Dimodelo Solutions

Data Warehousing and Business Intelligence Concepts

TABLE OF CONTENTS

DATA WAREHOUSING

WHAT IS A DATA WAREHOUSE?

A Data Warehouse is to be a database that contains data integrated from multiple source systems. It exists to support reporting and analysis in the organisation. It contains historical data, and is structured in what is known as a star schema. It is optimized for high speed loading, and for responding to queries over large data sets

WHY DO YOU NEED A DATA WAREHOUSE?

The benefits of a data warehouse include:

- The ability to integrate data from multiples source to provide analysis across business domains. I.e. across financial, HR, operations, sales etc.
- End User Productivity. In any organization there is a subset of people who spend part, or all of their day producing information in one form or another. Typically they spend much of their time wrangling dirty data. One of the benefits of a data warehouse is end user productivity. All that data manipulation is already done, and the users can concentrate on analysing and responding to information, rather than producing it.
- Single version of the truth. One of the issues encountered when users report from operational systems, is the inconsistencies they can create between reports using their own queries, formulas and definitions. Meetings become about measure definitions, instead of about strategy. A data warehouse provides consistent measures, periods, rollups, ranges, KPIs etc. across the business.
- Keep historical data and do analysis of the past as it was in the past. For example, imagine a sales person who works in region A. All the sales made by the sales person roll up into the sales figures for region A. Imagine said sales person moves from region A to region B. Now suddenly, if you are reporting directly from the operational system, the sales person's past sales now roll up to the sales figures for region B. Undesirable. A Data Warehouse has methods of preventing this issue, keeping past sales associated with region A, and new sales are attributed to region B.
- Remove load from operational systems. A single analytical query can cause major performance issues for an operational system. Separating reporting and analysis load to a data warehouse removes adverse impacts on operational systems.
- Complex measures and data augmentation. A Data Warehouse provides consistent data augmentation for reporting purposes that aren't available in a source system. A data warehouse provides analytics functions like relative period (e.g. MTD, YTD etc), periodic and rolling calendars (e.g. Christmas Period, Public holidays etc.) , definition of acceptable ranges, targets, KPIs, aggregation or disaggregation of data, and periodic balances (e.g. end of month balances).
- A Data Warehouse can keep historical data beyond the normal retention period of operational systems.

WHY YOU NEED DATA WAREHOUSE AUTOMATION

Dimodelo solutions provides a Data Warehouse automation tool called Dimodelo Architect.

Traditional methods of building data warehouses have taken too long and the result have been too inflexible.

A 2006 survey by DM Review and IDC and revel these statistics:

- 17 months is the average implementation time for a BI project.
- 5 months on average to deploy the first usable BI artefact.

- $1.1m is the mean, annual expenditure on a BI project for business with greater than 1,000 employees.
- Only 31% of BI projects are recognized as successful
- A confidence rate of 36% that the right data is available to the right people.

Not a great result!

While a Data Warehouse is an extremely effective way of managing data for reporting and analysis purposes, it can be difficult to create. This is where Data Warehouse automation comes in. It makes building a Data Warehouse much easier and faster, and makes the data warehouse more flexible and capable of changing as business requirements change.

DESIGN OF A DATA WAREHOUSE

FACTS TABLES AND MEASURES

A Fact table holds rows of data containing the measures/numbers you wish to analysis. For example, a Sales fact table contains one row per invoice line item with sale amounts, discounts and other measures. A Fact table usually represents a business process or an event in a business process that you want to analyse. Fact tables are often defined by their grain. The grain of a fact table represents the most atomic level by which the facts may be analysed. The grain of a Sales fact table might be individual invoice line items.

DIMENSIONS AND ATTRIBUTES

A standard Dimension defines an entity in a business (e.g. Product, Customer etc), and groups the attributes of that entity together. A Dimension holds the attributes (i.e. fields) you want to analyse your facts by. E.g. Product Type, Product Colour etc. The attributes are used to constrain and group fact data when performing data warehousing queries. E.g. Sales Amounts Fact, where Product Colour = Silver. Other dimensions like Time and Calendar are common.

STAR SCHEMA

A Star Schema refers to the way Facts and Dimensions are related in a Data Warehouse. A Star Schema is organized around a central fact table that is joined to its dimension tables using foreign keys. The name star schema comes from the pattern formed by the entities and relationships when they are represented as an entity-relationship diagram. The fact table of a specific business activity (i.e. the Fact) is at the centre of the star schema and is surrounded by dimensional tables with data on the people, places, and things that come together to perform the business activity. These dimensional tables are the points of the star.

