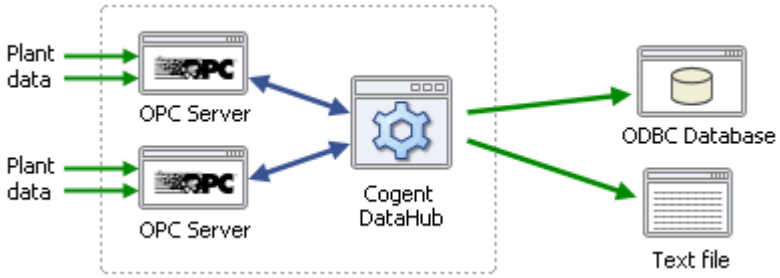
章節 8. 資料記錄

目錄

8.1. 簡介

Cogent DataHub可以將資料記錄到任何ODBC相容的資料庫或文字檔[1]



使用DataHub的這個功能，您可以:

- 將資料記錄到任何ODBC相容的資料庫，如MS Access、MS SQL Server、MySQL、Oracle等等。
- 從任何連接到DataHub的資料來源進行紀錄。
- 紀錄到現有的資料庫表格，或者是依需求建立新的表格。
- 使用`LogFile.g`指令碼將資料記錄到文字檔。

> 爲了將資料從資料庫寫進Cogent DataHub，請參閱**DataHub ODBC指令碼**使用手冊裡的**教學課程 3: 將資料從資料庫寫入DataHub**。

Cogent DataHub的ODBC Data Logging介面提供一個連接到DSN的簡單方法，建立或選取一個表格，還有指派資料點屬性給表格欄，以及指派紀錄點給一個觸發程序或條件。學習如何使用介面的最快方法是觀看網站的影片或是使用**快速入門**。

Click here to watch a video.

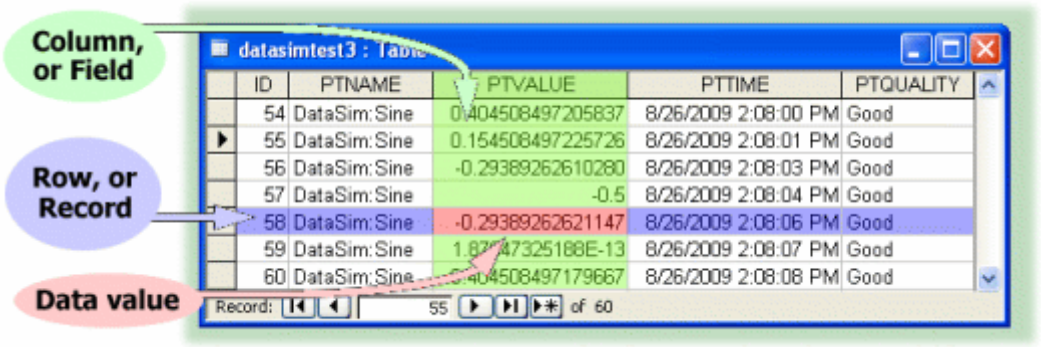
8.1.1. 資料庫的專有名詞和概念

一般而言，*資料庫*是資料的集合物件。電腦化的資料庫將資料儲存在*表格*裡，它可以透過*資料庫管理系統*或*DBMS*，如SQL Server、MS Access、MySQL、Oracle等等來進行存取。這些全部都能夠在相關聯、已連結的tables儲存資料，它們綜觀來說，稱爲*關聯式資料庫*。大多數現代電腦化的資料庫都是關聯式資料庫。

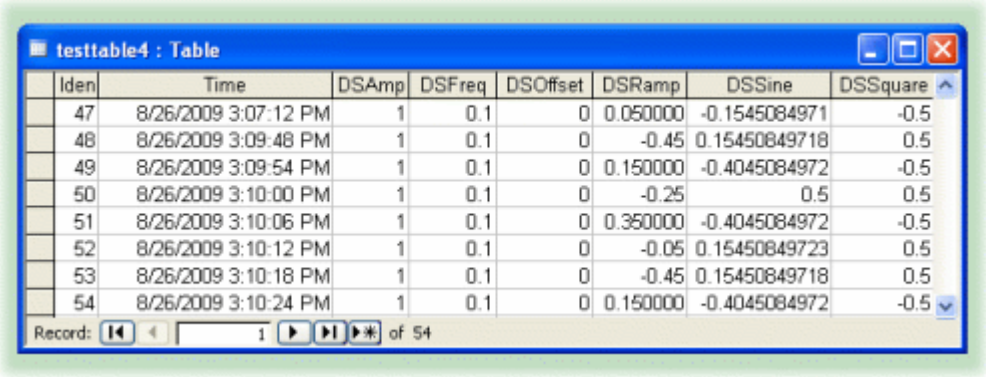
一個資料庫*表格*是相關聯資料的邏輯群組，由通用屬性或個別資料items的特性組織而成的。每個表格裡的*資料行*或*欄位*包含一個特定的屬性，而且是一個特定的*資料類型*。典型的資料類型包括boolean、字串、數值、日期/時間等等。典型的資料類型包括boolean、字串、數值、日期/時間等等。資料表裡的每一*行*或*資料列*包含每個與單一item相關聯之*資料值*的完整

組。

例如，一個包含Cogent DataHub資料的資料表可能會有點名稱、值、時間戳記、和品質的資料行(欄位)。每一行(資料列)會顯示不同的時間點裡面紀錄該點的各種資料值。



不同類型的資料表可能會有一個時間戳記的欄位，以及其它的欄位，裡面包含不同的DataHub點在不同時間裡紀錄的值，像是:



資料庫table也許會要求每一行(或資料列)都是唯一識別。這通常是透過一個索引欄來完成，其主要(有時是唯一)目的是提供一個唯一辨識的值給資料列。大多數的DBMSs允許該值讓資料庫使用者手動指定，或是透過*自動增量的計數器*或是其他機制設定。資料表擁有多個索引欄的可能性是存在的，但是Cogent DataHub裡的某些函式只會使用一個索引欄寫入資料表。欲了解更多使用Cogent DataHub設定資料庫tables的更多資訊，請參閱**章節 8.5，“設定資料庫Table”**。

8.1.2. 連接到資料庫: ODBC

連接到資料庫是透過DBMS完成的，它通常提供兩種可能性: *原生驅動程式*和*ODBC* (開放式資料連結)。原生驅動程式使用上不方便，因爲每個都需要它自己的程式設計介面。另一方面，ODBC指定一個標準化、通用的介面，幾乎所有的資料庫供應商都可供使用，包括SQL Server、MS Access、MySQL、Oracle以及許多等等。Cogent DataHub使用ODBC連接到資料庫。

ODBC支援在本機或橫跨網路與DBMS進行通訊，藉由使用一個*ODBC驅動程式*。每個ODBC相容的DBMS提供一個ODBC驅動程式，它需要安裝在使用者的機器上。舉例來說，這裡有個可用於MS Access、SQL Server和MySQL等等的ODBC驅動程式。

您可以使用Windows **ODBC資料來源管理員**來設定ODBC驅動程式和您想要共同運作的特定資料庫之間的連線。該設定被稱爲*Data Source Name*或是*DSN*。例如，Cogent DataHub參照DSN以及爲ODBC驅動程式使用已設定的連線以便連接到資料庫。

設定DSN是很簡單的，根據您正在運作的ODBC驅動程式而有些許不同。通常您需要選取一個ODBC驅動程式，爲DSN建立一個名稱，然後選取一個資料庫。其他資訊，如登錄名稱或密碼可能會被要求或選用。欲了解更多資訊，請參閱**章節 8.4，"建立DSN (資料來源名稱)"**

8.1.3. 資料存取: SQL

一旦連接到一個資料庫，任何擷取、修改、新增或刪除資料的*查詢*(請求)必須透過語言來製造。最普遍的資料庫查詢語言是SQL(結構化查詢語言)，讀音爲"sequel"或"ess-kyu-el"。建立於1960年代，雖然在某些提供的命令上有一些小變化，但這種語言已經成爲廣泛使用的標準，大部分的DBMSs都支援。

Cogent DataHub使用SQL以便寫入或讀取資料庫。當您設定Data Logging介面以便寫入DataHub點值時，機器表面之下的機制都是以SQL寫入的。DataHub指令碼也使用SQL命令寫入和讀取資料。例如，下面一行是來自**ODBCTutorial3.g**指令碼，它使用SQL SELECT命令以便從資料庫表格裡提取所有資料的。

```
result = .conn.QueryToClass (.tableclass, string ("select * from ", .tablename));
```

SELECT命令通常與FROM和WHERE運算子一起使用，如以下的查詢:

```
SELECT data_element_1
FROM table_1
WHERE data_element_1 > 32 and data_element_1 < 212;
```

SQL的語法相當簡單，而且有許多書籍和線上教學課程可以幫助您學習。一般來說，這裡所提供SQL、ODBC和資料庫的資訊，應該足以讓您使用Cogent DataHub**開始紀錄資料**。

---

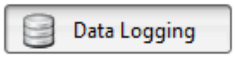
[1] 文字紀錄可以很快在Data Logging介面中供使用，但現在可以透過**DataHub指令碼**設定。

8.2. 快速入門

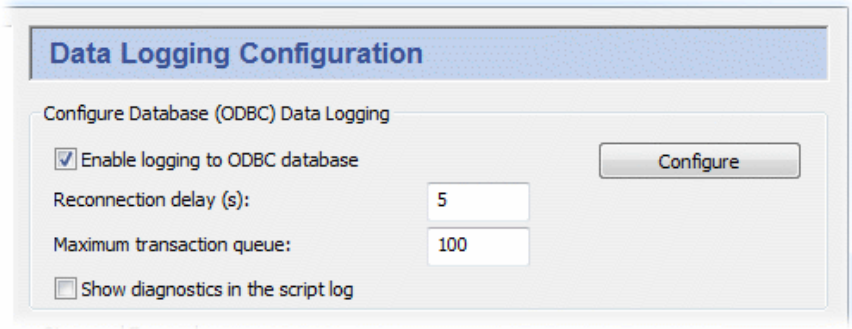
以下將指導您如何把Cogent DataHub設定成寫入資料到您所選取的資料庫。我們在此範例中使用DataSim程式裡的資料點，但也可以使用您自己的資料點，就如同範例般簡單。

Click here to watch a video.

