Performing a requirements analysis is critical to the success of any project. Without a clear goal in mind, success is dubious. There are a number of different philosophies about requirements analysis: top down, bottom up, inside out, etc. The method found to work the best is as follows:

1. Clearly state the problem(s) you wish to solve.
2. Identify all data sources and formats.
3. Identify the users of the completed system.
4. Formulate a specific budget time, money, personnel.
5. Ask users to specifically state what they expect the system to do.
6. Ask management to specifically state their success criteria.
7. Separate their requirements from their "desirements." Only design to requirements. The enhancement phase is where you address the "desirements."
8. Group and "bubble up" requirements.
9. Generate a prioritized requirements table listing the requirement, where it came from, the success criteria, and priority. Keep this table high level. A table with a dozen requirements will be much easier to manage than one with hundreds.
10. Produce a detailed development schedule including hardware, software, personnel, documentation, and reviews. Include outsourcing requirements and long lead time items.
11. Get a sign off of the requirements, resource allocation, and schedule from top management before you go any further.
2.1 Information/Data Modeling

Information and Data modeling, along with the definition of the metadata, is the single most important activity in the design of a data warehouse. If this step is done correctly, success is almost ensured. If not, then areas such as flexibility, scalability, and usability will suffer. Information modeling is best left to the experts. I recommend that you hire a professional information modeling firm or consultant. The consultant should be brought in at the tail end of the requirements analysis and be kept though the design and prototyping and then be called in as necessary during the development and documentation phases. Don't skimp on this phase or you will regret it.

2.2 Design & Prototyping

There are many design methodologies. The ones I've used most successfully are 1. Rapid Prototyping (for small to medium projects) and 2. Structured Development (for large or very complex projects). Rapid Prototyping

The rapid prototyping method is illustrated . There are 5 keys to a successful rapid prototyping methodology:

1. Assemble a small, very bright team of database programmers, hardware technicians, designers, quality assurance technicians, documentation and decision support specialists, and a single manager.

2. Define and involve a small "focus group" consisting of users (both novice and experienced) and managers (both line and upper). These are the people who will provide the feedback necessary to drive the prototyping cycle. Listen to them.

3. Generate a user's manual and user interface first. You will be amazed at what you will find out by producing a user's manual first!
4. Use tools specifically designed for rapid prototyping. Stay away from C, C++, COBOL, SQL, etc. Instead use the visual development tools included with the database.

5. Remember a prototype is NOT the final application. Prototypes are meant to be copied into production models. Once the prototypes are successful, then begin the development processing using development tools, such as C, C++, Java, SQL, etc. Structured.

Development. When a project has more than 10 people involved or when multiple companies are performing the development, a more structured development management approach is required. Note that rapid prototyping can be a subset of the structured development approach. This approach applies a more disciplined approach to the data warehouse development. Documentation requirements are larger, quality control is critical, and the number of reviews increases. While some parts may seem like overkill at the time, they can save a project from problems, especially late in the development cycle.

2.3 Development & Documentation

Once the requirements analysis is well underway, the prototypes are working, and the focus groups are becoming happy, it's time to begin the development. Coordinating hardware and software purchases and upgrades, server and hardware installation, software and database development, documentation guides and manuals, reviews, and testing can become a full time job. The key to keeping a handle on all of this to maintain a good written schedule that everyone can view and to have periodic "all hands" reviews. Remember that working with vendors can be a frustrating experience. Hardware incompatibilities, data format incompatibilities,
software bugs, late deliveries, etc. are more the norm than the exception. Outsourcing can help, but you must be continually involved to ensure success.

2.4 Test & Review

Testing and Reviews take place throughout the development cycle, including prototyping, development, deployment, operations, and enhancements. It never ends. It's wise to place a single individual in charge of testing and reviews. This is not a popular job, but it is critical for developing a system that works and meets each of the requirements. Be sure to empower this person (usually a quality assurance engineer) with the appropriate authority. Also, provide them with an appropriately sized staff. Testing is time consuming, tedious work and preparing for reviews and analyzing results can take much longer than you might think. Fortunately this person can save you from being surprised at budget review time and usually catches most problems before they become too big. If you outsource this task, make sure that you make it clear to the others on the team what the outsourcer's role is and what level of authority they have.

2.5 Deployment & Training

OK, the development is complete, quality assurance is satisfied, the documentation is ready, and all the "off the shelf" products have arrived. Now it's time to put everything together. This can be a highly disruptive time. Make sure that you have full management support and that they understand the nature and effect of the installation and deployment disruption. Scheduling training sessions concurrently with the installation can be an effective use of time. Don't skimp on the training. Make sure you have training in the budget from the beginning and don't dip into it. The best way to ensure success is to effectively train the users so that they will actually use
the system and possibly sing its praises. Also remember that training is ongoing. New employees or employees being moved or promoted will need to be trained.

2.6 Operation
Data warehouses usually contain two or more servers. Tasks such as backups, bug fixes, software updates, hardware maintenance and upgrades, media services, account maintenance, security patches, and other similar tasks must be performed regularly. Operation and maintenance of such services requires an operations staff. It is not enough to "let the users take care of it." If you are providing these services in house then you will need on site support from either an outsourcing agency or in house staff. The current trend is to outsource most of these services. More and more companies are outsourcing the entire data warehouse and access it via the Internet or private network. Outsourcing can result in a substantial savings. Just make sure that your provider can supply the services you require and is available when you need them. Also, be sure to discuss security requirements with them before you hire them.

2.7 Enhancement
There is always one thing you can count on: "Requirements Creep." The more successful the data warehouse, the faster requirements creep will occur. As your users become more sophisticated they will want more and more capabilities. If you can respond quickly and efficiently, your users will again sing your praises (and upper management will definitely take notice). Make sure that you have designed in the ability to add features from the very beginning. Remember to design in scalability and flexibility at all phases of development.