Introducing multidisciplinary thinking in Computer Engineering: A new way of teaching database systems

Claudia Bauzer Medeiros
Institute of Computing
University of Campinas - UNICAMP
Campinas, SP, Brazil
cmbm@ic.unicamp.br

Peter Baumann and Constantin Jucovschi
Electrical Engineering and Computer Science
Jacobs University Bremen
Bremen, Germany
{p.baumann,c.jucovschi}@jacobs-university.de

Abstract—**This paper describes an experiment conducted in Brazil in teaching an undergraduate database course to Computer Engineering and Computer Science students at the University of Campinas (UNICAMP). The experiment has introduced advanced (graduate-level) material in this introductory course, showing students how to integrate course material into several other subjects. The ultimate goal is to engage students in a richer learning environment, enhance motivation, broadening their horizons, and showing them how to solve problems in other fields by looking at a problem from distinct angles.**

This work was conducted within a binational research project involving UNICAMP/Brazil and Jacobs University/Germany.

Keywords- computer science education; multi-disciplinary teaching; non-standard databases; raster databases

I. INTRODUCTION AND MOTIVATION

Computer Engineering (CE) and Computer Science (CS) courses are seeing a progressive drop in student enrollment in several countries – see, e.g., the annual CRA report [1]. Given the pressing need for qualified manpower in this area, universities all over the world are updating their curricula, or introducing new course modalities, to attract a wider spectrum of candidates. Furthermore, traditional subjects are being restructured to show students how to apply what they learn to new domains.

The University of Campinas (UNICAMP, Brazil) is not an exception to this activity. It is one of the country's foremost universities. Undergraduate computer science courses have been introduced in 1969, making them oldest in the country. The UNICAMP computing courses are consistently rated among the country's top five courses (out of roughly 2,000 computing courses).

Besides being responsible for two undergraduate curriculae (in Computer Science and one modality of Computer Engineering) and the graduate program in Computer Science, the Institute of Computing is also responsible for the basic education in computing for the entire university. This increases its responsibility in keeping its courses up-to-date and tailoring them to specific needs (e.g., for Arts or Biology students). Hence, the contents of computing courses go through frequent updates as part of the university's policy of training high-level professionals.

This paper describes results of applying one such modification, in an experiment which was conducted during the first semester of 2009. The experiment concerns teaching database management systems to second year CS and CE majors.

The main goal of this experiment was to introduce a new way of looking at databases, showing students how they could combine what they were learning during this course with what they had seen in other subjects, so that they could solve problems in other sciences. As an experimental platform the rasdaman system [2][3][4] was adopted. This is an open source database system geared towards the management of multi-dimensional raster data, i.e., arrays in a programming sense. Rasdaman allows students to query different kinds of scientific data stored and managed by a database management system; in our case, this is PostgreSQL [5]. With its raster query language, rasql, rasdaman smoothly extends the retrieval capabilities of SQL to multi-dimensional sensor, image, and statistics data as they appear in manifold variations in the earth and life sciences and beyond, such as astronomy.

The paper discusses our findings on extending classical database teaching with a non-standard, but SQL-like retrieval language which allows performing visual query experiments in domains appealing to "techies". Discussion is based on tests applied during the course and a questionnaire filled by the students. These findings point at the interest of motivating basic computing courses with multidisciplinary material from other disciplines. They also show that it is possible to adapt graduate level subjects to an undergraduate course. This not only helps those who are interested in pursuing graduate work, but also shows students the need for first learning the basics, so that afterwards they can understand advanced topics.

The remainder of this paper is organized as follows. Section II briefly presents the contents of a basic database course, and comments on how it is being taught at UNICAMP. Section III discusses advanced material used in our experiment, usually considered at graduate level. Section IV presents rasdaman, the retrieval tool used here. Section V follows on this by comment-

This project is sponsored by CNPq (Brasil) and DFG (Germany) and partially supported by grants from Brazilian funding agencies CAPES and FAPESP.

ing on the exercises applied, and analyzing the results. Section VI covers some related work. Section VII concludes the paper.

II. TEACHING DATABASE SYSTEMS TO UNDERGRADUATE CS STUDENTS

A. Basic course material

Traditionally, basic database management courses cover two complementary aspects: modeling and design, centered on creating a database from a given set of user requirements; and internals of database systems, covering issues such as query optimization, transaction processing and storage structures. These two aspects are often treated in separate courses, the first emphasizing user-related problems, and the second systems implementation issues.

These subjects are usually considered as part of the core of Computing curricula – e.g., [6] – being taught within not only Computer Science and Computer Engineering programs, but also in MIS-related programs [7]. However, in spite of the immediate application of what they learn in this course, students are seldom able to generalize the concepts.