開啓ODBC Data Logging視窗

1. 在Cogent DataHub Properties視窗中，選取**Data Logging** 

。
2. 點擊**Configure**按鈕。

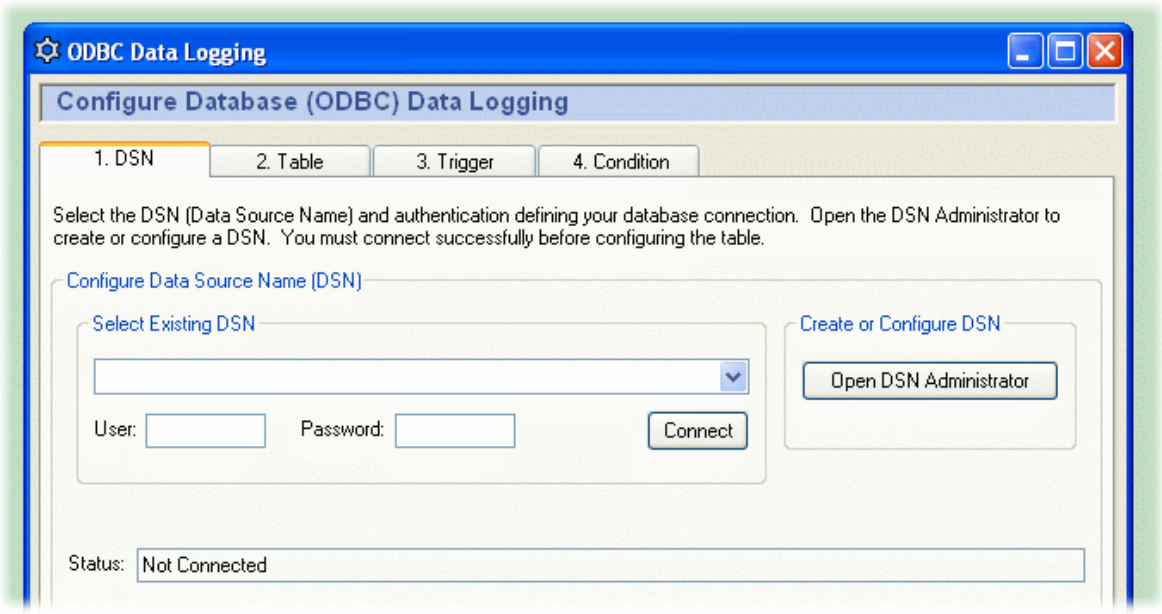


這會開啓ODBC Data Logging視窗，如下所示。

Data Logging Configuration介面的詳細解釋可在**章節 8.3， "設定Queue、Store和Forward"**中找到。至於現在，您可以使用預設值。

連接到資料庫

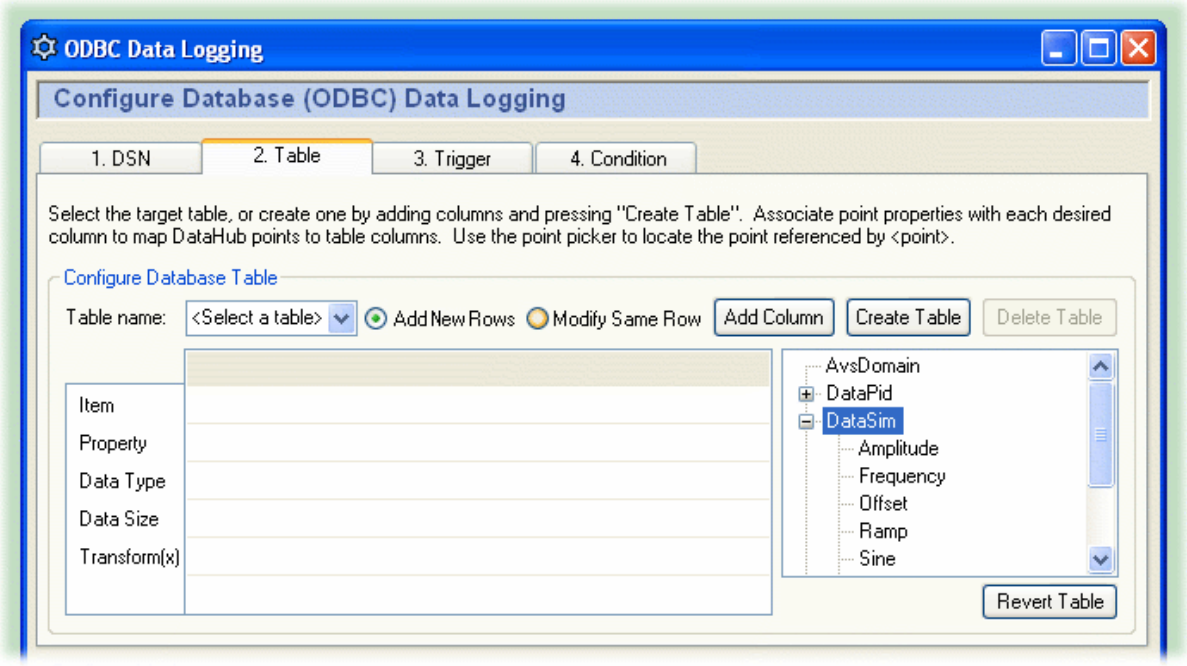
1. 選取**1. DSN**頁籤。*DSN*是資料來源名稱。Windows使用此名稱來識別您想連接的資料庫。



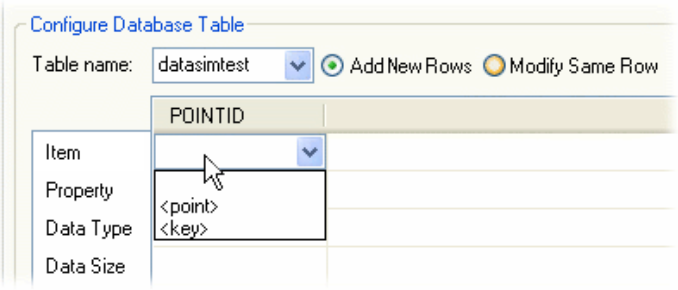
2. 在下拉式清單方框中，選取DSN。如果您沒有任何DSNs，或者是您想要建立一個新的DSN，您可以開啓DSN系統管理員建立。請參閱**建立DSN**以了解更多細節。
3. 輸入適當的使用者名稱和密碼(如果有要求)，並點擊**Connect**按鈕。會出現一個"Connected to ..."的訊息方塊。如果在方塊中出現錯誤訊息，請諮詢您的系統管理員。

設定表格

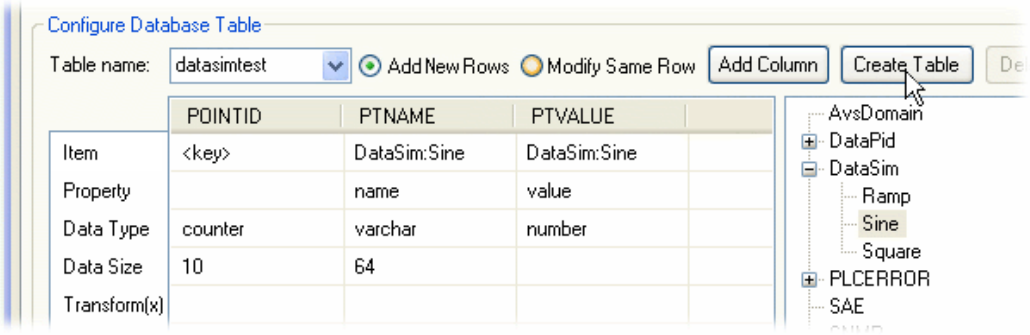
1. 選取**2. Table**頁籤



2. 如果尚未執行DataSim程式，請啓動它，並確保它已連接到DataHub。
3. 在**Table name**欄位中輸入: `datasimtest`。
4. 點擊**Add Column**按鈕，在快顯對話方塊中輸入`POINTID`，接著點擊**OK**。
5. 點擊**POINTID**標籤下方，**Item**列中的欄位。

