# MaxAccount

## Open Source Accounting Software

**with .net library MaxAccountExtension.dll**

## Github: https://github.com/YuWaiPang/MaxAccount

## Command List and Syntax

## *It looks like an Accounting ETL*

Version: 18-November-2021

## MaxAccount Command by Unit

### Unit 1: Copy Table

- CopyTable
- CopyTable.CommonTable
- CSV2LedgerRAM
- CSV2LedgerRAM.CommonTable
- DataTable2LedgerRAM
- LedgerRAM2CSV
- LedgerRAM2DataTable
- LedgerRAM2JSON
- LedgerRAM2HTML
- LedgerRAM2XML
- LedgerRAM2OneColumn
- ManyCSV2LedgerRAM
- OneColumn2LedgerRAM

### Unit 2: Add Column

- ComputeColumn
- ConditionalJoin
- FullJoin
- InnerJoin
- JoinTable
- Number2Text
- ResidualJoin

### Unit 3: Data Filter and Selection

- Filter.AndCondition
- Filter.AndConditionTable
- Filter.AndDistinctTable
- Filter.OrCondition
- Filter.OrCondition
- Filter.OrDistinctTable
- RemoveColumn
- SelectColumn

### Unit 4: Presentation of Data

- Crosstab
- Distinct
- GroupBy
- OrderBy
- ReverseCrosstabCSV2LedgerRAM
- ReverseManyCrosstabCSV2LedgerRAM

### Unit 5: Amendment

- AmendColumnName
- AmendDate
- ReverseNumber
- AmendDateFormat

### Unit 6: Add Row from Cell / Table

- AppendRow
- ComputeCell
- CombineTableByCommonColumn
- MergeCommonTable
- MergeTable
- Table2Cell

**Unit 7: Accounting**

- Amortization
- CurrentBuildBalanceSetting
- BuildDailyBalance
- BuildMonthlyBalance
- BuildWeeklyBalance
- BuildDailyBalanceCrosstabPeriod
- BuildMonthyBalanceCrosstabPeriod
- BuildWeeklyBalanceCrosstabPeriod
- Date2EffectiveDate
- Date2DailyPeriod
- Date2WeeklyPeriod
- Date2YearlyPeriod
- DC2NegativePositive
- DC2PositiveNegative
- NegativePositive2DC
- PositiveNegative2DC
- ReverseDC
- ReverseDailyVoucher
- ReverseWeeklyVoucher
- ReverseMonthlyVoucher
- VoucherEntry

**Unit 8: Process Control**

- ContinueProcess
- CurrentTable
- Disable
- Enable
- EndProcess
- ParallelProcess
- Process
- ReplaceRule

**Unit 9: Conditional Process Control**

- AndCondition2Action
- AndCondition2Cell
- OrCondition2Action
- OrCondition2Cell

**Unit 10: Utility**

- FileList2LedgerRAM
- Rule2LedgerRAM

**Unit 11: MaxAccount SQL**

*Initial Setting*

- CurrentConnectionString
- CurrentSQLServer
- CreateSQLServerDatabase

*CopyTable*

- DataTableClone2SQLServer
- LedgerRAMClone2SQLServer
- SQLTable2LedgerRAM

*Data Filter and Selection*

- FilterSQLRow.AndCondition
- FilterSQLRow.AndConditionTable
- FilterSQLRow.AndDistinctTable
- FilterSQLRow.OrCondition
- FilterSQLRow.OrConditionTable
- FilterSQLRow.OrDistinctTable

*Presentation of Data*

- CrosstabSQLTable
- DistinctSQLTable
- GroupSQLTableBy

*Amendment*

- DataTableAppend2SQLServer
- LedgerRAMAppend2SQLServer
- LedgerRAMAmend2SQLServer
- RemoveSQLRow.AndCondition
- RemoveSQLRow.AndConditionTable
- RemoveSQLRow.AndDistinctTable
- RemoveSQLRow.OrCondition
- RemoveSQLRow.OrConditionTable
- RemoveSQLRow.OrDistinctTable
- RemoveSQLDatabase
- RemoveSQLColumn
- RemoveSQLTable

*Accounting*

- BuildDailySQLBalance
- BuildMonthlySQLBalance
- BuildWeeklySQLBalance
- BuildDailySQLBalanceCrosstabPeriod
- BuildMonthySQLBalanceCrosstabPeriod
- BuildWeeklySQLBalanceCrosstabPeriod

*Utility*

- RunNonQuerySQL
- RunSQL2DataTable
- RunSQL2LedgerRAM

## MaxAccount Command Syntax by Unit

### Unit 1: Copy Data

Except for using the symbol *, there is no change in the final meaningful content of the table after the copying process.

- CopyTable{Source Table | ~ Result Table}
- CopyTable.CommonTable{Source Table | @ CommonTable ~ Result Table}
- CSV2LedgerRAM{Source File ~ Result Table}
- CSV2LedgerRAM.CommonTable{Source File @ CommonTable ~ Result Table }
- DataTable2LedgerRAM{DataTable ~ Result Table}
- LedgerRAM2CSV{Source Table | Column 1, Column n ~ Result File}
  LedgerRAM2CSV{Source Table | * ~ Result File}
- LedgerRAM2DataTable{Source Table | Column 1, Column n ~ Result Table}
  LedgerRAM2DataTable{Source Table | * ~ Result Table}
  *Note: DataTable is C# ADO.Net memory table.*
- LedgerRAM2JSON{Source Table | Column 1, Column n ~ Result File}
  LedgerRAM2JSON{Source Table | * ~ Result File}
- LedgerRAM2HTML{Source Table | Column 1, Column n ~ Result File}
- LedgerRAM2HTML{Source Table | * ~ Result File}
- LedgerRAM2MSSQL{Source Table | * ~ DataBaseName(Table Name)}
- LedgerRAM2OneColumn{Source Table | * ~ Result File}
- LedgerRAM2XML{Source Table | Column 1, Column n ~ Result File}
- LedgerRAM2XML{Source Table | * ~ Result File}
- ManyCSV2LedgerRAM{FolderPath(Path)FileFilter(Filter)Subdirectory(Include/Exclude) ~ Result Table }
- OneColumn2LedgerRAM{Source File ~ Result Table}

