

Reading Data Schemas and Knowing a db Query Interface in Non technical Secondary Schools

G. Barbara Demo, *barbara@di.unito.it*

University of Torino, Department of Informatics, c.so Svizzera 185, 10143 Torino, Italy

Abstract

The ability of specifying simple queries on a database is felt necessary by a growing number of non-computer specialists in order to have direct access to their own data. As an example, physicians quite often ask for personal access to the data concerning therapies and treatments of their patients. Another example concerns open data. The research about this subject must also consider open access at least with simple queries in order to reach the data transparency that is among open data most considered aims, particularly for the public administration data. The experience here described is under development in two Italian secondary schools specialized in ancient cultures (Latin and Greek) called Classical Lyceum. Students are first introduced to reading the Entity-Relationship data model and to discovering via an activity of “reverse modelling” what part of the world is considered in a given database (db). The next steps concern how to specify queries here seen as composition of sets of information. Multiple problems of db design and of db management are intuitively evident also to people who do not have a computer science background. Thus an important component of this proposal is discussing with students two concrete examples of how complex computer problems, such as the design and the management of a db, are solved by layers of abstraction, one of the main principles of Informatics as a science.

The experience here described is proposed as a contribution to the discussion on which aspects of computer science shall be present in secondary schools other than in technical and vocational schools and possessed by all in the next future.

Keywords

informatics education in secondary school, open access to open data, informatics principles, informatics competences.

INTRODUCTION

The abilities of understanding a basic data model, for example the well known Entity-Relationship (ER) data model, and of specifying simple queries on a database (db) are becoming necessary to a growing number of persons and shall be part of Informatics education for all. Relevant examples concern medical environments and the open data.

The author often works with colleagues from the college of Medicine to design databases for their research data, i.e. data concerning therapies and treatments of their patients, not the administrative data. Once the db is available, often physicians ask to acquire (some) abilities to query their research data.

Open data have been considered by B. Obama the foundation of his government transparency on a 2009 document. Also in Europe, one of the strongest motivations of these last year's large interest around open data is the transparency of the data

originated by the public administration as one can read in reports in Europe's Open Data website. Nowadays activities around open data mainly concern publishing standards and legal aspects of privacy, but in the near future a broad diffusion of open access abilities shall be considered to guarantee a true data transparency.

On the educational side, the presence of computer science in schools is still unsatisfactory as from the 2010 report by Wilson, Sudol, Stephenson and Stehlik. Thus, after many discussions about how ICT is insufficient and how principles of Informatics must enter in schools, researchers are active in proposing what shall be taught, what is most suited for the different types of schools and how to attune Informatics concepts with traditional curricula. Brinda, Puhlmann and Schulte (2009) consider the problem in lower secondary education. We think urgent to discuss which computer science subjects other than programming are most suited to the different types of schools because a strong focus on programming turns out to appear too technical to politicians and to education organizations that have decision power over school curricula.

In Italy the problem is particularly difficult with respect to the secondary schools of the type called Liceo, Lyceum, and, among these, with respect to the classical Lyceum, where students learn Latin and ancient Greek languages and cultures. In this Lyceum informatics as a science is perceived as a discipline having little to do with their curricula where also traditional scientific subjects, such as mathematics and physics, are present only for a few hours a week. Here we sketch how the growing social demand for a new presence of computer science in schools converges into a proposal for informatics lectures in two classical Licei in Italy.

Antonisch, in a 2006 paper, and Stroeder, in 2012, have proposed databases as a subject for informatics in all types of secondary schools. Here we describe how some particular db aspects are proposed in a humanity or "classical" type of secondary school not only to ensure open access to data but also as an activity showing to students concrete examples of how complex computer problems, such as the design and the management of a db, are solved by layers of abstraction, thus showing one of the principles of Informatics as a science.

Motivations of this specific proposal are described in section 2 while in section 3 the real case proposed to students is sketched. The experience is under development and is organized along the steps summarized in section 4. In the conclusive section we point out that the activities of the project here summarized can also interest university students in humanities.

MOTIVATIONS FOR A DB PROJECT IN CLASSICAL SECONDARY SCHOOL

Informatics is present in Italian secondary schools as a mandatory technical subject in the branch of the Scientific Lyceum called Applied Sciences, in technical schools having Informatics specialization and in some vocational schools. Unfortunately students attending these types of schools are around the 20% of the country secondary school students. In the other secondary schools some, not all, have "Elements of Informatics" where students are introduced to different software to advance learning across their curricula disciplines.

