POWER OF SIMPLICITY

Tally 9 Release 3.0 Beta

Release Notes
29th July, 2008

Ver 3.0 Dated 29th July, 2008

Contents

Release Notes for Tally 9 Release 3.0 Beta

Going further with the success of Tally 9 Release 2.0, this amazing major release of Tally will take the product flexibility, performance and volume handling capacity to greater heights. The underlying philosophy of Tally has been to give our customers the best of speed and the latest technology with improved functionalities. We have incorporated major changes in the data processing capabilities to improve the overall performance and are proud to say that we have kept up this promise once again with the latest release.

The important features included in this release are as follows:

I. General Enhancements

1. Size and Performance changes

Automatic caching based on system memory

Earlier, report generation involving large amount of data was always impeded by the problem of high memory usage. In Release 3.0, a new feature has been introduced to overcome this hurdle. The application now uses an automatic caching based on the memory of the system it is running on. This results in the usage of lesser memory even with higher data volumes.

Reports which were running out of memory or consuming large amount of virtual memory on a 2GB system, will now respond even on a 1GB system. This can be observed by generating a report say a day book containing 50,000 to 100,000 vouchers on a 512MB machine using the last major release as well as this release.

Faster Data Entry

The data entry speed has been considerably increased in scenarios where there are a large number of ‘Tracking numbers’ and ‘list of pending orders’. In other words, the time taken to make a single entry has been reduced due to the faster retrieval and display of tables for selection of tracking numbers and pending orders.

Application size reduction by three-fourth

A faster, powerful and a SMALLER Tally application! The application has been modified and restructured to drastically reduce its size. The “exe” file of the application has been approximately reduced to three-fourth the size when compared to earlier versions. To be more precise, it is reduced from 10.7 MB to 1.96 MB. This will result in faster installation, loading, and opening of the application, thereby enhancing the overall user experience.

Tally DATA SIZE Reduction

Tally data size has been reduced in this release. Tally 9 Release 3.0 will effectively reduce the data size up to 40%. This data size reduction will help in enhancing the overall performance of the application.

Note: Data rewrite is mandatory in this release.

Tally 9 goes GREEN

Tally 9 is now eco-friendly.

While printing reports that are more than one page, the header and footer part used to consume equal height on all pages even if one mentions less or no height in Page Break Part. The Page Break behavior in TDL is now enhanced to utilize the entire page.

For example, Sales Invoice with 100 Stock Items being printed in 5 Pages on A4 Paper leaving the footer part un-utilized. With this enhancement, now it will be printed in 3 pages utilizing the full page thus saving lot of pages.

Performance - GST Analysis Report

Performance of GST Analysis Report (Singapore) for a specific period has improved compared to Tally7.2 and Tally 9 (earlier releases).

2. *Installation and Licensing changes*

Tally Installer

Tally 9 Release 3.0 comes with a new installer. The new installer is being provided for better management of licensing activities and to overcome few drawbacks that were faced in the previous versions.

Earlier when the port was set in the Tally.ini file, both Tally application and License Server were forced to use the same port for communication. As a result, in a multi-user environment, the user was forced to provide a separate port for either the License Server or for Tally. From Tally 9 Release 3.0 onwards, there is a clear separation between the port for Tally application and License Server.

The following options are available during installation:

- **Tally** application can be installed without the License **Server** for client or stand-alone machines.
- License Server alone can be installed in a Multi–User environment to serve as dedicated License Server for clients.
- Both Tally application and License Server can be installed.

The user can select different directories for Tally Application and the Tally License Server during installation.

If the customer is installing Tally as a client, the user will have an option to specify the License Server details like server name and port at the time of installation. This setting is stored in the Tally.ini file (which can be changed later, if required). By default, Tally application runs and uses the port 9000 for ODBC, Synchronization and other data exchange/interface activities.

The default port number for the License Server is specified as 9090 (during License Server installation). The user can change the same while installing the License Server (The same can be changed later in the Tally.ini file, if required).

Uninstaller: The Uninstaller now contains options to uninstall either Tally9 or License Server or both. Each component (either Tally 9 or License Server) can be uninstalled independent of the other.

Note: If the Tally License Server is serving license to clients, uninstalling of License Server may cause the clients to switch to Educational mode.

License Server Admin Tool

License Server Admin Tool, will be available with both License Server and Tally application. This tool can be used to perform the following License related activities:

1. License Server Operations (only for License Server – Multi User environment)

   a. **Install** License Server
   b. **Uninstall** License Server
   c. **Restart** License Server

2. License Operations (only for License Server – Multi User environment)

   a. **Activate** License
   b. **Update** License
   c. **Surrender** License

   Proxy Configuration:

   If the user is using Proxy Server to connect to internet in such cases, the Proxy server details can be set in the Proxy configuration screen by providing Proxy URL and User Name and Password in case of Authentication.

3. License Server Status (for client machines)

   a. You can now **PING** the **License Server** to check the status of **License Server** (whether it is Running or not)
   b. **PING** will also help to check the status of your **License** (whether it is Activated or not)

Note:

- **License Server Admin Tool** - Licensing Operations can be done only at the License Server machine.
- **License Server Admin Tool** - Currently, Licensing operations using LicAdminTool can be done only for online mode (Activation/Update/Surrender). For Offline licensing operations, the users need to use Tally application.
- **"LicServerPort"** - A new Tally.ini Parameter has been introduced to indicate the port where the Tally License Server is running (e.g.: LicServerPort = 9090).
- License Server can be installed only on NT based Operating Systems (e.g. Windows NT/2000/XP/2003) and not in Windows 98 and ME.

Note: For users upgrading to Tally 9 Release 3.0, Tally requires re-activation of license.

II. Tally TDL Enhancements

1. Major enhancements in TDL

Introduction of TDL Server DLL Component

Tally 9 Release 3 onwards, a new component TDLServer.DLL is introduced which gets copied to the folder in which Tally 9 is being installed. All the default TDL files of Tally have been integrated in this component. On invoking Tally 9, these TDLs will be loaded. The user TDLs will be subsequently loaded as usual from Tally.ini.

The componentization will result in more robust application. The benefits of this componentization are as follows –

- In case of any updates/ changes in TDL, only TDLServer.dll file needs to be downloaded and not the entire Tally application which enables faster downloads.
- This will help in dynamic loading and unloading of TDL Files while Tally application is running which will be available in subsequent builds.
- Forthcoming Tally Developer will be directly using this component which eliminates the need for a separate DefTDL.dat file.

New Collection capabilities for aggregation and reporting

Collection, the data processing artifact of TDL provides extensive capabilities to gather data not only from Tally database but also from external sources using ODBC, DLLs, HTTP and so on. A set of new capabilities have been added to Collection which will provide far more flexibility and power in the hands of the TDL programmer. This will allow him to write significantly complex reports with ease and still deliver enhanced performance with high volume of data. These are among the most significant changes in TDL in the last 10 years which will move the overall platform and product capability to a new dimension.

Data Roll up/summarization capability in TDL Collection - Group by/ Roll up

A major technological advancement in this release of Tally 9 is “Data Roll up in TDL Collection – GROUP BY”, which is a part of the TDL language capabilities. This is a milestone achievement over the past 10 years. This will now facilitate the creation of large summary tables of aggregations in a single shot using the new syntax/attributes of the Collection description.

This is something similar to **GROUP-BY** provided by relational database query language but is more flexible and powerful as it allows you to **Walk** down the object hierarchies and gathers values and summarizes them in one scan. Overall it reduces the TDL code complexity, resource requirement and increases performance drastically in case of reports generated using this new capability.

Till now, all the totals were generated using the Total and aggregation functions like **CollAmtTotal** or **FilterAmtTotal** via collections. These have certain advantages and disadvantages. While they provide excellent granularity and control, each call is largely an independent activity to gather the data set and then aggregate it. This can make the code very complex and may not scale up if a large number of totals need to be generated as in the case of most business summary reports or a large underlying data set being used. Considering the object oriented nature of Tally data and existence of sub-objects up to any level, the task becomes even more complex. These functions require multiple data scans to produce a summary report with multiple rows and columns.

This methodology has now been complemented with a single scan to get all the totals including those based on User Defined Fields (UDFs). Native aggregation capability has now been added to a collection itself. A TDL programmer can use a source collection of vouchers and create a summary collection out of it by specifying which elements or sub-elements to walk, what criteria to use for uniquely identifying an aggregation and what to aggregate as totals. The overall effect is a reduction in TDL code complexity and resource requirement, enhanced performance by orders of magnitude especially concerning reports generation.

This aspect is shown in the code snippet below:

```
[Collection: My Source Collection]        ;; source collection defined to use for aggregation

  Type                : Voucher


[Collection: My Summary Collection]

  Source Collection   : My Source Collection              ;; source collection used for aggregation
  Walk                : Ledger Entries                    ;; sub-object if any to walk from source object
  By                  : MyLedgerName : $LedgerName        ;; create an aggregation for every unique ledger name
  Aggr Method         : MyTotal  :  Sum: $Amount     ;; total the amount of entry into the respective aggregation
```

New collection attributes

The following attributes have been added to Collection syntax to support roll-up and extraction.

Source Collection : Collection name, Collection Name, …

(multiple, list, mandatory, which collection(s) to use for source data)

Walk : Sub-Object Type / Sub-Collection …

(multiple, list, optional, further elements to walk on the source)

By : Method-Name : Method-Formula

(multiple, named-list, mandatory, criteria on which aggregation is to be done)

Method : Method-Name : Method-Formula

(multiple, named-list, optional, normal method, neither key nor aggregate)

Native Method : Existing-Method-Name-in-Source, …

(multiple, list, optional, short-cut for method)

Aggr Method : Method-Name : Aggr-Type : Method-Formula

(multiple, named-list, mandatory, aggregation to be done on By)

(Aggr-type can accept values Sum, Max or Min)

Keep Source : Yes/No

(Boolean, optional, default=no, to preserve source for re-use or release it)

Sample TDL Code demonstrating the above

Note:

Multiple **Source Collections** can be specified as a list, separated by commas.

Summary Collection does not have any object type.

Walk is optional and if not specified the source object itself is used. On the other hand, Walk can be specified to any depth for the source object. This gives enormous flexibility and power. For example, you can specify to Walk each inventory entry, each of its batch allocation and subsequently, apply aggregations. The Walk list has to be specified in the order in which they occur in the object or can be collected so.

Walk : Inventory Entries, Batch Allocations

Aggregation criteria can be one or more. For example, if we have a UDF category stored in each ledger entry. You can choose to aggregate both by ledger name and category also by specifying them independently. So you will get the totals for each ledger name and category combination

By : MyLedgerName : $LedgerName

By : MyCategory : $CategoryName

A programmer can create his own **Method** name(s) for grouping as well as totaling. The same can be referred to in rest of the TDL report.

Filters specified in the **Summary Collection** are applied on the **Summarized Object** and not on the **Source Object**.

Sorting can be specified on the methods defined and created in the **Summary Collection** and **Object**.

If a value from the source has to be taken as it is, use the **Method** keyword instead of **By** or **Aggr Method**.

Method : Date : $Date

If multiple values from the source have to be taken as it is with the same name, use the **Native Method** Keyword. It is equal to specifying it with the **Method** Keyword with the same method name and formula -

Native Method : Date, Narration

is similar to writing

```
Method          : Date          : $Date
Method          : Narration     : $Narration
```

The default data type for all columns is **String**.

Search Key is **Case Sensitive**.

Collection re-use, extraction and chaining support in TDL Collection

Another important enhancement is that a collection can now extract information from other collections including its sub-objects with the choice of method(s), filter(s) and sort-order. This can be best demonstrated with an example.

You can create a summary collection of Departments & employees once and then extract information using this capability into another collection dynamically to present them as a hierarchy. The data has been collected only once, but is being extracted in parts and displaying them without re-gathering them from the database.

With the introduction of **Source Collection** within a collection, collection(s) can now be chained. This can be used in various ways. For example – you can extract a sub-set of information from a collection or re-apply summarization to an already summarized collection. You have the flexibility of creating new methods, sorting and filtering. Extending the same example above:-

```
[Collection: My Source Collection]      ;; source collection defined to use for aggregation

    Type            : Voucher

[Collection: My Summary Collection]

    Source Collection : My Source Collection            ;; source collection used for aggregation
    Walk              : Ledger Entries                  ;; sub-object if any to walk from source object
    By                : MyLedgerName : $LedgerName      ;; create an aggregation for every unique ledger name
    Aggr Method       : MyTotal  : Sum : $Amount        ;; total the amount of entry into the respective aggregation

[Collection: My Parent Summary Collection]

    Source Collection : My Summary Collection                           ;; use summary as new source
    By                : MyParent        : $Parent:Ledger:$LedgerName    ;; aggregation for every unique group
    Aggr Method       : MyParentTotal   : Sum : $MyTotal                ;; total the amount for each group
```

Similarly you can pick up data from external sources like Excel via ODBC or another application via DLL and then process them in various ways using this extract-collection capability.

Indexed or Searchable Collection on TDL defined keys

The above capabilities extend the data gathering capabilities of TDL. However business reporting in general and more so in Tally uses hierarchical presentation or columnar presentation rather than simple table representation. This creates a unique and natural experience of working with the product & business data.

In case you can simply repeat the summarized collection and get the desired report, everything works fine with the existing capabilities. However if your representation is not necessarily a simple repeat of the collection, things become more complex.

For example, if you have collected data on two or more dimensions like Ledger and Cost Center or Employee and Pay head and so on but want to present it as row by columns or a hierarchy; a simple repeat on the summarized collection will not suffice.

Assume you have gathered the following in your summarized collection on cost center ledger combination; the same has been collected as a set of n objects in the collection as follows:-

```
Object 1 - CC-a   Ledger-l    Amount-x
Object 2 - CC-b   Ledger-m   Amount-y
Object 3 - CC-c   Ledger-n   Amount-z
...
Object n
```

Now you have the combination totals available in the collection but you want to represent ledgers as rows and cost centers as columns. The following options are available:-

Use function(s) like **CollectionField** or **FilterValue** in each column.

Create Summary collection for each column.

The first one will scan through the whole collection for every value required. The second one will scan the whole source data as many times as there are columns. Both of them will take a significant hit on the scale and volume that it can handle and affect the resultant performance.

To provide presentation capabilities beyond simple tables, a new capability has been added to the collection definition. A search key can be defined in the collection using the Search Key attribute. This implies that a unique key is created for every object which can be used to instantly access the

corresponding objects and its values without needing to scan or re-collect. The corresponding function created to access the same is $$CollectionFieldByKey which has the following signature:

```
$$CollectionFieldByKey:Method:Key-Formula:Collection-Name
```

Using the above example of Ledger and Cost Center, we can create a key for the combination by adding the following code to the collection (method names are representative only)

```
Search Key : $LedgerName + $CostCenterName
```

Now on any row/column in your report, you can access the combination total using

```
$$CollectionFieldByKey:$MyTotal:@MySearchKey:MySummaryCollection
```

where **MySearchKey** is the formula to get the **Ledger Name + Cost Center** name at that point. Effectively you can create your report by repeating on your collection for ledgers on rows and cost centers on columns but picking up the total value from this summarized and indexed collection using search key, thus removing your limitation of representing the data necessarily in the way it has been collected.

