

Fast Formula, the Key to Simplifying Your User's Life

Patricia Keeley
BOSS Corporation

Introduction:

Fast Formula is a powerful tool. While there are many types of Fast Formula, today we will be discussing the use of the Fast Formula Type "Payroll Calculation."

- We'll discuss relationships between Fast Formula, Formula Results, Global Values, Utility Application Lookups (Quick Codes for those of you not on 11i) and Input Values.
- We'll begin with a quick evaluation of the Seeded Elements and the related Fast Formulas.
- We'll develop a simple deduction element that has both a percentage and a flat amount that can be withheld.
- We'll develop a bonus element that will generate multiple types of bonuses that can be costed differently.
- We'll develop a pretax refund element that will generate multiple non-recurring elements, including the option of updating an Employer Match, using one element.

While this is may be a complex setup, it will eliminate the need for your users to remember when they have to use one element versus another for the "same" deduction or "type" of Earnings. Your users will love you!

The Basics:

Fast Formulas are combinations of *statements* and *comments*. Statements are instructions telling Oracle Fast Formula how to process *constants* and *variables* that are the units in a formula. The several types of statements, that describe the formula's calculations, they are:

1. The Alias statement;
2. The Default statement;
3. The Assignment statement;
4. The Inputs statement;
5. The If statement; and
6. The Return statement.

These statements can include *expressions* that manipulate constants and variables using *arithmetic operators* and *functions*. The operators and functions you can use and the results they give depend upon the data type of the constants and variables. Other 'If' statements, as well as 'Assignment' and 'Return' statements can be included within a single nested 'If' statement. In the following pages are examples with a brief analysis of seeded, generated and custom formulas. It is my intention, that by providing you with an understanding of Fast Formula, you will begin to realize the wonderful tool you have with Fast Formulas.

Application Utility Lookup Tables:

Oracle provides many seeded Lookup tables, but you can create Lookup tables for your organization's unique uses. The Lookup tables can be used to create a list of values for your elements. This list of values can prevent invalid entries that may cause a formula to error. When creating custom formulas, consider the use of Lookup tables. In the following pages, I have provided examples of two custom Lookup tables.

Global Values:

Global Values should be used whenever your Fast Formula requires a 'fixed value.' You do not want to be looking for this value hidden within your formula, when a change occurs. Even when your user tells you the value will never change. It may only change once in a Blue Moon, but know it can and most likely will change in the future. Using the Global Values table, the user can update these values as they change. The Fast Formula references the Global Value name and will process with no changes to the Fast Formula.

Global Values are date tracked, so a value will not take affect until the date specified, another great reason for using Global Values.

User Tables:

Oracle provides many seeded User Tables, but you can create custom user tables for your organization’s unique uses by navigating to Table Structures form and applying the values as required via the Table Values form. So, why would you want to create a new table? Consider using a custom table if you require fixed values based on different conditions or different default values based on different employee groups. A perfect example of this would be hourly rates based on a Union Local and Apprentice level. If you have to create several links with a variety of eligibility rules, you may want to consider the use of Tables to identify the valid variables and let the Fast Formula use the table to establish eligibility.

Earnings and Deduction Templates:

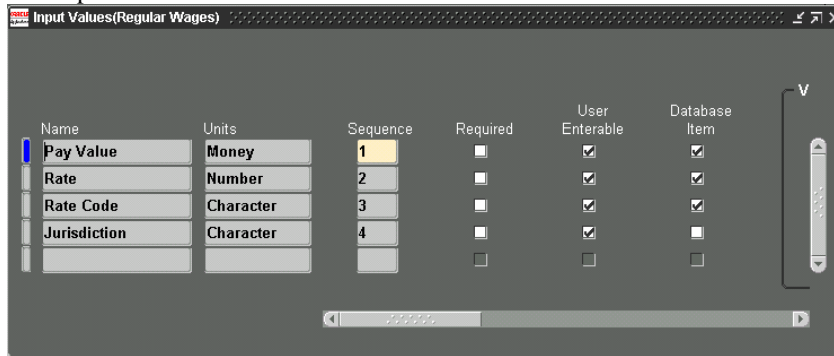
The US Oracle Payroll module supplies templates for creating earnings and deduction elements. These templates use the values you provide with seeded Fast Formulas, formula results, and views to generate the unique elements, Fast Formula, and input values needed to support your requirements. Note, that in addition to the base element the template generates two additional elements “Special Inputs” and “Special Features.” These elements are generated for all elements created by using the templates, whether or not your element requires these two additional elements. The client can modify all the generated elements, values and formulas. By modifying the generated element input values and the Fast Formula you can create an element that will meet all the needs of your organization. You are limited only by your own imagination and creativity.

Seeded Elements:

The US Oracle Payroll module supplies seeded elements. Oracle does not allow the client to modify any of the seeded elements, but I find it helpful to understand how the formulas for these elements function. Once you understand how the formula functions, you may copy the formula and use it as a base for custom elements.

1. Regular Wages

- **Input Values**



- **Fast Formula**

```

/* *****
$Header: pelegus1.dat 115.33 2000/05/19 15:41:57 pkm ship $
FORMULA NAME: REGULAR_WAGES
FORMULA TYPE: Payroll
Change History
29 Sep 1993 hparicha Created.
14 Oct 1993 hparicha Moved skip rule formulae to separate
file ".SKIP".
30 Nov 1993 hparicha G187

```

Text between /* and */ are comments. At the beginning of all seeded and generated formulas are comments from the Oracle developers who originally wrote or modified the Fast Formula. This is an attempt by the developers to inform the client of the use of the formula and the intentions of the developer when writing or modifying the formula. This is always a good place to start when attempting to determine what the main purpose of the formula.

06 Dec 1993 jmychale G305. Added aliases, entered database items and tidied up formula

07 Jan 1994 jmychale G491. Removed actual hours worked parameter from Calculate_Period_Earnings ()

13 Jan 1994 hparicha G497. Reverted calc period earnings to use ACTUAL_HOURS_WORKED! Used by Statement of Earnings report. Replaced TIME_ENTRY_WAGES_COUNT with LABOR_RECORDING_COUNT for timecard req'd employees.

24 Feb 1994 hparicha G560. ASS -> ASG; TU -> GRE

24 Feb 1994 hparicha G581. Handles negative earnings.

09 Jun 1994 hparicha G907. New implementation of generated and startup earnings and deductions using "<ELE_NAME> Special Features" shadow element to feed balances and handle Addl/Repl Amounts.

04 Jan 1995 hparicha Vacation/Sick correlation to Regular Pay. New results need to be passed when vac/sick pay are present.

04 May 1995 hparicha Defaulted values for PAY_PROC_PERIOD_START/END_DATE. Default dates should be obvious when default is used.

10 Jan 1996 hparicha 323639 Major cleanup effort involving proration and other user defined functions and formulae changes.

16 Apr 1996 ssdesai Latest balances creation.

25 Apr 1996 hparicha 344018 Added check for USER_ENTERED_TIME.

17 Apr 1996 djeng Changed PAY_PROC_PERIOD_START_DATE, and PAY_PROC_PERIOD_END_DATE to PAY_EARNED_START_DATE and

PAY_EARNED_END_DATE

13 AUG 1997 Lwhtomps BUG 525859. Payments dimension can not be held as a latest balance.

--

INPUTS: Rate
Rate Code (text)

--

DBI Required: ASG_SALARY_BASIS
TERMINATED_EMPLOYEE
FINAL_PAY_PROCESSED
LABOR_RECORDING_COUNT
SCL_ASG_US_WORK_SCHEDULE
ASG_HOURS
SCL_ASG_US_TIMECARD_REQUIRED

DESCRIPTION:

Computes earnings per pay period for hourly employees. Proration function must be available to determine if employee worked entire period - earnings will be adjusted accordingly by proration fn to account for new hire, termination, leave of absence, etc.

*** Hourly handling ***

Regular wages earned per pay period for employees paid by the hour. Hourly employees can either be "Hourly-Automatic" (ie. timecard not required) or "Hourly-Timecard" where a timecard is required for pay. The hourly rate for an employee is entered as the

input value for this element. This rate is used with the number of hours worked to calculate earnings.

Hours worked will be indicated by one of the following:

- time entry or entries (ie. timecard)
- ASGigned Work Schedule
- standard hours entered at the Organization and ASGignment levels.

For an Hourly-Timecard or "timecard required" employee, when a timecard is not submitted by the payroll input cutoff date - the wages for that employee will not be calculated and will have to wait for a subsequent payroll run for processing. When a timecard is submitted for an Hourly-Automatic employee, then the time entry (or entries) is treated as the source for Hours - if a rate is entered on the time entry, then this rate is used along with the hours to compute pay, otherwise the normal rate (Regular Wages rate) is used for computation. If this is the Final Pay run for the employee's ASGignment, then the Regular Wages element will be discontinued after this run.

ALGORITHM:

If timecard required and time entries NOT FOUND, then message='No timecards entered for Hourly, Timecard Required employee.'

```

return message
-- NOTE: If tc was req'd and time entries WERE found - then the skip rule
-- for this Regular Salary element would have skipped this processing.
Endif

```

```

Call proration function with hourly rate; --> Regular_Wage_Earnings
If this is final pay for employee(ASGignment), then discontinue further processing of this
element -- This is last time.
endif

```

```

Return Regular_Wage_Earnings
-- The earnings calculation for hourly employees will primarily be calculated by the
calculate_period_earnings() function. This function has its' own hld (in Work Schedules
Functionality doc) and lld to be called calc_period_earnings.lld
*****

```

FORMULA_TEXT:

*****/

```

/* Alias Section */
ALIAS SCL_ASG_US_WORK_SCHEDULE AS Work_Schedule
ALIAS SCL_ASG_US_TIMECARD_REQUIRED AS Timecard_Required

```

Alias statements allow the user to use an alternate name for Database Item and Global Value names.

```

/* dbi defaults */
DEFAULT FOR ASG_SALARY_BASIS IS 'NOT ENTERED'
DEFAULT FOR TERMINATED_EMPLOYEE IS 'N'
DEFAULT FOR FINAL_PAY_PROCESSED IS 'N'
DEFAULT FOR LABOR_RECORDING_COUNT IS 0
DEFAULT FOR USER_ENTERED_TIME IS 'Y'
default for PAY_PROC_PERIOD_START_DATE is '01-JAN-0001' (DATE)
default for PAY_PROC_PERIOD_END_DATE is '02-JAN-0001' (DATE)
default for PAY_EARNED_START_DATE is '01-JAN-0001' (DATE)
default for PAY_EARNED_END_DATE is '02-JAN-0001' (DATE)
DEFAULT FOR Work_Schedule IS 'NOT ENTERED'
DEFAULT FOR ASG_HOURS IS 0
DEFAULT FOR ASG_FREQ IS 'NOT ENTERED' /* WWBug 323639 */

