Spotlight

Bloor

Data Discovery

A Spotlight Paper by Bloor Research
Author : Philip Howard
Publish date : September 2009

We believe that the ability to discover and understand the relationships that exist across your data, wherever it resides, is of fundamental importance to a number of IT disciplines

Philip Howard

Introduction

Data discovery or, more precisely, data relationship discovery, is of fundamental importance to a wide range of functions ranging from business intelligence through master data management to data governance and data archival. Nevertheless, it has not traditionally been treated as a market or requirement in its own right. This is largely because the tools that have been available to discover data relationships have all been data profiling tools which, in turn, have been closely associated with data quality rather than more general-purpose usage. However, that is no longer the case: there are now a number of products on the market that can discover data relationships that do not fall within the category of either data profiling or data quality. As a result, it is time to consider the importance of data discovery, and its requirements, as a market in its own right.

In this paper we will discuss what data discovery is, why it is important, what sort of functionality you should be looking for in a data discovery product and the different approaches to data discovery that are currently available.

What is data discovery?

To put it simply, data discovery is:

"the discovery of relationships between data elements, regardless of where the data is stored."

There are two basic reasons why you might want to discover these relationships: either for their own sake, so that you can ensure the integrity of these relationships, or because you want to follow these relationships from a single data element in order to retrieve related data.

In so far as this definition is concerned there are three important elements:

1. Data may be stored in a source of any type: relational or other databases, flat files, XML files, spreadsheets or whatever.

2. These data sources may be multiple and heterogeneous but you may also wish to discover relationships within a single data source.

3. Relationships may be of any type: simple or complex, direct, indirect or fuzzy. Relationships may involve more than two data elements.

We also need to address the question of why any specialised tools or procedures should be necessary in order to enable data discovery. The uninitiated, for example, might suppose that, at least within a relational database, all relationships would be defined within the database schema. There are a number of reasons why this is not so. The first of these is derived from the fact that there is no such thing as a perfect schema. When implementing a database there are any number of different schemas that you could build so that schema design is an art as much as a science and, of course, this results in some schemas being better and more efficient than others. Indeed, poor schema design can result in some relationships being omitted simply by mistake. More particularly, there is a balance to be struck between pure design and performance. For example, one of the functions of data discovery tools is to identify redundant columns. That is, where you have two or more columns in different tables in a database, where those columns contain the same data. Now, if you want a fully normalised, relationally-pure database design then these redundant columns should be removed. On the other hand, you may get much better performance in the real world by duplicating these values, because you eliminate the need to make joins between these two tables. Thus schema design is a balancing act between relational purity on the one hand and practical requirements on the other, and data relationships can get muddled in the middle. Moreover, this is just one database schema. Typically, there is no relationship between schemas for different databases: very few companies have done any top-down modelling and binding across databases and, indeed, one of the benefits of data discovery is to discover such binding relationships.

The need for performance can also have other implications that subvert schema design. A common example is that referential integrity is often turned off in production databases because of its performance overhead. This means that you cannot ensure that primary/foreign key relationships are properly maintained and can result in orphaned rows (with no relationships at all).

However, in practice, it is the definition of rules within applications as opposed to within the database that causes the biggest issues with regard to data relationships, primarily because it is so frequent. Where this is the case there are two possibilities: either you may be able to infer the presence of a business rule simply by examining data values (we have 1 million customers with various credit limits between nothing and 50,000 but no values over that) and then presenting that inference to a business analyst for confirmation; or you need the ability to discover data rules within the application software itself.

Finally, there are many environments that do not have anything approaching a schema: consider VSAM for example, or spreadsheets. There is typically even less known about these as data sources.

Why is data discovery important?

There are a variety of areas where the understanding of data relationships is important. The following is not intended to be a comprehensive list but to highlight some of the major use cases of data discovery. It is not in any particular order.

Master data management

A master data hub will typically be populated by data from multiple systems that make use of the same types of data, for example data about customers. It will frequently be the case that the same customer (for instance) will appear in multiple data sources. You therefore need to conduct overlap analysis (that is, determining where the same data appears in more than one location) to decide which sources of data are the most complete, accurate and trustworthy in order to populate your hub. This is only a very simple relationship (equality) so that you might imagine that you could do it manually and, if you only have a couple of such sources then you probably could. However, such a process will become exponentially more difficult, and time consuming, as the number of relevant sources multiplies and therefore some sort of automated process will be a significant boon.