This is a dynamic index creation capability where the TDL programmer can define the key and the collection is indexed in the memory using the key. Once the index is created, any object in the collection can be instantly searched without needing a scan as in the case of a filter.

Sample TDL Code demonstrating the above

Function, IsObjectBelongsTo

A new function $$IsObjectBelongsTo has been provided with the following signature:

```
$$IsObjectBelongsTo:ObjType:ObjName:BelongsToName
```

As compared to the existing function **IsBelongsTo**, this provides more explicit control in the hands of the programmer by allowing him to specify the object type and name in addition to parentage against which it needs to be checked. You can easily relate to it by considering the fact that the summarized objects are not of any native type, they are just aggregation

objects. This function allows an easy link back into the native object type and **Walk** up the chain. It is very useful when creating hierarchical reports on summarized collections.

Sample TDL Code demonstrating the above

Direct access to any Object, Sub-Collection(s) Method

The $ operator has been enhanced with a new syntax. This allows direct access to any object method including its sub-collections to any level with a dotted notation framework.

The Syntax to access a Method prior to this Release was:

```
$MethodName
```

OR

```
$MethodName:PrimaryObjType:ObjNameFormula
```

Apart from the above, the new extended Syntax is:

```
$<PrimaryObjectSpec>.<SubObjectPathSpec>.MethodName
```

Where,

- **Primary Object Path Specification can either be relative or absolute.**

Relative Path is referred using empty parenthesis () or Dotted path to refer to the Parent object relatively. SINGLE DOT denotes the current object, DOUBLE DOT the Parent Object, TRIPLE DOT the Grand Parent Object and so on within an Internal Object. **Absolute** Path refers to the path in which the Primary Object is explicitly specified.

Syntax – Access to Methods of Primary Object using **Relative** Path

```
$().MethodName or $..MethodName or $... MethodName
```

Example:

Being in the context of LedgerEntries Object within Voucher Object, the following has to be written to access the Date from its Parent Object which is the Voucher Object.

```
$..Date
```

Syntax – Access to Methods of Primary Object using **Absolute** Path

```
(<Primary Object Type Keyword>, <Primary Object Identifier Formula>)
```

Example:

```
$(Ledger, “Cash”).OpeningBalance
```

- **SubObjectPathSpec**

Syntax for the **Sub-Object** Path Specification

CollectionName[Index Formula, [<Condition>]]

where Index Formula should return a number which acts as a position specifier in the Collection of Objects matching the given condition.

Example:

```
$BillAllocations[First,$Name="MyBill"]
```

- **MethodName**

MethodName refers to the name of the method in the specified path.

Primary Object specification, Sub collection and index specification is optional. In this case the current object will be considered as primary object. <Primary Object Identifier Formula>, <Index Formula> and Condition can be a value or formula.

Please refer the following examples –

To get the Ledger Name of the first Ledger Entry from the current Voucher,

```
Set as          : $LedgerEntries[1].LedgerName
```

To get the amount of the first Ledger Entry on the Ledger Sales from the current voucher (Sales Invoice),

```
Set as          : $LedgerEntries[1,@@LedCondition].Amount
LedCondition    : $LedgerName = “Sales”
```

To get the first Bill Name of the first Ledger entry on the Party Ledger from the current voucher (Sales Invoice),

```
Set as      : $LedgerEntries[1,@@LedCondition].BillAllocations[1].Name
LedCondition : $LedgerName = @@InvPartyName
```

To get the OpeningBalance of the first Bill for the Party Ledger Acme Corp

```
Set as      : $(Ledger,@@PartyLedger).BillAllocations[1].OpeningBalance
PartyLedger : "Acme Corp"
```

Examples using relative path –

To access the Amount in LedgerEntries Object while the current object in context is BillAllocations,

```
Set as      : $..Amount
```

To access the Date from Voucher Object while the current object in context is BillAllocations,

```
Set as      : $...Date
```

OR

```
Set as      : $().Date
```

Note:

Using the new capability, there is no need to repeat a line over a sub-collection to access it or one can pick up values from any object anywhere without needing to make it the current object.

Primary Object Specification **(PrimaryObjType, PrimaryObjNameFormula)** is optional. If not specified, the current Object is used.

Sub-Collection specification **Sub-Collection[Index Formula, Condition Formula]** is optional. If not specified, methods from the current or specified primary object will be available. Index specifies the position of the Sub-Object to be picked up from the Sub-Collection. Condition is the filter which is checked on the objects of the specified Sub-Collection.

Any one of the Index or Condition **[Index Formula, Condition Formula]** can also be used. **Index Formula** can be any formula evaluating to a number. Positive Number indicates a forward search and negative number indicates backward search. This can also be a keyword **First** or **Last** which is equivalent to specifying **1** or **-1** respectively.

If both **Index** and **Condition** is specified, the index is applicable on the Object(s) which satisfy the condition so one gets the nth Object which clears the condition. Let's say for example, if the Index specified is 2 and Condition is Name = "Sales", then the second object which matches the name Sales will be picked up.

Suffixing of **:PrimaryObjType:ObjNameFormula** is still supported for backward compatibility. In cases where both are specified; the enhanced new primary object specification will be considered.

Sample TDL Code demonstrating the above

Enhancements in Key Actions

Multi-Line Selection Capability

Multi-line selection and performing various operations on the selected/ unselected lines is a major feature which has been introduced in Tally 9 Release 3. Multiple vouchers can now be selected/ unselected and various actions such as deletion, modification, etc. can now be performed on the same. Currently, this feature is available will work in Display mode only.

To achieve this,

Following attributes have been introduced / altered:

Following attributes have been introduced / altered:

- In Key description,
  Scope Attribute allows specifying a scope for the Action(s) specified in 'Action' Attribute

  Syntax:

  ```
  Scope : <Scope Keyword>
  ```

  Scope Keywords possible are **Current Line / Line**, **All Lines**, **Selected Lines, Unselected Lines** and **Report**

- In Part description,

Selectable Attribute indicates if the lines owned by this Part are selectable or not & the default value for the same is **Yes**.

Syntax:

```
Selectable : <Logical Formula>
```

- In Line Description,
  Selectable Attribute indicates if the line (or lines within this line) is/are selectable or not (Defaults to 'Yes' for Repeated lines and 'No' for non-repeated lines). Also inherits from Parent Part/Line & the same can be overridden here.

  Syntax:

  ```
  Selectable : <Logical Formula>
  ```

1. Following actions have been introduced/ changed.

   - **Toggle Select** – Selects / deselects a line
   - **Select All** – Selects all the lines within a part
   - **Unselect All** – Deselects all the lines within a part
   - **Invert Selection** – Selects all the unselected lines within a part
   - **Modify Object**: – Modifies the values stored in the methods of an Object

   The behaviour of existing actions **Cancel Object**, **Delete Object, Remove Line** and **Multi Field Set** have been modified to obey the scope specified in the Key description.

   Actions like **Cancel Object, Audit Object, Delete Object** and **Modify Object** enhanced to work with **Report** scope.

2. Some additional color formulae have been introduced in TDL to allow background, foreground and cursor colors while they are selected or unselected.

   - **SV_SEL_CURSOR_BG-** Selection + Current Background Color
   - **SV_SEL_CURSOR_FG-** Selection + Current Foreground Color
   - **SV_SELECTED_BG -** Background Color for the selected lines
   - **SV_SELECTED_FG -** Foreground Color for the selected lines

   Note: If an action is fired and scope is not defined in the Key, it defaults to Selected Lines if there are any selections else defaults to Current Line. This will reduce the number of Key definition to support a given action with two different scopes.

   Sample TDL Code demonstrating the above

New Action - Modify Object

Similar to **Direct Access** Action **Modify Object** is enhanced to alter a method of an object at any level. Modify Object action also supports modifying multiple values of an object by a comma separated specification of Method : Value pair. Here is an example which modifies the First Bill Allocation's Opening Balance amount to 100 as well as last line of address to "Bangalore". If multiple values are specified Primary object specification is allowed only at first level. A single Modify Object action cannot modify Multiple Objects, but can modify Multiple Values of an Object.

Syntax:

Action : Modify Object :
<PrimaryObjectSpec>.<SubObjectPathSpec>.MethodName : value>[,Method Name: <value> , …] [,<SubObjectPathSpec>.MethodName :<value>, …..]

The specifications given for <PrimaryObjectSpec>, <SubObjectPathSpec>, MethodName remain same as described in the New Method syntax section.

Example 1*:*

```
[Key: Alter My Object]
    Key     : Ctrl + Z
    Action  : Modify Object : +
   (Ledger,"MyLedger").BillAllocations[First, $Name="MyBill"].OpeningBalance : 100, +
   Address[Last].Address : "Bangalore"
```

Example 2*:*

```
[Key: Alter My Object]
    Key     : Ctrl + Z
    Action  : Modify Object : +
   (Ledger,"MyLedger").BillAllocations[1].OpeningBalance : 1000,Name:"My New Bill", +
   Address[First].Address : "Hongasandra Bangalore ",Opening Balance:5000
```

Example 3: In Voucher context

```
[Key : Alter My Object]
    Key :Ctrl + Z
    Action: ModifyObject: LedgerEntries[1].BillAllocations[1].Name : "Test1", Amount :
"1000.00", ..BillAllocations[2].Name : "Test2", Amount : "2000.00", ().Date : "1.4.08"
```

Notice the use of double dot and () operator in the above snippet

Note:

1. **Modify Object** is allowed to have Primary Object Specification only once i.e., for the first Value. Further values permissible are optional Sub Object and Method Specification only.

2. **Sub Object Specification** is optional for second value and onwards.

    a. If Sub Object Specification is specified, the context is assumed to be the Primary Object specified for the first value.
    b. In absence of Sub Object Specification, the previous value specification's Leaf Object is considered as the context.

3. Keys with Action **Modify Object** are supported both at Menu and Report Definition. In a Menu Definition,

    a. Since menu does not have any Info Objects in context, specifying Primary Object becomes mandatory.
    b. Since Menu cannot work on scopes like Selected, Unselected, etc. scopes specified are ignored.
    c. Any formula specified in the value is evaluated assumes Menu Object as requestor.
    d. Even Method values pertaining to Company Objects can be modified.
    e. A button can be added at the menu to specify the action Modify Object at the menu level

New Action - Browse URL

A New action Browse URL has been introduced in this release. This can be used to open web browser with any URL formula passed as a parameter.

Syntax:

```
Action: Browse URL: <URL>
```

Example: Field acting as a hyperlink

```
[Key : Execute Hyperlink]
    Key       : Left Click
    Action    : Browse URL : “www.tallysolutions.com”

[Field: Hyperlink Company]
    Color     : Blue
    Border    : Thin Bottom
    Key       : Execute Hyperlink
    Set as    : "Tally Solutions Pvt. Ltd"
    Local     : Key  : Execute Hyperlink  : Action : Browse URL: +
                                            http://www.tally.co.in
```

Note: *Existing Action* ***Register Tally*** *has been removed for generalization which is now replaced with Action* ***Browse URL****.*

Enhanced Actions - Print/Mail/Export/Upload Report

The actions Print Report, Mail Report, Export Report, Upload Report is now enhanced to obey scope and also to print a report other than the current.

Syntax:

```
Action : Print Report       [: ReportName]
Action : Mail Report        [: ReportName]
Action : Export Report      [: ReportName]
Action : Upload Report      [: ReportName]
```

The above actions can use the scope as All Lines, Selected Lines, Unselected Lines, Current Line or Report. They also support an additional optional parameter which is the Report Name. This allows us to use the different Report using one of the contexts specified in the scope.

Example:

```
/* This button can be attached to any Report displaying list of employees, for e.g., List of Accounts ->
Employees */

[Button: Print Selected Ledgers]

    Key         : Alt + F1
    Action      : Print Report : Multi Payslip Print
    Scope       : Selected Lines

/* Now the existing Report Multi Payslip Print must be altered to include the Collection Parameter
Collection */

[#Report: Multi Payslip Print]

    Collection  : Parameter Collection
```

Now the Report Ledger Vouchers can use an internally available collection Parameter Collection to achieve Multi Account Printing using the Report Attribute Collection. This collection can also be used for repeating a line, etc.

Note:

- Absence of **Report Name** parameter will default to the current Report Name.
- If Scope is not specified for the above **actions**, it defaults to **Report** Scope.
- **Report** Scope makes the Report Level Object available in the Acted Report.
- **Selected Lines, Current Line, Unselected Lines** and **All Lines** Scopes make an internal collection **Parameter Collection** available to the Acted Report.
- This **Internal Collection** will have Objects which are in the specified scope in the Main Report.

Sample TDL Code demonstrating the above

Switch attribute support in TDL

"Switch" is a new attribute modifier that has been incorporated in this release. This attribute is similar to the Option attribute but reduces code complexity and improves the performance.

The function Option compulsorily evaluates the conditions for all the options provided in the description code and applies all which satisfy the evaluation conditions. This means that it is not always easy to write the code where you just want one of the options to be applied, you have to make sure that other options are not applied using a negative condition. The new attribute provider Switch has been provided to support these types of scenarios where evaluation is carried out only up to the point where the first evaluation process has been cleared.

Apart from this, the Switch can be grouped using a label. Therefore, multiple switch groups can be created and zero or one of the switch cases would be applied from each such group.

Apart from this, the switch can be grouped using a label, as shown below:

```
Switch : Label : Desc-name : Condition
Switch : Label : Desc-name : Condition
```

Sample TDL Code demonstrating the above

Field Attribute, SetbyCondition

A new field attribute SetbyCondition has been introduced in this Release. This is a conditional "Set as" at Field Level. This attribute behaves similar to Switch Case but with this attribute, we can avoid defining multiple optional fields and complex if else statements. The last satisfied SetbyCondition will be executed.

Sample TDL Code demonstrating the above

Field Attribute, ToolTip

A new Field attribute ToolTip has been introduced in this Release. As the name suggests, the value specified with this attribute is displayed when the mouse pointer is placed on a particular field. This means that in addition to the static information displayed by "Info or Set as" attributes; we can also communicate additional meaningful information with the help of this attribute. As against Attributes "Info or Set as", this attribute value is independent of the Field Width. In other words, when the user hovers the mouse pointer over the Field, a small hover box appears with supplementary information regarding the item being hovered over.

Sample TDL Code demonstrating the above

Multi- line commenting in TDL source code using /* and */

Multi-line commenting is a new feature in this release which renders the TDL code more user-friendly and easy to maintain. A simple Multi-line comment would look like:

```
/*
<Comment Line 1>
<Comment Line 2>
<Comment Line 3>
*/
```

Extension of modifying definitions using

The scope of modifying definitions using # is extended to System Formula Definition. Now, you can alter the value of the existing system formula using #. This helps to improve the performance with optimized formulae.

```
/*
The following code alters the existing System Formula to change the values specified to Formulae NameWidth and
MaxNameWidth in DefTDL
*/
[#System: Formula]

   NameWidth         : 40
   MaxNameWidth      : 60

;; End-of-code
```

Unlimited Namespace support

The new TDL language processing engine supports unlimited name-space allowing large TDLs to be loaded without any constraint of internal name-space.