```

Default statements are required for Database Items with the default indicator equal to 'Yes.'

```

DEFAULT FOR Timecard_Required IS 'N'
DEFAULT FOR REGULAR_WAGES_NEG_EARNINGS_ASG_GRE_ITD IS 0
DEFAULT FOR REGULAR_WAGES_ADDITIONAL_ASG_GRE_ITD IS 0
DEFAULT FOR REGULAR_WAGES_REPLACEMENT_ASG_GRE_ITD IS 0
DEFAULT FOR REGULAR_WAGES_ASG_GRE_RUN IS 0
DEFAULT FOR REGULAR_HOURS_WORKED_ASG_GRE_RUN IS 0
/* input defaults */
DEFAULT FOR Rate IS 0
DEFAULT FOR Rate_Code (text) IS 'NOT ENTERED'

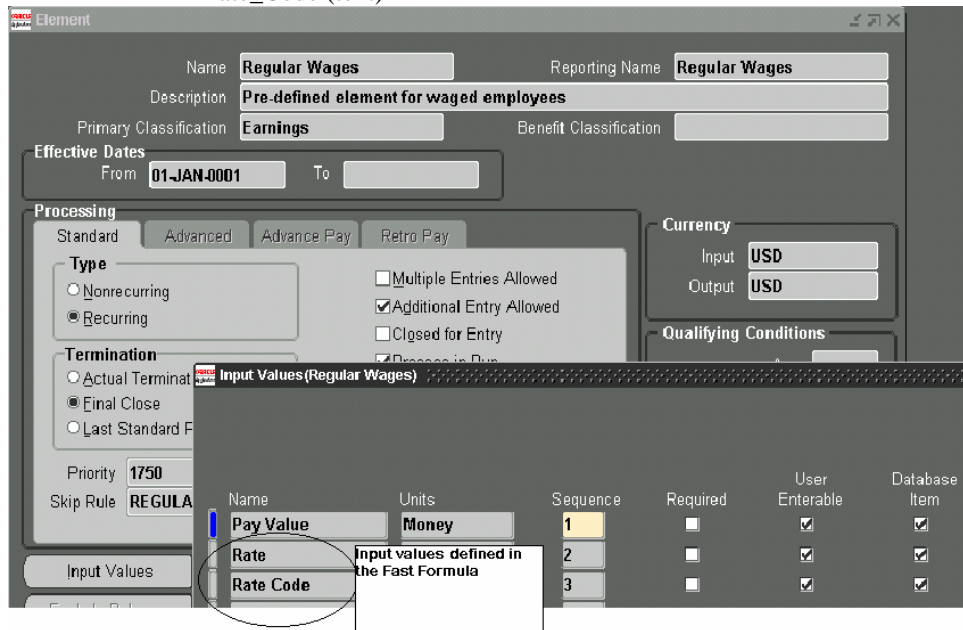
```

```

/* Input Values */
INPUTS ARE Rate,
Rate_Code (text)

```

Input statements identify input values used by the element

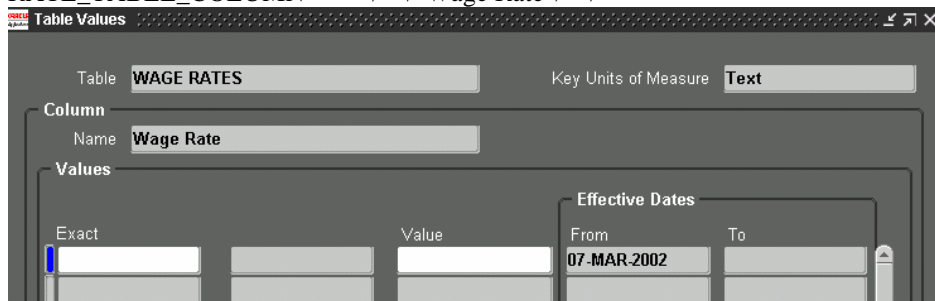


```

/* Updatable Values */
RATE_TABLE = /**/ 'WAGE RATES' /**/
RATE_TABLE_COLUMN = /**/ 'Wage Rate' /**/

```

This allows the formula to get its values from the Wage Rates table when values are provided. To provide values for use with Regular Wages, navigate to Other Definitions → Table Values.



If the employee is terminated, the formula returns the Stop Entry value that results in end dating the element. The formula stops processing for terminated employees at this point.

```
IF TERMINATED_EMPLOYEE = 'Y' AND USER_ENTERED_TIME = 'Y' THEN  
(STOP_ENTRY = 'Y'  
  msg = 'Regular Wages being stopped after Final Pay.'  
  RETURN STOP_ENTRY, msg  
)
```

If the employee is Time Card Required and no values were entered for Labor, the formula returns only a message. The formula stops processing for these employees at this point.

```
IF Timecard_Required = 'Y' AND LABOR_RECORDING_COUNT = 0 THEN  
(msg = 'No timecards entered for Hourly, Timecard Required employee'  
  RETURN msg  
)
```

The formula checks for a replacement amount entered via the Special Inputs element. If no value is found the formula proceeds to determine the wage amount. Note the next 'If' statement and the open parenthesis. If the replacement value was entered, the formula jumps to the next 'If' statement after the closed parenthesis.

```
IF REGULAR_WAGES_REPLACEMENT_ASG_GRE_ITD WAS DEFAULTED OR  
REGULAR_WAGES_REPLACEMENT_ASG_GRE_ITD = 0  
THEN
```

If the replacement value was not entered, the formula will check for the rate being entered. If the rate was entered as an input value, the formula proceeds to set several 'local variable' defaults to 0 and precedes to calculate the earnings based on scheduled hours or actual hours worked.

```

IF Rate WAS NOT DEFAULTED THEN
(hourly_rate = Rate
actual_hours_worked = 0
t_vac_hours_taken = 0
t_vac_pay = 0
t_sick_hours_taken = 0
t_sick_pay = 0
regular_wage_earnings = Calculate_Period_Earnings(
    ASG_SALARY_BASIS,
    'RATE',
    hourly_rate,
    PAY_EARNED_START_DATE,
    PAY_EARNED_END_DATE,
    Work_Schedule,
    ASG_HOURS,
    actual_hours_worked,
    t_vac_hours_taken,
    t_vac_pay,
    t_sick_hours_taken,
    t_sick_pay,
    'Y',
    ASG_FREQ) /* WWBug 323639 */
)

```

If the replacement value was not entered, the formula will check for the rate being entered. If the rate was not entered as an input value, the formula checks for the Rate Code. If the Rate Code was entered the formula proceeds to set several 'local variable' defaults to 0 and precedes to calculate the earnings based on scheduled hours or actual hours worked.

```

ELSE
IF Rate_Code WAS NOT DEFAULTED THEN
(hourly_rate = To_Number(Get_Table_Value( RATE_TABLE,
    RATE_TABLE_COLUMN,
    Rate_Code))
actual_hours_worked = 0
t_vac_hours_taken = 0
t_vac_pay = 0
t_sick_hours_taken = 0
t_sick_pay = 0
regular_wage_earnings = Calculate_Period_Earnings(
    'HOURLY',
    'RATE CODE',
    hourly_rate,
    PAY_EARNED_START_DATE,
    PAY_EARNED_END_DATE,
    Work_Schedule,
    ASG_HOURS,
    actual_hours_worked,
    t_vac_hours_taken,
    t_vac_pay,
    t_sick_hours_taken,
    t_sick_pay,
    'Y',
    ASG_FREQ) /* WWBug 323639 */
)

```

If the replacement value was not entered, the formula will check for the rate and the rate code being entered. If the neither the rate or the rate code was not entered as an input value, the formula will exit at this point and return a message only.

```
ELSE
(mesg = 'No Hourly Rate or Rate Code entered for this employee'
RETURN mesg
)
```

If a replacement value was entered, the formula will use the replacement value as the Regular Wage Earnings.

```
ELSE
(regular_wage_earnings = REGULAR_WAGES_REPLACEMENT_ASG_GRE_ITD
clear_repl_amt = -1 * REGULAR_WAGES_REPLACEMENT_ASG_GRE_ITD
/* WWBug 323639 */
actual_hours_worked = 0
t_vac_hours_taken = 0
t_vac_pay = 0
t_sick_hours_taken = 0
t_sick_pay = 0
)
```

Here the formula is taking the previously calculated results and adding the Special Inputs 'Additional Amount'

```
regular_wage_earnings = regular_wage_earnings
+ REGULAR_WAGES_ADDITIONAL_ASG_GRE_ITD
+ REGULAR_WAGES_NEG_EARNINGS_ASG_GRE_ITD
```

