
Struct
{element of idata}
+name:char
+dtype:char
+d1:xts
+w1:xts
+meta:list{
splits:xts
dividends:xts
interdaytimezone:char
intradaytimezone:char
rollon:char
volccount:int
gluedcontracts:list{
cfmonth:char
c2month:char
c3month:char
cfexpir:char
datasub:char
}}
+intraday:list
{element:list{
n
idata
Struct
{imetadata data frame}
IDNAM:char
TKR:char
TYP:char
LBL:char
grp:char
imetadata
n
«MIdataDtn»
{impl for DTN data}
+loadDataContractsDTN(o:MIdata,rollon:char,..)
+updDataContractsDTN(o:MIdata,rollon:char,..)
«MIdata»
Package::pmktdata
{principal package object}
idata:list
-imetadata:data.frame
-groupsordered:char
-dataprovider:char
-curmktidx:int
-mode:char
-dseq=:POSIXct
-enddate:POSIXct
-cdate:POSIXct
-histd1:int
-histh1:int
--constr
+instrlistdatacsv(csvfile:char,..):MIdata
+instrlistdata(instrvec:char,..):MIdata
--getters setters
+getCinstrument(o:MIdata):char
+getInstrumentLabel(o:MIdata:iname:char):char
+getSortedinstruments(o:MIdata):vector
+setInstrumentPosition(o:MIdata,pos:char):void
+setSpecificInstrumentPosition(o:MIdata,idnam:char):void
+{get/set}Mode(o:MIdata,..)
+{get/set}Cdate(o:MIdata,..)
+{get/set}Groups(o:MIdata,..)
+getCdatedata(o:MIdata,iname:char,tfvalue:char):Xts
+getcd1(o:MIdata):Xts
+getcw1(o:MIdata):Xts
+setInterdayTZ(o:MIdata,tz:char)
+getInterdayTZ(o:MIdata)
--load data func
+iniDataDld(o:MIdata,sdate:POSIXct)
+updDataDld(o:MIdata):void
+fetchNextDay(o:MIdata,n:int)
+fetchNextWeek(o:MIdata)
+iniDataContracts(o:MIdata,rollon:char,..)
+loadUp2dateContracts(o:MIdata,..)
+updDataContracts(o:MIdata,..)
--misc funcs
+insertGroup(o:MIdata,grp:char,pos:int):void
+removeDatatail(o:MIdata,rmd:bool,..)
+addMarketdata(o:MIdata,name:char,..)
+rmMarketdata(o:MIdata,name:char,..)
+printIdatMeta(o:MIdat,name:char,..)
+getIdatMetaRolls(o:MIdata,name:char):list
+getIdatMetaFrontcontract(o:MIdata,name:char,ctime:POSIXct):list
+getIdatMetaRollcrit(o:MIdata,name:char):char
«ContractSeqList»
Package::pmktdata
{helper object for list of contract seqs}
-csl:list
-topstoredir:char
-loggername:char
--constr
+nreadCSL(ivec:vector,topstoredir:char,freq:in..):ContractSeqList
+readCSL(topstoredir:char):ContractSeqList
--getters setters
+getInstrument(o:ContractSeqList):vector
+setLogger(o:contractSeqList):void
+{get/set}CS(o:ContractSEqList,cs:ContractSeq)
--data mgmt func
+updEOD(o:ContractSeqList,ctime:POSIXct,updMeta:bool,persist:bool)
+updIntra(o:ContractSeqList,ctime:POSIXct,interval:int,mofirst:bool,..)
+storecsl(o:ContractSeqList)
+storeIntracsl(o:ContractSeqList,interval:int)
--misc funcs
+lstliq(o:ContractSeqList,ctime:POSIXct)
+lstcur(o:ContractSeqList,ctime:POSIXct
+show(o:ContractSeqList)
csl
n
«ContrcactSeqDtn»
Package::pmktdata
{implementation for DTN data}
+iniDownloadDTN(o:ContractSEq,lstqtime:POSIXct)
+iniDownloadIntraDTN(o:ContractSEq,lstqtime:POSIXct,interval:int,..)
+updateEODDTN(o:ContractSEq,scontract:char,ctime:POSIXct)
+updateIntraDTN(o:ContractSEq,lstqtime:POSIXct,interval:int,..)
