

Integration using ODBC Interface

Last updated on: August 22, 2025 | 7 min read

Open Database Connectivity (ODBC) is an interface that enables access to data across a heterogeneous environment of relational and non-relational database management systems.

ODBC is an Application Program Interface (API) specification that uses SQL (Structured Query Language) to interact with multiple databases, providing high interoperability. This means a developer can create a single application that accesses various data sources without targeting a specific one.

Components of a Typical ODBC Implementation

- **ODBC Client**

Implements the ODBC API, which interacts with the database-specific ODBC Driver.

- **ODBC Driver**

A library that processes ODBC function calls, submits SQL requests, and returns results.

- **ODBC Server**

A database that understands SQL and can respond to requests from the ODBC Driver.

TallyPrime can function both as an **ODBC Server** and an **ODBC Client**.

TallyPrime as an ODBC Client

Retrieving Data from External Databases

TallyPrime is ODBC-enabled and can communicate with ODBC Drivers of external databases. This is done using TDL Collections, which fetch data using SQL queries.

Ways to connect:

- **Using DSN (Data Source Name)**
- **DSN-less connections** by specifying the driver, driver ID, and path.

Each SQL result row becomes an object in Tally, and each column becomes a method.

Syntax

```
[Collection:<Collection Name>]
```

```
    ODBC : <Driver Info>
```

```
    SQL : <SQL Statement>
```

Where <Driver Info> can be a DSN or ODBC driver, Driver ID & path of the data source can be mentioned and <SQL Statement> is SELECT query.

Example 1: Import the Ledger Master from MS Access

Sample Data

Ledger Master		
Led Name	Led Parent	Opening Balance
Ram & Shyam Party	Sundry Debtors	-101
Universal Agencies	Sundry Creditors	5000
Paramount Ltd	Sundry Debtors	1000
Bangalore Cable Network	Sundry Creditors	1000

TDL Collection to gather data from MS Access

[Collection: Led Coll From Access]

ODBC: "Driver={Microsoft Access Driver (*.mdb)};Dbq=C:Masters.mdb;Uid=;Pwd=;"

SQL : Select * From LedgerMaster

[Collection: Led Coll]

Source Collection : Led coll From Access

Compute : LedName : \$_1

Compute : LedParent : \$_2

Compute : LedOpBal : \$\$AsAmount:\$_3

Alternatively TDL Collection gathers data from MS Access in following way

[Collection: Led Coll From Access]

ODBC : "Driver={Microsoft Access Driver (*.mdb)};Dbq=C:Masters.mdb;Uid=;Pwd=;"

SQL : Select * From LedgerMaster

SQL Object : AccessObj

[Object: AccessObj]

LedName : \$_1

LedParent : \$_2

LedOpBal : \$\$AsAmount:\$_3

Utilizing the ODBC Collection in a user defined Function to store the Ledger Objects in Tally DB

[Function: Ledger Import]

01 : WALK COLLECTION: Led Coll

02 : IF : @@LedgerExists > 0

03 : NEW OBJECT: Ledger : \$LedName

04 : CALL : Set Values

05 : SAVE TARGET

06 : ELSE

```
07 : NEW OBJECT: Ledger  
08 : CALL : Set Values  
09 : SAVE TARGET  
10 : END IF  
11 : END WALK  
12 : MSGBOX : "Status" : "Process completed successfully!!"  
13 : RETURN
```

[Function : Set Values]

```
01 : SET VALUE : Name : $LedName  
02 : SET VALUE : Parent: $LedParent  
03 : SET VALUE : Opening Balance : $LedOpBal
```

[System : Formula]

```
Ledger Exists : $$FilterCount:Ledger:IsMyLedger  
IsMyLedger : $Name = $$ReqObject:$LedName
```

Example 2 : Import the Ledger Master from MS Excel

TDL Collection to gather data from MS Excel

```
[Collection: ExcelData]  
ODBC : "Driver={Microsoft Excel Driver (*.xls)};Dbq=C:Masters.xls;DriverId=790"  
SQL : "Select * From [Sheet1$]"
```

TallyPrime as an ODBC Server

Accessing Tally Data from External Applications

External applications can access Tally data via:

- Tables
- Calling a Procedure

Retrieving Data Using Tables

Tally stores data as Objects. Externally, each Object = a row, and a Collection = a table.

Exposing Methods to ODBC

By prefixing '_' to external method(s) of an internal object or method(s) of an external Object can be exposed to ODBC.

Note:

- ◆ By default all the methods of the internal objects are exposed to ODBC
- ◆ Only First level methods of an Object can be exposed directly

Example

```
[#Object : Ledger]
```

```
_Difference : $ClosingBalance -$OpeningBalance
```

The code snippet alters an internal object, Ledger, to add an external method, _Difference and exposes it to ODBC.