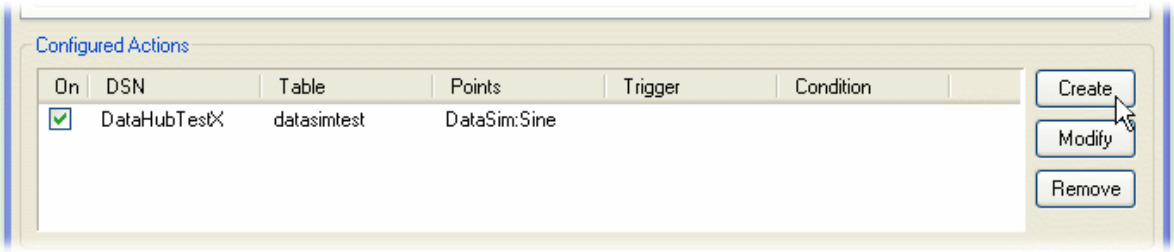


6. 在下拉式清單中選取**<key>**。請注意，在您做出選取之後，文字counter會自動輸入**Data Type**表格內。
7. 點擊**Add Column**按鈕並輸入名稱`PTNAME`。
8. 在右方的點選取清單中，展開DataSim Data Domain，並選取名爲Sine的點。
9. 在**PTNAME**欄位中，點擊**Item**列中的欄位並選取<point>。應會顯示完整的點名稱，DataSim:Sine。
   在**Property**欄位中選取name。
   在**Data Type**欄位中選取`varchar`(或是equivalent)。
   在**Data Size**欄位中，系統應該已經輸入值了。 若沒有，請輸入值如64並點擊**Enter**。
10. 再次點擊**Add Column**按鈕並新增欄位名稱`PTVALUE`。接著輸入項目:
    在**Item**欄位中選取`<point>`。(應會顯示點的名稱，`DataSim:Sine`。)
    在**Property**欄位中選取`value`.
    在**Data Type**欄位中選取`number`(或是equivalent)。
    在**Data Size**欄位中，依照您的資料庫而有所不同，您可能無法輸入任何文字。如果可以的話，輸入位元數，接著點擊Enter。
    現在，輸入欄位應該類似下圖所示:



點擊**Create Table**按鈕。如果成功的話，您現在已經在DSN指定的資料庫中建立一個新的資料表。您可以開啓資料庫程式並且檢視以驗證。如果您得到一個錯誤訊息，請仔細檢查上面的輸入項目以確保它們與您的資料庫相容。舉例來說，某些資料庫不允許資料表或欄位名稱中有空格或特殊字元。
只要資料表建立後，您無法再新增更多欄位。但是您可以使用**Delete Table**按鈕刪除資料表。這會從資料庫刪除表格，但是您所有的輸入項目將會保留在輸入欄位中。接著，您可以依照需求新增更多欄位，並重新建立資料表。您可以在欄位名稱上點擊右鍵，接著在快顯功能表中輕鬆地重新命名、插入或刪除欄位。欲了解有關建立和修改資料表的更多資訊，請參閱**章節 8.5，"建立一個資料庫Table"**。

12. 當您製作出想要的資料表時，往下至**Configured Actions**方框並點擊**Create**按鈕。

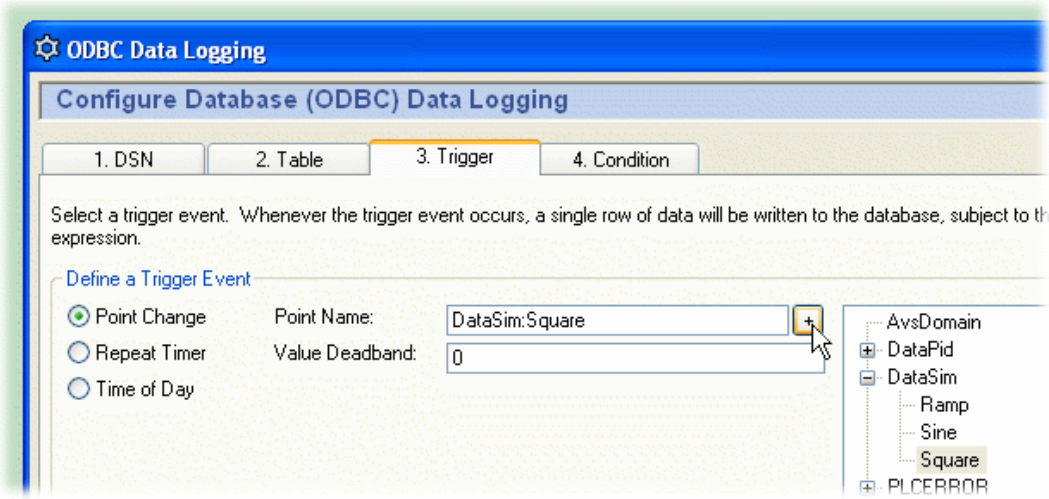


一個新的設定動作會出現在清單中。欲了解有關設定動作的更多資訊，請參閱**章節 8.9，"已設定的動作"**
下一步，欲讓DataHub寫入資料，您需要指派一個觸發程序。

指派一個觸發程序

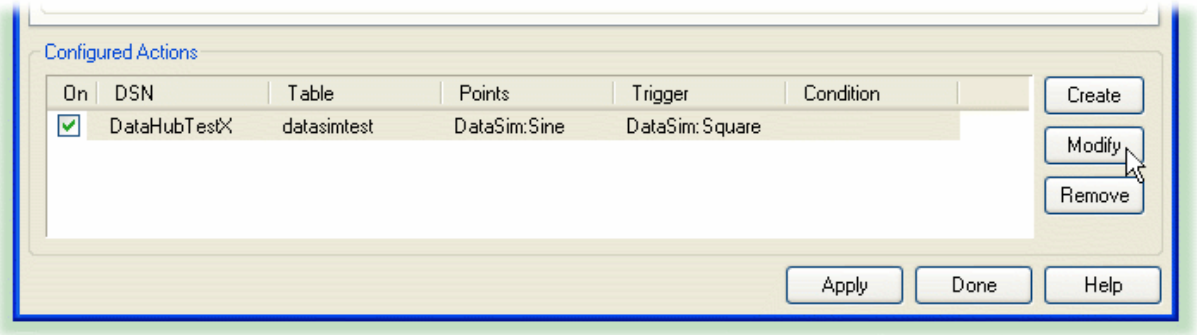
在這個範例中，無論何時，只要DataSim:Square點的值變更，我們就會觸發此行動。

1. 選取**3. Trigger**標籤。
2. 在點選取器中，展開DataSim資料網域然後選取點Square。
3. 點擊**Point Name** 欄位右邊的+按鈕。會填入點名稱`DataSim:Square`。



您可以爲該觸發程序選取任何點，包含被寫入的點，如範例中的DataSim:Sine。欲了解觸發程序的更多資訊，請參閱**章節 8.7，"指派一個觸發程序"**。

4. 在**Configured Actions**方框中，確保您剛剛建立的設定動作是反白的。如果沒有，請點擊來反白它。之後點擊**Modify**按鈕。



您的設定動作的**Trigger**欄位現在應該會顯示`DataSim:Square`點。

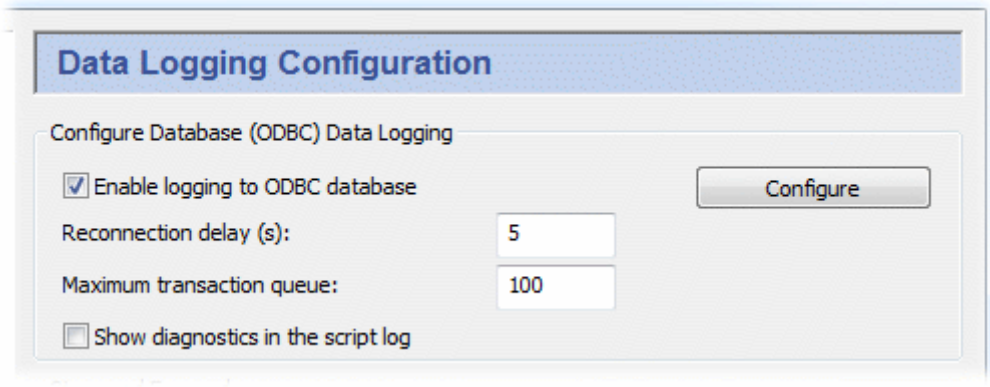
5. 點擊**Apply**按鈕以啓動該設定動作。
6. 開啓**Script Log**以檢查錯誤訊息以及確認您的資料已被成功地寫入。每個寫入資料庫的動作都會紀錄在這裡。您也可以藉由查詢資料庫本身來驗證寫入。

您剛剛已經設定一個動作，無論何時，只要點Square的值變更，就會紀錄DataSim資料網域裡點Sine的名稱和值。現在您可以建立資料表以紀錄您自己的資料，使用根據點或計時器的觸發程序。