«ContractSeq»
Package::pmktdata
{futures contracts}
+instName:char
+tickerPref:char
+contSuff:char
+yrContracts:int
+topstoredir:char
+offsetd1fixed:int
+datasrc:char
+lastDataDaily:POSIXct
+cslogger:char
+contrseq:data.frame
+contrefdata:list
+data:list
+intrdconf:list
+intrdata:list
+tkdata:list
+rollrule:list
--constr
+crBareContractSeq(name:char,..):ContractSeq
+loadseq(rdatfile:char):ContractSeq
+readcs(name:char,topstoredir:char):ContractSeq
+nreadcs(name:char,topstoredir:char,..):ContractSeq
+csfromcfg(fname:char, cfgdir:char):ContractSeq
--getters setters
+getInstrumentName(o:ContractSeq):char
+getTickerPref(o:ContractSeq):char
+getLoggername(o:ContractSeq):char
+{get/set}Topstoredir(o:ContractSeq,..)
+setRollrule(o:ContractSeq:value:list)
+setRegularRollRule(o:ContractSeq,rollprd:int,fmvolpct:float)
+resetLogger(o:ConractSeq,level:char)
+getD1offset(o:ConractSeq):int
+{get/set}ContractSequence(o:ContractSeq,..)
+getContract(o:ContractSeq:month:char):Contract
+getUnexpiredContract(o:ContractSeq,ctime:POSIXct,n:int):char
+getUnexpiredExpectedExpiry(o:ContractSeq,ctime:POSIXct,n:int)
+getIntraIntervalDataIndex(o:ContractSeq,interval:int):int
+getIntraIntervalData(o:ContractSeq,interval:int):xts
+getLastQuoteDatePosix(o:ContractSeq)
+setRefContIntradaySerie(o:ContractSeq,rollon:char)
+getRefContserie(o:ContractSeq,freq:int)
+getGluedsequence(o:ContractSeq:stime:POSIXct,etime:POSIXct,..
+getGluedserie(o:ContractSeq:stime:POSIXct,etime:POSIXct,..):xts
--misc funcs
+dtnConf(env:char)
+mkContractChain(yrstart:char,yrend:char,..):vector
+cshelp(o:ContractSeq)
+show(o:ContractSeq)
+showcur(o:ContractSeq,ctime:POSIXct)
-- check funcs
+printLiquidity(o:ContractSeq,ctime:POSIXct,n:int)
+chkprintEODroll(o:ContractSeq,ctime:POSIXct,n:int)
+chkMeta(o:ContractSEq)
+chkData(o:ContractSEq)
+sanityCheckMeta(o:ContractSeq)
-- config mgmt funcs
+findRollonvol(o:ContractSeq,ctime:POSIXct,..)
+fillExpiries(o:ContractSeq,kkeys:char)
-- data mgmt funcs
+addFutureContracts(o:ContractSeq,n:int)
+updateMeta(o:ContractSeq,ctime:POSIXct,persist:bool)
+updateRollonvol(o:ContractSeq,ctime:POSIXct)
+updateDataExpiries(o:ContractSeq)
+fixMetafromcsv(o:ContractSeq,persist:bool)
+iniDownload(o:ContractSeq,lstqtime:POSIXct,..)
+iniDownloadIntra(o:ContractSeq,lstqtime:POSIXct,..)
+loadIntraday(o:ContractSeq,interval:int,..)
+updateEOD(o:ContractSeq,scontract:char,ctime:POSIXct,..)
+updateIntra(o:ContractSeq,scontract:char,lstqtime:POSIXct,..)
+toPrdIntra(o:ContractSeq,sinerval:int,tointerval:int)
+allPrdIntra(o:ContractSeq,contracts:vector)
data
n
Struct
{element of data}
+ticker:char
+month:char
+expiry:char
+d1start:char
+d1end:char
+bd1:xts
intrdata
n
Struct
{element of intrdata}
+ticker:char
+month:char
+idat:list{
+freq:char
+dtstart:char
+dtstop:char
+xdat:xts
}
intrdcon
n
Struct v.2
{element of intrdconf}
+intradayfreq:vector(numeri
+daydoublevolume:bool
+outcrytimezone:char
+tick_size:double
+tick_price:double
+imnt_currency:char


Wstorybook
-curwstoryid:int
wstories:list
+addWstory(story:Wstory)
+rmWstory(instName:char)
+updateWstory(story:Wstory)
+getStory(instName:char):Wstory
+globalreplceWylbl(oldsl:char,newsl:char,newll:char
..
