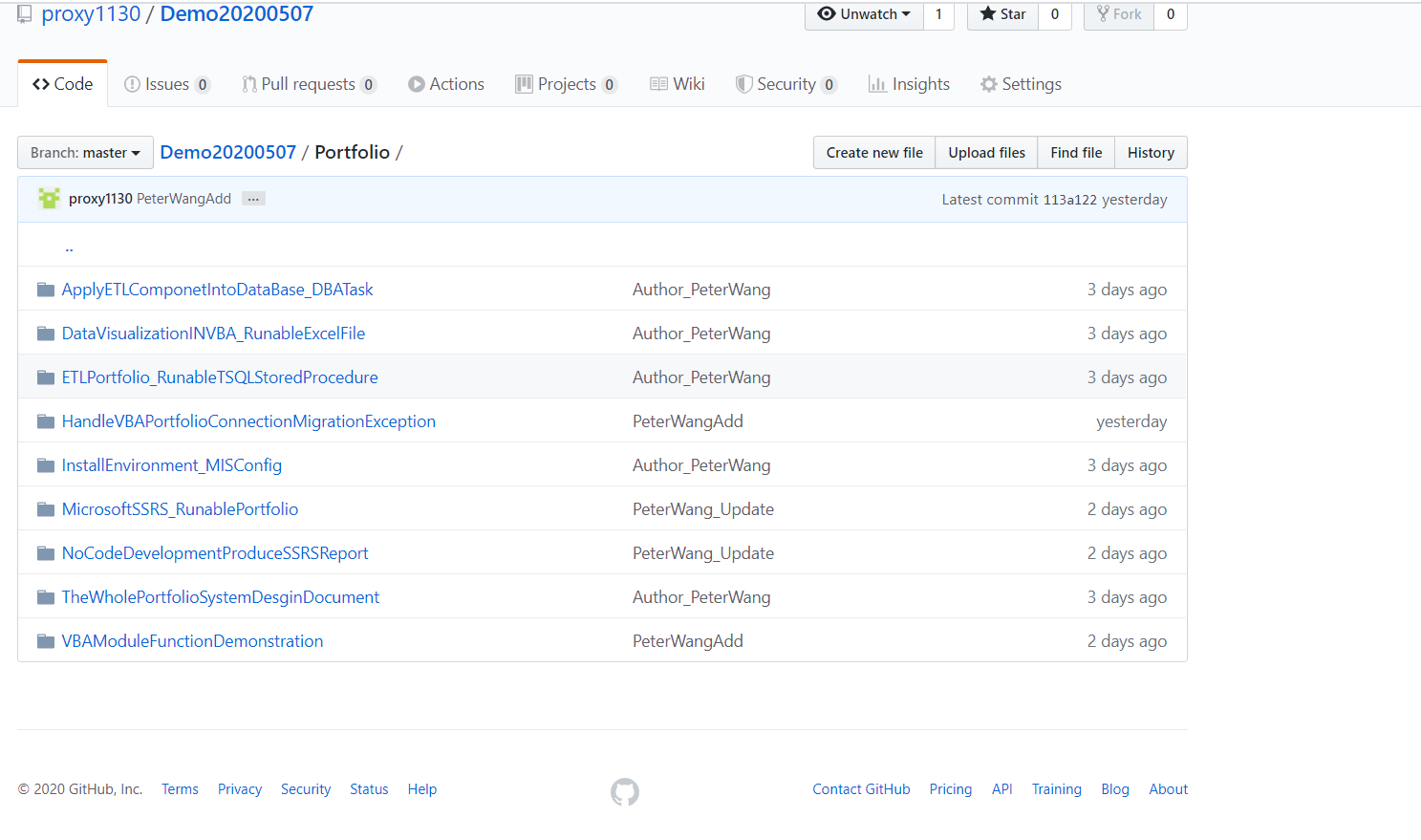
1. 為何如下圖所示，會有這個GitHub repository 所存放相關成品於GitHub呢?