This is, of course, not a problem particular to database teaching, but rather concerns the way courses are often taught in the credit-based system. One way to approach this problem is to introduce projects to be solved by teams [8]. This approach, sometimes called "Project-based learning", is common to many engineering and technological courses, helping students to learn to work together, an essential skill for all professionals. In particular, in database courses, students are given a problem and have to solve it by designing and constructing a database, and subsequently develop an information system on top of this database, accessible by "end-users".

This requires several basic computing skills, such as requirement analysis, programming and designing data structures. For this reason, database courses should never be offered in the first year of study of CS or CE majors.

B. Teaching databases – connecting with other courses

The approach of teaching databases separately from the users' and from the system's point of view gives lecturers and students more time for each subject. On the other hand, students find it difficult to connect the two views. Thus, at UNICAMP, we have designed a basic course that combines both aspects. Students may afterwards take additional credits on advanced topics.

The first author has been teaching database-related topics for over 24 years at UNICAMP, striving to connect course content with other courses [8]. This approach is based on showing students that they can use material covered in databases as a means to partially reviewing other courses they have taken, as well as a preview of other courses.

Course material and exercises are divided into user-related and systems-related. Each topic is introduced with examples and motivation from other topics, always reminding students that subjects cannot be considered in isolation. Table 1 summarizes this approach. Column 1 enumerates a few basic course subjects, and Column 2 indicates courses in which related subjects are taught. The first three lines are mostly user-related concerns, and the SQL language is used as an entry point to systems issues, which are covered in the last four columns.

C. Barriers to overcome – bringing the curriculum up to date

Like in any other CS subject, there is a need to combine two sometimes conflicting goals – to cover the necessary basic material, and introduce advances and state of the art questions to prepare students for their professional life.

Our experiment, discussed from now on, combined these goals. As will be seen, we introduced advanced graduate material in the introductory database course – namely, non-standard queries and applications. To further motivate students and enhance learning, we used a raster query engine so that they had a practical, hands-on experience of visually interesting material.

III. NEW PARADIGMS – NON-STANDARD QUERIES AND APPLICATIONS

Databases are omnipresent in industry, science, and government, hence there is a plethora of practical examples just waiting to be exploited for teaching. Still, in today's classrooms the prevailing scenarios center around business applications or academic administration ("students – teachers – courses"). However, our observation is that at least Computer Science and Engineering students typically have a high technical/scientific affinity and feel much less enthusiastic about administrative and business scenarios. Hence, scientific use cases bear a significant motivation potential for this audience.

TABLE I. RELATING DATABASE AND OTHER SUBJECTS

Course Subject	Other courses
Database modeling and design Requirement elicitation	Software engineering Interface design
Map conceptual to logical levels (e.g, ER to relational) Normalization	Algorithm design Logics
Formal query languages SQL basics	Compilers Logics Interface design
Query optimization	Compilers Operating systems Software engineering
Transaction management	Operating systems Algorithm design Distributed systems
Concurrency control Recovery mechanisms	Operating systems Distributed systems Algorithm analysis
I/O and data structures	Algorithm analysis Computer networks

An additional shortcoming is the lack of appealing visual effects – generally, a database course is perceived much less exciting than, e.g., computer graphics and visualization. Typically, visualization in databases is only addressed when it comes to a condensed presentation of query results to support human grasping of properties embedded in large data sets [10]. Taking advantage of a truly visual semantics normally is not addressed; even multimedia databases aim at extracting semantic information from images to later on query only alpha-numeric data, but not the images themselves.

Aside from motivational reasons, exposing students to non-classical data and query models bears a value because it widens the horizon and avoids hardwiring the equation

"databases ≡ SQL"

in the students' brains. Addressing XML databases is one possible counter measure, but these are not necessarily exciting in terms of visual presentation and use case scenarios either.

Our approach attempts to leverage the benefits of truly visual information for database education. In this approach, it does not matter whether students later on in their professional lives deal with such kind of data and query models – the driving forces in this experiment are (i) student motivation through visual experiences with scientific semantics and (ii) extending the students' horizon beyond "classical" SQL.

To this end, we chose scientific sensor, image, and statistics data as our target. Most often, when sampled or generated from simulations, these come as raster data of some specific dimension and cell ("pixel", "voxel") type. In the earth sciences, for example, we find 1-D sensor time series, 2-D satellite images, 3-D x/y/t image time series and x/y/z exploration data, as well as 4-D x/y/z/t climate and ocean simulation data. Queries on them can rely on use case scenarios which are practically motivated, related to hi-tech appealing to students, and lead to query results that can immediately be verified visually. Actually, this work to some extent is a spin-off from ongoing standardization activities by the German partners where, for the Open GeoSpatial Consortium (OGC) Web Coverage Processing Service (WCPS) standard developed by Jacobs University; a demonstration site with a hands-on sandbox has been established at [10].

