UI 界面显示代码 Objective-C 和C++

//

// dataTableView.h

// CPK\_Test

//

// Created by RyanGao on 2020/6/25.

// Copyright © 2020 RyanGao. All rights reserved.

//

#import <Cocoa/Cocoa.h>

#import "csvListController.h"

#import "../StartUp.framework/Headers/StartUp.h"

NS\_ASSUME\_NONNULL\_BEGIN

@interface dataTableView : NSViewController<NSApplicationDelegate, NSTableViewDataSource, NSTableViewDelegate,NSSplitViewDelegate>

{

csvListController \*csvView;

IBOutlet NSView \*csvViewMain;

IBOutlet NSView \*leftViewMain;

StartUp \* startPython;

}

@property (strong) IBOutlet NSView \*viewWindow;

- (IBAction)btLoadCsvData:(id)sender;

- (IBAction)btnSearchCsv:(id)sender;

//@property (weak) IBOutlet NSImageView \*cpkImageMap;

//@property (weak) IBOutlet NSImageView \*correlationImageMap;

@property (weak) IBOutlet NSView \*leftPane;

@property (weak) IBOutlet NSView \*rightPanel;

@property (weak) IBOutlet NSSplitView \*splitView;

@property (weak) IBOutlet NSTextField \*txtTestCount;

@property (weak) IBOutlet NSTextField \*txtPass;

@property (weak) IBOutlet NSTextField \*txtFail;

@property (weak) IBOutlet NSTextField \*txtYieldP;

@property (weak) IBOutlet NSTextField \*txtRetestR;

@property (weak) IBOutlet NSTextField \*txtRetestC;

@property (weak) IBOutlet NSTableCellView \*itemCellView;

@property (weak) IBOutlet NSTextField \*txtTotalC;

@property (strong) IBOutlet NSSearchField \*txtSearch;

@end

NS\_ASSUME\_NONNULL\_END

//

// dataTableView.m

// CPK\_Test

//

// Created by RyanGao on 2020/6/25.

// Copyright © 2020 RyanGao. All rights reserved.

//

#import "dataTableView.h"

#import "../SCparseCSV.framework/Headers/parseCSV.h"

#import "../SCNSEventEx.framework/Headers/NSEventEx.h"

#import "../SCRedis.framework/Headers/RedisInterface.hpp"

#import "dataPlotView.h"

#import "defineHeader.h"

#import "../SCZmq.framework/Headers/Client.h"

//#import "../XlsxReaderWriter.framework/Headers/BRAOfficeDocumentPackage.h"

#import "../SCopensslSha1.framework/Headers/sha.h"

#import "loadCsvControl.h"

#import <Quartz/Quartz.h>

#import <QuartzCore/QuartzCore.h>

//#import "AppConstants.h"

//#import "AppUtils.h"

extern NSMutableDictionary \*m\_configDictionary;

extern NSInteger tbDataTableSelectItemRow;

extern NSMutableArray \*\_dataReverse;

extern NSMutableArray \*\_rawData;

extern int selectColorBoxIndex; //left color by

extern int selectColorBoxIndex2;//right color by

extern RedisInterface \*myRedis;

extern Client \*cpkClient;

extern Client \*correlationClient;

extern Client \*reportHashClient;//hash

extern Client \*scatterClient;

Client \*retestPlotClient;

Client \*calculateClient;

Client \*retestRateClient;

int n\_Start\_Data\_Col;

int n\_Pass\_Fail\_Status;

int n\_Product\_Col;

int n\_SerialNumber;

int n\_SpecialBuildName\_Col;

int n\_Special\_Build\_Descrip\_Col;

int n\_StationID\_Col;

int n\_StartTime;

int n\_Version\_Col;

int n\_Diags\_Version\_Col;

int n\_OS\_VERSION\_Col;

int n\_passdata;

/\*

#define Start\_Data\_Row 7

#define Start\_Data\_Col 11

#define Pass\_Fail\_Status 7

#define Product\_Col 1

#define SerialNumber 2

#define SpecialBuildName\_Col 3

#define Special\_Build\_Descrip\_Col 4

#define StationID\_Col 6

#define StartTime 8

#define Version\_Col 10

\*/

@interface dataTableView ()

{

BOOL enableEditing;

NSUInteger clickItemIndex;

NSUInteger editLimitRow;

NSString \*retestValue; //记录上次结果

NSString \*removeValue; //记录上次结果

NSString \*desktopPath;

int select\_btn\_x;

int select\_btn\_y;

int click\_item\_flag;

NSMutableArray \*arrSearch;

NSMutableArray \*arrSearchRed;

NSMutableArray \*arrSearchGreen;

NSMutableArray \*arrSearchRedCPK;

NSMutableArray \*arrSearchGreenCPK;

NSMutableArray \*arrSearchYellowCPK;

int n\_search;

CGFloat \_lastLeftPaneWidth;

int n\_loadCsvBtn;

int n\_sort\_col1;

int n\_sort\_col5;

int n\_sort\_col9; //reviwer

int n\_sort\_col12; //bmc

NSMutableArray \* tmpColorArr;

NSInteger click\_tb\_row;

NSInteger n\_reviewer\_col;

NSInteger n\_double\_click;

NSMutableArray \*hash\_value;

BOOL b\_ClearComment;

NSMutableString \*inputCharacter;

int n\_firstItemClick;

NSInteger n\_clickApplyRow;

}

@property (weak) IBOutlet NSTableView \*dataTableView;

@property (nonatomic,strong)NSMutableArray \*data;

@property (nonatomic,strong) NSMutableArray \*dataBackup;

@property (nonatomic,strong)NSMutableArray \*scriptData;

@property (nonatomic,strong)NSMutableArray \*limitUpdateData;

@property (nonatomic,strong)NSMutableArray \*sortDataBackup;

//@property (nonatomic,strong)NSMutableArray \*rawData;

@property (nonatomic,strong)NSMutableDictionary \*indexItemNameDic;

@property (nonatomic,strong)NSMutableArray \*ListAllItemNameArr;

@property (nonatomic,strong)NSMutableDictionary \*textEditLimitDic;

@property (nonatomic,strong)NSMutableArray \*colorRedIndex; //不相同的item，后面追加的数据，显示红色

@property (nonatomic,strong)NSMutableArray \*colorGreenIndex; //相同的item，显示绿色

@property (nonatomic,strong)NSMutableArray \*colorRedIndexBackup; //不相同的item，后面追加的数据，显示红色

@property (nonatomic,strong)NSMutableArray \*colorGreenIndexBackup; //相同的item，显示绿色

@property (nonatomic,strong)NSMutableArray \*colorRedIndexSearchBackup; //搜索的时候，颜色备份

@property (nonatomic,strong)NSMutableArray \*colorGreenIndexSearchBackup; //搜素的时候，颜色备份

@property (nonatomic,strong)NSMutableArray \*colorRedIndexCpk; //cpk颜色

@property (nonatomic,strong)NSMutableArray \*colorGreenIndexCpk; //cpk颜色

@property (nonatomic,strong)NSMutableArray \*colorYellowIndexCpk; //cpk颜色

@property (nonatomic,strong)NSMutableArray \*colorRedIndexSearchCpk; //cpk颜色

@property (nonatomic,strong)NSMutableArray \*colorGreenIndexSearchCpk; //cpk颜色

@property (nonatomic,strong)NSMutableArray \*colorYellowIndexSearchCpk; //cpk颜色

@property (nonatomic,strong)NSMutableArray \*colorRedIndexCpkBackup; //cpk red颜色备份

@property (nonatomic,strong)NSMutableArray \*colorGreenIndexCpkBackup; //cpk green颜色备份

@property (nonatomic,strong)NSMutableArray \*colorYellowIndexCpkBackup; //cpk yellow颜色备份

@property (nonatomic,strong)NSMutableArray \*reviewerNameIndex; //reviewer name

@property (nonatomic,strong)NSMutableArray \*reviewerNameIndexBackup; //reviewer name 备份

@property (nonatomic,strong)NSMutableArray \*bmcYesIndex; //bmc YES

@property (nonatomic,strong)NSMutableArray \*bmcNoIndex; //bmc YES

@property (weak) IBOutlet NSTextField \*txtcsvDataName;

@property (weak) IBOutlet NSTextField \*txtScriptName;

@property (weak) IBOutlet NSTextField \*txtLimitUpdate;

@property loadCsvControl \*modalCsvController;

@end

@implementation dataTableView

-(instancetype)init

{

self = [super init];

if (self)

{

\_data = [[NSMutableArray alloc]init];

\_scriptData = [[NSMutableArray alloc]init];

\_limitUpdateData = [[NSMutableArray alloc]init];

//\_dataReverse = [[NSMutableArray alloc]init];

//\_rawData = [[NSMutableArray alloc]init];

\_indexItemNameDic = [[NSMutableDictionary alloc] init];

\_textEditLimitDic = [[NSMutableDictionary alloc] init];

\_ListAllItemNameArr = [[NSMutableArray alloc]init];

hash\_value = [[NSMutableArray alloc]init];

enableEditing = YES;

clickItemIndex = -1;

editLimitRow = -1;

retestValue=@"";

removeValue = @"";

select\_btn\_x = 0;

select\_btn\_y = 0;

click\_item\_flag = 0;

arrSearch = [[NSMutableArray alloc]init];

arrSearchRed = [[NSMutableArray alloc]init];

arrSearchGreen = [[NSMutableArray alloc]init];

arrSearchRedCPK = [[NSMutableArray alloc]init];

arrSearchGreenCPK = [[NSMutableArray alloc]init];

arrSearchYellowCPK = [[NSMutableArray alloc]init];

\_dataBackup = [[NSMutableArray alloc]init];

n\_search = 0;

n\_loadCsvBtn = 0;

n\_sort\_col1 = 0;

n\_sort\_col5 = 0;

n\_sort\_col9 = 0;

n\_sort\_col12 = 0;

\_sortDataBackup = [[NSMutableArray alloc]init];

n\_Start\_Data\_Col = -1;

n\_Pass\_Fail\_Status = -1;

n\_Product\_Col =-1;

n\_SerialNumber = -1;

n\_SpecialBuildName\_Col = -1;

n\_Special\_Build\_Descrip\_Col =-1;

n\_StationID\_Col =-1;

n\_StartTime = -1;

n\_Version\_Col =-1;

n\_Diags\_Version\_Col = -1;

n\_OS\_VERSION\_Col = -1;

click\_tb\_row = 0;

n\_reviewer\_col = 0;

n\_double\_click = 0;

n\_firstItemClick = 0;

n\_passdata = 0;

\_colorRedIndex = [[NSMutableArray alloc]init];

\_colorGreenIndex = [[NSMutableArray alloc]init];

\_colorRedIndexBackup = [[NSMutableArray alloc]init];

\_colorGreenIndexBackup = [[NSMutableArray alloc]init];

\_colorRedIndexSearchBackup =[[NSMutableArray alloc]init];

\_colorGreenIndexSearchBackup = [[NSMutableArray alloc]init];

\_colorRedIndexCpk = [[NSMutableArray alloc]init];

\_colorGreenIndexCpk = [[NSMutableArray alloc]init];

\_colorYellowIndexCpk = [[NSMutableArray alloc]init];

\_colorRedIndexSearchCpk = [[NSMutableArray alloc]init];

\_colorGreenIndexSearchCpk = [[NSMutableArray alloc]init];

\_colorYellowIndexSearchCpk = [[NSMutableArray alloc]init];

\_colorRedIndexCpkBackup = [[NSMutableArray alloc]init];

\_colorGreenIndexCpkBackup = [[NSMutableArray alloc]init];

\_colorYellowIndexCpkBackup = [[NSMutableArray alloc]init];

\_reviewerNameIndex = [[NSMutableArray alloc]init];

\_bmcYesIndex = [[NSMutableArray alloc]init];

\_bmcNoIndex = [[NSMutableArray alloc]init];

tmpColorArr = [[NSMutableArray alloc]init];

b\_ClearComment = YES;

inputCharacter = [[NSMutableString alloc] init];

n\_clickApplyRow = -1;

}

return self;

}

- (void)viewDidLoad

{

[\_dataTableView setDelegate:self];

[\_dataTableView setDataSource:self];

[self.dataTableView reloadData];

[self.dataTableView setTarget:self];

[self.dataTableView setDoubleAction:@selector(DblClickOnTableViewDouble:)];

[self.dataTableView setAction:@selector(DblClickOnTableView:)];

[NSEvent addLocalMonitorForEventsMatchingMask:NSEventMaskKeyDown handler:^NSEvent \* \_Nullable(NSEvent \* \_Nonnull aEvent) {

[self keyDown:aEvent];

return aEvent;

}];

desktopPath = [NSSearchPathForDirectoriesInDomains(NSDesktopDirectory, NSUserDomainMask, YES)objectAtIndex:0];

NSString \*logPath = [NSString stringWithFormat:@"%@/CPK\_Log",desktopPath];

[self createFileDirectories:logPath];

NSString \*logPath2 = @"/tmp/CPK\_Log";

[self createFileDirectories:logPath2];

NSString \*failPlot = [NSString stringWithFormat:@"%@/fail\_plot",logPath2];

[self createFileDirectories:failPlot];

NSString \*retestPlot = [NSString stringWithFormat:@"%@/retest",logPath2];

[self createFileDirectories:retestPlot];

NSString \*plot = [NSString stringWithFormat:@"%@/plot",logPath2];

[self createFileDirectories:plot];

NSString \*temp = [NSString stringWithFormat:@"%@/temp",logPath2];

[self createFileDirectories:temp];

[@"none" writeToFile:[NSString stringWithFormat:@"%@/temp/.logcpk.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:[NSString stringWithFormat:@"%@/temp/.logcor.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:[NSString stringWithFormat:@"%@/temp/.logscatter.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:[NSString stringWithFormat:@"%@/temp/.logcalc.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:[NSString stringWithFormat:@"%@/temp/.logretest.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:[NSString stringWithFormat:@"%@/temp/.excel.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:[NSString stringWithFormat:@"%@/temp/.excel\_hash.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:[NSString stringWithFormat:@"%@/temp/.keynote.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:[NSString stringWithFormat:@"%@/temp/.reporttags.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"" writeToFile:[NSString stringWithFormat:@"%@/temp/.recordSelctItem.csv",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"" writeToFile:[NSString stringWithFormat:@"%@/temp/.cpknew.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"" writeToFile:[NSString stringWithFormat:@"%@/retest/.retest\_plot.txt",logPath2] atomically:YES encoding:NSUTF8StringEncoding error:nil];

NSFileManager \*manager = [NSFileManager defaultManager];

NSString \*nonpicPath =[[NSBundle mainBundle]pathForResource:@"none\_pic.png" ofType:nil];

[manager copyItemAtPath:nonpicPath toPath:@"/tmp/CPK\_Log/retest/.none\_pic.png" error:nil];

NSString \*appl\_picPath =[[NSBundle mainBundle]pathForResource:@"apple\_log\_black.png" ofType:nil];

[manager copyItemAtPath:appl\_picPath toPath:@"/tmp/CPK\_Log/retest/.apple\_log\_black.png" error:nil];

// [@"none" writeToFile:[NSString stringWithFormat:@"%@/CPK\_Log/temp/.logparam.txt",desktopPath] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[self.txtcsvDataName setStringValue:@""];

[self.txtScriptName setStringValue:@""];

[self.txtLimitUpdate setStringValue:@""];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(settingTableViewData:) name:kNotificationSetColorByLeft object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(settingTableViewData:) name:kNotificationSetColorByRight object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(settingTableViewData:) name:kNotificationSelectX object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(settingTableViewData:) name:kNotificationSelectY object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(settingTableViewData:) name:kNotificationSetParameters object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(settingTableViewData:) name:kNotificationSaveUIdata object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(settingTableViewData:) name:kNotificationSetCpkNew object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(toLoadCsv:) name:kNotificationToLoadCsv object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(toLoadLocalCsv:) name:kNotificationToLocalLoadCsv object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(clickOneItem:) name:kNotificationClickOneItem object:nil];

//

\_lastLeftPaneWidth = self.leftPane.frame.size.width;

csvView = [[csvListController alloc]init];

[self LoadSubView:csvView.view];

[self.splitView setPosition:0 ofDividerAtIndex:0];

startPython = [[StartUp alloc] init];

}

-(void)awakeFromNib

{

[self.splitView setPosition:0 ofDividerAtIndex:0];

}

-(void)settingTableViewData:(NSNotification \*)nf

{

NSString \* name = [nf name]; // set color by choose

if ([ name isEqualToString:kNotificationSetColorByLeft])

{

NSDictionary\* info = [nf userInfo];

int colorIndex = [[info valueForKey:select\_Color\_Box\_left] intValue];

NSInteger row = [self.dataTableView selectedRow];

if (row>=0)

{

\_data[row][tb\_color\_by\_left]=[NSNumber numberWithInt:colorIndex];

}

}

else if ([ name isEqualToString:kNotificationSetColorByRight])

{

NSDictionary\* info = [nf userInfo];

int colorIndex = [[info valueForKey:select\_Color\_Box\_Right] intValue];

NSInteger row = [self.dataTableView selectedRow];

if (row>=0)

{

\_data[row][tb\_color\_by\_right]=[NSNumber numberWithInt:colorIndex];

}

}

else if ([ name isEqualToString:kNotificationSelectX])

{

NSDictionary\* info = [nf userInfo];

int x = [[info valueForKey:btn\_select\_x] intValue];

NSInteger row = [self.dataTableView selectedRow];

if (row>=0)

{

\_data[row][button\_select\_x]=[NSNumber numberWithInt:x];

n\_firstItemClick = 1;

[self triggerGeneratePlot:row withApplyBox:YES withSelectXY:1];

}

//add recore select item for correlation

}

else if ([ name isEqualToString:kNotificationSetCpkNew])

{

NSDictionary\* info = [nf userInfo];

NSString \* ret = [info valueForKey:cpkNewNumber];

if (n\_clickApplyRow>=0)

{

\_data[n\_clickApplyRow][tb\_cpk\_new] =ret;

int n\_i = 0;

for (n\_i=0; n\_i<[[self.dataTableView tableColumns] count]; n\_i++)

{

if ([[[[self.dataTableView tableColumns] objectAtIndex:n\_i] identifier] isEqualTo:identifier\_cpknew]) {

break;

}

}

[self.dataTableView reloadDataForRowIndexes:[NSIndexSet indexSetWithIndex:n\_clickApplyRow] columnIndexes:[NSIndexSet indexSetWithIndex:n\_i]];

n\_clickApplyRow = -1;

}

}

else if ([ name isEqualToString:kNotificationSelectY])

{

NSDictionary\* info = [nf userInfo];

int y = [[info valueForKey:btn\_select\_y] intValue];

NSInteger row = [self.dataTableView selectedRow];

if (row>=0)

{

\_data[row][button\_select\_y]=[NSNumber numberWithInt:y];

n\_firstItemClick = 1;

[self triggerGeneratePlot:row withApplyBox:YES withSelectXY:10];

}

//add recore select item for correlation

}

else if ([ name isEqualToString:kNotificationSaveUIdata])

{

//NSMutableArray \*csvData = [NSMutableArray arrayWithArray:[self reverseArray:\_data]];

NSMutableString \*strCsv = [NSMutableString string];

[strCsv appendString:@"index,item,low,upper,new\_lsl,new\_usl,apply\n"];

for(NSMutableArray \*lineArray in \_data)

{

NSString \*arrString = [NSString stringWithFormat:@"%@,%@,%@,%@,%@,%@,%@\n",lineArray[0],lineArray[1],lineArray[5],lineArray[4],lineArray[7],lineArray[8],lineArray[9]];

[strCsv appendString:arrString];

}

NSString \*csv\_Path = @"/tmp/CPK\_Log/temp/item\_limit.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/item\_limit.csv",desktopPath];

[strCsv writeToFile:csv\_Path atomically:YES encoding:NSUTF8StringEncoding error:nil];

}

else if ([ name isEqualToString:kNotificationRetestRate])

{

/\* @try

{

NSFileManager \*fh\_csv = [NSFileManager defaultManager];

NSString \*pathRate = [NSString stringWithFormat:@"%@/CPK\_Log/temp/yield\_rate\_param.csv",desktopPath];

NSData \*data = [fh\_csv contentsAtPath:pathRate];

NSString \*str = [[NSString alloc] initWithData:data encoding:NSUTF8StringEncoding];

NSArray \*line = [str componentsSeparatedByString:@"\n"];

NSLog(@"--->yield\_rate\_param %@",line[1]);

if ([line count]>1)

{

NSArray \*lineArr = [line[1] componentsSeparatedByString:@","];

//test\_count,fail\_count,pass\_count,retest\_count,retest\_rate,yield\_percentage

[self.txtTestCount setStringValue:lineArr[0]];

[self.txtPass setStringValue:lineArr[2]];

[self.txtFail setStringValue:lineArr[1]];

[self.txtYieldP setStringValue:lineArr[5]];

[self.txtRetestR setStringValue:lineArr[4]];

[self.txtRetestC setStringValue:lineArr[3]];

[self.txtTotalC setStringValue:lineArr[6]];

}

}

@catch (NSException \*exception)

{

NSLog(@"-----update yiled retest faile");

}

\*/

}

else if ([ name isEqualToString:kNotificationSetParameters])

{

@try {

[\_bmcYesIndex removeAllObjects];

[\_bmcNoIndex removeAllObjects];

NSDictionary\* info = [nf userInfo];

NSString \*path = [info valueForKey:paramPath];

//NSLog(@"======>>>>>>:%@",path);

NSFileManager \*fh\_csv = [NSFileManager defaultManager];

NSData \*data = [fh\_csv contentsAtPath:path];

NSString \*str = [[NSString alloc] initWithData:data encoding:NSUTF8StringEncoding];

NSArray \*line = [str componentsSeparatedByString:@"\n"];

for (int i = 1; i< [line count]; i++)

{

NSArray \*lineArr = [line[i] componentsSeparatedByString:@","];

//NSLog(@"=====> count: %zd",[lineArr count]);

if ([lineArr count]>7)

{

for (int j = 0; j< [\_ListAllItemNameArr count]; j++)

{

if ([\_ListAllItemNameArr[j] isEqualToString:lineArr[0]])

{

// \_data[j][BC\_Col] = lineArr[1];

// \_data[j][p\_val\_Col] = lineArr[2];

// \_data[j][a\_q\_Cal] = lineArr[3];

// \_data[j][a\_irr\_Cal] = lineArr[4];

\_data[j][tb\_cpk\_orig] =lineArr[6] ;

\_data[j][tb\_bmc] =lineArr[7] ;

if ([lineArr[7] containsString:@"YES"])

{

[\_bmcYesIndex addObject:[NSNumber numberWithInt:j]];

}

if ([lineArr[7] containsString:@"NO"])

{

[\_bmcNoIndex addObject:[NSNumber numberWithInt:j]];

}

[self setCpkOrigColor:lineArr[6] withRow:j];

break;

}

}

}

}

//NSLog(@"--\_colorRedIndexCpk:%@, \_colorGreenIndexCpk:%@",\_colorRedIndexCpk,\_colorGreenIndexCpk);

[self.dataTableView reloadData];

}

@catch (NSException \*exception)

{

NSLog(@"-----update 3CV,a\_q,a\_irr error");

}

}

}

//-(void)readParameterCsv:(NSString \*)path

//{

// NSFileManager \*fh\_csv = [NSFileManager defaultManager];

// NSData \*data = [fh\_csv contentsAtPath:path];

// NSString \*str = [[NSString alloc] initWithData:data encoding:NSUTF8StringEncoding];

// NSArray \*line = [str componentsSeparatedByString:@"\n"];

//

//

// for (int i = 0; i< [line count]; i++)

// {

// NSArray \*lineArr = [line[i] componentsSeparatedByString:@","];

// NSString \* finnalStr=[line[i] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

//

// }

//}

-(void)setCpkOrigColor:(NSString \*)value withRow:(int)row

{

float cpkL = [[m\_configDictionary valueForKey:cpk\_Lowthl] floatValue];

float cpkH = [[m\_configDictionary valueForKey:cpk\_Highthl] floatValue];

if ([self isPureFloat:value]|| [self isPureInt:value])

{

if ([value floatValue]<cpkL)

{

[\_colorRedIndexCpk addObject:[NSNumber numberWithInt:row]];

[\_colorRedIndexCpkBackup addObject:[NSNumber numberWithInt:row]];

}

else if([value floatValue] >cpkH)

{

[\_colorYellowIndexCpk addObject:[NSNumber numberWithInt:row]];

[\_colorYellowIndexCpkBackup addObject:[NSNumber numberWithInt:row]];

}

else

{

[\_colorGreenIndexCpk addObject:[NSNumber numberWithInt:row]];

[\_colorGreenIndexCpkBackup addObject:[NSNumber numberWithInt:row]];

}

}

else

{

// NSLog(@"======white===");

}

}

- (void)createFileDirectories:(NSString \*)folderPath

{

// 判断文件夹是否存在，不存在则创建对应文件夹

NSFileManager \*fileManager = [NSFileManager defaultManager];

BOOL isExist = [fileManager fileExistsAtPath:folderPath];

if (isExist)

{

//NSLog(@"目录已经存在");

}

else

{

BOOL ret = [fileManager createDirectoryAtPath:folderPath withIntermediateDirectories:YES attributes:nil error:nil];

if (ret)

{

// NSLog(@"目录创建成功");

}

else

{

NSLog(@"目录创建失败");

return;

}

}

}

-(IBAction)DblClickOnTableViewDouble:(id)sender

{

NSInteger row = [self.dataTableView selectedRow];

if (row == -1)

{

return;

}

// [self.dataTableView setAccessibilityEnabled:YES];

n\_double\_click = row;

[self.dataTableView reloadDataForRowIndexes:[NSIndexSet indexSetWithIndex:row] columnIndexes:[NSIndexSet indexSetWithIndex:1]];

}

-(IBAction)DblClickOnTableView:(id)sender

{

[m\_configDictionary setValue:[NSNumber numberWithBool:NO] forKey:kInputRangeFlag];

NSInteger row = [self.dataTableView selectedRow];

if (row == -1)

{

[self btnSearchCsv:@""];

NSInteger col = [self.dataTableView selectedColumn];

NSString \*col\_identifier = [[[self.dataTableView tableColumns] objectAtIndex:col] identifier];

NSLog(@">>>>click select row: %zd double\_click %zd , col\_identifier: %@",row,col,col\_identifier);

if ([col\_identifier isEqualToString:identifier\_index])//==col select index

{

if ([\_colorGreenIndexBackup count]>0 || [\_colorRedIndexBackup count]>0)

{

if (n\_sort\_col9 %2 ==1 ||n\_sort\_col12 !=0) //还原

{

n\_sort\_col9 = 0;

n\_sort\_col12 = 0;

//[self AlertBox:@"Error" withInfo:@"Please Click Reviewer Name column to data back."];

[\_data removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorRedIndex removeAllObjects];

[\_colorRedIndexCpk removeAllObjects];

[\_colorGreenIndexCpk removeAllObjects];

[\_colorYellowIndexCpk removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_data setArray:\_sortDataBackup];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

if ([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:i]];

}

else if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:i]];

}

}

[tmpColorArr removeAllObjects];

[tmpColorArr setArray:\_data];

n\_sort\_col1 = 0;

n\_sort\_col5 = 0;

}

n\_sort\_col1 ++;

[\_data removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorRedIndex removeAllObjects];

[\_colorRedIndexCpk removeAllObjects];

[\_colorGreenIndexCpk removeAllObjects];

[\_colorYellowIndexCpk removeAllObjects];

if (n\_sort\_col1%3 ==1) //绿色排序显示在前面

{

NSMutableArray \*sort = [NSMutableArray array];

int i;

for (i=0; i<[\_colorGreenIndexBackup count]; i++)

{

int index = [\_colorGreenIndexBackup[i] intValue];

[sort addObject:\_sortDataBackup[index]];

[\_colorGreenIndex addObject:[NSNumber numberWithInt:i]];

}

[\_data setArray:sort];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

for (int i=0; i<[\_colorGreenIndex count]; i++)

{

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

}

}

if (n\_sort\_col1%3 ==2) //红色排序显示在前面

{

NSMutableArray \*sort = [NSMutableArray array];

int i;

for (i=0; i<[\_colorRedIndexBackup count]; i++)

{

int index = [\_colorRedIndexBackup[i] intValue];

[sort addObject:\_sortDataBackup[index]];

[\_colorRedIndex addObject:[NSNumber numberWithInt:i]];

}

[\_data setArray:sort];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

for (int i=0; i<[\_colorRedIndex count]; i++)

{

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

}

}

if (n\_sort\_col1%3 ==0) //正常排序显示

{

[\_data setArray:\_sortDataBackup];

[\_colorRedIndex setArray:\_colorRedIndexBackup];

[\_colorGreenIndex setArray:\_colorGreenIndexBackup];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

}

NSLog(@"-----n\_sort\_col1 还原");

n\_sort\_col1 = 0;

}

[tmpColorArr removeAllObjects];

[tmpColorArr setArray:\_data];

[self.dataTableView reloadData];

}

else

{

NSLog(@"--only isight data ,can not sort!!! %zd",col);

}

}

else if([col\_identifier isEqualToString:identifier\_cpk\_orig]) //cpk-original color sort

{

NSLog(@"====n\_sort\_col1: %d",n\_sort\_col1);

if ([\_colorRedIndexCpkBackup count]>0 || [\_colorGreenIndexCpkBackup count]>0 || [\_colorYellowIndexCpkBackup count]>0)

{

if (n\_sort\_col9 %2 ==1 || n\_sort\_col12 !=0) //还原

{

n\_sort\_col9 =0;

n\_sort\_col12 =0;

//[self AlertBox:@"Error" withInfo:@"Please Click Reviewer Name column to data back."];

[\_data removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorRedIndex removeAllObjects];

[\_colorRedIndexCpk removeAllObjects];

[\_colorGreenIndexCpk removeAllObjects];

[\_colorYellowIndexCpk removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_data setArray:\_sortDataBackup];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

if ([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:i]];

}

else if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:i]];

}

}

[tmpColorArr removeAllObjects];

[tmpColorArr setArray:\_data];

n\_sort\_col1 = 0;

n\_sort\_col5 = 0;

}

n\_sort\_col5 ++;

[\_data removeAllObjects];

[\_colorGreenIndexCpk removeAllObjects];

[\_colorRedIndexCpk removeAllObjects];

[\_colorYellowIndexCpk removeAllObjects];

NSLog(@"====n\_sort\_col5: %d",n\_sort\_col5);

float cpkL = [[m\_configDictionary valueForKey:cpk\_Lowthl] floatValue];

float cpkH = [[m\_configDictionary valueForKey:cpk\_Highthl] floatValue];

if (n\_sort\_col5%4 ==1) //绿色排序显示在前面

{

NSMutableArray \*sort = [NSMutableArray array];

if (n\_sort\_col1%3 ==1) //找到相同元素

{

int j =0;

for (int i=0; i<\_colorGreenIndex.count; i++)

{

if ([self isPureFloat:tmpColorArr[i][tb\_cpk\_orig]]|| [self isPureInt:tmpColorArr[i][tb\_cpk\_orig]])

{

if (([tmpColorArr[i][tb\_cpk\_orig] floatValue]>=cpkL)&&([tmpColorArr[i][tb\_cpk\_orig] floatValue]<=cpkH))

{

[sort addObject:tmpColorArr[i]];

[\_colorGreenIndexCpk addObject:[NSNumber numberWithInt:j]];

j++;

}

}

}

[\_data setArray:sort];

}

else if (n\_sort\_col1%3 ==2) //找到相同元素

{

int j =0;

for (int i=0; i<\_colorRedIndex.count; i++)

{

if ([self isPureFloat:tmpColorArr[i][tb\_cpk\_orig]]|| [self isPureInt:tmpColorArr[i][tb\_cpk\_orig]])

{

if (([tmpColorArr[i][tb\_cpk\_orig] floatValue]>=cpkL)&&([tmpColorArr[i][tb\_cpk\_orig] floatValue]<=cpkH))

{

[sort addObject:tmpColorArr[i]];

[\_colorGreenIndexCpk addObject:[NSNumber numberWithInt:j]];

j++;

}

}

}

[\_data setArray:sort];

}

else if (n\_sort\_col1%3 ==0) //找到相同元素

{

int j=0;

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([\_colorGreenIndexCpkBackup containsObject:[NSNumber numberWithInt:i]])

{

[sort addObject:\_sortDataBackup[i]];

[\_colorGreenIndexCpk addObject:[NSNumber numberWithInt:j]];

if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:j]];

}

if (([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]]))

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:j]];

}

j++;

}

}

[\_data setArray:sort];

}

}

if (n\_sort\_col5%4 ==2) //红色排序显示在前面

{

NSMutableArray \*sort = [NSMutableArray array];

if (n\_sort\_col1%3 ==1) //找到相同元素

{

int j=0;

for (int i=0; i<\_colorGreenIndex.count; i++)

{

if ([self isPureFloat:tmpColorArr[i][tb\_cpk\_orig]]|| [self isPureInt:tmpColorArr[i][tb\_cpk\_orig]])

{

if ([tmpColorArr[i][tb\_cpk\_orig] floatValue]<cpkL)

{

[sort addObject:tmpColorArr[i]];

[\_colorRedIndexCpk addObject:[NSNumber numberWithInt:j]];

j++;

}

}

}

[\_data setArray:sort];

}

else if (n\_sort\_col1%3 ==2) //找到相同元素

{

int j=0;

for (int i=0; i<\_colorRedIndex.count; i++)

{

if ([self isPureFloat:tmpColorArr[i][tb\_cpk\_orig]]|| [self isPureInt:tmpColorArr[i][tb\_cpk\_orig]])

{

if ([tmpColorArr[i][tb\_cpk\_orig] floatValue]<cpkL)

{

[sort addObject:tmpColorArr[i]];

[\_colorRedIndexCpk addObject:[NSNumber numberWithInt:j]];

j++;

}

}

}

[\_data setArray:sort];

}

else if (n\_sort\_col1%3 ==0) //找到相同元素

{

int j =0;

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([\_colorRedIndexCpkBackup containsObject:[NSNumber numberWithInt:i]])

{

[sort addObject:\_sortDataBackup[i]];

[\_colorRedIndexCpk addObject:[NSNumber numberWithInt:j]];

if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:j]];

}

if (([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]]))

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:j]];

}

j++;

}

}

[\_data setArray:sort];

}

}

if (n\_sort\_col5%4 ==3) //黄色排序显示在前面

{

NSMutableArray \*sort = [NSMutableArray array];

if (n\_sort\_col1%3 ==1) //找到相同元素

{

int j=0;

for (int i=0; i<\_colorGreenIndex.count; i++)

{

if ([self isPureFloat:tmpColorArr[i][tb\_cpk\_orig]]|| [self isPureInt:tmpColorArr[i][tb\_cpk\_orig]])

{

if ([tmpColorArr[i][tb\_cpk\_orig] floatValue]>cpkH)

{

[sort addObject:tmpColorArr[i]];

[\_colorYellowIndexCpk addObject:[NSNumber numberWithInt:j]];

j++;

}

}

}

[\_data setArray:sort];

}

else if (n\_sort\_col1%3 ==2) //找到相同元素

{

int j=0;

for (int i=0; i<\_colorRedIndex.count; i++)

{

if ([self isPureFloat:tmpColorArr[i][tb\_cpk\_orig]]|| [self isPureInt:tmpColorArr[i][tb\_cpk\_orig]])

{

if ([tmpColorArr[i][tb\_cpk\_orig] floatValue]>cpkH)

{

[sort addObject:tmpColorArr[i]];

[\_colorYellowIndexCpk addObject:[NSNumber numberWithInt:j]];

j++;

}

}

}

[\_data setArray:sort];

}

else if (n\_sort\_col1%3 ==0) //找到相同元素

{

int j = 0;

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([\_colorYellowIndexCpkBackup containsObject:[NSNumber numberWithInt:i]])

{

[sort addObject:\_sortDataBackup[i]];

[\_colorYellowIndexCpk addObject:[NSNumber numberWithInt:j]];

if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:j]];

}

if (([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]]))

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:j]];

}

j++;

}

}

[\_data setArray:sort];

}

}

if (n\_sort\_col5%4 ==0) //正常排序显示

{

[\_data setArray:\_sortDataBackup];

[\_colorRedIndex removeAllObjects];

[\_colorGreenIndex removeAllObjects];

// [\_colorRedIndex setArray:\_colorRedIndexBackup];

// [\_colorGreenIndex setArray:\_colorGreenIndexBackup];

[\_colorGreenIndexCpkBackup removeAllObjects];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:i]];

}

else if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:i]];

}

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

}

[tmpColorArr removeAllObjects];

[tmpColorArr setArray:\_data];

n\_sort\_col1 = 0;

n\_sort\_col5 = 0;

/\* NSMutableArray \*sort = [NSMutableArray array];

if (n\_sort\_col1%3 ==1) //找到相同元素

{

for (int i=0; i<\_colorGreenIndex.count; i++)

{

if ([self isPureFloat:tmpColorArr[i][tb\_cpk\_orig]]|| [self isPureInt:tmpColorArr[i][tb\_cpk\_orig]])

{

[self setCpkOrigColor:tmpColorArr[i][tb\_cpk\_orig] withRow:i];

}

[sort addObject:tmpColorArr[i]];

}

[\_data setArray:sort];

}

else if (n\_sort\_col1%3 ==2) //找到相同元素

{

for (int i=0; i<\_colorRedIndex.count; i++)

{

if ([self isPureFloat:tmpColorArr[i][tb\_cpk\_orig]]|| [self isPureInt:tmpColorArr[i][tb\_cpk\_orig]])

{

[self setCpkOrigColor:tmpColorArr[i][tb\_cpk\_orig] withRow:i];

}

[sort addObject:tmpColorArr[i]];

}

[\_data setArray:sort];

}

else if (n\_sort\_col1%3 ==0) //找到相同元素

{

[\_data setArray:\_sortDataBackup];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([self isPureFloat:\_sortDataBackup[i][tb\_cpk\_orig]]|| [self isPureInt:\_sortDataBackup[i][tb\_cpk\_orig]])

{

[self setCpkOrigColor:\_sortDataBackup[i][tb\_cpk\_orig] withRow:i];

}

}

}

\*/

}

[self.dataTableView reloadData];

}

else

{

NSLog(@"====click cpk orig");

}

}

else if([col\_identifier isEqualToString:identifier\_reviewer]) //reviewer col ==colselect reviewer \_name

{

if ([\_reviewerNameIndex count] > 0)

{

n\_sort\_col9 ++;

[\_data removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorRedIndex removeAllObjects];

[\_colorGreenIndexCpk removeAllObjects];

[\_colorRedIndexCpk removeAllObjects];

[\_colorYellowIndexCpk removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

if (n\_sort\_col9 %2 ==1)

{

NSMutableArray \*sort = [NSMutableArray array];

int j = 0;

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([\_reviewerNameIndex containsObject:[NSNumber numberWithInt:i]])

{

[sort addObject:\_sortDataBackup[i]];

if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:j]];

}

if (([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]]))

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:j]];

}

j++;

}

}

[\_data setArray:sort];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

for (int i=0; i<[\_data count]; i++)

{

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

}

}

else if(n\_sort\_col9 %2 ==0)

{

[\_data setArray:\_sortDataBackup];

[\_colorRedIndex removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:i]];

}

else if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:i]];

}

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

// if ([\_data[i][tb\_reviewer] isNotEqualTo:@""])

// {

// [\_reviewerNameIndex addObject:[NSNumber numberWithInt:i]];

// }

}

[tmpColorArr removeAllObjects];

[tmpColorArr setArray:\_data];

}

[self.dataTableView reloadData];

}

}

else if([col\_identifier isEqualToString:identifier\_bmc]) //BM col ==col select bm

{

if ([\_bmcYesIndex count] > 0 || [\_bmcNoIndex count] > 0)

{

n\_sort\_col12 ++;

[\_data removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorRedIndex removeAllObjects];

[\_colorGreenIndexCpk removeAllObjects];

[\_colorRedIndexCpk removeAllObjects];

[\_colorYellowIndexCpk removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

if (n\_sort\_col12 %3 ==1)

{

NSMutableArray \*sort = [NSMutableArray array];

int j = 0;

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([\_bmcYesIndex containsObject:[NSNumber numberWithInt:i]])

{

[sort addObject:\_sortDataBackup[i]];

if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:j]];

}

if (([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]]))

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:j]];

}

j++;

}

}

[\_data setArray:sort];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

for (int i=0; i<[\_data count]; i++)

{

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

}

}

if (n\_sort\_col12 %3 ==2)

{

NSMutableArray \*sort = [NSMutableArray array];

int j = 0;

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([\_bmcNoIndex containsObject:[NSNumber numberWithInt:i]])

{

[sort addObject:\_sortDataBackup[i]];

if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:j]];

}

if (([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]]))

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:j]];

}

j++;

}

}

[\_data setArray:sort];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

for (int i=0; i<[\_data count]; i++)

{

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

}

}

else if(n\_sort\_col12 %3 ==0)

{

[\_data setArray:\_sortDataBackup];

[\_colorRedIndex removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([\_colorRedIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorRedIndex addObject:[NSNumber numberWithInt:i]];

}

else if ([\_colorGreenIndexBackup containsObject:[NSNumber numberWithInt:i]])

{

[\_colorGreenIndex addObject:[NSNumber numberWithInt:i]];

}

[self setCpkOrigColor:\_data[i][tb\_cpk\_orig] withRow:i];

}

[tmpColorArr removeAllObjects];

[tmpColorArr setArray:\_data];

}

[self.dataTableView reloadData];

}

}

else

{

NSLog(@"--select data tbs item is wrong!!! %zd",col);

}

return;

}

click\_tb\_row = row;

[self triggerGeneratePlot:row withApplyBox:NO withSelectXY:-1];

}

-(NSString \*)opensslSha1:(NSString \*)inputStr

{

unsigned char digest[SHA\_DIGEST\_LENGTH];

const char\* string = [inputStr UTF8String];

SHA\_CTX ctx;

SHA1\_Init(&ctx);

SHA1\_Update(&ctx, string, strlen(string));

SHA1\_Final(digest, &ctx);

char mdString[SHA\_DIGEST\_LENGTH\*2+1];

for (int i = 0; i < SHA\_DIGEST\_LENGTH; i++)

sprintf(&mdString[i\*2], "%02x", (unsigned int)digest[i]);

NSString \* hashCode = [NSString stringWithFormat:@"%s",mdString];

return hashCode;

}

-(NSString \*)opensslSha1FilePath:(NSString \*)path

{

FILE\* file = fopen([path UTF8String], "rb");

SHA\_CTX c;

unsigned char md[SHA\_DIGEST\_LENGTH];

int fd;

ssize\_t i;

unsigned char buf[BUFSIZE];

fd=fileno(file);

SHA1\_Init(&c);

for (;;)

{

i=read(fd,buf,BUFSIZE);

if (i <= 0) break;

SHA1\_Update(&c,buf,(unsigned long)i);

}

SHA1\_Final(&(md[0]),&c);

char mdString[SHA\_DIGEST\_LENGTH\*2+1];

for (i=0; i<SHA\_DIGEST\_LENGTH; i++)

sprintf(&mdString[i\*2], "%02x", (unsigned int)md[i]);

NSString \* hashCode = [NSString stringWithFormat:@"%s",mdString];

return hashCode;

}

-(void)clickOneItem:(NSNotification \*)nf

{

[self triggerGeneratePlot:click\_tb\_row withApplyBox:YES withSelectXY:-1];

}

-(BOOL)triggerGeneratePlot:(NSInteger)rowtb withApplyBox:(BOOL)ApplyBoxflag withSelectXY:(int)xy

{

NSInteger row = 0;

for (NSInteger i= 0; i<[\_ListAllItemNameArr count]; i++) //当UI 选择search 的时候，数据变了，row 也变了，要找到对应值

{

if ([\_ListAllItemNameArr[i] isEqualToString:\_data[rowtb][1]])

{

row = i;

break;

}

}

tbDataTableSelectItemRow = row;

NSString \*retest = [m\_configDictionary valueForKey:kRetestSeg];

NSString \*removeFail = [m\_configDictionary valueForKey:kRemoveFailSeg];

NSString \*bins = [m\_configDictionary valueForKey:kBins];

NSString \*typeZoom = [m\_configDictionary valueForKey:kzoom\_type];

//NSLog(@"==>row:%zd %@ %@ bin:%@ zoom type: %@ ",row,retest,removeFail,bins,typeZoom);

\_data[rowtb][tb\_color\_by\_left]= [NSNumber numberWithInteger:selectColorBoxIndex]; //设置color By左边那个,给python生成图表用

\_data[rowtb][tb\_color\_by\_right]= [NSNumber numberWithInteger:selectColorBoxIndex2]; //设置color By左边那个,给python生成图表用

//NSLog(@"---->select color by1: %d by2: %d",selectColorBoxIndex,selectColorBoxIndex2);

\_data[rowtb][tb\_bins] = bins;

\_data[rowtb][tb\_zoom\_type] = typeZoom;

if (clickItemIndex!=row|| [retestValue isNotEqualTo:retest] || [removeValue isNotEqualTo:removeFail]||ApplyBoxflag)

{

clickItemIndex = row;

retestValue = retest;

removeValue = removeFail;

if (selectColorBoxIndex == 0 && selectColorBoxIndex2 == 0) //color by box 关闭

{

NSString \*choose\_item\_name = [\_indexItemNameDic valueForKey:[NSString stringWithFormat:@"%zd",row]];

NSString \* itemName = [self combineItemName:choose\_item\_name];

//NSLog(@"--ClickOnTableView--:%zd selectColorBoxIndex:%d, selectColorBoxIndex2:%d,item name : %@",row,selectColorBoxIndex,selectColorBoxIndex2,itemName);

NSInteger row\_0 = [[m\_configDictionary valueForKey:kChooseItemIndex] integerValue];

NSMutableArray \* itemData\_0 = [self calculateData:row\_0];

NSMutableArray \* itemData = [self calculateData:row]; // must after itemData\_0, because of show data

if ([[m\_configDictionary valueForKey:kInputRangeFlag] boolValue])

{

NSString \*rangelsl = [m\_configDictionary valueForKey:krangelsl];

NSString \*rangeusl = [m\_configDictionary valueForKey:krangeusl];

itemData\_0[tb\_range\_lsl] = rangelsl;

itemData\_0[tb\_range\_usl] = rangeusl;

itemData[tb\_range\_lsl] = rangelsl;

itemData[tb\_range\_usl] = rangeusl;

NSLog(@">>range: %@,%@",rangelsl,rangeusl);

}

else

{

NSString \*rangelsl = itemData[tb\_lower];

NSString \*rangeusl = itemData[tb\_upper];

itemData\_0[tb\_range\_lsl] = rangelsl;

itemData\_0[tb\_range\_usl] = rangeusl;

itemData[tb\_range\_lsl] = rangelsl;

itemData[tb\_range\_usl] = rangeusl;

NSLog(@".>>range: %@,%@",rangelsl,rangeusl);

}

if (((n\_firstItemClick ==0) && ([itemData count]>38 ))|| (n\_firstItemClick ==1)) // n\_firstItemClick 为1 就是点击了select x 或者 select y

{

[m\_configDictionary setValue:[NSNumber numberWithInteger:row] forKey:kChooseItemIndex];

[m\_configDictionary setValue:choose\_item\_name forKey:kChooseItemName];

n\_firstItemClick =10;

}

if (n\_firstItemClick == 10)

{

NSString \*itemName\_0 = [NSString stringWithFormat:@"%@\_XY",[m\_configDictionary valueForKey:kChooseItemName]];

itemData[tb\_correlation\_xy] = itemName\_0;

itemData\_0[tb\_correlation\_xy] = itemName\_0;

[self sendDataToRedis:itemName\_0 withData:itemData\_0];

}

if (xy==-1)

{

// do nothing

}

else

{

itemName = [NSString stringWithFormat:@"%@$$%d",itemName,xy];

}

[self sendDataToRedis:itemName withData:itemData];

[self sendCpkZmqMsg:itemName];

[self sendCorrelationZmqMsg:itemName];

[self sendScatterZmqMsg:itemName];

}

else

{

if (n\_firstItemClick ==0|| n\_firstItemClick ==1)

{

NSString \*choose\_item\_name = [\_indexItemNameDic valueForKey:[NSString stringWithFormat:@"%zd",row]];

[m\_configDictionary setValue:[NSNumber numberWithInteger:row] forKey:kChooseItemIndex];

[m\_configDictionary setValue:choose\_item\_name forKey:kChooseItemName];

n\_firstItemClick =10;

}

if (xy==-1)

{

// NSDictionary \*dic = [NSDictionary dictionaryWithObject:[NSNumber numberWithBool:ApplyBoxflag] forKey:applyBoxCheck];

if (selectColorBoxIndex > 0)

{

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickPlotTable object:nil userInfo:nil];

}

else if (selectColorBoxIndex2 > 0)

{

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickPlotTable2 object:nil userInfo:nil];

}

}

else

{

NSDictionary \*dic = [NSDictionary dictionaryWithObject:[NSNumber numberWithInt:xy] forKey:selectXY];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickPlotTable\_selectXY object:nil userInfo:dic];

}

}

}

else

{

if (selectColorBoxIndex > 0 )

{

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickPlotTable object:nil userInfo:nil];

}

else if(selectColorBoxIndex2 > 0)

{

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickPlotTable2 object:nil userInfo:nil];

}

else

{

NSLog(@"--click the same item, do nothing");

}

}

return ApplyBoxflag;

}

- (IBAction)btnClickItem:(NSTextField \*)sender

{

NSLog(@"=====>>>>>>item click");

[sender setEditable:YES];

[sender setBordered:NO];

}

//-(BOOL)tableView:(NSTableView \*)tableView shouldEditTableColumn:(NSTableColumn \*)tableColumn row:(NSInteger)row

//{

// return YES;

//}

-(IBAction)btnClickKeynoteApply:(NSButton\*)sender

{

NSInteger btnTag = sender.tag; // select row

NSInteger state = sender.state;

NSLog(@"btnClickKeynoteApply: %zd,%zd",btnTag,state);

\_data[btnTag][tb\_keynote] = [NSNumber numberWithInteger:state];

[self.dataTableView reloadData];

[m\_configDictionary setValue:[NSNumber numberWithBool:btnTag] forKey:K\_dic\_keynoteBoxCheck];

}

-(void)reloadReviewerDate:(NSInteger)row

{

NSString \*lsl = [\_data[row][tb\_lsl] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

NSString \*usl = [\_data[row][tb\_usl] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

if ([lsl length]<1 && [usl length]<1)

{

return;

}

float lowPrevious = -9999;

float highPrevious = -9999;

NSString \*reviewer\_name = @"";

NSString \*reviewer\_date = @"";

NSString \*user\_comment = @"";

if (\_limitUpdateData.count >0)

{

for (int i=0; i<[\_limitUpdateData count]; i++) // find load excel之前的设置

{

if ([\_limitUpdateData[i] count]==27)

{

if ([\_limitUpdateData[i][1] isEqualToString:\_data[row][tb\_item]] )

{

if ([\_limitUpdateData[i][updatelimit\_newLower-1] isNotEqualTo:@""] &&[\_limitUpdateData[i][updatelimit\_newUpper-1] isNotEqualTo:@""]) //load excel之前的设置

{

lowPrevious = [[\_limitUpdateData[i][updatelimit\_newLower-1]stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]] floatValue];

highPrevious = [[\_limitUpdateData[i][updatelimit\_newUpper-1]stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]] floatValue];

reviewer\_name = [\_limitUpdateData[i][updatelimit\_reviewer\_name-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

reviewer\_date = [\_limitUpdateData[i][updatelimit\_reviewer\_date-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

user\_comment = [\_limitUpdateData[i][updatelimit\_user\_comment-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

break;

}

}

}

else if ([\_limitUpdateData[i] count]==28)

{

if ([\_limitUpdateData[i][1] isEqualToString:\_data[row][tb\_item]] )

{

if ([\_limitUpdateData[i][updatelimit\_newLower] isNotEqualTo:@""] &&[\_limitUpdateData[i][updatelimit\_newUpper] isNotEqualTo:@""]) //load excel之前的设置

{

lowPrevious = [[\_limitUpdateData[i][updatelimit\_newLower]stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]] floatValue];

highPrevious = [[\_limitUpdateData[i][updatelimit\_newUpper]stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]] floatValue];

reviewer\_name = [\_limitUpdateData[i][updatelimit\_reviewer\_name] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

reviewer\_date = [\_limitUpdateData[i][updatelimit\_reviewer\_date] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

user\_comment = [\_limitUpdateData[i][updatelimit\_user\_comment] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

break;

}

}

}

}

}

if (lowPrevious != highPrevious && lowPrevious != -9999)

{

float low = [lsl floatValue];

float high = [usl floatValue];

//NSString \*comment = [\_data[row][tb\_comment] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

if (low == lowPrevious && high == highPrevious)

{

\_data[row][tb\_comment] = user\_comment;

\_data[row][tb\_reviewer] = reviewer\_name;

\_data[row][tb\_date] = reviewer\_date;

b\_ClearComment = YES;

}

else

{

if (b\_ClearComment)

{

\_data[row][tb\_comment] = @"";

}

\_data[row][tb\_reviewer] = @"";

\_data[row][tb\_date] = @"";

}

[self.dataTableView reloadData];

}

}

-(IBAction)btnClickApply:(NSButton\*)sender

{

n\_clickApplyRow = -1;

NSLog(@"==>%@ %@",[m\_configDictionary valueForKey:kRetestSeg],[m\_configDictionary valueForKey:kRemoveFailSeg]);

NSInteger btnTag = sender.tag; // select row

NSInteger state = sender.state;

float lowPrevious = 0;

float highPrevious = 0;

NSString \*reviewer\_name = @"";

NSString \*reviewer\_date = @"";

NSString \*user\_comment = @"";

if (\_limitUpdateData.count >0)

{

for (int i=0; i<[\_limitUpdateData count]; i++) // find load excel之前的设置

{

if ([\_limitUpdateData[i] count]==27)

{

if ([\_limitUpdateData[i][1] isEqualToString:\_data[btnTag][tb\_item]] )

{

if ([\_limitUpdateData[i][updatelimit\_newLower-1] isNotEqualTo:@""] &&[\_limitUpdateData[i][updatelimit\_newUpper-1] isNotEqualTo:@""]) //load excel之前的设置

{

lowPrevious = [[\_limitUpdateData[i][updatelimit\_newLower-1]stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]] floatValue];

highPrevious = [[\_limitUpdateData[i][updatelimit\_newUpper-1]stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]] floatValue];

reviewer\_name = [\_limitUpdateData[i][updatelimit\_reviewer\_name-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

reviewer\_date = [\_limitUpdateData[i][updatelimit\_reviewer\_date-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

user\_comment = [\_limitUpdateData[i][updatelimit\_user\_comment-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

break;

}

}

}

else if ([\_limitUpdateData[i] count]==28)

{

if ([\_limitUpdateData[i][1] isEqualToString:\_data[btnTag][tb\_item]] )

{

if ([\_limitUpdateData[i][updatelimit\_newLower] isNotEqualTo:@""] &&[\_limitUpdateData[i][updatelimit\_newUpper] isNotEqualTo:@""]) //load excel之前的设置

{

lowPrevious = [[\_limitUpdateData[i][updatelimit\_newLower]stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]] floatValue];

highPrevious = [[\_limitUpdateData[i][updatelimit\_newUpper]stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]] floatValue];

reviewer\_name = [\_limitUpdateData[i][updatelimit\_reviewer\_name] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

reviewer\_date = [\_limitUpdateData[i][updatelimit\_reviewer\_date] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

user\_comment = [\_limitUpdateData[i][updatelimit\_user\_comment] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

break;

}

}

}

}

}

// if ([\_data[btnTag][tb\_lsl] isNotEqualTo:@""] &&[\_data[btnTag][tb\_usl] isNotEqualTo:@""]) //load excel之前的设置

// {

// lowPrevious = [\_data[btnTag][tb\_lsl] floatValue];

// highPrevious = [\_data[btnTag][tb\_usl] floatValue];

// }

\_data[btnTag][tb\_apply] = [NSNumber numberWithInteger:state];

\_data[btnTag][tb\_color\_by\_left]= [NSNumber numberWithInteger:selectColorBoxIndex]; //设置color By左边那个,给python生成图表用

\_data[btnTag][tb\_color\_by\_right]= [NSNumber numberWithInteger:selectColorBoxIndex2]; //设置color By right那个,给python生成图表用

[self.dataTableView reloadData];

if ([\_data[btnTag][tb\_lsl] isEqualTo:@""] && [\_data[btnTag][tb\_usl] isEqualTo:@""])

{

[self AlertBox:@"Warning!!!" withInfo:@"Please input LSL or USL firstly!!!"];

\_data[btnTag][tb\_apply] = [NSNumber numberWithInt:0];

[self.dataTableView reloadData];

return;

}

if ([\_data[btnTag][tb\_lsl] isEqualTo:@""])

{

[self AlertBox:@"Warning!!!" withInfo:@"Please input LSL firstly!!!"];

\_data[btnTag][tb\_apply] = [NSNumber numberWithInt:0];

[self.dataTableView reloadData];

return;

}

if ([\_data[btnTag][tb\_usl] isEqualTo:@""])

{

[self AlertBox:@"Warning!!!" withInfo:@"Please input USL firstly!!!"];

\_data[btnTag][tb\_apply] = [NSNumber numberWithInt:0];

[self.dataTableView reloadData];

return;

}

if ([\_data[btnTag][tb\_lsl] isNotEqualTo:@""] &&[\_data[btnTag][tb\_usl] isNotEqualTo:@""] && ([\_data[btnTag][tb\_lsl] isNotEqualTo:@"NA"] && [\_data[btnTag][tb\_usl] isNotEqualTo:@"NA"]))

{

float low = [\_data[btnTag][tb\_lsl] floatValue];

float high = [\_data[btnTag][tb\_usl] floatValue];

if (low>high)

{

[self AlertBox:@"Error:023" withInfo:@"Input LSL is bigger than USL!!!"];

\_data[btnTag][tb\_apply] = [NSNumber numberWithInt:0];

\_data[btnTag][tb\_lsl] = @"";

\_data[btnTag][tb\_usl] = @"";

[self.dataTableView reloadData];

return;

}

if (lowPrevious != highPrevious)

{

if (low != lowPrevious || high != highPrevious)

{

if (b\_ClearComment)

{

\_data[btnTag][tb\_comment] = @"";

}

\_data[btnTag][tb\_reviewer] = @"";

\_data[btnTag][tb\_date] = @"";

}

else

{

\_data[btnTag][tb\_comment] = user\_comment;

\_data[btnTag][tb\_reviewer] = reviewer\_name;

\_data[btnTag][tb\_date] = reviewer\_date;

b\_ClearComment = YES;

}

}

}

/\*if (state == 1)

{

NSDateFormatter\* DateFomatter = [[NSDateFormatter alloc] init];

[DateFomatter setDateFormat:@"yyyy-MM-dd HH:mm:ss"];

NSString\* systemTime = [DateFomatter stringFromDate:[NSDate date]];

\_data[btnTag][13] = systemTime;

}

else

{

\_data[btnTag][13] = @"";

}

\*/

n\_clickApplyRow = btnTag;

if (state == 1)

{

}

else

{

\_data[btnTag][tb\_cpk\_new] =@"";

}

[m\_configDictionary setValue:[NSNumber numberWithBool:btnTag] forKey:K\_dic\_ApplyBoxCheck];

NSString \* itemName =[self combineItemName: [\_indexItemNameDic valueForKey:[NSString stringWithFormat:@"%zd",btnTag]]];

NSLog(@">tag:%zd state:%zd, item name: %@",btnTag,state,itemName);

[self triggerGeneratePlot:btnTag withApplyBox:YES withSelectXY:-1];

}

-(NSString \*)openCSVLoadPanel

{

//[[NSWorkspace sharedWorkspace] openFile:@"~/desktop"];

//[[NSWorkspace sharedWorkspace] openFile:desktopPath];

// [panel setDirectoryURL:[NSURL URLWithString:desktopPath]];

//[panel beginSheetModalForWindow:self.view.window completionHandler:^(NSInteger result) //[NSApp mainWindow]

// {

// if (result == NSModalResponseOK) {

// @try {

// csvpath = [[[panel URLs] objectAtIndex:0] path];

// [self.txtScriptPath setStringValue:csvpath];

// }

// @catch (NSException \*exception) {

// NSLog(@"Load file failed,please check the data");

// }

// @finally {

// }

// }

// }];

NSString \*csvpath =nil;

NSOpenPanel \* panel = [NSOpenPanel openPanel];

[panel setAllowsMultipleSelection:NO]; //设置多选模式

[panel setCanChooseFiles:YES];

[panel setCanCreateDirectories:YES];

[panel setCanChooseDirectories:YES];

[panel setAllowedFileTypes:[NSArray arrayWithObjects:@"CSV", @"csv", @"Csv",nil]];

[panel setDirectoryURL:[NSURL URLWithString:desktopPath]];

[panel runModal];

if ([[panel URLs] count]>0)

{

csvpath = [[[panel URLs] objectAtIndex:0] path];

[self.txtcsvDataName setStringValue:csvpath];

}

else

{

[self.txtcsvDataName setStringValue:@"--"];

}

if (csvpath==nil || [csvpath isEqualToString:desktopPath])

{

return nil;

}

return csvpath;

}

- (IBAction)btnSearchCsv:(id)sender

{

if (n\_search == 0)

{

[\_dataBackup setArray:\_data];

[\_colorRedIndexSearchBackup setArray:\_colorRedIndex];

[\_colorGreenIndexSearchBackup setArray:\_colorGreenIndex];

[\_colorRedIndexSearchCpk setArray: \_colorRedIndexCpk];

[\_colorGreenIndexSearchCpk setArray:\_colorGreenIndexCpk];

[\_colorYellowIndexSearchCpk setArray:\_colorYellowIndexCpk];

/\*[\_dataBackup setArray:\_sortDataBackup];

[\_colorRedIndexSearchBackup setArray:\_colorRedIndexBackup];

[\_colorGreenIndexSearchBackup setArray:\_colorGreenIndexBackup];

[\_colorRedIndexSearchCpk setArray: \_colorRedIndexCpkBackup];

[\_colorGreenIndexSearchCpk setArray:\_colorGreenIndexCpkBackup];

[\_colorYellowIndexSearchCpk setArray:\_colorYellowIndexCpkBackup];

\*/

n\_search ++;

}

[arrSearch removeAllObjects];

[arrSearchRed removeAllObjects];

[arrSearchGreen removeAllObjects];

[arrSearchRedCPK removeAllObjects];

[arrSearchGreenCPK removeAllObjects];

[arrSearchYellowCPK removeAllObjects];

NSString \*content = @"";

if ([sender isNotEqualTo:@""])

{

content = [sender stringValue];

}

if (content.length<2)

{

[\_data setArray:\_dataBackup];

[\_colorRedIndex setArray:\_colorRedIndexSearchBackup];

[\_colorGreenIndex setArray:\_colorGreenIndexSearchBackup];

[\_colorRedIndexCpk setArray: \_colorRedIndexSearchCpk];

[\_colorGreenIndexCpk setArray:\_colorGreenIndexSearchCpk];

[\_colorYellowIndexCpk setArray:\_colorYellowIndexSearchCpk];

[self.dataTableView reloadData];

n\_search = 0;

return;

}

[self searchFind:content];

}

-(void)searchFind:(NSString \*)content

{

NSUInteger m\_length1 = [\_sortDataBackup count];

NSUInteger m\_length2 = [\_dataBackup count];

if(m\_length1 == m\_length2)

{

[arrSearchRedCPK removeAllObjects];

[arrSearchGreenCPK removeAllObjects];

[arrSearchYellowCPK removeAllObjects];

int n\_num = 0;

for (NSArray \*lineData in \_dataBackup)

{

NSString \*lineStr = lineData[1];

if ([lineStr.uppercaseString containsString:content.uppercaseString])

{

[arrSearch addObject:lineData];

int x\_row = [lineData[0] intValue]-1;

if ([\_colorRedIndexSearchBackup containsObject:[NSNumber numberWithInt:x\_row]])

{

[arrSearchRed addObject:[NSNumber numberWithInt:n\_num]];

}

else if ([\_colorGreenIndexSearchBackup containsObject:[NSNumber numberWithInt:x\_row]])

{

[arrSearchGreen addObject:[NSNumber numberWithInt:n\_num]];

}

if ([\_colorRedIndexSearchCpk containsObject:[NSNumber numberWithInt:x\_row]])

{

[arrSearchRedCPK addObject:[NSNumber numberWithInt:n\_num]];

}

else if ([\_colorGreenIndexSearchCpk containsObject:[NSNumber numberWithInt:x\_row]])

{

[arrSearchGreenCPK addObject:[NSNumber numberWithInt:n\_num]];

}

else if ([\_colorYellowIndexSearchCpk containsObject:[NSNumber numberWithInt:x\_row]])

{

[arrSearchYellowCPK addObject:[NSNumber numberWithInt:n\_num]];

}

n\_num++;

}

}

[\_data setArray:arrSearch];

}

else

{

[arrSearchRedCPK removeAllObjects];

[arrSearchGreenCPK removeAllObjects];

[arrSearchYellowCPK removeAllObjects];

int n\_num = 0;

for (NSArray \*lineData in \_dataBackup)

{

NSString \*lineStr = lineData[1];

if ([lineStr.uppercaseString containsString:content.uppercaseString])

{

[arrSearch addObject:lineData];

if ([\_colorRedIndexSearchBackup containsObject:[NSNumber numberWithInt:n\_num]])

{

[arrSearchRed addObject:[NSNumber numberWithInt:n\_num]];

}

else if ([\_colorGreenIndexSearchBackup containsObject:[NSNumber numberWithInt:n\_num]])

{

[arrSearchGreen addObject:[NSNumber numberWithInt:n\_num]];

}

n\_num++;

}

}

[\_data setArray:arrSearch];

int n\_num2 = 0;

float cpkL = [[m\_configDictionary valueForKey:cpk\_Lowthl] floatValue];

float cpkH = [[m\_configDictionary valueForKey:cpk\_Highthl] floatValue];

for (NSArray \*lineData in arrSearch)

{

NSString \*value = lineData[tb\_cpk\_orig];

if ([self isPureFloat:value]|| [self isPureInt:value])

{

if ([value floatValue]<cpkL)

{

[arrSearchRedCPK addObject:[NSNumber numberWithInt:n\_num2]];

}

else if([value floatValue] >cpkH)

{

[arrSearchYellowCPK addObject:[NSNumber numberWithInt:n\_num2]];

}

else

{

[arrSearchGreenCPK addObject:[NSNumber numberWithInt:n\_num2]];

}

}

n\_num2++;

}

}

[\_colorRedIndex setArray:arrSearchRed];

[\_colorGreenIndex setArray:arrSearchGreen];

[\_colorRedIndexCpk setArray: arrSearchRedCPK];

[\_colorGreenIndexCpk setArray:arrSearchGreenCPK];

[\_colorYellowIndexCpk setArray:arrSearchYellowCPK];

[self.dataTableView reloadData];

}

-(BOOL)isContinuous:(NSArray \*)array

{

if([array count]<1)

return true;

int min = [array[0] intValue];

int max = [array[0] intValue];

for (int i = 1; i < [array count]; i++)

{

if (array[i] != 0)

{

if (min>[array[i] intValue])

{

min = [array[i] intValue];

}

if (max < [array[i] intValue])

{

max = [array[i] intValue];

}

}

}

if ((max - min)>([array count] - 1))

return false;

else

return true;

}

- (IBAction)btLoadCsvData:(id)sender // for button use

{

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.25; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

//if ([self.splitView isSubviewCollapsed:self.rightPanel])

// if(n\_loadCsvBtn %2==1)

// {

// // -> expand

// [self.splitView setPosition:\_lastLeftPaneWidth ofDividerAtIndex:0];

// }

// else {

// // <- collapse

// \_lastLeftPaneWidth = self.leftPane.frame.size.width; // remember current width to restore

// [self.splitView setPosition:0 ofDividerAtIndex:0];

// }

if (\_lastLeftPaneWidth==0 )

{

[self.splitView setPosition:1200 ofDividerAtIndex:0];

\_lastLeftPaneWidth = 1200;

}

else

{

[self.splitView setPosition:0 ofDividerAtIndex:0];

\_lastLeftPaneWidth = 0;

}

[self.splitView layoutSubtreeIfNeeded];

}];

//return;

// [self openSheet:sender];

/\* NSString \*csvPath = [self openCSVLoadPanel];

if (!csvPath) {

NSLog(@"--no csv select");

return;

}

[m\_configDictionary setValue:[NSNumber numberWithBool:NO] forKey:K\_dic\_Load\_Csv\_Finished];

[m\_configDictionary setValue:csvPath forKey:Load\_Csv\_Path];

[self sendCalculateZmqMsg:@"calculate-param"];

NSTimeInterval starttime = [[NSDate date]timeIntervalSince1970];

//NSString \*csvPath = @"/Users/RyanGao/Desktop/cpk/cpk\_data\_0611/J5xx-FCT.csv"; //J5xx-FCT test

[self reloadDataWithPath:csvPath];

NSTimeInterval now = [[NSDate date]timeIntervalSince1970];

[self initRetestAndRemoveFailSeg];

[self initColorByTableView];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationInitColorTable object:nil userInfo:nil];

//再次加载，需要init

enableEditing = YES;

clickItemIndex = -1;

editLimitRow = -1;

retestValue=@"";

removeValue = @"";

NSLog(@"====load csv执行时间: %f",now-starttime);

NSString \*file1 = [NSString stringWithFormat:@"%@/CPK\_Log/temp/cpk.png",desktopPath];

NSFileManager \*manager = [NSFileManager defaultManager];

[manager removeItemAtPath:file1 error:nil];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"cpk.png" ofType:nil];

[manager copyItemAtPath:picPath toPath:file1 error:nil];

NSString \*file2 = [NSString stringWithFormat:@"%@/CPK\_Log/temp/correlation.png",desktopPath];

[manager removeItemAtPath:file2 error:nil];

NSString \*picPath2 =[[NSBundle mainBundle]pathForResource:@"correlation.png" ofType:nil];

[manager copyItemAtPath:picPath2 toPath:file2 error:nil];

\*/

}

-(void)toLoadLocalCsv:(NSNotification \*)nf

{

[self launch\_yield\_rate];

[self launch\_calculate\_test];

[self launch\_retest\_plot];

[m\_configDictionary setValue:[NSNumber numberWithBool:NO] forKey:K\_dic\_Load\_Csv\_Finished];

NSDictionary\* info = [nf userInfo];

NSString \*csvPath = [info valueForKey:@"data\_csv"];

[m\_configDictionary setValue:[info valueForKey:cpk\_Lowthl] forKey:cpk\_Lowthl];

[m\_configDictionary setValue:[info valueForKey:cpk\_Highthl] forKey:cpk\_Highthl];

NSLog(@"---->>>local csvpath: %@ ",csvPath);

[m\_configDictionary setValue:csvPath forKey:Load\_Local\_Csv\_Path];

[m\_configDictionary setValue:@"" forKey:Load\_Csv\_Path]; //清掉

[m\_configDictionary setValue:@"" forKey:Load\_Script\_Path];

// NSString \*csvPath = [m\_configDictionary valueForKey:Load\_Csv\_Path];

//for debug

//[self AlertBox:@"!!!" withInfo:[NSString stringWithFormat:@"Under development:\r\n%@",csvPath]];

//return;

if (!csvPath)

{

NSLog(@"--no csv select");

return;

}

NSFileManager \*manager = [NSFileManager defaultManager];

if (![manager fileExistsAtPath:csvPath])

{

[self AlertBox:@"Error:001" withInfo:[NSString stringWithFormat:@"Local data file did not exist at Path:\r\n%@",csvPath]];

return;

}

//[self.txtcsvDataName setStringValue:[csvPath lastPathComponent]];

//[self.txtScriptName setStringValue:@""];

//[self.txtLimitUpdate setStringValue:@""];

[self initRetestPlotAndCsv];

[self sendYieldRateZmqMsgLocal:@"yield\_rate-param-local"];

[self sendCalculateZmqMsgLocal:@"calculate-param-local"];

//NSTimeInterval starttime = [[NSDate date]timeIntervalSince1970];

//NSString \*csvPath = @"/Users/RyanGao/Desktop/cpk/cpk\_data\_0611/J5xx-FCT.csv"; //J5xx-FCT test

if(![self reloadLocalDataWithPath:csvPath])

{

return;

}

//解析脚本csv

//解析XLSX

//NSTimeInterval now = [[NSDate date]timeIntervalSince1970];

[self initRetestAndRemoveFailSeg];

[self initColorByTableView];

[m\_configDictionary setValue:\_ListAllItemNameArr forKey:k\_All\_Item\_Name];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationInitColorTable object:nil userInfo:nil];

//再次加载，需要init

enableEditing = YES;

clickItemIndex = -1;

editLimitRow = -1;

retestValue=@"";

removeValue = @"";

//NSLog(@"====load csv执行时间: %f",now-starttime);

NSString \*file1 = @"/tmp/CPK\_Log/temp/cpk.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/cpk.png",desktopPath];

[manager removeItemAtPath:file1 error:nil];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"cpk.png" ofType:nil];

[manager copyItemAtPath:picPath toPath:file1 error:nil];

NSString \*file2 = @"/tmp/CPK\_Log/temp/correlation.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/correlation.png",desktopPath];

[manager removeItemAtPath:file2 error:nil];

NSString \*picPath2 =[[NSBundle mainBundle]pathForResource:@"correlation.png" ofType:nil];

[manager copyItemAtPath:picPath2 toPath:file2 error:nil];

NSString \*file3 = @"/tmp/CPK\_Log/temp/scatter.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/scatter.png",desktopPath];

[manager removeItemAtPath:file3 error:nil];

NSString \*picPath3 =[[NSBundle mainBundle]pathForResource:@"scatter.png" ofType:nil];

[manager copyItemAtPath:picPath3 toPath:file3 error:nil];

\_lastLeftPaneWidth = self.leftPane.frame.size.width; // remember current width to restore

[self.splitView setPosition:0 ofDividerAtIndex:0];

\_lastLeftPaneWidth = 0;

[self.splitView layoutSubtreeIfNeeded];

[self.dataTableView setFocusRingType:NSFocusRingTypeNone];

[self.dataTableView setAccessibilityFocused:YES];//isAccessibilityFocused

[[NSNotificationCenter defaultCenter]postNotificationName:kLoadGroupPanel object:nil userInfo:nil];

[self setHiddenCol:YES];

b\_ClearComment = YES;

}

-(int)deleteFiles:(NSString \*)path

{

NSFileManager \* fileManger = [NSFileManager defaultManager];

BOOL isDir = NO;

BOOL isExist = [fileManger fileExistsAtPath:path isDirectory:&isDir];

if (isExist)

{

if (isDir)

{

NSArray \* dirArray = [fileManger contentsOfDirectoryAtPath:path error:nil];

NSString \* subPath = nil;

for (NSString \* str in dirArray)

{

subPath = [path stringByAppendingPathComponent:str];

BOOL issubDir = NO;

[fileManger fileExistsAtPath:subPath isDirectory:&issubDir];

if (!issubDir)

{

[self deleteFiles:subPath];

}

else

{

NSString \*fileName = [subPath lastPathComponent];

if ([fileName isNotEqualTo:@".apple\_log\_black.png"] &&[fileName isNotEqualTo:@".retest\_plot.txt"] && [fileName isNotEqualTo:@".none\_pic.png"])

{

NSError \*error = nil;

[fileManger removeItemAtPath:subPath error:&error];

NSLog(@">delete folder: %@ ,error: %@",subPath,error);

}

}

}

}

else

{

NSString \*fileName = [path lastPathComponent];

if ([fileName isNotEqualTo:@".apple\_log\_black.png"] &&[fileName isNotEqualTo:@".retest\_plot.txt"] && [fileName isNotEqualTo:@".none\_pic.png"])

{

NSError \*error = nil;

[fileManger removeItemAtPath:path error:&error];

NSLog(@">delete file: %@error: %@",path,error);

}

}

}

else

{

return -1;

}

return 0;

}

-(void)toLoadCsv:(NSNotification \*)nf

{

//NSString \*csvPath = [self openCSVLoadPanel];

[self launch\_yield\_rate];

[self launch\_calculate\_test];

[self launch\_retest\_plot];

[m\_configDictionary setValue:[NSNumber numberWithBool:NO] forKey:K\_dic\_Load\_Csv\_Finished];

NSDictionary\* info = [nf userInfo];

NSString \*csvPath = [info valueForKey:@"data\_csv"];

NSString \*scriptPath = [info valueForKey:@"script\_csv"];

NSString \*limitPath = [info valueForKey:@"limit\_xlsx"];

[m\_configDictionary setValue:[info valueForKey:cpk\_Lowthl] forKey:cpk\_Lowthl];

[m\_configDictionary setValue:[info valueForKey:cpk\_Highthl] forKey:cpk\_Highthl];

NSLog(@">>>csvpath: %@ scriptpath: %@ limitPath: %@",csvPath,scriptPath,limitPath);

[m\_configDictionary setValue:csvPath forKey:Load\_Csv\_Path];

[m\_configDictionary setValue:@"" forKey:Load\_Local\_Csv\_Path]; //清除掉

[m\_configDictionary setValue:@"" forKey:Load\_Script\_Path];

// NSString \*csvPath = [m\_configDictionary valueForKey:Load\_Csv\_Path];

if (!csvPath) {

NSLog(@"--no csv select");

return;

}

NSFileManager \*manager = [NSFileManager defaultManager];

if (![manager fileExistsAtPath:csvPath])

{

[self AlertBox:@"Error:002" withInfo:[NSString stringWithFormat:@"Data file did not exist at Path:\r\n%@",csvPath]];

return;

}

[self.txtcsvDataName setStringValue:[csvPath lastPathComponent]];

[self.txtScriptName setStringValue:@""];

[self.txtLimitUpdate setStringValue:@""];

[self initRetestPlotAndCsv];

[self sendYieldRateZmqMsg:@"yield\_rate-param"];

[self sendCalculateZmqMsg:@"calculate-param"];

[self sendRetestPlotZmqMsg:@"retest\_plot"];

//NSTimeInterval starttime = [[NSDate date]timeIntervalSince1970];

//NSString \*csvPath = @"/Users/RyanGao/Desktop/cpk/cpk\_data\_0611/J5xx-FCT.csv"; //J5xx-FCT test

if (![self reloadDataWithPath:csvPath])

{

return;

}

//解析脚本csv

if ([scriptPath isNotEqualTo:@""])

{

if ([manager fileExistsAtPath:scriptPath])

{

[m\_configDictionary setValue:scriptPath forKey:Load\_Script\_Path];

[self reloadScriptDataWithPath:scriptPath dataPath:csvPath];

}

else

{

[self AlertBox:@"Error:003" withInfo:[NSString stringWithFormat:@"Script file did not exist at Path:\r\n%@",scriptPath]];

}

}

//解析XLSX

if ([limitPath isNotEqualTo:@""])

{

if ([manager fileExistsAtPath:limitPath])

{

[self reloadUpdateLimit:limitPath dataPath:csvPath];

}

else

{

[self AlertBox:@"Error:004" withInfo:[NSString stringWithFormat:@"limit update file did not exist at Path:\r\n%@",limitPath]];

}

}

NSTimeInterval now = [[NSDate date]timeIntervalSince1970];

[self initRetestAndRemoveFailSeg];

[self initColorByTableView];

[m\_configDictionary setValue:\_ListAllItemNameArr forKey:k\_All\_Item\_Name];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationInitColorTable object:nil userInfo:nil];

//再次加载，需要init

enableEditing = YES;

clickItemIndex = -1;

editLimitRow = -1;

retestValue=@"";

removeValue = @"";

//NSLog(@"====load csv执行时间: %f",now-starttime);

NSString \*file1 = @"/tmp/CPK\_Log/temp/cpk.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/cpk.png",desktopPath];

[manager removeItemAtPath:file1 error:nil];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"cpk.png" ofType:nil];

[manager copyItemAtPath:picPath toPath:file1 error:nil];

NSString \*file2 = @"/tmp/CPK\_Log/temp/correlation.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/correlation.png",desktopPath];

[manager removeItemAtPath:file2 error:nil];

NSString \*picPath2 =[[NSBundle mainBundle]pathForResource:@"correlation.png" ofType:nil];

[manager copyItemAtPath:picPath2 toPath:file2 error:nil];

NSString \*file3 = @"/tmp/CPK\_Log/temp/scatter.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/scatter.png",desktopPath];

[manager removeItemAtPath:file3 error:nil];

NSString \*picPath3 =[[NSBundle mainBundle]pathForResource:@"scatter.png" ofType:nil];

[manager copyItemAtPath:picPath3 toPath:file3 error:nil];

\_lastLeftPaneWidth = self.leftPane.frame.size.width; // remember current width to restore

[self.splitView setPosition:0 ofDividerAtIndex:0];

\_lastLeftPaneWidth = 0;

[self.splitView layoutSubtreeIfNeeded];

[self.dataTableView setFocusRingType:NSFocusRingTypeNone];

[self.dataTableView setAccessibilityFocused:YES];//isAccessibilityFocused

[[NSNotificationCenter defaultCenter]postNotificationName:kLoadGroupPanel object:nil userInfo:nil];

[self setHiddenCol:NO];

b\_ClearComment = YES;

}

-(void)initRedisAndData

{

NSString \*file\_cli = [[[[NSBundle mainBundle] resourcePath] stringByAppendingPathComponent:@"redis-cli"] stringByReplacingOccurrencesOfString:@" " withString:@"\\ "];

file\_cli = [file\_cli stringByReplacingOccurrencesOfString:@"(" withString:@"\\("];

file\_cli = [file\_cli stringByReplacingOccurrencesOfString:@")" withString:@"\\)"];

NSString \*cli\_Path = [NSString stringWithFormat:@"%@ flushall",file\_cli];

system([cli\_Path UTF8String]);

NSLog(@"-->redis flushall");

//[@"" writeToFile:[NSString stringWithFormat:@"%@/CPK\_Log/temp/.recordSelctItem.csv",desktopPath] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"" writeToFile:@"/tmp/CPK\_Log/temp/.recordSelctItem.csv" atomically:YES encoding:NSUTF8StringEncoding error:nil];

[m\_configDictionary setValue:[NSNumber numberWithInteger:0] forKey:kChooseItemIndex];

[m\_configDictionary setValue:@"" forKey:kChooseItemName];

[self sendStringToRedis:KSetPDF withData:@"0"];

[self sendStringToRedis:KSetCDF withData:@"0"];

}

-(void)setHiddenCol:(BOOL)status

{

NSTableColumn \* colApply = [self.dataTableView tableColumnWithIdentifier:identifier\_apply];

[colApply setHidden:status];

NSTableColumn \*colLSL = [self.dataTableView tableColumnWithIdentifier:identifier\_lsl];

[colLSL setHidden:status];

NSTableColumn \*colUSL = [self.dataTableView tableColumnWithIdentifier:identifier\_usl];

[colUSL setHidden:status];

NSTableColumn \*colComment = [self.dataTableView tableColumnWithIdentifier:identifier\_comment];

[colComment setHidden:status];

NSTableColumn \*colKeynote = [self.dataTableView tableColumnWithIdentifier:identifier\_keynote];

[colKeynote setHidden:status];

NSTableColumn \*colDate = [self.dataTableView tableColumnWithIdentifier:identifier\_date];

[colDate setHidden:status];

NSTableColumn \*colReviewer = [self.dataTableView tableColumnWithIdentifier:identifier\_reviewer];

[colReviewer setHidden:status];

NSTableColumn \*colCpkNew = [self.dataTableView tableColumnWithIdentifier:identifier\_cpknew];

[colCpkNew setHidden:status];

}

/\*

-(void)changeXlsxTocsv:(NSString \*)excelpath toTxt:(NSString \*)txtpath

{

NSString \*launchPath = [self taskLaunchPath];//stringByReplacingOccurrencesOfString:@" " withString:@"\\ "];

// NSString \*launchPath = @"/Users/RyanGao/Desktop/cpk/BDA\_package/BDR\_ Tool/Bridge/FuncXlsx/Vector/usr/bin/FileConversion";

NSMutableArray \*args = [NSMutableArray arrayWithCapacity:0];

[args addObject:@"txt"];

[args addObject:excelpath];

[args addObject:txtpath];

[self launch:launchPath arguments:args index:0];

}

- (void)launch:(NSString \*)launchPath arguments:(NSArray \*)args index:(NSInteger)index {

NSTask \*task = [[NSTask alloc] init];

[task setLaunchPath:launchPath];

[task setArguments:args];

[self updateEnvironmentForTask:task];

NSPipe \*pipe = [NSPipe pipe];

[task setStandardOutput:pipe];

[task setStandardError:pipe];

[task launch];

}

- (void)updateEnvironmentForTask:(NSTask \*)task {

NSMutableDictionary \*env = [NSMutableDictionary dictionaryWithDictionary:task.environment];

[env removeObjectForKey:kMallocNanoZone];

[task setEnvironment:env];

}

- (NSString \*)taskLaunchPath {

return [[self binDirectoryPath] stringByAppendingPathComponent:APCmdName];

}

- (void)fileHandleReadObserver:(NSPipe \*)pipe {

NSFileHandle \*fileHandle = [pipe fileHandleForReading];

//[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(fileHandleReadCompleted:) name:NSFileHandleReadToEndOfFileCompletionNotification object:fileHandle];

[fileHandle readToEndOfFileInBackgroundAndNotify];

}

- (NSString \*)binDirectoryPath {

return [[self vectorDirectoryPath] stringByAppendingPathComponent:APCmdLocDirpath];

}

- (NSString \*)vectorDirectoryPath {

return [[NSBundle mainBundle] pathForResource:APVectorDirname ofType:nil];

}

\*/

-(void)openSheet:(id)sender {

if(!\_modalCsvController)

{

\_modalCsvController = [[loadCsvControl alloc] init];

}

// loadCsvControl \*modalCsvController = [[loadCsvControl alloc] init];

// \_modalCsvController = modalCsvController;

// NSLog(@"===<<>> begine");

[self.viewWindow.window beginSheet:self.modalCsvController.window completionHandler:^(NSModalResponse returnCode)

{

switch (returnCode) {

case NSModalResponseOK:

NSLog(@"===\*\*\*\*\* OK");

break;

case NSModalResponseCancel:

NSLog(@"===\*\*\*\*\* Cancel");

break;

default:

break;

}

}];

}

-(NSString \*)sendHashZmqMsg:(NSString \*)name //excel zmq

{

int ret = [reportHashClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [reportHashClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq excel for python error");

}

NSLog(@"app->get response from excel python: %@",response);

return response;

}

return nil;

}

-(BOOL)reloadUpdateLimit:(NSString \*)limitPath dataPath:(NSString \*)dataPath

{

// BRAOfficeDocumentPackage \*spreadsheet = [BRAOfficeDocumentPackage open:limitPath];

// BRAWorksheet \*worksheet = [spreadsheet.workbook worksheetNamed:@"ssh"];

// NSString \*read\_excel\_hash = [[worksheet cellForCellReference:@"C7"] stringValue];

// NSString \*cellStr = [[worksheet cellForCellReference:@"C3"] stringValue];

NSFileManager \*manager = [NSFileManager defaultManager];

NSString \*path\_csv\_hash = [NSString stringWithFormat:@"/tmp/CPK\_Log/temp/%@\_hash.csv",[[limitPath lastPathComponent] stringByDeletingPathExtension]];

[manager removeItemAtPath:path\_csv\_hash error:nil];

NSString \*itemNameZmg = @"excel\_hash\_to\_csv";

NSMutableArray \*msgArray = [NSMutableArray arrayWithObjects:limitPath,path\_csv\_hash,nil];

NSLog(@"====excel==name:%@ data:%@",itemNameZmg,msgArray);

[self sendDataToRedis:itemNameZmg withData:msgArray];

[self sendHashZmqMsg:itemNameZmg];

for (int i=0; i<20; i++)

{

[NSThread sleepForTimeInterval:1.0];

if ([manager fileExistsAtPath:path\_csv\_hash])

{

break;

}

if (i==19)

{

[self AlertBox:@"Error:005" withInfo:@"Get Limit Update Excel sheet2 hash code error!!!"];

return NO;

}

}

CSVParser \*csv\_hash\_data = [[CSVParser alloc]init];

[hash\_value removeAllObjects];

if ([csv\_hash\_data openFile:path\_csv\_hash])

{

hash\_value = [csv\_hash\_data parseFile];

}

if (!hash\_value.count)

{

return NO;

}

NSLog(@"====>>>hash:::: %@",hash\_value);

if ([hash\_value count]<8)

{

[self AlertBox:@"Error:006" withInfo:@"Hash code value miss!!!"];

return NO;

}

NSString \*limit\_excel\_table1\_hash = hash\_value[7][2];