'*' (Reinitialize) Description operator introduced

This is similar to description operators such as '#' (Modify), '!' (Option) and '@' Const/Fixed, when used for an existing description; overwrites the existing description. This is very useful when there is a need to completely replace the existing description content with a new code.

Sample TDL Code demonstrating the above

2. Enhanced integration capabilities with HTTP-XML collection and HTTP-POST action

Collection capability has been enhanced to gather live data from HTTP/web-service delivering XML. The entire XML is automatically now converted to TDL objects and is available natively in TDL reports as $ based methods. There is no need to access the data via specialized functions like $$XMLValue. Reports can be shown live from an HTTP server. Coupled with the new [OBJECT:] extensions and POST action you can also submit data back to the server almost operating Tally as a client to HTTP-XML web-services.

HTTP-XML collection

Following are the attributes in collection for gathering XML based collection from a remote server over HTTP

[Collection: MyXMLCollection]

RemoteURL : http-url
RemoteRequest : request-report-name, pre-request-display-report : encoding type
XMLObjectPath : Start-node: Path-to-start-node
XMLObject : tdl-object-name

where

- *Remote-URL* attribute is used to specify the URL of the HTTP server delivering the XML data
- *RemoteRequest attribute is used to specify the R*eport name which is to be sent to the HTTP server as an XML Request. If the report requires user inputs then it has to be accepted before the request is sent.
- *XMlObjectPath attribute* is used to take only a specific fragment of the response XML and converts the same to TDL Objects in Collection. By default, it takes the root node.

Start-Node = Node Name : Position
Path-to-Start-Node = Root-node : Child Node : Start Pos : Child Node : Start Pos ...

- *XMLObject* attribute is used to specify the name of the TDL Object to be stored in the Tally DB.

The following syntax is used for object specification

[Object: Object Name]

Storage : Name : Type

Collection : Name : Type

/*The 2nd Parameter in the Collection, Type can be a Object type in case of a complex collection or a simple data type in case of simple collection*/

All these attributes cater to specific requirements

Let us consider following four cases:

Case 1: Simple GET Request

If it is required to access the data (XML format) from remote server in a collection it is sufficient to specify the URL of the server only. The data thus obtained is available in the collection as objects and can be accessed as native methods.

XML Data available at http://Remoteserver/Test.xml

```
<Customer>
    <Name>Customer 1</Name>
    <Place>Bangalore</Place>
</Customer>
```

Example:

[Collection: TestHttpDetails]

RemoteURL : “http://Remoteserver/Test.xml”

This collection can be used in a TDL Report.The method names will be same as the XML tag names.

Sample TDL Code demonstrating the above

Case 2: Simple GET Request and mapping the response to TDL Object

The data thus obtained in the collection can be stored in Tally DB as TDL Objects. The following example demonstrates this capability.

Example:

[Collection: TestHttpDetails]

RemoteURL : “http://Remoteserver/Test.xml”

XMLObjectPath : CUSTOMER

XMLObject : Customer

[Object : Customer]

Storage : Name : String

Collection : PrimaryAddress : String ;; Simple

Collection : OTHERADDERSS : MyOtherAddressCollection ;; Complex

[Object : MyOtherAddressCollection]

Storage : Line1 : String

Storage : Line2 : String

Storage : Line3 : String

Sample TDL Code demonstrating the above

Case 3: Request to Post a Request with and without a Pre Request Report

If a TDL report is to be sent to the HTTP server as an XML request and the XML response is to be obtained in the collection, then the attribute Remote Request has to be used.

Example 1

[Collection: TestHttpDetails]

RemoteURL : http://Remoteserver/Test.asp

Remote Request: Request Rep

A Pre Request Report is required when some inputs are to be accepted from the user and the XML Request is to be generated out of those inputs. In that case a TDL report is used in create mode and has be accepted first. This report name is specified as Pre Request Report.

[Collection: TestHttpDetails]

RemoteURL : http://Remoteserver/Test.asp

Remote Request : Request Rep, Input Rep

HTTP POST Action

A new key/button action HTTP Post has been introduced which will help in exchanging data with external applications using web services. In other words, Http Post Action can be used to submit data to a server over http and gather the response. This will enable a TDL Report to perform a HTTP Post to a remote location.

Syntax:

```
[Key: <Key Name>]
        Key    : <Key Combination>
        Action : HTTP Post : <URL Formula> : <Encoding> : <Request
                                        Report>[, <Response Report>]
```

The details pertaining to URL (at the receiving end), Encoding Format, Request Report and the Response Report should be specified along with HTTP Post Action.

- Universal Resource Locator (URL) for which information is intended has to be specified through a System Formula.
- Encoding Format specifies the encoding to be used while transmitting information to the receiving end. The valid encoding formats are ASCII and UNICODE. UNICODE is set by default.
- Request Report is the name of the TDL Report which will be used for generating XML Request to be sent.
- Response Report is optional and will enable the programmer to display a Report with the details of the XML response received.
- Both the Requests / Response are exchanged in XML format.

Example:

```
[Key: XMLReqResp]
        Key    : Ctrl + R
        Action : HTTP Post : @@XMLR : ASCII : ReqRep, RespRep
        Scope  : Selected Lines

[System : Formula]
        XMLR : http://127.0.0.1:9000  ;; URL Specification must be done here
```

The defined Key **XMLReqResp** in the snippet above must be attached to an initial Report. When the report is activated and this Key is pressed, the Action **HTTP Post** activates a defined report **ReqRep** which generates the **request XML** and the response is sent to the new report **RespRep** which uses the response received to display in the Report.

3. Behavioral changes in TDL

Changes in Set as / Info Attributes

As of Release 2.x, the attributes Set as and Info were treated as the same attribute with aliases, when 'Info' is used, it had a special Skip and Prompt privilege. If both were specified the last specification would override the previous specification and would be the effective specification.

Tally 9 Release 3 onwards, this behavior has been modified to treat both attributes as individual attributes. When both these attributes are specified in any field, 'Info' always takes the precedence and 'Set as' is ignored. In other words, Info carries more privilege than Set AS.

Sample TDL Code demonstrating the above

Changes in Attribute "Use"

The behavior of attribute USE which is used to inherit the properties from other definition has now changed. Irrespective of their order across other attributes, USE will be evaluated first. In other words, the order in which USE is specified is immaterial as in any case it will be evaluated first. If multiple USE attributes are specified in a single definition, they are evaluated in the order of their occurrence.

Sample TDL Code demonstrating the above

Changes in Delayed Attributes "Add/Delete/Replace"

Delayed Attributes like Add, Delete and Replace are now generalized across all definitions. Earlier for definitions Report, Key, Color, Style, Border and Variable, the delayed attributes were expanded as per the sequence across the other attributes. Now they behave like all other definitions. This has been done to bring consistency across the definitions.

```
;; Example - Old behavior
;; End result of this description is the report will have *ONE* form 'Form2'

[Report: Test Report]

    Form    : Form1
    Delete  : Form
    Form    : Form2

;; End result of this description is the Part will have *NO* Lines
```

```
[Part: Test Part]

    Line      : Line1
    Delete    : Line
    Line      : Line2

;; Example - New behavior
;; End result of this description is the report will have *NO* forms

[Report: Test Report]

    Form      : Form1
    Delete    : Form
    Form      : Form2

;; End result of this description is the Part will have *NO* Lines

[Part: Test Part]

    Line      : Line1
    Delete    : Line
    Line      : Line2
```

The above code snippet clearly distinguishes the current behavior with the behavior prior to Tally 9 Release 3.

Sample TDL Code demonstrating the above

Change in Delayed Attribute "Local"

Delayed Attribute Local which is used to locally modify the attributes of any child definition is now enhanced to accept nested Locals. Prior to this release, if one needs to locally modify the value of a Field within any line (which is locally modified within the line), deleting the field and adding a new field becomes necessary. This problem is now eliminated by supporting multiple locals within a single local statement.

```
[Line: TitleLine]

    Use       : DetailLine
    Local     : Field : AmtField : Set as : "Purchase Amount"

;; Old Behavior

[Report: Sales Report]

    Use   : Purchase Report

    Local : Field : AmtField : Set as : "Sales Amount"
;;  The above will not work since the Field is locally modified within the Line TitleLine which overrides the
;;  above local statement.  Therefore, the following lines becomes necessary
    Local : Line : TitleLine      : Delete : Right Field : AmtField
    Local : Line : TitleLine      : Add    : Right Field : At Beginning : SalesAmtField

    [Field: SalesAmtField]

        Use      : AmtField
```

```
        Set as   : “Sales Amount”
;; New Behavior

[Report: Custom Report]

    Local  : Line : TitleLine : Local : Field : AmtField : Set as : “Sales Amount”

;;  One can achieve the desired result in a single line by locally modifying the Line and the Field
;;  within the Line.  This eliminates the need for unimportant definitions and increases efficiency in coding.
;;  Also enables the TDL Programmers to code faster.
```

Change in usage of 'BLANK' Keyword in Menu Items

To insert empty line between menu items, BLANK keyword was used. Also Item Attribute without any "Value" is considered as BLANK. For consistency in TDL coding, the later is now disallowed. Only BLANK keyword can now be used to indicate an empty menu item.

Sample TDL Code demonstrating the above

Partial Attribute Support discontinued

Prior to Release 3.0, all descriptions supported partial search on their attribute words, for e.g., Set as could have been written as Set a, Set or Se, which would allow minimum number of characters to be present to an extent where another attribute does not start with those characters. This behavior is now removed as it is not practical to use partial words. But multiple aliases are now supported to allow meaningful attribute names.

For example,

- Set as can be written as Set
- Float Bottom Lines at Part Definition can be written as Float
- Top Part can be written as Part, Parts or Top Parts

Since these aliases have been introduced, most of the existing TDL will work without any changes. In case of Partial words / Non-meaningful words used in any TDL, Tally would throw an error which needs to be corrected in TDL.

4. Sample TDL Codes

New Collection Attributes

```
/*

The following report displays all the Stock Items computing the Total Quantity sold along with Maximum & Minimum
rates at which they are sold for a specified period.  This program uses the Aggregate Types; Sum, Min and Max.

*/

[#Menu: Inventory Books]

    Add      : Key Item        : Product Sales : R : Display        : ProductwiseRpt

[Report: ProductwiseRpt]

    Use      : DSP Template
    Form     : ProductwiseRpt
    Title    : "Product-wise Sales Report"

[Form: ProductwiseRpt]

    Use      : DSP Template
    Parts    : ProductwiseRptHdr, ProductwiseRpt
    Height   : 100% Screen
    Width    : 100% Screen
    Delete   : Buttons         : ExplodeFlag
    Delete   : Bottom Buttons : DSPAutoColumns, BudgetAnalysis

[Part: ProductwiseRptHdr]

    Lines    : ProductwiseRptHeader

    [Line: ProductwiseRptHeader]

        Field          : Name Field
        Right Field    : DSP MainDateTitle
        Local : Field  : Name Field     : Set as          : $$LocaleString:"Inventory Details"
        Local : Field  : Name Field     : Style           : Normal Bold
        Space Bottom: 1

[Part: ProductwiseRpt]

    Lines             : ProductwiseRptTitle, ProductwiseRptInfo
    Repeat            : ProductwiseRptInfo              : MinMaxStockItem
    Scrolled          : Vertical
    Common Borders: Yes
    Border            : Thin Bottom
    Float             : No

    [Line: ProductwiseRptTitle]

        Use      : ProductwiseRptInfo
        Local    : Field       : Default                : Type           : String
        Local    : Field       : Default                : Lines          : 0
        Local    : Field       : Default                : Style          : Normal Bold

        Local    : Field       : ProductTotalAmt        : Align          : Centre
        Local    : Field       : ProductMaxRate         : Align          : Centre
```

```
        Local     : Field        : ProductMinRate          : Align          : Centre
        Local     : Field        : ProductName             : Set as         : "Particulars"
        Local     : Field        : ProductTotalAmt         : Set as         : "Total Quantity"
        Local     : Field        : ProductMaxRate          : Set as         : "Maximum Rate"
        Local     : Field        : ProductMinRate          : Set as         : "Minimum Rate"
        Border    : Thin Column Titles

    [Line: ProductwiseRptInfo]

        Fields         : ProductName
        Right Fields   : ProductTotalAmt, ProductMaxRate, ProductMinRate
        Local: Field   : Default: Style   : Normal

        [Field: ProductName]

            Use      : Name Field
            Set as   : $Name
            Display  : Stock Vouchers
            Variable : Stock Item Name

        [Field: ProductTotalAmt]

            Use      : Qty Primary Field
            Set as   : $$CollectionField:$StkTotalQty:1:SummaryColl
            Border   : Thin Left Right
            Width    : 10

        [Field: ProductMaxRate]

            Use      : Rate Field
            Set as   : $$CollectionField:$StkMaxRate:1:SummaryColl
            Border   : Thin Right
            Width    : 10

        [Field: ProductMinRate]

            Use      : Rate Field
            Set as   : $$CollectionField:$StkMinRate:1:SummaryColl
            Width    : 10

;; Collection Definition begins

[Collection: MinMaxStockItem]

    Type         : Stock Item
    Filter       : NonEmpty Items

[Collection: SalesVouchersColl]

    Type         : Voucher
    Filter       : OnlySalesCreditNote

[Collection: SummaryColl]

    Source Collection: SalesVouchersColl

    Walk             : Inventory Entries

    NativeMethod     : Stock Item Name, Date

    By               : StkItemName   : $StockItemName

    Aggr Method      : StkTotalQty     : Sum            : $BilledQty

;; Total the Billed Quantity for each Stock Item
```

```
    Aggr Method     : StkMinRate     : Min            : $Rate

;; Compute Minimum Rate at which Item is sold for a given period

    Aggr Method     : StkMaxRate     : Max            : $Rate

;; Compute Maximum Rate at which Item is sold for a given period

    Keep Source     : No

    Filter          : SearchForStockItem

[System: Formula]

    NonEmpty Items          : $StkOutQty > 0
    OnlySalesCreditNote     : $$IsSales:$VoucherTypeName OR $$IsCreditNote:$VoucherTypeName
    SearchForStockItem      : $StockItemName = #ProductName

;; End-of-code
```