Moreover, while overlap analysis is one hurdle to be overcome, it is not the only one for master data management. Once you know where your overlaps occur you then want to be able to determine matching keys to join these sources together and, further, you need to determine precedence (typically based on statistical analysis) so that you know which source to trust when two sources conflict.

Similar principles will also apply after mergers and acquisitions, when the new company needs to merge details from the former companies to create a single customer database, for example.

Data migration

When migrating data (whether upgrading from one version of an ERP package to another or migrating from a legacy database to a modern one or anything similar) there are two potential approaches: either you migrate database tables or you migrate business entities. We will discuss the distinction between these two later. In either case it is vital to keep data relationships intact. As a simple example, if you migrate your customer table and your sales order table and lose the link (relationship) between these, then your new system will not work as you will no longer know which orders belong to which customers. If all relevant relationships are explicitly defined within the database then this should not be a major issue but it may be if you have data rules defined within the application code instead of the database. Applications based on non-relational databases where relationships are implicit may also require special handling. The danger, of course, is that you assume that all rules are explicitly defined within the database when they are not. In all cases the migration process requires an understanding of the relationships that exist between different data elements and then the ability to ensure that these relationships are maintained as you move the data.

Note that if you are consolidating data as a part of the migration process then you will also run into the sort of issues discussed under the master data management heading.

Legacy migration

There are additional migration issues to consider if you are migrating from a legacy environment such as IMS or VSAM to, say, DB2 or Oracle. Two, in particular, relate to the COBOL redefine statement on the one hand, and database comparisons on the other. In the case of the former the issue is that you can have a field in a database that has a different meaning depending on the use of the redefine function. For example, you might have a statement that effectively says "if the first digit is a '1' then this field is a customer, if it is a '2' then it is ... " and so on. To give you an idea of the scope of the problem we know of one site that had 74 redefinitions of a single field. Now, if you want to discover the data relationships that pertain to that field you need to have some seriously clever understanding of the COBOL involved because you could have 74 different sets of relationships: so you need a tool that understands application code as well as data and metadata.

The second complication is with database comparisons. It is good practice to compare your old implementation and new implementation using comparison tools that allow you to visually compare the old and new system to check that the data has been correctly migrated. There are several such products on the market. However, these work on the basis that you are migrating from one relational environment to another. If you want to make similar

Why is data discovery important?

comparisons when migrating from a legacy environment then you need some more sophisticated capabilities that take into account that difference.

Data archival

The same principles apply to data archival as to data migration with respect to the fact that you have a choice between working with tables or business entities but, again, it is ensuring that relationships are maintained intact during the archival process that is of primary importance. Note that a particular consideration with respect to archival is that you may still want to access the archived data using the original application(s) if the archive is in near-line data storage, for example. Thus there is a direct parallel with data migration.

Sensitive data

There is also the issue of discovering sensitive data, such as personally-identifying information, credit card numbers or other confidential data contained in a database or data source. Most organisations recognise the need to mask or de-identify such data for purposes of compliance and good stewardship. Note that there are two aspects to this: in the first place discovering the sensitive data (either standalone, or embedded within text or comment fields or other unexpected places) and in the second, what happens when you actually mask an identifier for security reasons. In the latter case you get similar issues as for data archival. That is, you need to maintain the relationship to sub-tables. Thus, if you use a social security number as a primary/foreign key and you don't de-identify both the parent and child columns consistently you can orphan all the data in the child table.

Data warehousing

While there are tools and vendors that specifically support the movement of business entities rather than tables for data migration and archival, we are not aware of any companies supporting such techniques in the ETL (extract, transform and load) space for loading data warehouses, and it is probably not necessary. Of course you still need to understand, work with and maintain relationships but ETL tools typically include facilities for such things as shunting orphaned records into reject files (though we would argue that is a case of cure rather than prevention) and provide support for validation mappings to check the preservation of relationships, or you can use specialised validation tools.

Nevertheless, there are also other issues with data warehousing. For instance, depending on your warehouse architecture (for example, if you want to implement a single enterprise data warehouse as opposed to using data marts) you can end up with similar issues as for master data management. Then there is the issue of conformed dimensions (where data attributes are replicated across data marts and warehouses). First, you have to establish which attributes are going to be conformed and, secondly, you need to be able to check that conformity is maintained, as unconformed dimensions imply that the relevant data relationships have been broken.

