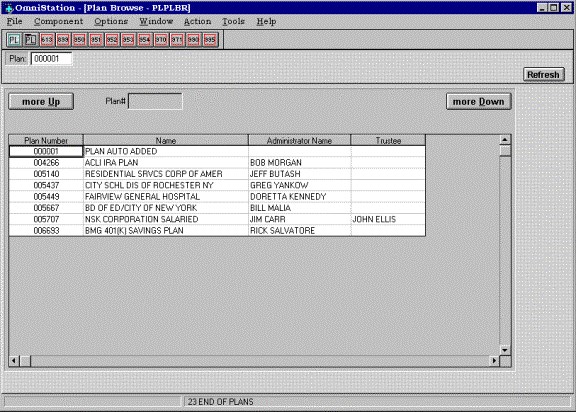




**Plan Brows**

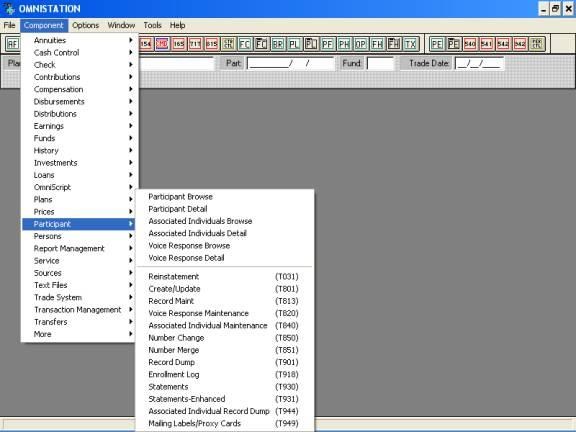
**e**

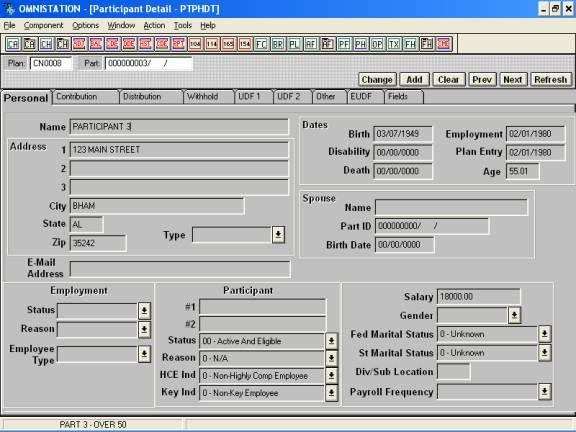
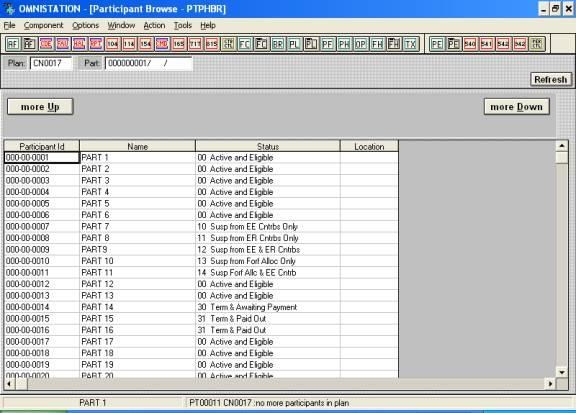
**Plan Detail**



### 3.4.5 Participant Component Menu

The Participant Component Menu on Omni Plus lists Browsing, Creating, Updating and Statement functions available for Participants. This component menu also includes functions for the Associated Individuals and Voice Response Components.





**Participant Browse Participant Detail**

### 3.4.1 Fund Component Menu

Determine whether whole or fractional units will be held

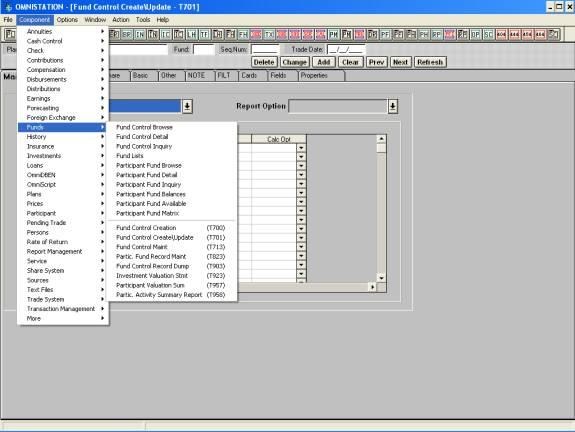


Determine whether units will be automatically purchased during a financial transaction within the plan.

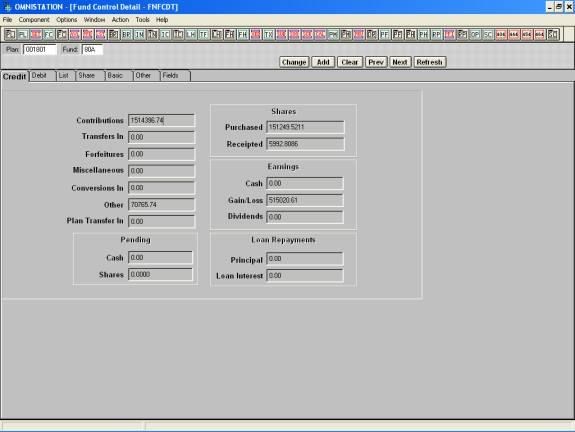
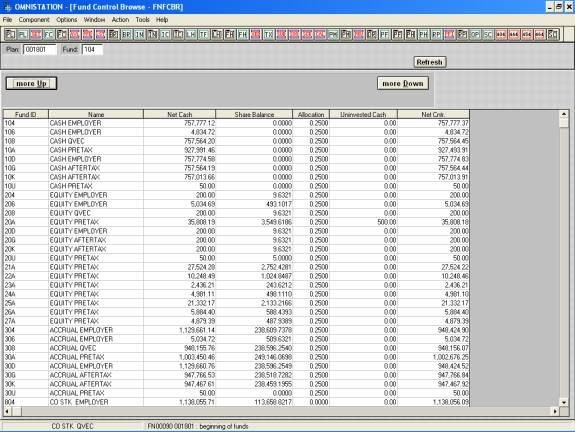
Define the processing sequences for fund hierarchy



Define whether prices are static or daily



.

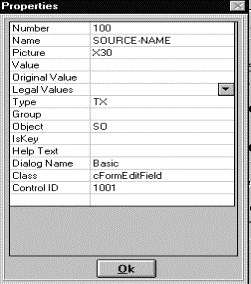


**Fund Browse Fund Detail**

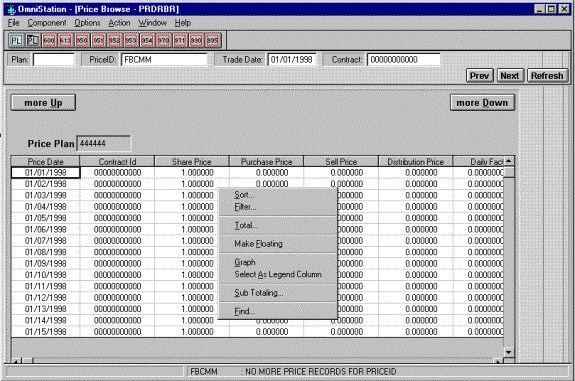
### 3.4.2 Right Click Mouse Functions

* **Key Field Functions** - Key information, which controls the data displayed on the screen, can be accessed by using the right mouse button

* **Field Level Information** - screen providing useful info about each field in OmniPlus



* **Spreadsheet Right Click Functions** - numbers to be totaled, locates a searched numeric or alpha field, column to be graphed, makes the spreadsheet a floating display



# 4.0 Omni Script

**4.1 Overview**

“Omni Script", formerly referred to as “OmniCalc” or “Calculator”, is the programming language used to customize Omni Plus in order to meet specific processing and reporting requirements.

Omni Script extends the functionality of Omni Plus without the need for traditional code.

Omni Script allows:

* Customization of the Omni Plus system to meet plan-specific rules of transaction eligibility, processing, and amounts available.
* Processing of custom edits and validations at the plan or participant level.
* Creation of custom reports, statements, and data extracts to meet the needs of the record keeper, client, trust, or other recipient.
* Adjustments and maintenance of database records and fields.
* Generation of formatted XML or HTML web pages.

## 4.2 Features

### 4.2.1 Transaction-Level Processing Extensions

Omni Script statements can be entered directly to many transaction screens. These are usually smaller, temporary programs not stored in a separate file on the database.

Transaction-level code can:

* Supply values or factors for unique allocations. (e.g., The Company issues a special stock bonus.)

* Correct and maintain participant accounts. (e.g., an ineligible participant shared in an allocation and the account must be corrected.)

* Research and verify information from the database. (e.g., Verify a participant's vested account value.)

### 4.2.2 Processing Overrides

The system can also call and execute specific, stored Omni Script programs during transaction processing. These 'Omni Script processing exits' allow custom modifications/extensions to the usual transaction processing.

Omni Script processing exits can:

* Build allocation bases for earnings, contributions, forfeitures, and fees (e.g., Time weighting of earnings)
* Determine participant's eligibility for plan or share in allocations (e.g., only active participants are assessed fees)
* Customize edits and validations to ensure plan rules are enforced (e.g., Participants cannot transfer money that has been in the plan for less than 12 months)
* Determine amounts available for loans, transfers, terminations, and withdrawals (e.g., Participants cannot withdraw employer money until they reach age 50 1/2)

### 4.2.3 Customized Reports and Screens

* **Full-page reports**: Omni Plus contains a feature called Report Writer that produces detailed custom reports (Participant Statements, Proxy Letters, Withdrawal Confirmations, etc.).
* **Columnar reports**: Omni Plus contains a preformatted report facility that reports on specified database or computed values with customized headings and text.
* **Custom screens**: Using Report Writer, custom screens of either computed or extracted Omni Plus information can be generated.

### 4.2.4 Data and XML/HTML Generation

* **Data generation:** Data files of either computed or extracted OmniPlus information can be generated through Report Writer. This feature is used to create proxy tapes (e.g., transaction files, voice response extracts).
* **HTML generation:** HTML pages or XML documents of either computed or extracted Omni Plus information can be generated and displayed as Internet/Intranet web pages through OmniNet.

## 4.3 Basic Concepts

### 4.3.1 Programming Structure

An Omni Script is a collection of statements. A statement is any single, clearly defined instruction to the program or computer. The collection of statements is sometimes referred to as "Omni Script text", which can be included on an Omni Plus transaction or retrieved from a stored file.

Omni Script text, like computer programs, can be short or long, containing a single Omni Script statement, or a string of many statements. As part of the function of the Omni Script facility, there are some precise rules about punctuation both within and between individual Omni Script statements. These rules are best learned through practice and example, but they are summarized here.

* Statements

An Omni Script statement is a single, clearly defined instruction to the computer. Omni Plus Omni Script statements are separated by a semicolon (; ). Statements typically perform a single function, such as:

* + Assigning a value to a variable (WK001=5 ;)
  + Controlling the Omni Script program (IF …, LOOP …, EACH@ …, etc.)
  + Using keywords to call certain actions (AMORT, INITKV, etc.)
  + Calling subroutines (ROUTINE, PERFORM, etc.)
  + Executing an OmniScript Function Operation (OCTEXT\_SETLENG (TX001 5);)

Individual elements of an Omni Script statement may appear on one line, or on multiple lines. Multiple statements may be included on a single line. Generally, spaces between elements are not significant.

Note: OmniScript statements can be mixed (UPPER or lower) case. For example:

if Wk1>wk2; tX001=' Work1 > Work2‟; END;

* Formatting

The Omni Script facility allows considerable freedom in physically arranging the Omni Script statements within a text file or on a transaction screen. It is advantageous to use this freedom to enhance the readability and maintainability of OmniScript text by using spacing and indenting to set off major portions or statements within an Omni Script. This arrangement makes the logical structure of the program apparent to anyone reading it.

* Comments

In addition to formatting, comments within Omni Script text can also ease the understanding of Omni Script statements and make them more maintainable. Comments are notes, explanations, or instructions intended to document what an Omni Script statement does. Comments within an OmniScript are not processed by Omni Plus.

Comments can be placed on a line by themselves, or after Omni Script text. Depending on where they appear, the coding is slightly different. If a comment appears on a line with no other information, it needs to be preceded by an asterisk (\*). Omni Plus ignores any Omni Script statement that begins with an asterisk as the first character on the line. This type of comment line may only be used in Omni Script text files. It may not be used in Omni Script coded directly on transaction screens.

Another type of comment appears on the same line with other Omni Script statements. In this case, the comment must be included within /\* … \*/ parameters (ex. /\* This is a comment \*/) that set the comment off from other Omni Script text. These parameters indicate both the beginning and end of the comment, so they must be used in pairs. This type of comment may be used in a text file or on a transaction screen. The table below summaries the two formats for comments in Omni Script statements.

**Examples of comments:**

|  |  |
| --- | --- |
| **Text in**  **OmniScript** | **Explanation** |
| \* This  OmniScript is used \* to calculate earnings. | Because of the leading '\*' on each line, these comments will NOT be processed by OmniPlus. |
| IF  WK1=WK3;  /\*  Compares  WK1 &  WK3 \*/ | The OmniScript statement (IF WK1=WK3) is processed, but the comment that describes what the OmniScript is doing is set off by the /\* … \*/ parameters and will NOT be processed. |

 Expressions

The Omni Script facility supports mathematical expressions by using a combination of operators, data values, and parentheses. These mathematical expressions result in numeric values (most commonly) used for assignment or comparison.

Operators

Operators are symbols used by Omni Plus to perform operations in Omni Script statements or expressions. Operators perform a function using two data values to produce one result, e.g., adding together two numbers to get one sum, or multiplying two numbers to get one product. The tables below lists the arithmetic operators recognized by the OmniScript facility and provide a simple example.

1. Arithmetic Operators

|  |  |  |
| --- | --- | --- |
| **Operator** **Explanation** | | **Example** |
| = | Assign | WK001 = 3; |
| \* | Multiplication | WK001 \* 3; |
| / | Division | WK001 / 2; |
| + | Addition | WK001 + 1; |
| - | Subtraction | WK001-5; |
| \*\* | Exponent (example: 3 squared) | WK001=3 \*\* 2; |
| << | Lesser of two values | WK001 = WK002 << WK003; |
| >> | Greater of two values | WK001 = WK002 >> WK003; |
| =: | Assign and propagate a value into multiple fields | WK001=WK002=:WK003; |

1. Assignment Operators

Omni Script assignment statements store values in data fields. Values may be simple (from another element or literal) or intermediate (computed from an expression). When an assignment is made from a text (alphanumeric) field to a numeric field, the Omni Script facility scans the text field and converts it to a numeric value.

Assignment operators store values in data fields. The table below lists the assignment operators recognized by the OmniScript facility.

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Explanation** | **Example** | **Ending WK1 Value** |
| = | Field assigned specified value | WK1=5 | 5 |
| =$ | Value rounded to two decimal places | WK1=$5.1234 | 5.12 |
| =+ | Stored Value is Added to Target Sets WK1 to 1 | WK1=5;WK1=+1; WK1=0; WK1= +1; | 6  1 |
| =- | Stored Value is Subtracted from Target  Sets WK1 to –1 | WK1=5;WK1=-1; WK1=0;WK1= -1; | 4  -1 |
| =: | Assign and propagate a value into multiple fields WK001=WK002=:WK003; | |  |

1. Assign & Propagate Operator (‘=:’)

The operator „=:‟ is used to store a value in the left parameter, while also giving the value as a result. „=:‟ can be used to store a value and also provide the value for further processing.

**Example :**

#### WK1=WK2=:100; Stores 100 in WK1 and WK2; WK1=WK2=:(WK3=:200); Stores 200 in WK1, WK2, and WK3

**Note: The first variable must be followed by an equal (‘=’) all other variables must have an equal colon (‘=:’).**

1. Comparison Operators

Comparison operators produce a true or false result when comparing two operands. The table below lists the comparison operators recognized by the Omni Script facility and provides a simple example.

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** **Explanation** | | | **Example** |
| < | Less than | If (WK001<20);… | |
| > | Greater than | If (WK001>20);… | |
| = | Equal | If (WK001=20);… If (TX001=‟Fred‟);… | |
| <= | Less than or equal to If (WK001<=20);… | | |
| >= | Greater than or equal to If (WK001>=20);… | | |
| >< or  <> | Not equal to If (WK001><20);… If (WK001<>20);…  If (TX001><‟Fred‟);…  If (TX001<>‟Fred‟);… | | |

Logical operators compare two or more operands and generate a true or false result. A FALSE value is returned for a zero result, and a TRUE value is returned for a non-zero result. The table below lists the logical operators recognized by the Omni Script facility and provides a simple example.

1. Logical Operators

Logical operators compare two or more operands and generate a true or false result. A FALSE value is returned for a zero result, and a TRUE value is returned for a non-zero result. The table below lists the logical operators recognized by the OmniScript facility and provides a simple example.

**Operator** **Explanation** **Example**

AND Compares two or more operands, all of which must be true If (WK001>20) and (WK001<40);… OR Compares two or more operands at least one of which is true If (WK001<20) or (WK001>40);…

Expression Evaluation

Omni Script expressions are evaluated according to set rules. Evaluation of expressions proceeds from left to right with no priority when there are no parentheses. Intermediate numeric results are evaluated up to 17 digits (eleven to the left of the decimal and six to the right). Thus, WK1 = 2 + 3 \* 4 – 2;

results in WK1 having a value of 18. (2+3=5, 5\*4=20, and 20-2=18.)

When parentheses are present, operations within parentheses are evaluated first, and left to right when more than one set is included. Thus, WK1 = 2 + (4 \* 3) – 9;

results in WK1 = 5. (3\*4=12, 2+12=14, and 14-9=5.), and WK1= (4\*2) + 1 - (2\*3);

results in WK 1 = 3. (4\*2=8, 2\*3=6, 8+1=9, and 9-6=3.)

Particular care must be taken in using 'IF' statements in conjunction with 'AND' or 'OR' operators to ensure the correct order of evaluation. Operations must always be enclosed within parenthesis when using 'AND' and 'OR' operators to force the proper order of resolution. Thus, IF (WK6 > 5) OR (WK1 > 0); is a valid statement, but IF WK6 > 5 OR WK1 > 0;

is not a valid statement.and 'OR' operators to force the proper order of resolution. Thus,

#### IF (WK6 > 5) OR (WK1 > 0); is a valid statement, but

IF WK6 > 5 OR WK1 > 0; is not a valid statement.

 Using Text Fields with Numeric Fields

An alphanumeric field must first be moved to a WK field before being compared to a numeric value.

**Example (VALID):**

***Task:*** Determine whether the Location Number (PH244) is greater than '1000'. If so, make WK212 equal to '0'.

#### WK001=PH244; IF (WK001 > 1000); WK212 = 0; END;

***Result:*** This operation is valid because the alphanumeric field (PH244) was moved into a numeric field (WK001) before the comparison was made in the 'IF' Statement.

**Example (INVALID):**

***Task:*** Determine whether the Location Number (PH244) is greater than '1000'. If so, make WK212 equal to '0'.

#### IF (PH244 > 1000); WK212 = 0; END;

***Result:*** This code will be rejected because the alphanumeric field (PH244) was not moved into a numeric field before the comparison was made in the 'IF' Statement.

An alphanumeric field can be compared to a numeric value expressed as a text literal:

***Task:*** If the Location Number (PH244) is not equal to „1000‟, then do not print the person.

IF (PH244 <> '1000'); WK212 = 1; END;

### 4.3.2 Omni Script Data

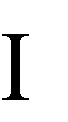
The Omni Plus Omni Script facility uses system data elements, data values, variables, and literals to allow the user to create simple and complex calculations with access to nearly every piece of information available on the Omni Plus system.

* Data Elements

Data elements are referenced in Omni Script text by a two-character prefix and a one-to-three digit numeric data element number (e.g. PH104, BR100, WK1, etc.). Some data elements may also be referenced by an alias name. Alias Names are a substitute for a data element name. The Omni Script will substitute the actual data element for the alias. For a list of aliases, see the appropriate "field" discussion of the data element in question.

* Data Values

Data values in Omni Script fall into a variety of categories depending on the format of the information. These categories include:

* Numeric - used for cash, counters, shares, date, etc.
* Text - used for alphanumeric names, values, etc.
* Work products - values computed or stored within the Omni Script (WK, TX, IT).
* Database values - values stored or calculated within Omni Plus as actual or artificial data elements (PH, PF, BR).
* nterface values - values defined by a particular function of the system (FD).
* Environment values - system-defined values based on processing circumstances (SD).

Variables

A Variable is a work, database, or environment field from which data can be fetched or to which data can be stored (As opposed to a literal which is unchanging and predefined). Variables are numeric or text.

* Literals

Specific, fixed values specified in an Omni Script program, used to assign values or compare to variables.

* Numeric Literals

Numeric literals consist of digits and an optional decimal point.

Examples: WK1=0; WK2=123.4; WK3=.005;

* Signed Numeric Literals

WK1=+1; Will add 1 to WK1.

WK1= +1; Will set WK1 to 1.

WK1=-1; Will subtract 1 from WK1.

WK1= -1; Will set WK1 to negative 1.

WK1=1; Will set WK1 to 1.

WK1= 1; Will set WK1 to 1.

**Note: When setting a variable to a literal negative, there is a space between the equal sign (=) and the negative sign (-).**

* Text Literals

A pair of single or double quotes delimits text literals.

Examples: TX1=‟Value‟; TX2=”Value”; IF TX1=‟10A‟; Literals can be up to 200 characters in length.

 Hexadecimal Literals Omni Script supports the specification of hexadecimal literals.

**Example**

#### TX1=X‟F1F2F3OCOD‟; or IF TX2=X‟OAOC‟; TX2=‟CRLF‟; END

This allows users to value or compare fields to exact hexadecimal values. This can be useful in setting certain printer control sequences or imbedding CRLF sequences in output data. Hexadecimal literals are a type of TEXT literal, and can be specified anywhere TEXT literals can be specified.

 Picture Clause (PIC)

Defines the characteristics (type (Numeric or Text), length, decimal places, etc.) of a data element.

* X(20) specifies a 20 character text (alpha-numeric) field
* 9(9)V9 (2) specifies a cash type field of up to 9 digits with 2 decimal digits

Each data element may be described by a picture clause that defines the type of field and number of characters (or digits) stored in the field. For example, the picture clause 'PIC X(3)' indicates that the field is alphanumeric (X) and contains three characters.

### 4.3.3 Intermediate Work Field

Various fields are available to all Omni Script to hold intermediate results and to interface to certain pre-written routines. The types of Intermediate/Work fields used in Omni Script are described in this section.

 (WK) Work Fields

Work Fields **(WK)** are used to hold intermediate and/or final numeric calculation results and can be printed on selected reports. These fields can be used to simplify a calculation process or to allow a computed value to be used multiple times within the Omni Script Text without recalculation.

* WKs cannot accommodate numbers greater than 99,999,999,999.999999. „PIC 9(11)V9(6)
* WKs calculation results can be used multiple times in the same program
* Omni Script contains up to 6200 work fields (WK001 through WK6200). Up to 3000 can be used with the T966 Custom Report Generator
* All 'WK' Work Fields are reinitialized to zero between participants so that calculations are performed for each participant (not accumulated for all participants).
* WKs can be user defined or predefined by the transaction you are processing. Most transactions set aside a block of WKs to perform certain tasks. Before creating a program, you must be familiar with the pre-defined WKs for the transaction. (See the Omni Script Processing Exits section of the documentation manual you are using. For a list of all Calculator Processing Exits, see the System Cross References) **Example:**

This example illustrates the importance of knowing the predefined fields for each transaction. The following statement can be interpreted in different ways depending on where it is used.

#### WK001=PF120@\*\*\*;

Work Fields can have particular uses in transactions that use Omni Script for custom exit processing.

For example, WK212 - WK217 are predefined for the T966 transaction, and therefore cannot be user-defined for that transaction.

 (IW) Indexed Work Fields

Indexed work fields (IW) are used to store temporary values and assign them to work fields (WK). IWs reduce the number of repetitive statements required because they can be repeatedly used in a single Omni Script, but have their values assigned to incrementing work fields. This gives IWs the unique advantage of being especially useful with repeating Omni Script statements such as the LOOP and EACH@ commands.

Indexed work fields (IWs) are assigned to corresponding work fields (WK) based on the setting of the Indexed Work Field Base Value (SD030). This field is established in the Omni Script text and drives the incrementing of the IW/WK link in the Omni Script text. Normally, IW000 assigns a value to WK000. If SD030 contains a value, IW000 assigns a value to WK000 increased by the value in SD030. Thus, if:

SD030=100, then

IW000 will assign a value to WK100,

IW001 will assign a value to WK101,

IW002 will assign a value to WK102

When the end of a repeating Omni Script command is reached (EACH@ or LOOP), SD030 is reset to the amount to increment IW/WK link. Thus, if the above example were complete after WK102, then:

SD030=+3, then

IW000 will assign a value to WK103 (on the next pass through the EACH@ or LOOP)

IW001 will assign a value to WK104,

IW002 will assign a value to WK105

Then the EACH@ or LOOP continues and the sequence is repeated again beginning with WK106, WK107, etc. The example below demonstrates how a LOOP command could be used with indexed work fields to increment values that could be used for a statement or report on loan balances. An explanation of the function of each Omni Script statement follows.

#### Example

In this example, loan numbers and balances are obtained for the first three loans with balances for each participant and stored in work fields. IW000 gathers the loan numbers for the first three loans with balances, and stores these numbers in WK100, WK102 and WK104. IW001 gathers the loan balances for these corresponding loans and stores the balances in WK101, WK103 and WK105. When the system reaches a fourth loan, the WK001 counter exceeds a value of '3' and exits the loop for that participant.

|  |  |
| --- | --- |
| **Omni Script Text** | **Explanation of Omni Script Statement** |
| SD030=100; | Sets the Indexed Work Field Base Value to 100. So now IW000 will store information in WK100, IW001 in WK101, IW002 in WK102. |
| WK001=1; | Pre-sets WK001 to a value of '1'. This work field will be used to store the incrementing loan numbers. |
| LOOP; | Establishes a LOOP statement. All subsequent commands will be repeated until the OmniScript reaches a condition that ends the loop. |
| KL001=WK001; | Sets the loan number field to '1' (because WK001 was pre-set to this value). This ensures that each pass of the Omni Script for a participant starts with Loan 1. |
| IF WK001>3; ENDEXIT; | Determines if the loan number is greater than 3, and if so, ends the loop statement. |
| IF LH315>0; | Determines if the participant has a loan balance. LH315 is a loan header data element that contains current balance. |
| IW000=LH300; | Sets IW000 to the loan number. |
| IW001=LH315; | Sets IW001 to the loan balance. |
| SD030=+2;END; | Increments the Indexed Work Field Base Value by +2 so that now IW000 will store information in WK102, IW001 in WK103 for the next loan. |
| WK001=+1; | Increments the WK001 field that is used to identify the loan number by +1, so that on the second pass the Omni Script will gather information on Loan 2. |
| ENDLOOP; | Ends the LOOP statement. |

* (WA) Work Accumulator Fields

Work accumulator fields (WA) accumulate results that are not to be reinitialized to zero between each participant processing. All WA fields are user-defined. Values must be loaded to work fields (WK) to be printed. Up to 100 WA fields are available for use (WA001 through WA100). The example below demonstrates how WA fields might be used to increment check numbers and print them sequentially. An explanation of the function of each Omni Script statement follows.

|  |  |
| --- | --- |
| **Omni Script Text** | **Explanation of Omni Script Statement** |
| WA003=WA003+1 | Sets WA003 to increment by +1 with each participant. The initial value of WA003 is '0'. |
| WK001=WA003; | Sets WK001 equal to the incrementing value of WA003 so that check numbers can be printed. |

* (WF) Work Fund Fields

Work Fund Fields (WF) allows fund related intermediate calculated results to be held. Work Fund fields are particularly helpful in the Report Writer facilities, where a great many figures or balances are to be printed on one page. For instance, if you need to print Market Value, Share Balance, Contributions, and Dividends for all investments in a Plan, a WF field can be assigned to each fund in the CALC SECTION of the Report Definition. The TEXT SECTION can be set up with the appropriate Fund-ID for each calculated Work Fund Field.

* WFs can be followed by an '@' symbol and a three character specific Fund ID. The Fund ID can be 'source' generic (i.e., 10\*), 'investment' generic (i.e., \*\*A), or all generic (i.e., \*\*\*).
* If a WF is not followed by an „@‟ Fund ID qualifier, the FUND-ID usage rules are followed: the CURRENT-FUND-ID (SD033) is used.
* WFs are only valid for selected applications.
* OmniScript contains up to 999 work fund fields (WF001 through WF999). Note that the number of available Work Fund Fields may be limited based on independent variables unique to each Plan.
* WFs can be user defined or predefined by the transaction you are processing. Most transactions set aside a block of WFs to perform certain tasks. Before creating a program, you must be familiar with the pre-defined WFs for the transaction. Note: Each documentation manual contains a list of all pre-defined fields.
* WFs reduce the number of repetitive statements required, making the program easier to read.
* The FUNDS= XXX Command is used to reserve available storage within the Report Writer when using WFs. Please refer to the Report Writer - Parms section of this manual.

The example below demonstrates how WF fields might be used to store participant amounts. An explanation of the function of each OmniScript statement follows.

|  |  |
| --- | --- |
| **OmniScript Text** | **Explanation of OmniScript Statement** |
| EACH@\*\*\*; | Initializes an EACH@ statement so that amounts stored in WF fields are by specific investment and source. |
| WF004=PF999;  WF005=PF130;  WF006=PF100;  ENDEACH; | Stores the participant account value (PF999) in WF004 for each fund and source. Stores the participant share balance (PF130) in WF005 for each fund and source. Stores the participant uninvested cash amount (PF100) in WF006 for each fund and source. |

* (TX) Text Work Fields

Text work fields (TX) are used to store alphanumeric text for use in messages generated by Omni plus Omni Script processing. TX fields are limited to 200 characters. Omni Plus Omni Script may use up to 120 TX fields (TX001 through TX120).Omni Plus Text fields are strings of up to 200 characters. A length field is maintained for each string, and the „+‟ operator can be used to concatenate strings.

#### Example

PARM SECTION.

TEXT SECTION.

>>L1

WITHDRAWAL REQUEST

>PH911 >TDATEA:D3

>TX001 >100:F52 >>C50

CALC SECTION.

##### IF (PF999@\*\*G>0); TX001='YOU CAN WITHDRAW THE FOLLOWING AMOUNT:'; ELSE; TX001='WITHDRAWAL NOT AVAILABLE'; END; WK100=PF999@\*\*G; IF (WK100>0); WK050=1; END;

 (IT) Indexed Text Fields

Indexed text fields (IT) are used to store temporary values and assign them to text work fields (TX). ITs reduce the number of repetitive statements required because they can be repeatedly used in a single Omni Script, but have their values assigned to incrementing text work fields. These fields work much the same way as Indexed Work fields (IW).

Indexed text fields (ITs) are assigned to corresponding text work fields (TX) based on the setting of the Indexed Text Field Base Value (SD031). This field is established in the Omni Script text and drives the incrementing of the IT/TX link in the Omni Script text. Normally, IT000 assigns a value to TX000. If SD031 contains a value, IT000 assigns a value to TX000 increased by the value in SD031.

#### Example

In the following example, four IT fields are used with the EACH command; therefore, SD031 must be increased by a value of '4' before each subsequent execution of the loop.

SD031=1; /\* Start with the first one \*/ EACH@\*\*\*; IT1=FC030;

IT2=FC031; IT3=FC034; IT4=FC036; SD031=+4; ENDEACH;

### 4.3.4 Environment Fields

Writing useful and effective OmniScript programs to support record-keeping functions depends on a reasonable knowledge of where information is stored in the OmniPlus database. The following section details the System-Defined Data Elements, Function Data Elements, and Key Value Fields available to the various execution environments.

 (SD) System-Defined Data Elements

Several categories of System Defined Data Elements have been established for use within OmniScript. System-defined data elements represent values automatically calculated by the system that can be used in OmniScript.

* System Defined Values can be used in any OmniScript context.
* Literal equivalent values (Aliases) may be substituted for SD data element identifiers.

Available SD aliases are listed in the Literal Equivalents column of the table below.

* System Defined (SD) data elements consist of:

Storage areas accessed by different functions Flags that perform various tasks  Some common uses for SDs include:

Date Conversion Facility

OmniScript Text Substring Facility

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** |
| SD001 | SDPlan | Current Plan ID | System-assigned |
| SD002 | SDPartNum | Current Participant Number | System-assigned |
| SD003 | SDRDate | Current System Date | System-assigned |
| SD004 | SDRTime | Current System Time | System-assigned |
| SD005 | SDProcDt | Current Processing Date | System-assigned |
| SD006 | SDNumFCs | Total Number of Fund Control Records | System-assigned |
| SD007 | SDNumPFs | Total Number of Participant Fund Records | System-assigned |
| SD008 | SDPartCtr | Total Participant Count | System-assigned |
| SD009 | SDActive | Record Status Flag. Values are:   1. – Active 2. – Inactive | System-assigned |
| SD010 | SDEdit | Edit Mode Indicator. Values are:   1. or “E” – Edit mode 2. or “P” – Update mode | System-assigned |
| SD011 | SDTSeq | Current Transaction Sequence# | System-assigned |
| SD012 | SDNumICs | Total Number of Investment Control Records | System-assigned |
| SD013 | SDTotPage | Total Page Flag. Values are:   1. – Not a totals page 2. – Totals page | System-assigned |
| SD014 | SDFMRpt | File Maintenance Reporting Option. Values are: | Defined in the  OmniScript text by the |

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** |
|  |  | 1. – Report only changed field values 2. – Report regardless of whether field value is changed 3. – Report only unchanged field values 4. – Do not report any field values | user. |
| SD017 |  | Display Log Messages Flag. Allows users to specify where LOG messages should be routed, either to OP-02 reports (Information Log, Warning Log, Reject Log), to SYSOUT logs, or both.  “Y” – Route log messages to SYSOUT (default for Jobcalc)  “ “ – Route log messages to OP-02 reports (default for Unified Jobs)  “B” – Route log messages to both SYSOUT and OP-02 reports. | Defined in the  OmniScript text by the user. |
| SD020 | SDFmt1 | Message Field Mask-1. Indicates the print format for numeric values in OmniScript messages. Values are:   1. – Dollar S9(9)V99 format 2. – Share S9(9)V9(4) format 3. – Whole number S9(9) format 4. – Price S9(7)V9(6) format 5. – Date S9(6) format 6. – Time S9(4) format 7. – Percent S9(3)V9(4) format 8. – Social security number S9(9) format 9. – Count S9(9) format | Defined in the  OmniScript text by the user. |
| SD021 | SDFmt2 | Message Field Mask -2. Values are same as SD020. | Defined in the  OmniScript text by the user. |
| SD022 | Tran Type | Transaction Type. Refers to the TYPE field on a Participant InterFund Transfer (T381) transaction or the TYPE field on a Withdrawal transaction (T444). | System-assigned |
| SD023 | TranType1 | Transaction Type - 1st character. First character of SD022. | System-assigned |
| SD024 | TranType2 | Transaction Type - 2nd character. Second character of SD022. | System-assigned |
| SD025 |  | Transaction Code. Currently executing transaction. | System-assigned |
| SD030 | SDWKIDX | Index Work Field - Base Value. (See "Indexed Work Fields" section above.) | Defined in the  OmniScript text by the user. |
| SD031 | SDTXIDX | Index Text Field - Base Value. (See "Indexed Text Fields" section above.) | Defined in the  OmniScript text by the user. |
| SD033 | SDFund | Default Fund-ID. The current Fund-ID applicable to a fund level variable fetch or store in which no „@‟ parameter is specified. See the „Fund Usage‟ section. | Defined in the  OmniScript text by the user. |
| SD035 | SDPrDate | Price Date | System-assigned if not included in OmniScript text by the user. |
| SD036 | SDLPart | Last Participant Flag. Indicates last participant in master file processed. Values are:  N – No. Last participant not processed.  Y – Yes. Last participant processed. | System-assigned |
| SD037 | SDAllow0Pr | Allow Zero Price Flag. Allows market value calculation (PA999) without a price on file. Values are:  Y – Yes. (Default)  N – No. A price is required to calculate market value (PA999). | 'N' value must be set by user. |
| SD039 |  | Transaction Date. The Current Transaction Date. | System-assigned |
| SD040 | SDYYMMDD | Date Conversion Field-1. Used for converting date formats. This field will infer a century designation and return an eight-digit date in CCYYMMDD format. | System or user. |
| SD041 | SDDays | Date Conversion Field-2. Used for converting date formats. This | System or user. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** |
|  |  | field will return the number of days since 01/01/1900. |  |
| SD042 | SDWeekday | Date Conversion Field-3 (Day of the Week format). Used for converting date formats. Values are:   1. – Monday 5 – Friday 2. – Tuesday 6 – Saturday 3. – Wednesday 7 – Sunday 4. – Thursday | System-assigned |
| SD043 | SDMMDDYY | Date Conversion Field-4. Used for converting date formats. This field will return a six-digit date in MMDDYY format. | System or user. |
| SD044 | SDJullian | Date Conversion Field-5. Used for converting date formats. This field will return a seven-digit Julian date in CCYYDDD format. | System or user. |
| SD045 |  | Date Conversion Field-6. Used for converting date formats. This field will return an eight-digit date in MMDDCCYY format. | System or user. |
| SD046 |  | Business Calendar Date Input. This field is set by the user to determine if a specified date is a business date. | Defined in the  OmniScript text by the user. |
| SD047 |  | Business Calendar Date Output. If SD046 is valued, this field contains the corresponding business date according to the Business Calendar textfile. If SD046 is a holiday, this field will contain the previous or following business date. | System defined based on the value of SD046. |
| SD050 |  | OmniScript Return Code | System-assigned |
| SD051 | SDEnvir | Environment Operating System (OS)  MVS NT  UNIX  AS400 | System-assigned |
| SD052 | SDReleaseID | Plus Release Identifier including ptf level  (e.g., „5.20.20', '5.30.01', 'DEV2‟) | System-assigned |
| SD053 | SDRelDate | Plus Release Date | System-assigned |
| SD054 | SDOrigPgm | Originating Program – highest level program (e.g., OCRUN) | System-assigned |
| SD055 | SDEditMode | Edit Mode  P – Post  E – Edit  T – Trade Advance  X – Edit Error  T – Trade Error | System-assigned |
| SD056 | SDProcMethod | Process Method  B – Batch  O – Online | System-assigned |
| SD057 | SDCompiler | Compiler  COB2 – MVS COBOL  MFC – Merant COBOL  AS4 – AS400 | System-assigned |
| SD060 |  | Text String 1 - Text Work Area. Stores up to 40-character alphanumeric value that can be retrieved by providing start character and length. Works with SD061 below. | Defined in the  OmniScript text by the user. |
| SD061 |  | Text String 1 - Number Work Area. The first of two fields (SD066 is the other) provided to convert numeric OmniScript values to text values. A numeric value entered in this field is converted to a text value and placed in the first 15 bytes of SD060. The '@' parameter is used to designate the format of the value placed in SD060. It is expressed as SD061@00#. Values for the '@' parameter used to define format are:  @001 - ZZZ,ZZZ,ZZ9.99  @002 - ZZZ,ZZZ,ZZ9.999999  @003 - ZZZ,ZZZ,ZZ9.999 | Defined in the  OmniScript text by the user. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** |
|  |  | @004 - ZZZ,ZZZ,ZZ9.9999  @005 - ZZZ,ZZZ,ZZ9  @006 – 99B99B99 |  |
| SD065 |  | Text String 2 - Text Work Area. Stores up to 40-character alphanumeric value that can be retrieved by providing start character and length. Works with SD066 below. | Defined in the  OmniScript text by the user. |
| SD066 |  | Text String 2 - Number Work Area. The second of two fields (SD061 is the other) provided to convert numeric OmniScript values to text values. A numeric value entered in this field is converted to a text value and placed in the first 15 bytes of SD065. See definition of SD061 above for values and correct formatting in OmniScript text. | Defined in the  OmniScript text by the user. |
| SD068 |  | Text String - Start Character Position. Numeric position of the starting character within the text string. This field referenced if the first two bytes of the „@‟ parameter are „00‟. | Defined in the  OmniScript text by the user. |
| SD069 |  | Text String - Length. Number of characters to be referenced from the text string. This field referenced if the third byte of the „@‟ parameter is „0‟. | Defined in the  OmniScript text by the user. |
| SD070 |  | Voucher Number | System-assigned |
| SD071 |  | VTRAN Folder. Folder name of the file submitted for processing. | System-assigned |
| SD072 |  | PSW User ID. Online User ID from certain transactions that are being processed. | System-assigned |
| SD073 |  | Oper ID. Online Operator ID from certain transactions that are being processed. | System-assigned |
| SD075 |  | EACH Mode Flag. Controls records processed by the EACH@ command. Values are: 0 or P – Accesses PF records  1 or F – Accesses FC records  I – Accesses IC records  S – Accesses SC records   |  | | --- | | Note: This field is obsolete, but is included for backward compatibility. The improved syntax EACH\_PF,  EACH\_FC, EACH\_IC and EACH\_SC is documented in  the basic OmniScript concepts section of this user guide. | | Defined in the  OmniScript text by the user. |
| SD080 | SDLoopMax | MAX-LOOP-COUNTER. Overrides for maximum number of times a LOOP will execute. Default is 500. The maximum value is 999,999,999. | Defined in the  OmniScript text by the user. |
| SD085 |  | Override date for loan balances. Overrides the processing date when calculating the highest outstanding loan balance (LH991). | Defined in the  OmniScript text by the user. |
| SD090 |  | Trigger Event Name. The event name associated with an OmniScript.  (e.g., “RunStart”, “RunEnd”, “PlanStart”)  Refer to the **Base Event Processing guide** for more information. | System-assigned |
| SD091 |  | Events Enabled Flag. Indicates that base processing events have been enabled or disabled.  Y – Events are enabled.  N – Events are ignored or no events have been added.  SD091 is set to “Y” when events are added and may be set to “N” to disable event processing. | System-assigned, but may be overridden by the user in the  OmniScript text. |
| SD100 |  | OmniScript Function Numeric Return Code | System-assigned |
| SD101 |  | OmniScript Function Text Return Message | System-assigned |
| SD105 |  | OmniScript Return Code Level. This field is set when an OmniScript calls an external program (i.e., user exit).  0 – OK 3 – Error | System-assigned |

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** |
|  |  | 1. – Info 4 – Severe 2. – Warning 5 – Fatal |  |
| SD106 |  | OmniScript Return Code Reason This field is set when a OmniScript calls an external program (i.e., user exit).  00 – Reason OK  05 – Missing Routine  99 – Unimplemented COBOL program, i.e., module not available | System-assigned |
| SD200 |  | Current Transaction Active Flag. SD200 is the currently executing transaction active flag. It is set to 'Y' by the transaction processor at StartTran, and must be 'Y' after StartTran for the transaction to continue to process. SD200 can be set to 'N' to stop the current transaction from further processing. For example, setting it to 'N' in the middle of a plan level pass will cause the transaction to end without processing any more participants. | System-assigned |
| SD201 |  | Current Transaction Code (VTD060) | System-assigned |
| SD202 |  | Current Folder Name (VTD050 FILEID) | System-assigned |
| SD203 |  | Current Transaction Participant ID (VTD070)  Note: This data element is not always the same as SD002, since the SD203 may be 000000000 for plan level transactions | System-assigned |
| SD204 |  | Current Transaction Sequence Number (VTD090) | System-assigned |
| SD205 |  | Current Transaction Plan Num (VTD005) | System-assigned |
| SD206 |  | Current Transaction Date | System-assigned |
| SD207 |  | Current Transaction Document ID (VTD400) | System-assigned |
| SD208 |  | Current Transaction Usage Codes (VT580) | System-assigned |
| SD220 |  | Current Transaction Error Message Id. If a transaction error occurred, SD220 will contain the most recent error message ID for the transaction. Refer also to the OmniScript function **RPMsgs**, which allows full access to Current Transaction Error Messages. | System-assigned |
| SD221 |  | Current Transaction Error Parms. If a transaction error occurred, SD221 may contain any parameters used with the error message found in SD220.  Note: SD221 is limited to the first 14 characters of parameter data. Refer also to the OmniScript function **RPMsgs**, which allows full access to the parameters for Current Transaction Error Messages. | System-assigned |
| SD222 |  | Current Transaction Number of Errors. If a transaction error occurs, SD222 will contain the number of error messages issued for the transaction.  Refer also to the OmniScript function **RPMsgs**, which allows full access to the parameters for Current Transaction Error Messages. | System-assigned |
| SD230 |  | Skip This Participant. Can be set to 'Y' at PartStart, PartTranStart. Skips all transaction processing for the current participant. | Defined in the  OmniScript text by the user. |
| SD231 |  | Skip This Transaction. Can be set to 'Y' at TranStart. Skips processing this transaction. | Defined in the  OmniScript text by the user. |
| SD232 |  | Skip This Plan. Can be set to 'Y' at PlanStart. Skips processing this plan. | Defined in the  OmniScript text by the user. |
| SD901 | SDCR | The Carriage Return character. | System-assigned |
| SD902 | SDLF | The Line Feed character. | System-assigned |
| SD903 | SDFF | The Form Feed character. | System-assigned |
| SD904 | SDTab | The Horizontal Tab character. | System-assigned |
| SD905 | SDEsc, SDEscape | The Escape character. | System-assigned |
| SD906 | SDBel, SDBell | The Bell character. | System-assigned |

* + (FD) Function Data Elements

Function Data Elements provide an area of storage used to pass data and perform specific OmniScript keyword functions.