Rule details as indicated by blue color are optional settings.

| Reserve Symbol | Description |
|---|---|
| { } | Start/end of rule detail of a current rule type |
| # | Block name for a group of rule |
| \| | Read data from a source table (must be LedgerRAM) |
| , | Separate different refer names i.e. cell, column, table name |
| " " | Indicate a text or number value (it is not column name) |
| * | Select all column names |
| [] | Include row number of a column |
| () | Parameter(Setting) |
| = | Assign new value of a column name |
| => | Start mathematical or statistical calculation |
| @ | Set relation with alternative table |
| . | Start indicate number of decimal places of result column |
| ~ | Save data to result table (must be LedgerRAM) |

LedgerRAM represents in-memory data tables. Symbols as indicated in red color are designed to work with LedgerRAM tables and cells.

For all LedgerRAM tables and cells created or copied by using the symbol ~ and () respectively, they will be kept within memory until the process is completed. It can be reused or overwritten using the same table/cell name with ~ and ().

**Unit 2: Add Column**

Column can be added after the last column. To select or reorganize column names, you can use the rule type **Select Column**.

- ComputeColumn{Source Table | Column 1, Column n => Maths(Result Column.d) ~ Result Table}
  *Where Maths can be Add, Subtract, Multiply, Divide, CombineText.*
  *d represents decimal places which support >0.*
  *For Add, Subtract, Multiply and Divide, if you apply the number instead of the cell name, please double quote "" with the number, e.g. "2".*
  *For CombineText, if you apply the text instead of the cell name, , please double quote "" with the number, e.g. "apple".*

- ConditionalJoin{Transaction Table(Column 1, Column n) @ Master Table(Column 1, Column n) ~ Result Table}
- FullJoin{Transaction Table(Column 1, Column n) @ Master Table(Column 1, Column n) ~ Result Table}
- InnerJoin{Transaction Table(Column 1, Column n) @ Master Table(Column 1, Column n) ~ Result Table}
- JoinTable{Transaction Table(Column 1, Column n) @ Master Table(Column 1, Column n) ~ Result Table}
- Number2Text{Source Table | Number Column 1, Number Column n ~ Result Table}
- ResidualJoin{Transaction Table(Column 1, Column n) @ Master Table(Column 1, Column n) ~ Result Table}

**ResidualJoin** also follows condition rules of **ConditionalJoin**, but ResidualJoin runs slower than ConditionalJoin because it runs each master table row after the process completed of prior master table row. If your condition rules do not have any dependency across each master table row, you can use ConditionalJoin to achieve faster processing speed.

**Unit 3: Data Filter and Selection**

Source table will not be changed if the new table is saved as an alternative name.

- AndFilter{Source Table | Column 1(Condition) Column n(Condition) ~ Result Table}
- AndFilter.ConditionList{Source Table | @ ConditionTable ~ Result Table}
- AndFilter.DistinctList{Source Table | @ DistinctTable ~ Result Table}
- OrFilter{Source Table | Column 1(Condition) Column n(Condition) ~ Result Table}
- OrFilter.ConditionList{Source Table | @ ConditionTable ~ Result Table}
  *Where condition can be any combination of > value, < value, >= value, <= value, = value, != Value*
- OrFilter.DistinctList{Source Table | @ DistinctTable ~ Result Table}
- RemoveColumn{Source Table | Column 1, Column n ~ Result Table}
- SelectColumn{Source Table | Column 1, Column n ~ Result Table}

**Unit 4: Presentation of Data**

Unit 3 and unit 4 are essential to support your different scenarios of reporting.

- ✧ Crosstab{Source Table | X(Column 1, Column n) Y(Column 1, Column n)
  => Statistics 1(Result Column) Statistic n(Result Column) ~ Result Table}
- ✧ Distinct{Source Table | Column 1, Column n ~ Result Table}
- ✧ GroupBy{Source Table | Column 1, Column n => Statistics 1(Result Column) Statistic n(Result Column)
  ~ Result Table}
  *Where statistics can be Count(), Sum(Column), Max(Column), Min(Column),*
  *if using count, no need to specify column name*
- ✧ OrderBy{Source Table | Column 1(A/D) Column n(A/D) ~ Result Table}
  *Where A is ascending order and D is descending order*
- ✧ ReverseCrosstabCSV2LedgerRAM{Source File ~ Result Table}
- ✧ ReverseManyCrosstabCSV2LedgerRAM{FolderPath(Path)FileFilter(Filter)Subdirectory(Include/Exclude) ~ Table1}

**Unit 5: Amend Column**

No additional column will be generated. Result table will be amended accordingly. If no setting of the result table, the source table will be amended.

- ✧ AmendColumnName{Source Table | Column = Column ~ Result Table}
- ✧ AmendDate{Cell Name => AddYear(n) AddMonth(n) AddDay(n) ~ Result Table}
  AmendDate{Cell Name => Year(n) Month(n) Day(n) ~ Result Table}
  AmendDate{Source Table | Column 1, Column n => AddYear(n) AddMonth(n) AddDay(n) ~ Result Table}
  AmendDate{Source Table | Column 1, Column n => Year(n) AddMonth(n) AddDay(n) ~ Result Table}
  AmendDate{Source Table | * => Year(n) AddMonth(n) AddDay(n) ~ Result Table}

  *To effect calculation of date, the date format of the source table must be in OLEAutomationDate format.*

- ✧ AmendDateFormat{Cell Name => OLEAutomationDate = MM-dd-yyyy ~ Result Table}
  AmendDateFormat{Source Table | Column 1, Column n => MM-dd-yyyy = OLEAutomationDate
  ~ Result Table}
  AmendDateFormat{Source Table | * => MM-dd-yyyy = OLEAutomationDate ~ Result Table}
  *Where n Day can be First, Last, 1,2,3*
- ✧ ReverseNumber{Source Table | Number Column 1, Number Column n ~ Result Table}

**Unit 6: Add Row from Cell / Table**

Prior to understanding how to **AppendRow**, you shall practice how to manipulate cells by **ComputeCell** and **Table2Cell**.

