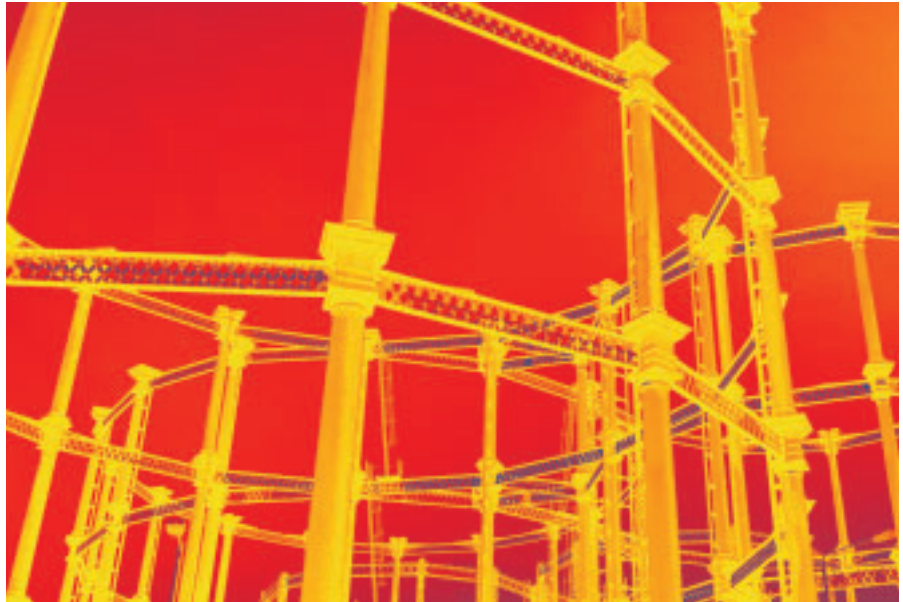


Data Management in KOffice

Modular Architecture

The Kexi project with its KDE database program not only provides useful basic functions for end users, but also interesting technologies for database developers. The complete rewrite of Kexi has allowed the developers to add new ideas to the existing architecture.

BY LUCIJAN BUSCH



No office suite is complete without a database front-end. This is Kexi, in KOffice's case. In contrast to comparable programs, Kexi can be used to create small databases without needing to first set up a server. Kexi stores the data in a single file in this case. Of course, if you need to handle larger amounts of data, or provide access to multiple users, it does make sense to use a server.

Starting Over is Hard to Do

Although the beta1 version of Kexi had a good range of features, the developers decided to go in for a complete rework of the architecture. This explains why the front-end provides only basic database management facilities at time of writing. Users can create tables and edit the data in them, but Kexi does not currently have a front-end that would allow users without SQL skills to define database queries.

Users that speak SQL can use drag and drop to create a query, as Figure 1 shows.

To perform a query across multiple tables, users first need to define the relationships between the tables in the database schema editor. Again, they can use drag and drop to do so. More experienced users who prefer direct SQL entries can use an editor with syntax highlighting and a history function. Self-

programmed SQL queries can be reviewed and modified in draft mode. Both in draft and in SQL direct entry mode, Kexi will store only the SQL input. The program will store incorrect statements allowing you to correct them later.

Kexi includes an integrated database engine called KexiSQL that can be used to create simple projects. Kexi will store your project data in a file, allowing users to exchange data by email for example, or publish the data on a website. KexiSQL is derived from the popular SQLite engine. Although Kexi has a native database format, it can still access other DBM systems such as MySQL and PostgreSQL. Kexi is independent of the underlying DBM system both from the user's and the developer's perspective. An ODBC driver is in the pipeline.

Back to the Future

The beta1 version of Kexi, which the developers released in April 2003, had a quite a few additional

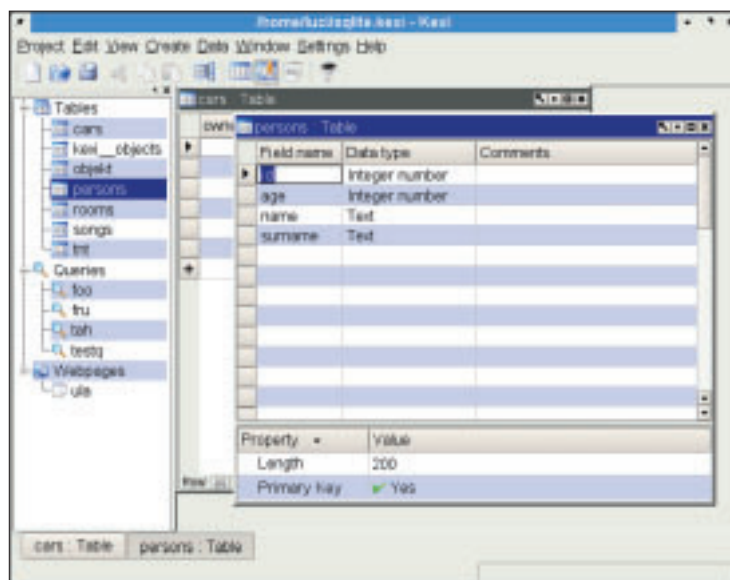


Figure 1: Convenient database manipulation with Kexi. Although the program is at an early beta stage, it can handle basic operations such as creating tables.

simply build on the KexiDB abstractions. Despite the high degree of abstraction, the operations are fast and do not need a complex API.