+show()
wstories
n
Wstory
-instr:char
facts:list
+getSinstrument():char
+getFact(index:int):Artifact
+addFact(fact:Artifact)
+delFact(index:int)
..
+printStory(timeFrame:char,lflag:bool,ndays:int)
+show()
n
facts
«Artifact»
Package::mtrlchart
{base class for facts}
-instr:char
-typeStr:char
-slabel:char
-llabel:char
-dunit:char
-dfrequency:int
+appendStory()
+show()
Wfact
yvalue:int
label2char:char
setYval()
appendStory()
LevelFact
appendStory()
TrendFact
appendStory()
Bvfact
{bar view}
appendStory()
Bpfact
{bar property}
appendStory()
Nfact
{deprecate}
appendStory()
Hfact
{hypo}
appendStory()
«Wwatchlist»
Package::mtrlchart
{principal operate object}
-curwstoryid:int
-curtimeframe:char
-dsubset:int
-wsubset:int
-hsubset:int
-rdatfile:char
-wdict:Wdict
-wstories:list(depr)
-mdata:Mldata
-wgconf:Wgconf
-storybook:Wstorybook
-orderbook:Orderbook
--constr
+wwlist(instrvec:vector):Wwatchlist
+wwlistfromdata(mdata:Mldata):Wwatchlist
+lwl(rdatfile:char):Wwatchlist
--getters setters
+getCurinstrument(l:Wwatchlist)
+getCurstory(l:Wwatchlist)
+{get/set}Curtimeframe(l:Wwatchlist)
+{get/set}Subset(l:Wwatchlist,tframe:char)
+{get/set}Wdict(l:Wwatchist):Wdict
+get/set}Graphconf(l:Wwatchist)Wgconf
+{get/set}Storybook(l:Wwatchist):Wstorybook
+{get/set}Orderbook(l:Wwatchist):Orderbook
+{get/set}Marketdata(l:Wwatchist):Mldata
+{get/set}Rdatfile((l:Wwatchist):char
--misc funcs
+addWatchGroup(l:Wwatchlist,ngrp:char,pos:int)
+addWatchInstrument(l:Wwatchlist,name:char,..)
+rmWatchInstrument(l:Wwatchlist,name:char)
+bulkreplaceWyklbl(l:Wwatchlist,..)
+persist(l:Wwatchlist)
--help funcs
+whelp(l:Wwatchlist)
+ohelp(l:Wwatchlist)
+show(l:Wwatchlist)
«Wwatchlist»
Package::mtrlchart
{WwatchlistCharting}
--misc funcs
+flipFctlbl(o:Wwatchlis,persist:bool)
+flipbg(o:Wwatchslist,persist:bool)
+ugpar(o:Wwatchslist,persist:bool)
+dngpar(o:Wwatchslist,persist:bool)
+inigpar(o:Wwatchslist,persist:bool)
--charting funcs
+cc(o:Wwatchlist,tf:char,nof:bool,pl:bool,..)
+ccd(o:Wwatchlist,nof:bool,pl:bool,..)
+ccw(o:Wwatchlist,nof:bool,pl:bool,..)
+ccc(o:Wwatchlist,pl:bool,plf:bool,..)
+cci(o:Wwatchlist,n:int, wave:bool,..)
+ccv(o:Wwatchlist,tl:float,th:float,..)
--help funcs
+crthelp(l:Wwatchlist)
«Wwatchlist»
Package::mtrlchart
{WwatchlistUserops}
-market data related funcs
+stfw1(o:Wwatchlist)
+stdw1(o:Wwatchlist)
+stfh1(o:Wwatchlist)
+gnd(o:Wwatchlist,f:bool,n:int,sav:bool)
+gndc(o:Wwatchlist,itrf:bool)
+gnw(o:Wwatchlist,itrf:bool)
+gndcs(o:Wwatchlist,itrf:bool,rollon:char,intrd:char,..)
+gndfcs(o:Wwatchlist,cslist:ContractSeqList,..)