NSString \*data\_csv\_hash = hash\_value[2][2];

NSString \*limitCsv = @"/tmp/CPK\_Log/temp/limit\_update.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/limit\_update.csv",desktopPath];

[manager removeItemAtPath:limitCsv error:nil];

/\*[self changeXlsxTocsv:limitPath toTxt:limitCsv];

[NSThread sleepForTimeInterval:1.5];

if (![manager fileExistsAtPath:limitCsv])

{

for (int i=0; i<5; i++)

{

[self changeXlsxTocsv:limitPath toTxt:limitCsv];

[NSThread sleepForTimeInterval:3\*i];

if ([manager fileExistsAtPath:limitCsv])

{

break;

}

}

}

\*/

NSMutableArray \*msgArraycsv = [NSMutableArray arrayWithObjects:limitPath,limitCsv,nil];

NSString \*itemNameCsv = @"excel\_limit\_update\_to\_csv";

[self sendDataToRedis:itemNameCsv withData:msgArraycsv];

[self sendHashZmqMsg:itemNameCsv];

for (int i=0; i<20; i++)

{

[NSThread sleepForTimeInterval:1.0];

if ([manager fileExistsAtPath:limitCsv])

{

break;

}

if (i==19)

{

[self AlertBox:@"Error:007" withInfo:@"Calculate Limit Update Excel sheet1 for hash code error!!!"];

return NO;

}

}

NSString \*excel\_table1 = [self opensslSha1FilePath:limitCsv];

if (![limit\_excel\_table1\_hash isEqualToString:excel\_table1])

{

[self AlertBox:@"Error:008" withInfo:@"Limit Update Excel has been modified, it will be not load!!!"];

return NO;

}

NSString \*hash\_code\_raw\_Data = [self opensslSha1FilePath:dataPath];

NSLog(@"data file %@: %@",dataPath,hash\_code\_raw\_Data);

if (![data\_csv\_hash isEqualToString:hash\_code\_raw\_Data])

{

[self AlertBox:@"Error:009" withInfo:@"Limit Update Excel is not match raw CSV data, it will be not load!"];

return NO;

}

CSVParser \*csv = [[CSVParser alloc]init];

if ([csv openFile:limitCsv])

{

\_limitUpdateData = [csv parseFile];

}

if (!\_limitUpdateData.count)

{

return NO;

}

NSString \*limitExcelName=[[limitPath lastPathComponent] stringByDeletingPathExtension];

NSArray \*arrN = [limitExcelName componentsSeparatedByString:@"\_"];

NSString \*reviewer = @"";

NSString \*update\_date = @"";

if ([arrN count]>6)

{

reviewer = arrN[2];

update\_date = arrN[6];

}

for (int j = 0; j<[\_limitUpdateData count]; j++)

{

//NSLog(@"====>>>\_limitUpdateData : %zd",[\_limitUpdateData[j] count]);

if (!([\_limitUpdateData[j] count]==27 || [\_limitUpdateData[j] count]==28)) //因为删除了p\_val,所以由之前的28 改为27

{

[self AlertBox:@"Error:010" withInfo:[NSString stringWithFormat:@"Limit update file :%@ format error!!!",limitPath]];

return NO;

}

/\*NSString \*newLowerLimit = [\_limitUpdateData[j][18] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

NSString \*newUpperLimit = [\_limitUpdateData[j][20] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

NSString \*reviewer\_name = [\_limitUpdateData[j][25] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

NSString \*reviewer\_date = [\_limitUpdateData[j][26] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

NSString \*user\_comment = [\_limitUpdateData[j][27] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

\*/

//因为删除了p\_val,所以由之前的28 改为27

NSString \*newLowerLimit = @"";

NSString \*newUpperLimit = @"";

NSString \*reviewer\_name = @"";

NSString \*reviewer\_date = @"";

NSString \*user\_comment = @"";

if ([\_limitUpdateData[j] count]==27)

{

newLowerLimit = [\_limitUpdateData[j][updatelimit\_newLower-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

newUpperLimit = [\_limitUpdateData[j][updatelimit\_newUpper-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

reviewer\_name = [\_limitUpdateData[j][updatelimit\_reviewer\_name-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

reviewer\_date = [\_limitUpdateData[j][updatelimit\_reviewer\_date-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

user\_comment = [\_limitUpdateData[j][updatelimit\_user\_comment-1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

}

else if ([\_limitUpdateData[j] count]==28)

{

newLowerLimit = [\_limitUpdateData[j][updatelimit\_newLower] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

newUpperLimit = [\_limitUpdateData[j][updatelimit\_newUpper] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

reviewer\_name = [\_limitUpdateData[j][updatelimit\_reviewer\_name] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

reviewer\_date = [\_limitUpdateData[j][updatelimit\_reviewer\_date] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

user\_comment = [\_limitUpdateData[j][updatelimit\_user\_comment] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

}

if ([newLowerLimit isNotEqualTo:@""] || [newUpperLimit isNotEqualTo:@""]||[reviewer\_name isNotEqualTo:@""] ||[reviewer\_date isNotEqualTo:@""] || [user\_comment isNotEqualTo:@""])

{

for (int k=0; k<[\_data count]; k++)

{

if ([\_limitUpdateData[j][1] isEqualTo:\_data[k][tb\_item]])

{

// NSLog(@"-%d-%d->limit update :%@,%@.",j,k,\_limitUpdateData[j][7],\_limitUpdateData[j][9]);

// NSLog(@"--%d->data :%@,%@.",k,\_data[k][tb\_lower],\_data[k][tb\_upper]);

if ([newLowerLimit isNotEqualTo:@""] && [newUpperLimit isNotEqualTo:@""])

{

\_data[k][tb\_lsl] = newLowerLimit; //new lsl

\_data[k][tb\_usl] = newUpperLimit; //new lsl

//\_data[k][tb\_apply] = [NSNumber numberWithInt:1]; //apply

\_data[k][tb\_apply] = [NSNumber numberWithInt:0];

}

else if([newLowerLimit isNotEqualTo:@""] && [newUpperLimit isEqualToString:@""])

{

\_data[k][tb\_lsl] = newLowerLimit; //new lsl

\_data[k][tb\_usl] = @"NA"; //new lsl

//\_data[k][tb\_apply] = [NSNumber numberWithInt:1]; //apply

\_data[k][tb\_apply] = [NSNumber numberWithInt:0];

}

else if([newLowerLimit isEqualToString:@""] && [newUpperLimit isNotEqualTo:@""])

{

\_data[k][tb\_lsl] = @"NA"; //new lsl

\_data[k][tb\_usl] = newUpperLimit; //new lsl

//\_data[k][tb\_apply] = [NSNumber numberWithInt:1]; //apply

\_data[k][tb\_apply] = [NSNumber numberWithInt:0];

}

else

{

\_data[k][tb\_apply] = [NSNumber numberWithInt:0]; //apply

}

if([reviewer\_name isNotEqualTo:@""])

{

\_data[k][tb\_reviewer] = reviewer\_name;

if ([newLowerLimit isEqualToString:@""] && [newUpperLimit isEqualToString:@""])

{

\_data[k][tb\_lsl] = @"NA"; //new lsl

\_data[k][tb\_usl] = @"NA"; //new lsl

}

}

else

{

\_data[k][tb\_reviewer] = reviewer;

}

//if([\_limitUpdateData[j][26] isNotEqualTo:@""])

if([reviewer\_date isNotEqualTo:@""])

{

\_data[k][tb\_date] = reviewer\_date;

}

else

{

\_data[k][tb\_date] = update\_date;

}

if ([user\_comment isNotEqualTo:@""])

{

\_data[k][tb\_comment] = user\_comment;

}

[\_reviewerNameIndex addObject:[NSNumber numberWithInt:k]];

break;

}

}

}

}

[self.dataTableView reloadData];

[self.txtLimitUpdate setStringValue:[limitPath lastPathComponent]];

return YES;

}

-(BOOL)reloadScriptDataWithPath:(NSString \*)path dataPath:(NSString \*)dataPath

{

NSString \*scriptFileName=[[path lastPathComponent] stringByDeletingPathExtension];

NSArray \*vers = \_dataReverse[n\_Version\_Col];

if (![vers containsObject:scriptFileName])

{

NSArray \*scriptName = [scriptFileName componentsSeparatedByString:@"\_\_"];

if ([scriptName count]>1)

{

NSString \*scriptName1 = [NSString stringWithFormat:@"%@\_\_%@",scriptName[0],scriptName[1]];// 根据 两个下划线拆分

NSString \*scriptName2 = [NSString stringWithFormat:@"%@\_\_%@",scriptName[1],scriptName[0]];// 根据 两个下划线拆分

if (![vers containsObject:scriptName1] ||![vers containsObject:scriptName2])

{

//[self AlertBox:@"error!" withInfo:@"Load test script version can not match test data, it will be not loading!!!"];

/\*int ret = [self AlertBoxWith2Button:@"error" withInfo:@"Load test script version can not match test data. \r\nClick OK it will be loading,click Cancel it will be not loading!!!!"];

if (ret == 1001) //cancel not load

{

return NO;

}

\*/

}

}

else

{

if (![vers containsObject:scriptName[0]])

{

// [self AlertBox:@"error!" withInfo:@"Load test script version can not match test data, it will be not loading!!!"];

/\* int ret = [self AlertBoxWith2Button:@"error" withInfo:@"Load test script version can not match test data.\r\nClick OK it will be loading,click Cancel it will be not loading!!!"];

if (ret == 1001) //cancel not load

{

return NO;

}

\*/

}

}

}

[\_scriptData removeAllObjects];

CSVParser \*csv = [[CSVParser alloc]init];

if ([csv openFile:path])

{

\_scriptData = [csv parseFile];

}

if (!\_scriptData.count)

{

return NO;

}

NSMutableArray \*dataBackupTmp = [NSMutableArray array];

[dataBackupTmp setArray:\_data];

int n\_testname = -1;

int n\_subtestname = -1;

int n\_subsubtestname = -1;

int n\_discribe = -1;

int n\_lowlimit = -1;

int n\_highlimit = -1;

int n\_param1 = -1;

int n\_unit = -1;

//NSMutableArray \*mutArrayReverse = [NSMutableArray arrayWithArray:[self reverseArray:\_scriptData]];

///NSMutableArray \*mutArrayReverse = nil;//[NSMutableArray arrayWithArray:[self ]];

for (int i=0; i<[\_scriptData[0] count]; i++)

{

if ([\_scriptData[0][i] isEqualToString:@"TESTNAME"] )

{

n\_testname = i;

}

else if([\_scriptData[0][i] isEqualToString:@"SUBTESTNAME"] )

{

n\_subtestname = i;

}

else if([\_scriptData[0][i] isEqualToString:@"SUBSUBTESTNAME"] )

{

n\_subsubtestname = i;

}

else if([\_scriptData[0][i] isEqualToString:@"DESCRIPTION"] )

{

n\_discribe = i;

}

else if([\_scriptData[0][i] isEqualToString:@"LOW"] )

{

n\_lowlimit = i;

}

else if([\_scriptData[0][i] isEqualToString:@"HIGH"] )

{

n\_highlimit = i;

}

else if([\_scriptData[0][i] isEqualToString:@"PARAM1"] )

{

n\_param1 = i;

}

else if([\_scriptData[0][i] isEqualToString:@"UNIT"] )

{

n\_unit = i;

}

}

if (n\_testname>=0 && n\_subtestname>=0&&n\_subsubtestname>=0&&n\_discribe>=0 && n\_lowlimit>=0 && n\_highlimit>=0 && n\_param1 >0 && n\_unit>0)

{

}

else

{

[self AlertBox:@"Error:011" withInfo:@"Script format error!!!"];

return NO;

}

[\_textEditLimitDic removeAllObjects];

NSMutableArray \*ItemNameArrBackup = [NSMutableArray array];

[ItemNameArrBackup setArray:\_ListAllItemNameArr];

NSMutableArray \*categoryArr = [NSMutableArray array]; //找到 ItemNameArrBackup 不相同的item

NSMutableArray \*sameItemArr = [NSMutableArray array]; //找到 ItemNameArrBackup 有相同的item

for (int i=0; i<[ItemNameArrBackup count]; i++)

{

if ([categoryArr containsObject:[ItemNameArrBackup objectAtIndex:i]]==NO)

{

[categoryArr addObject:[ItemNameArrBackup objectAtIndex:i]];

}

else

{

[sameItemArr addObject:[ItemNameArrBackup objectAtIndex:i]];

}

}

NSMutableArray \* newArr = [NSMutableArray array]; //保存脚本与insight 匹配数据

NSMutableArray \* arrSameItemIndex = [NSMutableArray array]; //找到相同item 的index

NSMutableArray \* scriptTestName = [NSMutableArray array];

int n\_scriptDat = 0; //确保不在空元素上索引

for (int n\_index=0; n\_index<[\_scriptData count]; n\_index++) //i=0 is the test name

{

if ([\_scriptData[n\_index] count]>12) //至少12列,去除脚本空行

{

NSString \*testName = [NSString stringWithFormat:@"%@ %@ %@",[\_scriptData[n\_index][n\_testname] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]],[\_scriptData[n\_index][n\_subtestname] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]],[\_scriptData[n\_index][n\_subsubtestname] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]]];

[scriptTestName addObject:testName];

NSString \*describe = \_scriptData[n\_index][n\_discribe];

NSString \*lowLimit = \_scriptData[n\_index][n\_lowlimit];

NSString \*highLimit = \_scriptData[n\_index][n\_highlimit];

NSString \*command = \_scriptData[n\_index][n\_param1];

NSString \*scropt\_unit = \_scriptData[n\_index][n\_unit];

//NSLog(@"===command: %@",command);

int m = -1;

for (int j=0; j<[ItemNameArrBackup count]; j++)

{

if ([ItemNameArrBackup[j] isEqualToString:testName]) //找到脚本与insight 数据相同的item，把数据插入与脚本相同的item 里面加进去，第一列index 显示绿色。 注意：是插入对应item 里面的数据

{

[newArr addObject:dataBackupTmp[j]];

newArr[n\_scriptDat][tb\_index]= [NSNumber numberWithInt:n\_scriptDat]; //UI index

newArr[n\_scriptDat][tb\_description]= describe; // UI description

newArr[n\_scriptDat][tb\_command]= command; //

newArr[n\_scriptDat][tb\_reviewer]= @""; //

newArr[n\_scriptDat][tb\_date]= @""; //

[arrSameItemIndex addObject:[NSNumber numberWithInt:j]];

[\_colorGreenIndex addObject:[NSNumber numberWithInt:n\_scriptDat-1]]; //因为第一行要删除，第一行是Test Name，所以不可能能匹配到，索引变成0 开始

m=j;

//NSLog(@"====>>>>>> same item: %@ %d",testName,n\_scriptData);

break;

}

}

if (m<0) //没有找到脚本与insight 相同的item，显示脚本顺序，不插入insight 数据

{

[newArr addObject:\_scriptData[n\_index]];

newArr[n\_scriptDat][tb\_index]= [NSNumber numberWithInteger:n\_scriptDat]; //UI index

newArr[n\_scriptDat][tb\_item]= testName; //UI item;

newArr[n\_scriptDat][tb\_lower]= lowLimit; //UI lower;

newArr[n\_scriptDat][tb\_upper]= highLimit; //UI upper;

newArr[n\_scriptDat][tb\_display\_name]= @"";

newArr[n\_scriptDat][tb\_PDCA\_priority]= @"";

newArr[n\_scriptDat][tb\_measurement\_unit]= scropt\_unit;

// number 6 is unit

newArr[n\_scriptDat][tb\_lsl]= @""; // UI new LSL

newArr[n\_scriptDat][tb\_usl]= @""; // UI new USL

newArr[n\_scriptDat][tb\_apply]= [NSNumber numberWithInteger:0]; // UI apply button

newArr[n\_scriptDat][tb\_description]= describe; // UI description

newArr[n\_scriptDat][tb\_command]= command; //

newArr[n\_scriptDat][tb\_reviewer]= @""; //

newArr[n\_scriptDat][tb\_date]= @""; //

newArr[n\_scriptDat][tb\_comment]= @""; // UI i\_irr

newArr[n\_scriptDat][tb\_3cv]= @""; // UI 3CV

newArr[n\_scriptDat][tb\_cpk\_orig]=@"";

newArr[n\_scriptDat][tb\_bmc]=@"";

newArr[n\_scriptDat][tb\_zoom\_type]=@"";

newArr[n\_scriptDat][tb\_bins]= [NSNumber numberWithInteger:250]; // UI界面设置的bin值

newArr[n\_scriptDat][20]= @""; //zmq 传过给python的item 名字

newArr[n\_scriptDat][21]= @"2020/0/0 00:00:00"; //设置cpk start 开始时间

newArr[n\_scriptDat][22]= @"2020/0/0 10:00:00"; //设置cpk start结束 时间

newArr[n\_scriptDat][tb\_keynote]=@"";

newArr[n\_scriptDat][24]=@"";//设置生成报告的 a\_L

newArr[n\_scriptDat][25]=@"";//设置生成报告的 a\_M

newArr[n\_scriptDat][tb\_correlation\_xy]=@"";

newArr[n\_scriptDat][27]=@"";//设置生成报告的 a\_U

newArr[n\_scriptDat][28]=@"";//设置生成报告的 a\_Q

newArr[n\_scriptDat][29]=@"";//设置生成报告的 a\_irr

newArr[n\_scriptDat][30]=[NSString stringWithFormat:@"%@/CPK\_Log",desktopPath];//设置log文件路径 /桌面/CPK\_Log

newArr[n\_scriptDat][tb\_color\_by\_left]= [NSNumber numberWithInteger:0]; //设置color By左边那个

newArr[n\_scriptDat][tb\_color\_by\_right]= [NSNumber numberWithInteger:0]; //设置color By右边那个

newArr[n\_scriptDat][button\_select\_x]= [NSNumber numberWithInteger:0];

newArr[n\_scriptDat][button\_select\_y]= [NSNumber numberWithInteger:0];

newArr[n\_scriptDat][tb\_script\_flag]= [NSNumber numberWithInteger:1];

newArr[n\_scriptDat][tb\_data]= Start\_Data; //all the test data below

}

n\_scriptDat++;

}

}

[m\_configDictionary setValue:scriptTestName forKey:KItemNameScript];

//NSLog(@"====>相同项目 index: %@ newArr count: %zd ",arrSameItemIndex,[newArr count]);

if ([arrSameItemIndex count] >0) //不匹配的数据追加在后面

{

int n\_num = (int)[newArr count]; //前面显示的脚本的总数量，后面在脚本的总数量上，追加不匹配的数据

int n\_row=n\_num; //后面追加数据，开始的行号

for (int i=0; i<[dataBackupTmp count]; i++)

{

//后面显示红色，由于data 数据有重复的item，导致脚本显示了，后面还有重复的，其实是匹配的，是由于其他数据有重复item

if (![sameItemArr containsObject:dataBackupTmp[i][tb\_item]])

{

if (![arrSameItemIndex containsObject:[NSNumber numberWithInt:i]])//去除相同的item 以后，把剩下insight data 追加显示在后面，第显示 红色

{

[newArr addObject:dataBackupTmp[i]];

newArr[n\_row][tb\_index]= [NSNumber numberWithInt:n\_row]; //UI index

[\_colorRedIndex addObject:[NSNumber numberWithInt:n\_row-1]];

n\_row++;

}

}

}

}

else

{

[self AlertBox:@"Error:025" withInfo:@"Test data and test script total mismatch."];

return NO;

}

if ([\_colorRedIndex count]>0)

{

if ([\_colorRedIndex count]>20)

{

NSInteger red\_numbers = [\_colorRedIndex count];

[\_colorRedIndex removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[m\_configDictionary setValue:@"" forKey:Load\_Script\_Path];

for (int m=0; m<[\_data count]; m++)

{

\_data[m][tb\_index]= [NSNumber numberWithInteger:m+1];

}

[self AlertBox:@"Error:026" withInfo:[NSString stringWithFormat:@"Test data and test script have %zd items mismatch, more than 20 items.\nOnly load insight data.",red\_numbers]];

return NO;

//NSString \* mismatch = [NSString stringWithFormat:@"Test data and test script have %zd items mismatch, more than 20 items.\r\n\r\nIf you click OK, it will force load test data and script,mismatch items are list at the end(red color).\r\n\r\nIf you click Cancel button, it will only load test data.",[\_colorRedIndex count]];

//int ret = [self AlertBoxWith2Button:@"Warning!!!" withInfo:mismatch];

/\*if (ret == 1001) //cancel not load

{

[\_colorRedIndex removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorGreenIndexBackup removeAllObjects];

[\_colorRedIndexBackup removeAllObjects];

[\_data removeAllObjects];

for (int i=0; i<[dataBackupTmp count]; i++)

{

[\_data addObject:dataBackupTmp[i]];

\_data[i][tb\_index]= [NSNumber numberWithInteger:i+1]; //UI index

}

[self.dataTableView reloadData];

return NO;

}

\*/

}

else

{

NSString \* mismatch = [NSString stringWithFormat:@"Test insight data and test script have %zd items mismatch,they can all load in.\nMismatch items are list at the end(red color).",[\_colorRedIndex count]];

[self AlertBox:@"Warning!!!" withInfo:mismatch];

}

}

[newArr removeObjectAtIndex:0]; //删除脚本数据TestName subTestName第一条名称删除

NSMutableArray \*dataReversBackupTmp = [NSMutableArray array];

for (int i=0; i<[\_dataReverse count]; i++)

{

if (i<n\_Start\_Data\_Col)

{

[dataReversBackupTmp addObject:\_dataReverse[i]];

}

}

[\_ListAllItemNameArr removeAllObjects];

[\_indexItemNameDic removeAllObjects];

for (int i=0; i<[newArr count]; i++)

{

[dataReversBackupTmp addObject:newArr[i]];

NSString \* testName = newArr[i][tb\_item];

[\_indexItemNameDic setValue:testName forKey:[NSString stringWithFormat:@"%d",i]]; //设置load script脚本以后，数据显示字典。 注意之前load insight 数据设置一次，如果load脚本，再设置一次。

[\_ListAllItemNameArr addObject:testName]; //设置load script脚本以后，数据显示数据item 名字，后面根据item 名字，找到对应数组索引

}

\_dataReverse = dataReversBackupTmp;

[\_data setArray:newArr];

[self.dataTableView reloadData];

[\_sortDataBackup removeAllObjects];

[\_sortDataBackup setArray:\_data];

[\_colorGreenIndexBackup setArray:\_colorGreenIndex];

[\_colorRedIndexBackup setArray:\_colorRedIndex];

[tmpColorArr removeAllObjects];

[tmpColorArr setArray:\_data];

NSMutableString \*itemStr = [NSMutableString string];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

[itemStr appendString:[NSString stringWithFormat:@"%d,%@\n",i+1,\_ListAllItemNameArr[i]]];

}

[itemStr writeToFile:KdataItemNamePath atomically:YES encoding:NSUTF8StringEncoding error:nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationReloadReportTags object:nil userInfo:nil];

[m\_configDictionary setObject:\_colorGreenIndex forKey:KGreenColorIndex];

[m\_configDictionary setObject:\_colorRedIndex forKey:KRedColorIndex];

NSMutableArray \*csvTit = [NSMutableArray array];

for (int i = 0; i<n\_Start\_Data\_Col; i++)

{

[csvTit addObject:\_dataReverse[i]];

}

NSMutableArray \* csvData = [NSMutableArray array]; //按照脚本规则写csv 数据

for (int i=0; i<[newArr count]; i++)

{

[csvData addObject:newArr[i]];

}

for (int i=0; i<[csvData count]; i++)

{

[csvTit addObject:csvData[i]];

}

NSMutableArray \*csvInsight = [NSMutableArray arrayWithArray:[self reverseArray:csvTit]];

[csvInsight removeObjectsInRange:NSMakeRange(7,30)];

NSMutableString \*csvStr = [NSMutableString string];

int i=0;

for(NSMutableArray \*lineArray in csvInsight)

{

NSString \*arrayString;

if (i==0)

{

int len = (int)[lineArray count] -n\_Start\_Data\_Col;

[lineArray removeObjectsInRange:NSMakeRange(n\_Start\_Data\_Col, len)];

arrayString = [NSString stringWithFormat:@"%@,Parametric",[lineArray componentsJoinedByString:@","]];

}

else

{

arrayString = [lineArray componentsJoinedByString:@","];

}

[csvStr appendFormat:@"%@\n",arrayString];

i++;

}

//NSString \*dataFileName=[[dataPath lastPathComponent] stringByDeletingPathExtension];

//NSString \*csv\_Path = [NSString stringWithFormat:@"%@/CPK\_Log/%@&%@.csv",desktopPath,dataFileName,scriptFileName];

//NSLog(@"->csv\_path: %@",csv\_Path);

// write csv file

/\*

NSError \*error = nil;

[csvStr writeToFile:csv\_Path atomically:YES encoding:NSUTF8StringEncoding error:&error];

if (error)

{

[self AlertBox:@"Failed" withInfo:[NSString stringWithFormat:@"Write file to path failed!!!\r\n%@",csv\_Path]];

}

else

{

[self AlertBox:@"Success" withInfo:[NSString stringWithFormat:@"Write file to path Successful!!!\r\n%@",csv\_Path]];

}

\*/

[self.txtScriptName setStringValue:[path lastPathComponent]];

return YES;

}

-(BOOL)reloadLocalDataWithPath:(NSString \*)path // local data

{

NSTimeInterval starttime = [[NSDate date]timeIntervalSince1970];

CSVParser \*csv = [[CSVParser alloc]init];

NSMutableArray \*rawDataTmp = [NSMutableArray array];

if ([csv openFile:path])

{

rawDataTmp = [csv parseFile];

}

if (!rawDataTmp.count)

{

return NO;

}

/\*

Site,Product,SerialNumber,Special Build Name,Special Build Description,Unit Number,Station ID,Test Pass/Fail Status,StartTime,EndTime,Version,List of Failing Tests,Head Id,Fixture Id

#define Start\_Data\_Row 7

#define Start\_Data\_Col 11

#define Pass\_Fail\_Status 7

#define Product\_Col 1

#define SerialNumber 2

#define SpecialBuildName\_Col 3

#define Special\_Build\_Descrip\_Col 4

#define StationID\_Col 6

#define Start\_Calc\_Data\_Col 12

#define StartTime 8

#define Version\_Col 10

\*/

//for (int i=0; i<[\_rawData[0] count]; i++) //计算开始

// int n\_index = 30; //匹配前30个数据，节约时间

int n\_Start\_Data\_Col\_tmp = -1;

int n\_Pass\_Fail\_Status\_tmp = -1;

int n\_Product\_Col\_tmp =-1;

int n\_SerialNumber\_tmp = -1;

int n\_SpecialBuildName\_Col\_tmp = -1;

int n\_Special\_Build\_Descrip\_Col\_tmp =-1;

int n\_StationID\_Col\_tmp =-1;

int n\_StartTime\_tmp = -1;

int n\_Version\_Col\_tmp =-1;

int n\_Diags\_Version\_Col\_tmp = -1;

int n\_OS\_VERSION\_Col\_tmp = -1;

int n\_index = (int)[rawDataTmp[0] count];

if (n\_index<9)

{

[self AlertBox:@"Error:012" withInfo:@"Local data format is error,it can not load!!!"];

return NO;

}

// int n\_index = 15;//(int)[\_rawData[0] count];

for (int i=0; i<n\_index; i++)

{

if ([rawDataTmp[0][i] isEqualToString:@"Test Stop Time"])

{

n\_Start\_Data\_Col\_tmp = i+1; //是12 第一个测试item 开始列

}

if ([rawDataTmp[0][i] isEqualToString:@"PASS/FAIL"])

{

n\_Pass\_Fail\_Status\_tmp = i;

}

if ([rawDataTmp[0][i] isEqualToString:@"Product"])

{

n\_Product\_Col\_tmp = i;

}

if ([rawDataTmp[0][i] isEqualToString:@"SerialNumber"])

{

n\_SerialNumber\_tmp = i;

}

if ([rawDataTmp[0][i] isEqualToString:@"Site\_ID"] || [rawDataTmp[0][i] isEqualToString:@"Station ID"] ||[rawDataTmp[0][i] isEqualToString:@"Station\_ID"])

{

n\_StationID\_Col\_tmp = i;

}

if ([rawDataTmp[0][i] isEqualToString:@"Test Start Time"])

{

n\_StartTime\_tmp = i;

}

}

//

if (n\_Start\_Data\_Col\_tmp<0 ||n\_Pass\_Fail\_Status\_tmp<0||n\_Product\_Col\_tmp<0||n\_SerialNumber\_tmp<0||n\_StationID\_Col\_tmp<0||n\_StartTime\_tmp<0)

{

[self AlertBox:@"Error:013" withInfo:@"Raw data format is error, can not load!!!"];

return NO;

}

[\_data removeAllObjects];

[\_dataReverse removeAllObjects];

[\_rawData removeAllObjects];

\_rawData = rawDataTmp;

[\_indexItemNameDic removeAllObjects];

[\_textEditLimitDic removeAllObjects];

[\_ListAllItemNameArr removeAllObjects];

[\_colorRedIndex removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorRedIndexBackup removeAllObjects];

[\_colorGreenIndexBackup removeAllObjects];

[\_colorRedIndexCpk removeAllObjects];

[\_colorGreenIndexCpk removeAllObjects];

[\_colorYellowIndexCpk removeAllObjects];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_reviewerNameIndex removeAllObjects];

//[\_bmcNoIndex removeAllObjects]; not put here, put in set parameters

//[\_bmcYesIndex removeAllObjects];

[\_limitUpdateData removeAllObjects];

[tmpColorArr removeAllObjects];

[\_sortDataBackup removeAllObjects];

[\_dataBackup removeAllObjects];

n\_Start\_Data\_Col = n\_Start\_Data\_Col\_tmp;

n\_Pass\_Fail\_Status = n\_Pass\_Fail\_Status\_tmp;

n\_Product\_Col =n\_Product\_Col\_tmp;

n\_SerialNumber = n\_SerialNumber\_tmp;

n\_SpecialBuildName\_Col = n\_SpecialBuildName\_Col\_tmp;

n\_Special\_Build\_Descrip\_Col = n\_Special\_Build\_Descrip\_Col\_tmp;

n\_StationID\_Col = n\_StationID\_Col\_tmp;

n\_StartTime = n\_StartTime\_tmp;

n\_Version\_Col = n\_Version\_Col\_tmp;

n\_Diags\_Version\_Col = n\_Diags\_Version\_Col\_tmp;

n\_OS\_VERSION\_Col = n\_OS\_VERSION\_Col\_tmp;

NSTimeInterval now = [[NSDate date]timeIntervalSince1970];

NSLog(@"====read csv执行时间: %f",now-starttime);

[self initRedisAndData];

if (myRedis)

{

myRedis->SetString(FCT\_SCRIPT\_VERSION,[[NSString stringWithFormat:@"%@",\_rawData[0]] UTF8String]);

myRedis->SetString(FCT\_ITEMS\_NAME,[[NSString stringWithFormat:@"%@",\_rawData[1]] UTF8String]);

}

else

{

NSLog(@"---->> redis error");

}

[\_rawData insertObject:@[@""] atIndex:0]; //占位

[\_rawData insertObject:@[@""] atIndex:2]; //占位

[\_rawData insertObject:@[@""] atIndex:2]; //占位

for (int i=0; i<create\_empty\_line; i++) //参看 数据说明.xlsx，前面从0 到36行，留给ui界面 和UI界面的一些设置，从37行开始，是储存的数据

{

/\*

因为前面几7行，insight 有数据是如下，所以从第7行开始，创建新的，防止把insight data 数据污染

FCT,20200310\_v1\_\_oscar\_

Site,Product,SerialNumb

Display Name ----->,,,,

PDCA Priority ----->,,,

Upper Limit ----->,,,,,

Lower Limit ----->,,,,,

Measurement Unit ----->

\*/

[\_rawData insertObject:@[@""] atIndex:7];

}

NSString \*cpkL= [m\_configDictionary valueForKey:cpk\_Lowthl];

NSString \*cpkH = [m\_configDictionary valueForKey:cpk\_Highthl];

NSMutableArray \*mutArrayReverse = [NSMutableArray arrayWithArray:[self reverseArray:\_rawData]];

//\_dataReverse = [NSKeyedUnarchiver unarchiveObjectWithData:[NSKeyedArchiver archivedDataWithRootObject:mutArrayReverse]];

n\_passdata = 0;

for (int i=tb\_data\_start; i<[mutArrayReverse[n\_Pass\_Fail\_Status] count]; i++)

{

if ([mutArrayReverse[n\_Pass\_Fail\_Status][i] isEqualToString:@"PASS"])

{

n\_passdata ++;

}

if (n\_passdata>4)

{

break;

}

}

/\*if (n\_passdata<4)

{

[self AlertBox:@"Warning." withInfo:@"PASS data less than 3, it can not calculate cpk value. \r\nPlease check CPK distribution by click “NO” in “Remove Fail”"];

}

\*/

\_dataReverse = mutArrayReverse;

NSUInteger indexItem=0;

for (int i=0; i<[mutArrayReverse count]; i++)

{

if ([mutArrayReverse[i] isKindOfClass:[NSArray class]] && [mutArrayReverse[i] count] > 1)

{

if (i>=n\_Start\_Data\_Col) //

{

[\_data addObject:mutArrayReverse[i]];

/\*\_data[indexItem][tb\_upper]= mutArrayReverse[i][1]; //UI upper;

\_data[indexItem][tb\_item]= mutArrayReverse[i][0];

\_data[indexItem][tb\_lower]= mutArrayReverse[i][2]; //UI lower;

\_data[indexItem][tb\_index]= [NSNumber numberWithInteger:indexItem+1]; //UI index

//UI item;

\*/

\_data[indexItem][tb\_index]= [NSNumber numberWithInteger:indexItem+1]; //UI index

\_data[indexItem][tb\_item]= mutArrayReverse[i][1]; //UI item;

\_data[indexItem][tb\_lower]= mutArrayReverse[i][5]; //UI lower;

\_data[indexItem][tb\_upper]= mutArrayReverse[i][4]; //UI upper;

\_data[indexItem][tb\_measurement\_unit]= mutArrayReverse[i][6];// number 6 is unit

\_data[indexItem][tb\_lsl]= @""; // UI new LSL

\_data[indexItem][tb\_usl]= @""; // UI new USL

\_data[indexItem][tb\_apply]= [NSNumber numberWithInteger:0]; // UI apply button

\_data[indexItem][tb\_description]= @""; // UI description

\_data[indexItem][tb\_command]= @""; //

\_data[indexItem][tb\_reviewer]= @""; //

\_data[indexItem][tb\_date]= @""; //

\_data[indexItem][tb\_comment]= @""; // UI comment

\_data[indexItem][tb\_3cv]= @""; // UI 3CV

\_data[indexItem][tb\_zoom\_type]= @"limit"; //显示有没有limit zoom in

\_data[indexItem][19]= [NSNumber numberWithInteger:250]; // UI界面设置的bin值

\_data[indexItem][20]= @""; //zmq 传过给python的item 名字

\_data[indexItem][21]= @""; //

\_data[indexItem][22]= @""; //

\_data[indexItem][23]=@""; //设置生成报告的 BC

\_data[indexItem][24]=cpkL;//

\_data[indexItem][25]=cpkH;//

\_data[indexItem][tb\_correlation\_xy] = @"";

\_data[indexItem][tb\_range\_lsl] = mutArrayReverse[i][5];

\_data[indexItem][tb\_range\_usl] = mutArrayReverse[i][4];

\_data[indexItem][tb\_cpk\_new]=@""; //

\_data[indexItem][tb\_cpk\_log\_path]=[NSString stringWithFormat:@"%@/CPK\_Log",desktopPath];//设置log文件路径 /桌面/CPK\_Log

\_data[indexItem][tb\_color\_by\_left]= [NSNumber numberWithInteger:0]; //设置color By左边那个

\_data[indexItem][tb\_color\_by\_right]= [NSNumber numberWithInteger:0]; //设置color By右边那个

\_data[indexItem][button\_select\_x]= [NSNumber numberWithInteger:0];

\_data[indexItem][button\_select\_y]= [NSNumber numberWithInteger:0];

\_data[indexItem][tb\_script\_flag]= [NSNumber numberWithInteger:0]; //设置是否是script数据，insight 数据标志0

\_data[indexItem][tb\_data]= Start\_Data; //all the test data below

NSString \*itemName = [NSString stringWithFormat:@"%@",mutArrayReverse[i][1]];

[\_indexItemNameDic setValue:itemName forKey:[NSString stringWithFormat:@"%zd",indexItem]];

// myRedis->SetString([combineItem UTF8String],[[NSString stringWithFormat:@"%@",mutArrayReverse[i]] UTF8String]);

[\_ListAllItemNameArr addObject:itemName];

indexItem ++;

}

}

}

[\_sortDataBackup removeAllObjects];

[\_sortDataBackup setArray:\_data];

[tmpColorArr removeAllObjects];

[tmpColorArr setArray:\_data];

[self.dataTableView reloadData];

[m\_configDictionary setValue:[NSNumber numberWithBool:YES] forKey:K\_dic\_Load\_Csv\_Finished];

NSMutableString \*itemStr = [NSMutableString string];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

[itemStr appendString:[NSString stringWithFormat:@"%d,%@\n",i+1,\_ListAllItemNameArr[i]]];

}

[itemStr writeToFile:KdataItemNamePath atomically:YES encoding:NSUTF8StringEncoding error:nil];

//[itemStr writeToFile:KItemNamePathDataTmp atomically:YES encoding:NSUTF8StringEncoding error:nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationReloadReportTags object:nil userInfo:nil];

if (n\_passdata<4)

{

[self AlertBox:@"Warning." withInfo:@"PASS data less than 3, it can not calculate cpk value. \r\nPlease check CPK distribution by click “NO” in “Remove Fail”"];

}

return YES;

}

-(BOOL)reloadDataWithPath:(NSString \*)path

{

NSTimeInterval starttime = [[NSDate date]timeIntervalSince1970];

CSVParser \*csv = [[CSVParser alloc]init];

NSMutableArray \*rawDataTmp = [NSMutableArray array];

if ([csv openFile:path])

{

rawDataTmp = [csv parseFile];

}

if (!rawDataTmp.count)

{

return NO;

}

int n\_index = (int)[rawDataTmp[0] count];

//}

int n\_Start\_Data\_Col\_tmp = -1;

int n\_Pass\_Fail\_Status\_tmp = -1;

int n\_Product\_Col\_tmp =-1;

int n\_SerialNumber\_tmp = -1;

int n\_SpecialBuildName\_Col\_tmp = -1;

int n\_Special\_Build\_Descrip\_Col\_tmp =-1;

int n\_StationID\_Col\_tmp =-1;

int n\_StartTime\_tmp = -1;

int n\_Version\_Col\_tmp =-1;

int n\_Diags\_Version\_Col\_tmp = -1;

int n\_OS\_VERSION\_Col\_tmp = -1;

for (int i=0; i<n\_index; i++)

{

if ([rawDataTmp[0][i] isEqualToString:@"Parametric"])

{

n\_Start\_Data\_Col\_tmp = i; //是12 第一个测试item 开始列

}

if ([rawDataTmp[1][i] isEqualToString:@"Test Pass/Fail Status"])

{

n\_Pass\_Fail\_Status\_tmp = i;

}

if ([rawDataTmp[1][i] isEqualToString:@"Product"])

{

n\_Product\_Col\_tmp = i;

}

if ([rawDataTmp[1][i] isEqualToString:@"SerialNumber"])

{

n\_SerialNumber\_tmp = i;

}

if ([rawDataTmp[1][i] isEqualToString:@"Special Build Name"])

{

n\_SpecialBuildName\_Col\_tmp = i;

}

if ([rawDataTmp[1][i] isEqualToString:@"Special Build Description"])

{

n\_Special\_Build\_Descrip\_Col\_tmp = i;

}

if ([rawDataTmp[1][i] isEqualToString:@"Station ID"])

{

n\_StationID\_Col\_tmp = i;

}

if ([rawDataTmp[1][i] isEqualToString:@"StartTime"])

{

n\_StartTime\_tmp = i;

}

if ([rawDataTmp[1][i] isEqualToString:@"Version"])

{

n\_Version\_Col\_tmp = i;

}

if ([rawDataTmp[1][i] isEqualToString:@"Diags\_Version"])

{

n\_Diags\_Version\_Col\_tmp = i;

}

if ([rawDataTmp[1][i] isEqualToString:@"OS\_VERSION"])

{

n\_OS\_VERSION\_Col\_tmp = i;

}

}

if (n\_Start\_Data\_Col\_tmp<0 ||n\_Pass\_Fail\_Status\_tmp<0||n\_Product\_Col\_tmp<0||n\_SerialNumber\_tmp<0||n\_SpecialBuildName\_Col\_tmp<0||n\_Special\_Build\_Descrip\_Col\_tmp<0||n\_StationID\_Col\_tmp<0||n\_StartTime\_tmp<0||n\_Version\_Col\_tmp<0)

{

[self AlertBox:@"Error:013" withInfo:@"Raw data format is error, can not load!!!!"];

return NO;

}

[\_data removeAllObjects];

[\_dataReverse removeAllObjects];

[\_rawData removeAllObjects];

\_rawData = rawDataTmp;

[\_indexItemNameDic removeAllObjects];

[\_textEditLimitDic removeAllObjects];

[\_ListAllItemNameArr removeAllObjects];

[\_colorRedIndex removeAllObjects];

[\_colorGreenIndex removeAllObjects];

[\_colorRedIndexBackup removeAllObjects];

[\_colorGreenIndexBackup removeAllObjects];

[\_colorRedIndexCpk removeAllObjects];

[\_colorGreenIndexCpk removeAllObjects];

[\_colorYellowIndexCpk removeAllObjects];

[\_colorRedIndexCpkBackup removeAllObjects];

[\_colorGreenIndexCpkBackup removeAllObjects];

[\_colorYellowIndexCpkBackup removeAllObjects];

[\_reviewerNameIndex removeAllObjects];

//[\_bmcNoIndex removeAllObjects]; not put here, put in set parameters

//[\_bmcYesIndex removeAllObjects];

[\_limitUpdateData removeAllObjects];

[tmpColorArr removeAllObjects];

[\_sortDataBackup removeAllObjects];

[\_dataBackup removeAllObjects];

n\_Start\_Data\_Col = n\_Start\_Data\_Col\_tmp;

n\_Pass\_Fail\_Status = n\_Pass\_Fail\_Status\_tmp;

n\_Product\_Col = n\_Product\_Col\_tmp;

n\_SerialNumber = n\_SerialNumber\_tmp;

n\_SpecialBuildName\_Col = n\_SpecialBuildName\_Col\_tmp;

n\_Special\_Build\_Descrip\_Col = n\_Special\_Build\_Descrip\_Col\_tmp;

n\_StationID\_Col =n\_StationID\_Col\_tmp;

n\_StartTime = n\_StartTime\_tmp;

n\_Version\_Col =n\_Version\_Col\_tmp;

n\_Diags\_Version\_Col = n\_Diags\_Version\_Col\_tmp;

n\_OS\_VERSION\_Col = n\_OS\_VERSION\_Col\_tmp;

NSTimeInterval now = [[NSDate date]timeIntervalSince1970];

NSLog(@"====read csv执行时间: %f",now-starttime);

// ====store data fct

//myRedis->SetString(FCT\_RAW\_DATA,[[NSString stringWithFormat:@"%@",\_rawData] UTF8String]);

[self initRedisAndData];

if (myRedis)

{

myRedis->SetString(FCT\_SCRIPT\_VERSION,[[NSString stringWithFormat:@"%@",\_rawData[0]] UTF8String]);

myRedis->SetString(FCT\_ITEMS\_NAME,[[NSString stringWithFormat:@"%@",\_rawData[1]] UTF8String]);

}

else

{

NSLog(@"---->> redis error");

}

/\*

Site,Product,SerialNumber,Special Build Name,Special Build Description,Unit Number,Station ID,Test Pass/Fail Status,StartTime,EndTime,Version,List of Failing Tests,Head Id,Fixture Id

#define Start\_Data\_Row 7

#define Start\_Data\_Col 11

#define Pass\_Fail\_Status 7

#define Product\_Col 1

#define SerialNumber 2

#define SpecialBuildName\_Col 3

#define Special\_Build\_Descrip\_Col 4

#define StationID\_Col 6

#define Start\_Calc\_Data\_Col 12

#define StartTime 8

#define Version\_Col 10

\*/

//for (int i=0; i<[\_rawData[0] count]; i++) //计算开始

// int n\_index = 30; //匹配前30个数据，节约时间

//if ([\_rawData[0] count] >30)

// {

for (int i=0; i<create\_empty\_line; i++) //参看 数据说明.xlsx，前面从0 到36行，留给ui界面 和UI界面的一些设置，从37行开始，是储存的数据

{

/\*

因为前面几7行，insight 有数据是如下，所以从第7行开始，创建新的，防止把insight data 数据污染

FCT,20200310\_v1\_\_oscar\_

Site,Product,SerialNumb

Display Name ----->,,,,

PDCA Priority ----->,,,

Upper Limit ----->,,,,,

Lower Limit ----->,,,,,

Measurement Unit ----->

\*/

[\_rawData insertObject:@[@""] atIndex:7]; //占位，给UI显示 从第七开始，

}

NSMutableArray \*mutArrayReverse = [NSMutableArray arrayWithArray:[self reverseArray:\_rawData]];

//\_dataReverse = [NSKeyedUnarchiver unarchiveObjectWithData:[NSKeyedArchiver archivedDataWithRootObject:mutArrayReverse]];

n\_passdata = 0;

for (int i=tb\_data\_start; i<[mutArrayReverse[n\_Pass\_Fail\_Status] count]; i++)

{

if ([mutArrayReverse[n\_Pass\_Fail\_Status][i] isEqualToString:@"PASS"])

{

n\_passdata ++;

}

if (n\_passdata>4)

{

break;

}

}

/\*if (n\_passdata<4)

{

[self AlertBox:@"Warning." withInfo:@"PASS data less than 3, it can not calculate cpk value. \r\nPlease check CPK distribution by click “NO” in “Remove Fail”"];

}\*/

\_dataReverse = mutArrayReverse;

NSUInteger indexItem=0;

NSString \*cpkL= [m\_configDictionary valueForKey:cpk\_Lowthl];

NSString \*cpkH = [m\_configDictionary valueForKey:cpk\_Highthl];

NSMutableArray \*insightTestName = [NSMutableArray array];

for (int i=0; i<[mutArrayReverse count]; i++)

{

if ([mutArrayReverse[i] isKindOfClass:[NSArray class]] && [mutArrayReverse[i] count] > 1)

{

if (i>=n\_Start\_Data\_Col) //

{

[\_data addObject:mutArrayReverse[i]];

\_data[indexItem][tb\_index]= [NSNumber numberWithInteger:indexItem+1]; //UI index

\_data[indexItem][tb\_item]= mutArrayReverse[i][1]; //UI item;

[insightTestName addObject:mutArrayReverse[i][1]];

\_data[indexItem][tb\_lower]= mutArrayReverse[i][5]; //UI lower;

\_data[indexItem][tb\_upper]= mutArrayReverse[i][4]; //UI upper;

\_data[indexItem][tb\_measurement\_unit]= mutArrayReverse[i][6]; // number 6 is unit

\_data[indexItem][tb\_lsl]= @""; // UI new LSL

\_data[indexItem][tb\_usl]= @""; // UI new USL

\_data[indexItem][tb\_apply]= [NSNumber numberWithInteger:0]; // UI apply button

\_data[indexItem][tb\_description]= @""; // UI description

\_data[indexItem][tb\_command]= @""; //

\_data[indexItem][tb\_reviewer]= @""; //

\_data[indexItem][tb\_date]= @""; //

\_data[indexItem][tb\_comment]= @""; // UI comment

\_data[indexItem][tb\_3cv]= @""; // UI 3CV

\_data[indexItem][tb\_zoom\_type]= @"limit"; //显示有没有limit zoom in

\_data[indexItem][19]= [NSNumber numberWithInteger:250]; // UI界面设置的bin值

\_data[indexItem][20]= @""; //zmq 传过给python的item 名字

\_data[indexItem][21]= @"";

\_data[indexItem][22]=@"";

\_data[indexItem][tb\_keynote]=[NSNumber numberWithInteger:0]; //设置keynote 勾选按钮

\_data[indexItem][24]=cpkL; //设置low ThLD -->1.5

\_data[indexItem][25]=cpkH; //设置High THLD -->10.0

\_data[indexItem][tb\_correlation\_xy]=@"";

\_data[indexItem][tb\_range\_lsl] = mutArrayReverse[i][5];

\_data[indexItem][tb\_range\_usl] = mutArrayReverse[i][4];

\_data[indexItem][tb\_cpk\_new]=@"";//

\_data[indexItem][tb\_cpk\_log\_path]=[NSString stringWithFormat:@"%@/CPK\_Log",desktopPath];//设置log文件路径 /桌面/CPK\_Log

\_data[indexItem][tb\_color\_by\_left]= [NSNumber numberWithInteger:0]; //设置color By左边那个

\_data[indexItem][tb\_color\_by\_right]= [NSNumber numberWithInteger:0]; //设置color By右边那个

\_data[indexItem][button\_select\_x]= [NSNumber numberWithInteger:0];

\_data[indexItem][button\_select\_y]= [NSNumber numberWithInteger:0];

\_data[indexItem][tb\_script\_flag]= [NSNumber numberWithInteger:0]; //设置是否是script数据，insight 数据标志0

\_data[indexItem][tb\_data]= Start\_Data; //all the test data below

NSString \*itemName = [NSString stringWithFormat:@"%@",mutArrayReverse[i][1]];

[\_indexItemNameDic setValue:itemName forKey:[NSString stringWithFormat:@"%zd",indexItem]];

// myRedis->SetString([combineItem UTF8String],[[NSString stringWithFormat:@"%@",mutArrayReverse[i]] UTF8String]);

[\_ListAllItemNameArr addObject:itemName];

indexItem ++;

}

}

}

[m\_configDictionary setValue:insightTestName forKey:KItemNameInsight];

[\_sortDataBackup removeAllObjects];

[\_sortDataBackup setArray:\_data];

[tmpColorArr removeAllObjects];

[tmpColorArr setArray:\_data];

[self.dataTableView reloadData];

[m\_configDictionary setValue:[NSNumber numberWithBool:YES] forKey:K\_dic\_Load\_Csv\_Finished];

NSMutableString \*itemStr = [NSMutableString string];

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

[itemStr appendString:[NSString stringWithFormat:@"%d,%@\n",i+1,\_ListAllItemNameArr[i]]];

}

[itemStr writeToFile:KdataItemNamePath atomically:YES encoding:NSUTF8StringEncoding error:nil];

//[itemStr writeToFile:KItemNamePathDataTmp atomically:YES encoding:NSUTF8StringEncoding error:nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationReloadSkipSettingData object:nil userInfo:nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationReloadReportTags object:nil userInfo:nil];

if (n\_passdata<4)

{

[self AlertBox:@"Warning." withInfo:@"PASS data less than 3, it can not calculate cpk value. \r\nPlease check CPK distribution by click “NO” in “Remove Fail”"];

}

return YES;

}

-(NSString \*)combineItemName:(NSString \*)name

{

NSString \*str\_name = @"";

str\_name = [NSString stringWithFormat:@"%@##%@&%@",name,[m\_configDictionary valueForKey:kRetestSeg],[m\_configDictionary valueForKey:kRemoveFailSeg]];

return str\_name;

}

-(NSArray \*)reverseArray:(NSArray \*)array

{

NSArray \*tmpArray = array[1];

NSMutableArray \*newArray = [NSMutableArray arrayWithCapacity:tmpArray.count];

for (NSInteger i=0; i<tmpArray.count; i++) {

NSMutableArray \*lineArray = [NSMutableArray arrayWithCapacity:array.count];

for (NSInteger j=0; j<array.count; j++) {

[lineArray addObject:@""];

}

[newArray addObject:lineArray];

}

for (NSInteger i=0; i<array.count; i++) {

for (NSInteger j=0; j<tmpArray.count; j++) {

if ([array[i] count]<=j)

{

newArray[j][i] = @"";

}

else

{

newArray[j][i] = array[i][j];

}

}

}

return newArray;

}

-(NSArray \*)reverseArray\_ext:(NSArray \*)array

{

NSArray \*tmpArray = array[1];

NSMutableArray \*newArray = [NSMutableArray arrayWithCapacity:tmpArray.count];

for (NSInteger i=0; i<tmpArray.count; i++) {

NSMutableArray \*lineArray = [NSMutableArray arrayWithCapacity:array.count];

for (NSInteger j=0; j<array.count; j++) {

[lineArray addObject:@""];

}

[newArray addObject:lineArray];

}

for (NSInteger i=0; i<array.count; i++) {

for (NSInteger j=0; j<tmpArray.count; j++) {

if ([array[i] count]<=j)

{

if (i==0 && j==0)

{

newArray[j][i] = @"Parametric";

}

else

{

newArray[j][i] = @"";

}

}

else

{

if (i==0 && j==0)

{

newArray[j][i] = @"Parametric";

}

else

{

newArray[j][i] = array[i][j];

}

}

}

}

return newArray;

}

-(void)sendDataToRedis:(NSString \*)name withData:(NSMutableArray \*)arrData

{

if (myRedis)

{

myRedis->SetString([name UTF8String],[[NSString stringWithFormat:@"%@",arrData] UTF8String]);

}

else

{

[self AlertBox:@"Error:027" withInfo:@"Redis server is shut down.!!!"];

}

NSLog(@"--->>set name to redis:%@ %zd",name,[arrData count]);

// NSArray \*nameArr = [name componentsSeparatedByString:@"###"];

// if ([nameArr count]>1)

// {

// NSArray \*nameArrOp = [nameArr[1] componentsSeparatedByString:@"&"];

// if ([nameArrOp count]>1)

// {

// NSLog(@"==retest: %@ remove: %@",nameArrOp[0],nameArrOp[1]);

// }

// }

}

-(void)sendStringToRedis:(NSString \*)name withData:(NSString \*)strData

{

if (myRedis)

{

myRedis->SetString([name UTF8String],[strData UTF8String]);

}

else

{

[self AlertBox:@"Error:027" withInfo:@"Redis server is shut down!"];

}

}

//-(void)setCpkImage:(NSString \*)path

//{

// NSImage \*imageCPK = [[NSImage alloc]initWithContentsOfFile:path];

// dispatch\_async(dispatch\_get\_main\_queue(), ^{

// [self.cpkImageMap setImage:imageCPK];

// });

//}

//-(void)setCorrelationImage:(NSString \*)path

//{

// NSImage \*imageCorrelation = [[NSImage alloc]initWithContentsOfFile:path];

// dispatch\_async(dispatch\_get\_main\_queue(), ^{

// [self.correlationImageMap setImage:imageCorrelation];

// });

//}

-(NSString \*)sendCpkZmqMsg:(NSString \*)name

{

NSString \*file1 = @"/tmp/CPK\_Log/temp/cpk.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/cpk.png",desktopPath];

NSFileManager \*manager = [NSFileManager defaultManager];

[manager removeItemAtPath:file1 error:nil];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"cpk.png" ofType:nil];

[manager copyItemAtPath:picPath toPath:file1 error:nil];

int ret = [cpkClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [cpkClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python error");

}

NSLog(@"app->get response from python: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendCalculateZmqMsg:(NSString \*)name

{

NSString \*path1 = [m\_configDictionary valueForKey:Load\_Csv\_Path];

NSString \*path2 = @"/tmp/CPK\_Log/temp/calculate\_param.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/calculate\_param.csv",desktopPath];

NSString \*path3 = @"/tmp/CPK\_Log/temp/.logcalc.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.logcalc.txt",desktopPath];

NSString \*cpkL= [m\_configDictionary valueForKey:cpk\_Lowthl];

NSString \*cpkH = [m\_configDictionary valueForKey:cpk\_Highthl];

NSString \*msg = [NSString stringWithFormat:@"%@$$%@$$%@$$%@$$%@$$%@",name,path1,path2,path3,cpkL,cpkH]; //calculate-param

int ret = [calculateClient SendCmd:msg];

if (ret > 0)

{

NSString \* response = [calculateClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python calculate error");

}

NSLog(@"app->get response from python calculate: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendCalculateZmqMsgLocal:(NSString \*)name

{

NSString \*path1 = [m\_configDictionary valueForKey:Load\_Local\_Csv\_Path];

NSString \*path2 = @"/tmp/CPK\_Log/temp/calculate\_param.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/calculate\_param.csv",desktopPath];

NSString \*path3 = @"/tmp/CPK\_Log/temp/.logcalc.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.logcalc.txt",desktopPath];

NSString \*cpkL= [m\_configDictionary valueForKey:cpk\_Lowthl];

NSString \*cpkH = [m\_configDictionary valueForKey:cpk\_Highthl];

NSString \*msg = [NSString stringWithFormat:@"%@$$%@$$%@$$%@$$%@$$%@",name,path1,path2,path3,cpkL,cpkH]; //calculate-param\_local

NSLog(@"--calculte local name: %@",msg);

int ret = [calculateClient SendCmd:msg];

if (ret > 0)

{

NSString \* response = [calculateClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python calculate local error");

}

NSLog(@"app->get response from python calculate local: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendYieldRateZmqMsg:(NSString \*)name

{

NSString \*path1 = @"/tmp/CPK\_Log/temp/yield\_rate\_param.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/yield\_rate\_param.csv",desktopPath];

NSString \*path2 = @"/tmp/CPK\_Log/temp/.logretest.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.logretest.txt",desktopPath];

NSString \*csv\_Data = [m\_configDictionary valueForKey:Load\_Csv\_Path];

NSString \*msg = [NSString stringWithFormat:@"%@$$%@$$%@$$%@",name,csv\_Data,path1,path2]; //retest rate-param

int ret = [retestRateClient SendCmd:msg];

if (ret > 0)

{

NSString \* response = [retestRateClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python retest rate error");

}

NSLog(@"app->get response from python retest rate: %@",response);

return response;

}

return nil;

}

-(void)initRetestPlotAndCsv

{

NSArray \*remove\_path = @[pie\_retest\_csv,retest\_csv\_csv,retest\_vs\_station\_id\_csv,retest\_vs\_version\_csv,summary\_retest\_csv,cpk\_min\_max\_csv,daily\_retest\_summary\_png,fail\_item\_overall\_csv,fail\_pareto\_png,retest\_breakdown\_fixture\_csv,retest\_item\_overall\_csv,retest\_pareto\_png,retest\_pie\_png,retest\_vs\_station\_id\_png,retest\_vs\_version\_png,header\_info\_csv\_csv,fail\_csv\_csv,total\_count\_by\_version\_csv,total\_count\_by\_station\_slot\_id\_csv,total\_count\_by\_date\_product\_csv,yield\_rate\_param\_tmp\_csv,cpk\_range\_csv,daily\_all\_retest\_summary\_png];

NSFileManager \*fileManager = [NSFileManager defaultManager];

for (NSString \*path\_object in remove\_path)

{

[fileManager removeItemAtPath:path\_object error:NULL];

}

for (int i=0; i<100; i++)

{

NSString \*filePath1 = [NSString stringWithFormat:@"%@%d.png",daily\_retest\_summary\_x,i+1];

BOOL isExist1 = [fileManager fileExistsAtPath:filePath1];

if (isExist1)

{

[fileManager removeItemAtPath:filePath1 error:NULL];

}

NSString \*filePath2 = [NSString stringWithFormat:@"%@%d.png",retest\_vs\_station\_id\_x,i+1];

BOOL isExist2 = [fileManager fileExistsAtPath:filePath2];

if (isExist2)

{

[fileManager removeItemAtPath:filePath2 error:NULL];

}

NSString \*filePath3 = [NSString stringWithFormat:@"%@%d.png",retest\_vs\_version\_x,i+1];

BOOL isExist3 = [fileManager fileExistsAtPath:filePath3];

if (isExist3)

{

[fileManager removeItemAtPath:filePath3 error:NULL];

}

NSString \*filePath4 = [NSString stringWithFormat:@"%@%d.png",daily\_all\_retest\_summary\_x,i+1];

BOOL isExist4 = [fileManager fileExistsAtPath:filePath4];

if (isExist4)

{

[fileManager removeItemAtPath:filePath4 error:NULL];

}

if (!isExist1 && !isExist2 && !isExist3 && !isExist4)

{

break;

}

}

NSString \*pathretest =@"/tmp/CPK\_Log/retest/.retest\_plot.txt";

[@"Finished,init retest folder empty!" writeToFile:pathretest atomically:YES encoding:NSUTF8StringEncoding error:nil];

}

-(NSString \*)sendRetestPlotZmqMsg:(NSString \*)name

{

NSString \*path1 = @"/tmp/CPK\_Log/retest/.retest\_csv.csv";

NSString \*path2 = @"/tmp/CPK\_Log/retest/.pie\_retest.csv";

NSString \*csv\_Data = [m\_configDictionary valueForKey:Load\_Csv\_Path];

NSString \*msg = [NSString stringWithFormat:@"%@$$%@$$%@$$%@",name,csv\_Data,path1,path2]; //retest rate-param

int ret = [retestPlotClient SendCmd:msg];

if (ret > 0)

{

NSString \* response = [retestPlotClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python retest plot error");

}

NSLog(@"app->get response from python retest plot: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendYieldRateZmqMsgLocal:(NSString \*)name

{

NSString \*path1 = @"/tmp/CPK\_Log/temp/yield\_rate\_param.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/yield\_rate\_param.csv",desktopPath];

NSString \*path2 = @"/tmp/CPK\_Log/temp/.logretest.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.logretest.txt",desktopPath];

NSString \*csv\_Data = [m\_configDictionary valueForKey:Load\_Local\_Csv\_Path];

NSString \*msg = [NSString stringWithFormat:@"%@$$%@$$%@$$%@",name,csv\_Data,path1,path2]; //retest rate-param

int ret = [retestRateClient SendCmd:msg];

if (ret > 0)

{

NSString \* response = [retestRateClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python retest rate error");

}

NSLog(@"app->get response from python retest rate: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendCorrelationZmqMsg:(NSString \*)name

{

NSString \*file1 = @"/tmp/CPK\_Log/temp/correlation.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/correlation.png",desktopPath];

NSFileManager \*manager = [NSFileManager defaultManager];

[manager removeItemAtPath:file1 error:nil];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"correlation.png" ofType:nil];

[manager copyItemAtPath:picPath toPath:file1 error:nil];

NSLog(@"---set sendCorrelationZmqMsg:%@",name);

int ret = [correlationClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [correlationClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python error");

}

NSLog(@"app->correlation get response from python: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendScatterZmqMsg:(NSString \*)name

{

NSString \*file1 = @"/tmp/CPK\_Log/temp/scatter.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/scatter.png",desktopPath];

NSFileManager \*manager = [NSFileManager defaultManager];

[manager removeItemAtPath:file1 error:nil];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"scatter.png" ofType:nil];

[manager copyItemAtPath:picPath toPath:file1 error:nil];

NSLog(@">>set send Scatter Zmq Msg:%@",name);

int ret = [scatterClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [scatterClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python error");

}

NSLog(@"app->scatter get response from python: %@",response);

return response;

}

return nil;

}

-(NSMutableArray \*)removeFailData:(NSInteger)seletRow

{

NSMutableArray \*tempArray = [NSMutableArray array];

NSMutableArray \*itemArray = \_dataReverse[seletRow+n\_Start\_Data\_Col];

NSArray \*arrayCol = \_dataReverse[n\_Pass\_Fail\_Status];

for (NSInteger i=0; i<[arrayCol count]; i++)

{

if (![arrayCol[i] isEqualToString:@"FAIL"]) {

[tempArray addObject:itemArray[i]];

}

}

return tempArray;

}

-(NSMutableArray \*)removeFailDataIndex:(int)removeF

{

NSMutableArray \*tempArray = [NSMutableArray array];

if (removeF==0) // remove fail = yes

{

NSArray \*arrayCol = \_dataReverse[n\_Pass\_Fail\_Status];

for (int i=0; i<[arrayCol count]; i++)

{

if ([arrayCol[i] isEqualToString:@"FAIL"])

{

[tempArray addObject:[NSNumber numberWithInt:i]];

}

}

}

else if (removeF==1) //remove fail = no

{

// nothing need to do

}

return tempArray;

}

-(NSMutableArray \*)addPassDataIndex

{

NSMutableArray \*tempArrayIndex = [NSMutableArray array];

NSArray \*arrayCol = \_dataReverse[n\_Pass\_Fail\_Status];

for (NSInteger i=0; i<[arrayCol count]; i++)

{

if (![arrayCol[i] isEqualToString:@"FAIL"]) {

[tempArrayIndex addObject:[NSNumber numberWithInteger:i]];

}

}

return tempArrayIndex;

}

-(int)compareTime:(NSString\*)date01 withDate:(NSString\*)date02

{

int ci;

NSDateFormatter \*df = [[NSDateFormatter alloc] init];

[df setDateFormat:@"yyyy-MM-dd HH:mm"];

NSDate \*dt1 = [[NSDate alloc] init];

NSDate \*dt2 = [[NSDate alloc] init];

dt1 = [df dateFromString:date01];

dt2 = [df dateFromString:date02];

NSComparisonResult result = [dt1 compare:dt2];

switch (result)

{

case NSOrderedAscending: ci=1; break; //date02比date01大

case NSOrderedDescending: ci=-1; break; //date02比date01小

case NSOrderedSame: ci=0; break; //date02=date01

default: NSLog(@"erorr dates %@, %@", dt2, dt1); break;

}

return ci;

}

-(int)compareStartTime:(NSString\*)date01 withDate:(NSString\*)date02

{

if ([date01 isEqualToString:@""])

{

return 1;

}

long time01 = [self getTimeNumberWithString:date01];

long time02 = [self getTimeNumberWithString:date02];

if (time01>time02)

{

return -1;

}

else if (time01<time02)

{

return 1;

}

else

{

return 0;

}

}

-(int)compareStartTime2:(NSString\*)date01 withDate:(NSString\*)date02

{

if ([date01 isEqualToString:@""])

{

return -1;

}

long time01 = [self getTimeNumberWithString:date01];

long time02 = [self getTimeNumberWithString:date02];

if (time01>time02)

{

return -1;

}

else if (time01<time02)

{

return 1;

}

else

{

return 0;

}

}

//字符串转时间戳 如：2017-4-10 17:15:10

- (NSString\*)getTimeStrWithString:(NSString \*)str

{

NSDateFormatter \*dateFormatter = [[NSDateFormatter alloc] init];// 创建一个时间格式化对象

if ([str length]>17)

{

[dateFormatter setDateFormat:@"YYYY-MM-dd HH:mm:ss"]; //设定时间的格式

}

else

{

[dateFormatter setDateFormat:@"YYYY-MM-dd HH:mm:ss"]; //设定时间的格式

}

NSDate \*tempDate = [dateFormatter dateFromString:str];//将字符串转换为时间对象

NSString \*timeStr = [NSString stringWithFormat:@"%ld", (long)[tempDate timeIntervalSince1970]];//字符串转成时间戳,精确到毫秒\*1000

return timeStr;

}

- (long)getTimeNumberWithString:(NSString \*)str{

NSDateFormatter \*dateFormatter = [[NSDateFormatter alloc] init];// 创建一个时间格式化对象

if ([str length]>17)

{

[dateFormatter setDateFormat:@"YYYY-MM-dd HH:mm:ss"]; //设定时间的格式

}

else

{

[dateFormatter setDateFormat:@"YYYY-MM-dd HH:mm"]; //设定时间的格式

}

NSDate \*tempDate = [dateFormatter dateFromString:str];//将字符串转换为时间对象

return (long)[tempDate timeIntervalSince1970];

}

-(NSMutableArray \*)failPassItemDataIndex:(NSInteger)seletRow withRemoveOption:(int)removeF

{

NSMutableArray \*dataTemp = nil;

if (removeF==0) // remove fail = yes

{

dataTemp = [self removeFailData:seletRow];

}

else if (removeF==1) //remove fail = no

{

dataTemp = \_dataReverse[seletRow+n\_Start\_Data\_Col];

}

return dataTemp;

}

-(NSArray \*)getItemDataIndexWithRetestOption:(int)retestSeg withRemoveOption:(int)removeFSeg

{

if (retestSeg == 1) // retest = all

{

return nil;

}

NSMutableArray \*snArray = \_dataReverse[n\_SerialNumber];

NSMutableArray \*startTimeArray = \_dataReverse[n\_StartTime];

NSArray \*arrayFailPass = \_dataReverse[n\_Pass\_Fail\_Status];

NSMutableArray \*arrayUnique = [NSMutableArray array];

NSMutableArray \*arraySame = [NSMutableArray array];

for (unsigned i = 0; i<[snArray count]; i++)

{

if ([arrayUnique containsObject:[snArray objectAtIndex:i]] == NO)

{

[arrayUnique addObject:[snArray objectAtIndex:i]];

}

else

{

[arraySame addObject:[snArray objectAtIndex:i]];

}

}

NSSet \*setX = [NSSet setWithArray:arraySame];

NSArray \* arrayD = [setX allObjects];

NSMutableArray \*timeArrIndex = [NSMutableArray array]; //retest 选项所有相同的元素 索引

NSMutableArray \*timeArrMaxIndex = [NSMutableArray array]; //retest last 即时间最大元素

if ([arrayD count] >0)

{

for (NSString \*snDuplicate in arrayD)

{

if (snDuplicate && [snDuplicate isNotEqualTo:@""])

{

NSString \* maxStartTime=@"";

int maxTimeIndex = 0;

int ii=0;

for (NSString \*object in snArray)

{

if ([snDuplicate isEqualToString:object])

{

[timeArrIndex addObject:[NSNumber numberWithInt:ii]];

if (retestSeg == 2) // retest last

{

int result = [self compareStartTime:maxStartTime withDate:startTimeArray[ii]];

//NSLog(@"====retult: %d",result);

if(result==1)

{

if (removeFSeg == 0)

{

if (![arrayFailPass[ii] isEqualToString:@"FAIL"])

{

maxStartTime = startTimeArray[ii];

maxTimeIndex = ii;

}

}

else

{

maxStartTime = startTimeArray[ii];

maxTimeIndex = ii;

}

}

}

else if (retestSeg == 0) // retest first

{

int result = [self compareStartTime2:maxStartTime withDate:startTimeArray[ii]];

//NSLog(@"====1 retult: %d",result);

if(result==-1)

{

if (removeFSeg == 0) // remove fail=yes

{

if (![arrayFailPass[ii] isEqualToString:@"FAIL"])

{

maxStartTime = startTimeArray[ii];

maxTimeIndex = ii;

}

}

else // remove fail=no

{

maxStartTime = startTimeArray[ii];

maxTimeIndex = ii;

}

}

}

}

ii++;

}

[timeArrMaxIndex addObject:[NSNumber numberWithInt:maxTimeIndex]];

//NSLog(@"\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

}

}

}

NSPredicate \* filterPredicate = [NSPredicate predicateWithFormat:@"NOT (SELF IN %@)",timeArrMaxIndex];

NSArray \* filterLast = [timeArrIndex filteredArrayUsingPredicate:filterPredicate]; //==剔除Last 之前数据

//NSLog(@"===剔除Last 之前数据 : %@ %@ %@",timeArrIndex,timeArrMaxIndex,filterLast);

return filterLast;

}

-(NSMutableArray \*)getItemDataWithRetestIndex:(NSArray \*)filterData withRemoveFailIndex:(NSArray \*)filterData2 bySelectRow:(NSInteger )seletRow //根据index 删除数据

{

NSMutableArray \*itemArray = \_dataReverse[seletRow+n\_Start\_Data\_Col];

NSMutableArray \*tempArray = [NSMutableArray array];

for (int i=0; i<[itemArray count]; i++)

{

if (![filterData containsObject:[NSNumber numberWithInt:i]] && ![filterData2 containsObject:[NSNumber numberWithInt:i]])

{

[tempArray addObject:itemArray[i]];

}

}

//NSLog(@"====tempArray==>> %zd %@",[tempArray count],tempArray);

NSLog(@"====tempArray==>> %zd",[tempArray count]);

return tempArray;

}

-(NSMutableArray \*)getItemDataWithRetestIndex:(NSArray \*)filterData bySelectRow:(NSInteger )seletRow //根据index 删除数据

{

NSMutableArray \*itemArray = \_dataReverse[seletRow+n\_Start\_Data\_Col];

NSMutableArray \*snArray = \_dataReverse[n\_SerialNumber];

NSMutableArray \*tempArray = [NSMutableArray array];

NSMutableArray \*snTmpArray = [NSMutableArray array];

for (int i=0; i<[itemArray count]; i++)

{

if (![filterData containsObject:[NSNumber numberWithInt:i]])

{

[tempArray addObject:itemArray[i]];

[snTmpArray addObject:snArray[i]];

// if ([\_dataReverse[n\_StartTime][i] isNotEqualTo:@"StartTime"] &&[\_dataReverse[n\_StartTime][i] isNotEqualTo:@""]) //这里如果获取时间大小，浪费时间

// {

// if ([self compareStartTime:max\_time withDate:\_dataReverse[n\_StartTime][i]]==1)

// {

// max\_time = \_dataReverse[n\_StartTime][i];

// }

//

// if ([self compareStartTime2:min\_time withDate:\_dataReverse[n\_StartTime][i]]==-1)

// {

// min\_time = \_dataReverse[n\_StartTime][i];

// }

// }

}

}

[tempArray addObject:End\_Data];

[snTmpArray addObject:End\_Data];

NSDictionary \*dic = [NSDictionary dictionaryWithObjectsAndKeys:snTmpArray,kSerial\_number,tempArray,kData\_Value, nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationShowData object:nil userInfo:dic];

if (![[m\_configDictionary valueForKey:kInputRangeFlag] boolValue])

{

NSDictionary \*dic2 = [NSDictionary dictionaryWithObjectsAndKeys:tempArray[tb\_lower],krangelsl,tempArray[tb\_upper],krangeusl, nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetRangeLslUsl object:nil userInfo:dic2];

}

return tempArray;

}

-(NSMutableArray \*)calculateData:(NSInteger )seletRow withRetest:(NSString \*)opt1 withRemove:(NSString \*)opt2

{

// retest: first=0 all=1,last=2

//remove fail: yes=0, no=1

int retest = 0;

int removeF = 0;

if ([opt1 isEqualToString:vRetestAll] && [opt2 isEqualToString:vRemoveFailYes])

{

retest = 1;

removeF = 0;

}

else if ([opt1 isEqualToString:vRetestAll] && [opt2 isEqualToString:vRemoveFailNo])

{

// for save time no need do anything

retest = 1;

removeF = 1;

return \_dataReverse[seletRow+n\_Start\_Data\_Col];

}

else if ([opt1 isEqualToString:vRetestFirst] && [opt2 isEqualToString:vRemoveFailYes])

{

retest = 0;

removeF = 0;

}

else if ([opt1 isEqualToString:vRetestFirst] && [opt2 isEqualToString:vRemoveFailNo])

{

retest = 0;

removeF = 1;

}

else if ([opt1 isEqualToString:vRetestLast] && [opt2 isEqualToString:vRemoveFailYes])

{

retest = 2;

removeF = 0;

}

else if ([opt1 isEqualToString:vRetestLast] && [opt2 isEqualToString:vRemoveFailNo])

{

retest = 2;

removeF = 1;

}

else{

return [NSMutableArray arrayWithObject:@[@"0"]];

}

NSMutableArray \* removeArrIndex = [self removeFailDataIndex:removeF];

NSArray \*arrIndex = [self getItemDataIndexWithRetestOption:retest withRemoveOption:removeF];

return [self getItemDataWithRetestIndex:removeArrIndex withRemoveFailIndex:arrIndex bySelectRow:seletRow];

}

-(NSMutableArray \*)calculateData:(NSInteger )seletRow

{

NSString \*opt1 = [m\_configDictionary valueForKey:kRetestSeg];

NSString \*opt2 = [m\_configDictionary valueForKey:kRemoveFailSeg];

NSString \*dic\_key = [NSString stringWithFormat:@"%@&%@",opt1,opt2];

NSMutableArray \*indexArr = [m\_configDictionary valueForKey:dic\_key];

return [self getItemDataWithRetestIndex:indexArr bySelectRow:seletRow];

}

-(void)initRetestAndRemoveFailSeg

{

int removeF = 0; //RemoveFail=Yes

int retest = 1; //Retest=All

NSMutableArray \* removeArrIndex0 = [self removeFailDataIndex:removeF];

NSArray \*retestArrIndex0 = [self getItemDataIndexWithRetestOption:retest withRemoveOption:removeF];

for (int i=0; i<[retestArrIndex0 count]; i++)

{

[removeArrIndex0 addObject:retestArrIndex0[i]];

}

[m\_configDictionary setObject:removeArrIndex0 forKey:k\_dic\_RetestAll\_RemoveFailYes];

removeF = 1; // RemoveFail=No

retest = 1; //Retest=All

NSMutableArray \* removeArrIndex1 = [self removeFailDataIndex:removeF];

NSArray \*retestArrIndex1 = [self getItemDataIndexWithRetestOption:retest withRemoveOption:removeF];

for (int i=0; i<[retestArrIndex1 count]; i++)

{

[removeArrIndex1 addObject:retestArrIndex1[i]];

}

[m\_configDictionary setObject:removeArrIndex1 forKey:k\_dic\_RetestAll\_RemoveFailNo];

removeF = 0; // RemoveFail=Yes

retest = 0; //Retest=First

NSMutableArray \* removeArrIndex2 = [self removeFailDataIndex:removeF];

NSArray \*retestArrIndex2 = [self getItemDataIndexWithRetestOption:retest withRemoveOption:removeF];

for (int i=0; i<[retestArrIndex2 count]; i++)

{

[removeArrIndex2 addObject:retestArrIndex2[i]];

}

[m\_configDictionary setObject:removeArrIndex2 forKey:k\_dic\_RetestFirst\_RemoveFailYes];

removeF = 1; // RemoveFail=No

retest = 0; //Retest=First

NSMutableArray \* removeArrIndex3 = [self removeFailDataIndex:removeF];

NSArray \*retestArrIndex3 = [self getItemDataIndexWithRetestOption:retest withRemoveOption:removeF];

for (int i=0; i<[retestArrIndex3 count]; i++)

{

[removeArrIndex3 addObject:retestArrIndex3[i]];

}

[m\_configDictionary setObject:removeArrIndex3 forKey:k\_dic\_RetestFirst\_RemoveFailNo];

retest = 2; //Retest=Last

removeF = 0; //RemoveFail=Yes

NSMutableArray \* removeArrIndex4 = [self removeFailDataIndex:removeF];

NSArray \*retestArrIndex4 = [self getItemDataIndexWithRetestOption:retest withRemoveOption:removeF];

for (int i=0; i<[retestArrIndex4 count]; i++)

{

[removeArrIndex4 addObject:retestArrIndex4[i]];

}

[m\_configDictionary setObject:removeArrIndex4 forKey:k\_dic\_RetestLast\_RemoveFailYes];

retest = 2; //Retest=Last

removeF = 1; //vRemoveFail=No

NSMutableArray \* removeArrIndex5 = [self removeFailDataIndex:removeF];

NSArray \*retestArrIndex5 = [self getItemDataIndexWithRetestOption:retest withRemoveOption:removeF];

for (int i=0; i<[retestArrIndex5 count]; i++)

{

[removeArrIndex5 addObject:retestArrIndex5[i]];

}

[m\_configDictionary setObject:removeArrIndex5 forKey:k\_dic\_RetestLast\_RemoveFailNo];

}

-(void)initColorByTableView

{

//version

if (n\_Version\_Col>=0)

{

NSArray \*arrayVer = \_dataReverse[n\_Version\_Col];

NSSet \*set = [NSSet setWithArray:arrayVer];

NSArray \*tempVer = [set allObjects];

NSMutableArray \*vers = [NSMutableArray array];

for (int i=0; i<[tempVer count]; i++)

{

if ([tempVer[i] isNotEqualTo:@""] && [[tempVer[i] uppercaseString] isNotEqualTo:@"VERSION"])

{

[vers addObject:tempVer[i]];

}

}

[m\_configDictionary setObject:vers forKey:k\_dic\_Version];

}

else

{

[m\_configDictionary setObject:@"" forKey:k\_dic\_Version];

}

// station id

if (n\_StationID\_Col>=0)

{

NSArray \*arrayStations = \_dataReverse[n\_StationID\_Col];

NSSet \*setStation = [NSSet setWithArray:arrayStations];

NSArray \*tempId = [setStation allObjects];

NSMutableArray \*IDs = [NSMutableArray array];

for (int i=0; i<[tempId count]; i++)

{

if ([tempId[i] isNotEqualTo:@""] && [[tempId[i] uppercaseString] isNotEqualTo:@"STATION ID"] && [[tempId[i] uppercaseString] isNotEqualTo:@"SITE\_ID"] && [[tempId[i] uppercaseString] isNotEqualTo:@"STATION\_ID"])

{

[IDs addObject:tempId[i]];

}

}

[m\_configDictionary setObject:IDs forKey:k\_dic\_Station\_ID];

}

else

{

[m\_configDictionary setObject:@"" forKey:k\_dic\_Station\_ID];

}

//Special Build Name

if (n\_SpecialBuildName\_Col>=0)

{

NSArray \*arrayBuildN = \_dataReverse[n\_SpecialBuildName\_Col];

NSSet \*setBuild = [NSSet setWithArray:arrayBuildN];

NSArray \*tempBuildN = [setBuild allObjects];

NSMutableArray \*BuildNs = [NSMutableArray array];

for (int i=0; i<[tempBuildN count]; i++)

{

if ([tempBuildN[i] isNotEqualTo:@""] && [[tempBuildN[i] uppercaseString] isNotEqualTo:@"SPECIAL BUILD NAME"])

{

[BuildNs addObject:tempBuildN[i]];

}

}

[m\_configDictionary setObject:BuildNs forKey:k\_dic\_Special\_Build\_Name];

}

else

{

[m\_configDictionary setObject:@"" forKey:k\_dic\_Special\_Build\_Name];

}

//Special Build Description

if (n\_Special\_Build\_Descrip\_Col>=0)

{

NSArray \*arrayBuildD = \_dataReverse[n\_Special\_Build\_Descrip\_Col];

NSSet \*setBuildD = [NSSet setWithArray:arrayBuildD];

NSArray \*tempBuildD = [setBuildD allObjects];

NSMutableArray \*BuildDe = [NSMutableArray array];

for (int i=0; i<[tempBuildD count]; i++)

{

if ([tempBuildD[i] isNotEqualTo:@""] && [[tempBuildD[i] uppercaseString] isNotEqualTo:@"SPECIAL BUILD DESCRIPTION"])

{

[BuildDe addObject:tempBuildD[i]];

}

}

[m\_configDictionary setObject:BuildDe forKey:k\_dic\_Special\_Build\_Desc];

}

else

{

[m\_configDictionary setObject:@"" forKey:k\_dic\_Special\_Build\_Desc];

}

//Product

if (n\_Product\_Col>=0)

{

NSArray \*arrayProduct = \_dataReverse[n\_Product\_Col];

NSMutableArray \*Produc = [NSMutableArray array];

for (int i=tb\_data\_start; i<[arrayProduct count]; i++)

{

if ([arrayProduct[i] isNotEqualTo:@""] && [[arrayProduct[i] uppercaseString] isNotEqualTo:@"PRODUCT"])

{

[Produc addObject:arrayProduct[i]];

}

}

NSSet \*setProduct = [NSSet setWithArray:Produc];

NSArray \*tempProduct = [setProduct allObjects];

[m\_configDictionary setObject:tempProduct forKey:k\_dic\_Product];

}

else

{

[m\_configDictionary setObject:@"" forKey:k\_dic\_Product];

}

//channel id

int index\_channelId = -1;

NSString \*keyWord = @"FIXTURE CHANNEL ID";

NSString \*keyWord2 = @"FIXTURE INITILIZATION SLOT\_ID";//@"Fixture Initilization SLOT\_ID";

NSString \*keyWord3 = @"FIXTURE RESET CALC FIXTURE\_CHANNEL";//@"Fixture Reset CALC fixture\_channel";

NSString \*keyWord4 = @"HEAD ID";//@"Head Id";

NSString \*keyWord5 = @"FIXTURE\_CHANNEL CHANNEL CHANNEL\_ID";

NSString \*keyWord6 = @"FIXTURE CHANNEL CHANNEL\_ID";

NSString \*keyWord7 = @"CHANNEL ID";

NSString \*keyWord8 = @"CHANNEL\_ID";

NSString \*keyWord9 = @"SLOT ID";

NSString \*keyWord10 = @"SLOT\_ID";

NSString \*keyWord11 = @"FIXTURE\_SETUP CHANNEL CHANNEL\_ID";

/\*for (int i=0; i<[\_rawData[1] count]; i++)

{

if ([[\_rawData[1][i] uppercaseString] isEqualToString:keyWord] || [[\_rawData[1][i] uppercaseString] isEqualToString:keyWord2]||[[\_rawData[1][i] uppercaseString] isEqualToString:keyWord3]||[[\_rawData[1][i] uppercaseString] isEqualToString:keyWord4])

{

index\_channelId = i;

break;

}

}\*/

for (int i=0; i<[\_ListAllItemNameArr count]; i++)

{

if ([[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord] || [[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord2]||[[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord3]||[[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord4] || [[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord5] || [[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord6] || [[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord7] || [[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord8] || [[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord9] || [[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord10] || [[\_ListAllItemNameArr[i] uppercaseString] isEqualToString:keyWord11])

{

index\_channelId = i;

break;

}

}

if (index\_channelId == -1)

{

[m\_configDictionary setObject:[NSNumber numberWithInt:index\_channelId] forKey:k\_dic\_Channel\_ID\_Index];

[m\_configDictionary setObject:@[@"NULL"] forKey:k\_dic\_Channel\_ID];

}

else

{

index\_channelId = index\_channelId+ n\_Start\_Data\_Col;

[m\_configDictionary setObject:[NSNumber numberWithInt:index\_channelId] forKey:k\_dic\_Channel\_ID\_Index];

if (index\_channelId>=n\_Start\_Data\_Col)

{

NSArray \*arrayChannel = \_dataReverse[index\_channelId];

NSMutableArray \*channels = [NSMutableArray array];

for (int i=tb\_data\_start; i<[arrayChannel count]; i++)

{

// if ([arrayChannel[i] isNotEqualTo:@""])

// {

// [channels addObject:arrayChannel[i]];

// }

if ([arrayChannel[i] isEqualToString:@""])

{

//[channels addObject:@""];

}

else

{

[channels addObject:arrayChannel[i]];

//[channels addObject:[NSString stringWithFormat:@"slot:%@",arrayChannel[i]]];

}

}

NSSet \*setChannel = [NSSet setWithArray:channels];

NSArray \*channelIDs = [setChannel allObjects];

[m\_configDictionary setObject:channelIDs forKey:k\_dic\_Channel\_ID];

}

else

{

[self AlertBox:@"Warning!" withInfo:@"The data has no Channel ID!!!"];

[m\_configDictionary setObject:@[@"NULL"] forKey:k\_dic\_Channel\_ID];

}

}

//diags version

if (n\_Diags\_Version\_Col>0)

{

NSArray \*arrayVer = \_dataReverse[n\_Diags\_Version\_Col];

NSSet \*set = [NSSet setWithArray:arrayVer];

NSArray \*tempVer = [set allObjects];

NSMutableArray \*vers = [NSMutableArray array];

for (int i=0; i<[tempVer count]; i++)

{

if ([tempVer[i] isNotEqualTo:@""] && [[tempVer[i] uppercaseString] isNotEqualTo:@"DIAGS\_VERSION"])

{

[vers addObject:tempVer[i]];

}

}

[m\_configDictionary setObject:vers forKey:k\_dic\_Diags\_Version];

}

else

{

[m\_configDictionary setObject:@"" forKey:k\_dic\_Diags\_Version];

}

//OS\_VERSION

if (n\_OS\_VERSION\_Col>0)

{

NSArray \*arrayVer = \_dataReverse[n\_OS\_VERSION\_Col];

NSSet \*set = [NSSet setWithArray:arrayVer];

NSArray \*tempVer = [set allObjects];

NSMutableArray \*vers = [NSMutableArray array];

for (int i=0; i<[tempVer count]; i++)

{

if ([tempVer[i] isNotEqualTo:@""] && [[tempVer[i] uppercaseString] isNotEqualTo:@"OS\_VERSION"])

{

[vers addObject:tempVer[i]];

}

}

[m\_configDictionary setObject:vers forKey:k\_dic\_OS\_Version];

}

else

{

[m\_configDictionary setObject:@"" forKey:k\_dic\_OS\_Version];

}

// station id & channel id

/\*if (index\_channelId>0)

{

NSMutableArray \*staionChannel = [NSMutableArray array];