- ✧ AppendRow{Source Table | Column 1(Value or Cell Name) Column n(Value or Cell Name) ~ Result Table}
- ✧ ComputeCell{CellName1 / "Number 1", CellName n / "Number n"=> Maths(Cell Name.d)}
  *Where Maths can be Add, Subtract, Multiply, Divide, CombineText*
  *d represents decimal places which support >0.*
  *If you apply the number instead of the cell name, please double quote "" with the number, e.g. "2".*

- ✧ CombineTableByCommonColumn {Source Table 1, Source Table n ~ Result Table}

- ✧ MergeCommonTable{Source Table 1, Source Table n ~ Result Table}
- ✧ MergeTable{Source Table 1, Source Table n ~ Result Table}
- ✧ Table2Cell{Source Table | Column 1, Column n => Statistics(Cell Name)}
  *Where statistics can be Count, Sum, Max, Min or Average.*
- ✧ Table2Cell{Source Table | Column[Row] => CellAddress(Cell Name)}
  *Where row can be First, Last or row number such as 1,10,99. First represents row 1, Last represents last row.*

**CombineTableByCommonColumn** vs **MergeTable**: resulting column of MergeTable may be more than CombineTableByCommonColumn. If more columns are number type, CombineTableByCommonColumn is faster than MergeTable, If more columns are text type, MergeTable is faster than CombineTableByCommonColumn.

**Unit 7: Accounting**

Following rules are designed specifically for accounting practice. **VoucherEntry** offers simple settings to support general voucher preparation. **Amortization** supports more on specific voucher preparation. **BuildBalance** allows your massive volume of vouchers to be summarized in periodical balances with multiple currencies and segments.

- ✧ Amortization{Source Table | Cost($) StartDate(d) TotalTenor(t) Method(m) ~ Result Table}
  *Where m can be StraightLine, Rule78, ReducingBalance*
- ✧ BuildDailyBalance{Source Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result Table}
- ✧ BuildMonthlyBalance{Source Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result Table}
- ✧ BuildWeeklyBalance{Source Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result Table}
- ✧ BuildDailyBalanceCrosstabPeriod{Source Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result Table}
- ✧ BuildWeeklyBalanceCrosstabPeriod{Source Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result Table}
- ✧ BuildMonthlyBalanceCrosstabPeriod{Source Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result Table}

**Example of LedgerMaster**

| Ledger | Account | Account Name | Year End:Account |
|---|---|---|---|
| BP01..BP02 | 1000 | Investment properties | 1000 |
| BP01..BP02 | 1090 | Fair Value Adjustment | 1090 |
| BP01..BP02 | 1100 | Fixed assets - cost | 1100 |
| BP01..BP02 | 1150 | Accumulate depreciation | 1150 |
| BP01..BP02 | 1510 | Lease commitment receivable | 1510 |
| BP01..BP02 | 1520 | Unearn lease income | 1520 |
| BP01..BP02 | 2100 | Prepaid charges | 2100 |
| BP01..BP02 | 2200 | Deposits | 2200 |
| BP01..BP02 | 2300 | Accounts receivable | 2300 |
| BP01..BP02 | 2310 | Accrued rental income | 2310 |
| BP01..BP02 | 2320 | Rental receivable | 2320 |
| BP01..BP02 | 2500 | Bank balances | 2500 |

| | | | |
|---|---|---|---|
| BP01..BP02 | 3100 | Tenants deposits received | 3100 |
| BP01..BP02 | 3200 | Accrued expenses | 3200 |
| BP01..BP02 | 3300 | Interest payable | 3300 |
| BP01..BP02 | 3400 | Tax Payable | 3400 |
| BP01..BP02 | 3500 | Bank loan due within one year | 3500 |
| BP01..BP02 | 4500 | Bank loan due after one year | 4500 |
| BP01..BP02 | 5100 | Share capital | 5100 |
| **BP01..BP02** | **5200** | **Retained earning - b/f** | **5200** |
| BP01..BP02 | 6100 | Rental income | 5200 |
| BP01..BP02 | 6500 | Bank interest income | 5200 |
| BP01..BP02 | 6900 | Sundry income | 5200 |
| BP01..BP02 | 7100 | Building management fee | 5200 |
| BP01..BP02 | 7800 | Exchange Variation Account | 5200 |
| BP01..BP02 | 8110 | Audit fee | 5200 |
| BP01..BP02 | 8120 | Bank charges | 5200 |
| BP01..BP02 | 8130 | Bank loan interest expenses | 5200 |
| BP01..BP02 | 8140 | Business registration fee | 5200 |
| BP01..BP02 | 8160 | Depreciation | 5200 |
| BP01..BP02 | 8170 | Electricity & water | 5200 |
| BP01..BP02 | 8180 | Insurance | 5200 |
| BP01..BP02 | 8200 | Legal & professional fees | 5200 |
| BP01..BP02 | 8220 | Printing & stationery | 5200 |
| BP01..BP02 | 8230 | Rates & government rent | 5200 |
| BP01..BP02 | 8240 | Repair & maintenance | 5200 |
| BP01..BP02 | 8250 | Staff cost | 5200 |
| BP01..BP02 | 8260 | Sundry expenses | 5200 |
| BP01..BP02 | 8270 | Telephone & fax | 5200 |
| BP01..BP02 | 8280 | Travelling expenses | 5200 |
| BP01..BP02 | 8310 | Club House Net Expenses | 5200 |
| BP01..BP02 | 8320 | Coach Service | 5200 |
| BP01..BP02 | 8330 | Lifts & Escalators Maintenance | 5200 |
| BP01..BP02 | 8340 | Minor Asset Items | 5200 |
| BP01..BP02 | 8350 | Recreation/Promotion | 5200 |
| BP01..BP02 | 8360 | Security Service | 5200 |
| BP01..BP02 | 8370 | Uniform | 5200 |
| BP01..BP02 | 8900 | Taxation | 5200 |

For each end of financial year, account balance will be carried forward to account as indicated by the column **Year End:Account.** The system support none to many retained accounts.