An example:

Suppose our company sells products to customers. Every sale is a business event that happens within our company and the fact table is used to record these events. The Star schema would look like the diagram below.

Fact table: Sales

The fact table will contain the measures of the business event (in this case Units Sold and Total Amount) and the foreign keys back to the associated Dimension members. The grain of this fact is one row per sale line item.

Sale_ID	Calendar_ID	Product_ID	Customer_ID	Region_ID	Units Sold	Total Amount
1	20130321	17	2	1234	1	$ 500.00
2	20130406	21	3	1246	2	$ 345.87
2	20130406	4	3	1246	1	$ 12378.98

Dimension table: Customers

The Customer dimension contains 1 row per customer. Each row contains the attributes of that customer. Note that the values for the attributes are verbose descriptions, rather than codes (e.g. Male instead of say 1 for Male). The reason is that these values become column headings in a report, and a column heading of 1 is not very informative. This demonstrates one of the key differences between modelling for a data warehouse and an operational system. A data warehouse contains much data redundancy.

Customer_ID	Full Name	Gender	Club Member	Marital Status	Occupation
1	Brian Edge	Male	Is Club Member	Married	Fire Fighter
2	Fred Smith	Male	Is Not Club Member	Single	Police Man
3	Sally Jones	Female	Is Club Member	Married	Lawyer

By following the links we can see that, for example, the 2nd row in the fact table records the fact that customer 3 (Sally Jones) bought two items (sale 2) on the day corresponding with the Calendar Dimension member with the Id 20130406. And, in a complete example, we would also have a product dimension table, Calendar dimension table and Region dimension table, so that we know what she bought, when and where.

The fact table contains business events that happen in our company. The dimension tables contain the factors (Customer, Time, Product) by which we want to analyze the facts.

Given this fact table and these three dimension tables, we can ask questions like: How many diamond rings (product dimension) have we sold to unmarried male customers (customer dimension) in south region (region dimension) during the first quarter of 2008 (calendar dimension)?

In other words, the difference between dimension tables and fact tables is that fact tables hold the measures we want to analyze and the dimension tables hold the information necessary to allow us to break down the measures, group them and aggregate (SUM, AVG etc) them. Dimensions hold the information we want to analyse our Facts by.

TYPES OF FACT TABLES

There are 3 main types of Fact tables. Transaction Fact Table, Periodic Snapshot fact table and Accumulating Snapshot fact table. The three types are documented below:

TRANSACTION FACT

A transaction fact represents a discrete business event that itself doesn't proceed through a series of statuses. It is often identified as a single record within a source database. It could be part of a larger business process, but this one event within the process generates its own record in the source database. For example, let's examine a simple purchasing process; first an internal purchase order is generated, then the goods are received from the supplier, then a supplier invoice is received, then a payment is made to the supplier. Each of the discrete events of this business process result in a record being created in source system, and therefore there would be a transaction fact table for each purchase orders, goods receipts, supplier invoices and payments. Other examples are metering applications, like capturing a part passing through a station in an assembly line, visits on a website etc. Transaction facts don't tend to change alot, but history of a transaction can be captured using a ledger based approach where new versions of the fact are written to the fact table superseding older versions. This requires careful handling in both the ETL and any client that consumes the data (including Cubes and Column stores).

ACCUMULATING SNAPSHOT FACT

An accumulating snapshot fact captures multiple steps of business process within the one fact table. Usually the steps represent an entity proceeding through a series of know statuses. For example, a work item could go from proposed, to approved, to in progress, to complete. Accumulating snapshot fact measures usually include the duration it takes to move between each step of the process and other measures of each step. Usually each step of the process involves updating a single existing record in the source system. The history of the fact is captured by capturing the date on which the fact changes to each status.

PERIODIC SNAPSHOT FACT

A periodic snapshot fact captures the aggregate or balance of a business process or event for a given period. Common examples are monthly financial account balances, monthly bank account balances etc. Periodic

Snapshot fact tables are usually built from the data contained in a transaction fact table. They start with an opening balance (from the previous period) tally up the transactions for the current period and produce a closing balance. However periodic snapshot fact tables may also represent aggregations (SUM, AVG etc) of a period. For example, at the end of each day, the rolling 12 month sum of Asset failure minutes. The fact is a historical snapshot as at a point in time.