此章節的其他小節會更詳細的解釋介面，也會介紹設定紀錄特殊**狀況**的選項，如有需求的話。

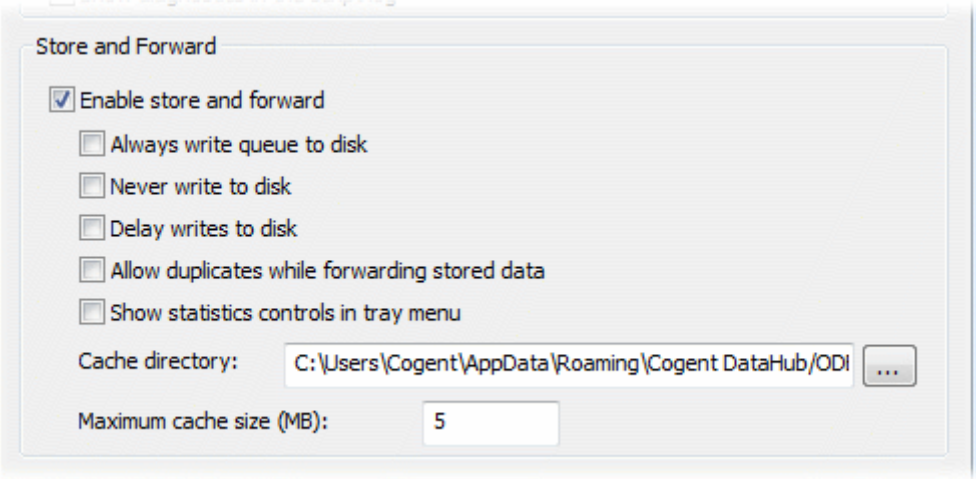
8.3. 設定Queue、Store和Forward

DataHub維護擱置作業中的記憶體內*異動佇列*。這個佇列有助於避免在忙碌期間或者是在資料庫和網路短期中斷時寫入磁碟。**Maximum transaction queue**選項可以讓您修改該佇列的深度，預設值是100則訊息。**Reconnection delay (s):**選項指定當ODBC連線中斷時，嘗試重新連線前的等候秒數。還有**Show diagnostics in the Script Log**選項 讓您檢視**Script Log**裡面的連線訊息。



儲存和轉送

*Store和Forward*詞彙是指一種資料庫連線類型，在此類型中，資料會先儲存在本機的磁碟，稍後會轉送到資料庫。Cogent DataHub執行一個store和forward的進階表單，如果資料庫沒有連線，就只能寫入到磁碟，或者是暫停。如果資料庫可供使用，資料會直接傳送到資料庫。這表示，在正常操作中，使用Store和Foward是沒有害處的。DataHub的store和forward機制使用兩個層級的磁碟快取以確保所有的資料都被紀錄，沒有任何資料遺失。



Enable store and forward
啓動store和forwarding功能。

Always write queue to disk
在異動佇列裡的資料會先被寫入磁碟快取，並從磁碟快取寫入資料庫。針對當機的最安全保護方法是勾選該方框，並取消勾選**Delay writes to disk**(如下)。

Never write queue to disk
異動佇列裡的資料只會儲存在記憶體，絕不會寫入到磁碟。

Delay writes to disk
異動佇列裡的資料會在最適當的時間寫入磁碟。針對當機的最安全保護方法就是取消勾選此方框，並勾選**Always write queue to disk**(如上)。

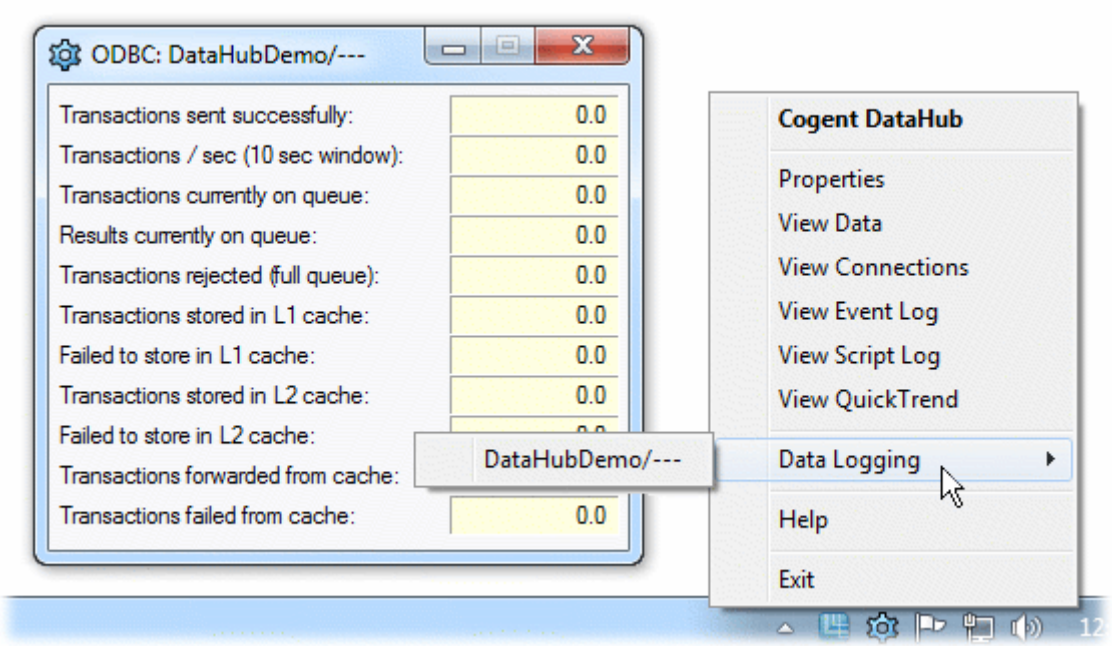
Allow duplicates while forwarding stored data

如果從快取裡傳輸資料時發生網路中斷，Cogent DataHub需要知道當重新連接時，如何處理任何已發送的資料。保持不勾選此方框 會要求Cogent DataHub隨時追蹤其快取的位置，並在每次值發送時修改資訊。這將影響每個傳輸的速度，但這會確保值不會被重複傳輸。

勾選此方框會造成DataHub在每次重新連接時，只從佇列或者是快取的最初開始，以及重複的傳輸某些資料。這大大的減少資料處理的複雜性和降低傳輸速率。如果網路頻繁的斷線而且可以接受一些重複的資料記錄，那麼這個選項特別有效。

Show statistics in tray menu

將**Data Logging**項目新增DataHub的System Tray功能表，這能讓您開啓一個統計的視窗:



Transactions sent successfully:
已傳輸的異動數量或直接傳輸到資料庫的兩者之一，或者是傳輸到磁碟快取的數量。

Transactions / sec (10 sec window):
異動的發送速率，會計算過去10秒裡的速率。

Transactions currently on queue:
佇列中的異動數量

Results currently on queue:
尚未有文件紀錄。

Transactions rejected (full queue)
因爲佇列已滿所拒絕的異動數量。

Transactions stored in L1 cache
移除佇列並將其放進第一層快取的異動數量。內部的演算法決定兩個快取中，哪個是最適合儲存指定異動的快取。

Failed to store in L1 cache
無法儲存在第一層快取的異動數量。

Transactions stored in L2 cache
移除佇列並將其放進第二層快取的異動數量。內部的演算法決定兩個快取中，哪個是最適合儲存指定異動的快取。

Failed to store in L2 cache
無法儲存在第二層快取的異動數量。

Transactions forwarded from cache
從兩個快取中轉送的異動總數量。這個數量應該是L1和L2的總合，一旦所有的傳輸都被轉送，且只要DataHub是啓動的，磁碟上就不會有任何快取。

Transactions failed from cache
在快取中，無法被成功傳輸，且儲存以供以後傳輸的嘗試異動數量。這種現象可能會發生在DataHub第一次得知此資料庫無法使用時。例如，如果您已經勾選**Always write queue to disk**，您將會在每次網路中斷時看到。

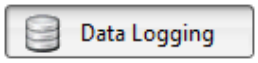
Cache directory:
快取的路徑和目錄。

Maximum cache size (MB):
快取配置的磁碟空間，以MB爲單位。

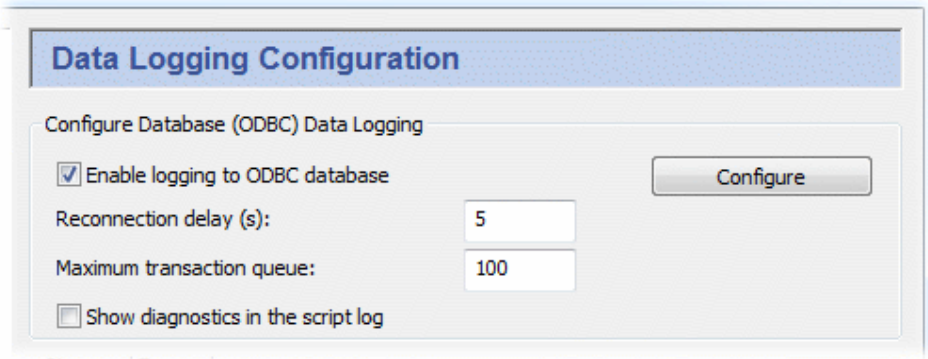
8.4. 建立DSN (Data Source Name)

