Auto-updatable Index Approach for OODBMSs

Tomasz M. Kowalski, Kamil Kuliberda, Jacek Wiślicki, and Radosław Adamus

Computer Engineering Department, Technical University of Lodz, Poland
{tkowals,kamil,jacenty,radamus}@kis.p.lodz.pl

Abstract. The paper contains a short introduction to robust approach (including architecture) for realisation of auto-updatable indexing of data in OODBMS, i.e. maintaining cohesion between data and indices. The authors work is based on the Stack-Based Query Language (SBQL) and has been implemented and verified in the ODRA (Object Database for Rapid Applications development) OODBMS prototype.

Keywords: automatic index updating, triggers, OODBMS, SBA, SBQL.

1 Indexing in ODRA OODBMS

The general idea of indexing in object-oriented databases does not differ from the one in relational systems [5]. Database indices ought to ensure two important properties: transparency and automatic updating. Indices, like all redundant structures, can lose cohesion if the data stored in the database are modified. Thus, to ensure validity of indices, the update of data has to be combined with rebuilding of appropriate index structures. The rebuild process should be transparent to abstract a programmer of this inconvenient and error prone task. Furthermore, additional time required for an index update in response to data modification should be minimised. To achieve this, database systems should efficiently find indices which became outdated due to performed data modification. Next, the appropriate index entries should be corrected. Such index updating routines should not influence performance of retrieving information from the database and the overhead introduced to writing data should be minimal. The theoretical idea for query optimisation using indices was developed and presented in [3]. The implementation of indexing is based on Linear Hashing structures which can be easily extended to its distributed version SDDS [2]. Moreover, the implementation provides extensive query optimisation supported by enabling: (1) support for optimising dense and range queries on *integer*, *real*, *string* and *date* keys, (2) dense indexing for *reference* key values, (3) indexing using multiple keys, (4) special support facilitating indexing of *integer*, *real*, *string, date*, *reference* and *boolean* keys (*enum* key type) with a countable limited set of values (low key value cardinality) giving additional possibility in applying multiple keys.

The architectural view of the proposed index update process is presented in Figure 1.We assume that an administrator adds an index, Triggers Definitions (TDs) are created before Index Update Triggers (IUTs) – see (1a) and (1b) in Figure 1. Index Manager (IM) initialises a new index and issues Triggers Manager (TM) a message to build TDs, next, the TM activates the Index Updating Mechanism (IUM)

R. Meersman, P. Herrero, and T. Dillon (Eds.): OTM 2009 Workshops, LNCS 5872, pp. 15–16, 2009.

which basing on the knowledge about indices and TDs proceeds to add IUTs – a Root-IUT for the databases root entry, while a Non Key-IUT is added to an indexed non-key object; then a key value is evaluated and an adequate entry is added to the created index.

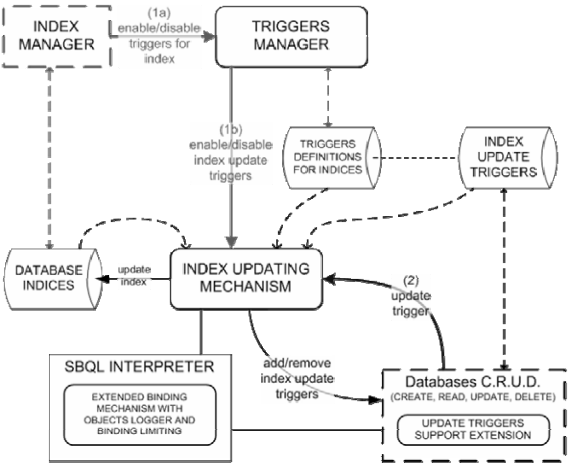

Fig. 1. Index Updating Engine architecture

Removing an index causes removal of IUTs (together with NK-IUTs corresponding index entries are deleted) and TDs. The mediator managing addition and removal of IUTs is a special extension of the CRUD interface. The other case is when the IUM is activated and when the stored CRUD interface receives a message to modify an object which is marked with one or more IUTs (see (2) in Figure 1). CRUD notifies the IUM about forthcoming modifications and all necessary preparation before database alternation are performed. After gathering required information, CRUD performs requested modifications and calls IUM to work. A significant element used by the Index Updating Mechanism is the SBQL interpreter – see [1, 4] for details (out of scope of the paper). For implementation details see [1].

References

1. Kowalski, T.M., Wislicki, J., Kuliberda, K., Adamus, R., Subieta, K.: Optimization by Indices in ODRA. In: First International Conference on Object Databases, Berlin, pp. 97–117 (2008), ISBN 078-7399-412-9
2. Litwin, W., Nejmat, M.A., Schneider, D.A.: LH*: Scalable, Distributed Database System. ACM Trans. Database Syst. 21(4), 480–525 (1996)
3. Płodzień, J.: Method in Object Query Languages. PhD Thesis. IPIPAN, Warszawa (2000)
4. SBA & SBQL Web pages, `http://www.sbql.pl/`
5. Elmasri, R., Navathe, S.B.: Fundamentals of Database Systems, 4th edn. Pearson Education, Inc., London (2004)