* Function data elements (FD) contain many of the data elements used by OmniPlus (i.e., Loan Amount, Payment Frequency, etc.). FD data elements allow you to *temporarily* enter and use this information to produce reports, accrual tables, amortization schedules, etc., without affecting the database.
* FDs are generally used in conjunction with keywords such as AMORT and BUMPDATE.

* + INITFD Keyword

Initializes all of the FD fields to zero or blank. Used to reset FDs between functions.

#### Example

FD101=14;

.. function call ..

##### INITFD; /\* All FDs reset to zero or blanks \*/ FD101=1;

.. another function call ..

The table below lists the function data elements available in the OmniScript facility.

|  |  |  |
| --- | --- | --- |
| **Field** | **Field Description** | **Valued By:** |
| FD101 | Interest Rate. Yearly interest rate being charged on a loan. | Set by user or value of PL746. |
| FD102 | Loan Amount. Amount of the original loan. | System |
| FD103 | Payment Frequency. Frequency of loan payments due. Values are:  1-Yearly 5-Semi-monthly  2-Semi-annually 6-Bi-weekly  3-Quarterly 7-Weekly  4-Monthly 8-Daily | System |
| FD104 | Payment Date. Date in CCYYMMDD format used as the sending date for the keyword BUMPDATE. If the 'CC' (century) is not supplied initially, the system will infer the century designation. | Defined in the  OmniScript text by the user. |
| FD105 | Next Payment Date. Next date in a series of payments in CCYYMMDD format. If the 'CC' (century) is not supplied, initially, the system will infer the century designation. | Defined in the  OmniScript text by the user. |
| FD106 | Last Payment Flag. Indicates whether the amortization has reached the last payment. Values are:  N-Not last payment  Y-Last payment | System |
| FD107 | Amortization Type Flag. Indicates amortization type. Values are:  0-Economic accrual method  3-Effective method | System |
| FD108 | Exact/Adjusted Interest Indicator. Indicates whether actual interest is calculated for the final loan payment, or the final payment is rounded to match all previous payment amounts. Values are:  0-Calculate actual interest for the final payment.  1-Round up interest to match other payments | System |

|  |  |  |
| --- | --- | --- |
| **Field** | **Field Description** | **Valued By:** |
| FD109 | Loan Use Indicator. Loan usage code which determines length of repayment. Values are:  0-N/A (repayment limited to 5 years maximum)  1-Principal residence (repayment up to 15 years maximum) | System |
| FD110 | Interest Applied to Principal. Amount of interest charged for the period not covered by the first loan payment. | System |
| FD111 | Days Interest. Overrides the calculation for payments in the amortization program. | Defined in the  OmniScript text by the user. |
| FD112 | Interest Paid. Indicates the total interest paid to date for the loan. | System |
| FD113 | Total Payments. Total number of payments for the loan requested. | System |
| FD114 | Payment Amount. Amount of the periodic loan payments. | System |
| FD115 | Total Expected Interest. Total dollar amount expected to be paid in interest for the loan. Required when processing the keyword AMORT with the Participant Loan Reports (T985) transaction code. | Defined in the  OmniScript text by the user. |
| FD116 | Amount Applied to Principal. Portion of the payment amount that was applied to the principle. | System |
| FD117 | Amount Applied to Interest. Portion of the payment amount that was applied to interest. | System |
| FD118 | Balance. Current balance of the loan (principal outstanding). | System |
| FD119 | Original Number of Payments. Total number of payments for the loan as determined by the amortization schedule. | System |
| FD120 | Loan Payment Date Flag. Indicates whether to use payment date from Loan Payment (T385) transaction or loan header record as the actual payment date. Values are:  0, 3 or 4-Use loan header date (LH311)  1-Use date on transaction  2-Use date on transaction, but alter sequence of repayment to pay all interest before paying any principal | System |
| FD121 | Date Direction. Indicates the direction desired for the keyword BUMPDATE. Values are:  F-Forward  B-Backward | Defined in the  OmniScript text by the user. |
| FD122 | Transaction Code. Indicates type of transaction. If a value is not entered while using the keyword AMORT, the default is '985'. Values are:  Loan Re-amort-383  Loan Repay-385  Loan Issue-384  Loan Report-985 | Defined in the  OmniScript text by the user. |
| FD130 | Annual Percentage Rate. Annual percentage rate of interest for the loan. | System |
| FD131 | Issue Date. Issue date of the loan. | System |
| FD132 | First Payment Date. Date of the first payment of the loan. | System |
| FD133 | Prepaid Fee. Amount of any prepayment fee for the loan. | System |
| FD134 | Processing Fee. Amount of any processing fee for the loan. | System |
| FD135 | Interest Balance Indicator. Indicates whether or not to calculate up-front interest for the loan (interest for the period between the issue date and the first payment). Values are:  0-Calculate up-front interest  1-Do not calculate up-front interest | System |
| FD200 | Fund ID. Fund-ID used with ACCRUAL and ACRURATE keywords. The Fund ID must be enclosed in apostrophes ( ' 10A ' ). | System |
| FD201 | End Date. Ending date for the ACCRUAL and ACRURATE keywords. If a value is not entered, the system defaults to the Run Date. | System |
| FD202 | Start Date. Start date for the ACRURATE keyword. If a value is not entered, the system defaults to PF-LAST-DIV-DATE or 40 day prior, whichever is greater. | System |
| FD203 | Accrual Amount. Dividend amount posted to the fund entered in FD200. | System |
| FD204 | Accrual Calculated Amount. Amount calculated for dividends for the fund entered in FD200. | System |
| FD205 | Cost. Cost of shares that could be purchased with dividend. | System |
| **Field** | **Field Description** | **Valued By:** |
| FD206 | Uninvested Cash. Amount of dividend that could be posted to uninvested cash. | System |
| FD207 | Shares Purchased. Number of shares that could be purchased with dividend. | System |
| FD208 | Accumulated Accrual Rate. Sum of all accrual rates on file in FD200 from the date entered in FD202 to the date entered in FD201. This applies only when the ACRURATE keyword is used. | System |
| FD209 | Plan Number. Overrides the PLAN# entered in PL907 to indicate where accrual rates are located. | System |

 Key Value (KV) Fields

Key value (KV) fields are used within the OmniScript facility to establish parameters and qualify the records that are to be accessed and used in OmniScript functions. Individual records can be accessed, or records can be counted and summed together. Records for selected files can be added, changed, or deleted. Caution should be exercised when using 'KV' fields because values are defined differently for different types of records. Records types that use „KV‟ fields for OmniScript access include, BR, BT, FA, PA and DR. Refer to the OmniScript Access section of the specific record type for a listing of applicable key value fields.

### Aliases

Aliases can be used in OmniScript statements in place of data element identifiers. The Literal Alias column of the following table lists the aliases for Key Value Field (KV) data elements.

|  |  |  |
| --- | --- | --- |
| **Field** |  | **Literal Alias** |
| KV001 | **Trade Date - Low**. Low value to be compared to the trade date in BR008. | **KVTDateLo** |
| KV002 | **Trade Date - High**. High value to be compared to the trade date in BR008. | **KVTDateHi** |
| KV003 | **Run Date - Low**. Low value to be compared to the run date in BR009. | **KVRDateLo** |
| KV004 | **Run Date - High**. High value to be compared to the run date in BR009. | **KVRDateHi** |
| KV005 | **Run Time - Low**. Low value to be compared to the run time in BR010. | **KVRTimeLo** |
| KV006 | **Run Time - High**. High value to be compared to the run time in BR010. | **KVRTimeHi** |
| KV007 | **Transaction Code**. Value to be compared to the transaction code in BR101. | **KVTran** |
| KV008 | **Activity Type**. Value to be compared to the activity type in BR102. | **KVActivity** |
| KV009 | **Sequence Option**. This field is used to select individual Base Records.     1. - Return and sum all qualified records for a Participant. 2. - Return only the first qualified record for a Participant. 3. - 9998 Return only the second qualified record, etc., up to the 9998th qualified record for a Participant.   9999 - Return only the last qualified record for a Participant. This field can be incremented by '1' each time a different Base Record is accessed until a non-qualifying record is returned for a Participant. | **KVSeqN** |
| KV010 | **Cash Value of Transaction - Low**. Low value to be compared to the cash amount in BR110. | **KVVAL1Lo** |
| KV011 | **Cash Value of Transaction - High**. High value to be compared to the cash amount in BR110. | **KVVAL1Hi** |
| KV012 | **Change in Share Balance - Low**. Low value to be compared to the share amount in BR121. | **KVVAL2Lo** |
| KV013 | **Change in Share Balance - High**. High value to be compared to the share amount in | **KVVAL2Hi** |

**Description**

|  |  |  |
| --- | --- | --- |
|  | BR121. |  |
| KV014 | **Uninvested Cash - Low**. Low value to be compared to the cash amount in BR130. | **KVVAL3Lo** |
| KV015 | **Uninvested Cash - High**. High value to be compared to the cash amount in BR130. | **KVVAL3Hi** |
| KV016 | **Share Cost - Low**. Low value to be compared to the cost in BR140. | **KVVAL4Lo** |
| KV017 | **Share Cost - High**. High value to be compared to the cost in BR140. | **KVVAL4Hi** |
| KV018 | **Other Cash - Low**. Low value to be compared to the amount in BR150. | **KVVAL5Lo** |
| KV019 | **Other Cash - High**. High value to be compared to the amount in BR150. | **KVVAL5Hi** |
| KV020 | **Other Shares - Low**. Low value to be compared to the share amount in BR151. | **KVVAL6Lo** |
| KV021 | **Other Shares - High**. High value to be compared to the share amount in BR151. | **KVVAL6Hi** |
| KV022 \* | **Usage Code 1 - Low**. Low value to be compared to data element BR104. | **KVUC1Lo** |
| KV023 \* | **Usage Code 1 - High**. High value to be compared to data element BR104. | **KVUC1Hi** |
| KV024 \* | **Usage Code 2 - Low**. Low value to be compared to data element BR105. | **KVUC2Lo** |
| KV025 \* | **Usage Code 2 - High**. High value to be compared to data element BR105. | **KVUC2Hi** |
| KV026 \* | **Usage Code 3 -Low**. Low value to be compared to data element BR106. | **KVUC3Lo** |
| KV027 \* | **Usage Code 3 - High**. High value to be compared to data element BR106. | **KVUC3Hi** |
| KV028 \* | **Usage Code 4 - Low**. Low value to be compared to data element BR107. | **KVUC4Lo** |
| KV029 \* | **Usage Code 4 - High**. High value to be compared to data element BR107. | **KVUC4Hi** |
| KV030 | **Voucher Number - Low**. Low value to be compared to data element BR172. | **KVVouLo** |
| KV031 | **Voucher Number - High**. High value to be compared to data element BR172. | **KVVouHi** |
| KV032 \* | **Usage Code 5 - Low**. Low value to be compared to data element BR250. |  |
| KV033 \* | **Usage Code 5 - High**. High value to be compared to data element BR250. |  |
| KV036 | **Pended Status Flag**. Value to be compared to data element BR171.  Legal values may be combined to filter based on multiple Statuses. For example, KV036="PI"; will return base records with a Pending or Immediate status. | **KVPendSt** |
| KV037 | **Base Text Type.** Represents the value to be compared to data element BT008 (Base Text Type). Refer the table in **BT008** for a complete list of Base Text Type values. |  |
| KV038 | **Key Data – Low.** Represents the low value to be compared to the Key Data stored in varying data elements depending on the Base Text Type stored in BT008. |  |
| KV039 | **Key Data – High.** Represents the high value to be compared to the Key Data stored in varying data elements depending on the Base Text Type stored in BT008 of the Base Text Record. |  |
| KV040 | **Plan Level BT/BR Indicator**. Used to determine if plan-level or participant-level BT or BR records are searched. Legal values are: N or blank - searches participant-level records  Y - searches plan-level records |  |
| KV041 | **Contract ID** (used only for GIC funds). Value compared to the Contract ID in DR012. Contract ID must contain seven (7) digits to the left of the decimal and four (4) to the right (N11D4). If Daily Price/Rate Retrieval Method (KV009) is set to '0', both the Trade Date (KV001) and Contract ID (KV041) are required fields. |  |
| KV045 | **Fund ID.** Value to be compared to data element BR100. |  |
| KV050 | **Post Number.** Value to be compared to data element BR170. |  |
| KV051 | **Plan ID Override.** Used to override the current processing Plan ID for cross plan access to BR records. This field should be reset to the current Plan ID once cross plan access is completed. |  |

**\*Note**: These KV fields are designed to handle numeric values only, and therefore cannot be assigned alphanumeric values. If BR's are to be filtered by usage codes which contain alpha characters, an "IF" statement should be utilized by the OmniScript.

 Key Value Access Rules

Certain rules apply to access of all key value fields. The rules include:

* A zero or undefined value in a KV field qualifies all records for retrieval
* Key value 'LO' and 'HI' ranges are inclusive
* A record must meet all KV selection criteria to be returned and/or summed
* Key Value Fields are initialized to zero between participants
* Once encountered in the text, the value of a KV remains constant until it is changed to another value elsewhere in the text
* To qualify records with a zero value, the 'LO' value must be set to '-.000001' and the HI value to '+.000001'
* The INITKV keyword is used to clear or reset all KV fields when changing access criteria

 INITKV Keyword

The INITKV keyword initializes all KV values in the OmniScript facility to zero. The keyword may be used anywhere in the OmniScript.

**Example**

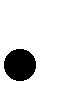
KV001=19990101; KV002=SD003; KV7=301; WK1=PA200@\*\*\*; INITKV; /\*INITKV resets all “KV” values to zero. “KV” values can now be re-defined for new record access and retrieval \*/

#### 4.3.5 Program Control Statements

Statements can be executed in a sequential order from beginning to end. However, most useful programs require decisions or carry out a process many times and therefore alter the flow of statement execution. To this end, OmniScript offers a set of programming commands that control the flow of your program. OmniScript commands are flexible, powerful, and easily understood.

|  |  |
| --- | --- |
| Conditional Statements | **IF / ELSE / END Statements** |
| Iteration Statements | **EACH@ / ENDEACH Statements** **LOOP / ENDLOOP Statements** |
| Termination Statements | **QUIT** **ABEND** |

 Conditional Statements

 IF / ELSE / END

The 'IF / ELSE / END' statements test a condition and execute statements based on the results of the test. An 'IF/END' statement tests a condition and may have up to four hundred intermediate work fields held within the one IF statement. Intermediate work fields hold the result of logical operations such as found in IF statements in OCSEM. If the condition is TRUE, OmniScript statements dictate the action to be taken before an “END” statement occurs. If the condition is FALSE, the OmniScript ignores the OmniScript statements for that participant, unless there is an ELSE statement. In this case, the OmniScript statements following the ELSE will be executed up until the END statement. For every “IF” statement, there must be an accompanying 'END' or 'ENDEXIT' statement. ENDIF may also be used in place of an END. See examples below.

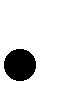
##### Simple 'IF/END'

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF (PH021=0);  PH600=1;  END; | Tests the condition 'PH021=0'. PH021 is participant status. If the participant has a status of '0' (active), this OmniScript sets the value of PH600 (a user-defined field) to 1. If the participant status is something other than '0', the OmniScript will ignore the participant. The 'END' statement is necessary to stop the 'IF' conditional query. |

The next example adds the 'ELSE' option to the conditional statement. 'ELSE' introduces OmniScript statements that dictate action to be taken if the condition is FALSE.

##### Simple 'IF/ELSE/END'

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF (PH021=0);  PH600=1; ELSE;  PH600=2;  END; | Again, tests the condition 'PH021=0'. PH021 is participant status. If the participant has a status of '0' (active), this OmniScript sets the value of PH600 (a user-defined field) to '1'. If the participant status is something other than '0', this OmniScript does not ignore the participant, but sets the value of PH600 to '2' (ELSE statement). The 'END' statement is necessary to stop the 'IF' conditional query. |

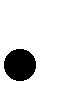
 ELSEIF

The ELSEIF; statement can be used in combination with an IF statement to allow checking a list of exclusive conditions, then ending with a single END statement. Use of the simple ELSE;IF combination requires an END statement for each IF statement.

Example of using the ELSEIF statement:

###### IF (WK1=1); TX1=‟a‟; ELSEIF (WK1=2); TX1=‟b‟; ELSEIF (WK1=3); TX1=‟c‟; ELSEIF (WK1=4); TX1=‟d‟; END;

 Iteration Statements

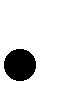
 EACH@ / ENDEACH

The 'EACH@ / ENDEACH' statements execute OmniScript statements against qualified participant fund (PF), fund control (FC), investment control (IC), or source control (SC) records. The type of record accessed (PF, FC, IC, or SC) is specified as a suffix on the EACH statement. The default value is participant fund (\_PF).

The 'EACH@' statement identifies a Fund Id for processing that may be generic or specific for both fund and source (10A, 10\*, \*\*A, or \*\*\*) in the format 'EACH@10A', or 'EACH@\*\*\*', etc. Every 'EACH@' statement must have an accompanying 'ENDEACH' statement. Any number of commands may be entered between the 'EACH@' initial statement and the 'ENDEACH' closing statement. An 'IF/ENDEXIT' statement may be used to exit the 'EACH' loop prematurely.

See the FUND ID USAGE section for the use of SD033, „?‟, „\*‟ for specifying and selecting fund ids.

* The EACH statement sets the current fund Id (SD033) in turn to the key value of each specified record.
* SD033 is restored to its prior value after the ENDEACH.

 LOOP / ENDEXIT / ENDLOOP

The 'LOOP / ENDEXIT / ENDLOOP' commands execute a block of statements repeatedly to accomplish a specified task. OmniScript instructions found between the 'LOOP' and 'ENDLOOP' iterators are executed until an 'IF/ENDEXIT' is encountered or the Max Loop Counter (SD080) exceeds the maximum loop value set by the user (default is 500). Instructions within the loop command must include an 'IF/ENDEXIT' conditional statement to determine if the loop process is complete.

The 'ENDLOOP' command redirects the process flow back to the first statement of the block to start the loop processing on the next participant. In the example below the OmniScript uses a 'LOOP/ENDLOOP' sequence to store loan balances in work fields for reporting. The loop looks for loan numbers up to '3' and records the loan number and the loan balance in indexed work fields.

###### **Simple 'LOOP/ENDLOOP'**

|  |  |
| --- | --- |
| **OmniScript Text** | **Explanation of OmniScript Statement** |
| SD030=100; | Sets the Indexed Work Field Base Value (SD030) to 100. So now IW000 will store information in WK100, IW001 in WK101, IW002 in WK102. |
| WK001=1; | Pre-sets WK001 to a value of '1'. This work field will be used to store the incrementing loan numbers. |
| LOOP; | Establishes a LOOP statement. All subsequent commands will be repeated until the OmniScript reaches a condition that ends the loop. |
| KL001=WK001; | Sets the loan number field to '1' (because WK001 was pre-set to this value). This ensures that each pass of the OmniScript for a participant starts with Loan 1. |
| IF WK001>3; ENDEXIT; | Determines if the loan number is greater than 3, and if so, ends the loop statement. |
| IF LH315>0; | Determines if the participant has a loan balance. LH315 is a loan header data element that contains current balance. |
| IW000=LH300; | Sets IW000 to the loan number. |
| IW001=LH315; | Sets IW001 to the loan balance. |
| SD030=+2; END; | Increments the Indexed Work Field Base Value by +2 so that now IW000 will store information in WK102, IW001 in WK103 for the next loan. |
| WK001=+1; | Increments the WK001 field that is used to identify the loan number by +1, so that on the second pass the OmniScript will gather information on Loan 2. |
| ENDLOOP; | Ends the LOOP statement. |

**Example**

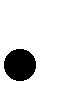
Task

Sum WK fields 1 through 50 into WK100 using Indexed Work (IW) fields.

SD30 = 0; /\* Initialize the IW subscript to zero \*/ LOOP; WK100 =+ IW1; SD030 =+ 1; /\* Increment the IW subscript) IF (SD30>=50); ENDEXIT; ENDLOOP;

Result

WK100 equals the sum of WK1 through WK50.

 LOOP WHILE/ UNTIL

A WHILE or UNTIL condition can be used with the loop, and is preferred over the IF/ENDEXIT processing.

LOOP UNTIL will process until the parameter condition is true.

LOOP WHILE will process until the parameter condition is false.

WK1=0;

LOOP WHILE WK1<5; ...

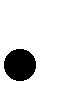
WK1=+1; ENDLOOP; WK1=0;

LOOP UNTIL WK1>5; ...

WK1=+1; ENDLOOP;

Eliminates the need to code ‟IF/ENDEXIT' within the loop.

 Termination Statements

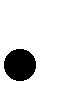
 QUIT

Normally, OmniScript execute until the end of the supplied statements at the bottom of the text. The 'QUIT' command is used with an 'IF' statement to exit the OmniScript immediately when a condition is met. This command differs from 'GOBACK' in that 'QUIT' always exits the OmniScript function completely, while 'GOBACK' only exits the current routine. In the example below the OmniScript is designed to exit if the participant status is terminated (31). Otherwise, work fields are used to store contribution, cash, and share balances.

###### **Simple 'QUIT'**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF PH021=31;  QUIT;  END;  WK001=PF200@10\*;  WK002=PF120@10\*;  WK003=PF130@10\*; | Examines a participant's status. If the status is '31' (terminated), the QUIT command is executed and the OmniScript goes to the next participant. If the status is not '31', the rest of the OmniScript executes to store contributions, cash, and shares in work fields. |

Note: The QUIT statement does not necessarily stop output from printing. When using a T966 consider setting the Report Selection Indicator (WK212) equal to “1” before using the QUIT statement."

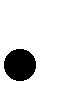
 ABEND

A second termination command is the 'ABEND' statement. This statement terminates the current job step for all plans processing, and on some platforms (e.g. MVS) generates a system dump for diagnostic purposes. 'ABEND' is typically reserved for debugging and locating serious problems. The command generates a message that states that the abnormal ending of the OmniScript was requested by the OmniScript text based on a conditional statement.

#### 4.3.6 Subroutines

 PERFORM/ ROUTINE/ GOBACK

OmniPlus OmniScript subroutines allow the user to define and invoke blocks of OmniScript statements from multiple points in a OmniScript. This eliminates needless repetition of the same OmniScript statements. Subroutines simplify program coding and maintenance and make OmniScript text easier to read and understand. The three basic subroutine instructions: 'PERFORM', 'ROUTINE', and 'GOBACK'.

 PERFORM

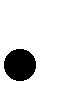
The 'PERFORM' instruction tells the OmniScript facility to find a specified 'ROUTINE' keyword and execute the OmniScript statements found in the 'ROUTINE' statement block before executing the remainder of the OmniScript statements. Normally the name of the 'ROUTINE' is text literal (the exact name of the program in single quotes), but it is possible to use a text variable name that is constructed through a series of statements. If the 'ROUTINE' specified is not found in the OmniScript file, a compile error occurs.

The 'PERFORM' command is formatted as follows (with an example 'ROUTINE' called 'INVEST.CHECK'):

##### PERFORM 'INVEST.CHECK';

The „PERFORM‟ command may also be executed for a variable routine name. When an OmniScript attempts to perform a routine with a variable name, **SD105** is set to “0” if the routine executes successfully and is set to a non-zero value otherwise (i.e., the routine does not exist).

##### TX001 = "INVEST.CHECK"; PERFORM TX001; /\* variable routine name \*/ IF SD105 = "0"; TX001 = "ROUTINE CALL SUCCESSFUL"; ELSE; TX001 = "ROUTINE CALL UNSUCCESSFUL"; TX002 = "RETURN CODE:" + SD105; TX003 = "RETURN REASON CODE:" + SD106; END;

 ROUTINE

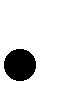
The 'ROUTINE' instruction defines a subroutine name and the beginning of a block of OmniScript statements. The block of statements consists of all OmniScript statements that immediately follow the 'ROUTINE' instruction and includes all OmniScript statements until either another 'ROUTINE' statement is encountered or the end of the program is reached. Because of this unique construction of blocks of OmniScript statements, 'ROUTINE' commands and statement blocks must be coded at the end of the OmniScript text file.

**Note: If the routines are not coded at the end of the text file, the OmniScript statements following the last routine will be ignored.**

'ROUTINE' names can be up to 20 characters in length. Multiple subroutines can be stored in a single text file. **However, the 'ROUTINE' instruction cannot be used within an 'IF', 'LOOP', or 'EACH@' command.** But, all of these commands can be part of a subroutine.

The 'ROUTINE' command is formatted as follows (with example 'ROUTINE' instructions called 'PH.CHECK' and 'STATUS.CHECK').

##### ROUTINE 'PH.CHECK'; ['PH.CHECK' OMNISCRIPT STATEMENTS] ROUTINE 'STATUS.CHECK'; ['STATUS.CHECK' OMNISCRIPT STATEMENTS] <END OF FILE>

 GOBACK

The 'GOBACK' instruction redirects the process flow to the first statement after the preceding

'PERFORM' instruction. This is an optional instruction and is not required to be used with the 'PERFORM' or 'ROUTINE' instructions.

The example below provides a simple example of the 'PERFORM' and 'ROUTINE' instructions. This sample verifies that investment elections are made in 25% increments. If they are not, the OmniScript outputs a warning message.

##### Simple 'PERFORM / ROUTINE / GOBACK'

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| EACH@\*\*\*;  WF001=PF040; | Initiates an 'EACH@' statement for all funds. Sets work fund WF001 to the value of the allocation percentages on file for the participant (PF040). |
| PERFORM 'ALLOC.CHECK'; | Instructs the OmniScript to execute the  'ALLOC.CHECK' subroutine, which is defined at the bottom of the OmniScript. |
| IF (WK007=1);  WK008=+1;  END;  ENDEACH; | After executing the 'ALLOC.CHECK', the error flag (WK007) is examined. If the flag is equal to '1', the error counter (WK008) is incremented by '1'. |
| IF (WK008>0);  TM001='INVEST % NOT 25%  INCRMENTS';  END; | Examines the error counter (WK008). If it is greater than '0', outputs the warning message that the investment allocations are not in increments of 25%. |
| **OmniScript statement** | **Explanation** |
| ROUTINE 'ALLOC.CHECK';  WK007=1;  IF (WF001=0.00) OR  (WF001=0.25) OR  (WF001=0.50) OR  (WF001=0.75) OR  (WF001=1.00);  WK007=0;  END; | Defines the 'ALLOC.CHECK' routine. In this routine, WK007 is initialized to the error setting. Then the participant allocations are examined to ensure that the are in increments of 25%. If they are, the error setting is changed back to '0'. |
| GOBACK; | Instructs the OmniScript to return to the first statement after the 'PERFORM' instruction. |

#### 4.3.7 Date Conversions

System-defined environment data elements SD040 through SD045 are reserved for converting date formats within OmniScript text. The date to be converted is stored in the system-defined field (SD) that has the same format. For example,

**SD040=PH050**; stores participant birthdate in YYMMDD format.

Now this date can be retrieved in other formats using SD041 through SD045. This same date is available in all of the date conversion fields until another date is assigned to one of the 'SD' date fields in the OmniScript. See the system-defined data element table previously presented in this book for the various formats available.

The date conversion fields can be used to calculate a variety of date parameters in all of the formats designated in the 'SD' fields. Some simple examples follow. The first example calculates the length of employment for a participant to determine whether a matching contribution can be made.

##### Simple Date Calculations - Example 1

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| SD040=SD003; | Sets SD040 equal to the system date in YYMMDD format. |
| WK001=SD041; | Now stores the system date in SD040 in the format designated by SD041 (number of days since 00/01/1900) in WK001. |
| SD040=PH054; | Resets the value of SD040 to the participant's plan entry date (PH054) in YYMMDD format. |
| WK002=SD041; | Stores the plan entry date in SD040 in the format designated by SD041 (number of days since 00/01/1900) in WK002. |
| WK003=WK001-WK002; | Stores the difference of WK001 and WK002 in WK003. This number is the length of employment (system date minus plan entry date) in days. |
| IF WK003 > 600;  TM1='MATCH ELIG';  END; | If the participant has been employed for more than 600 days, prints a message indicating eligibility for a matching contribution. |

The next example below adds 100 days to a defined date and stores the new date in various formats.

##### Example 2

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| SD040=19991001; | Sets SD040 equal to October 1, 1999. |
| SD041=+100; | Adds 100 days to the value of SD041 (number of days since  00/01/1900). This changes the value of all 'SD' date fields to 100 days after October 1, 1999. |
| WK001=SD040; | Stores the new date value in SD040 (YYMMDD format) in WK001. |
| WK002=SD045; | Stores the new date value in SD045 (MMDDCCYY format) in WK002. |

The final example below converts a standard date format to Julian format and stores it in a work field.

##### Example 3

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| SD040=PH050; | Sets SD040 equal to the participant birthdate in YYMMDD format. |
| WK001=SD044; | Stores the birthdate set in the first statement in Julian format (YYDDD) in WK001. |

#### 4.3.8 BUMPDATE Keyword

Used in conjunction with the function data elements (FDs), the BUMPDATE keyword is used to change the current date forward or backward in defined increments. This keyword uses FD103, FD104, FD105 and FD121 to perform this function. See the function data element table previously presented in this book for definitions of these fields. The example below illustrates how the BUMPDATE keyword is used with the function data elements.

##### Simple BUMPDATE Function

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| WK001=SD003; | Stores the current system date in WK001. |
| FD104=WK001; | Sets the date function field for 'Start Date' (FD104) equal to the date stored in WK001 (current system date). |
| FD121='F'; | Sets the BUMPDATE function to calculate a date forward ('F'). |
| FD103=1; | Sets the BUMPDATE function frequency to a year (1). |
| BUMPDATE; | Instructs the system to 'bump' the date in FD104 (Start Date) forward ('F') by 1 year (1). |
| WK002=FD105; | Stores the 'Next Date' (FD105) in WK002. This field will now contain the 'Start Date' (FD104) bumped forward 1 year. |

## 4.4 Omni Functions

### 4.4.1 Overview

OmniScript supports Function Libraries. Function operations allow invoking pre-written COBOL modules, passing parameters, and returning a result. Functions offer more concise and readable statement coding and increased performance.

* Features
  + Functions can return Text or Numeric Values
  + Functions can be passed up to 100 Parameters
  + Parameters can be labeled (e.g. DATE:SD003) or „Positional unlabeled‟
  + Functions may also return a numeric return code (**SD100**), and a text return message (**SD101**)
  + **TRACE facilities** are available to aid in Function development and use
  + Function libraries greatly increase the power and capabilities of OmniScript

* Syntax

Functions are invoked using the following syntax:

#### xxYYYYY\_OOOOO (LABEL: Value)

Where:

xx is an object identifier

YYYYY is a 1-5 character library identifier

xxYYYYY is the module name of the COBOL function library processor

OOOOO is a 1-23 character operation identifier

LABEL: is a 1-12 character parameter label identifier

Value is a text or numeric variable or literal

In the example below:

* „OC‟ is the object identifier (OmniCalc)
* „TEXT‟ is the library identifier
* „OCTEXT‟ is the COBOL module name
* „PROPER‟ is the operation
* „FIELD‟ is an optional parameter label
* „PH011‟ is a required variable text parameter

#### OCTEXT\_PROPER (FIELD: PH011); /\* returns name in proper case \*/

* Invocation The table below contains examples of the different methods used to invoke functions.

|  |  |
| --- | --- |
| OCFILE1\_CLOSE(); | Entire statement, return value not used |
| IF BARUN\_PLSTART(„111111‟); | Return value checked by an IF |
| WK1=OCNUM\_AVG(WK2 WK3 WK4) / 2; | Used in an assign statement |
| OCNUM\_MIN(WK1\*5 WK2) | An expression passed as a parameter |
| OCSHOW(OCNUM\_MAX(WK1 WK2 WK3)); | A function value passed as a parameter to another function |
| LOOP WHILE PTPHOBJ\_NEXT(); | Return value checked by a Loop While condition |

**Note: Since OmniScript interprets 0 as False, and any other value as True, functions which return 0 or 1 can be checked by IF or LOOP WHILE/UNTIL statements.**

* Conventions

Because parameters vary by operation, the following conventions have been used throughout the documentation to indicate the type of label/parameters that can be passed to a function.

* Brackets „[ ]‟around parameters indicate an optional component
* An „\*‟ indicates it can occur multiple times
* „tx‟ indicates a Text string value (variable or literal)
* „num‟ indicates a numeric value (variable or literal)

**Example**

#### Parameter Indicates

|  |  |
| --- | --- |
| [TEXT:]tx | The TEXT: label is optional, but the text value is required |
| TEXT:tx | A labeled parameter is required |
| [tx] | An optional non-labeled text value |
| [TEXT:tx] | An optional labeled parameter |
| Num | A required numeric value |
| Num\* | Multiple unlabeled numeric parameters |

### 4.4.2 Object Access Functions

 Purpose

The following section gives basic information common to all OBJ type function libraries. See the particular OBJ library for detailed information particular to that record (e.g. \_VIEW parameters). The xxyyOBJ$ modules provide OmniScript access to the records and other structures of OmniPlus. Each OmniPlus object (e.g. record type) will have a xxyyOBJ module to access the instance (record), and the individual Data Elements in it. In addition, an xxOBJ module may exist to reference the entire (Virtual) object.

**Example**

TXTFOBJ and TXTXOBJ function reference textfile header and textfile data records.

\* Delete the specified textfile header record, but not the associated detail (lines) TXTFOBJ\_DELETE(PLAN:'111111' FILENAME:'TEST.TEXTFILE');

TXOBJ references the entire textfile (header plus all detail lines).

#### \* Delete the textfile header record and all textfile detail (lines) TXOBJ\_DELETE(PLAN:'111111' FILENAME:'TEST.TEXTFILE');

 Terms and Definitions

**INSTANCE** - Refers to a unique database physical record (e.g. PH-REC), or sometimes to a collection of records (e.g., Participant) consisting of more than one record.

**VIEW -** Identifies criteria selecting a set of records for following sequential processing (usually via NEXT operations).

**OBJID** - Object ID is a TEXT field identifying the object. The \_OBJID operation of data structure libraries returns this text field. OBJID‟s are readable, and can be displayed to uniquely identify the object, or extracted/stored and later used to re-fetch the object. OBJID‟s may be passed as parameters in VIEW or GET operations to provide key fields.

#### Common Object Access Operations

|  |  |  |
| --- | --- | --- |
| Area | Operations | Description |
| View Oriented | **View**  **First**  **Next**  **Prev**  **Last**  **Count** | Define range for accessing records  Access the first qualified record  Access the next record within a VIEW  Access the previous record within a VIEW\*  Access the last qualified record\*  Count all qualified records |
| Record Level | **Init** | Initialize a record |

|  |  |  |
| --- | --- | --- |
|  | **OBJID**  **Get**  **GetGE**  **GetGT**  **GetLT**  **Delete**  **Copy**  **Move**  **Update**  **Add**  **Error**  **ErrMsg** | Return a record‟s unique Object identifier  Access a specific record  Access a record >= to a specified key  Access a record > a specified key  Access a record < a specified key  Delete a record  Copy a record  Move (Copy and Delete) a record  Update (Change) a record  Add (write) a record  Return an error code  Return an error message |
| Data Element Level (DE) | **DE**  **NumDE**  **SetDE**  **DEFmtLeng**  **DEName**  **DENext**  **DEPic**  **DEType**  **DEValDesc**  **DEValKey**  **DEValKeyN**  **DEValid**  **DEValName**  **DEXML** | Return the alphanumeric value of a DE  Return the numeric value of a DE  Set the value of a DE  Return the length of a formatted DE  Return a DE‟s name  Return the next DE number  Return the picture clause of a DE  Return a DE‟s type  Return the description for a DE value  Return the specified legal value (alphanumeric)  Return the specified legal value (numeric)  Validate a DE Number or Name  Return a DE‟s legal value name  Return a DE in descriptive XML format |
| Clipboard | **ToClip**  **ToClipAdd** | Copy record to the clipboard in XML or other format  Append record to the clipboard in XML or other format |
| Other | **Validate1**  **Validate2** | Verify all fields contain legal values  Verify all “foreign keys” are valid |

 View Oriented Operations

#### \_VIEW – Define range for accessing records

The VIEW operation sets the scope of the records to be returned by following \_NEXT operations. Parameter Fields provided with the \_VIEW can identify filters and keys so the user can examine only the records of interest. If no \_VIEW parameters are given, all records on the system database will be qualified.

##### Parameters

Key or Filter parameters, which vary by record type, are allowed.

Parameters fields vary with the particular function library based on the type of record. View parameters will include each Key field, usually with LO and HI values (e.g. TRADEDATELO/TRADEDATEHI). Filter parameters may also be allowed, such as STATUS:'31' or FILTFUND:'\*\*A'. See the individual object for its provided filters and key ranges.

|  |  |
| --- | --- |
| **Parameters** | **Description** |
| [OBJID:tx] | An OBJID, must be one of this object‟s ids, or a compatible object‟s id |
| [LIMIT:num] | Limits the maximum number of entries to be returned by \_NEXT. May be used to test processes on smaller volumes. After COUNT is exceeded, succeeding NEXT will return End of File. LIMIT: can be used when testing, or to limit accesses to reasonable values. |
| [STATUS:] | Record Dependent TEXT field to filter VIEW/NEXT fetches |

##### Example

The PLPLOBJ library allows the following \_VIEW filters:

**Parameter Description** [PLAN:tx] Plan Identifier

[PLANLO:tx] Lo plan number to start with

###### [PLANHI:tx] Stop after processing this plan number

The PTPHOBJ library allows the following \_VIEW filters:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [PLAN:tx] | Plan Identifier |
| [PLANLO:tx] | Lo plan number to start with |
| [PLANHI:tx] | Stop after processing this plan number |
| [STATUS:tx] | Qualifies only participants with PH-STAT indicating „ACTIVE‟, „INACTIVE‟  „SUSPENDED‟, „RETIRED‟, „DEFERRED‟, „TERMINATED‟, and „INELIGIBLE‟ |
| [STATUS:‟30‟] | Qualify only participants with PH-STAT = „30‟ |

VIEW Parameters vary by object and may include:

* Each key field, with a LO/HI value. (e.g. PLANLO:'000001', PLANHI:'222222')
* Filter criteria for various key or data fields. (e.g. STATUS:'31' or FUNDFILT:'1\*\*' or

RUNDATELO:19980101)

#### \_FIRST – Access the first qualified record

Specify access filters and get first occurrence

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

##### Returns

1 (True) if a record was found

0 (False) otherwise

##### Example

PTPHOBJ\_FIRST(PLAN:'777777'); \* Returns the first PH record occurrence in plan '777777'

#### \_NEXT – Access the next qualified record within a VIEW

Return next instance within a VIEW. A previous VIEW operation is required.

**Parameters**

None

##### Returns

1 (True) if a instance is available, 0 (False) Error or End of File

After a successful \_NEXT, there is a current record with which to perform record or data element operations.

**Note:If [Limit] is used and qualifying records exist beyond the set limit, \_Next will terminate and load SD101 with ‘Exceeded Given Limit’, SD100 with a ‘4’.**

##### Example

PTPHOBJ\_VIEW(PLAN:'777777'); LOOP WHILE PTPHOBJ\_NEXT(); OCSHOW (PTPHOBJ\_DE("NAME")); ENDLOOP;

#### \_PREV – Access the previous qualified record within a VIEW

Return previous instance within a VIEW. A previous VIEW operation is required. This operation is identical to NEXT, except that records are read in reverse order.

|  |
| --- |
| **Note: The PREV operation is only valid on the Windows and UNIX platforms. The PREV operation is not available on the Mainframe (MVS) platform.**  **Note: The PREV operation is currently only valid with the following Object Access Functions: BAI2OBJ, CMCMOBJ, CMSLOBJ, DNCROBJ, DNPBOBJ, DNPNOBJ, PLPLOBJ, PTPHOBJ, SVSTOBJ, and SVSVOBJ.** |

**Parameters**

None

##### Returns

1 (True) if a instance is available, 0 (False) Error or Beginning of File

After a successful \_PREV, there is a current record with which to perform record or data element operations.

##### Example

PTPHOBJ\_VIEW(PLAN:'777777'); LOOP WHILE PTPHOBJ\_PREV (); OCSHOW (PTPHOBJ\_DE("NAME")); ENDLOOP;

#### \_LAST – Access the last qualified record

Specify access filters and get last occurrence.

**Note: The LAST operation is currently only valid with the following Object Access Functions: BAI2OBJ, CMCMOBJ, CMSLOBJ, DNCROBJ, DNPBOBJ, DNPNOBJ, PLPLOBJ, PTPHOBJ, SVSTOBJ, and SVSVOBJ.**

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

##### Returns

1 (True) if a record was found

0 (False) otherwise

##### Example

PTPHOBJ\_LAST(PLAN:'777777'); \* Returns the last PH record occurrence in plan '777777'

#### \_COUNT – Count the number of qualified records

Executes a VIEW and then NEXT functions, counting each record, returning the total number of qualified records.

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

**Returns**

Number of Object Instances in this VIEW.

##### Example

###### WK1=PTPHOBJ\_COUNT(PLAN:'777777'); \* Stores the number of participants in plan '777777' in WK1

**Note: \_COUNT does in fact pass all qualified records to count them. For performance reasons, you should only use \_COUNT if you will not pass the records yourself with \_VIEW/\_NEXT. If you will process the records, it will be more efficient (especially for large record counts) to tally the records yourself as you process them; otherwise passing the records twice will be inefficient.**

 Record Level Operations

#### \_INIT – Initialize a record

INITIALIZE function to clear all fields in the current record. Can be used prior to valuing fields for an Add/Write operation. Clears each field in the instance (record).

**Parameters**

Key parameters, which vary by record type, are allowed.

##### Example

PTVROBJ\_INIT(PLAN:'777777' PARTID:PH007); PTVROBJ\_SETDE(DENUM:100 VALUE:'5789'); PTVROBJ\_SETDE(DENUM:105 VALUE:4); PTVROBJ\_ADD();

#### \_OBJID – Return a record’s unique object identifier

Return a text field with the object identifier

**Parameters**

None

**Returns**

A text field with the object identifier

##### Example

PTPHOBJ\_VIEW(PLAN:SD001); PTPHOBJ\_NEXT(); TX001=PTPHOBJ\_OBJID(); \* RETURNS: 'OBJID:PTPH PLAN:111111 PARTID:000000001'

#### \_GET – Access a specific record

Get using specified key fields or OBJID

**Parameters**

Key parameters, which vary by record type, are allowed.

**Returns**

1 (True) if record was found, 0 (False) otherwise

##### Example

PTPHOBJ\_GET(PLAN:'111111' PARTID:'222222222');

#### \_GETGE – Access a record greater than or equal to the specified key

Get greater than or equal to (>=) using specified Key fields or OBJID

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

##### Example

###### PTPHOBJ\_GETGE(PLAN:'777777' PARTID'000000001'); TX001=PTPHOBJ\_OBJID();

**Note: Key information can be specific (e.g. PLAN:'111111' PARTID:'222222222') or an ObjectID (e.g. OBJID:TX1) or generic (e.g. PLAN:'111111'')**

#### \_GETGT – Access a record greater than to the specified key

Get greater than (>) using specified Key fields or OBJID

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

#### \_GETLT – Access a record less than the specified key

Get less than (<) using specified Key fields or OBJID

##### Parameters

Key or Filter parameters, which vary by record type, are allowed.

**Note: The GETLT operation is currently only valid with the following Object Access Functions: BAI2OBJ, CMCMOBJ, CMSLOBJ, DNCROBJ, DNPBOBJ, DNPNOBJ, PLPLOBJ, PTPHOBJ, SVSTOBJ, and SVSVOBJ.**

#### \_DELETE – Delete the specified or current record

Delete specified Occurrence

##### Parameters

Key parameters, which vary by record type, are allowed.

If key information is given, delete the specified instance. If no parameters are given, delete the current instance

**Returns**

1 (True) if an instance is deleted, 0 (False) if no delete occurred due to Error/End of File

##### Example

IF DSDSOBJ\_GET (PLAN:'777777' DIVSUB:'1234'); DSDSOBJ\_DELETE (); TX001='REC DELETED'; ELSE; TX001='DELETE FAILED'; END;

#### \_COPY – Copy the current record

Copy specified occurrence, using given Key fields. Makes a copy of the current record, and writes it using the given new Key fields.

**Parameters**

Key parameters for the new record.