NSArray \*arrayChannel = \_dataReverse[index\_channelId];

for (int i=27; i<[arrayStations count]; i++) //从第7行开始

{

[staionChannel addObject:[NSString stringWithFormat:@"%@ & %@",arrayStations[i],arrayChannel[i]]];

}

NSSet \*setStationChannel = [NSSet setWithArray:staionChannel];

NSArray \*stationChannelID = [setStationChannel allObjects];

[m\_configDictionary setObject:stationChannelID forKey:k\_dic\_Station\_Channel\_ID];

}

else

{

[m\_configDictionary setObject:@[@"NULL"] forKey:k\_dic\_Station\_Channel\_ID];

}

\*/

}

#pragma mark **TableView Datasource & delegate**

-(NSInteger)numberOfRowsInTableView:(NSTableView \*)tableView

{

return [\_data count];

}

-(NSView \*)tableView:(NSTableView \*)tableView viewForTableColumn:(NSTableColumn \*)tableColumn row:(NSInteger)row

{

NSString \*columnIdentifier = [tableColumn identifier];

NSTableCellView \*view = [\_dataTableView makeViewWithIdentifier:columnIdentifier owner:self];

NSUInteger index = -1;

if ([columnIdentifier isEqualToString:identifier\_index])

{

index = tb\_index;

NSArray \*subviews = [view subviews];

NSTextField \*txtField = subviews[0];

if ([\_colorRedIndex count]>0 || [\_colorGreenIndex count]>0)

{

if ([\_colorGreenIndex containsObject:[NSNumber numberWithInteger:row]])

{

txtField.drawsBackground = YES;

txtField.backgroundColor = [NSColor greenColor];

}

if ([\_colorRedIndex containsObject:[NSNumber numberWithInteger:row]])

{

txtField.drawsBackground = YES;

txtField.backgroundColor = [NSColor systemRedColor];

}

if (![\_colorGreenIndex containsObject:[NSNumber numberWithInteger:row]] && ![\_colorRedIndex containsObject:[NSNumber numberWithInteger:row]])

{

txtField.drawsBackground = YES;

txtField.backgroundColor = [NSColor grayColor];

}

}

else

{

txtField.drawsBackground = YES;

txtField.backgroundColor = [NSColor whiteColor];

}

}

if ([columnIdentifier isEqualToString:identifier\_item])

{

index = tb\_item;

NSArray \*subviews = [view subviews];

NSTextField \*txtField = subviews[0];

if (n\_double\_click == row)

{

[txtField setEditable:YES];

[txtField setBordered:NO];

}

else

{

//NSLog(@"===========n\_double\_click>>>> %zd NO",n\_double\_click);

[txtField setEditable:NO];

[txtField setBordered:NO];

}

// txtField.tag = row;

// txtField.target = self;

// [txtField setAction:@selector(btnClickItem:)];

}

if ([columnIdentifier isEqualToString:identifier\_low]){

index = tb\_lower;

}

if ([columnIdentifier isEqualToString:identifier\_upper]){

index = tb\_upper;

}

if ([columnIdentifier isEqualToString:identifier\_unit]){

index = tb\_measurement\_unit;

}

if ([columnIdentifier isEqualToString:identifier\_lsl]){

index = tb\_lsl;

}

if ([columnIdentifier isEqualToString:identifier\_usl]) {

index = tb\_usl;

}

if ([columnIdentifier isEqualToString:identifier\_apply]) {

NSArray \*subviews = [view subviews];

NSButton \*checkBoxField = subviews[0];

checkBoxField.tag = row;

checkBoxField.target = self;

[checkBoxField setAction:@selector(btnClickApply:)];

index = tb\_apply;

if ([[\_data objectAtIndex:row] count]>index)

{

[checkBoxField setState:[[\_data objectAtIndex:row][index] intValue]];

}

return view;

}

if ([columnIdentifier isEqualToString:identifier\_keynote]) {

NSArray \*subviews = [view subviews];

NSButton \*checkBoxField = subviews[0];

checkBoxField.tag = row;

checkBoxField.target = self;

[checkBoxField setAction:@selector(btnClickKeynoteApply:)];

index = tb\_keynote;

if ([[\_data objectAtIndex:row] count]>index)

{

[checkBoxField setState:[[\_data objectAtIndex:row][index] intValue]];

}

return view;

}

if ([columnIdentifier isEqualToString:identifier\_description]) {

index = tb\_description;

}

if ([columnIdentifier isEqualToString:identifier\_cpknew])

{

index = tb\_cpk\_new;

NSString \*cpk\_new\_data = [\_data objectAtIndex:row][index];

NSArray \*subviews = [view subviews];

NSTextField \*txtField = subviews[0];

txtField.drawsBackground = YES;

if ([cpk\_new\_data isNotEqualTo:@""] && ([self isPureInt:cpk\_new\_data] ||[self isPureFloat:cpk\_new\_data]))

{

float cpkL = [[m\_configDictionary valueForKey:cpk\_Lowthl] floatValue];

float cpkH = [[m\_configDictionary valueForKey:cpk\_Highthl] floatValue];

float cpk\_new\_val = [cpk\_new\_data floatValue];

if (cpk\_new\_val> cpkH)

{

txtField.backgroundColor = [NSColor yellowColor];

}

else if(cpk\_new\_val< cpkL)

{

txtField.backgroundColor = [NSColor redColor];

}

else

{

txtField.backgroundColor = [NSColor greenColor];

}

}

else

{

txtField.backgroundColor = [NSColor whiteColor];

}

}

if ([columnIdentifier isEqualToString:identifier\_command]) {

index = tb\_command;

}

if ([columnIdentifier isEqualToString:identifier\_reviewer]) {

index = tb\_reviewer;

}

if ([columnIdentifier isEqualToString:identifier\_date]) {

index = tb\_date;

}

if ([columnIdentifier isEqualToString:identifier\_bmc]) {

index = tb\_bmc;

}

if ([columnIdentifier isEqualToString:identifier\_comment]) {

index = tb\_comment;

}

if ([columnIdentifier isEqualToString:identifier\_cpk\_orig]) {

index = tb\_cpk\_orig;

NSArray \*subviews = [view subviews];

NSTextField \*txtField = subviews[0];

if ([\_colorRedIndexCpk count]>0 || [\_colorGreenIndexCpk count]>0 || [\_colorYellowIndexCpk count]>0)

{

if ([\_colorRedIndexCpk containsObject:[NSNumber numberWithInteger:row]])

{

txtField.drawsBackground = YES;

txtField.backgroundColor = [NSColor redColor];

}

if ([\_colorGreenIndexCpk containsObject:[NSNumber numberWithInteger:row]])

{

txtField.drawsBackground = YES;

txtField.backgroundColor = [NSColor greenColor];

}

if ([\_colorYellowIndexCpk containsObject:[NSNumber numberWithInteger:row]])

{

txtField.drawsBackground = YES;

txtField.backgroundColor = [NSColor yellowColor];

}

if (![\_colorRedIndexCpk containsObject:[NSNumber numberWithInteger:row]] && ![\_colorGreenIndexCpk containsObject:[NSNumber numberWithInteger:row]] && ![\_colorYellowIndexCpk containsObject:[NSNumber numberWithInteger:row]])

{

txtField.drawsBackground = YES;

txtField.backgroundColor = [NSColor whiteColor];

}

}

else

{

txtField.drawsBackground = YES;

txtField.backgroundColor = [NSColor whiteColor];

}

}

// if (index == -1)

// {

// return nil;

// }

if ([[\_data objectAtIndex:row] count]>index)

{

[[view textField] setStringValue:[\_data objectAtIndex:row][index]];

}

else

{

[[view textField] setStringValue:@""];

}

return view;

}

- (void)tableView:(NSTableView \*)tableView setObjectValue:(nullable id)object forTableColumn:(nullable NSTableColumn \*)tableColumn row:(NSInteger)row;

{

NSLog(@"====edit: %@",object);

}

- (BOOL)isAllNum:(NSString \*)string{

unichar c;

for (int i=0; i<string.length; i++) {

c=[string characterAtIndex:i];

if (!isdigit(c)) {

return NO;

}

}

return YES;

}

-(BOOL)isPureInt:(NSString \*)string

{

NSScanner\* scan = [NSScanner scannerWithString:string];

int val;

return [scan scanInt:&val] && [scan isAtEnd];

}

-(BOOL)isPureFloat:(NSString \*)string

{

NSScanner\* scan = [NSScanner scannerWithString:string];

float val;

return [scan scanFloat:&val] && [scan isAtEnd];

}

-(BOOL)isPureNumandCharacters:(NSString \*)string

{

string = [string stringByTrimmingCharactersInSet:[NSCharacterSet decimalDigitCharacterSet]];

if(string.length > 0)

{

return NO;

}

return YES;

}

-(BOOL)isOnlyhasNumberAndpointWithString:(NSString \*)string{

NSCharacterSet \*cs=[[NSCharacterSet characterSetWithCharactersInString:NUMBERS] invertedSet];

NSString \*filter=[[string componentsSeparatedByCharactersInSet:cs] componentsJoinedByString:@""];

return [string isEqualToString:filter];

}

-(void)AlertBox:(NSString \*)msgTxt withInfo:(NSString \*)strmsg

{

NSAlert \* alert = [[NSAlert alloc] init];

alert.messageText = msgTxt;

alert.informativeText = strmsg;

[alert runModal];

}

-(int)AlertBoxWith2Button:(NSString \*)msgTxt withInfo:(NSString \*)strmsg

{

NSAlert \*alert = [[NSAlert alloc] init];

[alert setMessageText:msgTxt];

[alert setInformativeText:strmsg];

[alert addButtonWithTitle:@"OK"];

[alert addButtonWithTitle:@"Cancel"];

// [alert addButtonWithTitle:@"abort"];

[alert setAlertStyle:NSAlertStyleWarning];

NSUInteger action = [alert runModal];

if(action == NSAlertFirstButtonReturn) //1000

{

return 1000;

}

else if(action == NSAlertSecondButtonReturn )//1001

{

return 1001;

}

// else if(action == NSAlertThirdButtonReturn)//1002

// {

// NSLog(@"Abort");

// }

else

{

return -1;

}

}

- (void)controlTextDidBeginEditing:(NSNotification \*)obj

{

NSInteger row =self.dataTableView.selectedRow;

editLimitRow = row;

NSLog(@"===edit==> row: %zd",row);

// NSTextField \*textF =obj.object;

// NSInteger col = [self.dataTableView columnForView:textF];

// if (col == 5 || col == 6 || col == 8)

// {

// \_data[editLimitRow][tb\_apply] = [NSNumber numberWithInt:0];

// [self.dataTableView reloadDataForRowIndexes:[NSIndexSet indexSetWithIndex:row] columnIndexes:[NSIndexSet indexSetWithIndex:7]];

// }

}

-(void)controlTextDidChange:(NSNotification \*)obj

{

NSTextField \*textF =obj.object;

NSInteger row =self.dataTableView.selectedRow;

NSInteger col = [self.dataTableView columnForView:textF];

n\_reviewer\_col = col;

//NSString \*identifier = self.dataTableView.tableColumns[col].identifier;

//NSLog(@"===edit==>identifier: %@ row:%zd col:%zd %@",identifier,row,col,textF.stringValue);

NSString \*col\_identifier = [[[self.dataTableView tableColumns] objectAtIndex:col] identifier];

//NSString \*key = [NSString stringWithFormat:@"%zd-%zd",row,col];

//[\_textEditLimitDic setValue:textF.stringValue forKey:key];

if ([col\_identifier isEqualToString:identifier\_lsl] || [col\_identifier isEqualToString:identifier\_usl]) //if (col == col edit\_new\_lsl || col == col edit\_new\_usl)

{

if ([textF.stringValue isEqualToString:@"N"] || [textF.stringValue isEqualToString:@"NA"])

{

if ([col\_identifier isEqualToString:identifier\_lsl]) //(col == col edit\_new\_lsl)

{

if ([textF.stringValue isEqualToString:@"NA"])

{

\_data[editLimitRow][tb\_lsl] = [NSString stringWithFormat:@"%@",[textF stringValue]];

}

}

if ([col\_identifier isEqualToString:identifier\_usl]) //(col == col edit\_new\_usl)

{

if ([textF.stringValue isEqualToString:@"NA"])

{

\_data[editLimitRow][tb\_usl] = [NSString stringWithFormat:@"%@",[textF stringValue]];

}

}

\_data[editLimitRow][tb\_reviewer] = @"";

\_data[editLimitRow][tb\_date] = @"";

return;

}

if(![self isOnlyhasNumberAndpointWithString:textF.stringValue])

{

[self AlertBox:@"Error:014" withInfo:@"Please input number type or NA!"];

\_data[editLimitRow][col+2] = @"";

[self.dataTableView reloadData];

return;

}

if (row ==-1)

{

}

else

{

editLimitRow = row;

}

if (row>=0 && row<[\_data count])

{

//NSLog(@"===edit==>identifier : %@ row:%zd col:%zd %@",identifier,editLimitRow,col+3,textF.stringValue);

if([col\_identifier isEqualToString:identifier\_lsl])//(col == col edit\_new\_lsl)

{

\_data[editLimitRow][tb\_lsl] = [NSString stringWithFormat:@"%@",[textF stringValue]];

}

if([col\_identifier isEqualToString:identifier\_usl]) //(col == col edit\_new\_usl)

{

\_data[editLimitRow][tb\_usl] = [NSString stringWithFormat:@"%@",[textF stringValue]];

}

\_data[editLimitRow][tb\_reviewer] = @"";

\_data[editLimitRow][tb\_date] = @"";

//\_data[editLimitRow][tb\_apply] = [NSNumber numberWithInt:0];

//[self.dataTableView reloadDataForRowIndexes:[NSIndexSet indexSetWithIndex:row] columnIndexes:[NSIndexSet indexSetWithIndex:7]];

}

}

if ([col\_identifier isEqualToString:identifier\_reviewer] || [col\_identifier isEqualToString:identifier\_date]) //(col == col edit\_reviewer\_name || col == col edit\_review\_date)

{

if (row ==-1)

{

}

else

{

editLimitRow = row;

}

if (row>=0 && row<[\_data count])

{

// NSLog(@"=====00>identifier : %@ row:%zd col:%zd %@",identifier,editLimitRow,col+3,textF.stringValue);

if ([[textF stringValue] isNotEqualTo:@""])

{

\_data[editLimitRow][col+3] = [NSString stringWithFormat:@"%@",[textF stringValue]];

if([col\_identifier isEqualToString:identifier\_reviewer])//(col == col edit\_reviewer\_name)

{

NSDateFormatter\* DateFomatter = [[NSDateFormatter alloc] init];

[DateFomatter setDateFormat:@"yyyy-MM-dd HH:mm:ss"];

NSTimeZone \*timezone = [[NSTimeZone alloc] initWithName:@"PST"];

[DateFomatter setTimeZone:timezone];

NSString\* systemTime = [DateFomatter stringFromDate:[NSDate date]];

\_data[editLimitRow][tb\_date] = systemTime;

}

}

}

}

if ([col\_identifier isEqualToString:identifier\_comment])//(col == col edit\_comment)

{

if (row ==-1)

{

}

else

{

editLimitRow = row;

}

if (row>=0 && row<[\_data count])

{

if ([[textF stringValue] isNotEqualTo:@""])

{

\_data[editLimitRow][tb\_comment] = [NSString stringWithFormat:@"%@",[textF stringValue]];

b\_ClearComment = NO;

}

else

{

\_data[editLimitRow][tb\_comment] = [NSString stringWithFormat:@"%@",[textF stringValue]];

b\_ClearComment = YES;

}

//\_data[editLimitRow][tb\_apply] = [NSNumber numberWithInt:0];

//[self.dataTableView reloadDataForRowIndexes:[NSIndexSet indexSetWithIndex:row] columnIndexes:[NSIndexSet indexSetWithIndex:7]];

}

}

}

-(void)mouseEntered:(NSEvent \*)event

{

//NSLog(@"====>>>>%@",event);

}

-(void)mouseMoved:(NSEvent \*)event

{

//NSLog(@"====>>>>%@",event);

}

- (void)keyDown:(NSEvent \*)event

{

[m\_configDictionary setValue:[NSNumber numberWithBool:NO] forKey:kInputRangeFlag];

if (![self.dataTableView isAccessibilityFocused])

{

unichar key = [[event charactersIgnoringModifiers] characterAtIndex:0];

NSString \*identifier = self.dataTableView.tableColumns[n\_reviewer\_col].identifier;

NSLog(@">key :%hu %@",key,identifier);

// if(key == 0xf700)

// {

// NSLog(@"----0xf700");

// }

// if(key == 0xf701)

// {

// NSLog(@"----0xf701");

// }

/\*if (key==13 && ([identifier isEqualToString:@"lsl"] || [identifier isEqualToString:@"usl"] || [identifier isEqualToString:@"comment"])) //press Enter

{

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

NSInteger row = [self.dataTableView selectedRow];

if (row >= 0)

{

[self reloadReviewerDate:row];

}

}

\*/

// NSString \*identifier = self.dataTableView.tableColumns[n\_reviewer\_col].identifier;

// if (n\_reviewer\_col == 9)

// {

// n\_reviewer\_col = 0;

// }

//NSLog(@"====not foucus on editor view 0x%X , %@",key,identifier);

//if (key==13 && [identifier isEqualToString:@"reviewer"]) //press Enter

// {

// [\_reviewerNameIndex removeAllObjects];

// for (int i=0; i<[\_data count]; i++)

// {

// if ([\_data[i][tb\_reviewer] isNotEqualTo:@""])

// {

// [\_reviewerNameIndex addObject:[NSNumber numberWithInt:i]];

// }

// }

// [self.dataTableView reloadData];

// [self.dataTableView setAccessibilityFocused:YES];//isAccessibilityFocused

//}

return;

}

if(![self.dataTableView selectedCell])

{

if (event.isCommandDown)

{

if ([event.characters isEqual:@"c"]) //copy

{

if([self.dataTableView selectedRow] == -1 && [self.dataTableView selectedColumn] == -1)

{

return;

}

NSLog(@"====>copy ");

}

else if ([event.characters isEqual:@"v"]) //paste

{

if(!enableEditing)

{

return;

}

NSLog(@"====>paste ");

[self.dataTableView reloadData];

//[self paste:nil];

}

else if ([event.characters isEqual:@"x"]) //cut

{

if(([self.dataTableView selectedRow] == -1 && [self.dataTableView selectedColumn] == -1) || !enableEditing)

{

return;

}

NSLog(@"====>cut ");

[self.dataTableView reloadData];

// [self cut:nil];

}

else if ([event.characters isEqual:@"z"]) // undo

{

[self.undoManager undo];

NSLog(@"====>undo ");

}

else if (event.isShiftDown && [event.characters isEqual:@"z"]) // redo

{

[self.undoManager redo];

NSLog(@"====>redo ");

}

else

{

return;

}

}

else if (event.isShiftDown)

{

NSLog(@"isShiftDown");

}

else if (event.isOptionDown)

{

NSLog(@"isOptionDown");

}

else if (event.isControlDown)

{

NSLog(@"isControlDown");

}

else

{

unichar key = [[event charactersIgnoringModifiers] characterAtIndex:0];

if(key == NSDeleteCharacter)

{

if(([self.dataTableView selectedRow] == -1 && [self.dataTableView selectedColumn] == -1)|| !enableEditing)

{

return;

}

//[self delete:nil];

NSLog(@"====>delete :%x",key);

[self.dataTableView reloadData];

return;

}

if(key == 0xf700)

{

NSLog(@"==>%@ %@",[m\_configDictionary valueForKey:kRetestSeg],[m\_configDictionary valueForKey:kRemoveFailSeg]);

if(([self.dataTableView selectedRow] == -1 && [self.dataTableView selectedColumn] == -1)|| !enableEditing)

{

return;

}

NSInteger selectRow = [self.dataTableView selectedRow]-1;

if (selectRow < 0)

{

selectRow = [self.dataTableView selectedRow];

}

\_data[selectRow][tb\_color\_by\_left]= [NSNumber numberWithInteger:selectColorBoxIndex]; //设置color By左边那个,给python生成图表用

\_data[selectRow][tb\_color\_by\_right]= [NSNumber numberWithInteger:selectColorBoxIndex2]; //设置color By左边那个,给python生成图表用

NSString \*typeZoom = [m\_configDictionary valueForKey:kzoom\_type];

\_data[selectRow][tb\_zoom\_type] = typeZoom;

NSString \*bins = [m\_configDictionary valueForKey:kBins];

\_data[selectRow][tb\_bins] = bins;

NSInteger rowActual = 0;

for (NSInteger i= 0; i<[\_ListAllItemNameArr count]; i++) //当UI 选择search 的时候，数据变了，row 也变了，要找到对应值

{

if ([\_ListAllItemNameArr[i] isEqualToString:\_data[selectRow][1]])

{

rowActual = i;

break;

}

}

tbDataTableSelectItemRow = rowActual;

click\_tb\_row = selectRow;

//click\_tb\_row = rowActual;

if (selectColorBoxIndex > 0 ) //color by 打开，用color by 那边发指令给python

{

if (n\_firstItemClick ==0|| n\_firstItemClick ==1)

{

NSString \*choose\_item\_name = [\_indexItemNameDic valueForKey:[NSString stringWithFormat:@"%zd",rowActual]];

[m\_configDictionary setValue:[NSNumber numberWithInteger:rowActual] forKey:kChooseItemIndex];

[m\_configDictionary setValue:choose\_item\_name forKey:kChooseItemName];

n\_firstItemClick =10;

}

//NSDictionary \*dic = [NSDictionary dictionaryWithObject:[m\_configDictionary valueForKey:K\_dic\_ApplyBoxCheck] forKey:applyBoxCheck];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickPlotTable object:nil userInfo:nil];

}

else if(selectColorBoxIndex2>0 )

{

if (n\_firstItemClick ==0|| n\_firstItemClick ==1)

{

NSString \*choose\_item\_name = [\_indexItemNameDic valueForKey:[NSString stringWithFormat:@"%zd",rowActual]];

[m\_configDictionary setValue:[NSNumber numberWithInteger:rowActual] forKey:kChooseItemIndex];

[m\_configDictionary setValue:choose\_item\_name forKey:kChooseItemName];

n\_firstItemClick =10;

}

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickPlotTable2 object:nil userInfo:nil];

}

else // color by 关闭。直接发指令给python

{

NSString \* itemName = [self combineItemName:[\_indexItemNameDic valueForKey:[NSString stringWithFormat:@"%zd",rowActual]]];

NSLog(@"====>down key :%x row: %ld item name: %@",key,rowActual,itemName);

// 写发送代码

// NSMutableArray \*itemArray = \_dataReverse[selectRow+n\_Start\_Data\_Col];

NSString \*itemName\_0 = [NSString stringWithFormat:@"%@\_XY",[m\_configDictionary valueForKey:kChooseItemName]];

NSInteger row\_0 = [[m\_configDictionary valueForKey:kChooseItemIndex] integerValue];

NSMutableArray \* itemData\_0 = [self calculateData:row\_0];

NSMutableArray \* itemData = [self calculateData:rowActual];

itemData\_0[tb\_correlation\_xy] = itemName\_0;

itemData[tb\_correlation\_xy] = itemName\_0;

if ([[m\_configDictionary valueForKey:kInputRangeFlag] boolValue])

{

NSString \*rangelsl = [m\_configDictionary valueForKey:krangelsl];

NSString \*rangeusl = [m\_configDictionary valueForKey:krangeusl];

itemData\_0[tb\_range\_lsl] = rangelsl;

itemData\_0[tb\_range\_usl] = rangeusl;

itemData[tb\_range\_lsl] = rangelsl;

itemData[tb\_range\_usl] = rangeusl;

NSLog(@">>>range: %@,%@",rangelsl,rangeusl);

}

else

{

NSString \*rangelsl = itemData[tb\_lower];

NSString \*rangeusl = itemData[tb\_upper];

itemData\_0[tb\_range\_lsl] = rangelsl;

itemData\_0[tb\_range\_usl] = rangeusl;

itemData[tb\_range\_lsl] = rangelsl;

itemData[tb\_range\_usl] = rangeusl;

NSLog(@".>>>range: %@,%@",rangelsl,rangeusl);

}

[self sendDataToRedis:itemName\_0 withData:itemData\_0];

[self sendDataToRedis:itemName withData:itemData];

[self sendCpkZmqMsg:itemName];

[self sendCorrelationZmqMsg:itemName];

[self sendScatterZmqMsg:itemName];

}

return;

}

if(key == 0xf701)

{

NSLog(@"==>%@ %@",[m\_configDictionary valueForKey:kRetestSeg],[m\_configDictionary valueForKey:kRemoveFailSeg]);

if(([self.dataTableView selectedRow] == -1 && [self.dataTableView selectedColumn] == -1)|| !enableEditing)

{

return;

}

NSInteger selectRow = [self.dataTableView selectedRow]+1;

if (selectRow >= [\_data count])

{

selectRow = [self.dataTableView selectedRow];

}

\_data[selectRow][tb\_color\_by\_left]= [NSNumber numberWithInteger:selectColorBoxIndex]; //设置color By左边那个,给python生成图表用

\_data[selectRow][tb\_color\_by\_right]= [NSNumber numberWithInteger:selectColorBoxIndex2]; //设置color By right那个,给python生成图表用

NSString \*typeZoom = [m\_configDictionary valueForKey:kzoom\_type];

\_data[selectRow][tb\_zoom\_type] = typeZoom;

NSString \*bins = [m\_configDictionary valueForKey:kBins];

\_data[selectRow][tb\_bins] = bins;

NSInteger rowActual = 0;

for (NSInteger i= 0; i<[\_ListAllItemNameArr count]; i++) //当UI 选择search 的时候，数据变了，row 也变了，要找到对应值

{

if ([\_ListAllItemNameArr[i] isEqualToString:\_data[selectRow][1]])

{

rowActual = i;

break;

}

}

tbDataTableSelectItemRow = rowActual;

//click\_tb\_row = rowActual;

click\_tb\_row = selectRow;

if (selectColorBoxIndex > 0) //color by 打开，用color by 那边发指令给python

{

if (n\_firstItemClick ==0|| n\_firstItemClick ==1)

{

NSString \*choose\_item\_name = [\_indexItemNameDic valueForKey:[NSString stringWithFormat:@"%zd",rowActual]];

[m\_configDictionary setValue:[NSNumber numberWithInteger:rowActual] forKey:kChooseItemIndex];

[m\_configDictionary setValue:choose\_item\_name forKey:kChooseItemName];

n\_firstItemClick =10;

}

// NSDictionary \*dic = [NSDictionary dictionaryWithObject:[m\_configDictionary valueForKey:K\_dic\_ApplyBoxCheck] forKey:applyBoxCheck];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickPlotTable object:nil userInfo:nil];

}

else if(selectColorBoxIndex2>0)

{

if (n\_firstItemClick ==0|| n\_firstItemClick ==1)

{

NSString \*choose\_item\_name = [\_indexItemNameDic valueForKey:[NSString stringWithFormat:@"%zd",rowActual]];

[m\_configDictionary setValue:[NSNumber numberWithInteger:rowActual] forKey:kChooseItemIndex];

[m\_configDictionary setValue:choose\_item\_name forKey:kChooseItemName];

n\_firstItemClick =10;

}

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickPlotTable2 object:nil userInfo:nil];

}

else // color by 关闭。直接发指令给python

{

NSString \* itemName = [self combineItemName:[\_indexItemNameDic valueForKey:[NSString stringWithFormat:@"%zd",rowActual]]];

NSLog(@"====>down key :%x row: %ld item name: %@",key,rowActual,itemName);

//写发送代码

// NSMutableArray \*itemArray = \_dataReverse[selectRow+n\_Start\_Data\_Col];

//NSLog(@"--ClickOnTableView--:%zd selectColorBoxIndex:%d, item name : %@",selectRow,selectColorBoxIndex,itemName);

NSString \*itemName\_0 = [NSString stringWithFormat:@"%@\_XY",[m\_configDictionary valueForKey:kChooseItemName]];

NSInteger row\_0 = [[m\_configDictionary valueForKey:kChooseItemIndex] integerValue];

NSMutableArray \* itemData\_0 = [self calculateData:row\_0];

NSMutableArray \* itemData = [self calculateData:rowActual];

itemData\_0[tb\_correlation\_xy] = itemName\_0;

itemData[tb\_correlation\_xy] = itemName\_0;

if ([[m\_configDictionary valueForKey:kInputRangeFlag] boolValue])

{

NSString \*rangelsl = [m\_configDictionary valueForKey:krangelsl];

NSString \*rangeusl = [m\_configDictionary valueForKey:krangeusl];

itemData\_0[tb\_range\_lsl] = rangelsl;

itemData\_0[tb\_range\_usl] = rangeusl;

itemData[tb\_range\_lsl] = rangelsl;

itemData[tb\_range\_usl] = rangeusl;

NSLog(@">>range: %@,%@",rangelsl,rangeusl);

}

else

{

NSString \*rangelsl = itemData[tb\_lower];

NSString \*rangeusl = itemData[tb\_upper];

itemData\_0[tb\_range\_lsl] = rangelsl;

itemData\_0[tb\_range\_usl] = rangeusl;

itemData[tb\_range\_lsl] = rangelsl;

itemData[tb\_range\_usl] = rangeusl;

NSLog(@".>>range: %@,%@",rangelsl,rangeusl);

}

[self sendDataToRedis:itemName\_0 withData:itemData\_0];

[self sendDataToRedis:itemName withData:itemData];

[self sendCpkZmqMsg:itemName];

[self sendCorrelationZmqMsg:itemName];

[self sendScatterZmqMsg:itemName];

}

return;

}

NSLog(@"no shorcut: %x",key);

}

}

else

{

NSLog(@"nothing");

}

}

#pragma mark **luanch function methods**

-(void)launch\_calculate\_test

{

[startPython Lanuch\_calculate];

calculateClient = [[Client alloc] init]; // connect calculate zmq for calculate.py

[calculateClient CreateRPC:calculate\_zmq\_addr withSubscriber:nil];

[calculateClient setTimeout:20\*1000];

}

-(void)launch\_retest\_plot

{

[startPython Lanuch\_retest\_plot];

retestPlotClient = [[Client alloc] init]; //

[retestPlotClient CreateRPC:retest\_plot\_zmq\_addr withSubscriber:nil];

[retestPlotClient setTimeout:20\*1000];

}

-(void)launch\_yield\_rate

{

[startPython Lanuch\_yield\_rate];

retestRateClient = [[Client alloc] init]; // connect calculate zmq for calculate.py

[retestRateClient CreateRPC:retest\_rate\_zmq\_addr withSubscriber:nil];

[retestRateClient setTimeout:20\*1000];

}

-(void)notifySetImage:(NSString \*)path

{

//NSDictionary \*dic = [NSDictionary dictionaryWithObject:path forKey:imagePath];

//[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetCpkImage object:nil userInfo:dic];

}

#pragma mark **NSSplitViewDelegate methods**

- (BOOL)splitView:(NSSplitView \*)splitView canCollapseSubview:(NSView \*)subview

{

if (subview == self.leftPane)

{

return YES;

}

else if (subview == self.rightPanel)

{

return YES;

}

return NO;

}

-(void)LoadSubView:(NSView \*)view

{

[[leftViewMain superview] replaceSubview:leftViewMain with:view];

[view setFrame:[leftViewMain frame]];

leftViewMain = view;

//[self loadView];

}

//

-(void)setLoadCsvView:(NSView \*)view

{

[self replaceView:csvViewMain with:view];

csvViewMain =view;

}

-(void)replaceView:(NSView \*)oldView with:(NSView \*)newView

{

[newView setFrame:[oldView frame]];

[[oldView superview] addSubview:newView];

[[oldView superview] replaceSubview:oldView with:newView];

[oldView setHidden:YES];

}

@end

//

// dataPlotView.m

// CPK\_Test

//

// Created by RyanGao on 2020/6/25.

// Copyright © 2020 RyanGao. All rights reserved.

//

#import "dataPlotView.h"

#import "defineHeader.h"

#import "../SCRedis.framework/Headers/RedisInterface.hpp"

#import "../SCZmq.framework/Headers/Client.h"

#import "../SCNSEventEx.framework/Headers/NSEventEx.h"

//#import "../XlsxReaderWriter.framework/Headers/BRAOfficeDocumentPackage.h"

#import "reportSettingCfg.h"

#import "keynoteSetting.h"

//#import "keynoteItemSkipSetting.h"

#import "keynote\_skip\_setting.h"

#import "reportTags.h"

#import "../SCopensslSha1.framework/Headers/sha.h"

//#import "../SCopensslSha1.framework/Headers/sha.h"

//#import "AppConstants.h"

//#import "AppUtils.h"

#import "yieldRetestRate.h"

#import "showDataControl.h"

#import <CoreGraphics/CGImage.h>

#import <Quartz/Quartz.h>

NSMutableDictionary \*m\_configDictionary;

NSInteger tbDataTableSelectItemRow;

NSMutableArray \*\_dataReverse;

NSMutableArray \*\_rawData;

int selectColorBoxIndex; //left color by

int selectColorBoxIndex2; //right color by

extern RedisInterface \*myRedis;

extern Client \*cpkClient;

extern Client \*correlationClient;

extern Client \*calculateClient;

Client \*reportExcelClient; //excel

//extern Client \*reportKeynoteClient; //keynote

Client \*reportKeynoteClient;

Client \*reportTagsClient;

extern Client \*copyImageClient; //

extern Client \*scatterClient;

extern int n\_Start\_Data\_Col;

extern int n\_Pass\_Fail\_Status;

extern int n\_Product\_Col;

extern int n\_SerialNumber;

extern int n\_SpecialBuildName\_Col;

extern int n\_Special\_Build\_Descrip\_Col;

extern int n\_StationID\_Col;

extern int n\_StartTime;

extern int n\_Version\_Col;

extern int n\_Diags\_Version\_Col;

extern int n\_OS\_VERSION\_Col;

extern int n\_passdata;

@interface dataPlotView ()

{

NSMutableArray \* lastItemSelectColorTbLeft; //记录上次结果

NSMutableArray \* lastItemSelectColorTbRight; //记录上次结果

NSInteger lastTbDataTableSelectItemRow;//记录上次结果

NSString \*desktopPath;

NSString \*userDocuments;

int n\_select\_x;

int n\_select\_y;

NSArray \*colorByName;

NSString \*inputLSL;

NSString \*inputUSL;

int last\_SilderL;

int last\_SilderR;

int last\_SilderScatter;

CGFloat \_lastCpkPaneWidth;

CGFloat \_lastCorrelationPaneWidth;

CGFloat \_lastScatterPaneWidth;

CGFloat \_lastSettingPaneWidth;

CGFloat \_lastFilter1PaneWidth;

CGFloat \_lastFilter2PaneWidth;

CGFloat \_lastPaneWidth;

CGFloat \_cpkPercentage;

CGFloat \_correlationPercentage;

CGFloat \_scatterPercentage;

CGFloat \_settingPanelPercentage;

CGFloat \_filter1lPercentage;

CGFloat \_filter2lPercentage;

BOOL b\_setRangeTxt;

int n\_flag\_scatterBtn;

int n\_flag\_cpkBtn;

}

@property (nonatomic,strong)NSMutableArray \*data; //left color by data

@property (nonatomic,strong)NSMutableArray \*data2; //right color by data

@property(strong) reportSettingCfg \*reportSetWin;

@property(strong) keynoteSetting \*keynoteSetWin;

//@property(strong) keynoteItemSkipSetting \*keynoteItemSkipWin;

@property(strong) keynote\_skip\_setting \*keynoteskipSettingWin;

@property(strong) reportTags \*reportTagsWin;

@property(strong)yieldRetestRate \*yieldRetestWin;

@property(strong)showDataControl \*showDatatWin;

@end

@implementation dataPlotView

-(instancetype)init

{

self = [super init];

if (self)

{

\_data = [[NSMutableArray alloc]init];

\_data2 = [[NSMutableArray alloc]init];

m\_configDictionary = [[NSMutableDictionary alloc]init];

\_rawData = [[NSMutableArray alloc]init];

\_dataReverse = [[NSMutableArray alloc]init];

tbDataTableSelectItemRow = -1;

lastTbDataTableSelectItemRow = -1;

selectColorBoxIndex = 0; //left

selectColorBoxIndex2 = 0; //right

lastItemSelectColorTbLeft = [NSMutableArray array];

lastItemSelectColorTbRight = [NSMutableArray array];

n\_select\_x=0;

n\_select\_y=0;

colorByName = @[Off,Version,Station\_ID,Special\_Build\_Name,Special\_Build\_Descrip,Product,Channel\_ID,Diags\_Version,OS\_VERSION];

b\_setRangeTxt = NO;

n\_flag\_scatterBtn = 0;

n\_flag\_cpkBtn = 0;

}

return self;

}

- (void)viewDidLoad {

[super viewDidLoad];

[\_colorByTableView setDelegate:self];

[\_colorByTableView setDataSource:self];

[\_colorByTableView2 setDelegate:self];

[\_colorByTableView2 setDataSource:self];

[self initColorTabView:nil];

[\_progressExcel setHidden:YES];

[\_progressKeynote setHidden:YES];

[\_cpkSaveButton setHidden:NO];

[\_correlationSaveButton setHidden:NO];

NSString \*savepath =[[NSBundle mainBundle]pathForResource:@"for\_filesave.png" ofType:nil];

NSImage \*saveIcon = [[NSImage alloc]initWithContentsOfFile:savepath];

dispatch\_async(dispatch\_get\_main\_queue(), ^{

[self.cpkSaveButton setImage:saveIcon];

[self.correlationSaveButton setImage:saveIcon];

[self.scatterSaveButton setImage:saveIcon];

});

desktopPath = [NSSearchPathForDirectoriesInDomains(NSDesktopDirectory, NSUserDomainMask, YES)objectAtIndex:0];

userDocuments = [NSSearchPathForDirectoriesInDomains(NSDocumentDirectory, NSUserDomainMask, YES) objectAtIndex:0];

[m\_configDictionary setValue:[NSNumber numberWithBool:NO] forKey:K\_dic\_ApplyBoxCheck];

[m\_configDictionary setValue:[NSNumber numberWithBool:NO] forKey:K\_dic\_Load\_Csv\_Finished];

[m\_configDictionary setValue:@"" forKey:krangelsl];

[m\_configDictionary setValue:@"" forKey:krangeusl];

[m\_configDictionary setValue:[NSNumber numberWithBool:NO] forKey:kInputRangeFlag];

//left color by

[self.colorByTableView setTarget:self];

[self.colorByTableView setDoubleAction:@selector(DblClickOnTableView:)];

[self.colorByTableView setAction:@selector(DblClickOnTableView:)];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(DblClickOnTableView:) name:kNotificationClickPlotTable object:nil];

//right color by

[self.colorByTableView2 setTarget:self];

[self.colorByTableView2 setDoubleAction:@selector(DblClickOnTableView2:)];

[self.colorByTableView2 setAction:@selector(DblClickOnTableView2:)];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(DblClickOnTableView2:) name:kNotificationClickPlotTable2 object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(ClickOnSelectXY:) name:kNotificationClickPlotTable\_selectXY object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(initColorTabView:) name:kNotificationInitColorTable object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(setUiImage:) name:kNotificationSetCpkImage object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(setUiImage:) name:kNotificationSetCorrelationImage object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(setUiImage:) name:kNotificationSetScatterImage object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(setReportButton:) name:kNotificationGenerateExcel object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(addExcelHash:) name:kNotificationAddExcelHash object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(setReportButton:) name:kNotificationGenerateKeynote object:nil];

//[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(settingTableViewData:) name:kNotificationRetestRate object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(setReportButton:) name:kNotificationToLoadCsv object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(setReportButton:) name:kNotificationToLocalLoadCsv object:nil];

[[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(setRangeLSLandUSL:) name:kNotificationSetRangeLslUsl object:nil];

[NSTimer scheduledTimerWithTimeInterval:0.02 target:self selector:@selector(OnTimer:) userInfo:nil repeats:YES];

[self.sliderL setHidden:YES];

self.sliderL.floatValue = 50;

last\_SilderL = 50;

[self.sliderR setHidden:YES];

self.sliderR.floatValue = 50;

last\_SilderR = 50;

self.sliderScatter.floatValue = 50;

last\_SilderScatter = 50;

[self.rangeLsl setHidden:YES];

[self.rangeUsl setHidden:YES];

[self.rangeTxtLsl setHidden:YES];

[self.rangeTxtUsl setHidden:YES];

[self.cpkViewWin setHidden:NO];

[self.correlationViewWin setHidden:NO];

[self.scatterViewWin setHidden:NO];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%@",@"limit"] forKey:kzoom\_type];

\_showDatatWin=[[showDataControl alloc]initWithWindowNibName:@"showDataControl"];

[\_showDatatWin.window orderFront:nil];

[\_showDatatWin.window close];

[NSEvent addLocalMonitorForEventsMatchingMask:NSEventMaskKeyDown handler:^NSEvent \* \_Nullable(NSEvent \* \_Nonnull aEvent) {

[self keyDown:aEvent];

return aEvent;

}];

[self.checkPDF setAlphaValue:0.9];

[self.checkCDF setAlphaValue:0.9];

[self.checkPDF setState:0];

[self.checkCDF setState:0];

[self sendStringToRedis:KSetPDF withData:@"0"];

[self sendStringToRedis:KSetCDF withData:@"0"];

self.cpkImageView.imageScaling = NSImageScaleProportionallyUpOrDown;

self.correlationImageView.imageScaling = NSImageScaleProportionallyUpOrDown;

self.scatterImageMapView.imageScaling = NSImageScaleProportionallyUpOrDown;

[self.sliderL setAlphaValue:0.9];

[self.sliderR setAlphaValue:0.9];

[self.sliderScatter setAlphaValue:0.9];

[self.cpkSaveButton setAlphaValue:1];

[self.correlationSaveButton setAlphaValue:1];

[self.scatterSaveButton setAlphaValue:1];

[self.cpkFitScreen setAlphaValue:1];

[self.correlationFitScreen setAlphaValue:1];

[self.scatterFitScreen setAlphaValue:1];

//self.scrollViewLeft.documentCursor = [NSCursor openHandCursor];

//self.scrollViewRight.documentCursor = [NSCursor openHandCursor];

//self.scatterScrollView.documentCursor = [NSCursor openHandCursor];

self.scrollViewLeft.backgroundColor = [NSColor colorWithWhite:0.5 alpha:0.8];

self.scrollViewRight.backgroundColor = [NSColor colorWithWhite:0.5 alpha:0.8];

self.scatterScrollView.backgroundColor = [NSColor colorWithWhite:0.5 alpha:0.8];

self.clipViewLeft.backgroundColor = [NSColor colorWithWhite:0.5 alpha:1];

self.clipViewRight.backgroundColor = [NSColor colorWithWhite:0.5 alpha:1];

self.clipViewScatter.backgroundColor = [NSColor colorWithWhite:0.5 alpha:1];

\_lastCpkPaneWidth = self.cpkViewWin.frame.size.width;

\_lastCorrelationPaneWidth = self.correlationViewWin.frame.size.width;

\_lastScatterPaneWidth = self.scatterViewWin.frame.size.width;

\_lastSettingPaneWidth =self.settingViewWin.frame.size.width;

\_lastFilter1PaneWidth = self.filter1ViewWin.frame.size.width;

\_lastFilter2PaneWidth =self.filter2ViewWin.frame.size.width;

\_lastPaneWidth = self.customerMainView.frame.size.width;

\_cpkPercentage = \_lastCpkPaneWidth/\_lastPaneWidth;

\_correlationPercentage = \_lastCorrelationPaneWidth/\_lastPaneWidth;

\_scatterPercentage = \_lastScatterPaneWidth/\_lastPaneWidth;

\_settingPanelPercentage = \_lastSettingPaneWidth/\_lastPaneWidth;

\_filter1lPercentage = \_lastFilter1PaneWidth/\_lastPaneWidth;

\_filter2lPercentage = \_lastFilter2PaneWidth /\_lastPaneWidth;

startPython = [[StartUp alloc] init];

}

- (void)keyDown:(NSEvent \*)event

{

if ([self.colorByTableView isAccessibilityFocused])

{

unichar key = [[event charactersIgnoringModifiers] characterAtIndex:0];

BOOL b\_shiftDown = NO;

if (event.isShiftDown)

{

b\_shiftDown = YES;

}

if(key == 0xf700)

{

NSInteger selectRow = [self.colorByTableView selectedRow]-1;

if (selectRow < 0)

{

selectRow = [self.colorByTableView selectedRow];

}

[self keyMoveFilter1:selectRow withShiftDown:b\_shiftDown];

}

else if (key == 0xf701)

{

NSInteger selectRow = [self.colorByTableView selectedRow]+1;

if (selectRow >= [\_data count])

{

selectRow = [self.colorByTableView selectedRow];

}

[self keyMoveFilter1:selectRow withShiftDown:b\_shiftDown];

}

}

else if ([self.colorByTableView2 isAccessibilityFocused])

{

unichar key = [[event charactersIgnoringModifiers] characterAtIndex:0];

BOOL b\_shiftDown = NO;

if (event.isShiftDown)

{

b\_shiftDown = YES;

}

if(key == 0xf700)

{

NSInteger selectRow = [self.colorByTableView2 selectedRow]-1;

if (selectRow < 0)

{

selectRow = [self.colorByTableView2 selectedRow];

}

[self keyMoveFilter2:selectRow withShiftDown:b\_shiftDown];

}

else if (key == 0xf701)

{

NSInteger selectRow = [self.colorByTableView2 selectedRow]+1;

if (selectRow >= [\_data2 count])

{

selectRow = [self.colorByTableView2 selectedRow];

}

[self keyMoveFilter2:selectRow withShiftDown:b\_shiftDown];

}

}

}

-(NSMutableArray \*)getNeedDeletDataIndex //更具UI retest 和remove fail 按钮，移除相关index 数据

{

NSString \*opt1 = [m\_configDictionary valueForKey:kRetestSeg];

NSString \*opt2 = [m\_configDictionary valueForKey:kRemoveFailSeg];

NSString \*dic\_key = [NSString stringWithFormat:@"%@&%@",opt1,opt2];

NSMutableArray \*indexArr = [m\_configDictionary valueForKey:dic\_key];

//NSLog(@"====>>>>>delet: %@",indexArr);

return indexArr;

}

-(void)OnTimer:(NSTimer \*)timer

{

NSString \*pathcpk =@"/tmp/CPK\_Log/temp/.logcpk.txt";// [NSString stringWithFormat:@"%@/CPK\_Log/temp/.logcpk.txt",desktopPath];

NSString \*logcpk = [NSString stringWithContentsOfFile:pathcpk encoding:NSUTF8StringEncoding error:nil];

if ([logcpk containsString:@"PASS"]|| [logcpk containsString:@"FAIL"])

{

[@"none" writeToFile:pathcpk atomically:YES encoding:NSUTF8StringEncoding error:nil];

NSString \*path = @"/tmp/CPK\_Log/temp/cpk.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/cpk.png",desktopPath];

NSDictionary \*dic = [NSDictionary dictionaryWithObject:path forKey:imagePath];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetCpkImage object:nil userInfo:dic];

NSLog(@">set cpk pic.");

}

NSString \*pathcpknew = @"/tmp/CPK\_Log/temp/.cpknew.txt";

NSString \*logcpknew = [NSString stringWithContentsOfFile:pathcpknew encoding:NSUTF8StringEncoding error:nil];

if ([logcpknew containsString:@"DONE,"])

{

[@"" writeToFile:pathcpknew atomically:YES encoding:NSUTF8StringEncoding error:nil];

NSArray \*subArr = [logcpknew componentsSeparatedByString:@","];

NSDictionary \*dic = [NSDictionary dictionaryWithObject:subArr[1] forKey:cpkNewNumber];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetCpkNew object:nil userInfo:dic];

}

NSString \*pathcor = @"/tmp/CPK\_Log/temp/.logcor.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.logcor.txt",desktopPath];

NSString \*logcor = [NSString stringWithContentsOfFile:pathcor encoding:NSUTF8StringEncoding error:nil];

if ([logcor containsString:@"PASS"]||[logcor containsString:@"FAIL"])

{

[@"none" writeToFile:pathcor atomically:YES encoding:NSUTF8StringEncoding error:nil];

NSString \*path = @"/tmp/CPK\_Log/temp/correlation.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/correlation.png",desktopPath];

NSDictionary \*dic = [NSDictionary dictionaryWithObject:path forKey:imagePath];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetCorrelationImage object:nil userInfo:dic];

NSLog(@">set correlation pic.");

}

NSString \*pathscatter = @"/tmp/CPK\_Log/temp/.logscatter.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.logscatter.txt",desktopPath];

NSString \*logscatter = [NSString stringWithContentsOfFile:pathscatter encoding:NSUTF8StringEncoding error:nil];

if ([logscatter containsString:@"PASS"]||[logscatter containsString:@"FAIL"])

{

[@"none" writeToFile:pathscatter atomically:YES encoding:NSUTF8StringEncoding error:nil];

NSString \*path = @"/tmp/CPK\_Log/temp/scatter.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/scatter.png",desktopPath];

NSDictionary \*dic = [NSDictionary dictionaryWithObject:path forKey:imagePath];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetScatterImage object:nil userInfo:dic];

// NSLog(@">set scatter pic.");

}

NSString \*pathcalc = @"/tmp/CPK\_Log/temp/.logcalc.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.logcalc.txt",desktopPath];

NSString \*logcalc = [NSString stringWithContentsOfFile:pathcalc encoding:NSUTF8StringEncoding error:nil];

BOOL isFinished = [[m\_configDictionary valueForKey:K\_dic\_Load\_Csv\_Finished] boolValue];

if ([logcalc containsString:@"PASS"] && isFinished)

{

// update UI display

[@"none" writeToFile:pathcalc atomically:YES encoding:NSUTF8StringEncoding error:nil];

NSString \*path = @"/tmp/CPK\_Log/temp/calculate\_param.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/calculate\_param.csv",desktopPath];

NSDictionary \*dic = [NSDictionary dictionaryWithObject:path forKey:paramPath];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetParameters object:nil userInfo:dic];

}

NSString \*pathretest = @"/tmp/CPK\_Log/temp/.logretest.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.logretest.txt",desktopPath];

NSString \*logretest = [NSString stringWithContentsOfFile:pathretest encoding:NSUTF8StringEncoding error:nil];

if ([logretest containsString:@"Finished"])

{

[@"none" writeToFile:pathretest atomically:YES encoding:NSUTF8StringEncoding error:nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationRetestRate object:nil userInfo:nil];

}

NSString \*pathexcel = @"/tmp/CPK\_Log/temp/.excel.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.excel.txt",desktopPath];

NSString \*logexcel = [NSString stringWithContentsOfFile:pathexcel encoding:NSUTF8StringEncoding error:nil];

if ([logexcel containsString:@"Finished"])

{

[@"none" writeToFile:pathexcel atomically:YES encoding:NSUTF8StringEncoding error:nil];

NSArray \*limit\_path = [logexcel componentsSeparatedByString:@","];

NSString \*path\_limit\_file = @"NULL";

if ([limit\_path count]>1)

{

path\_limit\_file = [limit\_path[1] stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

}

NSDictionary \*dic = [NSDictionary dictionaryWithObject:path\_limit\_file forKey:limit\_update\_path];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationAddExcelHash object:nil userInfo:dic];

}

NSString \*pathexcel\_hash = @"/tmp/CPK\_Log/temp/.excel\_hash.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.excel\_hash.txt",desktopPath];

NSString \*logexcel\_hash = [NSString stringWithContentsOfFile:pathexcel\_hash encoding:NSUTF8StringEncoding error:nil];

if ([logexcel\_hash containsString:@"Finished"])

{

[@"none" writeToFile:pathexcel\_hash atomically:YES encoding:NSUTF8StringEncoding error:nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationGenerateExcel object:nil userInfo:nil];

}

NSString \*pathkeynote = @"/tmp/CPK\_Log/temp/.keynote.txt";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/.keynote.txt",desktopPath];

NSString \*logkeynote = [NSString stringWithContentsOfFile:pathkeynote encoding:NSUTF8StringEncoding error:nil];

if ([logkeynote containsString:@"Finished"])

{

[@"none" writeToFile:pathkeynote atomically:YES encoding:NSUTF8StringEncoding error:nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationGenerateKeynote object:nil userInfo:nil];

}

NSString \*modulePath = [NSString stringWithFormat:@"%@/.errormodule.txt",userDocuments];

NSString \*logmodule = [NSString stringWithContentsOfFile:modulePath encoding:NSUTF8StringEncoding error:nil];

if ([logmodule length]>0)

{

[self AlertBox:@"Import Pyhton module Error!!!" withInfo:logmodule];

[@"" writeToFile:modulePath atomically:YES encoding:NSUTF8StringEncoding error:nil];

}

NSString \*pathreporttags = @"/tmp/CPK\_Log/temp/.reporttags.txt";

NSString \*logreporttags = [NSString stringWithContentsOfFile:pathreporttags encoding:NSUTF8StringEncoding error:nil];

if ([logreporttags containsString:@"Finished"])

{

[@"none" writeToFile:pathreporttags atomically:YES encoding:NSUTF8StringEncoding error:nil];

NSString \*cmdKillPythonLaunch = @"ps -ef |grep -i python |grep -i report\_targs\_test |grep -v grep|awk '{print $2}' |xargs kill -9";

system([cmdKillPythonLaunch UTF8String]);

NSString \*tags\_path = [m\_configDictionary valueForKey: KreportTagsExcelPath];

NSFileManager \*fileManager = [NSFileManager defaultManager];

BOOL isExist = [fileManager fileExistsAtPath:tags\_path];

if (isExist)

{

[self AlertBox:@"Save Report Tags Excel successful." withInfo:tags\_path];

}

else

{

[self AlertBox:@"Save Report Tags Excel Failed." withInfo:tags\_path];

}

}

// NSString \*pathretest\_plot =@"/tmp/CPK\_Log/retest/.retest\_plot.txt";

// NSString \*logContext\_plot = [NSString stringWithContentsOfFile:pathretest\_plot encoding:NSUTF8StringEncoding error:nil];

// if ([logContext\_plot containsString:@"Finished"])

// {

// [@"none" writeToFile:pathretest\_plot atomically:YES encoding:NSUTF8StringEncoding error:nil];

// [[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetRetestImage object:nil userInfo:nil];

// [[NSNotificationCenter defaultCenter]postNotificationName:kNotificationRetestRate object:nil userInfo:nil];

// }

}

-(void)updateBuildSummaryWin:(int)x

{

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetRetestImage object:nil userInfo:nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationRetestRate object:nil userInfo:nil];

}

-(void)initColorTabView:(NSNotification \*)nf

{

//NSDictionary\* info = [nf userInfo];

if ([[nf name] isEqualToString:kNotificationInitColorTable])

{

[self.retestSegment setSelectedSegment:1];

[self.removeFailSegment setSelectedSegment:0];

[self.zoomTypeSeg setSelectedSegment:0];

[m\_configDictionary setValue:[self switchZoomDataLimit:0] forKey:kzoom\_type];

[self.plotTypeSeg setSelectedSegment:0];

[m\_configDictionary setValue:[self switchRetest:1] forKey:kRetestSeg];

[m\_configDictionary setValue: [self switchRemoveFail:0] forKey:kRemoveFailSeg];

NSLog(@"Init> %@ %@",[m\_configDictionary valueForKey:kRetestSeg],[m\_configDictionary valueForKey:kRemoveFailSeg]);

[self.txtBins setIntValue:250];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%@",@"250"] forKey:kBins];

[self.colorByBox removeAllItems];

[self.colorByBox addItemsWithObjectValues:colorByName];

[self.colorByBox selectItemAtIndex:0];

[\_data removeAllObjects];

[self.colorByBox2 removeAllItems];

[self.colorByBox2 addItemsWithObjectValues:colorByName];

[self.colorByBox2 selectItemAtIndex:0];

[\_data2 removeAllObjects];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"none\_pic.png" ofType:nil];

[self setCpkImage:picPath];

[self setCorrelationImage:picPath];

[m\_configDictionary setValue:@[Off] forKey:kSelectColorByTableRowsLeft];

[m\_configDictionary setValue:@[Off] forKey:kSelectColorByTableRowsRight];

[\_data removeAllObjects];

[\_data2 removeAllObjects];

selectColorBoxIndex = 0;

selectColorBoxIndex2 = 0;

[self.colorByTableView reloadData];

[self initSplitScatter];

[self setRangeCtlHidden:YES];

[self.colorByTableView reloadData];

[self.colorByTableView2 reloadData];

[self.checkPDF setState:0];

[self.checkCDF setState:0];

[self sendStringToRedis:KSetPDF withData:@"0"];

[self sendStringToRedis:KSetCDF withData:@"0"];

}

}

-(void)setUiImage:(NSNotification \*)nf

{

NSString \* name = [nf name];

if ([ name isEqualToString:kNotificationSetCpkImage])

{

NSDictionary \* dic = [nf userInfo];

NSString \* path = [dic valueForKey:imagePath];

[self setCpkImage:path];

}

else if([ name isEqualToString:kNotificationSetCorrelationImage])

{

NSDictionary \* dic = [nf userInfo];

NSString \* path = [dic valueForKey:imagePath];

[self setCorrelationImage:path];

}

else if([ name isEqualToString:kNotificationSetScatterImage])

{

NSDictionary \* dic = [nf userInfo];

NSString \* path = [dic valueForKey:imagePath];

[self setScatterImage:path];

}

}

-(void)addExcelHash:(NSNotification \*)nf

{

NSString \* name = [nf name];

NSDictionary\* info = [nf userInfo];

if ([ name isEqualToString:kNotificationAddExcelHash])

{

NSString \*hash\_Path = @"/tmp/CPK\_Log/temp/data\_hash.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/data\_hash.csv",desktopPath];

NSString \*str = [NSString stringWithContentsOfFile:hash\_Path encoding:NSUTF8StringEncoding error:nil];

NSString \*limit\_update\_excel\_path = [info valueForKey:limit\_update\_path];

NSString \*update\_limit\_tmp\_csv = @"/tmp/CPK\_Log/temp/update\_limit\_temp\_for\_hash.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/update\_limit\_temp\_for\_hash.csv",desktopPath]; //把Limit Update 转换成csv，方便计算Excel 第一个sheet的hash code，写入到第二个sheet,目的是只计算第一个sheet的内容，因为第二个sheet 要写入hash 值，所以内容会变化

NSFileManager \*manager = [NSFileManager defaultManager];

[manager removeItemAtPath:update\_limit\_tmp\_csv error:nil];

/\* [self changeXlsxTocsv:limit\_update\_excel\_path toTxt:update\_limit\_tmp\_csv];

[NSThread sleepForTimeInterval:1.5];

if (![manager fileExistsAtPath:update\_limit\_tmp\_csv])

{

for (int i=0; i<5; i++)

{

[self changeXlsxTocsv:limit\_update\_excel\_path toTxt:update\_limit\_tmp\_csv];

[NSThread sleepForTimeInterval:3\*i];

if ([manager fileExistsAtPath:update\_limit\_tmp\_csv])

{

break;

}

}

}

\*/

NSMutableArray \*msgArraycsv = [NSMutableArray arrayWithObjects:limit\_update\_excel\_path,update\_limit\_tmp\_csv,nil];

NSString \*itemNameCsv = @"excel\_limit\_update\_to\_csv\_report";

[self sendDataToRedis:itemNameCsv withData:msgArraycsv];

[self sendExcelZmqMsg:itemNameCsv];

for (int i=0; i<30; i++)

{

[NSThread sleepForTimeInterval:2.0];

if ([manager fileExistsAtPath:update\_limit\_tmp\_csv])

{

break;

}

if (i==59)

{

[self sendDataToRedis:itemNameCsv withData:msgArraycsv];

[self sendExcelZmqMsg:itemNameCsv];

for (int j=0; j<60; i++)

{

[NSThread sleepForTimeInterval:2.0];

if ([manager fileExistsAtPath:update\_limit\_tmp\_csv])

{

break;

}

if(j==59)

{

[self AlertBox:@"Error:018" withInfo:@"Limit Update Excel file sheet1 calculate hash code error.!!!"];

return;

}

}

}

}

NSString \* excel\_sheet1\_hash = [self opensslSha1FilePath:update\_limit\_tmp\_csv];

NSString \*hash\_content = [ NSString stringWithFormat:@"%@7,%@,%@",str,[limit\_update\_excel\_path lastPathComponent],excel\_sheet1\_hash];

[hash\_content writeToFile:hash\_Path atomically:YES encoding:NSUTF8StringEncoding error:nil];

NSString \*update\_limit\_path = [NSString stringWithFormat:@"%@/CPK\_Log",desktopPath];

NSString \*push2git = [m\_configDictionary valueForKey:kpush2GitHub];

NSString \*gitAddr = [m\_configDictionary valueForKey:kgitWebAddr];

NSString \*gitComment = [m\_configDictionary valueForKey:kgitComment];

NSMutableArray \*msgArray = [NSMutableArray arrayWithObjects:hash\_Path,update\_limit\_path,push2git,gitAddr,gitComment,nil];

NSString \*itemName = @"generate\_excel\_sheet1\_hash";

NSLog(@">generate excel sheet1 hash: name: %@,msgArray:%@",itemName,msgArray);

[self sendDataToRedis:itemName withData:msgArray];

[self sendExcelZmqMsg:itemName];

}

}

/\*

-(void)changeXlsxTocsv:(NSString \*)excelpath toTxt:(NSString \*)txtpath

{

NSString \*launchPath = [self taskLaunchPath];

NSMutableArray \*args = [NSMutableArray arrayWithCapacity:0];

[args addObject:@"txt"];

[args addObject:excelpath];

[args addObject:txtpath];

[self launch:launchPath arguments:args index:0];

}

- (void)launch:(NSString \*)launchPath arguments:(NSArray \*)args index:(NSInteger)index {

NSTask \*task = [[NSTask alloc] init];

[task setLaunchPath:launchPath];

[task setArguments:args];

[self updateEnvironmentForTask:task];

NSPipe \*pipe = [NSPipe pipe];

[task setStandardOutput:pipe];

[task setStandardError:pipe];

[task launch];

}

- (void)updateEnvironmentForTask:(NSTask \*)task {

NSMutableDictionary \*env = [NSMutableDictionary dictionaryWithDictionary:task.environment];

[env removeObjectForKey:kMallocNanoZone];

[task setEnvironment:env];

}

- (NSString \*)taskLaunchPath {

return [[self binDirectoryPath] stringByAppendingPathComponent:APCmdName];

}

- (void)fileHandleReadObserver:(NSPipe \*)pipe {

NSFileHandle \*fileHandle = [pipe fileHandleForReading];

//[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(fileHandleReadCompleted:) name:NSFileHandleReadToEndOfFileCompletionNotification object:fileHandle];

[fileHandle readToEndOfFileInBackgroundAndNotify];

}

- (NSString \*)binDirectoryPath {

return [[self vectorDirectoryPath] stringByAppendingPathComponent:APCmdLocDirpath];

}

- (NSString \*)vectorDirectoryPath {

return [[NSBundle mainBundle] pathForResource:APVectorDirname ofType:nil];

}

\*/

-(void)setReportButton:(NSNotification \*)nf

{

NSString \* name = [nf name];

if ([ name isEqualToString:kNotificationGenerateExcel])

{

[\_btn\_report\_excel setEnabled:YES];

[\_progressExcel stopAnimation:nil];

[\_progressExcel setHidden:YES];

NSString \*cmdKillPythonLaunch = @"ps -ef |grep -i python |grep -i report\_excel\_test.py |grep -v grep|awk '{print $2}' |xargs kill -9";

system([cmdKillPythonLaunch UTF8String]);

NSString \*nameExcel = [NSString stringWithContentsOfFile:@"/tmp/CPK\_Log/temp/.excelreportname.txt" encoding:NSUTF8StringEncoding error:nil];

NSString \*namePath = [NSString stringWithFormat:@"%@/CPK\_Log/%@",desktopPath,nameExcel];

NSFileManager \*fileManager = [NSFileManager defaultManager];

BOOL isExist = [fileManager fileExistsAtPath:namePath];

if (isExist)

{

[self AlertBox:@"Save Excel successful." withInfo:namePath];

}

else

{

[self AlertBox:@"Save Excel Failed." withInfo:namePath];

}

}

else if ([ name isEqualToString:kNotificationGenerateKeynote])

{

[\_btn\_report\_keynote setEnabled:YES];

[\_progressKeynote stopAnimation:nil];

[\_progressKeynote setHidden:YES];

NSString \*cmdKillPythonLaunch = @"ps -ef |grep -i python |grep -i report\_keynote\_test |grep -v grep|awk '{print $2}' |xargs kill -9";

NSString \*cmdKillKeynote = @"ps -ef |grep -i Keynote |grep -i Keynote |grep -v grep|awk '{print $2}' |xargs kill -9";

system([cmdKillPythonLaunch UTF8String]);

system([cmdKillKeynote UTF8String]);

NSString \*nameKeynote = [NSString stringWithContentsOfFile:@"/tmp/CPK\_Log/temp/.keynotereportname.txt" encoding:NSUTF8StringEncoding error:nil];

NSFileManager \*fileManager = [NSFileManager defaultManager];

BOOL isExist = [fileManager fileExistsAtPath:nameKeynote];

if (isExist)

{

[self AlertBox:@"Save Keynote successful." withInfo:nameKeynote];

}

else

{

[self AlertBox:@"Save Keynote Failed." withInfo:nameKeynote];

}

}

else if ([ name isEqualToString:kNotificationToLoadCsv])

{

[\_btn\_report\_keynote setEnabled:YES];

[\_btn\_report\_excel setEnabled:YES];

}

else if ([ name isEqualToString:kNotificationToLocalLoadCsv])

{

[\_btn\_report\_excel setEnabled:NO];

[\_btn\_report\_keynote setEnabled:NO];

}

}

-(void)setRangeLSLandUSL:(NSNotification \*)nf

{

[m\_configDictionary setValue:[NSNumber numberWithBool:NO] forKey:kInputRangeFlag];

NSDictionary \* dic = [nf userInfo];

NSString \*lsl = [dic valueForKey:krangelsl];

NSString \*usl = [dic valueForKey:krangeusl];

dispatch\_async(dispatch\_get\_main\_queue(), ^{

[self.rangeTxtLsl setStringValue:lsl];

[self.rangeTxtUsl setStringValue:usl];

});

inputUSL = usl;

inputLSL = lsl;

}

-(void)sendDataToRedis:(NSString \*)name withData:(NSMutableArray \*)arrData

{

if (myRedis)

{

myRedis->SetString([name UTF8String],[[NSString stringWithFormat:@"%@",arrData] UTF8String]);

}

else

{

[self AlertBox:@"Error:027" withInfo:@"Redis server is shut down!"];

}

//NSLog(@"--->>set name to redis:%@ %@",name,arrData);

}

-(void)sendStringToRedis:(NSString \*)name withData:(NSString \*)strData

{

if (myRedis)

{

myRedis->SetString([name UTF8String],[strData UTF8String]);

}

else

{

[self AlertBox:@"Error:027" withInfo:@"Redis server is shut down!"];

}

}

-(NSString \*)sendCpkZmqMsg:(NSString \*)name

{

NSString \*file1 = @"/tmp/CPK\_Log/temp/cpk.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/cpk.png",desktopPath];

NSFileManager \*manager = [NSFileManager defaultManager];

[manager removeItemAtPath:file1 error:nil];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"cpk.png" ofType:nil];

[manager copyItemAtPath:picPath toPath:file1 error:nil];

int ret = [cpkClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [cpkClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python error");

}

NSLog(@"app->get response from python: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendCorrelationZmqMsg:(NSString \*)name

{

NSString \*file1 = @"/tmp/CPK\_Log/temp/correlation.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/correlation.png",desktopPath];

NSFileManager \*manager = [NSFileManager defaultManager];

[manager removeItemAtPath:file1 error:nil];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"correlation.png" ofType:nil];

[manager copyItemAtPath:picPath toPath:file1 error:nil];

int ret = [correlationClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [correlationClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python error");

}

NSLog(@"app->get response from python: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendScatterZmqMsg:(NSString \*)name

{

NSString \*file1 = @"/tmp/CPK\_Log/temp/scatter.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/scatter.png",desktopPath];

NSFileManager \*manager = [NSFileManager defaultManager];

[manager removeItemAtPath:file1 error:nil];

NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"scatter.png" ofType:nil];

[manager copyItemAtPath:picPath toPath:file1 error:nil];

NSLog(@">set send Scatter Zmq Msg:%@",name);

int ret = [scatterClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [scatterClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq for python error");

}

NSLog(@"app->scatter get response from python: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendExcelZmqMsg:(NSString \*)name //excel zmq

{

int ret = [reportExcelClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [reportExcelClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq excel for python error");

}

NSLog(@"app->get response from excel python: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendReportTagsZmqMsg:(NSString \*)name //

{

int ret = [reportTagsClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [reportTagsClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq report tags for python error");

}

NSLog(@"app->get response from report tags python: %@",response);

return response;

}

return nil;

}

-(NSString \*)sendKeynoteZmqMsg:(NSString \*)name //keynote zmq

{

int ret = [reportKeynoteClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [reportKeynoteClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq keynote for python error");

}

NSLog(@"app->get response from keynote python: %@",response);

return response;

}

return nil;

}

-(NSString \*)combineItemName:(NSString \*)name

{

NSString \*str\_name = @"";

// 传过来的的name 后面已经有##，因name 是自动往后拼接的##

str\_name = [NSString stringWithFormat:@"%@%@&%@",name,[m\_configDictionary valueForKey:kRetestSeg],[m\_configDictionary valueForKey:kRemoveFailSeg]];

return str\_name;

}

-(void)toClickOnTableView:(NSNotification \*)nf

{

}

-(void)ClickOnSelectXY:(NSNotification \*)nf

{

NSDictionary \* dic = [nf userInfo];

int xy = [[dic valueForKey:selectXY] intValue];

[self getTwoColorTableDataAndSend:xy];

}

-(IBAction)DblClickOnTableView:(id )sender

{

NSInteger row = [self.colorByTableView selectedRow];

[self FilterBy1:row];

}

-(void)FilterBy1:(NSInteger)row

{

if (row == -1 && selectColorBoxIndex2 == 0)

{

NSLog(@"--select item is wrong--++-!!!");

return;

}

bool checkApplyBox = [[m\_configDictionary valueForKey:K\_dic\_ApplyBoxCheck] boolValue];

NSMutableArray \*selectItem = [NSMutableArray array];

NSIndexSet \*rowIndexes = [self.colorByTableView selectedRowIndexes];

NSLog(@">>select: %@",rowIndexes);

if ([rowIndexes count]) {

[rowIndexes enumerateIndexesUsingBlock:^(NSUInteger idx, BOOL \* \_Nonnull stop) {

[selectItem addObject:\_data[idx]];

}];

[m\_configDictionary setValue:selectItem forKey:kSelectColorByTableRowsLeft];

if (tbDataTableSelectItemRow>=0)

{

NSMutableArray \*selectItemColorTbRight = [NSMutableArray arrayWithArray:[m\_configDictionary valueForKey:kSelectColorByTableRowsRight]];

if ([selectItem isNotEqualTo:lastItemSelectColorTbLeft] || lastTbDataTableSelectItemRow!= tbDataTableSelectItemRow|| selectItemColorTbRight !=lastItemSelectColorTbRight ||checkApplyBox) //判断是否点击相同的item，如果是相同item，就直接返回

{

lastItemSelectColorTbLeft = selectItem;

lastItemSelectColorTbRight = selectItemColorTbRight;

lastTbDataTableSelectItemRow = tbDataTableSelectItemRow;

}

else

{

lastItemSelectColorTbLeft = selectItem;

lastItemSelectColorTbRight = selectItemColorTbRight;

lastTbDataTableSelectItemRow = tbDataTableSelectItemRow;

NSLog(@"=====click the same items");

return;

}

[self getTwoColorTableDataAndSend:-1];

}

else

{

[self AlertBox:@"Warning!!!" withInfo:@"Please select item firstly!!!"];

}

}

else

{

NSLog(@">>> %@",Off);

[m\_configDictionary setValue:@[Off] forKey:kSelectColorByTableRowsLeft];

}

}

-(void)keyMoveFilter1:(NSInteger)row withShiftDown:(BOOL)status

{

if (row == -1 && selectColorBoxIndex2 == 0)

{

return;

}

bool checkApplyBox = [[m\_configDictionary valueForKey:K\_dic\_ApplyBoxCheck] boolValue];

NSMutableArray \*selectItem = [NSMutableArray array];

NSIndexSet \*rowIndexes = [self.colorByTableView selectedRowIndexes];

if ([rowIndexes count])

{

if (status)

{

[rowIndexes enumerateIndexesUsingBlock:^(NSUInteger idx, BOOL \* \_Nonnull stop) {

[selectItem addObject:\_data[idx]];

}];

}

[selectItem addObject:\_data[row]];

[m\_configDictionary setValue:selectItem forKey:kSelectColorByTableRowsLeft];

if (tbDataTableSelectItemRow>=0)

{

NSMutableArray \*selectItemColorTbRight = [NSMutableArray arrayWithArray:[m\_configDictionary valueForKey:kSelectColorByTableRowsRight]];

if ([selectItem isNotEqualTo:lastItemSelectColorTbLeft] || lastTbDataTableSelectItemRow!= tbDataTableSelectItemRow|| selectItemColorTbRight !=lastItemSelectColorTbRight ||checkApplyBox) //判断是否点击相同的item，如果是相同item，就直接返回

{

lastItemSelectColorTbLeft = selectItem;

lastItemSelectColorTbRight = selectItemColorTbRight;

lastTbDataTableSelectItemRow = tbDataTableSelectItemRow;

}

else

{

lastItemSelectColorTbLeft = selectItem;

lastItemSelectColorTbRight = selectItemColorTbRight;

lastTbDataTableSelectItemRow = tbDataTableSelectItemRow;

NSLog(@"=====click the same items");

return;

}

[self getTwoColorTableDataAndSend:-1 withLeftRow:row withRightRow:0];

}

else

{

[self AlertBox:@"Warning!!!" withInfo:@"Please select item firstly!!!."];

}

}

}

-(void)FilterBy2:(NSInteger)row

{

if (row == -1 && selectColorBoxIndex==0) {

NSLog(@"--select item is wrong- 2--!!!");

return;

}

bool checkApplyBox = [[m\_configDictionary valueForKey:K\_dic\_ApplyBoxCheck] boolValue];

NSMutableArray \*selectItem = [NSMutableArray array];

NSIndexSet \*rowIndexes = [self.colorByTableView2 selectedRowIndexes];

if ([rowIndexes count])

{

[rowIndexes enumerateIndexesUsingBlock:^(NSUInteger idx, BOOL \* \_Nonnull stop)

{

[selectItem addObject:\_data2[idx]];

}];

[m\_configDictionary setValue:selectItem forKey:kSelectColorByTableRowsRight];

if (tbDataTableSelectItemRow>=0)

{

NSMutableArray \*selectItemColorTbLeft = [NSMutableArray arrayWithArray:[m\_configDictionary valueForKey:kSelectColorByTableRowsLeft]];

if ([selectItem isNotEqualTo:lastItemSelectColorTbRight] ||[selectItemColorTbLeft isNotEqualTo:lastItemSelectColorTbLeft] ||lastTbDataTableSelectItemRow!= tbDataTableSelectItemRow||checkApplyBox) //判断是否点击相同的item，如果是相同item，就直接返回

{

lastItemSelectColorTbLeft = selectItemColorTbLeft;

lastItemSelectColorTbRight = selectItem;

lastTbDataTableSelectItemRow = tbDataTableSelectItemRow;

}

else

{

lastItemSelectColorTbLeft = selectItemColorTbLeft;

lastItemSelectColorTbRight = selectItem;

lastTbDataTableSelectItemRow = tbDataTableSelectItemRow;

NSLog(@"=====click the same items");

return;

}

[self getTwoColorTableDataAndSend:-1];

//[self getTwoColorTableDataAndSend:-1 withLeftRow:0 withRightRow:row];

}

else

{

[self AlertBox:@"Warning!!!" withInfo:@"Please select item firstly!!!"];

}

}

else

{

NSLog(@"==>>> %@",Off);

[m\_configDictionary setValue:@[Off] forKey:kSelectColorByTableRowsLeft];

}

}

-(void)keyMoveFilter2:(NSInteger)row withShiftDown:(BOOL)status

{

if (row == -1 && selectColorBoxIndex==0) {

return;

}

bool checkApplyBox = [[m\_configDictionary valueForKey:K\_dic\_ApplyBoxCheck] boolValue];

NSMutableArray \*selectItem = [NSMutableArray array];

NSIndexSet \*rowIndexes = [self.colorByTableView2 selectedRowIndexes];

if ([rowIndexes count])

{

if (status)

{

[rowIndexes enumerateIndexesUsingBlock:^(NSUInteger idx, BOOL \* \_Nonnull stop)

{

[selectItem addObject:\_data2[idx]];

}];

}

[selectItem addObject:\_data2[row]];

[m\_configDictionary setValue:selectItem forKey:kSelectColorByTableRowsRight];

if (tbDataTableSelectItemRow>=0)

{

NSMutableArray \*selectItemColorTbLeft = [NSMutableArray arrayWithArray:[m\_configDictionary valueForKey:kSelectColorByTableRowsLeft]];

if ([selectItem isNotEqualTo:lastItemSelectColorTbRight] ||[selectItemColorTbLeft isNotEqualTo:lastItemSelectColorTbLeft] ||lastTbDataTableSelectItemRow!= tbDataTableSelectItemRow||checkApplyBox) //判断是否点击相同的item，如果是相同item，就直接返回

{

lastItemSelectColorTbLeft = selectItemColorTbLeft;

lastItemSelectColorTbRight = selectItem;

lastTbDataTableSelectItemRow = tbDataTableSelectItemRow;

}

else

{

lastItemSelectColorTbLeft = selectItemColorTbLeft;

lastItemSelectColorTbRight = selectItem;

lastTbDataTableSelectItemRow = tbDataTableSelectItemRow;

NSLog(@"=====click the same items");

return;

}

[self getTwoColorTableDataAndSend:-1 withLeftRow:0 withRightRow:row];

}

else

{

[self AlertBox:@"Warning!!!" withInfo:@"Please select item firstly!!!."];

}

}

}

-(void)DblClickOnTableView2:(id )sender

{

NSInteger row = [self.colorByTableView2 selectedRow];

[self FilterBy2:row];

}

-(NSArray\*)combineMutiArray:(NSMutableArray \*)arrayLeft withArray:(NSMutableArray \*)arrayRight withDeleteArray:(NSMutableArray \*)array3

{

NSPredicate \* filterPredicate\_same = [NSPredicate predicateWithFormat:@"SELF IN %@",arrayLeft];

NSArray \* filter\_no = [arrayRight filteredArrayUsingPredicate:filterPredicate\_same];

// NSLog(@"%@",filter\_no);

NSPredicate \* filterPredicate1 = [NSPredicate predicateWithFormat:@"NOT (SELF IN %@)",arrayLeft];

NSArray \* filter1 = [arrayRight filteredArrayUsingPredicate:filterPredicate1];

//找到在arr1中不在数组arr2中的数据

NSPredicate \* filterPredicate2 = [NSPredicate predicateWithFormat:@"NOT (SELF IN %@)",arrayRight];

NSArray \* filter2 = [arrayLeft filteredArrayUsingPredicate:filterPredicate2];

//拼接数组

NSMutableArray \*array = [NSMutableArray arrayWithArray:filter1];

[array addObjectsFromArray:filter2];

NSArray \*result = [[filter\_no arrayByAddingObjectsFromArray:array] arrayByAddingObjectsFromArray:array3];

//NSLog(@"==> %@",result);

return result;

// NSPredicate \* filter\_same = [NSPredicate predicateWithFormat:@"SELF IN %@",selectItemColorTbItemLeft]; //找到相同元素

// NSArray \* filter\_selectItemColorTbItem = [selectItemColorTbItemRight filteredArrayUsingPredicate:filter\_same];

}

-(void)getTwoColorTableDataAndSend:(int)xy //计算两个filter 选择的值

{

if (selectColorBoxIndex ==0 && selectColorBoxIndex2 == 0)

{

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

return;

}

NSMutableArray \*delectArrIndex = [self getNeedDeletDataIndex];

NSArray \*selectItemColorTbItemLeft = [m\_configDictionary valueForKey:kSelectColorByTableRowsLeft];

NSArray \*selectItemColorTbItemRight = [m\_configDictionary valueForKey:kSelectColorByTableRowsRight];

NSArray \*itemsArr = \_dataReverse[selectColorBoxIndex]; //left

NSUInteger itemCountL = [itemsArr count];

NSUInteger selectCountL = [selectItemColorTbItemLeft count];

NSMutableArray \*itemDataIndexLeft = [NSMutableArray array];

NSInteger row\_left = [self.colorByTableView selectedRow];

if (selectColorBoxIndex >0 && row\_left>= 0)

{

for (int i=0; i<selectCountL; i++) // color by table select item,显示item名字

{

NSMutableArray \*tmp = [NSMutableArray array];

for (int j =0; j<itemCountL; j++)

{

if (j<tb\_data\_start)

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

if ([itemsArr[j] isEqualTo:selectItemColorTbItemLeft[i]])

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

}

[itemDataIndexLeft addObject:tmp];

}

}

else

{

for (int i=0; i<selectCountL; i++) // color by table select item,显示item名字

{

NSMutableArray \*tmp = [NSMutableArray array];

for (int j =0; j<itemCountL; j++)

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

[itemDataIndexLeft addObject:tmp];

}

}

// NSLog(@"=====item index left: %@",itemDataIndexLeft);

NSArray \*itemsArr2 = \_dataReverse[selectColorBoxIndex2]; //right

NSUInteger itemCountR = [itemsArr2 count];

NSUInteger selectCountR = [selectItemColorTbItemRight count];

NSMutableArray \*itemDataIndexRight = [NSMutableArray array];

NSInteger row\_right = [self.colorByTableView2 selectedRow];

if (selectColorBoxIndex2>0 && row\_right>=0)

{

for (int i=0; i<selectCountR; i++) // color by table select item,显示item名字

{

NSMutableArray \*tmp = [NSMutableArray array];

for (int j =0; j<itemCountR; j++)

{

if (j<tb\_data\_start)

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

if ([itemsArr2[j] isEqualTo:selectItemColorTbItemRight[i]])

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

}

[itemDataIndexRight addObject:tmp];

}

}

else

{

for (int i=0; i<selectCountR; i++) // color by table select item,显示item名字

{

NSMutableArray \*tmp = [NSMutableArray array];

for (int j =0; j<itemCountR; j++)

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

[itemDataIndexRight addObject:tmp];

}

}

//NSLog(@"=====item index right: %@",itemDataIndexRight);

NSMutableArray \*selectItemsIndex = [NSMutableArray array];

NSMutableArray \*selectItemsName = [NSMutableArray array];

for (int m = 0; m<[itemDataIndexLeft count]; m++)

{

for (int n = 0; n<[itemDataIndexRight count]; n++)

{

NSPredicate \* filter\_same = [NSPredicate predicateWithFormat:@"SELF IN %@",itemDataIndexLeft[m]]; //找到相同元素

NSArray \* filter\_selectItem = [itemDataIndexRight[n] filteredArrayUsingPredicate:filter\_same];

[selectItemsIndex addObject:filter\_selectItem];

[selectItemsName addObject:[NSString stringWithFormat:@"%@&%@",selectItemColorTbItemLeft[m],selectItemColorTbItemRight[n]]];

}

}

//NSLog(@"=====>select item index: %@",selectItemsIndex);

NSMutableArray \*itemsData\_0 = [NSMutableArray array];

NSInteger itemRow\_0 = [[m\_configDictionary valueForKey:kChooseItemIndex] integerValue];

NSMutableArray \*itemsData = [NSMutableArray array];

NSMutableArray \*snData = [NSMutableArray array];

NSMutableString \*colorItemName = [NSMutableString string];

for (int k = 0; k<[selectItemsIndex count]; k++)

{

//NSLog(@"====<<keep>>: %@",selectItemsIndex[k]);

//NSLog(@"====<<delete>>: %@",delectArrIndex);

int last\_selectItemsIndex = -1;

for (int h = 0; h<[selectItemsIndex[k] count]; h++)

{

if (last\_selectItemsIndex == [selectItemsIndex[k][h] intValue]) //去掉重复的

{

continue;

}

if (![delectArrIndex containsObject:selectItemsIndex[k][h]]) //在index delete 列没有的元素

{

int okrow = [selectItemsIndex[k][h] intValue];

if ([\_dataReverse[tbDataTableSelectItemRow+n\_Start\_Data\_Col] count]>okrow)

{

[itemsData addObject:\_dataReverse[tbDataTableSelectItemRow+n\_Start\_Data\_Col][okrow]];

[snData addObject:\_dataReverse[n\_SerialNumber][okrow]];

}

else

{

[itemsData addObject:@""];

[snData addObject:@""];

}

if ([\_dataReverse[itemRow\_0+n\_Start\_Data\_Col] count]>okrow)

{

[itemsData\_0 addObject:\_dataReverse[itemRow\_0+n\_Start\_Data\_Col][okrow]];

}

else

{

[itemsData\_0 addObject:@""];

}

//NSLog(@"===??>>>> %@",\_dataReverse[tbDataTableSelectItemRow+n\_Start\_Data\_Col][okrow]);

}

last\_selectItemsIndex = [selectItemsIndex[k][h] intValue];

}

[itemsData addObject:End\_Data];

[snData addObject:End\_Data];

[itemsData\_0 addObject:End\_Data];

[colorItemName appendString:[NSString stringWithFormat:@"%@##",selectItemsName[k]]];

}

//NSLog(@"===>>?>>>> %@",itemsData);

NSString \* itemName = [self combineItemName:colorItemName];

if (xy==-1)

{

// do nothing

}

else

{

itemName = [NSString stringWithFormat:@"%@$$%d",itemName,xy];

}

NSLog(@"======send item name to redis: %@ itemsData count:%zd",itemName,[itemsData count]);

itemsData[tb\_color\_by\_left]= [NSNumber numberWithInteger:selectColorBoxIndex]; //设置color By左边那个,给python生成图表用

itemsData[tb\_color\_by\_right]= [NSNumber numberWithInteger:selectColorBoxIndex2]; //设置color By左边那个,给python生成图表用

NSString \*typeZoom = [m\_configDictionary valueForKey:kzoom\_type];

itemsData[tb\_zoom\_type] = typeZoom;

NSString \*bins = [m\_configDictionary valueForKey:kBins];

itemsData[tb\_bins] = bins;

itemsData\_0[tb\_color\_by\_left]= [NSNumber numberWithInteger:selectColorBoxIndex]; //设置color By左边那个,给python生成图表用

itemsData\_0[tb\_color\_by\_right]= [NSNumber numberWithInteger:selectColorBoxIndex2]; //设置color By左边那个,给python生成图表用

itemsData\_0[tb\_zoom\_type] = typeZoom;

itemsData\_0[tb\_bins] = bins;

NSString \*itemName\_0 = [NSString stringWithFormat:@"%@\_XY",[m\_configDictionary valueForKey:kChooseItemName]];

itemsData[tb\_correlation\_xy] = itemName\_0;

itemsData\_0[tb\_correlation\_xy] = itemName\_0;

NSDictionary \*dic = [NSDictionary dictionaryWithObjectsAndKeys:snData,kSerial\_number,itemsData,kData\_Value, nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationShowData object:nil userInfo:dic];

if (b\_setRangeTxt)

{

b\_setRangeTxt = NO;

NSString \*rangelsl = [m\_configDictionary valueForKey:krangelsl];

NSString \*rangeusl = [m\_configDictionary valueForKey:krangeusl];

itemsData\_0[tb\_range\_lsl] = rangelsl;

itemsData\_0[tb\_range\_usl] = rangeusl;

itemsData[tb\_range\_lsl] = rangelsl;

itemsData[tb\_range\_usl] = rangeusl;

NSLog(@">.>>>range: %@,%@",rangelsl,rangeusl);

NSDictionary \*dic2 = [NSDictionary dictionaryWithObjectsAndKeys:rangelsl,krangelsl,rangeusl,krangeusl, nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetRangeLslUsl object:nil userInfo:dic2];

}

else

{

NSString \*rangelsl = itemsData[tb\_lower];

NSString \*rangeusl = itemsData[tb\_upper];

itemsData\_0[tb\_range\_lsl] = rangelsl;

itemsData\_0[tb\_range\_usl] = rangeusl;

itemsData[tb\_range\_lsl] = rangelsl;

itemsData[tb\_range\_usl] = rangeusl;

NSLog(@".>.>>>range: %@,%@",rangelsl,rangeusl);

NSDictionary \*dic2 = [NSDictionary dictionaryWithObjectsAndKeys:rangelsl,krangelsl,rangeusl,krangeusl, nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetRangeLslUsl object:nil userInfo:dic2];