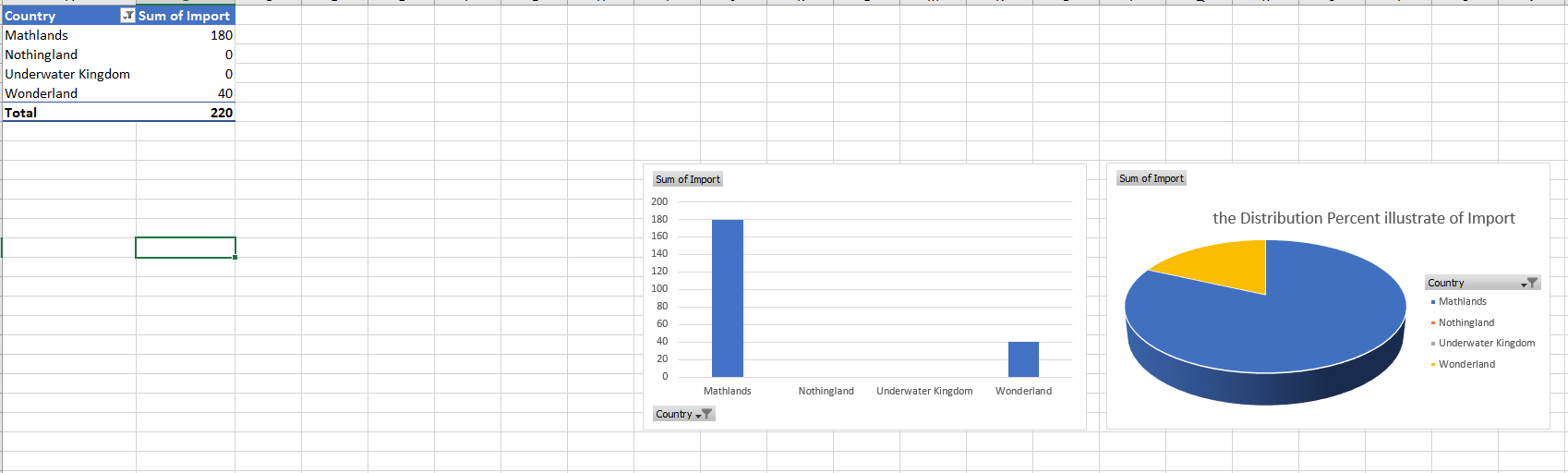
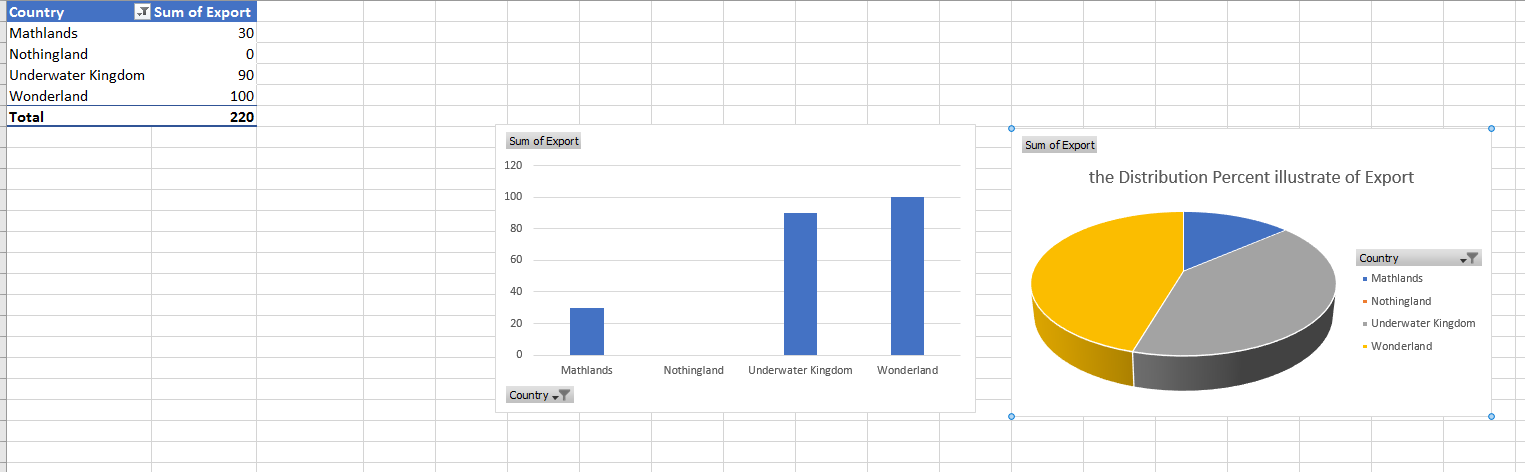