The formula is checking for negative values. If the earnings is negative, the formula will check termination status and final pay status. Negative earnings are put into a "Regular Wages Neg Earnings" balance. If the balance is not zero, the formula multiplies the balance by a negative 1, making the result a positive number. **Note:** this may be where you encounter issues if you are attempting to minus regular wages and add them to another earnings.

```

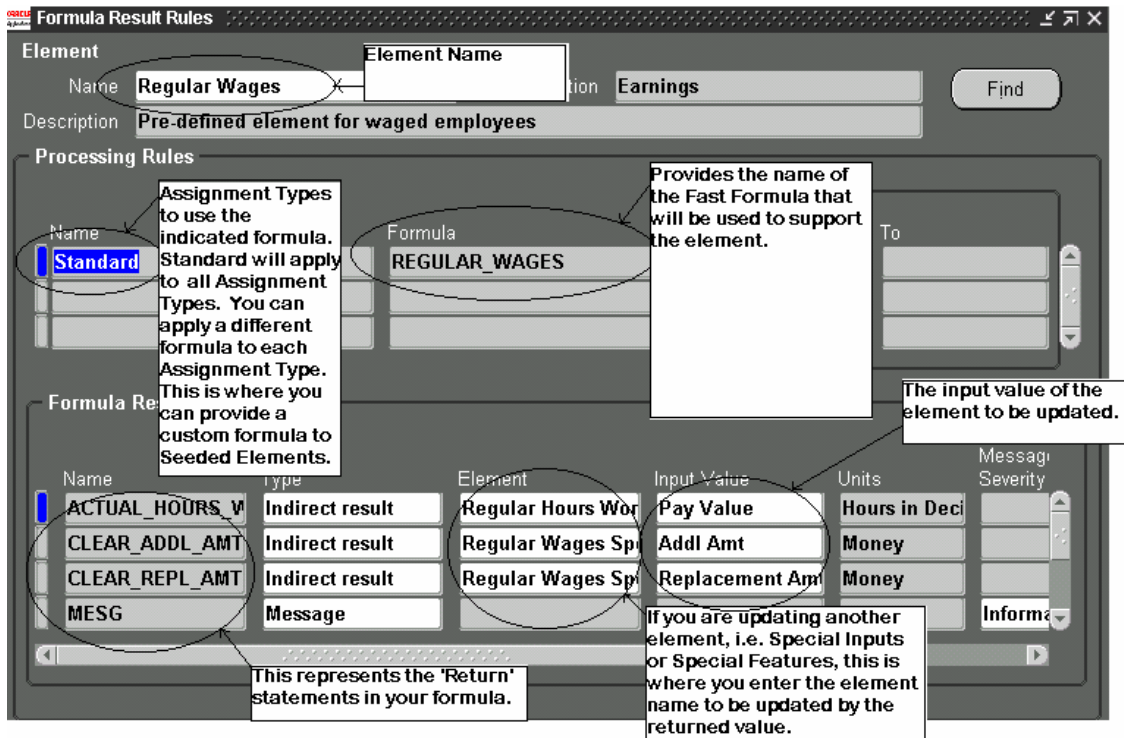
IF regular_wage_earnings < 0 THEN
(IF TERMINATED_EMPLOYEE = 'Y' AND FINAL_PAY_PROCESSED = 'N' THEN
  neg_earn = 0
  ELSE
  (neg_earn = regular_wage_earnings
  regular_wage_earnings = 0
  )
)
)
ELSE
(IF REGULAR_WAGES_NEG_EARNINGS_ASG_GRE_ITD <> 0 THEN
  neg_earn = -1 * REGULAR_WAGES_NEG_EARNINGS_ASG_GRE_ITD
  )
)
IF REGULAR_WAGES_ADDITIONAL_ASG_GRE_ITD <> 0 THEN
  clear_addl_amt = -1 * REGULAR_WAGES_ADDITIONAL_ASG_GRE_ITD
  IF t_vac_pay <> 0 THEN
  (vac_pay = t_vac_pay
  vac_hours_taken = t_vac_hours_taken
  regular_wage_earnings = regular_wage_earnings - vac_pay
  actual_hours_worked = actual_hours_worked - vac_hours_taken
  )
)
IF t_sick_pay <> 0 THEN
  (sick_pay = t_sick_pay
  sick_hours_taken = t_sick_hours_taken
  regular_wage_earnings = regular_wage_earnings - sick_pay
  actual_hours_worked = actual_hours_worked - sick_hours_taken
  )
)
/* Create latest balances */
  soe_run = REGULAR_WAGES_ASG_GRE_RUN
  soe_ytd = REGULAR_WAGES_ASG_GRE_YTD
  soe_hrs = REGULAR_HOURS_WORKED_ASG_GRE_RUN
  IF TERMINATED_EMPLOYEE = 'Y' AND FINAL_PAY_PROCESSED = 'N' THEN
  ( STOP_ENTRY = 'Y'
  RETURN          regular_wage_earnings, actual_hours_worked, clear_repl_amt,
  clear_addl_amt, neg_earn, vac_pay, vac_hours_taken, sick_pay, sick_hours_taken,
  STOP_ENTRY
  )
)
ELSE
  RETURN          regular_wage_earnings, actual_hours_worked, clear_repl_amt,
  clear_addl_amt, neg_earn, vac_pay, vac_hours_taken, sick_pay, sick_hours_taken
  /***** End of Formula *****/

```

If the employee is terminated and this is not the final pay and none of the previous conditions have been met the formula returns a Stop Entry equal to 'Yes' and the earnings values. The element will not process in future payrolls.

This is the final 'Return' statement.

- **Formula Results**
This is where it all comes together. It is with the Formula Results window that you provide the formula name to be applied to the element.

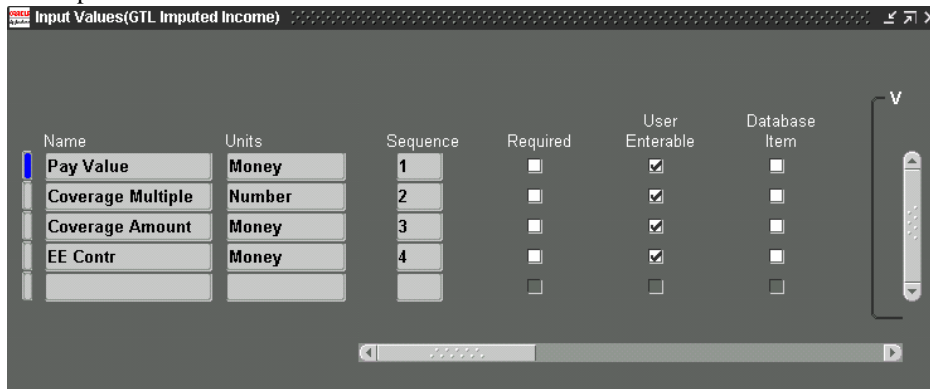


Seeded elements use the name “Standard” value in the name field under the Processing Rules. If your organization is committed to the use of Regular Wages due to configuration choices, you can create a new formula for Regular Wages and by using the Name “Active Employee” (or other names – as it applies to your organization) the element will recognize your formula in place of the seeded formula. DO NOT DELETE the “Standard” name from the Processing Rules.

Note: Using Messages in the formula, can be a great way to debug your formula. Within each “If” statement include a “Return” statement with a message. Number each message so that it is unique for each “If” statement. Add each “message” to the Formula Results. Remember to comment or delete your message statements from your formula before you put it into production.

2. GTL Imputed Income

- Input Values



- Fast Formula

```

/*****
-- FORMULA NAME: GTL IMPUTED INCOME
--
-- DESCRIPTION: Computes imputed income from Group Term
-- Life Insurance Premiums.

```

Comments with explanation of the intention of the formula by the developers.

-- Imputed Income = cost of coverage over
 -- \$50,000 as determined by IRS Uniform
 -- Premiums -Table I.
 --
 -- IRS UNIFORM PREMIUMS TABLE: held in user-defined table
 -- called GTL_PREMIUMS.
 -- Function calculate_premium(\$\$, age) returns
 -- cost of monthly premium for given amount of coverage.
 -- Imputed earnings is determined by calculating cost of monthly
 -- premiums for amount of coverage over \$50000, then subtracting
 -- employee's monthly contribution.

Calculates Group Term Life Insurance premiums to determine imputed earnings as appropriate. Current law determines that the cost (premiums) or less than \$50000 is exempt from taxation. The premiums for coverage over \$50000 are taxable. The cost of coverage is determined by the IRS Uniform Premiums Table (for GTL).

Imputed Income is derived by taking the amount of coverage over \$50000 (ie. \$25000 for coverage of \$75000), and figuring the cost of that "taxable" coverage from the table. The table displays premiums per \$1000 of coverage for 1 month, by age group. So in this case, 25 times the cost of coverage found on the table will be included in the employee's earnings (ie. imputed income). The amount of imputed income can be reduced by employee after-tax contributions to GTL premiums. This imputed income is not subject to FIT Withholding and FUTA; however, imputed income is subject to FICA and various state taxes as mandated by each state. GTL functionality provided with the system will properly handle taxation of GTL imputed income.

The GTL process begins with determining the cost of coverage (ie. The [monthly] premium). From this calculation, an imputed income amount may be present. Any imputed income will be reduced by the employee contribution.

Formula Description:

- 1) Check if the amount of coverage exceeds \$50000.
 If not, then no imputed
 income calculation is necessary (so formula will only return amount of GTL
 Premium as indirect result. This will ensure that GTL imputed income
 will only appear on
 paystub when necessary.
- 2) Find amount of coverage: usually a multiple of annual salary.
 Figure used for coverage is rounded to ...
 current tax year, ie. 31 Dec);
- 3) Reduce monthly premium amount by monthly employee contribution;
- 4) If imputed income exists, reduce imputed income by monthly EE Contr
 (this the EE Contr per \$1,000 of coverage).
- 5) Convert imputed income to per-pay-period amount.
- 6) Return Imputed Income (if necessary) and GTL EE Portion
 (if necessary).

Formula Result Rules

Direct Result: GTL_Imputed
 Indirect Result: feeds Pay Value of GTL EE Contribution (Vol Ded) element.

Change History

15 Oct 93 hparicha Created.
 14 Dec 1993 jmychale G432. Major revision. Removed ER indirect
 result.

19 Jan 1994 hparicha Replaced PER_AGE with PER_END_OF_YEAR_AGE db item.
 24 Feb 1994 hparicha G560. ASS -> ASG; TU -> GRE
 23 Jun 1994 hparicha Improved handling of EE Contr and reducing monthly imputed amount.

 FORMULA TEXT:
 ***** */

/* Alias Section */

ALIAS SCL_ASG_US_WORK_SCHEDULE AS Work_Schedule

The formula begins with the Alias statement where the developer is replacing the database item name SCL_ASG_US_WORK_SCHEDULE with the name 'work_schedule.'

Default Statements

DEFAULT FOR Coverage_Multiple IS 1
 DEFAULT FOR Coverage_Amount IS 0
 DEFAULT FOR EE_Contr IS 0
 DEFAULT FOR Work_Schedule IS 'NOT ENTERED'
 DEFAULT FOR TERMINATED_EMPLOYEE IS 'N'
 DEFAULT FOR FINAL_PAY_PROCESSED IS 'N'
 DEFAULT FOR PER_END_OF_YEAR_AGE IS 0
 DEFAULT FOR ASG_HOURS IS 0
 DEFAULT FOR ASG_SALARY IS 0
 DEFAULT FOR ASG_SALARY_BASIS IS 'NOT ENTERED'
 DEFAULT FOR PAY_PROC_PERIOD_DATE_PAID IS '01-JAN-0001' (date)

/* Inputs : note that the EE_Contr input holds the monthly after-tax contribution by the employee in \$ per \$1,000 of Coverage elected */

Input Statement

INPUTS ARE Coverage_Multiple,
 Coverage_Amount,
 EE_Contr

/*
 Outputs are GTL_Imputed (direct result if necessary),
 GTL_EE_Contr (indirect result, voluntary deduction if necessary)
 */

Setting local variables

GTL_TABLE = /**/ 'GTL PREMIUMS' /**/
 GTL_TABLE_COLUMN = /**/ 'Age Range' /**/
 Max_Non_Imputed = /**/ 50000.00 /**/

If the input values are blank or zero the formula will return a message only and exit the routine. If the Coverage Amount is blank but the multiple is not blank, the formula will calculate the annual compensation based on the current amount/rate on the salary record times the number of pay periods based on the period type. It then proceeds to determine the coverage amount by using the calculated salary times the multiple and rounded to the nearest thousand.

```
IF Coverage_Amount WAS DEFAULTED THEN
IF Coverage_Multiple WAS DEFAULTED THEN
(mesg = 'No GTL coverage amount or multiple was entered for this employee'
RETURN mesg )
```

```
ELSE
```

```
/* Convert the salary as identified by the Salary Admin Basis to
an Annual Salary */
```

```
(Annual_Salary = Convert_Period_Type ( Work_Schedule,
ASG_HOURS,
ASG_SALARY,
ASG_SALARY_BASIS,
'YEAR' )
```

```
/* To closest thousand. */
```

```
coverage = ROUND( Coverage_Multiple * Annual_Salary, -3)
)
```

```
ELSE
```

```
coverage = ROUND(Coverage_Amount, -3) /* To nearest thousand */
```

If coverage amount was entered the formula will use the amount entered rounded to the nearest thousand.

