



# QubeXL TECHNO-FINANCIAL MODELER

2024



**PrimeThought**  
Software Solutions

# Introduction to PrimeThought

- PrimeThought Software Solutions is a well-established South African company in operation for over a decade.
- We provide to end spatial analytic solutions

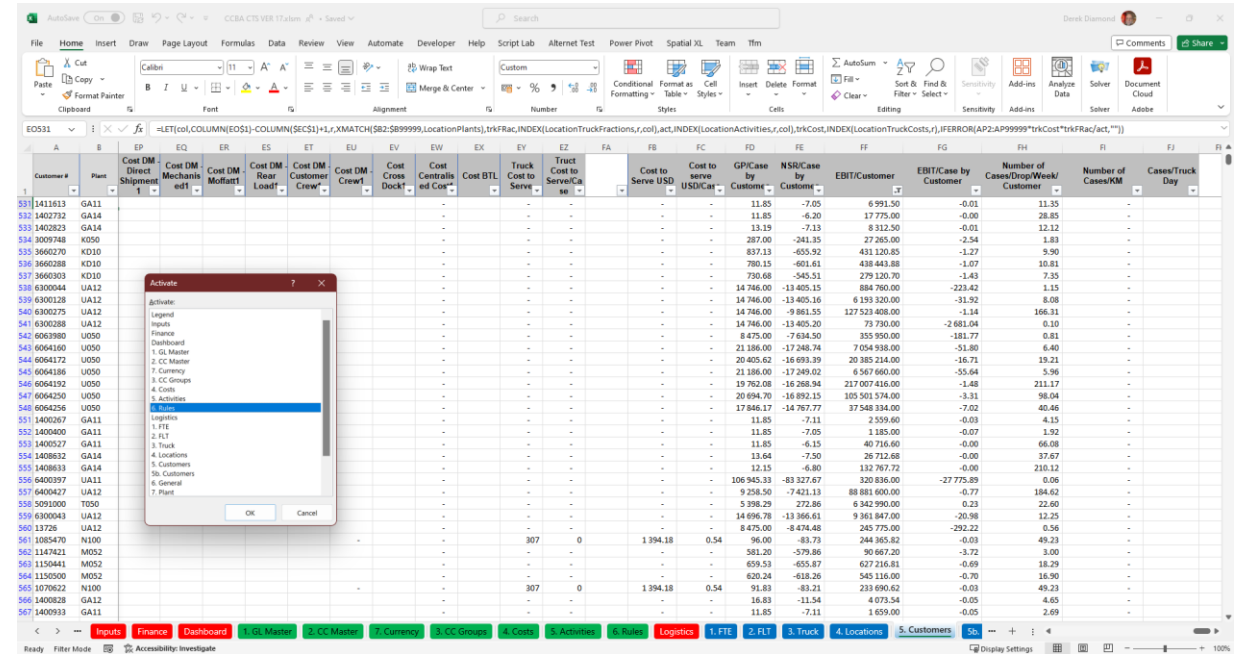


# Our clients



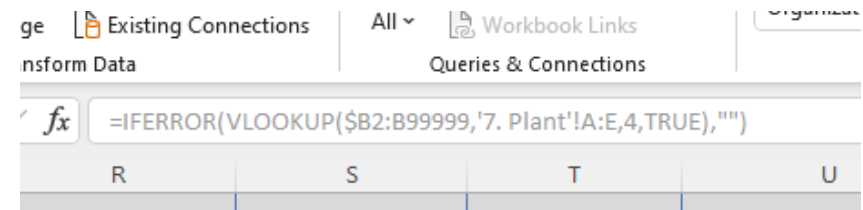
# QubeXL what is it?