}

[self sendDataToRedis:itemName\_0 withData:itemsData\_0];

[self sendDataToRedis:itemName withData:itemsData];

[self sendCpkZmqMsg:itemName];

[self sendCorrelationZmqMsg:itemName];

[self sendScatterZmqMsg:itemName];

}

-(void)getTwoColorTableDataAndSend:(int)xy withLeftRow:(NSInteger)row\_left withRightRow:(NSInteger)row\_right

{

if (selectColorBoxIndex ==0 && selectColorBoxIndex2 == 0)

{

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

return;

}

NSMutableArray \*delectArrIndex = [self getNeedDeletDataIndex];

NSArray \*selectItemColorTbItemLeft = [m\_configDictionary valueForKey:kSelectColorByTableRowsLeft];

NSArray \*selectItemColorTbItemRight = [m\_configDictionary valueForKey:kSelectColorByTableRowsRight];

NSArray \*itemsArr = \_dataReverse[selectColorBoxIndex]; //left

NSUInteger itemCountL = [itemsArr count];

NSUInteger selectCountL = [selectItemColorTbItemLeft count];

NSMutableArray \*itemDataIndexLeft = [NSMutableArray array];

//NSInteger row\_left = [self.colorByTableView selectedRow];

if (!row\_left)

{

row\_left = [self.colorByTableView selectedRow];

}

if (selectColorBoxIndex >0 && row\_left>= 0)

{

for (int i=0; i<selectCountL; i++) // color by table select item,显示item名字

{

NSMutableArray \*tmp = [NSMutableArray array];

for (int j =0; j<itemCountL; j++)

{

if (j<tb\_data\_start)

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

if ([itemsArr[j] isEqualTo:selectItemColorTbItemLeft[i]])

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

}

[itemDataIndexLeft addObject:tmp];

}

}

else

{

for (int i=0; i<selectCountL; i++) // color by table select item,显示item名字

{

NSMutableArray \*tmp = [NSMutableArray array];

for (int j =0; j<itemCountL; j++)

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

[itemDataIndexLeft addObject:tmp];

}

}

// NSLog(@"=====item index left: %@",itemDataIndexLeft);

NSArray \*itemsArr2 = \_dataReverse[selectColorBoxIndex2]; //right

NSUInteger itemCountR = [itemsArr2 count];

NSUInteger selectCountR = [selectItemColorTbItemRight count];

NSMutableArray \*itemDataIndexRight = [NSMutableArray array];

if (!row\_right)

{

row\_right = [self.colorByTableView2 selectedRow];

}

if (selectColorBoxIndex2>0 && row\_right>=0)

{

for (int i=0; i<selectCountR; i++) // color by table select item,显示item名字

{

NSMutableArray \*tmp = [NSMutableArray array];

for (int j =0; j<itemCountR; j++)

{

if (j<tb\_data\_start)

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

if ([itemsArr2[j] isEqualTo:selectItemColorTbItemRight[i]])

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

}

[itemDataIndexRight addObject:tmp];

}

}

else

{

for (int i=0; i<selectCountR; i++) // color by table select item,显示item名字

{

NSMutableArray \*tmp = [NSMutableArray array];

for (int j =0; j<itemCountR; j++)

{

[tmp addObject:[NSNumber numberWithInt:j]];

}

[itemDataIndexRight addObject:tmp];

}

}

NSMutableArray \*selectItemsIndex = [NSMutableArray array];

NSMutableArray \*selectItemsName = [NSMutableArray array];

for (int m = 0; m<[itemDataIndexLeft count]; m++)

{

for (int n = 0; n<[itemDataIndexRight count]; n++)

{

NSPredicate \* filter\_same = [NSPredicate predicateWithFormat:@"SELF IN %@",itemDataIndexLeft[m]]; //找到相同元素

NSArray \* filter\_selectItem = [itemDataIndexRight[n] filteredArrayUsingPredicate:filter\_same];

[selectItemsIndex addObject:filter\_selectItem];

[selectItemsName addObject:[NSString stringWithFormat:@"%@&%@",selectItemColorTbItemLeft[m],selectItemColorTbItemRight[n]]];

}

}

//NSLog(@"=====>select item index: %@",selectItemsIndex);

NSMutableArray \*itemsData\_0 = [NSMutableArray array];

NSInteger itemRow\_0 = [[m\_configDictionary valueForKey:kChooseItemIndex] integerValue];

NSMutableArray \*itemsData = [NSMutableArray array];

NSMutableArray \*snData = [NSMutableArray array];

NSMutableString \*colorItemName = [NSMutableString string];

for (int k = 0; k<[selectItemsIndex count]; k++)

{

//NSLog(@"====<<keep>>: %@",selectItemsIndex[k]);

//NSLog(@"====<<delete>>: %@",delectArrIndex);

int last\_selectItemsIndex = -1;

for (int h = 0; h<[selectItemsIndex[k] count]; h++)

{

if (last\_selectItemsIndex == [selectItemsIndex[k][h] intValue]) //去掉重复的

{

continue;

}

if (![delectArrIndex containsObject:selectItemsIndex[k][h]]) //在index delete 列没有的元素

{

int okrow = [selectItemsIndex[k][h] intValue];

if ([\_dataReverse[tbDataTableSelectItemRow+n\_Start\_Data\_Col] count]>okrow)

{

[itemsData addObject:\_dataReverse[tbDataTableSelectItemRow+n\_Start\_Data\_Col][okrow]];

[snData addObject:\_dataReverse[n\_SerialNumber][okrow]];

}

else

{

[itemsData addObject:@""];

[snData addObject:@""];

}

if ([\_dataReverse[itemRow\_0+n\_Start\_Data\_Col] count]>okrow)

{

[itemsData\_0 addObject:\_dataReverse[itemRow\_0+n\_Start\_Data\_Col][okrow]];

}

else

{

[itemsData\_0 addObject:@""];

}

//NSLog(@"===??>>>> %@",\_dataReverse[tbDataTableSelectItemRow+n\_Start\_Data\_Col][okrow]);

}

last\_selectItemsIndex = [selectItemsIndex[k][h] intValue];

}

[itemsData addObject:End\_Data];

[snData addObject:End\_Data];

[itemsData\_0 addObject:End\_Data];

[colorItemName appendString:[NSString stringWithFormat:@"%@##",selectItemsName[k]]];

}

//NSLog(@"===>>?>>>> %@",itemsData);

NSString \* itemName = [self combineItemName:colorItemName];

if (xy==-1)

{

// do nothing

}

else

{

itemName = [NSString stringWithFormat:@"%@$$%d",itemName,xy];

}

NSLog(@"======send item name to redis: %@ itemsData count:%zd",itemName,[itemsData count]);

itemsData[tb\_color\_by\_left]= [NSNumber numberWithInteger:selectColorBoxIndex]; //设置color By左边那个,给python生成图表用

itemsData[tb\_color\_by\_right]= [NSNumber numberWithInteger:selectColorBoxIndex2]; //设置color By左边那个,给python生成图表用

NSString \*typeZoom = [m\_configDictionary valueForKey:kzoom\_type];

itemsData[tb\_zoom\_type] = typeZoom;

NSString \*bins = [m\_configDictionary valueForKey:kBins];

itemsData[tb\_bins] = bins;

itemsData\_0[tb\_color\_by\_left]= [NSNumber numberWithInteger:selectColorBoxIndex]; //设置color By左边那个,给python生成图表用

itemsData\_0[tb\_color\_by\_right]= [NSNumber numberWithInteger:selectColorBoxIndex2]; //设置color By左边那个,给python生成图表用

itemsData\_0[tb\_zoom\_type] = typeZoom;

itemsData\_0[tb\_bins] = bins;

NSString \*itemName\_0 = [NSString stringWithFormat:@"%@\_XY",[m\_configDictionary valueForKey:kChooseItemName]];

itemsData[tb\_correlation\_xy] = itemName\_0;

itemsData\_0[tb\_correlation\_xy] = itemName\_0;

NSDictionary \*dic = [NSDictionary dictionaryWithObjectsAndKeys:snData,kSerial\_number,itemsData,kData\_Value, nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationShowData object:nil userInfo:dic];

if (b\_setRangeTxt)

{

b\_setRangeTxt = NO;

NSString \*rangelsl = [m\_configDictionary valueForKey:krangelsl];

NSString \*rangeusl = [m\_configDictionary valueForKey:krangeusl];

itemsData\_0[tb\_range\_lsl] = rangelsl;

itemsData\_0[tb\_range\_usl] = rangeusl;

itemsData[tb\_range\_lsl] = rangelsl;

itemsData[tb\_range\_usl] = rangeusl;

NSLog(@">>>>>range: %@,%@",rangelsl,rangeusl);

NSDictionary \*dic2 = [NSDictionary dictionaryWithObjectsAndKeys:rangelsl,krangelsl,rangeusl,krangeusl, nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetRangeLslUsl object:nil userInfo:dic2];

}

else

{

NSString \*rangelsl = itemsData[tb\_lower];

NSString \*rangeusl = itemsData[tb\_upper];

itemsData\_0[tb\_range\_lsl] = rangelsl;

itemsData\_0[tb\_range\_usl] = rangeusl;

itemsData[tb\_range\_lsl] = rangelsl;

itemsData[tb\_range\_usl] = rangeusl;

NSLog(@".>>>>>range: %@,%@",rangelsl,rangeusl);

NSDictionary \*dic2 = [NSDictionary dictionaryWithObjectsAndKeys:rangelsl,krangelsl,rangeusl,krangeusl, nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetRangeLslUsl object:nil userInfo:dic2];

}

[self sendDataToRedis:itemName\_0 withData:itemsData\_0];

[self sendDataToRedis:itemName withData:itemsData];

[self sendCpkZmqMsg:itemName];

[self sendCorrelationZmqMsg:itemName];

[self sendScatterZmqMsg:itemName];

}

-(void)notifySetImage:(NSString \*)path

{

NSDictionary \*dic = [NSDictionary dictionaryWithObject:path forKey:imagePath];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetCpkImage object:nil userInfo:dic];

}

-(void)setCpkImage:(NSString \*)path

{

NSImage \*imageCPK = [[NSImage alloc]initWithContentsOfFile:path];

dispatch\_async(dispatch\_get\_main\_queue(), ^{

[self.cpkImageView setImage:imageCPK];

});

}

-(void)setCorrelationImage:(NSString \*)path

{

NSImage \*imageCorrelation = [[NSImage alloc]initWithContentsOfFile:path];

dispatch\_async(dispatch\_get\_main\_queue(), ^{

[self.correlationImageView setImage:imageCorrelation];

});

}

-(void)setScatterImage:(NSString \*)path

{

NSImage \*imageCorrelation = [[NSImage alloc]initWithContentsOfFile:path];

dispatch\_async(dispatch\_get\_main\_queue(), ^{

[self.scatterImageMapView setImage:imageCorrelation];

});

}

- (BOOL)isAllNum:(NSString \*)string{

unichar c;

for (int i=0; i<string.length; i++) {

c=[string characterAtIndex:i];

if (!isdigit(c)) {

return NO;

}

}

return YES;

}

-(void)AlertBox:(NSString \*)msgTxt withInfo:(NSString \*)strmsg

{

NSAlert \* alert = [[NSAlert alloc] init];

alert.messageText = msgTxt;

alert.informativeText = strmsg;

[alert runModal];

}

-(int)AlertBoxWith2Button:(NSString \*)msgTxt withInfo:(NSString \*)strmsg

{

NSAlert \*alert = [[NSAlert alloc] init];

[alert setMessageText:msgTxt];

[alert setInformativeText:strmsg];

[alert addButtonWithTitle:@"OK"];

[alert addButtonWithTitle:@"Cancel"];

// [alert addButtonWithTitle:@"abort"];

[alert setAlertStyle:NSAlertStyleWarning];

NSUInteger action = [alert runModal];

if(action == NSAlertFirstButtonReturn) //1000

{

return 1000;

}

else if(action == NSAlertSecondButtonReturn )//1001

{

return 1001;

}

// else if(action == NSAlertThirdButtonReturn)//1002

// {

// NSLog(@"Abort");

// }

else

{

return -1;

}

}

-(NSString\*)switchRetest:(NSInteger)num

{

NSString \*value = @"";

switch (num) {

case 0:

value = vRetestFirst;

break;

case 1:

value = vRetestAll;

break;

case 2:

value = vRetestLast;

break;

default:

break;

}

return value;

}

-(NSString\*)switchRemoveFail:(NSInteger)num

{

NSString \*value = @"";

switch (num) {

case 0:

value = vRemoveFailYes;

break;

case 1:

value = vRemoveFailNo;

break;

default:

break;

}

return value;

}

-(NSString\*)switchZoomDataLimit:(NSInteger)num

{

NSString \*value = @"";

switch (num) {

case 0:

value = @"limit";

break;

case 1:

value = @"data";

break;

case 2:

value = @"range";

break;

default:

break;

}

return value;

}

-(int)indexOfColorByItem:(NSString\*)item

{

for (int i=0; i<[colorByName count]; i++)

{

if ([colorByName[i] isEqualToString:item])

{

return i;

}

}

return 0;

}

// Version,Station\_ID,Special\_Build\_Name,Special\_Build\_Descrip,Product,Channel\_ID

- (IBAction)selectColorByBoxAction:(id)sender {

NSString \*title = [(NSComboBox \*)sender stringValue];

NSLog(@"=>title: %@",title);

[self sendDataToRedis:@"select\_filter\_by\_1" withData:[NSMutableArray arrayWithObject:title]];

int n\_index = [self indexOfColorByItem:title];

if (n\_index>0)

{

[self.colorByBox2 removeAllItems];

[self.colorByBox2 addItemsWithObjectValues:colorByName];

[self.colorByBox2 removeItemAtIndex:n\_index];

}

[\_data removeAllObjects];

if ([title isEqualToString:Off])

{

[\_data removeAllObjects];

[m\_configDictionary setValue:@[Off] forKey:kSelectColorByTableRowsLeft];

selectColorBoxIndex = 0;

[self.colorByBox2 removeAllItems];

[self.colorByBox2 addItemsWithObjectValues:colorByName];

[self getTwoColorTableDataAndSend:-1];

}

else if ([title isEqualToString:Version])

{

if (n\_Version\_Col>=0)

{

selectColorBoxIndex = n\_Version\_Col;

NSMutableArray \*vers = [m\_configDictionary valueForKey:k\_dic\_Version];

if ([vers count]>0)

{

\_data = [NSMutableArray arrayWithArray:vers];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[Version\_Col]);

}

}

}

else

{

selectColorBoxIndex = 0;

[\_data removeAllObjects];

}

}

else if ([title isEqualToString:Station\_ID])

{

if (n\_StationID\_Col>=0)

{

selectColorBoxIndex = n\_StationID\_Col;

NSMutableArray \*IDs = [m\_configDictionary valueForKey:k\_dic\_Station\_ID];

if ([IDs count]>0) {

\_data = [NSMutableArray arrayWithArray:IDs];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[StationID\_Col]);

}

}

}

else

{

selectColorBoxIndex = 0;

[\_data removeAllObjects];

}

}

else if ([title isEqualToString:Special\_Build\_Name])

{

if (n\_SpecialBuildName\_Col>=0)

{

selectColorBoxIndex = n\_SpecialBuildName\_Col;

NSMutableArray \*BuildN = [m\_configDictionary valueForKey:k\_dic\_Special\_Build\_Name];

if ([BuildN count]>0) {

\_data = [NSMutableArray arrayWithArray:BuildN];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex = 0;

[\_data removeAllObjects];

}

}

else if ([title isEqualToString:Special\_Build\_Descrip])

{

if (n\_Special\_Build\_Descrip\_Col >=0)

{

selectColorBoxIndex = n\_Special\_Build\_Descrip\_Col;

NSMutableArray \*BuildN = [m\_configDictionary valueForKey:k\_dic\_Special\_Build\_Desc];

if ([BuildN count]>0) {

\_data = [NSMutableArray arrayWithArray:BuildN];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex = 0;

[\_data removeAllObjects];

}

}

else if ([title isEqualToString:Product])

{

if (n\_Product\_Col>=0)

{

selectColorBoxIndex = n\_Product\_Col;

NSMutableArray \*BuildN = [m\_configDictionary valueForKey:k\_dic\_Product];

if ([BuildN count]>0) {

\_data = [NSMutableArray arrayWithArray:BuildN];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex = 0;

[\_data removeAllObjects];

}

}

else if ([title isEqualToString:Channel\_ID])

{

int selRow = [[m\_configDictionary valueForKey:k\_dic\_Channel\_ID\_Index] intValue];

if (selRow>=0)

{

selectColorBoxIndex = selRow;

NSMutableArray \*channelId = [m\_configDictionary valueForKey:k\_dic\_Channel\_ID];

if ([channelId count]>0) {

NSLog(@"=====<<<--->>> %@ %zd",channelId,[channelId count]);

\_data = [NSMutableArray arrayWithArray:channelId];

if (tbDataTableSelectItemRow>=0)

{

if (selRow>0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[selRow]);

}

}

}

}

else

{

selectColorBoxIndex = 0;

[\_data removeAllObjects];

}

}

else if ([title isEqualToString:Diags\_Version])

{

if (n\_Diags\_Version\_Col>0)

{

selectColorBoxIndex = n\_Diags\_Version\_Col;

NSMutableArray \*diagsN = [m\_configDictionary valueForKey:k\_dic\_Diags\_Version];

if ([diagsN count]>0) {

\_data = [NSMutableArray arrayWithArray:diagsN];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex = 0;

[\_data removeAllObjects];

}

}

else if ([title isEqualToString:OS\_VERSION])

{

if (n\_OS\_VERSION\_Col>0)

{

selectColorBoxIndex = n\_OS\_VERSION\_Col;

NSMutableArray \*OSVer = [m\_configDictionary valueForKey:k\_dic\_OS\_Version];

if ([OSVer count]>0) {

\_data = [NSMutableArray arrayWithArray:OSVer];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex = 0;

[\_data removeAllObjects];

}

}

/\*else if ([title isEqualToString:Station\_Channel\_ID])

{

int selRow = [[m\_configDictionary valueForKey:k\_dic\_Channel\_ID\_Index] intValue];

selectColorBoxIndex = StationID\_Col\*10000+selRow; //取出来的时候除以10000，结果是station id，余就是channel id

NSMutableArray \*station\_channelId = [m\_configDictionary valueForKey:k\_dic\_Station\_Channel\_ID];

if ([station\_channelId count]>0) {

\_data = [NSMutableArray arrayWithArray:station\_channelId];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[tbsDataTableSelectItemRow]);

}

}

}\*/

// if ([self.color\_dicDatas.allKeys containsObject:title]) {

//

// self.color\_datas =[self.color\_dicDatas objectForKey:title];

// }else{

// self.color\_datas =nil;

// }

[self.colorByTableView reloadData];

NSDictionary \*dic = [NSDictionary dictionaryWithObject:[NSNumber numberWithInt:selectColorBoxIndex] forKey:select\_Color\_Box\_left];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetColorByLeft object:nil userInfo:dic];

}

- (IBAction)selectColorByBoxAction2:(id)sender

{

NSString \*title = [(NSComboBox \*)sender stringValue];

NSLog(@"=>title2: %@",title);

[self sendDataToRedis:@"select\_filter\_by\_2" withData:[NSMutableArray arrayWithObject:title]];

int n\_index = [self indexOfColorByItem:title];

if (n\_index>0)

{

[self.colorByBox removeAllItems];

[self.colorByBox addItemsWithObjectValues:colorByName];

[self.colorByBox removeItemAtIndex:n\_index];

}

[\_data2 removeAllObjects];

if ([title isEqualToString:Off])

{

[\_data2 removeAllObjects];

selectColorBoxIndex2 = 0;

[m\_configDictionary setValue:@[Off] forKey:kSelectColorByTableRowsRight];

[self.colorByBox removeAllItems];

[self.colorByBox addItemsWithObjectValues:colorByName];

[self getTwoColorTableDataAndSend:-1];

}

else if ([title isEqualToString:Version])

{

if (n\_Version\_Col>=0)

{

selectColorBoxIndex2 = n\_Version\_Col;

NSMutableArray \*vers = [m\_configDictionary valueForKey:k\_dic\_Version];

if ([vers count]>0) {

\_data2 = [NSMutableArray arrayWithArray:vers];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[Version\_Col]);

}

}

}

else

{

selectColorBoxIndex2 = 0;

[\_data2 removeAllObjects];

}

}

else if ([title isEqualToString:Station\_ID])

{

if (n\_StationID\_Col>=0)

{

selectColorBoxIndex2 = n\_StationID\_Col;

NSMutableArray \*IDs = [m\_configDictionary valueForKey:k\_dic\_Station\_ID];

if ([IDs count]>0) {

\_data2 = [NSMutableArray arrayWithArray:IDs];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[StationID\_Col]);

}

}

}

else

{

selectColorBoxIndex2 = 0;

[\_data2 removeAllObjects];

}

}

else if ([title isEqualToString:Special\_Build\_Name])

{

if (n\_SpecialBuildName\_Col>=0)

{

selectColorBoxIndex2 = n\_SpecialBuildName\_Col;

NSMutableArray \*BuildN = [m\_configDictionary valueForKey:k\_dic\_Special\_Build\_Name];

if ([BuildN count]>0) {

\_data2 = [NSMutableArray arrayWithArray:BuildN];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex2 = 0;

[\_data2 removeAllObjects];

}

}

else if ([title isEqualToString:Special\_Build\_Descrip])

{

if (n\_Special\_Build\_Descrip\_Col>=0)

{

selectColorBoxIndex2 = n\_Special\_Build\_Descrip\_Col;

NSMutableArray \*BuildN = [m\_configDictionary valueForKey:k\_dic\_Special\_Build\_Desc];

if ([BuildN count]>0) {

\_data2 = [NSMutableArray arrayWithArray:BuildN];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex2 = 0;

[\_data2 removeAllObjects];

}

}

else if ([title isEqualToString:Product])

{

if (n\_Product\_Col>=0)

{

selectColorBoxIndex2 = n\_Product\_Col;

NSMutableArray \*BuildN = [m\_configDictionary valueForKey:k\_dic\_Product];

if ([BuildN count]>0) {

\_data2 = [NSMutableArray arrayWithArray:BuildN];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex2 = 0;

[\_data2 removeAllObjects];

}

}

else if ([title isEqualToString:Channel\_ID])

{

int selRow = [[m\_configDictionary valueForKey:k\_dic\_Channel\_ID\_Index] intValue];

if (selRow>=0)

{

selectColorBoxIndex2 = selRow;

NSMutableArray \*channelId = [m\_configDictionary valueForKey:k\_dic\_Channel\_ID];

if ([channelId count]>0) {

NSLog(@"=====<<<--->>> %@ %zd",channelId,[channelId count]);

\_data2 = [NSMutableArray arrayWithArray:channelId];

if (tbDataTableSelectItemRow>=0)

{

if (selRow>0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[selRow]);

}

}

}

}

else

{

selectColorBoxIndex2 = 0;

[\_data2 removeAllObjects];

}

}

else if ([title isEqualToString:Diags\_Version])

{

if (n\_Diags\_Version\_Col>0)

{

selectColorBoxIndex2 = n\_Diags\_Version\_Col;

NSMutableArray \*diagsN = [m\_configDictionary valueForKey:k\_dic\_Diags\_Version];

if ([diagsN count]>0) {

\_data2 = [NSMutableArray arrayWithArray:diagsN];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex2 = 0;

[\_data2 removeAllObjects];

}

}

else if ([title isEqualToString:OS\_VERSION])

{

if (n\_OS\_VERSION\_Col>0)

{

selectColorBoxIndex2 = n\_OS\_VERSION\_Col;

NSMutableArray \*OSVer = [m\_configDictionary valueForKey:k\_dic\_OS\_Version];

if ([OSVer count]>0) {

\_data2 = [NSMutableArray arrayWithArray:OSVer];

if (tbDataTableSelectItemRow>=0)

{

//NSLog(@"---data table select row: %zd , %@",tbDataTableSelectItemRow,\_dataReverse[SpecialBuildName\_Col]);

}

}

}

else

{

selectColorBoxIndex2 = 0;

[\_data2 removeAllObjects];

}

}

[self.colorByTableView2 reloadData];

NSDictionary \*dic = [NSDictionary dictionaryWithObject:[NSNumber numberWithInt:selectColorBoxIndex2] forKey:select\_Color\_Box\_Right];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSetColorByRight object:nil userInfo:dic];

}

- (IBAction)clickRetestSegmentAction:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

NSInteger ret = self.retestSegment.selectedSegment;

NSLog(@"==%zd",ret);

[m\_configDictionary setValue:[self switchRetest:ret] forKey:kRetestSeg];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

}

- (IBAction)clickZoomType:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

NSInteger ret = self.zoomTypeSeg.selectedSegment;

if(ret >1)

{

[self setRangeCtlHidden:NO];

}

else

{

[self setRangeCtlHidden:YES];

}

[m\_configDictionary setValue:[self switchZoomDataLimit:ret] forKey:kzoom\_type];

[m\_configDictionary setValue:inputLSL forKey:krangelsl];

[m\_configDictionary setValue:inputUSL forKey:krangeusl];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

}

-(void)setRangeCtlHidden:(BOOL)status

{

[self.rangeLsl setHidden:status];

[self.rangeUsl setHidden:status];

[self.rangeTxtLsl setHidden:status];

[self.rangeTxtUsl setHidden:status];

}

- (IBAction)btnShowData:(id)sender {

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

if ([\_showDatatWin.window isVisible]==0)

{

if (!\_showDatatWin)

{

\_showDatatWin=[[showDataControl alloc]initWithWindowNibName:@"showDataControl"];

}

[\_showDatatWin.window orderFront:nil];

}

//just for test

// NSString \*picPath =[[NSBundle mainBundle]pathForResource:@"1.png" ofType:nil];

// [self setCpkImage:picPath];

// picPath =[[NSBundle mainBundle]pathForResource:@"2.png" ofType:nil];

// [self setCorrelationImage:picPath];

}

- (IBAction)clickRemoveFailSegmentAction:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

NSInteger ret = self.removeFailSegment.selectedSegment;

NSLog(@"== %zd",ret);

[m\_configDictionary setValue:[self switchRemoveFail:ret] forKey:kRemoveFailSeg];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

}

-(NSString \*)opensslSha1:(NSString \*)inputStr

{

unsigned char digest[SHA\_DIGEST\_LENGTH];

const char\* string = [inputStr UTF8String];

SHA\_CTX ctx;

SHA1\_Init(&ctx);

SHA1\_Update(&ctx, string, strlen(string));

SHA1\_Final(digest, &ctx);

char mdString[SHA\_DIGEST\_LENGTH\*2+1];

for (int i = 0; i < SHA\_DIGEST\_LENGTH; i++)

sprintf(&mdString[i\*2], "%02x", (unsigned int)digest[i]);

NSString \* hashCode = [NSString stringWithFormat:@"%s",mdString];

return hashCode;

}

-(NSString \*)opensslSha1FilePath:(NSString \*)path

{

FILE\* file = fopen([path UTF8String], "rb");

SHA\_CTX c;

unsigned char md[SHA\_DIGEST\_LENGTH];

int fd;

ssize\_t i;

unsigned char buf[BUFSIZE];

fd=fileno(file);

SHA1\_Init(&c);

for (;;)

{

i=read(fd,buf,BUFSIZE);

if (i <= 0) break;

SHA1\_Update(&c,buf,(unsigned long)i);

}

SHA1\_Final(&(md[0]),&c);

char mdString[SHA\_DIGEST\_LENGTH\*2+1];

for (i=0; i<SHA\_DIGEST\_LENGTH; i++)

sprintf(&mdString[i\*2], "%02x", (unsigned int)md[i]);

NSString \* hashCode = [NSString stringWithFormat:@"%s",mdString];

return hashCode;

}

-(BOOL)isPureInt:(NSString \*)string

{

NSScanner\* scan = [NSScanner scannerWithString:string];

int val;

return [scan scanInt:&val] && [scan isAtEnd];

}

-(BOOL)isPureFloat:(NSString \*)string

{

NSScanner\* scan = [NSScanner scannerWithString:string];

float val;

return [scan scanFloat:&val] && [scan isAtEnd];

}

-(NSArray \*)reverseArray:(NSArray \*)array

{

NSArray \*tmpArray = array[1];

NSMutableArray \*newArray = [NSMutableArray arrayWithCapacity:tmpArray.count];

for (NSInteger i=0; i<tmpArray.count; i++) {

NSMutableArray \*lineArray = [NSMutableArray arrayWithCapacity:array.count];

for (NSInteger j=0; j<array.count; j++) {

[lineArray addObject:@""];

}

[newArray addObject:lineArray];

}

for (NSInteger i=0; i<array.count; i++) {

for (NSInteger j=0; j<tmpArray.count; j++) {

if ([array[i] count]<=j)

{

newArray[j][i] = @"";

}

else

{

newArray[j][i] = array[i][j];

}

}

}

return newArray;

}

-(NSString \*)clickApply2NewCsv

{

NSString \*csv\_temp\_Item\_Path = @"/tmp/CPK\_Log/Temp/Excel\_data\_temp\_select\_apply.csv";

NSMutableArray \*csvTmpItem = [NSMutableArray array];

int i\_col=0;

for(NSMutableArray \*lineArray in \_dataReverse)

{

if (i\_col >= n\_Start\_Data\_Col)

{

if ([lineArray[tb\_apply] intValue]==1)

{

[csvTmpItem addObject:\_dataReverse[i\_col]];

}

}

else

{

[csvTmpItem addObject:\_dataReverse[i\_col]];

}

i\_col++;

}

NSMutableArray \*csvInsight = [NSMutableArray arrayWithArray:[self reverseArray:csvTmpItem]];

[csvInsight removeObjectsInRange:NSMakeRange(7,30)];

NSMutableString \*csvStr = [NSMutableString string];

int i=0;

for(NSMutableArray \*lineArray in csvInsight)

{

NSString \*arrayString;

if (i==0)

{

int len = (int)[lineArray count] -n\_Start\_Data\_Col;

[lineArray removeObjectsInRange:NSMakeRange(n\_Start\_Data\_Col, len)];

arrayString = [NSString stringWithFormat:@"%@,Parametric",[lineArray componentsJoinedByString:@","]];

}

else

{

arrayString = [lineArray componentsJoinedByString:@","];

}

[csvStr appendFormat:@"%@\n",arrayString];

i++;

}

NSError \*error = nil;

[csvStr writeToFile:csv\_temp\_Item\_Path atomically:YES encoding:NSUTF8StringEncoding error:&error];

if (error)

{

NSLog(@"write apply csv failed: %@",csv\_temp\_Item\_Path);

}

else

{

NSLog(@"write apply csv successful: %@",csv\_temp\_Item\_Path);

}

return csv\_temp\_Item\_Path;

}

- (IBAction)btnReportExcel:(id)sender

{

// [@"none" writeToFile:[NSString stringWithFormat:@"%@/CPK\_Log/temp/.excel.txt",desktopPath] atomically:YES encoding:NSUTF8StringEncoding error:nil];

// [@"none" writeToFile:[NSString stringWithFormat:@"%@/CPK\_Log/temp/.excel\_hash.txt",desktopPath] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:@"/tmp/CPK\_Log/temp/.excel.txt" atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:@"/tmp/CPK\_Log/temp/.excel\_hash.txt" atomically:YES encoding:NSUTF8StringEncoding error:nil];

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

if (n\_passdata<4)

{

//[self AlertBox:@"Warning" withInfo:@"PASS data less than 3, it can not calculate cpk value."];

//return;

}

NSString \*cmdKillPythonLaunch = @"ps -ef |grep -i python |grep -i report\_excel\_test.py |grep -v grep|awk '{print $2}' |xargs kill -9";

system([cmdKillPythonLaunch UTF8String]);

//NSString \*cmdKillExcel = @"ps -ef |grep -i Excel |grep -i Excel |grep -v grep|awk '{print $2}' |xargs kill -9";

//system([cmdKillExcel UTF8String]);

[startPython Lanuch\_excel\_report];

reportExcelClient = [[Client alloc] init];

[reportExcelClient CreateRPC:excel\_report\_zmq\_addr withSubscriber:nil];

[reportExcelClient setTimeout:20\*1000];

if(!\_reportSetWin)

{

\_reportSetWin=[[reportSettingCfg alloc]initWithWindowNibName:@"reportSettingCfg"];

}

NSModalResponse result = [NSApp runModalForWindow:\_reportSetWin.window];

if (result == NSModalResponseOK)

{

NSMutableString \*strCsv = [NSMutableString string];

NSMutableString \*new\_USL = [NSMutableString string];

NSMutableString \*new\_LSL = [NSMutableString string];

NSString \*reviewer\_n;

NSMutableString \*update\_Date = [NSMutableString string];

NSDateFormatter\* DateFomatter = [[NSDateFormatter alloc] init];

[DateFomatter setDateFormat:@"yyyy-MM-dd HH:mm:ss"];

NSTimeZone \*timezone = [[NSTimeZone alloc] initWithName:@"PST"];

[DateFomatter setTimeZone:timezone];

NSString\* systemTime = [DateFomatter stringFromDate:[NSDate date]];

NSString \*modify\_date = @"";

[strCsv appendString:@"index,item,low,upper,new\_lsl,new\_usl,apply,reviewer,date,comment\n"];

NSMutableString \* newLimitStr = [NSMutableString string];

int i=0;

int falgApplly = 0;

int flagApply2 = 0;

int flagnewLimit = 0;

for(NSMutableArray \*lineArray in \_dataReverse)

{

if (i>=n\_Start\_Data\_Col)

{

if ([lineArray[tb\_apply] intValue]==0 && ([lineArray[tb\_lsl] isNotEqualTo:@""] || [lineArray[tb\_usl] isNotEqualTo:@""]) &&[lineArray[tb\_reviewer] isEqualToString:@""])

{

falgApplly = 1;

}

if ([lineArray[tb\_apply] intValue]==1)

{

flagApply2 ++;

}

if ([lineArray[tb\_lsl] isNotEqualTo:@""] || [lineArray[tb\_usl] isNotEqualTo:@""])

{

flagnewLimit ++;

}

NSString \*new\_lsl\_str = lineArray[tb\_lsl];

NSString \*new\_usl\_str = lineArray[tb\_usl];

//NSLog(@"--->>new\_lsl\_str : %@ new\_usl\_str: %@",new\_lsl\_str,new\_usl\_str);

if ([self isPureFloat:new\_lsl\_str] && [self isPureInt:new\_lsl\_str] && [self isPureInt:new\_usl\_str] && [self isPureFloat:new\_usl\_str])

{

float newL = [new\_lsl\_str floatValue];

float newU = [new\_usl\_str floatValue];

if (newL>newU)

{

[newLimitStr appendFormat:@"Index:%d, Item:%@\nLSL: %@, USL: %@\n\n",i-n\_Start\_Data\_Col+1,lineArray[tb\_item],new\_lsl\_str,new\_usl\_str];

}

}

else if (([new\_lsl\_str length]>0 && [new\_usl\_str isEqualToString:@""]) || ([new\_usl\_str length]>0 && [new\_lsl\_str isEqualToString:@""]))

{

[newLimitStr appendFormat:@"Index:%d, Item:%@\nLSL: %@, USL: %@\n\n",i-n\_Start\_Data\_Col+1,lineArray[tb\_item],new\_lsl\_str,new\_usl\_str];

}

if ([lineArray[tb\_reviewer] isNotEqualTo:@""])

{

reviewer\_n = lineArray[tb\_reviewer];

}

//else if ([lineArray[tb\_date] isNotEqualTo:@""])

else if ([lineArray[tb\_lsl] isNotEqualTo:@""] || [lineArray[tb\_usl] isNotEqualTo:@""])

{

reviewer\_n = [m\_configDictionary valueForKey:kuserName];

}

else

{

reviewer\_n = @"";

}

if ([lineArray[tb\_date] isEqualToString:@""])

{

if ([reviewer\_n isNotEqualTo:@""])

{

modify\_date = systemTime;

}

else

{

modify\_date = @"";

}

}

else

{

modify\_date = lineArray[tb\_date];

}

NSString \*arrString = [NSString stringWithFormat:@"%@,%@,%@,%@,%@,%@,%@,%@,%@,%@\n",lineArray[tb\_index],lineArray[tb\_item],lineArray[tb\_lower],lineArray[tb\_upper],lineArray[tb\_lsl],lineArray[tb\_usl],lineArray[tb\_apply],reviewer\_n,modify\_date,lineArray[tb\_comment]];

[strCsv appendString:arrString];

[new\_USL appendString:[NSString stringWithFormat:@"%@,",lineArray[tb\_usl]]];

[new\_LSL appendString:[NSString stringWithFormat:@"%@,",lineArray[tb\_lsl]]];

[update\_Date appendString:[NSString stringWithFormat:@"%@,",modify\_date]];

}

i++;

}

NSString \*csv\_Path = @"/tmp/CPK\_Log/temp/item\_limit.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/item\_limit.csv",desktopPath];

[strCsv writeToFile:csv\_Path atomically:YES encoding:NSUTF8StringEncoding error:nil];

// =========hash code=====

NSString \*csv\_path1 = [m\_configDictionary valueForKey:Load\_Csv\_Path];

NSString \*csv\_path2\_local = [m\_configDictionary valueForKey:Load\_Local\_Csv\_Path];

NSString \*csv\_data\_Path = @"";

NSString \*itemName = @"";

if ([csv\_path1 length]>0)

{

csv\_data\_Path = csv\_path1;

itemName = @"generate\_excel";

}

else if ([csv\_path2\_local length]>0)

{

csv\_data\_Path = csv\_path2\_local;

itemName = @"local\_csv\_generate\_excel";

}

NSMutableString \*hashCsv = [NSMutableString string];

[hashCsv appendString:@"index,item,ssh code\n"];

NSString \*csv\_data\_Name = [csv\_data\_Path lastPathComponent];

[hashCsv appendString:[NSString stringWithFormat:@"1,Data CSV Name Hash:%@,%@\n",csv\_data\_Name,[self opensslSha1:csv\_data\_Name]]];

[hashCsv appendString:[NSString stringWithFormat:@"2,Data CSV File Hash :%@,%@\n",csv\_data\_Path,[self opensslSha1FilePath:csv\_data\_Path]]];

NSFileManager\* manager = [NSFileManager defaultManager];

unsigned long long csv\_data\_size= [[manager attributesOfItemAtPath:csv\_data\_Path error:nil] fileSize];

[hashCsv appendString:[NSString stringWithFormat:@"3,Data CSV Size(Exact %llu Bytes),%@\n",csv\_data\_size,[self opensslSha1:[NSString stringWithFormat:@"%llu",csv\_data\_size]]]];

NSRange deleteusl = {[new\_USL length] - 1, 1};

[new\_USL deleteCharactersInRange:deleteusl]; //删除最后一个逗号

[hashCsv appendString:[NSString stringWithFormat:@"4,\"New USL\" column hash,%@\n",[self opensslSha1:new\_USL]]];

NSRange deletelsl = {[new\_LSL length] - 1, 1};

[new\_LSL deleteCharactersInRange:deletelsl]; //删除最后一个逗号

[hashCsv appendString:[NSString stringWithFormat:@"5,\"New LSL\" column hash,%@\n",[self opensslSha1:new\_LSL]]];

NSRange deletedate = {[update\_Date length] - 1, 1};

[update\_Date deleteCharactersInRange:deletedate]; //删除最后一个逗号

[hashCsv appendString:[NSString stringWithFormat:@"6,\"Date\" column hash,%@\n",[self opensslSha1:update\_Date]]];

NSString \*hash\_Path = @"/tmp/CPK\_Log/temp/data\_hash.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/data\_hash.csv",desktopPath];

[hashCsv writeToFile:hash\_Path atomically:YES encoding:NSUTF8StringEncoding error:nil];

//=========================

if ([newLimitStr length]>0)

{

[self AlertBox:@"Below items limit reverse, or no input new LSL or new USL!!!" withInfo:[NSString stringWithFormat:@"%@",newLimitStr]];

system([cmdKillPythonLaunch UTF8String]);

return;

}

NSString \*exportAllItems = [m\_configDictionary valueForKey:kexportAllItems];

NSString \*exportPassItems = [m\_configDictionary valueForKey:kexportPassItems];

NSString \*onlyexportlimitupdated = [m\_configDictionary valueForKey:konlyLimitUpdated];

if ([onlyexportlimitupdated isEqualToString:@"1"])

{

if (flagApply2 != flagnewLimit || flagApply2 == 0)

{

[self AlertBox:@"Error:028" withInfo:@"You did not click apply button."];

system([cmdKillPythonLaunch UTF8String]);

return;

}

NSString \*tempPath = [self clickApply2NewCsv];

exportAllItems = @"1";

csv\_data\_Path = tempPath;

}

if (falgApplly == 1)

{

int ret = [self AlertBoxWith2Button:@"Warning!" withInfo:@"You have updated some items limits, but didn't click apply. You want to proceed with report generation?"];

if (ret == 1001) //cancel not load

{

system([cmdKillPythonLaunch UTF8String]);

return;

}

}

//[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSaveUIdata object:nil userInfo:nil];

NSString \*cpkLow = [m\_configDictionary valueForKey:kcpkLowThd];

NSString \*cpkHigh = [m\_configDictionary valueForKey:kcpkHighThd];

NSString \*populate = [m\_configDictionary valueForKey:kpopulateDistri];

NSString \*userName = [m\_configDictionary valueForKey:kuserName];

NSString \*projectName = [m\_configDictionary valueForKey:kprojectName];

NSString \*targetBuild = [m\_configDictionary valueForKey:ktargetBuild];

NSString \*cpk\_path = [NSString stringWithFormat:@"%@/CPK\_Log/",desktopPath];

NSString \*set\_bin = [m\_configDictionary valueForKey:kBins];

NSString \*push2git = [m\_configDictionary valueForKey:kpush2GitHub];

NSString \*gitAddr = [m\_configDictionary valueForKey:kgitWebAddr];

NSString \*gitComment = [m\_configDictionary valueForKey:kgitComment];

NSString \*p\_val\_status = [m\_configDictionary valueForKey:kp\_val\_status];

NSMutableArray \*msgArray = [NSMutableArray arrayWithObjects:exportAllItems,exportPassItems,cpkLow,cpkHigh,populate,userName,projectName,targetBuild,cpk\_path,set\_bin,csv\_data\_Path,push2git,gitAddr,gitComment,p\_val\_status,onlyexportlimitupdated,nil];

NSLog(@"====excel==name:%@ data:%@",itemName,msgArray);

[self sendDataToRedis:itemName withData:msgArray];

[self sendExcelZmqMsg:itemName];

[\_btn\_report\_excel setEnabled:NO];

[\_progressExcel setHidden:NO];

[\_progressExcel startAnimation:nil];

} else if (result == NSModalResponseCancel)

{

system([cmdKillPythonLaunch UTF8String]);

NSLog(@"====cancel==");

// NSLog(@"=======hash code: %@",[self opensslSha1:@"hello worldrrrrr"]);

}

}

- (IBAction)btnReport:(id)sender // keynote report

{

//[@"none" writeToFile:[NSString stringWithFormat:@"%@/CPK\_Log/temp/.keynote.txt",desktopPath] atomically:YES encoding:NSUTF8StringEncoding error:nil];

[@"none" writeToFile:@"/tmp/CPK\_Log/temp/.keynote.txt" atomically:YES encoding:NSUTF8StringEncoding error:nil];

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load!"];

return;

}

if (n\_passdata<4)

{

//[self AlertBox:@"Warning" withInfo:@"PASS data less than 3, it can not calculate cpk value."];

//return;

}

NSString \*udfLanguageCode = [[NSUserDefaults standardUserDefaults] objectForKey:@"AppleLanguages"][0];

if (![udfLanguageCode containsString:@"en"])

{

[self AlertBox:@"Error:020" withInfo:@"Your Mac OS Language is not English version, it can not generate Keynote!!!\r\nPlease set your Mac OS Language to English!"];

return;

}

NSString \*cmdKillPythonLaunch = @"ps -ef |grep -i python |grep -i report\_keynote\_test |grep -v grep|awk '{print $2}' |xargs kill -9";

NSString \*cmdKillKeynote = @"ps -ef |grep -i Keynote |grep -i Keynote |grep -v grep|awk '{print $2}' |xargs kill -9";

system([cmdKillPythonLaunch UTF8String]);

system([cmdKillKeynote UTF8String]);

[startPython Lanuch\_keynote\_report];

reportKeynoteClient = [[Client alloc] init]; // connect keynote

[reportKeynoteClient CreateRPC:keynote\_report\_zmq\_addr withSubscriber:nil];

[reportKeynoteClient setTimeout:20\*1000];

if (!\_keynoteSetWin)

{

\_keynoteSetWin = [[keynoteSetting alloc] initWithWindowNibName:@"keynoteSetting"];

}

NSString \* cancelButton = [m\_configDictionary valueForKey:Kkeynote\_skip\_setting\_Cancel];

if ([cancelButton isEqualTo:@"Cancel"])

{

[\_keynoteSetWin initAllCtl];

}

NSModalResponse result = [NSApp runModalForWindow:\_keynoteSetWin.window];

if (result == NSModalResponseOK)

{

int check\_AdvancedYes = [[m\_configDictionary valueForKey:KitemAdvancedYes] intValue];

int check\_AdvancedNo = [[m\_configDictionary valueForKey:KitemAdvancedNo] intValue];

int check\_1aYes = [[m\_configDictionary valueForKey:Kitem1aYes] intValue];

int check\_1aNo = [[m\_configDictionary valueForKey:Kitem1aNo] intValue];

int check\_1bYes = [[m\_configDictionary valueForKey:Kitem1bYes] intValue];

int check\_1bNo = [[m\_configDictionary valueForKey:Kitem1bNo] intValue];

NSString \*cpkLow = [m\_configDictionary valueForKey:kcpkKeynoteLowThd];

NSString \*projectName = [m\_configDictionary valueForKey:kkeynotePrjName];

NSString \*targetBuild = [m\_configDictionary valueForKey:kkeynoteBuild];

NSLog(@"--1 setting: %d,%d,%d,%d,%d,%d, %@",check\_AdvancedYes,check\_AdvancedNo,check\_1aYes,check\_1aNo,check\_1bYes,check\_1bNo,cpkLow);

NSString \*cpk\_path = [NSString stringWithFormat:@"%@/CPK\_Log/",desktopPath];

NSString \*set\_bin = [m\_configDictionary valueForKey:kBins];

//NSString \*csv\_path1 = [m\_configDictionary valueForKey:Load\_Csv\_Path];

if (check\_1aNo==0)

{

int check\_k = [[m\_configDictionary valueForKey:kchooseUIK] intValue];

if (check\_k == 0)

{

[self AlertBox:@"Warning!!!" withInfo:@"You need click \"K\" cloumn check box!!!"];

system([cmdKillPythonLaunch UTF8String]);

system([cmdKillKeynote UTF8String]);

return;

}

NSString \*itemName = @"generate\_keynote\_1a\_yes";

NSString \*cpkHigh = @"99999999.9";

NSString \*csv\_data\_Path = @"/tmp/CPK\_Log/Temp/keynote\_data\_temp.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp.csv",desktopPath];

NSMutableArray \*msgArray = [NSMutableArray arrayWithObjects:cpkLow,cpkHigh,cpk\_path,set\_bin,csv\_data\_Path,projectName,targetBuild,nil];

NSLog(@"====keynote 1a yes,name:%@;data:%@",itemName,msgArray);

[self sendDataToRedis:itemName withData:msgArray];

[self sendKeynoteZmqMsg:itemName];

[\_btn\_report\_keynote setEnabled:NO];

[\_progressKeynote setHidden:NO];

[\_progressKeynote startAnimation:nil];

}

else if (check\_1bYes==1)

{

int check\_biggerThanLowThd = [[m\_configDictionary valueForKey:khasBiggerThanLowThd] intValue];

if (check\_biggerThanLowThd == 0) // 全部数据 cpk 都小于lthd

{

NSString \*itemName = @"generate\_keynote\_1b\_yes";

//NSString \*cpkHigh = @"99999999.9";

//NSString \*csv\_data\_Path = [m\_configDictionary valueForKey:Load\_Csv\_Path];

NSString \*csv\_data\_Path = @"/tmp/CPK\_Log/Temp/keynote\_data\_temp.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp.csv",desktopPath];

NSString \*keynote\_data\_temp\_select\_k =@"/tmp/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv";// [NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv",desktopPath];

int check\_skipOneLimitYes = 1;//[[m\_configDictionary valueForKey:KskipOneLimitYes] intValue];

int check\_skipOneLimitNo = 0;//[[m\_configDictionary valueForKey:KskipOneLimitNo] intValue];

int check\_skipHTHLDYes = 1;//[[m\_configDictionary valueForKey:KskipHTHLDYes] intValue];

int check\_skipHTHLDNo = 0;//[[m\_configDictionary valueForKey:KskipHTHLDNo] intValue];

NSString \*check\_cpk\_thhld = @"";

NSString \*check\_one\_limit = @"";

if (check\_skipHTHLDYes ==1)

{

check\_cpk\_thhld = @"yes";

}

if (check\_skipHTHLDNo == 1)

{

check\_cpk\_thhld = @"no";

}

if (check\_skipOneLimitYes == 1)

{

check\_one\_limit = @"yes";

}

if (check\_skipOneLimitNo == 1)

{

check\_one\_limit = @"no";

}

NSString \*cpkHigh = @"10.0";//[m\_configDictionary valueForKey:kcpkKeynoteHighThd];

NSMutableArray \*msgArray = [NSMutableArray arrayWithObjects:cpkLow,cpkHigh,cpk\_path,set\_bin,csv\_data\_Path,keynote\_data\_temp\_select\_k,check\_cpk\_thhld,check\_one\_limit,projectName,targetBuild,nil];

NSLog(@"====keynote 1b yes directly,name:%@;data:%@",itemName,msgArray);

[self sendDataToRedis:itemName withData:msgArray];

[self sendKeynoteZmqMsg:itemName];

[\_btn\_report\_keynote setEnabled:NO];

[\_progressKeynote setHidden:NO];

[\_progressKeynote startAnimation:nil];

}

else

{

if (!\_keynoteskipSettingWin)

{

\_keynoteskipSettingWin = [[keynote\_skip\_setting alloc] initWithWindowNibName:@"keynote\_skip\_setting"];

}

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationReloadSkipSettingData object:nil userInfo:nil];

NSModalResponse resultSkip = [NSApp runModalForWindow:\_keynoteskipSettingWin.window];

if (resultSkip == NSModalResponseOK)

{

NSString \*itemName = @"generate\_keynote\_1b\_yes";

//NSString \*cpkHigh = @"99999999.9";

//NSString \*csv\_data\_Path = [m\_configDictionary valueForKey:Load\_Csv\_Path];

NSString \*csv\_data\_Path = @"/tmp/CPK\_Log/Temp/keynote\_data\_temp.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp.csv",desktopPath];

NSString \*keynote\_data\_temp\_select\_k = @"/tmp/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv",desktopPath];

int check\_skipOneLimitYes = [[m\_configDictionary valueForKey:KskipOneLimitYes] intValue];

int check\_skipOneLimitNo = [[m\_configDictionary valueForKey:KskipOneLimitNo] intValue];

int check\_skipHTHLDYes = [[m\_configDictionary valueForKey:KskipHTHLDYes] intValue];

int check\_skipHTHLDNo = [[m\_configDictionary valueForKey:KskipHTHLDNo] intValue];

NSString \*check\_cpk\_thhld = @"";

NSString \*check\_one\_limit = @"";

if (check\_skipHTHLDYes ==1)

{

check\_cpk\_thhld = @"yes";

}

if (check\_skipHTHLDNo == 1)

{

check\_cpk\_thhld = @"no";

}

if (check\_skipOneLimitYes == 1)

{

check\_one\_limit = @"yes";

}

if (check\_skipOneLimitNo == 1)

{

check\_one\_limit = @"no";

}

NSString \*cpkHigh = [m\_configDictionary valueForKey:kcpkKeynoteHighThd];

NSMutableArray \*msgArray = [NSMutableArray arrayWithObjects:cpkLow,cpkHigh,cpk\_path,set\_bin,csv\_data\_Path,keynote\_data\_temp\_select\_k,check\_cpk\_thhld,check\_one\_limit,projectName,targetBuild,nil];

NSLog(@"====keynote 1b yes,name:%@;data:%@",itemName,msgArray);

[self sendDataToRedis:itemName withData:msgArray];

[self sendKeynoteZmqMsg:itemName];

[\_btn\_report\_keynote setEnabled:NO];

[\_progressKeynote setHidden:NO];

[\_progressKeynote startAnimation:nil];

}

}

}

else if (check\_1bNo==1)

{

if (!\_keynoteskipSettingWin)

{

\_keynoteskipSettingWin = [[keynote\_skip\_setting alloc] initWithWindowNibName:@"keynote\_skip\_setting"];

}

//[\_keynoteskipSettingWin.window makeKeyAndOrderFront:nil];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationReloadSkipSettingData object:nil userInfo:nil];

NSModalResponse resultSkip = [NSApp runModalForWindow:\_keynoteskipSettingWin.window];

if (resultSkip == NSModalResponseOK)

{

NSLog(@"-----keynote skip item win ok");

NSString \*itemName = @"generate\_keynote\_1b\_no";

int check\_skipOneLimitYes = [[m\_configDictionary valueForKey:KskipOneLimitYes] intValue];

int check\_skipOneLimitNo = [[m\_configDictionary valueForKey:KskipOneLimitNo] intValue];

int check\_skipHTHLDYes = [[m\_configDictionary valueForKey:KskipHTHLDYes] intValue];

int check\_skipHTHLDNo = [[m\_configDictionary valueForKey:KskipHTHLDNo] intValue];

NSString \*cpkHigh = [m\_configDictionary valueForKey:kcpkKeynoteHighThd];

//NSLog(@"--2 setting: %d,%d,%d,%d, %@",check\_skipOneLimitYes,check\_skipOneLimitNo,check\_skipHTHLDYes,check\_skipHTHLDNo,cpkHigh);

NSString \*csv\_data\_Path = @"/tmp/CPK\_Log/Temp/keynote\_data\_temp.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp.csv",desktopPath];

NSString \*keynote\_data\_temp\_select\_k = @"/tmp/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv",desktopPath];

NSString \*check\_cpk\_thhld = @"";

NSString \*check\_one\_limit = @"";

if (check\_skipHTHLDYes ==1)

{

check\_cpk\_thhld = @"yes";

}

if (check\_skipHTHLDNo == 1)

{

check\_cpk\_thhld = @"no";

}

if (check\_skipOneLimitYes == 1)

{

check\_one\_limit = @"yes";

}

if (check\_skipOneLimitNo == 1)

{

check\_one\_limit = @"no";

}

NSMutableArray \*msgArray = [NSMutableArray arrayWithObjects:cpkLow,cpkHigh,cpk\_path,set\_bin,csv\_data\_Path,keynote\_data\_temp\_select\_k,check\_cpk\_thhld,check\_one\_limit,projectName,targetBuild,nil];

NSLog(@"====keynote 1b no,name:%@;data:%@",itemName,msgArray);

[self sendDataToRedis:itemName withData:msgArray];

[self sendKeynoteZmqMsg:itemName];

[\_btn\_report\_keynote setEnabled:NO];

[\_progressKeynote setHidden:NO];

[\_progressKeynote startAnimation:nil];

}

else if (resultSkip == NSModalResponseCancel)

{

NSLog(@"---..--keynote skip setting item will cancel");

}

}

}

else if (result == NSModalResponseCancel)

{

system([cmdKillPythonLaunch UTF8String]);

system([cmdKillKeynote UTF8String]);

NSLog(@"====keynote cancel==");

}

}

- (IBAction)btnShowYield:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

if ([\_yieldRetestWin.window isVisible]==0)

{

if (!\_yieldRetestWin)

{

\_yieldRetestWin=[[yieldRetestRate alloc]initWithWindowNibName:@"yieldRetestRate"];

}

[\_yieldRetestWin.window orderFront:nil];

}

}

- (IBAction)btnSelectY:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

n\_select\_y ++;

int y = n\_select\_y%2;

NSDictionary \*dic = [NSDictionary dictionaryWithObject:[NSNumber numberWithInt:y] forKey:btn\_select\_y];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSelectY object:nil userInfo:dic];

NSLog(@"====select y: %d",y);

}

- (IBAction)btnSelectX:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

n\_select\_x ++;

int x = n\_select\_x%2;

NSDictionary \*dic = [NSDictionary dictionaryWithObject:[NSNumber numberWithInt:x] forKey:btn\_select\_x];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationSelectX object:nil userInfo:dic];

NSLog(@"====select x: %d",x);

}

- (IBAction)setTxtBinsValue:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

//NSLog(@"==%@",[self.txtBins stringValue]);

[m\_configDictionary setValue:[self.txtBins stringValue] forKey:kBins];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

}

-(void)controlTextDidEndEditing:(NSNotification \*)obj

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

NSTextField \*textF =obj.object;

if ([textF.identifier isEqualToString:@"bins"])

{

NSString \*ret = [textF stringValue];

NSLog(@"===edit bins: %@",ret);

if ([self isAllNum:ret])

{

if ([ret intValue] <50 ||[ret intValue] >250)

{

[self AlertBox:@"error:021" withInfo:@"set bins range must between 50~250"];

NSString \*val = [m\_configDictionary valueForKey:kBins];

[self.txtBins setStringValue:val];

return;

}

[m\_configDictionary setValue:ret forKey:kBins];

}

else

{

[self AlertBox:@"Error:022" withInfo:@"Input Bins should be number!!!"];

NSString \*val = [m\_configDictionary valueForKey:kBins];

[self.txtBins setStringValue:val];

}

}

}

#pragma mark **TableView Datasource & delegate**

-(NSInteger)numberOfRowsInTableView:(NSTableView \*)tableView

{

if (tableView== self.colorByTableView) { //left color by

return [\_data count];

}

else if(tableView== self.colorByTableView2) //right color by

{

return [\_data2 count];

}

return -1;

}

-(NSView \*)tableView:(NSTableView \*)tableView viewForTableColumn:(NSTableColumn \*)tableColumn row:(NSInteger)row

{

if (tableView == self.colorByTableView) {

NSString \*columnIdentifier = [tableColumn identifier];

NSTableCellView \*view = [\_colorByTableView makeViewWithIdentifier:columnIdentifier owner:self];

if ([\_data count] > row)

{

[[view textField] setStringValue:\_data[row]];

}

else

{

[[view textField] setStringValue:@"--"];

}

return view;

}

else if (tableView == self.colorByTableView2)

{

NSString \*columnIdentifier = [tableColumn identifier];

NSTableCellView \*view = [\_colorByTableView2 makeViewWithIdentifier:columnIdentifier owner:self];

if ([\_data2 count] > row)

{

[[view textField] setStringValue:\_data2[row]];

}

else

{

[[view textField] setStringValue:@"--"];

}

return view;

}

return nil;

}

- (IBAction)sliderActionR:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

[self.sliderR setIntValue:50];

return;

}

int scaleFactor = self.sliderR.intValue;

if (scaleFactor==50)

{

[self.scrollViewRight magnifyToFitRect:self.scrollViewRight.bounds];

last\_SilderR = scaleFactor;

[self.sliderR setIntValue:50];

return;

}

if (last\_SilderR>scaleFactor)

{

static const CGFloat kZoomInFactor = 0.7071068;

[self.scrollViewRight setMagnification:self.scrollViewRight.magnification \* kZoomInFactor];

}

else if(last\_SilderR<scaleFactor)

{

static const CGFloat kZoomOutFactor = 1.414214;

[self.scrollViewRight setMagnification:self.scrollViewRight.magnification \* kZoomOutFactor];

}

last\_SilderR = scaleFactor;

}

- (IBAction)sliderActionL:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

[self.sliderL setIntValue:50];

return;

}

int scaleFactor = self.sliderL.intValue;

if (scaleFactor ==50)

{

[self.scrollViewLeft magnifyToFitRect:self.scrollViewLeft.bounds];

last\_SilderL = scaleFactor;

[self.sliderL setIntValue:50];

return;

}

if (last\_SilderL >scaleFactor)

{

static const CGFloat kZoomInFactor = 0.7071068;

[self.scrollViewLeft setMagnification:self.scrollViewLeft.magnification \* kZoomInFactor];

}

else if(last\_SilderL < scaleFactor)

{

static const CGFloat kZoomOutFactor = 1.414214;

[self.scrollViewLeft setMagnification:self.scrollViewLeft.magnification \* kZoomOutFactor];

}

last\_SilderL = scaleFactor;

}

- (IBAction)clickSaveButton:(NSButton \*)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

NSInteger btnTag = sender.tag;

if (btnTag == 0)

{

NSLog(@"--save cpk image");

NSString \*path = @"/tmp/CPK\_Log/temp/cpk.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/cpk.png",desktopPath];

NSString \*name = [NSString stringWithFormat:@"copy-image$$%@",path];

[self sendCopyImageZmqMsg:name];

}

else if (btnTag == 1)

{

NSLog(@"--save correlation image");

NSString \*path = @"/tmp/CPK\_Log/temp/correlation.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/correlation.png",desktopPath];

NSString \*name = [NSString stringWithFormat:@"copy-image$$%@",path];

[self sendCopyImageZmqMsg:name];

}

else if (btnTag == 2)

{

NSLog(@"--save scatter image");

NSString \*path = @"/tmp/CPK\_Log/temp/scatter.png";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/scatter.png",desktopPath];

NSString \*name = [NSString stringWithFormat:@"copy-image$$%@",path];

[self sendCopyImageZmqMsg:name];

}

}

-(NSString \*)sendCopyImageZmqMsg:(NSString \*)name

{

int ret = [copyImageClient SendCmd:name];

if (ret > 0)

{

NSString \* response = [copyImageClient RecvRquest:1024];

if (!response)

{

NSLog(@"zmq copy image for python error");

}

//NSLog(@"app->get response from copy image python: %@",response);

return response;

}

return nil;

}

- (IBAction)clickReportTags:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

NSString \*cmdKillPythonLaunch = @"ps -ef |grep -i python |grep -i report\_targs\_test |grep -v grep|awk '{print $2}' |xargs kill -9";

system([cmdKillPythonLaunch UTF8String]);

[startPython Lanuch\_report\_tags];

reportTagsClient = [[Client alloc] init]; // connect keynote

[reportTagsClient CreateRPC:report\_tags\_zmq\_addr withSubscriber:nil];

[reportTagsClient setTimeout:20\*1000];

if (!\_reportTagsWin)

{

\_reportTagsWin = [[reportTags alloc] initWithWindowNibName:@"reportTags"];

}

[\_reportTagsWin.window orderFront:nil];

}

- (IBAction)zoomInActionRight:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

static const CGFloat kZoomInFactor = 0.7071068;

[self.scrollViewRight setMagnification:self.scrollViewRight.magnification \* kZoomInFactor];

}

- (IBAction)zoomOutActionRight:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

static const CGFloat kZoomOutFactor = 1.414214;

[self.scrollViewRight setMagnification:self.scrollViewRight.magnification \* kZoomOutFactor];

}

- (IBAction)fitToScreenActionRight:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

[self.scrollViewRight magnifyToFitRect:self.scrollViewRight.bounds];

self.sliderR.intValue = 50;

last\_SilderR = 50;

}

- (IBAction)fittoScreenActionScatter:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

[self.scatterScrollView magnifyToFitRect:self.scatterScrollView.bounds];

self.sliderScatter.intValue = 50;

last\_SilderScatter = 50;

}

- (IBAction)clickSliderScatter:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

[self.sliderScatter setIntValue:50];

return;

}

int scaleFactor = self.sliderScatter.intValue;

if (scaleFactor ==50)

{

[self.scatterScrollView magnifyToFitRect:self.scatterScrollView.bounds];

last\_SilderScatter = 50;

[self.sliderScatter setIntValue:50];

return;

}

if (last\_SilderScatter>scaleFactor)

{

static const CGFloat kZoomInFactor = 0.7071068;

[self.scatterScrollView setMagnification:self.scatterScrollView.magnification \* kZoomInFactor];

}

else if(last\_SilderScatter<scaleFactor)

{

static const CGFloat kZoomOutFactor = 1.414214;

[self.scatterScrollView setMagnification:self.scatterScrollView.magnification \* kZoomOutFactor];

}

last\_SilderScatter = scaleFactor;

}

- (IBAction)zoomInActionLeft:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

static const CGFloat kZoomOutFactor = 0.7071068;

[self.scrollViewLeft setMagnification:self.scrollViewLeft.magnification \* kZoomOutFactor];

}

- (IBAction)zoomOutActionLeft:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

static const CGFloat kZoomOutFactor = 1.414214;

[self.scrollViewLeft setMagnification:self.scrollViewLeft.magnification \* kZoomOutFactor];

}

- (IBAction)fitTpScreenActionLeft:(id)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

[self.scrollViewLeft magnifyToFitRect:self.scrollViewLeft.bounds];

self.sliderL.intValue = 50;

last\_SilderL = 50;

}

- (IBAction)btTxtUsl:(NSTextField \*)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

NSString \*usl = [sender stringValue];

if ([usl length]==0)

{

return;

}

if ([self isPureInt:usl] || [self isPureFloat:usl] || [usl isEqualToString:@"NA"])

{

inputUSL = usl;

if (([self isPureInt:inputLSL] || [self isPureFloat:inputLSL]) && ([self isPureInt:inputUSL] || [self isPureFloat:inputUSL]))

{

float n\_lsl = [inputLSL floatValue];

float n\_usl = [inputUSL floatValue];

if (n\_lsl>n\_usl)

{

[self AlertBox:@"Error:023" withInfo:@"LSL is bigger than USL!!!"];

return;

}

[m\_configDictionary setValue:inputLSL forKey:krangelsl];

[m\_configDictionary setValue:inputUSL forKey:krangeusl];

[m\_configDictionary setValue:[NSNumber numberWithBool:YES] forKey:kInputRangeFlag];

b\_setRangeTxt = YES;

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

}

}

else

{

[self AlertBox:@"Error:016" withInfo:@"Please input a number."];

}

}

- (IBAction)btTxtLsl:(NSTextField \*)sender

{

if ([\_dataReverse count]<1)

{

[self AlertBox:@"Error:019" withInfo:@"no data to load."];

return;

}

NSString \*lsl = [sender stringValue];

if ([lsl length]==0)

{

return;

}

if ([self isPureInt:lsl] || [self isPureFloat:lsl] || [lsl isEqualToString:@"NA"])

{

inputLSL = lsl;

if (([self isPureInt:inputLSL] || [self isPureFloat:inputLSL]) && ([self isPureInt:inputUSL] || [self isPureFloat:inputUSL]))

{

float n\_lsl = [inputLSL floatValue];

float n\_usl = [inputUSL floatValue];

if (n\_lsl>n\_usl)

{

[self AlertBox:@"Error:023" withInfo:@"LSL is bigger than USL!!!"];

return;

}

[m\_configDictionary setValue:inputLSL forKey:krangelsl];

[m\_configDictionary setValue:inputUSL forKey:krangeusl];

[m\_configDictionary setValue:[NSNumber numberWithBool:YES] forKey:kInputRangeFlag];

b\_setRangeTxt = YES;

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

}

}

else

{

[self AlertBox:@"Error:016" withInfo:@"Please input a number."];

}

}

- (IBAction)btCorrelationScatterPlot:(id)sender

{

// if ([\_dataReverse count]<1)

// {

// [self AlertBox:@"Error!" withInfo:@"no data to load."];

// return;

// }

NSInteger ret = self.plotTypeSeg.selectedSegment;

if (ret ==0)

{

n\_flag\_scatterBtn=0;

if (n\_flag\_cpkBtn >0)

{

return;

}

n\_flag\_cpkBtn ++;

[self initSplitScatter];

}

else if (ret == 1)

{

n\_flag\_cpkBtn = 0;

if (n\_flag\_scatterBtn>0)

{

return;

}

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:0 ofDividerAtIndex:0];

[self.splitPlotView setPosition:0 ofDividerAtIndex:1];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

n\_flag\_scatterBtn ++;

}

}

-(void)initSplitScatter

{

CGFloat panelWinth = self.customerMainView.frame.size.width;

CGFloat offsetWidth = 20;

CGFloat x0 = panelWinth\*\_cpkPercentage;

CGFloat x1 = panelWinth\*\_correlationPercentage+ x0;

CGFloat x2 = \_scatterPercentage\*panelWinth + x1;

CGFloat x3 = \_settingPanelPercentage\*panelWinth + x2;

CGFloat x4 = \_filter1lPercentage\*panelWinth + x3;

CGFloat x5 = \_filter2lPercentage\*panelWinth+x4;

NSLog(@">>> %f %f %f %f %f %f",x0,x1,x2,x3,x4,x5);

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:0 ofDividerAtIndex:2];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:panelWinth\*\_correlationPercentage+ panelWinth\*\_cpkPercentage+offsetWidth ofDividerAtIndex:1];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:panelWinth\*\_cpkPercentage ofDividerAtIndex:0];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

/\*

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x0 ofDividerAtIndex:0];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x1 ofDividerAtIndex:1];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x2 ofDividerAtIndex:2];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x3 ofDividerAtIndex:3];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x4 ofDividerAtIndex:4];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x5 ofDividerAtIndex:5];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

\*/

}

-(void)moveSplitScatter

{

CGFloat panelWinth = self.customerMainView.frame.size.width;

CGFloat x0 = panelWinth\*\_cpkPercentage;

CGFloat x1 = panelWinth\*\_correlationPercentage+ x0;

CGFloat x2 = \_scatterPercentage\*panelWinth + x1;

CGFloat x3 = \_settingPanelPercentage\*panelWinth + x2;

CGFloat x4 = \_filter1lPercentage\*panelWinth + x3;

CGFloat x5 = \_filter2lPercentage\*panelWinth+x4;

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:0 ofDividerAtIndex:0];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:0 ofDividerAtIndex:1];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x0+x1+x2 ofDividerAtIndex:2];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x3 ofDividerAtIndex:3];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x4 ofDividerAtIndex:4];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

[NSAnimationContext runAnimationGroup:^(NSAnimationContext \* \_Nonnull context) {

context.allowsImplicitAnimation = YES;

context.duration = 0.1; // seconds

context.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];

[self.splitPlotView setPosition:x5 ofDividerAtIndex:5];

[self.splitPlotView layoutSubtreeIfNeeded];

}];

}

- (IBAction)checkActionPDF:(id)sender

{

[self sendStringToRedis:KSetPDF withData:[NSString stringWithFormat:@"%zd",[self.checkPDF state]]];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

}

- (IBAction)checkActionCDF:(id)sender

{

[self sendStringToRedis:KSetCDF withData:[NSString stringWithFormat:@"%zd",[self.checkCDF state]]];

[[NSNotificationCenter defaultCenter]postNotificationName:kNotificationClickOneItem object:nil userInfo:nil];

}

@end

//Keynote 代码

//

// keynoteSetting.m

// Bridge

//

// Created by RyanGao on 2020/7/20.

// Copyright © 2020 RyanGao. All rights reserved.

//

#import "keynoteSetting.h"

#import "defineHeader.h"

#import "../SCparseCSV.framework/Headers/parseCSV.h"

extern NSMutableDictionary \*m\_configDictionary;

extern int n\_Start\_Data\_Col;

extern int n\_Pass\_Fail\_Status;

extern int n\_Product\_Col;

extern int n\_SerialNumber;

extern int n\_SpecialBuildName\_Col;

extern int n\_Special\_Build\_Descrip\_Col;

extern int n\_StationID\_Col;

extern int n\_StartTime;

extern int n\_Version\_Col;

extern int n\_Diags\_Version\_Col;

extern int n\_OS\_VERSION\_Col;

extern NSMutableArray \*\_dataReverse;

@interface keynoteSetting ()

{

NSMutableArray \*selelct\_1b\_yes;

}

@end

@implementation keynoteSetting

- (void)windowDidLoad {

[super windowDidLoad];

selelct\_1b\_yes = [[NSMutableArray alloc] init];

[\_keynoteWin setLevel:kCGFloatingWindowLevel];

[self initAllCtl];

}

-(void)initAllCtl

{

[\_txtCPKLow setStringValue:@"1.5"];

//[\_txtCPKHigh setStringValue:@"9999999.0"];

[m\_configDictionary setValue:@"1.5" forKey:kcpkKeynoteLowThd];

self.buttonOk.enabled = NO;

[self.prjName setStringValue:@""];

[self.targetBuild setStringValue:@""];

[\_item\_Advanced\_Yes setState:0];

[m\_configDictionary setValue:[NSNumber numberWithInt:0] forKey:KitemAdvancedYes];

[\_item\_Advanced\_No setState:1];

[m\_configDictionary setValue:[NSNumber numberWithInt:1] forKey:KitemAdvancedNo];

[\_item\_1a\_Yes setState:1];

[m\_configDictionary setValue:[NSNumber numberWithInt:1] forKey:Kitem1aYes];

[\_item\_1a\_No setState:0];

[\_item\_1a\_No setEnabled:NO];

[m\_configDictionary setValue:[NSNumber numberWithInt:0] forKey:Kitem1aNo];

[\_item\_1b\_Yes setState:0];

[\_item\_1b\_Yes setEnabled:NO];

[m\_configDictionary setValue:[NSNumber numberWithInt:0] forKey:Kitem1bYes];

[\_item\_1b\_No setState:0];

[\_item\_1b\_No setEnabled:NO];

[m\_configDictionary setValue:[NSNumber numberWithInt:0] forKey:Kitem1bNo];

}

- (IBAction)btActionDefault:(id)sender

{

[self initAllCtl];

}

- (IBAction)btOk:(id)sender

{

NSString \*low = [self.txtCPKLow stringValue];

if ([self isPureInt:low] || [self isPureFloat:low])

{

[m\_configDictionary setValue:low forKey:kcpkKeynoteLowThd];

[m\_configDictionary setValue:self.prjName.stringValue forKey:kkeynotePrjName];

[m\_configDictionary setValue:self.targetBuild.stringValue forKey:kkeynoteBuild];

[m\_configDictionary setValue:[NSNumber numberWithInt:0] forKey:khasBiggerThanLowThd];

if ([\_item\_1a\_No state]==0)

{

[self uiSelectItemData2CSV];

}

else if([\_item\_1b\_Yes state]==1)

{

[self uiDispaly2CSV];

[self uiSelectItemName2Csv];

[self cpk\_lowerThan\_lsl\_item];

if ([[m\_configDictionary valueForKey:khasBiggerThanLowThd] intValue] == 0)

{

[self baseOnSelectgenerateDara2CSV];

}

}

else if ([\_item\_1b\_No state]==1)

{

[self uiDispaly2CSV];

[self uiSelectItemName2Csv];

}

[NSApp stopModalWithCode:NSModalResponseOK];

[[sender window] orderOut:self];

}

else

{

[self AlertBox:@"Error:016" withInfo:@"Please input a number"];

}

//NSString \*high = [self.txtCPKHigh stringValue];

/\*if (([self isPureInt:low] || [self isPureFloat:low]) && ([self isPureInt:high] || [self isPureFloat:high]))

{

float lowV = [low floatValue];

float highV = [high floatValue];

if (lowV>highV)

{

[self AlertBox:@"Error!!!" withInfo:@"CPK high threshold should be bigger than CPK low threshold!"];

return;

}

[m\_configDictionary setValue:low forKey:kcpkKeynoteLowThd];

[m\_configDictionary setValue:high forKey:kcpkKeynoteHighThd];

[NSApp stopModalWithCode:NSModalResponseOK];

[[sender window] orderOut:self];

}

else

{

[self AlertBox:@"Error!!!" withInfo:@"Input CPK threshold should be a number!"];

}

\*/

}

-(void)uiDispaly2CSV

{

int i=0;

NSMutableString \*strCsv = [NSMutableString string];

for(NSMutableArray \*lineArray in \_dataReverse)

{

if (i>=n\_Start\_Data\_Col)

{

NSString \*arrString = [NSString stringWithFormat:@"%@,%@,%@\n",lineArray[tb\_index],lineArray[tb\_item],lineArray[tb\_keynote]];

[strCsv appendString:arrString];

}

i++;

}

//NSString \*desktopPath = [NSSearchPathForDirectoriesInDomains(NSDesktopDirectory, NSUserDomainMask, YES)objectAtIndex:0];

NSString \*csv\_Path = @"/tmp/CPK\_Log/temp/items\_for\_skip\_setting.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/items\_for\_skip\_setting.csv",desktopPath];

[strCsv writeToFile:csv\_Path atomically:YES encoding:NSUTF8StringEncoding error:nil];

}

-(void)cpk\_lowerThan\_lsl\_item //把

{

NSString \*low = [self.txtCPKLow stringValue];

if ([self isPureInt:low] || [self isPureFloat:low])

{

//NSString \*desktopPath = [NSSearchPathForDirectoriesInDomains(NSDesktopDirectory, NSUserDomainMask, YES)objectAtIndex:0];

NSString \*param\_path = @"/tmp/CPK\_Log/temp/calculate\_param.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/temp/calculate\_param.csv",desktopPath];

CSVParser \*csvParamItem = [[CSVParser alloc]init];

NSMutableArray \*dataParam = [NSMutableArray array];

if ([csvParamItem openFile:param\_path])

{

dataParam = [csvParamItem parseFile];

}

if (dataParam.count<1)

{

NSLog(@"-no dataParam --");

return;

}

float cpk\_lthl = [low floatValue];

NSMutableString \*csvStr = [NSMutableString string];

int is\_bigger\_cpk\_lthl = 0;

[selelct\_1b\_yes removeAllObjects];

for (int i=0; i<dataParam.count; i++)

{

if ([dataParam[i] count]>5)

{

if ([self isPureInt:dataParam[i][6]] || [self isPureFloat:dataParam[i][6]])

{

float orig\_cpk = [dataParam[i][6] floatValue];

if (orig\_cpk < cpk\_lthl)

{

// NSLog(@"---orig\_cpk: %@ cpk\_lthl: %@",dataParam[i][6],low);

[csvStr appendString:[NSString stringWithFormat:@"%@\n",dataParam[i][0]]];

[selelct\_1b\_yes addObject:dataParam[i][0]];

}

else

{

is\_bigger\_cpk\_lthl ++;

}

}

}

}

[m\_configDictionary setValue:[NSNumber numberWithInt:is\_bigger\_cpk\_lthl] forKey:khasBiggerThanLowThd];

if (is\_bigger\_cpk\_lthl == 0)

{

[\_buttonOk setTitle:@"OK"];

}

NSString \*csv\_temp\_Item\_Path = @"/tmp/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv",desktopPath];

NSString \*str = [NSString stringWithContentsOfFile:csv\_temp\_Item\_Path encoding:NSUTF8StringEncoding error:nil];

NSString \*allStr = [NSString stringWithFormat:@"%@%@",str,csvStr];

NSError \*error = nil;

[allStr writeToFile:csv\_temp\_Item\_Path atomically:YES encoding:NSUTF8StringEncoding error:&error];

if (error)

{

NSLog(@"write csv all cpk<cpk\_lthl item and select K item fail: %@",csv\_temp\_Item\_Path);

}

else

{

NSLog(@"write csv all cpk<cpk\_lthl item and select K item successful: %@",csv\_temp\_Item\_Path);

}

}

}

-(void)uiSelectItemName2Csv

{

//NSString \*desktopPath = [NSSearchPathForDirectoriesInDomains(NSDesktopDirectory, NSUserDomainMask, YES)objectAtIndex:0];

//--按照k 勾选，生成新的csv

NSString \*csv\_temp\_Item\_Path = @"/tmp/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp\_select\_k.csv",desktopPath];

// NSFileManager \*manager = [NSFileManager defaultManager];

// [manager removeItemAtPath:csv\_temp\_Item\_Path error:nil];

NSMutableArray \*csvTmpItem = [NSMutableArray array];

int i\_col=0;

for(NSMutableArray \*lineArray in \_dataReverse)

{

if (i\_col >= n\_Start\_Data\_Col)

{

if ([lineArray[tb\_keynote] intValue]==1)

{

[csvTmpItem addObject:\_dataReverse[i\_col][1]]; // item name for K choose

}

}

i\_col++;

}

NSMutableString \*csvStr = [NSMutableString string];

for(NSString \*lineArray in csvTmpItem)

{

[csvStr appendFormat:@"%@\n",lineArray];

}

NSError \*error = nil;

[csvStr writeToFile:csv\_temp\_Item\_Path atomically:YES encoding:NSUTF8StringEncoding error:&error];

if (error)

{

NSLog(@"write csv item for keynote failed: %@",csv\_temp\_Item\_Path);

}

else

{

NSLog(@"write csv for keynote successful: %@",csv\_temp\_Item\_Path);

}

}

-(void)baseOnSelectgenerateDara2CSV

{

//NSString \*desktopPath = [NSSearchPathForDirectoriesInDomains(NSDesktopDirectory, NSUserDomainMask, YES)objectAtIndex:0];

NSString \*csv\_temp\_Path = @"/tmp/CPK\_Log/Temp/keynote\_data\_temp.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp.csv",desktopPath];

NSMutableArray \*csvTmpData = [NSMutableArray array];

int i\_col=0;

for(NSMutableArray \*lineArray in \_dataReverse)

{

if (i\_col <n\_Start\_Data\_Col)

{

[csvTmpData addObject:\_dataReverse[i\_col]];

}

else

{

if ([selelct\_1b\_yes containsObject:lineArray[tb\_item]])

{

////

if (([lineArray[tb\_keynote] intValue]==1) && ([lineArray[tb\_apply] intValue]==1))

{

NSMutableArray \*tmp\_arr =[NSMutableArray array];

for (int m = 0; m<[\_dataReverse[i\_col] count]; m++)

{

if (m==tb\_lower)

{

NSString \*new\_lsl = \_dataReverse[i\_col][tb\_lsl];

[tmp\_arr addObject:new\_lsl];

}

else if (m==tb\_upper)

{

NSString \*new\_usl = \_dataReverse[i\_col][tb\_usl];

[tmp\_arr addObject:new\_usl];

}

else

{

[tmp\_arr addObject:\_dataReverse[i\_col][m]];

}

}

[csvTmpData addObject:tmp\_arr];

}

else

{

[csvTmpData addObject:\_dataReverse[i\_col]];

}

}

}

i\_col++;

}

NSMutableArray \*csvInsight = [NSMutableArray arrayWithArray:[self reverseArray:csvTmpData]];

[csvInsight removeObjectsInRange:NSMakeRange(7,30)];

NSMutableString \*csvStr = [NSMutableString string];

int i=0;

for(NSMutableArray \*lineArray in csvInsight)

{

NSString \*arrayString;

if (i==0)

{

int len = (int)[lineArray count] -n\_Start\_Data\_Col;

[lineArray removeObjectsInRange:NSMakeRange(n\_Start\_Data\_Col, len)];

arrayString = [NSString stringWithFormat:@"%@,Parametric",[lineArray componentsJoinedByString:@","]];

}

else

{

arrayString = [lineArray componentsJoinedByString:@","];

}

[csvStr appendFormat:@"%@\n",arrayString];

i++;

}

NSError \*error = nil;

[csvStr writeToFile:csv\_temp\_Path atomically:YES encoding:NSUTF8StringEncoding error:&error];

if (error)

{

NSLog(@"write csv 1b yes directly for keynote failed: %@",csv\_temp\_Path);

}

else

{

NSLog(@"write csv 1b yes directly for keynote successful: %@",csv\_temp\_Path);

}

}

-(void)uiSelectItemData2CSV

{

//NSString \*desktopPath = [NSSearchPathForDirectoriesInDomains(NSDesktopDirectory, NSUserDomainMask, YES)objectAtIndex:0];

NSString \*csv\_temp\_Path = @"/tmp/CPK\_Log/temp/keynote\_data\_temp.csv";//[NSString stringWithFormat:@"%@/CPK\_Log/Temp/keynote\_data\_temp.csv",desktopPath];

NSLog(@"---temp keynote: %@",csv\_temp\_Path);

//--按照k 勾选，生成新的csv

NSMutableArray \*csvTmpData = [NSMutableArray array];

int i\_col=0;

int falgK = 0;

for(NSMutableArray \*lineArray in \_dataReverse)

{

if (i\_col <n\_Start\_Data\_Col)

{

[csvTmpData addObject:\_dataReverse[i\_col]];

}

else

{

if ([lineArray[tb\_keynote] intValue]==1)

{

falgK = 1;

if ([lineArray[tb\_apply] intValue]==1) //当k列与apply 都选中，用新limit

{

NSMutableArray \*tmp\_arr =[NSMutableArray array];

for (int m = 0; m<[\_dataReverse[i\_col] count]; m++)

{

if (m==tb\_lower)

{

NSString \*new\_lsl = \_dataReverse[i\_col][tb\_lsl];

[tmp\_arr addObject:new\_lsl];

}

else if (m==tb\_upper)

{

NSString \*new\_usl = \_dataReverse[i\_col][tb\_usl];

[tmp\_arr addObject:new\_usl];

}

else

{

[tmp\_arr addObject:\_dataReverse[i\_col][m]];

}

}

[csvTmpData addObject:tmp\_arr];

}

else

{

[csvTmpData addObject:\_dataReverse[i\_col]];

}

}

}

i\_col++;

}

[m\_configDictionary setValue:[NSNumber numberWithInt:falgK] forKey:kchooseUIK];

if (falgK == 1)

{

NSMutableArray \*csvInsight = [NSMutableArray arrayWithArray:[self reverseArray:csvTmpData]];

[csvInsight removeObjectsInRange:NSMakeRange(7,30)];

NSMutableString \*csvStr = [NSMutableString string];

int i=0;

for(NSMutableArray \*lineArray in csvInsight)

{

NSString \*arrayString;

if (i==0)

{

int len = (int)[lineArray count] -n\_Start\_Data\_Col;

[lineArray removeObjectsInRange:NSMakeRange(n\_Start\_Data\_Col, len)];

arrayString = [NSString stringWithFormat:@"%@,Parametric",[lineArray componentsJoinedByString:@","]];

}

else

{

arrayString = [lineArray componentsJoinedByString:@","];

}

[csvStr appendFormat:@"%@\n",arrayString];

i++;

}

NSError \*error = nil;

[csvStr writeToFile:csv\_temp\_Path atomically:YES encoding:NSUTF8StringEncoding error:&error];

if (error)

{

NSLog(@"write csv for keynote failed: %@",csv\_temp\_Path);

}

else

{

NSLog(@"write csv for keynote successful: %@",csv\_temp\_Path);

}

}

}

-(NSArray \*)reverseArray:(NSArray \*)array

{

NSArray \*tmpArray = array[1];

NSMutableArray \*newArray = [NSMutableArray arrayWithCapacity:tmpArray.count];

for (NSInteger i=0; i<tmpArray.count; i++) {

NSMutableArray \*lineArray = [NSMutableArray arrayWithCapacity:array.count];

for (NSInteger j=0; j<array.count; j++) {

[lineArray addObject:@""];

}

[newArray addObject:lineArray];

}

for (NSInteger i=0; i<array.count; i++) {

for (NSInteger j=0; j<tmpArray.count; j++) {

if ([array[i] count]<=j)

{

newArray[j][i] = @"";

}

else

{

newArray[j][i] = array[i][j];

}

}

}

return newArray;

}

- (IBAction)btCancel:(id)sender

{

[self initAllCtl];

[NSApp stopModalWithCode:NSModalResponseCancel];

[[sender window] orderOut:self];

}

- (IBAction)checkBox:(id)sender

{

int check = (int)[sender state];

switch ([sender tag])

{

case 10:

{

if (check)

{

[self AlertBox:@"note" withInfo:@"This Advanced Bimodality Breakdown Analysis is under development!!!"];

[\_item\_Advanced\_No setState:1];

[\_item\_Advanced\_Yes setState:0];

return;

}

else

{

[\_item\_Advanced\_No setState:check];

[\_item\_Advanced\_Yes setState:!check];

}

}

break;

case 11:

{

if (check)

{

[\_item\_Advanced\_Yes setState:!check];

[\_item\_Advanced\_No setState:check];

}

else

{

[\_item\_Advanced\_Yes setState:check];

[\_item\_Advanced\_No setState:!check];

}

}

break;

case 12:

{

if (check)

{

[\_item\_1a\_Yes setState:check];

//[\_item\_1a\_No setState:!check];

[\_item\_1a\_No setState:0];

[\_item\_1a\_No setEnabled:NO];

[\_item\_1b\_Yes setState:0];

[\_item\_1b\_No setState:0];

[\_item\_1b\_Yes setEnabled:NO];

[\_item\_1b\_No setEnabled:NO];

}

else

{

[\_item\_1a\_Yes setState:0];

[\_item\_1a\_No setState:1];

[\_item\_1a\_No setEnabled:YES];

[\_item\_1b\_Yes setEnabled:YES];

[\_item\_1b\_No setEnabled:YES];

[\_item\_1b\_Yes setState:1];

[\_item\_1b\_No setState:0];

}

}

break;

case 13:

{

if (check)

{

[\_item\_1a\_No setState:check];

[\_item\_1a\_Yes setState:!check];

[\_item\_1b\_Yes setState:1];

[\_item\_1b\_No setState:0];

}

else

{

[\_item\_1a\_No setState:!check];

[\_item\_1a\_Yes setState:check];

}

}

break;

case 14:

{

if (check)

{

[\_item\_1b\_Yes setState:check];

[\_item\_1b\_No setState:!check];

[\_item\_1a\_No setState:1];

[\_item\_1a\_Yes setState:0];

}

else

{

[\_item\_1b\_Yes setState:!check];

[\_item\_1b\_No setState:check];

}

}

break;

case 15:

{

if (check)

{

[\_item\_1b\_No setState:check];

[\_item\_1b\_Yes setState:!check];

[\_item\_1a\_No setState:1];

[\_item\_1a\_Yes setState:0];

}

else

{

[\_item\_1b\_No setState:!check];

[\_item\_1b\_Yes setState:check];

}

}

break;

default:

break;

}

if (\_item\_1b\_No.state == 1|| \_item\_1b\_Yes.state == 1)

{

[\_buttonOk setTitle:@"NEXT"];

}

else

{

[\_buttonOk setTitle:@"OK"];

}

[m\_configDictionary setValue:[NSNumber numberWithInteger:[\_item\_Advanced\_Yes state]] forKey:KitemAdvancedYes];

[m\_configDictionary setValue:[NSNumber numberWithInteger:[\_item\_Advanced\_No state]] forKey:KitemAdvancedNo];

[m\_configDictionary setValue:[NSNumber numberWithInteger:[\_item\_1a\_Yes state]] forKey:Kitem1aYes];

[m\_configDictionary setValue:[NSNumber numberWithInteger:[\_item\_1a\_No state]] forKey:Kitem1aNo];

[m\_configDictionary setValue:[NSNumber numberWithInteger:[\_item\_1b\_Yes state]] forKey:Kitem1bYes];

[m\_configDictionary setValue:[NSNumber numberWithInteger:[\_item\_1b\_No state]] forKey:Kitem1bNo];

}

-(void)controlTextDidChange:(NSNotification \*)obj

{

if (self.prjName.stringValue.length &&self.targetBuild.stringValue.length)

{

self.buttonOk.enabled = YES;

}

else

{

self.buttonOk.enabled = NO;

}

}

-(BOOL)isPureInt:(NSString \*)string

{

NSScanner\* scan = [NSScanner scannerWithString:string];

int val;

return [scan scanInt:&val] && [scan isAtEnd];

}

-(BOOL)isPureFloat:(NSString \*)string

{

NSScanner\* scan = [NSScanner scannerWithString:string];

float val;

return [scan scanFloat:&val] && [scan isAtEnd];

}

-(void)AlertBox:(NSString \*)msgTxt withInfo:(NSString \*)strmsg

{

NSAlert \* alert = [[NSAlert alloc] init];

alert.messageText = msgTxt;

alert.informativeText = strmsg;

[alert runModal];

}

@end

// Excel代码

//

// reportSettingCfg.m

// BDR\_Tool

//

// Created by RyanGao on 2020/7/5.