Once the coverage amount has been calculated the formula attempts to find the premium amount from the GTL Table by using the coverage amount and the age of the employee on the pay date. If no match is found the local variables are set to zero. If a match is found the formula attempts determine the monthly-imputed amount. If the imputed amount is greater than zero the formula now determines the amount of imputed income for the pay period. The amount is reduced by the amount of the employee contribution, if any. If the employee contribution equals the imputed amount, the imputed amount is set to zero.

```

/* Calculate gtl premium and imputed income (if necessary) */
prem_per_thousand = get_table_value (
    GTL_TABLE,
    GTL_TABLE_COLUMN,
    to_char(PER_END_OF_YEAR_AGE),
    PAY_PROC_PERIOD_DATE_PAID)

IF prem_per_thousand = 'NO_DATA_FOUND' THEN
    (GTL_Imputed = 0
    Monthly_Imputed = 0
    )
ELSE
/* Calculate the Monthly Imputed */
    Monthly_Imputed = TO_NUMBER(prem_per_thousand) *
        (coverage - Max_Non_Imputed) / 1000

IF Monthly_Imputed > 0 THEN

/* Calculate the imputed earnings for a payroll period */

IF EE_Contr WAS DEFAULTED THEN

    GTL_Imputed = Convert_Period_Type ( Work_Schedule,
        ASG_HOURS,
        Monthly_Imputed,
        'Calendar Month',
        'PERIOD')

ELSE
    (
    Reduced_Monthly_Imputed = Monthly_Imputed - EE_Contr
    IF Reduced_Monthly_Imputed > 0 THEN
        GTL_Imputed = Convert_Period_Type ( Work_Schedule,
            ASG_HOURS,
            Reduced_Monthly_Imputed,
            'Calendar Month',
            'PERIOD')

    ELSE
        GTL_Imputed = 0

    GTL_EE_Contr = Convert_Period_Type ( Work_Schedule,
        ASG_HOURS,
        EE_Contr,
        'Calendar Month',
        'PERIOD')

    )

/* Final Pay Section */

IF TERMINATED_EMPLOYEE = 'Y' AND FINAL_PAY_PROCESSED = 'N' THEN
    STOP_ENTRY = 'Y'

RETURN GTL_Imputed, GTL_EE_Contr, STOP_ENTRY
/*****End of Formula*****/

```

Final return statement.

3. State Tax Levy

The Oracle Payroll Module provides five seeded involuntary deduction elements. These elements DO NOT check state or federal legislative limits and all have the same input values. By understanding the formula, you can determine if these elements can be applied to your organization's involuntary deduction requirements.

- Input Values

Name	Units	Sequence	Required	User Enterable	Database Item	Val
Amount	Money	2	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	D
Total Owed	Money	3	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Guaranteed Net	Money	4	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Clear Arrears	Character	5	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Court Order	Character	6	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

In addition to the input values displayed here, is Pay Value that is also 'User Enterable.' Whenever 'Pay Value' is entered, that value will override anything that would be generated by the Fast Formula.

- Fast Formula

/******
 FORMULA NAME: STATE_TAX_LEVIES

DESCRIPTION: Contains formula to handle State Tax Levies.

Change History

17 Jan 94 hparicha Created from SYSTEM_DEDN_CALC_FORMULA
 02 Feb 94 hparicha G542. Changed input value names; added Replacement/Additional Amounts.
 24 Feb 94 hparicha G560. ASS -> ASG; TU -> GRE
 31 May 94 hparicha G815 (Completes G564). Period Type segment added to Involuntary Deductions.
 09 Jun 1994 hparicha G907. Neg Earnings, Addl/Repl amounts; Redesign of Earnings.
 24 Oct 1994 hparicha Benchmark changes - do not populate to_arrears and not_taken amounts if not necessary.
 21 Dec 1994 hparicha G1681: Clear/Adjust arrears fn'ality.
 28 Aug 1995 hparicha On final pay, try and collect remainder of levy. If NET not sufficient to collect remaining balance, then deduct amount calculated via arrearage function - ie. guaranteed net has already been considered!

NB: The handling of all Wage Attachments will be changing...

FORMULA TEXT

Formula Results :

Developer's explanation and intention of the formula.

```

dedn_amt      Direct Result for Deduction Amount
not_taken     Indirect to Special Features - Not Taken
to_arrears    Indirect to Special Features - Arrears Contr
set_clear     Update recurring entry for Clear Arrears
mesg          Fatal Error Message
clear_repl_amt Indirect to Special Features - Replacement Amount
clear_addl_amt Indirect to Special Features - Additional Amount
STOP_ENTRY    Stop current recurring entry

```

*****/

```

/* Database Item Defaults */
default for STATE_TAX_LEVIES_PARTIAL_EE_CONTRIBUTIONS is 'NOT
ENTERED'
default for STATE_TAX_LEVIES_ARREARS_ASG_GRE_ITD      is 0
default for STATE_TAX_LEVIES_ASG_GRE_ITD              is 0
/*
default for STATE_TAX_LEVIES_PERIOD_TYPE              is 'NOT ENTERED'
*/
DEFAULT FOR STATE_TAX_LEVIES_ADDITIONAL_ASG_GRE_ITD  IS 0
DEFAULT FOR STATE_TAX_LEVIES_REPLACEMENT_ASG_GRE_ITD IS 0

default for TERMINATED_EMPLOYEE                       is 'N'
default for FINAL_PAY_PROCESSED                       is 'N'
default for NET_ASG_RUN                               is 0

```

Default statements.
Note: No Alias statements are included in this formula.

```

/* Input Value Defaults */
default for Amount                is 0
default for Total_Owed            is 0
default for Clear_Arrears (text)  is 'N'
default for Guaranteed_Net       is 0

```

/* Inputs */

Input statement

```

Inputs are  Amount          /* Flat Amount calc rule. */
           ,Total_Owed     /* Total Amount of Garnishment. */
           ,Clear_Arrears (text) /* Clears down any amount in arrears */
           ,Guaranteed_Net /* Preserve this amount. */

```

/*===== Amount Calculation Section Begin =====*/

If no Replacement value is entered and no other input values are entered the formula returns a message only and exit the routine. If no replacement was entered and an Amount was entered the formula proceeds.

```

IF STATE_TAX_LEVIES_REPLACEMENT_ASG_GRE_ITD WAS DEFAULTED OR
STATE_TAX_LEVIES_REPLACEMENT_ASG_GRE_ITD = 0 THEN
(IF Amount WAS NOT DEFAULTED THEN
(deduction_frequency_factor = 1
 /* dedn_freq_factor(STATE_TAX_LEVIES_PERIOD_TYPE) */
 dedn_amt = Amount * deduction_frequency_factor
)
ELSE
(mesg = 'Could not find appropriate input value to calculate deduction'
RETURN mesg
)
)
)

```

If a replacement amount was entered, the deduction amount is set to equal the replacement amount.

```
ELSE
(dedn_amt = STATE_TAX_LEVIES_REPLACEMENT_ASG_GRE_ITD
 clear_repl_amt = -1 * STATE_TAX_LEVIES_REPLACEMENT_ASG_GRE_ITD
 )
/*===== Amount Calculation Section End =====*/
/*===== Adjust Amount Section Begin =====*/
```

If an additional amount was entered for the employee the formula adds this amount to the deduction amount.

```
dedn_amt = dedn_amt + STATE_TAX_LEVIES_ADDITIONAL_ASG_GRE_ITD
IF STATE_TAX_LEVIES_ADDITIONAL_ASG_GRE_ITD <> 0 THEN
 clear_addl_amt = -1 * STATE_TAX_LEVIES_ADDITIONAL_ASG_GRE_ITD
/*===== Adjust Amount Section End =====*/
/*===== Arrears Section Begin =====*/
```

The formula is determining if there is any arrearages that need to be added to the current deduction amount.

```
IF Clear_Arrears = 'N' THEN
(
 temp_to_arrears = 0
 temp_not_taken = 0
 dedn_amt = arrearage( STATE_TAX_LEVIES_PARTIAL_EE_CONTRIBUTIONS,
 NET_ASG_RUN,
 STATE_TAX_LEVIES_ARREARS_ASG_GRE_ITD,
 Guaranteed_Net,
 dedn_amt,
 temp_to_arrears,
 temp_not_taken)
IF temp_to_arrears <> 0 THEN
 to_arrears = temp_to_arrears
IF temp_not_taken <> 0 THEN
 not_taken = temp_not_taken
)
ELSE /* Clear down arrears balance. */
(
 to_arrears = -1 * STATE_TAX_LEVIES_ARREARS_ASG_GRE_ITD
 set_clear = 'No'
 temp_to_arrears = 0
 temp_not_taken = 0
/* Call with 0 amount for ARREARS_ASG_ITD b/c balance has been "cleared". */
 dedn_amt = arrearage( STATE_TAX_LEVIES_PARTIAL_EE_CONTRIBUTIONS,
 NET_ASG_RUN,
 0,
```

Guaranteed_Net,
dedn_amt,
temp_to_arrears,
temp_not_taken)

IF temp_to_arrears <> 0 THEN
to_arrears = to_arrears + temp_to_arrears

IF temp_not_taken <> 0 THEN
not_taken = temp_not_taken

)

/*===== Arrears Section End =====*/

/*===== Stop Rule Section Begin =====*/

IF Total_Owed WAS NOT DEFAULTED THEN

(total_accrued = dedn_amt + STATE_TAX_LEVIES_ASG_GRE_ITD

IF total_accrued >= Total_Owed THEN
(dedn_amt = Total_Owed - STATE_TAX_LEVIES_ASG_GRE_ITD

/* The total has been reached - the return will stop the entry under
these conditions */

STOP_ENTRY = 'Y'

RETURN dedn_amt, not_taken, to_arrears, clear_repl_amt,
clear_addl_amt, set_clear, STOP_ENTRY

)
)

/*===== Stop Rule Section End =====*/

/*===== Final Pay Section Begin =====*/

IF TERMINATED_EMPLOYEE = 'Y' AND FINAL_PAY_PROCESSED = 'N' THEN

/* This IS final pay run, considerations:

1) Try and collect remainder of Total Owed;
(consider Partial Deduction, not Arrears)

2) Refund EE Bond "Towards Purchase" balance

*/

(STOP_ENTRY = 'Y'

IF Total_Owed WAS NOT DEFAULTED THEN

/* Try and collect Total Amount. */

(dedn_left = Total_Owed - STATE_TAX_LEVIES_ASG_GRE_ITD

Here the formula is checking the total owed input value with the 'Inception to Date' balance. If the Inception to Date is equal to the Total Owed the element will be end dated and stopped. If the Inception to date is greater than the Total Owed, the overage will be refunded to the employee and the deduction end dated. **Note:** This is why I recommend not using the Total Owed input value.