DSN 是資料庫來源名稱。Windows使用這個名稱來識別您欲連接的資料庫。這個索引標籤可讓您選取一個現存的DSN或是依照需求建立一個新的DSN。

開啓ODBC Data Logging視窗

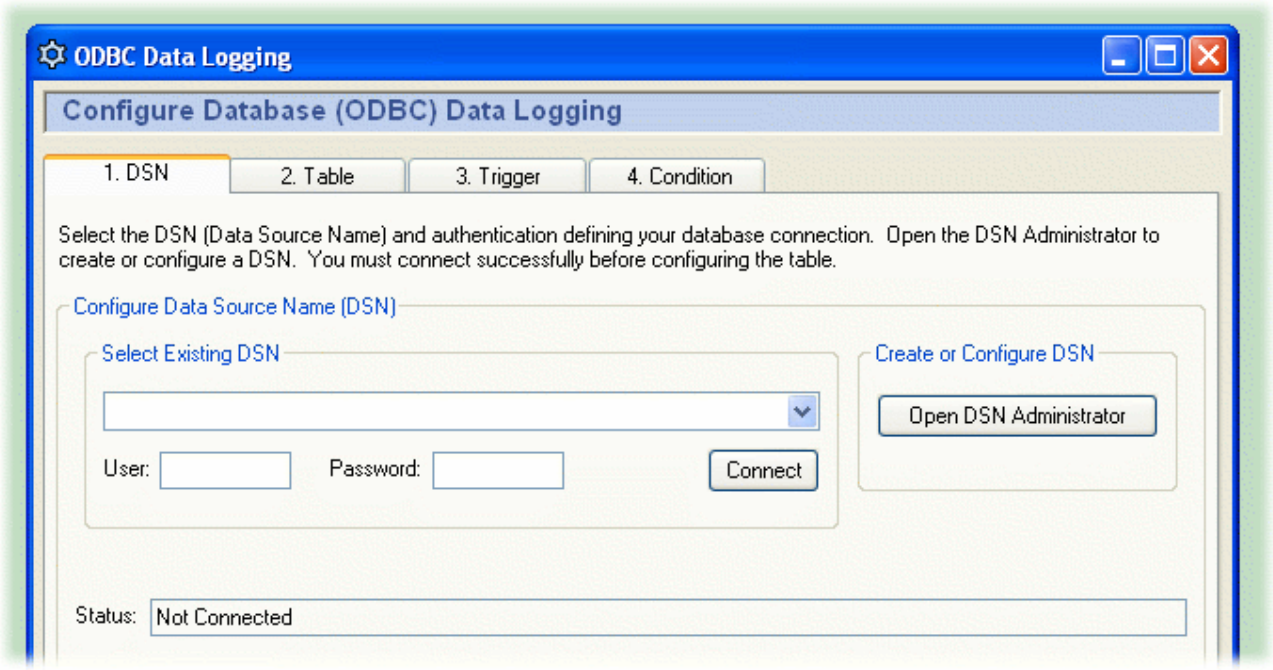
1. 在Cogent DataHub Properties視窗中，選取 **Data Logging** 

。
2. 在**Configure Database (ODBC) Data Logging**區塊中點擊**Configure**按鈕。



Data Logging Configuration介面的詳解在章節 8.3，"設定Queue、Store和Forward"。

3. 請依照下列步驟設定您的DSN。

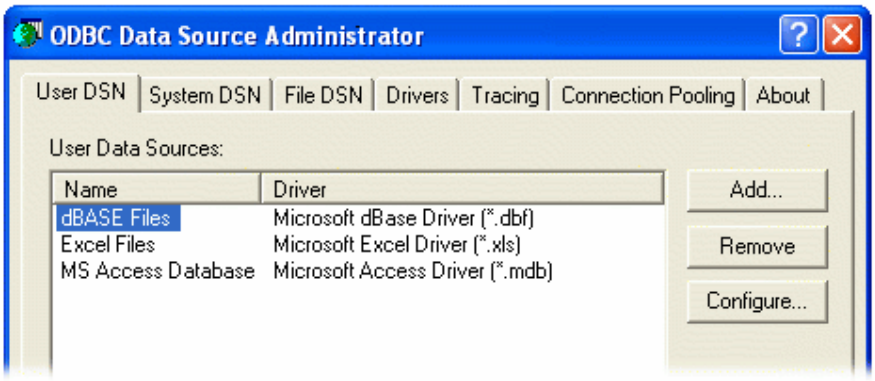


選取一個DSN

1. 欲選取一個DSN，在下拉式清單方塊中選取其一，接著輸入適當的使用者名稱和密碼。
2. 點擊**Connect**按鈕。訊息框中會出現一個"`Connected to ...`"訊息。如果您的訊息框中出現錯誤訊息，請洽詢您的系統管理者。

建立或設定一個DSN

1. 欲建立或設定一個DSN，點擊**Open DSN Administrator**按鈕。這會開啓ODBC資料來源管理員視窗。



2. 選取**User DSN**或**System DSN**頁籤，這取決於您打算如何存取您的資料庫。
   一個使用者DSN只可以用於現在的使用者帳戶，而一個系統DSN可用於此電腦中的任何使用者帳戶。
3. 現在您可以新增一個新的資料庫或是設定現有的資料庫。

新增一個新的資料庫

1. 點擊**Add**按鈕。會開啓Create New Data Source視窗， 會顯示資料來源驅動程式的清單。
2. 選取對應您ODBC資料來源的資料來源驅動程式。會開啓一個資料來源安裝視窗。每個資料來源安裝視窗都不一樣，但是您應該可以很輕鬆地找到適當的輸入欄位。
3. 輸入資料來源名稱並選取資料庫。
4. 輸入任何必需或選擇性資訊，如登錄名稱、密碼等等。需要輸入的項目和在哪個項目輸入，是根據您所使用的特定資料來源安裝視窗而有所不同。
5. 點擊**OK**以返回ODBC資料來源管理員視窗。您應該可以看到列出新資料庫和驅動程式的清單。如果您需要進行任何變更，您可以設定現有的資料庫，如下列步驟所述。

設定現有的資料庫

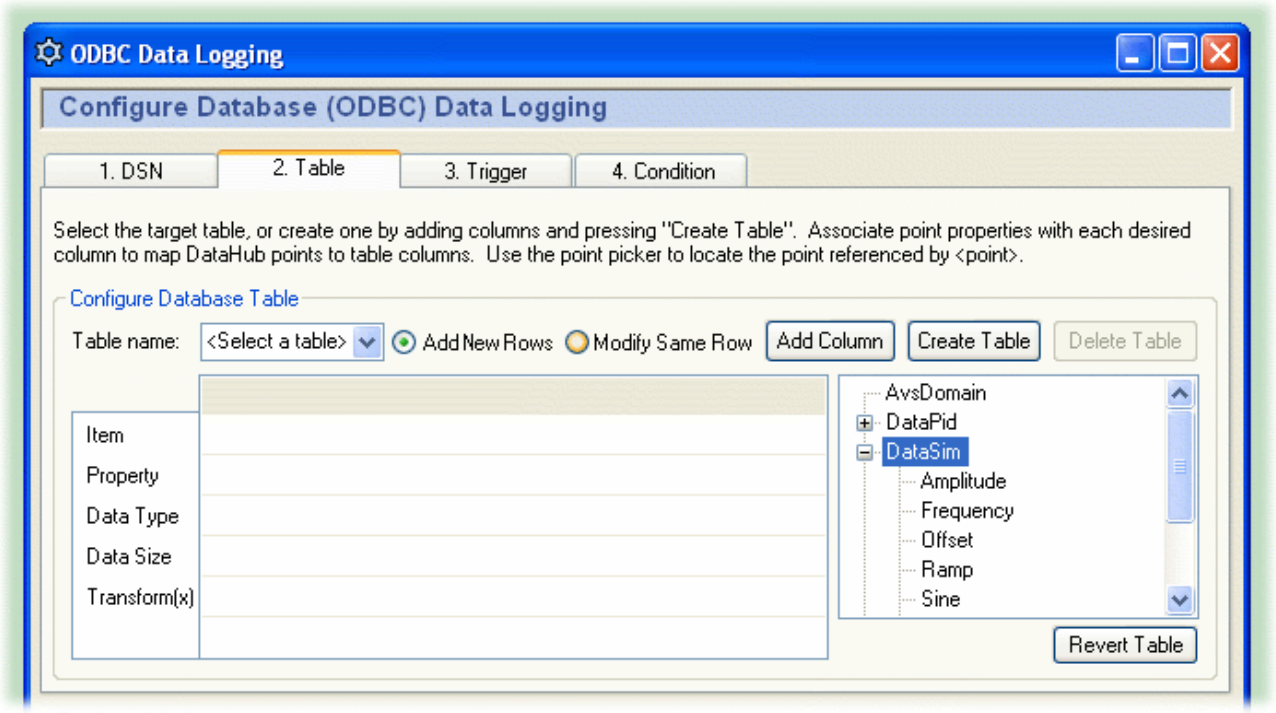
1. 選取一個資料庫來源名稱並點擊**Configure...** 按鈕。這會帶您到資料來源安裝視窗(如上所述)，在此您可以對設定做出變更。
2. 做出變更，然後點擊**OK**以返回ODBC資料來源管理員視窗。只要需要，您可以隨時進入這個視窗以便進行變更。

4. 當您滿意所有設定時，點擊**OK**按鈕以退出ODBC資料來源管理員。

只要您已經建立DSN，您就可以選取它，如上所述。

8.5. 設定資料庫Database Table

當您**建立一個DSN**後，您可以建立一個table或是選取現存的table，接著指派點和屬性給table裡面的欄位。



選取Table或建立Table

您需要選取一個現有的table，或是建立一個新的table。

- **Selecting a table** 讓您使用在資料庫程式中建立的table。當您選取一個現有的table，您無法新增欄位或變更欄位名稱。欲選取一個table，在下拉式清單中選取table名稱。
- **Creating a table** 提供在這個介面中設計和建立table的方法。

1. 在下拉式清單**Table Name**中，選取`<Create a new table>`。
2. 在對話框中，輸入table名稱並點擊**OK**。
   現在您必須新增至少一列欄位。
3. 點擊**Add Column**按鈕以建立一個欄位。
4. 在對話框中，輸入欄位名稱並點擊**OK**。
   在這個步驟中，您可以新增更多欄位，或者您可以指派點(如下所述)。您可以在欄位名稱上點擊右鍵，接著在快選功能表中選取**Rename Column**、**Insert Column**或**Delete Column**，藉以輕鬆的重新命名、插入或是刪除欄位。
5. 當您的所有欄位都已建立和命名，您可以在資料庫中建立table。一旦建立後，您無法在新增欄位到table裡。欲建立一個table，點擊**Create Table**按鈕。

   如果在建立table之後，您需要變更它，且裡面沒有您需要的資料，您可以點擊**Delete Table**按鈕。這會從資料庫中刪除table，但會完整的保留您在此視窗中做出的設定。您可以新增更多欄位或對點和屬性進行變更，以及重新建立table。您可以依照需求次數來建立。

新增新的資料列或修改相同的資料列

對於任何指定名稱，無論是現有的或是新建立的，您會需要決定您要新增新的資料行，每次新的資料變更時修改相同的資料列。

- **Add New Rows** 每次資料記錄時，在table中建立一個新的資料列。

Click here to watch a video.