- QubeXL is a Techno-Financial modelling system.
- These models are traditionally created in Microsoft Excel.
- For small and simpler models this can be acceptable but when the models result in 100's of columns and many sheets they become unwieldy.
- This manifests in the following ways:



# Formulae are not immediately understandable

- For example
- We can't immediately see what is being referenced in the VLOOKUP



# Complex formulae are almost undecipherable

```
=LET(col,COLUMN(BK$1)-COLUMN($BA$1)+1,r,XMATCH($B2:$B99999,LocationPlants),act,INDEX(LocationActivities,r,col),IFERROR(AN2:AN99999/act*INDEX(LocationDirectCosts,r,col),""))
```

- In the above this is despite using named ranges and the new LET function

# Large workbooks can take minutes to calculate

- This is usually caused by the very inefficient VLOOKUP function and the fact that many intermediate cells need to be used to handle complicated formulae

# Errors can occur in formulae that will not be detected

- All cells need to be inspected which can take ages to verify with and instances can easily be missed



# Invalid data can be entered and not detected

- For example entering a number as text.

# Lookups on more than one column

- Require additional columns
- Notoriously slow
- Crux of any complex model!

# Auditing these models is very complex

- You can get dependents and precedents of a cell, but only to one level, which makes it very time consuming and error prone to trace the calculation of a cell

# Complex logic is very hard to encode

- Even harder to verify and test

# Can't organize many worksheets

- Only have one large flat list of worksheets

494	1400594	GA11	-	-	-	-	-	-	9 600	-	-	-	-	-	-	-	-	-	-	-
495	1402208	GA14	-	-	-	-	-	-	4 085	-	-	-	-	-	-	-	-	-	-	-
496	1402218	GA14	-	-	-	-	-	-	1 240	-	-	-	-	-	-	-	-	-	-	-
497	1402248	GA14	-	-	-	-	-	-	1 325	-	-	-	-	-	-	-	-	-	-	-
498	1402333	GA14	-	-	-	-	-	-	964	-	-	-	-	-	-	-	-	-	-	-
499	1402387	GA14	-	-	-	-	-	-	326	-	-	-	-	-	-	-	-	-	-	-
500	1402407	GA14	-	-	-	-	-	-	530	-	-	-	-	-	-	-	-	-	-	-
501	1402418	GA14	-	-	-	-	-	-	1 355	-	-	-	-	-	-	-	-	-	-	-
502	1402472	GA14	-	-	-	-	-	-	3 190	-	-	-	-	-	-	-	-	-	-	-
503	3660094	KD10	-	-	-	-	-	-	2 817	-	-	-	-	-	-	-	-	-	-	-
504	3660301	KD10	-	-	-	-	-	-	1 114	-	-	-	-	-	-	-	-	-	-	-
505	3660302	KD10	-	-	-	-	-	-	1 382	-	-	-	-	-	-	-	-	-	-	-
506	3660304	KD10	-	-	-	-	-	-	1 306	-	-	-	-	-	-	-	-	-	-	-

Navigation bar: Inputs Finance Dashboard 1. GL Master 2. CC Master 7. Currency 3. CC Groups 4. Costs 5. Activities 6. Rules Logistics 1. FTE 2. FLT 3. Truck 4. Locations 5. Customers 5b. ... + : <

# Getting data into the model

- Excel has no simple customizable way to get your data into the model
- Very manual resulting in errors

# Getting data out of the model

- No standard way to get the data out for advanced reporting
- Manual and error prone

# QubeXL is designed to address these issues

The screenshot displays the QubeXL V1.1.0.8 interface. The main window is titled 'Table data: Customers' and contains a data table with columns: #, Customer #, Plant, Distributi... (incl ret net = sales vol in PC), Picked Volume, Ret Qty SKU, Pick %, Return-a... %, Number of Drops, Gross Revenue, Net Revenue, Gross Contribu... @ std, and Diet Disc. The table shows data for four customer entries.