Back

CollectionFieldByKey Function

```
/*

The following report lists all the Stock Items with their Opening, Inwards with Purchase & Transfers, Outwards with
Sales & Transfers and Closing Balance using the new function '$$CollectionFieldByKey'. The function enables the TDL
programmer to create *in-memory indexed collections* i.e., collection based on 'Search Key' assigned.  This indexed
collection will have significant performance improvement.

*/

[#Menu: Inventory Books]

    Add     : Key Item       : Item Transaction Breakup : R : Display    : Stock Index Coll

[Report: Stock Index Coll]

    Use     : DSP Template
    Form    : Stock Index Coll
    Title   : $$LocaleString:"Stock Report using Index Collection"

[Form: Stock Index Coll]

    Use     : DSP Template
    Parts   : Stock Index Coll Header, Stock Index Coll Main
    Height  : 100% Page
    Width   : 100% Page

[Part: Stock Index Coll Header]

    Lines           : Stock Index Coll Header
    Space Bottom    : 1

     [Line: Stock Index Coll Header]

        Fields        : Name Field
        Right Fields  : DSP MainDateTitle

        Local : Field  : Name Field     : Set as  : $$LocaleString:"Inventory Details"
        Local : Field  : Name Field     : Style   : Normal Bold
```

```
[Part: Stock Index Coll Main]

    Lines            : Stock Index Coll Titles1, Stock Index Coll Titles2, Stock Index Coll Details
    Bottom Lines     : Stock Index Coll Totals
    Repeat           : Stock Index Coll Details  : Stock Item Coll
    Scroll           : Vertical
    Float            : Yes
    Total            : Stock IC OpenBal, Stock IC Inwards, Stock IC InwardsT, Stock IC Outwards, +
                       Stock IC OutwardsT, Stock IC ClosBal
    CommonBorder     : Yes
    [Line: Stock Index Coll Titles1]

        Fields       : Stock IC ItemName
        Right Fields : Stock IC OpenBal, Stock IC Inwards, Stock IC Outwards, Stock IC ClosBal

        Local: Field : Default                : Type          : String
        Local: Field : Default                : Style         : Small Bold
        Local: Field : Default                : Align         : Center
        Local: Field : Stock IC Inwards       : Width         : 13.5
        Local: Field : Stock IC Outwards      : Width         : 12
        Local: Field : Stock IC Inwards       : Border        : Full Thin Bottom
        Local: Field : Stock IC Outwards      : Border        : Full Thin Bottom

        Local: Field : Stock IC ItemName      : Set AS        : $$LocaleString:"Particulars"
        Local: Field : Stock IC OpenBal       : Set AS        : $$LocaleString:"Opening"
        Local: Field : Stock IC Inwards       : Set AS        : $$LocaleString:"Inwards"
        Local: Field : Stock IC Outwards      : Set AS        : $$LocaleString:"Outwards"
        Local: Field : Stock IC ClosBal       : Set AS        : $$LocaleString:"Closing"
        Border       : Thin Top
        Fixed        : Yes

    [Line: Stock Index Coll Titles2]

        Use          : Stock Index Coll Details

        Local: Field : Default                : Type          : String
        Local: Field : Default                : Style         : Small Bold
        Local: Field : Default                : Align         : Center

        Local: Field : Stock IC ItemName      : Set AS        : $$LocaleString:""
        Local: Field : Stock IC OpenBal       : Set AS        : $$LocaleString:"Balance"
        Local: Field : Stock IC Inwards       : Set AS        : $$LocaleString:"Purchase"
        Local: Field : Stock IC InwardsT      : Set AS        : $$LocaleString:"Transfers"
        Local: Field : Stock IC Outwards      : Set AS        : $$LocaleString:"Sales"
        Local: Field : Stock IC OutwardsT     : Set AS        : $$LocaleString:"Transfers"
        Local: Field : Stock IC ClosBal       : Set AS        : $$LocaleString:"Balance"
        Border       : Thin Bottom
        Fixed        : Yes

    [Line: Stock Index Coll Details]

        Fields       : Stock IC ItemName
        Right Fields : Stock IC OpenBal, Stock IC Inwards, Stock IC InwardsT, Stock IC Outwards, +
                       Stock IC OutwardsT, Stock IC ClosBal

    [Line: Stock Index Coll Totals]

        Use          : Stock Index Coll Details

        Local: Field : Default                : Style         : Normal Bold
        Local: Field : Stock IC ItemName      : Align         : Center

        Local: Field : Stock IC ItemName      : Set AS        : $$LocaleString:"Total"
        Local: Field : Stock IC OpenBal       : Set AS        : $$Total:StockICOpenBal
        Local: Field : Stock IC Inwards       : Set AS        : $$Total:StockICInwards
```

```
        Local: Field   : Stock IC InwardsT       : Set AS          : $$Total:StockICInwardsT
        Local: Field   : Stock IC Outwards       : Set AS          : $$Total:StockICOutwards
        Local: Field   : Stock IC OutwardsT      : Set AS          : $$Total:StockICOutwardsT
        Local: Field   : Stock IC ClosBal        : Set AS          : $$Total:StockICClosBal
        Border         : Flush Column Titles
        Fixed          : Yes

;; Field Definition begins

[Field: Stock IC ItemName]

    Use          : Name Field
    Set AS       : $Name
    FullWidth    : Yes
    Display      : Item Monthly Summary
    Variable     : Stock Item Name

[Field: Stock IC OpenBal]

    Use          : Qty Primary Field
    Set AS       : $OpeningBalance
    Width        : 12
    Border       : Thin Left
    Format       : NoZero

[Field: Stock IC Inwards]

    Use          : Stock IC OpenBal
    Set AS       : $StockICInwards
    Border       : Thin Left
    Width        : 6

[Field: Stock IC InwardsT]

    Use          : Stock IC OpenBal
    Set AS       : $StockICInwardsT
    Width        : 6
    Delete       : Border

[Field: Stock IC Outwards]

    Use          : Stock IC OpenBal
    Set AS       : $StockICOutwards
    Border       : Thin Left
    Width        : 6

[Field: Stock IC OutwardsT]

    Use          : Stock IC OpenBal
    Set AS       : $StockICOutwardsT
    Width        : 6
    Delete       : Border

[Field: Stock IC ClosBal]

    Use          : Stock IC OpenBal
    Set AS       : $ClosingBalance
    Border       : Thin Left

;; Collection Definition begins

[Collection: Stock Item Coll]

    Type         : Stock Item

[Collection: VCH Source InColl]
```

```
    Type          : Voucher
    Filter        : InFilter

[Collection: VCH Summary InColl]

    Source Collection : VCH Source InColl

   ;; the Inventory Entries object is available directly under Voucher Object
   ;; in case of multilevel hierarchy, we can specify as 'Object type, Sub Object type, Sub Object type...'
   ;; for e.g., Cost Centre Allocations can be specified as
   ;; 'All Ledger Entries, Cost Category Allocations, Cost Centre Allocations'

    Walk              : Inventory Entries

   ;; the following method will refer from the least object and minimum one 'By' method should be available

    By                : Name          : $StockItemName

   ;; minimum one 'Aggr Method' is required for the summary collection

    Aggr Method       : BillQty       : Sum    : $BilledQty

   ;; the below can be a list for e.g., '$Name + $StockItem + ...'

    Search Key        : $Name

[Collection: VCH Source OutColl]

    Type              : Voucher
    Filter            : OutFilter

[Collection: VCH Summary OutColl]

    Source Collection : VCH Source OutColl
    Walk              : Inventory Entries
    By                : Name          : $StockItemName
    Aggr Method       : BillQty       : Sum    : $BilledQty
    Search Key        : $Name

[Collection: VCH Source InTColl]

    Type              : Voucher
    Filter            : InOutTFilter

[Collection: VCH Summary InTColl]

    Source Collection : VCH Source InTColl
    Walk              : Inventory Entries In
    By                : Name          : $StockItemName
    Aggr Method       : BillQty       : Sum    : $BilledQty
    Search Key        : $Name

[Collection: VCH Source OutTColl]

    Type              : Voucher
    Filter            : InOutTFilter

[Collection: VCH Summary OutTColl]

    Source Collection : VCH Source OutTColl     ;; VCH Source InTColl
    Walk              : Inventory Entries Out
    By                : Name          : $StockItemName
    Aggr Method       : BillQty       : Sum    : $BilledQty
    Search Key        : $Name
;; Object Alteration begins
```

```
[#Object: Stock Item]

;; the following function '$$CollectionFieldByKey' will go and directly LOCATE the record that is indexed at the
collection
;; based on the search key. It doesn't need to scan through the collection to fetch the value.

     Stock IC Inwards          : $$ReportObject:$$CollectionFieldbyKey:$BillQty:$Name:VCHSummaryInColl
     Stock IC InwardsT         : $$ReportObject:$$CollectionFieldbyKey:$BillQty:$Name:VCHSummaryInTColl
     Stock IC Outwards         : $$ReportObject:$$CollectionFieldbyKey:$BillQty:$Name:VCHSummaryOutColl
     Stock IC OutwardsT        : $$ReportObject:$$CollectionFieldbyKey:$BillQty:$Name:VCHSummaryOutTColl

;; if the new capability had not been available, the above would have been written as follows

;;   Stock IC Inwards          : $$ReportObject:$$FilterValue:$BillQty:VCHSummaryInColl:First:ForThisItem
;;   Stock IC InwardsT         : $$ReportObject:$$FilterValue:$BillQty:VCHSummaryInTColl:First:ForThisItem
;;   Stock IC Outwards         : $$ReportObject:$$FilterValue:$BillQty:VCHSummaryOutColl:First:ForThisItem
;;   Stock IC OutwardsT        : $$ReportObject:$$FilterValue:$BillQty:VCHSummaryOutTColl:First:ForThisItem

;; System Formula begins

[System: Formula]

     InFilter          : $$IsPurchase:$VoucherTypeName OR $$IsDebitNote:$VoucherTypeName
     OutFilter         : $$IsSales:$VoucherTypeName OR $$IsCreditNote:$VoucherTypeName
     InOutTFilter      : $$IsStockJrnl:$VoucherTypeName

     ForThisItem       : $Name = $$ReqObject:$Name

;; End-of-code
```

Back

IsObjectBelongsTo Function

```
/*

The following auto column report is generated using the enhanced collection abilities. The collections are created in
two levels, first level will summarize the data using the vouchers as a source collection and the second level will
extract the information from the summary level (first level) collection for the sub totals.

*/

[#Menu: Payroll Reports]

     Add      : Key Item        : Payroll Columnar : R : Display     : Sample Columnar

[Report: Sample Columnar]

   Use        : DSP Template
   Title      : $LocaleString:"Pay Sheet"
   Variable   : GroupName, PayHeadName, LedgerName, CostCategoryName, CostCentreName
   Variable   : DoSetAutoColumn, DSPRepeatCollection, SVFromDate, SVToDate
   Repeat     : PayHeadName
   Set        : PayHeadName             : #PayHeadName
   Set        : DoSetAutoColumn         : Yes
   Set        : DSPRepeatCollection     : "Pay Heads"
   Set        : SVFromDate              : $$MonthStart:##SVCurrentDate
   Set        : SVToDate                : $$MonthEnd:##SVCurrentDate
   Form       : Sample Columnar
```

```
[Form: Sample Columnar]

   Use        : DSP Template
   Part       : Sample ColumnarTitle, Sample Columnar
  Option      : Set Auto Col Option       : ##DoSetAutoColumn AND $$SetAutoColumns:PayHeadName

[Part: Sample ColumnarTitle]

   Lines      : Sample ColumnarTitle

[Part: Sample Columnar]

   Lines              : Sample Columnar
   Repeat             : Sample Columnar        : PayHead SourceCC
   Scroll             : Vertical
   Common Border      : Yes

   [Line: Sample ColumnarTitle]

      Fields        : Sample EmpName, Sample EmpSalary
      Repeat        : Sample EmpSalary
      Local : Field: Sample EmpSalary   : Type            : String
      Local : Field: Sample EmpSalary   : Align           : Centre
      Local : Field: Sample EmpName     : Set as          : "Employee Name"
      Local : Field: Sample EmpSalary   : Set as          : ##PayHeadName
      Border        : Flush Totals

     [Line: Sample Columnar]

      Fields        : Sample EmpName, Sample EmpSalary
      Explode       : Sample Columnar    : $$NumItems:PayHeadChildCostCentres > 0
      Indent        : $$ExplodeLevel * 2
      Repeat        : Sample EmpSalary
      Total         : Sample EmpSalary

;; Field Definition begins

[Field: Sample EmpName]

    Use           : Name Field
    Set as        : $Name
    Style         : If $$ExplodeLevel > 2 then "Small" else "Small Bold"
    Fixed         : Yes

[Field: Sample EmpSalary]

    Use           : Amount Field
    Set as        : If $$IsEmpty:@PayHeadAmt then @SubTotAmt else @PayHeadAmt
    SubTotAmt     : If $$IsEmpty:$CCPaySubTot then $CCAggrPaySubTot else $CCPaySubTot

   ;; the method $CCAggrPayHeadFor itself capable of giving the value for the all the cost
   ;; centres. But using the method $CCPayHeadFor will increase the performance in case of
   ;; least level cost centre

    PayHeadAmt : If $$IsEmpty:$CCPayHeadFor then $CCAggrPayHeadFor else $CCPayHeadFor
    Style         : If $$ExplodeLevel > 2 then "Small" else "Small Bold"
    Align         : Right
    Border        : Thin Left Right

;; Object Alteration begins

[#Object: Cost Centre]

   ;; this can be directly referred for the cost centres having same name (least level)
```

```
    CCPayHeadFor        : +
$$ReportObject:$$CollectionFieldByKey:$CCTotAmount:@@locLedCCSearch:PayHeadEmpCollection

   ;; this is required to aggregate the amount for the groups

    CCAggrPayHeadFor    : $$ReportObject:$$FilterAmtTotal:PayHeadEmpCollection:ForThisChildCC:$CCTotAmount

   ;; this below method is used to arrive at the sub totals for earnings & deductions

    CCPaySubTot         : +
$$ReportObject:$$CollectionFieldBYKey:$ComponentSubTot:@@locLedCCSearch:PayHeadSubTotals

   ;; this is required to aggregate the sub total amount for the groups

    CCAggrPaySubTot     :
$$ReportObject:$$FilterAmtTotal:PayHeadSubTotals:ForThisCompType:$ComponentSubTot

    ParentFullList      : If $$IsSysName:$Parent then "" else $ConcatenateParent
    ConcatenateParent   : "^"+ $Parent + "^" + $RecursiveParent:CostCentre:$Parent
    RecursiveParent     : $ParentFullList

;; Collection Definitions begins

[Collection: PayHead SourceCC]

    Type        : Cost Centre
    Filter      : PayHeadEmpCat
    ChildOf     : $Name
    Sort        : Default     : $Name

[Collection: PayHeadVch Collection]

    Type        : Voucher

[Collection: PayHeadEmp Collection]

    Source Collection : PayHeadVch Collection
    Walk              : All Ledger Entries, Category Allocations, Cost Centre Allocations
    By                : Name                : $Name
    By                : Ledger Name         : $LedgerName
    Aggr Method       : CCTotAmount         : Sum    : $Amount
    Search Key        : $Name + $LedgerName
    Keep Source       : Yes

[Collection: PayHead SubTotals]

    Source Collection : PayHeadEmp Collection
    By                : Name                : $Name
    By                : ComponentType       : $Parent:Ledger:$LedgerName
    Aggr Method       : ComponentSubTot     : Sum    : $CCTotAmount
    Search Key        : $Name + $ComponentType
    Keep Source       : Yes

[Collection: PayHeadChildCostCentres]

    Type              : Cost Centre
    ChildOf           : $Name
    BelongsTo         : Yes


[Collection: Pay Heads]

    Object            : TotDeductions, TotEarnings
    Collection        : Pay Head Ledgers
    Sort              : Default          : @@ParentIndexPosition
```

```
[Object: TotEarnings]

    Name            : @@EarningGrpName
    PayHeadName     : $Name
    SortPosition    : 5002

[Object: TotDeductions]

    Name            : @@DeductionGrpName
    PayHeadName     : $Name
    SortPosition    : 5004

[Collection: Pay Head Ledgers]

    Type            : Ledger
    Filter          : PayHeadLedgers

;; System Formulae

[System: Formula]

    PayHeadEmpCat       : $$IsChildOfCategory:"Employee"
    ForThisEmpType      : $Name = $$ReqObject:$Name AND $LedgerName = ##PayHeadName

;; new function to check BelongsTo

    ForThisChildCC      : +
$$IsObjectBelongsTo:CostCenter:$Name:@@ReqObjName AND $LedgerName = ##PayHeadName

    ForThisCompType     : +
$$IsObjectBelongsTo:CostCenter:$Name:@@ReqObjName AND $ComponentType=##PayHeadName

    PayHeadLedgers      : $Parent = @@EarningGrpName OR $Parent = @@DeductionGrpName
    EarningGrpName      : "Total Earnings"
    DeductionGrpName    : "Total Deductions"
    loc LedCCSearch     : $Name + ##PayHeadName
    ParentIndexPosition : If $$IsTDLObject then $SortPosition else $SortPosition:Group:$Parent

;; End-of-code
```