Modify Same Row 每次資料紀錄時，在table中寫入相同的資料列。

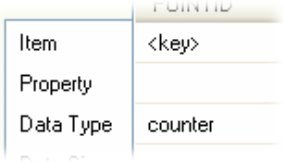
Click here to watch a video.

指派一個key

您可以選擇製作一個欄位，通常是table中的第一個欄位，一個索引欄。

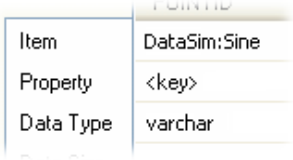
- **一個自動遞增的索引鍵**通常設定爲**新增新的資料行**。新增新的資料行時，一個索引欄是可選的，但如果您選擇擁有一個索引欄，它**必須**是自動遞增的。請參閱**章節 8.6，"索引欄"**以了解更多細節。一個自動遞增的索引鍵可以被設定，如下所述:

  Item: 在下拉式清單中選取<key>。
  Property: (無)
  Data Type: 適合資料庫的自動遞增整數類型。



- **一個單一索引欄**是**修改相同資料行**要求的。DataHub使用索引鍵的值來決定修改哪個資料行。請參閱**章節 8.6，"索引欄"** 以了解更多細節。索引鍵可以被設定如下:

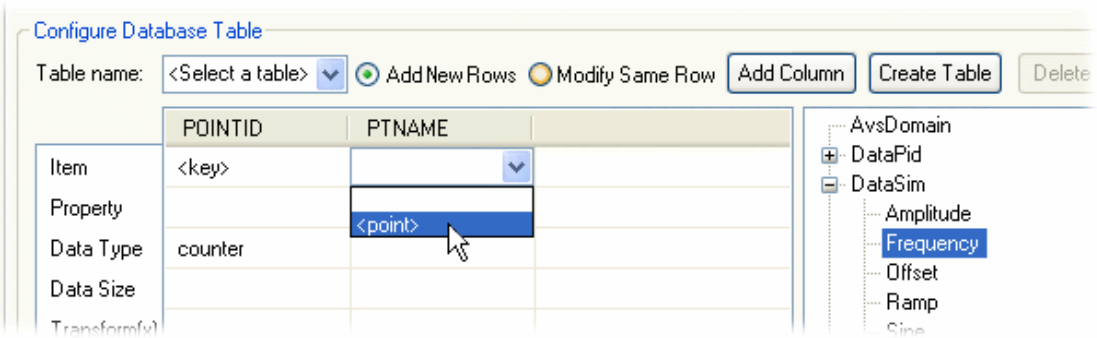
  Item: 輸入一個DataHub點名稱，或是任何其他的值。
  Property: <key>
  Data Type: 資料庫適當的類型。(通常是VARCHAR、NVARCHAR或CHAR)。



由於Windows的bug，介面不會回應第一欄，除非您直接點擊下方欄位名稱的文字。我們正在努力解決此問題。

指派點和屬性

當您選取一個table，或者您建立的table中至少有一個欄位，您可以指派一個點以及其屬性給各個欄位。



Item

首先，在右側的點選取清單中選取點。然後點擊Item列並在下拉式清單中選取<point>。或者，您可以輸入點的名稱，把這個**Item**留白可讓您選取時鐘的**Property**以顯示系統時間，或是clockms以便顯示系統時間的秒數之後的milliseconds秒數。

Property

選取您想要在此欄位中，寫入資料庫的點屬性:

`name`
　　顯示在**Item**欄位中的點名稱。
`value`
　　顯示在**Item**欄位中的點值。
`quality`
　　顯示在**Item**欄位中點的品質。

`timestamp`

點的時間戳記顯示在**Item**欄位，這會包括milliseconds，但許多資料庫如MS Access，會忽略milliseconds，只儲存秒數。其他的資料庫如MySQL和MS SQL Server在時間戳記中會包括milliseconds。例如:

- 資料庫如MySQL和MS SQL Server:

	欄位A
輸入**Property**	`timestamp`
輸入**Data Type**	`datetime` or `timestamp` or `date`

- 資料庫如MS Access:

	欄位A	欄位B
輸入**Property**	`timestamp`	`timems`
輸入**Data Type**	`datetime`	`number` or `integer`

`timems`
時間戳記的millisecond組件，通常與timestamp一起使用。如果您的資料庫無法儲存時間戳記的millisecond組件.timestamp，您才需要這個。

`timestampUTC`
和timestamp相同，但是以UTC的時間。您也可以與timems一起使用這個。

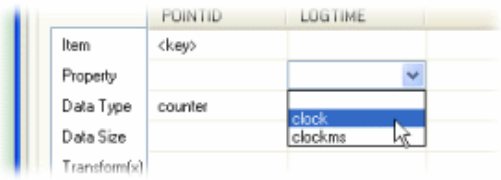
`clock`
目前的系統時間。這將包括milliseconds，但如同timestamp(如上)，許多資料庫會忽略milliseconds，只儲存秒數。

`clockms`

時鐘的millisecond組件，通常與clock一起使用。如同timems(如上)，如果您的資料庫無法儲存時鐘的milliecond組件，您才需要這個。

Using `clock` and `clockms` - 小秘訣:

- 您必須將此Item欄位保留空白以選取這些選項之一。



- **範例:** 如果時間是`12:34:56.789`， 會在接受milliseconds的資料庫中寫入時間`12:34:56.789`，然後會在不接受milliseconds的資料庫中寫入`12:34:56`。在所有的資料庫中clockms屬性會被寫入789。
- clock和clockms屬性讓您紀錄table裡的欄位時也紀錄系統時間，所以您的紀錄可以包含一個系統時間之外也能有不同點值的數量，例如:

```
System Time      Point1      Point2      Point3
08:12:56.000     43.883       3.727     213.905
```

Data Type
資料庫應與屬性相關聯的資料類型。

Data Size
在某些情況下，它會自動輸入。在其他情況下是不使用的，但有時候可能可以使用，或者是必須輸入一個大小，如文字字串中字元的數量，或是位元的數量。

Transform(x)

這會允許您修改輸入項目或是插入一個文字字串。例如:

- (x * 100) + 25可用來將點值乘以100倍並在結果中增加25。
- `"Tank Level"` 會插入字串Tank Level，而不是點名稱。

8.6. 索引欄

一個資料庫table可以有特別的欄位(或一組欄位)被指定爲*索引鍵*。每個索引欄裡的值都是獨一無二的。藉由在table中指定一個索引鍵，您能提供資料庫一個保證，就是任何索引鍵中的搜尋都會產生0或1的資料行。事實上，當搜尋符合索引欄中的值時，包含索引鍵的資料行保證是table中唯一的一個。

一個資料庫table不要求索引鍵。索引鍵是資料庫引擎的有效提示，以便提高效能和保證唯一性。您仍然可以搜尋和修改沒有索引鍵的table。

當使用 DataHub Data Logging介面時，您可以選擇指定哪個索引鍵。介面只允許單一資料行索引鍵。您將無法指定一個以上的索引欄，而且如果您用多個索引欄選取一個現有的table，介面將會失敗。

您所選取的索引鍵將取決於您資料庫table選取的是**新增新的資料列**或是**修改相同的資料列**。

新增新的資料列

如果您選擇新增(插入)新的資料列進您的資料庫table，您不需要一個索引欄。每個新的紀錄介面會建立一個新的資料列並在其填入您已設定的資料。因爲您不斷地在table中新增資料列，所以無法保證欄位中的資料是唯一的。任何點名稱、品質、值或是時間戳記都可能會重複。

因此，如果您想要在table中擁有一個關鍵欄，它必須擁有一個重要的屬性: 它必須是自動建立以及保證獨一無二的。索引鍵的值不能衍生自DataHub點。大部分的資料庫引擎支援計數器或是自動遞增的數值。如果您選取使用一個包含索引鍵的table，索引欄必須是適合您資料庫的自動遞增整數類型。

當透過Data Logging介面設定table時，您可以藉由在資料庫table的**item**輸入項目中選取<key>選項來指定一個索引鍵。如果您正在使用一個包含索引鍵的現有table，您必須在Data Logging介面中指定該欄位爲索引欄。