The formula is checking the termination status of the employee. If there is a Total Owed and the employee is terminated the formula will attempt to collect the remainder owed. This may result in the entire net payroll for the employee and no other deductions being withheld.

```

IF NET_ASG_RUN > dedn_left THEN /* Recoup all Remainder */

    ( dedn_amt = dedn_left
      RETURN dedn_amt, to_arrears, not_taken, clear_repl_amt,
clear_addl_amt, set_clear, STOP_ENTRY
    )
ELSE

    RETURN dedn_amt, to_arrears, not_taken, mesg, clear_repl_amt,
clear_addl_amt, set_clear, STOP_ENTRY

)
ELSE
    RETURN dedn_amt, to_arrears, not_taken, clear_repl_amt,
clear_addl_amt, set_clear, STOP_ENTRY
)
ELSE /* Not Final Pay */

    RETURN dedn_amt, to_arrears, not_taken, clear_repl_amt,
clear_addl_amt, set_clear

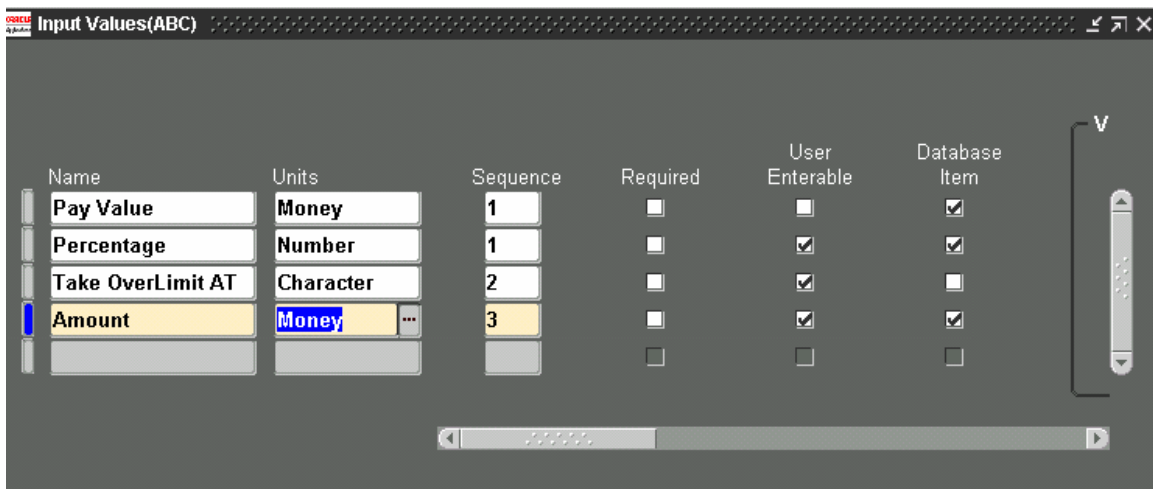
/*===== Final Pay Section End =====*/

/*===== End Payroll Formula Body =====*/

```

Modifying a Generated Deduction Element to Use a Percentage or a Flat Amount:

Use the Deduction template and select '% Earnings' rather than the default of 'Flat Amount.' This will cause the template to generate Percentage as one of the input values. When you have completed the entries for the template, navigate to the Element Description form and select the Input Values button. Add 'Amount' as an input value.



There is no need to modify the Fast Formula generated for your element, as the generated element will use either the Amount or the Percentage to calculate the pay value.

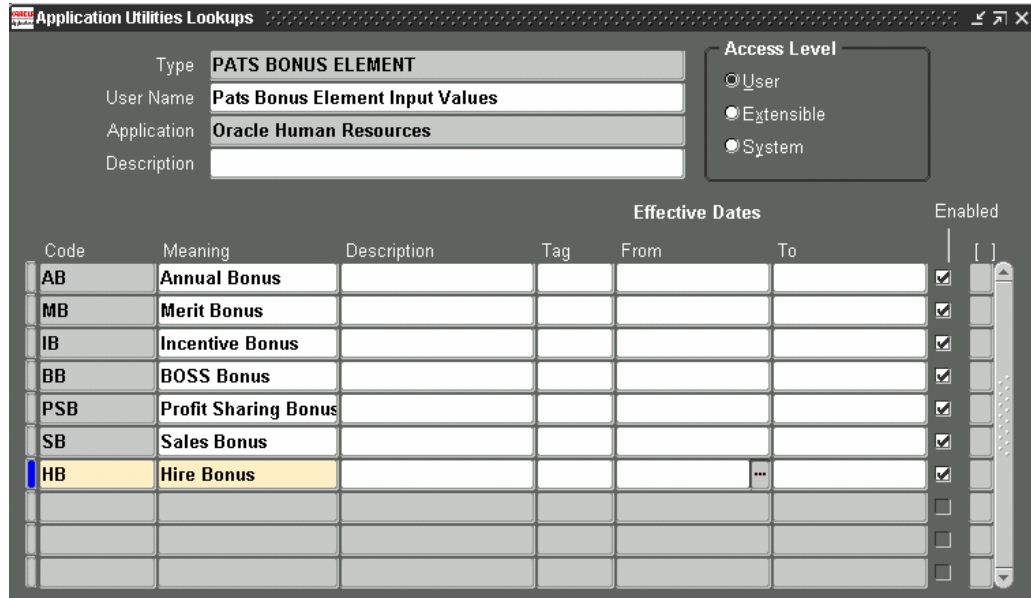
Creating an Information Element to Generate Other Elements:

This process can be used for generating any type of element and minimize the input requirements for the user. Remember to link all of the elements. For generated elements, be sure to check the "Indirect Results" box on the Element Description form. This will prevent the element from being displayed on the list of values when establishing element entries for employees.

1. Generate Bonus Supplemental Earnings Elements:

- **Create Lookup Table:**

Create a new Lookup Type that will include the names of your organization's bonus or award elements.



The screenshot shows the 'Application Utilities Lookups' window. At the top, there are input fields for 'Type' (PATS BONUS ELEMENT), 'User Name' (Pats Bonus Element Input Values), 'Application' (Oracle Human Resources), and 'Description'. To the right, there is an 'Access Level' section with radio buttons for 'User', 'Extensible', and 'System'. Below these fields is a table with columns: Code, Meaning, Description, Tag, Effective Dates (From, To), and Enabled. The table contains several rows of bonus types, with 'HB Hire Bonus' highlighted in yellow.

Code	Meaning	Description	Tag	Effective Dates	Enabled
				From To	
AB	Annual Bonus				<input checked="" type="checkbox"/>
MB	Merit Bonus				<input checked="" type="checkbox"/>
IB	Incentive Bonus				<input checked="" type="checkbox"/>
BB	BOSS Bonus				<input checked="" type="checkbox"/>
PSB	Profit Sharing Bonus				<input checked="" type="checkbox"/>
SB	Sales Bonus				<input checked="" type="checkbox"/>
HB	Hire Bonus				<input checked="" type="checkbox"/>
					<input type="checkbox"/>
					<input type="checkbox"/>
					<input type="checkbox"/>

- **Populate Lookup Values:**

You will need to update the values in this table as your organization adds or drops bonus plans to the compensation package.

- **Create Information Element:**

Create a new information element.

Select the Input Values button and provide the input names and characteristics.

Name	Units	Sequence	Required	User Enterable	Database Item
Bonus Amount	Money	1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Bonus Percent	Number	2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Bonus Plan	Character	3	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Future Use 1	Money	4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Future Use 2	Date	5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

The following values are suggestions only; use the value names that make sense to your organization. The Oracle payroll module does not allow the client to add or modify characteristics of input values after the element has been linked and balances exist. I allow for this by creating input values that are not user enterable but can be used at a later date if needed. The Oracle payroll module, does allow for name changes, however if you have checked the Database Item box, the Database Item name will keep the original name. This is important to know if you need this input value for your Fast Formula.

Bonus Amount	Money	1	*	*						1-Jan-2001
Bonus Percent	Number	2	*	*						1-Jan-2001
Bonus Plan	Money	3	*	*	*		New Lookup table name here		Error	1-Jan-2001
Future Use 1	Money	4								1-Jan-2001
Future Use 2	Date	5								1-Jan-2001
Future Use 3	Money	6								1-Jan-2001
Future Use 4	Money	7								1-Jan-2001


```

IB_Pct = Bonus_Percent
Return IB_Amount, IB_Pct)
Else
If Bonus_Plan = 'BOSS Bonus' then
(BB_Amount = Bonus_Amount
BB_Pct = Bonus_Percent
Return BB_Amount, BB_Pct)
Else
If Bonus_Plan = 'Profit Sharing Bonus' then
(PSB_Amount = Bonus_Amount
PSB_Pct = Bonus_Percent
Return PSB_Amount, PSB_Pct)
Else
If Bonus_Plan = 'Sales Bonus' then
(SB_Amount = Bonus_Amount
SB_Pct = Bonus_Percent
Return SB_Amount, SB_Pct)
Else
If Bonus_Plan = 'Hire Bonus' then
(HB_Amount = Bonus_Amount
HB_Pct = Bonus_Percent
Return HB_Amount, HB_Pct)
/*****End of custom Formula*****/

```

- **Create Earnings Elements:**

These elements will receive the information generated from the Bonus Information Element. Create one for each bonus type element you want to generate, as represented in your Lookup table. Respond to the values as appropriate for your organization's bonus plans.

The screenshot shows the 'Earnings' configuration window with the following details:

- Name:** Annual Bonus
- Reporting Name:** Annual Bonus
- Description:** Bonus element updated by OAUG Bonus Information Element
- Classification:** Supplemental Earnings
- Category:** Bonuses
- Alien Supplemental Category:** (Empty)
- Options:** Overtime Base, FLSA Hours, Reduce Regular, Grossup
- Standard / Grossup Processing:** (Grossup Processing is selected)
- Element Processing:**
 - Type:** Recurring, Nonrecurring
 - Priority:** 2500
 - Standard Link
 - Calculation Rule:** PERCENTAGE_OF_REG_EARNINGS_NONRECUR_V2
- Separate Check:** No, Yes
- Deduction Processing:** All (D), Tax Only (I), PreTax and Tax Only
- Effective Date:** From [] To []

- **Modify Element Description:**

Check the 'Indirect Results' box. This is critical, in order to prevent this element from appearing on the user's list of values.