Below the table is a code editor with the following script:

```
var ccImportFile = Model.FileFolder + "\\Inputs\\CCSabco 2022 Actual US";
var table = Model.Objects.Item("Finance").Table("Accounts");
var cTab = Model.Objects.Item("Input Output").Table("Countries");
table.ClearRows();
Host.LogProgress("Importing Accounts " + ccImportFile);
table.LoadExcelRange(
    ccImportFile,
    "$B184:L30000",
    ["CO Area", "CC", "CC Description", "Function", "Capability", "Sub-...", "Plant"],
    [],
    null,
    null,
    n => {
        if (!Lib.IsNullOrEmpty(n.RowValues[0].Value)) {
            return false;
        }
    }
);
var crow = cTab.Lookup("Country", n.RowValues[0].Value.slice(0, 2)).FirstOrDefault();
```

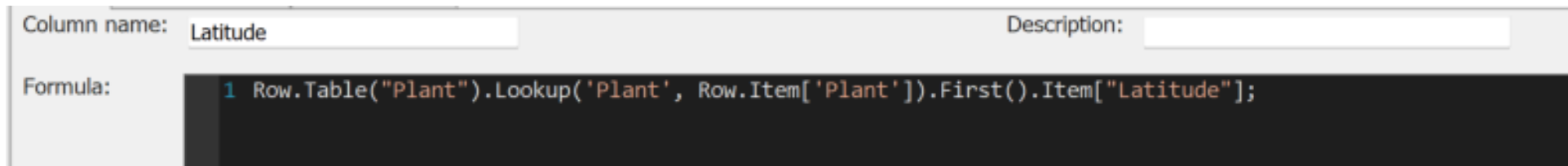
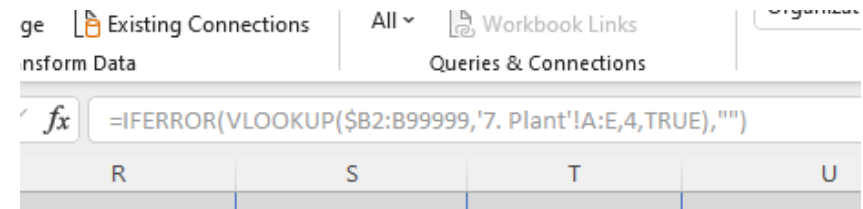
On the right, a dashboard titled 'Locations CTS by Cost Type' features a pivot table and a stacked bar chart. The pivot table shows 'Grand Total' and 'Locations CTS by Cost Type' with columns for Direct CTS \$/UC, FTE CTS \$/UC, FLT CTS \$/UC, TRUCK CTS \$/UC, and Total CTS \$/UC. The chart displays 'USD / UC' for five locations: KEETMANSHOOP, OSHAKATI, OTJIWARONGO, WALVISBAY, and WINDHOEK, with a legend for Direct CTS, FTE CTS, FLT CTS, and TRUCK CTS.

The bottom status bar indicates 'CTS Namibia Master.QubeXLC Loaded, 0 Calculation error(s)'.



# Formulae are immediately understandable

- For example
- In QubeXL we can immediately see we are looking up the Latitude column from the Plants table













# Complex formulae are more understandable

```
=LET(col,COLUMN(BK$1)-COLUMN($BA$1)+1,r,XMATCH($B2:$B99999,LocationPlants),act,INDEX(LocationActivities,r,col),IFERROR(AN2:AN99999/act*INDEX(LocationDirectCosts,r,col),""))
```

```
var r = Row.Tables['Locations'].Lookup('Plant', Row.Item['Plant']).First();  
Lib.Divide(Row.Item['Bt1'], r.Item['Plant Bt11']) * r.Item['Direct Costs Bt11'];
```

- Here we lookup the plant of the customer in line 1 and do a divide in line two of the rows value by the plants value and multiply it by an amount in the row

# Large models take up little space

Name	Status	Date modified	Type	Size
Inputs	 	2023/09/11 09:30	File folder	
Templates	 	2023/09/11 09:30	File folder	
 CCBA CTS VER 17.xlsm	 	2022/02/22 13:48	Microsoft Excel M...	61 715 KB
 CCBA CTS VER 17.QubeXLC	 	2023/09/08 11:30	QubeXL Project (C...	1 449 KB