At the same time, the information category of raster data is so generic that use cases can be found in many domains beyond geographic applications. Other areas include topics as diverse as diverse human brain imaging, gene expression analysis, radio astronomy and cosmological simulations, and even psychology (where our current research addresses quantitative emotion analysis).

IV. RASQL OVERVIEW

The conceptual model of rasdaman centers around the notion of an n-D array (in the programming language sense) which can be of any dimension, spatial extent, and array cell type. Arrays are functions $f:D \rightarrow V$ from some domain D, which is a subset of d-dimensional Euclidean space $\mathbf{Z}^d$, into some value set V. A rasdaman database can be conceived as a set of tables where each table contains a single array-valued attribute, augmented with a system-generated object identifier suitable for foreign key references (Figure 1).

The rasdaman query language, rasql, offers an algebra-based query language [4] which extends standard SQL92 with declarative MDD operators suitable for a wide range of signal processing, imaging, and statistical operations. Server-based query evaluation relies on extensive logical and physical optimisation and a streamlined array storage manager [3]. Fro a teaching perspective, therefore, all conventional (i.e.: known from SQL) query language principles can be found and discussed.

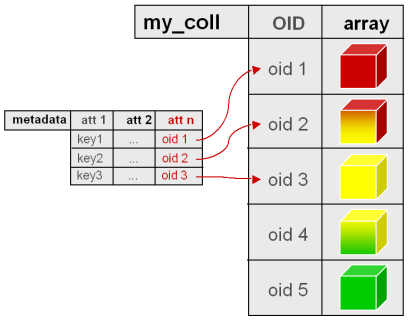

Figure 1. The rasdaman conceptual array model

A. Raster Retrieval

We introduce rasql only briefly, with a walk-through guided by didactics rather than by completeness; see [1] for a more comprehensive and formal treatment.

Like SQL, a query returns a set of items (in this case: either arrays or, for summarization queries, scalars). *Subsetting* of arrays includes *trimming* (rectangular cutouts) and *slicing* (extraction of lower-dimensional sub-arrays). The following query retrieves a 2000x3000 cutout from every Landsat satellite image in collection `LandsatImages`:

```
SELECT ls[ 1001:3000, 1001:4000 ]
FROM   LandsatImages AS ls
```

This works for any number of dimensions, and also allows wildcards to refer to the array boundaries. Let us assume that `ClimateCubes` contains 4-D cubes with time being the first dimension. The following query, then, extracts a 3-D spatial volume at time step 1020:

```
SELECT cc[ 1020, *:*, *:*, *:* ]
FROM   ClimateCubes AS cc
```

For each operation available on the raster cell type, a corresponding so-called *induced operation* is provided which applies the base operation to all cells of an array simultaneously. Both unary operations (e.g., record access, cast operations, or constant multiplication for contrast enhancement) and binary operations (e.g., masking an image) can be induced. An example is "*Band 3 of all Landsat images, with intensity reduced by a factor of 2*":

```
SELECT (char) (ls.band3 / 2)
FROM   LandsatImages AS ls
```

Not only arithmetic, but also boolean operators can be induced. Figure 2 (left) shows the three visible bands of a

satellite view on the South Spanish coast. The query below masks out all non-green pixels from this image; the boolean result of the parenthesis expression is interpreted as 0 or 1, resp., so that a *false* value maps to black and a *true* value retains the original color value:

```
SELECT (    ls.green > 130
        AND ls.red   < 110
        AND ls.blue  < 140
       ) * ls
FROM   LandsatImages AS ls
```

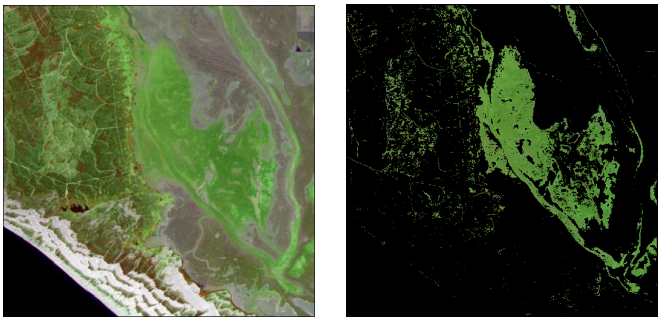

Figure 2. Landsat scene (left) and green detection (right)

