Data transformation is the bread and butter technology involved in most aspects of information management. Transformations are needed to migrate legacy systems to more modern applications, to optimize queries, to translate from one data model to another, to integrate heterogeneous systems into federated databases or warehouses, to perform data cleansing or scrubbing, to evolve a schema and its associated database as driven by changing user requirements, to construct user-customized web sites, and to achieve enterprise-wide integration. While the specific data models being transformed have grown over time to include network model, relational data model, object-oriented schema, and now possibly XML web models, the problems of how to specify such mappings, what language to employ, and how to efficiently execute them have persisted. As demonstrated by the large number of authors that contributed to this issue, interest in this technology is as strong as ever. Although there are a variety of products on the market that achieve some forms of transformations, many are limited to a specific problem and/or a specific data model. How to provide a general umbrella approach solving all or at least a large class of the above problems remains unanswered. This issue reports upon recent efforts from academia and industry both on addressing specific transformation problems as well as on developing more generic transformation approaches.

Abiteboul, Cluet, et al. address data translation for heterogeneous sources. Their solution assumes a middleware data model to which data sources are mapped, along with a declarative language for specifying translations within this common middlelayer data model. Automation of some of the tasks of translation are studied.

Bernstein and Bergstraesser report on facilities integrated with Microsoft SQL Server 7.0 for transforming both data and meta-data. They describe the repository functions of the meta-data manager called Microsoft Repository, such as the Data Transformation Service which helps users develop programs to populate a data warehouse.

Carrer, Joshi, et al. describe the Oracle MediaAnnotator, a metadata translation and management tool to be used with Oracle8i interMedia. MediaAnnotator automatically extracts and transforms metadata from multimedia objects into logical annotations, thus simplifying the indexing and searching of multimedia objects.

Claypool and Rundensteiner present a generic framework for flexible database transformations, called SERF. SERF can be added as thin transformation support layer on top of current schema evolution systems, enhancing them with flexibility, extensibility and re-usability of transformations.

Davidson and Kosky introduce a declarative (Horn-clause) language, WOL, developed for specifying transformations involving complex types and recursive data-structures. WOL transformations, which map from a source to a target schema, are easily modified to respond to schema evolution of the source schema. WOL mappings can explicitly resolve incompatibilities between source and target.

Haas, Miller, et al. advocate database middleware systems as transformation engines. They argue that a middleware transformer must provide database-like features, in particular, transformation support for heterogeneous data and efficient query processing capabilities to compensate for less-capable sources.

Heiler, Lee, and Mitchell apply software repository technology to the problem of legacy system migration. Correctly transforming systems (including data, programs and processes) requires metadata about the two objects and their interrelationships. Hence, metadata repositories that provide tools for capturing, transforming, storing, and manipulating metadata are advocated for supporting the migration process.

Hellerstein, Stonebraker, and Caccia introduce the Cohera Federated DBMS as a way to add data independence back into the process of integrating heterogeneous databases. They advocate physical independence for scalable physical design of enterprise-wide data, and logical independence via the use of SQL99 to achieve an open conversion framework.

Finally, Orenstein describes the object/relational mapping system called Enterprise Business Objects (EBO) which is part of a Java application server. Transformations used by EBO to provide transparent, high-performance database access to middle-tier Java applications are characterized.