Files are 1 /20 the size of the Excel workbook

# Errors prevent the model from calculating

- Any error in the model stops calculation

The screenshot shows a software interface with a 'Details: column: Direct Cost BTL1 table: Logistics[Customers]' window. The 'Formula' field contains the following code:

```
1 //  
2 // We take the customer activity divided by plant or OCCD activity  
3 //  
4 if (Lib.IsNullOrEmpty(Row.Item['OCCD'])) {  
5     var r = Row.Tables['Locations'].Lookup('Plant', Row.Item['Plant'])  
6     Lib.Divide(Row.Item['Btl'], r.Item['Plant Btl1']) * r.Item['Direct Cost']  
7 }  
8 } xxx  
9 else {  
10    var r = Row.Tables['OCCD'].Lookup('OCCD #', Row.Item['OCCD'])  
11    Lib.Divide(Row.Item['Btl'], r.Item['Btl']) * r.Item['Direct Cost']  
12 }  
13 }
```

A 'Calculation Errors' dialog box is open, displaying the following table:

Error	Name	Parent
Error calculating row 3614: Line 9: Unexp...	Direct Cost BTL1	Logistics[Customers]

# Data is typed per column of tables

Details: table: Logistics[Customers]

Columns | Script | Documentation

Name: Customers  Read Only

Drag a column header here to group by that column

	Column Name	Type	Group	Format
Y	#c	#c	#c	=
▶	Customer #	int		
	Plant	string		
	Distribution Volume (incl ret) net = sales v...	float		N0
	Picked Volume	float		N0
	Ret Qty SKU	float		N0
	Pick %	float		N1
	Return-able %	float		N0
	Number of Drops	float		N0
	Gross Revenue	float		C0
	Net Revenue	float		
	Gross Contribution @ std	float		
	Dist Discount	float		
	Other Discounts	float		C2
	Mode - NOW	string		
	Net Revenue USD	float		C2
	Gross Profit USD	float		C2
	Sales Volume UC	float		N0
	OCCD	string		
	Scenario Pick%	float		N2
	Distance from Depot (km)	float		N1
	Nominal Distance Travelled	float		

# Lookups on more than one column

Details: column: Plant table: Finance[Costs]

Column  Format\_Lookups Documentation

Column name:  Description:

Formula: 

```
1 Row.Tables['Accounts'].Lookup(['Account','CC'], [Row.Item['Account'], Row.Item['Cost Center']]).First().Item['Plant']
```

From table Accounts, lookup using columns Account and CC and values from column Account and Cost Center in this table, find first matching row and get Plant name

# Auditing these models is easy

The screenshot displays the QubeXL V1.1.0.8 software interface. The main window shows a project structure on the left and a details pane for a column named 'Plant table: Finance[Costs]'. The formula for this column is: `= Row.Tables['Accounts'].Lookup(['Account','CC'], [Row.Item['Account'], Row.Item['Cost Center']]).First().Item['Plant']`. A dialog box titled 'Precedents of EBIT/Customer' is open, showing a table of dependencies. A red arrow points from the 'EBIT/Customer (float)' item in the structure to the dialog box.

Parent	Name	Index	Type	Changed
	=			
	Logistics	49	Object	False
	Customers	448	Table	False
	Logistics[Customers]	463	Column	False
	Logistics[Customers]	1078	Column	False
>	Logistics[Customers]	975	Column	False
>	Logistics[Customers]	998	Column	False
>	Logistics[Customers]	1067	Column	False
>	Logistics[Customers]	1076	Column	False
	Logistics	49	Object	False
	Customers	448	Table	False
	Logistics[Customers]	514	Column	False
	Logistics[Customers]	515	Column	False
	Logistics[Customers]	516	Column	False
	Logistics[Customers]	517	Column	False
	Logistics[Customers]	518	Column	False
>	Logistics[Customers]	843	Column	False
	Logistics	49	Object	False
>	Logistics[Locations]	332	Column	False
	Logistics[Locations]	327	Column	False
>	Logistics[Locations]	441	Column	False
	Logistics	448	Table	False
	Logistics[Customers]	450	Column	False
	Logistics[Customers]	466	Column	False
>	Logistics[Customers]	481	Column	False
>	Logistics[Locations]	812	Column	False
>	Logistics[Customers]	844	Column	False
>	Logistics[Customers]	1000	Column	False