Element

Name: **Annual Bonus** Reporting Name: **Annual Bonus**

Description: **Bonus element updated by OAUG Bonus Information Element**

Primary Classification: **Supplemental Earnings** Benefit Classification:

Effective Dates: From **14-MAR-2002** To

Processing

Standard **Advanced** Advance Pay Retro Pay

Type

Nonrecurring
 Recurring

Termination

Actual Termination
 Final Close
 Last Standard Process

Priority: **2500**

Skip Rule: **SUPPLEMENTAL_EARNINGS**

Multiple Entries Allowed
 Additional Entry Allowed
 Closed for Entry
 Process in Run
 Indirect Results
 Adjustment Only
 Third Party Payment

Currency: Input **USD** Output **USD**

Qualifying Conditions

Age
Length of Service
Units

Standard []

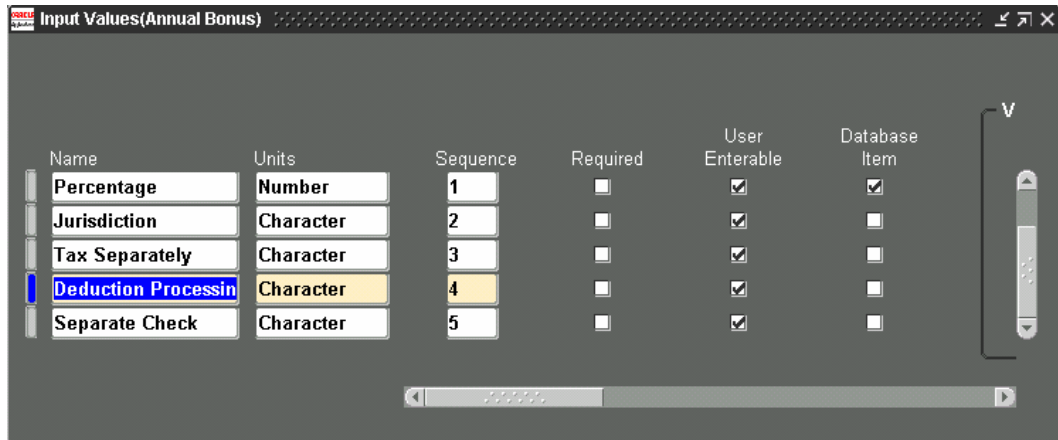
Further Information: **Bonuses..**

Input Values Balance Feeds Balance Feed Control Frequency Rules
Exclude Balances Iterative Rules Extra Information(g) Usages

Modify Input Values, if appropriate for your element. You may wish to uncheck the “Required” box for “Separate Check” and “Deduction Processing.” If you do not uncheck these boxes, it will be necessary for your information element to populate these input values.

Input Values(Annual Bonus)

Name	Units	Sequence	Required	User Enterable	Database Item
Pay Value	Money	1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Amount	Money	6	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Percentage	Number	1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Jurisdiction	Character	2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Tax Separately	Character	3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>



- **Modify Fast Formula:**

```

/*****
$Header: pelegus1.dat 115.33 2000/05/19 15:41:57 pkm ship $
FORMULA NAME: PERCENTAGE_OF_REG_EARNINGS_NONRECUR
FORMULA TYPE: Payroll
DESCRIPTION: This formula applies a percentage to the appropriate
              regular earnings of an employee according to the following
              rules in descending priority ::
              1) Salary Admin Pay Basis: ASG_SALARY * Percentage
                 if Pay Basis not hourly; else
                 ASG_SALARY * Percentage * normal period hours
              2) Regular Salary: Monthly_Salary * Percentage
              3) Regular Wages: Normal Period hours * Rate * Percentage
                 Normal hours per period are found by work schedule or standard
                 hours on ASGignment converted to the pay period timescale.
--
INPUTS:      Percentage
              Replacement Amount
              Additional Amount
--
DBI Required:  TERMINATED_EMPLOYEE
              FINAL_PAY_PROCESSED
              ASG_SALARY
              ASG_SALARY_BASIS
              ASG_FREQ
              ASG_HOURS
              REGULAR_WAGES_RATE_ENTRY_VALUE
              REGULAR_WAGES_RATE_CODE_ENTRY_VALUE
              REGULAR_SALARY_MONTHLY_SALARY_ENTRY_VALUE
--
Updatable Values:  RATE_TABLE
                   RATE_TABLE_COLUMN
--
Change History
17 Nov 1993  hparicha    Major revisions after initial creation.
06 Dec 1993  jmy chale   G305. Tidied up and output salary * percentage
                        if pay_basis is not hourly, else salary rate
                        * percentage * period_hours.
                        Percentage re-interpreted as non-fractional

```

on input value eg 10 %, 15 % etc.

07 Dec 1993 jmychale G317. Corrected syntax parenthesis error

14 Dec 1993 jmychale G432. Added Rounding

24 Feb 1994 hparicha G560. ASS -> ASG; TU -> GRE

09 Jun 1994 hparicha G907. Neg Earnings, Addl/Repl amounts;
 Redesign of Earnings. This formula
 is now copied by generated earnings
 and "individualized" by replacing
 ANNUAL_BONUS appropriately.

11 Mar 1996 hparicha Defaulted values for PAY_PROC_PERIOD_
 START/END_DATE. Default dates should
 be obvious when default is used.

323639 Major cleanup effort involving proration and
 other user defined functions and formulae
 changes.

```
--
*****
/
/* Alias Section */
ALIAS SCL_ASG_US_WORK_SCHEDULE AS Work_Schedule
/* Defaults Section */
default for STOP_ENTRY is 'Y'
default for TERMINATED_EMPLOYEE is 'N'
default for FINAL_PAY_PROCESSED is 'N'
default for ASG_SALARY is 0
default for ASG_SALARY_BASIS is 'NOT ENTERED'
default for REGULAR_WAGES_RATE_ENTRY_VALUE is 0
default for REGULAR_WAGES_RATE_CODE_ENTRY_VALUE is 'NOT
ENTERED'
default for REGULAR_SALARY_MONTHLY_SALARY_ENTRY_VALUE is 0
Default for ANNUAL_BONUS_ADDITIONAL_ASG_GRE_ITD is 0
Default for ANNUAL_BONUS_REPLACEMENT_ASG_GRE_ITD is 0
Default for ANNUAL_BONUS_ASG_GRE_RUN is 0
Default for ANNUAL_BONUS_ASG_GRE_YTD is 0
default for Work_Schedule is 'NOT ENTERED'
default for ASG_HOURS is 0
default for Percentage is 0
default for Amount is 0 /*Added by P Keeley */
/* WWBug 323639 begin */
default for PAY_PROC_PERIOD_START_DATE is '01-JAN-0001' (DATE)
default for PAY_PROC_PERIOD_END_DATE is '02-JAN-0001' (DATE)
DEFAULT FOR ASG_FREQ IS 'NOT ENTERED'
/* WWBug 323639 end */
/* Inputs Section */
Inputs are Percentage
,Amount /*Added by P Keeley */
/* Updatable Values Section */
RATE_TABLE = /**/ 'WAGE RATES' /**/
RATE_TABLE_COLUMN = /**/ 'Wage Rate' /**/
/*Modification Added to support 'Amount' input value */
If Amount was not defaulted
then
Template_Earning = Amount
Else
/* End of Modifiation */
```

For our sample bonus element, I added an Input Value for Amount, so either an Amount or a Percent can be entered for this element. The formula required modification to process this new input value.

```

IF ANNUAL_BONUS_REPLACEMENT_ASG_GRE_ITD WAS DEFAULTED OR
  ANNUAL_BONUS_REPLACEMENT_ASG_GRE_ITD = 0 THEN
/* Find appropriate earnings figure. */
  IF ASG_SALARY WAS DEFAULTED THEN
    /* Salary Admin is not being used */
    IF REGULAR_SALARY_MONTHLY_SALARY_ENTRY_VALUE WAS
      DEFAULTED THEN
      (
        IF REGULAR_WAGES_RATE_ENTRY_VALUE WAS DEFAULTED THEN
          IF REGULAR_WAGES_RATE_CODE_ENTRY_VALUE WAS DEFAULTED
            THEN
              ( msg = 'ANNUAL_BONUS: No Regular Earnings Can be calculated for this
                employee.'
                soe_run = ANNUAL_BONUS_ASG_GRE_RUN
                soe_ytd = ANNUAL_BONUS_ASG_GRE_YTD
                RETURN msg )
              ELSE
                /* Find Rate */
                regular_rate = To_Number( Get_Table_Value( RATE_TABLE,
                  RATE_TABLE_COLUMN,
                  REGULAR_WAGES_RATE_CODE_ENTRY_VALUE ) )
              ELSE
                regular_rate = REGULAR_WAGES_RATE_ENTRY_VALUE
              /* For REGULAR_WAGES, continue by finding the period_hours (per
                pay period ie. "PERIOD" value in To_Freq parameter of
                Convert_Period_Type ) and then compute result :
                template_earning */
                period_hours = Convert_Period_Type(Work_Schedule,
                  ASG_HOURS,
                  ASG_HOURS,
                  ASG_FREQ,
                  'PERIOD',
                  PAY_PROC_PERIOD_START_DATE,
                  PAY_PROC_PERIOD_END_DATE,
                  ASG_FREQ) /* WWBug 323639 */
                template_earning = ROUNDUP(
                  (Percentage * regular_rate * period_hours / 100),2)
              )
              ELSE
                /* Use Monthly Salary entered on Regular Salary element. */
                template_earning = ROUNDUP(
                  (Percentage * REGULAR_SALARY_MONTHLY_SALARY_ENTRY_VALUE /
                  100),2)
              ELSE
                /* Using Salary Admin Pay Basis */
                IF ASG_SALARY_BASIS = 'Hourly Salary' THEN
                  ( regular_rate = ASG_SALARY
                    period_hours = Convert_Period_Type(Work_Schedule,
                      ASG_HOURS,
                      ASG_HOURS,
                      ASG_FREQ,
                      'PERIOD',
                      PAY_PROC_PERIOD_START_DATE,
                      PAY_PROC_PERIOD_END_DATE,
                      ASG_FREQ) /* WWBug 323639 */
                    template_earning = ROUNDUP(

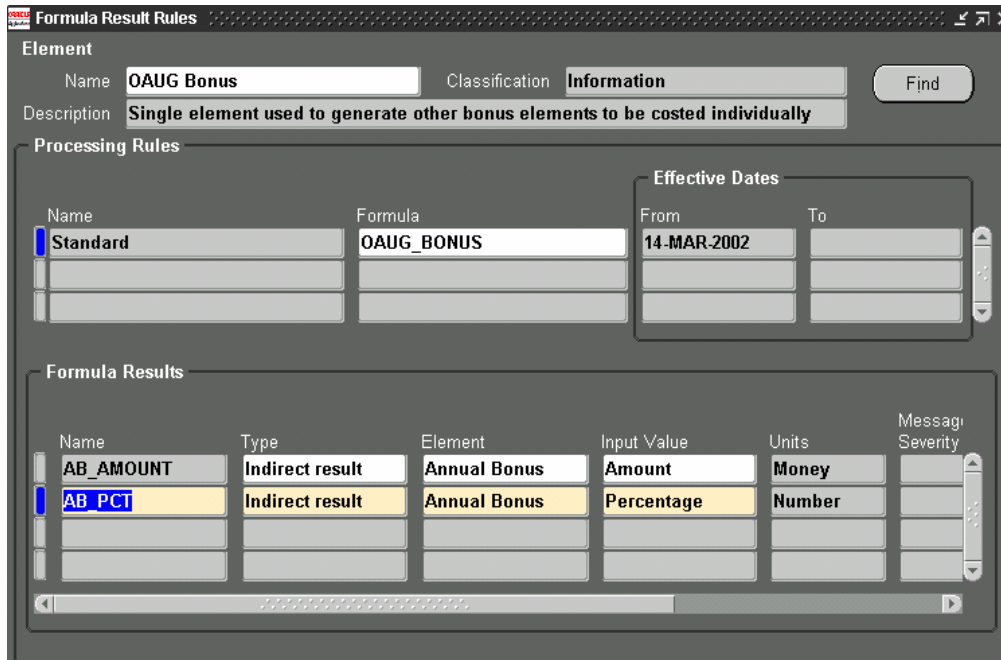
```

```

)
ELSE
template_earning = ROUNDUP((Percentage * ASG_SALARY / 100 ),2)
ELSE
/* ----- Replacement Amount WAS entered ----- */
(
template_earning = ANNUAL_BONUS_REPLACEMENT_ASG_GRE_ITD
clear_repl_amt = -1 * ANNUAL_BONUS_REPLACEMENT_ASG_GRE_ITD
)
template_earning = template_earning
+ ANNUAL_BONUS_ADDITIONAL_ASG_GRE_ITD
IF ANNUAL_BONUS_ADDITIONAL_ASG_GRE_ITD <> 0 THEN
clear_addl_amt = -1 * ANNUAL_BONUS_ADDITIONAL_ASG_GRE_ITD
if ( 1 = 1 ) then (
soe_run = ANNUAL_BONUS_ASG_GRE_RUN
soe_ytd = ANNUAL_BONUS_ASG_GRE_YTD
)
RETURN template_earning, clear_repl_amt, clear_addl_amt