修改相同的資料列

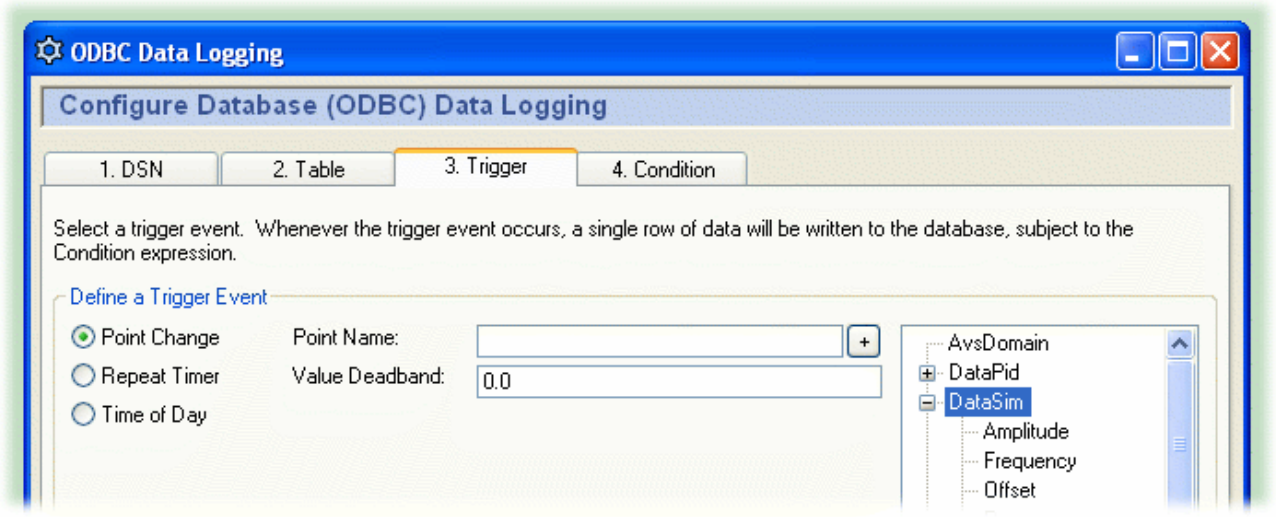
如果您選擇在資料庫中修改一個資料列，使得任何新的資料會覆寫進資料列裡的現有資料，您必須能夠唯一識別該資料列。這表示您**必須**擁有一個索引欄。

索引欄可以是任何類型，且不需要被自動遞增。因爲該資料列在新資料可用時就會覆寫，並沒有建立新的索引鍵的值。提供DataHub的點名稱作爲索引鍵，而索引鍵欄位定義爲一個字串類型是很普遍的(常見的是VARCHAR、NVARCHAR或CHAR)。您可以在樹狀的點選取器中選取一個點，接著選取<point>，選擇資料庫欄位的**Item**項目，並把**Property**欄位設爲<key>，藉以完成上述內容。

如果您想要指定一個索引鍵的值而不是一個點的名稱，輸入您自己的值到**Item**項目，而不是使用點的名稱。您接著必須在**Property**輸入項目中選取<key>。

8.7. 指定一個Trigger

Trigger是一個事件，它可以讓一排資料行寫入到您的資料庫table。一個Trigger可以是一個點值的變更、一個計時器事件或是一個日曆事件。您可以替每個資料行指派一個不同的Trigger，或是爲任意數量的資料行指派相同的Trigger。一個動作可以設定來執行在每個觸發事件，或者您也可以指派一個**觸發條件**來隨時評估觸發條件的發生，以確定是否該執行此動作。



三種類型的Trigger如下:

- **Point Change** 無論何時，只要特定的觸發點變更，就會觸發。

    1. 輸入點的名稱到**Point Name**對話框中，或在右側的樹狀資料點擊+號展開以選取點。
    2. (可選)如果您想要過濾掉多餘的資料就輸入一個值的死區。您輸入的號碼會指定一個高和低(加或減)的範圍。任何落於該範圍內的值變更不會導致引發此Trigger。正向或負向的變化大於此值會啓動此Trigger，並會讓此資料行被寫入。

    欲建立一個能自動重設的Trigger，請參閱**章節 8.8，"設定觸發條件"** 裡的**自動重設的Trigger**。

- **Repeat Timer** 循環引發，每次秒數流逝時。
- **Time of Day** 在您指定的時間裡引發。您可以輸入:

    - 一個數字，指出一個特定值。例如，seconds欄位填入0會造成只在分鐘的第0秒引發事件。
    - 一列數字，用逗號分隔。例如，在minutes欄位裡輸入0,15,30,45會指出應該在每個小時引發事件，以及在每個小時的15分、30分和45分之間引發事件。
    - 一個數量的範圍，由破折號分隔。例如在hours欄位中輸入8-18會指出應該在8 a.m.到6 p.m的每個小時中引發事件。範圍可以混合列表，如0,4,8-16,20。
    - 星號(*)指出應該要爲欄位裡的每個可能的值引發事件。例如，在seconds欄位填入*號會造成每秒引發事件。hour欄位填入*號會造成每一個小時觸發事件。

    欲定期紀錄一週裡指定日期的紀錄，請參閱**章節 8.8, "設定觸發條件"**。

    欄位的範圍是:

Year:	`1970-*`	**Hour:**	`0-23`
Month:	`1-12`	**Minute:**	`0-59`
Day:	`1-31`	**Second:**	`0-59`

    年份和月份在此是輸入不同的，至於Gamma `localtime`函式，如**時間條件**所述。

    範例:

- 輸入項目爲:

會造成每天、每個月以及每一年的8:45記錄資料行。

- 輸入項目爲:

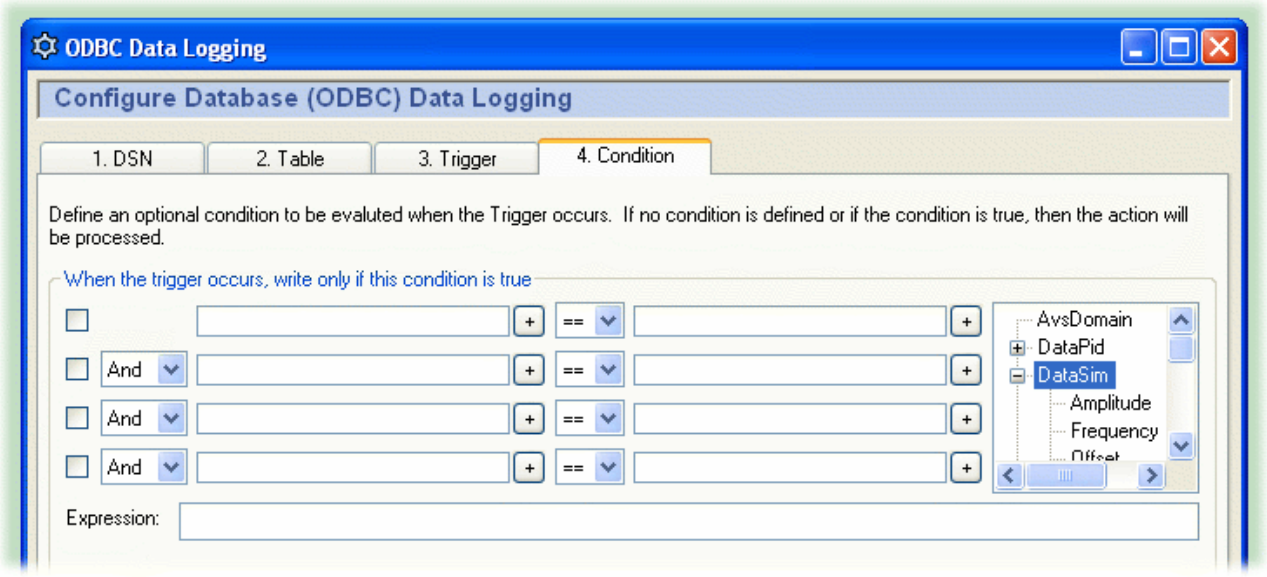
會造成每個月、每一年第15天的每個小時紀錄一次資料行。

- 輸入項目爲:

會造成8 a.m.和6 p.m.之間每兩個小時的每5分鐘引發事件。

8.8. 設定觸發條件

每個紀錄動作最多可以有4個條件以決定當引發Trigger時，是否將資料行寫入資料庫。



勾選Condition旁的方框以套用。當您製作輸入項目時，相對應的*Gamma*代碼會出現在顯示中。Gamma是DataHub的內建指令碼語言。在**Expression**方框中出現的代碼是由Gamma執行的實際代碼。"And"和"Or"運算子(&&和||)的優先順序是And，接著才是Or。

點值的條件

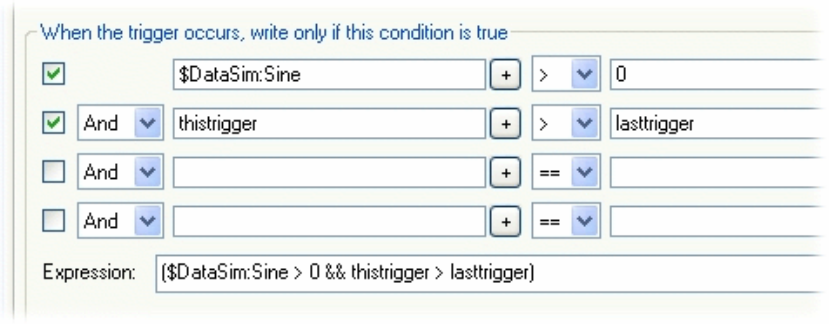
點的名稱可以被輸入比對雙方的任何一方。它們可以從樹狀資料清單中選取，或者是輸入。每個點名稱的前面需要有一個錢幣符號($)以便指示Gamma其爲DataHub點。您可以把數值放進比對的任何一方。