In general, array expressions can be used in the `SELECT` part of a query and, if the outermost expression result type is boolean, also in the `WHERE` part. Therefore, we need a means to "collapse" raster-valued expressions, such as induced comparisons, into scalar boolean values. This is accomplished through so-called *condensers* which summarize over the array values, representing the counterpart to SQL aggregation. Based on a higher-order generalized construct the usual aggregates are defined, such as *count*, *average*, *min*, *max*, *all*, and existence quantifiers. For example, the following query retrieves the "*percentage of green area per region in Landsat scenes*".

```
SELECT   count_cells( ls.green > 127 and r )
       / count_cells( ls and r )
FROM   LandsatImages AS ls, Regions as r
```

The `MARRAY` operator constructs a new array with some given extent and a contents determined by a scalar expression used to assign a value into each raster cell position. To this end, it defines locally scoped cell coordinate iteration variables. For example, a 256x256 checker-board array can be defined as

```
MARRAY x IN [0:255], y IN [0:255]
VALUES mod( x + y, 2 )
```

In practice, `MARRAY` frequently appears in conjunction with condensers. The following example illustrates this, deriving a histogram from 16-bit brain scans of unknown dimension:

```
SELECT MARRAY n IN [0:16535]
       VALUES count_cells(b) = n
FROM   BrainScans AS b
```

Advanced applications of this pattern include filter kernels, general convolutions, up to the Fast Fourier Transform. Interesting parallels also can be found to the relational GROUP BY / HAVING clause. A theoretical analysis of the expressiveness and complexity of array indexing expressions is given in [4].

In contrast to alphanumeric query results which can be readily printed, raster results need to be encoded for display. The following query delivers spectral band 3 from all Landsat images, encoded in JPEG:

```
SELECT jpeg( ls.band3 )
FROM   LandsatImages AS ls
```

B. Storage and Processing

The storage model adopted by rasdaman is based on multi-dimensional *tiling*, i.e., a user-definable decomposition into rectangular sub-arrays of suitable size (Figure 3). Tiles are stored as BLOBs (binary large objects, i.e.: linearized byte arrays) in a standard relational database system, such as PostgreSQL.

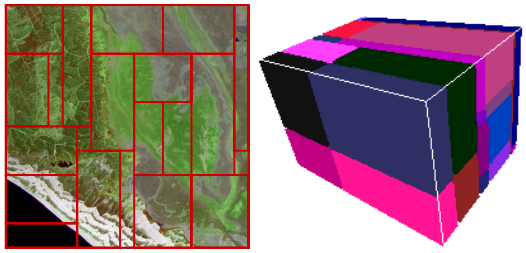

Figure 3. Sample 2-D and 3-D tiling

Query processing relies on *tile streaming*: operator tree nodes read data tile by tile, process them, and pass on result tiles to the next higher node. This way, arrays of unlimited size can be processed without limitations imposed by server main memory. See [3] for further details.

C. Front-End Tools

For the purpose of our experiment, two tools have been provided. *Rview* is a graphical-interactive tool which allows submitting queries typed into a text window, retrieving result sets, and visualizing them in various ways, including 1-D charts, 2-D tables and images, and 3-D animations. The *rasql* tool allows submitting queries via command line and store results in files or, alternatively, output them directly to the console. Especially the latter feature, in combination with an ASCII output format, allows inspecting particular data values in detail and with Unix tools well known to the students at this stage.

V. EXPERIMENT EVALUATION

In order to describe the experiment, we first situate the database course in the CS and CE curriculum and next discuss the students' academic and work profile.

The experiment of reorganizing the database course was conducted during 4 months (from March to June 2009) in a 90-hour undergraduate introductory database course. This course is organized in 6 hours per week, of which 2 are for hands-on exercises and the other 4 dedicated to theoretical classes.

In the first semester of 2009, this course was offered to a class of 65 students in the evening. The Institute of Computing offers both day and night computing courses. The night course

offers a bachelors degree in Computer Science; the day course offers a degree in Computer Engineering. The contents of most computing related courses are the same, though CE has many more credits to cover engineering requirements. In particular, the database course is offered on the fourth or fifth semester of both degrees, and students have already taken courses on programming, data structures and operating systems.

Academically, day and night students are very similar – university entrance examinations are very competitive at UNICAMP, and are conducted on a country-wide basis. Every year, acceptance rate for CE and CS is between 20:1 and 25:1 (though in the beginning of the nineties the competition was as high as 35:1). Hence, only very good students are accepted in our courses.