+dmode(o:Wwatchlist,mode:char,startdate:char,enddate:char)
+dcdate(o:Wwatchlist, date:char)
-instrument related funcs
+gif(o:Wwatchlist)
+gin(o:Wwatchlist,rechart:bool)
+gip(o:Wwatchlist)
+gis(o:Wwatchlist,instr:char)
+pd(o:Wwatchlist,tf:char,n:int,all:bool)
+pmeta(o:Wwatchlist,all:bool,n:int,ctime:char)
+pliquid(o:Wwatchlist,ctime:char,n:int,topdir:char)
+rmdTail(o:Wwatchlist,n:int,tf:char)
-story,report related funcs
+ps(o:Wwatchlist,tf:char,lf:bool,n:int,uhist:bool,ct:bool,tex:bool,sig:bool)
+psc(o:Wwatchlist,lf:bool,nd:int,nw:int,uhist:bool)
+pst(o:Wwatchlist,uhist:bool)
-Wyckoff facts related funcs
+iSelectfact(o:Wwatchlist)
+pWfct(o:Wwatchlist,slb:char,cand:bool,cbt:bool,cbb:boo,sav:bool)
+swLbl(o:Wwatchlist,slbl:char,dtime:char,sav:bool)
+uWfct(o:Wwatchlist,slbo:char,slbn:char,dtime:char,cand:bool,sav:bool)
+upWfct(o:Wwatchlist,slbn:char,cand:bool,sav:bool)
+pCfct(o:Wwatchlist,type:char,cont:char,dtime:char,cbt:bool,cbb:bool)
+pVfct(o:Wwatchlist,cont:char,dtime:char,cbt:bool,cbb:bool,sav:bool)
+pBfct(o:Wwatchlist,cont:char,dtime:char,cbt:bool,cbb:bool,sav:bool)
+pHfct(o:Wwatchlist,cont:char,dtime:char,cbt:bool,cbb:bool,sav:bool)
+pAfct(o:Wwatchlist,wc:char,cand:bool,bc:char,vc:char,hc:char,dtime:char,..)
+pNob(o:Wwatchlist,cont:char,dtime:char,cbt:bool,cbb:bool,sav:bool)
+apndFct(o:Wwatchlist, acont:char,type:char,dtime:char,sav:bool)
+pLvl(o:Wwatchlist,stopdate:char,sav:bool)
+pIcl(o:Wwatchlist,stopdate:char,sav:bool)
+pHfl(o:Wwatchlist,sav:bool)
+pTrl(o:Wwatchlist,sav:bool)
+pChn(o:Wwatchlist,sav:bool)
+pLepr(o:Wwatchlist,putl:bool,sav:bool)
+upHfl(o:Wwatchlist,sav:bool)
+rmFact(o:Wwatchlist,slbl:char,dtime:char, ctf:char,f:bool,sav:bool)
+rmpFact(o:Wwatchlist,sav:bool)
-trades related funcs
+pptr(o:Wwatchlist,ftarg:float,pin:float,pstop:float,riskpt:float,capital:int,mult:1,..)
--help funcs
+wldhelp(l:Wwatchlist)
+wfhelp(l:Wwatchlist)
«Wwatchlist»
Package::mtrlchart
{Orderops}
--trade analytics funcs
+ck4tr(o:Wwatchlist,clb:bool,cbs:bool,arby:bool,pent:float,pstop:float,ft:float,mt:float,..)
+ptr(o:Wwatchlist,sdate:char,n:int,iall:bool,otn:bool,ct:bool)
+portf(o:Wwatchlist,sdate:char,tf:char)
+pctr(o:Wwatchlist)
+calcPortfolioNav(o:Wwatchlist,sdate:char,tf:char)
--trade market data processing funcs
+cfotr(o:Wwatchlist, sav:bool)
+gnbfo(o:Wwatchlist,tf:char,single:bool,sav:bool)
+lpd(o:Wwatchlist,tf:char,dtype:char,gfirst:bool,wt:int,mwt:float,osunix:bool)
--order management funcs
+rmatr(o:Wwatchlist,tid:int,sav:bool)
+cantr(o:Wwatchlist,tid:int,sav:bool)
+optr(o:Wwatchlist,cmnt:char,qty:int,mkto:bool,plstop:bool,verbose:bool,sav:bool)
+adjsoptr(o:Wwatchlist,tid:int,sav:bool)
+addOrder(o:Wwatchlist,kind:char,tid:int,qty:int,pr:float,sav:bool)
+rmotr(o:Wwatchlist,oid:int,sav:bool)
+canotr(o:Wwatchlist,oid:in,sav:bool)
+astoptr(o:Wwatchlist,tid:int,qty:int,pr:float,sav:bool)
+atartr(o:Wwatchlist,tid:int,qty:int,pr:float,sav:bool)
+aaaotr(o:Wwatchlist,tid:int,type:char,qty:int,pr:float,sav:bool)
+amotr(o:Wwatchlist,tid:int,qty:int,sav:bool)
+alotr(o:Wwatchlist,tid:int,qty:int,pr:float,sav:bool)
+moprtr(o:Wwatchlist,oid:int,pr:float,sav:bool)
+moqtytr(o:Wwatchlist,oid:int,qty:int,sav:bool)
+fixotr(o:Wwatchlist,oid:int,oncl:bool,sav:bool)
Tmptrade
1
«Orderbook»
Package::mtrlchart
{functional area object}
-pfpars:list
-ometadat:data.frame
-Tmptrade:Trade
-oidcur:int
-oidseed:int
-tidcur:int
-tidseed:int
-odf:data.frame
-tdf:data.frame
--constr
+initOrderbook(pparams:list,instrvec:vec,typevec:vec,...)