Exposing Collections to ODBC

A Collection is exposed to ODBC by using the attribute, IsODBCTable .

Example

```
[Collection: Vouchers]
```

```
Type : Voucher
```

```
Is ODBCTable: Yes
```

The Collection Vouchers is exposed to ODBC by using the attribute IsODBCTable.

Example 1: Firing SQL statements from MS Excel to TallyPrime ODBC Server

Step 1 : - Open a New Work Book in MS Excel

Step 2: Go to Data > Get Data > From other sources > From ODBC

Step 3 : Select Tally ODBC Driver from 'From ODBC' window.

Step 4 : Select Tally ODBC Driver from 'From ODBC' window.

Step 5 : View the result in Excel Sheet

Example 2: Firing SQL statements from a VB application to TallyPrime ODBC Server

Below mentioned code snippet can be used to establish ODBC connection with TallyPrime and to then SQL queries can be fired.

```
Dim TallyCn As ADODB.Connection  
Set TallyCn = New ADODB.Connection  
TallyCn.Open "TallyODBC_9000"  
Set rst = New ADODB.Recordset  
rst.Open "Select $Name From Ledger", TallyCn, adOpenDynamic, adLockOptimistic
```

Retrieving Data By Calling an SQL Procedure

A Client application can call a SQL Procedure of a TallyPrime. But within TallyPrime this is a Collection with its name prefixed with an underscore. The Collection attribute, SQLParms is used to pass parameters to procedures. This Collection takes the parameter from the Client application by using the Collection attribute 'SQL Params'. The Collection attributes 'SQL Values' is used to return the values from procedure back to the client application.

Collection Attribute – SQLParms

This attribute is used to pass parameter(s) to a SQL Procedure. The parameter is a System variable.

SQL Params : <Parameter>

Where <Parameter> is a name of the Variable

Collection Attribute – SQLValues

This attribute returns value to client application. SQLValues require two parameters, the Column Name and the values for the column.

SQL Values : <Column Name> : <Expression>

Where <Column Name> is a name of the column i.e. column header and <Expression> is the expression which evaluates to a value and returned back to the client application.

Usage of SQL Procedure

The SQL Procedure '_StkBatches' displays the Batch name and Closing Balance for a given Stock Item.

[Collection : _StkBatches]

Type : Batches

Childof : ##StkItemName

SQLParms : StkItemName

SQLValues : Name : \$Name

SQLValues : Amount :\$ClosingBalance

[Variable:StkItemName]

Type : String

Example 1: Calling the SQL Procedure '_StkBatches' in MS Excel

Step 1 :- Open a New Work Book in MS Excel

Step 2 :- Go to Data -> Import External Data -> New Database Query

Step 3:- Select Tally ODBC Driver from 'Choose Data Source' window to open 'Microsoft Query' screen

Step 4 :- Go to File -> Execute SQL

Step 5:- In 'Execute SQL' window click on 'Procedures'

Step 6 : In 'Select Procedures' window, select procedure '_StkBatches'

Step 7: Pass appropriate stock item name as parameter to the procedure and 'Execute'

Step 8:- View the result in 'Query1' window

Step 9:- From 'Microsoft Query' screen, Go to File -> Return Data to Microsoft Office Excel

Step 10 :- View the Result in Excel sheet .

Example 2: Calling the SQL Procedure from a VB Application

```
Dim DBcon As New ADODB.Connection  
Dim objCmd As New ADODB.Command  
Dim objRs As New ADODB.Recordset  
DBcon.CursorLocation = adUseClient  
  
'Establish the connection using Tally ODBC Driver DBcon.Open "TallyODBC_9000"  
objCmd.ActiveConnection = DBcon objCmd.CommandType = adCmdStoredProc objCmd.CommandText  
= "_PartyBills"  
  
'Pass the the Stock Item Name as Parameter  
objCmd.CreateParameter (ODBCMAIN.CmbLedger.Text)  
  
'Call the SQL procedure  
Set objRs = objCmd.Execute
```

Using calculator pane for testing SQL commands

TallyPrime has an in-built SQL processor that processes SQL Select statements on collections. By default, only the collections at first level are available for selection.

Syntax

Select [<Method Name/s> <*>] from <Collection / Table> where <Condition> order by <Method Name/s>

Example

```
Select $Name from Ledger  
  
Select $Name, $ClosingBalance from Ledger Select * from Ledger  
  
Select $Name from ODBCTables  
  
Select $Name, $ClosingBalance from Ledger where $$IsDr:$ClosingBalance order by $ClosingBalance  
DESC  
  
Select $Name, $ClosingBalance from Ledger where $$IsDr:$ClosingBalance order by $ClosingBalance  
Select TOP 2 from Ledger
```

Have another query?

 Scan and Chat



OR "Save the number
+91 9019910043 to chat with us"

» On this
page