##### Returns

1 (True) if an instance is copied, 0 (False) if no COPY occurred due to Error

**Note: If [Plan:] or [Partid:] is not coded for a new plan or participant, SD101 will be loaded with ‘Cannot copy onto self’.**

##### Example

IF PTPHOBJ\_GET(PLAN:'777777' PARTID:'444000000'); /\* copy the current PTPH record from plan 777777 to plan 777780 \*/ PTPHOBJ\_COPY(PLAN:'777780'); TX001='PH RECORD COPIED'; ELSE; TX001='COPY FAILED'; END;

#### \_MOVE – Move the current record

Delete specified occurrence and copy to given Key

**Parameters**

Key parameters for the new record

**Returns**

1 (True) if an instance is moved, 0 (False) if no MOVE occurred due to Error

##### Example

IF PTPHOBJ\_GET(PLAN:'777777' PARTID:'444000000'); PTPHOBJ\_MOVE(PLAN:'777780'); TX001='PH RECORD MOVED'; ELSE; TX001='MOVE FAILED'; END;

#### \_UPDATE – Update (Change) the current record

Rewrite current occurrence.

**Parameters**

None

**Returns**

1 (True) if an instance is copied, 0 (False) if failed occurred due to Error

##### Example

DSDSOBJ\_VIEW(PLAN:'777780'); LOOP WHILE DSDSOBJ\_NEXT(); DSDSOBJ\_SETDE(DENUM:200 VALUE:1); DSDSOBJ\_UPDATE(); ENDLOOP;

**Note: This function cannot be used to change the Key fields of a record.**

#### \_ADD – Add the current record

ADD the current occurrence to the database

**Parameters**

None

**Returns**

1 (True) if add was successful, 0 (False) otherwise

##### Example

\* ADD PIN NUMBER PTVROBJ\_INIT(PLAN:SD001 PARTID:PH007); PTVROBJ\_SETDE(DENUM:100 VALUE:'5789'); PTVROBJ\_SETDE(DENUM:105 VALUE:4); PTVROBJ\_ADD();

#### \_ERROR – Return the error status of the previous IO operation

**Parameters**

None

##### Returns

1 (True) if the last IO operation terminated in error, 0 (False) if the last IO operation was successful

##### Example

PLPLOBJ\_GET(PLAN:'111111'); /\* Attempt to read this record \*/ WK1=PLPLOBJ\_ERROR(); /\* WK1 might be 0 indicating success, or 1 indicating No such record \*/

#### \_ERRMSG – Return the error message of the previous IO operation Return code and message of previous IO operation

**Parameters**

None

##### Returns

A TEXT message containing the File Status and a description of the last IO completion FILESTATUS

##### Example

###### PLPLOBJ\_GET(PLAN:'111111'); /\* Attempt to read this record \*/ TX1=PLPLOBJ\_ERRMSG(); /\* TX1 might be '00 - Successful' or '23 - Record not Found' \*/

 Data Element Level Operations

Once a record is current, either by reading it or by initializing an empty instance, the DE operations can retrieve or set values.

#### \_DE – Return the text value of a data element

Fetch Text value of the specified Data Element. If necessary, converts a numeric element to it‟s TEXT equivalent using its default format. If the actual numeric value is wanted, use \_NUMDE to fetch it.

##### Parameters

Either of the following:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats:   * The name as it appears on the Omni Station Fields tab * The data element number in 999 or DE999 format * The data element in YY999 format, were YY is the record type (e.g.   “PH011”)  (NOTE: This entry is NOT case-sensitive.) |

**Returns**

Text value of the Data Element

##### Examples

PTPHOBJ\_GET(PLAN:'777777' PARTID:'1230004567'); TX1=PTPHOBJ\_DE(011); /\* OR \*/ TX1=PTPHOBJ\_DE('NAME'); /\* OR \*/ TX1=PTPHOBJ\_DE('naMe'); /\* OR \*/ TX1=PTPHOBJ\_DE(NAME:'011'); /\* OR \*/ TX1=PTPHOBJ\_DE(NAME:'DE011') /\* OR \*/ TX1=PTPHOBJ\_DE(NAME:'PH011');

#### \_NUMDE – Return the numeric value of a data element

Fetch Numeric value of Data Element

##### Parameters

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats:   * The name as it appears on the OmniStation Fields tab * The data element number in 999 or DE999 format * The data element in YY999 format, were YY is the record type (e.g.   “PH011”)  (**NOTE:** This entry is **NOT** case-sensitive.) |

**Returns**

The value of the specified Data Element

##### Example

PTPHOBJ\_VIEW (PLAN:'777777'); PTPHOBJ\_NEXT (); TX1=PTPHOBJ\_NUMDE('SALARY'); /\* OR \*/ TX1=PTPHOBJ\_NUMDE(170);

#### \_SETDE – Set the value of a data element

Sets the specified value of the specified Data Element

##### Parameters

Either of the following:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats:   * The name as it appears on the Omni Station Fields tab * The data element number in 999 or DE999 format * The data element in YY999 format, were YY is the record type (e.g.   “PH011”)  (**NOTE:** This entry is **NOT** case-sensitive.) |

Followed by:

**Parameter Description**

[VALUE:]tx or num The value to set the element to

##### Returns

1 (DE stored), 0 (Invalid field or value)

**Note: Returns SD101 loaded with ‘No Such Data Element: DE999’ if requested DE doesn’t exist. Returns SD101 loaded with ‘Mismatched Data Type: DE999’ for attempts to give numeric data elements text values (or vice versa).**

##### Example

PTPHOBJ\_VIEW(PLAN:'777777'); PTPHOBJ\_NEXT (); PTPHOBJ\_SETDE(015 '100044'); /\* OR \*/ PTPHOBJ\_SETDE(DENUM:015 VALUE:'100044'); /\* OR \*/ PTPHOBJ\_SETDE(NAME:'EMPL-NUM' VALUE:'100044');

#### \_DEFMTLENG – Get the formatted length of a data element

Return the # of bytes required to show a data element. For Text elements, this will be the # of bytes. For numeric elements, it will be the total formatted length, including any formatting characters such as commas, decimals, signs, slashes, etc. This operation should be used for output formatting (such as HTML screens or reports) where it is necessary to get the formatted length of a field.

##### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]:num | The selected data element number. |

**Returns**

The number of characters in the actual formatted length of the element.

##### Example

\* Show all legal values and descriptions for PH170 (salary) WK1=PTPHOBJ\_DEFMTVAL(170);

#### \_DENAME – Return the name of a Data Element

##### Parameters

Either of the following:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats:   * The name as it appears on the Omni Station Fields tab * The data element number in 999 or DE999 format |

- The data element in YY999 format, were YY is the record type (e.g.

“PH011”)

(**NOTE:** This entry is **NOT** case-sensitive.)

##### Example

TX1=PTPHOBJ\_DENAME(DENUM:'021'); /\* Returns: „STAT‟ \*/

#### \_DENEXT – Return the next valid data element number

Return the next highest Valid DE number for this record. DENEXT can be used to sequentially fetch all the elements for a record, in data element order.

##### Parameters

Either of the following:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats:   * The name as it appears on the Omni Station Fields tab * The data element number in 999 or DE999 format * The data element in YY999 format, were YY is the record type (e.g.   “PH011”)  (**NOTE:** This entry is **NOT** case-sensitive.) |

##### Returns

the next highest valid Data Element number for this record

0 – no more data elements exist

##### Example

WK1= PTPHOBJ\_DENEXT(0); LOOP WHILE WK1>0; /\* Code to process each Data Element goes here \*/ WK1=PTPHOBJ\_DENEXT(WK1); ENDLOOP;

#### \_DEPIC – Return the OmniPlus DE Picture of a DE

Returns the internal Picture of the specified data element, such as “N11D00” or “X40”. See the OmniPlus Data Element overview for an explanation of OmniPlus element pictures. Refer also to the DETYPE operation.

##### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]:num | The selected data element number. |

**Returns**

The text of the Internal OmniPlus picture of the element

##### Example

TX1=PTPHOBJ\_DEPIC(170); /\* Returns: „N11D02‟ \*/

#### \_DETYPE – Return the type of a specified data element

Return the type of the specified Data Element („X‟ or „N‟)

##### Parameters

Either of the following:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats:   * The name as it appears on the Omni Station Fields tab * The data element number in 999 or DE999 format * The data element in YY999 format, were YY is the record type (e.g.   “PH011”)  (**NOTE:** This entry is **NOT** case-sensitive.) |

##### Returns

* „X‟ if the given element is a TEXT element  „N‟ if it is a NUMERIC element.
* Spaces if no such element exists

##### Example

TX001=PTPHOBJ\_DETYPE(007); \* Returns: 'X' /\* OR \*/ TX001=PTPHOBJ\_DETYPE(DENUM:170); \* Returns: 'N'

#### \_DeValDesc – Return the description of a DE Value

This operation is used to display the legal values for data elements that have multiple legal values. In Omni Station, these data elements show as a “pull down menu” with all of the legal values displayed. This operation returns the text description for the specified legal value for the specified data element. Refer also to the **DeValKey** operation.

##### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]num | The selected data element number. |
| INDEX:num | The selected subscript of the value, starting at 1 |

##### Returns

The text value of the selected value description, or „‟ if no such value.

SD101 will be set to the description of the selected value.

##### Example

\* Get the first legal value and it‟s description for PH023 TX1=PTPHOBJ\_DEVALKEY(023 INDEX:1);

#### \_DeValid – Determine if a Data Element exists

Return 1 (True) if the parameter Data Element exists

##### Parameters

Either of the following:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats:   * The name as it appears on the Omni Station Fields tab * The data element number in 999 or DE999 format * The data element in YY999 format, were YY is the record type (e.g.   “PH011”)  (NOTE: This entry is **NOT** case-sensitive.) |

**Returns**

1 (True) if the parameter DE exists, 0 (no such DE)

#### \_DeValKey – Get the specified legal value

This operation is used to display the legal values for data elements that have multiple legal values. In OmniStation, these data elements show as a “pull down menu” with all of the legal values displayed. This operation returns the text value of the specified element‟s value. Refer also to the **DeValDesc** and DeValKeyN operations.

##### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]:num | The selected data element number. |
| INDEX:num | The selected subscript of the value, starting at 1 |

##### Returns

The text value of the selected element, or „‟ if no such value.

SD101 will be set to the description of the selected value.

##### Example

\* Show all legal values and descriptions for PH023 WK1=0; TX1=PTPHOBJ\_DEVALKEY(023 INDEX:WK1); LOOP WHILE TX1 >< „‟; OCSHOW(TX1 SD101) WK1=+1; TX1=PTPHOBJ\_DEVALKEY(023 INDEX:WK1); ENDLOOP;

#### \_DeValKeyN – Get the numeric value of DeValKey

Refer to the **DeValKey** operation. Returns the numeric value of a data element‟s legal value.

##### Example

WK1=PTPHOBJ\_DEVALKEYN(023 INDEX:WK1);

#### \_DeValName – Return the name a data elements value

This operation is used to display the legal values for data elements that have multiple legal values. In Omni Station, these data elements show as a “pull down menu” with all of the legal values displayed. This operation returns the descriptive value associated with the data element‟s current value.

##### Parameters

Either of the following:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats:   * The name as it appears on the Omni Station Fields tab * The data element number in 999 or DE999 format * The data element in YY999 format, were YY is the record type (e.g.   “PH011”)  (NOTE: This entry is **NOT** case-sensitive.) |

##### Example

TX1=PTPHOBJ\_DEVALNAME(NAME:'STAT'); Returns: 'Active and Eligible'

#### \_DEXML – Format a Data Element’s Values

Return a text line describing the indicated data element

##### General Parameter

**Parameter**

**Description**

[DENUM:]num The 3 digit numeric data element number of the element

##### Formatting Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| [FMT:txt] | The selected format type.  „XML2‟ – XML, with a <Tag> value </Tag> format (default)  „XML‟ – Basic XML  „PRINT‟ – Data Element=‟Value‟ format |
| [TAG:tx] | Select the per-element XML tag.  „DE‟ – The tag will be „<de>‟  „DENUM‟ – The tag will be <de###> where ### is the De Number. |
|  | „LABEL‟ – The tag will be a generated DE name. (e.g. <PLANNUM>. |
| [ATTRIBS:TX] | Identifies which Data Element fields are to be included as XML attributes. One or more items may be specified.  „#‟ – DE Lumber (num=”005”)  „L‟ – DE Label  „N‟ – DE Name (name=”PLANNUM”)  „P‟ – DE Picture (pic=”X06”)  „V‟ – DE Value (value=“000001”)  „D‟ – DE Value Description (desc=“Retired”)  „\*‟ – All the above  The default is no attributes.  Example: ATTRIBS:‟#N‟ |
| [NODES:TX] | Identifies which Data Element fields are to be included as XML nodes.  One or more items may be specified.  „#‟ – DE number (<num>005</num>)  „L‟ – DE Label  „N‟ – DE Name (<name>PLANNUM)</name>  „P‟ – DE Picture <pic>X06)</pic>  „V‟ – DE Value (<value>000001</value>  „D‟ – DE Value Description (<desc>Retired</desc>)  „\*‟ – All the above The default is no nodes.  Example: NODES:‟#ND‟ |

##### Returns

The text values associated with the selected element

„‟ if no data element exists (**SD101** will contain the error reason)

##### Examples

|  |  |
| --- | --- |
| **OmniScript Text:** | **Returns:** |
| PTPHOBJ\_DEXML(050); | <DE050> 1970/07/01 </DE050> |
| PTPHOBJ\_DEXML(050 TAG:'LABEL'); | <BrthDate> 1970/07/01 </BrthDate> |
| PTPHOBJ\_DEXML(050 FMT:'XML2' TAG:'DE' NODES:'#V'); | <DE> <num> 050 </num><value> 1970/07/01  </value> </DE> |
| PTPHOBJ\_DEXML(050 FMT:'XML2' ATTRIBS:'#L'); | <DE num="050" label="BrthDate"> 1970/07/01  </DE> |
| PTPHOBJ\_DEXML(050 FMT:'XML2' NODES:'#LV'); | <DE> <num> 050 </num><value> 1970/07/01  </value><label> BrthDate </label> </DE> |
| PTPHOBJ\_DEXML(050 FMT:'XML' ATTRIBS:'#V' TAG:'LABEL'); | <BrthDate num="050" value="1970/07/01"/> |
| PTPHOBJ\_DEXML(050 FMT:'XML2'  ATTRIBS:'#' TAG:'LABEL'); | <BrthDate num="050"> 1970/07/01 </BrthDate> |
| PTPHOBJ\_DEXML(050 FMT:'PRINT'); | DE050=1970/07/01 |

**4.4.3 Standard Functions**

* OmniScript Utility Functions

|  |  |  |
| --- | --- | --- |
| **Function**  **Name** | **Description** | **Object Guide** |
| **BARUN** | Establish OmniPlus Processing Environment | OmniScript |
| **DDRESP** | Generate Omni Station Packet Response | OmniScript |
| **OCCLIP** | Clipboard Facility | OmniScript |
| **OCCNTL** | Parameter Control File Facility | OmniScript |
| **OCCSV** | Interface with Comma Separated Values (.CSV) formatted data | OmniScript |
| **OCDATA** | Data Container Facility | OmniScript |
| **OCDATE** | Date Utility | OmniScript |
| **OCFACT** | Financial Factor / Allocation Library | OmniScript |
| **OCFILE** | External File Read and Write Access | OmniScript |
| **OCFILV** | External File Read and Write Access (Wide Files) | OmniScript |
| **OCFIND** | Find (Lookup) Item Among Given Parameters | OmniScript |
| **OCFMT** | Format Numeric Fields into Text with Given Picture | OmniScript |
| **OCFUNC** | OmniScript Function Execution Facility | OmniScript |
| **OCLABEL** | Custom Variable Label Utility | OmniScript |
| **OCLINE** | Produce OP50 Report Line Output | OmniScript |
| **OCNUM** | Arithmetic and Numeric Library | OmniScript |
| **OCPAGE** | Produce T966 Report Output Directly from OmniScript | OmniScript |
| **OCPARMS** | Retrieve Parameters from T588/T988 transaction entry | OmniScript |
| **OCSCREEN** | Interactive Screen Facility | OmniScript |
| **OCSCRIPT** | OmniScript Execution Facility | OmniScript |
| **OCSEL** | Select Item from Parameter List | OmniScript |
| **OCSHOW** | 'Exhibit Named' Type Display of Variables | OmniScript |
| **OCSPLIT** |  | OmniScript |
| **OCSUB** | Perform Text Sub-string Operations | OmniScript |
| **OCTEST** | Self Checking Canned Test Data Support | OmniScript |
| **OCTEXT** | Text Field Utility | OmniScript |
| **OCTIME** | Perform Various Time Related Operations | OmniScript |
| **OCTX** | TX Field Array Utility Library | OmniScript |
| **OCVALUE** | Iterate Numeric Values for the Loop While | OmniScript |
| **OCWK** | WK Field Array Utility Library | OmniScript |
| **OCWORD** | Convert Numeric Values Into Word Equivalents | OmniScript |
| **OCXML** | XML (eXtended Markup Language) Parsing Facility | OmniScript |
| **UTFSYS** | File System Object Facility | OmniScript |
| **UTINDX** | External Indexed File Read and Write Access | OmniScript |

* Object-Specific Utility Functions by Object Guide

|  |  |  |
| --- | --- | --- |
| **Object Guide** | **Function Name** | **Description** |
| **Adjustments** | AD301O | Miscellaneous Adjustments Transaction Access (obsolete) |
| **Adjustments** | ADGADJ | Provide Access to Gross Adjustment Area |
| **Base** | BABKOBJ | Backup File Access Utility |
| **Base** | BATCUT | Transaction Card Access Facility |
| **Base** | BATIUT | Transaction Input Facility |
| **Compensation** | CMSUM | Sum Compensation History for a Given Date Range |
| **Distributions** | DI404O | Termination Transaction Data Access (obsolete) |
| **Distributions** | DI444O | Withdrawal Transaction Data Access (obsolete) |
| **Distributions** | DI454O | Return of Excess Transaction Data Access (obsolete) |
| **Distributions** | DILIFE | Life Expectancy Table Access |
| **Eligibility** | PLELIG | Eligibility Calculation Library |
| **Fees** | FEALLOC | Fee Allocation Utility |
| **Fees** | FEPROC | Fee Processing Functions |
| **Funds** | FNFCUT | Fund Control Record Utility |
| **Funds** | FNPFAMT | Participant Fund amounts available calculations |
| **Funds** | FNTRAN | Transaction Fund Amounts Access |
| **History** | HIBRPO | Access to Internal History Posting records |
| **Loans** | LNACINT | Calculate Accumulated Loan Interest |
| **Loans** | LNLDA | Loan Delinquency Analyzer |
| **OmniCash** | CAEXP | OmniCash Billing Functions |
| **OmniDBEN \*** | DNACTE |  |
| **OmniDBEN \*** | DNFAE |  |
| **OmniDBEN \*** | DNPROJ |  |
| **OmniDBEN \*** | DNSSA |  |
| **OmniNet** | DDPARMS | Retrieve Form Data from Web Pages |
| **OmniNet** | DDSDATA | Access DDMS Session Data Records |
| **OmniNet** | DDSESS | Access DDMS Session Records |
| **OmniNet** | DDWEB | String Text into Response Buffer |
| **OmniSecurity** | USPROC | OmniSecurity User Validation |
| **OmniTrade \*** | TXCTRL | Create Table of OmniTrade Messages |
| **Plans** | PLYEAR | Plan Year Dates Function |
| **Rate of Return** | RRITEM | Rate of Return Calculations Function |
| **Report**  **Management** | RPLOG | Issue Reject, Warning, and Information Log Messages |
| **Report**  **Management** | RPMSGS | Access Error/Reject Messages for the current transaction |
| **Service** | SVCALC | Determination of Service Time as of a Given Date |
| **Service** | SVSTAT | Employment and Participant Status Calculations |
| **Sub-accounting** | SACOBV | Contract Balance Access |
| **Textfiles** | TXTIER | Tiered Table access |
| **Object Guide** | **Function**  **Name** | **Description** |
| **Transfers** | XF381O | Inter-fund transfer transaction data access (obsolete) |
| **Utilites** | UTPROSE | OmniPlus Proration Logic Access |
| **Utilities** | UTCACHE |  |
| **Vesting** | FOVECNV | Vesting Control File Conversion Utility |
| **Vesting** | FOVEST | Market Value, Vesting, and Forfeiture Amount calculations |
| **Vesting** | PTPHRET | Participant Retirement Date Functions |
| **Withholding** | TAINFO | Calculate SSA Taxable Wage Base for the current year |

* Object Access Functions by Object Guide

|  |  |  |
| --- | --- | --- |
| **Object Guide** | **Function Name** | **Object (Record Type)** |
| **Addresses** | AAAAOBJ | Alternate Address |
| **Annuities** | ANAMOBJ | Annuity (Annuity Master) |
| **Annuities** | ANASOBJ | Annuity (Annuity History) |
| **Associated Individuals** | PTAIOBJ | Participant (Associated Individual) |
| **Base** | BAI2OBJ | Alternate Inquiry |
| **Base Text** | BABTOBJ | Base Text |
| **Base Text** | BAUDOBJ | Base Text (User Defined Base Text) |
| **Base Text** | UTLOOBJ | Utility (Log Base Text) |
| **Cash** | CACAOBJ | Cash (Cash Account) |
| **Cash** | CACHOBJ | Cash (Cash History) |
| **Checks** | CKCDOBJ | Check (Check Detail) |
| **Checks** | CKCKOBJ | Check (Check Header) |
| **Compensation** | CMCMOBJ | Compensation |
| **Compliance** | PLCOOBJ | Plan (Compliance) |
| **Compliance** | TALMOBJ | Tax and Compliance Limits |
| **Disbursements** | DBDBOBJ | Disbursements |
| **Distributions** | DIAPOBJ | Distributions (Alternate Payee) |
| **Distributions** | DIMDOBJ | Distributions (Minimum Distributions) |
| **Distributions** | DISGOBJ | Distributions (Source Group) |
| **DivSub** | DSDSOBJ | Division/Subsidiary |
| **Forecasting** | FRFAOBJ | Forecasting (Forecasting Projection Annuity) |
| **Forecasting** | FRFBOBJ | Forecasting (Forecasting Beneficiary) |
| **Forecasting** | FRFPOBJ | Forecasting (Forecasting Projection) |
| **Forecasting** | FRFROBJ | Forecasting (Forecasting Header) |
| **Foreign Exchange Rate** | HIFXOBJ | Foreign Exchange History |
| **Foreign Exchange Rate** | XRXROBJ | Exchange Rate |
| **Funds** | FNFCOBJ | Fund (Fund Activity and Fund Control) |
| **Funds** | FNTRAN | Transaction Fund Amounts |
| **Funds** | PTPFOBJ | Fund (Participant Fund) |
| **History** | BAFMOBJ | File Maintenance |

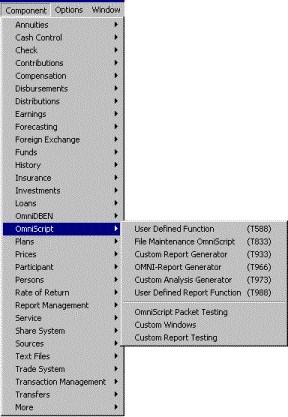
|  |  |  |
| --- | --- | --- |
| **Object Guide** | **Function**  **Name** | **Object (Record Type)** |
| **History** | HIBROBJ | History (Base Record) |
| **Installments** | ININOBJ | Installments |
| **Insurance** | PIEVOBJ | Insurance (Participant Event) |
| **Insurance** | PIPDOBJ | Insurance (Participant Detail) |
| **Investments** | IVIAOBJ | Investment (Investment Action) |
| **Investments** | IVICOBJ | Investment |
| **Loans** | LNLFOBJ | Loans (Loan Fund) |
| **Loans** | LNLHOBJ | Loans (Loan Header) |
| **Loans** | LNLPOBJ | Loans (Loan Payment) |
| **Messages** | MSMSOBJ | Messages |
| **Notes** | NTNDOBJ | Notes (Note Detail) |
| **Notes** | NTNHOBJ | Notes (Note Header) |
| **Notes** | NTOBJ | Notes (Header and Detail) |
| **OmniDBEN \*** | CMSLOBJ | OmniDBEN Compensation (Salary) |
| **OmniDBEN \*** | DNBEOBJ | OmniDBEN (Beneficiary) |
| **OmniDBEN \*** | DNCROBJ | OmniDBEN (Computation Results) |
| **OmniDBEN \*** | DNIROBJ | OmniDBEN (Interest Rate) |
| **OmniDBEN \*** | DNPBOBJ | OmniDBEN (Prior Benefit) |
| **OmniDBEN \*** | DNPNOBJ | OmniDBEN (Participant) |
| **OmniDBEN \*** | SVSTOBJ | Service (OmniDBEN Service History) |
| **OmniExport \*** | HISHOBJ | History (Sequential History File) |
| **OmniTrade \*** | SSSAOBJ | OmniTrade (Share Account) |
| **OmniTrade \*** | SSSHOBJ | OmniTrade (Share History) |
| **OmniTrade \*** | SSSROBJ | OmniTrade (Share Request) |
| **OmniTrade \*** | TRTHOBJ | OmniTrade (Trade History) |
| **OmniTrade \*** | TRTOOBJ | OmniTrade (Trade Order) |
| **Participants** | PTAFOBJ | Participant (Annual Financial) |
| **Participants** | PTPHOBJ | Participant (Participant Header) |
| **Persons** | PEPEOBJ | Person |
| **Plans** | PLPLOBJ | Plan |
| **Plans** | PMPMOBJ | Product Master |
| **Prices** | PRDROBJ | Price (Daily Rate) |
| **Prices** | PRPROBJ | Price (Price Header) |
| **Rate of Return** | RRRROBJ | Rate of Return |
| **Report Management** RPRHOBJ | | Reports (UCOM Report Header) |
| **Report Management** RPRPOBJ | | Reports (UCOM Report Page) |
| **Service** SVSVOBJ | | Service |
| **Sources** PTPSOBJ | | Source (Participant Source) |
| **Sources** SOSCOBJ | | Source (Source Control) |
| **Sub accounting** SAEPOBJ | | Sub-Accounting (GIC Processing) |
| **Sub accounting** SAGIOBJ | | Sub-Accounting (GIC Base Text) |
| **Sub accounting** SAILOBJ  **SubPlan** SPSPOBJ | | SubPlan |
| **Textfiles** TXOBJ | | Textfile (Headers and Records) |
| **Textfiles** TXTFOBJ | | Textfile (Textfile Header) |
| **Textfiles** TXTXOBJ | | Textfile (Textfile Record) |
| **Transaction**  **Management** VTTCOBJ | | Transactions (Common Transaction Data) |
| **Transaction** VTTDOBJ | | Transactions (Transaction Detail) |
| **Object Guide** | **Function**  **Name** | **Object (Record Type)** |
| **Management**  **Transaction**  **Management** | VTTHOBJ | Transactions (Transaction Header) |
| **Transaction Management** | VTTSOBJ | Transactions (Transaction Submit) |
| **Voice Response** | PTVROBJ | Voice Response |

## 4.5 Online Access

### 4.5.1 OmniScript Menu

The OmniPlus OmniScript component option accesses six transaction screens and two custom windows for developing and testing OmniScript text files. The OmniScript functions are accessed via the COMPONENT menu option, and the OmniScript component, which expands to the transaction and screen options shown in the graphic below. The OmniScript component includes transactions for universal update (T588), participant file maintenance (T833), custom report generation (T933), full-page custom reporting (T966), and universal report (T988). The transactions are discussed under [OmniScript Transactions.](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/transactions/index.htm)

**Note: OmniScript text files used to store OmniScript text are accessed via the Text Files component. This component provides a list of all text files on a plan by selecting the Text Files/Text Files Browse component and option.**



### 4.5.2 Custom Windows (OCCSCR)

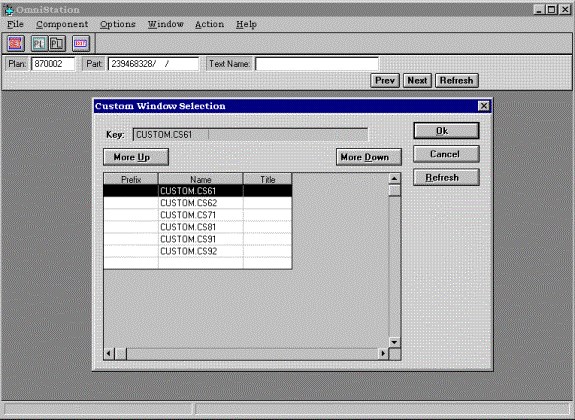
The OmniScript Packet Testing, Custom Windows, and Custom Report Testing options on the OMNISCRIPT component item allow the user to select and test report definition files with immediate system response that shows whether the text file is operating correctly. If the text file contains an error, the Custom Report Testing screen will show an error message.

The first menu option, Custom Windows (OCCSCR), provides a list of all of the custom windows available for testing by the user (screen below). The custom textfiles are named with a **'CUSTOM.CSnn'** format, where 'nn' is a user-defined sequence number. Select any custom screen by double-clicking on the text file name. The custom file is opened and "tested" using the Custom Report Testing (OCCS99) function. A screen will appear showing either the correct results of the text file, or an error message indicating problems with the custom text file.

* Key Fields

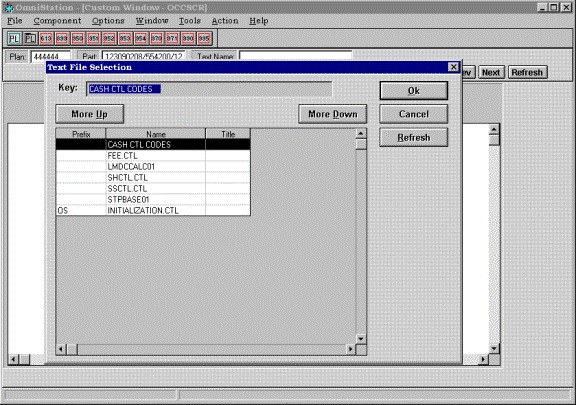
|  |  |  |
| --- | --- | --- |
| ***Field*** | ***Field Name*** | ***Object*** |
| ***Plan*** | [Plan Identifier](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/DATAELEMENTS/DATAELEMENTS.htm%23PL011) | PLAN |
| ***Part*** | [Participant Identification](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/DataElements/DataElements.htm%23PH007) | PART |
| ***Text Name*** | Text File Name |  |

* Custom Window (OCCSCR) - Custom Textfile Selection Option



* Field Descriptions
* ***Key***  *-* This field shows the starting text file name for this browse list.
* ***Prefix*** *-* This field shows any prefix for the text file name.
* ***Name*** - This field shows the name of the text file.
* ***Title***  - This field shows any available description of the text file. In addition to accessing the custom report definitions, if the user desires to see other report definition text files a simple "right-click" in the 'TEXT NAME:' field on the Custom Screen provides a 'LIST' option to see all text files available on the plan (screen below). The fields are identical to the Custom Window Selection screen. Any textfile may now be selected. Once selected, the textfile is automatically "tested" using the Custom Report Testing screen (OCCS99).
* ***Plan*** - This field shows the plan the custom text file resides in***.***
* ***Date 1, Date 2, & Initial Value # 1, 3 &6*** - The custom screen facility allows the use of up to five (5) user-defined fields to supply values for the OmniScript and textfile sections for screen processing. These may be labeled with the default values or with user-defined captions. Refer to [Custom Screens](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/CustomScreens/CustomScreens.htm%23custom_screens) for information on how to code custom screen text files.

 Custom Window (OCCSCR) - All Text Files Selection Option

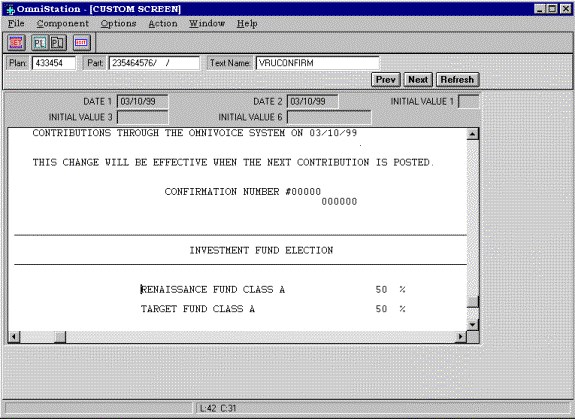


### 4.5.3 Custom Report Testing (OCCS99)

The Custom Report Testing (OCCS99) screen is used to test a report definition textfile on-line. The user may enter directly any text file name in the 'TEXT NAME:' field or right-click for a list of text files on the plan and select one. Once the file is selected, clicking on the 'REFRESH' button will execute the text file. Once the text file is executed, The Custom Report Testing screen will either display the report text for that participant or will display error messages describing a problem with the text file.

* Key Fields

|  |  |  |
| --- | --- | --- |
| ***Field*** | ***Field Name*** | ***Object*** |
| ***Plan*** | [Plan Identifier](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/DATAELEMENTS/DATAELEMENTS.htm%23PL011) | PLAN |
| ***Part*** | [Participant Identification](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/DataElements/DataElements.htm%23PH007) | PART |
| ***Text Name*** | Text File Name |  |



* Field Descriptions
* ***Plan*** - This field identifies the plan on which the custom screen definition exists.
* ***Part*** - This field identifies the participant whose account was used for the custom screen definition test.
* ***Text Name*** - This field identifies the report definition text file that was used for the custom screen definition test.
* ***Date 1, Date 2, & Initial Value #*** - The custom screen facility allows the use of up to five

(5) user-defined fields to supply values for the OmniScript and textfile sections for screen processing. These may be labeled with the default values or with user-defined captions. Refer to [Custom Screens](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/CustomScreens/CustomScreens.htm%23custom_screens) for information on how to code custom screen text files.

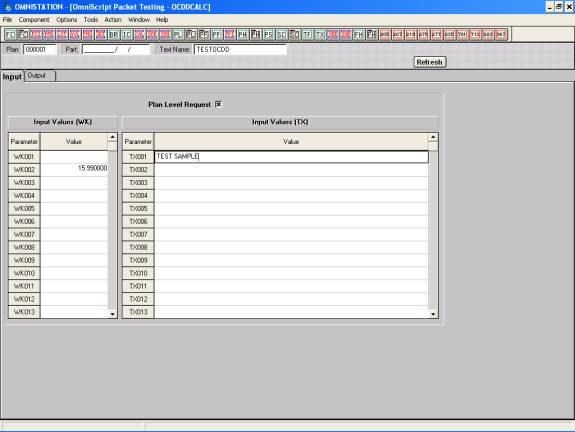
### 4.5.4 OmniScript Packet Testing (OCDDCALC)

The OmniScript Packet Testing (OCDDCALC) screen is used for on-line testing of work and text field returns. The user may enter directly any textfile name in the 'TEXT NAME:' field, or rightclick for a list of textfiles on the plan and select one. Once the file is selected, clicking on the 'REFRESH' button will execute the text file.

* Key Fields

|  |  |  |
| --- | --- | --- |
| ***Field*** | ***Field Name*** | ***Object*** |
| ***Plan*** | [Plan Identifier](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/DATAELEMENTS/DATAELEMENTS.htm%23PL011) | PLAN |
| ***Part*** | [Participant Identification](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/DataElements/DataElements.htm%23PH007) | PART |
| ***Text Name*** | Text File Name |  |

Sample OmniScript Packet Testing input:



* Field Descriptions
* ***Plan*** - This field identifies the plan on which the OmniScript Packet definition exists.
* ***Part*** - This field identifies the participant whose account was used for the OmniScript Packet definition test.
* ***Text Name*** - This field identifies the OmniScript textfile that was used for the OmniScript Packet definition test.
* ***Returned Values (WK) and Returned Values (TX)*** - These two sections hold the values returned by the work and text fields. The numeric work fields are reported on the left side of the screen, and the alphanumeric text fields are reported on the right side of the screen.
* ***Parameter*** - These are the “valued” work and text fields to be reported. The work and text fields may be valued from either the textfile being used or from the “Input” screen of OCDDCALC. Values from the textfile may consist of literal values or variable values pulled from data elements or calculated using OmniScript.
* ***Values*** - These fields hold the amounts or text for each work or text field found in the Parameter.
* ***Plan Level Request*** - This flag allows the use of OCDDCALC on a plan level rather then on a participant level. This allows the use of the OCDDCALC packet on a plan with no Participant Header records.

## 4.6 Report Writer

Report definition text files are created in OmniPlus using Report Writer - a text facility that allows the user to create and maintain text files to generate custom output. The Custom Full Page Report Generator (T966) transaction is used to execute the report definition text files. This section presents information on how to create report definitions that can be used for custom printing. Included are discussions about custom screens, transactions, and data files.

Report definition text files **must** contain three sections:

* **PARM SECTION**. - contains the parameters and values controlling report or data generation.
* **TEXT SECTION**. - contains the actual text layout of the report including headings, titles, columns, rows, etc.
* **CALC SECTION.** - contains the OmniScript code that extracts and manipulates data from the OmniPlus database.

The section commands (PARM SECTION., etc.) must begin in column 1 in the text file. Each must be entered in all capital letters and must be followed by a period. Even if one of the sections is not needed to define any element of the report, **all three section commands must appear in the correct order** in the report definition text file, otherwise a reject message will appear due to the missing section. The following sections describe the three report definition elements.

### 4.6.1 Parm Section

The "PARM SECTION." command is always the first entered in a report definition text file. The parameters use to establish the output mediums are entered on the lines immediately following the PARM SECTION command. Only one output parameter may be loaded on a single line. The parameters available within this command include:

* DATA
* FORM
* FUNDS
* OR\_START
* PRINTFIRST, PRINTSECOND, PRINTLAST
* PRINT WIDE
* TOT (totals)

### 4.6.2 Text Section

The "TEXT SECTION." is always the second section entered. This command contains control characters and parameters that define the actual layout of the report, including report headings, messages, print identifiers, print formats, and conditional print options. Parameters, symbols, and commands include:

* Variables and Fields

Report definition text files can access system-generated, calculated, mnemonic, or OmniPlus database records. Within the TEXT SECTION of a text file, the user may access the following types of records:

* Function data elements (FD)
* Fund Control data elements (FC)
* Investment Control data elements (IC)
* Participant Fund data elements (PF & PA)
* Participant Header data elements (PH)
* Plan codes (PL)
* Source Control data elements (SC)
* System-defined data elements (SD)
* Work fields (WK) and Work funds (WF)
* Text literal fields (TX)
* System time and date elements (CTIME, CDATE)
* Social security numbers (PART)
* Canadian format SSN (PARTC)
* Report dates (TDATEA)
* Alternate report dates (TDATEB)

### 4.6.3 CALC SECTION

The "CALC SECTION." is always the last section entered. This section contains the OmniPlus OmniScript code used to extract and compute values for the report. After the initial CALC command line, the report definition file contains typical OmniScript code used to compute the values for the data fields defined on the report in the TEXT section.

# 5.0 Omni Plus Technical Overview

## 5.1 System Architecture

 Overview :

The OmniPlus System is a COBOL based financial application. The system is modular, with structured COBOL programs, using Indexed files. In a large system such as OmniPlus, in which many programmer‟s efforts are accumulated over a long period of time, concrete standards and practices must be established to insure the system is easy to extend and maintain.

**OMNI**

**STATION**

**OMNI PLUS**

**EXTERNAL**

**SOURCE**

**UNIFIED**

**BATCH**

**PLAN**

**PART**

**FUND**

**AHST**

**VTRAN**

**UCOM**

**FORMAT**

**MATCH ?**

**TRANSACTION**

**TRANSLATOR**

**NO**

**YES**

**UPDATE**

**EXTRACT**

**UPDATE**

**UPDATE**

**UPDATE**

**UPDATE**

**REPORTS**

**DATA**

**EXTRACT**

**REPORT**

**SEQ. HIST**

**EXTRACT**

**BRAR**

**MUPD**

**EXTK**

The technical features of Omni can be stated as follows:

* **Platforms** - The OmniPlus system is implemented on a variety of hardware/software environments. The Coding Standards are designed to help insure successful compiling on each platform.

* **System Architecture** - The software is very modular, with many shared modules to perform common functions.

* **Client/Server Environment -** The OmniStation product is a workstation/client GUI implementation. The Host/Server code executes on the host system.

* **Driver Modules -** A basic architectural feature is the use of driver modules to help enforce standardized module templates/functions.

* **SYSINIT** - Runtime Environment Control - this file contains .INI type run Setup information, in a VARIABLE=VALUE format. OmniPlus (BAENV) uses these values to set variables which can control system operation.

* **SYSDATA** - Global Common Parameter Area - use of this area allows components (modules) of the system to communicate with each other.

* **SYSOUT/DMSG/LOG** - Processing programs call the RPDMSG facility to issue DISPLAY type messages.

* **Run Date vs. ASOF Date** - Events in OmniPlus occur on both a system Run Date/Time (via COBOL ACCEPT verb), and a virtual processing date (ASOF or TRADE date) logical to the user.

* **User Defined Fields (UDF’s)** - Many user/sites want to keep additional data with OmniPlus data. Such data can be kept on OmniPlus records, but in a place/format which does not harm OmniPlus processing.

* **Base Text (BT) Records -** AHBT records are 200 byte data records, with a generic key. BT records can be user defined, and used for system extensions, or to store related data.

 File Back Ups :

An integral part of any information system is to establish and implement backup procedures. OmniPlus master file backups may be of 2 types:

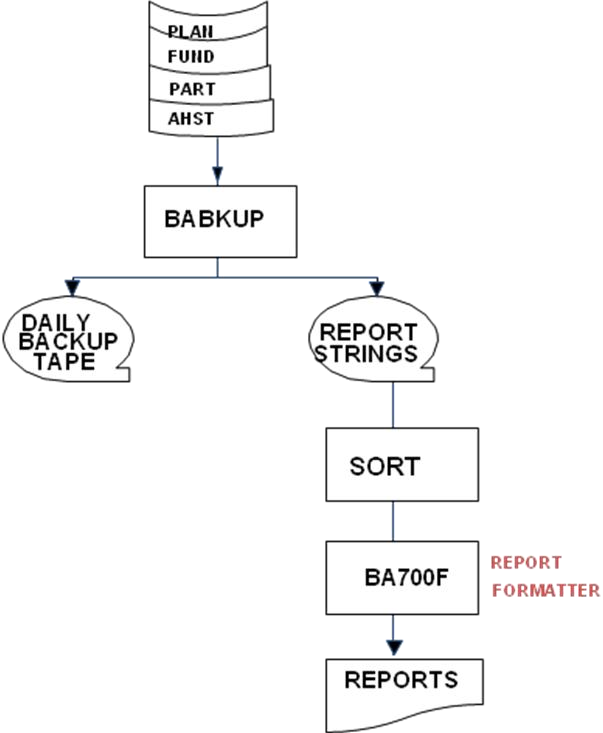
* **System Utility Backups** - SYSTEM utility backups (e.g. PKZIP etc.)
* **Master File Integrated Backup** - The OmniPlus integrated Backup/Restore jobs

1. **System Utility Backups –** PKZIP, etc. are often much quicker than OmniPlus backups, but do not provide direct capability to restore individual plans, or logs that the OmniPlus integrated backups provide.

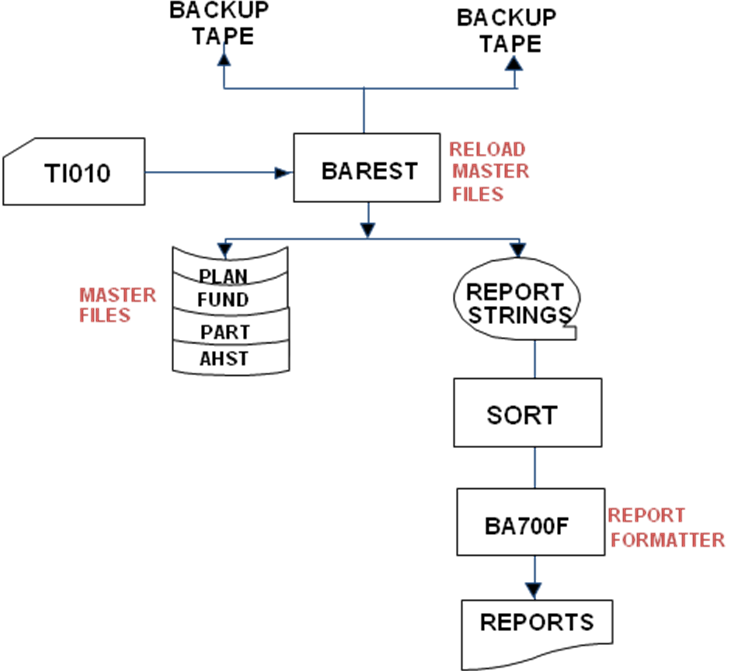
1. **Master File Integrated Backup -** The OmniPlus integrated Backup/Restore jobs backup the PLAN/FUND/PART/AHST master files to a sequential dataset. The RESTORE process has transaction options to select individual plans from a BACKUP dataset, adding or replacing data as necessary.

1. **Master File Standard File Backup -** Taken by default. Each record on this sequential variable record length file has a 1 byte prefix identifying the file (PLAN/FUND/PART/AHST) from which the record came.
2. **Master File Extended File Backup** - sequential variable record length files for which each record has an extended, fixed format key area with the standard OmniPlus fields (PLAN/PART-NUM, FUND-ID, ASOF-DATE, etc.) in standard positions. This enables the files to be sorted and processed by various utilities in a selected order.