// Copyright © 2020 RyanGao. All rights reserved.

//

#import "reportSettingCfg.h"

#import "defineHeader.h"

extern NSMutableDictionary \*m\_configDictionary;

@interface reportSettingCfg ()

@end

@implementation reportSettingCfg

-(void)initAllCtl

{

[\_buttonOK setEnabled:NO];

[\_lowTH setStringValue:@"1.5"];

[\_highTH setStringValue:@"9999999.0"];

[m\_configDictionary setValue:@"1" forKey:kexportAllItems];

[\_exportAllItems setState:NSControlStateValueOn];

[m\_configDictionary setValue:@"0" forKey:kexportPassItems];

[\_exportAllItemsOutOf setState:NSControlStateValueOff];

[m\_configDictionary setValue:@"0" forKey:konlyLimitUpdated];

[\_onlyLimitUpdated setState:NSControlStateValueOff];

[m\_configDictionary setValue:@"1.5" forKey:kcpkLowThd];

[m\_configDictionary setValue:@"9999999.0" forKey:kcpkHighThd];

[m\_configDictionary setValue:@"0" forKey:kpopulateDistri];

[self.populate setState:NSControlStateValueOff];

[m\_configDictionary setValue:@"0" forKey:kp\_val\_status];

[self.p\_val\_Check setState:NSControlStateValueOff];

[m\_configDictionary setValue:@"" forKey:kuserName];

[self.userName setStringValue:@""];

[m\_configDictionary setValue:@"" forKey:kprojectName];

[self.projectName setStringValue:@""];

[m\_configDictionary setValue:@"" forKey:ktargetBuild];

[self.TargetBuild setStringValue:@""];

[\_push2Git setState:0];

[\_gitAddress setEnabled:NO];

[\_gitComment setEnabled:NO];

}

- (void)windowDidLoad {

[super windowDidLoad];

[\_settingCfgWin setLevel:kCGFloatingWindowLevel];

[self initAllCtl];

// Implement this method to handle any initialization after your window controller's window has been loaded from its nib file.

}

- (IBAction)btnOK:(id)sender

{

NSString \*low = [self.lowTH stringValue];

NSString \*high = [self.highTH stringValue];

if (([self isPureInt:low] || [self isPureFloat:low]) && ([self isPureInt:high] || [self isPureFloat:high]))

{

float lowV = [low floatValue];

float highV = [high floatValue];

if (lowV>highV)

{

[self AlertBox:@"Error:024" withInfo:@"CPK high threshold should be bigger than CPK low threshold!"];

return;

}

[m\_configDictionary setValue:low forKey:kcpkLowThd];

[m\_configDictionary setValue:high forKey:kcpkHighThd];

[m\_configDictionary setValue:self.userName.stringValue forKey:kuserName];

[m\_configDictionary setValue:self.projectName.stringValue forKey:kprojectName];

[m\_configDictionary setValue:self.TargetBuild.stringValue forKey:ktargetBuild];

int push2git\_checkBox = (int)[self.push2Git state];

[m\_configDictionary setValue:[NSNumber numberWithInt:push2git\_checkBox] forKey:kpush2GitHub];

[m\_configDictionary setValue:self.gitAddress.stringValue forKey:kgitWebAddr];

[m\_configDictionary setValue:self.gitComment.stringValue forKey:kgitComment];

[NSApp stopModalWithCode:NSModalResponseOK];

[[sender window] orderOut:self];

}

else

{

[self AlertBox:@"Error:016" withInfo:@"Please input a number."];

}

}

-(void)AlertBox:(NSString \*)msgTxt withInfo:(NSString \*)strmsg

{

NSAlert \* alert = [[NSAlert alloc] init];

alert.messageText = msgTxt;

alert.informativeText = strmsg;

[alert runModal];

}

- (IBAction)btnCancel:(id)sender

{

[self initAllCtl];

[NSApp stopModalWithCode:NSModalResponseCancel];

[[sender window] orderOut:self];

}

- (IBAction)click\_p\_val:(id)sender

{

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_p\_val\_Check state]] forKey:kp\_val\_status];

}

- (IBAction)clickPopulate:(id)sender

{

[self AlertBox:@"Warning" withInfo:@"It will take long time and just for debug use when select this."];

[self.populate setState:0];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_populate state]] forKey:kpopulateDistri];

}

- (IBAction)clickAllItemsOutOf:(id)sender

{

[\_exportAllItems setState:NSControlStateValueOff];

[\_exportAllItemsOutOf setState:NSControlStateValueOn];

[\_onlyLimitUpdated setState:NSControlStateValueOff];

//NSLog(@"======1=>>> %zd",[\_exportAllItems state]);

// NSLog(@"======1=>>> %zd",[\_exportAllItemsOutOf state]);

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_exportAllItems state]] forKey:kexportAllItems];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_exportAllItemsOutOf state]] forKey:kexportPassItems];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_onlyLimitUpdated state]] forKey:konlyLimitUpdated];

}

- (IBAction)clickAllItems:(id)sender

{

[\_exportAllItems setState:NSControlStateValueOn];

[\_exportAllItemsOutOf setState:NSControlStateValueOff];

[\_onlyLimitUpdated setState:NSControlStateValueOff];

// NSLog(@"======2=>>> %zd",[\_exportAllItems state]);

//NSLog(@"======2=>>> %zd",[\_exportAllItemsOutOf state]);

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_exportAllItems state]] forKey:kexportAllItems];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_exportAllItemsOutOf state]] forKey:kexportPassItems];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_onlyLimitUpdated state]] forKey:konlyLimitUpdated];

}

-(void)controlTextDidChange:(NSNotification \*)obj

{

//NSLog(@"---->>>controlTextDidChange");

if (self.userName.stringValue.length &&self.projectName.stringValue.length&&self.TargetBuild.stringValue.length)

{

self.buttonOK.enabled = YES;

}

else

{

self.buttonOK.enabled = NO;

}

}

-(BOOL)isPureInt:(NSString \*)string

{

NSScanner\* scan = [NSScanner scannerWithString:string];

int val;

return [scan scanInt:&val] && [scan isAtEnd];

}

-(BOOL)isPureFloat:(NSString \*)string

{

NSScanner\* scan = [NSScanner scannerWithString:string];

float val;

return [scan scanFloat:&val] && [scan isAtEnd];

}

- (IBAction)clickPush2Git:(id)sender

{

[self AlertBox:@"Warning" withInfo:@"Git function is under development!!! "];

[self.push2Git setState:0];

// int check = (int)[sender state];

//[\_gitAddress setEnabled:check];

//[\_gitComment setEnabled:check];

}

- (IBAction)btActionDefault:(id)sender

{

[self initAllCtl];

}

- (IBAction)ClickOnlyLimitUpdate:(id)sender

{

[\_exportAllItems setState:NSControlStateValueOff];

[\_exportAllItemsOutOf setState:NSControlStateValueOff];

[\_onlyLimitUpdated setState:NSControlStateValueOn];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_exportAllItems state]] forKey:kexportAllItems];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_exportAllItemsOutOf state]] forKey:kexportPassItems];

[m\_configDictionary setValue:[NSString stringWithFormat:@"%zd",[\_onlyLimitUpdated state]] forKey:konlyLimitUpdated];

}

@end

Python部分代码，实现计算和图像生成

#! /usr/bin/env python3

# --\*-- coding: utf-8 ---\*---

errorMsgs = ''

try:

import sys,os,time,math,re

except Exception as e:

print('e---->',e)

errorMsgs = str(e)+'\r\n'

try:

import datetime

except Exception as e:

print('e---->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

try:

import pytz

except Exception as e:

print('e---->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

try:

from pytz import timezone

except Exception as e:

print('e---->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

try:

import time

except Exception as e:

print('e---->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

try:

import threading

except Exception as e:

print('e---->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

BASE\_DIR=os.path.dirname(os.path.abspath(\_\_file\_\_))

sys.path.insert(0,BASE\_DIR+'/site-packages/')

try:

import csv

except Exception as e:

print('e---->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ---->matplotlib')

try:

import matplotlib

matplotlib.use("Agg")

import matplotlib.pyplot as plt

except Exception as e:

print('e---->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ----> matplotlib.colors')

try:

import matplotlib.colors as colors

except Exception as e:

print('e---->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ----> FontProperties')

try:

from matplotlib.font\_manager import FontProperties

except Exception as e:

print('e---->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ----> numpy')

try:

import numpy as np

except Exception as e:

print('e--->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ----> pandas')

try:

import pandas as pd

except Exception as e:

print('e--->',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ----> openpyxl')

try:

import openpyxl

except Exception as e:

print('import openpyxl error:',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ----> xlsxwriter')

try:

import xlsxwriter

except Exception as e:

print('import xlsxwriter error:',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ----> diptest')

try:

import diptest

except Exception as e:

print('import diptest error:',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ----> zmg')

try:

import zmq

except Exception as e:

print('import zmq error:',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

# print('python import ----> redis')

try:

import redis

except Exception as e:

print('import redis error:',e)

errorMsgs = errorMsgs + str(e) +'\r\n'

print('check for error Msgs module:',errorMsgs)

userdocuments = os.path.join(os.path.expanduser("~"), 'Documents')

with open(userdocuments + '/.errormodule.txt', 'w') as file\_obj:

file\_obj.write(errorMsgs)

print(sys.getdefaultencoding())

redisClient = redis.Redis(host='localhost', port=6379, db=0)

context = zmq.Context()

socket = context.socket(zmq.REP)

socket.setsockopt(zmq.LINGER,0)

socket.bind("tcp://127.0.0.1:3100")

filelogname = '/tmp/CPK\_Log/temp/.logcpk.txt'

def get\_redis\_data(zmqMsg):

tb = redisClient.get(zmqMsg)

tb\_data=[]

if tb:

tb=tb.decode('utf-8')

tb=tb.split("\n")

tb=(tb[1:-1]) #去掉数据库首尾元素

for i in tb:

k=re.sub('\"','',i) #去掉数据库引号

h=re.sub(',','',k) #去掉数据库逗号

m=h.strip() #去掉数据库首尾空白

if is\_number(m):

tb\_data.append(eval(m)) #去掉数字的引号

else:

tb\_data.append(m)

else:

tb\_data.append('')

return tb\_data

def is\_number(s):

try:

float(s)

return True

except ValueError:

pass

try:

import unicodedata

unicodedata.numeric(s)

return True

except (TypeError, ValueError):

pass

return False

def normfun(x, mu, sigma):

pdf = np.exp(-((x-mu)\*\*2)/(2\*sigma\*\*2))/(sigma\*np.sqrt(2\*np.pi))

return pdf

def get\_coefficients(value\_l):

'''

param value\_l: need float list

return: bc,p\_val,a\_Q,a\_irr,three\_σ\_x100\_divide\_mean

1σ＝690000／1000000 #fault rate

2σ＝308000／1000000

3σ＝66800／1000000

4σ＝6210／1000000

5σ＝230／1000000

6σ＝3.4／1000000

7σ＝0／1000000

'''

if len(value\_l) <= 3:

return '','','','',''

column\_stdev = np.std(value\_l,ddof=1,axis=0)

three\_sigma= 3\*column\_stdev

# print('three\_sigma:',column\_stdev,three\_sigma)

temp\_l= value\_l

if len(value\_l) < 10:

temp\_l = value\_l + value\_l

if len(temp\_l)<10:

temp\_l = value\_l + value\_l + value\_l + value\_l+value\_l + value\_l + value\_l + value\_l+value\_l + value\_l

# print('------>>>temp\_l:',temp\_l)

data0 = np.array(temp\_l)

# print('------>>>data0:',data0)

try:

dip, p\_val = diptest.diptest(data0)

p\_val = '%f'%p\_val

# print('------>>>p\_val :',p\_val)

except RuntimeWarning as w:

print('calculate dip,p\_val RuntimeWarning:',w)

return '','','','',''

except Exception as e:

print('calculate dip,p\_val error:',e)

return '','','','',''

# print('dip,p\_val:',dip,p\_val)

item\_name ='value1'

data = pd.DataFrame({item\_name:value\_l})

# print('data--->',type(data),data)

u1 = data[item\_name].mean() # 计算均值

# std1 = data[item\_name].std() # 计算标准差

# t,pval=stats.kstest(data[item\_name], 'norm', (u1, std1))

# print('normality test t,pval:',str(t),str(pval))

# print('------')

# 正态性检验 → pvalue >0.05

n= float(len(value\_l))

#Item (xi-ẍ)^2

item\_l\_1 =[]

item\_l\_2 = []

item\_l\_3 = []

for i in value\_l:

temp1 = (i-u1)\*\*2

temp2 = (i-u1)\*\*3

temp3 = (i-u1)\*\*4

item\_l\_1.append(temp1)

item\_l\_2.append(temp2)

item\_l\_3.append(temp3)

# print('item\_l\_1--->',item\_l\_1)

# print('item\_l\_2--->',item\_l\_2)

# print('item\_l\_3--->',item\_l\_3)

sum\_item\_l\_1 = sum(item\_l\_1)

sum\_item\_l\_2 = sum(item\_l\_2)

sum\_item\_l\_3 = sum(item\_l\_3)

# print('sum\_item\_l\_1',sum\_item\_l\_1)

# print('sum\_item\_l\_2',sum\_item\_l\_2)

# print('sum\_item\_l\_3',sum\_item\_l\_3)

if n<=3 or sum\_item\_l\_1==0 or sum\_item\_l\_2==0 or sum\_item\_l\_3==0:

# print('len < 3--->')

if abs(u1) == 0:

return 'Nan',str(p\_val),'Nan','Nan','Nan'

else:

three\_CV = three\_sigma\*100/abs(u1)

# print('three\_CV:',three\_CV)

return 'Nan',str(p\_val),'Nan','Nan',str(round(three\_CV,6))

else:

try:

m3 = np.sqrt(n\*(n-1))/(n-2)\*((1/n\*sum\_item\_l\_2)/np.sqrt(1/n\*sum\_item\_l\_1)\*\*3)

# print('m3=',m3)

# print('d8:',n+1)

# print('d15:',1/n\*sum\_item\_l\_3/(1/n\*sum\_item\_l\_1)\*\*2)

# print('d16:',(n+1)\*1/n\*sum\_item\_l\_3/(1/n\*sum\_item\_l\_1)\*\*2-3\*(n-1))

# print('d6/d7',(n-1)/((n-2)\*(n-3)))

m4 = ((n-1)/((n-2)\*(n-3)))\*((n+1)\*1/n\*sum\_item\_l\_3/(1/n\*sum\_item\_l\_1)\*\*2-3\*(n-1))#(d6/d7)\*d16

# print('m4=',m4)

#(d14\*\*2+1)/(d17+3\*(d10/d7))

bc =(m3\*\*2+1)/(m4+3\*((n-1)\*\*2/((n-2)\*(n-3))))

# print('bc:',bc)

a\_L=0.05

a\_M=0.1

a\_U=0.32

a\_Q = (a\_U-a\_L)\*bc\*\*2+a\_L

# print ('a\_Q:',a\_Q)

a\_irr = np.sqrt((a\_U-a\_L)\*\*2\*bc)+a\_L

# print('a\_irr:',a\_irr)

except Exception as e:

# print('calculate error',e)

if abs(u1) == 0:

return 'Nan',str(p\_val),'Nan','Nan','Nan'

else:

three\_CV = three\_sigma\*100/abs(u1)

# print('three\_CV:',three\_CV)

return 'Nan',str(p\_val),'Nan','Nan',str(round(three\_CV,6))

if abs(u1) == 0:

return str(round(bc,6)),str(p\_val),str(round(a\_Q,6)),str(round(a\_irr,6)),'Nan'

else:

three\_CV = three\_sigma\*100/abs(u1)

# print('three\_CV:',three\_CV)

return str(round(bc,6)),str(p\_val),str(round(a\_Q,6)),str(round(a\_irr,6)),str(round(three\_CV,6))

def plot\_display\_y\_name(zmqSelectItems,filter1,filter2,tb\_data2\_len):

scatter\_item = []

for v in zmqSelectItems:

filterItems = v.split('&')

i = 0

for z in filterItems:

if i%2 ==0:

if filter1[0] == 'Station ID':

tmpFilter1 = filterItems[i].split('-')

tmpFilter1\_len = len(tmpFilter1)

if tmpFilter1\_len>2:

tmpFilter1\_str = []

for x in range(2,tmpFilter1\_len):

if x==2:

tmpFilter1\_str.append(tmpFilter1[x])

else:

tmpFilter1\_str.append('-')

tmpFilter1\_str.append(tmpFilter1[x])

filter\_str1=''.join(tmpFilter1\_str)

scatter\_item.append(filter\_str1)

elif tmpFilter1\_len>1:

tmpFilter1\_str = []

for x in range(1,tmpFilter1\_len):

if x==1:

tmpFilter1\_str.append(tmpFilter1[x])

else:

tmpFilter1\_str.append('-')

tmpFilter1\_str.append(tmpFilter1[x])

filter\_str1=''.join(tmpFilter1\_str)

scatter\_item.append(filter\_str1)

else:

scatter\_item.append(tmpFilter1[0])

else:

scatter\_item.append(filterItems[0])

else:

if filter2[0] == 'Station ID':

tmpFilter2 = filterItems[i].split('-')

tmpFilter2\_len = len(tmpFilter2)

if tmpFilter2\_len>2:

tmpFilter2\_str = []

for x in range(2,tmpFilter2\_len):

if x==2:

tmpFilter2\_str.append(tmpFilter2[x])

else:

tmpFilter2\_str.append('-')

tmpFilter2\_str.append(tmpFilter2[x])

filter\_str2=''.join(tmpFilter2\_str)

scatter\_item.append(filter\_str2)

elif tmpFilter2\_len>1:

tmpFilter2\_str = []

for x in range(1,tmpFilter2\_len):

if x==1:

tmpFilter2\_str.append(tmpFilter2[x])

else:

tmpFilter2\_str.append('-')

tmpFilter2\_str.append(tmpFilter1[x])

filter\_str2=''.join(tmpFilter2\_str)

scatter\_item.append(filter\_str2)

else:

scatter\_item.append(tmpFilter2[0])

else:

scatter\_item.append(filterItems[1])

i=i+1

plot\_scatter\_item = []

tmp = []

x\_flg = False

for x in range(0,len(scatter\_item)):

if x%2 ==0:

tmp = []

x\_flg = False

if not scatter\_item[x] == 'Off':

x\_flg = False

tmp.append(scatter\_item[x])

else:

x\_flg = True

else:

if not scatter\_item[x] == 'Off':

if x\_flg:

tmp.append(scatter\_item[x])

else:

tmp.append('&'+scatter\_item[x])

xx = int(x/2)

if len(tb\_data2\_len)>xx:

plot\_scatter\_item.append(''.join(tmp) + '('+str(tb\_data2\_len[xx])+')')

else:

plot\_scatter\_item.append(''.join(tmp))

return plot\_scatter\_item

def cpk\_plot(table\_data,zmqMsg):

try:

global filelogname

filelcpknew = '/tmp/CPK\_Log/temp/.cpknew.txt'

info = ''

i\_start = [i for i,x in enumerate(table\_data) if x=='Start\_Data']

i\_stop = [i for i,x in enumerate(table\_data) if x=='End\_Data']

tbdata=[] #取出数据

for i, v in enumerate(i\_start): #因start\_data和End\_data成对出现

tbdata.append(table\_data[i\_start[i]-36:i\_stop[i]])

item\_name=tbdata[0][1]

usl = tbdata[0][4]

lsl = tbdata[0][5]

usl\_orig = usl

lsl\_orig = lsl

set\_bins = tbdata[0][19]

path= '/tmp/CPK\_Log' #tbdata[0][30]

filelogname = path + '/temp/.logcpk.txt'

start\_time\_first = tbdata[0][21]

start\_time\_last = tbdata[0][22]

select\_new\_lsl = tbdata[0][7]

select\_new\_usl = tbdata[0][8]

limit\_apply = tbdata[0][9] #是否点击了UI apply

cpkLTHLD = tbdata[0][24]

cpkHTHLD = tbdata[0][25]

select\_color\_by = tbdata[0][31] #是否点击了color by left

select\_color\_by\_right = tbdata[0][32] #是否点击了color by right

range\_set\_lsl = tbdata[0][27] #get range lsl

range\_set\_usl = tbdata[0][28] #get range usl

zoom\_type = tbdata[0][18]

if limit\_apply == 1:

lsl = select\_new\_lsl

usl = select\_new\_usl

if select\_color\_by >0 or select\_color\_by\_right>0: # color by choose

tb\_data\_raw=[] #取出数据

for i, v in enumerate(i\_start): #因start\_data和End\_data成对出现

tb\_data\_raw.append(table\_data[i\_start[i]+1:i\_stop[i]])

tb\_data2=[]

tb\_data2\_len = []

for i,v in enumerate(tb\_data\_raw): # 删除列表空值

tmp = [i for i in v if i !='']

tb\_data2.append(tmp)

tb\_data2\_len.append(len(tmp))

all\_data = ([i for item in tb\_data2 for i in item]) # 二维列表拼接成一维列表

bc,p\_val,a\_Q,a\_irr,three\_CV = get\_coefficients(all\_data)

BMC = ''

if bc != '' and bc != 'Nan' and p\_val != '' and a\_Q != '' and p\_val != 'Nan' and a\_Q != 'Nan':

if float(p\_val) <= float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) <= float(a\_Q) and float(bc)<0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))>=-0.1:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))<-0.1:

BMC = 'NO'

else:

BMC = ''

else:

BMC = ''

mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk = cpk\_calc(all\_data, lsl, usl)

if cpk:

if is\_number(str(cpk)):

if float(cpk)>cpkHTHLD:

BMC = ''

if limit\_apply == 1:

if cpk ==None:

cpk\_value =''

else:

if cpk > 999999:

cpk\_value = str("%.3e" % cpk)

else:

cpk\_value = str("%.3f" % cpk)

with open(filelcpknew, 'w') as file\_object:

file\_object.write("DONE,"+str(cpk\_value))

image\_name ='cpk.png'

pic\_path = path +'/temp/'

pic\_path = pic\_path + image\_name

zmqItem = zmqMsg.split('##')

filter1 = ['']

filter2 = ['']

if select\_color\_by>0:

filter1 = get\_redis\_data('select\_filter\_by\_1')

if select\_color\_by\_right>0:

filter2 = get\_redis\_data('select\_filter\_by\_2')

# print('filter1,filter2:',filter1,filter2,zmqMsgName)

zmqSelectItems = plot\_display\_y\_name(zmqItem,filter1,filter2,tb\_data2\_len)

draw\_more\_histogram(tb\_data2,all\_data,zmqSelectItems,item\_name,lsl, usl, mean,

max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk,

pic\_path,set\_bins,start\_time\_first,start\_time\_last,

BMC,zoom\_type,range\_set\_lsl,range\_set\_usl,lsl\_orig,usl\_orig)

else: #没有选择color by

tb\_raw\_data = tbdata[0][37:] #注意 原始数据有空值

tb\_data=[]

# for i,v in enumerate(tb\_raw\_data): # 删除列表空值

# tmp = [i for i in v if i !='']

# tb\_data.append(tmp)

tb\_data = [i for i in tb\_raw\_data if i !='']

# print(tb\_data) # 没有空值的数据

bc,p\_val,a\_Q,a\_irr,three\_CV = get\_coefficients(tb\_data)

BMC = ''

if bc != '' and bc != 'Nan' and p\_val != '' and a\_Q != '' and p\_val != 'Nan' and a\_Q != 'Nan':

if float(p\_val) <= float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) <= float(a\_Q) and float(bc)<0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))>=-0.1:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))<-0.1:

BMC = 'NO'

else:

BMC = ''

else:

BMC = ''

mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk = cpk\_calc(tb\_data, lsl, usl)

if cpk:

if is\_number(str(cpk)):

if float(cpk)>float(cpkHTHLD):

BMC = ''

if limit\_apply == 1:

if cpk ==None:

cpk\_value =''

else:

if cpk > 999999:

cpk\_value = str("%.3e" % cpk)

else:

cpk\_value = str("%.3f" % cpk)

with open(filelcpknew, 'w') as file\_object:

file\_object.write("DONE,"+str(cpk\_value))

image\_name ='cpk.png'

pic\_path = path +'/temp/'

pic\_path = pic\_path + image\_name

draw\_histogram(tb\_data,item\_name,lsl, usl, mean, max\_num, min\_num,

stdev, x1, y1, cpu, cpl, cpk, pic\_path,set\_bins,

start\_time\_first,start\_time\_last,BMC,zoom\_type,

range\_set\_lsl,range\_set\_usl,lsl\_orig,usl\_orig)

info = 'cpk and plot draw finished!!!'

with open(filelogname, 'w') as file\_object:

file\_object.write("PASS,cpk and plot draw finished")

return info

except Exception as e:

with open(filelogname, 'w') as file\_object:

file\_object.write("FAIL,error cpk\_plot function")

print('error cpk\_plot function:',e)

def draw\_more\_histogram(column\_category\_data\_list,column\_data,zmqItem,item\_name,

lsl, usl, mean, max\_num, min\_num, stdev, x1, y1,cpu, cpl,

cpk, pic\_path,set\_bins,start\_time\_first,start\_time\_last,

bmc,zoom\_type,range\_set\_lsl,range\_set\_usl,lsl\_orig,usl\_orig):

"""

"""

# range = get\_limit\_range(lsl, usl)

# range = round((range /set\_bins), 5)

# bins = np.arange(lsl, usl, range) # 必须是单调递增的

bins,bins\_l,bins\_h = get\_bins(min\_num,max\_num,lsl,usl,set\_bins,zoom\_type)

range\_bins,range\_bins\_l,range\_bins\_h = bins,bins\_l,bins\_h

if zoom\_type == 'range':

range\_bins,range\_bins\_l,range\_bins\_h = get\_bins(min\_num,max\_num,range\_set\_lsl,range\_set\_usl,set\_bins,zoom\_type)

probability\_distribution\_extend\_by\_color(column\_category\_data\_list,column\_data,zmqItem,bins, 0, item\_name,

lsl, usl, mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk,

pic\_path,bins\_l,bins\_h,start\_time\_first,start\_time\_last,

bmc,zoom\_type,range\_set\_lsl,range\_set\_usl,range\_bins,range\_bins\_l,range\_bins\_h,

lsl\_orig,usl\_orig)

return True

def checkItemName(item\_name,item\_length):

if len(item\_name) > item\_length:

if item\_name[item\_length] == '\_' or item\_name[item\_length] == ' ':

item\_name = item\_name[0:item\_length+1] + '\n' + item\_name[item\_length+1:]

else:

item\_name1 = item\_name[0:item\_length]

item\_name\_tmp = item\_name1[::-1]

x1=item\_name\_tmp.find('\_')

x2=item\_name\_tmp.find(' ')

x\_len = 0

if x1 == -1 and x2 == -1:

x\_len = 0

elif x1 == -1:

x\_len = x2

elif x2 ==-1:

x\_len = x1

elif x1>x2:

x\_len = x2

else:

x\_len = x1

x\_len = len(item\_name1) - x\_len

item\_name = item\_name[0:x\_len] + '\n' + item\_name[x\_len:]

return item\_name

def probability\_distribution\_extend\_by\_color(column\_category\_data\_list,data,zmqItem,bins,margin,item\_name,

lsl,usl,mean,max\_num,min\_num,stdev,x1,y1,cpu,cpl,cpk,pic\_path,

bins\_l,bins\_h,start\_time\_first,start\_time\_last,bmc,zoom\_type,

range\_set\_lsl,range\_set\_usl,range\_bins,range\_bins\_l,range\_bins\_h,

lsl\_orig,usl\_orig):

max\_num\_orig = max\_num

min\_num\_orig = min\_num

mean\_orig = mean

stdev\_orig = stdev

if zoom\_type == 'range':

bins = sorted(range\_bins)

length = len(range\_bins)

else:

bins = sorted(bins)

length = len(bins)

try:

permin=round(np.percentile(data,2),3)

except Exception as e:

permin = ''

try:

permax=round(np.percentile(data,98),3)

except Exception as e:

permax = ''

intervals = np.zeros(length+1)

for value in data:

i = 0

while i < length and value >= bins[i]:

i += 1

intervals[i] += 1

intervals = intervals / float(len(data))

plt.ion() # 开启interactive mode

plt.figure(1) # 创建图表1

plt.xlim(min(bins) - margin, max(bins) + margin)

bins.insert(0, -999)

# plt.title("probability-distribution")

plt.bar(bins, intervals,color=['r'], label='')#频率分布

x\_ticks,labels = plt.xticks()

# x\_ticks\_start=round(x\_ticks[0],2)

# x\_ticks\_end = round(x\_ticks[len(x\_ticks) - 1],2)

y\_ticks, labels = plt.yticks()

# print('y\_ticks--->',y\_ticks,y\_ticks[len(y\_ticks) - 1],y\_ticks[0])

y\_tick\_cure=round(y\_ticks[len(y\_ticks) - 3],5)

y\_ticks=round(y\_ticks[len(y\_ticks) - 1],5)

# print("x\_ticks\_start,x\_ticks\_end--->",x\_ticks\_start,x\_ticks\_end)

# print('y\_ticks\_end--->',y\_ticks)

# plt.show()

plt.close(1)

plt.figure(2,dpi=150) # 创建图表2,facecolor='blue',edgecolor='black'

fig, axes = plt.subplots(1, 0, figsize=(6, 5), facecolor='#ccddef')

plt.axes([0.1, 0.17, 0.65, 0.7]) # [左, 下, 宽, 高] 规定的矩形区域 （全部是0~1之间的数，表示比例）

if stdev =='nan':

pass

else:

if stdev > 999999:

stdev = str("%.3e" % stdev)

else:

stdev = str("%.3f" % stdev)

if cpl ==None:

cpl\_value =''

else:

if cpl > 999999:

cpl\_value = str("%.3e" % cpl)

else:

cpl\_value = str("%.3f" % cpl)

if cpu ==None:

cpu\_value =''

else:

if cpu > 999999:

cpu\_value = str("%.3e" % cpu)

else:

cpu\_value = str("%.3f" % cpu)

if cpk ==None:

cpk\_value =''

else:

if cpk > 999999:

cpk\_value = str("%.3e" % cpk)

else:

cpk\_value = str("%.3f" % cpk)

if max\_num > 999999:

max\_num = str("%.6e" % max\_num)

else:

max\_num = str("%.3f" % max\_num)

if mean > 999999:

mean = str("%.6e" % mean)

else:

mean = str("%.3f" % mean)

if min\_num > 999999:

min\_num = str("%.6e" % min\_num)

else:

min\_num = str("%.3f" % min\_num)

if len(data) > 1000000000000:

sample\_n = str("%.e" % len(data))

else:

sample\_n = str("%.f" % len(data))

info = "Samples:" + sample\_n + ' ' +"Max:" + str(max\_num\_orig) + ' ' + "Min:" + str(min\_num\_orig) + '\n' + "Mean:" + str(mean\_orig) + ' '+ "Std:" + stdev + ' ' + "Cpl:" + cpl\_value + ' ' + "Cpu:" + cpu\_value + '\n' + "Cpk:" + cpk\_value + ' ' + '02%:'+ str(permin) + ' ' + '98%:' + str(permax) + ' '+ "Bimodal:" + bmc

item\_name = checkItemName(str(item\_name),55)

plt.title(item\_name,size=11,verticalalignment='bottom')

# plt.xlabel(str(start\_time\_first)+' -- '+str(start\_time\_last))

plt.ylabel('Count')

# plt.title(item\_name+"\nCpk={0}".format(str("%.6f" % cpk)))

# plt.hist(x=data, bins=bins, density=False, histtype='bar', color=['r'])

bins = [round(x,5) for x in bins]

bins=sorted(bins)

# print('----->bins-->',bins)

# print(' more draw category data:',len(column\_category\_data\_list),column\_category\_data\_list)

color\_l = ['#0000FF','#FF0000','#008000','#00FFFF','#9400D3','#8B008B','#B8860B','#FFA500','#A9A9A9','#FFFF00']

color\_dpf = ['#BFF128','#87A922','#B9CC81','#4B5D16','#5CB200','#B1FF65','#8EE53F','#58BC08','#4DA409','#C1FD95']

l=0

l\_len=[]

x = 0

for category\_data in column\_category\_data\_list:

try:

pdf\_check = redisClient.get('Set\_CPK\_CheckBox\_PDF').decode('utf-8')

if pdf\_check == '1':

category\_pdf = color\_dpf[x%10]

# if lsl\_orig =='' or lsl\_orig =='NA':

# lsl\_orig = min(category\_data)

# if usl\_orig =='' or usl\_orig =='NA':

# usl\_orig = max(category\_data)

# x\_axle = np.arange(lsl\_orig, usl\_orig, 0.01)

xx\_min = float(min(category\_data))

xx\_max = float(max(category\_data))

x\_axle = np.arange(xx\_min,xx\_max, (xx\_max-xx\_min)/100)

pdf\_values = normfun(x\_axle, float(mean\_orig), float(stdev\_orig))

cur\_y\_max=max(pdf\_values)

y\_axle = (len(category\_data) \* y\_tick\_cure/cur\_y\_max)\*pdf\_values

plt.plot(x\_axle,y\_axle,linewidth=1, color=category\_pdf)

except Exception as e:

print('error\_pdf: '+str(e))

category\_color = color\_l[x%10]

n=len(category\_data)

l=l+n

# print('category len:',n)

l\_len.append(n)

if len(column\_category\_data\_list) == 1:

if n>0:

plt.hist(category\_data, bins=bins, label=zmqItem[x], color=category\_color ,histtype='stepfilled',edgecolor=category\_color,linewidth=1.0,align='mid',density=False) #time分布

else:

plt.hist([0], bins=[0], label=zmqItem[x], color=category\_color ,histtype='stepfilled',edgecolor=category\_color,linewidth=1.0,align='mid',density=False)

else:

plt.hist(category\_data, bins=bins, label=zmqItem[x], color='white' ,histtype='step',edgecolor=category\_color,linewidth=1.0,align='mid',density=False) #time分布

x=x+1

y\_ticks = max(l\_len) \* (y\_ticks+0.04)

if zoom\_type == 'range':

range\_value = get\_limit\_range(range\_bins\_l, range\_bins\_h)

else:

range\_value = get\_limit\_range(bins\_l, bins\_h)

range\_value =round(range\_value/5,5)

if zoom\_type =='data':

range\_offset = abs(float(max\_num\_orig) - float(min\_num\_orig))\*0.2

plt.xlim(float(min\_num\_orig)-float(range\_offset), float(max\_num\_orig)+float(range\_offset))

elif zoom\_type =='range':

if (range\_set\_lsl =='' or range\_set\_lsl =='NA') and (range\_set\_usl!='NA'and range\_set\_usl!=''):

plt.xlim(float(min\_num\_orig)\*0.999, range\_set\_usl)

elif (range\_set\_usl =='' or range\_set\_usl =='NA') and (range\_set\_lsl!='NA'and range\_set\_lsl!=''):

plt.xlim(range\_set\_lsl, float(max\_num\_orig)\*1.001)

elif (range\_set\_usl =='' or range\_set\_usl =='NA') and (range\_set\_lsl=='NA'or range\_set\_lsl==''):

plt.xlim(float(min\_num\_orig)\*0.999, float(max\_num\_orig)\*1.001)

elif range\_set\_lsl == range\_set\_usl:

plt.xlim(range\_set\_lsl, range\_set\_usl\*1.001)

else:

plt.xlim(range\_set\_lsl, range\_set\_usl)

else:

if (lsl =='' or lsl =='NA') and (usl!='NA'and usl!=''):

plt.xlim(float(min\_num\_orig)\*0.999, usl+range\_value)

elif (usl =='' or usl =='NA') and (lsl!='NA'and lsl!=''):

plt.xlim(lsl-range\_value, float(max\_num\_orig)\*1.001)

elif (usl =='' or usl =='NA') and (lsl=='NA'or lsl==''):

plt.xlim(float(min\_num\_orig)\*0.999, float(max\_num\_orig)\*1.001)

else:

plt.xlim(bins\_l-range\_value, bins\_h+range\_value)

plt.ylim((0, y\_ticks)) # 设置y轴scopex

ax=plt.gca()

ax.spines['bottom'].set\_linewidth(1.0)

ax.spines['left'].set\_linewidth(1.0)

ax.spines['right'].set\_linewidth(1.0)

ax.spines['top'].set\_linewidth(1.0)

# ax.spines['bottom'].set\_position(('data', 5))

# ax.spines['left'].set\_position(('data', 5))

# ax.spines['right'].set\_position(-5)

# ax.spines['top'].set\_position(-5)

# plt.plot(x1, y1, 'k--', label="", linewidth=1.0, color='lime') # 画正态分布曲线

if lsl !='' and lsl !='NA' and zoom\_type =='limit':

plt.plot([lsl, lsl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画lower limit线，

plt.text(lsl, y\_ticks / 3, ' LSL\n' + ' ' + str(lsl), fontdict={'size': 8, 'color': 'r'})

if usl != '' and usl != 'NA' and zoom\_type =='limit':

plt.plot([usl, usl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画upper limit线，

plt.text(usl, y\_ticks / 2, ' USL\n' + ' ' + str(usl), fontdict={'size': 8, 'color': 'r'})

plt.text(0.0,-0.236, info,fontsize=10,ha="left",transform=ax.transAxes)

plt.legend(bbox\_to\_anchor=(1.37,1),loc="upper right",fontsize=8,framealpha=1,edgecolor='royalblue',borderaxespad=0.07)

plt.grid(linestyle=':',c='gray') # 生成网格

plt.savefig(pic\_path,dpi=200)

plt.draw()

# plt.show()

plt.close('all')

plt.ioff()

def verify\_limit(lsl,usl):

if lsl != None:

lsl.replace(' ','')

if usl != None:

usl.replace(' ','')

try:

lsl = float(eval(lsl))

except:

lsl = None

try:

usl = float(eval(usl))

except:

usl = None

# if type(lsl) == float and type(usl) == float:

# print('after verify lsl===>',lsl)

# print('after verify usl===>',usl)

return lsl,usl

def cpk\_calc(df\_data,lsl,usl):

"""

:param df\_data: list

:param usl: 数据指标上限

:param lsl: 数据指标下限

:return:

"""

sigma = 3

# print('limit--->',lsl,usl)

# 数据平均值

# print('df\_data in cpk\_calc:',df\_data)

mean = np.mean(df\_data)#

# mean = round(mean,3)

# print('mean ---->',mean)

# 数据max值

if len(df\_data)>0:

max\_num = max(df\_data)

else:

max\_num = 0

# print('max\_num ---->',max\_num)

# 数据min值

if len(df\_data)>0:

min\_num = min(df\_data)

else:

min\_num = 0

# print('min\_num ---->',min\_num)

# a = np.array([[1, 2], [3, 4]])

# print('a--->', type(a))

# print('gobal std:',np.std(a))#全局标准差

# print('each line std:',np.std(a, axis=0,ddof=1))

# print("each row std:",np.std(a, axis=1,ddof=1))

# 数据标准差

if len(df\_data)==1:

stdev =0.00

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

else:

try:

stdev = np.std(df\_data,ddof=1,axis=0)

except Exception as e:

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

# print('stdev ---->',stdev)

if stdev == 0:#stop count cpk

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

# 生成横轴数据平均分布

# x1 = np.linspace(mean - sigma \* stdev, mean + sigma \* stdev, 1000)

# print('x1 ---->',x1)

# 计算正态分布曲线

# y1 = np.exp(-(x1 - mean) \*\* 2 / (2 \* stdev \*\* 2)) / (math.sqrt(2 \* math.pi) \* stdev)

# print('y1 ---->',y1)

x1,y1 = None,None

if (lsl != 'NA' and lsl != '') and (usl == 'NA' or usl == ''):

cpl = (mean - lsl) / (sigma \* stdev)

# print('====>>>>>>cpl',cpl)

return (mean,max\_num,min\_num,stdev,None,None,None,cpl,None)

if (usl != 'NA' and usl != '') and (lsl == 'NA' or lsl == ''):

# print('====>>>>>>=====cpu')

cpu = (usl - mean) / (sigma \* stdev)

# print('====>>>>>>cpu',cpu)

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,None,None,cpu,None,None)

if lsl == 'NA' or usl == 'NA' or lsl == '' or usl == '':

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

cpu = (usl - mean) / (sigma \* stdev)

cpl = (mean - lsl) / (sigma \* stdev)

# print('cpu ---->',cpu)

# print('cpl ---->',cpl)

# 得出cpk

cpk = min(cpu, cpl)

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,x1,y1,cpu,cpl,cpk)

def draw\_histogram(column\_data,item\_name,lsl,usl,mean,max\_num,min\_num,

stdev,x1,y1,cpu,cpl,cpk,pic\_path,set\_bins,start\_time\_first,

start\_time\_last,bmc,zoom\_type,range\_set\_lsl,range\_set\_usl,lsl\_orig,usl\_orig):

bins,bins\_l,bins\_h = get\_bins(min\_num,max\_num,lsl,usl,set\_bins,zoom\_type)

range\_bins,range\_bins\_l,range\_bins\_h = bins,bins\_l,bins\_h

if zoom\_type == 'range':

range\_bins,range\_bins\_l,range\_bins\_h = get\_bins(min\_num,max\_num,range\_set\_lsl,range\_set\_usl,set\_bins,zoom\_type)

probability\_distribution\_extend(column\_data,bins,0,item\_name,lsl,usl,mean,max\_num,min\_num,

stdev,x1,y1,cpu,cpl,cpk,pic\_path,bins\_l,bins\_h,start\_time\_first,

start\_time\_last,bmc,zoom\_type,range\_set\_lsl,range\_set\_usl,

range\_bins,range\_bins\_l,range\_bins\_h,lsl\_orig,usl\_orig)

return True

def get\_bins(min\_num,max\_num,lsl,usl,set\_bins,zoom\_type='limit'):

bins\_l = 0

bins\_h = 0

if lsl == 'NA' or usl == 'NA' or lsl == '' or usl == '' or zoom\_type == 'data':

bins\_l,bins\_h = min\_num,max\_num

else:

if min\_num < lsl and max\_num < lsl:

bins\_l = min\_num

bins\_h = usl

elif min\_num < lsl and max\_num > lsl and max\_num <= usl:

bins\_l = min\_num

bins\_h = usl

elif min\_num < lsl and max\_num > usl:

bins\_l = min\_num

bins\_h = max\_num

elif lsl <= min\_num and min\_num <= usl and max\_num <= usl:

bins\_l = lsl

bins\_h = usl

elif lsl <= min\_num and min\_num <= usl and max\_num > usl:

bins\_l = lsl

bins\_h = max\_num

elif min\_num > usl:

bins\_l = lsl

bins\_h = max\_num

range\_value = get\_limit\_range(bins\_l,bins\_h)

if lsl == 'NA' or usl == 'NA' or lsl == '' or usl == '' or zoom\_type == 'data':

if range\_value ==0 and min\_num > 0:

range\_value = min\_num\*0.2

bins\_l = (bins\_l - min\_num\*0.1)

bins\_h = (bins\_h + min\_num\*0.1)

elif range\_value ==0 and min\_num == 0:

range\_value = 6

bins\_l = - 3

bins\_h = 3

elif range\_value ==0 and min\_num <0:

range\_value = 6

bins\_l = min\_num - 3

bins\_h = min\_num + 3

# print('range\_value0-->',range\_value,min\_num,bins\_l,bins\_h)

else:

if min\_num > 1 and range\_value < 1 and range\_value !=0:

range\_value = min\_num\*0.05

bins\_l = (bins\_l - min\_num\*0.025)

bins\_h = (bins\_h + min\_num\*0.025)

elif min\_num > 0.001 and min\_num < 1 and range\_value < 1 and range\_value !=0:

range\_value = min\_num\*0.2

bins\_l = (bins\_l - min\_num\*0.1)

bins\_h = (bins\_h + min\_num\*0.1)

else:

range\_value = range\_value\*0.4

bins\_l = (bins\_l - range\_value\*0.2)

bins\_h = (bins\_h + range\_value\*0.2)

else:

if range\_value == 0 and lsl !=0:

range\_value = lsl\*0.2

bins\_l = bins\_l - lsl\*0.1

bins\_h = bins\_h + usl\*0.1

elif range\_value == 0 and lsl ==0:

range\_value = 6

bins\_l = - 3

bins\_h = 3

range\_value = round((range\_value/set\_bins),12)

# print('range\_value2-->',range\_value)

# print('=====>',bins\_l,bins\_h,range\_value)

bins = np.arange(bins\_l, bins\_h, range\_value)#必须是单调递增的

if bins\_l<0:

bins = np.arange(bins\_h, bins\_l, -range\_value)

# print('lsl,usl,min\_num,max\_num,bins\_l,bins\_h in get\_bins=====>',lsl,usl,min\_num,max\_num,bins\_l,bins\_h)

return bins,bins\_l,bins\_h

def probability\_distribution\_extend(data,bins,margin,item\_name,lsl,usl,mean,max\_num,

min\_num,stdev,x1,y1,cpu,cpl,cpk,pic\_path,bins\_l,bins\_h,

start\_time\_first,start\_time\_last,bmc,zoom\_type,range\_set\_lsl,

range\_set\_usl,range\_bins,range\_bins\_l,range\_bins\_h,lsl\_orig,usl\_orig):

try:

permin=round(np.percentile(data,2),3)

except Exception as e:

permin = ''

try:

permax=round(np.percentile(data,98),3)

except Exception as e:

permax = ''

max\_num\_orig = max\_num

min\_num\_orig = min\_num

mean\_orig = mean

stdev\_orig = stdev

data\_orig = data

if zoom\_type =='range':

bins = sorted(range\_bins)

length = len(range\_bins)

else:

bins = sorted(bins)

length = len(bins)

intervals = np.zeros(length+1)

for value in data:

i = 0

while i < length and value >= bins[i]:

i += 1

intervals[i] += 1

intervals = intervals / float(len(data))

plt.ion() # 开启interactive mode

plt.figure(1) # 创建图表1

plt.xlim(min(bins) - margin, max(bins) + margin)

bins.insert(0, -999)

# plt.title("probability-distribution",size=8,verticalalignment='bottom')

plt.bar(bins, intervals,color=['r'], label='')#频率分布

x\_ticks,labels = plt.xticks()

#x\_ticks\_start=round(x\_ticks[0],2)

#x\_ticks\_end = round(x\_ticks[len(x\_ticks) - 1],2)

y\_ticks, labels = plt.yticks()

# print('=====y\_ticks, labels:',y\_ticks)

# print('y\_ticks--->',y\_ticks,y\_ticks[len(y\_ticks) - 1],y\_ticks[0])

y\_tick\_cure=round(y\_ticks[len(y\_ticks) - 3],5)

y\_ticks=round(y\_ticks[len(y\_ticks) - 1],5)

# print('y\_tick\_cure:',y\_tick\_cure)

# print("x\_ticks\_start,x\_ticks\_end--->",x\_ticks\_start,x\_ticks\_end)

# print('y\_ticks--->',y\_ticks)

# plt.show()

plt.close(1)

plt.figure(2,dpi=150) # 创建图表2

fig, axes = plt.subplots(1, 0, figsize=(6, 5), facecolor='#ccddef')

plt.axes([0.11, 0.17, 0.85, 0.7]) # [左, 下, 宽, 高] 规定的矩形区域 （全部是0~1之间的数，表示比例）

if stdev =='nan':

pass

else:

if stdev > 999999:

stdev = str("%.3e" % stdev)

else:

stdev = str("%.3f" % stdev)

if cpl ==None:

cpl\_value =''

else:

if cpl > 999999:

cpl\_value = str("%.3e" % cpl)

else:

cpl\_value = str("%.3f" % cpl)

if cpu ==None:

cpu\_value =''

else:

if cpu > 999999:

cpu\_value = str("%.3e" % cpu)

else:

cpu\_value = str("%.3f" % cpu)

if cpk ==None:

cpk\_value =''

else:

if cpk > 999999:

cpk\_value = str("%.3e" % cpk)

else:

cpk\_value = str("%.3f" % cpk)

if max\_num > 999999:

max\_num = str("%.6e" % max\_num)

else:

max\_num = str("%.3f" % max\_num)

if mean > 999999:

mean = str("%.6e" % mean)

else:

mean = str("%.3f" % mean)

if min\_num > 999999:

min\_num = str("%.6e" % min\_num)

else:

min\_num = str("%.3f" % min\_num)

if len(data) > 1000000000000:

sample\_n = str("%.e" % len(data))

else:

sample\_n = str("%.f" % len(data))

info = "Samples:" + sample\_n + ' ' +"Max:" + str(max\_num\_orig) + ' ' + "Min:" + str(min\_num\_orig) + '\n' + "Mean:" + str(mean\_orig) + ' '+ "Std:" + stdev + ' ' + "Cpl:" + cpl\_value + ' ' + "Cpu:" + cpu\_value + '\n' + "Cpk:" + cpk\_value + ' ' + '02%:'+ str(permin) + ' ' + '98%:' + str(permax) + ' '+ "Bimodal:" + bmc

item\_name = checkItemName(str(item\_name),55)

plt.title(item\_name,size=11,verticalalignment='bottom')

# plt.xlabel(str(start\_time\_first)+' -- '+str(start\_time\_last))

plt.ylabel('Count')

bins = [round(x,5) for x in bins]

bins=sorted(bins)

try:

pdf\_check = redisClient.get('Set\_CPK\_CheckBox\_PDF').decode('utf-8')

if pdf\_check == '1':

# if lsl\_orig =='' or lsl\_orig =='NA':

# lsl\_orig = min(data\_orig)

# if usl\_orig =='' or usl\_orig =='NA':

# usl\_orig = max(data\_orig)

# x\_axle = np.arange(lsl\_orig, usl\_orig, 0.01)

x\_axle = np.arange(float(min\_num\_orig),float(max\_num\_orig), (float(max\_num\_orig)-float(min\_num\_orig))/100)

pdf\_values = normfun(x\_axle, float(mean\_orig), float(stdev\_orig))

cur\_y\_max=max(pdf\_values)

y\_axle = (len(data\_orig) \* y\_tick\_cure/cur\_y\_max)\*pdf\_values

plt.plot(x\_axle,y\_axle,linewidth=1, color='#FD411E')

except Exception as e:

print('error pdf: '+str(e))

# plt.hist(data, bins=bins, label=info, histtype='stepfilled',color = 'blue', edgecolor='blue', linewidth=1.0,align='mid',density=False) #time分布

plt.hist(data, bins=bins, histtype='stepfilled',color = 'blue', edgecolor='blue', linewidth=1.0,align='mid',density=False) #time分布

if zoom\_type =='range':

range\_value = get\_limit\_range(range\_bins\_l, range\_bins\_h)

else:

range\_value = get\_limit\_range(bins\_l, bins\_h)

range\_value =round(range\_value/5,5)

if zoom\_type =='data':

range\_offset = abs(float(max\_num\_orig) - float(min\_num\_orig))\*0.2

plt.xlim(float(min\_num\_orig)-float(range\_offset), float(max\_num\_orig)+float(range\_offset))

elif zoom\_type =='range':

if (range\_set\_lsl =='' or range\_set\_lsl =='NA') and (range\_set\_usl!='NA'and range\_set\_usl!=''):

plt.xlim(float(min\_num\_orig)\*0.999, range\_set\_usl)

elif (range\_set\_usl =='' or range\_set\_usl =='NA') and (range\_set\_lsl!='NA'and range\_set\_lsl!=''):

plt.xlim(range\_set\_lsl, float(max\_num\_orig)\*1.001)

elif (range\_set\_usl =='' or range\_set\_usl =='NA') and (range\_set\_lsl=='NA'or range\_set\_lsl==''):

plt.xlim(float(min\_num\_orig)\*0.999, float(max\_num\_orig)\*1.001)

elif range\_set\_lsl == range\_set\_usl:

plt.xlim(range\_set\_lsl, range\_set\_usl\*1.001)

else:

plt.xlim(range\_set\_lsl, range\_set\_usl)

else:

if (lsl =='' or lsl =='NA') and (usl!='NA'and usl!=''):

plt.xlim(float(min\_num\_orig)\*0.999, usl+range\_value)

elif (usl =='' or usl =='NA') and (lsl!='NA'and lsl!=''):

plt.xlim(lsl-range\_value, float(max\_num\_orig)\*1.001)

elif (usl =='' or usl =='NA') and (lsl=='NA'or lsl==''):

plt.xlim(float(min\_num\_orig)\*0.999, float(max\_num\_orig)\*1.001)

else:

plt.xlim(bins\_l-range\_value, bins\_h+range\_value)

y\_ticks = len(data) \* y\_ticks

plt.ylim((0, y\_ticks)) # 设置y轴scopex

ax=plt.gca()

ax.spines['bottom'].set\_linewidth(1)

ax.spines['left'].set\_linewidth(1)

ax.spines['right'].set\_linewidth(1)

ax.spines['top'].set\_linewidth(1)

if lsl !='' and lsl !='NA' and zoom\_type =='limit':

plt.plot([lsl, lsl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画lower limit线，

plt.text(lsl, y\_ticks / 3, ' LSL\n' + ' ' + str(lsl), fontdict={'size': 8, 'color': 'r'})

if usl != '' and usl != 'NA' and zoom\_type =='limit':

plt.plot([usl, usl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画upper limit线，

plt.text(usl, y\_ticks / 2, ' USL\n' + ' ' + str(usl), fontdict={'size': 8, 'color': 'r'})

plt.text(0.0,-0.236, info,fontsize=10,ha="left",transform=ax.transAxes)

plt.legend(bbox\_to\_anchor=(0.9,-0.09),loc="best",fontsize=10,framealpha=0,edgecolor='royalblue',borderaxespad=0.1)

plt.grid(linestyle=':',c='gray') # 生成网格

plt.savefig(pic\_path,dpi=200)

plt.draw()

plt.close('all')

plt.ioff()

def get\_limit\_range(lsl,usl):

# print('lsl,usl----->', lsl, usl)

range\_v = 0

if lsl < 0 and usl <= 0:

range\_v = abs(lsl) - abs(usl)

elif lsl < 0 and usl >= 0:

range\_v = abs(lsl) + usl

elif lsl >= 0 and (usl > 0):

range\_v = usl - lsl

else:

print('get\_limit\_range 00000')

range\_v = round(range\_v, 5)

# print('range in get\_limit\_range----->', range)

return range\_v

def cpk(message):

print("this function is generate cpk plot......")

val = r.get(message)

# time.sleep(5) #测试python 执行时间 5s

if val:

return val

else:

return b'None'

def run(n):

while True:

try:

print("wait for cpk client ...")

zmqMsg = socket.recv()

socket.send(b'cpk.png')

if len(zmqMsg)>0:

key = zmqMsg.decode('utf-8')

print("message from cpk client:", key)

table\_data = get\_redis\_data(key)

if len(table\_data)>0:

cpk\_plot(table\_data,key)

else:

print("---get data error")

# socket.send(ret.decode('utf-8').encode('ascii'))

else:

time.sleep(0.05)

# socket.send(b'cpk.png')

except Exception as e:

print('error:',e)

if \_\_name\_\_ == '\_\_main\_\_':

# t1 = threading.Thread(target=run, args=("<<cpk1>>",))

# t1.start()

run(0)

#! /usr/bin/env python3

# --\*-- coding: utf-8 ---\*---

import sys,os,time,math,re

import time

import threading

BASE\_DIR=os.path.dirname(os.path.abspath(\_\_file\_\_))

sys.path.insert(0,BASE\_DIR+'/site-packages/')

try:

import csv

except Exception as e:

print('e---->',e)

# print('python import ---->matplotlib')

try:

import matplotlib

matplotlib.use("Agg")

import matplotlib.pyplot as plt

except Exception as e:

print('e---->',e)

# print('python import ----> matplotlib.colors')

try:

import matplotlib.colors as colors

except Exception as e:

print('e---->',e)

# print('python import ----> FontProperties')

try:

from matplotlib.font\_manager import FontProperties

except Exception as e:

print('e---->',e)

# print('python import ----> numpy')

try:

import numpy as np

except Exception as e:

print('e--->',e)

# print('python import ----> pandas')

try:

import pandas as pd

except Exception as e:

print('e--->',e)

# print('python import ----> openpyxl')

try:

import openpyxl

except Exception as e:

print('import openpyxl error:',e)

# print('python import ----> xlsxwriter')

try:

import xlsxwriter

except Exception as e:

print('import xlsxwriter error:',e)

try:

import zmq

except Exception as e:

print('import zmq error:',e)

# print('python import ----> zmg')

try:

import redis

except Exception as e:

print('import redis error:',e)

# print('python import ----> redis')

print(sys.getdefaultencoding())

# import zmq

# import redis

redisClient = redis.Redis(host='localhost', port=6379, db=0)

context = zmq.Context()

socket = context.socket(zmq.REP)

socket.setsockopt(zmq.LINGER,0)

socket.bind("tcp://127.0.0.1:3110")

filelogname = '/tmp/CPK\_Log/temp/.logcor.txt'

table\_data\_correlation\_y = []

select\_x\_flag = 0

xy\_reverse\_flag = 0

def get\_redis\_data(zmqMsg):

tb = redisClient.get(zmqMsg)

tb\_data=[]

if tb:

tb=tb.decode('utf-8')

tb=tb.split("\n")

tb=(tb[1:-1]) #去掉数据库首尾元素

for i in tb:

k=re.sub('\"','',i) #去掉数据库引号

h=re.sub(',','',k) #去掉数据库逗号

m=h.strip() #去掉数据库首尾空白

if is\_number(m):

tb\_data.append(eval(m)) #去掉数字的引号

else:

tb\_data.append(m)

else:

tb\_data.append('')

return tb\_data

def is\_number(s):

try:

float(s)

return True

except ValueError:

pass

try:

import unicodedata

unicodedata.numeric(s)

return True

except (TypeError, ValueError):

pass

return False

def correlation\_plot(table\_data\_x,table\_data\_y,forXorY):

try:

global filelogname

info = ''

i\_start\_x = [i for i,x in enumerate(table\_data\_x) if x=='Start\_Data']

i\_stop\_x = [i for i,x in enumerate(table\_data\_x) if x=='End\_Data']

tbdata\_x=[] #取出数据

for i, v in enumerate(i\_start\_x): #因start\_data和End\_data成对出现

tbdata\_x.append(table\_data\_x[i\_start\_x[i]-36:i\_stop\_x[i]])

i\_start\_y = [i for i,x in enumerate(table\_data\_y) if x=='Start\_Data']

i\_stop\_y = [i for i,x in enumerate(table\_data\_y) if x=='End\_Data']

tbdata\_y=[] #取出数据

for i, v in enumerate(i\_start\_y): #因start\_data和End\_data成对出现

tbdata\_y.append(table\_data\_y[i\_start\_y[i]-36:i\_stop\_y[i]])

y\_item\_name = str(tbdata\_y[0][1])

x\_item\_name = str(tbdata\_x[0][1])

# print('---->y\_item\_name==>',y\_item\_name)

# print('---->x\_item\_name==>',x\_item\_name)

y\_usl = tbdata\_y[0][4]

y\_lsl = tbdata\_y[0][5]

x\_usl = tbdata\_x[0][4]

x\_lsl = tbdata\_x[0][5]

set\_bins = tbdata\_x[0][19]

path= '/tmp/CPK\_Log'#tbdata\_x[0][30]

start\_time\_first = tbdata\_x[0][21]

start\_time\_last = tbdata\_x[0][22]

new\_y\_lsl = tbdata\_y[0][7]

new\_y\_usl = tbdata\_y[0][8]

# new\_lsl,new\_usl = verify\_limit(new\_y\_lsl,new\_y\_usl)

limit\_apply\_y = tbdata\_y[0][9]

if limit\_apply\_y == 1:

y\_lsl = new\_y\_lsl

y\_usl = new\_y\_usl

new\_x\_lsl = tbdata\_x[0][7]

new\_x\_usl = tbdata\_x[0][8]

# new\_lsl,new\_usl = verify\_limit(new\_x\_lsl,new\_x\_usl)

limit\_apply\_x = tbdata\_x[0][9]

if limit\_apply\_x== 1:

x\_lsl = new\_x\_lsl

x\_usl = new\_x\_usl

# image\_name = item\_name.replace('/','\_')+".png"

filelogname = '/tmp/CPK\_Log/temp/.logcor.txt'

image\_name ='correlation.png'

pic\_path = path +'/temp/'

# if not os.path.exists(pic\_path):

# os.makedirs(pic\_path)

# os.system('mkdir '+pic\_path)

pic\_path = pic\_path + image\_name

# print('pic\_path--->',pic\_path)

if forXorY == 'setY':

select\_color\_by\_left = tbdata\_y[0][31] #判断是否点击了color by

select\_color\_by\_right = tbdata\_y[0][32] #判断是否点击了color by

else:

select\_color\_by\_left = tbdata\_x[0][31] #判断是否点击了color by

select\_color\_by\_right = tbdata\_x[0][32] #判断是否点击了color by

# select\_btn\_x = tbdata\_x[0][33] #select x button

# select\_btn\_y = tbdata\_x[0][34] #select y button

# print("----select x,y:",select\_btn\_x,select\_btn\_y)

print("--select color by correlation: ",select\_color\_by\_left,select\_color\_by\_right)

if select\_color\_by\_left == 0 and select\_color\_by\_right == 0: #没有点击color by

x\_tb\_value = tbdata\_x[0][37:]

y\_tb\_value = tbdata\_y[0][37:]

# print("=====x\_tb\_value:",x\_tb\_value)

# print("=====x\_tb\_value:",y\_tb\_value)

tb\_len\_x = len(x\_tb\_value)

tb\_len\_y = len(y\_tb\_value)

xValue=[]

yValue=[]

if tb\_len\_x>tb\_len\_y: #删除空元素 和长度不匹配的，按照做小的长度为基准

for i,v in enumerate(y\_tb\_value):

if x\_tb\_value[i]!='' and v!='':

xValue.append(x\_tb\_value[i])

yValue.append(v)

else:

for i,v in enumerate(x\_tb\_value):

if y\_tb\_value[i]!='' and v!='':

xValue.append(v)

yValue.append(y\_tb\_value[i])

if len(xValue)<1 and len(yValue)<1 and len(yValue) != len(xValue): # 判断确认一下长度，理论上经过上面的判断，都应该是一样的长度

print('xValue and yValue are not same length!Can not calculate pearson/generate correlation\_plot')

# redisClient.set('correlation\_png','xValue and yValue are not same')

with open(filelogname, 'w') as file\_object:

file\_object.write("FAIL,xValue and yValue are not same")

return 'xValue and yValue are not same length!Can not calculate pearson/generate correlation\_plot'

draw\_correlation(xValue,yValue,x\_item\_name,y\_item\_name,pic\_path,x\_lsl,x\_usl,y\_lsl,y\_usl,start\_time\_first,start\_time\_last)

else:

#[[],[],[]...],only value category lists

x\_tb\_val=[] #取出数据

for i, v in enumerate(i\_start\_x): #因start\_data和End\_data成对出现

x\_tb\_val.append(table\_data\_x[i\_start\_x[i]+1:i\_stop\_x[i]])

y\_tb\_val=[] #取出数据

for i, v in enumerate(i\_start\_y): #因start\_data和End\_data成对出现

y\_tb\_val.append(table\_data\_y[i\_start\_x[i]+1:i\_stop\_x[i]])

x\_tb\_len = len(x\_tb\_val) #存的二位数组x

y\_tb\_len = len(y\_tb\_val) #存的二位数组y

x\_category\_value = [] # 二位数组x

y\_category\_value = [] #二位数组y

if x\_tb\_len>y\_tb\_len:

for i,v in enumerate(y\_tb\_val):

x\_tb\_value = x\_tb\_val[i] # 取出一位数组值x

y\_tb\_value = y\_tb\_val[i] # 取出一位数组值y

tb\_len\_x = len(x\_tb\_value)

tb\_len\_y = len(y\_tb\_value)

x\_category\_tmp = [] #一维数组值x

y\_category\_tmp = [] #一维数组值y

if tb\_len\_x>tb\_len\_y: #删除空元素 和长度不匹配的，按照做小的长度为基准

for i,v in enumerate(y\_tb\_value):

if x\_tb\_value[i]!='' and v!='':

x\_category\_tmp.append(x\_tb\_value[i])

y\_category\_tmp.append(v)

else:

for i,v in enumerate(x\_tb\_value):

if y\_tb\_value[i]!='' and v!='':

y\_category\_tmp.append(v)

y\_category\_tmp.append(y\_tb\_value[i])

x\_category\_value.append(x\_category\_tmp)

y\_category\_value.append(x\_category\_tmp)

else:

for i,v in enumerate(x\_tb\_val):

x\_tb\_value = x\_tb\_val[i] # 取出一位数组值x

y\_tb\_value = y\_tb\_val[i] # 取出一位数组值y

tb\_len\_x = len(x\_tb\_value)

tb\_len\_y = len(y\_tb\_value)

x\_category\_tmp = [] #一维数组值x

y\_category\_tmp = [] #一维数组值y

if tb\_len\_x>tb\_len\_y: #删除空元素 和长度不匹配的，按照做小的长度为基准

for i,v in enumerate(y\_tb\_value):

if x\_tb\_value[i]!='' and v!='':

x\_category\_tmp.append(x\_tb\_value[i])

y\_category\_tmp.append(v)

else:

for i,v in enumerate(x\_tb\_value):

if y\_tb\_value[i]!='' and v!='':

x\_category\_tmp.append(v)

y\_category\_tmp.append(y\_tb\_value[i])

x\_category\_value.append(x\_category\_tmp)

y\_category\_value.append(y\_category\_tmp)

if len(x\_category\_value)<1 and len(y\_category\_value)<1 and len(x\_category\_value)!=len(y\_category\_value):

print("no data, can not generate plot")

with open(filelogname, 'w') as file\_object:

file\_object.write("FAIL,x\_category\_value and y\_category\_value are not same")

return 'X table or Y table no data, can not generate plot'

xValue = ([i for item in x\_category\_value for i in item]) # 二维列表拼接成一维列表

yValue = ([i for item in y\_category\_value for i in item]) # 二维列表拼接成一维列表

draw\_correlation\_by\_color(xValue,yValue,x\_category\_value,y\_category\_value,x\_item\_name,y\_item\_name,pic\_path,x\_lsl,x\_usl,y\_lsl,y\_usl,start\_time\_first,start\_time\_last)

info = 'correlation plot draw finished!'

print(info)

with open(filelogname, 'w') as file\_object:

file\_object.write("PASS,correlation plot draw finished")

# redisClient.set('correlation\_png','PASS,correlation plot draw finished!')

return info

except Exception as e:

with open(filelogname, 'w') as file\_object:

file\_object.write("FAIL,error correlation\_plot function")

print('error correlation\_plot function:',e)

def checkItemName(item\_name,item\_length):

if len(item\_name) > item\_length:

if item\_name[item\_length] == '\_' or item\_name[item\_length] == ' ':

item\_name = item\_name[0:item\_length+1] + '\n' + item\_name[item\_length+1:]

else:

item\_name1 = item\_name[0:item\_length]

item\_name\_tmp = item\_name1[::-1]

x1=item\_name\_tmp.find('\_')

x2=item\_name\_tmp.find(' ')

x\_len = 0

if x1 == -1 and x2 == -1:

x\_len = 0

elif x1 == -1:

x\_len = x2

elif x2 ==-1:

x\_len = x1

elif x1>x2:

x\_len = x2

else:

x\_len = x1

x\_len = len(item\_name1) - x\_len

item\_name = item\_name[0:x\_len] + '\n' + item\_name[x\_len:]

return item\_name

def draw\_correlation(xValue,yValue,x\_item\_name,y\_item\_name,pic\_save\_path,x\_lsl,x\_usl,y\_lsl,y\_usl,start\_time\_first,start\_time\_last):

pearson,spearman = correlation\_coefficient\_calc(xValue, yValue, x\_item\_name, y\_item\_name)

plt.ion() # 开启interactive mode

# font = FontProperties(fname=r"/Library/Fonts/Songti.ttc", size=12)

fig, axes = plt.subplots(1, 0, figsize=(8, 6), facecolor='#ccddef')

plt.axes([0.15, 0.15, 0.75, 0.75]) # [左, 下, 宽, 高] 规定的矩形区域 （全部是0~1之间的数，表示比例）

plt.title('Correlation pearson coefficient = ' + str(pearson),size=13)

# if len(x\_item\_name) > 60:

# x\_item\_name = x\_item\_name[0:60] + '\n' + x\_item\_name[60:]

x\_item\_name = checkItemName(x\_item\_name,60)

# if len(y\_item\_name) > 55:

# y\_item\_name = y\_item\_name[0:55] + '\n' + y\_item\_name[55:]

y\_item\_name = checkItemName(y\_item\_name,55)

plt.xlabel(x\_item\_name,size=12)

plt.ylabel(y\_item\_name,size=12)

x\_min\_num, x\_max\_num = min(xValue), max(xValue)

# print('xvalue min,max:', x\_min\_num, x\_max\_num)

x\_ticks\_l, x\_ticks\_h = get\_ticks(x\_min\_num, x\_max\_num, x\_lsl, x\_usl)

if x\_ticks\_l == x\_ticks\_h and x\_ticks\_l !=0:

x\_ticks\_l = x\_ticks\_l - round((x\_ticks\_l/5.0),2)

x\_ticks\_h = x\_ticks\_h + round((x\_ticks\_h/5.0),2)

elif x\_ticks\_l == x\_ticks\_h and x\_ticks\_l ==0:

x\_ticks\_l = - 3

x\_ticks\_h = 3

# print('x\_ticks\_l x\_ticks\_h:', x\_ticks\_l, x\_ticks\_h)

plt.xlim(x\_ticks\_l, x\_ticks\_h)

y\_min\_num, y\_max\_num = min(yValue), max(yValue)

# print('yvalue min,max:', y\_min\_num, y\_max\_num)

y\_ticks\_l, y\_ticks\_h = get\_ticks(y\_min\_num, y\_max\_num, y\_lsl, y\_usl)

if y\_ticks\_l == y\_ticks\_h and y\_ticks\_l !=0:

y\_ticks\_l = y\_ticks\_l - round((y\_ticks\_l/5.0),2)

y\_ticks\_h = y\_ticks\_h + round((y\_ticks\_h/5.0),2)

elif y\_ticks\_l == y\_ticks\_h and y\_ticks\_l ==0:

y\_ticks\_l = - 3

y\_ticks\_h = 3

# print('y\_ticks\_l y\_ticks\_h:', y\_ticks\_l, y\_ticks\_h)

plt.ylim((y\_ticks\_l, y\_ticks\_h)) # 设置y轴scopex

ax=plt.gca()

ax.spines['bottom'].set\_linewidth(1.0)

ax.spines['left'].set\_linewidth(1.0)

ax.spines['right'].set\_linewidth(1.0)

ax.spines['top'].set\_linewidth(1.0)

# if x\_lsl != 'NA' and x\_usl != 'NA':

# plt.plot([x\_lsl, x\_lsl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x lower limit线，

# plt.plot([x\_usl, x\_usl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x upper limit线，

# if y\_lsl != 'NA' and y\_usl != 'NA':

# plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_lsl, y\_lsl, ], 'k--', linewidth=1.0, color='red') # y lower limit线，

# plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_usl, y\_usl, ], 'k--', linewidth=1.0, color='red') # y upper limit线，

# print('=====x\_lsl x\_usl:',x\_lsl,x\_usl)

# print('=====y\_lsl y\_usl',y\_lsl,y\_usl)

# print('=====x\_ticks\_l x\_ticks\_h',x\_ticks\_l,x\_ticks\_h)

# print('=====y\_ticks\_l y\_ticks\_h',y\_ticks\_l,y\_ticks\_h)

if x\_lsl != 'NA' and x\_lsl != '':

plt.plot([x\_lsl, x\_lsl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x lower limit线，

# plt.plot([x\_usl, x\_usl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x upper limit线，

if x\_usl != 'NA' and x\_usl != '':

# plt.plot([x\_lsl, x\_lsl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x lower limit线，

plt.plot([x\_usl, x\_usl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x upper limit线，

if y\_lsl != 'NA' and y\_lsl != '':

plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_lsl, y\_lsl, ], 'k--', linewidth=1.0, color='red') # y lower limit线，

# plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_usl, y\_usl, ], 'k--', linewidth=1.0, color='red') # y upper limit线，

if y\_usl != 'NA' and y\_usl != '':

# plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_lsl, y\_lsl, ], 'k--', linewidth=1.0, color='red') # y lower limit线，

plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_usl, y\_usl, ], 'k--', linewidth=1.0, color='red') # y upper limit线，

# plt.scatter(x, y, s, c, marker)

# x: x轴坐标

# y：y轴坐标

# s：点的大小/粗细 标量或array\_like 默认是 rcParams['lines.markersize'] \*\* 2

# c: 点的颜色

# marker: 标记的样式 默认是 'o'

# plt.legend()

plt.rcParams['savefig.dpi'] = 250 # 图片像素

plt.rcParams['figure.dpi'] = 150 # 分辨率

plt.scatter(xValue, yValue, s=45,linewidth =1, c="blue", marker='+')

plt.grid(linestyle=':', c='gray', linewidth=1, alpha=0.6) # 生成网格

plt.savefig(pic\_save\_path, dpi=250)

plt.draw()

# plt.show()

plt.close()

plt.ioff()

def draw\_correlation\_by\_color(xValue,yValue,x\_category\_value,y\_category\_value,x\_item\_name,y\_item\_name,pic\_save\_path,x\_lsl,x\_usl,y\_lsl,y\_usl,start\_time\_first,start\_time\_last):

# print("===xValue",xValue)

# print("===yValue",yValue)

# print("===x\_category\_value",x\_category\_value)

# print("===y\_category\_value",y\_category\_value)

pearson, spearman = correlation\_coefficient\_calc(xValue, yValue, x\_item\_name, y\_item\_name)

plt.ion() # 开启interactive mode

# font = FontProperties(fname=r"/Library/Fonts/Songti.ttc", size=12)

fig, axes = plt.subplots(1, 0, figsize=(8, 6), facecolor='#ccddef')

plt.axes([0.15, 0.15, 0.75, 0.75]) # [左, 下, 宽, 高] 规定的矩形区域 （全部是0~1之间的数，表示比例）

# plt.title('Correlation pearson coefficient = ' + str(pearson)+'\n'+str(start\_time\_first)+' -- '+str(start\_time\_last),size=12)

plt.title('Correlation pearson coefficient = ' + str(pearson),size=13)

# if len(x\_item\_name) > 60:

# x\_item\_name = x\_item\_name[0:60] + '\n' + x\_item\_name[60:]

x\_item\_name = checkItemName(x\_item\_name,60)

# if len(y\_item\_name) > 55:

# y\_item\_name = y\_item\_name[0:55] + '\n' + y\_item\_name[55:]

y\_item\_name = checkItemName(y\_item\_name,55)

plt.xlabel(x\_item\_name,size=12)

plt.ylabel(y\_item\_name,size=12)

if len(xValue) == 0:

x\_min\_num, x\_max\_num = 0,0

else:

x\_min\_num, x\_max\_num = min(xValue), max(xValue)

print('xvalue min,max:', x\_min\_num, x\_max\_num)

x\_ticks\_l, x\_ticks\_h = get\_ticks(x\_min\_num, x\_max\_num, x\_lsl, x\_usl)

if x\_ticks\_l == x\_ticks\_h and x\_ticks\_l !=0:

x\_ticks\_l = x\_ticks\_l - round((x\_ticks\_l/5.0),2)

x\_ticks\_h = x\_ticks\_h + round((x\_ticks\_h/5.0),2)

elif x\_ticks\_l == x\_ticks\_h and x\_ticks\_l ==0:

x\_ticks\_l = - 3

x\_ticks\_h = 3

# print('x\_ticks\_l x\_ticks\_h:', x\_ticks\_l, x\_ticks\_h)

plt.xlim(x\_ticks\_l, x\_ticks\_h)

if len(yValue)==0:

y\_min\_num, y\_max\_num = 0,0

else:

y\_min\_num, y\_max\_num = min(yValue), max(yValue)

y\_ticks\_l, y\_ticks\_h = get\_ticks(y\_min\_num, y\_max\_num, y\_lsl, y\_usl)

if y\_ticks\_l == y\_ticks\_h and y\_ticks\_l !=0:

y\_ticks\_l = y\_ticks\_l - round((y\_ticks\_l/5.0),2)

y\_ticks\_h = y\_ticks\_h + round((y\_ticks\_h/5.0),2)

elif y\_ticks\_l == y\_ticks\_h and y\_ticks\_l ==0:

y\_ticks\_l = - 3

y\_ticks\_h = 3

# print('y\_ticks\_l y\_ticks\_h:', y\_ticks\_l, y\_ticks\_h)

plt.ylim((y\_ticks\_l, y\_ticks\_h)) # 设置y轴scopex

ax=plt.gca()

ax.spines['bottom'].set\_linewidth(1.0)

ax.spines['left'].set\_linewidth(1.0)

ax.spines['right'].set\_linewidth(1.0)

ax.spines['top'].set\_linewidth(1.0)

if x\_lsl != 'NA' and x\_lsl != '':

plt.plot([x\_lsl, x\_lsl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x lower limit线，

# plt.plot([x\_usl, x\_usl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x upper limit线，

if x\_usl != 'NA' and x\_usl != '':

# plt.plot([x\_lsl, x\_lsl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x lower limit线，

plt.plot([x\_usl, x\_usl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x upper limit线，

if y\_lsl != 'NA' and y\_lsl != 'NA':

plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_lsl, y\_lsl, ], 'k--', linewidth=1.0, color='red') # y lower limit线，

# plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_usl, y\_usl, ], 'k--', linewidth=1.0, color='red') # y upper limit线，

if y\_usl != 'NA' and y\_usl != 'NA':

# plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_lsl, y\_lsl, ], 'k--', linewidth=1.0, color='red') # y lower limit线，

plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_usl, y\_usl, ], 'k--', linewidth=1.0, color='red') # y upper limit线，

# if x\_lsl != 'NA' and x\_usl != 'NA':

# plt.plot([x\_lsl, x\_lsl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x lower limit线，

# plt.plot([x\_usl, x\_usl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red') # x upper limit线，

# if y\_lsl != 'NA' and y\_usl != 'NA':

# plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_lsl, y\_lsl, ], 'k--', linewidth=1.0, color='red') # y lower limit线，

# plt.plot([x\_ticks\_l, x\_ticks\_h, ], [y\_usl, y\_usl, ], 'k--', linewidth=1.0, color='red') # y upper limit线，

# plt.scatter(x, y, s, c, marker)

# x: x轴坐标

# y：y轴坐标

# s：点的大小/粗细 标量或array\_like 默认是 rcParams['lines.markersize'] \*\* 2

# c: 点的颜色

# marker: 标记的样式 默认是 'o'

# plt.legend()

plt.rcParams['savefig.dpi'] = 250 # 图片像素

plt.rcParams['figure.dpi'] = 150 # 分辨率

# print('x\_category\_value,y\_category\_value length:---->',len(x\_category\_value),len(y\_category\_value),x\_category\_value[0], y\_category\_value[0])

color\_l = ['#0000FF','#FF0000','#008000','#00FFFF','#9400D3','#8B008B','#B8860B','#FFA500','#A9A9A9','#FFFF00']

for i,v in enumerate(x\_category\_value): # 因为传过来的参数x\_category\_value 和y\_category\_value 经过判断的，长度都是一样的，任意选择一个

# print('i,v',i,v)

set\_color = color\_l[i%10]

plt.scatter(x\_category\_value[i], y\_category\_value[i], s=45,linewidth =1, c=set\_color, marker='+')

# for i in range(0,len(x\_category\_value),1):

# print(x\_category\_value[i][1],y\_category\_value[i][1])

# set\_color = x\_category\_value[i][1]

# plt.scatter(x\_category\_value[i][5:], y\_category\_value[i][5:], s=40,linewidth =0.6, c=set\_color, marker='+')

plt.grid(linestyle=':', c='gray', linewidth=1, alpha=0.6) # 生成网格

plt.savefig(pic\_save\_path, dpi=250)

plt.draw()

# plt.show()

plt.close()

plt.ioff()

def get\_ticks(min\_num,max\_num,lsl,usl):

#for correlation plot

print('min\_num,max\_num,lsl,usl=====>',min\_num,max\_num,lsl,usl)

# return

ticks\_l = 0

ticks\_h = 0

if lsl =='NA' or lsl =='' :

ticks\_l = min\_num

if usl =='NA' or usl =='':

ticks\_h = max\_num

if lsl !='NA' and lsl !='' and usl !='NA' and usl !='':

if min\_num < lsl and max\_num < lsl:

ticks\_l = min\_num

ticks\_h = usl

elif min\_num < lsl and max\_num > lsl and max\_num <= usl:

ticks\_l = min\_num

ticks\_h = usl

elif min\_num < lsl and max\_num > usl:

ticks\_l = min\_num

ticks\_h = max\_num

elif lsl <= min\_num and min\_num <= usl and max\_num <= usl:

ticks\_l = lsl

ticks\_h = usl

elif lsl <= min\_num and min\_num <= usl and max\_num > usl:

ticks\_l = lsl

ticks\_h = max\_num

elif min\_num > usl:

ticks\_l = lsl

ticks\_h = max\_num

if (lsl !='NA' and lsl !='') and (usl =='NA' or usl ==''):

if min\_num < lsl:

ticks\_l = min\_num

else:

ticks\_l = lsl

if (lsl =='NA' or lsl =='') and (usl !='NA' and usl !=''):

if max\_num<usl:

ticks\_h = usl

else:

ticks\_h = max\_num

# print('ticks\_l,ticks\_h:',ticks\_l,ticks\_h)

range\_val = get\_limit\_range(ticks\_l,ticks\_h)

# print('ticks\_range:',range,round((ticks\_l-range/5),2),round((ticks\_h+range/5),2))

ticks\_l,ticks\_h = round((ticks\_l-range\_val/5),2),round((ticks\_h+range\_val/5),2)

return ticks\_l,ticks\_h

def get\_limit\_range(lsl,usl):

# print('lsl,usl----->', lsl, usl)

range\_val = 0

if lsl < 0 and usl <= 0:

range\_val = abs(lsl) - abs(usl)

elif lsl < 0 and usl >= 0:

range\_val = abs(lsl) + usl

elif lsl >= 0 and (usl > 0):

range\_val = usl - lsl

else:

print('get\_limit\_range 00000')

range\_val = round(range\_val, 5)

# print('range in get\_limit\_range----->', range)

return range\_val

def correlation\_coefficient\_calc(data\_l1,data\_l2,item\_name1,item\_name2):

'''

item\_name1 : string

item\_name2 : string

data\_l2 :[]

data\_l1 :[]

'''

if item\_name1 == item\_name2:# can not be the same name.

item\_name2 = item\_name2+'\_2'

data = pd.DataFrame({item\_name1:data\_l1,

item\_name2:data\_l2})

# print('correlation data:',data)

pearson = data.corr(method='pearson').values[0].tolist()[1]

spearman = data.corr(method='spearman').values[0].tolist()[1]

pearson = round(pearson,5)

spearman = round(spearman,5)

# print('pearson:',pearson)

# print('spearman:',spearman)

return pearson,spearman

def verify\_limit(lsl,usl):

if lsl != None:

lsl.replace(' ','')

if usl != None:

usl.replace(' ','')

try:

lsl = float(eval(lsl))

except:

lsl = None

try:

usl = float(eval(usl))

except:

usl = None

# if type(lsl) == float and type(usl) == float:

# print('after verify lsl===>',lsl)

# print('after verify usl===>',usl)

return lsl,usl

def calculate\_value(message):

print("this function is calculate\_value......")

val = r.get(message) # 注意 等到的都是字符串

if val:

val = float(val)\*200 # 数学运算

val = str(val).encode('utf-8')

return val

else:

return b'0'

# def readSelectItem():

# with open('all\_csv\_path', 'r') as f:

# reader = csv.reader(f)

# i = 1

# for row in reader:

def run(n):

global table\_data\_correlation\_y

global select\_x\_flag

global xy\_reverse\_flag

while True:

try:

print("wait for correlation ...")

zmqMsg = socket.recv()

socket.send(b'correlation.png')

if len(zmqMsg)>0:

key = zmqMsg.decode('utf-8')

print("-->message from correltion client:", key)

table\_a = get\_redis\_data(key)

select\_name\_xy = table\_a[26]

table\_b = get\_redis\_data(select\_name\_xy)

msg =key.split("$$")

if len(msg)>1:

if msg[1]=='1':

xy\_reverse\_flag = 0 #select X 设置

if msg[1]=='10':

xy\_reverse\_flag = 1 #select Y 设置

if len(table\_a)>0 and len(table\_b)>0:

if xy\_reverse\_flag == 0:

correlation\_plot(table\_b,table\_a,'setY')

else:

correlation\_plot(table\_a,table\_b,'setX')

else:

print("---get data error")

# socket.send(ret.decode('utf-8').encode('ascii'))

else:

time.sleep(0.05)

except Exception as e:

print('error:',e)

if \_\_name\_\_ == '\_\_main\_\_':

run(0)

