ATI-MIRAGE
TRAINING & BUSINESS SOLUTIONS

Power BI Essentials

Reference Information

ATI-MIRAGE
TRAINING & BUSINESS SOLUTIONS
Quality•Results•Success

For details of other course programs and services provided by ATI-Mirage, please visit our website www.ati-mirage.com.au or contact us on (08) 92189059

ATI-MIRAGE
TRAINING & BUSINESS SOLUTIONS
Quality•Results•Success

About ATI-Mirage

ATI-Mirage is a wholly West Australian owned organisation. The company was formed in October 2003 by the amalgamation of Australian Training Institutes (established 1983) and Mirage Technology (established 1990). The company provides training in Computer Skills, Office and Secretarial Skills, Human Resources & Management Skills and Information Technology training to the Government and Private sectors. The company is also involved in the supply of professional training and room hire facilities to a number of organisations Australia-wide.

ATI-Mirage is committed to providing quality training to its customers. Course curricula encompass current National Training Reform Agenda recommendations of nationally recognised training governed by the Australian National Training Authority (ANTA). The credential courses are accredited through the Training Accreditation Council (TAC) and ATI-Mirage is a registered provider of these courses.

It is the philosophy of ATI-Mirage to provide FLEXIBLE, BROADLY BASED and MODULARISED training programmes for its customers, in order to ensure a maximum return on the training dollar spent coupled with optimum learning for skills transfer. ATI-Mirage firmly embraces the principles and theories of Adult Learning and incorporates these techniques in their training programmes.

Training methods used by ATI-Mirage include instructor led classroom based training, roving training delivered within the workplace, flexi-learn (self-paced supported learning) and a range of e-learning options.

If you would like to find out more about the different delivery methods available and how they could work for you, please visit our website www.ati-mirage.com.au or contact us on (08) 92189059

How to Use This Manual

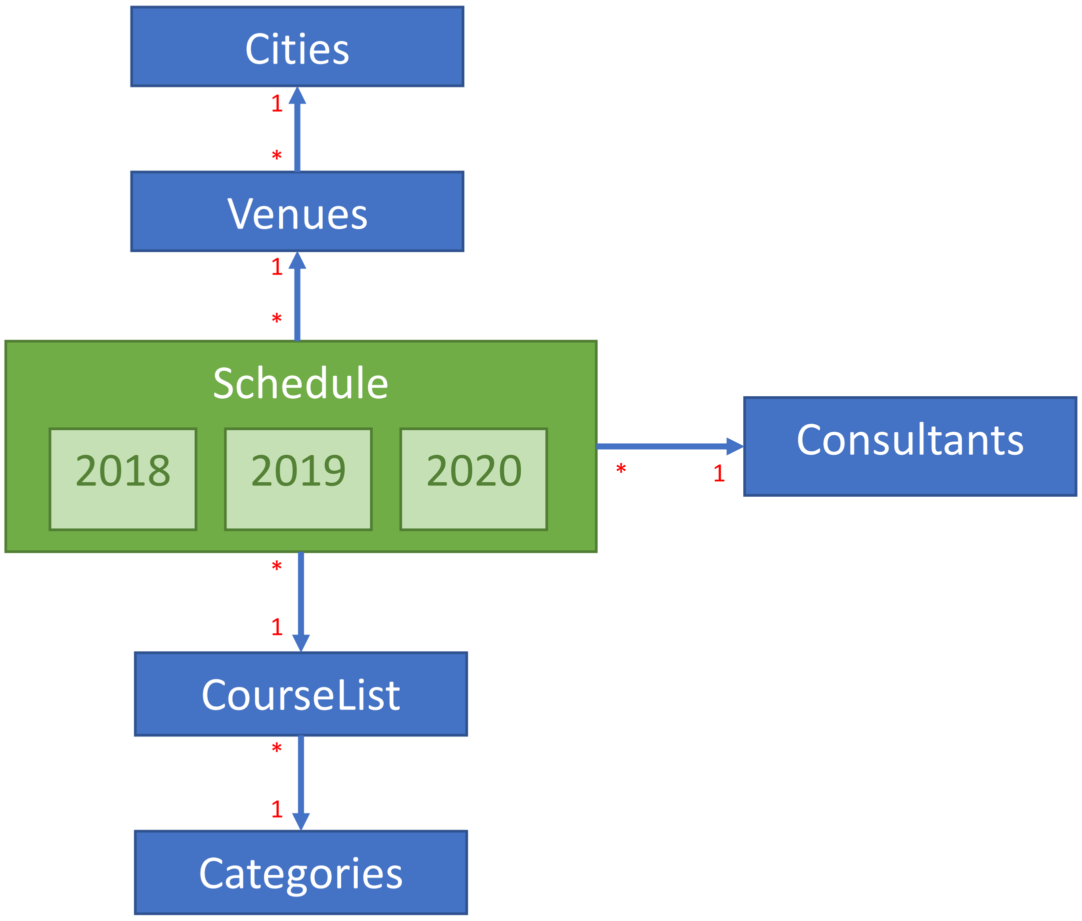
This manual is provided as a reference for use during and after the course. It has been divided into modules that broadly support the course objectives. Each module contains an explanation of the topic, some examples and exercises that support the topic. In some cases, additional topics and exercises are provided that can be completed to reinforce the outcomes of the course.