```

- **Update Formula Results for Information Element:**



In the Processing Rules, enter Standard in the Name field and your formula name in the Formula Field. Move your cursor to the Formula Results area and enter the names of your return statements and related information:

Name	Type	Element	Input Value	Units	Message Severity	Date
AB_AMOUNT	Indirect result	Annual Bonus	Amount	Money		1-Jan-2001
AB_PCT	Indirect result	Annual Bonus	Percentage	Number		1-Jan-2001
MB_AMOUNT	Indirect result	Merit Bonus	Amount	Money		1-Jan-2001
MB_PCT	Indirect result	Merit Bonus	Percentage	Number		1-Jan-2001
IB_AMOUNT	Indirect result	Incentive Bonus	Amount	Money		1-Jan-2001
IB_PCT	Indirect result	Incentive Bonus	Percentage	Number		1-Jan-2001

Name	Type	Element	Input Value	Units	Message Severity	Date
BB_AMOUNT	Indirect result	BOSS Bonus	Amount	Money		1-Jan-2001
BB_PCT	Indirect result	BOSS Bonus	Percentage	Number		1-Jan-2001
PSB_AMOUNT	Indirect result	Profit Sharing Bonus	Amount	Money		1-Jan-2001
PSB_PCT	Indirect result	Profit Sharing Bonus	Percentage	Number		1-Jan-2001
SB_AMOUNT	Indirect result	Sales Bonus	Amount	Money		1-Jan-2001
SB_PCT	Indirect result	Sales Bonus	Percentage	Number		1-Jan-2001
HB_AMOUNT	Indirect result	Hire Bonus	Amount	Money		1-Jan-2001
HB_PCT	Indirect result	Hire Bonus	Percentage	Number		1-Jan-2001

2. Generate Pretax Adjustment Elements:

Usually the first question I get is “Why do I need a pretax adjustment element?” Experience has indicated that people often make mistakes when establishing elements for employees. Most often this is a case of setting up the incorrect element for the employee. While ‘Special Inputs’ work very well for active elements, they do not work for inactive elements. This Pretax Adjustment element will allow your users to make adjustments to discontinued elements. This element will also work for terminated employees where a refund is required. If neither of these conditions apply to your organization, it is not necessary to create a pretax adjustment element. This element will also adjust any employer match when applicable.

- **Create Lookup Table:**

Create a new Lookup Type that will include the names of your pretax elements that can be adjusted using this element.

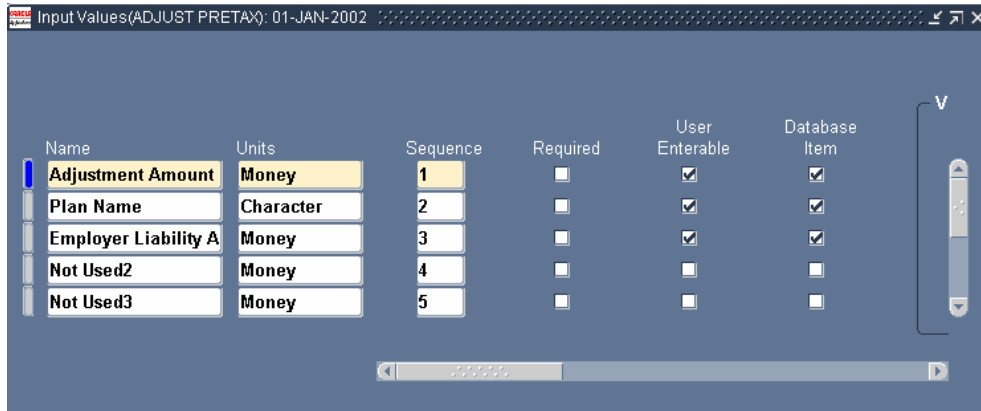
- **Populate Lookup Values:**

You will need to update the values in this table as you add or drop pretax plans from your benefit programs.

- **Create Information Element:**

Create a new information element.

Select the Input Values button and provide the input names and characteristics.



The following values are suggestions only; use the value names that make sense to your organization. The Oracle payroll module does not allow the client to add or modify characteristics of input values after the element has been linked and balances exist. I allow for this by creating input values that are not user enterable but can be used at a later date if needed. The Oracle payroll module, does allow for name changes, however if you have checked the Database Item box, the Database Item name will keep the original name. This is important to know if you need this input value for your Fast Formula.

Adjustment Amount	Money	1	*	*	*															1-Jan-2001
Plan Name	Character	2	*	*	*					New Lookup table name here									Error	1-Jan-2001
Employer Liability Amt	Money	3		*	*															1-Jan-2001
Not used2	Money	4																		1-Jan-2001
Not used3	Money	5																		1-Jan-2001
Not used4	Money	6																		1-Jan-2001
Not used5	Date	7																		1-Jan-2001
Not used6	Date	8																		1-Jan-2001
Not used7	Date	9																		1-Jan-2001
Not used8	Money	10																		1-Jan-2001
Not used9	Money	11																		1-Jan-2001
Not used10	Money	12																		1-Jan-2001
Not used11	Money	13																		1-Jan-2001
Not used12	Character	14																		1-Jan-2001
Not used13	Character	15																		1-Jan-2001

• **Write Formula**

Select the Edit button and use the following formula as your starting point. Keep in mind; you may have to modify this formula to meet your specific business requirements. After you have completed your formula, verify your work, close the edit window and save your work. Please note the blank lines are not required for the formula, they have been inserted to allow for the text boxes.

```

/*****
*****/

```

```

/*Custom formula written by BOSS Corporation to provide input values for
non-recurring non-payroll payments for pretax adjustments*/

```

Default statements for Input Values needed for this formula. Use the names you used to create your pretax adjustment element.

Default for Adjustment_Amount is 0
 Default for Plan_name is 'Not Entered'
 Default for Employer_Liability_Amt is 0

Input statements.
Again use the input
values for your
element.

```
Inputs are Adjustment_Amount  
      ,Plan_Name  
      ,Employer_Liability_Amt
```

The local variables
represent formula
names I created for
use in this formula
only. These values
do not carry to other
formulas or to other
balances unless they
are part of the return
statement. Use
names that make
sense to you. Fast
Formula is NOT
case sensitive.

```
/*Set local variables to 0 */  
P401K_amount = 0  
ADD_Amount = 0  
Catch_401_Amount = 0  
TSP_Amount = 0  
dental_Amount = 0  
life_Amount = 0  
ltd_Amount = 0  
medical_Amount = 0  
nqdc_Amount = 0  
vision_Amount = 0  
P401K_ER_amount = 0  
ADD_ER_amount = 0  
DENTAL_ER_amount = 0  
NQDC_ER_Amount = 0  
FSA_DAY_CARE_amount = 0  
FSA_HEALTH_amount = 0  
ER_Liability_Amt = Employer_Liability_Amt  
/*End of local variables */
```

Begin 'If' statements.
Note: Plan_Name was
defined as Character
so the value is 'text'
we are matching it to a
value from the Lookup
table meaning field
and must be exactly as
entered on the input
field, including blanks
and case. With each
'If' statement, if the
value matches, the
formula will return the
value and exit the
routine. Because you
created a Lookup table
and the Lookup table
value is the only value
permitted, the formula
will not encounter a
Plan_name that is not
appropriate.

```
If Plan_Name = 'PTX 401K' then  
(P401K_Amount = Adjustment_Amount  
P401K_ER_Amount = ER_Liability_Amt  
Return P401K_amount, P401K_ER_Amount)  
Else  
If Plan_name = 'PTX ADD' then  
(ADD_Amount = Adjustment_Amount  
Return ADD_amount)  
Else  
If Plan_name = 'PTX DENTAL' then  
(DENTAL_Amount = Adjustment_Amount  
Return DENTAL_amount)  
Else  
If Plan_name = 'PTX LIFE' then  
(LIFE_Amount = Adjustment_Amount  
Return LIFE_amount)  
Else  
If Plan_name = 'PTX LTD' then  
(LTD_Amount = Adjustment_Amount  
Return LTD_amount)  
Else  
If Plan_name = 'PTX MEDICAL' then  
(MEDICAL_Amount = Adjustment_Amount  
Return MEDICAL_amount)  
Else  
If Plan_name = 'PTX NQDC' then
```

```

(NQDC_Amount =Adjustment_Amount
NQDC_ER_Amount =ER_Liability_Amt
Return NQDC_amount, NQDC_ER_Amount)
Else
If Plan_name = 'PTX VISION' then
(VISION_Amount =Adjustment_Amount
Return VISION_amount)
Else
If Plan_name = 'PTX FSA DAY CARE' then
(FSA_DAY_CARE_Amount =Adjustment_Amount
Return FSA_DAY_CARE_Amount)
Else
If Plan_name = 'PTX FSA HEALTH' then
(FSA_HEALTH_Amount =Adjustment_Amount
Return FSA_HEALTH_Amount)
Else
If Plan_name = 'TSP' then
(TSP_Amount =Adjustment_Amount
Return TSP_Amount)
Else
If Plan_name = 'PTX CATCH 401' then
(CATCH_401_Amount =Adjustment_Amount
Return CATCH_401_Amount)

```

/******End of custom Formula******/

• Create Non-Payroll Payment Elements:

These elements will receive the information generated from the Adjust Pretax Element. Create one for each pretax element you want to adjust.