* + - Daily master File Back up



* + - Daily master File Restore



* + - Transaction Processing / VTRAN :

Transactions flow into the OmniPlus from numerous sources at different times of the day. The OmniPlus Transaction Management System (VTRAN) is used to perform or assist in performing the following functions:

* + Input transactions online or batch load trans. to the Transaction System from tape, diskette, or other means
  + Browse transactions in a data entry folder
  + Process transactions in "edit" or "update" modes
  + Change the status of a Folder or transaction. For example, individual transactions or entire folders may be placed on hold or reactivated
  + Replicate a transaction for all plans on the system or subsets of plans
  + Provide a Business Calendar to ensure the Transaction Scheduler (a.k.a. Periodic Propagation) does not create a transaction(s) with trade dates that fall on a weekend or a holiday.
  + View the input transaction and related history after updating records
  + View transactions with similar status codes within the plan or across plans
  + View folders system wide by status
  + Move or copy transactions between Folders
  + Allow for future dating of transactions
  + Report on different transactions and transaction statuses
  + Submit transactions for processing (automatically or manually)
  + Access the Transaction Management files via OmniScript
  + Purge transactions on predefined parameters
  + Backup and restore transaction folders by plan or globally
  + Provide transaction audit trail
  + Provide multiple levels of transaction approval
  + Allow online balancing of contribution transactions
  + Define currency of input transaction if Multi-Currency functionality is being used
  + Interface with other systems through usage codes
  + Identify transactions that produced external report string files
  + Complete one transaction and have the system copy and split the transaction to participant occurrences selected (i.e. Expand Sub Plan and PARTCROSSPLAN).

**Pre**

**-**

**Update**

**Backups**

**Post**

**-**

**Update**

**Backups**

Transactions

from

OMNI Station

Transaction Translator

Transactions from External Interfaces

Transactions in OMNI PLUS format

VTUTIL

VTRAN

Outbound Interfaces

**UNIFIED**

[

Updates in the OMNIPLUS Master File

]

VTRAN provides extensive functionality to efficiently manage the daily flow of transactions.

* + The OmniPlus System reads the VTRAN Files and extracts transactions based on current status. The system maintains folder and transaction statuses that allow the user to control what transactions are processed. Transactions may be future dated, placed on hold by the user, placed on hold by the system automatically (such as in cases of missing unit value), or they may be in "active" status and automatically extracted for the Unified Process.

* + The status of folders and transactions may be viewed by plan or across all plans. This is extremely important since most users operate in a daily mode and cannot process financial transactions until daily prices are available at the end of a day. The VTRAN System allows users to view folders that are to process overnight and then view any rejected activity across the system or by plan the next day.

* + The VTRAN Detail record key field information is stored on the report string output files (i.e., RS03, RS06) produced by the OmniPlus system. This allows the user to identify the transaction that produced the report string initially.

* + The Edit Programs accumulate and edit incoming 80 byte transactions, converting accepted ones to variable length, internal format.

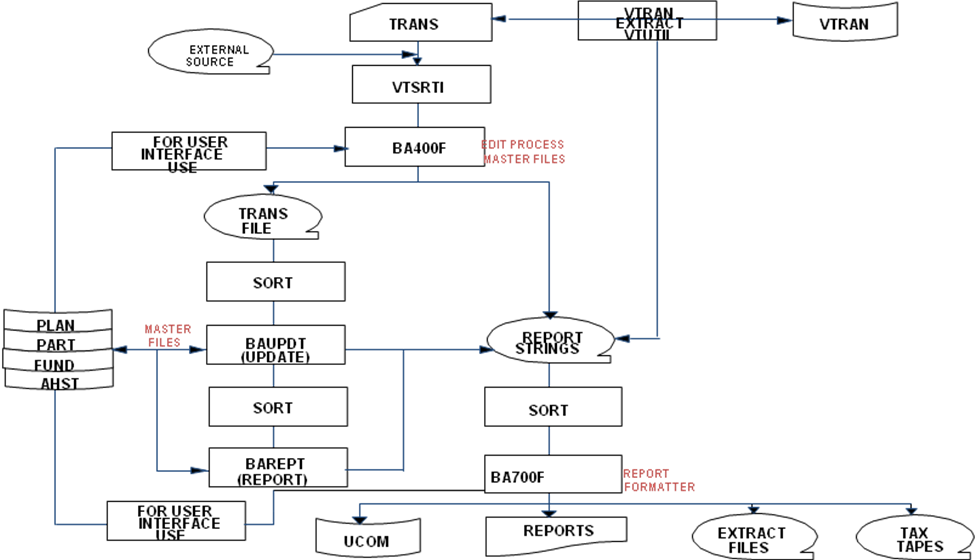
* + The Process Programs input the internal format transactions and process against the master files, performing updates and writing report strings.

* + The Report Programs input report strings produced by the previous processes and format printed output.

**Transaction number :**

* + 0\*\* transactions - accessory
  + 1\*\* transactions - contributions
  + 2\*\* transactions - earnings, stock/unit purchases/sales
  + 3\*\* transactions - miscellaneous/transfers
  + 4\*\* transactions - disbursements
  + 5\*\* transactions - cash, trade & share subsystems
  + 6\*\* transactions - plan file maintenance
  + 7\*\* transactions - fund control maintenance
  + 8\*\* transactions - participant file & AHST file maint.
  + 9\*\* transactions - report requests

**Transaction Processing Overview :**



 Reports :

Rather than generating reports directly, OmniPlus has a common Report String file. Programs generating reports write their data to the RS file. The Report String file is sorted, then input into the Report Formatter step which formats each report type.

Report strings are identified by a Report Code.

* + 3 digit codes starting with „9‟ are reserved for requested reports
  + 2 digit codes (e.g. 02) are used for activity generated reports. (a side effect of transaction activity)

Report Strings exist to:

* + Relieve the transaction processors from the burden of formatting reports
  + Allow common reports combining data from several different processes
  + Allow report format to be changed independent of the generating process
  + Allow external examination of internal OmniPlus processing (custom reports, interfaces, etc.)
  + Note: When writing any report string, be sure to move length of RSxxx-DATA to RSxxxLENG (Where xxx is the name of the report string) before calling RPRSTO.

**Common Error/Warning/Informational Logs:**

* + The RS02 report strings are used for Common Log reporting. Various programs call RPDMSG routines to output RS02 report strings. Code exists to format the RS02 report strings, formatting supplied amounts and TEXT values.
  + VTRAN code exists to append RS02 errors generated to the originating VTD transaction record for user online inspection.
  + The RS03 report strings are used to report File Maintenance changes to database records/fields. Included Activity: Record Adds/Deletes, and Data Element Changes.
  + Various Transaction and Calculator options exist to control/suppress such logs.

## 5.2 IO Modules

OmniPlus I/O modules perform all I/O requests against the files. Code may be added to perform special functions for each file access, or to perform utility functions. The I/O modules are all named OOSSIO where „OO‟ is the object and „SS‟ is the sub-object (e.g. PTPHIO).

Parameters passed include:

* I/O area containing returned errors, program name, etc.
* The record being passed or returned
* A function code to be performed (open, read, etc.)

It is (one and only one) Input/ Output module to perform access to the file. The I/O modules:

* Allows programs to assume OPENS and CLOSES are done at the run level and shared access to all files

* Provides a powerful single point hook at the point each record is read/written (Data

Compression is implemented at the IO module level )

* Provide low-level access to the basic COBOL verbs (OPEN, CLOSE, READ, REWRITE, etc.); higher level operations (e.g. READ-GE) can be implemented

Each OmniPlus object process business info stored in records. It hides the physical file implementation from record handling callers. Records are written/read from files. There is a separate OMNI I/O module for each record type. It can be many record types in a one file.

 Record Functions :

* include I/O operations (READ/WRITE/DELETE)
* allow Data Element dump
* allow Calculator/Report writer access to all Data Elements

Each OmniPlus record has individual data elements identified by it‟s 2 byte record prefix and a 3 byte Data Element Number (e.g. PH010, FC110, etc.).

Each Data Element Number in a record (e.g. PL011) is assigned a variable name (e.g. PL-PLANNUM). The DE number is used to reference the data via Calculator, DE screens, Dumps, and file maintenance.

Get/Store modules load and unload variables in the current record via the DE number. OmniPlus has Data Element GET/STORE modules for each record. These Get/Store modules are the only part of the system with a detailed knowledge of the elements a record has, their numbers/names, etc.

The DE modules are the single source of DE information. Generic routines/programs should call the DE modules to fetch or store elements generically. New elements can be added to any record type by modifying the DE modules. Any program wanting to access elements by Number or Name must call the DE module to do it (e.g. Record Dump, Data Element Screens, etc.).Get/Store modules load and unload variables in the current record via the DE number.

 Classification of I/O modules :

### Indexed files I/O modules

* PLPLIO Plan File
* FNIO FUND File
* PTPTIO PART File
* BAAHIO AHST File
* VTIO class VTRAN i/o interface
* BAPWIO Batch Record Access for Password record
* RPUCIO UCOM File I/O Processor

### Sequential Files I/O modules

* BABKPI Input from Backup file (BKUPIN)
* BABKPO Output to Backup file (BKUPOUT)
* BAEBKI Input from Extended Backup records (BKUPIN)
* BAEBKO Output to Extended Backup records (BKUPOUT)
* BAEXIO Output alphanumeric dumps of OmniPlus master files to external file
* (DPREXPO)
* BAPWIOO External Record Access for Password Backup File
* BATCIN Input TRS Records from External
* BATCOUT Output TRS Records to External
* RPFRMIO Output print to special forms
* RPPRTIO Output to proper Print DD
* RPRSTI Read in Report Strings from External
* RPRSTO Output Report Strings to External
* UT80IN Input from two External 80 character record length files
* UT80OUT1 Output 80 character record to additional file (UT802)
* UTI133IO Input from 133 character record length file
* UTI200IO Input from 200 character record length file (used on TEXT files)
* UTI80IO Input from External 80 character record length file
* UTO200IO Output 200 character record length to External file  UTO80IO Output 80 character record length to External file

### Record Level I/O Modules

* HIBRIO BR (Activity History) Record Access
* BABTIO BT (Base Text) Record Access
* CACAIO CA (Cash Account) Record Access
* CACHIO CH (Cash History) Record Access
* DBDBIO DB (Disbursement) Record Access
* PRDRIO DR (Daily Rate) Record Access
* LODSIO DS (Division Subsid/Loc) Record Access
* LOSUBS Subsidiary Record Access
* SAEPIO Object Interface to EP records
* SAEPIO2 EP (Extended PDF‟s) Record Access
* IVICIO IC (Investment Control) Record Access
* PTI2IO I2 (Alternate Key) Record Access
* LNLNIO LH (Header) and LF(Fund) Record Access
* LNLPIO LP (Loan Payment) Record Access
* SSIOSA SA (Share Account) Record Access
* SSIOSH SH (Share History) Record Access
* SSIOSR SR (Share Request) Record Access
* SOSCIO SC (Source Control) Record Access
* LOSPIO SP (Sub-Plan) Record Access
* TXTXIO TX (Text) Record Access

### High Level I/O Modules

* VTEASY High level access to VTRAN records
* VTPROC Object interface to VTRAN records
* VTTRAN Low Level External Record Access
* DB06RSIO Input RS06 (Payment Interface) from External RS

## 5.3 FIS programming Standard

Naming conventions and rules of function/coding simplify system implementation, and enforce standards and predictability across the system.

### Batch Program Naming convention

**OOSSVVVV or OOTTTX**

Where:

* OO 2 character Object (e.g. PT, LN, PR, etc.)
* SS 2 - Sub-Object (e.g. PH, PF, etc.)
* VVVV 4 - Verb/Function (e.g. PROC, UTIL, etc.)
* TTT 3 – Transaction number (e.g. 114, 966, etc.)
* X 1 – Function (e.g. E, T)

### Copybook Naming Convention

* PRMxxxxx - Control Parameters to Called Modules
* TIxxx - Transaction Input (80 byte records)
* TCxxx - Transaction Converted “Internal” (var.length)
* RSxxx - Report Strings
* USExxxxx - Call Using Copybooks (procedure division)
* LITxxxxx - Literals
* SYSxxxxx - System Processing Literals
* UXxxxxxx - User Exits
* WDRLxxxx - Withdrawal Process Data Structures
* XFERxxxx - Fund Transfer Data Structures
* OOxSS - Object and Sub-Object combination

## 5.4 User Exits

OmniScript can be used to build special processing rules on a plan by plan basis or Global level (shared by all plans or groups of plans). The OmniScript section in each User Guide describes the functional area of the system where OmniScript may be used to build special processing rules, how the rules are setup, and the predefined field values within each functional processing area. The system has three methods of invoking OmniScript:

* Some transactions support OmniScript text within the coding structure of the transaction.
* OmniScript Text Files with special names are established that are automatically invoked if the applicable transaction is present. These files are keyed with 'VLDTnnn' where 'nnn' is the transaction number (i.e. VLDT381 for transfers).
* The last method of building processing rules is via special application area Text Files that contain predefined names and a two digit numeric number i.e. WDRLMAX01. The number of the textfile is input on the applicable transaction.

### 5.4.1 Storage

OmniScript listed with a numeric Suffix Value allow storage of multiple routines. The system defined prefix and a user-defined suffix comprise the text file names. (i.e. CNTBBASE01, CNTBBASE02, CNTBBASE03, etc.)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Transaction Number** | **Description** | **OmniScript Allowed on**  **Transaction?** | **Stored Text File Name** | **OmniScript**  **Suffix Value** | **Object** |
| **All** | **Base Event Processing** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/AdditionalGuides/BaseEventProcessing/index.htm) | **N/A** | **Base Event Processing** |
| **On-Line** | **OmniScript Packet** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/OnlineAccess/OnlineAccess.htm%23OCDDCALC) | **N/A** | **OmniScript** |
| **On-Line Web** | **OmniScript Web** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/AdditionalGuides/OmniNet/Index.htm) | **N/A** | **OmniNet** |
| **T114/ T381/**  **T384/ T395/ T404/ T444** | **CT Fee Control File** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/OmniScript/OmniScriptProcessingExits.htm%23CTFEE) | **N/A** | **Fees** |
| **T1XX/ T301/ T995** | **Plan Eligibility Testing** | **N/A** | [**PLELIG**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Eligibility/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Eligibility** |
| **T002** | **Periodic Propagation** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TransactionManagement/OmniScript/OmniScriptProcessingExits.htm%23PPROP) | **N/A** | **Transaction Managemen**  **t** |
| **T021** | **Report Sorting for Contract Balances** | **N/A** | [**GICSORT**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Subaccounting/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Subaccounti ng** |
| **T021** | **Contract Merge** | **N/A** | **SUBAMERG** | **01-99** | **Subaccounti** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Transaction Number** | **Description** | **OmniScript Allowed on**  **Transaction?** | **Stored Text File Name** | **OmniScript**  **Suffix Value** | **Object** |
|  |  |  |  |  | **ng** |
| **T029** | **Price Adjustment for ADHR Fields** | **N/A** | [**PRCECALC**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Prices/OmniScript/OmniScriptProcessingExits.htm) | **01-99** | **Price** |
| **T114** | **Contribution Validation** | **N/A** | [**CNTBVLDT**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/OmniScript/OmniScriptProcessingExits/CNTBVLDT.htm) | **N/A** | **Contribution s** |
| **T154** | **ER**  **Contribution/Forfeiture Allocation** | **N/A** | [**CNTBBASE**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/OmniScript/OmniScriptProcessingExits/CNTBBASE.htm) | **01-99** | **Contribution s** |
| **T180/ T181/ T875** | **Compensation Validation** | **N/A** | [**VLDT.CM**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compensation/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Compensati on** |
| **T205** | **Cash Earnings** | **Yes** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Earnings/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Earnings** |
| **T245** | **Fee Allocation** | **Yes** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/OmniScript/OmniScriptProcessingExits.htm%23T245) | **N/A** | **Fee** |
| **T246** | **Daily Fees** | **N/A** | [**DAILY.FEES**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/OmniScript/OmniScriptProcessingExits.htm%23DAILYFEES) | **01-99** | **Fees** |
| **T247** | **Fee Allocation** | **Yes** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/OmniScript/OmniScriptProcessingExits.htm%23T245) | **N/A** | **Fee** |
| **T275** | **Stock/Unit Purchase** | **Yes** | [**STPBASE**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ParticipantShares/OmniScript/OmniScriptProcessingExits.htm) | **01-99** | **Participant Shares** |
| **T301** | **Miscellaneous**  **Receipts/**  **Disbursements** | **Yes** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Adjustments/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Adjustments** |
| **T381** | **Transfer Fee** | **N/A** | [**XFERFEE**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/OmniScript/OmniScriptProcessingExits/TransferFee-XFERFEE.html) | **N/A-** | **Transfers** |
| **T381** | **Transfer Maximum** | **N/A** | [**XFERMAX**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/OmniScript/OmniScriptProcessingExits/TransferMaximum-XFERMAX.html) | **01-99** | **Transfers** |
| **T381** | **Transfer Validation** | **N/A** | [**VLDT381**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/OmniScript/OmniScriptProcessingExits/Inter-FundTransferValidation-VLDT381.htm) | **N/A** | **Transfers** |
| **T384** | **Participant Loan Issuance** | **N/A** | [**LOANMAX**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/OmniScript/OmniScriptProcessingExits.htm) | **01-99** | **Loan** |
| **T388** | **Loan Default Validation** | **N/A** | [**VLDT.LOAN.DEFAULT**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/OmniScript/OmniScriptProcessingExits.htm%23VLDTDEFAULT) | **N/A** | **Loan** |
| **T394** | **Transfer Validation OmniScript** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/OmniScript/OmniScriptProcessingExits/IncomingPlanTransferValidationScript.html) | **N/A** | **Transfers** |
| **T395** | **Outgoing Transfer Validation** | **N/A** | [**VLDT395**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/OmniScript/OmniScriptProcessingExits/OutgoingPlanTransferValidationScript.htm) | **N/A** | **Transfers** |
| **T395** | **Transfer Max OmniScript** | **N/A** | [**XF\_PL\_MAX**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/OmniScript/OmniScriptProcessingExits/OutgoingPlanTransferMaximumScript.html) | **N/A** | **Transfers** |
| **T381/ T384/**  **T395/ T404/ T444** | **Adjusted Gross Payment** | **N/A** | [**ADJGROSSPMT**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Adjustments/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Adjustments** |
| **T404/ T444** | **Withdrawal Maximum** | **N/A** | [**WDRLMAX**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/OmniScript/OmniScriptProcessingExits/WithdrawalMaximum(WDRLMAXnn)/WithdrawalMaximum(WDRLMAXnn).htm) | **01-99** | **Distribution s** |
| **T444** | **Withdrawal Fee** | **N/A** | [**WDRLFEE**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/OmniScript/OmniScriptProcessingExits.htm%23WDRLFEE) | **N/A** | **Fees** |
| **T404/ T444** | **Withdrawal Forfeiture** | **N/A** | [**WDRLFORF**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Vesting/OmniScript/OmniScriptProcessingExits/WithdrawalForfeiture.htm) | **01-98** | **Vesting** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Transaction Number** | **Description** | **OmniScript Allowed on**  **Transaction?** | **Stored Text File Name** | **OmniScript**  **Suffix Value** | **Object** |
| **T444** | **Withdrawal Suspension** | **N/A** | [**WDRLSUSP**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/OmniScript/OmniScriptProcessingExits/WithdrawalSuspension(WDRLSUSPnn)/WithdrawalSuspension(WDRLSUSPnn).htm) | **01-99** | **Distribution s** |
| **T454** | **Excess Deferrals** | **N/A** | [**EXCSMAX**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/OmniScript/OmniScriptProcessingExits/MaximumExcessAvailableforReturn(EXCSMAXn/MaximumExcessAvailableforReturn(EXCSMAXn.htm) | **01-99** | **Distribution s** |
| **T454** | **Excess Deferrals** | **N/A** | [**EXCSXFER**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/OmniScript/OmniScriptProcessingExits/MaximumExcessAvailableforTransfer(EXCSXF/MaximumExcessAvailableforTransfer(EXCSXF.htm) | **01-99** | **Distribution s** |
| **T454** | **Excess Deferrals** | **N/A** | [**EXCSEARN**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/OmniScript/OmniScriptProcessingExits/ReturnofEarningsonExcess(EXCSEARNnn)/ReturnofEarningsonExcess(EXCSEARNnn).htm) | **01-99** | **Distribution s** |
| **T530/ T563/**  **T564/**  **ANANES** | **Annuity PreCalculation** | **N/A** | [**ANNUITY.OMNISCRIPT.###**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/OmniScript/Pre-CalcOmniScript.htm) | **001-999, AAA-ZZZ** | **Annuities** |
| **T530** | **Annuity Validate OmniScript** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/OmniScript/ValidateUpdate.htm) | **N/A** | **Annuities** |
| **T530** | **Annuity Update OmniScript** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/OmniScript/ValidateUpdate.htm) | **N/A** | **Annuities** |
| **T588** | **Universal Update** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/Transactions/Universal(T588-T988)/Universal(T588-T988).htm) | **N/A** | **OmniScript** |
| **T613** | **Plan Maintenance** | **N/A** | [**VLDT.PL**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Plans** |
| **T801** | **Validate Participant Allocation Table** | **N/A** | [**VLDT.ALLOC.PCT**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/OmniScript/OmniScriptProcessingExits/VLDT.ALLOC.PCT.htm) | **N/A** | **Participants** |
| **T801/ T813** | **Participant Header Maintenance** | **N/A** | [**VLDT.PH**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/OmniScript/OmniScriptProcessingExits/VLDT.PH.htm) | **N/A** | **Participants** |
| **T813** | **Participant Header Maintenance** | **N/A** | [**VLDT813**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/OmniScript/OmniScriptProcessingExits/VLDT813.htm) | **N/A** | **Participants** |
| **T815** | **Change Participant Allocation Table** | **N/A** | [**VLDT815**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/OmniScript/OmniScriptProcessingExits/VLDT815.htm) | **N/A** | **Participants** |
| **T833** | **File Maintenance OmniScript** | **Yes** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/Transactions/FileMaintenanceCalculator(T833)/FileMaintenanceCalculator(T833).htm) | **N/A** | **OmniScript** |
| **T840** | **Associated Individual Maintenance** | **N/A** | [**VLDT.ASSOC.IND**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/AssociatedIndividual/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Associated Individuals** |
| **T848** | **Rate of Return Maintenance** | **N/A** | [**HIBRFILT**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/RateOfReturn/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Rate of Return** |
| **T885** | **Loan Fund Allocation Maintenance** | **N/A** | **VLDT.885** | **N/A** | **Loan** |
| **T895** | **Generic Maintenance** | **N/A** | [**VLDT.895**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/OmniScript/OmniScriptProcessingExits.htm) | **N/A** | **Base**  **Utilities** |
| **T931** | **Statement OmniScript Text File** | **N/A** | [**STMTCALC**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/OmniScript/OmniScriptProcessingExits/STMTCALC.htm) | **0001-9999** | **Participants** |
| **T933** | **Custom Report Generator** | **Yes** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/Transactions/CustomReportGenerator(T933)/CustomReportGenerator(T933).htm) | **N/A** | **OmniScript** |
| **T950** | **Top-Heavy Test** | **N/A** | [**TOPHEAVY**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/OmniScript/OmniScriptProcessingExits.htm%23TOPHEAVY) | **01-98** | **Compliance** |
| **Transaction Number** | **Description** | **OmniScript Allowed on**  **Transaction?** | **Stored Text File Name** | **OmniScript**  **Suffix Value** | **Object** |
| **T953** | **Minimum Coverage Test** | **N/A** | [**MINCOV**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/OmniScript/OmniScriptProcessingExits.htm%23MINCOV) | **01-98** | **Compliance** |
| **T957** | **Participant Valuation Summary** | **N/A** | [**957CUST**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/OmniScript/OmniScriptProcessingExits.htm%23T957CUST) | **01-99** | **Funds** |
| **T966** | **Custom Full Page Report Generator** | **Yes** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/Transactions/CustomFullPageReportGenerator(T966)/CustomFullPageReportGenerator(T966).htm) | **N/A** | **OmniScript** |
| **T970** | **Non-Discrimination Test** | **Yes** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/OmniScript/OmniScriptProcessingExits.htm%23T970) | **N/A** | **Compliance** |
| **T972** | **402G/415 Contribution**  **Limit Test** | **Yes** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/OmniScript/OmniScriptProcessingExits.htm%23T972) | **N/A** | **Compliance** |
| **T988** | **Universal Report** | **N/A** | [**Any Unique Name**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/Transactions/Universal(T588-T988)/Universal(T588-T988).htm) | **N/A** | **OmniScript** |
| **T995** | **Master File Validation** | **N/A** | [**VLDTPLAN**](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/OmniScript/OmniScriptProcessingExits.htm%23VLDTPLAN) | **N/A** | **Plans** |

**Note: For more information on any of these textfiles, please see the OmniScript section of the appropriate Object Level Guide.**

### 5.4.2 OmniScript Text Allowed in Transaction

This field informs the user whether or not OmniScript text can be entered directly onto the transaction. A yes in this field says that a user can input OmniScript text directly onto the transaction. Including, but not limited to, commands such as the textfile= command. An N/A in this field indicates that OmniScript text cannot be directly input onto the transaction. This leaves two options:

* If the OmniScript suffix value option is allowed for the text file name, the user can enter the suffix value to call the text file.
* If the OmniScript suffix value is not allowed for the text file name, the OmniScript is automatically executed whenever the applicable transaction runs, unless the applicable transaction “OmniScript Override” is selected. Refer to the transaction documentation for additional information and options, which vary user has no control as to whether the OmniScript text will be used.

### 5.4.3 OmniScript Suffix Value

This is the two-digit code that must be entered on the transaction to call the text file. For example, to access the text file 'CNTBBASE01', you must enter '01' in the 'Base Calc Suffix' field on the T154.

## 5.5 Control Files

Textfiles are used to store many different kinds of OmniPlus system information, and can also be used to store user defined information. Textfiles are powerful tools that add flexibility to processing, reporting, and control of the OmniPlus system. They are used to store OmniScript, which extends the functionality of OmniPlus without the need for traditional code. In addition, text files are used to maintain user-defined fields and system control files.

OmniPlus text files are “containers” used to support the storing and access of multiple lines of text. Each line in a textfile can be up to 200 characters. Up to 9999 lines per textfile is supported.

**Naming Convention**



* A 2 byte Prefix, identifying the file type (e.g., 'CT' – Control information)
* A 1-25 byte File Name (Predefined for OmniPlus system files, variable for user defined

files)

Special Control files exist to establish additional system processing controls. Control files in the Local Plan may inherit variables from a Global Plan control file.

Control files are stored as Textfiles with the Prefix “CT” and the naming conventions shown in the table below. Control files can exist at either the global or local plan level.

|  |  |  |
| --- | --- | --- |
| **Object Guide** | **Description** | **Stored Control File Name** |
| Addresses | Alternate Address Control | [ALTERNATE.ADDRESS](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Addresses/ControlFiles/ControlFiles.htm) |
| Annuities | Mortality Table Controls | [ANNU.MORTALITY.\*\*\*\*\*\*\*\*](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/ControlFiles/ControlFiles.htm) |
| Annuities | Annuity Control Codes | [ANNUITY](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/ControlFiles/ControlFiles.htm) |
| Annuities | Annuity Control Definitions | [ANNUITY.DEFINITION.\*\*\*\*](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/ControlFiles/ControlFiles.htm) |
| Annuities | Annuity Global Group Control | [ANNUITY.GLOBAL.GROUP](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/ControlFiles/ControlFiles.htm) |
| Base Utilities | Business Calendar Control | [CCYY.BUSINESS.CALENDAR](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/ControlFiles/ControlFiles.htm) |
| Cash | Cash Control | [CASH](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/ControlFiles/ControlFiles.htm) |
| Checks | Check Control | [CHECK](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Checks/ControlFiles/ControlFiles.htm) |
| Compliance | Nondiscrimination Control | [COMPLIANCE](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/ControlFiles/CT.COMPLIANCE.htm) |
| Compliance | Form 5500 | [FORM.5500](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/ControlFiles/FORM.5500.htm) |
| Confirmation Processing | Confirm Control | [CONFIRM](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/AdditionalGuides/ConfirmProcessing/ControlFiles/ControlFile.htm) |
| Contributions | Contribution Control | [CONTRIBUTION](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/ControlFiles/ControlFiles.htm) |
| Distributions | Disbursement Worksheet | [DISB.WORK](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/ControlFiles/ControlFiles.htm) |
| Distributions | Distribution | [DISTRIBUTION](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/ControlFiles/ControlFiles.htm) |
| Earnings | Earnings Control | [EARNINGS](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Earnings/ControlFiles/ControlFiles.htm%23earningsctltextfile) |
| Eligibility | Eligibility Control | [ELIGIBILITY](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Eligibility/ControlFiles/ControlFiles.htm) |
| Fees | Fee Control | [FEE](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/ControlFiles/ControlFiles.htm) |
| Fees | Fee - Redemption Fees | [FEE.REDEMPTION](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/ControlFiles/ControlFiles.htm%23RedemptionFeeControlFile) |
| Forecasting | Forecasting Global Groups | [FORECAST.GLOBAL.GROUP](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Forecasting/ControlFiles/ControlFiles.htm%23Global) |
| Forecasting | Forecasting Control | [FORECASTER](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Forecasting/ControlFiles/ControlFiles.htm%23ForecastCTL) |
| Foreign Exchange | Foreign Exchange Currency  Control | [FOREIGN.EXCHANGE.CURRENCY](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ForeignExchangeRate/ControlFiles/ControlFiles.htm) |
| Funds | Participant Activity Summary  Control | [PARTACTSUM##](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/ControlFiles/ControlFiles.htm) |
| History | Activity History Summary Control | [HISTACTSUM##](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/History/ControlFiles/ControlFiles.htm) |
| History | Activity History Purge and Condense | [RETAIN.BR](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/History/ControlFiles/ControlFiles.htm) |
| Installation | Environment Control | [ENVIRONMENT](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/InstallationOperation/Common/JobOverview.htm%23hdg_3_15) |
| Installments | Installment Control Codes | [INSTALLMENT](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Installments/ControlFiles/ControlFiles.htm) |
| Insurance | Insurance Control | [INSURANCE](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Insurance/ControlFiles/ControlFiles.htm) |
| Loans | Loan Control | [LOAN](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/ControlFiles/index.htm) |
| Messages | Message Control | [MESSAGE](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Messages/ControlFiles/ControlFiles.htm) |
| Participants | Statement Control | [STMTCTL.####](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/ControlFiles/ControlFiles.htm) |
| Persons | Person Control Codes | [PERSON](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Persons/ControlFiles/ControlFiles.htm) |
| Rate of Return | Rate of Return | [RATE.OF.RETURN](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/RateOfReturn/ControlFiles/ControlFiles.htm) |
| Report Management | Report Control | [REPORT](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ReportManagement/ControlFiles/ControlFiles.htm) |
| Service | Leave of Absence | [LEAVE.OF.ABSENCE](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Service/ControlFiles/ControlFiles.htm) |
| Service | Service Control | [SERVICE](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Service/ControlFiles/ControlFiles.htm) |
| Share Trade | Share Request | [SHARE.REQUEST](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ShareTrade/ControlFiles/TextControlFiles.htm) |
| Share Trade | Share System | [SHARE.SYSTEM](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ShareTrade/ControlFiles/TextControlFiles.htm) |
| Share Trade | Trade Request | [TRADE.REQUEST](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ShareTrade/ControlFiles/TextControlFiles.htm) |
| Share Trade | Trade System | [TRADE.SYSTEM](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ShareTrade/ControlFiles/TextControlFiles.htm) |
| Sub Plans | Sub Plan Control | [SUBPLAN](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/SubPlan/ControlFiles/ControlFiles.htm) |
| Sub accounting | Sub accounting | [SUBA](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/SubAccounting/ControlFiles/ControlFiles.htm) |
| Tax Reporting | Tax Reporting | [TAX.RPT.OPT](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TaxReporting/ControlFiles/ControlFiles.htm%23TAXRPTOPT) |
| Tax Reporting | Trustee Information | [TRUSTEE](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TaxReporting/ControlFiles/ControlFiles.htm%23TRUSTEE) |
| Transaction Management | VTRAN Control | [VTRAN](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TransactionManagement/ControlFiles/ControlFiles.htm) |
| Transfers | Transfer Control | [TRANSFER](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/ControlFiles/ControlFiles.htm) |
| Vesting | Forfeiture Control | [FORFEITURE](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Vesting/ControlFiles/ForfeitureControlFile.htm) |
| Vesting | Vesting Control | [VESTING](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Vesting/ControlFiles/VestingControlFile.htm) |

# 6.0 Omni Plus Transactions & Records

## 6.1 Major Transactions

|  |  |  |
| --- | --- | --- |
| **Transaction** | **Description** | **Object Guide** |
| RETAINBR | Activity History Purge and Condense (RETAINBR) | History |
| T002 | [Transaction File Utility (T002)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TransactionManagement/Transactions/TransactionFileUtility(T002)/TranFileUtilityT002.htm) | Transaction Management |
| T005 | [PROCESS CONTROL (T005)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/Transactions/PROCESSCONTROL(T005)/PROCESSCONTROL(T005).htm) | Base Utilities |
| T006 | [Output Control (T006)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ReportManagement/Transactions/OUTPUTCONTROL(T006)/OUTPUTCONTROL(T006).htm) | Report Management |
| T010 | [PLAN RESTORE / ACTIVITY HISTORY PURGE (T010)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/Transactions/PLANRESTOREACTIVITYHISTORYPURGE(T010)/PLANRESTOREACTIVITYHISTORYPURGE(T010).htm) | Base Utilities |
| T013 | [EXTENDED MASTER FILE BACKUPUTILITY (T013)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/Transactions/EXTENDEDMASTERFILEBACKUPUTILITY(T013)/EXTENDEDMASTERFILEBACKUPUTILITY(T013).htm) | Base Utilities |
| T014 | [EXTENDED BACKUP SELECTION CRITERIA (T014)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/Transactions/EXTENDEDBACKUPSELECTIONCRITERIA(T014)/EXTENDEDBACKUPSELECTIONCRITERIA(T014).htm) | Base Utilities |
| T021 | [Sub Accounting Utility (T021)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Subaccounting/Transactions/SUBACCOUNTINGUTILITY(T021)/T021.htm) | Sub accounting |
| T022 | [Loan Utility (T022)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/LoanUtility(T022)/LoanUtility(T022).htm) | Loans |
| T027 | [FOREIGN EXCHANGE RECORD MAINTENANCE (T027)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ForeignExchangeRate/Transactions/EXCHRECMAINT(T027)/EXCHRECMAINT(T027).htm) | Foreign Exchange Rate |
| T028 | [Price ID File Maintenance (T028)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Prices/Transactions/PRICEIDMAINT(T028)/PRICEIDMAINT(T028).htm) | Prices |
| T029 | [Price Utility (T029)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Prices/Transactions/PRICEUTILITY(T029)/PRICEUTILITY(T029).htm) | Prices |
| T030 | [Date Card (T030)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/Transactions/DATECARD(T030)/DATECARD(T030).htm) | Base Utilities |
| T035 | [ACTIVITY ROLL (T035)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/Transactions/ACTIVITYROLL(T035)/ACTIVITYROLL(T035).htm) | Plans |
| T039 | [Investment Action Utility (T039)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Investments/Transactions/INVESTMENTACTIONUTILITY(T039)/INVESTMENTACTIONUTILITY(T039).htm) | Investments |
| T051 | [DELETE TRANSACTION (T051)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/Transactions/DELETEMASTERRECORDS(T051)/DELETEMASTERRECORDS(T051).htm) | Base Utilities |
| T055 | [Base File Maintenance Purge (T055)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/History/Transactions/BASEFILEMAINTPURGE(T055)/BASEFILMAINTPURGE(T055).htm) | History |
| T060 | [Activity Reversal (T060)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Adjustments/Transactions/ACTIVITYREVERSAL(T060)/ACTIVITYREVERSAL(T060).htm) | Adjustments |
| T070 | [ALTERNATE INQUIRY MAINTENANCE (T070)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/Transactions/ALTERNATEINQUIRYMAINTENANCE(T070)/ALTERNATEINQUIRYMAINTENANCE(T070).htm) | Base Utilities |
| T090 | [UCOM Report File Utility(T090)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ReportManagement/Transactions/UCOMREPORTFILEUTILITY(T090)/UCOMREPORTFILEUTILITY(T090).htm) | Report Management |
| T091 | [UCOM Report File Backup(T091)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ReportManagement/Transactions/UCOMREPORTFILEBACKUP(T091)/UCOMREPORTFILEBACKUP(T091).htm) | Report Management |
| T092 | [UCOM Report File Restore(T092)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ReportManagement/Transactions/UCOMREPORTFILERESTORE(T092)/UCOMREPORTFILERESTORE(T092).htm) | Report Management |
| T093 | [UCOM Report File Print Facility (T093)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ReportManagement/Transactions/UCOMREPORTFILEPRINTFACILITY(T093)/UCOMREPORTFILEPRINTFACILITY(T093).htm) | Report Management |
| T098 | [Compensation Record Purge/Condense (T098)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compensation/Transactions/CompensationPurgeCondense(T098)/CompPurge-Condense(T098).htm) | Compensation |
| T104 | [Plan Contribution Totals (T104)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/Transactions/PlanLevelContributionTotals(T104)/PlanLevelCntbTotals(T104).htm) | Contributions |
| T114 | [Participant Contribution Detail (T114)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/Transactions/PartContributionDetail(T114)/PartCntbDetail(T114).htm) | Contributions |
| T154 | [ER Contribution Based on EE Compensation (T154)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/Transactions/ERCntbBasedonEEComp(T154)/ERCntbBasedonEEComp(T154).htm) | Contributions |
| T165 | [ER Contribution Based on EE Contribution (T165)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/Transactions/ERCntbBasedonEECntb(T165)/ERCntbBasedonEECntb(T165).htm) | Contributions |
| T180 | [Compensation Plan Accumulation (T180)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compensation/Transactions/CompPlanAccumulation(T180)/CompPlanAccumulation.htm) | Compensation |
| T181 | [Compensation Participant Accumulation (T181)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compensation/Transactions/CompPartAccumulation(T181)/CompPartAccumulation.htm) | Compensation |
| T205 | [Cash Earnings (T205)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Earnings/Transactions/CASHEARNINGS(T205)/CASHEARNINGS(T205).htm) | Earnings |
| T207 | [Daily Accrual Earnings(T207)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Earnings/Transactions/DAILYACCRUALEARNINGS(T207)/DAILYACCRUALEARNINGS(T207).htm) | Earnings |

|  |  |  |
| --- | --- | --- |
| T219 | [Equity Dividends (T219)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Earnings/Transactions/EQUITYDIVIDENDS(T219)/EQUITYDIVIDENDS(T219).htm) | Earnings |
| T235 | [Stock Split (T235)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ParticipantShares/Transactions/STOCKSPLIT(T235)/STOCKSPLIT(T235).htm) | Participant Shares |
| T245 | [Fee Allocation (T245)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/Transactions/FEEALLOCATION(T245)/FEEALLOCATION(T245).htm) | Fees |
| T246 | [Daily Fees (T246)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/Transactions/DAILYFEES(T246)/DAILYFEES(T246).htm) | Fees |
| T247 | [Fee Allocation Utility (T247)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/Transactions/FEEALLOCATIONUTILITY(T247)/FEEALLOCATION(T247).htm) | Fees |
| T275 | [Stock/Unit Purchase (T275)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ParticipantShares/Transactions/STOCKUNITPURCHASE(T275)/STOCKUNITPURCHASE(T275).htm) | Participant Shares |
| T277 | [Stock/Unit Sell (T277)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ParticipantShares/Transactions/STOCKUNITSELL(T277)/STOCKUNITSELL(T277).htm) | Participant Shares |
| T301 | [Miscellaneous Receipts/Disbursements (T301)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Adjustments/Transactions/MISCRECEIPTSDISB(T301)/MISCRECEIPTSDISB(T301).htm) | Adjustments |
| T317 | [Reserve Transfer (T317)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/ReserveTransfer(T317)/ReserveTransfer(T317).htm) | Transfers |
| T363 | [Forfeiture Utility Transaction (T363)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Vesting/Transactions/FORFEITUREUTILITYTRANSACTION(T363)/FORFEITUREUTILITYTRANSACTION(T363).htm) | Vesting |
| T366AI | [Inter-Participant Transfer (T366AI)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/Inter-PartTransferAI(T366AI)/Inter-PartTransferAI(T366AI).htm) | Transfers |
| T381 | [Inter-Fund Transfer (T381)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/ParticipantInter-FundTransfer(T381)/ParticipantInter-FundTransfer(T381).htm) | Transfers |
| T381C | [Combined Inter - Fund Transfer -(T381C)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/CombinedInter-FundTransfer-(T381C)/CombinedInter-FundTransfer-(T381C).htm) | Transfers |
| T383 | [Loan Re-amortization (T383)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/LoanRe-amortization(T383)/LoanRe-amortization(T383).htm) | Loans |
| T384 | [Loan Issue (T384)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/LoanIssuance(T384)/LoanIssuance(T384).htm) | Loans |
| T385 | [Loan Payment (T385)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/LoanPayment(T385)/LoanPayment(T385).htm) | Loans |
| T386 | [Loan Adjustment (T386)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/LoanAdjustment(T386)/LoanAdjustment(T386).htm) | Loans |
| T387 | [Interest Accumulator (T387)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/InterestAccumulator(T387)/InterestAccumulator(T387).htm) | Loans |
| T388 | [Delinquency/Deemed Distribution Utility (T388)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/DelinquencyDeemedDistributionUtility(T388)/DelinquencyDeemedDistributionUtility(T388).htm) | Loans |
| T394 | [Incoming Plan Transfer (T394)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/IncomingPlanTransfer(T394)/IncomingPlanTransfer(T394).htm) | Transfers |
| T395 | [Outgoing Plan Transfer (T395)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/OutgoingPlanTransfer(T395)/OutgoingPlanTransfer(T395).htm) | Transfers |
| T404 | [Termination Distribution (T404)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/Transactions/TERMINATIONDISTRIBUTION(T404)/TERMINATIONDIST(T404).htm) | Distributions |
| T444 | [Withdrawal (T444)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/Transactions/WITHDRAWALS(T444)/WITHDRAWALS(T444).htm) | Distributions |
| T444 | [Loan Default (T444)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/LoanDefault(T444)/LoanDefault(T444).htm) | Loans |
| T454 | [Return of Excess (T454)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/Transactions/RETURNOFEXCESSCTBNDEF(T454)/RETURNOFEXCESSCTBNDEF(T454).htm) | Distributions |
| T464 | [Installment Payment Processing (T464)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Installments/Transactions/INSTALLMENTPAYMENTPROCESSINGTRANSACTION(/INSTALLMENTPAYMENTPROCESSINGTRANSACTION(.htm) | Installments |
| T500 | [Cash CCA Maintenance Transaction (T500)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CCAMaintenanceTran(T500)/CashCCAMaintTran(T500).htm) | Cash |
| T502 | [Cash Pointer Validation/Rebuild Transaction (T502)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashPointerValidRbld(T502)/CashPointerValidRbld(T502).htm) | Cash |
| T503 | [Cash Validation and Realignment Transaction (T503)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashValidRealignTran(T503)/CashValidRealign(T503).htm) | Cash |
| T504 | [Post Cash Expense Bill Transaction (T504)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/ExpenseBillTransaction(T504)/PostExpenseBillTran(T504).htm) | Cash |
| T506 | [T50CASH- DEPOSIT (T506)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashDeposit(T506)/CashDeposit(T506).htm) | Cash |
| T507 | [Cash Transaction (T507)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashTransaction(T507)/CashTransaction(T507).htm) | Cash |
| T508 | [CASH- PURGE (T508)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashPurge(T508)/CashPurge(T508).htm) | Cash |
| T509 | [Cash History File Maintenance (T509)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashHistoryFileMaintenance(T509)/CashHistFileMaint(T509).htm) | Cash |
| T510 | [Share System Basic Functions (T510)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ShareTrade/Transactions/ShareSystemTransactions/SHARECONTROLTRANSACTION(T510).htm) | ShareTrade |
| T511 | [Share System Financials (T511)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ShareTrade/Transactions/ShareSystemTransactions/SHARECONTROLTRANSACTION(T510).htm) | ShareTrade |
| T515 | [Trade System Transaction (T515)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ShareTrade/Transactions/TradeSystemTransactions/T515-TRADESYSTEMACTIONENTRYSCREEN.htm) | Share Trade |
| T530 | [Annuity Add (T530)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/ANNUITYADD(T530)/ANNUITYADD(T530).htm) | Annuities |
| T531 | [ModifyAnnuity (T531)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/MODIFYANNUITY(T531)/MODIFYANNUITY(T531).htm) | Annuities |