主要是因為求職者王竣生本身IT產業程式領域工作經歷豐富，多以大型企業集團ERP(Enterprise Resource Planning)與DMP(Data Management Platform)兩大類系統為工作主體，並且擔任MIS/DBA/SQA/PG一系列工作職務→這兩大類大型系統後端資料庫核心功能就會提供DW(data warehouse)領域相關功能，敝人的程式開發專業也多半以此為核心→更具體來講，就是開發ETL相關資料庫後端元件與前端負責Data Visualization相關的報表Dashboard→也就如下圖所示(文章出處: 擷取、轉換和載入 (ETL) 流程 [https://docs.microsoft.com/zh-tw/azure/architecture/data-guide/relational-data/etl ])

一張含有 螢幕擷取畫面 的圖片

自動產生的描述

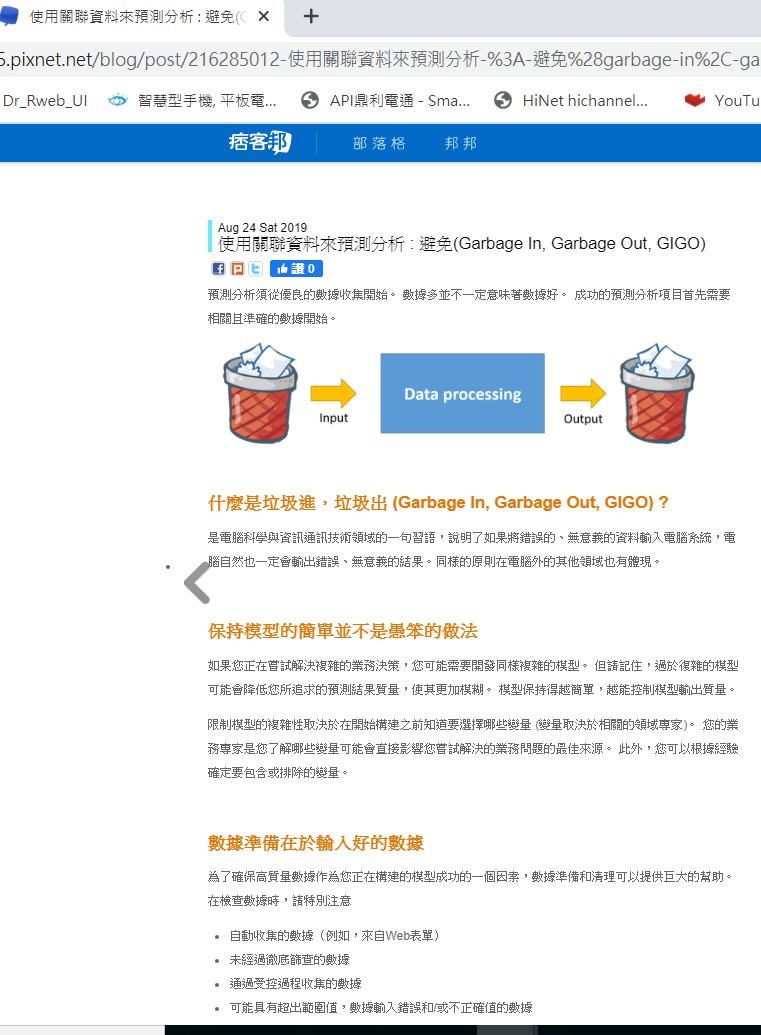
在GitHub repository(Demo2020507)所記載的專案，本身內容就是一個國際跨國集團所屬各國家分公司的特定期間總體營運→ETL後端資料處理元件TSQL Stored Procedure，就是將記載集團營運績效數字的兩張資料表，使用TSQL 關聯運算去產生集團管理階層所要的結果→然後當使用者按下Excel VBA分析模組的特定功能鍵，如下圖所示出現，用樞紐分析與圓餅長條圖表去呈現出每一個國家所屬分公司的具體銷售情況:



至於關於GitHub repository(Demo2020507)所記載的專案相關系統設計細節，可以參考以下路徑「,,,\Portfolio\TheWholePortfolioSystemDesginDocument」的以下檔案「SQL技術ETL成品用VBA程式來呈現.pdf」！而且為了驗證這一個ETL資料庫元件的成品可行度，因此在「SQL技術ETL成品用VBA程式來呈現.pdf」這一篇的第36~53頁，用初等會計學去針對ETL元件加工成品進行POC(Proof of Concept)驗證，即證明出ETL加工成品是正確無誤的！

隨後會於本GitHub repository的以下兩個目錄『NoCodeDevelopmentProduceSSRSReport』、「MicrosoftSSASBase\_MDXQuery\_Demonstration」，進行Microsoft SQL Server環境關於SSRS與SSAS相關演練與DMX分析環境架設→其中『NoCodeDevelopmentProduceSSRSReport』是將前述VBA 仿RPA的自動化會計資料檢核模組轉化到SQL Server內建SharePoint Server，因此Client 端使用者可以不用裝Excel用網頁瀏覽器就可以看到最新內容更豐富的成品→至於「MicrosoftSSASBase\_MDXQuery\_Demonstration」這一個目錄，是為了後續RPA自動化的Dashboard所需後端資料庫SSAS模組，還有MDX Cube等一系列後續開發的先遣者!

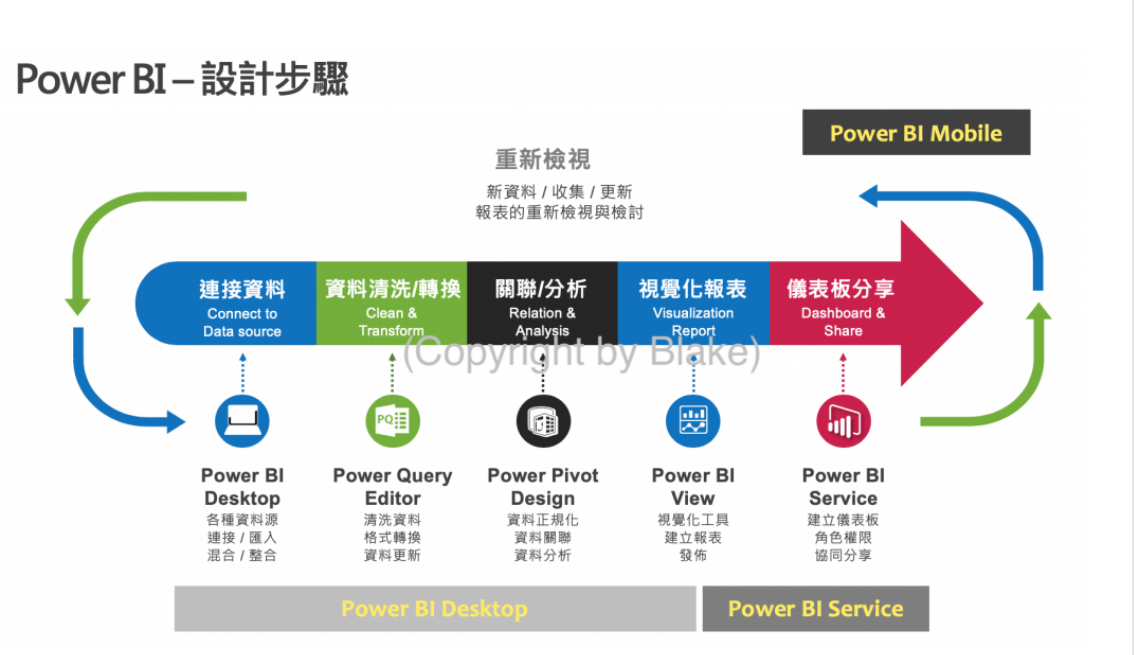
為何這一份個人作品，會花費大量篇幅去描述進行所謂資料庫層面的POC(Proof of Concept)驗證→並非只是單純展現作品作者本身MIS與DBA的專業能力，而是因為作品作者平生所經歷大型專案導入未能商業化正式營運的教訓，所累積經驗讓作者自然而然有感而發的!這些經驗總結起來，就可以用一句話來總結:「Garbage In, Garbage Out」，也就是如下圖所示:



文章出處: [使用關聯資料來預測分析 : 避免(Garbage In, Garbage Out, GIGO) ]: <https://wayne265265.pixnet.net/blog/post/216285012-%E4%BD%BF%E7%94%A8%E9%97%9C%E8%81%AF%E8%B3%87%E6%96%99%E4%BE%86%E9%A0%90%E6%B8%AC%E5%88%86%E6%9E%90-%3A-%E9%81%BF%E5%85%8D%28garbage-in%2C-garba>

總而言之，要做一個有分析與預測模組的ERP或DMP大金額的大型方案，必須先有一套可用真實營業資料庫，才能讓整個系統開發流程可以由簡而難循序漸進→否則由本作品作者實戰經驗看來，這類大型專案只是在浪費國家與所屬業主單位的財政資源而已!

接下來講一下本GitHub repository，所要繼續完成整體開發目標為何(也就是最後本repository所提供成品長成什麼樣)?總而言之，因為資料源/資料庫等全部後端部分都是Microsoft的，因此走如下圖所示的Power BI這一個體系:



由上圖可知，本GitHub repository的成品將會一個DMP( (Data Management Platform) 系統:

上圖中所謂「連接資料」(待開發中)，就是執行以下主功能流程:一旦使用者按下匯入成效按鍵→即是將終端使用者提供過來的靜態 EXCEL 空白報表樣板從網頁伺服器自動下載到回目前電腦上，然後自資料庫解析出該 EXCEL 空白報表樣板的資料儲存位址所在之後，再寫入使用者想要看的企業經營成效資料後，最後整張Excel活頁簿將其對應好回存到資料庫→ 一旦使用者按下報表匯出鍵，想去讀已經存檔的報表，就原封不動用template檔產生一份格式與內容符合其需求的成果讓其下載回去使用!

而上圖所謂資料清洗/轉換(待開發中)→簡單來說，就執行以下處理流程:檢查數據一致性，處理無效值和缺失值，必要時去檢視不同Excel活頁簿之間的關聯性驗證→資料清洗與轉換實作類型1:使用Excel程式內建以下功能函數與資料比對指令(IF，VLOOKUP，HLOOKUP，MATCH，INDEX，XLOOKUP)，去找出無效值和缺失值來加以處理→資料清與轉換實作類型2:使用T-SQL領域屬於MDX(Multidimensional Expressions)與SSIS(SQL Server Integration Services )兩主題相關的資料庫查詢語言中，關於資料完整性檢查(Data Integrity Check)強化語法，去除與處理處理無效值和缺失值即可!

而上圖所謂「關聯/分析」(已開發需擴展)→簡單來說，就在以下目錄「TheWholePortfolioSystemDesginDocument」已經完成一個簡易的後端資料庫體系→但是因為資料量不足，因此又在「MicrosoftSSASBase\_MDXQuery\_Demonstration」目錄中，去拓展出以下新的「Contoso\_Retail」資料庫→因此後續將於Microsoft SSIS與SSAS領域去擴充資料源與Cube等成品，讓「關聯/分析」這一個部分更有看頭!

而上圖所謂「視覺化報表」(已完成待轉化) →簡單來說，就在以下目錄「NoCodeDevelopmentProduceSSRSReport」已經完成四張完整成品→日後就是隨SSIS等模組的增加，去擴展該目錄的成品→最後因應Power BI 環境的建立，現有的SSRS報表都應該被轉化重新發行到Power BI Service之上!

而上圖所謂「視覺化報表」(環境架設與RPA自動化為其主要目標)→簡單來說，讓全部的Power BI Service轉化發布現有SSRS報表的操作，儘量使用RPA流程控管方式進行自動化程式來完成，而不是跟以往一樣用手動手工藝去完成的!