#! /usr/bin/env python3

# --\*-- coding: utf-8 ---\*---

import sys,os,time,math,re

import time

import threading

BASE\_DIR=os.path.dirname(os.path.abspath(\_\_file\_\_))

sys.path.insert(0,BASE\_DIR+'/site-packages/')

try:

import csv

except Exception as e:

print('e---->',e)

try:

import matplotlib

matplotlib.use("Agg")

import matplotlib.pyplot as plt

except Exception as e:

print('e---->',e)

try:

import matplotlib.colors as colors

except Exception as e:

print('e---->',e)

try:

from matplotlib.font\_manager import FontProperties

except Exception as e:

print('e---->',e)

try:

import numpy as np

except Exception as e:

print('e--->',e)

try:

import pandas as pd

except Exception as e:

print('e--->',e)

try:

import openpyxl

except Exception as e:

print('import openpyxl error:',e)

try:

import xlsxwriter

except Exception as e:

print('import xlsxwriter error:',e)

try:

import zmq

except Exception as e:

print('import zmq error:',e)

try:

import redis

except Exception as e:

print('import redis error:',e)

print(sys.getdefaultencoding())

redisClient = redis.Redis(host='localhost', port=6379, db=0)

context = zmq.Context()

socket = context.socket(zmq.REP)

socket.setsockopt(zmq.LINGER,0)

socket.bind("tcp://127.0.0.1:3170")

filelogname = '/tmp/CPK\_Log/temp/.logscatter.txt'

def get\_redis\_data(zmqMsg):

tb = redisClient.get(zmqMsg)

tb\_data=[]

if tb:

tb=tb.decode('utf-8')

tb=tb.split("\n")

tb=(tb[1:-1]) #去掉数据库首尾元素

for i in tb:

k=re.sub('\"','',i) #去掉数据库引号

h=re.sub(',','',k) #去掉数据库逗号

m=h.strip() #去掉数据库首尾空白

if is\_number(m):

tb\_data.append(eval(m)) #去掉数字的引号

else:

tb\_data.append(m)

else:

tb\_data.append('')

return tb\_data

def is\_number(s):

try:

float(s)

return True

except ValueError:

pass

try:

import unicodedata

unicodedata.numeric(s)

return True

except (TypeError, ValueError):

pass

return False

def plot\_display\_y\_name(zmqSelectItems,filter1,filter2):

scatter\_item = []

for v in zmqSelectItems:

filterItems = v.split('&')

i = 0

for z in filterItems:

if i%2 ==0:

if filter1[0] == 'Station ID':

tmpFilter1 = filterItems[i].split('-')

tmpFilter1\_len = len(tmpFilter1)

if tmpFilter1\_len>2:

tmpFilter1\_str = []

for x in range(2,tmpFilter1\_len):

if x==2:

tmpFilter1\_str.append(tmpFilter1[x])

else:

tmpFilter1\_str.append('-')

tmpFilter1\_str.append(tmpFilter1[x])

filter\_str1=''.join(tmpFilter1\_str)

scatter\_item.append(filter\_str1)

elif tmpFilter1\_len>1:

tmpFilter1\_str = []

for x in range(1,tmpFilter1\_len):

if x==1:

tmpFilter1\_str.append(tmpFilter1[x])

else:

tmpFilter1\_str.append('-')

tmpFilter1\_str.append(tmpFilter1[x])

filter\_str1=''.join(tmpFilter1\_str)

scatter\_item.append(filter\_str1)

else:

scatter\_item.append(tmpFilter1[0])

else:

scatter\_item.append(filterItems[0])

else:

if filter2[0] == 'Station ID':

tmpFilter2 = filterItems[i].split('-')

tmpFilter2\_len = len(tmpFilter2)

if tmpFilter2\_len>2:

tmpFilter2\_str = []

for x in range(2,tmpFilter2\_len):

if x==2:

tmpFilter2\_str.append(tmpFilter2[x])

else:

tmpFilter2\_str.append('-')

tmpFilter2\_str.append(tmpFilter2[x])

filter\_str2=''.join(tmpFilter2\_str)

scatter\_item.append(filter\_str2)

elif tmpFilter2\_len>1:

tmpFilter2\_str = []

for x in range(1,tmpFilter2\_len):

if x==1:

tmpFilter2\_str.append(tmpFilter2[x])

else:

tmpFilter2\_str.append('-')

tmpFilter2\_str.append(tmpFilter1[x])

filter\_str2=''.join(tmpFilter2\_str)

scatter\_item.append(filter\_str2)

else:

scatter\_item.append(tmpFilter2[0])

else:

scatter\_item.append(filterItems[1])

i=i+1

plot\_scatter\_item = []

tmp = []

x\_flg = False

for x in range(0,len(scatter\_item)):

if x%2 ==0:

tmp = []

x\_flg = False

if not scatter\_item[x] == 'Off':

x\_flg = False

tmp.append(scatter\_item[x])

else:

x\_flg = True

else:

if not scatter\_item[x] == 'Off':

if x\_flg:

tmp.append(scatter\_item[x])

else:

tmp.append('&'+scatter\_item[x])

plot\_scatter\_item.append(''.join(tmp))

return plot\_scatter\_item

def scatter\_plot(table\_data, zmqMsgName):

try:

global filelogname

info = ''

i\_start = [i for i,x in enumerate(table\_data) if x=='Start\_Data']

i\_stop = [i for i,x in enumerate(table\_data) if x=='End\_Data']

tbdata=[]

for i, v in enumerate(i\_start):

tbdata.append(table\_data[i\_start[i]-36:i\_stop[i]])

x\_item\_name=tbdata[0][1]

x\_usl = tbdata[0][4]

x\_lsl = tbdata[0][5]

new\_x\_lsl = tbdata[0][7]

new\_x\_usl = tbdata[0][8]

limit\_apply\_x = tbdata[0][9]

if limit\_apply\_x== 1:

x\_lsl = new\_x\_lsl

x\_usl = new\_x\_usl

set\_bins = tbdata[0][19]

start\_time\_first = tbdata[0][21]

start\_time\_last = tbdata[0][22]

cpkLTHLD = tbdata[0][24]

cpkHTHLD = tbdata[0][25]

select\_color\_by\_left = tbdata[0][31] #是否点击了color by left

select\_color\_by\_right = tbdata[0][32] #是否点击了color by right

range\_set\_lsl = tbdata[0][27] #get range lsl

range\_set\_usl = tbdata[0][28] #get range usl

zoom\_type = tbdata[0][18]

path= '/tmp/CPK\_Log' #tbdata[0][30]

filelogname = path + '/temp/.logscatter.txt'

image\_name ='scatter.png'

pic\_path = path +'/temp/'

pic\_path = pic\_path + image\_name

# print('select\_color\_by:',select\_color\_by\_left,select\_color\_by\_right)

if select\_color\_by\_left == 0 and select\_color\_by\_right == 0: #没有点击color by

print('you did not choose filter 1 or filter 2,will not generate scatter plot!!!')

else:

filter1 = ['']

filter2 = ['']

if select\_color\_by\_left>0:

filter1 = get\_redis\_data('select\_filter\_by\_1')

if select\_color\_by\_right>0:

filter2 = get\_redis\_data('select\_filter\_by\_2')

print('filter1,filter2:',filter1,filter2,zmqMsgName)

zmqSelectItems\_tmp = zmqMsgName.split('##')

zmqSelectItems = plot\_display\_y\_name(zmqSelectItems\_tmp,filter1,filter2)

tb\_val=[] #取出数据

for i, v in enumerate(i\_start): #因start\_data和End\_data成对出现

tb\_val.append(table\_data[i\_start[i]+1:i\_stop[i]])

tb\_data\_x=[] # 二位数组

tb\_data\_y=[] # 二位数组

yy = len(tb\_val)

y\_max = yy

y\_min = 1

for i,v in enumerate(tb\_val):

tmp = [i for i in v if i !='']

tb\_data\_x.append(tmp)

tmp\_y = [yy for i in v if i !='']

tb\_data\_y.append(tmp\_y)

yy -=1

# print('---->>>>tb\_data\_y',tb\_data\_y)

if len(tb\_data\_x)<1:

print("no data, can not generate plot")

with open(filelogname, 'w') as file\_object:

file\_object.write("FAIL,no data to generate the scatter")

return 'no date to generate plot'

xValue = ([i for item in tb\_data\_x for i in item]) # 二维列表拼接成一维列表

yValue = ([i for item in tb\_data\_y for i in item]) # 二维列表拼接成一维列表

y\_item\_name = ''

y\_lsl = ''

y\_usl = ''

start\_time\_first = ''

start\_time\_last = ''

draw\_correlation\_by\_color(xValue,yValue,tb\_data\_x,tb\_data\_y,x\_item\_name,y\_item\_name,pic\_path,

x\_lsl,x\_usl,y\_lsl,y\_usl,start\_time\_first,start\_time\_last,

zmqSelectItems,y\_max,y\_min)

info = 'scatter plot draw finished!'

print(info)

with open(filelogname, 'w') as file\_object:

file\_object.write("PASS,correlation plot draw finished")

return info

except Exception as e:

with open(filelogname, 'w') as file\_object:

file\_object.write("FAIL,error correlation\_plot function")

print('error correlation\_plot function:',e)

def draw\_correlation\_by\_color(xValue,yValue,x\_category\_value,y\_category\_value,x\_item\_name,y\_item\_name,pic\_save\_path,

x\_lsl,x\_usl,y\_lsl,y\_usl,start\_time\_first,start\_time\_last,y\_item,y\_max,y\_min):

pearson, spearman = correlation\_coefficient\_calc(xValue, yValue, x\_item\_name, y\_item\_name)

plt.ion() # 开启interactive mode

# font = FontProperties(fname=r"/Library/Fonts/Songti.ttc", size=12)

fig, axes = plt.subplots(1, 0, figsize=(13, 7), facecolor='#ccddef')

plt.axes([0.22, 0.12, 0.75, 0.75]) # [左, 下, 宽, 高] 规定的矩形区域 （全部是0~1之间的数，表示比例）

# plt.title('Correlation pearson coefficient = ' + str(pearson)+'\n'+str(start\_time\_first)+' -- '+str(start\_time\_last),size=12)

#plt.title('Correlation pearson coefficient = ' + str(pearson),size=13)

plt.title('Scatter plot',size=20)

if len(x\_item\_name) > 70:

x\_item\_name = x\_item\_name[0:70] + '\n' + x\_item\_name[70:]

if len(y\_item\_name) > 45:

y\_item\_name = y\_item\_name[0:45] + '\n' + y\_item\_name[45:]

plt.xlabel(x\_item\_name,size=14)

#plt.ylabel(y\_item\_name,size=12)

if len(xValue)==0:

x\_min\_num, x\_max\_num = 0,0

else:

x\_min\_num, x\_max\_num = min(xValue), max(xValue)

x\_ticks\_l, x\_ticks\_h = get\_ticks(x\_min\_num, x\_max\_num, x\_lsl, x\_usl)

if x\_ticks\_l == x\_ticks\_h and x\_ticks\_l !=0:

x\_ticks\_l = x\_ticks\_l - round((x\_ticks\_l/5.0),2)

x\_ticks\_h = x\_ticks\_h + round((x\_ticks\_h/5.0),2)

elif x\_ticks\_l == x\_ticks\_h and x\_ticks\_l ==0:

x\_ticks\_l = - 3

x\_ticks\_h = 3

plt.xlim(x\_ticks\_l, x\_ticks\_h)

y\_min\_num, y\_max\_num = y\_min,y\_max#min(yValue), max(yValue)

y\_ticks\_l, y\_ticks\_h = get\_y\_ticks(y\_min\_num, y\_max\_num)

if y\_ticks\_l == y\_ticks\_h and y\_ticks\_l !=0:

y\_ticks\_l = y\_ticks\_l - round((y\_ticks\_l/5.0),2)

y\_ticks\_h = y\_ticks\_h + round((y\_ticks\_h/5.0),2)

elif y\_ticks\_l == y\_ticks\_h and y\_ticks\_l ==0:

y\_ticks\_l = - 3

y\_ticks\_h = 3

plt.ylim(y\_ticks\_l, y\_ticks\_h)

plt.tick\_params(labelsize=12)

y\_name = []

for i in range(len(y\_category\_value),0,-1):

y\_name.append(i)

print(y\_name)

plt.yticks(y\_name,y\_item,fontsize=12)

plt.ylim(y\_ticks\_l, y\_ticks\_h)

ax=plt.gca()

ax.spines['bottom'].set\_linewidth(1.0)

ax.spines['left'].set\_linewidth(1.0)

ax.spines['right'].set\_linewidth(1.0)

ax.spines['top'].set\_linewidth(1.0)

if x\_lsl != 'NA' and x\_lsl != '':

plt.text(x\_lsl, len(y\_item) / 3, ' LSL\n' + ' ' + str(x\_lsl), fontdict={'size': 10, 'color': 'r'})

plt.plot([x\_lsl, x\_lsl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red')

if x\_usl != 'NA' and x\_usl != '':

plt.text(x\_usl, len(y\_item) / 2, ' USL\n' + ' ' + str(x\_usl), fontdict={'size': 10, 'color': 'r'})

plt.plot([(x\_lsl+x\_usl)/2, (x\_lsl+x\_usl)/2, ], [0, len(y\_item), ], 'k--', linewidth=1.0, color='green')

plt.text((x\_lsl+x\_usl)/2, 0.02, ' Center\n'+str(round((x\_lsl+x\_usl)/2,4)), fontdict={'size': 10, 'color': 'green'})

plt.plot([x\_usl, x\_usl, ], [y\_ticks\_l, y\_ticks\_h, ], 'k--', linewidth=1.0, color='red')

plt.rcParams['savefig.dpi'] = 250 # 图片像素

plt.rcParams['figure.dpi'] = 150 # 分辨率

color\_l = ['#0000FF','#FF0000','#008000','#00FFFF','#9400D3','#8B008B','#B8860B','#FFA500','#A9A9A9','#FFFF00']

for i,v in enumerate(x\_category\_value):

set\_color = color\_l[i%10]

plt.scatter(x\_category\_value[i], y\_category\_value[i], s=200,linewidth =1, c=set\_color, marker='+')

if len(x\_category\_value[i]) > 0:

plt.text(round((min(x\_category\_value[i])+max(x\_category\_value[i]))/2,2), y\_category\_value[i][0]+0.2,str(len(x\_category\_value[i])), fontdict={'size': 10, 'color': 'black'})

plt.grid(linestyle=':', c='gray', linewidth=1.0, alpha=0.8) # 生成网格

plt.savefig(pic\_save\_path, dpi=250)

plt.draw()

# plt.show()

plt.close()

plt.ioff()

def get\_y\_ticks(min\_val,max\_val):

ticks\_l = min\_val -1

ticks\_h = max\_val +1

return ticks\_l, ticks\_h

def get\_ticks(min\_num,max\_num,lsl,usl):

ticks\_l = 0

ticks\_h = 0

if lsl =='NA' or lsl =='' :

ticks\_l = min\_num

if usl =='NA' or usl =='':

ticks\_h = max\_num

if lsl !='NA' and lsl !='' and usl !='NA' and usl !='':

if min\_num < lsl and max\_num < lsl:

ticks\_l = min\_num

ticks\_h = usl

elif min\_num < lsl and max\_num > lsl and max\_num <= usl:

ticks\_l = min\_num

ticks\_h = usl

elif min\_num < lsl and max\_num > usl:

ticks\_l = min\_num

ticks\_h = max\_num

elif lsl <= min\_num and min\_num <= usl and max\_num <= usl:

ticks\_l = lsl

ticks\_h = usl

elif lsl <= min\_num and min\_num <= usl and max\_num > usl:

ticks\_l = lsl

ticks\_h = max\_num

elif min\_num > usl:

ticks\_l = lsl

ticks\_h = max\_num

if (lsl !='NA' and lsl !='') and (usl =='NA' or usl ==''):

if min\_num < lsl:

ticks\_l = min\_num

else:

ticks\_l = lsl

if (lsl =='NA' or lsl =='') and (usl !='NA' and usl !=''):

if max\_num<usl:

ticks\_h = usl

else:

ticks\_h = max\_num

range\_val = get\_limit\_range(ticks\_l,ticks\_h)

ticks\_l,ticks\_h = round((ticks\_l-range\_val/5),2),round((ticks\_h+range\_val/5),2)

return ticks\_l,ticks\_h

def get\_limit\_range(lsl,usl):

range\_val = 0

if lsl < 0 and usl <= 0:

range\_val = abs(lsl) - abs(usl)

elif lsl < 0 and usl >= 0:

range\_val = abs(lsl) + usl

elif lsl >= 0 and (usl > 0):

range\_val = usl - lsl

else:

print('get\_limit\_range 00000')

range\_val = round(range\_val, 5)

return range\_val

def correlation\_coefficient\_calc(data\_l1,data\_l2,item\_name1,item\_name2):

'''

item\_name1 : string

item\_name2 : string

data\_l2 :[]

data\_l1 :[]

'''

if item\_name1 == item\_name2:# can not be the same name.

item\_name2 = item\_name2+'\_2'

data = pd.DataFrame({item\_name1:data\_l1,

item\_name2:data\_l2})

pearson = data.corr(method='pearson').values[0].tolist()[1]

spearman = data.corr(method='spearman').values[0].tolist()[1]

pearson = round(pearson,5)

spearman = round(spearman,5)

# print('pearson:',pearson)

# print('spearman:',spearman)

return pearson,spearman

def verify\_limit(lsl,usl):

if lsl != None:

lsl.replace(' ','')

if usl != None:

usl.replace(' ','')

try:

lsl = float(eval(lsl))

except:

lsl = None

try:

usl = float(eval(usl))

except:

usl = None

return lsl,usl

def calculate\_value(message):

print("this function is calculate\_value......")

val = r.get(message) # 注意 等到的都是字符串

if val:

val = float(val)\*200 # 数学运算

val = str(val).encode('utf-8')

return val

else:

return b'0'

def run(n):

while True:

try:

print("wait for scatter ...")

zmqMsg = socket.recv()

socket.send(b'scatter.png')

if len(zmqMsg)>0:

key = zmqMsg.decode('utf-8')

print("-->message from scatter client:", key)

table\_data = get\_redis\_data(key)

if len(table\_data)>0 :

scatter\_plot(table\_data,key)

else:

print("---get data error")

# socket.send(ret.decode('utf-8').encode('ascii'))

else:

time.sleep(0.05)

except Exception as e:

print('error:',e)

if \_\_name\_\_ == '\_\_main\_\_':

run(0)

# -\*- coding: utf-8 -\*-

import os

import sys

BASE\_DIR=os.path.dirname(os.path.abspath(\_\_file\_\_))

#print('BASE\_DIR--->',BASE\_DIR)

sys.path.insert(0,BASE\_DIR+'/site-packages/')

import csv

import textwrap

import random

import pandas as pd

import matplotlib.pyplot as plt

import matplotlib

matplotlib.use("Agg")

#%matplotlib inline #not sure what this line does

import numpy as np

import matplotlib.cm as cm

try:

import zmq

except Exception as e:

print('import zmq error:',e)

try:

import redis

except Exception as e:

print('import redis error:',e)

try:

from datetime import datetime, timedelta

except Exception as e:

print('import datetime,timedelta error:',e)

station\_id\_key = ''

slot\_id\_key = ''

redisClient = redis.Redis(host='localhost', port=6379, db=0)

context = zmq.Context()

socket = context.socket(zmq.REP)

socket.setsockopt(zmq.LINGER,0)

socket.bind("tcp://127.0.0.1:3191")

def clear\_files(path): # '/tmp/CPK\_Log/fail\_plot/'

for root, dirs, files in os.walk(path):

for file in files:

os.remove(path + '/' + file)

def my\_mkdir(path):

isExists=os.path.exists(path)

if not isExists:

os.makedirs(path)

return True

else:

return False

def get\_csv\_file\_name(file\_dir):

csv\_file\_l = []

files = os.listdir(file\_dir)

files.sort(key=lambda x: str(x.split('.')[0]))

for file in files:

if os.path.splitext(file)[1] == '.csv':

csv\_file\_l.append(os.path.splitext(file)[0])

return csv\_file\_l

def checkItemName(string, minlength, maxlength):

lines = textwrap.wrap(string.replace('\_',' '), maxlength)

newlines = list(lines)

index = 0

for l in lines:

if len(l) < minlength and index > 0:

prelist = lines[index-1].split(" ")

postlist = l.split(" ")

lastword = prelist[-1]

prelist.remove(lastword)

postlist.insert(0,lastword)

if len( " ".join(prelist) ) >= minlength:

newlines[index-1] = " ".join(prelist)

newlines[index] = " ".join(postlist)

index += 1

wraptext = "\n".join(newlines)

return wraptext

def randomcolor():

colorArr = ['1','2','3','4','5','6','7','8','9','A','B','C','D','E','F']

color = ""

for i in range(6):

color += colorArr[random.randint(0,14)]

return "#"+color

def summary\_retest\_csv(csv\_data\_path,header\_item\_path,csv\_path\_output):

with open(header\_item\_path,'r') as csvfile:

reader = csv.reader(csvfile)

header\_rows = [row for row in reader]

data1 = pd.read\_csv(csv\_data\_path)

data0 = pd.read\_csv('/tmp/CPK\_Log/retest/..total\_count\_by\_date\_product.csv')

df0 = pd.DataFrame(data0)

df = pd.DataFrame(data1)

if len(data1) == 0:

f\_csv = open(csv\_path\_output,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

header\_list = header\_rows[0]

date\_total\_list = header\_rows[5]

data\_group\_name = []

data\_group\_name.append('Date')

for groupName in header\_list:

data\_group\_name.append(groupName)

data\_group\_name.append(groupName+'\_#TOTAL')

data\_group\_name.append(groupName+'\_#RATE')

data\_group\_name.append('ALL')

data\_group\_name.append('ALL\_#TOTAL')

data\_group\_name.append('ALL\_#RATE')

csv\_writer.writerow(data\_group\_name)

for date\_time in date\_total\_list:

data\_list\_1 = []

data\_list\_1.append(str(date\_time))

all\_count = 0

for groupName in header\_list:

t\_flag = False

try:

tt\_list = list(df0.loc[(df0['Date Product'] == str(date\_time) + ' ' + str(groupName)), 'TOTAL'].tolist())

t\_flag = True

if len(tt\_list) == 0:

tt\_list.append('0')

except Exception as e:

# print('error:',str(e))

t\_flag = False

if t\_flag:

data\_list\_1.append('0')

# print('---111>>>>>',str(date\_time) + ' ' + str(groupName))

data\_list\_1.append(str(tt\_list[0]))

data\_list\_1.append('0')

all\_count = all\_count + int(tt\_list[0])

data\_list\_1.append('0')

data\_list\_1.append(str(all\_count))

data\_list\_1.append('0')

csv\_writer.writerow(data\_list\_1)

f\_csv.close()

return

#-------if have data---------

x\_datetime = []

for x\_date in df0['Date']:

date\_time = pd.to\_datetime(x\_date).strftime("%Y/%m/%d")

x\_datetime.append(date\_time)

x\_datetime = sorted(list(set(x\_datetime)))

# print(x\_datetime)

data\_group\_name = []

data\_group\_name.append('Date')

dict\_d = {}

for groupName, groupDf in df.groupby(by='Product'):

# print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*',groupName)

if len(groupName)>0:

data\_group\_name.append(groupName)

data\_group\_name.append(groupName+'\_#TOTAL')

data\_group\_name.append(groupName+'\_#RATE')

l\_retest = []

for x\_time in x\_datetime:

timeStart = pd.to\_datetime(x\_time,format = "%Y/%m/%d")

timeEnd = timeStart + timedelta(days=0,hours= 23,minutes = 59,seconds = 59)

n\_retest\_count = 0

for index,row in groupDf.iterrows():

lineDate = pd.to\_datetime(row['Date'])

if lineDate >=timeStart and lineDate<=timeEnd:

n\_retest\_count = n\_retest\_count + 1

tt\_list = list(df0.loc[(df0['Date Product'] == str(x\_time) + ' ' + str(groupName)), 'TOTAL'].tolist())

try:

if len(tt\_list) == 0:

tt\_list.append("0")

list\_rate\_percent = '0'

else:

x\_percent = round(int(n\_retest\_count)/int(tt\_list[0])\*100,3)

list\_rate\_percent = str(x\_percent)

except Exception as e:

print('summary\_retest\_csv error',str(e))

tt\_list.append('0')

list\_rate\_percent = '0'

l\_retest.append(str(x\_time)+','+str(n\_retest\_count)+','+str(tt\_list[0])+','+str(list\_rate\_percent))

dict\_d[groupName] = l\_retest

for groupName in header\_rows[0]: # only pass data,no retest data also need to add

if groupName not in data\_group\_name:

data\_group\_name.append(groupName)

data\_group\_name.append(groupName+'\_#TOTAL')

data\_group\_name.append(groupName+'\_#RATE')

l\_retest = []

for x\_time in x\_datetime:

tt\_list = list(df0.loc[(df0['Date Product'] == str(x\_time) + ' ' + str(groupName)), 'TOTAL'].tolist())

try:

xx\_tmp = tt\_list[0]

except Exception as e:

# print('for tt\_list[0]:',e)

xx\_tmp = 0

l\_retest.append(str(x\_time)+',0,'+str(xx\_tmp)+',0')

dict\_d[groupName] = l\_retest

n\_count = 0

for name in data\_group\_name:

if (not name == 'Date') and ('\_#TOTAL' not in name) and ('\_#RATE' not in name):

if n\_count == 0:

f\_csv = open(csv\_path\_output,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(data\_group\_name)

for row\_name in dict\_d[name]:

date\_row = row\_name.split(',')

csv\_writer.writerow([date\_row[0],date\_row[1],date\_row[2],date\_row[3]])

f\_csv.close()

else:

with open(csv\_path\_output, 'r') as f:

reader = csv.reader(f)

store\_d = []

for row in reader:

store\_d.append(row)

f\_csv = open(csv\_path\_output,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(data\_group\_name)

n\_index = 1

for row\_name in dict\_d[name]:

date\_row = row\_name.split(',')

row\_line = store\_d[n\_index]

row\_line.insert(len(row\_line),date\_row[1])

row\_line.insert(len(row\_line),date\_row[2])

row\_line.insert(len(row\_line),date\_row[3])

csv\_writer.writerow(row\_line)

n\_index = n\_index +1

f\_csv.close()

n\_count = n\_count +1

n\_total\_all\_list = []

n\_total\_retest\_all\_list = []

for x\_time in x\_datetime:

timeStart = pd.to\_datetime(x\_time,format = "%Y/%m/%d")

timeEnd = timeStart + timedelta(days=0,hours= 23,minutes = 59,seconds = 59)

n\_total\_all = 0

for groupName0, groupDf0 in df0.groupby(by='Date Product'):

if x\_time in groupName0:

n\_total\_all = int(groupDf0['TOTAL']) + n\_total\_all

n\_total\_all\_list.append(n\_total\_all)

n\_total\_retest\_all = 0

for index,row in df.iterrows():

lineDate = pd.to\_datetime(row['Date'])

if lineDate >=timeStart and lineDate <= timeEnd:

n\_total\_retest\_all = n\_total\_retest\_all+1

n\_total\_retest\_all\_list.append(n\_total\_retest\_all)

with open(csv\_path\_output, 'r') as f:

reader = csv.reader(f)

store\_d = []

for row in reader:

store\_d.append(row)

f\_csv = open(csv\_path\_output,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

n\_index = 0

for row\_d in store\_d:

if n\_index == 0:

row\_line = store\_d[n\_index]

row\_line.insert(len(row\_line),'ALL')

row\_line.insert(len(row\_line),'ALL\_#TOTAL')

row\_line.insert(len(row\_line),'ALL\_#RATE')

csv\_writer.writerow(row\_line)

else:

row\_line = store\_d[n\_index]

try:

total\_num = n\_total\_all\_list[n\_index-1]

except Exception as e:

print('.error get retest',e)

total\_num = 0

try:

retest\_num = n\_total\_retest\_all\_list[n\_index-1]

except Exception as e:

print('.error get retest',e)

retest\_num = 0

if total\_num == 0:

rate\_percentage = 0

else:

rate\_percentage = round(float(retest\_num)/float(total\_num)\*100,3)

row\_line.insert(len(row\_line),str(retest\_num))

row\_line.insert(len(row\_line),str(total\_num))

row\_line.insert(len(row\_line),str(rate\_percentage))

csv\_writer.writerow(row\_line)

n\_index = n\_index+1

f\_csv.close()

def retest\_vs\_Version\_csv(csv\_data\_path,header\_item\_path,keyword,csv\_path\_output):

data0 = pd.read\_csv('/tmp/CPK\_Log/retest/..total\_count\_by\_version.csv')

df0 = pd.DataFrame(data0)

data1 = pd.read\_csv(csv\_data\_path)

df = pd.DataFrame(data1)

with open(header\_item\_path,'r') as csvfile:

reader = csv.reader(csvfile)

header\_rows = [row for row in reader]

print('>retest\_vs\_Version\_csv start')

list\_version\_id = []

list\_retest\_count = []

list\_total\_count = []

list\_rate\_percent = []

for groupName, groupDf in df.groupby(by=keyword): #'Station ID'

n\_count = 0

for index,row in groupDf.iterrows():

n\_count = n\_count + 1

list\_version\_id.append(groupName)

list\_retest\_count.append(n\_count)

tt\_list = list(df0.loc[(df0['Version'] == groupName), 'TOTAL'].tolist())

try:

xx\_tmp = tt\_list[0]

except Exception as e:

# print('for tt\_list[0]:',e)

xx\_tmp = 0

list\_total\_count.append(str(xx\_tmp))

try:

if int(xx\_tmp) == 0:

list\_rate\_percent.append('0')

else:

x\_percent = round(int(n\_count)/int(xx\_tmp)\*100,3)

list\_rate\_percent.append(str(x\_percent))

except Exception as e:

print('error convert int :',str(e))

list\_rate\_percent.append('0')

print('>retest\_vs\_Version\_csv done')

f\_csv = open(csv\_path\_output,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Index','Retest\_Count','TOTAL','RATE'])

n\_index = 0

for x\_row in list\_version\_id:

csv\_writer.writerow([str(list\_version\_id[n\_index]),str(list\_retest\_count[n\_index]),str(list\_total\_count[n\_index]),str(list\_rate\_percent[n\_index])])

n\_index = n\_index+1

diff\_list = find\_diff\_intwo\_list(list\_version\_id,header\_rows[4])

for diff\_value in diff\_list:

tt\_list = list(df0.loc[(df0['Version'] == diff\_value), 'TOTAL'].tolist())

try:

xx\_tmp = tt\_list[0]

except Exception as e:

# print('for tt\_list[0]:',e)

xx\_tmp = 0

csv\_writer.writerow([str(diff\_value),'0',str(xx\_tmp),'0'])

f\_csv.close()

def cut\_station\_name(staion\_name):

station\_name\_list = staion\_name.split('-')

if len(station\_name\_list)>2:

return ''.join(station\_name\_list[2:])

if len(station\_name\_list)>1:

return ''.join(station\_name\_list[1:])

return ''.join(station\_name\_list[0])

# for station id - slot id ,version

def retest\_vs\_plot(csv\_path,title\_name,pie\_retest\_csv\_path,pic\_path,descend = False):

isExists=os.path.exists(csv\_path)

if not isExists:

plt.gca().spines["top"].set\_alpha(.0)

plt.gca().spines["bottom"].set\_alpha(.5)

plt.gca().spines["right"].set\_alpha(.0)

plt.gca().spines["left"].set\_alpha(.5)

plt.tight\_layout()

plt.savefig(pic\_path, dpi=200)

return

data1= pd.read\_csv(csv\_path)

df = pd.DataFrame(data1)

if descend == True:

df = df.sort\_values('RATE', ascending=False)

df.index=range(len(df))

# pass

rowCount = int(df.iloc[:,0].size)+1

columnCount = int(df.columns.size)

print('rowCount:',rowCount)

n\_reference = 30

rowPage = int(rowCount/n\_reference)

row\_2 = rowCount%n\_reference

if rowCount <=n\_reference+2:

generate\_retest\_plot(df, title\_name, pie\_retest\_csv\_path, pic\_path, 0, rowCount)

else:

for i in range(rowPage):

if i == 0:

generate\_retest\_plot(df, title\_name, pie\_retest\_csv\_path, pic\_path, i\*n\_reference,i\*n\_reference+n\_reference-1)

else:

pic\_path\_new = pic\_path.replace('.png','') + str(i)+'.png'

generate\_retest\_plot(df, title\_name, pie\_retest\_csv\_path, pic\_path\_new, i\*n\_reference,i\*n\_reference+n\_reference-1)

if row\_2>1:

pic\_path\_new = pic\_path.replace('.png','') + str(i+1)+'.png'

generate\_retest\_plot(df, title\_name, pie\_retest\_csv\_path, pic\_path\_new,rowPage\*n\_reference,rowPage\*n\_reference+row\_2)

def generate\_retest\_plot(df\_data,title\_name,pie\_retest\_csv\_path,pic\_path,n\_start, n\_end):

df\_count = pd.read\_csv(pie\_retest\_csv\_path)

if len(df\_count['TOTAL']) == 0:

input\_total\_count = 0

else:

input\_total\_count= max(df\_count['TOTAL'])

df = df\_data.loc[n\_start:n\_end]

df.index=range(len(df))

plt.figure(figsize=(15,8))

colorList = list(plt.cm.colors.cnames.keys())

random.seed(100)

c = random.choices(colorList, k=100)

n\_index = 0

max\_y = 0

plt.style.use("bmh")

xtick\_location = df.index.tolist()[::1]

xtick\_labels = df.Index.tolist()[::1]

xtick\_retest\_count = df.Retest\_Count.tolist()[::1]

xtick\_total = df.TOTAL.tolist()[::1]

# print('X\_Y:',xtick\_location,xtick\_labels,xtick\_retest\_count,xtick\_total)

for header\_name in df.columns:

if header\_name == 'RATE':

data = df[header\_name].values

plt.plot('Index', header\_name, data=df, color=c[n\_index], label='Total Input Count('+str(input\_total\_count)+')', alpha=1)

if len(data) ==0:

max\_data = 0

else:

max\_data = max(data)

if max\_y<max\_data:

max\_y = max\_data

n\_index = n\_index +1

ii\_n = 0

for x0,y0 in zip(xtick\_location,data):

# plt.text(x0,y0,y0,fontdict={'fontsize':11},alpha=0.6)

plt.text(x0,y0,('%s/%s') % (str(xtick\_retest\_count[ii\_n]),str(xtick\_total[ii\_n])),fontdict={'fontsize':11},alpha=0.7)

ii\_n = ii\_n+1

# Get the Peaks and Troughs

# data = df['Retest\_Count'].values

# data2 = df['traffic2'].values

# doublediff = np.diff(np.sign(np.diff(data)))

# peak\_locations = np.where(doublediff == -2)[0] + 1

# doublediff2 = np.diff(np.sign(np.diff(-1\*data)))

# trough\_locations = np.where(doublediff2 == -2)[0] + 1

# Draw Plot

# plt.plot('Date', 'Retest\_Count', data=df, color=c[0], label='Air ====Traffic1',alpha=1)

# plt.plot('date', 'traffic2', data=df, color=c[1], label='Air Traffic2',alpha=1)

# plt.scatter(df.date[peak\_locations], df.traffic[peak\_locations], marker=matplotlib.markers.CARETUPBASE, color='tab:green', s=100, label='Peaks')

# plt.scatter(df.date[trough\_locations], df.traffic[trough\_locations], marker=matplotlib.markers.CARETDOWNBASE, color='tab:red', s=100, label='Troughs')

# # Annotate

# for t, p in zip(trough\_locations[::1], peak\_locations[::]):

# # plt.text(x0,y0,'%.2f' % y0,fontdict={'fontsize':14})

# plt.text(df.date[p], df.traffic[p]+15, df.date[p], horizontalalignment='center', color='darkgreen')

# plt.text(df.date[t], df.traffic[t]-35, df.date[t], horizontalalignment='center', color='darkred')

# Decoration

# print('->>max\_y:',max\_y)

if max\_y ==0:

max\_y = 0.001

plt.ylim(0,max\_y\*1.1)

plt.xticks(ticks=xtick\_location, labels=xtick\_labels, rotation=45, fontsize=9, alpha=1)

plt.title(title\_name, fontsize=22)

plt.yticks(fontsize=13, alpha=1)

plt.ylabel("Percentage(%)",fontsize=15)

plt.xticks(fontsize=13, alpha=1)

# Lighten borders

plt.gca().spines["top"].set\_alpha(.0)

plt.gca().spines["bottom"].set\_alpha(.5)

plt.gca().spines["right"].set\_alpha(.0)

plt.gca().spines["left"].set\_alpha(.5)

plt.legend(loc='upper left')

# plt.grid(axis='x', alpha=.3)

# plt.grid(axis='y', alpha=.4)

plt.tight\_layout()

plt.savefig(pic\_path, dpi=200)

def daily\_retest\_summary\_plot(csv\_path, pic\_all\_path, pic\_path):

isExists=os.path.exists(csv\_path)

if not isExists:

plt.gca().spines["top"].set\_alpha(.0)

plt.gca().spines["bottom"].set\_alpha(.5)

plt.gca().spines["right"].set\_alpha(.0)

plt.gca().spines["left"].set\_alpha(.5)

plt.tight\_layout()

plt.savefig(pic\_path, dpi=200)

return

data1= pd.read\_csv(csv\_path)

df = pd.DataFrame(data1)

rowCount = int(df.iloc[:,0].size)+1

columnCount = int(df.columns.size)

n\_reference = 30

rowPage = int(rowCount/n\_reference)

row\_2 = rowCount%n\_reference

if rowCount <=n\_reference + 2:

generate\_retest\_summary\_plot(df,pic\_path,0,rowCount)

generate\_retest\_all\_summary\_plot(df,pic\_all\_path,0,rowCount)

else:

for i in range(rowPage):

if i == 0:

generate\_retest\_summary\_plot(df,pic\_path,i\*n\_reference,i\*n\_reference+n\_reference-1)

generate\_retest\_all\_summary\_plot(df,pic\_all\_path,i\*n\_reference,i\*n\_reference+n\_reference-1)

else:

pic\_path\_new = pic\_path.replace('.png','') + str(i)+'.png'

generate\_retest\_summary\_plot(df,pic\_path\_new,i\*n\_reference,i\*n\_reference+n\_reference-1)

pic\_all\_path\_new = pic\_path.replace('.png','') + str(i)+'.png'

generate\_retest\_all\_summary\_plot(df,pic\_all\_path\_new,i\*n\_reference,i\*n\_reference+n\_reference-1)

if row\_2>1:

pic\_path\_new = pic\_path.replace('.png','') + str(i+1)+'.png'

generate\_retest\_summary\_plot(df,pic\_path\_new,rowPage\*n\_reference,rowPage\*n\_reference+row\_2)

pic\_all\_path\_new = pic\_path.replace('.png','') + str(i+1)+'.png'

generate\_retest\_all\_summary\_plot(df,pic\_all\_path\_new,rowPage\*n\_reference,rowPage\*n\_reference+row\_2)

def generate\_retest\_all\_summary\_plot(df\_data, pic\_path, n\_start, n\_end):

plt.style.use("bmh")

df = df\_data.loc[n\_start:n\_end]

df.index=range(len(df))

plt.figure(figsize=(15,8))

colorList = list(plt.cm.colors.cnames.keys())

random.seed(100)

c = random.choices(colorList, k=100)

n\_index = 0

max\_y = 0

xtick\_location = df.index.tolist()[::1]

xtick\_labels = df.Date.tolist()[::1]

for header\_name in df.columns:

if ('ALL\_#RATE' in header\_name):

data = df[header\_name].values

xtick\_retest\_count = df[header\_name.replace('\_#RATE','')]

xtick\_total = df[header\_name.replace('\_#RATE','\_#TOTAL')]

xtick\_retest\_count\_all = df\_data[header\_name.replace('\_#RATE','')]

xtick\_total\_all = df\_data[header\_name.replace('\_#RATE','\_#TOTAL')]

plt.plot('Date', header\_name, data=df, color= '#FE420F', label=header\_name.replace('\_#RATE','') + '('+str(sum(xtick\_retest\_count\_all))+'/'+str(sum(xtick\_total\_all))+')', alpha=0.8)

if len(data) == 0:

max\_data = 0

else:

max\_data = max(data)

ii\_n = 0

for x0,y0 in zip(xtick\_location,data):

plt.text(x0,y0,('%s/%s') % (str(xtick\_retest\_count[ii\_n]),str(xtick\_total[ii\_n])),fontdict={'fontsize':11},alpha=1,color=c[n\_index])

ii\_n = ii\_n+1

if max\_y<max\_data:

max\_y = max\_data

n\_index = n\_index +1

if max\_y ==0:

max\_y = 0.001

plt.ylim(0,max\_y\*1.1)

plt.xticks(ticks=xtick\_location, labels=xtick\_labels, rotation=45, fontsize=9, alpha=1)

plt.title("Daily Retest Summary Chart All Product Wise", fontsize=22)

plt.yticks(fontsize=13, alpha=1)

plt.ylabel("Percentage(%)",fontsize=15)

plt.xticks(fontsize=13, alpha=1)

plt.gca().spines["top"].set\_alpha(.0)

plt.gca().spines["bottom"].set\_alpha(.5)

plt.gca().spines["right"].set\_alpha(.0)

plt.gca().spines["left"].set\_alpha(.5)

plt.legend(loc='upper left')

plt.tight\_layout()

plt.savefig(pic\_path, dpi=200)

def generate\_retest\_summary\_plot(df\_data, pic\_path, n\_start, n\_end):

plt.style.use("bmh")

df = df\_data.loc[n\_start:n\_end]

df.index=range(len(df))

plt.figure(figsize=(15,8))

colorList = list(plt.cm.colors.cnames.keys())

random.seed(100)

# c = random.choices(colorList, k=100)

c = ['#014182','#FF0000','#008000','#00FFFF','#9400D3','#F0833A','#B8860B','#FFA500','#A9A9A9','#FFFF00','#BFF128','#87A922','#B9CC81','#4B5D16','#5CB200','#B1FF65','#8EE53F','#58BC08','#4DA409','#C1FD95']

n\_index = 0

max\_y = 0

xtick\_location = df.index.tolist()[::1]

xtick\_labels = df.Date.tolist()[::1]

for header\_name in df.columns:

if header\_name != 'Date' and ('\_#RATE' in header\_name):

data = df[header\_name].values

xtick\_retest\_count = df[header\_name.replace('\_#RATE','')]

xtick\_total = df[header\_name.replace('\_#RATE','\_#TOTAL')]

xtick\_retest\_count\_all = df\_data[header\_name.replace('\_#RATE','')]

xtick\_total\_all = df\_data[header\_name.replace('\_#RATE','\_#TOTAL')]

plt.plot('Date', header\_name, data=df, color=c[n\_index%20], label=header\_name.replace('\_#RATE','') + '('+str(sum(xtick\_retest\_count\_all))+'/'+str(sum(xtick\_total\_all))+')', alpha=0.8)

if len(data) == 0:

max\_data = 0

else:

max\_data = max(data)

ii\_n = 0

for x0,y0 in zip(xtick\_location,data):

plt.text(x0,y0,('%s/%s') % (str(xtick\_retest\_count[ii\_n]),str(xtick\_total[ii\_n])),fontdict={'fontsize':11},alpha=1,color=c[n\_index])

ii\_n = ii\_n+1

if max\_y<max\_data:

max\_y = max\_data

n\_index = n\_index +1

if max\_y ==0:

max\_y = 0.001

plt.ylim(0,max\_y\*1.1)

plt.xticks(ticks=xtick\_location, labels=xtick\_labels, rotation=45, fontsize=9, alpha=1)

plt.title("Daily Retest Summary Chart Product Wise", fontsize=22)

plt.yticks(fontsize=13, alpha=1)

plt.ylabel("Percentage(%)",fontsize=15)

plt.xticks(fontsize=13, alpha=1)

plt.gca().spines["top"].set\_alpha(.0)

plt.gca().spines["bottom"].set\_alpha(.5)

plt.gca().spines["right"].set\_alpha(.0)

plt.gca().spines["left"].set\_alpha(.5)

plt.legend(loc='upper left')

plt.tight\_layout()

plt.savefig(pic\_path, dpi=200)

def pareto\_plot(csv\_path, pic\_color, title, x=None, y=None, customer = None, show\_pct\_y=False, pct\_format='{0:.0%}',saveas = None):

df = pd.read\_csv(csv\_path)

plt.style.use("seaborn")

plt.figure(figsize=(15,8))

# if customer != None:

# df = df[df['Supplier'].str.match(customer)] # omits data not from supplier

#title = 'Pareto Chart for Top 5 Retest'

occurrences = df.groupby(x).count().reset\_index()

n\_count = df[x].unique().\_\_len\_\_()+1

ylabel = "Frequency of Occurrence"

tmp = occurrences.sort\_values(y, ascending=False)

x = tmp[x].tolist()

if len(x)>5:

x = x[0:5]

x = [checkItemName(i,5,13) for i in x]

y = tmp[y].tolist()

if len(y)>5:

y = y[0:5]

# x = [str(x[i]) + '\n(' + str(y[i]) + ')' for i in range(min(len(x),len(y)))]

# print(x)

# print(y)

#x = tmp[x].values

#y = tmp[y].values

# at this point, x should be an ordered list of x axis categories

# and y should be the number of occurrences

#weights = y / y.sum()

weights = []

# colorList = []

for count in y:

weights.append(count/sum(y))

# for count in weights:

# colorList.append(count+(1-max(weights)))

# print(weights)

# print(colorList)

colorList = list(plt.cm.colors.cnames.keys())

colorList = colorList[2:]

cumsum = []

for counter,percent in enumerate(weights):

cumsum.append(sum(weights[:counter+1]))

fig, ax1 = plt.subplots()

random.seed(100)

c = random.choices(colorList, k=n\_count)

ax1.bar(x, y,color=pic\_color,width=.5,alpha=0.8)

#ax1.set\_xlabel(xlabel)

# ax1.set\_ylabel(ylabel,fontsize=12)

ax1.set\_ylabel(ylabel)

# plt.ylabel(ylabel,fontsize=15)

if len(y)==0:

ax1.set\_ylim([0,1])

else:

ax1.set\_ylim([0,max(y)\*1.4])

ax2 = ax1.twinx()

ax2.plot(x, cumsum, '-s',color = "black",alpha=1)#, alpha=0.5)

ax2.set\_ylabel('', color='purple')

ax2.tick\_params('y')#, colors='purple') #right y axis label color

ax2.set\_ylim([0,1.05])

ax2.grid(alpha = 0)

# ax1.set\_xticklabels(x,rotation=45,fontsize=13)

ax1.set\_xticklabels(x,rotation=45)

# vals = ax2.get\_yticks()

# label\_format = '{:,.0%}'

# xx =[label\_format.format(x) for x in vals]

# ax2.set\_yticklabels(xx)

# hide y-labels on right side

if not show\_pct\_y:

ax2.set\_yticks([])

formatted\_weights = [pct\_format.format(x) for x in cumsum]

bbox\_props = dict(boxstyle="round,pad=0.5", fc="w", ec="0", lw=.5,alpha=1)

for i, txt in enumerate(formatted\_weights):

ax2.text(x[i], cumsum[i],txt, horizontalalignment='center',verticalalignment="bottom",bbox=bbox\_props,fontdict={'fontweight':300, 'size':10})

yy = y[i]

# y\_value = cumsum[i]

# if abs(yy - y\_value)<0.5:

# yy = yy\*1.07 # 防止数量标签被挡住

ax1.text(x[i], yy, str(y[i]), horizontalalignment='center', verticalalignment='bottom', fontdict={'fontweight':400, 'size':10})

plt.title(title,fontsize = 15)

pie\_retest\_csv\_path = '/tmp/CPK\_Log/retest/.pie\_retest.csv'

df\_count = pd.read\_csv(pie\_retest\_csv\_path)

if len(df\_count['TOTAL']) == 0:

input\_total\_count = 0

else:

input\_total\_count= max(df\_count['TOTAL'])

ax1.legend(['Total Input Count('+str(input\_total\_count)+')'],loc="upper left",fontsize=8,frameon=False)

# plt.plot(color=pic\_color, label='Total Input Count('+str(input\_total\_count)+')', alpha=1)

plt.tight\_layout()

# plt.show()

if saveas != None:

fig.savefig(saveas, dpi=200)

def generateRetestCSV(all\_csv\_path,retest\_csv\_path,pie\_retest\_csv\_path,fail\_csv\_path,header\_item\_path):

tmp\_lst = []

with open(all\_csv\_path, 'r') as f:

reader = csv.reader(f)

i = 1

for row in reader:

if row[0].lower().find('display name') != -1:

pass

elif row[0].lower().find('pdca priority') != -1:

pass

elif row[0].lower().find('upper limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('lower limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('measurement unit') != -1: # "Measurement Unit ----->" in row:

pass

elif row[0].lower().find('site') != -1:

tmp\_lst.append(row)

else:

tmp\_lst.append(row)

i = i + 1

header\_list = tmp\_lst[1]

df = pd.DataFrame(tmp\_lst[2:], columns=tmp\_lst[1])

try:

pd.to\_datetime(df['StartTime'])

except Exception as e:

print('check StartTime,csv format wrong!')

return e

header\_df =df[0:2]

# print('header\_df before--->', header\_df)

data\_df = df[2:]

# print('data\_df before--->', data\_df)

# print('csv data row number before remove SN empty--->', len(data\_df.values.tolist()))

data\_df=data\_df[~data\_df['SerialNumber'].isin([''])]#Remove SN Empty

# print('csv data row number after remove SN empty--->', len(data\_df.values.tolist()))

# print('csv data row number before remove fail--->', len(data\_df.values.tolist()))

total\_sn = list((data\_df['SerialNumber'].values.tolist()))

print('>>>>total\_sn:',len(total\_sn))

pass\_df=data\_df[data\_df['Test Pass/Fail Status'].isin(['PASS'])]

# print('pass\_df:',len(pass\_df))

overall\_pass\_sn = list(set(list((pass\_df['SerialNumber'].values.tolist()))))

# print('pass\_sn:',len(overall\_pass\_sn))

# print('--->>>',data\_df['SerialNumber'])

fail\_df=data\_df[data\_df['Test Pass/Fail Status'].isin(['FAIL'])]

overall\_fail\_sn = list(set(list((fail\_df['SerialNumber'].values.tolist()))))

no\_retest\_pass\_sn = [x for x in overall\_pass\_sn if x not in overall\_fail\_sn] #在list1列表中而不在list2列表中

no\_retest\_pass\_sn = list(set(no\_retest\_pass\_sn))

print('plot->no\_retest\_pass\_sn',len(no\_retest\_pass\_sn))

true\_fail\_sn = [y for y in overall\_fail\_sn if y not in overall\_pass\_sn] #在list2列表中而不在list1列表中

true\_fail\_sn = list(set(true\_fail\_sn))

# print('fail\_df:',len(fail\_df),len(overall\_fail\_sn))

total\_retest\_sn = [z for z in overall\_fail\_sn if z in overall\_pass\_sn] #在list2列表中而不在list1列表中

total\_retest\_sn = list(set(total\_retest\_sn))

keyword\_list = ['STATION ID','SITE\_ID']

keyword\_list2 = ['FIXTURE\_SETUP CHANNEL CHANNEL\_ID','FIXTURE CHANNEL ID','FIXTURE INITILIZATION SLOT\_ID','FIXTURE RESET CALC FIXTURE\_CHANNEL','HEAD ID','FIXTURE\_CHANNEL CHANNEL CHANNEL\_ID','FIXTURE CHANNEL CHANNEL\_ID','CHANNEL ID','CHANNEL\_ID','SLOT ID','SLOT\_ID']

#Fixture\_Setup Channel Channel\_id

global station\_id\_key

global slot\_id\_key

for keyword in keyword\_list:

header\_data = [s.upper() for s in data\_df.columns if isinstance(s,str)==True]

n\_count = 0

for item\_data in header\_data:

if keyword == item\_data:

station\_id\_key = data\_df.columns[n\_count]

break

n\_count = n\_count + 1

for keyword in keyword\_list2:

header\_data = [s.upper() for s in data\_df.columns if isinstance(s,str)==True]

n\_count = 0

for item\_data in header\_data:

if keyword == item\_data:

slot\_id\_key = data\_df.columns[n\_count]

break

n\_count = n\_count + 1

f\_csv = open(retest\_csv\_path,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Date','SerialNumber','RetestItem','Product','Station ID','Version','Slot ID','Measured Value','[LSL;USL]'])

total\_temp\_p\_l=[]

total\_temp\_r\_l=[]

for retest\_sn in total\_retest\_sn:

if retest\_sn !='':

start\_time\_l = list(data\_df.loc[ data\_df['SerialNumber'] == retest\_sn, 'StartTime'].tolist())

first\_test\_time = min(start\_time\_l)

sn\_status = list(data\_df.loc[ data\_df['StartTime'].isin([first\_test\_time]) & (data\_df['SerialNumber'] == retest\_sn), 'Test Pass/Fail Status'].tolist())

if sn\_status == ['PASS']:

total\_temp\_p\_l = total\_temp\_p\_l + [retest\_sn]

elif sn\_status == ['FAIL']:

total\_temp\_r\_l = total\_temp\_r\_l + [retest\_sn]

fail\_list = list(data\_df.loc[data\_df['StartTime'].isin([first\_test\_time]) & (data\_df['SerialNumber'] == retest\_sn), 'List of Failing Tests'].tolist())

product\_list = list(data\_df.loc[data\_df['StartTime'].isin([first\_test\_time]) & (data\_df['SerialNumber'] == retest\_sn), 'Product'].tolist())

# print('fail\_list:',retest\_sn, fail\_list,product\_list[0])

fail\_list\_first\_item = fail\_list[0].split(';',1)[0]

if station\_id\_key == '':

station\_id\_list = ['']

else:

station\_id\_list = list(data\_df.loc[data\_df['StartTime'].isin([first\_test\_time]) & (data\_df['SerialNumber'] == retest\_sn), station\_id\_key].tolist())

if slot\_id\_key == '':

slot\_id\_list = ['']

else:

slot\_id\_list = list(data\_df.loc[data\_df['StartTime'].isin([first\_test\_time]) & (data\_df['SerialNumber'] == retest\_sn), slot\_id\_key].tolist())

version\_list = list(data\_df.loc[data\_df['StartTime'].isin([first\_test\_time]) & (data\_df['SerialNumber'] == retest\_sn), 'Version'].tolist())

try:

measured\_value = list(data\_df.loc[data\_df['StartTime'].isin([first\_test\_time]) & (data\_df['SerialNumber'] == retest\_sn), fail\_list\_first\_item].tolist())

except Exception as e:

# print('error measured\_value'+ str(e))

measured\_value = ['']

try:

limit\_value = '['+str(header\_df[fail\_list\_first\_item][1])+';'+str(header\_df[fail\_list\_first\_item][0])+']'

except Exception as e:

# print('error limit\_value'+ str(e))

limit\_value = ''

csv\_writer.writerow([first\_test\_time,retest\_sn,fail\_list\_first\_item,product\_list[0],str(station\_id\_list[0]),str(version\_list[0]),str(slot\_id\_list[0]),str(measured\_value[0]),limit\_value])

else:

print('>no test status')

total\_temp\_p\_l = total\_temp\_p\_l + [retest\_sn]

f\_csv.close()

print('plot->total\_temp\_p\_l',len(total\_temp\_p\_l),len(total\_temp\_r\_l),len(true\_fail\_sn))

total\_count = len(no\_retest\_pass\_sn+total\_temp\_p\_l+true\_fail\_sn+total\_temp\_r\_l)

total\_fail\_sn = true\_fail\_sn

fail\_count = len(total\_fail\_sn)

no\_retest\_pass\_count = len(no\_retest\_pass\_sn)

first\_pass\_count = len(no\_retest\_pass\_sn) + len(list(set(total\_temp\_p\_l)))

pass\_count = len(no\_retest\_pass\_sn+total\_retest\_sn)

total\_retest\_sn\_no\_first\_pass = list(set(total\_temp\_r\_l))

f\_csv = open(pie\_retest\_csv\_path,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['1st PASS','Retest PASS','FAILED','PASSED','No Retest PASS','TOTAL'])

csv\_writer.writerow([str(first\_pass\_count),str(len(total\_retest\_sn\_no\_first\_pass)),str(fail\_count),str(pass\_count),str(no\_retest\_pass\_count),str(total\_count)])

f\_csv.close()

f\_csv = open(fail\_csv\_path,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Date','SerialNumber','FailItem'])

data\_df2 = data\_df[data\_df['Test Pass/Fail Status'] == 'FAIL']

for fail\_sn in true\_fail\_sn:

start\_time\_l = list(data\_df2.loc[data\_df['SerialNumber'] == fail\_sn, 'StartTime'].tolist())

first\_test\_time = min(start\_time\_l)

fail\_list = list(data\_df2.loc[data\_df['SerialNumber']==fail\_sn, 'List of Failing Tests'].tolist())

fail\_list\_first\_item = fail\_list[0].split(';',1)[0]

csv\_writer.writerow([str(first\_test\_time),str(fail\_sn),fail\_list\_first\_item])

f\_csv.close()

f\_csv = open(header\_item\_path,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

try:

product\_list=list(set(list((data\_df['Product'].values.tolist()))))

except Exception as e:

print('error product\_list',str(e))

product\_list = ['']

csv\_writer.writerow(product\_list)

try:

if station\_id\_key == '':

station\_id\_list = ['']

else:

station\_id\_list=list(set(list((data\_df[station\_id\_key].values.tolist()))))

if slot\_id\_key == '':

slot\_id\_list = ['']

else:

slot\_id\_list=list(set(list((data\_df[slot\_id\_key].values.tolist()))))

except Exception as e:

print('error station\_id\_key',str(e))

station\_id\_list = ['']

slot\_id\_list = ['']

station\_slot\_id\_list = []

tation\_slot\_id\_list\_orig = []

for station\_id in station\_id\_list:

for slot\_id in slot\_id\_list:

if slot\_id == '':

station\_slot\_id\_list.append(cut\_station\_name(str(station\_id)))

tation\_slot\_id\_list\_orig.append(str(station\_id))

else:

station\_slot\_id\_list.append(cut\_station\_name(str(station\_id))+' '+str(slot\_id))

tation\_slot\_id\_list\_orig.append(str(station\_id)+' '+str(slot\_id))

csv\_writer.writerow(station\_slot\_id\_list)

csv\_writer.writerow(station\_id\_list)

csv\_writer.writerow(slot\_id\_list)

try:

version\_list=list(set(list((data\_df['Version'].values.tolist()))))

except Exception as e:

print('error version\_list',str(e))

version\_list = ['']

csv\_writer.writerow(version\_list)

try:

date\_total\_list = []

for single\_sn in total\_sn:

date\_time\_list\_tmp = list(data\_df.loc[(data\_df['SerialNumber'] == single\_sn), 'StartTime'].tolist())

date\_time\_tstr\_mp = pd.to\_datetime(str(date\_time\_list\_tmp[0])).strftime("%Y/%m/%d")

date\_total\_list.append(str(date\_time\_tstr\_mp))

date\_list = sorted(list(set(date\_total\_list)))

except Exception as e:

print('error version\_list',str(e))

date\_list = ['2020-00-00','2020-00-00']

csv\_writer.writerow(date\_list)

f\_csv.close()

# =====================by station id and slot id, by version, by date=======================

# n\_total\_name\_by\_product\_list = [groupName for groupName, groupDf in data\_df.groupby(data\_df='Product')]

# n\_total\_date\_by\_product\_list = []

# for x\_date in data\_df['StartTime']:

# date\_time = pd.to\_datetime(x\_date).strftime("%Y/%m/%d")

# n\_total\_date\_by\_product\_list.append(date\_time)

# n\_total\_date\_by\_product\_list = sorted(list(set(n\_total\_date\_by\_product\_list)))

n\_total\_count\_by\_station\_list = []

n\_first\_pass\_value\_list = []

n\_total\_count\_by\_version\_list = []

n\_first\_pass\_value\_by\_version\_list = []

n\_total\_count\_by\_date\_list = []

n\_first\_pass\_by\_date\_list = []

print('-1>no\_retest\_pass\_sn',len(no\_retest\_pass\_sn))

print('-1>true\_fail\_sn',len(true\_fail\_sn))

print('-1>total\_retest\_sn',len(total\_retest\_sn))

for no\_retest\_pass\_sn\_value in no\_retest\_pass\_sn:

try:

station\_id\_list\_pass = list(data\_df.loc[(data\_df['SerialNumber'] == no\_retest\_pass\_sn\_value), station\_id\_key].tolist())

if slot\_id\_key == '':

n\_first\_pass\_value\_list.append(str(station\_id\_list\_pass[0]))

else:

slot\_id\_list\_pass = list(data\_df.loc[(data\_df['SerialNumber'] == no\_retest\_pass\_sn\_value), slot\_id\_key].tolist())

n\_first\_pass\_value\_list.append(str(station\_id\_list\_pass[0])+' '+str(slot\_id\_list\_pass[0]))

version\_list\_pass = list(data\_df.loc[(data\_df['SerialNumber'] == no\_retest\_pass\_sn\_value), 'Version'].tolist())

n\_first\_pass\_value\_by\_version\_list.append(str(version\_list\_pass[0]))

date\_time\_list\_pass = list(data\_df.loc[(data\_df['SerialNumber'] == no\_retest\_pass\_sn\_value), 'StartTime'].tolist())

date\_time\_only\_day\_list\_pass = pd.to\_datetime(str(date\_time\_list\_pass[0])).strftime("%Y/%m/%d")

product\_list\_pass = list(data\_df.loc[(data\_df['SerialNumber'] == no\_retest\_pass\_sn\_value), 'Product'].tolist())

n\_first\_pass\_by\_date\_list.append(str(date\_time\_only\_day\_list\_pass)+' '+ str(product\_list\_pass[0]))

except Exception as e:

print('list no retest sn error'+ str(e))

# first pass

n\_first\_pass\_value\_set = set(n\_first\_pass\_value\_list)

print('>:n\_first\_pass\_value\_list',len(n\_first\_pass\_value\_list))

for n\_first\_pass\_item in n\_first\_pass\_value\_set:

# print('-first pass>>>',n\_first\_pass\_item,n\_first\_pass\_value\_list.count(n\_first\_pass\_item))

tmp\_pass\_1 = []

tmp\_pass\_1.append(n\_first\_pass\_item)

tmp\_pass\_1.append(n\_first\_pass\_value\_list.count(n\_first\_pass\_item))

n\_total\_count\_by\_station\_list.append(tmp\_pass\_1)

n\_first\_pass\_value\_by\_version\_set = set(n\_first\_pass\_value\_by\_version\_list)

for n\_first\_pass\_item in n\_first\_pass\_value\_by\_version\_set:

tmp\_pass\_1 = []

tmp\_pass\_1.append(n\_first\_pass\_item)

tmp\_pass\_1.append(n\_first\_pass\_value\_by\_version\_list.count(n\_first\_pass\_item))

n\_total\_count\_by\_version\_list.append(tmp\_pass\_1)

n\_first\_pass\_by\_date\_list\_set = set(n\_first\_pass\_by\_date\_list)

for n\_first\_pass\_item in n\_first\_pass\_by\_date\_list\_set:

tmp\_pass\_1 = []

tmp\_pass\_1.append(n\_first\_pass\_item)

tmp\_pass\_1.append(n\_first\_pass\_by\_date\_list.count(n\_first\_pass\_item))

n\_total\_count\_by\_date\_list.append(tmp\_pass\_1)

# true fail

n\_true\_fail\_value\_list = []

n\_true\_fail\_value\_by\_version\_list = []

n\_true\_fail\_value\_by\_date\_list = []

for true\_fail\_sn\_value in true\_fail\_sn:

try:

station\_id\_list\_fail = list(data\_df.loc[(data\_df['SerialNumber'] == true\_fail\_sn\_value), station\_id\_key].tolist())

if slot\_id\_key =='':

n\_true\_fail\_value\_list.append(str(station\_id\_list\_fail[0]))

else:

slot\_id\_list\_fail = list(data\_df.loc[(data\_df['SerialNumber'] == true\_fail\_sn\_value), slot\_id\_key].tolist())

n\_true\_fail\_value\_list.append(str(station\_id\_list\_fail[0])+' '+str(slot\_id\_list\_fail[0]))

version\_list\_fail = list(data\_df.loc[(data\_df['SerialNumber'] == true\_fail\_sn\_value), 'Version'].tolist())

n\_true\_fail\_value\_by\_version\_list.append(str(version\_list\_fail[0]))

date\_time\_list\_fail = list(data\_df.loc[(data\_df['SerialNumber'] == true\_fail\_sn\_value), 'StartTime'].tolist())

date\_time\_only\_day\_list\_fail = pd.to\_datetime(str(date\_time\_list\_fail[0])).strftime("%Y/%m/%d")

product\_list\_fail = list(data\_df.loc[(data\_df['SerialNumber'] == true\_fail\_sn\_value), 'Product'].tolist())

n\_true\_fail\_value\_by\_date\_list.append(str(date\_time\_only\_day\_list\_fail)+' '+ str(product\_list\_fail[0]))

except Exception as e:

print('list no retest sn error'+ str(e))

n\_true\_fail\_value\_set = set(n\_true\_fail\_value\_list)

for n\_true\_fail\_item in n\_true\_fail\_value\_set:

tmp\_fail\_1 = []

# print('-aab fail>>>',n\_true\_fail\_item,n\_true\_fail\_value\_list.count(n\_true\_fail\_item))

tmp\_fail\_1.append(n\_true\_fail\_item)

tmp\_fail\_1.append(n\_true\_fail\_value\_list.count(n\_true\_fail\_item))

n\_total\_count\_by\_station\_list.append(tmp\_fail\_1)

n\_true\_fail\_value\_by\_version\_set = set(n\_true\_fail\_value\_by\_version\_list)

for n\_true\_fail\_item in n\_true\_fail\_value\_by\_version\_set:

tmp\_fail\_1 = []

# print('-aab fail>>>',n\_true\_fail\_item,n\_true\_fail\_value\_list.count(n\_true\_fail\_item))

tmp\_fail\_1.append(n\_true\_fail\_item)

tmp\_fail\_1.append(n\_true\_fail\_value\_by\_version\_list.count(n\_true\_fail\_item))

n\_total\_count\_by\_version\_list.append(tmp\_fail\_1)

n\_true\_fail\_value\_by\_date\_set = set(n\_true\_fail\_value\_by\_date\_list)

for n\_true\_fail\_item in n\_true\_fail\_value\_by\_date\_set:

tmp\_pass\_1 = []

tmp\_pass\_1.append(n\_true\_fail\_item)

tmp\_pass\_1.append(n\_true\_fail\_value\_by\_date\_list.count(n\_true\_fail\_item))

n\_total\_count\_by\_date\_list.append(tmp\_pass\_1)

# retest

n\_retest\_value\_by\_station\_list = []

n\_retest\_value\_by\_version\_list = []

n\_retest\_value\_by\_date\_list = []

# n\_total\_count\_by\_station\_list\_only\_retest = []

# n\_total\_count\_by\_version\_list\_only\_retest = []

for total\_retest\_sn\_value in total\_retest\_sn:

try:

station\_id\_list\_retest= list(data\_df.loc[(data\_df['SerialNumber'] == total\_retest\_sn\_value), station\_id\_key].tolist())

if slot\_id\_key == '':

n\_retest\_value\_by\_station\_list.append(str(station\_id\_list\_retest[0]))

else:

slot\_id\_list\_retest = list(data\_df.loc[(data\_df['SerialNumber'] == total\_retest\_sn\_value), slot\_id\_key].tolist())

n\_retest\_value\_by\_station\_list.append(str(station\_id\_list\_retest[0])+' '+str(slot\_id\_list\_retest[0]))

version\_list\_retest = list(data\_df.loc[(data\_df['SerialNumber'] == total\_retest\_sn\_value), 'Version'].tolist())

n\_retest\_value\_by\_version\_list.append(str(version\_list\_retest[0]))

date\_time\_list\_fail = list(data\_df.loc[(data\_df['SerialNumber'] == total\_retest\_sn\_value), 'StartTime'].tolist())

date\_time\_only\_day\_list\_fail = pd.to\_datetime(str(date\_time\_list\_fail[0])).strftime("%Y/%m/%d")

product\_list\_fail = list(data\_df.loc[(data\_df['SerialNumber'] == total\_retest\_sn\_value), 'Product'].tolist())

n\_retest\_value\_by\_date\_list.append(str(date\_time\_only\_day\_list\_fail)+' '+ str(product\_list\_fail[0]))

except Exception as e:

print('list no retest sn error'+ str(e))

n\_retest\_value\_by\_station\_set = set(n\_retest\_value\_by\_station\_list)

for n\_retest\_item in n\_retest\_value\_by\_station\_set:

tmp\_retest\_1 = []

tmp\_retest\_1.append(n\_retest\_item)

tmp\_retest\_1.append(n\_retest\_value\_by\_station\_list.count(n\_retest\_item))

n\_total\_count\_by\_station\_list.append(tmp\_retest\_1)

# n\_total\_count\_by\_station\_list\_only\_retest.append(tmp\_retest\_1)

n\_retest\_value\_by\_version\_set = set(n\_retest\_value\_by\_version\_list)

for n\_retest\_item in n\_retest\_value\_by\_version\_set:

tmp\_retest\_1 = []

tmp\_retest\_1.append(n\_retest\_item)

tmp\_retest\_1.append(n\_retest\_value\_by\_version\_list.count(n\_retest\_item))

n\_total\_count\_by\_version\_list.append(tmp\_retest\_1)

# n\_total\_count\_by\_version\_list\_only\_retest.append(tmp\_retest\_1)

n\_retest\_value\_by\_date\_set = set(n\_retest\_value\_by\_date\_list)

for n\_reset\_item in n\_retest\_value\_by\_date\_set:

tmp\_pass\_1 = []

tmp\_pass\_1.append(n\_reset\_item)

tmp\_pass\_1.append(n\_retest\_value\_by\_date\_list.count(n\_reset\_item))

n\_total\_count\_by\_date\_list.append(tmp\_pass\_1)

df\_station\_slot\_id = pd.DataFrame(n\_total\_count\_by\_station\_list, columns=['Station ID Slot ID','TOTAL'])

# df\_station\_slot\_id\_only\_retest = pd.DataFrame(n\_total\_count\_by\_station\_list\_only\_retest, columns=['Station ID Slot ID','RETEST'])

# print('--->2>>>',n\_total\_count\_by\_station\_list\_only\_retest,df\_station\_slot\_id\_only\_retest)

f\_csv = open('/tmp/CPK\_Log/retest/..total\_count\_by\_station\_slot\_id.csv','w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Station ID Slot ID','TOTAL'])

for groupName, groupDf in df\_station\_slot\_id.groupby(by='Station ID Slot ID'):

total\_num = 0

for index,row in groupDf.iterrows():

total\_num = total\_num+row['TOTAL']

if slot\_id\_key == '':

station\_name\_1 = cut\_station\_name(str(groupName))

csv\_writer.writerow([str(station\_name\_1),str(total\_num)])

else:

group\_name\_sub = groupName.rsplit(' ',1)

station\_name\_1 = cut\_station\_name(str(group\_name\_sub[0]))

slot\_name\_1 = str(group\_name\_sub[1])

csv\_writer.writerow([str(station\_name\_1)+' '+slot\_name\_1,str(total\_num)])

f\_csv.close()

df\_version = pd.DataFrame(n\_total\_count\_by\_version\_list, columns=['Version','TOTAL'])

f\_csv = open('/tmp/CPK\_Log/retest/..total\_count\_by\_version.csv','w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Version','TOTAL'])

for groupName, groupDf in df\_version.groupby(by='Version'):

total\_num = 0

for index,row in groupDf.iterrows():

total\_num = total\_num+row['TOTAL']

csv\_writer.writerow([str(groupName),str(total\_num)])

f\_csv.close()

f\_date = pd.DataFrame(n\_total\_count\_by\_date\_list, columns=['Date Product','TOTAL'])

f\_csv = open('/tmp/CPK\_Log/retest/..total\_count\_by\_date\_product.csv','w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Date','Date Product','TOTAL'])

# print('------pdate product--')

for groupName, groupDf in f\_date.groupby(by='Date Product'):

total\_num = 0

for index,row in groupDf.iterrows():

total\_num = total\_num+row['TOTAL']

group\_date = groupName.rsplit(' ',1)

csv\_writer.writerow([str(group\_date[0]),str(groupName),str(total\_num)])

f\_csv.close()

if slot\_id\_key == '':

yield\_path = '/tmp/CPK\_Log/temp/yield\_rate\_param.csv'

isExists=os.path.exists(yield\_path)

if not isExists:

for x in range(30):

time.sleep(2)

isExists=os.path.exists(yield\_path)

if isExists:

break

with open(yield\_path,'r') as csvfile:

reader = csv.reader(csvfile)

yield\_rows = [row for row in reader]

f\_csv = open('/tmp/CPK\_Log/retest/..total\_count\_by\_station\_slot\_id.csv','w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Station ID Slot ID','TOTAL'])

n\_count = 0

for yield\_row in yield\_rows:

if n\_count>1:

csv\_writer.writerow([str(cut\_station\_name(yield\_row[0])),str(yield\_row[1])])

n\_count = n\_count +1

f\_csv.close()

def func\_pie(pct, allvals):

absolute = int(round(pct/100.\*np.sum(allvals),1))

return "{:.1f}% ({:d} )".format(pct, absolute)

def pie\_retest\_plot(csv\_pah,pic\_path):

# df\_raw = pd.read\_csv('/Users/RyanGao/Downloads/ParetoChart-master-2/pie.csv')

# df = df\_raw.groupby('class').size().reset\_index(name='counts')

# Draw Plot

df = pd.read\_csv(csv\_pah)

fig, ax = plt.subplots(figsize=(10, 6), subplot\_kw=dict(aspect="equal"), dpi= 150)

categories = ['1st PASS','Retest PASS','FAILED'] #df['class']

data = [int(max(df[categories[0]])),int(max(df[categories[1]])),int(max(df[categories[2]]))]

explode = [0,0.1,0]

wedges, texts, autotexts = ax.pie(data,

autopct=lambda pct: func\_pie(pct, data),

textprops=dict(color="w"),

colors= ['lightskyblue','lightyellow','pink'], #plt.cm.Dark2.colors,

startangle=140,

explode=explode)

# Decoration

ax.legend(wedges, categories, title="Color Description:", loc="center left", bbox\_to\_anchor=(1, 0, 0.5, 1))

plt.setp(autotexts, size=10, weight=700,color = 'blue',alpha = 0.7)

ax.set\_title("Pie Chart for Yield")

plt.tight\_layout()

plt.savefig(pic\_path, dpi=150)

# plt.show()

def yield\_donut(csv\_pah,pic\_path):

df = pd.read\_csv(csv\_pah)

categories = ['1st PASS','Retest PASS','FAILED','PASSED','No Retest PASS','TOTAL'] #1st PASS,Retest PASS,FAILED,PASSED,No Retest PASS,TOTAL

the\_first\_pass\_count = max(df[categories[0]])

the\_second\_pass\_count = max(df[categories[1]])

fail\_count = max(df[categories[2]])

total\_count = max(df[categories[5]])

first\_pass\_rate = float(the\_first\_pass\_count)/float(total\_count)

send\_pass\_rate = float(the\_second\_pass\_count)/float(total\_count)

fail\_rate = float(fail\_count)/float(total\_count)

fig, ax = plt.subplots(figsize=(12, 9), subplot\_kw=dict(aspect="equal"))

recipe = ["1st PASS:"+str(round(first\_pass\_rate\*100,2))+"%",

"Retest PASS:"+str(round(send\_pass\_rate\*100,2))+"%",

"AAB FAILED:"+str(round(fail\_rate\*100,2))+"%"]

data = [first\_pass\_rate,send\_pass\_rate,fail\_rate]

colors = ['limegreen','yellow','red']

wedges, texts = ax.pie(data, wedgeprops=dict(width=0.5), startangle=-40,colors=colors,)

lend=[u'1st PASS:'+str(the\_first\_pass\_count)+"/"+str(total\_count),u'Retest PASS:'+str(the\_second\_pass\_count)+"/"+str(total\_count),u'AAB FAILED:'+str(fail\_count)+"/"+str(total\_count)]

bbox\_props = dict(boxstyle="square,pad=0.3", fc="w", ec="k", lw=0.72)

kw = dict(xycoords='data', textcoords='data', arrowprops=dict(arrowstyle="-"),bbox=bbox\_props, zorder=0, va="center")

for i, p in enumerate(wedges): # 遍历每一个扇形

ang = (p.theta2 - p.theta1)/2. + p.theta1 # 锁定扇形夹角的中间位置，对应的度数为ang

y = np.sin(np.deg2rad(ang)) # np.sin()求正弦

x = np.cos(np.deg2rad(ang)) # np.cos()求余弦

horizontalalignment = {-1: "right", 1: "left"}[int(np.sign(x))]

connectionstyle = "angle,angleA=0,angleB={}".format(ang) # 参数connectionstyle用于控制箭头连接时的弯曲程度

kw["arrowprops"].update({"connectionstyle": connectionstyle}) # 将connectionstyle更新至参数集kw的参数arrowprops中

ax.annotate(recipe[i], size=15, xy=(x, y), xytext=(1\*np.sign(x), 1.1\*y),horizontalalignment=horizontalalignment, \*\*kw)

ax.set\_title("Build Yield Chart",fontsize=20)

plt.legend(lend,loc="center",fontsize=15,bbox\_to\_anchor=(0.5,0.5),borderaxespad=0.3,edgecolor='silver',shadow=True,labelspacing=0.5)

plt.tight\_layout()

plt.savefig(pic\_path, dpi=150)

def retest\_top\_5\_to\_csv(retest\_csv\_path,pie\_retest\_csv\_path,retest\_top\_5\_path):

df = pd.read\_csv(retest\_csv\_path)

occurrences = df.groupby('RetestItem').count().reset\_index()

n\_count = df['RetestItem'].unique().\_\_len\_\_()+1

tmp = occurrences.sort\_values('SerialNumber', ascending=False)

item\_names = tmp['RetestItem'].tolist()

retest\_counts = tmp['SerialNumber'].tolist()

total\_retest\_count = sum(retest\_counts)

if len(item\_names)>5:

item\_names = item\_names[0:5]

if len(retest\_counts)>5:

retest\_counts = retest\_counts[0:5]

# df = pd.read\_csv('/tmp/CPK\_Log/retest/.pie\_retest.csv')

df = pd.read\_csv(pie\_retest\_csv\_path)

if len(df['TOTAL']) == 0:

input\_total\_count = 0

else:

input\_total\_count= max(df['TOTAL'])

f\_csv = open(retest\_top\_5\_path,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Input (TOTAL)',str(input\_total\_count),'\*\*\*\*'])

csv\_writer.writerow(['Top 5 Retest Items','Qty','Fail rate'])

ii = 0

for item\_name in item\_names:

perct = round((float(retest\_counts[ii])/float(input\_total\_count))\*100,3)

csv\_writer.writerow([str(item\_names[ii]),str(retest\_counts[ii]),str(perct)+'%'])

ii = ii+1

f\_csv.close()

def fail\_top\_5\_to\_csv(fail\_csv\_path,pie\_retest\_csv\_path,fail\_top\_5\_path):

df = pd.read\_csv(fail\_csv\_path)

occurrences = df.groupby('FailItem').count().reset\_index()

n\_count = df['FailItem'].unique().\_\_len\_\_()+1

tmp = occurrences.sort\_values('SerialNumber', ascending=False)

item\_names = tmp['FailItem'].tolist()

fail\_counts = tmp['SerialNumber'].tolist()

total\_fail\_count = sum(fail\_counts)

if len(item\_names)>5:

item\_names = item\_names[0:5]

if len(fail\_counts)>5:

fail\_counts = fail\_counts[0:5]

df = pd.read\_csv(pie\_retest\_csv\_path)

input\_total\_count= max(df['TOTAL'])

f\_csv = open(fail\_top\_5\_path,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Input (TOTAL)',str(input\_total\_count),'\*\*\*\*'])

csv\_writer.writerow(['Top 5 Fail Items','Qty','Fail rate'])

ii = 0

for item\_name in item\_names:

perct = round((float(fail\_counts[ii])/float(input\_total\_count))\*100,3)

csv\_writer.writerow([str(item\_names[ii]),str(fail\_counts[ii]),str(perct)+'%'])

ii = ii+1

f\_csv.close()

def find\_diff\_intwo\_list(list1,list2):

'''

:param list1: 列表1

:param list2: 列表2

:return:

'''

same,diff=[],[]

seq=list(set(list2))

for i in list(set(list1)):

if i not in list2:

diff.append(i)

else:

same.append(i)

for j in same:

seq.remove(j)

# print("same is {},diff is {}".format(same,diff+seq))

return diff+seq

def retest\_vs\_station\_slot\_id\_csv(csv\_data\_path,header\_item\_path,keyword,csv\_path\_output):

data0 = pd.read\_csv('/tmp/CPK\_Log/retest/..total\_count\_by\_station\_slot\_id.csv')

df0 = pd.DataFrame(data0)

data1 = pd.read\_csv(csv\_data\_path)

df = pd.DataFrame(data1)

with open(header\_item\_path,'r') as csvfile:

reader = csv.reader(csvfile)

header\_rows = [row for row in reader]

list\_station\_id = []

list\_retest\_count = []

list\_total\_count = []

list\_rate\_percent = []

for groupName, groupDf in df.groupby(by=keyword[0]): #'Station ID'

if slot\_id\_key == '':

n\_count = 0

for index,row in groupDf.iterrows():

n\_count = n\_count + 1

tmp\_name = cut\_station\_name(str(groupName))

list\_station\_id.append(tmp\_name)

list\_retest\_count.append(n\_count)

tt\_list = list(df0.loc[(df0['Station ID Slot ID'] == tmp\_name), 'TOTAL'].tolist())

try:

xx\_tmp = tt\_list[0]

except Exception as e:

# print('for tt\_list[0]:',e)

xx\_tmp = 0

list\_total\_count.append(str(xx\_tmp))

try:

if int(xx\_tmp) == 0:

list\_rate\_percent.append('0')

else:

x\_percent = round(int(n\_count)/int(xx\_tmp)\*100,3)

list\_rate\_percent.append(str(x\_percent))

except Exception as e:

print('error convert int',str(e))

list\_rate\_percent.append('0')

else:

for groupName\_sub, groupDf\_sub in groupDf.groupby(by=keyword[1]):

n\_count = 0

for index,row in groupDf\_sub.iterrows():

n\_count = n\_count + 1

tmp\_name = cut\_station\_name(str(groupName)) + ' '+str(groupName\_sub)

list\_station\_id.append(tmp\_name)

list\_retest\_count.append(n\_count)

tt\_list = list(df0.loc[(df0['Station ID Slot ID'] == tmp\_name), 'TOTAL'].tolist())

try:

xx\_tmp = tt\_list[0]

except Exception as e:

# print('for tt\_list[0]:',e)

xx\_tmp = 0

list\_total\_count.append(str(xx\_tmp))

try:

if int(xx\_tmp) == 0:

list\_rate\_percent.append('0')

else:

x\_percent = round(int(n\_count)/int(xx\_tmp)\*100,3)

list\_rate\_percent.append(str(x\_percent))

except Exception as e:

print('error convert int',str(e))

list\_rate\_percent.append('0')

f\_csv = open(csv\_path\_output,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

csv\_writer.writerow(['Index','Retest\_Count','TOTAL','RATE'])

n\_index = 0

for x\_row in list\_station\_id:

csv\_writer.writerow([str(list\_station\_id[n\_index]),str(list\_retest\_count[n\_index]),str(list\_total\_count[n\_index]),str(list\_rate\_percent[n\_index])])

n\_index = n\_index+1

diff\_list = find\_diff\_intwo\_list(list\_station\_id,header\_rows[1])

for diff\_value in diff\_list:

tt\_list = list(df0.loc[(df0['Station ID Slot ID'] == diff\_value), 'TOTAL'].tolist())

try:

xx\_tmp = tt\_list[0]

except Exception as e:

# print('for tt\_list[0]:',e)

xx\_tmp = 0

csv\_writer.writerow([str(diff\_value),'0',str(xx\_tmp),'0'])

f\_csv.close()

def is\_number(s):

try:

float(s)

return True

except ValueError:

pass

try:

import unicodedata

unicodedata.numeric(s)

return True

except (TypeError, ValueError):

pass

return False

def retest\_breakdown\_by\_fixture\_to\_csv(csv\_data\_path,pie\_retest\_csv\_path,csv\_path\_output):

df\_count = pd.read\_csv(pie\_retest\_csv\_path)

input\_total\_count= max(df\_count['TOTAL'])

data1 = pd.read\_csv(csv\_data\_path)

df = pd.DataFrame(data1)

f\_csv = open(csv\_path\_output,'w',encoding='utf-8')

csv\_writer = csv.writer(f\_csv)

row\_count = len(df['Date'])

if row\_count == 0:

csv\_writer.writerow(['Fixture ID','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*'])

csv\_writer.writerow(['Input Qty',str(input\_total\_count),'\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*'])

csv\_writer.writerow(['Retest Item','Qty', 'Retest rate','Slot ID','UUT S/N','Measured Value','[LSL;USL]'])

f\_csv.close()

return

data0 = pd.read\_csv('/tmp/CPK\_Log/temp/yield\_rate\_param.csv')

df0 = pd.DataFrame(data0)

for groupName, groupDf in df.groupby(by='Station ID'): #'Station ID'

try:

tt\_list = list(df0.loc[(df0['Station ID'] == groupName), 'Input (TOTAL)'].tolist())

if len(tt\_list) == 0:

tt\_list.append('0')

except Exception as e:

print('error ,', str(e))

tt\_list.append('0')

csv\_writer.writerow(['Fixture ID',str(groupName),'\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*'])

csv\_writer.writerow(['Input Qty',str(tt\_list[0]),'\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*','\*\*\*\*\*'])

csv\_writer.writerow(['Retest Item','Qty', 'Retest rate','Slot ID','UUT S/N','Measured Value','[LSL;USL]'])

occurrences = groupDf.groupby('RetestItem').count().reset\_index()

tmp = occurrences.sort\_values('SerialNumber', ascending=False)

item\_name\_list = tmp['RetestItem'].tolist()[0:5]

if len(item\_name\_list)>5:

item\_name\_list = item\_name\_list[0:5]

# n\_count\_list = tmp['SerialNumber'].tolist()[0:5]

# if len(n\_count\_list)>5:

# n\_count\_list = n\_count\_list[0:5]

nn\_index = 0

for item\_name in item\_name\_list:

n\_retest\_count = 0

serial\_number\_list =[]

slot\_id\_list = []

measured\_value\_list = []

limit\_range\_list = []

for index,row in groupDf.iterrows():

if row['RetestItem'] == item\_name:

serial\_number\_list.append(str(row['SerialNumber']))

slot\_id\_list.append(str(row['Slot ID']))

meas\_val = row['Measured Value']

if is\_number(meas\_val):

meas\_val = round(meas\_val, 3)

if str(meas\_val) == '' or str(meas\_val) == 'nan' or meas\_val == None or str(meas\_val) == 'NaN':

meas\_val = ''

measured\_value\_list.append(str(meas\_val))

limit\_range\_list.append(str(row['[LSL;USL]']))

n\_retest\_count = n\_retest\_count +1

retest\_qty = str(n\_retest\_count)

retest\_rate = round(float(retest\_qty)/float(input\_total\_count)\*100,2)

slot\_id\_str = ';'.join(slot\_id\_list)

serial\_number\_str = ';'.join(serial\_number\_list)

measured\_value\_str = ';'.join(measured\_value\_list)

try:

limit\_range\_str = str(limit\_range\_list[0])

if limit\_range\_str == 'NaN' or limit\_range\_str == 'nan':

limit\_range\_str = ''

except Exception as e:

print('error limit range',str(e))

limit\_range\_str = ''

csv\_writer.writerow([str(item\_name),str(retest\_qty), str(retest\_rate)+'%',slot\_id\_str,serial\_number\_str, measured\_value\_str,limit\_range\_str])

nn\_index = nn\_index + 1

if nn\_index<5:

for i in range(5-nn\_index):

csv\_writer.writerow(['','', '','', '','',''])

csv\_writer.writerow(['','', '','', '','',''])

f\_csv.close()

def run(n):

while True:

try:

print("wait for retest plot client ...")

zmqMsg = socket.recv()

socket.send(b'retest\_plot\_sendback')

if len(zmqMsg)>0:

keyMsg = zmqMsg.decode('utf-8')

print("message from retest plot client:", keyMsg)

msg =keyMsg.split("$$")

if len(msg)>3:

if msg[0] == 'retest\_plot':

all\_csv\_path = msg[1]

retest\_csv\_path = msg[2]

pie\_retest\_csv\_path = msg[3]

fail\_csv\_path = '/tmp/CPK\_Log/retest/.fail\_csv.csv'

header\_item\_path = '/tmp/CPK\_Log/retest/.header\_info\_csv.csv'

print('>generateRetestCSV start')

generateRetestCSV(all\_csv\_path,retest\_csv\_path,pie\_retest\_csv\_path,fail\_csv\_path,header\_item\_path)

print('>generateRetestCSV finished')

title = 'Pareto Chart for Top 5 Retest'

color = 'steelblue'

pareto\_plot(retest\_csv\_path, color, title, x='RetestItem', y='SerialNumber', show\_pct\_y=False,saveas="/tmp/CPK\_Log/retest/retest\_pareto.png")

print('>pareto\_plot Pareto Chart for Top 5 Retest finished')

title = 'Pareto Chart for Top 5 Fail'

color = 'pink'

pareto\_plot(fail\_csv\_path, color, title, x='FailItem', y='SerialNumber', show\_pct\_y=False,saveas="/tmp/CPK\_Log/retest/fail\_pareto.png")

print('>pareto\_plot Pareto Chart for Top 5 Fail finished')