On the other hand, the profile of students who take night classes is very different from those in day classes. First, most students are employed, and even though in the same age range as day students (e.g., between 18 and 23) they are usually more mature and take their course work more serious. Second, they have a more practical mindset, in the sense that they are always looking for ways to apply what they are learning to solve problems they are facing in their jobs. Finally, since courses take place from 7 to 11 PM every day, plus Saturday mornings, lecturers have to work hard to keep students interested. In this sense, rasdaman exercises were a big success – the students had never experimented with such a system, or worked with such examples (at work or school) and were noticeably motivated by the new kind of exercises. This was also indicated in their answers to the questionnaire – all said they would be willing to participate in similar rasdaman experiments in the future (last question of the questionnaire).

The course material was reorganized in three stages. Eight percent of the 90 hours allotted to the course were occupied exclusively by introducing rasdaman – roughly, 3 hours of theoretical issues and 4 hours of hands-on exercises. Additional hours of graduate level material were also taught, to provide the theoretical basis on non-standard database management issues and multi-dimensional data. From the start, students were told that the course would be "different", i.e., that besides covering the required material they would use a new kind of database system, and that they would be experimenting with novel applications.

First, students were introduced to the basics of database management systems. This was accompanied by motivation on how database systems are needed (and profit from) other subjects, such as VLSI design, climate modeling and astronomy. Here, the notion of raster data structures was also introduced, preparing the ground for the use of rasdaman.

This first half of the course comprised the subjects involving user-related external issues (i.e., database design and querying formalism, see Section II). Each subject was accompanied by practical exercises. Students also had to develop a set of small database applications (in PostgreSQL [5]), to familiarize themselves with database creation and query specification.

Next, students were presented with the internals of database management systems, especially query optimization and transaction management. At the same time, additional material was presented, covering rasdaman basics [3]. First, there was an introduction to the rasql language, showing how it supported queries to raster data. Basic database material seen in the first half of the course was revisited during this stage, showing how it could be used to solve problems in scientific data handling - e.g., in environmental control, image analysis in health studies and in biodiversity. This was accompanied by hands-on exercises on examples of rasql.

Finally, they were introduced to rasdaman architecture, which was compared with the architecture of standard DBMS. For instance, they were shown how query processing in rasdaman was just an instance of DBMS query processing, with adaptations to support vectors and matrices. Further experimentation on rasql was introduced at this stage.

Each exercise was followed by challenges, e.g., asking the students to pose variations of an initial problem. Exercises grew in complexity, starting with simple one-dimensional data handling to processing movies and data cubes (e.g., of geological data, or temperature time series). The tasks to which the students were exposed included the following challenges:

Challenge 1: What is the difference in terms of the result structure of these two queries?

```
SELECT b                  SELECT avg_cells(b)>10
FROM   BrainScans AS b    FROM   BrainScans AS b
WHERE  avg_cells(b)>10
```

The first query obviously would return a subset of the stored images while the second one would return one scalar boolean value per image. Result visualization made the students aware of the data structures generated during query evaluation and allowed them to visually compare the results.

Challenge 2: Generate an array as shown in Figure 4.

A possible answer is this query:

```
SELECT MARRAY x IN [0:10],
              y IN [0:10]
       VALUES (y > x)
FROM   rgb
```

This shows how coordinate positions can influence a cell's value through index arithmetics and, at the same time, highlights one of SQL's issues: no result can be delivered without indicating a table, even if this table is not used for the result.

Challenge 3: Generate the result shown in Figure 5 (right) for a given circle center (x0/y0) and radius (r).

	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	0	0
3	1	1	1	0	0	0	0	0	0	0	0
4	1	1	1	1	0	0	0	0	0	0	0
5	1	1	1	1	1	0	0	0	0	0	0
6	1	1	1	1	1	1	0	0	0	0	0
7	1	1	1	1	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1	1	0	0	0
9	1	1	1	1	1	1	1	1	1	0	0
10	1	1	1	1	1	1	1	1	1	1	0

Figure 4. Task input for generating given data patterns

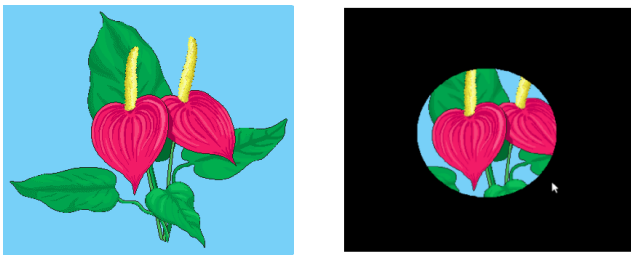

Figure 5. Extracting geometrically defined areas

The expected answer is a query like this one:

```
SELECT rgb *
       MARRAY x IN [0:199],
              y IN [0:199]
       VALUES (sq(x-x0) + sq(y-y0) < sq(r))
FROM   rgb
```

During this part of the classes, several students suggested the use of rasdaman to help teach calculus and linear algebra. They proposed examples of its use in such classes, and were excited at being able to connect the contents of a database course with the more theoretical subjects they had seen the previous year.