All the exercises used in this manual are available to download from the ATI-Mirage Website at http://www.ati-mirage.com.au/IT_Solutions-Course_Data_Files.htm

Storage and Analysis Models

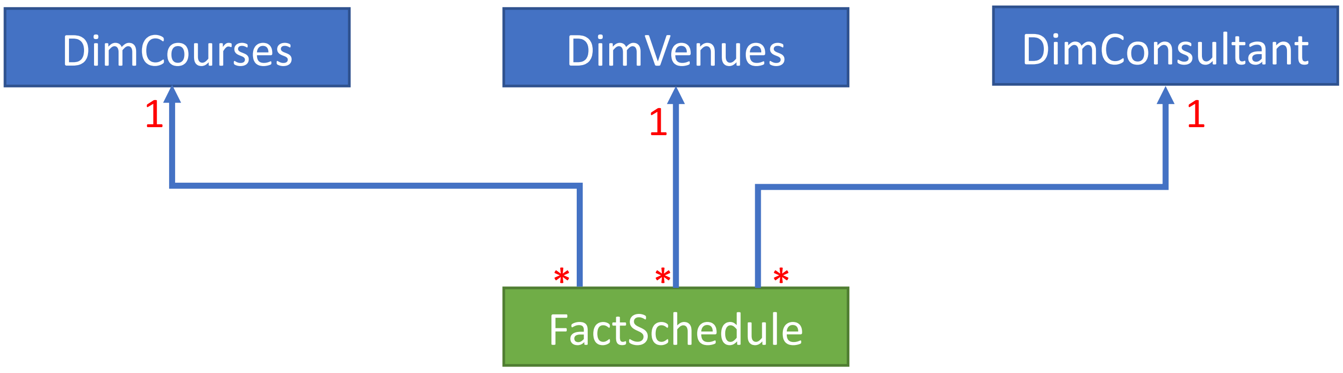
Our initial Storage Model

After data is originally stored like this:



Our desired Analysis Model

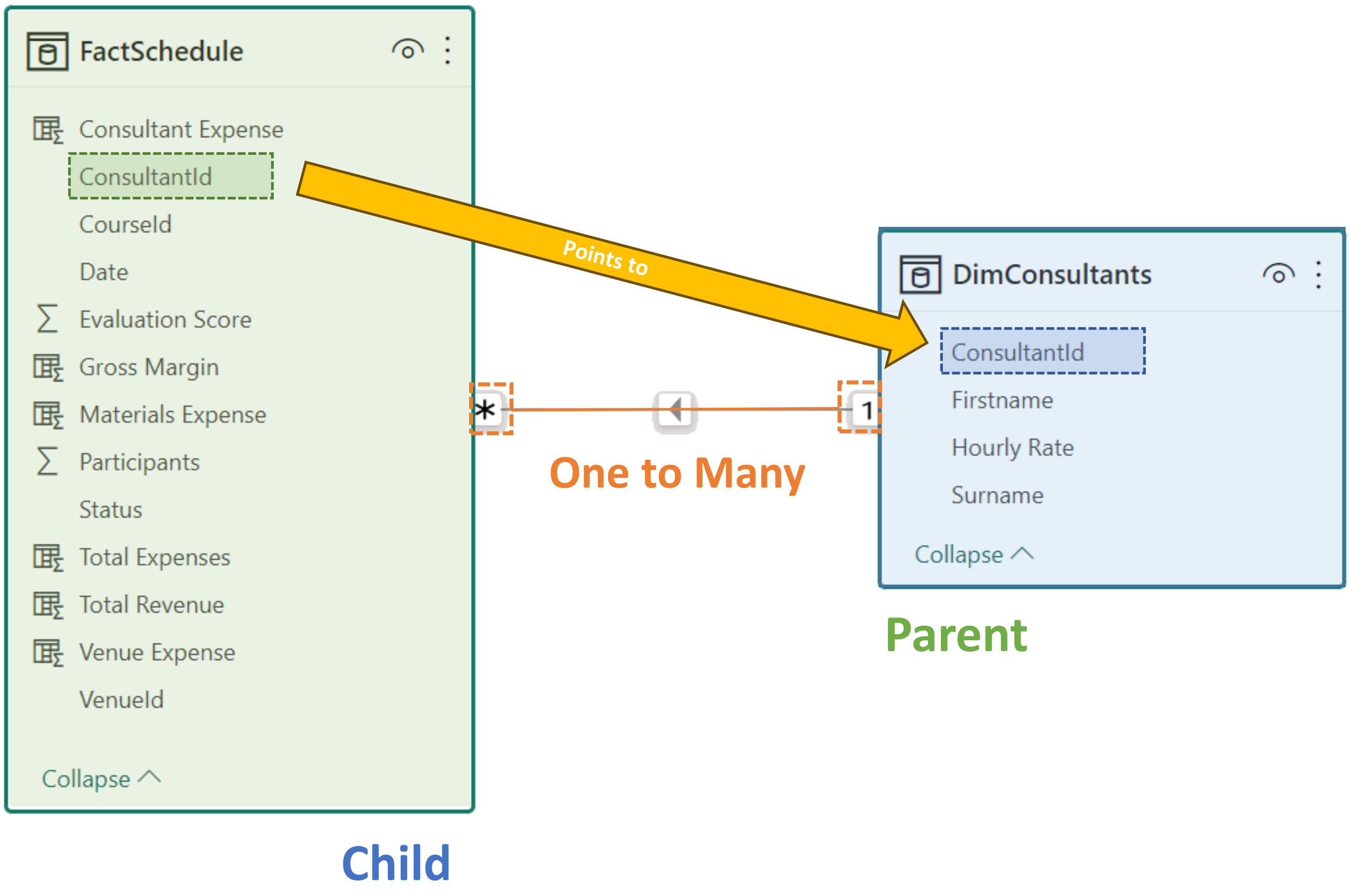
After the Power Query transformations, we want this:



Column Name Changes

Table	Old Name	New Name
FactSchedule	ConsultantEvalScore	Consultant Evaluation Score
DimCourses	MaterialsPerParticipant	Materials Cost Per Participant
DimCourses	PricePerDay	Course Price Per Day
DimCourses	VendorName	Vendor
DimVenues	StreetAddress	Street Address
DimVenues	PostCode	Post Code
DimVenues	CostPerDay	Cost Per Day

One to Many relationship Example



Process for general Transformations

1. Check the Data Type for each column in each table
   - Straight after "Change Types" step
2. Perform *Table Combinations* to simplify the Model structure
   - Appending – Same columns, more rows
   - Merging – Same rows, more columns
3. Perform tidy-up Transformations on individual tables
   - Pivot / unpivot / group table
   - Remove / Add Columns
   - Transform column data
   - Remove / Filter Rows
4. Finalise your table
   - Rename columns / fields
   - Rename query / table
5. Organise in Query groups (optional)

Process for Merging Tables

Part 1

1. Choose CHILD table (on many side)
2. Click the Merge button
3. Choose PARENT table
4. Choose the column in each which join them
5. Check for Matches at bottom
6. Click OK

Part 2

1. Expand table column
   1. No Name Prefix
   2. Leave out linking column
2. Remove linking column from child table
3. Uncheck "Enable Load" from parent table
4. Rename Child Table as Fact… or Dim… (optional)

DAX Formulas

Date Table creation

DimDates	CALENDARAUTO(12)
	
	CALENDAR(DATE(2018,01,08), DATE(2019,12,20))

Basic Columns

Year	YEAR([Date])
QuarterNum	QUARTER([Date])
QuarterName	"Qtr " & QUARTER([Date])
MonthNum	MONTH([Date])
MonthNameShort	FORMAT([Date],"MMM")
MonthNameLong	FORMAT([Date],"MMMM")
MonthYearName	FORMAT([Date],"MMM YY")
WeekdayNum	WEEKDAY([Date], 2)
DayNameShort	FORMAT([Date], "DDD")
DayNameLong	FORMAT([Date], "DDDD")