Modelling

There are two sides to modelling: data modelling and modelling to support application development. Data modelling is the technique used to design database schemas, either from scratch or on the basis of reverse engineering from an existing legacy database to be used as the basis for the new design. However, although it is called data modelling, in practice such tools work with metadata rather than data. As we have already discussed, this implies limitations and it is therefore useful for data modellers to compare their metadata with the combined data and metadata analysis that data discovery tools provide, in order to provide a holistic view. In addition, there is the potential for appropriate discovery tools to inform and enrich the reverse engineering process so that a more complete model results.

However, we do not need to confine ourselves to data modelling: if you are developing application software against an existing database or if you are re-developing an existing application then, again, it may well be appropriate to be able to understand discovered relationships within the context of your application development environment. This will certainly be true in the case of re-developments where you will want to make use of, and understand, the existing data and its structures. This will probably be easiest to represent if the development environment is model-driven because then there will be an appropriate mechanism for viewing these relationships.

Why is data discovery important?

Data integration

Just as data discovery can be used to inform data modelling it can also act as a pre-cursor to data integration processes. In particular, the relationships you discover can be used to form the basis of the transformations that are required for ETL and other data integration tasks, thereby saving significant time and effort in the design of those processes.

Data quality

The discovery of relationships as a function of data profiling is well-established as a part of the whole process of ensuring data quality. However, data profiling also collects statistics about the validity and correctness of existing data, which can be used both in assisting with data cleansing and in on-going monitoring of data quality. As we shall see data profiling is only one of a number of ways of discovering data relationships but as the statistical analysis inherent in data profiling is also important for data quality purposes, it would be data profiling tools we would recommend for data quality purposes rather than any other approach. Note that the same thing would apply if you were concerned with data quality in conjunction with other requirements such as data warehousing or master data management.

Data governance

In a sense, data governance covers all of the subjects just discussed. However, data stewards need to have an understanding of the data in a more general sense and this, again, will be facilitated by the discovery of the ways in which different data elements are related to one another. Moreover, there is the whole question of compliance and periodic regulatory reporting, which will typically require an understanding of data relationships. Arguably we could include a separate heading for this but it certainly falls within the scope of data governance.

A proactive approach to data governance would be to be able to prove that there is no data hidden away in unexpected places, such as email addresses or postal codes in a comments field, or credit card numbers in delivery instructions. This type of analysis could be based on example values, format or domains.

Business intelligence

If you think about it, the foundation of all business intelligence is an understanding of data relationships. You cannot query product sales by store, for example, unless you have details about which products are sold by which stores. Of course, you should not have to discover such relationships but the premise remains valid. At the opposite extreme, data mining is precisely about discovering relationships in areas such as market basket analysis, fraud detection, correlations and the like. However, there is also, potentially, a whole class of relationships that fall somewhere between these two. For example, suppose that you are a salesman responsible for a particular customer account and you want to find out why the latest invoice has not been paid: you might want to access customer, invoicing, product and service details before calling the customer. However, if these details reside in different locations there is a fair chance that no standard query exists. This can be resolved using data discovery because there are tools that can find relationships between customer, product and salesman, for example, across heterogeneous data sources. Note that this sort of discovery is operating at a higher level (in this case) than the other environments we have discussed: at the level of a particular business entity.

Technical versus business level discovery

It is important to appreciate that different types of project will require different levels of discovery. For example, data migration or archival may well warrant activity at the level of business entities rather than at the table level, and business intelligence applications will certainly require understanding at this level as well as the retrieval of actual data values. Tools that facilitate collaboration between business analysts on the one hand, and the IT department on the other, will be useful here as will business glossaries and other techniques that allow these different constituencies to understand the data and its relationships in their own terms.

The reason for suggesting that you might want to use business entities for data migration (and archival) purposes is that this is, essentially, a business issue. Not only it is driven by business requirements it is the business (or the relevant domain experts and business analysts) that understands the data best. It is therefore the business that is best placed to manage the migration process, at least in terms of understanding the relationships that (should) exist. This mandates less of a technical approach to data migration and a more business-oriented one. This can be done by migrating business entities rather than IT constructs such as tables. In other words, rather than moving the data table by table you move it customer by customer (say). Moreover, when we say customer we mean a customer with his legal address, his billing address, his delivery addresses, his orders, his invoices, his service history, his credit rating and everything else that pertains to that customer. In other words, you migrate the customer plus all the other data that is related to him. Of course, this will be easier if all this information is within an integrated suite of applications (such as ERP) than if it is in disparate applications.