Back

Modifier - Switch Case

```
/*

The following Report demonstrates the usage of Switch Case

*/

[#Menu: Display Menu]

    Add     : Key Item      : Understanding Switch Case : U : Display   : Sample Switch

[Report: Sample Switch]

   Form     : Sample Switch
   ;; the following variables have been added to switch over between the options/conditions

   Variable : SampleSwitch1, SampleSwitch2, SampleSwitch3, SampleSwitch4
```

```
   Local       : Variable          : Default           : Default           : No

[Form: Sample Switch]

   Parts       : Sample Switch
   ;; the below buttons have been added to switch over between the options/conditions
   Button      : SampleSwitch1, SampleSwitch2, SampleSwitch3, SampleSwitch4

[Part: Sample Switch]

    Lines      : Sample Switch

[Line: Sample Switch]

    Fields     : Sample Switch

;; Field definitions
[Field: Sample Switch]

    Set as     : "Default Value"

;; in 'case 1' the first satisfying switch will (only one) be activated at a given time
   Switch      : Case1             : Sample Switch1          : ##SampleSwitch1
   Switch      : Case1             : Sample Switch2          : ##SampleSwitch2
   Switch      : Case1             : Sample Switch3          : ##SampleSwitch3
;; the below will be activated if the condition is true
;; by specifying different label we can achieve the same functionality of 'Option'

   Switch      : Case2             : Sample Switch4          : ##SampleSwitch4

   [!Field: Sample Switch1]

      Set as   : "Switch 1 is activated"

   [!Field: Sample Switch2]

      Set as   : "Switch 2 is activated"

   [!Field: Sample Switch3]

      Set as   : "Switch 3 is activated"

   [!Field: Sample Switch4]

      Style    : Large Bold

;; Variable Definitions
[Variable: SampleSwitch1]

     Type      : Logical

[Variable: SampleSwitch2]

     Type      : Logical

[Variable: SampleSwitch3]

     Type      : Logical

[Variable: SampleSwitch4]

     Type      : Logical

;; Button Definitions
[Button: SampleSwitch1]
```

```
   Title       : "Switch 1"
   Key         : Alt + 1
   Action      : Set     : SampleSwitch1          : NOT ##SampleSwitch1

[Button: SampleSwitch2]

   Title       : "Switch 2"
   Key         : Alt + 2
   Action      : Set     : SampleSwitch2          : NOT ##SampleSwitch2

[Button: SampleSwitch3]

   Title       : "Switch 3"
   Key         : Alt + 3
   Action      : Set     : SampleSwitch3          : NOT ##SampleSwitch3

[Button: SampleSwitch4]

   Title       : "Make Bold"
   Key         : Alt + B
   Action      : Set     : SampleSwitch4          : NOT ##SampleSwitch4

;; End-of-code
```

Back

Modifier - SetbyCondition

```
/*
Set by condition is Multiple Name / Value attribute with the following syntax:
Set by Condition : Condition Formula : Value Formula
The very first satisfied condition is evaluated; if no condition is available / evaluated and Set as is available, it is
considered
*/

[#Menu: Display Menu]

    Add       : Key Item        : Understanding SetbyCondition : U : Display          : Sample SetbyCondition

[Report: Sample SetbyCondition]

   Form       : Sample SetbyCondition
   ;; the below variables have been added to switch over between the options/conditions
   Variable   : SampleSetbyCondition1, SampleSetbyCondition2
   Local      : Variable  : Default : Default : No

[Form: Sample SetbyCondition]

   Part       : Sample SetbyCondition
   ;; the below buttons have been added to switch over between the options/conditions
   Button     : SampleSetbyCondition1, SampleSetbyCondition2

[Part: Sample SetbyCondition]

   Line       : Sample SetbyCondition

   [Line: Sample SetbyCondition]

      Field   : Sample SetbyCondition

;; field definitions
[Field: Sample SetbyCondition]
```

```
   Set as              : "Default Value"
   ;; by writing the code using 'Set by Condition' we can avoid defining unwanted
   ;; optional fields and can avoid writing complex 'if else' conditions
   SetbyCondition      : ##SampleSetbyCondition1   : "Set by Condition 1"
   SetbyCondition      : ##SampleSetbyCondition2   : "Set by Condition 2"
   SetbyCondition      : ##SampleSetbyCondition1   : "Set by Condition 1 Again"
   Width               : 0

;; Variable definitions
[Variable: SampleSetbyCondition1]

   Type       : Logical

[Variable: SampleSetbyCondition2]

   Type       : Logical

;; Button definitions
[Button: SampleSetbyCondition1]

   Key        : Alt + 1
   Title      : If NOT ##SampleSetbyCondition1 then "Setby1-Active" else "Setby1-Deactive"
   Action     : Set   : SampleSetbyCondition1 : NOT ##SampleSetbyCondition1

[Button: SampleSetbyCondition2]

   Key        : Alt + 2
   Title      : If NOT ##SampleSetbyCondition2 then "Setby2-Active" else "Setby2-Deactive"
   Action     : Set   : SampleSetbyCondition2 : NOT ##SampleSetbyCondition2

;; End-of-code
```

Back

New Method Syntax

```
[#Menu: Gateway of Tally]

   Add        : Item    : Before  : Quit  : New Method Syntax   : Display   : Understanding Functions

[Report: Understanding Functions]

   Form       : Understanding Functions
   Title      : "New Method Syntax"
   Set        : ExplodeFlag      : Yes

[Form: Understanding Functions]

   Use        : DSP Template
   Parts      : Understanding Functions
   Height     : 100% Page
   Width      : 100% Page


[Part: Understanding Functions]

   Lines           : Object Title1, Object Details
   Repeat          : Object Details    : My Ledgers
   Scroll          : Vertical
   CommonBorder: Yes

      [Line: Object Title1]
```

```
    Fields       : OD Fld1, OD Fld2, OD Fld3, OD Fld4, OD Fld5
    Local: Field: Default    : Align     : Center
    Local: Field: OD Fld1    : Set as    : "Particulars"
    Local: Field: OD Fld2    : Set as    : "Dot"
    Local: Field: OD Fld3    : Set as    : "Double Dots"
    Local: Field: OD Fld4    : Set as    : "Triple Dots"
    Local: Field: OD Fld5    : Set as    : "Quadruple Dots"
    Border     : Thin Top Bottom

[Line: Object Details]

    Fields       : OD Fld1, OD Fld2, OD Fld3, OD Fld4, OD Fld5
    Explode    : VoucherDetails       : ##ExplodeFlag
    Border     : Double Top: $$Line > 1
    Local:Field : OD Fld1   : Display   : Ledger Vouchers
    Local:Field : OD Fld1   : Variable  : LedgerName

    [Field: OD Fld1]

        Use      : Name Field
        Set as   : $Name
        Width    : @@MyWidth% Screen

    [Field: OD Fld2]

        Use      : Name Field
        Set as   : $.Name
        Border   : Thin Left
        Width    : @@MyWidth% Screen

    [Field: OD Fld3]

        Use      : Name Field
        Set as   : $..Name
        Border   : Thin Left
        Width    : (@@MyWidth + 5) % Screen

    [Field: OD Fld4]

        Use      : Name Field
        Set as   : $...Name
        Border   : Thin Left
        Width    : @@MyWidth% Screen

    [Field: OD Fld5]

        Use      : Name Field
        Set as   : $....Name
        Border   : Thin Left
        Width    : @@MyWidth% Screen

[Part: VoucherDetails]

    Lines    : VoucherDetails
    Repeat   : VoucherDetails    : MyLedVouchers

    [Line : VoucherDetails]

        Fields         : OD Fld1, OD Fld2, OD Fld3, OD Fld4, OD Fld5
        Local: Field   : OD Fld1     : Set as      : "Voucher : " + $VoucherTypeName + "/" +  $VoucherNumber
        Local: Field   : OD Fld1     : Alter       : Voucher
        Local: Field   : OD Fld1     : Space Left: 1
        Local: Field   : Default     : Style       : Small
        Explode        : MyInvEntries             : ##ExplodeFlag
        Option         : AlterOnEnter
```

```
    [Part: MyInvEntries]

       Lines      : MyInvEntries
       Repeat     : MyInvEntries         : InventoryEntries
       Local      : Field: Default       : Color     : Blue

       [Line: MyInvEntries]

                  Fields      : OD Fld1, OD Fld2, OD Fld3, OD Fld4, OD Fld5
                  Local: Field: OD Fld1   : Set as    : $StockItemName
                  Local: Field: OD Fld1   : Delete    : Display
                  Local: Field: OD Fld1   : Space Left: 2
                  Local: Field: Default   : Style       : Small Italic
                  Explode     : MyBatchEntries       : ##ExplodeFlag

    [Part : MyBatchEntries]

       Lines   : MyBatchEntries
       Repeat  : MyBatchEntries    : Batch Allocations
       Scroll  : Vertical
       Local   : Field: Default    : Color     : Deep Grey

       [Line: MyBatchEntries]

        Fields      : OD Fld1, OD Fld2, OD Fld3, OD Fld4, OD Fld5
        Local: Field: OD Fld1   : Set as    : $BatchName
        Local: Field: OD Fld1   : Delete    : Display
        Local: Field: OD Fld1   : Space Left: 3
        Local: Field: Default   : Style     : Tiny Bold

;; Object Modifications

[#Object: Voucher]

    Name       : $VoucherNumber

[#Object: Inventory Entry]

    Name       : $StockItemName

[#Object: Batch Allocations]

    Name       : $BatchName

;; Collection Definition

[Collection: My Ledgers]

    Type       : Ledger
    Childof    : $$GroupSundryDebtors
    BelongsTo  : Yes

[Collection: MyLedVouchers]

    Type       : Vouchers  : Ledger
    Childof    : $Name

[System: Formula]

    MyWidth    : 20
```

Back

ToolTip Attribute introduced at Field Definition

```
/*

The following Report demonstrates the usage of the Field Attribute, Tooltip with the help of code for Ledger
Creation Program.

*/

[#Menu: Gateway of Tally]

   Add        : Key Item  : Before        : @@locQuit       : Ledger Creation : L : Create : Led Report

[Report: Led Report]

   Form       : Led Report
   Object     : Ledger

[Form: Led Report]

   Parts      : Led ReportName, Led ReportOthers

[Part: Led ReportName]

   Lines      : Led Note, Led ReportName
   Repeat     : Led ReportName  : Name
   Break      : $$Line = 4 OR $$IsEmpty:$Name
   Height     : 31% Page

   [Line: Led Note]

      Fields   : Simple Field
      Local    : Field: Simple Field        : Set as  : "Please hover over the Fields to have a brief information"
      Local    : Field: Simple Field        : Style   : Normal Italic
      Local    : Field: Simple Field        : Align   : Center
      Local    : Field: Simple Field        : Color   : Red
      Local    : Field: Simple Field        : Skip    : Yes
       Space Bottom: 1

   [Line: Led ReportName]

      Fields   : Medium Prompt, Led ReportName
      Local    : Field: Medium Prompt     : Set as   : "Name :"
      Local    : Field: Medium Prompt     : Inactive: $$Line > 1
      Local    : Field: Medium Prompt     : Tooltip  : "Prompt for entering Ledger Names"

      [Field: Led ReportName]

         Use            : Name Field
         Storage        : Name
         Unique         : Yes
         Tooltip        : @@LedNameTTP

[Part: Led ReportOthers]

   Lines      : Led ReportGroup, Led ReportOpBal

   [Line: Led ReportGroup]

      Fields   : Medium Prompt, Led ReportGroup
      Local    : Field: Medium Prompt     : Set as          : "Parent:"
      Local    : Field: Medium Prompt     : Tooltip         : "Prompt for entering Group"

   [Field: Led ReportGroup]
```

```
        Use             : Name Field
        Table           : Group          ;; Default Collection of Groups
        Show Table      : Always

        Set Always      : Yes
        Tooltip         : @@LedGroupTTP

    [Line: Led ReportOpBal]

      Fields    : Medium Prompt, Led ReportOpBal
      Local     : Field: Medium Prompt     : Set as          : "Opening Balance :"
      Local     : Field: Medium Prompt     : Tooltip         : "Prompt for entering Opening Balance"

      [Field: Led ReportOpBal]

        Use             : Amount Forex Field
        Storage         : OpeningBalance
        Tooltip         : @@LedOpeningTTP

[System: Formula]

    LedNameTTP      : "Please give suitable ledger names. Maximum of 4 Names are allowed."
    LedGroupTTP     : "Please select an appropriate Group"
    LedOpeningTTP   : "Please enter Opening Balance as on " + $$String:@@LedOpeningDate + ", if any"
    LedOpeningDate  : $BooksFrom:Company:##SVCurrentCompany

;; End-of-code
```

Back

ReInitialize Operator *

```
/*
This Report demonstrates the usage of ReInitialize Operator "*"
*/

[#Menu: Display Menu]

     Add       : Key Item         : Understanding ReInitialize : U : Display     : Sample ReInitialize

[Report: Sample ReInitialize]

   Form        : Sample ReInitialize

[Form: Sample ReInitialize]
   Parts       : Sample ReInitialize

[Part: Sample ReInitialize]

   Lines       : Sample ReInitialize

   [Line: Sample ReInitialize]

        Fields     : Sample ReInitialize

        [Field: Sample ReInitialize]
             Info         : "Original Value"
             Style        : Large Bold
             Color        : Blue

;; to check the functionality comment the below definition
;; when ever we want to delete all the attributes that are available in the previous
;; definition, we can reinitialize the definition by adding * and redefine
;; if the definition is used in some reports and locally modified, the same will not be deleted
             [*Field: Sample ReInitialize]

                 Info      : "Re Initialized - All the attribute values are deleted and newly defined"
                 Border  : Thick Box

;; End-of-code
```