SLOWLY CHANGING DIMENSIONS

Dimensions are often referred to as 'Slowly Changing Dimensions'. This describes the fact that dimensions members are relatively static but do change, albeit slowly, over time. How this change is managed in the data warehouse usually falls into 4 types. Type 1, Type 2, Type 3 and Type 6 (which is a hybrid of type 1 + 2 + 3). The most common are type 1 and type 2. Different attributes within the one dimension can have different slowly changing dimension types.

SLOWLY CHANGING DIMENSION TYPE 1 - OVERWRITE

When the Slowly changing dimension (SCD) type 1 is applied, if a change occurs to an attribute, the existing dimension member row is overwritten with the new value of the attribute. Essentially no history is kept. A good example of a Type 1 Attribute is Employee Name. If an employee name changes due to marriage, the Name should be overwritten. This has the effect of moving all history to the Employee's new name. This highlights unanticipated consequences with type 1. Let's say you have a Sales Division org unit dimension, and each Sales Division has a parent Region attribute. If the Sales Division moves to a new Region, and the Region attribute of the Sales Division is treated as type 1, then all the historical sales facts that were associated with the original version would suddenly appear in Regional reports as if they belong to the new Region, usually, an undesirable outcome. This is where Type 2 comes in.

SLOWLY CHANGING DIMENSION TYPE 2 – ADD A NEW VERSION

In the SCD type 2 scenario, a new version of the Dimension member row is written when a Type 2 Attribute changes. History is preserved. Existing facts remain associated with the old version of the dimension member and new data is associated with the new version of the dimension member. Let's say you have a Sales Division org unit dimension, and each Sales Division has a parent Region attribute. If the Sales Division moves to a new Region, and the Region attribute of the Sales Division is treated as type 2, then a new row is written to the dimension for the new version of the dimension member. All the existing sales facts that were associated with the old version would still appear in Regional reports as if they belong to the old Region, which is desirable, because this region was responsible for the sales at the time it was made. Only new facts that are associated with the Sales Division after the change will be associated with the new Region.

ETL (EXTRACT TRANSFORM LOAD)

ETL stands for Extract, Transform Load and refers to the process of extracting data from the Source systems, transforming it into the star schema format and loading it into the relational Data Warehouse. Development of an ETL process (along with the Data Warehouse itself) is the major cost in delivering a Business Intelligence Solution. Up to 80% of your cost will be in developing the Data Warehouse. Usually the ETL is run over night in a Batch process.

EXTRACT AND STAGING

The extract process pulls data from a source system, usually on a nightly basis. Source systems can include Databases, Text Files, Excel spread sheets, or any other kind of source data. The data is written to a Staging

database ready for transformation and loading into the Data Warehouse. It is necessary to stage data for a number of reasons:

- Staging places less load on source systems. Extract procedures are keep as simple as possible. The next step, transformation, may require complex queries, that you don't want to run on the source, mission critical, system.
- If you are combining data from more than 1 source system, then you need to stage data from all those systems before you can combine the data in the transformation step.
- Staging gives the Data Warehouse the opportunity to implement its own change data capture and data quality screening across source systems.
- Staging allows more rapid failure recovery, because the data does not need to be Extracted a second time on recovery.

TRANSFORMATION AND LOAD

Transformation and Load is the process of transforming the data extracted from source systems into the star schema format, and loading it into the Data Warehouse. Data staged from multiple source systems is combined. This along with the Star schema structure gives your organisation a 'single version of the truth'. There are many techniques required to identify change and implement high performance data loading into the data warehouse.

BUILD CUBE/COLUMN STORE

A Cube is defined as a Star Schema, and thus mimics the structure of the Data Warehouse closely. The process of building the cube involves simply executing a build command on the cube database. Usually the build command drops the existing cube, pulls data from the data warehouse, and rebuilds the cube from scratch. There are other build options, which involve updating the cube, but this is only recommended when the fact table(s) is very large (billion + rows).

The cube adds user defined hierarchies (like Year -> Month -> Day), Calculated Measures (like Gross Margin %, 12 Month Moving Avg etc), Relative Periods (like This Year, Last Year, Last Quarter) and perspectives to enhance the analytical and reporting nature of the data.