- ✧ CurrentBuildBalanceSetting{LedgerMasterTable}
- ✧ Date2EffectiveDate{Source Table | Date Column ~ Result Table}
- ✧ Date2DailyPeriod{Source Table | Column(Date Column) StartDay(n) ~ Result Table}
- ✧ Date2WeeklyPeriod{Source Table | Column(Date Column) StartWeek(n) ~ Result Table}
- ✧ Date2MonthlyPeriod{Source Table | Column(Date Column) StarMonth(n) ~ Result Table}
- ✧ DC2NegativePositive{Source Table | Number Column 1, Number Column n ~ Result Table}

- DC2PositiveNegative{Source Table | Number Column 1, Number Column n ~ Result Table}
- NegativePositive2DC{Source Table | Number Column 1, Number Column n ~ Result Table}
- PositiveNegative2DC{Source Table | Number Column 1, Number Column n ~ Result Table}
- ReverseDC{ Source Table | ~ Result Table}
- ReverseDailyVoucher{Column 1, Column n => AddYear(n) AddMonth(n) AddDay(n) ~ Result Table}
  ReverseDailyVoucher{Source Table | Column 1, Column n => Year(n) Month(n) Day(n) StartDay(n) }
  ReverseDailyVoucher{Source Table | * => Year(n) AddMonth(n) Day(n) @ LedgerMaster ~ Result Table}
- ReverseWeeklyVoucher{Column 1, Column n => NextPeriodAddDay(n) StartWeek(n) ~ Result Table}
- ReverseMonthlyVoucher{Column 1, Column n => AddYear(n) AddMonth(n) AddDay(n) ~ Result Table}
  ReverseMonthlyVoucher{Source Table | Column 1, Column n => Year(n) Month(n) Day(n) StartMonth(n)}
  ReverseMonthlyVoucher{Source Table | * => Year(n) AddMonth(n) Day(n) @ LedgerMaster ~ Result Table}
- VoucherEntry{ Source Table | Debit(Column) Credit(Column) Balance(Column)
  ExcludeBalanceGroupBy(Column) ~ Result Table}

**Unit 8: Process Control**

Unit 8 introduces simple process control while unit 9 introduces more advanced process control.

- ContinueProcess{Command Group Name}
- CurrentTable{Table}
- Disable{Message2Screen, Message2File}
- Enable{Message2Screen, Message2File}
- EndProcess{}
- ParallelProcess{Command Group Name}
- Process{Command Group Name}
  *Where "Exit" is an exit function of the Command Group Name of Process{}*
- ReplaceRule{Rule1, Rulen}
  *Support first row to define Replace Rule*

**Unit 9: Conditional Process Control**

Actual cell value is essential to work with condition. You can use **ComputeCell** and **Table2Cell** as introduced in unit 6 to generate one or more cell value(s) to work with condition(s).

- AndCondition2Cell{CellName 1(Condition)CellName n(Condition) => Cell Name}
- AndCondition2Action{CellName 1(Condition)CellName n(Condition) => Process(Rule)}
  *Where Condition can be any combination of > value, < value, >= value, <= value, = value, != Value*
- OrCondition2Action{CellName 1(Condition)CellName n(Condition) => Process(Rule)}
- OrCondition2Cell{CellName 1(Condition)CellName n(Condition) => Cell Name}
  *Where Condition can be any combination of > value, < value, >= value, <= value, = value, != Value*

**Unit 10: Utility**

These 2 commands are used to support your management of rule files.

- FileList2LedgerRAM{FolderPath(Path)FileFilter(Filter)Subdirectory(Exclude/Include) ~ Result Table}
- Rule2LedgerRAM{Source Table | Column Name of CalcRule File Path ~ Result Table}

**Unit 11: MaxAccount SQL**

Startup

- CurrentConnectionString{Connection String}
  *e.g. Server=localhost\SQLEXPRESS;Database=master;Trusted_Connection=True;*
- CurrentSQLServer{Microsoft SQL Server, Version}
- CreateSQLDatabase{Database Name}

*CopyTable*

- DataTableClone2SQLServer{DataTable ~ Result Table}
- LedgerRAMClone2SQLServer{LedgerRAM | Column 1, Column n ~ Result LedgerRAM}
  LedgerRAMClone2SQLServer{ LedgerRAM | * ~ Result LedgerRAM}
- SQLTable2LedgerRAM{SQL Table | * ~ Result LedgerRAM}
  SQLTable2LedgerRAM{SQL Table | Column 1, Column n ~ Result LedgerRAM}

*Data Filter and Selection*

- FilterSQLRow.AndCondition{SQL Table | Column 1(Condition), Column n(Condition)}
- FilterSQLRow.AndConditionTable{SQL Table | @ LedgerRAM ~ Result LedgerRAM}
- FilterSQLRow.AndDistinctTable{SQL Table | @ LedgerRAM ~ Result LedgerRAM}
- FilterSQLRow.OrCondition{SQL Table | Column 1(Condition), Column n(Condition) ~ Result LedgerRAM}
- FilterSQLRow.OrConditionTable{SQL Table | @ LedgerRAM ~ Result LedgerRAM}
- FilterSQLRow.OrDistinctTable{SQL Table | @ LedgerRAM ~ Result LedgerRAM}