Back

XML HTTP Collection

```
/*
The following TDL Code is written to get the customer data from XML File and display it in a Report.
To execute this Report, the following XML Data should be available in the form of a file in the path specified with the
URL.

;; XML Data

<CUSTOMER>
    <Name>Sample Name</Name>
    <ADDRESS>
     <LINE1>Line1</LINE1>
      <LINE2>Line2</LINE2>
```

```
      <LINE3>Line3</LINE3>
    </ADDRESS>
    <ADDRESS>
     <LINE1>Line1</LINE1>
      <LINE2>Line2</LINE2>
      <LINE3>Line3</LINE3>
    </ADDRESS>
 </CUSTOMER>

*/
```

```
[#Menu: Gateway of Tally]

    Add      : Key Item : HTTP Collection : H : Display    : HTTP Collection

[Report: HTTP Collection]

    Form     : HTTP Collection

[Form: HTTP Collection]

    Height   : 100% Screen
    Width    : 100% Screen
    Part     : HTTP Collection

[Part: HTTP Collection]

    Line     : HTTP Collection
    Repeat   : HTTP Collection          : TestHTTPDetails
    Scroll   : Vertical
```

```
    [Line: HTTP Collection]

            Field     :  MyName
            Explode   :  MyAddress

            [Field: MyName]

                    Use      : Name Field
                    Set As   : $NAME

;; Explode Part for Address Details begin here

            [Part: MyAddress]

                 Line         : MyAddress
                 Repeat       : MyAddress       : ADDRESS
                 Scroll       : Vertical

                 [Line: MyAddress]

                       Field : MyLine1, MyLine2, MyLine3

                       [Field: MyLine1]

                            Use      : Name Field
                            Set As   : $LINE1

                      [Field: MyLine2]

                            Use      : Name Field
                            Set As   : $LINE2

                      [Field: MyLine3]
```

```
                        Use       : Name Field
                        Set As    : $LINE3

[Collection: TestHTTPDetails]

    RemoteURL       : "http://localhost/testxml_snippet3.xml"
    XMLObjectPath   : CUSTOMER
```

Back

XML HTTP Collection with Object specification

```
/*
The following TDL Code is written to import the customer data from XML File and store it in a user created Object.
To execute this Report, the following XML Data should be available in the form of a file in the path specified with the
URL.

;; XML Data

<CUSTOMER>
   <Name>Sample Name</Name>
   <ADDRESS>
    <LINE1>Line1</LINE1>
     <LINE2>Line2</LINE2>
     <LINE3>Line3</LINE3>
   </ADDRESS>
   <ADDRESS>
    <LINE1>Line1</LINE1>
     <LINE2>Line2</LINE2>
     <LINE3>Line3</LINE3>
   </ADDRESS>
 </CUSTOMER>

*/
```

```
[#Menu: Gateway of Tally]

    Add     : Key Item : HTTP Collection : H : Display    : HTTP Collection

[Report: HTTP Collection]

    Form    : HTTP Collection

[Form: HTTP Collection]

    Height  : 100% Screen
    Width   : 100% Screen
    Part    : HTTP Collection

[Part: HTTP Collection]

    Line    : HTTP Collection
    Repeat  : HTTP Collection : TestHTTPDetails
    Scroll  : Vertical

    [Line: Http Collection]

            Field     : MyName
            Explode : MyAddress
```

```
            [Field: MyName]

                    Use      : Name Field
                    Set As   : $NAME

;; Explode Part for Address details begins here

            [Part : MyAddress]

                    Line     : MyAddress
                    Repeat   : MyAddress      : ADDRESS
                    Scroll   : Vertical

                    [Line: MyAddress]

                         Field        : MyLine1, MyLine2, MyLine3

                         [Field : MyLine1]

                              Use            : Name Field
                              Set As         : $LINE1

                         [Field : MyLine2]

                              Use            : Name Field
                              Set As         : $LINE2

                         [Field : MyLine3]

                              Use            : Name Field
                              Set As         : $LINE3

[Collection: TestHttpDetails]

    RemoteURL       : "http://localhost/testxml_snippet3.xml"
    XMLObjectPath   : CUSTOMER
    XMLObject       : Customer


[Object: Customer]

    Storage         :  Name         : String
    Collection      :  Address      : MyAddressCollection

[Object: MyAddressCollection]

    Storage         : Line1         : String
    Storage         : Line2         : String
    Storage         : Line3         : String
```

Back

Actions and Scope

```
/*
The following Report demonstrates the usage of various actions and scope
*/

[#Menu: Gateway of Tally]

    Add      : Item   : MultiSample : Display    : MultiSample

[Report: MultiSample]
```

```
    Use      : Day book
    Delete   : Form
    Add      : Form   : MultiSample

[Form: MultiSample]

    Use      : Day Book
    Button   : RemoveSelected, RemoveUnselected, RemoveAll
    Button   : DeleteSelected, DeleteUnselected, DeleteAll
    Button   : Btn Select All, Btn Select None, Btn Invert Selection
    Key      : Shift Down Select

[Button: RemoveSelected]

    Key      : Ctrl + 1
    Action   : Remove Line
    Mode     : Display
    Scope    : Selected Lines

[Button: RemoveUnselected]

    Key      : Ctrl + 2
    Action   : Remove Line
    Mode     : Display
    Scope    : Unselected Lines

[Button: RemoveAll]

    Key      : Ctrl + 3
    Action   : Remove Line
    Mode     : Display
    Scope    : All Lines

[Button: DeleteSelected]

    Key      : Ctrl + 4
    Action   : Delete Object
    Mode     : Display
    Scope    : Selected Lines

[Button: DeleteUnselected]

    Key      : Ctrl + 5
    Action   : Delete Object
    Mode     : Display
    Scope    : Unselected Lines

[Button: DeleteAll]

    Key      : Ctrl + 6
    Action   : Delete Object
    Mode     : Display
    Scope    : All Lines

[Button: Btn Select All]

    Key      : Ctrl+7
    Action   : Select All
    Mode     : Display

[Button: Btn Select None]

    Key      : Ctrl + 8
    Action   : Unselect All
    Mode     : Display
```

```
[Button: Btn Invert Selection]

    Key      : Ctrl + 9
    Action   : Invert Selection
    Mode     : Display

[Key: Shift Down Select]

    Key      : Shift + Down
    Action   : Toggle Select
    Mode     : Display

[#System: Formula]

    SV_SEL_CURSOR_BG      : Red
    SV_SEL_CURSOR_FG      : Released Orange

    SV_SELECTED_BG        : Black
    SV_SELECTED_FG        : Red

;; End-of-code
```

Back

Enhanced Actions

```
/* The Following Report prints Payslips for the Selected Employees.  This is achieved by defining the scope within the Key/Button and Action used is Print Report and Report Name as its parameter which means that on clicking this Button, do not print the current report but use the current selection and print a different report.  This is achieved by using an Internal collection Parameter Collection which passes the user's choice to the report which is being initiated by the button. */

[Button: Print Selected Payslips]

    Key              : Alt + F11
    Title            : "Print Payslips"
    Action           : Print Report      : Multi Payslip Print
/* Actions, Print/Mail/Export/Upload Report have been enhanced to accept Report Name as its parameter.  This clearly indicates that these actions can act upon another report using the information passed on from the previous report */

    Scope            : Selected Lines
/* Scope decides the scope in which the key will work i.e., Selected Lines which means that this Action will be performed only on the selected lines in the Report. */

[#Report: Multi Payslip Print]

    Collection       : Parameter Collection
/* Since the same report needs to be repeated for Multiple Objects, a Collection of selected objects is required to be passed as a parameter to the called report.  This is performed by an Internal collection Parameter Collection which is a collection of all the required objects which were selected from the base report. Report Multi Payslip Print then repeats for all the objects within the collection Parameter Collection which is how concept of Multi Account Printing works. */

[#Form: List of Accounts]

    Add              : Buttons        : Print Selected Payslips
;; Attaching the Button to the required/ relevant Form
```

```
    Option              : Multi Emp PS    : ##AccountType = $$SysName:Employees
    Option              : Multi Emp NPS   : ##AccountType NOT EQUALS $$SysName:Employees
/* These Options check whether the user is viewing List of Employees or some other accounts.  If list of employees
is being viewed, then we set the Start Date and End Date of the Current Month. */

    [!Form: Multi Emp PS]

            Set       : SV FromDate     : $$MonthStart:##SVCurrentDate
            Set       : SV ToDate       : $$MonthEnd:##SVCurrentDate

    [!Form: Multi Emp NPS]

            Set       : SVFromDate      : $BooksFrom:Company:##SVCurrentCompany
            Set       : SVToDate        : ""

;; End-of-Code
```

Back

Changes in behavior of Set As and Info

```
/*
The following Report demonstrates the changes in the Field Attributes, Set as / Info
*/

[#Menu: Display Menu]

    Add       : Key Item          : Understanding Setasinfo : U : Display       : Sample Setasinfo

[Report: Sample Setasinfo]

    Form      : Sample Setasinfo

[Form: Sample Setasinfo]

    Parts     : Sample Setasinfo

[Part: Sample Setasinfo]

    Lines     : Sample Setasinfo


[Line: Sample Setasinfo]

    Fields    : Sample Setasinfo

;; even though the below field is modified locally, the 'info' at the field definition level will take precedence

    Local     : Field    : Sample Setasinfo : Set as          : "Locally Modified"

;; to check the functionality comment the below 'Info' as 'Set as'

    [Field: Sample Setasinfo]

          Info     : "Field Value"

;; End-of-code
```

Back

Changes in Behavior of Use

```
/*

This TDL code demonstrates the changes done in evaluating 'Use' attribute.
'Use' attributes are always evaluated *FIRST* irrespective of their order across other attributes
If there are multiple 'Use' they are evaluated in the order specified by the user

*/

[#Menu: Display Menu]

    Add       : Key Item         : Understanding Use : U : Display   : Attr Use

[Report: Attr Use]

    Form      : Attr Use

[Form: Attr Use]

    Parts     : Attr Use

[Part: Attr Use]

    Lines     : Attr Use1, Attr Use2

    [Line: Attr Use1]

            Fields     : Attr Use1

            [Field: Attr Use1]

                Set as     : "This demonstrates the changed behavior of 'Use' attribute"
                Style      : Large Bold
                Full Width: Yes
                Align      : Centre
                Use        : Name Field

;; Use though specified after rest of the attributes will be evaluated first.  Hence, the Style specified in the current
;; Field Attr Use1 will be considered and the one specified in the Name Field will be overridden

     [Line: Attr Use2]

            Fields     : Attr Use2

            [Field: Attr Use2]

              Set as       : "(The above field is using name field and 'Use' attribute is used at last in sequence. " + +
                             "But still it will be evaluated first as per the changed behavior of 'Use' attribute." + +
                             "Because of this, the style 'Large Bold' which is there in 'Attr Use1' field will not" + +
                             "get overwritten by the 'Normal Bold' which is there in name field.) "
              Lines        : 0
              Align        : Justified
              Style        : Small Italic
              Full Width : Yes

;; End-of-code
```

Back

Changes in Delayed Attributes – Add/Delete/Replace

```
/*
The following program demonstrates the behavioral uniformity of Delayed Attributes across definitions.
*/

[#Menu: Display Menu]

    Add      : Key Item         : Understanding Delayed Attributes : U : Display      : Delayed Attr

[Report: Delayed Attr]

    Form     : Delayed Attr1
    Delete   : Form

    ;; If we use the following line without using Add Attribute to add a new child, the end result will be,
    ;; the report will have NO form as against the previous behavior.

    ;; Since the behavior is changed to bring the consistency across the definitions,
    ;; now we need to use 'Add' attribute to add any new child if we have previously deleted
    Add      : Form    : Delayed Attr2

[Form: Delayed Attr2]

    Parts    : Delayed Attr
    Option   : Small Size Form

[Part: Delayed Attr]

   Line    : Delayed Attr

 [Line: Delayed Attr]

      Field   : Delayed Attr

      [Field: Delayed Attr]

            Use        : Name Field
            Set as     : "This report is visible because 'Add' attribute is used to add a new form after deleting " ++
                         "the previous one.  If you don't use the 'Add' attribute at the report to add a new " ++
                         "form, then Tally throws an exception saying that 'Form: DelayedAttr' could not be " ++
                         "found. This is because when you don't use 'Add' attribute, Tally doesn't add a new " ++
                         "form since 'Delete:Form' was executed last and deleted all the forms. If no forms " ++
                         "are available, then Tally searches for a form with the same name as Report. To " ++
                         "check this, you can remove the 'Add' attribute at the report level and see."

            Lines      : 0
            Align      : Justified
            Full Width: Yes

;; End-of-code
```

Back

Changes in Menu Item value - BLANK Keyword

```
/*

Sample code to explain the behaviour of BLANK Keyword
```

```
*/

;; old behavior

[#Menu: Gateway of Tally]
    Add      : Item    : New Item1
    Add      : Item :                          ;; BLANK Keyword was not mandatory
    Add      : Item : New Item2

;; new behavior

[#Menu: Gateway of Tally]
    Add      : Item    : New Item1
    Add      : Item : BLANK                    ;; BLANK Keyword is mandatory for consistency
    Add      : Item : New Item2

;; End-of-code
```

Back

III Tally Functional Enhancements

1. Tally.NET

Tally 9 Release 3.0 brings a collaborative technology, which provides a wide range of Internet based services.

Tally.NET Services

Tally.NET provides a host of capabilities. Following are list of capabilities available in the current release:

1. Register and Connect companies from Tally
2. Create and maintain Remote Users
3. Remote availability of Auditors' Edition of Tally License

2. Auditors' Edition of Tally

Tally 9 offers a special Gold Auditors' Edition of Tally, which provides auditing and compliance capabilities exclusively for Chartered Accountants. The Audit capabilities provided in Auditors' Edition are available by default, regardless of Tally.NET subscription. Using Tally.NET features, Tally provides the capability to Chartered Accountants to Audit in following scenarios :

1. Audit at CA's office by accessing local data
2. Audit at Client's place by accessing local data

The Auditors' Edition of Tally enables the auditor to compile information in comparatively less time with required accuracy to form his opinion and print annexures as required under Tax Audit under Sec. 44AB. The following clauses are available as required in annexure Form 3CD :

3. Bonus PF, ESI Recoveries (Clause 16)
4. Amounts inadmissable u/s 40A(3)
5. Payments u/s 43B (Clause 21)
    a. Employer's Contribution
    b. Tax Collected at Source
    c. Service Tax
    d. Value Added Tax
6. Loans / Deposits Accepted (Clause 24(a))
7. Loans / Deposits Repaid (Clause 24(b))
8. Fringe Benefits Tax (Annexure II)

3. Data Synchronization

Using Direct - Client and Server Capability

Direct- Client and Server capability for Synchronization through private network will be part of the default product from Release 3 onwards.