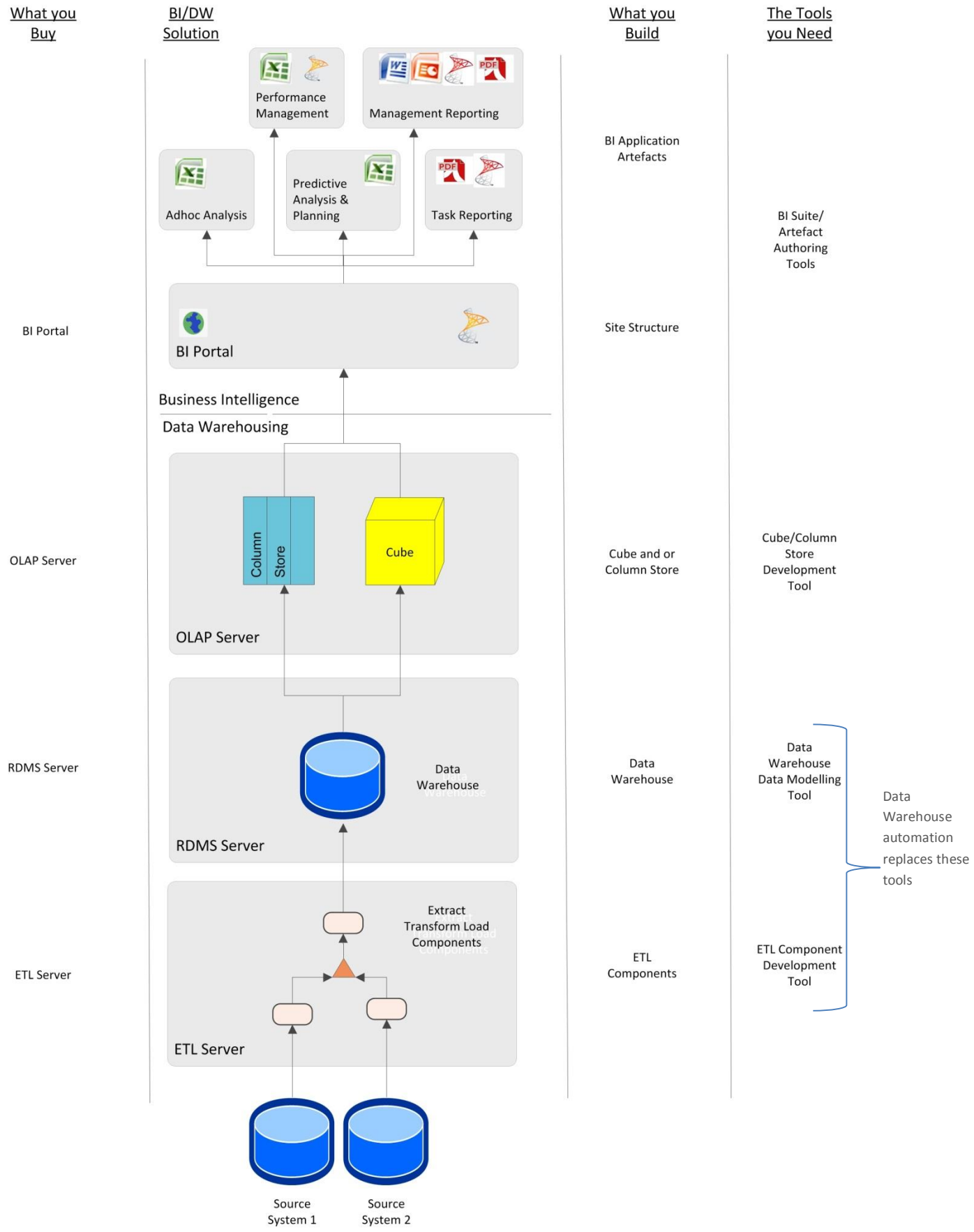
BUSINESS INTELLIGENCE AND DATA WAREHOUSING SOLUTION COMPONENTS

The diagram on the next pages shows the components of a full data warehousing and business intelligence solution. The diagram is divided in 4 streams.

1. The Component diagram.
2. What you Buy.
3. What you Build.
4. What Tools you need.

Note the distinction between Business Intelligence and Data Warehousing above and below the line.

BI Artefacts: This document refers to BI Artefacts. A BI Artefact is any Report, Dashboard, Scorecard, or Ad-hoc Analysis that is delivered as part of a BI Application. Our diagram uses the Microsoft toolset to illustrate the types of BI Artefacts you might deliver.