*Presentation of Data*

- CrosstabSQLTable{Source Table | X(Column 1, Column n) Y(Column 1, Column n)
  => Statistics 1(Result Column) Statistic n(Result Column) ~ Result Table}
- DistinctSQLTable{Source SQL Table | Column 1, Column n ~ Result LedgerRAM}
- GroupSQLTableBy{Source SQL Table | Column 1, Column n => Statistics 1(Result Column) Statistic n(Result Column) ~ Result LedgerRAM }
  *Where statistics can be Count(), Sum(Column), Max(Column), Min(Column),*
  *if using count, no need to specify column name*

*Amendment*

- DataTableAppend2SQLServer{DataTable ~ Result SQL Table}
- LedgerRAMAppend2SQLServer{LedgerRAM | Column 1, Column n ~ Result SQL Table}
  LedgerRAMAppend2SQLServer{LedgerRAM | * ~ Result SQL Table}
- LedgerRAMAmend2SQLServer{LedgerRAM | AmendKey(Column 1, Column n) AmendMode (Period/Fiscal Year/Calendar Year/Month/Day) ~ Result LedgerRAM}
- RemoveSQLRow.AndCondition{SQL Table | Column 1(Condition), Column n(Condition)}
- RemoveSQLRow.AndConditionTable{SQL Table | @ LedgerRAM}
- RemoveSQLRow.AndDistinctTable{SQL Table | @ LedgerRAM}
- RemoveSQLRow.OrCondition{SQL Table | Column 1(Condition), Column n(Condition)}
- RemoveSQLRow.OrConditionTable{SQL Table | @LedgerRAM}
- RemoveSQLRow.OrDistinctTable{SQL Table | @LedgerRAM}
- RemoveSQLDatabase{Database Name}
- RemoveSQLColumn{SQL Table | Column 1, Column n}
- RemoveSQLTable{SQL Table}

*Accounting (Current version has not included this functions)*

- BuildDailySQLBalance{SQL Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result LedgerRAM}
- BuildMonthlySQLBalance{SQL Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result LedgerRAM}
- BuildWeeklySQLBalance{SQL Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result LedgerRAM}
- BuildDailySQLBalanceCrosstabPeriod{SQL Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result LedgerRAM}
- BuildMonthlySQLBalanceCrosstabPeriod{SQL Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result LedgerRAM}
- BuildWeeklySQLBalanceCrosstabPeriod{SQL Table | Text Column 1, Text Column n, Amount Column 1, Amount Column n @ LedgerMaster ~ Result LedgerRAM}

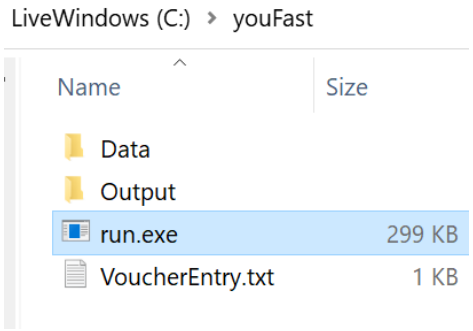
*Utility*

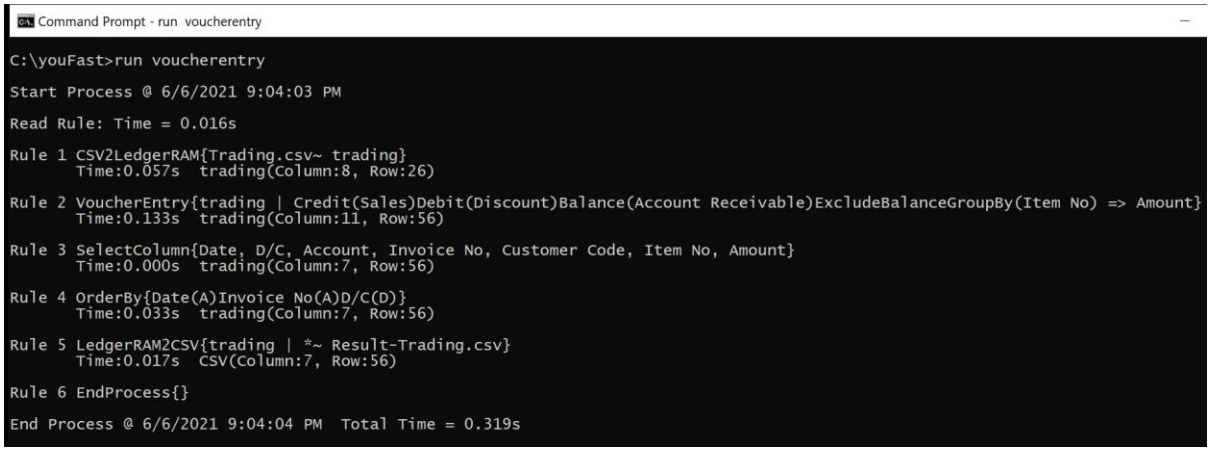
- RunNonQuerySQL{SQL Statement}
- RunSQL2DataTable{SQL Statement ~ Result DataTable}
- RunSQL2LedgerRAM{SQL Statement ~ Result LedgerRAM}

## Sample App: Voucher Entry

CSV2LedgerRAM{Trading.csv ~ trading}
VoucherEntry{Credit(Sales) Debit(Discount) Balance(Account Receivable) ExcludeBalanceGroupBy(Item No) => Amount}
*Use **ExcludeBalanceGroupBy** can prevent Account Receivable be aggregated at item no level when you require it aggregated at invoice level.*
SelectColumn{Date, D/C, Account, Invoice No,Customer Code, Item No, Amount}
OrderBy{Date(A) Invoice No(A) D/C(D)}
LedgerRAM2CSV{~ Result-Trading.csv}
EndProcess{}