Using Tally.NET Server

The process of Data Synchronization using TallyLink Server is changed in Tally 9 Release 3.0. Going forward Synchronization will be available though Tally.NET Framework as one of the services. To use this service the company has to be Registered and Connected to Tally.NET server at the time of Synchronization.

Note: Tally.NET Server (TNS) will be used for Synchronization instead of TRB Server

4. Payroll

Payroll in Tally 9 is now **simplified & enhanced** to handle all the functional, accounting and statutory requirements of your business. The following Statutory features and processes have been incorporated for accurate, better and faster payroll processing.

Payroll Statutory Features

Tally 9 Release 3.0, the following statutory features are provided:

Employees Provident Fund (EPF)

- Provides automated computation and deduction/contribution of Employees' & Employer's Contribution towards Provident Fund.
- Provides predefined PF process to expedite voucher entry and accurate processing of contributions
- Generates following Monthly / Annual statutory Challans & Reports :
  - **Form 12A** – Statement of PF Contribution during the particular month
  - **Form 5** – Details of the Employees joining during the particular month
  - **Form 10** – Details of the Employees leaving during the particular month
  - **Combined Challan** – Challan form to be submitted to the bank while paying the PF amount
  - **PF Monthly Statement**
  - **Form 3A** – Employee wise Annual Payment details
  - **Form 6A** –Annual Statement of PF contribution
- Facilitates automated computation and payment of following administrative charges with the help of predefined processes.
  - **PF Administration Charges @ 1.10%**

- **Employees' Deposit-Linked Insurance Scheme (EDLI) @ 0.5%**
- **EDLI Administration Charges @ 0.01%**
- **Other Admin Charges** (inspection charges, interest paid etc)

Employees State Insurance (ESI)

- Provides automated computation and deduction/contribution of Employees' & Employer's Contribution towards Employees' State Insurance
- Provides predefined ESI process to expedite voucher entry and accurate processing of contributions
- Generates following Monthly / Half Yearly statutory Challans & Reports :
  - **ESI Monthly Statement**
  - **ESI Monthly Payment Challan** - Challan form to be submitted to the bank while paying the ESI amount
  - **Form 3** – Half Yearly declaration form
  - **Form 5** – Half Yearly Return of ESI Contributions
  - **Form 6** – Half Yearly Employee Register

Professional Tax (PT)

- Provides automated computation and deduction of Professional Tax based on user-defined Slab rates
- Provides predefined PT payment process to expedite voucher entry and accurate payment of Professional Tax
- Generates following PT Reports :
  - **PT Computation Report** - Print report with Slab-wise details of Professional Tax by the Employer
  - **PT Monthly Report** – Print report with Employee wise details of Professional Tax

Payroll Processes

The following predefined payroll processes are provided to automate different functions in Payroll:

- **PF Process –** Automates the Computation of Employer PF Contribution
- **ESI Process -** Automates the Computation of Employer ESI Contribution
- **Salary Process -** Automates the Computation of Salary
- **PF Challan -** Automates the Payment of PF
- **ESI Challan -** Automates the Payment of ESI
- **Professional Tax Payment -** Automates the Payment of PT
- **Salary Payment -** Automates the Payment of Salary

5. Fringe Benefits Tax

- Fringe Benefits Tax (FBT) Annual Returns in Form ITR 8 with details of computation of Fringe Benefits & Fringe Benefits Tax and Schedule FBI is available.
- **Exemption Amount** field is provided in **FBT transactions** to specify the applicable Exemption amount.
- Bifurcations like FBT, Surcharge, Education Cess and Secondary Education Cess can be displayed in separate columns by enabling 'Show Tax Details' in F12 configuration from FBT computation.

Minor Enhancements:

Accounting Masters

In Group Creation /Alteration screen, the option No Appropriation has been removed as it had the same behaviour as of Not Applicable.

On disabling TDS / TCS, in the Ledger Alteration, Tally does allow the selection of State manually.

The field Rounding Method is provided in Ledger masters grouped under Duties & Taxes, Direct Expenses, Indirect Expenses, Direct Income, Indirect Income, Current Asset, Current Liability, Sales Accounts, Purchase Accounts and Primary Group in Purchase/Sales Invoice mode only.

Accounting Reports

F8: Other Report option is disabled in Statistics and Group Payment Performance under Group Summary.

Accounting Vouchers

In sales invoice, the Company's/Buyer's Tax Registration numbers will be printed in the order given below:

1. Company's VAT TIN
2. Company's CST No.
3. Company's Service Tax No.
4. Buyer's VAT TIN/Sales Tax No.
5. Buyer's Service Tax No.

The Option 'Show Ledger Current Balances' in F12: Voucher Configuration is now applicable for Ledgers in Invoice Format. If this field is set to NO, the Current Balance of the ledgers will not be displayed in invoices.

A Purchase order entered in a company with GST Registration number can be exported using Excel (Spreadsheet) or HTML format.

Application supports the printing of Inventory Details for Debit Note, Credit Note, Payment, Receipt and Journal entered in voucher mode, by enabling the option Print Inventory Details in F12 Print Configuration.

A Facility to print the serial number for each item in the Sales Invoice has been provided.

An option has been provided to print the system date and time in all the reports including the Invoice.

Cost Categories/Cost Centers

Cost Centers will appear along with Cost Centre Class in all Vouchers.

CST

In CST Forms Issuable / Receivable report, F12: Configure option is provided to sort details of Party in alphabetical order.

Emailing

SSL Interfaces are provided to enable the user to E-mail any information from Tally to external Mail servers.

Excise for Dealers

In Excise Sales Invoice Supplier Details for Stock item displays Rate of Basic Excise Duty and Excise Duty Per Unit along with supplier details and quantity purchased in the List of Purchases table.

A separate column has been provided to display the Special Additional Duty in printed format of Excise Purchase Invoice.

In Gujarat Excise Sales invoice, a separate column has been added to displaying the HSN Code.

In an Excise Purchase Invoice, an option is provided to create a ledger under Manufacturer/Importer Details section in Supplementary Details screen.

Configuration Use Excise Format in F12: Configuration is displayed only on enabling the option Allow Excise rules for Invoicing in F11: Features.

Final Accounts

If the option Income/Expense Statement instead of P & L is enabled in F11: Accounting Features, then in Income and Expenses A/c report, F12: Configure displays Show Vertical Income & Expense Stmt. is displayed instead of Show Vertical Profit & Loss.

In Statistics report, a facility has been provided to sort the voucher types using the configuration Sort by Parent Voucher Type.-OK.

Installation

The language support for Bahasa Indonesia and Bahasa Melayu has been provided on all operating systems.

Inventory Masters

In the Stock Group and Stock Item creation and alteration screen, a minimum width of 30 characters is provided to ensure visibility of names entered.

Inventory Vouchers

The option “Show using Alternate Units” is provided in F12: configure of Stock Vouchers screen to display the alternate units details for stock items.

Licensing

A Confirmation message is displayed when user opts to surrender the licensed version of Tally.

In the process of activating license, an hourglass will be displayed as an indicator to show the progress of activation

Multi-currency

Bank Reconciliation displays the accurate Closing Balance of Transacted Currency as per Company Books and Actual Balance, when "Show Forex Transactions only" is activated in F12: Configure.

Payroll & Taxes

In the PayHead master, an option has been provided to round off the Gratuity amount.

The Option 'Show Zero Value Payslips Also' is provided in Multi Payslip Printing to print the Blank Payslips.

In Paysheet, the Grand Total of attendance is displayed and user can drill down to attendance voucher from attendance details.

Position Index for Reports of the Employee is set as Default Sorting Method for all the Payroll Reports, in addition Alphabetical (increasing) and Alphabetical (decreasing) options are added to the Sorting Methods list, of Pay Sheet Configuration.

Tally now allows the user to group the TDS Payhead under Duties & Taxes and Salary Payable Payhead under Provisions

The option Show new joinees only has been renamed as Show zero valued Employees also In F12 Configuration of Gratuity Report.

In Pay sheet report the option "Display Attendance/Production Types in Tail Units ?" is provided under F12: Configure.

New Payhead Type Bonus is provided in the Payhead masters creation.

In International Version of Tally Gratuity Report Option is provided in Payroll Reports Menu.

POS

Now Party Details can be Entered and Printed in the POS Invoice.

Printing

The Option "Print Narration for Each Entry" is provided in F12 : Invoice printing Configuration to print narration for each entry in Purchase or Sales voucher.

The Option "Print Narration" is provided in F12: Configure (Physical Stock Printing Setup) to print Narration in Physical Stock Voucher. The same facility is provided to print Stock Journal Voucher.

Security Control

A warning message Forgetting your password will render your data unusable!! will be displayed when the user is setting or changing the Admin User Name and/or Password in Company Creation/alteration screen.

Service Tax

In Sales Register columnar report, a separate column has been provided for displaying the Service Tax Registration number.

In Service Tax Payables report, F4: Category button has been added to filter the services provided, based on the service category.

According to the new notification the “Focal Bank Details” has been withdrawn from Company F11: Statutory and Taxation under Company Service Tax Details.

Options provided to display/alter the Exemption Notification Number in item invoice.

In service tax payment voucher, cheques/Draft/Pay order No and Bank Name fields are provided under ‘Payment Details’ sub screen in ‘Provide details’ field.

Option to enter Challan No. and Challan Data is provided under Payment Details in Service Tax GAR 7 Payment Voucher, which will get printed in ST3 Form.

Cost centre allocation has been provided for Service Tax ledger created under Duties and Taxes.

VAT

In the voucher class screen of sales voucher type, the Calculation Methods Additional Tax and Cess on VAT can be selected only if the VAT Classification is selected in the respective sales ledger.

When a Company opts from Composition to Regular VAT and tries to pass purchase entry during composition period, in Purchase Accounting Allocation Screen VAT/TAX Class displays Composition VAT Classifications for selection.

When a MRP Stock Items Invoice is printed with VAT Analysis, total MRP Assessable Value is displayed as Net Value under VAT analysis

Issue Resolved:

Accounting Masters

Opening Balance bill Details in the ledger master were neither updating nor erasing when the Opening balance was altered in the ledger.

This issue has been resolved.

Accounting Reports

In Stock Item Monthly Summary on selecting Budget variance and then on selecting any particular month Memory Access violation is being displayed.

This issue has been resolved.

In Purchase/Sales Voucher Register report, where the sort option is selected as Amount-wise(Decreasing) and Amount-wise (Increasing), is not displaying the amount as per the method selected.

This issue has been resolved.

In case of journal entry consisting of a sundry creditor ledger, the Day Book Extract report of Journal voucher was not displaying Sundry Creditors under Current Liabilities.

This issue has been resolved.

The Post Dated vouchers report was displaying regular vouchers along with post dated vouchers when the report period was changed.

This issue has been resolved.

Date range was not refreshing when user was viewing Outstanding Report from the Ledger Report.

This Issue has been resolved.

A credit note entry made in voucher mode, was displaying the invoice format without Party name when printed from Multi Voucher Printing option. The same entry when made in invoice mode, was displaying the invoice format with party name when printed from Multi Voucher Printing.

This issue has been resolved. The credit note entry made both in voucher and invoice mode, when printed from Multi Voucher Printing option is displaying the Party name.

In case of multi year data, date range provided for previous year in ledger outstanding report from respective Ledger account is not showing as per the date range specification.

This issue has been resolved.

Press Enter on the exploded line of Group Vouchers; it would display Forex Gain / Loss Statement as the report heading.

This issue is resolved and it is not allowing to drill down.

In the Outstanding Ledger Report when Show Bills of following type is set to Debit Bills in F12: Configure. Both credit bills and debit bills are displayed in the report.

This issue is resolved and only Debit Bill are displayed when Show Bills of following type is set to Debit Bills in F12: Configure.

Accounting Vouchers

In sales invoice, if one VAT ledger was selected and then changed to another VAT ledger before saving the entry, the percentage and amount were not being refreshed.

This issue has been resolved and the VAT Percentage and amount are being calculated based on the re-selected VAT ledger.

A voucher class consisting of multiple excise duty ledgers with Type of Calculation based On Item Rate was calculating the amount of duty on the item for all excise ledgers included in voucher class and displaying it in the Excise Sales Invoice.

This issue has been resolved. The calculation of excise duty on the item is being displayed only for the relevant excise ledger.

On activating Use Additional Description(s) for Ledger Name in Invoice Configuration, the Description of Ledger screen was not being displayed for Accounts only companies.

This issue has been resolved. The **Description of Ledger** screen is now being displayed even for Accounts only companies.

When same Bank Ledger was selected for both Cheque and Card Payment in POS entry, the amount in both the fields was becoming the same when opened in alteration mode.

This issue has been resolved.

On changing the invoice date in the alteration screen of a sales invoice with voucher class, consisting of discount ledger and method of calculation set to As User Defined Value, the percentage entered used to disappear.

This issue has been resolved.

In a Sales / Purchase Invoice containing two different VAT sales/ purchase ledger, the sales/purchase figure of an item is modified either by changing rate, quantity or the sales/purchase amount directly, the sales/purchase value was not picked up automatically in the "Assessable Value" field.

This issue has been resolved.

When a single excise stock item is accounted in a sales voucher with Voucher Class which contains multiple Excise Ledgers with different Rates (Grouped under Duties & Taxes) and Type of calculation as Tax based on Item Rate, Excise amount is getting calculated for all the Excise Ledgers.

This issue has been resolved.

When a voucher recorded by selecting a Voucher class which is set to calculate the Excise on Current Sub Total for for the Excise duty Ledger and the option Calculate Tax on current Sub-Total is activated in F12:Configure, the Excise amount is calculated based on the Current Subtotal. If the same voucher is altered by deactivating the option Calculate Tax on current Sub-Total, Excise amount is calculated based on the Item rate but not on the current sub total.

This issue has been Resolved and now excise amount is calculated properly.

In the alteration mode of a POS invoice, different amounts entered in Credit/Debit Card Payment and Cheque fields by selecting the same bank ledger in the voucher creation mode, was being displayed as same amount in both the fields. Also, the entry was not being saved in alteration mode.

This issue has been resolved and the values entered in the above mentioned fields are being retained with the flexibility to save the changes made to the invoice.

While altering a Sales Invoice the VAT/TAX Class does not appear for the second and subsequent items.

This issue is resolved and the VAT/TAX class appears for any number of items added to the sales invoice in alteration mode.

Audit Features

A voucher accepted during Audit and then deleted displays particulars as (DELETED – Audit Voucher) in Tally Audit Listing. On pressing Enter to view such voucher, Ledger Forex Gain/Loss report was being displayed.

This issue has been resolved.

Budgets and Credit limits

Sales Voucher Registers with Extract of all vouchers, crashes when Budget Variance is selected.