KexiDB provides a useful set of meta-information for queries or tables. The querying application can access this information by means of schema classes. Schemas are editable, allowing users to add new fields to tables or define queries. After completing the extended schema definition, drivers translate the meta-information into SQL statements. These will be select statements for queries, and create or alter statements in the case of tables and indices.

The KexiDB library ensures that programmers can do without hard coded queries to a greater extent, at the same time avoiding issues with SQL dialects or misinterpretations. KexiDB also has a parser that converts stored SQL queries back into metadata. Kexi then uses these data to generate the query view as shown in Figure 3.

One of the Kexi parser's special features is its ability to display error messages, for example in case of invalid input, in the local language. Figure 4

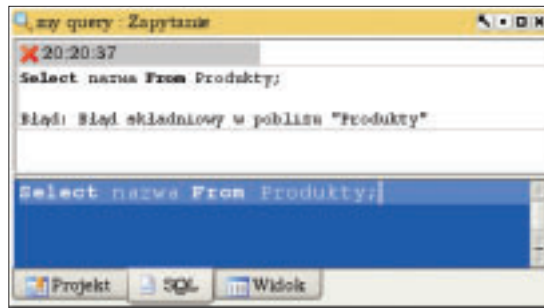


Figure 4: Thanks to the flexible implementation of the KexiDB SQL parser, it is quite easy to output multilingual messages. This shows an example in Polish.

shows an example in Polish. Additionally, the parser marks the point where the error occurred in the user interface. This could be the position of an invalid entry in the input window for example.

The parser is quite error-tolerant and speaks a subset of SQL, allowing it to understand the SQL dialects used by the most popular engines. KexiDB also supports an extended transaction system, allowing for named and nested transactions, assuming the database supports them. The developer plans to support asynchronous database operations in a future release.

Programmers can use KexiDB independently of Kexi itself. Some time in the future, there may even be

a special developer version that will run without KOffice, and only require Qt.

Future

The basic Kexi API still needs to be stabilized. As soon as this has happened, developers can look forward to engine-independent database access, a tolerant parser, and a transaction system that is easy to handle. This said, Kexi will probably need some time before it can provide a powerful and versatile

GUI to users. The development team plans to release a test version that should be of interest to non-developers in the third quarter of 2004. The next stable version will accompany KOffice 2.0. ■

INFO

- [1] Kexi project: <http://www.kexi-project.org/>
- [2] Quanta: <http://quanta.sourceforge.net/>
- [3] Kugar report generator: <http://www.koffice.org/kugar/>
- [4] Qt Script for Applications: <http://www.trolltech.com/products/qsqa/>
- [5] Open Office Polska: <http://www.openoffice.com.pl/>
- [6] The Kompany: <http://www.thekompany.com/>

Rekall

Of course, KDE has a few other database front-ends besides Kexi. Rekall, which was originally developed as a commercial product by TheKompany.com [6], is one of them. Rekall has been available under the GPL since late last year. The program uses the Qt libraries, and thus runs on Linux, MacOS X, and Windows.

Rekall has facilities for creating tables, data entry, creating forms, and even scripting database applications which can be run as executables. This range of features qualifies Rekall as a RAD application, and allows developers to create applications that run independently of Rekall.

One of the major differences between Kexi and Rekall is the fact that Kexi provides direct KDE integration, whereas Rekall runs independently of KDE. Also, Kexi uses ECMAScript for scripting, whereas Rekall uses Python. The user interfaces are also quite different. There is a version of Rekall for the Zaurus PDA, which was developed by TheKompany.com.

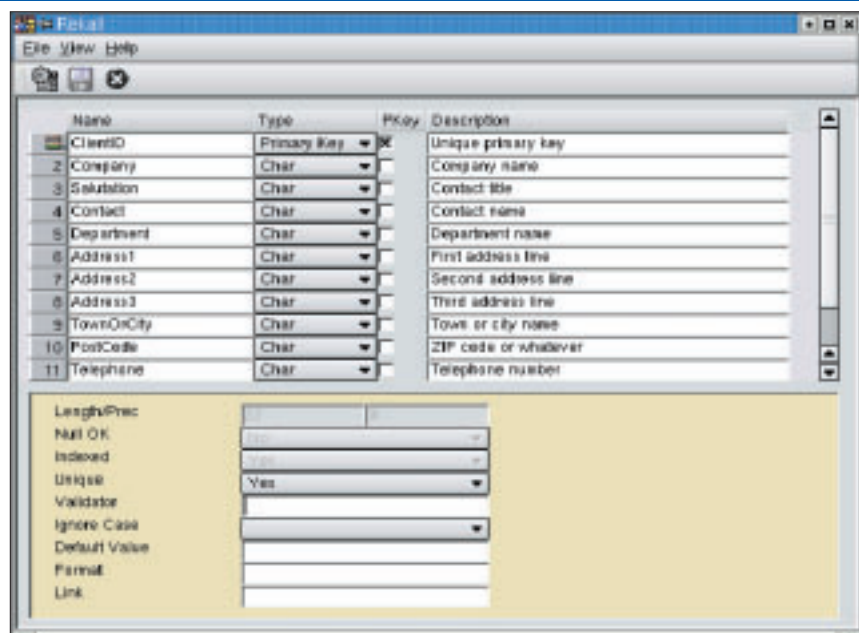


Figure 5: The Rekall database front-end is an alternative to Kexi. It became available under the GPL late last year. Rekall has a wide range of functions. Programmers can use it as a Rapid Application Development environment.