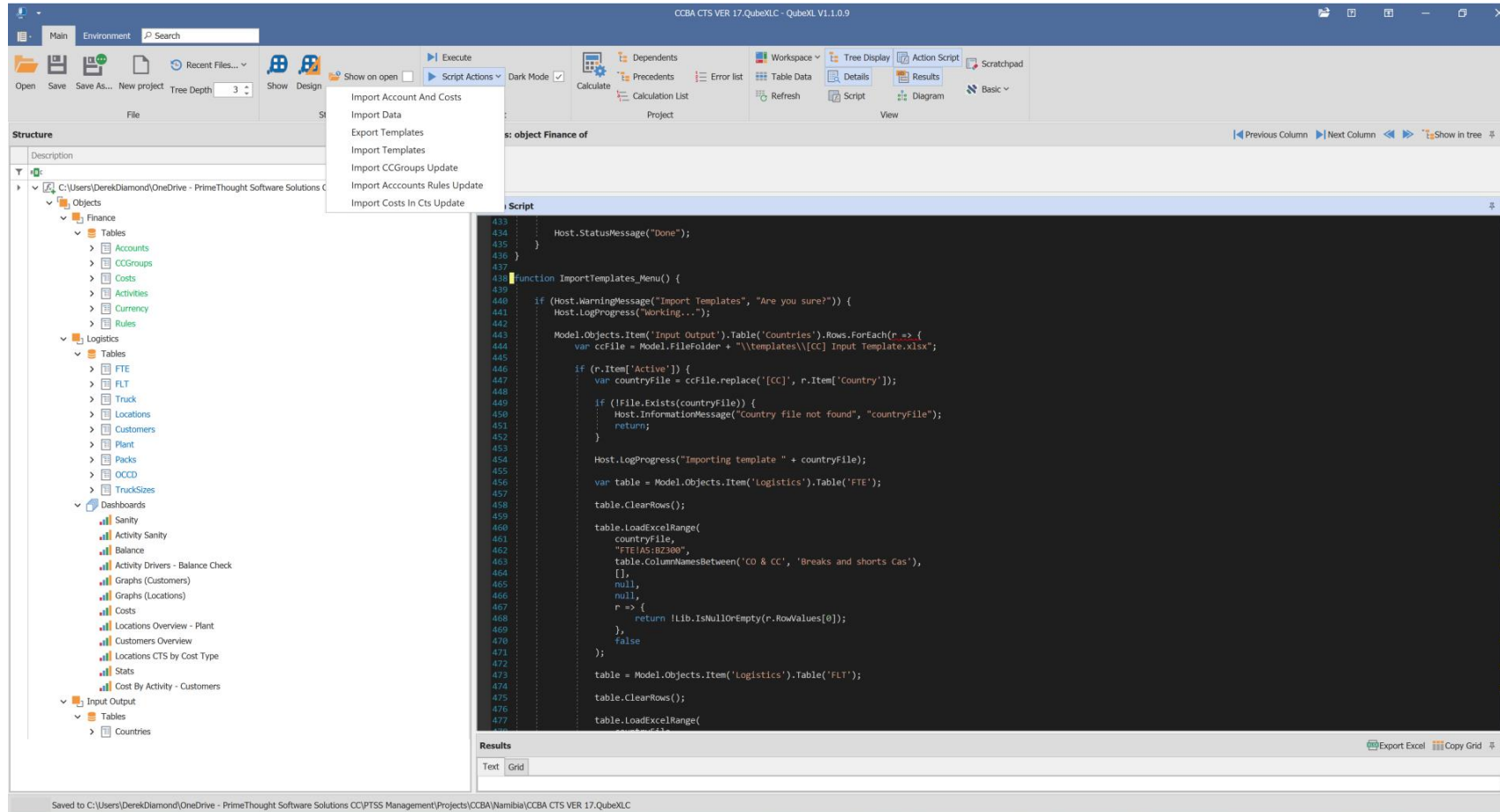
# Complex logic can be encoded

```
1 //
2 // We take the customer activity divided by plant or OCCD activity times the plant or OCCD costs times the fraction
3 //
4 if (Lib.IsNullOrEmpty(Row.Item['OCCD'])) {
5     var r = Row.Tables['Locations'].Lookup('Plant', Row.Item['Plant']).First();
6
7     Lib.Divide(Row.Item['Picking'], r.Item['Plant Picking']) * r.Item['Truck Costs'] * r.Item['Truck Costs Fraction Picking'];
8 }
9 else {
10    var r = Row.Tables['OCCD'].Lookup('OCCD #', Row.Item['OCCD']).First();
11
12    Lib.Divide(Row.Item['Picking'], r.Item['Picking']) * r.Item['Truck Costs'] * r.Item['Truck Costs Fraction Picking'];
13 }
14
```