|  |  |  |
| --- | --- | --- |
| T532 | [Delete Annuity Master(T532)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/DELETEANNUITYMASTER(T532)/DELETEANNUITYMASTER(T532).htm) | Annuities |
| T533 | [Purge Annuities (T533)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/PURGEANNUITIES(T533)/PURGEANNUITIES(T533).htm) | Annuities |
| T534 | [UpdateAnnuity Payments (T534)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/UPDATEFIXEDANNPMTAMT(T534)/UPDATEFIXEDANNPMTAMT(T534).htm) | Annuities |
| T536 | [Transfer Annuity (T536)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/TRANSFERANNUITY(T536)/TRANSFERANNUITY(T536).htm) | Annuities |
| T537 | [Change Annuitant ID (T537)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/CHANGEANNUITANTID(T537)/CHANGEANNUITANTID(T537).htm) | Annuities |
| T538 | [AnnuityPayment Processing (T538)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/ANNUITYPAYMENTPROCESSING(T538)/ANNUITYPAYMENTPROCESSING(T538).htm) | Annuities |
| T540 | [Establish Person Transaction (T540)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Persons/Transactions/EstablishPerson(T540)/EstablishPerson(T540).htm) | Persons |
| T545 | [Note Transaction (T545)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Notes/Transactions/NOTETRANSACTION(T545)/NOTETRANSACTION(T545).htm) | Notes |
| T560 | [Address Record Entry Screen (T560)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Addresses/Transactions/ADDRESSRECORDENTRYSCREEN(T560)/ADDRESSRECORDENTRYSCREEN(T560).htm) | Addresses |
| T561 | [Change Address Record (T561)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Addresses/Transactions/CHANGEADDRESSRECORD(T561)/CHANGEADDRESSRECORD(T561).htm) | Addresses |
| T562 | [Delete Address (T562)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Addresses/Transactions/DELETEADDRESS(T562)/DELETEADDRESS(T562).htm) | Addresses |
| T563 | [Forecast Projection Estimate (T563)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Forecasting/Transactions/FORECASTPROJECTIONESTIMATE(T563)/FORECASTPROJECTIONESTIMATE(T563).htm) | Forecasting |
| T564 | [Forecast Contribution Estimate (T564)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Forecasting/Transactions/FORECASTCONTRIBUTIONESTIMATE(T564)/FORECASTCONTRIBUTIONESTIMATE(T564).htm) | Forecasting |
| T566 | [Forecast Record Purge(T566)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Forecasting/Transactions/FORECASTRECORDPURGE(T566)/FORECASTRECORDPURGE(T566).htm) | Forecasting |
| T588 or T988 | [Universal Transactions (T588 or T988)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/Transactions/Universal(T588-T988)/Universal(T588-T988).htm) | Omni Script |
| T600 | [PLAN ADD (T600)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/Transactions/PLANADD(T600)/PLANADD.htm) | Plans |
| T613 | [PLAN MAINTENANCE (T613)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/Transactions/PLANMAINTENANCE(T613)/PLANMAINTENANCE(T613).htm) | Plans |
| T650 | [Product Master Utility(T650)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/Transactions/PRODUCTMAINTENANCE(T650)/PRODUCTMAINTENANCETRANSACTION(T650).htm) | Plans |
| T660 | [DEDUCTION HEADERMAINTENANCE (T660)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Disbursements/Transactions/DeductionHeaderMaint(T660)/DeductionHeaderMaint(T660).htm) | Disbursements |
| T700 | [Add a New Fund (T700)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/Transactions/ADDANEWFUND(T700)/ADDANEWFUND(T700).htm) | Funds |
| T701 | [Fund Control Record Add/Change (T701)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/Transactions/FUNDCONTROLADDCHANGE(T701)/FUNDCONTROLRECORDADDCHANGE(T701).htm) | Funds |
| T713 | [Fund Control Record Maintenance (T713)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/Transactions/FUNDCONTROLRECORDMAINTENANCE(T713)/FUNDCONTROLRECORDMAINTENANCE(T713).htm) | Funds |
| T717 | [Plan Level Allocations Percents Change (T717)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/Transactions/PlanLevelAllocPercentsChange(T717)/PlanAllocPercentsChange(T717).htm) | Contributions |
| T801 | [Participant Create/Update (T801)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/Transactions/PARTICIPANTCREATEUPDATE(T801)/PARTICIPANTCREATEUPDATE(T801).htm) | Participants |
| T813 | [Participant Header Record Maintenance (T813)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/Transactions/PARTICIPANTHEADERRECORDMAINTENANCE(T813)/PARTICIPANTHEADERRECORDMAINTENANCE(T813).htm) | Participants |
| T815 | [Change Participant's Allocation (T815)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/Transactions/PartAllocationChange(T815)/PartAllocationChange(T815).htm) | Contributions |
| T820 | [Participant Voice Response Record Maintenance (T820)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/VoiceResponse/Transactions/PARTICIPANTVOICERESPONSERECORDMAINTENANC/PARTICIPANTVOICERESPONSERECORDMAINTENANC.htm) | Voice Response |
| T823 | [Participant Fund Record Maintenance (T823)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/Transactions/PARTICIPANTFUNDRECORDMAINTENANCE(T823)/PARTICIPANTFUNDRECORDMAINTENANCE(T823).htm) | Funds |
| T824 | [Insurance Event Processor (T824)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Insurance/Transactions/InsEventProcessor(824).htm) | Insurance |
| T828 | [Insurance Detail Create/Update (T828)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Insurance/Transactions/InsDetailCreateUpdate(828).htm) | Insurance |
| T833 | [File Maintenance Omni Script (T833)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/Transactions/FileMaintenanceCalculator(T833)/FileMaintenanceCalculator(T833).htm) | Omni Script |
| T840 | [Associated Individual Maintenance (T840)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/AssociatedIndividual/Transactions/AssociatedIndMnt(T840).htm) | Associated Individual |
| T844 | [Minimum Distribution (T844)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/Transactions/MINDISTPROCESSING(T844)/MINDISTPROCESSING(T844).htm) | Distributions |
| T848 | [Rate of Return Record Maintenance (T848)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/RateOfReturn/Transactions/RORMAINT(T848).htm) | Rate Of Return |
| T860 | [Inter-Plan Transfer Control(T860)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/Inter-PlanTransferControl(T860)/Inter-PlanTransferControl(T860).htm) | Transfers |
| T861 | [Inter-Plan Transfer - Detail (T861)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/Inter-PlanTransfer-Detail(T861)/Inter-PlanTransfer-Detail(T861).htm) | Transfers |
| T868 | [Equity Wash Utility (T868)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/EquityWashUtility(T868)/EquityWashUtility(T868).htm) | Transfers |
| T870 | [Base Record Maintenance (T870)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/History/Transactions/BASERECORDMAINT(T870)/BASERECORDMAINT(T870).htm) | History |

|  |  |  |
| --- | --- | --- |
| T871 | [Foreign Exchange History Record Maintenance (T871)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ForeignExchangeRate/Transactions/EXCHHISTORYRECMAINT(T871)/EXCHHISTORYRECMAINT(T871).htm) | Foreign Exchange Rate |
| T875 | [Compensation Record Maintenance (T875)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compensation/Transactions/CompRecMaint(T875)/CompRecMaint.htm) | Compensation |
| T876 | [Salary Record Maintenance (T876)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compensation/Transactions/SalaryRecMaint(T876)/SalaryRecMaint.htm) | Compensation |
| T877 | [Service Record Maintenance (T877)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Service/Transactions/SERVICERECORDMAINTENANCE(T877).htm) | Service |
| T880 | [Activity History Maintenance (T880)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/History/Transactions/ACTHISTMAINT(T880)/ACTHISTMAINT(T880).htm) | History |
| T881 | [Loan File Maintenance (T881)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/LoanFileMaintenance(T881)/LoanFileMaintenance(T881).htm) | Loans |
| T882 | [Extended PDF Record Maintenance (T882)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Subaccounting/Transactions/EXTENDEDPDFRECORDMAINTENANCE(T882)/T882.htm) | Sub accounting |
| T884 | [Investment Control File Maintenance (T884)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Investments/Transactions/INVESTMENTCONTROLFILEMAINTENANCE(T884)/INVESTMENTCONTROLFILEMAINTENANCE(T884).htm) | Investments |
| T885 | [Change Loan Fund Allocation (T885)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/ChangeLoanFundAllocation(T885)/ChangeLoanFundAllocation(T885).htm) | Loans |
| T886 | [DISBURSEMENT ROLLOVERMAINTENANCE (T886)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Disbursements/Transactions/DisbRollMaint(T886)/DisbRollMaint(T886).htm) | Disbursements |
| T887 | [DEDUCTION GROUPMAINTENANCE (T887)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Disbursements/Transactions/DeductionGroupMaint(T887)/DeductionGroupMaint(T887).htm) | Disbursements |
| T888 | [Div/Sub Record Maintenance (T888)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/DivSub/Transactions/DIVSUBRECORDMAINTENANCE(T888)/DIVSUBRECORDMAINTENANCE(T888).htm) | Div Sub |
| T889 | [DISBURSEMENT RECORD MAINTENANCE (T889)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Disbursements/Transactions/DisbRecordMaint(T889)/DISBRECMAINT(T889).htm) | Disbursements |
| T890 | [SOURCE CONTROL RECORD MAINTENANCE (T890)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Sources/Transactions/SOURCECONTROLRECORDMAINTENANCE(T890)/SOURCECONTROLRECORDMAINTENANCE(T890).htm) | Sources |
| T892 | [Base Text Record File Maintenance (T892)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseText/Transactions/BASETEXTRECORDFILEMAINTENANCE(T892)/BASETEXTRECORDFILEMAINTENANCE(T892).htm) | Base Text |
| T894 | [Sub Plan Record Maintenance (T894)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/SubPlan/Transactions/SUBPLANRECMAINT(T894)/SUBPLANRECMAINT(T894).htm) | Sub Plan |
| T895 | [Generic Record Maintenance Transaction (T895)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/Transactions/GENERICRECORDMAINTENANCE(T895)/GENERICRECORDMAINTENANCE(T895).htm) | Base Utilities |
| T896 | [Installment Maintenance (T896)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Installments/Transactions/INSTALLMENTMAINTENANCE(T896)/INSTALLMENTMAINTENANCE(T896).htm) | Installments |
| T899 | [PLANCARD CREATION (T899)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/Transactions/PLANCARDCREATION(T899)/PLANCARDCREATION(T899).htm) | Plans |
| T901 | [Participant Record Dump (T901)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/Transactions/PARTICIPANTRECORDDUMP(T901)/PARTICIPANTRECORDDUMP(T901).htm) | Participants |
| T903 | [Fund Record Dump (T903)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/Transactions/FUNDRECORDDUMP(T903)/FUNDRECORDDUMP(T903).htm) | Funds |
| T907 | [MASTER FILE DATA EXTRACT PROCESS (T907)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/BaseUtilities/Transactions/MASTERFILEDATAEXTRACTPROCESS(T907)/MASTERFILEDATAEXTRACTPROCESS(T907).htm) | Base Utilities |
| T910 | [CHECK DATA ELEMENT REPORT (T910)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Checks/Transactions/CHECKDATAELEMENTREPORT(T910)/CHECKDATAELEMENTREPORT(T910).htm) | Checks |
| T912 | [Installment Data Element Report (T912)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Installments/Transactions/InstallmentReport(T912)/InstallmentReport(T912).htm) | Installments |
| T914 | [EFT Data Element Dump(T914)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/History/Transactions/EFTDATAELEMENTDUMP(T914)/EFTDATAELEMENTDUMP(T914).htm) | History |
| T916 | [Activity History Summary(T916)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/History/Transactions/ACTHISTSUMMARY(T916)/ACTHISTSUMMARY(T916).htm) | History |
| T919 | [Compensation Data Element Report (T919)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compensation/Transactions/CompDataElementRep(T919)/CompDataElementReport.htm) | Compensation |
| T920 | [Salary Data Element Report (T920)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compensation/Transactions/SalaryDataElementRep(T920)/SalaryDataElementReport.htm) | Compensation |
| T923 | [Investment Valuation Statement (T923)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/Transactions/INVESTMENTVALUATIONSTATEMENT(T923)/INVESTMENTVALUATIONSTATEMENT(T923).htm) | Funds |
| T926 | [Share System Reports (T926)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ShareTrade/Transactions/ShareSystemTransactions/SHARECONTROLTRANSACTION(T510).htm) | Share Trade |
| T929 | [Service Data Element Report (T929)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Service/Transactions/SERVICEDATAELEMENTREPORT(T929).htm) | Service |
| T933 | [CUSTOMREPORT GENERATOR (T933)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/Transactions/CUSTOMREPORTGENERATOR(T933)/CUSTOMREPORTGENERATOR(T933).htm) | Omni Script |
| T935 | [Annuity Master Validation (T935)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/ANNUITYMASTERVALIDATION(T935)/ANNUITYMASTERVALIDATION(T935).htm) | Annuities |
| T936 | [Annuity Beneficiary Estimate (T936)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/ANNUITYBENEFICIARYESTIMATE(T936)/ANNUITYBENEFICIARYESTIMATE(T936).htm) | Annuities |
| T943 | [Insurance Policy Report (T943)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Insurance/Transactions/InsurancePolicyRpt(T943).htm) | Insurance |
| T944 | [Associated Individual Record Dump (T944)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/AssociatedIndividual/Transactions/AssociatedIndDump(T944).htm) | Associated Individual |
| T945 | [TAX REPORTING (T945)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TaxReporting/Transactions/TaxReporting(T945)/TAXREPORTING(T945).htm) | Tax Reporting |
| T948 | [5498IRA QVEC REPORT REQUEST (T948)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TaxReporting/Transactions/TaxReportingIRAQVEC(T948)/5498IRAQVECRPTREQ(T948).htm) | Tax Reporting |
| T950 | [Top Heavy Test Plan Control (T950)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/Transactions/TOPHEAVY/TOPHEAVY(T950-T951).htm) | Compliance |
| T951 | [TOP HEAVY TEST PARTICIPANT DETAIL(T951)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/Transactions/TOPHEAVY/TOPHEAVY(T950-T951).htm) | Compliance |
| T953 | [Minimum Coverage Test (T953)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/Transactions/MinimumCoverage/MINIMUMCOVERAGE(T953-T954).htm) | Compliance |
| T954 | [MINIMUM COVERAGE TEST - PARTICIPANT DETAIL (T954)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/Transactions/MinimumCoverage/MINIMUMCOVERAGE(T953-T954).htm) | Compliance |
| T955 | [Form 5500 Report (T955)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/Transactions/Form5500/Form.5500(T955).htm) | Compliance |
| T957 | [Participant Valuation Summary (T957)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/Transactions/PARTICIPANTVALUATIONSUMMARY(T957)/PARTICIPANTVALUATIONSUMMARY(T957).htm) | Funds |
| T958 | [Participant Activity Summary Report (T958)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/Transactions/PARTICIPANTACTIVITYSUMMARYREPORT(T958)/PARTICIPANTACTIVITYSUMMARYREPORT(T958).htm) | Funds |
| T960 | [Address Data (T960)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Addresses/Transactions/ADDRESSDATA(T960)/ADDRESSDATA(T960).htm) | Addresses |
| T963 | [Formatted Forecast Report (T963)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Forecasting/Transactions/FORMATTEDFORECASTREPORT(T963)/FORMATTEDFORECASTREPORT(T963).htm) | Forecasting |
| T965 | [Product Master Records(T965)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/Transactions/PRODUCTMASTERRECORDDUMP(T965)/PRODUCTMAINTENANCETRANSACTION(T965).htm) | Plans |
| T966 | [CUSTOM FULL PAGE REPORT GENERATOR(T966)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/OmniScript/Transactions/CUSTOMFULLPAGEREPORTGENERATOR(T966)/CUSTOMFULLPAGEREPORTGENERATOR(T966).htm) | Omni Script |
| T967 | [Forecast Dump (T967)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Forecasting/Transactions/FORECASTDUMP(T967)/FORECASTDUMP(T967).htm) | Forecasting |
| T970 | [Nondiscrimination Test - Plan Controls (T970)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/Transactions/NonDiscrimination/NonDiscrimination(T970-T971).htm) | Compliance |
| T971 | [Nondiscrimination Test Participant Detail (T971)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Compliance/Transactions/NonDiscrimination/NonDiscrimination(T970-T971).htm) | Compliance |
| T974 | [Investment Action Report Request (T974)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Investments/Transactions/INVESTMENTACTIONREPORTREQUEST(T974)/InvestmentActionReportRequest(T974).htm) | Investments |
| T975 | [FOREIGN EXCHANGE REPORT (T975)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ForeignExchangeRate/Transactions/EXCHREPORT(T975)/EXCHREPORT(T975).htm) | Foreign Exchange Rate |
| T976 | [Price/Rate Record Report Request(T976)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Prices/Transactions/PRICERATERECREPORT(T976)/PRICERATERECREPORT(T976).htm) | Prices |
| T977 | [Cash- Share Activity Report (T977)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/ParticipantShares/Transactions/CASH-SHAREACTIVITYREPORT(T977)/CASH-SHAREACTIVITYREPORT(T977).htm) | Participant Shares |
| T978 | [Activity History Listing(T978)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/History/Transactions/ACTHISTLISTING(T978)/ACTHISTLISTING(T978).htm) | History |
| T979 | [Pending Activity Report(T979)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/History/Transactions/PENDINGACTREPORT(T979)/PENDINGACTIVITYREPORT(T979).htm) | History |
| T981 | [Loan Record Dump Request (T981)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/LoanRecordDumpRequest(T981)/LoanRecordDumpRequest(T981).htm) | Loans |
| T983 | [Insurance Dump (T983)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Insurance/Transactions/InsuranceDump(T983).htm) | Insurance |
| T985 | [Participant Loan Report(T985)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/ParticipantLoanReport(T985)/ParticipantLoanReport(T985).htm) | Loans |
| T987 | [Loan Delinquency /Default (T987)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/Transactions/LoanDelinquencyDefault(T987)/LoanDelinquencyDefault(T987).htm) | Loans |
| T989 | [ACTIVITYHISTORY DISBURSEMENT RECORD DUMP (T989)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Disbursements/Transactions/DisbActHistRecDump(T989)/ACTHISTDISBRECDUMP(T989).htm) | Disbursements |
| T990 | [PLANRESUME GENERATOR (T990)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/Transactions/PLANRESUMEGENERATOR(T990)/PLANRESUMEGENERATOR(T990).htm) | Plans |
| T991 | [CASH CONTROL ACCOUNTLISTING (T991)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashControlAccountListing(T991)/CCAListing(T991).htm) | Cash |
| T992 | [CASH CONTROL HISTORY LISTING (T992)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashControlHistoryListing(T992)/CashHistoryListing(T992).htm) | Cash |
| T993 | [Cash Report Transaction(T993)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashReportTransaction(T993)/CashReportTran(T993).htm) | Cash |
| T994 | [Cash Online Activity History Report Transaction (T994)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/CashOnlineActivityHistRpt(T994)/CashOnlineHistRpt(T994).htm) | Cash |
| T995 | [MASTERFILE VALIDATION (T995)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Plans/Transactions/MASTERFILEVALIDATION(T995)/MASTERFILEVALIDATION2(T995).htm) | Plans |
| T996 | [EXPENSE CASH REPORTTRANSACTION (T996)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/Transactions/ExpenseCashReportTran(T996)/ExpenseRptTran(T996).htm) | Cash |
| T997 | [Annuity Master Dump(T997)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/ANNUITYMASTERDUMP(T997)/ANNUITYMASTERDUMP(T997).htm) | Annuities |
| T998 | [Annuity Master Formatted Report (T998)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/Transactions/ANNUITYMASTERFORMATTEDREPORT(T998)/ANNUITYMASTERFORMATTEDREPORT(T998).htm) | Annuities |

**6.2 Activity Codes**

|  |  |
| --- | --- |
| **Activity** | **Description** |
| 001 | Contribution Received |
| 002 | Cash Earnings |
| 003 | Earn Gain/Loss |
| 004 | Dividends |
| 005 | Forfeiture Credit |
| 006 | Miscellaneous Receipts |
| 007 | Transfer In |
| 008 | Conversion Credit |
| 009 | Other Credit |
| 010 | Stock Split |
| 011 | Contract Override Credit |
| 012 | Plan Transfer In |
| 013 | Reserved Cash/Shares |
| 014 | Shares Purchased: Traded |
| 015 | Shares Purchased: Settled |
| 016 | Shares Purchased Traded/Settled |
| 017 | Shares Sold: Traded |
| 018 | Shares Sold: Settled |
| 019 | Shares Sold: Traded/Settled |
| 020 | Installment Payment |
| 021 | Transfer Out |
| 022 | Forfeiture Debit |
| 023 | Termination Distribution |
| 024 | Miscellaneous Debit |
| 025 | Withdrawal Distribution |
| 026 | Fees |
| 027 | Conversion Debit |
| 028 | Other Debit |
| 029 | Loan Issue |
| 030 | Insurance Premium |
| 031 | Stock Purchase |
| 032 | Stock Sale |
| 033 | Cost Adjustment (Federal and State) |
| 034 | Loan Default Vesting Adjustment |
| 035 | Loan Repayment |
| 036 | Loan Fund Repayment Allocation |

|  |  |
| --- | --- |
| 037 | Plan Dividend Information |
| 038 | Loan Default |
| 039 | Loan Balance |
| 040 | Allocation Percent |
| 041 | File Maintenance User Defined Fields |
| 042 | Related Participant # |
| 043 | Excess Deferral Contribution |
| 044 | Loan Arrears |
| 045 | Loan Re-amortization |
| 046 | Forfeiture Reserve In |
| 047 | Miscellaneous Reserve In |
| 048 | Earn Reserve In |
| 049 | Contribution Reserve In |
| 050 | Fees Reserve In |
| 051 | Forfeiture Reserve Out |
| 052 | Miscellaneous Reserve Out |
| 053 | Earn Reserve Out |
| 054 | Contribution Reserve Out |
| 055 | Fees Reserve Out |
| 056 | Earnings Base Save |
| 057 | Sell Request Pending |
| 058 | Buy Request Pending |
| 059 | Plan Transfer Out |
| 061 | Stock Distribution Contribution |
| 062 | Gross Adjustment |
| 063 | Pre87 EOY Balance |
| 064 | Stock Distribution Earnings |
| 065 | Stock Distribution Forfeiture Credit |
| 066 | Stock Distribution Miscellaneous Credit |
| 067 | Stock Distribution Transfer Credit |
| 068 | Stock Distribution Conversion Credit |
| 069 | Stock Distribution Other Credit |
| 070 | Pre87 Minimum Distribution Exclusion |
| 071 | Distribution of Insurance Inkind |
| 072 | Distribution of Insurance Death Benefit |
| 073 | Other Fees |
| 074 | Loan Reversal Indicative |
| 075 | Loan Re-amortization(2) |
| 076 | Pending Adjustment Debit |

|  |  |
| --- | --- |
| 077 | Pending Adjustment Credit |
| 078 | Loan Re-amortization(3) |
| 079 | Loan Re-amortization(4) |
| 080 | Stock Distribution Installment |
| 081 | Stock Distribution Transfer Debit |
| 082 | Stock Distribution Forfeiture Debit |
| 083 | Stock Distribution Termination |
| 084 | Stock Distribution Miscellaneous Debit |
| 085 | Stock Distribution Withdrawal |
| 086 | Loan Quote Change |
| 087 | Stock Distribution Conversion Debit |
| 088 | Stock Distribution Other Debit |
| 089 | Distribution Unit to Shares Conversion |
| 090 | Extended PDF Units Sold |
| 091 | Extended PDF Units Distributed |
| 092 | Post Interest |
| 093 | Stock Distribution Annuity |
| 094 | Fund Balance Save |
| 095 | Extended PDF Purchase |
| 096 | Plan Loan Issue |
| 097 | Plan Loan Repayment |
| 098 | Plan Loan Default |
| 099 | Memo |
| 100 | Plan Year End |
| 101 | Loan Payment Interest |
| 103 | Loan Accumulated Interest |
| 104 | Deemed Loan Offset |
| 105 | Process Payment |
| 106 | Deemed Loan Prior |
| 119 | Memo Shares Sold |
| 120 | Cash Bill Post |
| 130 | Installment Setup |
| 140 | Event Process |
| 141 | Dben Death |
| 142 | Dben Annuity |
| 143 | Dben Refund |
| 144 | Dben Minimum Distribution |
| 150 | Annuity Payment (OmniAnnuity™) |
| 151 | Annuity Add (OmniAnnuity™) |

|  |  |
| --- | --- |
| 152 | Annuity Pretax (Omni Annuity™) |
| 153 | Annuity After tax (Omni Annuity™) |
| 154 | Annuity Employer (Omni Annuity™) |
| 155 | Annuity MRD Pre87 (Omni Annuity™) |
| 160 | Minimum Required Distribution Adjustment |
| 161 | Voluntary Deduction Amount |
| 190 | Il Shares Sold |
| 191 | Il Shares Distributed |
| 192 | Il Ex Shares Sold |
| 194 | Il Shares Purchased |
| 195 | Il Shares Received |
| 196 | Il Ex Shares Purchased |
| 201 | Earnings Adjustment Reversal |
| 202 | Adjust Earnings |
| 299 | Contract UDF Date |
| 407 | Pended Transfer In |
| 900 | User Defined |

## 6.3 Transaction Trailers

Each transaction in Omni Plus can stand alone as a business task. Omni Plus also allows functionality from other transactions to be merged into related transactions to form business events.  [Transaction Trailers](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/SystemCrossReferences/TransactionTrailers.htm) are used to merge functionality from multiple transactions into a base transaction to perform an entire business event. The base transaction becomes the business event, and all history generated from the base transaction and Transaction Trailers are stored together.

[Transaction Trailers](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/SystemCrossReferences/TransactionTrailers.htm) are appended at the end of the base transaction, and identified as “Screen ID”. The Screen ID is a 4 character acronym that identifies a Transaction Trailer, and it is preceded with a $ (Ex. $INST is the Installment Trailer). The following is an example of an [Inter-Participant Transfer (T366)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Transfers/Transactions/Inter-ParticipantTransfer(T366)/Inter-ParticipantTransfer(T366).htm) where the [Participant Enrollment Trailer (PTEN)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Participants/TransactionTrailers/PARTICIPANTENROLLMENTTRAILERCARD(PTEN)/PARTICIPANTENROLLMENTTRAILERCARD(PTEN).htm) has been used to establish a new participant to transfer to.

00200 $TIHDR XFT366

00201 DEFINE TRIAL.T366

1. $TIHDR XFT366000000016 20050419
2. 10 20050401

36602

1. 111223333 \*\*\* 2 10000
2. $PTEN0
3. $PTEN1 011Name Examples
4. $PTEN1 0301
5. $PTEN1 289101 Test Street
6. $PTEN1 293Test City
7. $PTEN1 294AL
8. $PTEN1 29535244
9. $PTEN1 05019500101>

List of [Transaction Trailers](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/SystemCrossReferences/TransactionTrailers.htm) is given below:

|  |  |  |
| --- | --- | --- |
| **Trailer** | **Description** | **Object Guide** |
| ADDR | [Alternate Address (ADDR)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Addresses/TransactionTrailers/TransactionTrailers.htm) | Addresses |
| ADER | [Adjustments to Earnings Transaction Trailer](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Adjustments/TransactionTrailers/TransactionTrailers.htm)  [(ADER)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Adjustments/TransactionTrailers/TransactionTrailers.htm) | Adjustments |
| ANTR | [AnnuityTrailer (ANTR)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Annuities/TransactionTrailers/ANNUITYESTABLISHMENTTRAILERCARD(ANTR)/ANNUITYESTABLISHMENTTRAILERCARD(ANTR).htm) | Annuities |
| CALC | [CALC Trailer Card (CALC)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/TransactionTrailers/CalculatorTrailerCard(CALC)/CalculatorTrailerCard(CALC).htm) | Contributions |
| CATR | [Cash Trailer Card (CATR)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Cash/TransactionTrailers/CashTrailerCard(CATR).htm) | Cash |
| COOV | [Override Contract TransactionTrailer (COOV)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Subaccounting/TransactionTrailers/COOV-OVERRIDECONTRACTTRANSACTIONTRAILER/COOV-OVERRIDECONTRACTTRANSACTIONTRAILER.htm) | Subaccounting |
| DBDD | [Disbursement Deductions (DBDD)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Disbursements/TransactionTrailers/DISBURSEMENTDEDUCTIONS(DBDD)/DISBURSEMENTDEDUCTIONS(DBDD).htm) | Disbursements |
| DBDO | [Disbursement Deduction Override (DBDO)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Disbursements/TransactionTrailers/DISBURSEMENTDEDUCTIONOVERRIDE(DBDO)/DISBURSEMENTDEDUCTIONOVERRIDE(DBDO).htm) | Disbursements |
| DGRP | [Deduction Group (DGRP)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Disbursements/TransactionTrailers/DEDUCTIONGROUP(DGRP)/DEDUCTIONGROUP(DGRP).htm) | Disbursements |
| FEAO | [Fee Allocation Override Trailer (FEAO)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/TransactionTrailers/FEEALLOCOVERRIDETRAILER/FEAO.htm) | Fees |
| FEOV | [Fee Override Trailer (FEOV)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Fees/TransactionTrailers/FEEOVERRIDETRAILERCARD(FEOV)/FEEOVERRIDETRAILERCARD(FEOV).htm) | Fees |
| FILT | [Filter Transaction Trailer(FILT)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TransactionManagement/TransactionTrailers/FILT-FILTERTRANSACTIONTRAILER/FILT-FILTERTRANSACTIONTRAILER.htm) | Transaction Management |
| FLST | [Fund List Override (FLST)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Funds/TransactionTrailers/TransactionTrailers.htm) | Funds |
| ILOV | [Investor Lot Override Trailer (ILOV)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Subaccounting/TransactionTrailers/ILOV-INVESTORLOTOVERRIDETRAILER/ILOV-INVESTORLOTOVERRIDETRAILER.htm) | Subaccounting |
| INST | [Installment (INST)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Installments/TransactionTrailers/INSTALLMENT(INST)/INSTALLMENT(INST).htm) | Installments |
| LNHI | [Loan Highest Outstanding Balance(LNHI)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/TransactionTrailers/LOANHIGHESTOSBALANCE(LNHI)/LOANHIGHESTOSBALANCE.htm) | Loans |
| LNOV | [Loan Override Trailer (LNOV)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Loans/TransactionTrailers/LOANOVERRIDETRAILER(LNOV)/LOANOVERRIDETRAILER(LNOV).htm) | Loans |
| NOTE | [Note Transaction Trailer (NOTE)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TransactionManagement/TransactionTrailers/NOTE-NOTETRANSACTIONTRAILER/NOTE-NOTETRANSACTIONTRAILER.htm) | Transaction Management |
| OCUR | [MultipleParticipant Occurrence Trailer](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TransactionManagement/TransactionTrailers/OCUR-MULTIPLEPARTICIPANTOCCURRENCETRAILE/OCUR-OCCURRENCETRAILER.htm)  [(OCUR)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TransactionManagement/TransactionTrailers/OCUR-MULTIPLEPARTICIPANTOCCURRENCETRAILE/OCUR-OCCURRENCETRAILER.htm) | Transaction Management |
| PTAI | [Participant Associated Individuals Trailers](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/AssociatedIndividual/TransactionTrailers/PARTASSOIND(PTAI).htm)  [(PTAI)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/AssociatedIndividual/TransactionTrailers/PARTASSOIND(PTAI).htm) | Associated Individual |
| PTAL | [Participant Allocation Percentages Trailer](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/TransactionTrailers/ParticipantAllocationPercentagesTrailerCard(PTAL)/PARTICIPANTALLOCATIONPERCENTAGESTRAILERC.htm)  [Card (PTAL)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Contributions/TransactionTrailers/ParticipantAllocationPercentagesTrailerCard(PTAL)/PARTICIPANTALLOCATIONPERCENTAGESTRAILERC.htm) | Contributions |
| ROLL INTERNAL | [Rollover-Internal (ROLL INTERNAL)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/TransactionTrailers/ROLLOVER-INTERNAL(ROLLINTERNAL)/ROLLOVER-INTERNAL(ROLLINTERNAL).htm) | Distributions |
| ROLL INV | [Rollover-Investments (ROLL INV)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/TransactionTrailers/ROLLOVER-INVESTMENTS(ROLLINV)/ROLLOVER-INVESTMENTS(ROLLINV).htm) | Distributions |
| ROLL MAIN | [Rollover-Main (ROLL MAIN)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/TransactionTrailers/ROLLOVER-MAIN(ROLLMAIN)/ROLLOVER-MAIN(ROLLMAIN).htm) | Distributions |
| ROLL PAYEE | [Rollover-Payee (ROLL PAYEE)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/TransactionTrailers/ROLLOVER-PAYEE(ROLLPAYEE)/ROLLOVER-PAYEE(ROLLPAYEE).htm) | Distributions |
| ROLL WIRE | [Rollover-Wire (ROLL WIRE)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Distributions/TransactionTrailers/ROLLOVER-WIRE(ROLLWIRE)/ROLLOVER-WIRE(ROLLWIRE).htm) | Distributions |
| SVTR | [Service Trailer (SVTR)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Service/TransactionTrailers/SERVICETRAILER(SVTR).htm) | Service |
| TLOV | [Tax Lot Override Trailer (TLOV)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Subaccounting/TransactionTrailers/TLOV-TAXLOTOVERRIDETRAILER/TLOV-TAXLOTOVERRIDETRAILER.htm) | Subaccounting |
| VEOV | [Vesting Override Trailer Card (VEOV)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Vesting/TransactionTrailers/VESTINGOVERRIDETRAILERCARD(VEOV)/VESTINGOVERRIDETRAILERCARD(VEOV).htm) | Vesting |
| VTD | [VTRAN Detail Trailer Cards (VTD)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TransactionManagement/TransactionTrailers/VTRANDETAILTRAILERCARDS/VTRANDETAILTRAILERCARDS.htm) | Transaction Management |
| WHLD | [Withholding (WHLD)](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/Withholding/TransactionTrailers/WITHHOLDING(WHLD).htm) | Withholding |
| N/A | [Properties Transaction Tab](file://ctsincalsfna/ING_Omni_Win_WA/02%20KT%20Phase/03%20Understanding%20Documents/05%20Onsite%20Supplied/PL550_Kaushik/ObjectLevelGuides/TransactionManagement/TransactionTrailers/PROPERTIESTRANSACTIONTAB/PROPERTIESTRANSACTIONTAB.htm) | Transaction Management |

**6.4 Major Objects & Record types**

|  |  |  |
| --- | --- | --- |
| **Object Guide** | **Function Name** | **Object / Record Type** |
| Addresses | AAAAOBJ | Alternate Address |
| Annuities | ANAMOBJ | Annuity (Annuity Master) |
| Annuities | ANASOBJ | Annuity (Annuity History) |
| Associated Individuals | PTAIOBJ | Participant (Associated Individual) |
| Base | BAI2OBJ | Alternate Inquiry |
| Base Text | BABTOBJ | Base Text |
| Base Text | BAUDOBJ | Base Text (User Defined Base Text) |
| Base Text | UTLOOBJ | Utility (Log Base Text) |
| Cash | CACAOBJ | Cash (Cash Account) |
| Cash | CACHOBJ | Cash (Cash History) |
| Checks | CKCDOBJ | Check (Check Detail) |
| Checks | CKCKOBJ | Check (Check Header) |
| Compensation | CMCMOBJ | Compensation |
| Compliance | PLCOOBJ | Plan (Compliance) |
| Compliance | TALMOBJ | Tax and Compliance Limits |
| Disbursements | DBDBOBJ | Disbursements |
| Distributions | DIAPOBJ | Distributions (Alternate Payee) |
| Distributions | DIMDOBJ | Distributions (Minimum Distributions) |
| Distributions | DISGOBJ | Distributions (Source Group) |
| DivSub | DSDSOBJ | Division/Subsidiary |
| Forecasting | FRFAOBJ | Forecasting (Forecasting Projection Annuity) |
| Forecasting | FRFBOBJ | Forecasting (Forecasting Beneficiary) |
| Forecasting | FRFPOBJ | Forecasting (Forecasting Projection) |
| Forecasting | FRFROBJ | Forecasting (Forecasting Header) |
| Foreign Exchange Rate | HIFXOBJ | Foreign Exchange History |
| Foreign Exchange Rate | XRXROBJ | Exchange Rate |
| Funds | FNFCOBJ | Fund (Fund Activity and Fund Control) |
| Funds | FNTRAN | Transaction Fund Amounts |

|  |  |  |
| --- | --- | --- |
| Funds | PTPFOBJ | Fund (Participant Fund) |
| History | BAFMOBJ | File Maintenance |
| History | HIBROBJ | History (Base Record) |
| Installments | ININOBJ | Installments |
| Insurance | PIEVOBJ | Insurance (Participant Event) |
| Insurance | PIPDOBJ | Insurance (Participant Detail) |
| Investments | IVIAOBJ | Investment (Investment Action) |
| Investments | IVICOBJ | Investment |
| Loans | LNLFOBJ | Loans (Loan Fund) |
| Loans | LNLHOBJ | Loans (Loan Header) |
| Loans | LNLPOBJ | Loans (Loan Payment) |
| Messages | MSMSOBJ | Messages |
| Notes | NTNDOBJ | Notes (Note Detail) |
| Notes | NTNHOBJ | Notes (Note Header) |
| Notes | NTOBJ | Notes (Header and Detail) |
| OmniDBEN \* | CMSLOBJ | OmniDBEN Compensation (Salary) |
| OmniDBEN \* | DNBEOBJ | OmniDBEN (Beneficiary) |
| OmniDBEN \* | DNCROBJ | OmniDBEN (Computation Results) |
| OmniDBEN \* | DNIROBJ | OmniDBEN (Interest Rate) |
| OmniDBEN \* | DNPBOBJ | OmniDBEN (Prior Benefit) |
| OmniDBEN \* | DNPNOBJ | OmniDBEN (Participant) |
| OmniDBEN \* | SVSTOBJ | Service (OmniDBEN Service History) |
| OmniExport \* | HISHOBJ | History (Sequential History File) |
| OmniTrade \* | SSSAOBJ | OmniTrade (Share Account) |
| OmniTrade \* | SSSHOBJ | OmniTrade (Share History) |
| OmniTrade \* | SSSROBJ | OmniTrade (Share Request) |
| OmniTrade \* | TRTHOBJ | OmniTrade (Trade History) |
| OmniTrade \* | TRTOOBJ | OmniTrade (Trade Order) |
| Participants | PTAFOBJ | Participant (Annual Financial) |
| Participants | PTPHOBJ | Participant (Participant Header) |
| Persons | PEPEOBJ | Person |
| Plans | PLPLOBJ | Plan |
| Plans | PMPMOBJ | Product Master |
| Prices | PRDROBJ | Price (Daily Rate) |
| Prices | PRPROBJ | Price (Price Header) |
| Rate of Return | RRRROBJ | Rate of Return |
| Report Management | RPRHOBJ | Reports (UCOM Report Header) |
| Report Management | RPRPOBJ | Reports (UCOM Report Page) |
| Service | SVSVOBJ | Service |
| Sources | PTPSOBJ | Source (Participant Source) |
| Sources | SOSCOBJ | Source (Source Control) |
| Sub accounting | SAEPOBJ | Sub-Accounting (GIC Processing) |
| Sub accounting | SAGIOBJ | Sub-Accounting (GIC Base Text) |
| Sub accounting | SAILOBJ |  |
| SubPlan | SPSPOBJ | SubPlan |
| Textfiles | TXOBJ | Textfile (Headers and Records) |
| Textfiles | TXTFOBJ | Textfile (Textfile Header) |
| Textfiles | TXTXOBJ | Textfile (Textfile Record) |
| Transaction  Management | VTTCOBJ | Transactions (Common Transaction Data) |
| Transaction  Management | VTTDOBJ | Transactions (Transaction Detail) |
| Transaction  Management | VTTHOBJ | Transactions (Transaction Header) |
| Transaction  Management | VTTSOBJ | Transactions (Transaction Submit) |
| Voice Response | PTVROBJ | Voice Response |

**7.0 OmniPlus Batch Processing**

## 7.1 Batch Process Overview

The OmniPlus internal batch processing can be subdivided into 3 categories:

* The **EDIT (\*\*\*\*\*E)** programs edit incoming 80 byte transactions, converting accepted ones to internal format.

* The **UPDATE/REPORT EXTRACT (\*\*\*\*\*T)** programs input the internal format transactions and process against the master files, performing updates and report extraction.

* The **REPORT FORMATTER (\*\*\*\*\*R)** programs input report strings and format printed output.

* + OmniPlus Edit Module Flow

**BA400F**

**BA400E**

**BA400CCTL**

**VT002E**

**. . .**

**PL995E**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **UTIN80IO** |  | **PLPLIO** |  | **BATCOUT** |  | **RPRSTO** |

* + BA400F - Edit File Controller

BA400F is the high-level controller module for the Batch Transaction Edit Processing to centralize the basic functions required to process an OmniPlus transaction into high level common modules.

BA400F's primary functions include:

* Opening/closing the TRIN, TROT, RSTROUT files
* Detecting and processing PLAN breaks
* Reading and Tabling input transactions
* Invoking BA400E to supply the current records to each currently active transaction
* Reduces the code of each transaction module; transactions are edited for correctness before being passed to the transaction processors.
* Allows multiple transactions to be edited with one pass of the file; all current transactions tabled by ECTL, and the various currently active transactions area automatically invoked for each editor.
* Increases module standardization.
  + BA400E - Edit Module Processing Controller

This is the high level control module for the batch transaction edit process to supply the OmniPlus current records to each currently active transaction. Transaction Edit Processors are modules which edits the 80 column entry format input transactions (TIXXX) into the internal format output transactions (TCXXX). There is a user exit copy member used with this module which allows access to edited transactions.

BA400E performs the following functions for the Transaction Edit Processors (\*\*\*\*\*E modules):

* Inputs and tables the trans cards into the EDU area
* Checks for and rejects duplicate input cards
* Checks for and rejects cards with sequence errors
* Writes out valid, accepted transactions to the output file
* Reports the edited trans on the edit log
* Controls the sequence of edit module processing.
* Handles END-OF-FILE and plan breaks
* Calls BA400CCTL to invoke the proper Edit Processor
  + BA400CCTL - Edit Module Call Controller

BA400CCTL - Edit Module Call Controller - this module executes the calls to the various OmniPlus transaction edit modules, based on the current transaction code. When new transaction edit modules are added to OmniPlus, they must be added to the call list of this module.

* + OmniPlus Transaction Processor Module Flow

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **BAENV** | | **. . .**  **. . .**  **BAUPDT**  **BAREPT**  **BARGEN**  **BARGENTC**  **BARGENU**  **BARGENR**  **VT002T**  **PL899T**  **PT901T**  **PL995T** | | | | | | | |
| **\*\*IO**  **(IO MODULES)** |  | **E501**  **(SECONDARY**  **PROCESSES)** |  | **\*\*\*\*GT/ST**  **(GET/STORE**  **MODULES)** |  | **E900**  **(UTILITY)** | |

* + BARGEN - Transaction Process File Controller

BARGEN - this is the high-level controller module designed to centralize the common functions required to process an OmniPlus transaction into high level common modules. The trans processors are all named OOXXXT, where OO is the object and XXX is the 3 byte OmniPlus transaction code. It reads and tables the master file (PLAN, FUND, PART) records as necessary to process the current active transactions, then invokes the transaction table control module to call the current active processes.

* The transaction and master file records are set up for the transaction automatically.
* Allows multiple processes to execute with one pass of the files.
* Allows several concurrent and/or sequential processes to be applied against a participant's records with only one set of accesses to the master files.

* Reduces JCL required to process OmniPlus. Many processes may execute under one control program, allowing OmniPlus to execute various processes, instead of having separate jobs or job steps. New processes may be easily added without impact on existing JCL.
* Increases module standardization. Since the OmniPlus transaction processing system has several basic execution control options and procedures, transaction processors written under it will have many things in common.

BARGEN performs the following for the \*\*\*\*\*T modules:

* Reads the transactions to be processed.
* Opens the TRAN, RSTROUT, and MASTER files.
* Reads the appropriate plan, part, and fund records and supplies them matched up with the transaction.
* Updates the master files as requested.
* Synchronizing master file records with proc. transactions.
* Shares the master file records among active transactions, schedules the transactions properly against the proper records at the proper time.
* Invoking BARGENTC to supply records to active trans.

* + BARGENTC - Transaction Table Process Controller

BARGENTC - handle the trans table for update/report extract processing. Its basic functions include:

* Allocates transaction table entries for new transactions.
* Removes transaction table entries for inactive trans.
* Calls BARGENU and BARGENR to invoke the processes at the proper times.
* Generates 'PARTIC. NOT ON FILE' messages
* Notifies BARGENTC when a full pass of the PART file is required.
* Calls BARGENSY and BARGENFL to properly schedule transactions and insure currently executing transactions do not conflict.

* + BARGENR - Report Transaction Call Controller

BARGEN - this module contains the IF-CALL statements which inspect the processing transaction code and invoke the proper OmniPlus report/validation/roll totals processors. New report transactions may be added to OmniPlus simply by adding them to the call list in this program. The program sets RGEN-TRAN-USED to a 'Y' if the parameter transaction was used.