The screenshot shows a software window titled "Earnings: 01-JAN-2002". The main configuration area is for an element named "ADJUST PTX 401K". The "Reporting Name" is also "ADJUST PTX 401K". The "Description" is "Element generated by Pretax Adjustment". The "Classification" is "Non-payroll Payments" and the "Category" is "Expense Reimbursement". There are checkboxes for "Overtime Base", "FLSA Hours", "Reduce Regular", and "Grossup". Below this, there are two tabs: "Standard" and "Grossup Processing", with "Grossup Processing" selected. The "Element Processing" section includes a "Type" dropdown with "Recurring" and "Nonrecurring" options, a "Priority" field set to "750", and a "Calculation Rule" field set to "FLAT_AMOUNT_NONRECUR_V2". There is also a "Standard Link" checkbox. The "Separate Check" section has radio buttons for "No" (selected) and "Yes". The "Deduction Processing" section has radio buttons for "All (D)", "Tax Only (I)", and "PreTax and Tax Only". The "Effective Date" section has "From" and "To" date input fields.

• **Modify Non-Payroll Payment Element Description:**

Check the Indirect Results box – this is critical!! This prevents the element from appearing on the user’s list of values.

Select the Input Values button.

Delete the input values lines for “Separate Check” and “Deduction Processing”

Name	Units	Sequence	Required	User Enterable	Database Item	Val
Amount	Money	1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	D
Pay Value	Money	1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
Jurisdiction	Character	2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Deduction Processin	Character	3	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Separate Check	Character	4	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

When prompted to “Purge” click “OK.”

Name	Units	Sequence	Required	User Enterable	Database Item	Val
Amount	Money	1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	D
Pay Value	Money	1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
Jurisdiction	Character	2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Save your work and proceed to the next element. You will do this for each earnings element you created in the previous step. When prompted for Correction or Update, choose Correction.

- **Create Information Element to Update ER Liability:**

Navigate to the Element Description Form and create Information elements for all the ER elements that will be updated.

Sample:

Change Priority to 251

Check the Indirect Results box.

Save your work and select the Input Values button.

Enter the following value:

Name	Units	Sequence	Required	User Enterable	Database Item	Val
Amount	Money	1	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Repeat this process for all of the ER Adjustment elements.

- **Modify Formula Results:**

This is the point where you tell the system what you want to do with the return statements from the Adjust Pretax Fast Formula

In the name field enter name of your Pretax Adjustment element.

In the Processing Rules, enter Standard in the Name field and your formula name in the Formula Field. Move your cursor to the Formula Results area and enter the names of your return statements and related information:

Name	Type	Element	Input Value	Units	Message Severity	Date
P401_AMOUNT	Indirect result	ADJUST PTX 401K	Amount	Money		1-Jan-2001
ADD_AMOUNT	Indirect result	ADJUST PTX ADD	Amount	Money		1-Jan-2001
DENTAL_AMOUNT	Indirect result	ADJUST PTX DENTAL	Amount	Money		1-Jan-2001
LIFE_AMOUNT	Indirect result	ADJUST PTX LIFE	Amount	Money		1-Jan-2001
LTD_AMOUNT	Indirect result	ADJUST PTX LTD	Amount	Money		1-Jan-2001
MEDICAL_AMOUNT	Indirect result	ADJUST PTX MEDICAL	Amount	Money		1-Jan-2001
NQDC_AMOUNT	Indirect result	ADJUST PTX NQDC	Amount	Money		1-Jan-2001
VISION_AMOUNT	Indirect result	ADJUST PTX VISION	Amount	Money		1-Jan-2001
NQDC_ER_AMOUNT	Indirect result	ADJ NQDC ER	Amount	Money		1-Jan-2001
P401K_ER_AMOUNT	Indirect result	ADJ 401K ER	Amount	Money		1-Jan-2001
FSA_DAY_CARE_AMOUNT	Indirect result	ADJUST FSA DAY CARE AMOUNT	Amount	Money		1-Jan-2001
FSA_HEALTH_AMOUNT	Indirect result	ADJUST FSA HEALTH AMOUNT	Amount	Money		1-Jan-2001

- **Modify Balance Feeds**

You have created the elements and assigned the Fast Formula via the Formula Results, now in addition to linking each of the elements you need to update the appropriate balance feeds. Following is an example of the balances affected by this element. You will need to determine what balances need to be updated by your elements. It is always advised that you test thoroughly before moving these elements and formulas to Production.

Query Balance Name (this screen is case sensitive):

Balance: 01-JAN-2002

Name **PTX DENTAL**

Reporting Name **PTX DENTAL** Units **Money**

Currency **USD** Use For Remuneration Gross Up

Feeds Classifications Dimensions

Initial Feed

Select the Feeds button and enter the name of the pretax adjustment element and the input value.

Balance Feeds(PTX DENTAL): 01-JAN-2002

Element Name	Classification	Input Value Name	Add or Subtract	Effective Dates	
				From	To
PTX DENTAL	Pre-Tax Deductio	Pay Value	Add	01-JAN-1951	
ADJUST PTX DENTAL	Non-payroll Payr	Amount	Subtract	01-JAN-2002	

Balances for ER elements:

Query Balance Name:

Balance: 01-JAN-2002

Name **PTX 401K ER**

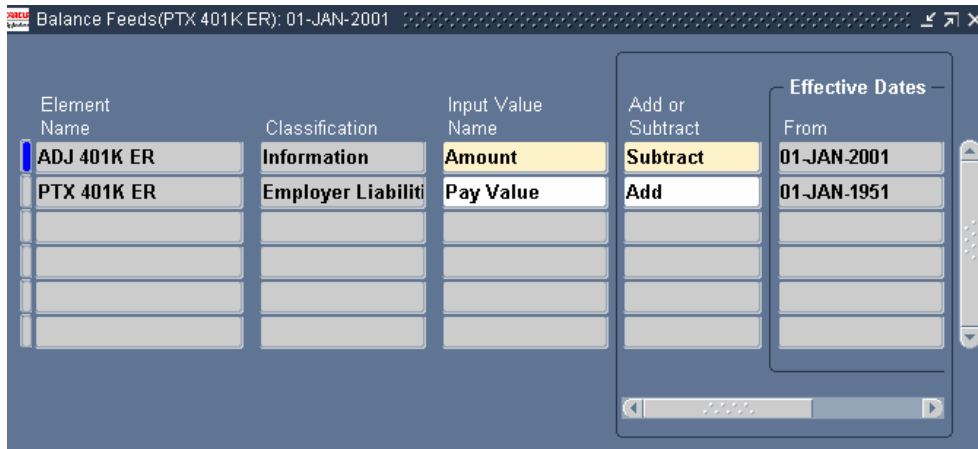
Reporting Name **PTX 401K ER** Units **Money**

Currency **USD** Use For Remuneration Gross Up

Feeds Classifications Dimensions

Initial Feed

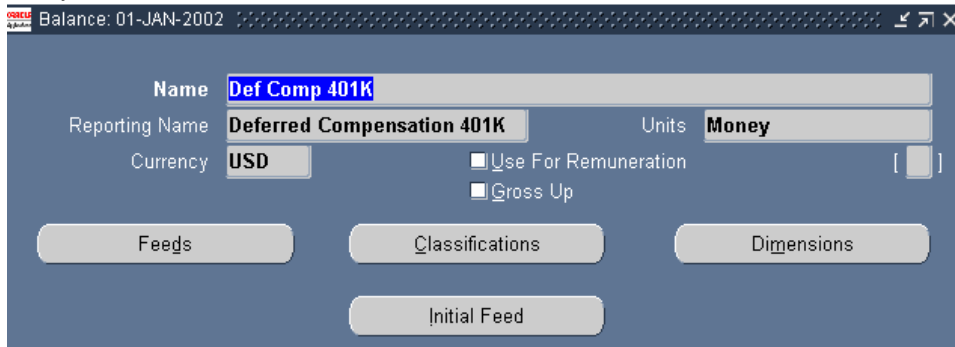
Select the Feeds button and enter the name of the pretax adjustment element and the input value.



Tax related balances.

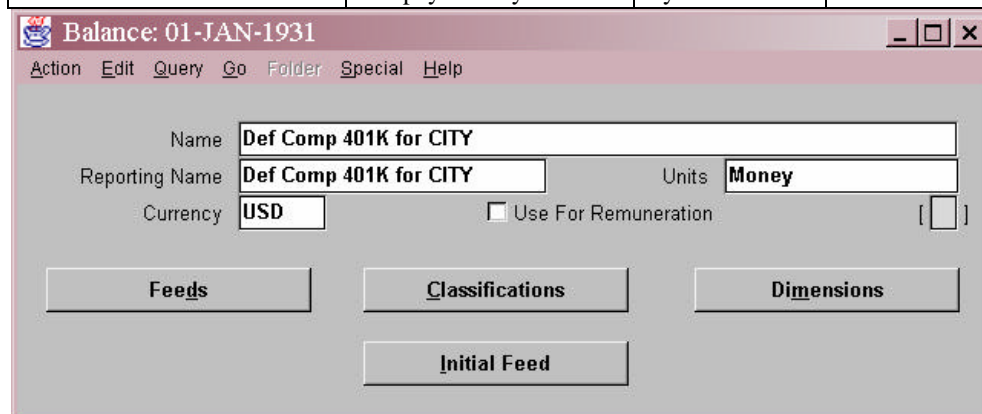
Helpful tip: Query the predominate characteristic of the balance name; this will allow you to page through the balance names rather than entering a new query each time.

Query Balance Name:



Select the Feeds button and enter the name of the pretax adjustment element and the input value. Add the following feeds:

Element Name	Classification	Input Value	Subtract/Add
ADJUST PTX 401K	Non-payroll Payments	Pay Value	Subtract



Select the Feeds button and enter the name of the pretax adjustment element and the input value. Add the following feeds:

Element Name	Classification	Input Value	Subtract/Add
ADJUST PTX 401K	Non-payroll Payments	Pay Value	Subtract

Continue until all applicable balances have been updated.

Recap:

Oracle Payroll depends on these basic components to create the run results used to pay and track information for your employees.

Elements are defined as: Earnings, Deductions, ER Liabilities, Tax Deductions, and Information. Each element type is further defined by Classification and Category.

Templates are used by Oracle payroll to create generated Fast Formulas that are unique to each element. If no template is used (i.e. element is created via the Element Description Form) and run results are required, the Fast Formula must be written by the client.

Fast Formulas are a series of statements and comments that receive values from one or more elements, Lookup table, or Global Value that can be manipulated to produce results that can be returned to the base element as a direct result, message, or Stop indicator or to other elements as an indirect result.

Fast Formulas are associated with the Element(s) via the Formula Results form. Once you have assigned the formula to the element name, you associate the returns with the elements and input values to be updated by the results of the formula.

Balances can be updated by input values or a generated pay value.

Additional Information Available:

- Using Oracle Fast Formula
- Oracle US Payroll Implementation Guide
- Oracle US Payroll Users Guide