# Getting data into the model fully customizable and repeatable



# Simple and Auditable Models

QubeXL simplifies things by having formulae and calculations per column, table and object instead of per cell as in Excel. This allows for simpler and auditable models which are easily verified and extended.

The screenshot displays the QubeXL software interface, divided into several panels:

- Structure Panel (Left):** A tree view showing the model's hierarchy. It includes 'Logistics', 'Annotations', 'Tables', 'FTE', and 'Columns'. The 'Total FTE (float)' column is highlighted in yellow.
- Details Panel (Top Right):** Titled 'Details: column Total FTE'. It shows the column name 'Total FTE', a 'Group' field, a 'Read Only' checkbox, and a 'Test' button. The formula field contains: `1 Row.Item['Perm'] + Row.Item['FTC'] + Row.Item['Casual']`.
- Diagram Panel (Bottom):** A visual dependency diagram showing the 'Total FTE' column as a central node connected to its constituent columns: 'Perm', 'FTC', and 'Casual'. It also includes a search bar and a 'Pan & Zoom' control.

# Simple and Powerful Scripting Language

QubeXL uses the most popular scripting language in the world: JavaScript for all its scripting and calculations. As you build your model QubeXL generates a diagram showing your model structure. You are also able to trace calculation dependencies

Details: column Total

Column: Total

Group:

Read Only  Test

Formula:

```
1 Row.Item['Total Small'] * Row.Tables['TruckSizes'].Formulae['SmallFactor'].Value +  
2 Row.Item['Total Med'] * Row.Tables['TruckSizes'].Formulae['MediumFactor'].Value +  
3 Row.Item['Total Large'] * Row.Tables['TruckSizes'].Formulae['LargeFactor'].Value
```