* + OmniPlus Report Formatter Module Flow

**BA700F**

**BA700R**

**BA02R**

**PL990R**

**. . .**

**RPPRTIO**

**RPUCIO**

**RPRSTI**

**REPORTS**

**UCOM**

**REPORT**

**STRINGS**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **RPRSTI RPRSTO** |  | **E701**  **(SECONDARY PROCESSES)** |  | **E900**  **(UTILITIES)** | | |  | | --- | | **\*\*\*\*GT/ST**  **(GET/STORE MODULES** | |

* + BA700F - Report Writer Root Process Controller

BA700F is the root program for all \*\*\*\*R series OmniPlus report formatters, including the VUCOM subsystem. It initializes processing and calls the proper report program based upon the report string type.

* It contains an IF-CALL sequence to call the proper report string processor based on the parameter report sequence.
* Any new report Formatter modules will have a call added to this module.
* OmniPlus report formatters are modules which process the internal report strings output by the transaction processors and produce the printout reports themselves.

BA700F is the call control program for the report writer series. The report formatters are all named OOXXXR, where XXX is the 3 byte OmniPlus report code (00-99 for activity reports, 900999 for requested reports via transaction). It contains an IF-CALL sequence to call the proper report string processor based on the parameter report sequence. Any new report Formatter modules will have a call added to this module.

* + \*\*\*\*\*R Series - Report Formatters

Report Formatters mainly performs the following:

* Opens the plan, RSTRIN, subsidiary, and print files
* Reads the initial report string
* Relieve the transaction processors from the burden of formatting reports
* Allow common reports combining data from several different processes
* Allow report format to be changed independent of the generating process
* Allows external examination of internal OmniPlus processings (custom reports, interfaces, etc.)
* Invokes the proper module to process the report string

## 7.2 Unified Processing

* Overview
  + UNIFIED Processing is the primary update/reporting process for Omni Plus and provides:
    - Multiple logical runs per job i.e. multiple transactions for a participant which might normally conflict may be processed in one cycle by using multiple steps.
    - Transaction sequencing controlled through use of steps: PF, PT, M1, M2, transactions run in one step will complete before those run in the next.
    - Separate transaction reports on UCOM, ordered by plan and step.

* + Each unified job points to different environmental cards. These text files are used to modify Omni Plus automated processes.
    - Get Active controls what action will be performed upon transactions in which step.
    - Another example is whether or not to perform the Omni Plus automatic backup of a plan if any of the transactions are financial.

* + Four processes are provided, each essentially another execution of the same program running in this order :

* + - PF - Transaction processing with file backup\*.
    - PT - Transaction processing with file backup\*.
    - M1 - Transaction processing with file backup\*.  M2 - Transaction processing with file backup\*.

**Note:** While Omni Plus defaults to automatically make a backup, the use of an environmental card as noted may shut off this option if desired.

* + Run Frequency: Daily and Nightly.

* + UNIFIEDs run on preset schedules throughout the day (09:00, 11:30, 14:30, 16:30, 18:30 and the nightly cycles.

* + The Unified cycles contain numerous types :

* + - Daily examples: Posting UNIFIEDs ( 11:30, 14:30 and 18:30) Edit-only, Zero par

* + - While the nightly cycle contains Post, Zeronite/Enroll, Prices and SDBO UNIFIEDs.

* + Four datasets are extracted from VTRAN to populate the CARDS. files.

* + At the end of each procedure, the CARDS.files are reinitialized in preparation for the next run. If the job abnormally ends, the .CARDS files may be examined to determine which transactions were being processed. The contents of all of the .CARDS files are combined and copied into an xx.SAVINPUT(+1) GDG dataset for recovery or audit purposes.

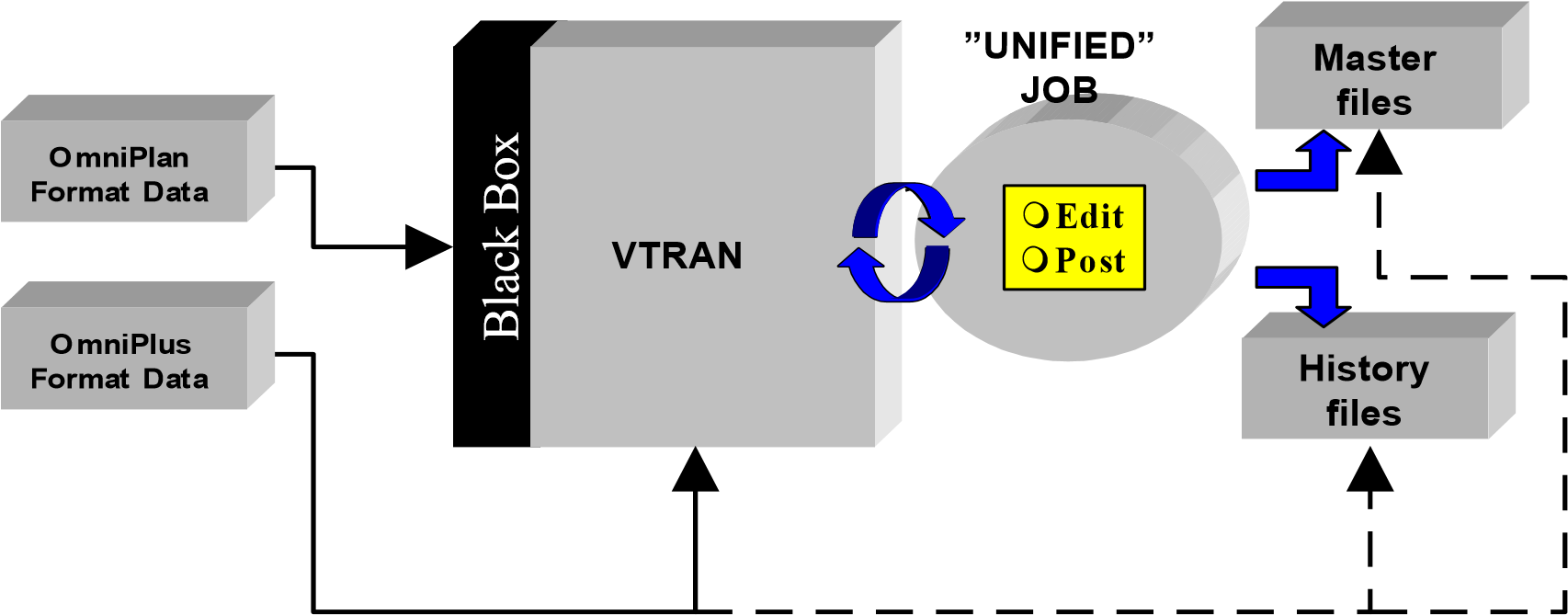
* + The NEXTTI feature allows users to generate transactions via the T966 transaction to be automatically input into the next process. The BA700F step includes a //NEXTTI DD which MODS onto the next process's .CARDS dataset.

* + The LOADVT feature allows users to generate transactions via the T844 transaction to be automatically input into either the next process via the NEXTTI feature or output to the .VTCARDS dataset to be processed later. The BA700F step includes a //LOADVT DD which MODS onto the .VT.CARDS dataset.

* + Output of the Unified process consists of :

* + - Updates to the master files.
    - Interface files for “downstream” applications.  Edit logs listing input transactions.
    - Activity logs showing summarized results of activity.
    - UCOM Reports.

* Process Flow Diagram



* UNIFIED Job Step Descriptions

The unified job primarily consists of the following steps:

* + - **SE400** - Sort Input Transactions. This step sorts the M1.CARDS dataset into the proper processing sequence for the edit.

* + - **PE400** - Financial Transaction Edit

* + - **SUPDT** - Sort Transactions for BAUPDT. This step sorts the transactions which were output from BA400F into the proper edit sequence.

* + - **PBKUP** - Pre-Backup of Master Files. Optional, installation dependent. If this step is included, backup will be taken if users enter financial transactions. Only plans with financial updates will be backed up.

* + - **PUPDT** - Update Transaction Processor

* + - **SREPT -** Sort Transactions for BAREPT. This step sorts the transaction file output from BAUPDT into the proper sequence for BAREPT.

* + - **PREPT** - Report Transaction Processor

* + - **DWUTIL** - Deferred Workfile Utility

* + - **SE700** - Report String Sort

* + - **IDE700** - Payment Interface Allocate

* + - **PE700** - Standard Report Generator

* UNIFIED Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Identifier** | **Length** | **Parameter** | **Description** |
| **EEEE** | **1-4** | **&PRMEDIT** | **Edit-only parameter**  'EDIT' will force all plans to run in an 'edit only' mode |
| **S** | **5** | **&PRMSEQ** | **Override for production of sequential AHST file** Legal Values:  ‘ ‘ - Sequential AHST file will be created only if a plan is running in 'post' mode  ‘1’ - The sequential AHST file will always be created. (Even if a plan is running in 'edit only' mode .)  ‘2’ - The sequential AHST file will never be created (Even if a plan is running in 'post' mode.) |
| **DDDDDD** | **6-11** | **&PRMRDATE** | **Run date override:YYMMDD format**  If input this will be used as the run date on all activity history records written. |
| **TTTT** | **12-15** | **&PRMRTIM** | **Run time override:HHMM format**  If input, this will be used as the run time on all activity history records written. |
| **R** | **16** | **&PRMRPT** | **Plan message suppression** |
| **M** | **17** | **&PRMMSG** | **Pass message suppression** |

# Omniscript Overview

Omniscript is a scripting language that is essentially used to perform the following functions for

the Retirement Services record keeping system viz Omniplus.

* Creation of custom reports & extraction of data at a record level, thereby easing the effort involved in data mining.
* Update data at a record level, thereby easing the effort involved in data setup.
* Creating transactions using the various Omniscript functions, thereby easing the effort involved in transaction creation.
* Processing of custom edits & validation at a plan and participant level, thereby adhering to the various business rules.
* Generation of formatted XML or HTML web pages.

Prior to Omniscript, Omnicalculator (Omnicalc) was used to perform the afore mentioned functions. Here’s the relationship between Omniscript & Omnicalc,

**Omniscript vs Omnicalc**

Think of it this way: OmniPlus now processes two scripting languages, the old language, which was referred to as “Calculator”, and the new language, Omniscript. (NOTE: In the System documentation, the entire scripting package, old code and new code, is referred to as Omniscript). These two languages may be used concurrently within the same text file BUT are independent of each other in terms of record access. This means that the parameters or records being examined by one language do not have any effect on the parameters or records being examined by the other. However, either language may be used at any time in a Text File.

This section attempts to provide some insight into the question on everyone’s mind: When do I use the old code (Calculator) versus the new code (Omniscript)? You will probably find a range of different answers depending on who you ask. Probably the best answer is that in time, as you develop your knowledge of Omniscript, you’ll be able to answer that question yourself.

However, there are at least 2 instances where it is probably better to use Calculator over Omniscript. These are listed below:

1. If you are generating a report which possibly may produce more than 1000 lines PER PARTICIPANT, you should use Calculator because Report Writer Integration has a limit of 1000 lines per participant. In this instance, the term “PER PARTICIPANT” means the pointer that is controlled by the system (or T966), not the one controlled by Omniscript.
2. In Omniscript, there is no equivalent of PA (as-of PF) functionality. If you need to use PA’s, you need to do so using the old code instead of the new. Also, Omnicalc will come in handy for performing complex calculations such as the Dividends / Earnings calculations.

Following are the essentials to build Omniscripts,

* Omniscript Variables / System Defined Functions
* Conditional, Looping Statements & Subroutines
* Omni Object Access Functions
* Omniscript Operators
* Omniscript Utility Functions

# Omniscript Structure

Following are the various sections available within an Omniscript program,

* PARM Section
* TEXT Section
* CALC Section

**PARM Section**

Parameter section is one of the Report Writer function that is included in an Omniscript especially if written directly in Omnistation, to specify the reporting feature of the output. Following are the Parm functions,

| **DE #** | **FIELD NAME** | **Values** | **Description** |
| --- | --- | --- | --- |
| N/A | OR\_RPT\_MODE | CALC  T966 | Report Writer Integration method (explicit multiple generations) SORT- Traditional method (implicit single generation) This command is REQUIRED if using Report Writer Integration. |
| N/A | OR\_DELIMITER |  | Changes TEXT SECTION delimiter from “\_” to value specified. |
| N/A | OR\_AUTO\_PAGE\_BREAK | YES  NO | Auto page break is ON  Auto page break is OFF |
| N/A | OR\_FOOTER\_LENGTH |  | Tells the page breaking facility how far from the bottom of the page to produce the Footer. Must be valued unless the user is using the Automatic Page Breaking facility. If the user is using the Automatic Page Breaking Facility, the length of the footer will be automatically calculated.  Default is 5 lines. |
| N/A | OR\_PAGE\_LENGTH |  | Tells the page breaking facility how many lines are contained within one page.  This field must be set to a desired page. Default is 50 lines. |
| OR210 | OR\_PRINT\_FIRST |  | Contains Xerox commands that print before detail statements. |
| OR220 | OR\_PRINT\_SECOND |  | Contains Xerox commands that print after OR\_PRINT\_FIRST, but before detail statements. |
| OR230 | OR\_PRINT\_LAST |  | Contains Xerox commands that print after all detail statements. |

**Text Section**

Text section is used for formatting the resulting values. The user can control the output parameters & the manner in which the output needs to be displayed in the Text Section. The various functions of the Text Section are as follows,

| **DE #** | **FIELD NAME** | **Values** | **Description** |
| --- | --- | --- | --- |
| OR001 | OR\_AREA\_1 |  | Use OR001 or OR\_AREA\_1 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_1\_ON and OR\_AREA\_1\_OFF can be used in the Calc Section to explicitly turn the text block on and off. |
| OR002 | OR\_AREA\_2 |  | Use OR002 or OR\_AREA\_2 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_2\_ON and OR\_AREA\_2\_OFF can be used in the Calc Section to explicitly turn the text block on and off. |
| OR003 | OR\_AREA\_3 |  | Use OR003 or OR\_AREA\_3 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_3\_ON and OR\_AREA\_3\_OFF can be used in the Calc Section to explicitly turn the text block on and off. |
| OR004 | OR\_AREA\_4 |  | Use OR004 or OR\_AREA\_4 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_4\_ON and OR\_AREA\_4\_OFF can be used in the Calc Section to explicitly turn the text block on and off. |
| OR005 | OR\_AREA\_5 |  | Use OR005 or OR\_AREA\_5 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_5\_ON and OR\_AREA\_5\_OFF can be used in the Calc Section to explicitly turn the text block on and off. |
|  | OR\_HEADER |  | will be produced at the top of each new page. |
|  | OR\_FOOTER |  | will be produced at the bottom of each new page. |
|  | OR\_TOTALS |  | will be produced as the last page of output print. |

**CALC Section**

Calc section is where all the Omniscript functions, statements, loops are specified. Omniscript executable statements are specified in the Calc section.

Following are the functions available in the Calc section,

| **DE #** | **FIELD NAME** | **Values** | | **Description** |
| --- | --- | --- | --- | --- |
| WK213 | OR\_SORT\_RPT\_OPT |  | | Sort Sequence Filed that overrides the value entered in the SORT-REPORT-OPTION-966 field. |
| WK214 | OR\_RPT\_DATE |  | | Refers to DATE-REPORT-966 |
| WK215 | OR\_RPT\_DATE\_ALTER |  | | Refers to DATE-ALTERNATE-966 |
| WK216 | OR\_PAGE\_NUM |  | | Contains the current page number that will be printed in the Text Section. |
| WK217 | OR\_PART\_CANADIAN |  | | Participant Number (Canadian format 999-999-999).  Reference only. |
|  | OR\_RPT\_GEN |  | | Causes OmniScript to issue commands to Report-Writer to print all Conditional Text Areas which have been set to ‘ON’. Text Blocks which are not Conditional Text Blocks will also be printed.  All Conditional Text Areas will be reset to ‘OFF’ after this command is issued. |
|  | OR\_CURR\_LINE |  | | equal to the value of the current line number. |
|  | OR\_PAGE\_BREAK |  | | causes OmniScript to issue commands to Report-Writer that will write out the footer, perform a page break, and set up the header for the next page.  This keyword is meant to allow user control over page breaking, and is not meant to be used with the auto page breaking facility. |
|  | OR\_PAGE\_INIT |  | | Prepare to output a page.  Clear page and set sort key. |
|  | OR\_PAGE\_WRITE |  | | Output the current page.  Should be proceeded by a OR\_PAGE\_INIT and one or more OR\_RPT-GENs. |
| OR001 | OR\_AREA\_1\_ON  OR\_AREA\_1\_OFF |  | Use OR\_AREA\_1 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_1\_ON and OR\_AREA\_1\_OFF can be used in the Calc Section to explicitly turn the text block on and off. | |
| OR002 | OR\_AREA\_2\_ON  OR\_AREA\_2\_OFF |  | Use OR\_AREA\_2 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_2\_ON and OR\_AREA\_2\_OFF can be used in the Calc Section to explicitly turn the text block on and off. | |
| OR003 | OR\_AREA\_3\_ON  OR\_AREA\_3\_OFF |  | Use OR\_AREA\_3 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_3\_ON and OR\_AREA\_3\_OFF can be used in the Calc Section to explicitly turn the text block on and off. | |
| OR004 | OR\_AREA\_4\_ON  OR\_AREA\_4\_OFF |  | Use OR\_AREA\_4 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_4\_ON and OR\_AREA\_4\_OFF can be used in the Calc Section to explicitly turn the text block on and off. | |
| OR005 | OR\_AREA\_5\_ON  OR\_AREA\_5\_OFF |  | Use OR\_AREA\_5 in the Text Section to define a text block.  If it is defined as a conditional text block, OR\_AREA\_5\_ON and OR\_AREA\_5\_OFF can be used in the Calc Section to explicitly turn the text block on and off. | |
|  | OR\_HEADER\_ON  OR\_HEADER\_OFF |  | turns the area in the Text Section on.  turns the area in the Text Section off. | |
|  | OR\_FOOTER\_ON  OR\_FOOTER\_OFF |  | turns the area in the Text Section on.  turns the area in the Text Section off | |
|  | OR\_TOTALS\_ON  OR\_TOTALS\_OFF |  | turns the area in the Text Section on.  turns the area in the Text Section off. | |
|  | OR\_AREA\_ALL\_ON  OR\_AREA\_ALL\_OFF |  | When multiple areas in the Text Section have been defined, you can use OR\_AREA\_all\_ON and OR\_AREA\_all\_OFF in the Calc Section to explicitly turn all the areas on and off. | |

# Omniscript Comments & Variables

# Omniscript Statements are not case sensitive. A combination of Upper & Lower case can be used in an Omniscript. Every Omniscript Statement should end with a semi colon (;)

**Comments**

Omniscript comments are an integral part of Omniscript programming. Comments help to understand the purpose of various program statements. It’s a good practice to include comments in every Omniscript, to ease the understandability of the script. Following are the two types of Comments used typically in an Omniscript,

* Informational Comments - > This type of Comment helps understand the purpose of an Omniscript Statement. This type of Comment starts with /\* and ends with \*/.
* Non-Executable Statement Comments -> This type of Comment is used to prevent a particular Statement from being executed. This type of Statement has a \* sign in the beginning of a Statement. Any Statement preceding the \* sign w--ill be ignored during Omniscript processing / execution.

Example: PTPHOBJ\_DE(050); / \* Participants Date of Birth \* / An example for Informational Comment.

\* TX001=WK002; An example for Non-Executable Statement.

**Variables**

Omniscript variables are similar to variables used in any other conventional programming language used essentially for storing values. Following are the various Variables available in Omniscript,

* FD – Function-Defined fields
* IT – Indexed Text fields
* IW – Indexed Work fields
* KV – Key Value fields
* SD – System-Defined fields
* TX – Text fields
* WA – Work Accumulator fields
* WF – Work Fund fields
* WK – Work Products fields

Text Fields(TX)

Text fields can hold alphanumeric values. TX001 through TX120 are available for usage. Values stored in Text fields are limited to 200 characters. Here’s an example for Text Fields,

TX001 = PTPHOBJ\_DE(289); /\* Stores the value of Address Line 1 in TX001 \* /

There are various operations that can be performed using Text variables, which will be addressed in the Omniscript Utility Function (in the subsequent chapter(s)).

Work Fields(WK)

Work Fields are used to house / accommodate numeric values. Following are some of the characteristics of Work fields

* WKs cannot accommodate numbers greater than 99,999,999,999.999999. ‘PIC 9(11)V9(6)’.
* WKs calculation results can be used multiple times in the same program.
* OmniScript contains up to 6200 work fields (WK001 through WK6200). Up to 3000 can be used with the T966 Custom Report Generator.
* All 'WK' Work Fields are reinitialized to zero between participants so that calculations are performed for each participant (not accumulated for all participants).
* WKs can be user defined or predefined by the transaction you are processing. Most transactions set aside a block of WKs to perform certain tasks. Before creating a program, you must be familiar with the pre-defined WKs for the transaction

Here’s an example of Work Fields,

WK001 = PTPFOBJ\_NumDE(120) ; /\* Read Net Dollars and assign the value to a Work Variable \*/

Work Fields can have particular uses in transactions that use OmniScript for custom exit processing. For example, WK212 - WK217 are predefined for the T966 transaction, and therefore cannot be user-defined for that transaction.

Key Value Fields(KV)

Key value (KV) fields are used within the OmniScript facility to establish parameters and qualify the records that are to be accessed and used in OmniScript functions. Individual records can be accessed, or records can be counted and summed together. Records for selected files can be added, changed, or deleted. Caution should be exercised when using 'KV' fields because values are defined differently for different types of records. Records types that use ‘KV’ fields for OmniScript access include, BR, BT, FA, PA and DR.

Following example will illustrate you the use of KV fields

KV001=19990101;

KV002=SD003;

KV7=301;

WK1=PA200@\*\*\*;

INITKV;

/\*INITKV resets all “KV” values to zero. “KV” values can now be re-defined for new record access and retrieval \*/

In the above example KV1 is being used to assign the low trade date, and KV2 is being used to assign the high trade date. SD003 is another special variable which stores the current trade date of the system. KV7 us being assigned the transaction code. When this code runs, it will assign the sum of all the PA200 (Gross contribution) posted from T301 transaction (KV7) starting from 01/01/1999 (KV1) to current trade date (KV2/SD3).

Following is the list of key KVs,

|  |  |
| --- | --- |
| KV001 | Trade Date - Low. Low value to be compared to the trade date in BR008. |
| KV002 | Trade Date - High. High value to be compared to the trade date in BR008. |
| KV003 | Run Date - Low. Low value to be compared to the run date in BR009. |
| KV004 | Run Date - High. High value to be compared to the run date in BR009. |
| KV005 | Run Time - Low. Low value to be compared to the run time in BR010. |
| KV006 | Run Time - High. High value to be compared to the run time in BR010. |
| KV007 | Transaction Code. Value to be compared to the transaction code in BR101. |
| KV008 | Activity Type. Value to be compared to the activity type in BR102. |

There are 53 Key Values available in Omni Scripting.

System Defined(SD) Variables

Several categories of System Defined Data Elements have been established for use within OmniScript. System-defined data elements represent values automatically calculated by the system that can be used in OmniScript.

* System Defined Values can be used in any OmniScript context.
* Literal equivalent values (Aliases) may be substituted for SD data element identifiers. Available SD aliases are listed in the Literal Equivalents column of the table in Appendix –B at the end of this chapter.
* System Defined (SD) data elements consist of:

Storage areas accessed by different functions

Flags that perform various tasks

* Some common uses for SDs include, Date Conversion Facility, OmniScript Text Substring Facility

Some of the commonly used SD data elements are

1. SD001:- Stores the current plan ID. Can be assigned to a text field to return the

Plan number. Example TX001=SD001; will return the current plan no.

1. SD002:- Stores the current participant on which the code is being executed.
2. TX001=SD002;

will return the current SSN id.

3. SD003:- Stores the current system date. WK001=SD003 will return current system date.

There are bunch of other SD Data elements which return the dates in various formats. Refer to SD040 to SD043 to find out what more you can do with SD data elements.

1. SD080:- This is one of the most important variables which you will be using most commonly
2. While doing the Omni Scripting.

SD080 is MAX-LOOP-COUNTER. Overrides for maximum number of times a LOOP will execute. Default is 500. The maximum value is 999,999,999. So if you wish to run a look any number of times between a range of 500 to 999,999,999 you can assign the value to SD080 before initiating the loop. Below example illustrates the same.

SD080=1000;

PTPHOBJ\_NEXT();

LOOP WHILE PTPHOBJ\_VIEW();

WK1=PTPHOBJ\_DE(021);

IF (WK1<20);

TX1=’ACTIVE’;

ENDIF;

IF (WK1>19);

TX1=’TERM’;

ENDIF;

OR\_RPT\_GEN;

ENDLOOP;

The above piece of code will traverse through 1000 participant records, and based upon the employment status it will print on the report if the participant is ACTIVE or TERM.

Here’s a complete list of the various SD variables,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** | **Format** |
| SD001 | SDPlan | Current Plan ID | System | X06 |
| SD002 | SDPartNum | Current Participant Number | System | X17 |
| SD003 | SDRDate | [Current System Date. The date at which the current process started. This value may be overridden by the OmniPlus environment variable ENV\_RUN-DATE and related environment variables. This field is in CCYYMMDD format. Use OcDate\_Current() to retrieve the actual CPU date](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\InstallationOperation\Common\JobOverview.htm#ENVRUNDATE) | System | N08 |
| SD004 | SDRTime | [Current Process Start Time. The time at which the current process started. This value may be overridden by the OmniPlus environment variable ENV\_RUN-TIME and related environment variables.  This field is in HHMMSS format  Use OcTime\_Current() to retrieve the actual CPU Time](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\InstallationOperation\Common\JobOverview.htm#ENVRUNTIME) | System | N06 |
| SD005 | SDProcDt | Current Processing Date. The date of the currently processing transaction, or supplied via a TI002 Date card. This field is in CCYYMMDD format. | System | N08 |
| SD006 | SDNumFCs | Total Number of Fund Control Records for the current plan | System | N03 |
| SD007 | SDNumPFs | Number of Participant Fund Records for the current participant | System | N03 |
| SD008 | SDPartCtr | Total Participants processed in this pass (Unified) | System | N09 |
| SD009 | SDActive | RGEN Status Flag.  Values are: | System | X01 |
| 0 – Active |
| 1 – Inactive |
| Note: Not in Use. |
| SD010 | SDEdit | [Edit Mode Indicator.  This value corresponds with the OmniPlus environment variable ENV\_GLOBAL-MODE. Values are:](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\InstallationOperation\Common\JobOverview.htm#ENVGLOBALMODE) | System or User | X01 |
| [0 or “E” – Edit mode](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\InstallationOperation\Common\JobOverview.htm#ENVGLOBALMODE) |
| [1 or “P” – Update mode](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\InstallationOperation\Common\JobOverview.htm#ENVGLOBALMODE) |
| [This field may be updated by the OmniScript text only in the online environment.](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\InstallationOperation\Common\JobOverview.htm#ENVGLOBALMODE) |
| SD011 | SDTSeq | Current Transaction Pass-Sequence | System | X01 |
| SD012 | SDNumICs | Total Number of Investment Control Records | System | N03 |
| SD013 | SDTotPage | Total Page Flag (T966).  Values are: | System | N01 |
| 0 – Not a totals page |
| 1 – Totals page |
| SD014 | SDFMRpt | File Maintenance Reporting Option. | User | N01 |
| Values are: |
| 0 – Report only changed field values |
| 1 – Report regardless of whether field value is changed |
| 2 – Report only unchanged field values |
| 3 – Do not report any field values |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** | **Format** |
| SD017 |  | Display Log Messages Flag.  Allows users to specify where LOG messages should be routed, either to OP-02 reports (Information Log, Warning Log, Reject Log), to SYSOUT logs, or both. | User | X01 |
| “Y” – Route log messages to SYSOUT (default for Jobcalc) |
| “ “ – Route log messages to OP-02 reports (default for Unified Jobs) |
| “B” – Route log messages to both SYSOUT and OP-02 reports. |
| SD020 | SDFmt1 | RS02 Message Field Mask-1.  Indicates the print format for numeric values in OmniScript messages.  Values are: | User | X01 |
| 0 – Dollar S9(9)V99 format |
| 1 – Share S9(9)V9(4) format |
| 2 – Whole number S9(9) format |
| 3 – Price S9(7)V9(6) format |
| 4 – Date S9(6) format |
| 5 – Time S9(4) format |
| 6 – Percent S9(3)V9(4) format |
| 7 – Social security number S9(9) format |
| 8 – Count S9(9) format |
| See Also: Output Report Message Facility |
| SD021 | SDFmt2 | RS02 Message Field Mask -2.  Values are same as SD020. | User | X01 |
| SD022 | TranType | Transaction Type.  Refers to the TYPE field on a Participant Inter-Fund Transfer (T381) transaction or the TYPE field on a Withdrawal transaction (T444). | System | X02 |
| SD023 | TranType1 | Transaction Type - 1st character.  First character of SD022. | System | X01 |
| SD024 | TranType2 | Transaction Type - 2nd character.  Second character of SD022. | System | X01 |
| SD025 |  | Transaction Code.  Currently executing transaction. | System | X03 |
| SD030 | SDWKIDX | Index Work Field - Base Value. | User | N03 |
| SD031 | SDTXIDX | Index Text Field - Base Value. | User | N03 |
| SD033 | SDFund | Default Fund-ID. The current Fund-ID applicable to a fund level variable fetch or store in which no ‘@’ parameter is specified. See the ‘Fund Usage’ section. | User | X03 |
| SD035 | SDPrDate | Price Date. Can be set to Override the date used for Price fetches. This field is in CCYYMMDD format. | User | N08 |
| SD036 | SDLPart | Last Participant Flag.  Indicates last participant in master file processed.  Values are: | System | X01 |
| N – No.  Last participant not processed. |
| Y – Yes.  Last participant processed. |
| SD037 | SDAllow0Pr | Allow Zero Price Flag.  Allows market value calculation (PA999) without a price on file. | 'N' value must be set by user. | X01 |
| Values are: |
| Y – Yes.  (Default) |
| N – No.  A price is required to calculate market value (PA999). |
| SD039 |  | Transaction Date.  The date on the Currently executing Transaction, in CCYYMMDD format. | System | N08 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** | **Format** |
| SD040 | SDYYMMDD | Date Conversion Field-1.  Used for converting date formats.  This field will infer a century designation and return an eight-digit date in CCYYMMDD format. | System or user. | N08 |
| SD041 | SDDays | Date Conversion Field-2.  Used for converting date formats.  This field will return the number of days since 01/01/1900. | System or user. | N05 |
| SD042 | SDWeekday | Date Conversion Field-3 (Day of the Week format).  Used for converting date formats.  Values are: | System | N02 |
| 1 – Monday             5 – Friday |
| 2 – Tuesday            6 – Saturday |
| 3 – Wednesday       7 – Sunday |
| 4 – Thursday |
| SD043 | SDMMDDYY | Date Conversion Field-4.  Used for converting date formats.  This field will return a six-digit date in MMDDYY format. | System or user. | N06 |
| SD044 | SDJullian | Date Conversion Field-5.  Used for converting date formats.  This field will return a seven-digit Julian date in CCYYDDD format. | System or user. | N07 |
| SD045 |  | Date Conversion Field-6.  Used for converting date formats.  This field will return an eight-digit date in MMDDCCYY format. | System or user. | N08 |
| SD046 |  | Business Calendar Date Input. This field is set by the user to determine if a specified date is a business date. This date is input in CCYYMMDD format. This field is in CCYYMMDD format. | User | N08 |
| SD047 |  | Business Calendar Date Output. If SD046 is valued, this field contains the corresponding business date according to the Business Calendar textfile in CCYYMMDD format.  If SD046 is a holiday, this field will contain the previous or following business date.  This field is in CCYYMMDD format. | System defined based on the value of SD046. | N08 |
| SD050 |  | OmniScript Return Code | System | X04 |
| SD051 | SDEnvir | OmniScript Return Code | System | X08 |
| Environment Operating System (OS) |
| MVS |
| PC |
| UNIX |
| SD052 | SDReleaseID | Plus Release Identifier including ptf level  (e.g., ‘5.20.20', '5.30.01', 'DEV2’) | System | X12 |
| SD053 | SDRelDate | Plus Release Date | System | k |
| SD054 | SDOrigPgm | Originating Program (e.g., "OCRUN", "BAUPDT", "BAREPT") | System | X12 |
| SD055 | SDEditMode | Edit Mode | System | X01 |
| P – Post |
| E – Edit |
| T – Trade Advance |
| X – Edit Error |
| R – Trade Error |
| SD056 | SDProcMethod | Process Method | System | X01 |
| B – Batch |
| O – Online (CICS) |
| 1 – OmniStation/Workstation |
| C – Conversion |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** | **Format** |
| SD057 | SDCompiler | COBOL Compiler used to compile OmniPlus. | System | X08 |
| COB2 – MVS COBOL |
| MFC – Micro Focus COBOL |
| FUJ – Fujitsu NetCobol |
| SD058 |  | Unified Process Step (PT, PF, M1, M2, etc.) | System | X02 |
| SD059 |  | Job Identifier | System | X20 |
| SD060 |  | Text String 1 - Text Work Area.  Stores up to 40-character alphanumeric value that can be retrieved by providing start character and length.  Works with SD061 below. | User | X200 |
| SD061 |  | Text String 1 - Number Work Area.  The first of two fields (SD066 is the other) provided to convert numeric OmniScript values to text values.  A numeric value entered in this field is converted to a text value and placed in the first 15 bytes of SD060.  The '@' parameter is used to designate the format of the value placed in SD060.  It is expressed as SD061@00#.  Values for the '@' parameter used to define format are: | User | N15D06 |
| @001 - ZZZ,ZZZ,ZZ9.99 |
| @002 - ZZZ,ZZZ,ZZ9.999999 |
| @003 - ZZZ,ZZZ,ZZ9.999 |
| @004 - ZZZ,ZZZ,ZZ9.9999 |
| @005 - ZZZ,ZZZ,ZZ9 |
| @006 – 99B99B99 |
| SD065 |  | Text String 2 - Text Work Area.  Stores up to 40-character alpha-numeric value that can be retrieved by providing start character and length.  Works with SD066 below. | User | X200 |
| SD066 |  | Text String 2 - Number Work Area.  The second of two fields (SD061 is the other) provided to convert numeric OmniScript values to text values.  A numeric value entered in this field is converted to a text value and placed in the first 15 bytes of SD065.  See definition of SD061 above for values and correct formatting in OmniScript text. | User | N15D06 |
| SD068 |  | Text String - Start Character Position.  Numeric position of the starting character within the text string.  This field referenced if the first two bytes of the ‘@’ parameter are ‘00’. | User | N02 |
| SD069 |  | Text String - Length.  Number of characters to be referenced from the text string.  This field referenced if the third byte of the ‘@’ parameter is ‘0’. | User | N02 |
| SD070 |  | Voucher Number | System | N09 |
| SD071 |  | Folder Name.  Folder name of the file submitted for processing. | System | X22 |
| SD072 |  | User ID. User ID from either the transaction being processed or the current online user . | System | X06 |
| SD075 |  | EACH Mode Flag.  Controls records processed by the EACH@ command.  Values are: | User | X01 |
| 0 or P – Accesses PF records |
| 1 or F – Accesses FC records |
| I – Accesses IC records |
| S – Accesses SC records |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** | **Format** |
| SD080 | SDLoopMax | MAX-LOOP-COUNTER.  Overrides for maximum number of times a LOOP will execute.  Default is 500.  The maximum value is 999,999,999. | User | N09 |
| SD085 |  | [Override date for loan balances.  Overrides the processing date when calculating the highest outstanding loan balance (LH991).](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Loans\DataElements\DataElements.htm#LH991) | User | N06D00 |
| SD090 |  | Trigger Event Name.  The event name associated with an OmniScript.  (e.g., “RunStart”, “RunEnd”, “PlanStart”) | System | X20 |
| SD091 |  | Events Enabled Flag.  Indicates that base processing events have been enabled or disabled.    Y – Events are enabled.    N – Events are ignored or no events have been added. SD091 is set to “Y” when events are added and may be set to “N” to disable event processing. | System, but may be overridden by the user in the OmniScript text. | X01 |
| SD095 |  | Validate DE Flag. Forces an OmniScript compile to reject if an Object Access Function (XXYYOBJ) requests an invalid data element number. Legal Values: "Y" - Produce a compile error if an invalid data element is requested. | User | X1 |
| SD100 |  | OmniScript Function Numeric Return Code. A numeric value that may be returned by an OmniScript function call.  Legal values vary by function and operation. | System | N07 |
| SD101 |  | OmniScript Function Text Return Message.  A text value that may be returned by a OmniScript function. | System | X40 |
| SD102 |  | OmniScript Function Trace flag. “N” or blank – Do not produce trace messages (default) “Y” – Produce trace messages For more information, see OmniScript Trace Facilities. | User | X01 |
| SD104 |  | Base Event Abend Flag. Set to ‘Y’ in Base Event script to cause the system to abend. | User | X01 |
| SD105 |  | OmniScript Return Code Level.  This field can be set when an OmniScript calls an external program (i.e., user exit) or a variable ROUTINE (i.e., PERFORM tx1). 0 – OK                     3 – Error 1 – Info                     4 – Severe 2 – Warning              5 – Fatal | System | X01 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** | **Format** |
| SD106 |  | OmniScript Return Code Reason This field is set when a OmniScript calls an external program (i.e., user exit) or a variable ROUTINE (i.e., PERFORM tx1). 00 – Reason OK 05 – Missing Routine 99 – Unimplemented COBOL program, i.e., module not available | System | X02 |
| SD109 |  | OmniScript behavior may be modified when a numeric value is directly assigned to an alphanumeric field (e.g. TX001 = 1234.56) ‘ ‘ – Default, return the value in a readable format, displaying all digits, without leading zeroes, and inserting commas where necessary, e.g. “1,234.56” ‘L’ – Legacy behavior, return as a 1-11 digit integer, depending on the length of the target field, e.g. “00000001234”. Leading zeroes are used, and all decimal positions are truncated. ‘E’ – Produce a compile error if a numeric value is directly assigned to an alphanumeric field.  If this value is set, OcFmt must be used to format the numeric field before assigning it to an alphanumeric field. This environment value may be used to override OC\_NTOX. | User | X01 |
| SD120 |  | Number of memory gets done | System | N13 |
| SD121 |  | Number of memory frees done | System | N13 |
| SD122 |  | Total allocated memory.  This field maintains a running total of memory allocated during a process.  It does not return the amount of memory that is currently allocated. | System | N13 |
| SD123 |  | Currently allocated segments (SD120-SD121).  This field tracks the number memory gets and subtracts out the number of memory frees. | System | N13 |
| SD125 |  | Trace Statements Seconds - Set to a number of seconds. Any Script instruction which takes longer will be shown on SYSOUT. Set to .01 to request that all statements traced and 0 to have no statements traced. | User | N07D06 |
| SD126 |  | Trace Limits -  Provides a maximum number of lines for which the SD125 DISPLAY will be done.  Limits output to a manageable size. | User | N09 |
| SD127 |  | Trace Statements - Number of instructions to trace the execution of via Sysout messages. | User | N09 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** | **Format** |
| SD127 |  | Trace Statements - Number of instructions to trace the execution of via Sysout messages. | User | N09 |
| SD200 |  | Current Transaction Active Flag. SD200 is the currently executing transaction active flag. It is set to 'Y' by the transaction processor at StartTran, and must be 'Y' after StartTran for the transaction to continue to process. SD200 can be set to 'N' to stop the current transaction from further processing. For example, setting it to 'N' in the middle of a plan level pass will cause the transaction to end without processing any more participants. | System, but may be overridden by the user in the OmniScript text. | X01 |
| SD201 |  | [Current Transaction Code (VTD060)](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\DataElements\DataElements.htm#VTD060) | System | X03 |
| SD202 |  | [Current Folder Name (VTD050)](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\DataElements\DataElements.htm#VTD050) | System | X30 |
| SD203 |  | [Current Transaction Participant ID (VTD070) Note:  This data element is not always the same as SD002, since the SD203 may be 000000000 for plan level transactions](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\DataElements\DataElements.htm#VTD070) | System | X17 |
| SD204 |  | [Current Transaction Sequence Number (VTD090)](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\DataElements\DataElements.htm#VTD090) | System | N07 |
| SD205 |  | [Current Transaction Plan Num (VTD005)](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\DataElements\DataElements.htm#VTD005) | System | X06 |
| SD206 |  | Current Transaction Date, in YYYYMMDD format | System | N08 |
| SD207 |  | [Current Transaction Document ID  (VTD400)](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\DataElements\DataElements.htm#VTD400) | System | X15 |
| SD208 |  | [Current Transaction Usage Codes (VTD580)](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\DataElements\DataElements.htm#VTD580) | System | X10 |
| SD220 |  | [Current Transaction Error Message Id.  If a transaction error occurred, SD220 will contain the most recent error message ID for the transaction.  Refer also to the OmniScript function RPMsgs, which allows full access to Current Transaction Error Messages.](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ReportManagement\OmniScript\Functions\RPMsgs.htm) | System | X07 |
| SD221 |  | Current Transaction Error Parms.  If a transaction error occurred, SD221 may contain any parameters used with the error message found in SD220. Note:  SD221 is limited to the first 14 characters of parameter data. Refer also to the OmniScript function RPMsgs, which allows full access to the parameters for Current Transaction Error Messages. | System | X40 |
| SD222 |  | Current Transaction Number of Errors.  If a transaction error occurs, SD222 will contain the number of error messages issued for the transaction. Refer also to the OmniScript function RPMsgs, which allows full access to the parameters for Current Transaction Error Messages. | System | N05 |
| SD230 |  | Skip This Participant.  Can be set to 'Y' at the PartStart or PartTranStart events. Skips all transaction processing for the current participant.Skip This Participant.  Can be set to 'Y' at the PartStart or PartTranStart events. Skips all transaction processing for the current participant. | User | X01 |
| SD231 |  | Skip This Transaction.  Can be set to 'Y' at the TranStart event. Skips processing this transaction. | User | X01 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Literal Alias** | **Field Description** | **Valued By:** | **Format** |
| SD232 |  | Skip This Plan.  Can be set to 'Y' at the PlanStart event. Skips processing this plan. | User | X01 |
| SD901 | SDCR | The Carriage Return character. | System | X01 |
| SD902 | SDLF | The Line Feed character. | System | X01 |
| SD903 | SDFF | The Form Feed character. | System | X01 |
| SD904 | SDTab | The Horizontal Tab character. | System | X01 |
| SD905 | SDEsc, SDEscape | The Escape character. | System | X01 |
| SD906 | SDBel, SDBell | The Bell character. | System | X01 |

Work Accumulator(WA) Variables

Work accumulator fields (WA) accumulate results that are not to be reinitialized to zero between each participant processing. All WA fields are user-defined. Values must be loaded to work fields (WK) to be printed. Up to 100 WA fields are available for use (WA001 through WA100). The example below demonstrates how WA fields might be used to increment check numbers and print them sequentially. An explanation of the function of each Omniscript statement follows.

Work accumulator (WA) field are not available with all transactions or Omniscript Processing Exits. In general, WA fields are used only with the Custom Full Page Report Generator (T966) transaction to accumulate values across participants.

|  |  |
| --- | --- |
| OmniScript Text | Explanation of OmniScript Statement |
| WA003=WA003+1 | Sets WA003 to increment by +1 with each participant. The initial value of WA003 is '0'. |
| WK001=WA003; | Sets WK001 equal to the incrementing value of WA003 so that check numbers can be printed. |

Indexed Work(IW) Variables

Indexed work fields (IW) are used to store temporary values and assign them to work fields (WK). IWs reduce the number of repetitive statements required because they can be repeatedly used in a single OmniScript, but have their values assigned to incrementing work fields. This gives IWs the unique advantage of being especially useful with repeating OmniScript statements such as the LOOP and EACH@ commands.

Indexed work fields (IWs) are assigned to corresponding work fields (WK) based on the setting of the Indexed Work Field Base Value (SD030). This field is established in the OmniScript text and drives the incrementing of the IW/WK link in the OmniScript text. Normally, IW000 assigns a value to WK000. If SD030 contains a value, IW000 assigns a value to WK000 increased by the value in SD030. Thus, if:

SD030=100, then

IW000 will assign a value to WK100,

IW001 will assign a value to WK101,

IW002 will assign a value to WK102

When the end of a repeating Omniscript command is reached (EACH@ or LOOP), SD030 is reset to the amount to increment IW/WK link. Thus, if the above example were complete after WK102, then:

SD030=+3, then

IW000 will assign a value to WK103 (on the next pass through the EACH@ or LOOP)

IW001 will assign a value to WK104,

IW002 will assign a value to WK105

Then the EACH@ or LOOP continues and the sequence is repeated again beginning with WK106, WK107, etc. The example below demonstrates how a LOOP command could be used with indexed work fields to increment values that could be used for a statement or report on loan balances. An explanation of the function of each Omniscript statement follows.

Example:-

Create a report for each participant to display the Market Value, Share Balance, Contributions, and Dividends for Funds 1, 2, 3, 4, and 5. The Report Writer recognizes >100, >101, >102, etc., as work fields (WK100, WK101, etc.).

Required OmniScript Text

PARM SECTION.

FORM=0001

TEXT SECTION.