Handling images and movies was also another source of interest. Several of them were familiar with movie editing tools, but never from a purely mathematical point of view. Though all were aware of images as 2D and movies as 3D objects, they had never practiced slicing and dicing frames – especially using a database framework. Details on course material for the rasdaman classes appear in [13].

As we all know, hands-on exercises are the most efficient form of learning. However, students mostly see such exercises as a means to practice course material, and are seldom able to see how courses can interconnect. Here, they not only experienced working with a database. To solve the exercises, they had to recall (and thus make the mental links to) notions in linear algebra, boolean algebra, calculus and physics. For instance, in the questionnaires one student commented that for the first time he understood how equations work in practice.

The appendix shows the questionnaire filled by the students. In the original questionnaire, all questions were accompanied by answer options (e.g. "YES, explain why"; "NO, explain why not"). These options have been omitted for the sake of brevity, and only the questions have been included.

All students said they had never seen any database system that was similar to rasdaman – including those that said they had already worked with maps, but never with multidimensional raster data management. They all said the classes were interesting, and 90% would be willing to participate in further exercises. Most said that this system could be used in other courses (examples cited included mathematics, physics, operating systems and image processing). Half of the respondents said that there should have been more of the internals covered, so they could understand "how it works".

Twenty percent of the respondents said that they thought this kind of topic should be left to advanced undergraduate courses. Interestingly enough, those who gave this opinion work as database administrators or database programmers. Thus, they looked at the course from an immediate, vocational view (which is not the main focus in UNICAMP teaching approach).

One of the course's exams also contained one question on rasdaman, asking students to describe its main characteristics and functionality. The idea was not to check whether they knew rasql syntax, but to see how they would describe the system's capabilities. Twenty percent of the students did not answer this question (a few said they had not gone to the hands-on classes). The majority correctly identified the possibility of managing raster data, with examples of what they had seen in class.

VI. RELATED WORK

This section discusses related work in teaching databases and in introducing multidisciplinarity in CS courses.

Curricular issues in computing are now treated in terms of Grand Challenges by joint ACM and IEEE education committees [14]. One of the problems is that computers and computing basics are used in all domains, and there is a need for approaching subjects at distinct levels of complexity and for all kinds of majors – from the Arts and Humanities to Engineering. One consequence of this ubiquity is that CS and CE majors now have to think in a multidisciplinary context, applying what they learn to problems in other domains. However, this requires major curricular modifications, and (re)training lecturers to teach traditional subjects establishing links with other areas.

Multidisciplinarity is being moreover seen as a solution to another problem afflicting CS and CE degree-offering programs – the decrease in student enrollment and the number of drop-outs. This second problem has given rise to the appearance of many new degrees – e.g., [15] – and to new ways of teaching computing, often emphasizing human and social aspects [16]. Multidisciplinary teaching has been established as one of the priorities of the Brazilian Computer Society as a prerequisite to training the future generations of professionals and researchers [17]. Our experiment in teaching databases has gone in this direction, showing students how to deal with problems arising in other domains by working with database concepts.

Within this multidisciplinary context, data management is seen as a primary concern, for professionals and researchers alike. If Computer Science is considered to be the "third pillar" of scientific research (along with the pillars of theory and experimentation), then data concerns the fourth one [18]. As exemplified in Section II, subjects covered in database courses can be linked to several other computing subjects. Moreover, as practiced in our course experiments, database-related issues can be used to support the solution of problems in other disciplines. For this reason, database courses present a very good starting point for implementing curriculum modifications.

Given this role of databases in curricular content, there have been many proposals for teaching this subject (e.g., [7], [8], [19], [20], [21], [22]). However, as mentioned by [19], educators face a dilemma when it comes to database education.

Indeed, database concepts have continued to expand, with the creation of many specialized database systems. Moreover, advanced topics of ten years ago have now become basic topics in many situations, especially with the advent of the Web. The latter is forcing computing courses to adapt to this new environment. Database projects have migrated from standalone applications to distributed Web-based platforms, which has been the case in our experiment, too –in the use of both PostgreSQL and rasdaman.