Prior to run this app, please copy run.exe file to c:\youFast



## Source Table

| Date | Customer Code | Item No | Sales | Discount | Invoice No | Supplier Code | Purchase |
|---|---|---|---|---|---|---|---|
| 42008 | AA01 | Product 1 | 6000 | 600 | 201501001 | EA01 | 5000 |
| 42008 | AA01 | Product 2 | 14640 | 1464 | 201501001 | EA01 | 12200 |
| 42008 | AA01 | Product 5 | 9432 | 943 | 201501001 | EA01 | 7860 |
| 42010 | AC02 | Product 6 | 13440 | 672 | 201501002 | EC02 | 11200 |
| 42010 | AC02 | Product 8 | 7980 | 399 | 201501002 | EC02 | 6650 |
| 42010 | AC02 | Product 11 | 9504 | 475 | 201501002 | EC02 | 7920 |
| 42010 | AC02 | Product 12 | 13344 | 667 | 201501002 | EC02 | 11120 |
| 42010 | AC02 | Product 20 | 20160 | 1008 | 201501002 | EC02 | 16800 |
| 42013 | BB01 | Product 10 | 15000 | 1200 | 201501003 | SB01 | 12500 |
| 42013 | BB01 | Product 3 | 15240 | 1219 | 201501003 | SB01 | 12700 |
| 42013 | BB01 | Product 4 | 12000 | 960 | 201501003 | SB01 | 10000 |
| 42013 | BB01 | Product 9 | 19872 | 1590 | 201501003 | SB01 | 16560 |
| 42013 | BB01 | Product 10 | 9864 | 789 | 201501003 | SB01 | 8220 |
| 42014 | BB10 | Product 1 | 16764 | 1509 | 201501004 | SB10 | 13970 |
| 42014 | BB10 | Product 2 | 15600 | 1404 | 201501004 | SB10 | 13000 |
| 42014 | BB10 | Product 3 | 3240 | 292 | 201501004 | SB10 | 2700 |
| 42014 | BB10 | Product 4 | 6144 | 553 | 201501004 | SB10 | 5120 |
| 42014 | BB10 | Product 5 | 7740 | 697 | 201501004 | SB10 | 6450 |
| 42014 | BB10 | Product 6 | 12960 | 1166 | 201501004 | SB10 | 10800 |
| 42014 | BB10 | Product 7 | 20160 | 1814 | 201501004 | SB10 | 16800 |
| 42015 | CC01 | Product 2 | 43200 | 2592 | 201501005 | ZC01 | 36000 |
| 42015 | CC01 | Product 7 | 15600 | 936 | 201501005 | ZC01 | 13000 |
| 42015 | CC01 | Product 11 | 29280 | 1757 | 201501005 | ZC01 | 24400 |
| 42015 | CC01 | Product 13 | 20700 | 1242 | 201501005 | ZC01 | 17250 |
| 42015 | CC01 | Product 15 | 8820 | 529 | 201501005 | ZC01 | 7350 |

## Result Table

| Date | D/C | Account | Invoice No | Customer Code | Item No | Amount |
|---|---|---|---|---|---|---|
| 42008 | D | Discount | 201501001 | AA01 | Product 1 | 600 |
| 42008 | D | Discount | 201501001 | AA01 | Product 2 | 1464 |
| 42008 | D | Discount | 201501001 | AA01 | Product 5 | 943 |
| 42008 | D | Account Receivable | 201501001 | AA01 | null | 27065 |
| 42008 | C | Sales | 201501001 | AA01 | Product 1 | 6000 |
| 42008 | C | Sales | 201501001 | AA01 | Product 2 | 14640 |
| 42008 | C | Sales | 201501001 | AA01 | Product 5 | 9432 |
| 42010 | D | Discount | 201501002 | AC02 | Product 6 | 672 |
| 42010 | D | Discount | 201501002 | AC02 | Product 8 | 399 |
| 42010 | D | Discount | 201501002 | AC02 | Product 11 | 475 |
| 42010 | D | Discount | 201501002 | AC02 | Product 12 | 667 |
| 42010 | D | Discount | 201501002 | AC02 | Product 20 | 1008 |
| 42010 | D | Account Receivable | 201501002 | AC02 | null | 61207 |
| 42010 | C | Sales | 201501002 | AC02 | Product 6 | 13440 |
| 42010 | C | Sales | 201501002 | AC02 | Product 8 | 7980 |
| 42010 | C | Sales | 201501002 | AC02 | Product 11 | 9504 |
| 42010 | C | Sales | 201501002 | AC02 | Product 12 | 13344 |
| 42010 | C | Sales | 201501002 | AC02 | Product 20 | 20160 |
| 42013 | D | Discount | 201501003 | BB01 | Product 10 | 1200 |
| 42013 | D | Discount | 201501003 | BB01 | Product 3 | 1219 |
| 42013 | D | Discount | 201501003 | BB01 | Product 4 | 960 |
| 42013 | D | Discount | 201501003 | BB01 | Product 9 | 1590 |
| 42013 | D | Discount | 201501003 | BB01 | Product 10 | 789 |
| 42013 | D | Account Receivable | 201501003 | BB01 | null | 66218 |
| 42013 | C | Sales | 201501003 | BB01 | Product 10 | 15000 |
| 42013 | C | Sales | 201501003 | BB01 | Product 3 | 15240 |
| 42013 | C | Sales | 201501003 | BB01 | Product 4 | 12000 |
| 42013 | C | Sales | 201501003 | BB01 | Product 9 | 19872 |
| 42013 | C | Sales | 201501003 | BB01 | Product 10 | 9864 |
| 42014 | D | Discount | 201501004 | BB10 | Product 1 | 1509 |
| 42014 | D | Discount | 201501004 | BB10 | Product 2 | 1404 |
| 42014 | D | Discount | 201501004 | BB10 | Product 3 | 292 |
| 42014 | D | Discount | 201501004 | BB10 | Product 4 | 553 |
| 42014 | D | Discount | 201501004 | BB10 | Product 5 | 697 |
| 42014 | D | Discount | 201501004 | BB10 | Product 6 | 1166 |

