The ability to interoperate is becoming an essential feature of modern data processing systems. First of all, interoperability is important because usually not one software module meets all the data processing requirements of a whole company. As a result, companies buy and install several components from potentially different vendors (including different database vendors), and these components need to interact in order to serve the company's purposes. One particular situation is that a company buys a new system for, say, sales and distribution and this system must coexist and interact with an old system for, say, stock management. Interoperability is also important to support e-commerce or automatic supply chain management in which different data processing systems of different companies must interact (e.g., on the Internet).

Of course, it will never be possible to plug any two systems together and make them cooperate seamlessly, but there has been significant progress in the last couple of years due to developments such as Java, CORBA, OLE-DB, and object-oriented and object-relational database technology. The purpose of this issue is to give an overview of the state of the art. The bulk of this issue consists of three papers that describe the interoperability concepts and products of three major database vendors: IBM, Sybase, and Informix. While the products of these three vendors have a great deal in common (e.g., use of the abovementioned standards and technology), there are significant differences in the approaches taken by these three vendors.

The first paper describes the IBM approach. The paper shows that IBM provides a whole range of products and features in order to integrate data from external data sources. Specifically, IBM's DB2 database products support (1) *table functions* which can be used to access external data just like (standard SQL) tables; (2) *extenders* which can be used to integrate non-standard data (e.g., geospatial data); (3) *DataLinks* which can be used to access data stored in ordinary files; and (4) *wrappers* which can be used to integrate data from complex external data sources.

The second paper gives an overview of the interoperability concepts implemented as part of the Sybase Adaptive Server products. Compared to DB2, Sybase does not attempt to integrate all kinds of (non-standard) data types and specializes instead on the integration of the most important data types: text, time series, and geospatial data. Furthermore, Sybase implements a special series of query optimization techniques in order to deal with external data and these techniques are also described in the paper. Finally, Java plays a dominant role in Sybase's strategy, and the paper, therefore, also describes SQLJ, a new standardization initiative to embed SQL into Java.

The third paper presents Informix' virtual table interface (VTI). The VTI complements Informix' data blades and object-relational features and plays a similar role than IBM's table functions; i.e., virtual tables make (non-standard) external data appear as (standard) relational tables. The paper describes the details of the interface including support for query optimization and update operations.

The fourth paper describes the design of a *data broker*. A data broker is a tool that can be used to build interoperable systems; a data broker could, for example, be used to connect IBM, Sybase, and/or Informix systems with other database and application systems. Putting it differently, a data broker provides the glue that holds the individual components together. One particular aspect is to achieve good performance, and the paper describes performance issues and experiences with scalable data brokers that perform well in the presence of hundreds of concurrent users and various different component databases.

The fifth paper describes the `Concert` research project at ETH Zurich. The approach devised in that project differs significantly from the general approach taken by the commercial products. The idea is to *export* the functionality of a database system. That is, rather than *importing* the external data and processing the data in the database system, the `Concert` project shows ways to move the database functionality to the data so that the data can be processed in-place. Specifically, the paper addresses issues that arise in the physical design and for transaction management.