I know not what I want, I want not what I get. ....Rabindra Nath Tagore (translated)

The topic of this special issue is *imprecise queries* arising where a user does not know how to precisely express his needs but when presented with mixed results will know what subset he wants and what he does not. The conventional approach of a one-shot subsetting of results specified through a fixed set of predicates will not work in such cases. The goal of this special issue is to make a compendium of tricks that can be used to answer imprecise queries. The topic of imprecise queries is different from the popular topic of approximate queries where the query is exact but the answer is imprecise or approximate.

Two trends in database usage has motivated the need to support imprecise queries in database engines. First is the increasing complexity of the types of data in the form of images, sound clips, and hypertext documents. The extent and variety of attributes of these data types make it hard for an end-user to specify exact queries on them. Second is the growing externalization of databases where an external user querying the data source does not know (and does not want to know) the exact schematic details of the data. Online retail catalog searches by end customers is a prime example of this kind of search. A customer would typically like to search the catalog database using a combination of smart navigation and natural keyword queries without any regard to the physical schema design.

Several techniques have been proposed to find relevant data even when a user cannot quantify relevance through a precise query. The first trick is iterative refinement where the user's query is composed in multiple rounds of interaction instead of the traditional one-shot approach of conventional querying. In each round, the user provides feedback to the system on the subset of answers that he found relevant and optionally modifies the original query. Another important idea is to exploit a notion of proximity or similarity between objects to retrieve objects that are close to a set of seed objects that the user found relevant. Finally, to compensate for the lack of a precise threshold to quantify the relevance data items, the result set is ranked according to a decreasing order of relevance measured via a user or system-induced scoring function.

In this issue, we present a collection of five articles that covers various kinds of data sources and deploys one or more of the above set of tricks to support the imprecise world of user queries. The first two papers "Query Refinement in Similarity Search Systems" by Chakrabarti and Mehlotra, and " Multimedia Queries by Example and Relevance Feedback" by Wu, Faloutsos, Sycara and Payne elaborate how the system refines its internal representation of the user's query concept through multiple rounds of relevance feedback. The query concept consists of a weighted set of objects already found relevant, a distance function to measure proximity between objects, and a function that combines the distances to each of the seed relevant objects. The two papers differ in the forms of these functions and how they get refined. The third paper "Keyword Searching in Databases by Bhalotia, Nakhey, Hulgeri, Chakrabarti and Sudarshan discusses how a relational database can be imprecisely queried using a set of keywords. The answer set returns tuples sorted according to an interesting notion of proximity spanning multiple relations connected through reference links. The paper also presents an exhaustive survey of other systems that support keyword search in relational databases, particularly in catalog databases. The fourth paper, "Using probabilistic argumentation systems to search and classify Web sites" by Picard and Savoy presents a theoretical framework for how hyperlinks in documents can be used to "spread activation" about the relevance of documents in a hypertext repository. The final paper "Towards Scalable Scoring for Preference-based Item Recommendation" by Stolze Rjaibi concentrates on the implementation issues of supporting user specified scoring functions for ranking products based on user preference.