The survey conducted by [7] also points out in this direction – database courses are now adapting to cover scalability, and the need to accommodate Web-based skills. Advanced topics are also proposed by [19][21][22]. While [19] introduces advanced topics in a second course, and [21] and [22] propose a suite of courses to accommodate topics, we introduced multidimensional data management (a graduate subject) in the first basic course. Students were motivated by this change, though those who work with database systems were more concerned with the immediate application of course content to their daily jobs, and thus did not see the usefulness of learning advanced material in a first course.

Finally, other authors stress specific points of curricular issues involving databases. Some, like [23] or [24], emphasize the need for treating data security as a concern of all that deal with data management [22], or the importance of basics such as transaction management [24] to provide students with a solid conceptual basis for subsequent studies. Others, like [25], see database courses as a way to help non-majors in collaborative learning. In his findings, a database project immerses students in the solution of real-life problems, thereby better motivating the learning experience. Like [24], we stress basic principles and mechanisms, to subsequently show how they can be instantiated in other situations. Like [25], we consider projects to be a good platform to enhance collaboration. The latter was also observed during the rasdaman exercises: since they required remembering subjects from linear algebra and mathematics, several students would get together to solve the exercises. Security, on the other hand, is just mentioned in UNICAMP's course, being covered in advanced topics. However, it could well be introduced as part of a broader problem – that of ethics, nowadays a part of CS curricula.

Finally, it must be stressed that there is no such thing as "one size fits all". While this refers to the discussion in [26] of issues in teaching an introductory CS course, we firmly believe there are domain-specific requirements that cannot be accommodated in a single course. Moreover, student background and expectations have to be taken into account. Thus, our experiment can be replicated only in similar environments – i.e., for students taking CS or CE degrees which require a solid background in mathematics, plus data structures and programming. In particular, such courses will have to take the evolution of data structures into consideration. This is the point stressed by [27], showing how data structures are evolving to consider advanced access methods and parallelism. Computing education must continue to enforce basics, while at the same time emphasizing new kinds of computational thinking.

VII. CONCLUSIONS AND ONGOING WORK

This paper discusses an experiment in teaching database management basics at the undergraduate level by combining standard course material with advanced topics. Besides working on traditional exercises with database tables, students had to practice handling and querying scientific raster data. Exercises were motivated by problems found in other disciplines, especially geography, biodiversity, and physics.

The results of this experiment, analyzed both in course exams and via a questionnaire applied after the course, indicate that it was successful in several ways. First, students' motivation in practical exercises was enhanced. Second, they became interested in browsing Web sources for additional material, connected with other (non business-oriented) uses of database systems. Finally, all students gave very high grades to the usefulness and interest of the exercises with multi-dimensional raster data.

The latter findings are particularly interesting since this was an undergraduate night course in which 90% of the students already work in IT-related jobs (e.g., as programmers or systems analysts). Since exercises relied on data from scientific domains, we did not expect that students would rate them as highly useful, considering that they do not directly relate to their jobs. However, comments on the questionnaire say that they relate usefulness to (a) having learnt new concepts, (b) having understood that database systems are not limited to tuple-oriented transactions, and (c) having discovered that a database system can be used to practice other subjects, such as computational geometry or linear algebra.

We intend to extend this experiment to other semesters to obtain a broader basis on which we can build a more solid exercise base. This will require, moreover, preparation of more concise data sets so that students can conduct several small experiments in a single class on subsets of the same data, and compare results.

Additionally, we will carry out the same experiment at the German university involved to get a broader basis for evaluating our results.

Another extension is to use the same approach in other courses. The first author is adapting some of the material used to teach an introductory CS course to Arts students, in 2010. In this case, the emphasis will be on examples for handling multimedia, and rasdaman will be used as a tool, but not a system to learn database concepts.

REFERENCES

[1] CRA – Computer Research Associates. www.cra.org. As of Nov. 2009

[2] P. Baumann. "A database array algebra for spatio-temporal data and beyond." Proc. 4th Int. Workshop on Next Generation Information Technologies and Systems (NGITS), Springer LNCS 1649, pp. 76 - 93

[3] P. Baumann: "Large-Scale raster services: A Case for Databases" (invited keynote). Proc. Workshop on Conceptual Modeling for Geographic Information Systems (CoMoGIS), Tucson, USA, 6 - 9 November 2006, Springer LNCS 4231, pp. 75 – 84

[4] www.rasdaman.org. As of Jan. 2010

[5] www.postgresql.org. As of Jan. 2010

[6] R. Shackelford et al. "Computing Curricula 2005 – the overview report". ACM and IEE Computer Society, 2005

[7] F. Springsteel., M. A. Robbert, C. M. Ricardo. "The next decade of the database course: three decades to speak to the next". Proc SIGCSE'00, ACM, 2000, pp 41-46