Diagram

Dependencies Show Columns Zoom Fit

Design Search

Pan & Zoom

Dependents of Total

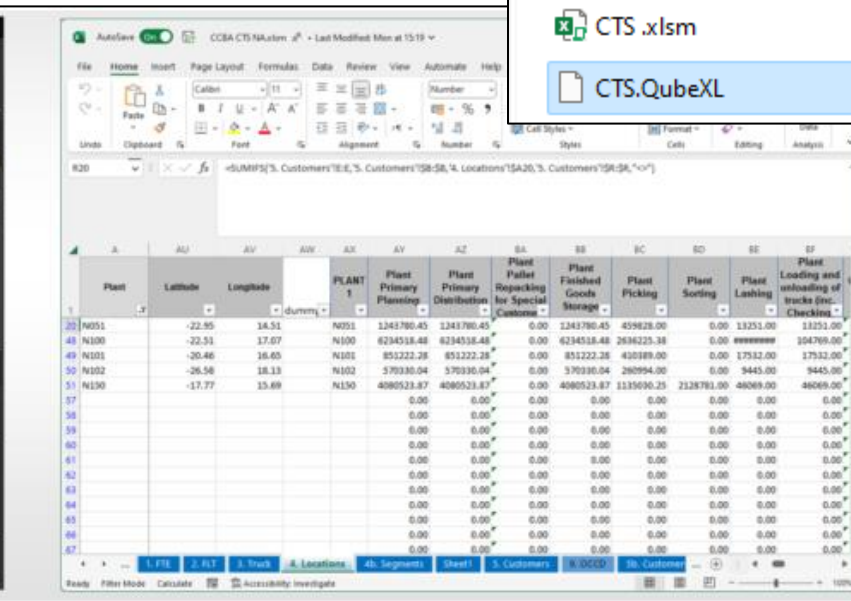
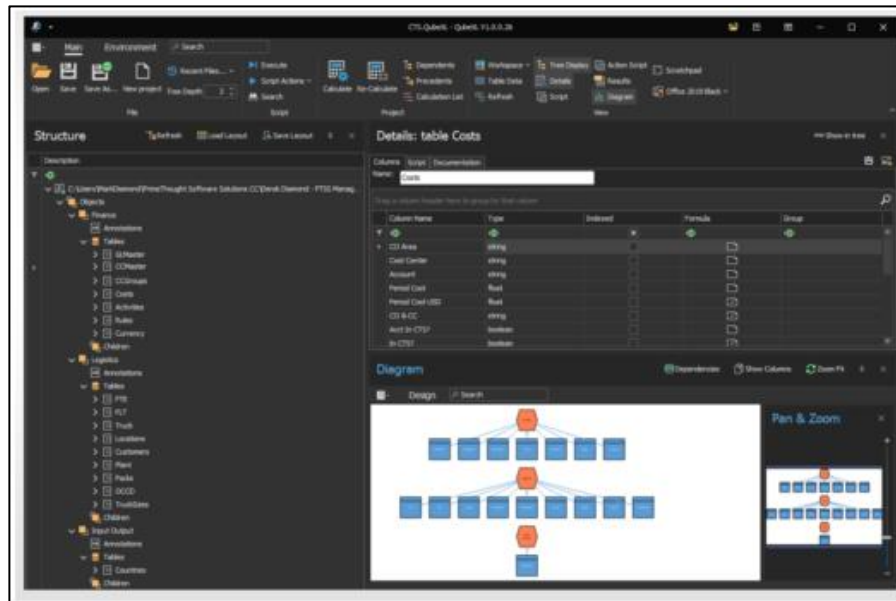
Dependents  Find

Enter text to search... Find

Parent	Name	Index	Type	Changed
=	=	=	Column	=
> Logistics[Locations]	Truck Costs Truck Total	767	Column	False
Logistics[Locations]	Truck Costs Actual Number of Trucks	789	Column	False
Logistics[OCCD]	Truck Actual Number of Trucks	803	Column	False

# Powerful and fast

- Because QubeXL uses data frames borrowed from data science systems it can handle millions of rows per table and calculate these in times incomparable with traditional Excel models.
- Further it is much more compact. As an example, the exact same financial model in Excel is 73MB while in QubeXL it is only 9MB:

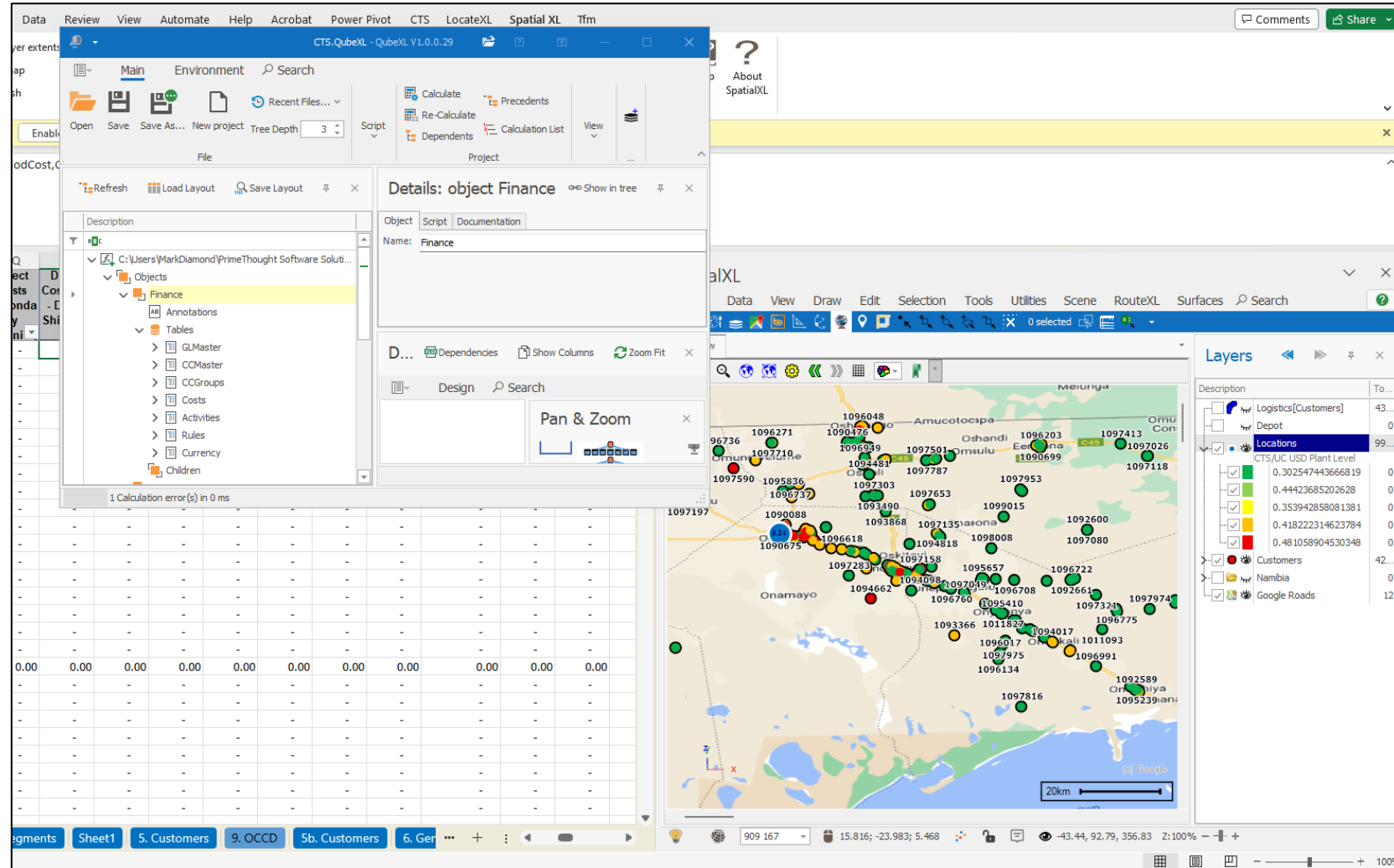


The image displays two side-by-side screenshots. The left screenshot shows the QubeXL interface with a dark theme, featuring a 'Structure' pane on the left and a 'Details: table Costs' pane on the right. The right screenshot shows the Microsoft Excel interface with a financial model spreadsheet. The spreadsheet has columns for Plant, Latitude, Longitude, PLANT 1, Plant Primary Planning, Plant Primary Distribution, Plant Pallet Respecting for Special Customer, Plant Finished Goods Storage, Plant Picking, Plant Sorting, Plant Lashing, and Plant Loading and unloading of trucks (inc. Checking). The data rows show values for various plants, with some cells containing zero or small numbers.

Name	Size
CTS.xlsm	73 522 KB
CTS.QubeXL	9 756 KB

# Excel Add-In

QubeXL is able to plug in to SpatialXL which is our mapping and spatial analytics tool that integrates with Excel. Your model data can then be plotted on a map and themed etc. with full business intelligence and GIS capability offered by SpatialXL:



# QubeXL key features

<http://qubexl.com/> & <https://primethought.biz/>

1. Object Oriented techno-financial modeller
2. Handles millions of rows of data
3. Scriptable imports and ETL from databases and Excel
4. Self describing formulae
5. Multi views on tables
6. Scriptable model generation and extension
7. Built AI to build prediction models
8. Built in dashboarding
9. Built-in auditability and traceability
10. Built-in Storyboarding™
11. Integration with our mapping, routing and spatial products
12. Excel Add-in





THANK YOU

Visit [primethought.biz](http://primethought.biz) for more information

or email

[sales@primethought.biz](mailto:sales@primethought.biz)