+initOrderbookcsv(pparams:list,csvfile:char)
--getters setters
+getNextIdentity(o:Orderbook,ivar:char):int
+storeNextIdentity(o:Orderbook,ivar:char)
+getTemtrade(o:Orderbook):Trade
+{get/set}ActiveInstruments(o:Orderbook)
+{get/set}WatchInstruments(o:Orderbook)
--misc funcs
+initPorfolioParams(accid:int,stratid:int,pname:char,..):list
+initOrderbookBlotter(o:Orderbook,tickers:vec)
+getInstrmetinfo(o:Orderbook,ticker:char,mtype:char)
+getTradesSubset(o:Orderbook, ticker:char, sdate:char,edate:char,crit:char)
+getOrdersSubset(o:Orderbook,ticker:char, sdate:char,edate:char,trcit:char,ocrit:char)
--Order operations
+insOrderRec(o:Orderbook,order:Order)
+updOrderRec(o:Orderbook,order:Order)
+rmOrderRec(o:Orderbook,id:int)
+getOrderRec(o:Orderbook,id:int)
+getOrderFromRec(o:Orderbook,id:int)
+eraseOrder(o:Orderbook,oid:int)
--trade operations
+insTradeRec(o:Orderbook,trade:Trade)
+updTradeRec(o:Orderbook,trade:Trade,cascade:bool)
+rmTradeRec(o:Orderbook,tid:int)
+getTradeRec(o:Orderbook,tid:int)
+getTradeFromRec(o:Orderbook,tid:int)
+eraseTrade(o:Orderbook,tid:int)
--biz functions
+check4trade(o:Orderbook,instr:char,incdate:char,pent:float,pstop:float,ftarget:float,mtarget:floa
+openTrade(o:Orderbook,mtype:char,incdate:char,qty:int,entry:float,comment:char,plstop:bool)
+cancelTrade(o:Orderbook,tid:int,chgdate:char)
+updOrdersTradePosition(o:Orderbook,tid:int,chgdate:char,verbose:bool)
+addTradeOrder(o:Orderbook,tid:int,incdate:char,mtype:char,qty:int,price:float)
+updTradeOrder(o:Orderbook,oid:int, chgdate:char,param:char,value:float)
+procOrderFills(o:Orderbook,ticker:char,cdate:char,data:xts,verbose:bool)
+printTrades(o:Orderbook,ticker:char,sdate:char,edate:char,mmprice:float,n:int,..)