What you Buy
BI/DW Solution
What you Build
The Tools you Need
Performance Management
Management Reporting
BI Application Artefacts
Adhoc Analysis
Predictive Analysis & Planning
Task Reporting
BI Suite/ Artefact Authoring Tools
BI Portal
BI Portal
Site Structure
Business Intelligence
Data Warehousing
Column Store
Cube
OLAP Server
OLAP Server
Cube and or Column Store
Cube/Column Store Development Tool
RDMS Server
Data Warehouse
RDMS Server
Data Warehouse
Data Warehouse Data Modelling Tool
Extract Transform Load Components
ETL Server
ETL Server
ETL Components
ETL Component Development Tool
Data Warehouse automation replaces these tools
Source System 1
Source System 2

BUSINESS INTELLIGENCE APPLICATIONS

Business Intelligence is an umbrella term which encompasses the processes, people and technology involved in business decision making. It can be broken down into a number of distinct applications:

PERFORMANCE MANAGEMENT

Performance Management complements a business's Strategic Planning. Once the Strategic plan is defined, Performance Management is the act of monitoring progress of the business against the goals of the strategic plan, and adjusting course when necessary. Progress is defined as a set of Key Performance Indicators (KPIs).

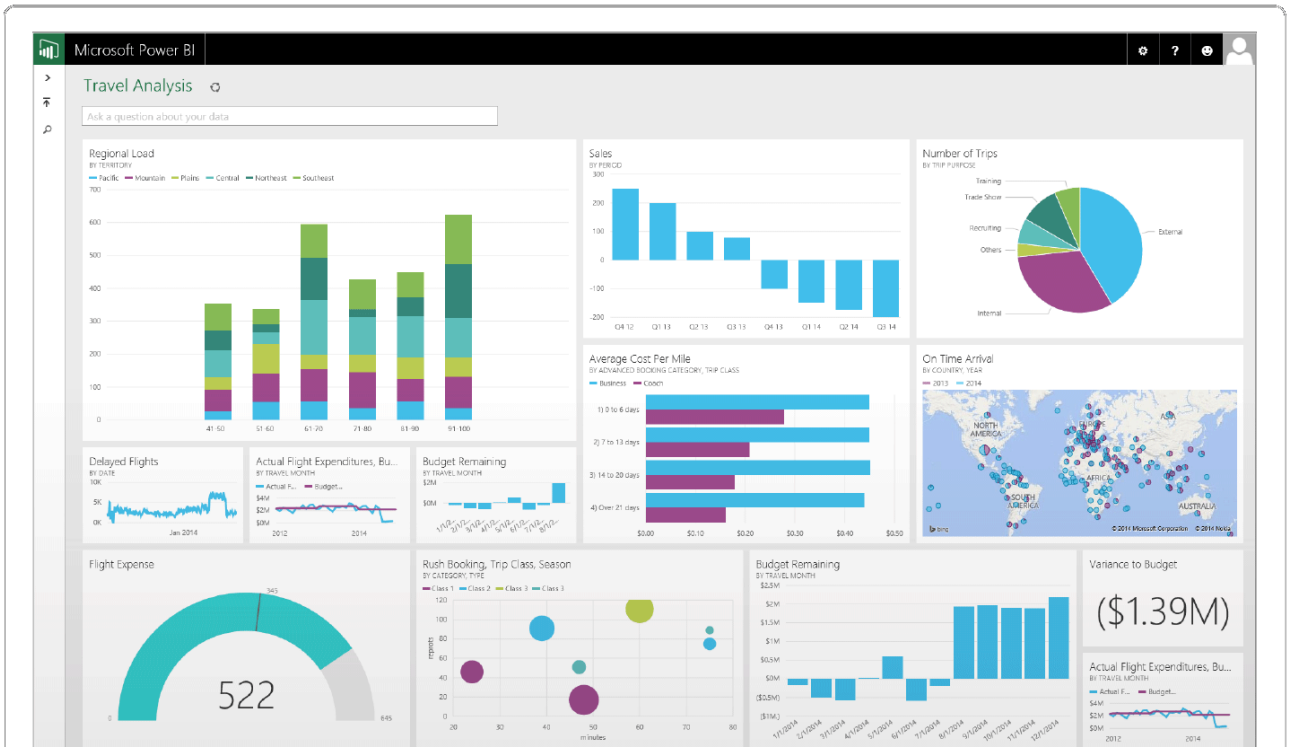
In Strategic planning, tools like Balanced Scorecards and Strategy maps are often utilized. In Performance Management visualizations of the Scorecards and/or strategy maps defined in the Strategic Plan are produced, with KPI mapped to each goal. The progress and trend of each KPI is shown, with data being sourced from the Cube.