[8] T. Connolly, C. Begg. "A constructivist-based approach to teaching database analysis and design." Journal of Information Systems Education 17(1), 2006, pp 43-53

[9] A. Salgado and C. B. Medeiros. "A pedagogical proposal for teaching database courses" (in Portuguese). Proc. VI Brazilian Workshop in Computer Educational Issues – Quality, 2000, pp 10 -24

[10] H. Ziegler, T. Nietzschmann, D. A. Keim: "Visual Analytics on the Financial Market: Pixel-based Analysis and Comparison of Long-Term Investments". Proc. 12th Conference On Information Visualization (iv 2008), December, 2008, London

[11] www.earthlook.org. As of Jan. 2010

[12] R. Machlin: "Index-based multidimensional array queries: safety and equivalence." Proc. ACM PoDS 2007, pp. 175 - 184.

[13] www.ic.unicamp.br/~cmbm/MC536/main.html. As of Nov. 2009

[14] A. McGettrick, R. Boyle, R. Ibbett, J. Lloyd, G. Lovegrove, K. Mander. "Grand challenges in computing education". British Computer Society, 2004

[15] J. Slonin, S. Scully, McAllister. "Crossroads for Canadian CS Enrollment", *Communications of the ACM*, 51(10):66-10, Oct 2008

[16] M. Buckley. "Computing as a social science". *Communications of the ACM*, 52(4):29-30, April 2009

[17] C. B. Medeiros. "Grand Research Challenges in Computer Science in Brazil". *IEEE Computer*, vol 41, pp 79-85, April 2008

[18] T. Hey, S. Tansley, K. Tolle (editors). "The fourth paradigm: data-intensive scientific discovery." Microsoft Research, Redmond, WA, October 2009

[19] S. Urban and S. Dietrich. "Advanced database concepts for undergraduates: experience with teaching a second course". Proc. SIGCSE 2001, ACM, pp 357-361, 2001

[20] C. Pahl, R. Barrett, C. Kenny. " Supporting active database learning and training through interactive multimedia". Proc. ITiCSE'04, ACM, 2004, pp 27-31

[21] M. Murray and M. Guimaraes. "Expanding the database curriculum". *Journal of Comput. Science in Colleges,* 23, 3 (Jan. 2008), pp 69-75

[22] S. Bagui and L. T. Haar. "Database education in the new millenium". *Journal of Computer Science in Colleges*, 24(4): 80-87, April 2009.

[23] H. Said, M. Guimaraes, Z. Maamar, L. Joolian. "Database and database application security". Proc ITiCSE'09, ACM, 2009, pp 90-93

[24] D. Hansen. "A sin of omission: database transactions". *Journal of Computer Science in Colleges*, 21(1):225-230, October 2005

[25] D. Goelman. "Databases, non-majors, and collaborative learning". Proc. ITiCSE'08, ACM, 2008, pp 27-31

[26] T. Huang, A. Briggs. "An introductory approach to unified computer science: can one size fit all?". Proc ITiCSE'09, ACM, 2009, 253-257

[27] D. Ernst, D. Stephenson, P. Wagner; "Hybrid and custom data structures. Evolution of the data structures course". Proc. ITiCSE´09, ACM, 2009, 213-217

APPENDIX: QUESTIONNAIRE

This appendix contains the questionnaire handed out to the students in both classes to obtain feedback on our approach.

This questionnaire on the classes you had about rasdaman is important for the planning of future offerings of the subject. It will also be used as a basis for a scientific paper. A summary of the main results will be sent to all respondents. Thank you for your collaboration.

Please answer all the questions. In doubt, answer "I have no opinion about this subject". Always answer in the space below the question. You can write as much as you want.

1. This kind of use of databases and DBMSs was new to you, or had you already seen something similar? If not new, please explain.

2. Were the classes on rasdaman useful in your training? Give an example of what you learnt towards this goal.

3. What were the highlights of the classes? And the main shortcomings?

4. Were there some topics that you think should have been covered?

5. Was the time alloted to cover the material adequate (3 classes – approximately 8 hours)? If not, how many hours would you recommend? [Why]

6. Was the number and type of classes adequate (1 theoretical class, 2 hands-on classes)? If not, what would you recommend? [Why]

7. Was the material used appropriate (PDF slides)? If not, what else should have been provided?

8. Do you think the material of these classes could also be presented in other courses? If yes, give an example.

9. Would you recommend we give these classes for the next offerings of the database course? [Why]

10. In an interval from 1 (minimum) to 5 (maximum), which grade would you assign to the rasdaman classes, considering the entire experiment (content, methodology, material)?

11. Would you like to participate in other activities using the same software?