Id Columns

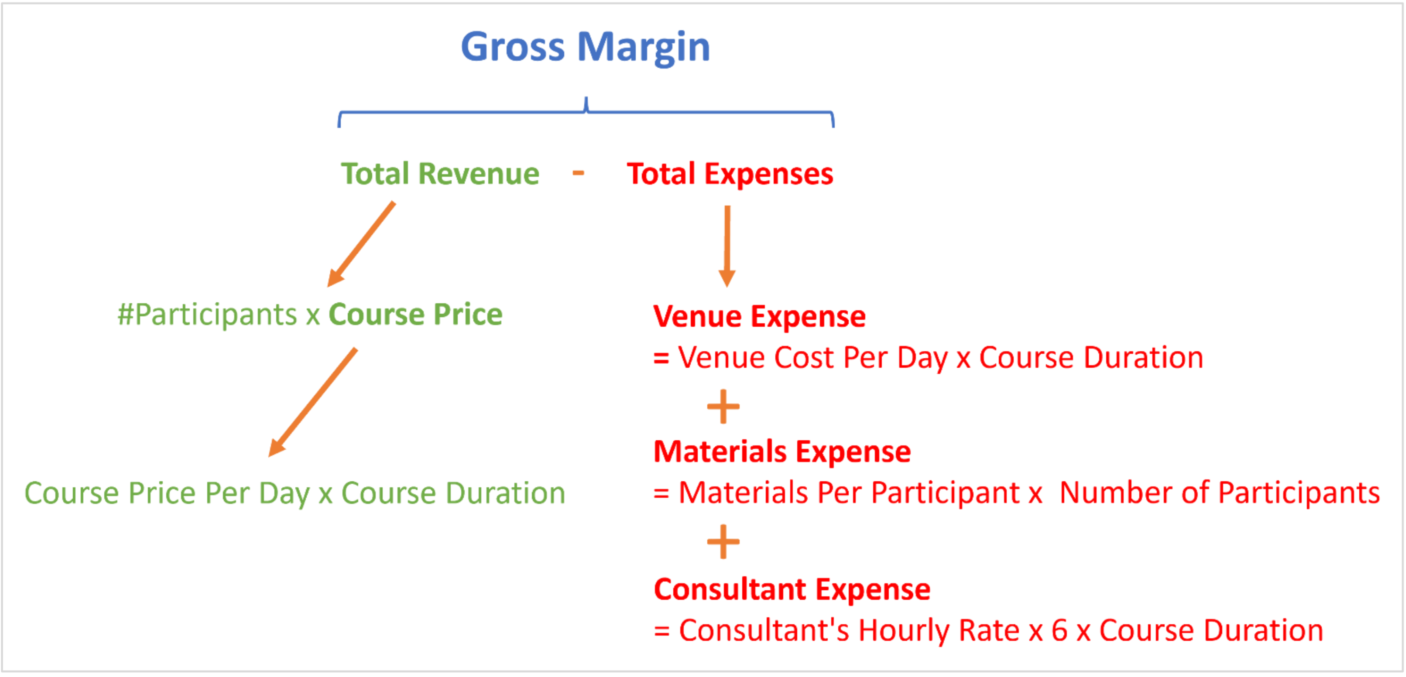
DateId	[Date] - MIN(DimDates[Date]) + 1
YearId	[Year] - MIN(DimDates[Year]) + 1
QuarterId	([YearId] -1) * 4 + [QuarterNum]
MonthId	([YearId] -1) * 12 + [MonthNum]

Conditional Columns

WeekdayWeekend	IF([WeekdayNum] <=5, "Weekday", "Weekend")
FinYear	IF([QuarterNum]>2, [Year] + 1, [Year])
FinQuarter	IF([QuarterNum]>2, [QuarterNum]-2, [QuarterNum]+2)

Gross Margin Calculations

Logic for the calculations …


Gross Margin
Total Revenue - Total Expenses
#Participants x Course Price
Course Price Per Day x Course Duration
Venue Expense
= Venue Cost Per Day x Course Duration
+
Materials Expense
= Materials Per Participant x Number of Participants
+
Consultant Expense
= Consultant's Hourly Rate x 6 x Course Duration

Calculation Formulas

DimCourse table

Course Price = [Days] * [Course Price Per Day]

FactSchedule Table

Total Revenue = [Participants] * **RELATED(**DimCourses[Course Price]**)**
Venue Expense = **RELATED(**DimVenues[Cost Per Day]**)** * **RELATED(**DimCourses[Days]**)**
Consultant Expense = **RELATED(**DimConsultants[Hourly Rate]**)** * **6** * **RELATED(**DimCourses[Days]**)**
Materials Expense = [Participants] * **RELATED(**DimCourses[Materials Cost Per Participant]**)**
Total Expenses = [Consultant Expense] + [Materials Expense] + [Venue Expense]
Gross Margin = [Total Revenue] - [Total Expenses]