Example Strategy Map:

Example Scorecard:

	2004			2005		
	Actual	Target	Person Responsible	Actual	Target	Person Responsible
Operational Scorecard			NORTHAMERICA\alysonp			NORTHAMERICA\alysonp
Increase Revenue			NORTHAMERICA\alysonp			NORTHAMERICA\alysonp
Sales Amt	$13,335,249	$217,975	NORTHAMERICA\alysonp		$14,668,774	NORTHAMERICA\alysonp
Sales Amt - % Growth PP	6,629.58%	10.00%	NORTHAMERICA\alysonp	-100.00%	10.00%	NORTHAMERICA\alysonp
Unit Sales	846,784	13,731	NORTHAMERICA\alysonp		931,462	NORTHAMERICA\alysonp
Unit Sales - % Growth PP	6,683.50%	10.00%	NORTHAMERICA\alysonp	-100.00%	10.00%	NORTHAMERICA\alysonp
Price Optimization			NORTHAMERICA\alysonp			NORTHAMERICA\alysonp
Avg Unit Price	$15.75	16	NORTHAMERICA\alysonp		16	NORTHAMERICA\alysonp
% Markdown	3.15%	3.00%	NORTHAMERICA\alysonp		3.00%	NORTHAMERICA\alysonp
Stores Optimization			NORTHAMERICA\alysonp			NORTHAMERICA\alysonp
Sales per Sq Ft	$6.40	$1.70	NORTHAMERICA\alysonp		$1.70	NORTHAMERICA\alysonp
Same Store Sales Growth	0.00	1	NORTHAMERICA\alysonp		1	NORTHAMERICA\alysonp
Inventory Optimization			NORTHAMERICA\alysonp			NORTHAMERICA\alysonp
Inventory Turns	18.7	24	NORTHAMERICA\alysonp	18.7	24	NORTHAMERICA\alysonp
% Unit Returns	9.6	10	NORTHAMERICA\alysonp	9.6	10	NORTHAMERICA\alysonp

Example Dashboard:



MANAGEMENT REPORTING

Business Analyst, Data Analysts and Managers of various departments/functions produce regular and uniform reports for Management. These reports often include a mix of data and commentary from the relevant Analyst.

Performance Management and Management Reporting are related. Often a Manager will have a set of KPIs for which they are responsible. A report on these KPIs would be produced on a periodic basis. Ideally these scorecards and management reports are linked, enabling 'click through' from a high level Scorecard to the detailed report.

Practically, Management Reports can take the form of:

- **A Spreadsheet**.
- **A Word document**. A Word document with embedded BI Artifacts is recommended where commentary is required.
- **A Web Based report**. Web based Management Reports can also be subscribed to by Managers, and emailed on a regular basis.
- **A PowerPoint presentation**.

or a combination of these.

AD HOC ANALYSIS

Ad Hoc Analysis is the domain of the subject matter experts, engineers and analysts within an organization. These users find fixed reports too constraining and want the flexibility to 'slice and dice' the information as they see fit, depending on the requirement at hand.

It is exceptionally difficult to define requirements for this group up-front. The reality is that their requirements change on a week to week basis.

The role of a Data Warehouse is to support the analysis effort by providing easily accessible, integrated and wide reaching information across a data domain. Coupled with powerful presentation layer analysis tools, the Analyst is equipped with the data and tools he needs to answer ad-hoc questions from management and do deep analysis of identified problems.

Typically an Analyst will use a tool (like Excel, Microsoft PowerView, Microsoft PowerPivot) to view the contents of the Cube/Column Store, and drag and drop the attributes and measures they require onto a 'report'. The tool will often allow them to visualize the report as a graph, and make other enhancements as required.

Although the output of most Ad Hoc analysis are meant to be run only once, in practice they often end up being reused and run on a regular basis. This is where the capability of uploading and managing reports in a BI portal comes into its own with analysts uploading and sharing their reports via the portal.