It is important to emphasise the significance of this. If you are an IT developer looking at a table and its relationships how do you spot that anything is missing? If something is not there that should be there, how do you know? Since tables are purely artificial constructs then the answer is that you can't. On the other hand, imagine that you are a domain expert considering customers and their attributes (relationships): now it should be relatively easy, or at least possible, to spot where things are missing. In other words, working at a business level is much more likely to be comprehensive and the more comprehensive you are in the understanding of relationships and their maintenance the less likely you are to have problems with relationships that get broken during the migration process. Note that Bloor Research is not alone in suggesting this business-centric approach: it is recommended by SAP, for example, when migrating from one version of its applications to another.

Unfortunately, working at the business entity level is not as easy it sounds. It is likely that some of the relevant detail about customers is owned by different departments: for example, the sales, marketing and financial departments are all likely to own data pertinent to customers. Furthermore, some of this data will be shared, and establishing who actually has ownership of it may be more of a political exercise than anything else.

However, the main point at issue is that you need to determine what your business entities consist of. While there is likely to always be some manual definition of pertinent business rules that describe business entities, it will clearly be beneficial to discover as many of these rules as possible, in an automated fashion. It will also be useful if there are appropriate visualisation techniques available to allow business analysts to see how different sub-entities inter-relate.

Functionality

Apart from commonplace requirements like scalability, performance and ease of use, there are really three types of functional consideration that you need to think about when looking at tools that might help discover relationships: what they do and where and how they do it. We shall consider each of these in turn. Note, again, that we are not attempting to give a comprehensive view of these areas but a general understanding of the sorts of capabilities to look for. In addition, bear in mind that not all of the functionality detailed will be necessary for all implementations: it will depend on what you want to do and the environment in which you are doing it.

What you can do

Before we consider the major functions that you might like to have it is worth pointing out the distinction between validation and discovery, because tools in the data discovery market will typically, or should, do both. In the first case you know, or think you know, what your business rule is and you want to check its validity and discover any exceptions; in the second case you want to discover if there are relationships that you don't know about. While neither of these easy, the latter is significantly more onerous and sophisticated.

The major things that you may like to be able to discover and/or support are as follows:

- Primary and foreign keys. You would like to know whether the existing primary/foreign key pairs are valid, consistent and complete; and you would like to know whether there are any broken pairs that create orphan rows. In addition, you would like to be able to scan for primary/foreign key relationships that are not explicitly defined in the database schema but which analysis of the data (rather than metadata) suggests the existence of. Such implicit relationships usually come about because of business rules defined at the application level. Where these are discovered you would also like the software to suggest potential candidates for new primary and foreign keys. Facilities to provide such capabilities are variously known as key, join or dependency analysis, though the last of these is often taken to apply across columns within a table rather than across tables.

- Matching keys. This is the ability to discover keys that can be used across data sources in order to join the relevant data.

- Column redundancy. As we have noted there may be valid performance reasons for supporting redundant columns but you would still like to know about them to ensure that they are synchronised. It is also possible that the redundancy is either inadvertent or there for historic reasons that no longer apply.

- Overlap analysis. This applies when you are looking for overlaps between data in different sources, especially as a precursor to master data management or consolidation projects, where you need to determine the most trusted sources of data for your new system. Precedence discovery will be useful here as will visual comparison capabilities.

- Business rule discovery. Here we are talking about the sort of data rules that are defined within applications and do not match to metadata, such as the range check for credit limits discussed previously. Given a sufficiently large dataset to process, tools should be able to infer and highlight the likely existence of such a rule.

- Transformation rule discovery. Transformation rules occur when relationships are transformed in some way, such as concatenation of two fields into a single one. For example, one data source might have product ID and product name as separate fields but a second source have them combined. Data discovery tools should be able to recognise such transformations. Note that transformations may be very complex and may include either business or technical rules or both.

- Rule exception discovery. You would like to be able to discover exceptions to your transformation and business rules, either based on pre-defined or discovered rules. Outlier analysis (of exceptions) may also be beneficial.