# pie\_retest\_plot(pie\_retest\_csv\_path,'/tmp/CPK\_Log/retest/retest\_pie.png')

yield\_donut(pie\_retest\_csv\_path,'/tmp/CPK\_Log/retest/retest\_pie.png')

print('>yield\_donut finished')

summary\_retest\_path\_output = '/tmp/CPK\_Log/retest/.summary\_retest.csv'

summary\_retest\_csv(retest\_csv\_path,header\_item\_path,summary\_retest\_path\_output)

print('>summary\_retest\_csv finished')

daily\_retest\_summary\_plot(summary\_retest\_path\_output,'/tmp/CPK\_Log/retest/daily\_all\_retest\_summary.png','/tmp/CPK\_Log/retest/daily\_retest\_summary.png')

print('>daily\_retest\_summary\_plot finished')

retest\_vs\_station\_id\_path\_output = '/tmp/CPK\_Log/retest/.retest\_vs\_station\_id.csv'

keyword = ['Station ID','Slot ID']

retest\_vs\_station\_slot\_id\_csv(retest\_csv\_path,header\_item\_path,keyword,retest\_vs\_station\_id\_path\_output)

print('>retest\_vs\_station\_slot\_id\_csv finished')

title\_name = 'Retest rate vs Station ID & Slot ID'

retest\_vs\_plot(retest\_vs\_station\_id\_path\_output,title\_name,pie\_retest\_csv\_path,'/tmp/CPK\_Log/retest/retest\_vs\_station\_id.png',False)

print('>retest\_vs\_plot Retest rate vs Station ID & Slot ID finished')

retest\_vs\_version\_path\_output = '/tmp/CPK\_Log/retest/.retest\_vs\_version.csv'

keyword = 'Version'

retest\_vs\_Version\_csv(retest\_csv\_path,header\_item\_path,keyword,retest\_vs\_version\_path\_output)

print('>retest\_vs\_Version\_csv finished')

title\_name = 'Retest rate vs SW version'

retest\_vs\_plot(retest\_vs\_version\_path\_output,title\_name,pie\_retest\_csv\_path,'/tmp/CPK\_Log/retest/retest\_vs\_version.png',True)

print('>retest\_vs\_plot Retest rate vs SW version finished')

retest\_top\_5\_path = '/tmp/CPK\_Log/retest/retest\_item\_overall.csv'

retest\_top\_5\_to\_csv(retest\_csv\_path,pie\_retest\_csv\_path,retest\_top\_5\_path)

print('>retest\_top\_5\_to\_csv finished')

fail\_top\_5\_path = '/tmp/CPK\_Log/retest/fail\_item\_overall.csv'

fail\_top\_5\_to\_csv(fail\_csv\_path,pie\_retest\_csv\_path,fail\_top\_5\_path)

print('>fail\_top\_5\_to\_csv finished')

retest\_breakdown\_path = '/tmp/CPK\_Log/retest/retest\_breakdown\_fixture.csv'

retest\_breakdown\_by\_fixture\_to\_csv(retest\_csv\_path,pie\_retest\_csv\_path,retest\_breakdown\_path)

print('>retest\_breakdown\_by\_fixture\_to\_csv finished')

filelogname = '/tmp/CPK\_Log/retest/.retest\_plot.txt'

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,retest plot is finished")

print('>all done retest plot exist.')

return

else:

time.sleep(0.05)

except Exception as e:

print('error retest plot rate:',e)

filelogname = '/tmp/CPK\_Log/retest/.retest\_plot.txt'

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,retest plot report error: " + str(e))

if \_\_name\_\_ == '\_\_main\_\_':

run(0)

# for count,customer in enumerate(df["Supplier"].unique().tolist()):

# savename = customer+" Pareto.png"

# print(savename)

# pareto\_plot(df, x='Reason', y='Supplier', customer=customer,show\_pct\_y=False,saveas=savename)

#! /usr/bin/env python3

# --\*-- coding: utf-8 ---\*---

import sys,os,time,math,re

import time

import threading

import datetime

from pytz import timezone

import pytz

BASE\_DIR=os.path.dirname(os.path.abspath(\_\_file\_\_))

sys.path.insert(0,BASE\_DIR+'/site-packages/')

try:

import csv

except Exception as e:

print('e---->',e)

# print('python import ---->matplotlib')

try:

import matplotlib

matplotlib.use("Agg")

import matplotlib.pyplot as plt

except Exception as e:

print('e---->',e)

# print('python import ----> matplotlib.colors')

try:

import matplotlib.colors as colors

except Exception as e:

print('e---->',e)

# print('python import ----> FontProperties')

try:

from matplotlib.font\_manager import FontProperties

except Exception as e:

print('e---->',e)

# print('python import ----> numpy')

try:

import numpy as np

except Exception as e:

print('e--->',e)

# print('python import ----> pandas')

try:

import pandas as pd

except Exception as e:

print('e--->',e)

# print('python import ----> openpyxl')

try:

import openpyxl

except Exception as e:

print('import openpyxl error:',e)

# print('python import ----> xlsxwriter')

try:

import xlsxwriter

except Exception as e:

print('import xlsxwriter error:',e)

# print('python import ----> diptest')

try:

import diptest

except Exception as e:

print('import diptest error:',e)

try:

import zmq

except Exception as e:

print('import zmq error:',e)

# print('python import ----> zmg')

try:

import redis

except Exception as e:

print('import redis error:',e)

# print('python import ----> redis')

print(sys.getdefaultencoding())

redisClient = redis.Redis(host='localhost', port=6379, db=0)

context = zmq.Context()

socket = context.socket(zmq.REP)

socket.setsockopt(zmq.LINGER,0)

socket.bind("tcp://127.0.0.1:3130")

filelogname = '/tmp/CPK\_Log/temp/.excel.txt'

filelognamehash = '/tmp/CPK\_Log/temp/.excel\_hash.txt'

p\_val\_checked = ''

# def correlation(message):

# print("this function is generate correlation plot......")

# val = r.get(message)

# # time.sleep(5) #测试python 执行时间 5s

# if val:

# return val

# else:

# return b'None'

def get\_redis\_data(zmqMsg):

tb = redisClient.get(zmqMsg)

tb\_data=[]

if tb:

tb=tb.decode('utf-8')

tb=tb.split("\n")

tb=(tb[1:-1]) #去掉数据库首尾元素

for i in tb:

k=re.sub('\"','',i) #去掉数据库引号

h=re.sub(',','',k) #去掉数据库逗号

m=h.strip() #去掉数据库首尾空白

if is\_number(m):

tb\_data.append(eval(m)) #去掉数字的引号

else:

tb\_data.append(m)

else:

tb\_data.append('')

return tb\_data

def is\_number(s):

try:

float(s)

return True

except ValueError:

pass

try:

import unicodedata

unicodedata.numeric(s)

return True

except (TypeError, ValueError):

pass

return False

def open\_one\_item\_csv(event,all\_csv\_path,data\_select,remove\_fail):

tmp\_lst = []

with open(all\_csv\_path, 'r') as f:

reader = csv.reader(f)

i = 1

for row in reader:

# print(row[0].lower())

if row[0].lower().find('fct') != -1:

# print("FW version---->")

pass

elif row[0].lower().find('display name') != -1:

pass

elif row[0].lower().find('pdca priority') != -1:

pass

elif row[0].lower().find('upper limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('lower limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('measurement unit') != -1: # "Measurement Unit ----->" in row:

pass

elif row[0].lower().find('site') != -1:

tmp\_lst.append(row)

else:

tmp\_lst.append(row)

i = i + 1

# print("index---->", tmp\_lst[0])

header\_list = tmp\_lst[0][:]#

temp\_header\_list = tmp\_lst[0]

if str(header\_list[12]).lower() == 'fixture channel id' or str(header\_list[13]).lower() == 'fixture channel id':

temp\_header\_list[14]= 'slot\_id' #rename fixture channel id

if header\_list[12] == header\_list[13]:

temp\_header\_list[13] = temp\_header\_list[13]+'\_2'

temp\_df = pd.DataFrame(tmp\_lst[1:], columns=temp\_header\_list)

correlation\_header\_df =temp\_df[0:2]

df = pd.DataFrame(tmp\_lst[1:], columns=tmp\_lst[0])

header\_df =df[0:2]

# print('header\_df before--->', header\_df)

data\_df = df[2:]

correlation\_data\_df = data\_df

# print('data\_df before--->', data\_df)

# print('99999888--->',type(correlation\_data\_df.columns),correlation\_data\_df.columns[12],correlation\_data\_df.columns[13])

if correlation\_data\_df.columns[12] != correlation\_data\_df.columns[13]:

print(' one\_item\_plot before remove empty--->', len(correlation\_data\_df.values.tolist()))

correlation\_data\_df=correlation\_data\_df[~correlation\_data\_df[correlation\_data\_df.columns[12]].isin([''])]#Remove SN Empty

correlation\_data\_df=correlation\_data\_df[~correlation\_data\_df[correlation\_data\_df.columns[13]].isin([''])]#Remove SN Empty

print(' one\_item\_plot after remove empty--->', len(correlation\_data\_df.values.tolist()))

print('csv data row number before remove SN empty--->', len(correlation\_data\_df.values.tolist()))

correlation\_data\_df=correlation\_data\_df[~correlation\_data\_df['SerialNumber'].isin([''])]#Remove SN Empty

print('csv data row number after remove SN empty--->', len(correlation\_data\_df.values.tolist()))

print('csv data row number before remove fail--->', len(correlation\_data\_df.values.tolist()))

if remove\_fail== 'yes':

correlation\_data\_df=correlation\_data\_df[correlation\_data\_df['Test Pass/Fail Status'].isin(['PASS'])]

# data\_df=data\_df[~data\_df['Test Pass/Fail Status'].isin(['FAIL'])]

print('csv data row number after remove fail--->', len(correlation\_data\_df.values.tolist()))

print('csv data row number before remove retest--->', len(correlation\_data\_df.values.tolist()))

# print('correlation\_data\_df--->',type(correlation\_data\_df),len(correlation\_data\_df.values.tolist()))

if data\_select == 'first':

correlation\_data\_df = correlation\_data\_df.sort\_values(axis=0,by=['StartTime'],ascending='True')

correlation\_data\_df.drop\_duplicates(['SerialNumber'],keep='first',inplace=True)

elif data\_select == 'last':

correlation\_data\_df = correlation\_data\_df.sort\_values(axis=0,by=['StartTime'],ascending='True')

correlation\_data\_df.drop\_duplicates(['SerialNumber'],keep='last',inplace=True)

elif data\_select == 'no\_retest':

correlation\_data\_df.drop\_duplicates(['SerialNumber'],keep=False,inplace=True)

elif data\_select == 'all':

pass

print('csv data row number after remove retest--->', len(correlation\_data\_df.values.tolist()))

print("==========================<<2222>>>>>>here")

correlation\_start\_time\_first,correlation\_start\_time\_last = '',''

print("==========================<<>>>>>>here")

if len(correlation\_data\_df.values.tolist()) >2:

correlation\_start\_time\_l = correlation\_data\_df['StartTime'].values.tolist() #StartTime

correlation\_start\_time\_first = min(correlation\_start\_time\_l)

correlation\_start\_time\_last = max(correlation\_start\_time\_l)

print('<correlation first time -- last time>',correlation\_start\_time\_first,correlation\_start\_time\_last)

if header\_list[12] == header\_list[13]:

df\_correlation = correlation\_header\_df.append(correlation\_data\_df)

else:

df\_correlation = header\_df.append(correlation\_data\_df)

print('csv data row number before remove SN empty--->', len(data\_df.values.tolist()))

data\_df=data\_df[~data\_df['SerialNumber'].isin([''])]#Remove SN Empty

print('csv data row number after remove SN empty--->', len(data\_df.values.tolist()))

print('csv data row number before remove fail--->', len(data\_df.values.tolist()))

if remove\_fail== 'yes':

data\_df=data\_df[data\_df['Test Pass/Fail Status'].isin(['PASS'])]

# data\_df=data\_df[~data\_df['Test Pass/Fail Status'].isin(['FAIL'])]

print('csv data row number after remove fail--->', len(data\_df.values.tolist()))

print('csv data row number before remove retest--->', len(data\_df.values.tolist()))

if data\_select == 'first':

data\_df = data\_df.sort\_values(axis=0,by=['StartTime'],ascending='True')

data\_df.drop\_duplicates(['SerialNumber'],keep='first',inplace=True)

elif data\_select == 'last':

data\_df = data\_df.sort\_values(axis=0,by=['StartTime'],ascending='True')

data\_df.drop\_duplicates(['SerialNumber'],keep='last',inplace=True)

elif data\_select == 'no\_retest':

data\_df.drop\_duplicates(['SerialNumber'],keep=False,inplace=True)

elif data\_select == 'all':

pass

print('csv data row number after remove retest--->', len(data\_df.values.tolist()))

start\_time\_l = data\_df['StartTime'].values.tolist() #StartTime

# print('start\_time\_l:',start\_time\_l)

if len(start\_time\_l)>0:

start\_time\_first = min(start\_time\_l)

start\_time\_last = max(start\_time\_l)

else:

start\_time\_first = ''

start\_time\_last = ''

# print('<first time -- last time>',start\_time\_first,start\_time\_last)

if header\_list[12] == header\_list[13]:

df = correlation\_header\_df.append(data\_df)

else:

df = header\_df.append(data\_df)

# print("index1---->", header\_list)

# print('df after--->',df\_correlation.columns.values.tolist(), df.\_stat\_axis.values.tolist(),df)

# print('df.values ---->', df.values)#array([[ ]])

return header\_list,df,df\_correlation,start\_time\_first,start\_time\_last,correlation\_start\_time\_first,correlation\_start\_time\_last

def clear\_files(path):

for root,dirs,files in os.walk(path):

for file in files:

os.remove(path+'/'+file)

def creat\_excel\_report\_file(path,file,cpk\_lsl,cpk\_usl,event,fail\_plot\_to\_excel):

if event == 'keynote-report':

excel\_file\_path = '/tmp/CPK\_Log/temp/cpk.xlsx'

else:

excel\_file\_path = path + file

global excel\_report\_name

excel\_report\_name = excel\_file\_path

global hash\_csv\_path

hash\_csv\_path = path

book = xlsxwriter.Workbook(excel\_file\_path)#'cpk.xlsx'

report\_sheet = book.add\_worksheet('report')#'report'

ssh\_sheet = book.add\_worksheet('ssh')#'hash number '

if event == 'excel-report' and fail\_plot\_to\_excel == 'yes':

plot\_sheet = book.add\_worksheet('fail plot')#'plot'

else:

plot\_sheet = None

report\_sheet.set\_column("B:B", 80) # 设定A列列宽为40

report\_sheet.set\_column("M:M", 10) #

report\_sheet.set\_column("AB:AB",40) #

report\_sheet.set\_row(0, 30) #设置行高度

cpk\_result ='CPK\_Result(cpk\_lsl:'+str(cpk\_lsl)+';cpk\_usl:'+str(cpk\_usl)+')'

if p\_val\_checked == '1':

report\_sheet\_title = [u'No', u'Item\_name',u'BC',u'P\_Val',u'a\_Q',u'a\_irr',u'3CV',

u'ORIG LSL',u'Target', u'ORIG USL', u'Min', u'Mean',u'Max',

u'Std',u'CPL', u'CPU', u'CPK',cpk\_result, u'New LSL',u'New Target',

u'New USL',u'New CPL',u'New CPU',u'New CPK','New '+cpk\_result,

u'Reviewer Name',u'Review Date',u'Comment']

else:

report\_sheet\_title = [u'No', u'Item\_name',u'BC',u'a\_Q',u'a\_irr',u'3CV',

u'ORIG LSL',u'Target', u'ORIG USL', u'Min', u'Mean',

u'Max', u'Std',u'CPL', u'CPU', u'CPK',cpk\_result,

u'New LSL',u'New Target',u'New USL',u'New CPL',

u'New CPU',u'New CPK','New '+cpk\_result,

u'Reviewer Name',u'Review Date',u'Comment']

ssh\_sheet.set\_column("B:B", 20)

ssh\_sheet.set\_column("C:C", 50)

ssh\_sheet.set\_row(0, 30) #设置行高度

ssh\_sheet\_title = [u'Index', u'Item',u'ssh code']

format\_normal = book.add\_format() # 定义format格式对象

format\_normal.set\_align('center') # 定义format\_titile对象单元格对齐方式

format\_normal.set\_valign('center') # 定义format\_titile对象单元格对齐方式

format\_normal.set\_border(1) # 定义format对象单元格边框加粗的格式

# format\_normal.set\_num\_format('0.00') # 格式化数据格式为小数点后两位

format\_normal.set\_text\_wrap() # 内容换行

new\_format\_pass = book.add\_format() # 定义format格式对象

new\_format\_pass.set\_align('center') # 定义format\_titile对象单元格对齐方式

new\_format\_pass.set\_valign('center') # 定义format\_titile对象单元格对齐方式

new\_format\_pass.set\_bg\_color('#00FF00') # 定义format\_titile对象单元格背景颜色

new\_format\_pass.set\_border(1) # 定义format对象单元格边框加粗的格式

# new\_format\_pass.set\_num\_format('0.00') # 格式化数据格式为小数点后两位

new\_format\_pass.set\_text\_wrap() # 内容换行

format\_highlight = book.add\_format() # 定义format\_title 格式对象

format\_highlight.set\_border(1) # 定义format\_titile 对象单元格边框加粗的格式

format\_highlight.set\_bg\_color('yellow') # 定义format\_titile对象单元格背景颜色

format\_highlight.set\_align('center') # 定义format\_titile对象单元格对齐方式

format\_highlight.set\_valign('center') # 定义format\_titile对象单元格对齐方式

# format\_highlight.set\_num\_format('0.00') # 格式化数据格式为小数点后两位

format\_highlight.set\_text\_wrap() # 内容换行

new\_format\_fail = book.add\_format() # 定义format\_title 格式对象

new\_format\_fail.set\_border(1) # 定义format\_titile 对象单元格边框加粗的格式

new\_format\_fail.set\_bg\_color('red') # 定义format\_titile对象单元格背景颜色

new\_format\_fail.set\_align('center') # 定义format\_titile对象单元格对齐方式

new\_format\_fail.set\_valign('center') # 定义format\_titile对象单元格对齐方式

# format\_fail.set\_num\_format('0.00') # 格式化数据格式为小数点后两位

new\_format\_fail.set\_text\_wrap() # 内容换行

format\_titile = book.add\_format() # 定义format\_title 格式对象

format\_titile.set\_border(1) # 定义format\_titile 对象单元格边框加粗的格式

format\_titile.set\_bg\_color('#cddccc') # 定义format\_titile对象单元格背景颜色

format\_titile.set\_align('center') # 定义format\_titile对象单元格对齐方式

format\_titile.set\_valign('center') # 定义format\_titile对象单元格对齐方式

format\_titile.set\_bold() # 定义format\_titile对象内容加粗

format\_titile.set\_text\_wrap() # 内容换行

format\_ave = book.add\_format() # 定义format\_ave格式对象

format\_ave.set\_border(1) # 边框加粗的格式

format\_ave.set\_num\_format('0.00') # 定义format\_ave对象单元格数字类别显示格式

report\_sheet.write\_row('A1', report\_sheet\_title, format\_titile)

ssh\_sheet.write\_row('A1', ssh\_sheet\_title, format\_titile)

print('Excel report head created finished!')

return book,report\_sheet,plot\_sheet,ssh\_sheet,format\_highlight,format\_normal,format\_titile,new\_format\_fail,new\_format\_pass

def valid\_column(test\_item\_name,column\_list):

# column\_list = df[test\_item\_name].tolist()

# print('valid\_column item\_name-->',test\_item\_name)

# print('valid\_column--->',len(column\_list),len(set(column\_list)),column\_list)

# print('column\_value\_list--->',column\_list)

if test\_item\_name.lower().find('fixture vendor\_id') == -1 and test\_item\_name.lower().find('unit number') == -1:

pass

else:

return 'not\_cpk state1'

if len(column\_list)< 3:

# column\_list[0] == '0' and column\_list[1] == '0' or column\_list[0] == '1' and column\_list[1] == '1' or column\_list[0] == column\_list[1] and column\_list[0] !='NA' or

# print('====>',column\_list[0],column\_list[1])

return "not\_cpk state2"

else:

# pattern = re.compile(r'^[+-]?[0-9]\*\.?[0-9]+$')

# print('valid\_column len----->',test\_item\_name,len(column\_list))

# print('v====>',column\_list[0],column\_list[1],column\_list,len(column\_list))

j=0

for i in range(2,len(column\_list),1):

# print(str(i),'valid\_column----->',test\_item\_name,pattern.match(column\_list[i]))

if is\_number(column\_list[i]):

j=j+1

# print(str(j)+',len(set(column\_list))-->',j,len(set(column\_list)))

if j>0 and len(set(column\_list))>=1:#

# print('need\_cpk')

return "need\_cpk"

return "not\_cpk state3"

def test\_value\_to\_numeric(test\_data\_list):

column\_list = []

i = 0

for x in test\_data\_list:

if (i ==0 and (x == 'NA' or x == '')) or (i ==1 and (x == 'NA' or x == '')):

column\_list.append(x)

else:

try:

x = float(x)

column\_list.append(x)

except Exception as e:

pass

# print('-------------------- it is not number --------------')

i = i + 1

# print('column\_list--->',column\_list)

return column\_list

def is\_empty\_list(l):

temp\_l = []

for value in l:

if value != '':

temp\_l.append(value)

n = len(temp\_l)

if n == 0:

# print('it is empty list!')

return True

else:

return False

def parse\_all\_csv\_local(header\_list,df,color\_by1,selected\_category1,event,color\_by2,selected\_category2,param\_item\_start\_index):

color\_l = ['#0000FF','#FF0000','#008000','#00FFFF','#000000',

'#8B008B','#B8860B','#FF6347','#A9A9A9','#FFFF00',

'#A52A2A','#7FFF00','#D2691E','#6495ED','#FF00FF']

table\_data = []#[[]]

table\_category\_data = []#[[[]]]

column\_list = []

n=0

no\_valid\_column\_name\_l = []

for item\_name in header\_list:

# if event == 'one\_item\_plot1':

# # print('one column--->',type(df.iloc[:, [n]].values),df.iloc[:, [n]].values.tolist())

# temp\_l=[]

# for i in df.iloc[:, [n]].values.tolist():

# temp\_l.append(i[0])

# column\_list = temp\_l # 读取指定键值列的所有行

# print('column\_list--->',column\_list)

# n=n+1

# else:

# print('item\_name:',item\_name)

# print('header\_list in parse\_all\_csv:',header\_list)

# print('df:',df)

try:

column\_list = df[item\_name].tolist()

except Exception as e:

if event == 'excel-report' and header\_list.index(item\_name) >= param\_item\_start\_index:

no\_valid\_column\_name\_l.append(item\_name)

print(item\_name,'is duplicate ? pls check!',e)

continue

# print('column\_list--->',type(df[item\_name]),column\_list)

need\_cpk = valid\_column(item\_name,column\_list)

# print('need\_cpk:---->',need\_cpk)

column\_num\_list = []

if need\_cpk == 'need\_cpk':

column\_list = test\_value\_to\_numeric(column\_list)

column\_list.insert(0, item\_name) # item name

usl = column\_list[1]

lsl = column\_list[2]

# print('color\_by1:',color\_by1)

# 'Off'/'SerialNumber'/'Version'/'Station ID'/'Special Build Name'/'Product'/'StartTime'/'Special Build Description'

if color\_by1 == 'SerialNumber' or color\_by1 == 'Version' or color\_by1 == 'Station ID' or color\_by1 == 'Special Build Name' or color\_by1 == 'Product' or color\_by1 == 'StartTime' or color\_by1 =='Special Build Description' or color\_by1 =='Fixture Channel ID' or color\_by1 =='Diags\_Version':

column\_temp = []

first\_filter\_category\_data\_len=0

second\_filter\_category\_data\_len=0

i =0

for x in selected\_category1:

# print('x:',x,color\_by2)

if color\_by2 == 'SerialNumber' or color\_by2 == 'Version' or color\_by2 == 'Station ID' or color\_by2 == 'Special Build Name' or color\_by2 == 'Product' or color\_by2 == 'StartTime' or color\_by2 =='Special Build Description' or color\_by2 =='Fixture Channel ID' or color\_by2 =='Diags\_Version':

for xx in selected\_category2:

# print('xxxxx:', item\_name,color\_by1,x[0],x[1],color\_by2,xx[0],xx[1])

if color\_by2 == 'Fixture Channel ID':

# print('the same!',df.columns.values.tolist()[14])

index2 = df.columns.values.tolist()[14]

index2 = color\_by2

one\_category\_list = df.loc[(df[color\_by1] == x[0]) & (df[index2] == xx[0]), item\_name].tolist() #

if is\_empty\_list(one\_category\_list) != True:

# print('one category\_data with second filter-->', len(one\_category\_list),one\_category\_list)

second\_filter\_category\_data\_len = second\_filter\_category\_data\_len +len(one\_category\_list)

one\_category\_list = test\_value\_to\_numeric(one\_category\_list)

column\_temp = column\_temp +one\_category\_list

# print('usl--->', usl)

# print('lsl--->', lsl)

category\_value = x[0]+'&'+xx[0]

# print('category\_value--->',category\_value)

one\_category\_list.insert(0, category\_value) # insert category

# print('i==>',i)

if i >14:

i = 0

one\_category\_list.insert(1, color\_l[i]) # insert color

one\_category\_list.insert(2,item\_name) # item\_name

if lsl != 'NA':

lsl = float(lsl)

if usl != 'NA':

usl = float(usl)

one\_category\_list.insert(3, usl) # usl

one\_category\_list.insert(4, lsl) # lsl

# print('one\_category\_list-->', one\_category\_list)

column\_num\_list.append(one\_category\_list)

i=i+1

elif color\_by2 == 'Off':

# print('color\_by2= Off xxxxx:', x[0],x[1])

one\_category\_list = df.loc[df[color\_by1] == x[0], item\_name].tolist() #

if is\_empty\_list(one\_category\_list) != True:

# print('one category\_data with first filter-->', one\_category\_list)

first\_filter\_category\_data\_len = first\_filter\_category\_data\_len +len(one\_category\_list)

one\_category\_list = test\_value\_to\_numeric(one\_category\_list)

column\_temp = column\_temp +one\_category\_list

# usl = column\_list[1]

# lsl = column\_list[2]

# print('usl--->', usl)

# print('lsl--->', lsl)

one\_category\_list.insert(0, x[0]) # insert category

one\_category\_list.insert(1, x[1]) # insert color

one\_category\_list.insert(2,item\_name) # item\_name

if lsl != 'NA':

lsl = float(lsl)

if usl != 'NA':

usl = float(usl)

one\_category\_list.insert(3, usl) # usl

one\_category\_list.insert(4, lsl) # lsl

# print('one\_category\_list-->', one\_category\_list)

column\_num\_list.append(one\_category\_list)

# print('one category\_data total with first filter-->', first\_filter\_category\_data\_len)

# print('one category\_data total with second filter-->', second\_filter\_category\_data\_len)

column\_temp.insert(0,item\_name) # item\_name

if lsl != 'NA':

lsl = float(lsl)

if usl != 'NA':

usl = float(usl)

column\_temp.insert(1, usl) # usl

column\_temp.insert(2, lsl) # lsl

# print('column\_temp-->', column\_temp)

table\_data.append(column\_temp)# one column's all category data []

# print('column\_num\_list-->', column\_num\_list)

table\_category\_data.append(column\_num\_list)# category [[],[],...]]

elif color\_by1 == 'Off':

if event == 'excel-report':

if header\_list.index(item\_name) >= param\_item\_start\_index:

table\_data.append(column\_list) # item\_name,usl,lsl,data

else:

table\_data.append(column\_list) # item\_name,usl,lsl,data

else:

# print('no cpk item:',event,header\_list.index(item\_name),param\_item\_start\_index)

if event == 'excel-report' and header\_list.index(item\_name) >= param\_item\_start\_index:

no\_valid\_column\_name\_l.append(item\_name)

# print('table\_data-->',len(table\_data[0]),len(table\_data),len(table\_category\_data),table\_data,table\_category\_data)

if event == 'excel-report':

pass

else:

no\_valid\_column\_name\_l = []

# print('no\_valid\_column\_name\_l-->',no\_valid\_column\_name\_l)

return table\_data,table\_category\_data,no\_valid\_column\_name\_l #[[[ ]]]

def parse\_all\_csv(header\_list,df,color\_by1,selected\_category1,event,color\_by2,selected\_category2,param\_item\_start\_index):

color\_l = ['#0000FF','#FF0000','#008000','#00FFFF','#000000',

'#8B008B','#B8860B','#FF6347','#A9A9A9','#FFFF00',

'#A52A2A','#7FFF00','#D2691E','#6495ED','#FF00FF']

table\_data = []#[[]]

table\_category\_data = []#[[[]]]

column\_list = []

n=0

no\_valid\_column\_name\_l = []

for item\_name in header\_list:

# if event == 'one\_item\_plot1':

# # print('one column--->',type(df.iloc[:, [n]].values),df.iloc[:, [n]].values.tolist())

# temp\_l=[]

# for i in df.iloc[:, [n]].values.tolist():

# temp\_l.append(i[0])

# column\_list = temp\_l # 读取指定键值列的所有行

# print('column\_list--->',column\_list)

# n=n+1

# else:

# print('item\_name:',item\_name)

# print('header\_list in parse\_all\_csv:',header\_list)

# print('df:',df)

try:

column\_list = df[item\_name].tolist()

except Exception as e:

if event == 'excel-report' and header\_list.index(item\_name) >= param\_item\_start\_index:

no\_valid\_column\_name\_l.append(item\_name)

print(item\_name,'is duplicate ? pls check!',e)

continue

# print('column\_list--->',type(df[item\_name]),column\_list)

need\_cpk = valid\_column(item\_name,column\_list)

# print('need\_cpk:---->',need\_cpk)

column\_num\_list = []

if need\_cpk == 'need\_cpk':

column\_list = test\_value\_to\_numeric(column\_list)

column\_list.insert(0, item\_name) # item name

usl = column\_list[1]

lsl = column\_list[2]

# print('color\_by1:',color\_by1)

# 'Off'/'SerialNumber'/'Version'/'Station ID'/'Special Build Name'/'Product'/'StartTime'/'Special Build Description'

if color\_by1 == 'SerialNumber' or color\_by1 == 'Version' or color\_by1 == 'Station ID' or color\_by1 == 'Special Build Name' or color\_by1 == 'Product' or color\_by1 == 'StartTime' or color\_by1 =='Special Build Description' or color\_by1 =='Fixture Channel ID' or color\_by1 =='Diags\_Version':

column\_temp = []

first\_filter\_category\_data\_len=0

second\_filter\_category\_data\_len=0

i =0

for x in selected\_category1:

# print('x:',x,color\_by2)

if color\_by2 == 'SerialNumber' or color\_by2 == 'Version' or color\_by2 == 'Station ID' or color\_by2 == 'Special Build Name' or color\_by2 == 'Product' or color\_by2 == 'StartTime' or color\_by2 =='Special Build Description' or color\_by2 =='Fixture Channel ID' or color\_by2 =='Diags\_Version':

for xx in selected\_category2:

# print('xxxxx:', item\_name,color\_by1,x[0],x[1],color\_by2,xx[0],xx[1])

if color\_by2 == 'Fixture Channel ID':

# print('the same!',df.columns.values.tolist()[14])

index2 = df.columns.values.tolist()[14]

index2 = color\_by2

one\_category\_list = df.loc[(df[color\_by1] == x[0]) & (df[index2] == xx[0]), item\_name].tolist() #

if is\_empty\_list(one\_category\_list) != True:

# print('one category\_data with second filter-->', len(one\_category\_list),one\_category\_list)

second\_filter\_category\_data\_len = second\_filter\_category\_data\_len +len(one\_category\_list)

one\_category\_list = test\_value\_to\_numeric(one\_category\_list)

column\_temp = column\_temp +one\_category\_list

# print('usl--->', usl)

# print('lsl--->', lsl)

category\_value = x[0]+'&'+xx[0]

# print('category\_value--->',category\_value)

one\_category\_list.insert(0, category\_value) # insert category

# print('i==>',i)

if i >14:

i = 0

one\_category\_list.insert(1, color\_l[i]) # insert color

one\_category\_list.insert(2,item\_name) # item\_name

if lsl != 'NA':

lsl = float(lsl)

if usl != 'NA':

usl = float(usl)

one\_category\_list.insert(3, usl) # usl

one\_category\_list.insert(4, lsl) # lsl

# print('one\_category\_list-->', one\_category\_list)

column\_num\_list.append(one\_category\_list)

i=i+1

elif color\_by2 == 'Off':

# print('color\_by2= Off xxxxx:', x[0],x[1])

one\_category\_list = df.loc[df[color\_by1] == x[0], item\_name].tolist() #

if is\_empty\_list(one\_category\_list) != True:

# print('one category\_data with first filter-->', one\_category\_list)

first\_filter\_category\_data\_len = first\_filter\_category\_data\_len +len(one\_category\_list)

one\_category\_list = test\_value\_to\_numeric(one\_category\_list)

column\_temp = column\_temp +one\_category\_list

# usl = column\_list[1]

# lsl = column\_list[2]

# print('usl--->', usl)

# print('lsl--->', lsl)

one\_category\_list.insert(0, x[0]) # insert category

one\_category\_list.insert(1, x[1]) # insert color

one\_category\_list.insert(2,item\_name) # item\_name

if lsl != 'NA':

lsl = float(lsl)

if usl != 'NA':

usl = float(usl)

one\_category\_list.insert(3, usl) # usl

one\_category\_list.insert(4, lsl) # lsl

# print('one\_category\_list-->', one\_category\_list)

column\_num\_list.append(one\_category\_list)

# print('one category\_data total with first filter-->', first\_filter\_category\_data\_len)

# print('one category\_data total with second filter-->', second\_filter\_category\_data\_len)

column\_temp.insert(0,item\_name) # item\_name

if lsl != 'NA':

lsl = float(lsl)

if usl != 'NA':

usl = float(usl)

column\_temp.insert(1, usl) # usl

column\_temp.insert(2, lsl) # lsl

# print('column\_temp-->', column\_temp)

table\_data.append(column\_temp)# one column's all category data []

# print('column\_num\_list-->', column\_num\_list)

table\_category\_data.append(column\_num\_list)# category [[],[],...]]

elif color\_by1 == 'Off':

if event == 'excel-report':

if header\_list.index(item\_name) >= param\_item\_start\_index:

table\_data.append(column\_list) # item\_name,usl,lsl,data

else:

table\_data.append(column\_list) # item\_name,usl,lsl,data

else:

# print('no cpk item:',event,header\_list.index(item\_name),param\_item\_start\_index)

if event == 'excel-report' and header\_list.index(item\_name) >= param\_item\_start\_index:

no\_valid\_column\_name\_l.append(item\_name)

# print('table\_data-->',len(table\_data[0]),len(table\_data),len(table\_category\_data),table\_data,table\_category\_data)

if event == 'excel-report':

pass

else:

no\_valid\_column\_name\_l = []

# print('no\_valid\_column\_name\_l-->',no\_valid\_column\_name\_l)

return table\_data,table\_category\_data,no\_valid\_column\_name\_l #[[[ ]]]

def get\_coefficients(value\_l):

'''

param value\_l: need float list

return: bc,p\_val,a\_Q,a\_irr,three\_σ\_x100\_divide\_mean

1σ＝690000／1000000 #fault rate

2σ＝308000／1000000

3σ＝66800／1000000

4σ＝6210／1000000

5σ＝230／1000000

6σ＝3.4／1000000

7σ＝0／1000000

'''

if len(value\_l) <= 3:

return '','','','',''

column\_stdev = np.std(value\_l,ddof=1,axis=0)

three\_sigma= 3\*column\_stdev

# print('three\_sigma:',column\_stdev,three\_sigma)

temp\_l= value\_l

if len(value\_l) < 10:

temp\_l = value\_l + value\_l

if len(temp\_l)<10:

temp\_l = value\_l + value\_l + value\_l + value\_l+value\_l + value\_l + value\_l + value\_l+value\_l + value\_l

# print('------>>>temp\_l:',temp\_l)

data0 = np.array(temp\_l)

# print('------>>>data0:',data0)

try:

dip, p\_val = diptest.diptest(data0)

p\_val = '%f'%p\_val

# print('------>>>p\_val :',p\_val)

except RuntimeWarning as w:

print('calculate dip,p\_val RuntimeWarning:',w)

return '','','','',''

except Exception as e:

print('calculate dip,p\_val error:',e)

return '','','','',''

# print('dip,p\_val:',dip,p\_val)

item\_name ='value1'

data = pd.DataFrame({item\_name:value\_l})

# print('data--->',type(data),data)

u1 = data[item\_name].mean() # 计算均值

# std1 = data[item\_name].std() # 计算标准差

# t,pval=stats.kstest(data[item\_name], 'norm', (u1, std1))

# print('normality test t,pval:',str(t),str(pval))

# print('------')

# 正态性检验 → pvalue >0.05

n= float(len(value\_l))

#Item (xi-ẍ)^2

item\_l\_1 =[]

item\_l\_2 = []

item\_l\_3 = []

for i in value\_l:

temp1 = (i-u1)\*\*2

temp2 = (i-u1)\*\*3

temp3 = (i-u1)\*\*4

item\_l\_1.append(temp1)

item\_l\_2.append(temp2)

item\_l\_3.append(temp3)

# print('item\_l\_1--->',item\_l\_1)

# print('item\_l\_2--->',item\_l\_2)

# print('item\_l\_3--->',item\_l\_3)

sum\_item\_l\_1 = sum(item\_l\_1)

sum\_item\_l\_2 = sum(item\_l\_2)

sum\_item\_l\_3 = sum(item\_l\_3)

# print('sum\_item\_l\_1',sum\_item\_l\_1)

# print('sum\_item\_l\_2',sum\_item\_l\_2)

# print('sum\_item\_l\_3',sum\_item\_l\_3)

if n<=3 or sum\_item\_l\_1==0 or sum\_item\_l\_2==0 or sum\_item\_l\_3==0:

# print('len < 3--->')

if abs(u1) == 0:

return 'Nan',str(p\_val),'Nan','Nan','Nan'

else:

three\_CV = three\_sigma\*100/abs(u1)

# print('three\_CV:',three\_CV)

return 'Nan',str(p\_val),'Nan','Nan',str(round(three\_CV,6))

else:

try:

m3 = np.sqrt(n\*(n-1))/(n-2)\*((1/n\*sum\_item\_l\_2)/np.sqrt(1/n\*sum\_item\_l\_1)\*\*3)

# print('m3=',m3)

# print('d8:',n+1)

# print('d15:',1/n\*sum\_item\_l\_3/(1/n\*sum\_item\_l\_1)\*\*2)

# print('d16:',(n+1)\*1/n\*sum\_item\_l\_3/(1/n\*sum\_item\_l\_1)\*\*2-3\*(n-1))

# print('d6/d7',(n-1)/((n-2)\*(n-3)))

m4 = ((n-1)/((n-2)\*(n-3)))\*((n+1)\*1/n\*sum\_item\_l\_3/(1/n\*sum\_item\_l\_1)\*\*2-3\*(n-1))#(d6/d7)\*d16

# print('m4=',m4)

#(d14\*\*2+1)/(d17+3\*(d10/d7))

bc =(m3\*\*2+1)/(m4+3\*((n-1)\*\*2/((n-2)\*(n-3))))

# print('bc:',bc)

a\_L=0.05

a\_M=0.1

a\_U=0.32

a\_Q = (a\_U-a\_L)\*bc\*\*2+a\_L

# print ('a\_Q:',a\_Q)

a\_irr = np.sqrt((a\_U-a\_L)\*\*2\*bc)+a\_L

# print('a\_irr:',a\_irr)

except Exception as e:

# print('calculate error',e)

if abs(u1) == 0:

return 'Nan',str(p\_val),'Nan','Nan','Nan'

else:

three\_CV = three\_sigma\*100/abs(u1)

# print('three\_CV:',three\_CV)

return 'Nan',str(p\_val),'Nan','Nan',str(round(three\_CV,6))

if abs(u1) == 0:

return str(round(bc,6)),str(p\_val),str(round(a\_Q,6)),str(round(a\_irr,6)),'Nan'

else:

three\_CV = three\_sigma\*100/abs(u1)

# print('three\_CV:',three\_CV)

return str(round(bc,6)),str(p\_val),str(round(a\_Q,6)),str(round(a\_irr,6)),str(round(three\_CV,6))

def cpk\_calc(df\_data,lsl,usl):

"""

:param df\_data: list

:param usl: 数据指标上限

:param lsl: 数据指标下限

:return:

"""

sigma = 3

# print('limit--->',lsl,usl)

# 数据平均值

# print('df\_data in cpk\_calc:',df\_data)

mean = np.mean(df\_data)#

# print('mean ---->',mean)

# 数据max值

max\_num = max(df\_data)

# print('max\_num ---->',max\_num)

# 数据min值

min\_num = min(df\_data)

# print('min\_num ---->',min\_num)

# a = np.array([[1, 2], [3, 4]])

# print('a--->', type(a))

# print('gobal std:',np.std(a))#全局标准差

# print('each line std:',np.std(a, axis=0,ddof=1))

# print("each row std:",np.std(a, axis=1,ddof=1))

# 数据标准差

if len(df\_data)==1:

stdev =0.00

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

else:

try:

stdev = np.std(df\_data,ddof=1,axis=0)

except Exception as e:

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

# print('stdev ---->',stdev)

if stdev == 0:#stop count cpk

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

# 生成横轴数据平均分布

# x1 = np.linspace(mean - sigma \* stdev, mean + sigma \* stdev, 1000)

# print('x1 ---->',x1)

# 计算正态分布曲线

# y1 = np.exp(-(x1 - mean) \*\* 2 / (2 \* stdev \*\* 2)) / (math.sqrt(2 \* math.pi) \* stdev)

# print('y1 ---->',y1)

x1,y1 = None,None

if (lsl != 'NA' and lsl != '') and (usl == 'NA' or usl == ''):

cpl = (mean - lsl) / (sigma \* stdev)

# print('====>>>>>>cpl',cpl)

return (mean,max\_num,min\_num,stdev,None,None,None,cpl,None)

if (usl != 'NA' and usl != '') and (lsl == 'NA' or lsl == ''):

# print('====>>>>>>=====cpu')

cpu = (usl - mean) / (sigma \* stdev)

# print('====>>>>>>cpu',cpu)

return (mean,max\_num,min\_num,stdev,None,None,cpu,None,None)

if lsl == 'NA' or usl == 'NA' or lsl == '' or usl == '':

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

cpu = (usl - mean) / (sigma \* stdev)

cpl = (mean - lsl) / (sigma \* stdev)

# print('cpu ---->',cpu)

# print('cpl ---->',cpl)

# 得出cpk

cpk = min(cpu, cpl)

return (mean,max\_num,min\_num,stdev,x1,y1,cpu,cpl,cpk)

def get\_target\_value(lsl,usl):

if lsl != 'NA' and usl != 'NA' and lsl != '' and usl != '':

target\_value = round(((lsl + usl) / 2.0), 5)

else:

target\_value = 'Nan'

return target\_value

def probability\_distribution\_extend(data,bins,margin,item\_name,lsl,usl,mean,

max\_num,min\_num,stdev,x1,y1,cpu,cpl,cpk,

pic\_path,bins\_l,bins\_h,start\_time\_first,

start\_time\_last,bmc,zoom\_type='limit'):

bins = sorted(bins)

length = len(bins)

intervals = np.zeros(length+1)

for value in data:

i = 0

while i < length and value >= bins[i]:

i += 1

intervals[i] += 1

intervals = intervals / float(len(data))

plt.ion() # 开启interactive mode

plt.figure(1) # 创建图表1

plt.xlim(min(bins) - margin, max(bins) + margin)

bins.insert(0, -999)

# plt.title("probability-distribution",size=8,verticalalignment='bottom')

plt.bar(bins, intervals,color=['r'], label='')#频率分布

x\_ticks,labels = plt.xticks()

#x\_ticks\_start=round(x\_ticks[0],2)

#x\_ticks\_end = round(x\_ticks[len(x\_ticks) - 1],2)

y\_ticks, labels = plt.yticks()

#print('=====y\_ticks, labels:',y\_ticks, labels)

# print('y\_ticks--->',y\_ticks,y\_ticks[len(y\_ticks) - 1],y\_ticks[0])

y\_ticks=round(y\_ticks[len(y\_ticks) - 1],5)

# print("x\_ticks\_start,x\_ticks\_end--->",x\_ticks\_start,x\_ticks\_end)

# print('y\_ticks\_end--->',y\_ticks)

# plt.show()

plt.close(1)

plt.figure(2,dpi=150) # 创建图表2

fig, axes = plt.subplots(1, 0, figsize=(6, 5), facecolor='#ccddef')

plt.axes([0.1, 0.17, 0.85, 0.7]) # [左, 下, 宽, 高] 规定的矩形区域 （全部是0~1之间的数，表示比例）

if stdev =='nan':

pass

else:

if stdev > 999999:

stdev = str("%.3e" % stdev)

else:

stdev = str("%.3f" % stdev)

if cpl ==None:

cpl\_value =''

else:

if cpl > 999999:

cpl\_value = str("%.3e" % cpl)

else:

cpl\_value = str("%.3f" % cpl)

if cpu ==None:

cpu\_value =''

else:

if cpu > 999999:

cpu\_value = str("%.3e" % cpu)

else:

cpu\_value = str("%.3f" % cpu)

if cpk ==None:

cpk\_value =''

else:

if cpk > 999999:

cpk\_value = str("%.3e" % cpk)

else:

cpk\_value = str("%.3f" % cpk)

if max\_num > 999999:

max\_num = str("%.3e" % max\_num)

else:

max\_num = str("%.3f" % max\_num)

if mean > 999999:

mean = str("%.3e" % mean)

else:

mean = str("%.3f" % mean)

if min\_num > 999999:

min\_num = str("%.3e" % min\_num)

else:

min\_num = str("%.3f" % min\_num)

if len(data) > 1000000000000:

sample\_n = str("%.e" % len(data))

else:

sample\_n = str("%.f" % len(data))

info = "Samples:" + sample\_n + ' ' +"Max:" + max\_num + ' ' + "Mean:" + mean + ' ' + "Min:" + min\_num + '\n' + "Std:" + stdev + ' ' + "Cpl:" + cpl\_value + ' ' + "Cpu:" + cpu\_value + ' ' + "Cpk:" + cpk\_value + '\n' + "Bimodal:" + bmc

if len(item\_name) > 60:

item\_name = item\_name[0:60] + '\n' + item\_name[60:]

# font = FontProperties(fname=r"/Library/Fonts/Songti.ttc", size=12)

# plt.title(item\_name,FontProperties=font)

plt.title(item\_name,size=11,verticalalignment='bottom')

# plt.xlabel(str(start\_time\_first)+' -- '+str(start\_time\_last))

plt.ylabel('Count')

bins = [round(x,5) for x in bins]

bins=sorted(bins)

plt.hist(data, bins=bins, label=info, histtype='stepfilled',color = 'blue', edgecolor='blue', linewidth=1.0,align='mid',density=False) #time分布

range\_value = get\_limit\_range(bins\_l, bins\_h)

range\_value =round(range\_value/5,5)

# print('plot bar--->',bins\_l,range)

if zoom\_type =='data':

plt.xlim(float(min\_num)\*0.999, float(max\_num)\*1.001)

else:

if (lsl =='' or lsl =='NA') and (usl!='NA'and usl!=''):

plt.xlim(float(min\_num)\*0.999, usl+range\_value)

elif (usl =='' or usl =='NA') and (lsl!='NA'and lsl!=''):

plt.xlim(lsl-range\_value, float(max\_num)\*1.001)

elif (usl =='' or usl =='NA') and (lsl=='NA'or lsl==''):

plt.xlim(float(min\_num)\*0.999, float(max\_num)\*1.001)

else:

plt.xlim(bins\_l-range\_value, bins\_h+range\_value)

y\_ticks = len(data) \* y\_ticks

plt.ylim((0, y\_ticks)) # 设置y轴scopex

ax=plt.gca()

ax.spines['bottom'].set\_linewidth(1)

ax.spines['left'].set\_linewidth(1)

ax.spines['right'].set\_linewidth(1)

ax.spines['top'].set\_linewidth(1)

# if lsl !='NA' and usl != 'NA' and zoom\_type =='limit':

# plt.plot([lsl, lsl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画lower limit线，

# plt.plot([usl, usl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画upper limit线，

# plt.text(lsl, y\_ticks / 3, ' LSL\n' + ' ' + str(lsl), fontdict={'size': 8, 'color': 'r'})

# plt.text(usl, y\_ticks / 2, ' USL\n' + ' ' + str(usl), fontdict={'size': 8, 'color': 'r'})

if lsl !='' and lsl !='NA' and zoom\_type =='limit':

plt.plot([lsl, lsl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画lower limit线，

# plt.plot([usl, usl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画upper limit线，

plt.text(lsl, y\_ticks / 3, ' LSL\n' + ' ' + str(lsl), fontdict={'size': 8, 'color': 'r'})

# plt.text(usl, y\_ticks / 2, ' USL\n' + ' ' + str(usl), fontdict={'size': 8, 'color': 'r'})

if usl != '' and usl != 'NA' and zoom\_type =='limit':

# plt.plot([lsl, lsl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画lower limit线，

plt.plot([usl, usl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画upper limit线，

# plt.text(lsl, y\_ticks / 3, ' LSL\n' + ' ' + str(lsl), fontdict={'size': 8, 'color': 'r'})

plt.text(usl, y\_ticks / 2, ' USL\n' + ' ' + str(usl), fontdict={'size': 8, 'color': 'r'})

plt.legend(bbox\_to\_anchor=(0.9,-0.09),loc="best",fontsize=10,framealpha=0,edgecolor='royalblue',borderaxespad=0.1)

plt.grid(linestyle=':',c='gray') # 生成网格

plt.savefig(pic\_path,dpi=200)

plt.draw()

plt.close('all')

plt.ioff()

def get\_bins(min\_num,max\_num,lsl,usl,set\_bins,zoom\_type='limit'):

# print('min\_num,max\_num,lsl,usl,set\_bins=====>',min\_num,max\_num,lsl,usl,set\_bins)

bins\_l = 0

bins\_h = 0

if lsl == 'NA' or usl == 'NA' or lsl == '' or usl == '' or zoom\_type == 'data':

bins\_l,bins\_h = min\_num,max\_num

else:

if min\_num < lsl and max\_num < lsl:

bins\_l = min\_num

bins\_h = usl

elif min\_num < lsl and max\_num > lsl and max\_num <= usl:

bins\_l = min\_num

bins\_h = usl

elif min\_num < lsl and max\_num > usl:

bins\_l = min\_num

bins\_h = max\_num

elif lsl <= min\_num and min\_num <= usl and max\_num <= usl:

bins\_l = lsl

bins\_h = usl

elif lsl <= min\_num and min\_num <= usl and max\_num > usl:

bins\_l = lsl

bins\_h = max\_num

elif min\_num > usl:

bins\_l = lsl

bins\_h = max\_num

range\_value = get\_limit\_range(bins\_l,bins\_h)

if lsl == 'NA' or usl == 'NA' or lsl == '' or usl == '' or zoom\_type == 'data':

if range\_value ==0 and min\_num > 0:

range\_value = min\_num\*0.2

bins\_l = (bins\_l - min\_num\*0.1)

bins\_h = (bins\_h + min\_num\*0.1)

elif range\_value ==0 and min\_num == 0:

range\_value = 6

bins\_l = - 3

bins\_h = 3

elif range\_value ==0 and min\_num <0:

range\_value = 6

bins\_l = min\_num - 3

bins\_h = min\_num + 3

# print('range\_value0-->',range\_value,min\_num,bins\_l,bins\_h)

else:

if min\_num > 1 and range\_value < 1 and range\_value !=0:

range\_value = min\_num\*0.05

bins\_l = (bins\_l - min\_num\*0.025)

bins\_h = (bins\_h + min\_num\*0.025)

elif min\_num > 0.001 and min\_num < 1 and range\_value < 1 and range\_value !=0:

range\_value = min\_num\*0.2

bins\_l = (bins\_l - min\_num\*0.1)

bins\_h = (bins\_h + min\_num\*0.1)

else:

range\_value = range\_value\*0.4

bins\_l = (bins\_l - range\_value\*0.2)

bins\_h = (bins\_h + range\_value\*0.2)

else:

if range\_value == 0 and lsl !=0:

range\_value = lsl\*0.2

bins\_l = bins\_l - lsl\*0.1

bins\_h = bins\_h + usl\*0.1

elif range\_value == 0 and lsl ==0:

range\_value = 6

bins\_l = - 3

bins\_h = 3

# print('range\_value1-->',range\_value)

range\_value = round((range\_value/set\_bins),12)

# print('range\_value2-->',range\_value)

# print('=====>',bins\_l,bins\_h,range\_value)

# bins = np.arange(bins\_l, bins\_h, range\_value)#必须是单调递增的

bins = np.arange(bins\_l, bins\_h, range\_value)#必须是单调递增的

if bins\_l<0:

bins = np.arange(bins\_h, bins\_l, -range\_value)

# print('lsl,usl,min\_num,max\_num,bins\_l,bins\_h in get\_bins=====>',lsl,usl,min\_num,max\_num,bins\_l,bins\_h)

return bins,bins\_l,bins\_h

def get\_limit\_range(lsl,usl):

# print('lsl,usl----->', lsl, usl)

range\_value = 0

if lsl < 0 and usl <= 0:

range\_value = abs(lsl) - abs(usl)

elif lsl < 0 and usl >= 0:

range\_value = abs(lsl) + usl

elif lsl >= 0 and (usl > 0):

range\_value = usl - lsl

else:

print('get\_limit\_range 00000')

range\_value = round(range\_value, 5)

# print('range in get\_limit\_range----->', range\_value)

return range\_value

def probability\_distribution\_extend\_by\_color(column\_category\_data\_list,data,bins,margin,item\_name,

lsl,usl,mean,max\_num,min\_num,stdev,x1,y1,cpu,cpl,cpk,

pic\_path,bins\_l,bins\_h,start\_time\_first,start\_time\_last,zoom\_type):

# print('one column data len:',len(data),item\_name,data,column\_category\_data\_list)

bins = sorted(bins)

length = len(bins)

intervals = np.zeros(length+1)

for value in data:

i = 0

while i < length and value >= bins[i]:

i += 1

intervals[i] += 1

intervals = intervals / float(len(data))

plt.ion() # 开启interactive mode

plt.figure(1) # 创建图表1

plt.xlim(min(bins) - margin, max(bins) + margin)

bins.insert(0, -999)

plt.title("probability-distribution")

plt.bar(bins, intervals,color=['r'], label='')#频率分布

x\_ticks,labels = plt.xticks()

x\_ticks\_start=round(x\_ticks[0],2)

x\_ticks\_end = round(x\_ticks[len(x\_ticks) - 1],2)

y\_ticks, labels = plt.yticks()

# print('y\_ticks--->',y\_ticks,y\_ticks[len(y\_ticks) - 1],y\_ticks[0])

y\_ticks=round(y\_ticks[len(y\_ticks) - 1],5)

# print("x\_ticks\_start,x\_ticks\_end--->",x\_ticks\_start,x\_ticks\_end)

# print('y\_ticks\_end--->',y\_ticks)

# plt.show()

plt.close(1)

plt.figure(2,dpi=150) # 创建图表2,facecolor='blue',edgecolor='black'

if stdev =='nan':

pass

else:

if stdev > 999999:

stdev = str("%.3e" % stdev)

else:

stdev = str("%.3f" % stdev)

if cpl ==None:

cpl\_value =''

else:

if cpl > 999999:

cpl\_value = str("%.3e" % cpl)

else:

cpl\_value = str("%.3f" % cpl)

if cpu ==None:

cpu\_value =''

else:

if cpu > 999999:

cpu\_value = str("%.3e" % cpu)

else:

cpu\_value = str("%.3f" % cpu)

if cpk ==None:

cpk\_value =''

else:

if cpk > 999999:

cpk\_value = str("%.3e" % cpk)

else:

cpk\_value = str("%.3f" % cpk)

if max\_num > 999999:

max\_num = str("%.3e" % max\_num)

else:

max\_num = str("%.3f" % max\_num)

if mean > 999999:

mean = str("%.3e" % mean)

else:

mean = str("%.3f" % mean)

if min\_num > 999999:

min\_num = str("%.3e" % min\_num)

else:

min\_num = str("%.3f" % min\_num)

if len(data) > 1000000000000:

sample\_n = str("%.e" % len(data))

else:

sample\_n = str("%.f" % len(data))

info = "Sample: " + sample\_n + '\n' +"Max: " + max\_num + '\n' + "Mean: " + mean + '\n' + "Min: " + min\_num + '\n' + "Std: " + stdev + '\n' + "Cpl: " + cpl\_value + '\n' + "Cpu: " + cpu\_value + '\n' + "Cpk: " + cpk\_value

# info = "Sample: " + str("%.f" % len(data)) + '\n' +"Max: " + str("%.3f" % max\_num) + '\n' + "Mean: " + str("%.3f" % mean) + '\n' + "Min: " + str(

# "%.3f" % min\_num) + '\n' + "Std: " + str("%.3f" % stdev) + '\n' + "Cpl: " + str(

# "%.3f" % cpl) + '\n' + "Cpu: " + str("%.3f" % cpu) + '\n' + "Cpk: " + str("%.3f" % cpk)

if len(item\_name) > 55:

item\_name = item\_name[0:55] + '\n' + item\_name[55:]

# font = FontProperties(fname=r"/Library/Fonts/Songti.ttc", size=12)

plt.title(item\_name,size=10)

plt.xlabel(str(start\_time\_first)+' -- '+str(start\_time\_last))

plt.ylabel('Count')

# plt.title(item\_name+"\nCpk={0}".format(str("%.6f" % cpk)))

# plt.hist(x=data, bins=bins, density=False, histtype='bar', color=['r'])

bins = [round(x,5) for x in bins]

bins=sorted(bins)

# print('----->bins-->',bins)

# print(' more draw category data:',len(column\_category\_data\_list),column\_category\_data\_list)

l=0

l\_len=[]

for category\_data in column\_category\_data\_list:

# print('category\_test\_data--->', category\_data[5:])

# print('category\_name,category\_color--->', category\_data[0],str(category\_data[1]))

category\_name,category\_color = category\_data[0],str(category\_data[1])

n=len(category\_data[5:])

l=l+n

# print('category len:',n)

l\_len.append(n)

if len(column\_category\_data\_list) == 1:

plt.hist(category\_data[5:], bins=bins, label=category\_data[0], color=category\_color ,histtype='stepfilled',edgecolor=category\_color,linewidth=1.5,align='mid',density=False) #time分布

else:

plt.hist(category\_data[5:], bins=bins, label=category\_data[0], color='white' ,histtype='step',edgecolor=category\_color,linewidth=1.5,align='mid',density=False) #time分布

# print('----->a column len,max in one category:-->',l,max(l\_len))

y\_ticks = max(l\_len) \* (y\_ticks+0.04)

range\_value = get\_limit\_range(bins\_l, bins\_h)

range\_value =round(range\_value/5,5)

# print('plot bar--->',bins\_l,range)

plt.xlim(bins\_l-range\_value, bins\_h+range\_value)

plt.ylim((0, y\_ticks)) # 设置y轴scopex

ax=plt.gca()

ax.spines['bottom'].set\_linewidth(1.5)

ax.spines['left'].set\_linewidth(1.5)

ax.spines['right'].set\_linewidth(1.5)

ax.spines['top'].set\_linewidth(1.5)

# plt.plot(x1, y1, 'k--', label="", linewidth=1.0, color='lime') # 画正态分布曲线

if lsl !='NA' and usl != 'NA' and zoom\_type=='limit':

plt.plot([lsl, lsl, ], [0, y\_ticks, ], 'k--', linewidth=3.0, color='red') # 画lower limit线，

plt.plot([usl, usl, ], [0, y\_ticks, ], 'k--', linewidth=3.0, color='red') # 画upper limit线，

# 添加文字

# plt.text(0.1,20, r'$\mu=100,\ \sigma=15$')

plt.text(lsl, y\_ticks / 3, ' LSL\n' + ' ' + str(lsl), fontdict={'size': 10, 'color': 'r'})

plt.text(usl, y\_ticks / 2, ' USL\n' + ' ' + str(usl), fontdict={'size': 10, 'color': 'r'})

# plt.axis([-0.2, 0.2,0, 100])#x 轴，y 轴

plt.text(bins\_l+range\_value/3, y\_ticks\*0.78, info, size=10, rotation=0.0, alpha=0.85,fontsize=8,ha="left",

va="center",bbox=dict(boxstyle="round", ec=('royalblue'),linestyle='-.',lw=1, fc=('white'), ))

#os.system('mkdir fail')

if len(column\_category\_data\_list) < 30:

plt.legend(loc="upper right",framealpha=1,edgecolor='royalblue',borderaxespad=0.3,fontsize=6)#facecolor ='None',

# plt.legend(bbox\_to\_anchor=(1.01, 1), loc=2, borderaxespad=0)

plt.grid(linestyle=':',c='gray') # 生成网格

# path="/Users/rex/PycharmProjects/my/fail/"

plt.savefig(pic\_path,dpi=200)

plt.draw()

# plt.show()

plt.close('all')

plt.ioff()

def save\_image\_to\_excel(location,pic\_path,plot\_sheet):

#save picture to excel

plot\_sheet.insert\_image(location,pic\_path,{'x\_scale': 1.2, 'y\_scale': 1.2})

return True

def get\_one\_item\_new\_limit\_from\_csv(new\_limit\_path,item\_name):

new\_limit\_file = '/tmp/CPK\_Log/temp/item\_limit.csv'

if not os.path.exists(new\_limit\_file):

return None,None

tmp\_lst = []

# try:

with open(new\_limit\_file, 'r') as f:

reader = csv.reader(f)

i = 1

for row in reader:

# print(row[0].lower())

if row[0].lower().find('item') != -1:

tmp\_lst.append(row)

else:

tmp\_lst.append(row)

i = i + 1

# except IOError as exc:

# print('ERROR:',exc)

header\_list = tmp\_lst[0]

# print('header\_list--->',header\_list)

df = pd.DataFrame(tmp\_lst[1:], columns=tmp\_lst[0])

# print(df)

# print(item\_name)

try:

new\_lsl=df.loc[df['item'] == item\_name, 'new\_lsl'].tolist()[0]

except IndexError as e:

# print('lsl-->',e)

new\_lsl = None

try:

new\_usl=df.loc[df['item'] == item\_name, 'new\_usl'].tolist()[0]

except IndexError as e:

# print('usl-->',e)

new\_usl = None

try:

reviewer = df.loc[df['item'] == item\_name, 'reviewer'].tolist()[0]

except IndexError as e:

# print('lsl-->',e)

reviewer = ''

try:

review\_date = df.loc[df['item'] == item\_name, 'date'].tolist()[0]

except IndexError as e:

# print('lsl-->',e)

review\_date = ''

try:

comment = df.loc[df['item'] == item\_name, 'comment'].tolist()[0]

except IndexError as e:

# print('lsl-->',e)

comment = ''

return new\_lsl,new\_usl,reviewer,review\_date,comment

def write\_hash\_to\_excel(file\_path,sheet\_name,cell\_format):

hash\_csv = '/tmp/CPK\_Log/temp/data\_hash.csv'

if not os.path.exists(hash\_csv):

return None,None

tmp\_lst = []

with open(hash\_csv, 'r') as f:

reader = csv.reader(f)

i = 1

for row in reader:

# print(row[0].lower())

if i>1:

location ='A'+str(i)

sheet\_name.write\_row(location,row,cell\_format)

i = i + 1

return 'hash save finish! ok'

def get\_file\_pic\_name(file\_dir):

pic\_file\_l = []

for root,dirs,files in os.walk(file\_dir):

# print(root)

# print('fail picture path:',dirs)

# print(files)

for file in files:

if os.path.splitext(file)[1] == '.png':

# print('suffix name---->',os.path.splitext(file)[1])

pic\_file\_l.insert(0,os.path.splitext(file)[0])

return pic\_file\_l

def append\_hash\_to\_excel(table\_data,excel\_file\_path,hashfile\_path):

workbook = openpyxl.load\_workbook(excel\_file\_path)

wb\_sheet = workbook.sheetnames

ssh\_sheet = workbook['ssh']

hash\_csv = '/tmp/CPK\_Log/temp/data\_hash.csv'

if not os.path.exists(hash\_csv):

return None,None

tmp\_lst = []

print('hashfile\_path:',hash\_csv)

with open(hash\_csv, 'r') as f:

csv\_reader = csv.reader(f)

i = 1

for row in csv\_reader:

print('row:',row)

if i>1:

for col,v in enumerate(row):

ssh\_sheet.cell(i,col+1).value = v.encode('utf-8')

i = i + 1

workbook.save(excel\_file\_path)

if 'fail plot' in wb\_sheet:

workbook = openpyxl.load\_workbook(excel\_file\_path)

# fail\_plot\_sheet = workbook['fail plot']

# fail\_pic\_path = hashfile\_path+'fail\_plot'

# file\_l=[]

# file\_l = get\_file\_pic\_name(fail\_pic\_path)

# j=0

# for f\_name in file\_l:

# if j==0:

# location='A1'

# else:

# location = 'A'+str(30\*j)

# one\_fail\_pic\_path = fail\_pic\_path +'/'+f\_name+'.png'

# print('===one\_fail\_pic\_path :',one\_fail\_pic\_path)

# # fail\_plot\_sheet.insert\_image(location,one\_fail\_pic\_path,{'x\_scale': 1.2, 'y\_scale': 1.2})

# j=j+1

workbook.save(excel\_file\_path)

push2git\_checkBox = table\_data[2]

if push2git\_checkBox == 1:

update\_limit\_path = table\_data[1]

gitAddr = table\_data[3]

gitComment = table\_data[4]

print('cd ' + str(update\_limit\_path))

os.chdir(str(update\_limit\_path))

file\_name = os.path.split(excel\_file\_path)[1]

print('git add ' + str(file\_name))

os.system('git add ' + str(file\_name))

print('git commit -m "'+ str(gitComment) + '"')

os.system('git commit -m "'+ str(gitComment) + '"')

print('git remote add origin "'+ str(gitAddr) + '"')

os.system('git remote add origin "'+ str(gitAddr) + '"')

print('git push -u origin master')

os.system('git push -u origin master')

with open(filelognamehash, 'w') as file\_object:

file\_object.write("Finished,create excel report finish")

print('hash save finish! done')

def verify\_limit(lsl,usl):

if lsl != None:

lsl.replace(' ','')

if usl != None:

usl.replace(' ','')

if lsl =='NA' or lsl =='':

lsl = None

else:

try:

lsl = float(eval(lsl))

except:

lsl = None

if usl =='NA' or usl == '':

usl = None

else:

try:

usl = float(eval(usl))

except:

usl = None

# if type(lsl) == float and type(usl) == float:

# print('after verify lsl===>',lsl)

# print('after verify usl===>',usl)

return lsl,usl

def write\_invalid\_item\_to\_excel(invalid\_item\_name\_l,sheet\_name,row,cell\_format,limit\_csv\_path):

for i,v in enumerate(invalid\_item\_name\_l):

sheet\_name.write(row+1+i,0,row+1+i,cell\_format)

sheet\_name.write(row+1+i,1,v,cell\_format)

new\_lsl,new\_usl,reviewer,review\_date,comment = get\_one\_item\_new\_limit\_from\_csv(limit\_csv\_path,v)

if p\_val\_checked == '1':

for n in range(2,28,1):

if new\_lsl != None and n == 18:

sheet\_name.write(row+1+i,n,new\_lsl,cell\_format)

elif new\_usl !=None and n == 20:

sheet\_name.write(row+1+i,n,new\_usl,cell\_format)

elif reviewer != '' and n == 25:

sheet\_name.write(row+1+i,n,reviewer,cell\_format)

elif review\_date != '' and n == 26:

sheet\_name.write(row+1+i,n,review\_date,cell\_format)

elif comment != '' and n == 27:

sheet\_name.write(row+1+i,n,comment,cell\_format)

else:

sheet\_name.write(row+1+i,n,'',cell\_format)

else:

for n in range(2,27,1):

if new\_lsl != None and n == 17:

sheet\_name.write(row+1+i,n,new\_lsl,cell\_format)

elif new\_usl !=None and n == 19:

sheet\_name.write(row+1+i,n,new\_usl,cell\_format)

elif reviewer != '' and n == 24:

sheet\_name.write(row+1+i,n,reviewer,cell\_format)

elif review\_date != '' and n == 25:

sheet\_name.write(row+1+i,n,review\_date,cell\_format)

elif comment != '' and n == 26:

sheet\_name.write(row+1+i,n,comment,cell\_format)

else:

sheet\_name.write(row+1+i,n,'',cell\_format)

def draw\_histogram(column\_data,item\_name,lsl,usl,mean,max\_num,min\_num,stdev,

x1,y1,cpu,cpl,cpk,pic\_path,set\_bins,start\_time\_first,

start\_time\_last,bmc,zoom\_type):

bins,bins\_l,bins\_h = get\_bins(min\_num,max\_num,lsl,usl,set\_bins,zoom\_type)

# print(len(column\_data),'---->',min(column\_data),max(column\_data),bins)

probability\_distribution\_extend(column\_data,bins,0,item\_name,lsl,usl,

mean,max\_num,min\_num,stdev,x1,y1,cpu,cpl,

cpk,pic\_path,bins\_l,bins\_h,start\_time\_first,

start\_time\_last,bmc,zoom\_type)

return True

def draw\_more\_histogram(column\_category\_data\_list,column\_data, item\_name, lsl, usl, mean,

max\_num, min\_num, stdev, x1, y1,cpu, cpl, cpk, pic\_path,set\_bins,

start\_time\_first,start\_time\_last,zoom\_type):

"""

"""

# print('9999 column\_category\_data\_list--->',column\_category\_data\_list)

# print('9999 column\_data--->',len(column\_data),column\_data)

# range = get\_limit\_range(lsl, usl)

# range = round((range /set\_bins), 5)

# bins = np.arange(lsl, usl, range) # 必须是单调递增的

bins,bins\_l,bins\_h = get\_bins(min\_num,max\_num,lsl,usl,set\_bins,zoom\_type)

# print('9999 bins len--->',len(bins))

# print(len(column\_data),'---->',min(column\_data),max(column\_data),bins)

probability\_distribution\_extend\_by\_color(column\_category\_data\_list,column\_data,bins, 0, item\_name, lsl, usl,

mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,bins\_l,

bins\_h,start\_time\_first,start\_time\_last,zoom\_type)

return True

def get\_project\_info(data\_df):

# print('one row data:--->',data\_df[1:2].values[0].tolist(),len(data\_df[1:2].values[0].tolist()))

# project\_code = data\_df[1:2].values[0].tolist()[1] #Product first cell

# print("=======get\_project\_info======",data\_df)

project\_code = get\_project\_code(data\_df)

# print('--------------------project\_code:'+project\_code+'------------------------')

build\_stage = get\_build\_stage(data\_df)

# print('--------------------build\_stage:'+build\_stage+'-------------------------')

station\_name = get\_station\_name(data\_df)

# print('--------------------station\_name:'+station\_name+'------------------------')

return project\_code,build\_stage,station\_name

def get\_build\_stage(data\_df):

special\_build\_name = data\_df['Special Build Name'].values.tolist() #Special Build Name

# pattern =re.compile(r'.\*\-(.+)\-')

# result=pattern.match(build\_stage)

# build\_stage = result.group(1)

build\_stage =''

build\_stage\_l = list(set(special\_build\_name))

print('build\_stage\_l-->',build\_stage\_l)

temp\_l=[]

for i in build\_stage\_l:

pattern = re.compile(r'.\*\-(.+)')

result = pattern.match(i)

if result:

temp\_l.append(result.group(1))

n=1

for j in list(set(temp\_l)):

if n==1:

build\_stage = j

else:

build\_stage = build\_stage +'&'+j

n=n+1

print("======>>>>>>>build\_stage",build\_stage)

return build\_stage

def get\_station\_name(data\_df):

# station\_name = data\_df[1:2].values[0].tolist()[6] #Station ID first cell

station\_name\_l = data\_df['Station ID'].values.tolist() #Station ID first cell

station\_name\_l = list(set(station\_name\_l))

station\_name =''

if len(station\_name\_l) == 0:

print('data is empty!')

station\_name == 'xxx'

else:

for i in station\_name\_l:

if i != '':

pattern =re.compile(r'.\*\\_([0-9]+)\\_(\D+)')

result=pattern.match(i)

try:

station\_name = result.group(2)

return station\_name

except Exception as e:

print('match station\_name error')

return station\_name

def get\_project\_code(data\_df):

product\_l = list(set(data\_df['Product'].values.tolist())) #Product first cell

print("=============>>>>>>product\_l",product\_l)

project\_code =''

n=1

for i in product\_l:

if n == 1:

project\_code = i

else:

project\_code = project\_code+'&'+i

n=n+1

return project\_code

def get\_pst\_time():

date\_format= '%Y-%m-%d' #'%m-%d%Y\_%H\_%M\_%S\_%Z'

date = datetime.datetime.now(tz=pytz.utc)

date = date.astimezone(timezone('US/Pacific'))

pstDateTime=date.strftime(date\_format)

return pstDateTime

def open\_all\_csv\_local(event,all\_csv\_path,data\_select,remove\_fail):

all\_csv\_path = os.path.join(all\_csv\_path+ '')

tmp\_lst = []

with open(all\_csv\_path, 'r') as f:

reader = csv.reader(f)

i = 1

for row in reader:

# print(row[0].lower())

# if row[0].lower().find('fct') != -1:

# # print("FW version---->")

# pass

if row[0].lower().find('upper limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('lower limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('measurement unit') != -1: # "Measurement Unit ----->" in row:

pass

elif row[0].lower().find('site') != -1:

tmp\_lst.append(row)

else:

tmp\_lst.append(row)

i = i + 1

# print("index---->", tmp\_lst[0])

param\_item\_start\_index = tmp\_lst[0].index('Test Stop Time')+1

# print('location parametric:',param\_item\_start\_index)

header\_list = tmp\_lst[0]

df = pd.DataFrame(tmp\_lst[1:], columns=tmp\_lst[0])

try:

pd.to\_datetime(df['Test Start Time'])

except Exception as e:

print('check Test Start Time,csv format wrong!')

return e

header\_df =df[0:2]

data\_df = df[2:]

print('csv data row number before remove SN empty--->', len(data\_df.values.tolist()))

data\_df=data\_df[~data\_df['SerialNumber'].isin([''])]#Remove SN Empty

print('csv data row number after remove SN empty--->', len(data\_df.values.tolist()))

print('csv data row number before remove fail--->', len(data\_df.values.tolist()))

if remove\_fail== 'yes':

data\_df=data\_df[data\_df['PASS/FAIL'].isin(['PASS'])]

# data\_df=data\_df[~data\_df['Test Pass/Fail Status'].isin(['FAIL'])]

print('csv data row number after remove fail--->', len(data\_df.values.tolist()))

print('csv data row number before remove retest--->', len(data\_df.values.tolist()))

if data\_select == 'first':

data\_df = data\_df.sort\_values(axis=0,by=['Test Start Time'],ascending='True')

data\_df.drop\_duplicates(['SerialNumber'],keep='first',inplace=True)

elif data\_select == 'last':

data\_df = data\_df.sort\_values(axis=0,by=['Test Start Time'],ascending='True')

data\_df.drop\_duplicates(['SerialNumber'],keep='last',inplace=True)

elif data\_select == 'no\_retest':

data\_df.drop\_duplicates(['SerialNumber'],keep=False,inplace=True)

elif data\_select == 'all':

pass

print('csv data row number after remove retest--->', len(data\_df.values.tolist()))

project\_code,build\_stage,station\_name = '','',''

start\_time\_l = data\_df['Test Start Time'].values.tolist() #StartTime

if len(start\_time\_l)>0:

start\_time\_first = min(start\_time\_l)

start\_time\_last = max(start\_time\_l)

else:

start\_time\_first = ''

start\_time\_last = ''

print('<first time -- last time>',start\_time\_first,start\_time\_last)

df = header\_df.append(data\_df)

# print('df after--->', df)

# print('df.values ---->', df.values)#array([[ ]])

return header\_list,df,project\_code,build\_stage,station\_name,start\_time\_first,start\_time\_last,param\_item\_start\_index

def open\_all\_csv(event,all\_csv\_path,data\_select,remove\_fail):

all\_csv\_path = os.path.join(all\_csv\_path+ '')

tmp\_lst = []

with open(all\_csv\_path, 'r') as f:

reader = csv.reader(f)

i = 1

for row in reader:

# print(row[0].lower())

# if row[0].lower().find('fct') != -1:

# # print("FW version---->")

# pass

if row[0].lower().find('display name') != -1:

pass

elif row[0].lower().find('pdca priority') != -1:

pass

elif row[0].lower().find('upper limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('lower limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('measurement unit') != -1: # "Measurement Unit ----->" in row:

pass

elif row[0].lower().find('site') != -1:

tmp\_lst.append(row)

else:

tmp\_lst.append(row)

i = i + 1

# print("index---->", tmp\_lst[0])

param\_item\_start\_index = tmp\_lst[0].index('Parametric')

# print('location parametric:',param\_item\_start\_index)

header\_list = tmp\_lst[1]

df = pd.DataFrame(tmp\_lst[2:], columns=tmp\_lst[1])

try:

pd.to\_datetime(df['StartTime'])

except Exception as e:

print('check StartTime,csv format wrong!')

return e

header\_df =df[0:2]

# print('header\_df before--->', header\_df)

data\_df = df[2:]

# print('data\_df before--->', data\_df)

print('csv data row number before remove SN empty--->', len(data\_df.values.tolist()))

data\_df=data\_df[~data\_df['SerialNumber'].isin([''])]#Remove SN Empty

print('csv data row number after remove SN empty--->', len(data\_df.values.tolist()))

print('csv data row number before remove fail--->', len(data\_df.values.tolist()))

if remove\_fail== 'yes':

data\_df=data\_df[data\_df['Test Pass/Fail Status'].isin(['PASS'])]

# data\_df=data\_df[~data\_df['Test Pass/Fail Status'].isin(['FAIL'])]

print('csv data row number after remove fail--->', len(data\_df.values.tolist()))

print('csv data row number before remove retest--->', len(data\_df.values.tolist()))

if data\_select == 'first':

data\_df = data\_df.sort\_values(axis=0,by=['StartTime'],ascending='True')

data\_df.drop\_duplicates(['SerialNumber'],keep='first',inplace=True)

elif data\_select == 'last':

data\_df = data\_df.sort\_values(axis=0,by=['StartTime'],ascending='True')

data\_df.drop\_duplicates(['SerialNumber'],keep='last',inplace=True)

elif data\_select == 'no\_retest':

data\_df.drop\_duplicates(['SerialNumber'],keep=False,inplace=True)

elif data\_select == 'all':

pass

print('csv data row number after remove retest--->', len(data\_df.values.tolist()))

#print("----------------<<<1>>>>>>>>>>>>>>")

if event == 'keynote-report' or event == 'excel-report':

project\_code,build\_stage,station\_name = get\_project\_info(data\_df)

else:

project\_code,build\_stage,station\_name = '','',''

start\_time\_l = data\_df['StartTime'].values.tolist() #StartTime

if len(start\_time\_l)>0:

start\_time\_first = min(start\_time\_l)

start\_time\_last = max(start\_time\_l)

else:

start\_time\_first = ''

start\_time\_last = ''

print('<first time -- last time>',start\_time\_first,start\_time\_last)

df = header\_df.append(data\_df)

# if event != 'one\_item\_plot':

# station\_id\_l = df['Station ID'].values.tolist()

# fixture\_channel\_id = df['Fixture Channel ID'].values.tolist()

# # print('station\_id\_l:',station\_id\_l)

# # print('fixture\_channel\_id:',fixture\_channel\_id)

# temp\_l = []

# for i in range(0,len(station\_id\_l[2:])):

# temp\_l.append(station\_id\_l[i+2]+'\_'+fixture\_channel\_id[i+2])

# temp\_l.insert(0,'')

# temp\_l.insert(0,'')

# fixture\_channel\_id = temp\_l

# # print(fixture\_channel\_id)

# df['Fixture Channel ID'] = pd.DataFrame({'Fixture Channel ID':fixture\_channel\_id})

# # print('fixture channel id:',df['Fixture Channel ID'].values.tolist())

# print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

# print('df after--->', df)

# print('df.values ---->', df.values)#array([[ ]])

return header\_list,df,project\_code,build\_stage,station\_name,start\_time\_first,start\_time\_last,param\_item\_start\_index

def create\_report(event,header\_list,df,color\_by1,pic\_path,select\_category1,cpk\_lsl,cpk\_usl,

save\_all\_cpk\_path,set\_bins,excel\_name,project\_code,build\_stage,station\_name,

start\_time\_first,start\_time\_last,color\_by2,select\_category\_l2,excel\_report\_item,

fail\_plot\_to\_excel,zoom\_type,param\_item\_start\_index):

clear\_files(pic\_path)

book,report\_sheet,plot\_sheet,ssh\_sheet,format\_highlight,format\_normal,format\_titile,new\_format\_fail,new\_format\_pass = creat\_excel\_report\_file(save\_all\_cpk\_path,excel\_name,cpk\_lsl,cpk\_usl,event,fail\_plot\_to\_excel)

# excel\_report\_data,excel\_report\_table,max\_row,max\_col = load\_excel\_table(save\_all\_cpk\_path+excel\_name)

table\_data,table\_category\_data,no\_valid\_column\_name\_l = parse\_all\_csv\_local(header\_list,df,color\_by1,select\_category1,event,color\_by2,select\_category\_l2,param\_item\_start\_index)#

i,j,n,t=0,0,0,0

path=save\_all\_cpk\_path

result='pass'

for column\_data in table\_data:

item\_name=column\_data[0]

wr\_excel = 'No'

# if str(item\_name).lower() != 'head id':

usl = column\_data[1]

lsl = column\_data[2]

column\_data = column\_data[3:]

# print('item\_name:',item\_name)

# print('---->>usl:',usl)

# print('---->>lsl:',lsl)

# print(str(t)+' column test value:',len(column\_data),column\_data)

if len(column\_data) >0:

bc,p\_val,a\_Q,a\_irr,three\_CV = get\_coefficients(column\_data)

BMC = ''

# if p\_val\_checked == '1':

if bc != '' and bc != 'Nan' and p\_val != '' and a\_Q != '' and p\_val != 'Nan' and a\_Q != 'Nan':

if float(p\_val) <= float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) <= float(a\_Q) and float(bc)<0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))>=-0.1:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))<-0.1:

BMC = 'NO'

else:

BMC = ''

else:

BMC = ''

row\_data = []

target\_value = 9999999999

mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk = cpk\_calc(column\_data, lsl, usl)

if cpk:

if is\_number(str(cpk)):

if float(cpk)>10:

BMC = ''

result=''

if stdev == 0 or stdev == 'nan':

n=n+1

# print('old limit--->')

# print(excel\_report\_item,item\_name+': do not draw plot due to stdev == 0!')#stdev ==0

location = 'A' + str(j+2)

result='Nan'

target\_value = get\_target\_value(lsl,usl)

# print('target\_value--->',lsl,usl,target\_value)

if p\_val\_checked == '1':

if event=='excel-report' and excel\_report\_item == 'all':#'Fail'

row\_data = [j + 1, item\_name,bc,p\_val,a\_Q,a\_irr,three\_CV,

lsl,target\_value, usl, min\_num, mean, max\_num,

stdev, cpl, cpu, cpk, result,'','','','','','','']

report\_sheet.write\_row(location,row\_data,format\_normal)

wr\_excel = 'Yes'

else:

if event=='excel-report' and excel\_report\_item == 'all':#'Fail'

row\_data = [j + 1, item\_name,bc,a\_Q,a\_irr,three\_CV,lsl,

target\_value, usl, min\_num, mean, max\_num,

stdev, cpl, cpu, cpk, result,'','','','','','','']

report\_sheet.write\_row(location,row\_data,format\_normal)

wr\_excel = 'Yes'

else:

location = 'A' + str(j+2)

# print("location,j--->",location,j)

# print('cpk:',cpk)

if cpk == None:

result='Nan'

target\_value = get\_target\_value(lsl,usl)

# print('target\_value--->',lsl,usl,target\_value)

if p\_val\_checked == '1':

if event=='excel-report' and excel\_report\_item == 'all':#'Fail'

row\_data = [str(j + 1), item\_name,bc,p\_val,a\_Q,a\_irr,

three\_CV,lsl,target\_value, usl, min\_num,

mean, max\_num, stdev, cpl, cpu, cpk, result,'','','','','','','']

report\_sheet.write\_row(location,row\_data,format\_normal)

wr\_excel = 'Yes'

else:

if event=='excel-report' and excel\_report\_item == 'all':#'Fail'

row\_data = [str(j + 1), item\_name,bc,a\_Q,a\_irr,three\_CV,lsl,

target\_value, usl, min\_num, mean, max\_num, stdev,

cpl, cpu, cpk, result,'','','','','','','']

report\_sheet.write\_row(location,row\_data,format\_normal)

wr\_excel = 'Yes'

else:

if cpk < cpk\_lsl or cpk > cpk\_usl:

# print('----------------------cpk value fail--------------------------')

# print('cpk\_lsl:',cpk\_lsl)

# print('cpk\_usl:',cpk\_usl)

result='Fail'

target\_value = get\_target\_value(lsl,usl)

# print('target\_value--->',lsl,usl,target\_value)

if p\_val\_checked == '1':

row\_data = [j + 1, item\_name,bc,p\_val,a\_Q,a\_irr,three\_CV,

lsl,target\_value, usl, min\_num, mean, max\_num,

stdev, cpl, cpu, cpk, result,'','','','','','','']

report\_sheet.write\_row(location,row\_data,format\_normal)

report\_sheet.write(j+1,17,result,new\_format\_fail)

else:

row\_data = [j + 1, item\_name,bc,a\_Q,a\_irr,three\_CV,

lsl,target\_value, usl, min\_num, mean,

max\_num, stdev, cpl, cpu, cpk, result,'','','','','','','']

report\_sheet.write\_row(location,row\_data,format\_normal)

report\_sheet.write(j+1,16,result,new\_format\_fail)

wr\_excel = 'Yes'

if not os.path.exists(path):

os.makedirs(path)

if fail\_plot\_to\_excel == 'yes' and event =='excel-report':

image\_name = item\_name.replace('/','\_')+".png"

pic\_path = '/tmp/CPK\_Log/fail\_plot/' + image\_name

# if fail\_plot\_to\_excel == 'yes' and event =='excel-report':

if len(table\_category\_data) == 0:

# print("============pic\_path:",pic\_path,image\_name)

draw\_histogram(column\_data,item\_name,lsl, usl,

mean, max\_num, min\_num, stdev, x1, y1,

cpu, cpl, cpk, pic\_path,set\_bins,

start\_time\_first,start\_time\_last,BMC,zoom\_type)

else:

draw\_more\_histogram(table\_category\_data[t],column\_data,

item\_name,lsl, usl, mean, max\_num, min\_num,

stdev, x1, y1, cpu, cpl, cpk, pic\_path,

set\_bins,start\_time\_first,start\_time\_last,zoom\_type)

if i==0:

location='A1'

else:

location = 'A'+str(30\*i)

if fail\_plot\_to\_excel == 'yes' and event =='excel-report':

print('=====save\_image\_to\_excel pic\_path:',pic\_path)

save\_image\_to\_excel(location,pic\_path,plot\_sheet)

i = i + 1

else:

if excel\_report\_item == 'all' or event == 'keynote-report':#'Fail'

result='Pass'

target\_value = get\_target\_value(lsl,usl)