The ad-hoc reports within the BI Portal should be reviewed periodically for efficiencies to determine whether they continue to serve a useful business purpose, and/or are candidates for more formal reporting processes.

DATA MINING, PREDICTIVE ANALYSIS AND PLANNING

There are a number of categories of predictive analysis:

- **Planning and Forecasting**. An example of planning and forecasting is financial budgeting. Other examples are sales forecasting or production forecasting. Planning and Forecasting will involve human input, and are the basis of many KPI targets. Typically these applications are implemented using the write back features of cubes.
- **Projections**. Projections use formulas based on past performance, and other factors, to predict the future. For example, future traffic volumes on a Route. Projections are typically implemented through Calculated measures in Cubes/Column Stores.
- **Data Mining**. A technique used to find relationships in data that are not easily apparent. For example modeling the causality and impact of Traffic Jams based on incidents, Road works and traffic conditions. Again data mining requires a Cube, and advanced Presentation layer tools (like those included in Excel) to operate.

TASK BASED REPORTING

A standard set of simple parameter driven reports can be produced by the IT department to support specific tasks performed on a regular basis as part of standard business processes.

DATA WAREHOUSE COMPONENTS

RDMS SERVER

The Data Warehouse is a relational Database, and thus, must be hosted on a Relational Database Management System (RDMS). Examples of an RDMS include IBM – DB2, Oracle, Microsoft SQL Server, My SQL.

ETL SERVER

Often the ETL Server and the RDMS are the same thing. Especially if you are relying on the power of the database and SQL to implement the ETL. Depending on the technology you rely on, you may need a separate ETL server to execute the ETL Batch process.

CUBE/OLAP/MULTIDIMENSIONAL DATABASE

A Cube is a data store and high performance aggregated query engine. A Cube is able to respond with exceptional performance to queries that would normally involve large, complex and slow aggregation (Sum, Max, Min, Avg etc) operations in a RDMS system. The primary purpose of the Cube is to be the query engine for Business Intelligence Applications. The Cube relies on the Data Warehouse to persist Historical data, and can be built from scratch. The build process builds the Cube from data stored in the Data Warehouse. A Cube is defined as a Star Schema, and thus mimics the structure of the Data Warehouse closely. A Cube can also be referred to as an OLAP Database.

A cube adds user defined hierarchies (like Year -> Month -> Day), Calculated Measures (like Gross Margin %, 12 Month Moving Avg etc), Relative Periods (like This Year, Last Year, Last Quarter) and perspectives to enhance the analytical and reporting nature of the data.

COLUMN STORE/ IN MEMORY/ TABULAR DATABASE

A Column store database is, like a cube, a data store and high performance query engine. A column store database can store data in a highly compressed way. A column store database is sometimes called an 'in memory' database because the compression makes it possible to store all the data in memory. In fact it is a pre requisite of some implementations that the entire database fit into memory. Storing all data in memory, along with the way data is organised makes column store databases highly responsive. Column store databases are also referred to as tabular databases.

OLAP SERVER

OLAP is an acronym for Online analytical processing. An OLAP server host Cubes/OLAP Databases. It may also host the Column store database, depending on implementation. An example of an OLAP Server is Microsoft Analysis Services (Shipped with Microsoft SQL Server).

BI PORTAL

A BI Portal can taking many forms, depending on the software vendor you engage. Usually a BI Portal is web based. The BI Portal is a publishing mechanism for BI Artefacts. BI artefacts (e.g. Reports, Excel etc) are hosted by the BI Portal and are made accessible to end users. Usually access control can be applied to each site/folder/artefact. Users would access BI Application artefacts through the BI Portal and, if sufficient permission is granted, have the ability to upload and manage new reports they create.

ABOUT 'DIMODELO SOLUTIONS'

'Dimodelo Solutions' is a Business Intelligence and Data Warehousing Consultancy and the creator of 'Dimodelo Architect' a Data Warehouse Automation tool which makes designing and building a Data Warehouse easier and faster.

At Dimodelo Solutions, our approach is different. Using 'Dimodelo Architect' and an Agile development approach we can build a data Warehouse in 1/3 the time it would normally take. We engage with the business often, provide hands-on demonstrations and guidance as we go, and garner feedback regularly.

If you would like to speak to us about Dimodelo Architect or our Data Warehousing consultancy services please email contact@dimodelo.com.