當您在condition欄位中輸入一個點名稱，點的現值會被用在評估。例如，您可以定義一個條件，規定只要觸發事件發生時，若其他點值在特定範圍之內，動作才會被執行。

這裡有一些可用的自動變數可與點值一起使用:

- `lasttrigger` - 該觸發程序最後一次引發後，觸發程序的點值(即使條件失敗)。
- `thistrigger` - 目前觸發程序的點值(即使條件失敗)。
- `lastevent` - 最後一次實際執行該事件之Trigger的值(條件成功)。
- `this` - 指觸發程序的點，而不是點的值。

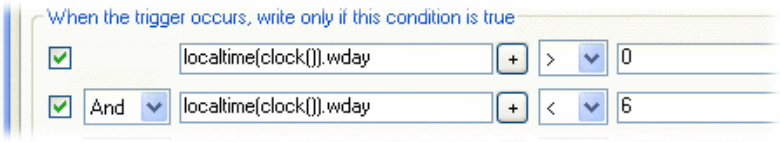
您可以在由點值變更觸發的任何條件之下使用這些變數。例如，您可以建立一些條件如下:



時間條件

當訊息被發送時，這能提供一個額外的方法來限制時間、日期、月份等等。除了在Trigger裡的選項之外，您還有day-of-week條件陳述式，根據一星期內的特定日子，它可給予您更彈性的事件。這些將能與任何類型的Trigger共同運作。

您可以使用Gamma函式**`clock`**和**`localtime`**指定一個星期裡特定的某幾天。例如，這些輸入項目:



會建立此Gamma程式碼:

```
(localtime(clock()).wday > 0 && localtime(clock()).wday < 6)
```

這會造成只在週一至週五之間紀錄資料。函式localtime返回一個類別，其成員包括有關資料的資訊，如下:

`.sec` 分鐘過後的秒數(`0 - 59`)。
`.min` 小時過後的分鐘數(`0 - 59`)。
`.hour` 午夜過後的時數(`0 - 23`)。
`.mday` 月份的天數 (`1 - 31`)。
`.mon` 從1月開始的月份數量(`0 - 11`)。
`.year` 從1900年開始的年份數量。
`.wday` 從星期日開始的天數(`0 - 6`)。
`.yday` 從1月1日開始的天數(`0 - 365`)。
`.isdst` 如果是在日光節約時間影響之下，爲`1`，如果沒有影響，爲`0`，如果資訊無法供使用，爲一負數。


年份和月份都輸入和**Time of Day**觸發條件不同的時間，如**章節 8.7， "指派一個Trigger"**所述。

這裡有兩種可用的自動變數可與點值一起使用:

- `lasteventtime` - 最後一個事件被執行的系統時鐘時間(UTC浮點數)。
- `curtime` - 目前Trigger發生的系統時鐘時間(UTC浮點數)。

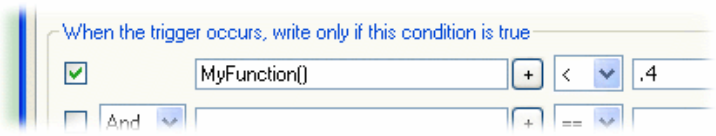
客製化條件

如果您需要滿足的條件已經超出該介面的範圍，您可以使用Gamma函式實際表達出您所需的任何條件。接著您可以插入函式到其中一個條件方框裡，並根據函式裡的傳回值設定條件。

欲做到這一點，您可以建立一個**DataHub指令碼** (`.g`檔案)其中只包含您在條件中會使用到的函式，沒有任何類別或方法。例如，以下是此類型檔案的完整內容，名爲MyCondition.g:

```
function MyFunction ()
{
    myvalue = $DataSim:Sine;
    princ("Value when the trigger fired: ", myvalue, "\n");
    myvalue;
}
```

這個函式打印出DataSim:Sine點的值，並傳回其值。我們可以從介面裡的其中一個條件方框呼叫此函式，以便使用該函式作爲一個條件，如下所示:



當Trigger引發時，會呼叫MyFunction，而且會檢查傳回值，看它是否小於.4，如果是的話，資料會被紀錄下來。

一個自動重設的Trigger

該指令碼可以將任何DataHub點變成一個自動重設的Trigger。欲使用它，首先您需要**載入並執行**TriggerFunctions.g指令碼(如下所示，並包含在安裝文件裡)。接著，如果您將此公式:

```
HighWithReset($default:TriggerPoint)!= nil
```

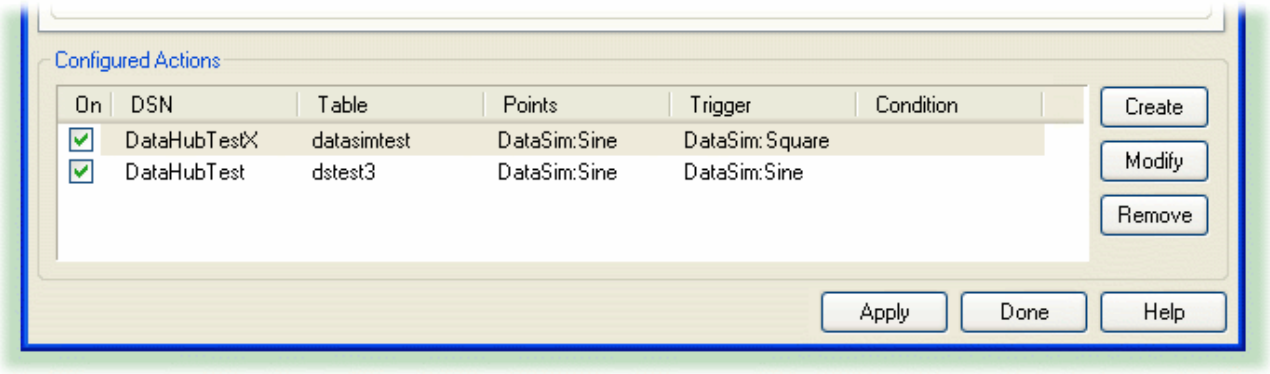
放進條件方框中，每當*TriggerPoint*在DataHub中變更爲一個非零的數字，就會引發您的Trigger。指令碼會等待一個millisecond，接著將*TriggerPoint*重設爲0。第二個函式運作方式相同，但它會在變更爲0時引發，而不是變更至非零時引發。

TriggerFunctions.g

```
/*
 * This file contains handy functions to perform more complex condition
 * handling in the Condition tab of the data logging and email interfaces.
 */

/*
 * Test a trigger point for a non-zero value.  If the point is non-zero,
 * create a delayed event to reset the point to zero, and return true,
 * indicating that the condition has succeeded and the action should proceed.
 * If the value is 0, then simply return nil indicating that the action
 * should not proceed.  We need to test for zero because when we reset the
 * trigger point to zero a second data change event will occur.
 *
 * The argument is unevaluated, so the condition should look like this:
 *    HighWithReset($default:TriggerPoint) != nil
 */

function HighWithReset(!triggerpoint)
{
    local   value;
    if (!undefined_p(value = eval(triggerpoint)) && value != 0)
    {
        after(0.001, `setq(@triggerpoint, 0));
        t;
    }
    else
    {
        nil;
    }
}

/*
 * This is the inverse of HighWithReset (see above).  If the trigger point
 * is zero, perform the action and set the trigger point to 1.  If the
 * trigger point is non-zero do nothing and return nil.
 */
function LowWithSet(!triggerpoint)
{
    local   value;
    if (!undefined_p(value = eval(triggerpoint)) && value == 0)
    {
        after(0.001, `setq(@triggerpoint, 1));
        t;
    }
    else
    {
        nil;
    }
}
```

8.9. 已設定的動作

*Configured Actions*會造成一個資料行寫入您的DSN中指定的table，根據一個Trigger以及可選的條件。這是此介面中設定動作的最後結果。Configured Actions清單顯示您所設定的動作，並允許您建立、修改或移除動作，以及將它們開啓或關閉。



設定動作的**清單**顯示您已經設定的動作。從清單中選取一個現有的動作並且自動在**DSN**、**Table**、**Trigger**和**Condition**頁籤中填入資訊。勾選或取消勾選左側**On**方框可以讓您開啓動作或關閉動作。

Create按鈕 爲目前輸入到**DSN**、**Table**、**Trigger**和**Condition**頁籤裡的資訊建立一個動作。如果您選取一個已設定動作的同時按下**Create**按鈕，它會建立一個該設定動作的複製，並將其新增至清單中。這是快速設定相同動作的方法。

Modify按鈕 使用目前輸入**DSN**、**Table**、**Trigger**和**Condition**頁籤中的資訊來覆寫選取的已設定動作。

Remove按鈕 移除一個設定的動作。

一旦建立或修改一個設定動作，變更將不會立即生效，直到您點擊**Apply**或是**Done**按鈕。每個寫入到資料庫的動作都會紀錄在DataHub的**指令碼紀錄檔**。每個寫入到資料庫的動作都會紀錄在DataHub裡。這能讓您檢查錯誤訊息並確保您的資料寫入成功。您也可以藉由查詢資料庫本身來驗證寫入。