# print('target\_value--->',lsl,usl,target\_value)

if p\_val\_checked == '1':

row\_data = [j + 1, item\_name,bc,p\_val,a\_Q,a\_irr,

three\_CV,lsl,target\_value,usl, min\_num,

mean, max\_num, stdev, cpl, cpu, cpk,

result,'','','','','','','']

report\_sheet.write\_row(location,row\_data,format\_normal)

report\_sheet.write(j+1,17,result,new\_format\_pass)

wr\_excel = 'Yes'

else:

row\_data = [j + 1, item\_name,bc,a\_Q,a\_irr,three\_CV,

lsl,target\_value,usl, min\_num, mean,

max\_num, stdev, cpl, cpu, cpk, result,'','','','','','','']

report\_sheet.write\_row(location,row\_data,format\_normal)

report\_sheet.write(j+1,16,result,new\_format\_pass)

wr\_excel = 'Yes'

else:

pass

#--------------------------------get new limit from csv start------------------------

new\_lsl,new\_usl,reviewer,review\_date,comment = get\_one\_item\_new\_limit\_from\_csv(save\_all\_cpk\_path,item\_name)

new\_lsl\_backup,new\_usl\_backup = new\_lsl,new\_usl

# print('=========new----',new\_lsl\_backup,new\_usl\_backup)

new\_lsl,new\_usl = verify\_limit(new\_lsl,new\_usl)

old\_mean, old\_max\_num,old\_target\_value,old\_min\_num, old\_stdev,old\_cpu, old\_cpl, old\_cpk,old\_lsl,old\_usl,old\_result = mean, max\_num,target\_value,min\_num, stdev, cpu, cpl, cpk,lsl,usl,result

if new\_lsl != None :

lsl = new\_lsl

if new\_usl != None :

usl = new\_usl

if new\_lsl == None and new\_usl != None:

lsl = ''

if new\_lsl != None and new\_usl == None:

usl = ''

if new\_lsl != None or new\_usl != None:

#--------------------------------get new limit from csv end------------------------

mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk = cpk\_calc(column\_data, lsl, usl)

if stdev == 0 or stdev == 'nan':

n=n+1

result = 'Nan'

target\_value = get\_target\_value(lsl,usl)

print(item\_name+': stdev == 0!(new limit)')#stdev ==0

if event=='excel-report' and excel\_report\_item == 'all':#'Fail'

if p\_val\_checked == '1':

report\_sheet.write(j+1,18,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,19,target\_value,format\_highlight)

report\_sheet.write(j+1,20,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,21,cpl,format\_highlight)

report\_sheet.write(j+1,22,cpu,format\_highlight)

report\_sheet.write(j+1,23,cpk,format\_highlight)

report\_sheet.write(j+1,24,result,format\_highlight)

else:

report\_sheet.write(j+1,17,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,18,target\_value,format\_highlight)

report\_sheet.write(j+1,19,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,20,cpl,format\_highlight)

report\_sheet.write(j+1,21,cpu,format\_highlight)

report\_sheet.write(j+1,22,cpk,format\_highlight)

report\_sheet.write(j+1,23,result,format\_highlight)

wr\_excel = 'Yes'

else:

location = 'A' + str(j+2)

# print("location,j--->",location,j)

# print('cpk:',cpk)

if cpk == None:

result='Nan'

target\_value = get\_target\_value(lsl,usl)

# print('target\_value--->',lsl,usl,target\_value)

# append\_new\_limit\_data\_to\_excel\_report(excel\_report\_data,excel\_report\_table,save\_all\_cpk\_path+excel\_name,j+2,lsl,usl,cpl,cpu,cpk)

if p\_val\_checked == '1':

row\_data = [j + 1, item\_name,bc,p\_val,a\_Q,a\_irr,three\_CV,

old\_lsl,old\_target\_value, old\_usl, old\_min\_num,

old\_mean, old\_max\_num, old\_stdev, old\_cpl,

old\_cpu, old\_cpk, old\_result,lsl,target\_value,usl,cpl,cpu,cpk,result]

report\_sheet.write\_row(location,row\_data,format\_normal)

if old\_result == 'Fail':

report\_sheet.write(j+1,17,old\_result,new\_format\_fail)

elif old\_result == 'Pass':

report\_sheet.write(j+1,17,old\_result,new\_format\_pass)

report\_sheet.write(j+1,18,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,19,target\_value,format\_highlight)

report\_sheet.write(j+1,20,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,21,cpl,format\_highlight)

report\_sheet.write(j+1,22,cpu,format\_highlight)

report\_sheet.write(j+1,23,cpk,format\_highlight)

report\_sheet.write(j+1,24,result,new\_format\_fail)

else:

row\_data = [j + 1, item\_name,bc,a\_Q,a\_irr,three\_CV,

old\_lsl,old\_target\_value, old\_usl,

old\_min\_num, old\_mean, old\_max\_num,

old\_stdev, old\_cpl, old\_cpu, old\_cpk,

old\_result,lsl,target\_value,usl,cpl,cpu,cpk,result]

report\_sheet.write\_row(location,row\_data,format\_normal)

if old\_result == 'Fail':

report\_sheet.write(j+1,16,old\_result,new\_format\_fail)

elif old\_result == 'Pass':

report\_sheet.write(j+1,16,old\_result,new\_format\_pass)

report\_sheet.write(j+1,17,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,18,target\_value,format\_highlight)

report\_sheet.write(j+1,19,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,20,cpl,format\_highlight)

report\_sheet.write(j+1,21,cpu,format\_highlight)

report\_sheet.write(j+1,22,cpk,format\_highlight)

report\_sheet.write(j+1,23,result,new\_format\_fail)

wr\_excel = 'Yes'

picFail\_path = '/tmp/CPK\_Log/fail\_plot/'

if not os.path.exists(picFail\_path):

os.makedirs(picFail\_path)

if fail\_plot\_to\_excel == 'yes' and event =='excel-report':

image\_name = item\_name.replace('/','\_')+" new\_limit.png"

pic\_path = picFail\_path + image\_name

if len(table\_category\_data) == 0:

draw\_histogram(column\_data,item\_name,lsl, usl, mean,

max\_num, min\_num, stdev, x1, y1, cpu,

cpl, cpk, pic\_path,set\_bins,start\_time\_first,

start\_time\_last,BMC,zoom\_type)

else:

#[[],[],[]...],only value category lists

draw\_more\_histogram(table\_category\_data[t],column\_data,

item\_name,lsl, usl, mean, max\_num,

min\_num, stdev, x1, y1, cpu, cpl,

cpk, pic\_path,set\_bins,start\_time\_first,

start\_time\_last,zoom\_type)

if i==0:

location='A1'

else:

location = 'A'+str(29\*i)

if fail\_plot\_to\_excel == 'yes' and event =='excel-report':

save\_image\_to\_excel(location,pic\_path,plot\_sheet)

i = i + 1

else:

if cpk < cpk\_lsl or cpk > cpk\_usl:

# print('----------------------cpk value result fail--------------------------')

# print('cpk\_lsl:',cpk\_lsl)

# print('cpk\_usl:',cpk\_usl)

result='Fail'

target\_value = get\_target\_value(lsl,usl)

# print('target\_value--->',lsl,usl,target\_value)

# append\_new\_limit\_data\_to\_excel\_report(excel\_report\_data,excel\_report\_table,save\_all\_cpk\_path+excel\_name,j+2,lsl,usl,cpl,cpu,cpk)

if p\_val\_checked == '1':

row\_data = [j + 1, item\_name,bc,p\_val,a\_Q,a\_irr,three\_CV,

old\_lsl,old\_target\_value, old\_usl, old\_min\_num,

old\_mean, old\_max\_num, old\_stdev, old\_cpl,

old\_cpu, old\_cpk, old\_result,lsl,target\_value,usl,cpl,cpu,cpk,result]

report\_sheet.write\_row(location,row\_data,format\_normal)

if old\_result == 'Fail':

report\_sheet.write(j+1,17,old\_result,new\_format\_fail)

elif old\_result == 'Pass':

report\_sheet.write(j+1,17,old\_result,new\_format\_pass)

report\_sheet.write(j+1,18,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,19,target\_value,format\_highlight)

report\_sheet.write(j+1,20,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,21,cpl,format\_highlight)

report\_sheet.write(j+1,22,cpu,format\_highlight)

report\_sheet.write(j+1,23,cpk,format\_highlight)

report\_sheet.write(j+1,24,result,new\_format\_fail)

else:

row\_data = [j + 1, item\_name,bc,a\_Q,a\_irr,three\_CV,

old\_lsl,old\_target\_value, old\_usl, old\_min\_num,

old\_mean, old\_max\_num, old\_stdev, old\_cpl, old\_cpu,

old\_cpk, old\_result,lsl,target\_value,usl,cpl,cpu,cpk,result]

report\_sheet.write\_row(location,row\_data,format\_normal)

if old\_result == 'Fail':

report\_sheet.write(j+1,16,old\_result,new\_format\_fail)

elif old\_result == 'Pass':

report\_sheet.write(j+1,16,old\_result,new\_format\_pass)

report\_sheet.write(j+1,17,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,18,target\_value,format\_highlight)

report\_sheet.write(j+1,19,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,20,cpl,format\_highlight)

report\_sheet.write(j+1,21,cpu,format\_highlight)

report\_sheet.write(j+1,22,cpk,format\_highlight)

report\_sheet.write(j+1,23,result,new\_format\_fail)

wr\_excel = 'Yes'

picFail\_path = '/tmp/CPK\_Log/fail\_plot/'

if not os.path.exists(picFail\_path):

os.makedirs(picFail\_path)

if fail\_plot\_to\_excel == 'yes' and event =='excel-report':

image\_name = item\_name.replace('/','\_')+" new\_limit.png"

pic\_path = picFail\_path + image\_name

if len(table\_category\_data) == 0:

draw\_histogram(column\_data,item\_name,lsl, usl, mean,

max\_num, min\_num, stdev, x1, y1, cpu, cpl,

cpk, pic\_path,set\_bins,start\_time\_first,

start\_time\_last,BMC,zoom\_type)

else:

#[[],[],[]...],only value category lists

draw\_more\_histogram(table\_category\_data[t],column\_data,

item\_name,lsl, usl, mean, max\_num,

min\_num, stdev, x1, y1, cpu, cpl, cpk,

pic\_path,set\_bins,start\_time\_first,start\_time\_last,zoom\_type)

if i==0:

location='A1'

else:

location = 'A'+str(29\*i)

if fail\_plot\_to\_excel == 'yes' and event =='excel-report':

save\_image\_to\_excel(location,pic\_path,plot\_sheet)

i = i + 1

else:

if excel\_report\_item == 'all' or event == 'keynote-report':#'Fail'

result='Pass'

target\_value = get\_target\_value(lsl,usl)

# print('target\_value--->',lsl,usl,target\_value)

# append\_new\_limit\_data\_to\_excel\_report(excel\_report\_data,excel\_report\_table,save\_all\_cpk\_path+excel\_name,j+2,lsl,usl,cpl,cpu,cpk)

if p\_val\_checked == '1':

row\_data = [j + 1, item\_name,bc,p\_val,a\_Q,a\_irr,

three\_CV,old\_lsl,old\_target\_value,

old\_usl, old\_min\_num, old\_mean, old\_max\_num,

old\_stdev, old\_cpl, old\_cpu, old\_cpk, old\_result,

lsl,target\_value,usl,cpl,cpu,cpk,result]

report\_sheet.write\_row(location,row\_data,format\_normal)

if old\_result == 'Fail':

report\_sheet.write(j+1,17,old\_result,new\_format\_fail)

elif old\_result == 'Pass':

report\_sheet.write(j+1,17,old\_result,new\_format\_pass)

report\_sheet.write(j+1,18,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,19,target\_value,format\_highlight)

report\_sheet.write(j+1,20,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,21,cpl,format\_highlight)

report\_sheet.write(j+1,22,cpu,format\_highlight)

report\_sheet.write(j+1,23,cpk,format\_highlight)

report\_sheet.write(j+1,24,result,new\_format\_pass)

else:

row\_data = [j + 1, item\_name,bc,a\_Q,a\_irr,three\_CV,

old\_lsl,old\_target\_value, old\_usl, old\_min\_num,

old\_mean, old\_max\_num, old\_stdev, old\_cpl,

old\_cpu, old\_cpk, old\_result,lsl,target\_value,usl,cpl,cpu,cpk,result]

report\_sheet.write\_row(location,row\_data,format\_normal)

if old\_result == 'Fail':

report\_sheet.write(j+1,16,old\_result,new\_format\_fail)

elif old\_result == 'Pass':

report\_sheet.write(j+1,16,old\_result,new\_format\_pass)

report\_sheet.write(j+1,17,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,18,target\_value,format\_highlight)

report\_sheet.write(j+1,19,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,20,cpl,format\_highlight)

report\_sheet.write(j+1,21,cpu,format\_highlight)

report\_sheet.write(j+1,22,cpk,format\_highlight)

report\_sheet.write(j+1,23,result,new\_format\_pass)

wr\_excel = 'Yes'

# print('reviewer,review\_date----->',reviewer,review\_date)

elif new\_lsl\_backup == 'NA' or new\_usl\_backup == 'NA':

result='Nan'

target\_value = 'NA'

cpl = 'NA'

cpu = 'NA'

cpk = 'NA'

if p\_val\_checked == '1':

report\_sheet.write(j+1,18,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,19,target\_value,format\_highlight)

report\_sheet.write(j+1,20,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,21,cpl,format\_highlight)

report\_sheet.write(j+1,22,cpu,format\_highlight)

report\_sheet.write(j+1,23,cpk,format\_highlight)

report\_sheet.write(j+1,24,result,new\_format\_fail)

wr\_excel = 'Yes'

else:

report\_sheet.write(j+1,17,new\_lsl\_backup,format\_highlight)

report\_sheet.write(j+1,18,target\_value,format\_highlight)

report\_sheet.write(j+1,19,new\_usl\_backup,format\_highlight)

report\_sheet.write(j+1,20,cpl,format\_highlight)

report\_sheet.write(j+1,21,cpu,format\_highlight)

report\_sheet.write(j+1,22,cpk,format\_highlight)

report\_sheet.write(j+1,23,result,new\_format\_fail)

wr\_excel = 'Yes'

if p\_val\_checked == '1':

report\_sheet.write(j+1,25,reviewer,format\_normal)

report\_sheet.write(j+1,26,review\_date,format\_normal)

report\_sheet.write(j+1,27,comment,format\_normal)

else:

report\_sheet.write(j+1,24,reviewer,format\_normal)

report\_sheet.write(j+1,25,review\_date,format\_normal)

report\_sheet.write(j+1,26,comment,format\_normal)

if event == 'excel-report' and excel\_report\_item == 'all' or event == 'excel-report' and excel\_report\_item == 'fail' and old\_result =='Fail' or event == 'excel-report' and excel\_report\_item == 'fail' and result =='Fail' or stdev != 0 and stdev != 'nan' and old\_cpk != None and event == 'keynote-report':

# print(stdev,event,'add j:')

j=j+1

else:

print('>data empty ,no record in excel!')

t = t + 1

# print('Total {} items not calulate cpk'.format('%.0f' % n))

if event == 'excel-report' and excel\_report\_item == 'all':

write\_invalid\_item\_to\_excel(no\_valid\_column\_name\_l,report\_sheet,j,format\_normal,save\_all\_cpk\_path)

book.close()

print('All items excel report finished!')

def local\_csv\_generate\_excel(table\_data):

print('---local csv generate excel')

# exportAllItems,exportPassItems,cpkLow,cpkHigh,populate,userName,projectName,targetBuild,cpk\_path,set\_bin,csv\_data\_Path

global filelogname

global filelognamehash

cpk\_lsl = table\_data[2]

# cpk\_usl = table\_data[3]

cpk\_usl = float("inf")

excel\_report\_user = table\_data[5]

project\_name = table\_data[6]

excel\_report\_stage = table\_data[7]

cpk\_path = table\_data[8]#"/Users/RyanGao/Desktop/CPK\_Log/"

set\_bins = table\_data[9] #250

filelogname = '/tmp/CPK\_Log/temp/.excel.txt'

filelognamehash = '/tmp/CPK\_Log/temp/.excel\_hash.txt'

all\_csv\_path = table\_data[10] #'/tmp/CPK\_Log/Temp/keynote\_data\_temp.csv'

data\_select = 'all'

remove\_fail = 'yes'

event = 'excel-report'

start\_time = get\_pst\_time()

excel\_report\_file\_name ='Limit\_Update\_'+str(excel\_report\_user)+'\_'+str(project\_name)+'\_For\_'+str(excel\_report\_stage)+'\_'+str(start\_time)+'.xlsx'#%Y-%m-%d\_%H-%M-%S

with open('/tmp/CPK\_Log/temp/.excelreportname.txt', 'w') as file\_object:

file\_object.write(excel\_report\_file\_name)

zoom\_type = 'limit'

color\_by1 = 'Off'

fail\_pic\_path =cpk\_path+'fail\_plot/'

select\_category\_l1 =[]

color\_by2 = 'Off'

select\_category\_l2 =[]

exportAllItems = table\_data[0]

exportPassItems = table\_data[1]

excel\_report\_item = 'all' #fail all

if exportAllItems == 1:

excel\_report\_item = 'all'

if exportPassItems == 1:

excel\_report\_item = 'fail'

populate = table\_data[4]

fail\_plot\_to\_excel = 'no'#'no' # yes

if populate==1:

fail\_plot\_to\_excel = 'yes'

header\_list,df,project\_code,build\_stage,station\_name,start\_time\_first,start\_time\_last,param\_item\_start\_index = open\_all\_csv\_local(event,all\_csv\_path,data\_select,remove\_fail)

create\_report(event,header\_list,df,color\_by1,fail\_pic\_path,select\_category\_l1,cpk\_lsl,

cpk\_usl,cpk\_path,set\_bins,excel\_report\_file\_name,project\_code,build\_stage,station\_name,

start\_time\_first,start\_time\_last,color\_by2,select\_category\_l2,excel\_report\_item,

fail\_plot\_to\_excel,zoom\_type,param\_item\_start\_index)

print('['+str(start\_time)+' '+str(datetime.datetime.now())+']','create excel report finished!')

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished," + excel\_report\_name)

def generate\_excel(table\_data):

# exportAllItems,exportPassItems,cpkLow,cpkHigh,populate,userName,projectName,targetBuild,cpk\_path,set\_bin,csv\_data\_Path

global filelogname

global filelognamehash

global p\_val\_checked

cpk\_lsl = table\_data[2]

# cpk\_usl = table\_data[3]

cpk\_usl = float("inf")

excel\_report\_user = table\_data[5]

project\_name = table\_data[6]

excel\_report\_stage = table\_data[7]

cpk\_path = table\_data[8] #"[NSString stringWithFormat:@"%@/CPK\_Log/",desktopPath]"

set\_bins = table\_data[9] #250

filelogname = '/tmp/CPK\_Log/temp/.excel.txt'

filelognamehash = '/tmp/CPK\_Log/temp/.excel\_hash.txt'

all\_csv\_path = table\_data[10] #'[m\_configDictionary valueForKey:Load\_Csv\_Path] raw data for load path'

p\_val\_checked = str(table\_data[14]) #choose p\_val

data\_select = 'all'

remove\_fail = 'yes'

event = 'excel-report'

start\_time = get\_pst\_time() #datetime.datetime.now()

excel\_report\_file\_name ='Limit\_Update\_'+str(excel\_report\_user)+'\_'+str(project\_name)+'\_For\_'+str(excel\_report\_stage)+'\_'+str(start\_time)+'.xlsx'#%Y-%m-%d\_%H-%M-%S

with open('/tmp/CPK\_Log/temp/.excelreportname.txt', 'w') as file\_object:

file\_object.write(excel\_report\_file\_name)

zoom\_type = 'limit'

color\_by1 = 'Off'

fail\_pic\_path ='/tmp/CPK\_Log/fail\_plot/'

select\_category\_l1 =[]

color\_by2 = 'Off'

select\_category\_l2 =[]

exportAllItems = table\_data[0]

exportPassItems = table\_data[1]

excel\_report\_item = 'all' #fail all

if exportAllItems == 1:

excel\_report\_item = 'all'

if exportPassItems == 1:

excel\_report\_item = 'fail'

populate = table\_data[4]

fail\_plot\_to\_excel = 'no'#'no' # yes

if populate==1:

fail\_plot\_to\_excel = 'yes'

header\_list,df,project\_code,build\_stage,station\_name,start\_time\_first,start\_time\_last,param\_item\_start\_index = open\_all\_csv(event,all\_csv\_path,data\_select,remove\_fail)

create\_report(event,header\_list,df,color\_by1,fail\_pic\_path,select\_category\_l1,cpk\_lsl,cpk\_usl,

cpk\_path,set\_bins,excel\_report\_file\_name,project\_code,build\_stage,station\_name,

start\_time\_first,start\_time\_last,color\_by2,select\_category\_l2,excel\_report\_item,

fail\_plot\_to\_excel,zoom\_type,param\_item\_start\_index)

print('==[='+str(start\_time)+' '+str(datetime.datetime.now())+']','create excel report finished!')

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished," + excel\_report\_name)

def excel\_hash\_to\_csv(excel\_path,csv\_hash\_path):

data = pd.read\_excel(excel\_path,'ssh',index\_col=0,keep\_default\_na=False)

data.to\_csv(csv\_hash\_path,encoding='utf-8')

def excel\_limitupdate\_to\_csv(excel\_path,csv\_limit\_path):

data = pd.read\_excel(excel\_path,'report',index\_col=0,keep\_default\_na=False)

data.to\_csv(csv\_limit\_path,encoding='utf-8')

def run(n):

while True:

try:

print("wait for excel client ...")

zmqMsg = socket.recv()

socket.send(b'excel.csv') # socket.send(ret.decode('utf-8').encode('ascii'))

if len(zmqMsg)>0:

key = zmqMsg.decode('utf-8')

print("message from excel client:", key)

if key == 'generate\_excel\_sheet1\_hash':

table\_data = get\_redis\_data(key)

if len(table\_data)>0:

append\_hash\_to\_excel(table\_data,excel\_report\_name,hash\_csv\_path)

else:

print("---get generate\_excel\_sheet1\_hash error")

with open(filelognamehash, 'w') as file\_object:

file\_object.write("Finished,create excel report error")

return

elif key == 'generate\_excel':

table\_data = get\_redis\_data(key)

if len(table\_data)>0:

generate\_excel(table\_data)

else:

print("---get generate\_excel error")

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished," + str(excel\_report\_name))

elif key == 'local\_csv\_generate\_excel':

table\_data = get\_redis\_data(key)

if len(table\_data)>0:

local\_csv\_generate\_excel(table\_data)

else:

print("---get local\_csv\_generate\_excel error")

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished," + str(excel\_report\_name))

elif key == 'excel\_hash\_to\_csv':

table\_data = get\_redis\_data(key)

if len(table\_data)>0:

excel\_path = table\_data[0]

csv\_hash\_path = table\_data[1]

excel\_hash\_to\_csv(excel\_path,csv\_hash\_path)

else:

print("---get excel\_hash\_to\_csv error")

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished," + str(excel\_report\_name))

elif key == 'excel\_limit\_update\_to\_csv\_report':

table\_data = get\_redis\_data(key)

if len(table\_data)>0:

excel\_path = table\_data[0]

csv\_limit\_path = table\_data[1]

excel\_limitupdate\_to\_csv(excel\_path,csv\_limit\_path)

else:

print("---get excel\_limit\_update\_to\_csv report error")

else:

time.sleep(0.05)

except Exception as e:

print('error excel:',e)

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,create excel report error: " + str(e))

with open(filelognamehash, 'w') as file\_object:

file\_object.write("Finished,create excel report error: " + str(e))

if \_\_name\_\_ == '\_\_main\_\_':

# t1 = threading.Thread(target=run, args=("<<correlation>>",))

# t1.start()

run(0)

# l=['All','yes',1.67,float("inf"),'','tom','j171a','pvt','/Users/rex/Desktop/CPK\_Log/',250,'/Users/rex/Desktop/CPK\_Log/loadFile/data/222---.csv']

# generate\_excel(l)

#! /usr/bin/env python3

# --\*-- coding: utf-8 ---\*---

import sys,os,time,math,re

import time

import threading

import datetime

from pytz import timezone

import pytz

BASE\_DIR=os.path.dirname(os.path.abspath(\_\_file\_\_))

sys.path.insert(0,BASE\_DIR+'/site-packages/')

try:

import csv

except Exception as e:

print('e---->',e)

try:

import matplotlib

matplotlib.use("Agg")

import matplotlib.pyplot as plt

except Exception as e:

print('e---->',e)

try:

import matplotlib.colors as colors

except Exception as e:

print('e---->',e)

try:

from matplotlib.font\_manager import FontProperties

except Exception as e:

print('e---->',e)

try:

import numpy as np

except Exception as e:

print('e--->',e)

try:

import pandas as pd

except Exception as e:

print('e--->',e)

try:

import openpyxl

except Exception as e:

print('import openpyxl error:',e)

try:

import xlsxwriter

except Exception as e:

print('import xlsxwriter error:',e)

try:

import diptest

except Exception as e:

print('import diptest error:',e)

current\_dir = os.path.dirname(os.path.realpath(\_\_file\_\_))

keynote\_lib\_path = current\_dir+'/python\_keynote'

try:

from python\_keynote import generate\_keynote

except Exception as e:

print('import python keynote---->',e)

try:

import zmq

except Exception as e:

print('import zmq error:',e)

try:

import redis

except Exception as e:

print('import redis error:',e)

print(sys.getdefaultencoding())

redisClient = redis.Redis(host='localhost', port=6379, db=0)

context = zmq.Context()

socket = context.socket(zmq.REP)

socket.setsockopt(zmq.LINGER,0)

socket.bind("tcp://127.0.0.1:3140")

def correlation(message):

print("this function is generate correlation plot......")

val = r.get(message)

# time.sleep(5) #测试python 执行时间 5s

if val:

return val

else:

return b'None'

def get\_redis\_data(zmqMsg):

tb = redisClient.get(zmqMsg)

tb\_data=[]

if tb:

tb=tb.decode('utf-8')

tb=tb.split("\n")

tb=(tb[1:-1]) #去掉数据库首尾元素

for i in tb:

k=re.sub('\"','',i) #去掉数据库引号

h=re.sub(',','',k) #去掉数据库逗号

m=h.strip() #去掉数据库首尾空白

if is\_number(m):

tb\_data.append(eval(m)) #去掉数字的引号

else:

tb\_data.append(m)

else:

tb\_data.append('')

return tb\_data

def is\_number(s):

try:

float(s)

return True

except ValueError:

pass

try:

import unicodedata

unicodedata.numeric(s)

return True

except (TypeError, ValueError):

pass

return False

def read\_csv\_to\_list(csv\_path):

tmp\_lst = []

with open(csv\_path, 'r') as f:

reader = csv.reader(f)

for i,row in enumerate(reader):

tmp\_lst = tmp\_lst + row

return tmp\_lst

def clear\_files(path):

for root, dirs, files in os.walk(path):

for file in files:

os.remove(path + '/' + file)

def valid\_column(test\_item\_name, column\_list):

if test\_item\_name.lower().find('fixture vendor\_id') == -1 and test\_item\_name.lower().find('unit number') == -1:

pass

else:

return 'not\_cpk state1'

if len(column\_list) < 3:

return "not\_cpk state2"

else:

j = 0

for i in range(2, len(column\_list), 1):

if is\_number(column\_list[i]):

j = j + 1

if j > 0 and len(set(column\_list)) >= 1: #

# print('need\_cpk')

return "need\_cpk"

return "not\_cpk state3"

def test\_value\_to\_numeric(test\_data\_list):

column\_list = []

i = 0

for x in test\_data\_list:

if (i ==0 and (x == 'NA' or x == '')) or (i ==1 and (x == 'NA' or x == '')):

column\_list.append(x)

else:

try:

x = float(x)

column\_list.append(x)

except Exception as e:

pass

# print('-------------------- it is not number --------------')

i = i + 1

return column\_list

def is\_empty\_list(l):

temp\_l = []

for value in l:

if value != '':

temp\_l.append(value)

n = len(temp\_l)

if n == 0:

# print('it is empty list!')

return True

else:

return False

def parse\_all\_csv(header\_list, df, color\_by1, selected\_category1, event, color\_by2, selected\_category2,param\_item\_start\_index):

color\_l = ['#0000FF', '#FF0000', '#008000', '#00FFFF', '#000000', '#8B008B', '#B8860B', '#FF6347', '#A9A9A9','#FFFF00', '#A52A2A', '#7FFF00', '#D2691E', '#6495ED', '#FF00FF']

table\_data = [] # [[]]

table\_category\_data = [] # [[[]]]

column\_list = []

n = 0

no\_valid\_column\_name\_l = []

for item\_name in header\_list:

try:

column\_list = df[item\_name].tolist()

except Exception as e:

if event == 'excel-report' and header\_list.index(item\_name) >= param\_item\_start\_index:

no\_valid\_column\_name\_l.append(item\_name)

print(item\_name, 'is duplicate ? pls check!', e)

continue

# print('column\_list--->',type(df[item\_name]),column\_list)

need\_cpk = valid\_column(item\_name, column\_list)

# print('need\_cpk:---->',need\_cpk)

column\_num\_list = []

if need\_cpk == 'need\_cpk':

column\_list = test\_value\_to\_numeric(column\_list)

column\_list.insert(0, item\_name) # item name

usl = column\_list[1]

lsl = column\_list[2]

# print('color\_by1:',color\_by1)

# 'Off'/'SerialNumber'/'Version'/'Station ID'/'Special Build Name'/'Product'/'StartTime'/'Special Build Description'

if color\_by1 == 'SerialNumber' or color\_by1 == 'Version' or color\_by1 == 'Station ID' or color\_by1 == 'Special Build Name' or color\_by1 == 'Product' or color\_by1 == 'StartTime' or color\_by1 == 'Special Build Description' or color\_by1 == 'Fixture Channel ID' or color\_by1 == 'Diags\_Version':

column\_temp = []

first\_filter\_category\_data\_len = 0

second\_filter\_category\_data\_len = 0

i = 0

for x in selected\_category1:

# print('x:',x,color\_by2)

if color\_by2 == 'SerialNumber' or color\_by2 == 'Version' or color\_by2 == 'Station ID' or color\_by2 == 'Special Build Name' or color\_by2 == 'Product' or color\_by2 == 'StartTime' or color\_by2 == 'Special Build Description' or color\_by2 == 'Fixture Channel ID' or color\_by2 == 'Diags\_Version':

for xx in selected\_category2:

if color\_by2 == 'Fixture Channel ID':

# print('the same!',df.columns.values.tolist()[14])

index2 = df.columns.values.tolist()[14]

index2 = color\_by2

one\_category\_list = df.loc[(df[color\_by1] == x[0]) & (df[index2] == xx[0]), item\_name].tolist() #

if is\_empty\_list(one\_category\_list) != True:

# print('one category\_data with second filter-->', len(one\_category\_list),one\_category\_list)

second\_filter\_category\_data\_len = second\_filter\_category\_data\_len + len(one\_category\_list)

one\_category\_list = test\_value\_to\_numeric(one\_category\_list)

column\_temp = column\_temp + one\_category\_list

category\_value = x[0] + '&' + xx[0]

one\_category\_list.insert(0, category\_value) # insert category

# print('i==>',i)

if i > 14:

i = 0

one\_category\_list.insert(1, color\_l[i]) # insert color

one\_category\_list.insert(2, item\_name) # item\_name

if lsl != 'NA':

lsl = float(lsl)

if usl != 'NA':

usl = float(usl)

one\_category\_list.insert(3, usl) # usl

one\_category\_list.insert(4, lsl) # lsl

# print('one\_category\_list-->', one\_category\_list)

column\_num\_list.append(one\_category\_list)

i = i + 1

elif color\_by2 == 'Off':

# print('color\_by2= Off xxxxx:', x[0],x[1])

one\_category\_list = df.loc[df[color\_by1] == x[0], item\_name].tolist() #

if is\_empty\_list(one\_category\_list) != True:

# print('one category\_data with first filter-->', one\_category\_list)

first\_filter\_category\_data\_len = first\_filter\_category\_data\_len + len(one\_category\_list)

one\_category\_list = test\_value\_to\_numeric(one\_category\_list)

column\_temp = column\_temp + one\_category\_list

one\_category\_list.insert(0, x[0]) # insert category

one\_category\_list.insert(1, x[1]) # insert color

one\_category\_list.insert(2, item\_name) # item\_name

# if lsl != 'NA' or usl != 'NA':

# usl = float(usl)

# lsl = float(lsl)

if lsl != 'NA':

lsl = float(lsl)

if usl != 'NA':

usl = float(usl)

one\_category\_list.insert(3, usl) # usl

one\_category\_list.insert(4, lsl) # lsl

# print('one\_category\_list-->', one\_category\_list)

column\_num\_list.append(one\_category\_list)

# print('one category\_data total with first filter-->', first\_filter\_category\_data\_len)

# print('one category\_data total with second filter-->', second\_filter\_category\_data\_len)

column\_temp.insert(0, item\_name) # item\_name

# if lsl != 'NA' or usl != 'NA':

# usl = float(usl)

# lsl = float(lsl)

if lsl != 'NA':

lsl = float(lsl)

if usl != 'NA':

usl = float(usl)

column\_temp.insert(1, usl) # usl

column\_temp.insert(2, lsl) # lsl

# print('column\_temp-->', column\_temp)

table\_data.append(column\_temp) # one column's all category data []

# print('column\_num\_list-->', column\_num\_list)

table\_category\_data.append(column\_num\_list) # category [[],[],...]]

elif color\_by1 == 'Off':

if event == 'excel-report' or event == 'keynote-report':

if header\_list.index(item\_name) >= param\_item\_start\_index:

table\_data.append(column\_list) # item\_name,usl,lsl,data

else:

table\_data.append(column\_list) # item\_name,usl,lsl,data

else:

if (event == 'excel-report' or event == 'keynote-report') and header\_list.index(item\_name) >= param\_item\_start\_index:

no\_valid\_column\_name\_l.append(item\_name)

if event == 'excel-report':

pass

else:

no\_valid\_column\_name\_l = []

# print('no\_valid\_column\_name\_l-->',no\_valid\_column\_name\_l)

return table\_data, table\_category\_data, no\_valid\_column\_name\_l # [[[ ]]]

def get\_coefficients(value\_l):

'''

param value\_l: need float list

return: bc,p\_val,a\_Q,a\_irr,three\_σ\_x100\_divide\_mean

1σ＝690000／1000000 #fault rate

2σ＝308000／1000000

3σ＝66800／1000000

4σ＝6210／1000000

5σ＝230／1000000

6σ＝3.4／1000000

7σ＝0／1000000

'''

if len(value\_l) <= 3:

return '','','','',''

column\_stdev = np.std(value\_l,ddof=1,axis=0)

three\_sigma= 3\*column\_stdev

# print('three\_sigma:',column\_stdev,three\_sigma)

temp\_l= value\_l

if len(value\_l) < 10:

temp\_l = value\_l + value\_l

if len(temp\_l)<10:

temp\_l = value\_l + value\_l + value\_l + value\_l+value\_l + value\_l + value\_l + value\_l+value\_l + value\_l

# print('------>>>temp\_l:',temp\_l)

data0 = np.array(temp\_l)

# print('------>>>data0:',data0)

try:

dip, p\_val = diptest.diptest(data0)

p\_val = '%f'%p\_val

# print('------>>>p\_val :',p\_val)

except RuntimeWarning as w:

print('calculate dip,p\_val RuntimeWarning:',w)

return '','','','',''

except Exception as e:

print('calculate dip,p\_val error:',e)

return '','','','',''

# print('dip,p\_val:',dip,p\_val)

item\_name ='value1'

data = pd.DataFrame({item\_name:value\_l})

# print('data--->',type(data),data)

u1 = data[item\_name].mean() # 计算均值

# std1 = data[item\_name].std() # 计算标准差

# t,pval=stats.kstest(data[item\_name], 'norm', (u1, std1))

# print('normality test t,pval:',str(t),str(pval))

# print('------')

# 正态性检验 → pvalue >0.05

n= float(len(value\_l))

#Item (xi-ẍ)^2

item\_l\_1 =[]

item\_l\_2 = []

item\_l\_3 = []

for i in value\_l:

temp1 = (i-u1)\*\*2

temp2 = (i-u1)\*\*3

temp3 = (i-u1)\*\*4

item\_l\_1.append(temp1)

item\_l\_2.append(temp2)

item\_l\_3.append(temp3)

sum\_item\_l\_1 = sum(item\_l\_1)

sum\_item\_l\_2 = sum(item\_l\_2)

sum\_item\_l\_3 = sum(item\_l\_3)

if n<=3 or sum\_item\_l\_1==0 or sum\_item\_l\_2==0 or sum\_item\_l\_3==0:

# print('len < 3--->')

if abs(u1) == 0:

return 'Nan',str(p\_val),'Nan','Nan','Nan'

else:

three\_CV = three\_sigma\*100/abs(u1)

return 'Nan',str(p\_val),'Nan','Nan',str(round(three\_CV,6))

else:

try:

m3 = np.sqrt(n\*(n-1))/(n-2)\*((1/n\*sum\_item\_l\_2)/np.sqrt(1/n\*sum\_item\_l\_1)\*\*3)

# print('m3=',m3)

# print('d8:',n+1)

# print('d15:',1/n\*sum\_item\_l\_3/(1/n\*sum\_item\_l\_1)\*\*2)

# print('d16:',(n+1)\*1/n\*sum\_item\_l\_3/(1/n\*sum\_item\_l\_1)\*\*2-3\*(n-1))

# print('d6/d7',(n-1)/((n-2)\*(n-3)))

m4 = ((n-1)/((n-2)\*(n-3)))\*((n+1)\*1/n\*sum\_item\_l\_3/(1/n\*sum\_item\_l\_1)\*\*2-3\*(n-1))#(d6/d7)\*d16

# print('m4=',m4)

#(d14\*\*2+1)/(d17+3\*(d10/d7))

bc =(m3\*\*2+1)/(m4+3\*((n-1)\*\*2/((n-2)\*(n-3))))

# print('bc:',bc)

a\_L=0.05

a\_M=0.1

a\_U=0.32

a\_Q = (a\_U-a\_L)\*bc\*\*2+a\_L

# print ('a\_Q:',a\_Q)

a\_irr = np.sqrt((a\_U-a\_L)\*\*2\*bc)+a\_L

# print('a\_irr:',a\_irr)

except Exception as e:

# print('calculate error',e)

if abs(u1) == 0:

return 'Nan',str(p\_val),'Nan','Nan','Nan'

else:

three\_CV = three\_sigma\*100/abs(u1)

# print('three\_CV:',three\_CV)

return 'Nan',str(p\_val),'Nan','Nan',str(round(three\_CV,6))

if abs(u1) == 0:

return str(round(bc,6)),str(p\_val),str(round(a\_Q,6)),str(round(a\_irr,6)),'Nan'

else:

three\_CV = three\_sigma\*100/abs(u1)

# print('three\_CV:',three\_CV)

return str(round(bc,6)),str(p\_val),str(round(a\_Q,6)),str(round(a\_irr,6)),str(round(three\_CV,6))

def cpk\_calc(df\_data,lsl,usl):

"""

:param df\_data: list

:param usl: 数据指标上限

:param lsl: 数据指标下限

:return:

"""

sigma = 3

# print('limit--->',lsl,usl)

# 数据平均值

# print('df\_data in cpk\_calc:',df\_data)

mean = np.mean(df\_data)#

# print('mean ---->',mean)

# 数据max值

max\_num = max(df\_data)

# print('max\_num ---->',max\_num)

# 数据min值

min\_num = min(df\_data)

# print('min\_num ---->',min\_num)

# a = np.array([[1, 2], [3, 4]])

# print('a--->', type(a))

# print('gobal std:',np.std(a))#全局标准差

# print('each line std:',np.std(a, axis=0,ddof=1))

# print("each row std:",np.std(a, axis=1,ddof=1))

# 数据标准差

if len(df\_data)==1:

stdev =0.00

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

else:

try:

stdev = np.std(df\_data,ddof=1,axis=0)

except Exception as e:

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

# print('stdev ---->',stdev)

if stdev == 0:#stop count cpk

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

# 生成横轴数据平均分布

# x1 = np.linspace(mean - sigma \* stdev, mean + sigma \* stdev, 1000)

# print('x1 ---->',x1)

# 计算正态分布曲线

# y1 = np.exp(-(x1 - mean) \*\* 2 / (2 \* stdev \*\* 2)) / (math.sqrt(2 \* math.pi) \* stdev)

# print('y1 ---->',y1)

x1,y1 = None,None

if (lsl != 'NA' and lsl != '') and (usl == 'NA' or usl == ''):

cpl = (mean - lsl) / (sigma \* stdev)

# print('====>>>>>>cpl',cpl)

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,None,None,None,cpl,None)

if (usl != 'NA' and usl != '') and (lsl == 'NA' or lsl == ''):

# print('====>>>>>>=====cpu')

cpu = (usl - mean) / (sigma \* stdev)

# print('====>>>>>>cpu',cpu)

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,None,None,cpu,None,None)

if lsl == 'NA' or usl == 'NA' or lsl == '' or usl == '':

return (mean,max\_num,min\_num,stdev,None,None,None,None,None)

cpu = (usl - mean) / (sigma \* stdev)

cpl = (mean - lsl) / (sigma \* stdev)

# 得出cpk

cpk = min(cpu, cpl)

mean = round(mean,3)

return (mean,max\_num,min\_num,stdev,x1,y1,cpu,cpl,cpk)

def get\_target\_value(lsl, usl):

if lsl != 'NA' and usl != 'NA' and lsl != '' and usl != '':

target\_value = round(((lsl + usl) / 2.0), 5)

else:

target\_value = 'Nan'

return target\_value

def checkItemName(item\_name,item\_length):

if len(item\_name) > item\_length:

if item\_name[item\_length] == '\_' or item\_name[item\_length] == ' ':

item\_name = item\_name[0:item\_length+1] + '\n' + item\_name[item\_length+1:]

else:

item\_name1 = item\_name[0:item\_length]

item\_name\_tmp = item\_name1[::-1]

x1=item\_name\_tmp.find('\_')

x2=item\_name\_tmp.find(' ')

x\_len = 0

if x1 == -1 and x2 == -1:

x\_len = 0

elif x1 == -1:

x\_len = x2

elif x2 ==-1:

x\_len = x1

elif x1>x2:

x\_len = x2

else:

x\_len = x1

x\_len = len(item\_name1) - x\_len

item\_name = item\_name[0:x\_len] + '\n' + item\_name[x\_len:]

return item\_name

def probability\_distribution\_extend(data,bins,margin,item\_name,lsl,usl,mean,max\_num,min\_num,

stdev,x1,y1,cpu,cpl,cpk,pic\_path,bins\_l,bins\_h,start\_time\_first,

start\_time\_last,bmc,zoom\_type='limit'):

try:

permin=round(np.percentile(data,2),3)

except Exception as e:

permin = ''

try:

permax=round(np.percentile(data,98),3)

except Exception as e:

permax = ''

max\_num\_orig = max\_num

min\_num\_orig = min\_num

mean\_orig = mean

bins = sorted(bins)

length = len(bins)

intervals = np.zeros(length+1)

for value in data:

i = 0

while i < length and value >= bins[i]:

i += 1

intervals[i] += 1

intervals = intervals / float(len(data))

plt.ion() # 开启interactive mode

plt.figure(1) # 创建图表1

plt.xlim(min(bins) - margin, max(bins) + margin)

bins.insert(0, -999)

# plt.title("probability-distribution",size=8,verticalalignment='bottom')

plt.bar(bins, intervals,color=['r'], label='')#频率分布

x\_ticks,labels = plt.xticks()

y\_ticks, labels = plt.yticks()

y\_ticks=round(y\_ticks[len(y\_ticks) - 1],5)

plt.close(1)

plt.figure(2,dpi=150) # 创建图表2

fig, axes = plt.subplots(1, 0, figsize=(6, 5), facecolor='#ccddef')

plt.axes([0.11, 0.17, 0.85, 0.7]) # [左, 下, 宽, 高] 规定的矩形区域 （全部是0~1之间的数，表示比例）

if stdev =='nan':

pass

else:

if stdev > 999999:

stdev = str("%.3e" % stdev)

else:

stdev = str("%.3f" % stdev)

if cpl ==None:

cpl\_value =''

else:

if cpl > 999999:

cpl\_value = str("%.3e" % cpl)

else:

cpl\_value = str("%.3f" % cpl)

if cpu ==None:

cpu\_value =''

else:

if cpu > 999999:

cpu\_value = str("%.3e" % cpu)

else:

cpu\_value = str("%.3f" % cpu)

if cpk ==None:

cpk\_value =''

else:

if cpk > 999999:

cpk\_value = str("%.3e" % cpk)

else:

cpk\_value = str("%.3f" % cpk)

if max\_num > 999999:

max\_num = str("%.6e" % max\_num)

else:

max\_num = str("%.3f" % max\_num)

if mean > 999999:

mean = str("%.6e" % mean)

else:

mean = str("%.3f" % mean)

if min\_num > 999999:

min\_num = str("%.6e" % min\_num)

else:

min\_num = str("%.3f" % min\_num)

if len(data) > 1000000000000:

sample\_n = str("%.e" % len(data))

else:

sample\_n = str("%.f" % len(data))

info = "Samples:" + sample\_n + ' ' +"Max:" + str(max\_num\_orig) + ' ' + "Min:" + str(min\_num\_orig) + '\n' + "Mean:" + str(mean\_orig) + ' '+ "Std:" + stdev + ' ' + "Cpl:" + cpl\_value + ' ' + "Cpu:" + cpu\_value + '\n' + "Cpk:" + cpk\_value + ' ' + '02%:'+ str(permin) + ' ' + '98%:' + str(permax) + ' '+ "Bimodal:" + bmc

item\_name = checkItemName(str(item\_name),55)

plt.title(item\_name,size=11,verticalalignment='bottom')

plt.ylabel('Count')

bins = [round(x,5) for x in bins]

bins=sorted(bins)

plt.hist(data, bins=bins, histtype='stepfilled',color = 'blue', edgecolor='blue', linewidth=1.0,align='mid',density=False) #time分布

range\_value = get\_limit\_range(bins\_l, bins\_h)

range\_value =round(range\_value/5,5)

if zoom\_type =='data':

range\_offset = abs(float(max\_num\_orig) - float(min\_num\_orig))\*0.2

plt.xlim(float(min\_num\_orig)-float(range\_offset), float(max\_num\_orig)+float(range\_offset))

else:

if (lsl =='' or lsl =='NA') and (usl!='NA'and usl!=''):

plt.xlim(float(min\_num\_orig)\*0.999, usl+range\_value)

elif (usl =='' or usl =='NA') and (lsl!='NA'and lsl!=''):

plt.xlim(lsl-range\_value, float(max\_num\_orig)\*1.001)

elif (usl =='' or usl =='NA') and (lsl=='NA'or lsl==''):

plt.xlim(float(min\_num\_orig)\*0.999, float(max\_num\_orig)\*1.001)

else:

plt.xlim(bins\_l-range\_value, bins\_h+range\_value)

y\_ticks = len(data) \* y\_ticks

plt.ylim((0, y\_ticks)) # 设置y轴scopex

ax=plt.gca()

ax.spines['bottom'].set\_linewidth(1)

ax.spines['left'].set\_linewidth(1)

ax.spines['right'].set\_linewidth(1)

ax.spines['top'].set\_linewidth(1)

if lsl !='' and lsl !='NA' and zoom\_type =='limit':

plt.plot([lsl, lsl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画lower limit线，

plt.text(lsl, y\_ticks / 3, ' LSL\n' + ' ' + str(lsl), fontdict={'size': 8, 'color': 'r'})

if usl != '' and usl != 'NA' and zoom\_type =='limit':

plt.plot([usl, usl, ], [0, y\_ticks, ], 'k--', linewidth=1.0, color='red') # 画upper limit线，

plt.text(usl, y\_ticks / 2, ' USL\n' + ' ' + str(usl), fontdict={'size': 8, 'color': 'r'})

plt.text(0.0,-0.236, info,fontsize=10,ha="left",transform=ax.transAxes)

plt.legend(bbox\_to\_anchor=(0.9,-0.09),loc="best",fontsize=10,framealpha=0,edgecolor='royalblue',borderaxespad=0.1)

plt.grid(linestyle=':',c='gray') # 生成网格

plt.savefig(pic\_path,dpi=200)

plt.draw()

plt.close('all')

plt.ioff()

def get\_bins(min\_num, max\_num, lsl, usl, set\_bins, zoom\_type):

bins\_l = 0

bins\_h = 0

if lsl == 'NA' or usl == 'NA' or zoom\_type == 'data':

bins\_l, bins\_h = min\_num, max\_num

else:

if min\_num < lsl and max\_num < lsl:

bins\_l = min\_num

bins\_h = usl

elif min\_num < lsl and max\_num > lsl and max\_num <= usl:

bins\_l = min\_num

bins\_h = usl

elif min\_num < lsl and max\_num > usl:

bins\_l = min\_num

bins\_h = max\_num

elif lsl <= min\_num and min\_num <= usl and max\_num <= usl:

bins\_l = lsl

bins\_h = usl

elif lsl <= min\_num and min\_num <= usl and max\_num > usl:

bins\_l = lsl

bins\_h = max\_num

elif min\_num > usl:

bins\_l = lsl

bins\_h = max\_num

range\_value = get\_limit\_range(bins\_l, bins\_h)

if lsl == 'NA' or usl == 'NA' or zoom\_type == 'data':

if range\_value == 0 and min\_num > 0:

range\_value = min\_num \* 0.2

bins\_l = (bins\_l - min\_num \* 0.1)

bins\_h = (bins\_h + min\_num \* 0.1)

elif range\_value == 0 and min\_num == 0:

range\_value = 6

bins\_l = - 3

bins\_h = 3

elif range\_value == 0 and min\_num < 0:

range\_value = 6

bins\_l = min\_num - 3

bins\_h = min\_num + 3

else:

if min\_num > 1 and range\_value < 1 and range\_value != 0:

range\_value = min\_num \* 0.05

bins\_l = (bins\_l - min\_num \* 0.025)

bins\_h = (bins\_h + min\_num \* 0.025)

elif min\_num > 0.001 and min\_num < 1 and range\_value < 1 and range\_value != 0:

range\_value = min\_num \* 0.2

bins\_l = (bins\_l - min\_num \* 0.1)

bins\_h = (bins\_h + min\_num \* 0.1)

else:

range\_value = range\_value \* 0.4

bins\_l = (bins\_l - range\_value \* 0.2)

bins\_h = (bins\_h + range\_value \* 0.2)

else:

if range\_value == 0 and lsl != 0:

range\_value = lsl \* 0.2

bins\_l = bins\_l - lsl \* 0.1

bins\_h = bins\_h + usl \* 0.1

elif range\_value == 0 and lsl == 0:

range\_value = 6

bins\_l = - 3

bins\_h = 3

range\_value = round((range\_value / set\_bins), 12)

# print('range\_value2-->',range\_value)

# print('=====>',bins\_l,bins\_h,range\_value)

# bins = np.arange(bins\_l, bins\_h, range\_value) # 必须是单调递增的

bins = np.arange(bins\_l, bins\_h, range\_value)#必须是单调递增的

if bins\_l<0:

bins = np.arange(bins\_h, bins\_l, -range\_value)

# print('lsl,usl,min\_num,max\_num,bins\_l,bins\_h in get\_bins=====>',lsl,usl,min\_num,max\_num,bins\_l,bins\_h)

return bins, bins\_l, bins\_h

def get\_limit\_range(lsl, usl):

# print('lsl,usl----->', lsl, usl)

range\_value = 0

if lsl < 0 and usl <= 0:

range\_value = abs(lsl) - abs(usl)

elif lsl < 0 and usl >= 0:

range\_value = abs(lsl) + usl

elif lsl >= 0 and (usl > 0):

range\_value = usl - lsl

else:

print('get\_limit\_range 00000')

range\_value = round(range\_value, 5)

return range\_value

def probability\_distribution\_extend\_by\_color(column\_category\_data\_list, data, bins, margin, item\_name, lsl, usl, mean,

max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path, bins\_l, bins\_h,

start\_time\_first, start\_time\_last, bmc, zoom\_type):

# print('one column data len:',len(data),item\_name,data,column\_category\_data\_list)

try:

permin=round(np.percentile(data,2),3)

except Exception as e:

permin = ''

try:

permax=round(np.percentile(data,98),3)

except Exception as e:

permax = ''

max\_num\_orig = max\_num

min\_num\_orig = min\_num

mean\_orig = mean

bins = sorted(bins)

length = len(bins)

intervals = np.zeros(length + 1)

for value in data:

i = 0

while i < length and value >= bins[i]:

i += 1

intervals[i] += 1

intervals = intervals / float(len(data))

plt.ion() # 开启interactive mode

plt.figure(1) # 创建图表1

plt.xlim(min(bins) - margin, max(bins) + margin)

bins.insert(0, -999)

plt.title("probability-distribution")

plt.bar(bins, intervals, color=['r'], label='') # 频率分布

x\_ticks, labels = plt.xticks()

x\_ticks\_start = round(x\_ticks[0], 2)

x\_ticks\_end = round(x\_ticks[len(x\_ticks) - 1], 2)

y\_ticks, labels = plt.yticks()

# print('y\_ticks--->',y\_ticks,y\_ticks[len(y\_ticks) - 1],y\_ticks[0])

y\_ticks = round(y\_ticks[len(y\_ticks) - 1], 5)

# print("x\_ticks\_start,x\_ticks\_end--->",x\_ticks\_start,x\_ticks\_end)

# print('y\_ticks\_end--->',y\_ticks)

# plt.show()

plt.close(1)

plt.figure(2, dpi=150) # 创建图表2,facecolor='blue',edgecolor='black'

if stdev == 'nan':

pass

else:

if stdev > 999999:

stdev = str("%.3e" % stdev)

else:

stdev = str("%.3f" % stdev)

if cpl == None:

cpl\_value = ''

else:

if cpl > 999999:

cpl\_value = str("%.3e" % cpl)

else:

cpl\_value = str("%.3f" % cpl)

if cpu == None:

cpu\_value = ''

else:

if cpu > 999999:

cpu\_value = str("%.3e" % cpu)

else:

cpu\_value = str("%.3f" % cpu)

if cpk == None:

cpk\_value = ''

else:

if cpk > 999999:

cpk\_value = str("%.3e" % cpk)

else:

cpk\_value = str("%.3f" % cpk)

if max\_num > 999999:

max\_num = str("%.3e" % max\_num)

else:

max\_num = str("%.3f" % max\_num)

if mean > 999999:

mean = str("%.3e" % mean)

else:

mean = str("%.3f" % mean)

if min\_num > 999999:

min\_num = str("%.3e" % min\_num)

else:

min\_num = str("%.3f" % min\_num)

if len(data) > 1000000000000:

sample\_n = str("%.e" % len(data))

else:

sample\_n = str("%.f" % len(data))

info = "Samples:" + sample\_n + ' ' +"Max:" + str(max\_num\_orig) + ' ' + "Min:" + str(min\_num\_orig) + '\n' + "Mean:" + str(mean\_orig) + ' '+ "Std:" + stdev + ' ' + "Cpl:" + cpl\_value + ' ' + "Cpu:" + cpu\_value + '\n' + "Cpk:" + cpk\_value + ' ' + '02%:'+ str(permin) + ' ' + '98%:' + str(permax) + ' '+ "Bimodal:" + bmc

item\_name = checkItemName(str(item\_name),55)

# if len(item\_name) > 55:

# item\_name = item\_name[0:55] + '\n' + item\_name[55:]

# font = FontProperties(fname=r"/Library/Fonts/Songti.ttc", size=12)

plt.title(item\_name, size=10)

plt.xlabel(str(start\_time\_first) + ' -- ' + str(start\_time\_last))

plt.ylabel('Count')

# plt.title(item\_name+"\nCpk={0}".format(str("%.6f" % cpk)))

# plt.hist(x=data, bins=bins, density=False, histtype='bar', color=['r'])

bins = [round(x, 5) for x in bins]

bins = sorted(bins)

l = 0

l\_len = []

for category\_data in column\_category\_data\_list:

# print('category\_test\_data--->', category\_data[5:])

# print('category\_name,category\_color--->', category\_data[0],str(category\_data[1]))

category\_name, category\_color = category\_data[0], str(category\_data[1])

n = len(category\_data[5:])

l = l + n

# print('category len:',n)

l\_len.append(n)

if len(column\_category\_data\_list) == 1:

plt.hist(category\_data[5:], bins=bins, label=category\_data[0], color=category\_color, histtype='stepfilled',

edgecolor=category\_color, linewidth=1.5, align='mid', density=False) # time分布

else:

plt.hist(category\_data[5:], bins=bins, label=category\_data[0], color='white', histtype='step',

edgecolor=category\_color, linewidth=1.5, align='mid', density=False) # time分布

# print('----->a column len,max in one category:-->',l,max(l\_len))

y\_ticks = max(l\_len) \* (y\_ticks + 0.04)

range\_value = get\_limit\_range(bins\_l, bins\_h)

range\_value = round(range\_value / 5, 5)

# print('plot bar--->',bins\_l,range)

plt.xlim(bins\_l - range\_value, bins\_h + range\_value)

plt.ylim((0, y\_ticks)) # 设置y轴scopex

ax = plt.gca()

ax.spines['bottom'].set\_linewidth(1.5)

ax.spines['left'].set\_linewidth(1.5)

ax.spines['right'].set\_linewidth(1.5)

ax.spines['top'].set\_linewidth(1.5)

# plt.plot(x1, y1, 'k--', label="", linewidth=1.0, color='lime') # 画正态分布曲线

if lsl != 'NA' and usl != 'NA' and zoom\_type == 'limit':

plt.plot([lsl, lsl, ], [0, y\_ticks, ], 'k--', linewidth=3.0, color='red') # 画lower limit线，

plt.plot([usl, usl, ], [0, y\_ticks, ], 'k--', linewidth=3.0, color='red') # 画upper limit线，

# 添加文字

# plt.text(0.1,20, r'$\mu=100,\ \sigma=15$')

plt.text(lsl, y\_ticks / 3, ' LSL\n' + ' ' + str(lsl), fontdict={'size': 10, 'color': 'r'})

plt.text(usl, y\_ticks / 2, ' USL\n' + ' ' + str(usl), fontdict={'size': 10, 'color': 'r'})

# plt.axis([-0.2, 0.2,0, 100])#x 轴，y 轴

plt.text(bins\_l + range\_value / 3, y\_ticks \* 0.78, info, size=10, rotation=0.0, alpha=0.85, fontsize=8, ha="left",

va="center", bbox=dict(boxstyle="round", ec=('royalblue'), linestyle='-.', lw=1, fc=('white'), ))

# os.system('mkdir fail')

if len(column\_category\_data\_list) < 30:

plt.legend(loc="upper right", framealpha=1, edgecolor='royalblue', borderaxespad=0.3,

fontsize=6) # facecolor ='None',

# plt.legend(bbox\_to\_anchor=(1.01, 1), loc=2, borderaxespad=0)

plt.grid(linestyle=':', c='gray') # 生成网格

# path="/Users/rex/PycharmProjects/my/fail/"

plt.savefig(pic\_path, dpi=200)

plt.draw()

# plt.show()

plt.close('all')

plt.ioff()

def verify\_limit(lsl, usl):

if lsl != None:

lsl.replace(' ', '')

if usl != None:

usl.replace(' ', '')

try:

lsl = float(eval(lsl))

except:

lsl = None

try:

usl = float(eval(usl))

except:

usl = None

return lsl, usl

def draw\_histogram(column\_data, item\_name, lsl, usl, mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,

set\_bins, start\_time\_first, start\_time\_last, bmc, zoom\_type):

bins, bins\_l, bins\_h = get\_bins(min\_num, max\_num, lsl, usl, set\_bins, zoom\_type)

# print(len(column\_data),'-==--->',min(column\_data),max(column\_data),bins)

probability\_distribution\_extend(column\_data, bins, 0, item\_name, lsl, usl, mean, max\_num, min\_num, stdev, x1, y1,

cpu, cpl, cpk, pic\_path, bins\_l, bins\_h, start\_time\_first, start\_time\_last,bmc,

zoom\_type)

return True

def draw\_more\_histogram(column\_category\_data\_list, column\_data, item\_name, lsl, usl, mean, max\_num, min\_num, stdev, x1,

y1, cpu, cpl, cpk, pic\_path, set\_bins, start\_time\_first, start\_time\_last, bmc,zoom\_type):

"""

"""

# print('9999 column\_category\_data\_list--->',column\_category\_data\_list)

# print('9999 column\_data--->',len(column\_data),column\_data)

# range = get\_limit\_range(lsl, usl)

# range = round((range /set\_bins), 5)

# bins = np.arange(lsl, usl, range) # 必须是单调递增的

bins, bins\_l, bins\_h = get\_bins(min\_num, max\_num, lsl, usl, set\_bins, zoom\_type)

probability\_distribution\_extend\_by\_color(column\_category\_data\_list, column\_data, bins, 0, item\_name, lsl, usl, mean,

max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path, bins\_l, bins\_h,

start\_time\_first, start\_time\_last, bmc,zoom\_type)

return True

def get\_project\_info(data\_df):

project\_code = get\_project\_code(data\_df)

# print('--------------------project\_code:'+project\_code+'------------------------')

build\_stage = get\_build\_stage(data\_df)

# print('--------------------build\_stage:'+build\_stage+'-------------------------')

station\_name = get\_station\_name(data\_df)

# print('--------------------station\_name:'+station\_name+'------------------------')

return project\_code, build\_stage, station\_name

def get\_build\_stage(data\_df):

special\_build\_name = data\_df['Special Build Name'].values.tolist() # Special Build Name

# pattern =re.compile(r'.\*\-(.+)\-')

# result=pattern.match(build\_stage)

# build\_stage = result.group(1)

build\_stage = ''

build\_stage\_l = list(set(special\_build\_name))

temp\_l = []

for i in build\_stage\_l:

pattern = re.compile(r'.\*\-(.+)')

result = pattern.match(i)

if result:

temp\_l.append(result.group(1))

else:

temp\_l.append(i)

n = 1

for j in list(set(temp\_l)):

if n == 1:

build\_stage = j

else:

build\_stage = build\_stage + '&' + j

n = n + 1

return build\_stage

def get\_station\_name(data\_df):

# station\_name = data\_df[1:2].values[0].tolist()[6] #Station ID first cell

station\_name\_l = data\_df['Station ID'].values.tolist() # Station ID first cell

station\_name\_l = list(set(station\_name\_l))

station\_name = ''

if len(station\_name\_l) == 0:

print('data is empty!')

station\_name == 'xxx'

else:

for i in station\_name\_l:

if i != '':

pattern = re.compile(r'.\*\\_([0-9]+)\\_(\D+)')

result = pattern.match(i)

try:

station\_name = result.group(2)

return station\_name

except Exception as e:

print('match station\_name error')

return station\_name

def get\_project\_code(data\_df):

product\_l = list(set(data\_df['Product'].values.tolist())) # Product first cell

# print("=============>>>>>>product\_l", product\_l)

project\_code = ''

n = 1

for i in product\_l:

if n == 1:

project\_code = i

else:

project\_code = project\_code + '&' + i

n = n + 1

return project\_code

def open\_all\_csv(event, all\_csv\_path, data\_select, remove\_fail):

tmp\_lst = []

print('>-all\_csv\_path:',all\_csv\_path)

with open(all\_csv\_path, 'r') as f:

reader = csv.reader(f)

i = 1

for row in reader:

if row[0].lower().find('display name') != -1:

pass

elif row[0].lower().find('pdca priority') != -1:

pass

elif row[0].lower().find('upper limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('lower limit') != -1:

tmp\_lst.append(row)

elif row[0].lower().find('measurement unit') != -1: # "Measurement Unit ----->" in row:

pass

elif row[0].lower().find('site') != -1:

tmp\_lst.append(row)

else:

tmp\_lst.append(row)

i = i + 1

# print("index---->", tmp\_lst[0])

param\_item\_start\_index = tmp\_lst[0].index('Parametric')

print('>location parametric:',param\_item\_start\_index)

header\_list = tmp\_lst[1]

df = pd.DataFrame(tmp\_lst[2:], columns=tmp\_lst[1])

header\_df = df[0:2]

# print('header\_df before--->', header\_df)

data\_df = df[2:]

# print('data\_df before--->', data\_df)

print('csv data row number before remove SN empty--->', len(data\_df.values.tolist()))

data\_df = data\_df[~data\_df['SerialNumber'].isin([''])] # Remove SN Empty

print('csv data row number after remove SN empty--->', len(data\_df.values.tolist()))

print('csv data row number before remove fail--->', len(data\_df.values.tolist()))

if remove\_fail == 'yes':

data\_df = data\_df[data\_df['Test Pass/Fail Status'].isin(['PASS'])]

# data\_df=data\_df[~data\_df['Test Pass/Fail Status'].isin(['FAIL'])]

print('csv data row number after remove fail--->', len(data\_df.values.tolist()))

print('csv data row number before remove retest--->', len(data\_df.values.tolist()))

if data\_select == 'first':

data\_df = data\_df.sort\_values(axis=0, by=['StartTime'], ascending='True')

data\_df.drop\_duplicates(['SerialNumber'], keep='first', inplace=True)

elif data\_select == 'last':

data\_df = data\_df.sort\_values(axis=0, by=['StartTime'], ascending='True')

data\_df.drop\_duplicates(['SerialNumber'], keep='last', inplace=True)

elif data\_select == 'no\_retest':

data\_df.drop\_duplicates(['SerialNumber'], keep=False, inplace=True)

elif data\_select == 'all':

pass

print('csv data row number after remove retest--->', len(data\_df.values.tolist()))

if event == 'keynote-report' or event == 'excel-report':

project\_code, build\_stage, station\_name = get\_project\_info(data\_df)

else:

project\_code, build\_stage, station\_name = '', '', ''

start\_time\_l = data\_df['StartTime'].values.tolist() #StartTime

if len(start\_time\_l)>0:

start\_time\_first = min(start\_time\_l)

start\_time\_last = max(start\_time\_l)

else:

start\_time\_first = ''

start\_time\_last = ''

print('<first time -- last time>',start\_time\_first,start\_time\_last)

df = header\_df.append(data\_df)

return header\_list, df, project\_code, build\_stage, station\_name, start\_time\_first, start\_time\_last, param\_item\_start\_index

# def get\_file\_pic\_name(file\_dir):

# pic\_file\_l = []

# for root,dirs,files in os.walk(file\_dir):

# for file in files:

# if os.path.splitext(file)[1] == '.png':

# # print('suffix name---->',os.path.splitext(file)[1])

# pic\_file\_l.insert(0,os.path.splitext(file)[0])

# return pic\_file\_l

def get\_file\_pic\_name(file\_dir):

pic\_file\_l = []

files = os.listdir(file\_dir)

files.sort(key=lambda x: str(x.split('.')[0]))

for file in files:

if os.path.splitext(file)[1] == '.png':

pic\_file\_l.append(os.path.splitext(file)[0])

return pic\_file\_l

def get\_pst\_time():

date\_format= '%Y-%m-%d\_%H-%M-%S' #'%m-%d%Y\_%H\_%M\_%S\_%Z'

date = datetime.datetime.now(tz=pytz.utc)

date = date.astimezone(timezone('US/Pacific'))

pstDateTime=date.strftime(date\_format)

return pstDateTime

def generate\_keynote\_report(project\_code,station\_name,build\_stage,dir\_path,project\_name,target\_build):

# keynote\_title\_name = project\_code+'-'+build\_stage+'-'+station\_name+' Data Analysis'

keynote\_title\_name = str(project\_name)+'-'+str(target\_build)+'-'+station\_name+'\nData Analysis'

description\_info = 'Issue description:\n'

root\_cause\_info = 'Root Cause:\n'

plan\_info = 'Next steps:'

# print('dir\_path--->',dir\_path)

# keynote\_save\_path = dir\_path+project\_code+'\_'+build\_stage+'\_Data\_Review\_'+datetime.datetime.now().strftime('%Y-%m-%d\_%H-%M-%S')+'.key'

if not os.path.exists(dir\_path):

os.makedirs(dir\_path)

keynote\_save\_path = dir\_path+str(project\_name)+'\_'+str(target\_build)+'\_Data\_Review\_'+str(get\_pst\_time())+'.key'

with open('/tmp/CPK\_Log/temp/.keynotereportname.txt', 'w') as file\_object:

file\_object.write(keynote\_save\_path)

doc,keynote = generate\_keynote.create\_keynote(keynote\_title\_name)

print('save title page to keynote report finished!')

generate\_keynote.add\_build\_summary\_overview(doc,keynote,'/tmp/CPK\_Log/temp/yield\_rate\_param.csv','/tmp/CPK\_Log/retest/retest\_pie.png')

print('add\_build\_summary\_overview finished')

generate\_keynote.add\_build\_yield\_pie(doc,keynote,'/tmp/CPK\_Log/retest/retest\_pie.png')

print('add\_build\_yield\_pie finished')

generate\_keynote.build\_failures\_retest\_pareto(doc,keynote,'/tmp/CPK\_Log/retest/retest\_pareto.png','/tmp/CPK\_Log/retest/fail\_pareto.png')

print('build\_failures\_retest\_pareto finished')

generate\_keynote.build\_summary\_all\_retest\_rates(doc,keynote,'/tmp/CPK\_Log/retest/daily\_all\_retest\_summary.png')

print('build\_summary\_all\_retest\_rates finished')

generate\_keynote.build\_summary\_retest\_rates(doc,keynote,'/tmp/CPK\_Log/retest/daily\_retest\_summary.png','/tmp/CPK\_Log/retest/retest\_vs\_station\_id.png','/tmp/CPK\_Log/retest/retest\_vs\_version.png')

print('build\_summary\_retest\_rates finished')

generate\_keynote.top\_5\_fail\_and\_retest(doc,keynote,'/tmp/CPK\_Log/retest/fail\_item\_overall.csv','/tmp/CPK\_Log/retest/retest\_item\_overall.csv')

print('top\_5\_fail\_and\_retest finished')

generate\_keynote.fixture\_retest\_breakdown(doc,keynote,'/tmp/CPK\_Log/retest/retest\_breakdown\_fixture.csv')

print('fixture\_retest\_breakdown finished')

generate\_keynote.max\_min\_cpk\_technology(doc,keynote,'/tmp/CPK\_Log/retest/cpk\_min\_max.csv')

print('max\_min\_cpk\_technology finished')

generate\_keynote.create\_abnormal\_distribution(doc,keynote,'Abnormal Distribution\nAnalysis')

fail\_pic\_path = '/tmp/CPK\_Log/fail\_plot'

file\_l=[]

file\_l = get\_file\_pic\_name(fail\_pic\_path)

for f\_name in file\_l:

one\_fail\_pic\_path = fail\_pic\_path +'/'+f\_name+'.png'

generate\_keynote.add\_fail\_pic(doc,keynote,one\_fail\_pic\_path,f\_name,description\_info,root\_cause\_info,plan\_info)

generate\_keynote.save\_keynote(doc,keynote,keynote\_save\_path)

print('save fail\_plot to keynote report finished!')

def create\_keynote\_report\_all(event,header\_list,df,color\_by1,pic\_path,select\_category1,cpk\_lsl,cpk\_usl,

save\_all\_cpk\_path,set\_bins,excel\_name,project\_code,build\_stage,station\_name,

start\_time\_first,start\_time\_last,color\_by2,select\_category\_l2,excel\_report\_item,

fail\_plot\_to\_excel,zoom\_type,param\_item\_start\_index,project\_name,target\_build):

clear\_files('/tmp/CPK\_Log/fail\_plot/')

table\_data,table\_category\_data,no\_valid\_column\_name\_l = parse\_all\_csv(header\_list,df,color\_by1,select\_category1,event,color\_by2,select\_category\_l2,param\_item\_start\_index)#

i,j,n,t=0,0,0,0

path=save\_all\_cpk\_path

result='pass'

picFail\_path = '/tmp/CPK\_Log/fail\_plot/'

for column\_data in table\_data:

# print('column\_data length,column\_data--->',t,len(column\_data),column\_data)

item\_name=column\_data[0]

if str(item\_name).lower() != 'fixture channel id\_' and str(item\_name).lower() != 'head id':

usl = column\_data[1]

lsl = column\_data[2]

column\_data = column\_data[3:]

if len(column\_data) >0:

bc,p\_val,a\_Q,a\_irr,three\_CV = get\_coefficients(column\_data)

BMC = ''

if bc != '' and bc != 'Nan' and p\_val != '' and a\_Q != '' and p\_val != 'Nan' and a\_Q != 'Nan':

if float(p\_val) <= float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) <= float(a\_Q) and float(bc)<0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))>=-0.1:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))<-0.1:

BMC = 'NO'

else:

BMC = ''

else:

BMC = ''

row\_data = []

target\_value = 9999999999

mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk = cpk\_calc(column\_data, lsl, usl)

if cpk:

if is\_number(str(cpk)):

if float(cpk)>float(cpk\_usl):

BMC = ''

result=''

if not os.path.exists(picFail\_path):

os.makedirs(picFail\_path)

image\_name = item\_name.replace('/','\_')+".png"

pic\_path = picFail\_path + image\_name

if len(table\_category\_data) == 0:

draw\_histogram(column\_data,item\_name,lsl, usl, mean, max\_num, min\_num,

stdev, x1, y1, cpu, cpl, cpk, pic\_path,set\_bins,start\_time\_first,

start\_time\_last,BMC,zoom\_type)

else:

draw\_more\_histogram(table\_category\_data[t],column\_data,item\_name,lsl, usl, mean,

max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,

set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

j=j+1

else:

pass

t = t + 1

print('All items cpk calulate/draw plots/excel report finished!')

generate\_keynote\_report(project\_code,station\_name,build\_stage,save\_all\_cpk\_path,project\_name,target\_build)

def create\_keynote\_report\_by\_cpk(event,header\_list,df,color\_by1,pic\_path,select\_category1,cpk\_lsl,cpk\_usl,

save\_all\_cpk\_path,set\_bins,excel\_name,project\_code,build\_stage,station\_name,

start\_time\_first,start\_time\_last,color\_by2,select\_category\_l2,excel\_report\_item,

fail\_plot\_to\_excel,zoom\_type,param\_item\_start\_index,check\_cpk\_thhld,k\_item\_list,

project\_name,target\_build):

clear\_files('/tmp/CPK\_Log/fail\_plot/')

table\_data,table\_category\_data,no\_valid\_column\_name\_l = parse\_all\_csv(header\_list,df,color\_by1,select\_category1,event,color\_by2,select\_category\_l2,param\_item\_start\_index)#

i,j,n,t=0,0,0,0

path=save\_all\_cpk\_path

picFail\_path = '/tmp/CPK\_Log/fail\_plot/'

result='pass'

for column\_data in table\_data:

item\_name=column\_data[0]

if str(item\_name).lower() != 'fixture channel id\_' and str(item\_name).lower() != 'head id':

usl = column\_data[1]

lsl = column\_data[2]

column\_data = column\_data[3:]

if len(column\_data) >0:

bc,p\_val,a\_Q,a\_irr,three\_CV = get\_coefficients(column\_data)

BMC = ''

if bc != '' and bc != 'Nan' and p\_val != '' and a\_Q != '' and p\_val != 'Nan' and a\_Q != 'Nan':

if float(p\_val) <= float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) <= float(a\_Q) and float(bc)<0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))>=-0.1:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))<-0.1:

BMC = 'NO'

else:

BMC = ''

else:

BMC = ''

row\_data = []

target\_value = 9999999999

mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk = cpk\_calc(column\_data, lsl, usl)

if cpk:

if is\_number(str(cpk)):

if float(cpk)>float(cpk\_usl):

BMC = ''

if cpk !=None:

ui\_select\_item = ''

if item\_name in k\_item\_list:

ui\_select\_item = 'yes'

if cpk < cpk\_lsl or ui\_select\_item == 'yes':

result='FAIL'

if not os.path.exists(picFail\_path):

os.makedirs(picFail\_path)

image\_name = item\_name.replace('/','\_')+".png"

pic\_path = picFail\_path + image\_name

if len(table\_category\_data) == 0:

draw\_histogram(column\_data,item\_name,lsl, usl, mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

else:

draw\_more\_histogram(table\_category\_data[t],column\_data,item\_name,lsl, usl, mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

j=j+1

elif cpk > cpk\_lsl and cpk < cpk\_usl and BMC == "YES" and check\_cpk\_thhld == 'yes':

result='FAIL'

if not os.path.exists(picFail\_path):

os.makedirs(picFail\_path)

image\_name = item\_name.replace('/','\_')+".png"

pic\_path = picFail\_path + image\_name

if len(table\_category\_data) == 0:

draw\_histogram(column\_data,item\_name,lsl, usl, mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

else:

draw\_more\_histogram(table\_category\_data[t],column\_data,item\_name,lsl, usl, mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

j=j+1

elif cpk > cpk\_lsl and BMC=="YES" and check\_cpk\_thhld == 'no':

result='FAIL'

if not os.path.exists(picFail\_path):

os.makedirs(picFail\_path)

image\_name = item\_name.replace('/','\_')+".png"

pic\_path = picFail\_path + image\_name

if len(table\_category\_data) == 0:

draw\_histogram(column\_data,item\_name,lsl, usl, mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

else:

draw\_more\_histogram(table\_category\_data[t],column\_data,item\_name,lsl, usl, mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

j=j+1

else:

pass

t = t + 1

print('All items cpk calulate/draw plots/excel report finished!')

generate\_keynote\_report(project\_code,station\_name,build\_stage,save\_all\_cpk\_path,project\_name,target\_build)

def create\_keynote\_report\_by\_p\_val(event,header\_list,df,color\_by1,pic\_path,select\_category1,cpk\_lsl,cpk\_usl,

save\_all\_cpk\_path,set\_bins,excel\_name,project\_code,build\_stage,station\_name,

start\_time\_first,start\_time\_last,color\_by2,select\_category\_l2,excel\_report\_item,

fail\_plot\_to\_excel,zoom\_type,param\_item\_start\_index,check\_cpk\_thhld,k\_item\_list,

project\_name,target\_build):

clear\_files('/tmp/CPK\_Log/fail\_plot/')

table\_data,table\_category\_data,no\_valid\_column\_name\_l = parse\_all\_csv(header\_list,df,color\_by1,select\_category1,

event,color\_by2,select\_category\_l2,

param\_item\_start\_index)

i,j,n,t=0,0,0,0

picFail\_path = '/tmp/CPK\_Log/fail\_plot/'

path=save\_all\_cpk\_path

result='pass'

for column\_data in table\_data:

item\_name=column\_data[0]

if str(item\_name).lower() != 'fixture channel id\_' and str(item\_name).lower() != 'head id':

usl = column\_data[1]

lsl = column\_data[2]

column\_data = column\_data[3:]

if len(column\_data) >0:

bc,p\_val,a\_Q,a\_irr,three\_CV = get\_coefficients(column\_data)

BMC = ''

if bc != '' and bc != 'Nan' and p\_val != '' and a\_Q != '' and p\_val != 'Nan' and a\_Q != 'Nan':

if float(p\_val) <= float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) <= float(a\_Q) and float(bc)<0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)>0.555:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))>=-0.1:

BMC = 'YES'

elif float(p\_val) > float(a\_Q) and float(bc)<0.555 and (float(bc)-float(p\_val))<-0.1:

BMC = 'NO'

else:

BMC = ''

else:

BMC = ''

row\_data = []

target\_value = 9999999999

mean, max\_num, min\_num, stdev, x1, y1, cpu, cpl, cpk = cpk\_calc(column\_data, lsl, usl)

if cpk:

if is\_number(str(cpk)):

if float(cpk)>float(cpk\_usl):

BMC = ''

if check\_cpk\_thhld == 'yes':

if cpk !=None:

ui\_select\_item = ''

if item\_name in k\_item\_list:

ui\_select\_item = 'yes'

if cpk < cpk\_usl and BMC =="YES" or ui\_select\_item == 'yes':

result='FAIL'

if not os.path.exists(picFail\_path):

os.makedirs(picFail\_path)

image\_name = item\_name.replace('/','\_')+".png"

pic\_path = picFail\_path + image\_name

if len(table\_category\_data) == 0:

draw\_histogram(column\_data,item\_name,lsl, usl, mean, max\_num,

min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,

set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

else:

draw\_more\_histogram(table\_category\_data[t],column\_data,item\_name,lsl,

usl, mean, max\_num, min\_num, stdev, x1, y1, cpu,

cpl, cpk, pic\_path,set\_bins,start\_time\_first,start\_time\_last,

BMC,zoom\_type)

j=j+1

elif check\_cpk\_thhld == 'no':

ui\_select\_item = ''

if item\_name in k\_item\_list:

ui\_select\_item = 'yes'

if BMC == 'YES' or ui\_select\_item == 'yes':

result='FAIL'

if not os.path.exists(picFail\_path):

os.makedirs(picFail\_path)

image\_name = item\_name.replace('/','\_')+".png"

pic\_path = picFail\_path + image\_name

if len(table\_category\_data) == 0:

draw\_histogram(column\_data,item\_name,lsl, usl, mean, max\_num,

min\_num, stdev, x1, y1, cpu, cpl, cpk, pic\_path,

set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

else:

draw\_more\_histogram(table\_category\_data[t],column\_data,

item\_name,lsl, usl, mean, max\_num,

min\_num, stdev, x1, y1, cpu, cpl, cpk,

pic\_path,set\_bins,start\_time\_first,start\_time\_last,BMC,zoom\_type)

j=j+1

else:

pass

t = t + 1

print('All items cpk calulate/draw plots/excel report finished!')

generate\_keynote\_report(project\_code,station\_name,build\_stage,save\_all\_cpk\_path,project\_name,target\_build)

def generate\_report\_for\_keynote\_1a\_yes(table\_data):

cpk\_lsl = table\_data[0] # cpk lsl

cpk\_usl = table\_data[1] #float("inf") #table\_data[1] # cpk usl

cpk\_path = table\_data[2] #'[NSString stringWithFormat:@"%@/CPK\_Log/",desktopPath];

filelogname = '/tmp/CPK\_Log/temp/.keynote.txt' #

set\_bins = table\_data[3] #250

all\_csv\_path = table\_data[4] #'csv 读取的数据路径，从此路径获得数据

project\_name = table\_data[5]

target\_build = table\_data[6]

event = 'keynote-report'

color\_by1 = 'Off'

select\_category\_l1 =[]

color\_by2 = 'Off'

select\_category\_l2 = []

remove\_fail = 'yes'

data\_select = 'all'

fail\_pic\_path ='/tmp/CPK\_Log/fail\_plot/'

excel\_report\_item = 'all' #fail all

fail\_plot\_to\_excel = 'no' # yes no

zoom\_type = 'limit'

header\_list,df,project\_code,build\_stage,station\_name,start\_time\_first,start\_time\_last,param\_item\_start\_index = open\_all\_csv(event,all\_csv\_path,data\_select,remove\_fail)

excel\_report\_file\_name = ''

create\_keynote\_report\_all(event,header\_list,df,color\_by1,fail\_pic\_path,select\_category\_l1,cpk\_lsl,cpk\_usl,cpk\_path,

set\_bins,excel\_report\_file\_name,project\_code,build\_stage,station\_name,start\_time\_first,

start\_time\_last,color\_by2,select\_category\_l2,excel\_report\_item,fail\_plot\_to\_excel,zoom\_type,

param\_item\_start\_index,project\_name,target\_build)

print('create keynote report finished!')

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,create keynote report finished")

def generate\_report\_for\_keynote\_1b\_yes(table\_data):

cpk\_lsl = table\_data[0] # cpk lsl

cpk\_usl = table\_data[1] #float("inf") #table\_data[1] # cpk usl

cpk\_path = table\_data[2] # [NSString stringWithFormat:@"%@/CPK\_Log/",desktopPath];

filelogname = '/tmp/CPK\_Log/temp/.keynote.txt'

set\_bins = table\_data[3] #250

all\_csv\_path = table\_data[4] #'csv 读取的数据路径，从此路径获得数据

csv\_select\_k\_path = table\_data[5] #'界面上UI K 列选择的item csv

check\_cpk\_thhld = table\_data[6] # yes/no

check\_one\_limit = table\_data[7] # yes/no

project\_name = table\_data[8]

target\_build = table\_data[9]

csv\_select\_k\_path = os.path.join(csv\_select\_k\_path+ '')

k\_item\_list = read\_csv\_to\_list(csv\_select\_k\_path)

event = 'keynote-report'

color\_by1 = 'Off'

select\_category\_l1 =[]

color\_by2 = 'Off'

select\_category\_l2 = []

remove\_fail = 'yes'

data\_select = 'all'

fail\_pic\_path ='/tmp/CPK\_Log/fail\_plot/'

excel\_report\_item = 'all' #fail all

fail\_plot\_to\_excel = 'no' # yes no

zoom\_type = 'limit'

header\_list,df,project\_code,build\_stage,station\_name,start\_time\_first,start\_time\_last,param\_item\_start\_index = open\_all\_csv(event,all\_csv\_path,data\_select,remove\_fail)

excel\_report\_file\_name = ''

create\_keynote\_report\_by\_cpk(event,header\_list,df,color\_by1,fail\_pic\_path,select\_category\_l1,cpk\_lsl,cpk\_usl,

cpk\_path,set\_bins,excel\_report\_file\_name,project\_code,build\_stage,

station\_name,start\_time\_first,start\_time\_last,color\_by2,select\_category\_l2,

excel\_report\_item,fail\_plot\_to\_excel,zoom\_type,param\_item\_start\_index,check\_cpk\_thhld,

k\_item\_list,project\_name,target\_build)

print('create keynote report finished!')

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,create keynote report finished")

def generate\_report\_for\_keynote\_1b\_no(table\_data):

cpk\_lsl = table\_data[0] # cpk lsl

cpk\_usl = table\_data[1] # cpk usl

cpk\_path = table\_data[2] #'[NSString stringWithFormat:@"%@/CPK\_Log/",desktopPath];

filelogname = '/tmp/CPK\_Log/temp/.keynote.txt'

set\_bins = table\_data[3] #250

all\_csv\_path = table\_data[4] #'csv 读取的数据路径，从此路径获得数据

csv\_select\_k\_path = table\_data[5] #'界面上UI K 列选择的item csv

csv\_select\_k\_path = os.path.join(csv\_select\_k\_path+ '')

k\_item\_list = read\_csv\_to\_list(csv\_select\_k\_path)

check\_cpk\_thhld = table\_data[6] # yes/no

check\_one\_limit = table\_data[7] # yes/no

project\_name = table\_data[8]

target\_build = table\_data[9]

event = 'keynote-report'

color\_by1 = 'Off'

select\_category\_l1 =[]

color\_by2 = 'Off'

select\_category\_l2 = []

remove\_fail = 'yes'

data\_select = 'all'

fail\_pic\_path ='/tmp/CPK\_Log/fail\_plot/'

excel\_report\_item = 'all' #fail all

fail\_plot\_to\_excel = 'no' # yes no

zoom\_type = 'limit'

header\_list,df,project\_code,build\_stage,station\_name,start\_time\_first,start\_time\_last,param\_item\_start\_index = open\_all\_csv(event,all\_csv\_path,data\_select,remove\_fail)

excel\_report\_file\_name = ''

create\_keynote\_report\_by\_p\_val(event,header\_list,df,color\_by1,fail\_pic\_path,select\_category\_l1,cpk\_lsl,cpk\_usl,cpk\_path,set\_bins,excel\_report\_file\_name,project\_code,build\_stage,station\_name,start\_time\_first,start\_time\_last,color\_by2,select\_category\_l2,excel\_report\_item,fail\_plot\_to\_excel,zoom\_type,param\_item\_start\_index,check\_cpk\_thhld,k\_item\_list,project\_name,target\_build)

print('create keynote report finished!')

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,create keynote report finished")

def run(n):

while True:

try:

print("wait for keynote client ...")

zmqMsg = socket.recv()

socket.send(b'keynote.key') # socket.send(ret.decode('utf-8').encode('ascii'))

if len(zmqMsg)>0:

key = zmqMsg.decode('utf-8')

if key == 'generate\_keynote\_1a\_yes':

print("message from keynote client:", key)

table\_data = get\_redis\_data(key)

if len(table\_data)>0:

generate\_report\_for\_keynote\_1a\_yes(table\_data)

else:

print("---get keynote data error")

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,create keynote report error")

return

if key == 'generate\_keynote\_1b\_yes':

print("message from keynote client:", key)

table\_data = get\_redis\_data(key)

if len(table\_data)>0:

generate\_report\_for\_keynote\_1b\_yes(table\_data)

else:

print("---get keynote data error")

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,create keynote report error")

return

if key == 'generate\_keynote\_1b\_no':

print("message from keynote client:", key)

table\_data = get\_redis\_data(key)

if len(table\_data)>0:

generate\_report\_for\_keynote\_1b\_no(table\_data)

else:

print("---get keynote data error")

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,create keynote report error")

return

else:

time.sleep(0.05)

except Exception as e:

print('error keynote:',e)

filelogname = '/tmp/CPK\_Log/temp/.keynote.txt'

with open(filelogname, 'w') as file\_object:

file\_object.write("Finished,create keynote error: " + str(e))

if \_\_name\_\_ == '\_\_main\_\_':

# t1 = threading.Thread(target=run, args=("<<correlation>>",))

# t1.start()

run(0)