Sample DAX Measures

```
Gross Margin Pct Measure =
DIVIDE(
    SUM( FactSchedule[@GrossMargin] ),
    SUM( FactSchedule[@Revenue] ),
    0
)
```

```
Gross Margin YTD =
CALCULATE(
    [Gross Margin],
    FILTER(
        ALL( DimDates ),
        DimDates[Year] = MAX( DimDates[Year] )
        && DimDates[Date] <= MAX( DimDates[Date] )
    )
)
```

```
Gross Margin MoM Ids =
SUM( FactSchedule[@GrossMargin] ) -
CALCULATE(
   SUM( FactSchedule[@GrossMargin] ),
   FILTER(
      ALL( DimDates ),
      DimDates[MonthId] = MIN( DimDates[MonthId] ) - 1
   )
)
```

Basic DAX
Function Reference

Text Functions		
BLANK [Scalar]	Returns a blank	= IF(ISERROR([TotalProductCost]/[SalesAmount]), BLANK(), [TotalProductCost]/[SalesAmount])
FORMAT [Scalar]	Converts a value to text according to the specified format	= FORMAT([StartDate],"ddd - MMM dd, yyyy")
REPLACE [Scalar]	Replaces part of a text string with a different text string	**REPLACE(<old_text>, <start_num>, <num_chars>, <new_text>)** = REPLACE('New Products'[Product Code],1,2,"OB")
SUBSTITUTE [Scalar]	Replaces existing text with new text in a text string	**SUBSTITUTE(<text>, <old_text>, <new_text>, <instance_num>)** = SUBSTITUTE([Product Code], "NW", "PA")
Date Time Functions		

CALENDAR
[Table]
	Returns a column of dates between Start Date and End Date.	DimDates = CALENDAR(Date(2019,7,1),Date(2020,6,30))
		DimDates = CALENDAR(MIN(Sales[Date]), MAX(Sales[Date]))

TODAY [Scalar] [Date]	Returns the current date	= If (MONTH(TODAY())>=MONTH([HireDate]), YEAR(TODAY()) - YEAR([HireDate]), YEAR(TODAY()) - YEAR([HireDate]) - 1)
DATE [Scalar] [Date]	Returns the specified date in datetime format	= Date(2019,7,1)
DATEVALUE [Scalar] [Date]	Converts a date in the form of text to a date in datetime format	= DateValue(“1/7/2019”)
DAY [Scalar] [Integer]	Returns the day of the month	= Day([StartDate])
MONTH [Scalar] [Integer]	Returns the month as a number from 1 (January) to 12 (December)	= Month([StartDate])
YEAR [Scalar] [Integer]	Returns the year of a date as a four-digit integer	= Year([StartDate])

WEEKDAY [Scalar] [Integer]	Returns a number from 1 to 7 identifying the day of the week of a date	= WEEKDAY([StartDate],2)
WEEKNUM [Scalar] [Integer]	Returns the week number for the given date	= WEEKNUM([StartDate])
DATEADD [Table]	Returns a column that adds (or removes) a certain number of intervals from the original date column. **Note**: The result table includes only dates that exist in the dates column.	**DATEADD(<dates>,<number_of_intervals>,<interval>)** NewDates = DATEADD('DimDates'[Date], -1, MONTH)
Logical / Informational Functions		
IF [Scalar]	Checks whether a condition provided as the first argument is met. Returns one value if the condition is TRUE and another value if the condition is FALSE	= IF(ISBLANK([MiddleName]), [FirstName] & " " & [LastName], [FirstName] & " " & [MiddleName] & " " & [LastName])
Maths / Statistical Functions		
SUM [Scalar]	Adds all the numbers in a column. Similar: Average, Count, CountA, Max, Min	= SUM('FactCourseSchedule'[GrossMargin])
SUMX [Scalar]	Iterates over the rows in a table, runs the row level context formula, and totals them.	**SUMX(<table>, <expression>)** Total Sales = SUMX(Sales, [Price]*[Quantity])
Lookup Functions		
RELATED [Scalar]	Returns a related value from another table	Estimated Weight = 'FactSale'[Quantity] * RELATED('DimStockItem'[WeightPerUnit])
LOOKUPVALUE [Scalar]	Looks a value up from a table	**LOOKUPVALUE(<result_col>, <search_col>, <search_value>…[<alternateResult>])** =LOOKUPVALUE(Product[SafetyStockLevel], [ProductName], " Mountain-400-W Silver, 46")