- Data usage. It will often be the case, particularly when pursuing data quality initiatives, that you will need to prioritise your efforts. In order to do this efficiently it will be useful to know which applications access particular pieces of data and how often. It will also be useful for possible chargeback purposes, or because there is a sponsoring department, if you know which users (at a departmental level) are using those applications. Further, if you have the ability to monitor

Functionality

user behaviour then you can identify when users join particular tables on a regular basis, thereby suggesting relationships that you may not have been aware of.

- Semantic awareness. It will be useful if you can determine the semantic type of the data in each column. For example, recognise that a particular column holds zip codes or city names or years.

- Context-free discovery. Understanding synonyms and near-synonyms would also be useful: cust_no (in a COBOL copybook, say) and custnum (in a database) being obvious examples. Similarly, credit card holder name is rarely the same as first name and surname but the similarities involved can help to identify unexpected relationships.

- Data classification. This is the ability to recognise the relationship between a pre-defined, business-user-maintained domain of values and the actual content of a field in order to automatically identify the content of a field, and thus to identify unexpected values.

- Business glossary and collaboration. It will frequently be the case that business and IT users need to work together on data discovery projects and tools that enable this will be beneficial.

Where you can do it

Data discovery within a single database may be all that you need if you are conducting a simple migration but in most cases you will require support for multiple, typically heterogeneous, sources. While support for relational databases is the baseline you would also like to be able to support other types of databases as well as flat files, spreadsheets, XML and so on, either directly or via JDBC (ODBC is not as versatile). Moreover, you would like to be able to discover relationships that span all data sources. In the case of XML some vendors support this directly but there are third party tools that will flatten XML so that this can be treated as if it were in a flat file. Thus all suppliers that support flat files can, in one way or another, support XML.

The other aspect here is the extent to which there is support for the discovery of business rules within application software. In most cases this is limited to such things as reading COBOL copybooks but in order to fully discover relationships in legacy code you really need to be able to understand such things as COBOL re-definitions, as discussed above.

How you can do it

Much of the process of discovery is manually intensive. While some manual involvement will always be required (the software may suggest relationships but only you can validate them) it will be beneficial to minimise this, which means that the software should incorporate as much automation as possible. For example, if we consider the discovery of candidate primary and foreign keys it will be useful if the software can automatically check all possible combinations before presenting them to you in ranked order of likelihood. Note that the more columns, tables and data sources you are analysing the more important this will be. With only one or two sources with few tables and columns the advantages of automation may not be particularly apparent but as the number of columns, tables and sources increases the number of combinations increases exponentially.

It is also worth bearing in mind that if you are constructing an MDM hub, or undertaking a data migration based on an application suite such as SAP R/3 or the Oracle eBusiness Suite, then it will be useful if you have a tool that has a deep understanding of these environments and that can extract all relevant relationships automatically along with business-based ontological information. This is analogous to the situation with COBOL redefinitions as discussed previously.

Types of tool

We are aware of four different approaches to data discovery, some of which are complementary rather than competitive.

1. Data modelling: you can use the reverse engineering capabilities of data modelling tools to capture metadata. However, as we have discussed, this approach is limited to metadata only whereas much of the discovery capability we have considered requires the ability to work at the data level as well as the metadata level. For this reason, data modelling on its own (unless it has been extended in some way) will not usually be sufficient for data discovery, though it may well be used to complement other solutions. That said, there are data modelling tools that have the sort of in-depth understanding of common application environments as discussed in the previous section.

2. Model-driven: we are making a distinction between this approach and that of data modelling, where the latter uses Entity-Relationship diagramming and the former is based on a three-tier physical/logical/conceptual model. The only product in this category (that we are aware of) uses its model-driven architecture in conjunction with in-depth source system analysis in order to discover business rules and relationships through working with both application code (including COBOL redefinitions) and data. This approach is particularly suitable for data migration (and especially legacy migrations) as well as in conjunction with master data management projects and as complementary to traditional data quality tools.

3. Structured search: there are several vendors that provide tools that work by indexing the data in relevant data sources and then providing a search front-end. Some vendors with this sort of capability specifically target business intelligence markets while others are also active in providing data discovery at a technical level. Note that you have to know what tables to index, but this is usually relatively straightforward. This approach is especially useful where it is data values that you want to retrieve in addition to, or instead of, the relationships themselves.