| | | | | | |
|---|---|---|---|---|---|
| 42014 | D | Discount | 201501004 BB10 | Product 7 | 1814 |
| 42014 | D | Account Receivable | 201501004 BB10 | null | 75173 |
| 42014 | C | Sales | 201501004 BB10 | Product 1 | 16764 |
| 42014 | C | Sales | 201501004 BB10 | Product 2 | 15600 |
| 42014 | C | Sales | 201501004 BB10 | Product 3 | 3240 |
| 42014 | C | Sales | 201501004 BB10 | Product 4 | 6144 |
| 42014 | C | Sales | 201501004 BB10 | Product 5 | 7740 |
| 42014 | C | Sales | 201501004 BB10 | Product 6 | 12960 |
| 42014 | C | Sales | 201501004 BB10 | Product 7 | 20160 |
| 42015 | D | Discount | 201501005 CC01 | Product 2 | 2592 |
| 42015 | D | Discount | 201501005 CC01 | Product 7 | 936 |
| 42015 | D | Discount | 201501005 CC01 | Product 11 | 1757 |
| 42015 | D | Discount | 201501005 CC01 | Product 13 | 1242 |
| 42015 | D | Discount | 201501005 CC01 | Product 15 | 529 |
| 42015 | D | Account Receivable | 201501005 CC01 | null | 110544 |
| 42015 | C | Sales | 201501005 CC01 | Product 2 | 43200 |
| 42015 | C | Sales | 201501005 CC01 | Product 7 | 15600 |
| 42015 | C | Sales | 201501005 CC01 | Product 11 | 29280 |
| 42015 | C | Sales | 201501005 CC01 | Product 13 | 20700 |
| 42015 | C | Sales | 201501005 CC01 | Product 15 | 8820 |

To output the date in accordance with other formatting e.g. dd-MMM-yyyy, please use the command **AmendDateFormat** as introduced in unit 5..

**Sample App: Monthly Trial Balance**

BuildBalanceCrosstabPeriod is the key rule of the app. This support to calculate monthly balance by reference to Voucher and LedgerMaster. LedgerMaster allows you to define how each account be carried forward it balance to next financial year. You can define from none to many retained account. Date2Period is used to define financial year.

```
CSV2LedgerRAM{LedgerMasterRange.csv ~ LedgerMaster}
CSV2LedgerRAM{VoucherList10X.csv ~ Voucher}
Date2MonthlyPeriod{DateColumn(Date) StartMonth(1)}
BuildMonthlyBalance{Voucher | Ledger, Account, Amount @ LedgerMaster ~ TrialBalance}
LedgerRAM2CSV{TrialBalance | * ~ Result-TrialBalanceByPeriod4a.csv}
```

**Sample App: Monthly Analysis Account Balance**

Traditional accounting software supports 3 ~ 5 level of analysis account balance. This software extend to support a limit of 12 ~ 16 levels of analysis account balances. Below are 2 levels of account balance by Contact and Analysus Rw.

```
CSV2LedgerRAM{LedgerMasterRange.csv ~ LedgerMaster}
CSV2LedgerRAM{VoucherList10X.csv ~ Voucher}
Date2MonthlyPeriod{Voucher | DateColumn(Date) StartMonth(1) ~ MonthlyVoucher}
BuildMonthlyBalanceCrosstabPeriod{MonthlyVoucher | Ledger, Account, Contact, Amount @ LedgerMaster ~
    AnalysisAccountBalanceCrosstabPeriod}
LedgerRAM2CSV{AnalysisAccountBalanceCrosstabPeriod | * ~ Result-MultiLevelAnalysisAccountBalanceCrosstabPeriod.csv}
BuildMonthlyBalance{MonthlyVoucher | Ledger, Account, Contact, Amount @ LedgerMaster ~ AnalysisAccountBalance}
LedgerRAM2CSV{AnalysisAccountBalance | * ~ Result-MultiLevelAnalysisAccountBalance.csv}
```

**Sample App: Amortization**

This app is not only generating amortization schedule, but also generate all relevant vouchers and summary reports. Using the command **Process** with symbol **#** are optional, however it allows rules be more organized.

```
Process{Import Data}
Process{Acquisition}
Process{Amortization}
Process{Disposal}
Process{Combine Voucher}
Process{Export Report}
EndProcess{}

# Import Data
CSV2LedgerRAM{Cost.csv ~ Table}
Amortization{Method(MonthlyBasis,StraightLine,ProRateActualDay,Round2) ~ AmortizedTable}
Date2MonthlyPeriod{DateColumn(Date) StartMonth(1)}
Number2Text{Tenor}
AmendColumnName{Text:Tenor = TextTenor}
LedgerRAM2CSV{AmortizedTable | * ~ Result-AmortizedTable.csv}

# Acquisition
AndFilter{Tenor(=0) ~ Acquisition}
SelectColumn{Date, Period Change, AssetID, Tenor, TextTenor, Acquisition}
VoucherEntry{Debit(Acquisition) Credit(Payable) => Amount}
ComputeColumn{"Acquisition" => CombineText(Voucher Type) ~ Acquisition2}
```

```
# Amortization
AndFilter{AmortizedTable | Amortization(>0) ~ Amortization}
SelectColumn{Date, Period Change, AssetID, Tenor, TextTenor, Amortization}
VoucherEntry{Debit(Amortization) Credit(AccAmortization) => Amount}
ComputeColumn{"Amortization" => CombineText(Voucher Type) ~ Amortization2}

# Disposal
AndFilter{AmortizedTable | Disposal(>0) ~ Disposal}
SelectColumn{Date, Period Change, AssetID, Tenor, Disposal}
ComputeColumn{Tenor, "1" => Subtract(Last Tenor)}
SelectColumn{AssetID, Last Tenor}
AmendColumnName{Last Tenor = Tenor}
AndFilter.DistinctList{@ AmortizedTable | Disposal ~ DisposalOfCost2}

SelectColumn{Date, Period Change, AssetID, Tenor, TextTenor, Acquisition, AccAmortization}
LedgerRAM2CSV{DisposalOfCost2 | * ~ Result-DisposalOfCost2.csv}
VoucherEntry{Debit(AccAmortization) Credit(Acquisition)Balance(Receivable) => Amount}
ComputeColumn{"Disposal" => CombineText(Voucher Type) ~ Disposal2}

# Combine Voucher
CombineTableByCommonColumn{Acquisition2, Amortization2, Disposal2 ~ Voucher}
OrderBy{AssetID(A) Period Change(A)}
AmendDateFormat{Date => OLEAutomationDate = dd-MMM-yyyy}
ComputeColumn{"Tenor ",TextTenor, ": ", Voucher Type => CombineText(Particular)}
SelectColumn{Date, Period Change, Voucher Type, D/C, Account, Amount, AssetID, Particular ~ VoucherList}

# Export Report
LedgerRAM2CSV{VoucherList | * ~ Result-VoucherList.csv}
Crosstab{X(Voucher Type, Account, D/C) Y(AssetID, Period Change) => Sum(Amount) ~ TrialBalanceByAssetIDByPeriod}
LedgerRAM2CSV{TrialBalanceByAssetIDByPeriod | * ~ Result-TrialBalanceByAssetIDByPeriod.csv}
Crosstab{VoucherList | X(Voucher Type, Account, D/C) Y(Period Change) => Sum(Amount) ~ TrialBalanceByPeriod}
LedgerRAM2CSV{TrialBalanceByPeriod | * ~ Result-TrialBalanceByPeriod.csv}
Crosstab{VoucherList | X(Voucher Type, Account, D/C) Y(AssetID) => Sum(Amount) ~ TrialBalanceByAssetID}
LedgerRAM2CSV{TrialBalanceByAssetID | * ~ Result-TrialBalanceByAssetID.csv}
```