>L001

>L005 >PH911

>PART

MARKET SHARE

VALUE BALANCE CONTRIB DIVIDEND

Fund 1 >100:F52 >101:F03 >102:F52 >103:F52

Fund 2 >104:F52 >105:F03 >106:F52 >107:F52

Fund 3 >108:F52 >109:F03 >110:F52 >111:F52

Fund 4 >112:F52 >113:F03 >114:F52 >115:F52

CALC SECTION.

SD030 = 100;

WK001 = 1;

EACH@\*\*\*;

IW000 = PF999;

IW001 = PF130;

IW002 = PF200;

IW003 = PF210 + PF220 + PF230;

IF (WK001 > 4);

ENDEXIT;

WK001=+1;

SD030=+3;

ENDEACH;

Result

Base Work Field Value (SD030) is initialized to a value of '100'. The first time the 'EACH' Statement is executed, IW000 through IW003 are converted to WK100 through WK103 respectively. Before exiting the 'EACH' Statement, SD030 is incremented by a value of '4' to change the value of SD030 to '104' prior to the next execution of the 'EACH' Statement.

The second time that the 'EACH' Statement is executed, IW000 through IW003 are converted to WK104 through WK107 respectively. Again, SD030 is incremented by a value of '4' prior to the next execution of the 'EACH' Statement.

Because WK100 through WK119 are the only work fields required to generate the values for the report, a special counter is stored in WK001 to monitor the number of times that the 'EACH' Statement has been executed. The WK001 counter is initialized to '1', and is incremented by '1' prior to the next execution of the 'EACH' Statement. An 'IF' Statement is included within the 'EACH' Statement to check if WK001>5, and if so, execution of the 'EACH' Statement is ended.

(WF) Work Fund Fields

Work Fund Fields (WF) allow fund-related intermediate calculated results to be held.

Work Fund fields are particularly helpful in the Report Writer facility, where a great many figures or balances are to be printed on one page. For instance, if you need to print Market Value, Share Balance, Contributions, and Dividends for all investments in a Plan, a WF field can be assigned to each fund in the CALC SECTION of the Report Definition. The TEXT SECTION can be set up with the appropriate Fund-ID for each calculated Work Fund Field.

* WFs can be followed by an '@' symbol and a three character specific Fund ID. The Fund ID can be 'source' generic (i.e., 10\*), 'investment' generic (i.e., \*\*A), or all generic (i.e., \*\*\*).
* If a WF is not followed by an ‘@’ Fund ID qualifier, the FUND-ID usage rules are followed: the CURRENT-FUND-ID (SD033) is used.
* WFs are only valid for selected applications.
* OmniScript contains up to 999 work fund fields (WF001 through WF999). Note that the number of available Work Fund Fields may be limited based on independent variables unique to each Plan.
* WFs can be user defined or predefined by the transaction you are processing. Most transactions set aside a block of WFs to perform certain tasks. Before creating a program, you must be familiar with the pre-defined WFs for the transaction. Note: Each documentation manual contains a list of all pre-defined fields.
* WFs reduce the number of repetitive statements required, making the program easier to read.
* The FUNDS= XXX Command is used to reserve available storage within the Report Writer when using WFs. Please refer to the Report Writer - Parms section of this manual.

Examples:-The example below demonstrates how WF fields might be used to store participant amounts. An explanation of the function of each OmniScript statement follows.

|  |  |
| --- | --- |
| OmniScript Text | Explanation of OmniScript Statement |
| EACH@\*\*\*; | Initializes an EACH@ statement so that amounts stored in WF fields are by specific investment and source. |
| WF004=PF999;  WF005=PF130;  YWF006=PF100;  ENDEACH; | Stores the participant account value (PF999) in WF004 for each fund and source. Stores the participant share balance (PF130) in WF005 for each fund and source. Stores the participant un-invested cash amount (PF100) in WF006 for each fund and source. |

The following program is used in conjunction with the withdrawal transaction.

\*IF THE PARTICIPANT IS LESS THAN AGE 50.5, PAY ALL FUNDS

\*EXCEPT SOURCE A

\*WF004=PRE-DEFINED AS 'THE MAX.DOLLAR AMOUNT'

\*WF005=PRE-DEFINED AS 'THE MAX.NUMBER OF SHARES'

\*WF006=PRE-DEFINED AS 'THE MAX.AMOUNT OF UNINVESTED CASH'

EACH@\*\*\*;

WF004=PF999;

WF005=PF130;

WF006=PF100;

ENDEACH;

IF PH980<59.06;

EACH@\*\*A;

WF004=0;

WF005=0;

WF006=0;

ENDEACH;

END;

(IT) Indexed Text Fields

Indexed text fields (IT) are used to store temporary values and assign them to text work fields (TX). ITs reduce the number of repetitive statements required because they can be repeatedly used in a single OmniScript, but have their values assigned to incrementing text work fields. These fields work much the same way as Indexed Work fields (IW).

Indexed text fields (ITs) are assigned to corresponding text work fields (TX) based on the setting of the Indexed Text Field Base Value (SD031). This field is established in the OmniScript text and drives the incrementing of the IT/TX link in the OmniScript text. Normally, IT000 assigns a value to TX000. If SD031 contains a value, IT000 assigns a value to TX000 increased by the value in SD031.

Example

In the following example, four IT fields are used with the EACH command; therefore, SD031 must be increased by a value of '4' before each subsequent execution of the loop.

SD031=1; /\* Start with the first one \*/

EACH@\*\*\*;

IT1=FC031;

IT2=FC032;

IT3=FC034;

IT4=FC036;

SD031=+4;

ENDEACH;

(FD) Function Data Elements : Function Data Elements are used to perform temporary loan calculations. Function Data Elements allow users to pass information to OmniPlus and allow OmniPlus to perform calculations, such as amortization and date calculations.

Possible Applications:

* Perform loan calculations for reporting or inquiry purposes (i.e., determine the payment amount for a specified loan amount and number of payments)
* Produce loan amortization schedules in a custom format, either before or after a loan has been issued.
* Produce a loan application or document

# Omniscript Operators

Operators are symbols / notations or expressions used in Omniscript programming to perform any specific operation. Following are the two types of Operators,

* Arithmetic Operators
* Assignment Operators
* Comparison Operators
* Logical Operators

**Arithmetic Operators**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator** | **Explanation** | **Example** | **Operator** | **Explanation** | **Example** |
| = | Assign | WK001 = 3; | - | Subtraction | WK001-5; |
| \* | Multiplication | WK001 \* 3; | \*\* | Exponent (example: 3 squared) | WK001=3 \*\* 2; |
| / | Division | WK001 / 2; | << | Lesser of two values | WK001 = WK002 << WK003; |
| + | Addition (Numeric Values) or | WK001 + 1; or | >> | Greater of two values | WK001 = WK002 >> WK003; |
| Append (Alphanumeric Values) | TX001 + TX002; | =: | Assign and propagate a value into multiple fields | WK001=WK002=:WK003; |

**Assignment Operators**

OmniScript assignment statements store values in data fields. Values may be simple (from another element or literal) or intermediate (computed from an expression). When an assignment is made from a text (alphanumeric) field to a numeric field, the OmniScript facility scans the text field and converts it to a numeric value.

Assignment operators store values in data fields.

The table below lists the assignment operators recognized by the OmniScript facility.

|  |  |  |
| --- | --- | --- |
| **Operator** | **Explanation** | **Example** |
| = | Assign | WK001 = 3; |
| \* | Multiplication | WK001 \* 3; |
| / | Division | WK001 / 2; |
| + | Addition (Numeric Values) or | WK001 + 1; or |
| Append (Alphanumeric Values) | TX001 + TX002; |
| - | Subtraction | WK001-5; |
| \*\* | Exponent (example: 3 squared) | WK001=3 \*\* 2; |
| << | Lesser of two values | WK001 = WK002 << WK003; |
| >> | Greater of two values | WK001 = WK002 >> WK003; |
| =: | Assign and propagate a value into multiple fields | WK001=WK002=:WK003; |

**Assign & Propagate Operator (‘=:’)**

The operator ‘=:’ is used to store a value in the left parameter, while also giving the value as a result. ‘=:’ can be used to store a value and also provide the value for further processing.

**Examples**

WK001=WK002=:100; Stores 100 in WK001 and WK002;

WK001=WK002=:(WK003=:200); Stores 200 in WK001, WK002, and WK003

Note: The first variable must be followed by an equal (‘=’) all other variables must have an equal colon (‘=:’).

**Comparison Operators**

Comparison operators produce a true or false result when comparing two operands. The table below lists the comparison operators recognized by the OmniScript facility and provides a simple example.

|  |  |  |
| --- | --- | --- |
| **Operator** | **Explanation** | **Example** |
| < | Less than | If (WK001<20);… |
| > | Greater than | If (WK001>20);… |
| = | Equal | If (WK001=20);… |
|  |
| If (TX001=’Fred’);… |
| <= | Less than or equal to | If (WK001<=20);… |
| >= | Greater than or equal to | If (WK001>=20);… |
| >< | Not equal to | If (WK001><20);… |
|  |  |
| or | If (WK001<>20);… |
|  |  |
| <> | If (TX001><’Fred’);… |
|  |  |
|  | If (TX001<>’Fred’);… |

**Logical Operators**

Logical operators compare two or more operands and generate a true or false result. A FALSE value is returned for a zero result, and a TRUE value is returned for a non-zero result. The table below lists the logical operators recognized by the OmniScript facility and provides a simple example.

|  |  |  |
| --- | --- | --- |
| **Operator** | **Explanation** | **Example** |
| AND | Compares two or more operands, all of which must be true | If (WK001>20) and (WK001<40);… |
| OR | Compares two or more operands at least one of which is true | If (WK001<20) or (WK001>40);… |

# Omniscript Programming Stmts

Following are some of the control statements used to control the flow of program,

|  |  |
| --- | --- |
| **Type of Statement** | **Statement Used** |
| Conditional | IF / ELSE / END |
| Iteration | EACH@/ ENDEACH  LOOP /END LOOP |
| Termination | QUIT  ABEND |

**Conditional Statements**

**IF / ELSE / END**

The 'IF / ELSE / END' statements test a condition and execute statements based on the results of the test. An 'IF/END' statement tests a condition and may have up to four hundred intermediate work fields held within the one IF statement. Intermediate work fields hold the result of logical operations such as found in IF statements in OCSEM. If the condition is TRUE, OmniScript statements dictate the action to be taken before an “END” statement occurs. If the condition is FALSE, the OmniScript ignores the OmniScript statements for that participant, unless there is an ELSE statement. In this case, the OmniScript statements following the ELSE will be executed up until the END statement. For every “IF” statement, there must be an accompanying 'END' or 'ENDEXIT' statement. ENDIF may also be used in place of an END. See examples below.

**Simple 'IF/END'**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF (PH021=0);  PH600=1;  END; | Tests the condition 'PH021=0'. PH021 is participant status. If the participant has a status of '0' (active), this OmniScript sets the value of PH600 (a user-defined field) to 1. If the participant status is something other than '0', the OmniScript will ignore the participant. The 'END' statement is necessary to stop the 'IF' conditional query. |

**Simple 'IF/ENDIF'**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF (PH021=0);  PH600=1;  ENDIF; | Tests the condition 'PH021=0'. PH021 is participant status. If the participant has a status of '0' (active), this OmniScript sets the value of PH600 (a user-defined field) to 1. If the participant status is something other than '0', the OmniScript will ignore the participant. The 'END' or ‘ENDIF’ statement is necessary to stop the 'IF' conditional query. |

The next example adds the 'ELSE' option to the conditional statement. 'ELSE' introduces OmniScript statements that dictate action to be taken if the condition is FALSE.

**Simple 'IF/ELSE/END'**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF (PH021=0);  PH600=1;  ELSE;  PH600=2;  END; | Again, tests the condition 'PH021=0'. PH021 is participant status. If the participant has a status of '0' (active), this OmniScript sets the value of PH600 (a user-defined field) to '1'. If the participant status is something other than '0', this OmniScript does not ignore the participant, but sets the value of PH600 to '2' (ELSE statement). The 'END' statement is necessary to stop the 'IF' conditional query. |

The third example adds a "nested" 'IF' statement, that is, a condition that must be met after an initial condition has been tested. Remember that each 'IF' statement must have a corresponding 'END' or 'ENDEXIT' statement.

**Simple Nested 'IF/END'**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF (PH021=0);  IF (PH170>150000);  PH600=1;  END;  END; | Tests the condition 'PH021=0'. PH021 is participant status. If the participant has a status of '0' (active), continues to the next conditional statement 'PH170 > 150000'. PH170 is salary. If the participant is active status (PH021=0) and has a salary greater than $150,000 annually, this OmniScript sets the value of PH600 (a user-defined field) to '1'. If the participant status is something other than '0', **or** the participant salary does not exceed $150,000, this OmniScript ignores the participant. **Both** conditions must be met before PH600 is set to a '1'. Note that two 'END' statements are necessary to stop the two 'IF' conditional queries. |

The next example illustrates a sequential 'IF' statement. In this example, multiple conditions are tested in sequence with OmniScript directions following each conditional test.

**Sequential 'IF/END'**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF (PH021=0);  PH600=1;  END;  IF (PH170>150000);  PH601=2;  END; | First, tests the condition 'PH021=0'. PH021 is participant status. If the participant has a status of '0' (active), this OmniScript sets the value of PH600 (a user-defined field) to '1'. Second, tests the condition 'PH170>150000'. If the participant salary (PH170) exceeds $150,000, this OmniScript sets the value of PH601 to a '2'. If the participant status is something other than '0' this OmniScript does not set PH600. If the salary is less than $150,000, this OmniScript does not set PH601. Note that the two conditional tests are independent of each other. The two 'END' statements are necessary to stop the two 'IF' conditional queries. |

The final example illustrates a complex nested 'IF' statement. In this example, a condition is tested and directions dictated before another condition is tested and further directions dictated.

**Complex Nested 'IF/END'**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF (PH021=0);  PH600=1;  IF (PH170>150000);  PH601=2;  END;  END; | First, tests the condition 'PH021=0'. PH021 is participant status. If the participant has a status of '0' (active), this OmniScript sets the value of PH600 (a user-defined field) to '1'. Then, for participants who met the initial condition and had PH600 set to '1', tests a second condition 'PH170>150000'. If this condition is met, this OmniScript sets PH601 to a value of '2' for participants who met both conditions. Note that for participants who did not meet the first condition (PH021=0), the second condition is not even considered. The two 'END' statements are necessary to stop the two 'IF' conditional queries. |

**ELSEIF**

The ELSEIF; statement can be used in combination with an IF statement to allow checking a list of exclusive conditions, then ending with a single END statement. Use of the simple ELSE;IF combination requires an END statement for each IF statement.

Example of using the ELSEIF statement:

IF (WK1=1);

TX1=’a’;

ELSEIF (WK1=2);

TX1=’b’;

ELSEIF (WK1=3);

TX1=’c’;

ELSEIF (WK1=4);

TX1=’d’;

END;

Following is the same example without using ELSEIF, in which an END is required for each IF statement.

IF (WK1=1);

TX1=’a’;

ELSE;

IF (WK1=2);

TX1=’b’;

ELSE;

IF (WK1=3);

TX1=’c’;

ELSE;

IF (WK1=4);

TX1=’d’;

END;

END;

END;

END;

**Iteration Statements**

**EACH@ / ENDEACH**

The 'EACH@ / ENDEACH' statements execute OmniScript statements against qualified participant fund (PF), fund control (FC), investment control (IC), or source control (SC) records. The type of record accessed (PF, FC, IC, or SC) is specified as a suffix on the EACH statement. The default value is participant fund (\_PF).

The 'EACH@' statement identifies a FundId for processing that may be generic or specific for both fund or source (10A, 10\*, \*\*A, or \*\*\*) in the format 'EACH@10A', or 'EACH@\*\*\*', etc. Every 'EACH@' statement must have an accompanying 'ENDEACH' statement. Any number of commands may be entered between the 'EACH@' initial statement and the 'ENDEACH' closing statement. An 'IF/ENDEXIT' statement may be used to exit the 'EACH' loop prematurely.

See the FUND ID USAGE section for the use of SD033, ‘?’, and ‘\*’ for specifying and selecting fund ids.

· The EACH statement sets the current fund Id (SD033) in turn to the key value of each specified record.

· SD033 is restored to its prior value after the ENDEACH.

**Process participant fund records**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| EACH\_PF@10\*;  WF001=PF200;  ENDEACH; | Requests processing any/all PF records for investment 10. For every occurrence of participant funds (PF) for fund 10, any source (\*), then stores the participant contributions in work fund field WF001. Note that a work fund field WF001 exists for each fund and source. As a result, the 'EACH@' statement stores contributions for as many sources as fund 10 has available, e.g., fund 10A in WF001 for 10A, fund 10D in WF001 for 10D, etc. The 'ENDEACH' statement ends the processing loop after PF200 has been stored for all sources of fund 10. |

**Process fund control records**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| SD031=1; | The indexed text base counter (SD031) is set to '1', indicating that IT work fields will correspond to TX work fields plus 1 (IT000=TX001, IT001=TX002, etc.) to begin with. |
| EACH\_FC@\*\*\*; | Initializes the 'EACH@' statement for all funds and sources. |
| IT000=FC026;  IT001=FC031; | Sets the fund ID number (FC026) and the fund name (FC031) in indexed text fields (which correspond to TX001 and TX002 for printing). |
| WK002=+1;  IF (WK002>10);  ENDEXIT; | Work field WK002 is being used as a counter. After each pass through the FC records, it is incremented by one. When it reaches a number greater than 10, the OmniScript uses an 'ENDEXIT' to stop the 'EACH@' loop and exit the program. |
| SD031=+2;  ENDEACH; | After each pass through the FC records, the indexed text base counter is incremented by '2' so that on the second pass IT000 stores the FC record information in TX003, and IT001 stores the FC record information in TX004, and so on with each pass. This way, TX records are not overwritten with each pass. The 'ENDEACH' is necessary to stop the 'EACH@' loop if less than 10 fund control records are found (and the 'ENDEXIT' is never executed). |

**Process investment control records**

The \_IC process always passes all investments.

EACH\_IC@\*\*\*;

...

ENDEACH;

**Process source control records**

The \_SC process always passes all sources.

EACH\_SC@\*\*\*;

...

ENDEACH;

**Dynamic Fund Ids**

Rather than using the @ fund identifier, a fund id can be given in a variable.

TX1=’\*\*A’;

EACH\_FC(TX1);

….

ENDEACH;

An EACH statement without an @ fund identifier or suffix, will reference the value stored in the current fund Id (SD033). If no value is stored in SD033, no records will be returned.

SD033=’10\*’;

EACH;

...

ENDEACH;

Note: The [SD075](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/Desktop%20Docs-SRISIR6A044/EnvironmentFields/EnvironmentFields.htm) flag still overrides the EACH \_PF default.

**LOOP / ENDEXIT / ENDLOOP**

The 'LOOP / ENDEXIT / ENDLOOP' commands execute a block of statements repeatedly to accomplish a specified task. OmniScript instructions found between the 'LOOP' and 'ENDLOOP' iterators are executed until an 'IF/ENDEXIT' is encountered or the Max Loop Counter (SD080) exceeds the maximum loop value set by the user (default is 500). Instructions within the loop command must include an 'IF/ENDEXIT' conditional statement to determine if the loop process is complete.

The 'ENDLOOP' command redirects the process flow back to the first statement of the block to start the loop processing on the next participant. In the example below the OmniScript uses a 'LOOP/ENDLOOP' sequence to store loan balances in work fields for reporting. The loop looks for loan numbers up to '3' and records the loan number and the loan balance in indexed work fields.

**Simple 'LOOP/ENDLOOP'**

|  |  |
| --- | --- |
| **OmniScript Text** | **Explanation of OmniScript Statement** |
| SD030=100; | Sets the Indexed Work Field Base Value (SD030) to 100. So now IW000 will store information in WK100, IW001 in WK101, IW002 in WK102. |
| WK001=1; | Pre-sets WK001 to a value of '1'. This work field will be used to store the incrementing loan numbers. |
| LOOP; | Establishes a LOOP statement. All subsequent commands will be repeated until the OmniScript reaches a condition that ends the loop. |
| KL001=WK001; | Sets the loan number field to '1' (because WK001 was pre-set to this value). This ensures that each pass of the OmniScript for a participant starts with Loan 1. |
| IF WK001>3;  ENDEXIT; | Determines if the loan number is greater than 3, and if so, ends the loop statement. |
| IF LH315>0; | Determines if the participant has a loan balance. LH315 is a loan header data element that contains current balance. |
| IW000=LH300; | Sets IW000 to the loan number. |
| IW001=LH315; | Sets IW001 to the loan balance. |
| SD030=+2;  END; | Increments the Indexed Work Field Base Value by +2 so that now IW000 will store information in WK102, IW001 in WK103 for the next loan. |
| WK001=+1; | Increments the WK001 field that is used to identify the loan number by +1, so that on the second pass the OmniScript will gather information on Loan 2. |
| ENDLOOP; | Ends the LOOP statement. |

**Example**

**Task**

Sum WK fields 1 through 50 into WK100 using Indexed Work (IW) fields.

SD30 = 0; /\* Initialize the IW subscript to zero \*/

LOOP;

WK100 =+ IW1;

SD030 =+ 1; /\* Increment the IW subscript)

IF (SD30>=50);

ENDEXIT;

ENDLOOP;

**Result**

WK100 equals the sum of WK1 through WK50.

**LOOP WHILE/UNTIL**

A WHILE or UNTIL condition can be used with the loop, and is preferred over the IF/ENDEXIT processing.

LOOP UNTIL will process until the parameter condition is true.

LOOP WHILE will process until the parameter condition is false.

WK1=0;

LOOP WHILE WK1<5;

...

WK1=+1;

ENDLOOP;

WK1=0;

LOOP UNTIL WK1>5;

...

WK1=+1;

ENDLOOP;

Eliminates the need to code 'IF/ENDEXIT' within the loop.

**Termination Statements**

**QUIT**

Normally, OmniScripts execute until the end of the supplied statements at the bottom of the text. The 'QUIT' command is used with an 'IF' statement to exit the OmniScript immediately when a condition is met. This command differs from 'GOBACK' in that 'QUIT' always exits the OmniScript function completely, while 'GOBACK' only exits the current routine. In the example below the OmniScript is designed to exit if the participant status is terminated (31). Otherwise, work fields are used to store contribution, cash, and share balances.

**Simple 'QUIT'**

|  |  |
| --- | --- |
| **OmniScript statement** | **Explanation** |
| IF PH021=31;  QUIT;  END;  WK001=PF200@10\*;  WK002=PF120@10\*;  WK003=PF130@10\*; | Examines a participant's status. If the status is '31' (terminated), the QUIT command is executed and the OmniScript goes to the next participant. If the status is not '31', the rest of the OmniScript executes to store contributions, cash, and shares in work fields. |

Note: The QUIT statement does not necessarily stop output from printing. When using a T966 consider setting the Report Selection Indicator (WK212) equal to “1” before using the QUIT statement.

**ABEND**

A second termination command is the 'ABEND' statement. This statement terminates the current job step for all plans processing, and on some platforms (e.g. MVS) generates a system dump for diagnostic purposes. 'ABEND' is typically reserved for debugging and locating serious problems. The command generates a message that states that the abnormal ending of the OmniScript was requested by the OmniScript text based on a conditional statement.

CAUTION: This keyword should only be used at the explicit direction of FIS customer support or programming personnel. The use of this keyword causes an entire processing job to terminate.

**Subroutine**

Subroutine allows the programmer to invoke blocks of Omniscript statements from multiple points in a Omniscript. Following are the three basic subroutine instructions,

* PERFORM – Executes the statements inside the PERFORM instruction before executing the rest of the statements. Eg: PERFORM ‘ENROLLMENT ‘;
* ROUTINE – Contains the subroutine name. Ex: ROUTINE ‘ENROLLMENT’;
* GOBACK - Redefines the process flow to the first statement after the following PERFORM instruction.

Following is an example

EACH@\*\*\*;

  WF001=PF040;

  PERFORM **'ALLOC.CHECK'**;

  IF (WK007=1);

    WK008=+1;

  END;

ENDEACH;

IF (WK008>0);

  TM001='INVEST % NOT 25% INCRMENTS';

END;

**ROUTINE** 'ALLOC.CHECK';

WK007=1;

IF (WF001=0.00) OR

  (WF001=0.25) OR

  (WF001=0.50) OR

  (WF001=0.75) OR

  (WF001=1.00);

  WK007=0;

END;

**GOBACK**;

Following is the explanation of what the above piece of code is doing.

* Initiates an 'EACH@' statement for all funds.  Sets work fund WF001 to the value of the allocation percentages on file for the participant (PF040).
* Instructs the Omniscript to execute the 'ALLOC.CHECK' subroutine, which is defined at the bottom of the Omniscript.
* After executing the 'ALLOC.CHECK', the error flag (WK007) is examined.  If the flag is equal to '1', the error counter (WK008) is incremented by '1'.
* Examines the error counter (WK008).  If it is greater than '0', outputs the warning message that the investment allocations are not in increments of 25%.
* Defines the 'ALLOC.CHECK' routine.  In this routine, WK007 is initialized to the error setting.  Then the participant allocations are examined to ensure that the are in increments of 25%.  If they are, the error setting is changed back to '0'.
* Instructs the Omniscript to return to the first statement after the 'PERFORM' instruction.

# Omniscript Business Object Functions

Omniscript business object functions refer to the built in functions to access the various business objects available in Omniplus under Components menu. Also, this chapter addresses the various operations that can be performed in conjunction with the various business objects.

Following are the various Omniscript business object functions,

|  |  |  |  |
| --- | --- | --- | --- |
| ***Function Name*** | ***Object (Record Type)*** | ***Function Name*** | ***Object (Record Type)*** |
| [AAAAOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\objectLevelGuides\addresses\OmniScript\OmniScriptFunctions.htm) | Alternate Address | [LNLHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Loans\OmniScript\OmniScriptFunctions.htm#LNLHOBJ) | Loans (Loan Header) |
| [ANAMOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Annuities\OmniScript\ANAMOBJ.htm#ANAMOBJ) | Annuity (Annuity Master) | [LNLPOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Loans\OmniScript\OmniScriptFunctions.htm#LNLPOBJ) | Loans (Loan Payment) |
| [PTAIOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\AssociatedIndividual\OmniScript\OmniScriptFunctions.htm) | Participant (Associated Individual) | [MSMSOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Messages\OmniScript\OmniScriptFunctions.htm#MSMSOBJ) | Messages |
| [BAI2OBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\BaseUtilities\OmniScript\Functions\BAI2OBJ.htm) | Base (Alternate Inquiry) | [NTNDOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Notes\OmniScript\OmniScriptFunctions.htm#NTNDOBJ) | Notes (Note Detail) |
| [BALOOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\BaseUtilities\OmniScript\Functions\BALOOBJ.htm) | Base (Log Base Text) | [NTNHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Notes\OmniScript\OmniScriptFunctions.htm#NTNHOBJ) | Notes (Note Header) |
| [BAI3OBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\BaseUtilities\OmniScript\Functions\BAI3OBJ.htm) | Base (Alternate Inquiry 3) | [NTOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Notes\OmniScript\OmniScriptFunctions.htm#NTOBJ) | Notes (Header and Detail) |
| [BAUDOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\BaseUtilities\OmniScript\Functions\BAUDOBJ.htm) | Base (User Defined Base Text) | [DNBEOBJ \*](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/SD.xls#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote) | OmniDBEN (Beneficiary) |
| [PLLCOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\BaseUtilities\OmniScript\Functions\PLLCOBJ.htm) | Plan Locator | [DNCROBJ \*](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/SD.xls#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote) | OmniDBEN (Computation Results) |
| [PLLHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\BaseUtilities\OmniScript\Functions\PLLHOBJ.htm) | Plan Locator Header | [DNIROBJ \*](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/SD.xls#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote) | OmniDBEN (Interest Rate) |
| [BABTOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\BaseText\OmniScript\OmniScriptFunctions.htm#BABTOBJ) | Base Text (Base Text) | [DNPBOBJ \*](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/SD.xls#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote) | OmniDBEN (Prior Benefit) |
| [BAEUOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\BaseText\OmniScript\OmniScriptFunctions.htm#BAEUOBJ) | Base Text (Extended UDF Record) | [DNPNOBJ \*](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/SD.xls#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote) | OmniDBEN (Participant) |
| [CACAOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Cash\OmniScript\OmniScriptFunctions.htm#CACAOBJ) | Cash (Cash Account) | [SVSTOBJ \*](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/SD.xls#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote) | Service (OmniDBEN Service History) |
| [CACHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Cash\OmniScript\OmniScriptFunctions.htm#CACHOBJ) | Cash (Cash History) | [HISHOBJ \*](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/SD.xls#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote#RANGE!Footnote) | History (Sequential History File) |
| [CACPOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Cash\OmniScript\OmniScriptFunctions.htm#CACPOBJ) | Cash (Cash Pointer) | [SSSAOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ShareTrade\OmniScript\BasicConcepts\OmniTrade_Calculator.htm#SSSAOBJ) | OmniTrade (Share Account) |
| [CKCDOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Checks\OmniScript\OmniScriptFunctions.htm#CKCDOBJ) | Check (Check Detail) | [SSSHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ShareTrade\OmniScript\BasicConcepts\OmniTrade_Calculator.htm#SSSHOBJ) | OmniTrade (Share History) |
| [CKCKOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Checks\OmniScript\OmniScriptFunctions.htm#CKCKOBJ) | Check (Check Header) | [SSSROBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ShareTrade\OmniScript\BasicConcepts\OmniTrade_Calculator.htm#SSSROBJ) | OmniTrade (Share Request) |
| [CMCMOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Compensation\OmniScript\CMCMOBJ.htm#CMCMOBJ) | Compensation | [TRTHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ShareTrade\OmniScript\BasicConcepts\OmniTrade_Calculator.htm#TRTHOBJ) | OmniTrade (Trade History) |
| [CMSLOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Compensation\OmniScript\CMSLOBJ.htm#CMSLOBJ) | Compensation (Salary) | [TRTOOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ShareTrade\OmniScript\BasicConcepts\OmniTrade_Calculator.htm#TRTOOBJ) | OmniTrade (Trade Order) |
| [PLCOOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Compliance\OmniScript\OmniScriptFunctions.htm#PLCOOBJ) | Plan (Compliance) | [PTAFOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Contributions\OmniScript\OmniScriptFunctions.htm#PTAFOBJ) | Participant (Annual Financial) |
| [PLCSOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Compliance\OmniScript\OmniScriptFunctions.htm#PLCSOBJ) | Plan (Compliance) | [PTPHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Participants\OmniScript\OmniScriptFunctions.htm#PTPHOBJ) | Participant (Participant Header) |
| [TALMOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Compliance\OmniScript\OmniScriptFunctions.htm#TALMOBJ) | Tax and Compliance Limits | [PEPEOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Persons\OmniScript\OmniScriptFunctions.htm#PEPEOBJ) | Person |
| [DBDBOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Disbursements\OmniScript\DBDBOBJ.htm) | Disbursements | [PLPLOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Plans\OmniScript\OmniScriptFunctions.htm) | Plan |
| [DBDGOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Disbursements\OmniScript\DBDGOBJ.htm) | Disbursements (Deduction Groups) | [PMPMOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Plans\OmniScript\OmniScriptFunctions.htm#PMPMOBJ) | Product Master |
| [DBDHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Disbursements\OmniScript\DBDHOBJ.htm) | Disbursements (Deduction Headers) | [PRDROBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Prices\OmniScript\OmniScriptFunctions.htm#PRDROBJ) | Price (Daily Rate) |
| [DBROOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Disbursements\OmniScript\DBROOBJ.htm) | Disbursements (Rollovers) | [PRPROBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Prices\OmniScript\OmniScriptFunctions.htm#PRPROBJ) | Price (Price Header) |
| [DIMDOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Distributions\OmniScript\OmniScriptFunctions.htm#DIMDOBJ) | Distributions (Minimum Distributions) | [RRRROBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\RateOfReturn\OmniScript\OmniScriptFunctions.htm) | Rate of Return |
| [DIROOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Distributions\OmniScript\OmniScriptFunctions.htm#DIROOBJ) | Distributions (Rollovers) | [RPRHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ReportManagement\OmniScript\Functions\RPRHOBJ.htm) | Reports (UCOM Report Header) |
| [DISGOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Distributions\OmniScript\OmniScriptFunctions.htm#DISGOBJ) | Distributions (Source Group) | [RPRPOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ReportManagement\OmniScript\Functions\RPRPOBJ.htm) | Reports (UCOM Report Page) |
| [DSDSOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\DivSub\OmniScript\OmniScriptFunctions.htm#DSDSOBJ) | Division/Subsidiary | [SVSVOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Service\OmniScript\SVSVOBJ.htm#SVSVOBJ) | Service |
| [FRFAOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Forecasting\OmniScript\FRFAOBJ.htm#FRFAOBJ) | Forecasting (Forecasting Projection Annuity) | [PTPSOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Sources\OmniScript\OmniScriptFunctions.htm#PTPSOBJ) | Source (Participant Source) |
| [FRFBOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Forecasting\OmniScript\FRFBOBJ.htm#FRFBOBJ) | Forecasting (Forecasting Beneficiary) | [SOSCOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Sources\OmniScript\OmniScriptFunctions.htm#SOSCOBJ) | Source (Source Control) |
| [FRFPOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Forecasting\OmniScript\FRFPOBJ.htm#FRFPOBJ) | Forecasting (Forecasting Projection) | [SAEPOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Subaccounting\OmniScript\OmniScriptFunctions.htm#SAEPOBJ) | Sub-Accounting (GIC Processing) |
| [FRFROBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Forecasting\OmniScript\FRFROBJ.htm#FRFROBJ) | Forecasting (Forecasting Header) | [SAGIOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Subaccounting\OmniScript\OmniScriptFunctions.htm#SAGIOBJ) | Sub-Accounting (GIC Base Text) |
| [HIFXOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ForeignExchangeRate\OmniScript\OmniScriptFunctions.htm) | Foreign Exchange History | ***SAILOBJ*** | Sub-Accounting (Investor Lots) |
| [XRXROBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\ForeignExchangeRate\OmniScript\OmniScriptFunctions.htm#XRXROBJ) | Exchange Rate | [SPSPOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\SubPlan\OmniScript\OmniScriptFunctions.htm#SPSPOBJ) | SubPlan |
| [FNFCOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Funds\OmniScript\FNFCOBJ.htm#FNFCOBJ) | Fund (Fund Activity and Fund Control) | [TXOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TextFiles\OmniScript\OmniScriptFunctions.htm#TXOBJ) | Textfile (Headers and Records) |
| [PTPFOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Funds\OmniScript\PTPFOBJ.htm#PTPFOBJ) | Fund (Participant Fund) | [TXTFOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TextFiles\OmniScript\OmniScriptFunctions.htm#TXTFOBJ) | Textfile (Textfile Header) |
| [BAFMOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\History\OmniScript\BAFMOBJ.htm) | File Maintenance | [TXTXOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TextFiles\OmniScript\OmniScriptFunctions.htm#TXTXOBJ) | Textfile (Textfile Record) |
| [HIBROBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\History\OmniScript\HIBROBJ.htm) | History (Base Record) | [VTFHOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\OmniScript\Functions\VTFHOBJ.htm) | Transactions (File Header) |
| [HIETOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\History\OmniScript\HIETOBJ.htm) | History (EFT Transfer History) | [VTTDOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\OmniScript\Functions\VTTDOBJ.htm) | Transactions (Transaction Detail) |
| [ININOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Installments\OmniScript\OmniScriptFunctions.htm#ININOBJ) | Installments | [VTTLOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\OmniScript\Functions\VTTLOBJ.htm) | Transactions (Transaction Log) |
| [PIEVOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Insurance\OmniScript\PIEVOBJ.htm#PIEVOBJ) | Insurance (Participant Event) | [VTTPOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\OmniScript\Functions\VTTPOBJ.htm) | Transactions (Transaction Pointer) |
| [PIPDOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Insurance\OmniScript\PIPDOBJ.htm#PIPDOBJ) | Insurance (Participant Detail) | [VTTSOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\TransactionManagement\OmniScript\Functions\VTTSOBJ.htm) | Transactions (Transaction Submit) |
| [IVIAOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Investments\OmniScript\OmniScriptFunctions.htm#IVIAOBJ) | Investment (Investment Action) | [XFEQOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Transfers\OmniScript\OmniScriptFunctions.htm#XFEQOBJ) | Transfer (Equity Wash) |
| [IVICOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Investments\OmniScript\OmniScriptFunctions.htm#IVICOBJ) | Investment | [PTVROBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\VoiceResponse\OmniScript\OmniScriptFunctions.htm#PTVROBJ) | Voice Response |
| [LNLFOBJ](file:///\\hwpfs307\Is_Development\Cognizant\Omniplus_5.50_Documentation_12-31-2008\PL550Doc%20(E)\ObjectLevelGuides\Loans\OmniScript\OmniScriptFunctions.htm#LNLFOBJ) | Loans (Loan Fund) |  |  |

Business Objects Operations

**XXYYOBJ OBJECT ACCESS Functions**

Purpose

The following section gives basic information common to all OBJ type function libraries. See the particular OBJ library for detailed information particular to that record (e.g. \_VIEW parameters). The xxyyOBJ modules provide OmniScript access to the records and other structures of OmniPlus. Each OmniPlus object (e.g. record type) will have a xxyyOBJ module to access the instance (record), and the individual Data Elements in it. In addition, an xxOBJ module **may** exist to reference the entire (Virtual) object.

**Example**

TXTFOBJ and TXTXOBJ function reference textfile header and textfile data records.

\* Delete the specified textfile header record, but not the associated detail (lines)

TXTFOBJ\_DELETE(PLAN:'111111' FILENAME:'TEST.TEXTFILE');

TXOBJ references the entire textfile (header plus all detail lines).

\* Delete the textfile header record and all textfile detail (lines)

TXOBJ\_DELETE(PLAN:'111111' FILENAME:'TEST.TEXTFILE');

**Terms and Definitions**

**INSTANCE** - Refers to a unique database physical record (e.g. PH-REC), or sometimes to a collection of records (e.g., Participant) consisting of more than one record.

**VIEW** - Identifies criteria selecting a set of records for following sequential processing (usually via NEXT operations).

**OBJID** - Object ID is a TEXT field identifying the object. The \_OBJID operation of data structure libraries returns this text field. OBJID’s are readable, and can be displayed to uniquely identify the object, or extracted/stored and later used to re-fetch the object. OBJID’s may be passed as parameters in VIEW or GET operations to provide key fields.

**Common Object Access Operations**

|  |  |  |
| --- | --- | --- |
| Area | Operations | Description |
| View Oriented | [View](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [First](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Next](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Prev](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Last](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Count](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [RecAreaPresent](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/ObjectAccess.htm) | Define range for accessing records  Access the first qualified record  Access the next record within a VIEW  Access the previous record within a VIEW\*  Access the last qualified record\*  Count all qualified records  Indicates a valid, current record area |
| Record Level | [Init](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [OBJID](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Get](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [GetGE](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [GetGT](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [GetLT](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Delete](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Copy](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Move](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Update](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Add](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Error](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [ErrMsg](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [RecordLabel](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [RecordName](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [RecordType](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) | Initialize a record  Return a record’s unique Object identifier  Access a specific record  Access a record >= to a specified key  Access a record > a specified key  Access a record < a specified key  Delete a record  Copy a record  Move (Copy and Delete) a record  Update (Change) a record  Add (write) a record  Return an error code  Return an error message  Return the record label  Return the record name  Return record type in numeric form |
| Data Element Level  (DE) | [DE](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [NumDE](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [SetDE](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEAttribute](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEFmtLeng](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEHtmInp](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DELabel](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DelsKey](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEName](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DENext](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEPic](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DERDBNum](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DERelated](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEType](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEValDesc](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEValKey](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEValKeyN](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEValid](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEValName](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [DEXML](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) | Return the alphanumeric value of a DE  Return the numeric value of a DE  Set the value of a DE  Returns the DE Attribute  Return the length of a formatted DE  Create HTML FORM input code for a DE  Return the DE label  Returns a value to signify a key field  Return a DE’s name  Return the next DE number  Return the picture clause of a DE  Return the RDB number for a DE  Returns the related DE number  Return a DE’s type  Return the description for a DE value  Return the specified legal value (alphanumeric)  Return the specified legal value (numeric)  Validate a DE Number or Name  Return a DE’s legal value name  Return a DE in descriptive XML format |
| Clipboard | [ToClip](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [ToClipAdd](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) | Copy record to the clipboard as XML or text  Append record to the clipboard as XML or text |
| Other | [Validate1](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l)  [Validate2](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) | Verify all fields contain legal values  Verify all “foreign keys” are valid |

**View Oriented Operations**

**\_VIEW – Define range for accessing records**

The VIEW operation sets the scope of the records to be returned by following \_NEXT operations. Parameter Fields provided with the \_VIEW can identify filters and keys so the user can examine only the records of interest. If no \_VIEW parameters are given, all records on the system database will be qualified.

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

Parameters fields vary with the particular function library based on the type of record. View parameters will include each Key field, usually with LO and HI values (e.g. TRADEDATELO/TRADEDATEHI). Filter parameters may also be allowed, such as STATUS:'31' or FILTFUND:'\*\*A'. See the individual object for it’s provided filters and key ranges.

|  |  |
| --- | --- |
| Parameters | Description |
| [OBJID:tx] | An OBJID, must be one of this object’s ids, or a compatible object’s id |
| [LIMIT:num] | Limits the maximum number of entries to be returned by \_NEXT. May be used to test processes on smaller volumes. After COUNT is exceeded, succeeding NEXT will return End of File. LIMIT: can be used when testing, or to limit accesses to reasonable values. |
| [STATUS:] | Record Dependent TEXT field to filter VIEW/NEXT fetches |

**Example**

The PLPLOBJ library allows the following \_VIEW filters:

|  |  |
| --- | --- |
| Parameter | Description |
| [PLAN:tx] | Plan Identifier |
| [PLANLO:tx] | Lo plan number to start with |
| [PLANHI:tx] | Stop after processing this plan number |

The PTPHOBJ library allows the following \_VIEW filters:

|  |  |
| --- | --- |
| Parameter | Description |
| [PLAN:tx] | Plan Identifier |
| [PLANLO:tx] | Lo plan number to start with |
| [PLANHI:tx] | Stop after processing this plan number |
| [STATUS:tx] | Qualifies only participants with PH-STAT indicating ‘ACTIVE’, ‘INACTIVE’ ‘SUSPENDED’, ‘RETIRED’, ‘DEFERRED’, ‘TERMINATED’, and ‘INELIGIBLE’ |
| [STATUS:’30’] | Qualify only participants with PH-STAT = ‘30’ |

VIEW Parameters vary by object and may include:

§ Each key field, with a LO/HI value. (e.g. PLANLO:'000001', PLANHI:'222222')

§ Filter criteria for various key or data fields. (e.g. STATUS:'31' or FUNDFILT:'1\*\*' or RUNDATELO:19980101)

**\_FIRST – Access the first qualified record**

Specify access filters and get first occurrence

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

**Returns**

1 (True) if a record was found

0 (False) otherwise

**Example**

PTPHOBJ\_FIRST(PLAN:'777777');

\* Returns the first PH record occurrence in plan '777777'

**\_NEXT – Access the next qualified record within a VIEW**

Return next instance within a VIEW. A previous VIEW operation is required.

**Parameters**

None

**Returns**

1 (True) if a instance is available, 0 (False) Error or End of File

After a successful \_NEXT, there is a current record with which to perform record or data element operations.

Note:If [Limit] is used and qualifying records exist beyond the set limit, \_Next will terminate and load SD101 with ‘Exceeded Given Limit’, SD100 with a ‘4’.

**Example**

PTPHOBJ\_VIEW(PLAN:'777777');

LOOP WHILE PTPHOBJ\_NEXT();

OCSHOW(PTPHOBJ\_DE("NAME"));

ENDLOOP;

**\_PREV – Access the previous qualified record within a VIEW**

Return previous instance within a VIEW. A previous VIEW operation is required. This operation is identical to NEXT, except that records are read in reverse order.

Note: The PREV operation is only valid on the Windows and UNIX platforms. The PREV operation is not available on the Mainframe (MVS) platform.

Note: The PREV operation is currently only valid with the following Object Access Functions: BAI2OBJ, CMCMOBJ, CMSLOBJ, DNCROBJ, DNPBOBJ, DNPNOBJ, HIBROBJ, PLPLOBJ, PTPHOBJ, SVSTOBJ, and SVSVOBJ.

**Parameters**

None

**Returns**

1 (True) if a instance is available, 0 (False) Error or Beginning of File

After a successful \_PREV, there is a current record with which to perform record or data element operations.

**Example**

PTPHOBJ\_VIEW(PLAN:'777777');

LOOP WHILE PTPHOBJ\_PREV();

OCSHOW(PTPHOBJ\_DE("NAME"));

ENDLOOP;

**\_LAST – Access the last qualified record**

Specify access filters and get last occurrence.

Note: The LAST operation is currently only valid with the following Object Access Functions: BAI2OBJ, CMCMOBJ, CMSLOBJ, DNCROBJ, DNPBOBJ, DNPNOBJ, HIBROBJ, PLPLOBJ, PTPHOBJ, SVSTOBJ, and SVSVOBJ.