+printInstrGlpos(o:Orderbook,ticker:char,sdate:char,mmprice:float,edate:char)
+calcInstrumentPosition(o:Orderbook,ticker:char, sdate:char,edate:char)
+calcInstrumentPnl(o:Orderbook, ticker:char,mmprice:float,sdate:char,edate:char,ptype:char)
+calcPortfolioEquity(o:Orderbook,sdate:char,edate:char)
+calcPortfolioCash(o:Orderbook, sdate:char,edate:char)
+eqtyAlloc4instrument(o:Orderbook, ticker:char,sdate:char,edate:char)
+blotterPnl(o:Orderbook,ticker:char,xdata:xts,sdate:char,edate:char,init:bool,chart:bool)
--help funcs
+crthelp(l:Wwatchlist)
Struct
{pfpars list}
account : num
straregy : num
name : char
iniDate : char
iniEquity: num
riskpct : num
activeinst : char
watchinst : char
Trade
package:mMtrlchart
-tid:int
-stratid:int
-accid:int
-itkr:char
-inceptDate:POSI
-tstatus:char
-tqty:int
-entry:float
-stop:float
-ftarget:float
-mtarget:float
-comment:char
-torders :list
--constr
+mkDummyTrade(ticker:char, incdate:POSI,entry:float,stop:flo
+mkTrade(strat:int,acc:int,tkr:char,incDate:POSI, qty:int, entry
+crTradeFromrec(tlist:list)
--getters setters
+{get/set}Tid(o:Trade,value:int)
+{get/set}Stratid(o:Trade,value:int)
+{get/set}Taccount(o:Trade,value:char)
+{get/set}Tticker(o:Trade,value:char)
+{get/set}Tstatus(o:Trade,value:char)
+{get/set}TincDate(o:Trade,value:POSI)
+{get/set}Tqty(o:Trade,value:int)
+{get/set}Entry(o:Trade,value:float)
+{get/set}Stop(o:Trade,value:float)
+{get/set}Ftarget(o:Trade,value:float)
+{get/set}Mtarget(o:Trade,value:float)
+{get/set}Comment(o:Trade,value:char)
+{get/set}Torders(o:Trade,value:list)
+{get/set}TradeOrder(o:Trade,value:Order)
+getOrdersattachedIds(o:Trade)
--operate methods
+removeTradeOrder(o:Trade,order:Order)
+updTradeStatus(o:Trade)
+updOrdersizeOpenposition(o:Trade,chgdate:POSI)
+getFtargetRR(o:Trade)
+getMtargetRR(o:Trade)
+hasOrdersattached(o:Trade)
+getTradePosition(o:Trade)
+getTradeMtm(o:Trade,mmprice:float)
+getTradeMtmvalue(o:Trade, mmprice:float, contrmult:int)
+getTradeUpnl(o:Trade, mmprice:float, contrmult:int)
+getTradeRpnl(o:Trade, mmprice:float, contrmult:int)
+getTradeLockedCash(o:Trade, mmprice:float, marginpct:int)
+getTradeAvrgPrice(o:Trade, crit:char)
+isTradeProtected(o:Trade, crit:int)
+findTradeOrphanedOrders(o:Trade)
+strTradeOrphanedOrders(o:Trade)
+getTradeWorkingOrders(o:Trade)
+getTradeOrdersStatus(o:Trade)
+attachOrdersFromrecset(o:Trade, rset:data.frame)
+setOrdersTrid(o:Trade)
+initTradedataframe(o:Trade)
list of {Order}
1
Order
package:mMtrlchart
-id:int
-trid:int
-accid:int
-itkr:char
-inceptDate:POSI
-type:char
-status:char
-changeDate:POSI
-qty:int
-rprice:float
-fprice:float
--constr
+mkLimitOrder(tradeid:int,acc:int,tkr:char,incDate:POSI,qty:in
+mMarketOrder(tradeid:int,acc:int,tkr:char,incDate:POSI,qty:i
+crOrderFromrec(olist:list)
--getters setters
+{get/set}Id(o:Order,value:int)
+{get/set}Trid(o:Order,value:int)
+{get/set}Type(o:Order,value:char)
+{get/set}Status(o:Order,value:char)
+getTicker(o:Order)
+getOaccount(o:Order)
+getIncDate(o:Order,chr:bool)
+{get/set}ChgDate(o:Order,value:POSI)
+{get/set}Qty(o:Order,value:int)
+{get/set}Regprice(o:Order,value:float)
+{get/set}Fillprice(o:Order,value:float)
+getMtm(o:Order,mmprice:float)
+getNotional(o:Order)
--operate methods
+cancelOrder(o:Order,chdate:POSI)
+fillOrder(o:Order,chdate:POSI,fillprice:float)
+checkFillOrder(o:Order, data:xts,cdate:POSI,slippage:float)
tdf
n
Struct
{trade data frame}
tid : num
strat : num
acc : num
tkr : char
incdate: POSI
status : char
tqty : num
entry : num
stop : num
ftarget: num
mtarget: num
comment: chr
ometadata
1
Struct
{book meta data frame}
IDNAM : chr
TYP : chr
CURR : chr
TKSIZ : num
TKVAL : num
UFEE : num
UINIMARG: num
TKSLIPP : num
RWGHT : num
odf
n
Struct
{order data frame}
id : num
trid : num
acc : num
tkr : char
incdate: POSI