4. Data profiling: this is by far the most common class of tool used for discovery purposes. Apart from the functionality we have already discussed the main differences between products is based on whether they work on the data in situ (within existing data sources) or they extract it into a database of their own, where you can analyse the data to your heart's content. The latter typically provides a richer set of capabilities because all of the information resides in a single place. For example, one of the companies using this approach is planning to move into the business intelligence space, applying the principle we referred to before, that all business intelligence is based on an understanding of relationships. We should note that there are also hybrid approaches that do simple work in situ but, for more complex functions, create appropriate artefacts that are stored internally to the tool: evidence suggests that such an approach can be at least as sophisticated as one in which the data is extracted. A further distinction should be made in that some data profiling tools are limited to profiling only one data source at a time, while others may be constrained to just a few cross-source capabilities. More advanced tools do not have such limitations. Between them, the tools in this category can pretty much meet all of the "what you can do" requirements detailed previously, though there is a wide range of capability between products and we are not aware of any vendor that meets all of the criteria listed. Most tools in this category are relatively weak when it comes to legacy systems and would not typically have the ability to understand application code.

Conclusion

Conventional data profiling is just fine if you are working with a single source of data and it could be argued that data profiling should expand beyond its present horizons to encompass the sorts of capabilities we have discussed but it is our opinion that data discovery should be regarded as a market in its own right rather than just being treated as a subset of data profiling. As we have demonstrated, the utility of data discovery is not limited to data quality projects and, in any case, there are other tools that are not data profiling products, which offer data discovery capabilities.

More importantly, we believe that the ability to discover and understand the relationships that exist across your data, wherever it resides, is of fundamental importance to a number of IT disciplines. Moreover, the larger and more distributed your IT resources are, the more benefits you should derive from appropriate data discovery products. However, we are not aware of any tool that, on its own, meets all of the requirements outlined in this paper: you should therefore apply appropriate diligence in determining which products, or combination of products, will best suit your needs.

Bloor Research overview

Bloor Research is one of Europe's leading IT research, analysis and consultancy organisations. We explain how to bring greater Agility to corporate IT systems through the effective governance, management and leverage of Information. We have built a reputation for 'telling the whole story' with independent, intelligent, well-articulated communications content and publications on all aspects of the ICT industry. We believe the objective of telling the whole story is to:

- Describe the technology in context to its business value and the other systems and processes it interacts with.
- Understand how new and innovative technologies fit in with existing ICT investments.
- Look at the whole market and explain all the solutions available and how they can be more effectively evaluated.
- Filter "noise" and make it easier to find the additional information or news that supports both investment and implementation.
- Ensure all our content is available through the most appropriate channel.

Founded in 1989, we have spent over two decades distributing research and analysis to IT user and vendor organisations throughout the world via online subscriptions, tailored research services, events and consultancy projects. We are committed to turning our knowledge into business value for you.

About the author

Philip Howard
Research Director - Data

Philip started in the computer industry way back in 1973 and has variously worked as a systems analyst, programmer and salesperson, as well as in marketing and product management, for a variety of companies including GEC Marconi, GPT, Philips Data Systems, Raytheon and NCR.

After a quarter of a century of not being his own boss Philip set up what is now P3ST (Wordsmiths) Ltd in 1992 and his first client was Bloor Research (then ButlerBloor), with Philip working for the company as an associate analyst. His relationship with Bloor Research has continued since that time and he is now Research Director. His practice area encompasses anything to do with data and content and he has five further analysts working with him in this area. While maintaining an overview of the whole space Philip himself specialises in databases, data management, data integration, data quality, data federation, master data management, data governance and data warehousing. He also has an interest in event stream/complex event processing.

In addition to the numerous reports Philip has written on behalf of Bloor Research, Philip also contributes regularly to www.IT-Director.com and www.IT-Analysis.com and was previously the editor of both "Application Development News" and "Operating System News" on behalf of Cambridge Market Intelligence (CMI). He has also contributed to various magazines and published a number of reports published by companies such as CMI and The Financial Times.

Away from work, Philip's primary leisure activities are canal boats, skiing, playing Bridge (at which he is a Life Master) and walking the dog.

Copyright & disclaimer

2nd Floor,
145–157 St John Street
LONDON,
EC1V 4PY, United Kingdom

Tel: +44 (0)207 043 9750
Fax: +44 (0)207 043 9748
Web: www.BloorResearch.com
email: info@BloorResearch.com