This issue has been resolved and Budget variance button is removed from Sales Register, Purchase Register, Journal Register under Accounts Books, Bill Outstanding Reports of Sundry Debtors and Creditors, Stock Items, Stock Transfers, Physical Stock Register under Inventory Books and Memorandum Vouchers under Exception Reports, as it is not required.

Cost Categories

The Cash Book was displaying cost centre details only when printed from Multi Column Cash/Bank and not from Account Books of Multi Account Printing menu.

This issue has been resolved.

Crash/MAV

While Filtering the Stock Items with Stock Groups from Sales Register, Tally is crashing by displaying the error message “Memory Access Violation”.

This issue has been resolved.

While Printing any report if the Key combination Alt+U is pressed instead of Alt+I to select the print preview option in the Print Configuration screen, Tally is crashing by displaying the error message “Memory Access Violation”.

This issue has been resolved.

Data/TCP migration

On migrating data from Tally 7.2 to Tally 9, the order reference number entered in supplementary details of some of the sales invoices were getting changed.

This issue has been resolved and the order reference number entered in every sales invoice is now being retained even after migrating data from Tally 7.2 to Tally 9.

Excise for Dealers

When multiple Purchases pertaining to a single Item is adjusted in a Excise Sales Invoice, the values of such entries were capturing only the first item in the Excise Stock Register/ Excise Purchase Bill Register.

This issue has been resolved.

In the detailed format of Excise Stock Register and Purchase Bill Register report, the address of the party was not being displayed in the Name and Address of the Party column only for the first entry.

This Issue has been resolved and the address of the party is now being displayed for all entries in the detailed format of Excise Stock Register and Purchase Bill Register.

Excise Opening Stock Entry details does not appear in the Excise Stock Register and when the sales invoice raised against the opening stock is printed, Excise Pass on Duty Values and Excise Purchase Section Duty Values do not appear in the printed Sales invoice.

This issues have been Resolved

Export/Import

When Stock Items with Part Numbers are exported using Excel (spread sheet) Format, in the excel sheet Part Numbers are displayed with negative sign.

This issue has been resolved and the part numbers, of the stock items are displayed properly in the excel sheet.

On exporting any of the Ledger vouchers from Tally to Excel Spread Sheet the balance at the end whether it's a debit/credit balance as displayed in Tally shows opposite (Debit balance is shown as credit and the credit balance is shown as debit) in Excel.

This issue has been Resolved.

When vouchers without effective date are imported in SDF format, application is crashing by displaying the error as Memory Access Violation.

This issue has been resolved and vouchers can be imported even when the effective date is not mentioned in the voucher.

When Ledgers are exported in ASCII format from Multi Account Printing as Group of Accounts by enabling the option Start fresh page for each Account in print configuration screen, the exported text file was displaying the group ledger which was debited even in the Credit field.

This issue has been resolved.

When the Bank Reconciliation Statement is exported using Excel Format, the details were not being displayed in appropriate columns and the Balance as per Bank and Company Books were being displayed twice.

This issue has been resolved.

Compound Units were not getting exported to Excel.

This issue has been resolved and compound units can be exported to excel as text and not as number.

Final Accounts

The column heading separator lines did not appear when a new column is inserted in the Profit and Loss account.

This issue is resolved and the **column heading separator lines** appear in the Profit and Loss Account.

Fringe Benefit Tax

In FBT Filters, on entering only the month, the first day of the month was being displayed as date in both From Date and To Date fields.

This issue has now been resolved.

The Additional Education Cess @ 1% which is applicable only from April 2008, was being calculated and displayed in the FBT Computation report generated for the period 2006-07.

This issue has been resolved and now, the Additional Education Cess @ 1% is not considered in the **FBT Computation** report of previous year.

The FBT Computation report, when generated for a financial year was getting hung if the volume of data is High.

This issue has been resolved and the **FBT Computation** report is now displaying the details like any other report regardless of data size.

In FBT Computation Report, Tax was being calculated even on the Credit Value of Expenditure and as a result the Grand Total of Net expenditure amount was displayed with negative sign.

This issue has been resolved.

FBT Ledger created in FBT Filters sub screen is not displayed in the List of FBT Ledgers for selection.

This issue has been resolved.

The FBT Payment entry marked as optional was affecting the FBT Computation report as it was being considered as regular payment entry.

This issue has now been resolved.

The value calculated on using FBT Filters in Payment voucher was more than the amount displayed in FBT Computation report. On restoring the data backup of company enabled for FBT, the Company FBT Assessee Details were missing in both Statutory & Taxation Features and FBT Challan. On activating the FBT Feature, the amount displayed in FBT Challan was more than that of FBT Computation report.

This issue has now been resolved. The value displayed in FBT Computation report is captured in FBT Payment voucher on using the auto fill option and

the Company FBT Assessee Details are also retained even after restoring the data.

FBT Computation amount is not matching with the FBT payment amount during FBT Payment Entry. It will show difference of Rs.1/-. This happens due to round off.

This issue has been resolved.

FBT Computation Report captures and prints the period including the post dated voucher date, when the post dated entry are passed using the ledgers for which FBT option is not enabled.

This Issue has been resolved.

Inventory Masters

To delete the components/items used in Bill of Material (BOM) item, wherein after deleting the Bill of Material (BOM) item, Tally has to be reloaded or the item had to be renamed.

This issue has been resolved, and the Components/items can now be deleted without reloading the Tally or renaming the item.

Inventory Reports

In Stock Ageing Analysis report Company Name and Address was not getting printed from the second page onwards.

This issue has been resolved

Inventory Vouchers

The voucher number of Sales Order was getting reflected in Delivery Note. Hence, the voucher number subsequent to that of sales order was being displayed as the reference number in delivery note.

This issue has been resolved.

On printing a sales invoice in simple format, even though there were tracking details for more than one Delivery Note, only one delivery note number was being captured and printed.

This issue has been resolved. Multiple delivery note numbers are now being printed in a single sales invoice.

In stock item allocation screen of sales order, when Any was selected under godown for a particular stock item, the warning message Negative Stock! Was displayed even though there was closing balance of that item in other godowns.

This issue has been resolved. On selecting **Any** as godown, warning message is not displayed if there is closing balance of stock items in other godowns.

The reference number entered in sales invoice was not getting captured in Bill-wise details screen if the method of voucher numbering was set to None in sales voucher type alteration screen.

This issue has been Resolved.

In a Manufacturing Journal, for a Stock Item with Component List etails (BOM) when amount is entered for the Component item without defining the rate, Rate of the Component item is not calculated automatically (Left blank) based on the quantity and amount.

This issue has been resolved and Rate is calculated based on the Quantity and amount specified.

When a Stock item, allocated with the amount of Expenses on purchase is transferred from one godown to another using Stock Journal, the Rate displayed for the item in Destination Column does not include the amount of expenses on purchase.

This issue has been resolved.

When a Sales Invoice is printed in simple format Delivery Note date was not getting printed, even when the details were entered in the invoice dispatch details.

This issue has been resolved and the Delivery Note date is printed in the sales invoice.

In sales invoice if reference is provided and if "Use defaults for bill allocation" is no then reference is captured in Bill wise window. If reference is not provided then voucher number is captured.

This issue has been Resolved.

Tracking number used in ‘Rejection In’ was not displayed in Item Allocation screen of Credit note (As Voucher mode).

This issue has been resolved.

For Company in which VAT and Use different Actual & Billed Quantity features are enabled, "Calculate MRP on billed quantity" option was not getting saved in F12: Invoice / Orders Entry configuration once you quit Tally and Reopen.

This issue has been resolved.

Licensing

Multi-Account Printing menu is not getting refreshed when the license is activated or surrendered. Takes effect only when the Tally is restarted.

This issue has been resolved.

Multi-currency

In Bank Ledger Voucher report, if a voucher is altered by drilling down from the report, the Opening and Closing Balances of Ledger Voucher report are displayed in Base currency instead of Foreign currency, when 'Show Forex details also' and 'Show Forex Transactions only' are enabled in F12: Configure.

This issue has been resolved and on altering a voucher containing foreign currency, opening and closing balances appear in foreign currency only.

On reconciling a bank account which contains foreign currency transaction the Balance as per Company Books differs.

This issue is resolved and now it displays correct reconciled values.

When we view any ledger account with Show Forex Transactions option as "Yes", it displays transacted values in foreign currency but opening and closing balances in the base currency.

This is resolved and now it displays the opening and closing balance in foreign currency.

ODBC/Export/Import

The Purchase Invoice consisting details of Company's Address and E-Mail ID, when exported in HTML format was displaying the error message 'Part:EXPINV Purchaser'.

This issue has been resolved. The Purchase Invoice getting exported in HTML format along with the Company's address & Email-ID.

When Sales Register is exported in HTML or Excel (Spreadsheet) format or printed by enabling Show Columnar Register with Type of Column as All Items (in one col.) was exporting or Printing the report in multiple columns.

This issue has been resolved.

On exporting Employee masters in XML format from List of Accounts Report, all the accounts and payroll masters were also being exported along with employee masters.

This issue has been resolved.

Optional & Scenario management

In Post Dated Voucher Report, optional voucher was also being displayed.

This issue has been resolved.

Payroll & Taxes

The Payment Advice was displaying the monthly installment of loan amount recovered from employee's salary, even though it was not included in payment voucher.

This issue has been resolved

In the PayHead creation screen, if the payhead name was entered and then altered, the changed name was not being displayed in the field Name to appear in Payslip.

This issue has now been resolved.

When the positive attendance type is used in the Pay Head, Payroll was not being calculated for employees who resign before/prior to the date on which attendance voucher was recorded.
This issue has been resolved.

When Payslips are printed through Multi Payslip Printing by enabling the option Remove Zero Entries, Tally still prints blank payslips of deactivated/zero entry employees. Still the blank payslips of deactivated/zero entry employees are printed.

This Issue has been resolved and now only the payslips with the entries are printed, when the option **Remove Zero Entries** is enabled in **Multi Payslip Printing**.

When you drill down from the attendance value of the employee in Attendance sheet, Voucher Details are not displayed in the Attendance Voucher

This issue has been resolved.

When Employee Voucher Report is printed the title of the report was getting printed as Cost Center instead of Employee.

This issue has been resolved.

Print Configuration Screen of Multi Payslip Printing was not aligned properly.

This issue has been resolved.

POS

After recording a POS transaction, the Sales Invoice Printing configuration screen title, was being displayed as 'POS Invoice Printing Configuration' instead of "Invoice Printing Configuration"

This issue has been resolved.

List of Ledgers Accounts is not displayed, when the cursor is in the Credit/Debit Card payment field in a Pos Voucher with Class in alteration mode

This issue has been resolved.

Amount is not calculated properly, when the Rate of Expense is altered in a POS voucher with class in alteration Mode.

This issue has been resolved.

When Sales Ledger is changed in the POS invoice in alteration mode VAT/Tax Class is not refreshing as applicable to the changed sales ledger and also on changing the VAT Ledger the VAT Percentage is Not changing.

This issue has been resolved.

Tally crashes when a POS Invoice with Class (voucher Class is created with 'Type of Calculation' set to "As Total Amount Rounding" under Additional Accounting Entries.) is altered.

This Issue has been resolved.

Printing

The report title of a Credit Note was being displayed as Cash Voucher, when printed from Multi Voucher Printing option of Multi Account Printing menu.

This issue has been resolved and the report titles are now displaying the names of the respective vouchers selected in the Printing Vouchers screen.

In Day Book – Sales Register, if the Sorting Method was set to Alphabetical (Decreasing/Increasing), the vouchers were displayed in the order set, but the same was not being reflected in printed format.

This issue has been resolved.

When the invoice title is not defined either in the Sales Voucher Type or in the F12: Invoice Print Configuration, Sales Invoice is printed without title.

This issue has been resolved and now **"Invoice"** is set as **Default Title**, when the title is not defined in the Voucher Type or in the F12: Invoice Printing Configuration.

In Multi Account Printing, if ledger is printed as Group of Account with Debit/Credit entries only, by setting the option Start fresh page for each Account to No, a blank report was being displayed.

This issue has been resolved.

Changes made to Print Configuration settings for Debit Note were not affecting the print.

This issue has been resolved.

While printing a cheque by selecting a ledger grouped under secured/unsecured loans in the payment voucher, Ledger name does not get printed on the cheque.

This issue has been resolved.

When a credit note is printed by enabling the option 'Print Quantity Column' in Print configuration, Quantity Column was not getting printed in the credit note.

This issue has been Resolved.

While printing the Confirmation of Accounts for Sundry Creditors, the purchase Voucher number was getting printed instead of purchase reference details.

This issue has been resolved.

Rewrite/Backup/Restore/Split

On successfully taking a backup on a removable device it is found that all the files in the removable files drive were deleted.

This issue is now resolved and on successfully taking backup on the removable drive the tally backup file gets added to the device.

Service Tax

In the process of Splitting the company data of two financial years (2006-2008), wherein service tax sale and receipt records are in One financial Year (2006-2007) and The Service Tax payment record is in another financial year (2007-2008), Service Tax paid is displayed as Input Credit in the new company split from 1-4-2007

This issue has been resolved.

The Service Tax amount is not getting captured in the Bill wise Details screen while passing a Journal Voucher.

This issue is resolved and the Service Tax Amount is captured in the bill-wise details screen.

Synchronization

When the user tries to synchronize the data which contains '<' or '>' symbol specified either in narration or ledger or stock item name field or in any other field, Tally Displays the XML tag (< or >) in the calculator panel without terminating the Synchronization process.

This issue has been resolved.

Tax Collected at Source

In the alteration screen of sales invoice, on changing the sale value, the TCS surcharge amount was not getting changed as per the new gross value.

This issue has been resolved.

When TCS (Tax) Ledger is included in a Sales Invoice in alteration mode, TCS amount was not getting calculated.

This issue has been resolved.

Tax Deducted at Source

e-TDS displays an error message as **"Date on which Amount paid/Credited/Debited must be within the Financial Year and before the End Date of Quarter"** for the 4th Quarter as the reconciliation date fall in the subsequent Financial year.

This issue has been resolved.

TDL issues

Rate type was not working in Batch wise details during Stock item creation but was working in alteration mode.

This issue has been resolved and Support for using Rate type in batch-wise details in Creation mode is now provided.

VAT

In the alteration screen of a sales voucher, the VAT/Tax class was not being displayed in the Accounting Details screen, if the VAT ledger is being edited in the alteration mode.

This issue has been resolved and the **VAT/Tax class** is now being updated in the **Accounting Details** screen of the entry even if the ledger name is edited after recording the transaction.

In a sales invoice of Accounts Only company, The Output VAT amount of transaction involving two different VAT rates was calculated on the current sub-totals and not the assessable value.

This issue has been resolved and now output VAT amount is being calculated on the **assessable value** of the respective sales ledgers.

When a Purchase or Sales Ledger Name is altered in the Accounting Details for screen of a VAT transaction in alteration mode, the respective VAT/Tax Classification of the altered sales or purchase ledger is not displayed in the VAT/Tax Class field.

This issue has been resolved.