## MaxAccount Command by Alphabetical Order

1. AmendColumnName
2. AmendDate
3. AmendDateFormat
4. Amortization
5. AndCondition2Action
6. AndCondition2Cell
7. AndFilter
8. AndFilter.ConditionList
9. AndFilter.DistinctList
10. AppendRow
11. BuildDailyBalance
12. BuildDailyBalanceCrosstabPeriod
13. BuildDailySQLBalance
14. BuildDailySQLBalanceCrosstabPeriod
15. BuildWeeklyBalance
16. BuildWeeklyBalanceCrosstabPeriod
17. BuildWeeklySQLBalance
18. BuildWeeklySQLBalanceCrosstabPeriod
19. BuildYearlyBalance
20. BuildYearlyBalanceCrosstabPeriod
21. BuildYearlySQLBalance
22. BuildYearlySQLBalanceCrosstabPeriod
23. CombineTableByCommonColumn
24. ComputeCell
25. ComputeColumn
26. ConditionalJoin
27. ContinueProcess
28. CopyTable
29. CopyTable.CommonTable
30. CreateSQLServerDatabase
31. Crosstab
32. CrosstabSQLTable
33. CSV2LedgerRAM
34. CSV2LedgerRAM.CommonTable
35. CurrentBuildBalanceSetting
36. CurrentConnectionString
37. CurrentSQLServer
38. CurrentTable
39. DataTable2LedgerRAM
40. DataTableAppend2SQLServer
41. DataTableClone2SQLServer
42. Date2DailyPeriod
43. Date2EffectiveDate
44. Date2WeeklyPeriod
45. Date2YearlyPeriod
46. DC2NegativePositive
47. DC2PositiveNegative
48. Disable
49. Distinct
50. DistinctSQLTable
51. Enable
52. EndProcess
53. FileList2LedgerRAM
54. FilterSQLRow.Condition
55. FilterSQLRow.AndConditionTable
56. FilterSQLRow.AndDistinctTable
57. FilterSQLRow.OrCondition
58. FilterSQLRow.OrConditionTable
59. FilterSQLRow.OrDistinctTable
60. FullJoin
61. GroupBy
62. GroupSQLTableBy
63. InnerJoin
64. JoinTable
65. LedgerRAM2CSV
66. LedgerRAM2DataTable
67. LedgerRAM2HTML
68. LedgerRAM2JSON
69. LedgerRAM2OneColumn
70. LedgerRAM2XML
71. LedgerRAMAmend2SQLServer
72. LedgerRAMAppend2SQLServer
73. LedgerRAMClone2SQLServer
74. ManyCSV2LedgerRAM
75. MergeCommonTable
76. MergeTable
77. NegativePositive2DC
78. Number2Text
79. OneColumn2LedgerRAM
80. OrCondition2Action
81. OrCondition2Cell
82. OrderBy
83. OrFilter
84. OrFilter.ConditionList
85. OrFilter.DistinctList
86. ParallelProcess
87. PositiveNegative2DC
88. Process
89. RemoveColumn
90. RemoveSQLColumn
91. RemoveSQLDatabase
92. RemoveSQLRow.AndCondition
93. RemoveSQLRow.AndConditionTable
94. RemoveSQLRow.AndDistinctTable
95. RemoveSQLRow.OrCondition
96. RemoveSQLRow.OrConditionTable
97. RemoveSQLRow.OrDistinctTable
98. RemoveSQLTable
99. ReplaceRule
100. ResidualJoin
101. ReverseCrosstabCSV2LedgerRAM
102. ReverseDailyVoucher
103. ReverseDC
104. ReverseManyCrosstabCSV2LedgerRAM
105. ReverseMonthlyVoucher
106. ReverseNumber
107. ReverseWeeklyVoucher
108. RunNonQuerySQL
109. RunSQL2DataTable
110. RunSQL2LedgerRAM
111. SelectColumn
112. SQLTable2LedgerRAM
113. Table2Cell
114. Rule2LedgerRAM
115. VoucherEntry

**Contact**
Email: yuwaipang@gmail.com
Linkedin: https://www.linkedin.com/in/max01/
Github: https://github.com/YuWaiPang/MaxAccount
YouTube: https://www.youtube.com/channel/UCouJHDI_7dkNbiEnuDpnFmg