Note: The LAST operation is not available on the MVS platform.

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

**Returns**

1 (True) if a record was found

0 (False) otherwise

**Example**

PTPHOBJ\_LAST(PLAN:'777777');

\* Returns the last PH record occurrence in plan '777777'

**\_COUNT – Count the number of qualified records**

Executes a VIEW and then NEXT functions, counting each record, returning the total number of qualified records.

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

**Returns**

Number of Object Instances in this VIEW.

**Example**

WK001=PTPHOBJ\_COUNT(PLAN:'777777');

\* Stores the number of participants in plan '777777' in WK1

Note: \_COUNT does in fact pass all qualified records to count them. For performance reasons, you should only use \_COUNT if you will not pass the records yourself with \_VIEW/\_NEXT. If you will process the records, it will be more efficient (especially for large record counts) to tally the records yourself as you process them; otherwise passing the records twice will be inefficient.

**\_RecAreaPresent- Indicates a valid, current record area is available**

The RecAreaPresent operation will return 1/true if an INIT was performed or a record was fetched using a GET, NEXT, or similar operation. RecAreaPresent will return 0/false if either no operation has been done to establish the record area or the prior GET, NEXT or similar operation failed.

Note that RecAreaPresent does not necessarily indicate that the record exists on the database. A previous SETDE operation may have changed a key field on the record, or the record area may have been established by an INIT operation.

**Alias**

CurrentRec and CurrentRecord

**Returns**

1 (True) if a record was found

0 (False) otherwise

**Example**

\* Fetch the 2005 TALM record if it isn't already available.

If TALMOBJ\_RecAreaPresent()=False;

If TALMOBJ\_NumDE(015) <> 2005;

TALMOBJ\_Get(Year:2005);

End;

End;

**Record Level Operations**

**\_INIT – Initialize a record**

INITIALIZE function to clear all fields in the current record. Can be used prior to valuing fields for an Add/Write operation. Clears each field in the instance (record).

**Parameters**

Key parameters, which vary by record type, are allowed.

**Example**

PTVROBJ\_INIT(PLAN:'777777' PARTID:PH007);

PTVROBJ\_SETDE(DENUM:100 VALUE:'5789');

PTVROBJ\_SETDE(DENUM:105 VALUE:4);

PTVROBJ\_ADD();

**\_OBJID – Return a record’s unique object identifier**

Return a text field with the object identifier

**Parameters**

None

**Returns**

A text field with the object identifier

**Example**

PTPHOBJ\_VIEW(PLAN:SD001);

PTPHOBJ\_NEXT();

TX001=PTPHOBJ\_OBJID();

\* RETURNS: 'OBJID:PTPH PLAN:111111 PARTID:000000001'

**\_GET – Access a specific record**

Get using specified key fields or OBJID

**Parameters**

Key parameters, which vary by record type, are allowed.

**Returns**

1 (True) if record was found, 0 (False) otherwise

**Example**

PTPHOBJ\_GET(PLAN:'111111' PARTID:'222222222');

**\_GETGE – Access a record greater than or equal to the specified key**

Get greater than or equal to (>=) using specified Key fields or OBJID

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

**Example**

PTPHOBJ\_GETGE(PLAN:'777777' PARTID'000000001');

TX001=PTPHOBJ\_OBJID();

Note: Key information can be specific (e.g. PLAN:'111111' PARTID:'222222222')

or an ObjectID (e.g. OBJID:TX1)

or generic (e.g. PLAN:'111111'')

**\_GETGT – Access a record greater than to the specified key**

Get greater than (>) using specified Key fields or OBJID

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

**\_GETLT – Access a record less than the specified key**

Get less than (<) using specified Key fields or OBJID

**Parameters**

Key or Filter parameters, which vary by record type, are allowed.

Note: The GETLT and GETGT operations are not currently valid with the following Objects: VTRAN and Share/Trade.

Note: The GETLT operation is not available on the Mainframe (MVS) platform.

**\_DELETE – Delete the specified or current record**

Delete specified Occurrence

**Parameters**

Key parameters, which vary by record type, are allowed.

If key information is given, delete the specified instance.

If no parameters are given, delete the current instance

**Returns**

1 (True) if an instance is deleted, 0 (False) if no delete occurred due to Error/End of File

**Example**

IF DSDSOBJ\_GET(PLAN:'777777' DIVSUB:'1234');

DSDSOBJ\_DELETE();

TX001='REC DELETED';

ELSE;

TX001='DELETE FAILED';

END;

**\_COPY – Copy the current record**

Copy specified occurrence, using given Key fields. Makes a copy of the current record, and writes it using the given new Key fields.

**Parameters**

Key parameters for the new record.

**Returns**

1 (True) if an instance is copied, 0 (False) if no COPY occurred due to Error

Note: If [Plan:] or [Partid:] is not coded for a new plan or participant, SD101 will be loaded with ‘Cannot copy onto self’.

**Example**

IF PTPHOBJ\_GET(PLAN:'777777' PARTID:'444000000');

/\* copy the current PTPH record from plan 777777 to plan 777780 \*/

PTPHOBJ\_COPY(PLAN:'777780');

TX001='PH RECORD COPIED';

ELSE;

TX001='COPY FAILED';

END;

**\_MOVE – Move the current record**

Delete specified occurrence and copy to given Key

**Parameters**

Key parameters for the new record

**Returns**

1 (True) if an instance is moved, 0 (False) if no MOVE occurred due to Error

**Example**

IF PTPHOBJ\_GET(PLAN:'777777' PARTID:'444000000');

PTPHOBJ\_MOVE(PLAN:'777780');

TX001='PH RECORD MOVED';

ELSE;

TX001='MOVE FAILED';

END;

**\_UPDATE – Update (Change) the current record**

Rewrite current occurrence.

**Parameters**

None

**Returns**

1 (True) if an instance is copied, 0 (False) if failed occurred due to Error

**Example**

DSDSOBJ\_VIEW(PLAN:'777780');

LOOP WHILE DSDSOBJ\_NEXT();

DSDSOBJ\_SETDE(DENUM:200 VALUE:1);

DSDSOBJ\_UPDATE();

ENDLOOP;

Note: This function cannot be used to change the Key fields of a record.

**\_ADD – Add the current record**

ADD the current occurrence to the database

**Parameters**

None

**Returns**

1 (True) if add was successful, 0 (False) otherwise

**Example**

\* ADD PIN NUMBER

PTVROBJ\_INIT(PLAN:SD001 PARTID:PH007);

PTVROBJ\_SETDE(DENUM:100 VALUE:'5789');

PTVROBJ\_SETDE(DENUM:105 VALUE:4);

PTVROBJ\_ADD();

**\_ERROR – Return the error status of the previous IO operation**

**Parameters**

None

**Returns**

1 (True) if the last IO operation terminated in error, 0 (False) if the last IO operation was successful

**Example**

PLPLOBJ\_GET(PLAN:'111111'); /\* Attempt to read this record \*/

WK001=PLPLOBJ\_ERROR(); /\* WK001 might be 0 indicating success, or 1 indicating no such record \*/

**\_ERRMSG – Return the error message of the previous IO operation**

Return code and message of previous IO operation

**Parameters**

None

**Returns**

A TEXT message containing the File Status and a description of the last IO completion FILE-STATUS

**Example**

PLPLOBJ\_GET(PLAN:'111111'); /\* Attempt to read this record \*/

TX001=PLPLOBJ\_ERRMSG(); /\* TX001 might be '00 - Successful' or '23 - Record not Found' \*/

**\_RecordLabel – Return the record label**

Returns the label for the object record.

**Parameters**

None

**Returns**

The label for the record being requested.

**Example**

TX003=AAAAOBJ\_RECORDLABEL(); /\* Show the record label value \*/

Returns: 'AltAddr'

**\_RecordName – Return the record name**

Returns the name for the object record.

**Parameters**

None

**Returns**

The name for the record being requested.

**Example**

TX002=PTPHOBJ\_RECORDNAME(); /\* Show the record label value \*/

Returns: 'Participant Header'

**\_RecordType – Return the numeric value for record type**

Returns a numeric value to represent the record type.

**Parameters**

None

**Returns**

1 (Record), 2 (Transaction), 3 (Trailer), 4 (Control)

**Example**

TX002=PTPHOBJ\_RECORDNAME(); /\* Show the record label value \*/

Returns: 'Participant Header'

**Data Element Level Operations**

Once a record is current, either by reading it or by initializing an empty instance, the DE operations can retrieve or set values. Data Elements may be accessed by Data Element Number or Name.

**\_DE – Return the text value of a data element**

Fetch Text value of the specified Data Element. If necessary, converts a numeric element to it’s TEXT equivalent using its default format. If the actual numeric value is wanted, use \_NUMDE to fetch it.

**Parameters**

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

Text value of the Data Element

**Examples**

PTPHOBJ\_GET(PLAN:'777777' PARTID:'1230004567');

TX001=PTPHOBJ\_DE(011);

/\* OR \*/

TX001=PTPHOBJ\_DE('NAME');

/\* OR \*/

TX001=PTPHOBJ\_DE('naMe');

/\* OR \*/

TX001=PTPHOBJ\_DE(NAME:'011');

/\* OR \*/

TX001=PTPHOBJ\_DE(NAME:'DE011')

/\* OR \*/

TX001=PTPHOBJ\_DE(NAME:'PH011');

**\_NUMDE – Return the numeric value of a data element**

Fetch Numeric value of Data Element

**Parameters**

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

The value of the specified Data Element

**Example**

PTPHOBJ\_VIEW(PLAN:'777777');

PTPHOBJ\_NEXT();

TX001=PTPHOBJ\_NUMDE('SALARY');

/\* OR \*/

TX001=PTPHOBJ\_NUMDE(170);

**\_SETDE – Set the value of a data element**

Sets the specified value of the specified Data Element

**Parameters**

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

Followed by:

|  |  |
| --- | --- |
| Parameter | Description |
| [VALUE:]tx or num | The value to set the element to |

**Returns**

1 (DE stored), 0 (Invalid field or value)

Note: Returns SD101 loaded with ‘No Such Data Element: DE999’ if requested DE doesn’t exist. Returns SD101 loaded with ‘Mismatched Data Type: DE999’ for attempts to give numeric data elements text values (or vice versa).

**Example**

PTPHOBJ\_VIEW(PLAN:'777777');

PTPHOBJ\_NEXT();

PTPHOBJ\_SETDE(015 '100044');

/\* OR \*/

PTPHOBJ\_SETDE(DENUM:015 VALUE:'100044');

/\* OR \*/

PTPHOBJ\_SETDE(NAME:'EMPL-NUM' VALUE:'100044');

**\_DEAttribute – Returns the DE Attribute**

Returns the special attribute of the specified Data Element

**Parameters**

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

‘ ‘ (Primary), ‘A’ (Artificial), ‘C’ (Composite), ‘P’ (Part of), ‘R’ (Redefinedl), ‘F’ (Fixed)

TX010=PTAIOBJ\_DEATTRIBUTE(210);

Returns: C

**\_DEFMTLENG – Get the formatted length of a data element**

Return the # of bytes required to show a data element. For Text elements, this will be the # of bytes. For numeric elements, it will be the total formatted length, including any formatting characters such as commas, decimals, signs, slashes, etc. This operation should be used for output formatting (such as HTML screens or reports) where it is necessary to get the formatted length of a field.

**Parameters**

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

The number of characters in the actual formatted length of the element.

**Example**

\* Show all legal values and descriptions for PH170 (salary)

WK001=PTPHOBJ\_DEFMTVAL(170);

**\_DEHTMINP – Create HTML FORM input code for a DE**

Adds HTML code to the OmniScript clipboard to format a data element as a pre-valued HTML FORM input field. For data elements with multiple legal values, this would return the HTML code for a drop down menu. This operation is useful when generating HTML FORM code using OmniScript.

Refer to [OCCLIP](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/OCClip.htm) documentation for more information on the OmniScript Clipboard Facility.

**Parameters**

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

The number of lines added to the clipboard. (HTML text is appended to the current clipboard contents).

**Example**

TX001 = PTPHOBJ\_DeHtmInp(030);

/\*

clipboard contents may contain the following, for example:

<select name="PTPH030">

<option select="1">Male</option>

<option select="2">Female</option>

</select>

\*/

TX001 = PTPHOBJ\_DeHtmInp(011);

/\*

clipboard contents may contain the following, for example:

<input type="text" name="PTPH011" size="30" maxlength="30" value="PARTICIPANT, SAMPLE " />

\*/

**\_DELabel – Returns the DE Label**

Returns the label value of a data element.

**Parameters**

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

‘ ‘ (Primary), ‘A’ (Artificial), ‘C’ (Composite), ‘P’ (Part of), ‘R’ (Redefinedl), ‘F’ (Fixed)

TX004=DBDBOBJ\_DELABEL(160);

Returns: DistType

**\_DElsKey – Returns a value if a key field**

Returns a value if the field is a key field.

**Parameters**

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

1 (Key field), 0 (Not a key field)

TX005=FNFCOBJ\_DEISKEY(026);

**\_DENAME – Return the name of a Data Element**

**Parameters**

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Example**

TX001=PTPHOBJ\_DENAME(DENUM:'021'); /\* Returns: ‘STAT’ \*/

**\_DENEXT – Return the next valid data element number**

Return the next highest Valid DE number for this record. DENEXT can be used to sequentially fetch all the elements for a record, in data element order.

**Parameters**

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

The next highest valid Data Element number for this record

0 – no more data elements exist

**Example**

WK001= PTPHOBJ\_DENEXT(0);

LOOP WHILE WK001>0;

/\* Code to process each Data Element goes here \*/

WK001=PTPHOBJ\_DENEXT(WK1);

ENDLOOP;

**\_DEPIC – Return the OmniPlus DE Picture of a DE**

Returns the internal Picture of the specified data element, such as “N11D00” or “X40”. See the OmniPlus Data Element overview for an explanation of OmniPlus element pictures. Refer also to the [DETYPE](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) operation.

**Parameters**

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

The text of the Internal OmniPlus picture of the element

**Example**

TX001=PTPHOBJ\_DEPIC(170); /\* Returns: ‘N11D02’ \*/

**\_DERDBNum – Returns the RDB number**

Returns the relational data base number of a specified data element.

**Parameters**

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

TX007=PTPFOBJ\_DERDBNUM(170);

Returns: 32

**\_DERelated – Returns the Related DE number**

Returns the DE number that the defined DE is a part of (redefined and part of de’s).

**Parameters**

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

TX006=NTNHOBJ\_DERELATED(105);

Returns: 100

**\_DETYPE – Return the type of a specified data element**

Return the type of the specified Data Element (‘X’ or ‘N’)

**Parameters**

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

Ø ‘X’ if the given element is a TEXT element

Ø ‘N’ if it is a NUMERIC element.

Ø Spaces if no such element exists

**Example**

TX001=PTPHOBJ\_DETYPE(007);

\* Returns: 'X'

/\* OR \*/

TX001=PTPHOBJ\_DETYPE(DENUM:170);

\* Returns: 'N'

**\_DeValDesc – Return the description of a DE Value**

This operation is used to display the legal values for data elements that have multiple legal values. In OmniStation, these data elements show as a “pull down menu” with all of the legal values displayed. This operation returns the text description for the specified legal value for the specified data element. Refer also to the [DeValKey](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) operation.

**Parameters**

Either of the following

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

And

|  |  |
| --- | --- |
| Parameter | Description |
| INDEX:num | The selected subscript of the value, starting at 1 |

**Returns**

The text value of the selected value description, or ‘’ if no such value.

SD101 will be set to the description of the selected value.

**Example**

\* Get the first legal value and it’s description for PH023

TX001=PTPHOBJ\_DEVALDESC(141 INDEX:1);

**\_DeValid – Determine if a Data Element exists**

Return 1 (True) if the parameter Data Element exists

**Parameters**

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Returns**

1 (True) if the parameter DE exists, 0 (no such DE)

**\_DeValKey – Get the specified legal value**

This operation is used to display the legal values for data elements that have multiple legal values. In OmniStation, these data elements show as a “pull down menu” with all of the legal values displayed. This operation returns the text value of the specified element’s value. Refer also to the [DeValDesc](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) and DeValKeyN operations.

**Parameters**

Either of the following

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

And

|  |  |
| --- | --- |
| Parameter | Description |
| INDEX:num | The selected subscript of the value, starting at 1 |

**Returns**

The text value of the selected element, or ‘’ if no such value.

SD101 will be set to the description of the selected value.

**Example**

\* Show all legal values and descriptions for PH023

WK001=1;

LOOP;

TX001=PTPHOBJ\_DEVALKEY(023 INDEX:WK001);

IF SD101 =''; ENDEXIT;

OCSHOW(TX001 SD101 WK001);

WK001=+1;

ENDLOOP;

**\_DeValKeyN – Get the numeric value of DeValKey**

Refer to the [DeValKey](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) operation. Returns the numeric value of a data element’s legal value.

**Example**

WK001=PTPHOBJ\_DEVALKEYN(023 INDEX:WK001);

**\_DeValName – Return the name a data elements value**

This operation is used to display the legal values for data elements that have multiple legal values. In OmniStation, these data elements show as a “pull down menu” with all of the legal values displayed. This operation returns the descriptive value associated with the data element’s current value.

**Parameters**

Either of the following:

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Example**

TX001=PTPHOBJ\_DEVALNAME(NAME:'STAT');

Returns: 'Active and Eligible'

**\_DEXML – Format a Data Element’s Values**

Return a text line describing the indicated data element

**General Parameter**

|  |  |
| --- | --- |
| Parameter | Description |
| [DENUM:]num | The 3 digit numeric data element number of the element |
| [NAME:]tx | The name of the data element (in quotes). The name may be in any of the following formats: - The name as it appears on the OmniStation Fields tab - The data element number in 999 or DE999 format - The data element in YY999 format, were YY is the record type (e.g. “PH011”) (**NOTE:** This entry is **NOT** case-sensitive.) Note that the following values may be accessed by data element name only: “RecordLength”, “KeyLength”. |

**Formatting Parameters**

|  |  |
| --- | --- |
| Parameter | Description |
| [FMT:txt] | The selected format type.  ‘XML2’ – XML, with a <Tag> value </Tag> format (default)  ‘XML’ – Basic XML  ‘PRINT’ – Data Element=’Value’ format |
| [TAG:tx] | Select the per-element XML tag.  ‘DE’ – The tag will be ‘<de>’  ‘DENUM’ – The tag will be <de###> where ### is the De Number.  ‘LABEL’ – The tag will be a generated DE name. (e.g. <PLANNUM>. |
| [ATTRIBS:TX] | Identifies which Data Element fields are to be included as XML attributes.  One or more items may be specified.  ‘#’ – DE Lumber (num=”005”)  ‘L’ – DE Label  ‘N’ – DE Name (name=”PLANNUM”)  ‘P’ – DE Picture (pic=”X06”)  ‘V’ – DE Value (value=“000001”)  ‘D’ – DE Value Description (desc=“Retired”)  ‘\*’ – All the above  The default is no attributes.  Example: ATTRIBS:’#N’ |
| [NODES:TX] | Identifies which Data Element fields are to be included as XML nodes.  One or more items may be specified.  ‘#’ – DE number (<num>005</num>)  ‘L’ – DE Label  ‘N’ – DE Name (<name>PLANNUM)</name>  ‘P’ – DE Picture <pic>X06)</pic>  ‘V’ – DE Value (<value>000001</value>  ‘D’ – DE Value Description (<desc>Retired</desc>)  ‘\*’ – All the above  The default is no nodes.  Example: NODES:’#ND’ |

**Returns**

The text values associated with the selected element

‘’ if no data element exists ([SD101](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/EnvironmentFields/EnvironmentFields.htm) will contain the error reason)

**Examples**

|  |  |
| --- | --- |
| OmniScript Text: | Returns: |
| PTPHOBJ\_DEXML(050); | <DE050> 1970/07/01 </DE050> |
| PTPHOBJ\_DEXML(050 TAG:'LABEL'); | <BrthDate> 1970/07/01 </BrthDate> |
| PTPHOBJ\_DEXML(050 FMT:'XML2' TAG:'DE' NODES:'#V'); | <DE> <num> 050 </num><value> 1970/07/01 </value> </DE> |
| PTPHOBJ\_DEXML(050 FMT:'XML2' ATTRIBS:'#L'); | <DE num="050" label="BrthDate"> 1970/07/01 </DE> |
| PTPHOBJ\_DEXML(050 FMT:'XML2' NODES:'#LV'); | <DE> <num> 050 </num><value> 1970/07/01 </value><label> BrthDate </label> </DE> |
| PTPHOBJ\_DEXML(050 FMT:'XML' ATTRIBS:'#V' TAG:'LABEL'); | <BrthDate num="050" value="1970/07/01"/> |
| PTPHOBJ\_DEXML(050 FMT:'XML2' ATTRIBS:'#' TAG:'LABEL'); | <BrthDate num="050"> 1970/07/01 </BrthDate> |
| PTPHOBJ\_DEXML(050 FMT:'PRINT'); | DE050=1970/07/01 |

**Clipboard Operations**

Refer to [OCCLIP](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/OCClip.htm) documentation for more information on the OmniScript Clipboard Facility.

**\_TOCLIP – Copy the current record to the clipboard**

Copies the current record to the clipboard as formatted text, replacing existing clipboard contents. This text may include all data element or only valued data elements. A wide variety of XML formats and options are available. Refer to the [DeXML](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) operation for more information.

**General Parameter**

|  |  |
| --- | --- |
| Parameter | Description |
| [VALUED:num] | Selects only valued elements, or all elements  0 – Only valued (non zero/blank) elements  1 – All elements |
| [NODEID:txt] | Create an XML node for this record’s data.  For example, if NODEID:’MyRec’ was specified, the XML text would contain  <MyRec>  … generated XML for each data element …  </MyRec> |

**Formatting Parameters**

[Additional Formatting parameters](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) are allowed to control the format of the data. These parameters are documented under the DeXML operation above.

**Returns**

The number of data elements copied to the clipboard. Zero if no none were selected or a clipboard error occurred.

**Example**

PTPHOBJ\_TOCLIP(NODEID:'PARTICIPANT' TAG:'LABEL' VALUED:0);

OCSHOW\_FROMCLIP(); /\* or \*/ OCFILE1\_FROMCLIP();

Might returns the following text to the clipboard

<PARTICIPANT>

<PlanNum> 123456 </PlanNum>

<PartId> 123456789 </PartId>

<PartNumX> 123456789 </PartNumX>

<Name> PARTICIPANT, TEST </Name>

<Stat> 00 </Stat>

--------- ADDITIONAL DATA ELEMENTS -----------------------

</PARTICIPANT>

**\_TOCLIPADD – Append the current record to the clipboard**

This operation is identical to the [TOCLIP](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/DKH8P1XN/l) operation, except that the current record is appended to the clipboard, instead of replacing the clipboard contents.

**Other Operations**

**\_VALIDATE1 – Verify all fields are valid**

Checks each field on the record to insure valid values (e.g. Flags, valid Dates, etc.), confining the checks to inspecting this particular instance.

**Parameters**

None

**Returns**

0 (Invalid Record), 1 (Valid Record)

Note: If invalid, SD100 will contain the DE number found to be invalid, and SD101 will contain a reason message.

**\_VALIDATE2 – Verify all ‘foreign keys’ are valid**

Verify all ‘foreign keys’ are valid. Accesses other records in the system to verify the proper relationship to this record. For example, PTPHOBJ\_VALIDATE2 would insure the specified Plan exists, the specified Participant exists, the specified Fund is valid for the plan.

**Parameters**

None

**Returns**

0 (Invalid Record), 1 (Valid Record)

Note: If invalid, SD100 will contain the DE number found to be invalid, and SD101 will contain a reason message.

# Omniscript Utility Functions

Omniscript Utility functions are tailor-made functions which aids in performing text, numeric manipulations and other calculations.

Following are the Omniscript Utility Functions,

|  |  |  |
| --- | --- | --- |
| ***Function Name*** | ***Description*** | ***Object Guide*** |
| [***BARUN***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/BARUN.htm) | **Establish OmniPlus Processing Environment** | **OmniScript** |
| [***DDRESP***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/DDRESP.htm) | **Generate OmniStation Packet Response** | **OmniScript** |
| [***OC966U***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/objectlevelguides/omniscript/functions/functions/OC966U.htm) | **Trial-Compile Report Writer Function** | **OmniScript** |
| [***OCCLEAR***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/objectlevelguides/omniscript/functions/functions/occlear.htm) | **Initialize OmniScript Field Values** | **OmniScript** |
| [***OCCLIP***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCClip.htm) | **Clipboard Facility** | **OmniScript** |
| [***OCCNTL***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCCNTL.htm) | **Parameter Control File Facility** | **OmniScript** |
| [***OCCSV***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCCSV.htm) | **Interface with Comma Separated Values (.CSV) formatted data** | **OmniScript** |
| [***OCDATA***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCData.htm) | **Data Container Facility** | **OmniScript** |
| [***OCDATE***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCDATE.htm) | **Date Utility** | **OmniScript** |
| [***OCEVAL***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OcEval.htm) | **Examine a field against a set of conditions/values** | **OmniScript** |
| [***OCFACT***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCFact.htm) | **Financial Factor / Allocation Library** | **OmniScript** |
| [***OCFILB***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/ocfilb.htm) | **External File Read and Write Access (Byte Stream Files)** | **OmniScript** |
| [***OCFILE***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCFILE.htm) | **External File Read and Write Access** | **OmniScript** |
| [***OCFILV***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCFILV.htm) | **External File Read and Write Access (Wide Files)** | **OmniScript** |
| [***OCFIND***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCFind.htm) | **Find (Lookup) Item Among Given Parameters** | **OmniScript** |
| [***OCFMT***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCFMT.htm) | **Format Numeric Fields into Text with Given Picture** | **OmniScript** |
| [***OCFUNC***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCFUNC.htm) | **OmniScript Function Execution Facility** | **OmniScript** |
| [***OCLABEL***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OcLabel.htm) | **User Defined Label Utility** | **OmniScript** |
| [***OCLABVAL***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCLABVAL.htm) | **Compare tables of labels and values** | **OmniScript** |
| [***OCLINE***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCLine.htm) | **Produce OP50 Report Line Output** | **OmniScript** |
| [***OCNUM***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCNUM.htm) | **Arithmetic and Numeric Library** | **OmniScript** |
| [***OCPAGE***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCPage.htm) | **Produce T966 Report Output Directly from OmniScript** | **OmniScript** |
| [***OCPARMS***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCParms.htm) | **Retrieve Parameters from T588/T988 transaction entry** | **OmniScript** |
| [***OCPRAM***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/objectlevelguides/omniscript/functions/functions/OCPram.htm) | **OmniScript Parameter Processing Facility** | **OmniScript** |
| [***OCSCREEN***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCScreen.htm) | **Interactive Screen Facility** | **OmniScript** |
| [***OCSCRIPT***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCScript.htm) | **OmniScript Execution Facility** | **OmniScript** |
| [***OCSEL***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCSel.htm) | **Select Item from Parameter List** | **OmniScript** |
| [***OCSHOW***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCSHOW.htm) | **'Exhibit Named' Type Display of Variables** | **OmniScript** |
| [***OCSPLIT***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/ocsplit.htm) | **Text Parsing Utility** | **OmniScript** |
| [***OCSUB***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCSUB.htm) | **Perform Text Sub-string Operations** | **OmniScript** |
| [***OCTEST***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCTest.htm) | **Self Checking Canned Test Data Support** | **OmniScript** |
| [***OCTEXT***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCText.htm) | **Text Field Utility** | **OmniScript** |
| [***OCTIME***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCTime.htm) | **Perform Various Time Related Operations** | **OmniScript** |
| [***OCTX***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCTX.htm) | **TX Field Array Utility Library** | **OmniScript** |
| [***OCVALUE***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCValue.htm) | **Iterate Numeric Values for the Loop While** | **OmniScript** |
| [***OCWK***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCWK.htm) | **WK Field Array Utility Library** | **OmniScript** |
| [***OCWORD***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCWord.htm) | **Convert Numeric Values Into Word Equivalents** | **OmniScript** |
| [***OCXML***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/OCXML.htm) | **XML (eXtended Markup Language) Parsing Facility** | **OmniScript** |
| [***UTFSYS***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/UTFSYS.htm) | **File System Object Facility** | **OmniScript** |
| [***UTINDX1***](../AppData/Local/Microsoft/Windows/Temporary%20Internet%20Files/Content.Outlook/ObjectLevelGuides/OmniScript/Functions/Functions/UTINDX.htm) | **External Indexed File Read and Write Access** | **OmniScript** |

# Omniscript Examples

Script -I

\* /\* Fetch all SSNs <59.06 in Non-Roth plans \*/

CALC SECTION

IF OCFILE1\_OPEN(NAME:'$FILE1' MODE:'INPUT')=0; /\* OPEN INPUT FILE \*/

DISPLAY 'OPEN FAILED INPUT FILE OCFILE1';

ABEND;

ENDIF;

IF OCFILE2\_OPEN(NAME:'$FILE2' MODE:'OUTPUT')=0; /\* OPEN OUTPUT FILE \*/

DISPLAY 'OPEN FAILED OUTPUT FILE OCFILE2';

ABEND;

ENDIF;

LOOP WHILE OCFILE1\_READ(INTO:TX110);/\*Area to read record into\*/

TX001=OCSUB(TX110 1 6); /\* Read Plan # from the text input file \*/

SOSCOBJ\_VIEW(PLAN:TX001);

LOOP WHILE SOSCOBJ\_NEXT();

TX002=SOSCOBJ\_DE(229); /\* Read the Roth Indicator \*/

IF (TX002!='1'); /\* If plan is Non-Roth \*/

PTPHOBJ\_VIEW(PLAN:TX001 PART:PTPHOBJ\_DE(007));

LOOP WHILE PTPHOBJ\_NEXT();

TX003=PTPHOBJ\_DE(007); /\* Read SSN \*/

TX004=PTPHOBJ\_DE(021); /\* Read SSN Status\*/

TX005=PTPHOBJ\_DE(050);/\*DoB\*/

WK001=OCDATE\_CURRENT(); /\* Returns the current System date \*/

WK002=OCTEXT\_TONUM(TX005);/\*Convert DoB to Numeric\*/

WK003=OCDATE\_DIFFYMD(WK002 WK001);/\*Return diff in YYMMDD\*/

TX007=WK003;

TX008=ocTEXT\_CONVERT(TX007 From:"/" To:".");

TX009=OCTEXT\_SET(TX008 3 '.' 1); /\*Inserts chars to a text string\*/

TX010=OCTEXT\_SET(TX008 5 '.' 1);

TX011=OCTEXT\_STRING(TX009 TX010); /\*Concatenate Text Strings into one\*/

TX012=OCTEXT\_SUB(TX011 1 4); /\*Return a portion of a string\*/

OCSHOW(TX009);

OCSHOW(TX010);

OCSHOW(TX011);

OCSHOW(TX012);

IF (TX003='00') AND (TX012<'59.06'); /\* Check for Active Status SSNs \*/

TX013=TX001+'|'+TX003+'|'+TX004+'|'+TX005+'|'+TX012+'|'+TX002;

OCFILE2\_WRITE(TX013);

ENDIF;

ENDLOOP;

ENDIF;

ENDLOOP;

ENDLOOP;

Script -II

\* /\* Fetch all SSNs <59.06 in Roth plans \*/

CALC SECTION

IF OCFILE1\_OPEN(NAME:'$FILE1' MODE:'INPUT')=0; /\* OPEN INPUT FILE \*/

DISPLAY 'OPEN FAILED INPUT FILE OCFILE1';

ABEND;

ENDIF;

IF OCFILE2\_OPEN(NAME:'$FILE2' MODE:'OUTPUT')=0; /\* OPEN OUTPUT FILE \*/

DISPLAY 'OPEN FAILED OUTPUT FILE OCFILE2';

ABEND;

ENDIF;

LOOP WHILE OCFILE1\_READ(INTO:TX110);/\*Area to read record into\*/

TX001=OCSUB(TX110 1 6); /\* Read Plan # from the text input file \*/

SOSCOBJ\_VIEW(PLAN:TX001);

LOOP WHILE SOSCOBJ\_NEXT();

TX002=SOSCOBJ\_DE(229); /\* Read the Roth Indicator \*/

IF (TX002='1'); /\* If plan is Roth \*/

PTPHOBJ\_VIEW(PLAN:TX001 PART:PTPHOBJ\_DE(007));

LOOP WHILE PTPHOBJ\_NEXT();

TX003=PTPHOBJ\_DE(007); /\* Read SSN \*/

TX004=PTPHOBJ\_DE(021); /\* Read SSN Status\*/

TX005=PTPHOBJ\_DE(050);/\*DoB\*/

WK001=OCDATE\_CURRENT(); /\* Returns the current System date \*/

WK002=OCTEXT\_TONUM(TX005);/\*Convert DoB to Numeric\*/

WK003=OCDATE\_DIFFYMD(WK002 WK001);/\*Return diff in YYMMDD\*/

TX007=WK003;

TX008=ocTEXT\_CONVERT(TX007 From:"/" To:".");

TX009=OCTEXT\_SET(TX008 3 '.' 1); /\*Inserts chars to a text string\*/

TX010=OCTEXT\_SET(TX008 5 '.' 1);

TX011=OCTEXT\_STRING(TX009 TX010); /\*Concatenate Text Strings into one\*/

TX012=OCTEXT\_SUB(TX011 1 4); /\*Return a portion of a string\*/

OCSHOW(TX009);

OCSHOW(TX010);

OCSHOW(TX011);

OCSHOW(TX012);

IF (TX003='00') AND (TX012<'59.06'); /\* Check for Active Status SSNs \*/

TX013=TX001+'|'+TX003+'|'+TX004+'|'+TX005+'|'+TX012+'|'+TX002;

OCFILE2\_WRITE(TX013);

ENDIF;

ENDLOOP;

ENDIF;

ENDLOOP;

ENDLOOP;

Script -III

\* /\* Fetch all SSNs >59.06 in plans that allow Inservice Wdrls \*/

CALC SECTION

IF OCFILE1\_OPEN(NAME:'$FILE1' MODE:'INPUT')=0; /\* OPEN INPUT FILE \*/

DISPLAY 'OPEN FAILED INPUT FILE OCFILE1';

ABEND;

ENDIF;

IF OCFILE2\_OPEN(NAME:'$FILE2' MODE:'OUTPUT')=0; /\* OPEN OUTPUT FILE \*/

DISPLAY 'OPEN FAILED OUTPUT FILE OCFILE2';

ABEND;

ENDIF;

LOOP WHILE OCFILE1\_READ(INTO:TX110);/\*Area to read record into\*/

TX001=OCSUB(TX110 1 6); /\* Read Plan # from the text input file \*/

BAUDOBJ\_VIEW(PLAN:TX001 PARTID:'000000000' BTTYPE:'UIDB');

LOOP WHILE BAUDOBJ\_NEXT();

TX002=BAUDOBJ\_DE(400);/\*Chk for Inservice Wdrl Plans\*/

TX003=OCSUB(TX003 1 1);

IF(TX003='1');/\* If plans allows Inservice Wdrl\*/

PLPLOBJ\_VIEW(PLAN:TX001);

LOOP WHILE PLPLOBJ\_NEXT();

TX004=PLPLOBJ\_DE(646); /\*spousal Consent Indicator\*/

TX005=OCSUB(TX004 7 1);

IF(TX005='0') OR (TX005='2') OR (TX005='3') OR (TX005='4') OR (TX005='5') OR (TX005='6') OR (TX005='7') OR (TX005='8');

PTPHOBJ\_VIEW(PLAN:TX001 PART:PTPHOBJ\_DE(007));

LOOP WHILE PTPHOBJ\_NEXT();

TX006=PTPHOBJ\_DE(007); /\* Read SSN \*/

TX007=PTPHOBJ\_DE(021); /\* Read SSN Status\*/

TX008=PTPHOBJ\_DE(050);/\*DoB\*/

WK001=OCDATE\_CURRENT(); /\* Returns the current System date \*/

WK002=OCTEXT\_TONUM(TX008);/\*Convert DoB to Numeric\*/

WK003=OCDATE\_DIFFYMD(WK002 WK001);/\*Return diff in YYMMDD\*/

TX009=WK003;

TX010=ocTEXT\_CONVERT(TX009 From:"/" To:".");

TX011=OCTEXT\_SET(TX010 3 '.' 1); /\*Inserts chars to a text string\*/

TX012=OCTEXT\_SET(TX010 5 '.' 1);

TX013=OCTEXT\_STRING(TX011 TX012); /\*Concatenate Text Strings into one\*/

TX014=OCTEXT\_SUB(TX013 1 4); /\*Return a portion of a string\*/

OCSHOW(TX010);

OCSHOW(TX011);

OCSHOW(TX012);

OCSHOW(TX013);

OCSHOW(TX014);

IF (TX003='00') AND (TX012>'59.06'); /\* Check for Active Status SSNs \*/

TX015=TX001+'|'+TX003+'|'+TX005+'|'+TX006+'|'+TX007+'|'+TX008+'|'+TX014

OCFILE2\_WRITE(TX013);

ENDIF;

ENDLOOP;

ENDIF;

ENDLOOP;

ENDIF;

ENDLOOP;

ENDLOOP;

Script -IV

\* /\* Fetch all SSNs which allows Loan Issue for Spousal consent \*/

CALC SECTION

IF OCFILE1\_OPEN(NAME:'$FILE1' MODE:'INPUT')=0; /\* OPEN INPUT FILE \*/

DISPLAY 'OPEN FAILED INPUT FILE OCFILE1';

ABEND;

ENDIF;

IF OCFILE2\_OPEN(NAME:'$FILE2' MODE:'OUTPUT')=0; /\* OPEN OUTPUT FILE \*/

DISPLAY 'OPEN FAILED OUTPUT FILE OCFILE2';

ABEND;

ENDIF;

LOOP WHILE OCFILE1\_READ(INTO:TX110);/\*Area to read record into\*/

TX001=OCSUB(TX110 1 6); /\* Read Plan # from the text input file \*/

PLPLOBJ\_VIEW(PLAN:TX001);

LOOP WHILE PLPLOBJ\_NEXT();

TX002=PLPLOBJ\_DE(646); /\*spousal Consent Indicator\*/

TX003=OCSUB(TX002 4 1);

TX004=PLPLOBJ\_DE(602); /\*Loan Application Fee\*/

TX005=OCSUB(TX002 7 1);

IF(TX003='3') AND (TX004<>' ') AND (TX005='0');/\* If plans allows Spousal Consent for Loan Issuance\*/

PTPHOBJ\_VIEW(PLAN:TX001);

LOOP WHILE PTPHOBJ\_NEXT();

TX008=PTPHOBJ\_DE(007);/\*SSN\*/

TX006=PTPHOBJ\_DE(633); /\* Spousal Consent Form Date\*/

TX007=OCSUB(TX006 11 8);

IF(TX007=' ');

LNLHOBJ\_VIEW(PLAN:TX001 PARTID:TX008);

LOOP WHILE LNLHOBJ\_NEXT();

TX009=LNLHOBJ\_DE(325);/\*Payroll Code\*/

IF(TX009='PDED');

TX010=TX001+'|'+TX003+'|'+TX004+'|'+TX005+'|'+TX008+'|'+TX007+'|'+TX009

OCFILE2\_WRITE(TX010);

ENDIF;

ENDLOOP;

ENDIF;

ENDLOOP;

ENDIF;

ENDLOOP;

ENDLOOP;

Script -V

\* /\* Fetch all SSNs with specific Loan Repayment codes & Frequency \*/

PARM SECTION

OR\_RPT\_MODE=CALC

OR\_DELIMITER='~'

OR\_AUTO\_PAGE\_BREAK=YES

OR\_PAGE\_LENGTH=50

TEXT SECTION

PLAN SSN# LH405(Loan Status) LH325(Payroll Code) LH318(Repmt Freq) FLAG>>CS,OR1=0 >>L1

--------------------------------------------------------------------->>CE

>TX1 >TX2 >TX3 >TX4 >TX5 >>CS,OR2=0>>CE

CALC SECTION.

OR\_AREA\_1\_ON;

SD080=099999;

PLPLOBJ\_VIEW();

LOOP WHILE PLPLOBJ\_NEXT();

TX001=PLPLOBJ\_DE(011);/\* Plan #\*/

PTPHOBJ\_VIEW();

LOOP WHILE PTPHOBJ\_NEXT();

TX002=PTPHOBJ\_DE(007);/\* Plan #\*/

LNLHOBJ\_VIEW(PLAN:TX001 PARTID:TX002)

LOOP WHILE LNLHOBJ\_NEXT();

TX003=LNLHOBJ\_DE(405); /\* Loan Status\*/

TX004=LNLHOBJ\_DE(325); /\* Loan Status\*/

TX005=LNLHOBJ\_DE(318); /\* Loan Status\*/

IF(TX003='0') AND (TX004='COUP') AND(TX005='3');

OR\_AREA\_2\_ON;

OR\_RPT\_GEN;

ENDIF;

ENDLOOP;

ENDLOOP;

ENDLOOP;

Div/Sub Scan

SD80=99999999;/\*To perform the number of loops\*/

IF OCFILE1\_OPEN(NAME:'$FILE1' MODE:'INPUT')=0; /\* OPEN OUTPUT FILE \*/

DISPLAY 'OPEN FAILED INPUT FILE OCFILE1';

ABEND;

ENDIF;

IF OCFILE2\_OPEN(NAME:'$FILE2' MODE:'OUTPUT')=0; /\* OPEN OUTPUT FILE \*/

DISPLAY 'OPEN FAILED OUTPUT FILE OCFILE2';

ABEND;

ENDIF;

LOOP WHILE OCFILE1\_READ(INTO:TX050);

TX001=OCSUB(TX050 1 6);

DSDSOBJ\_VIEW(PLAN:TX001);

LOOP WHILE DSDSOBJ\_NEXT();

TX002=DSDSOBJ\_DE(007);

TX003=TX001+':'+TX002;

OCFILE2\_WRITE(TX003);

ENDLOOP;

ENDLOOP;

SSN-BR Record Financial Extract Scan

IF OCFILE1\_OPEN(NAME:'$FILE1' MODE:'INPUT')=0; /\* OPEN INPUT FILE \*/

DISPLAY 'OPEN FAILED INPUT FILE OCFILE1';

ABEND;

ENDIF;

IF OCFILE2\_OPEN(NAME:'$FILE2' MODE:'OUTPUT')=0; /\* OPEN OUTPUT FILE \*/

DISPLAY 'OPEN FAILED OUTPUT FILE OCFILE2';

ABEND;

ENDIF;

LOOP WHILE OCFILE1\_READ(INTO:TX050);/\*Area to read record into\*/

TX001=OCSUB(TX050 1 6);/\*return a selected segment of a text string in this case plan#\*/

PTPHOBJ\_VIEW(PLAN:TX001);

LOOP WHILE PTPHOBJ\_NEXT();

TX011=PTPHOBJ\_DE(007);

HIBROBJ\_VIEW(PLAN:TX001 PARTID:TX011);

LOOP WHILE HIBROBJ\_NEXT();

TX003=HIBROBJ\_DE(100);/\*FUND ID\*/

TX004=HIBROBJ\_DE(101);/\*Transaction Code\*/

TX005=HIBROBJ\_DE(102);/\*Activity\*/

TX012=HIBROBJ\_DE(110);/\*Amount\*/

TX007=HIBROBJ\_DE(121);/\*Shares\*/

WK008=HIBROBJ\_DE(008);/\*Trade Date\*/

TX008=OCFMT(WK008 'DATE4');

WK009=HIBROBJ\_DE(009);/\*Run Date\*/

TX009=OCFMT(WK009 'DATE4');

TX010=HIBROBJ\_DE(008);/\*Run Time\*/

TX014=HIBROBJ\_DE(007);/\*SSN\*/

IF (TX004='165');

TX013=TX001+'|'+TX011+'|'+TX004+'|'+TX005+'|'+TX003+'|'+TX012+'|'+TX007+'|'+TX008+'|'+TX009+'|'+TX010;

OCSHOW(TX013);

OCFILE2\_WRITE(TX013);

ENDIF;

ENDLOOP;

ENDLOOP;

ENDLOOP;

\*\*\*\*\*\*\* script to update Participant Gender \*\*\*\*\*\*\*\*\*\*\*

SD080=99999999;

IF OCFILE1\_OPEN(NAME:'$FILE1' MODE:'INPUT')=0; /\* OPEN INPUT FILE \*/

DISPLAY 'OPEN FAILED INPUT FILE OCFILE1';

ABEND;

ENDIF;

LOOP WHILE OCFILE1\_READ(INTO:TX110);/\*Area to read record into\*/

TX001=OCSUB(TX110 1 6);/\*return a selected segment of a text string,PLAN#\*/

TX002=OCSUB(TX110 8 9);/\* SSN\*/

TX003=OCSUB(TX110 18 1);/\*Gender\*/

WK001=OCNUM\_VALUE(TX003);

PTPHOBJ\_GET(PLAN:TX001 PARTID:TX002);

PTPHOBJ\_SETDE(DENUM:030 VALUE:WK001);

PTPHOBJ\_UPDATE();

ENDLOOP;