After years attempting to have contacts with teachers in the Lyceum type of secondary schools we decided to face the Classical Lyceum that is the farthest from sciences in general. The activity we propose integrates two believes of ours: in the near future all persons will need basic abilities to access data and some aspects of the humanities culture, fostered in the Classic Lyceum, help in activities related to accessing a db. In short, we believe that open data access is particularly suited to both teachers and students of this type of school because of the attitudes developed by having translations from ancient languages as a central part of the education duties. In particular we consider well suited to students of a classical Lyceum the following aspects:

- *Comprehension of a formal language.* Translating from a lost language has similarities with translating from a formal language used to communicate with a computer. In our experience we focus on reverse modelling, i.e. we stress on reading a data schema more than on being able to design a data schema. This is done for short of time but also because being able to read a data schema is mandatory to know which parts of the reality are present in a db. Also, reverse modelling may appear to students as an experience similar to the translation from a dead language into a modern natural language
- *Text analysis.* Similarly, when we have to perform any kind of translation we must first discover the possible ambiguities of the text to be translated and be sure of our understanding: shortly we must do a type of text analysis similar to the one the students perform when dealing with Latin or Greek texts. Thus also the user requirements analysis, that is the step of a db design going with the requirements collection, is connected to the standard curriculum.
- *Composing sets and/or using first order logic.* Set theory is nowadays part of students' culture progressively introduced from primary education.. First order predicate calculus, with exercises on using the universal and existential quantifiers, is often covered in classical secondary schools as a part of philosophy and/or mathematics disciplines. Entities, relationships (or associations) and relational tables are sets and the answer set to a query, on a given db, is a composition of the sets present in the db by means of a formal language (such as the relational algebra) or a query interface based on the first order predicate calculus.

Finally, as an achievement quite positive for all types of schools, posing queries to a db also contributes to the students' inquiry capabilities. When they want to ask a query on a given db first of all they have to find out whether the set of information they would like to know is retrievable from the different sets existing in the db, i.e. they must recognize whether they have to solve a well posed problem. On this aspect a reference to the Stella Baruk's book, 1985, is always interesting.

Last, our secondary schools professors approved the experience also considering it important for the future careers of their students who, after this type of secondary school, mainly chose Medicine, Law and Ancient cultures for their university.

THE CASE STUDENTS WORK ON

In our project and for all db concepts in what follows we consider the book by Atzeni, Ceri, Paraboschi and Torlone, 1999, as a general reference. Students work on a few examples and on a real case type of exercise. Figure 1 concerns one of the examples and is related to an animal refuge organized in several sectors where animals can be recovered when needed. The use of the two different data schemas shown in the figure is described in the next section. For the real case exercise, students are given a data schema modelling a simple library that can be the students' own library. In a second time they are shown an extended version of the library schema considering not only books but also CDs and DVDs with an entity MyOwn-Objects generalizing the entities BOOK, DVDs and CDs. We have chosen to study this case both because it can be of interest for students and because there are Italian books that cover introduction to databases by developing similar examples with several queries using different Office suites. This shall help students in developing new queries or enriching their db experience after the activities of the project are over.

ORGANIZATION OF THE ACTIVITIES

The project is an optional activity involving two groups of 16-18 years old students, a group for each of the two secondary schools involved, and is organized in six steps covered in seven lectures-activities of two hours each. A full introduction to

db systems is of course not the aim of the proposal. The following description of the activities and of their schedule is to show that the perspective we focus on is that of a user of a db and not the one of a db engineer.

We also propose two talks open to all students of the two schools. The first is an introduction to the activities for presenting the project and for giving motivations. The second concerns a number of data models used in Artificial Intelligence in order to illustrate data models different from the one seen during the db activities to students taking part in the project.

Motivation and Introduction – step 1

(2 hours) The first activity is an introduction to formal languages (data models) for specifying in a data schema the data of interest that will be found in the db. Basic formal aspects are introduced, such as the entity identifiers and the relationships' cardinalities, and then we give an example of a set interpretation of the schemas (and of the data model). For the schema in figure 1.A a set interpretation of entities and relationships contains five sets, one for each entity and one for each relationship that can be as follows:

ANIMALS= {(c1, n1), (c2, n2), (c3, n2)}

SECTOR= {s1, s2, s3, s4}

SPECIES= {S1, S2, S3, S4, S5}

IsOf = {(c1, S1), (c2, S1), (c3, S2)}

Assigned = {(S1, s1), (S2, s1), (S3, s2), (S4, s3), (S5, s3)}.

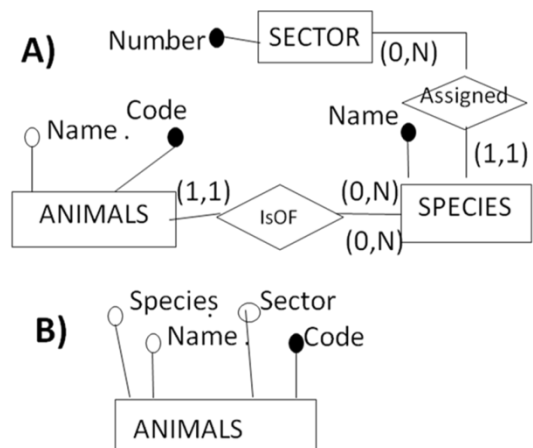


Figure 1. Different data schemas related to the same reality.

In figure 1 full dots indicate entities identifiers. The given set interpretation contains three animals having each an identifying code and a name (two animals have the same name n2). In the IsOf association set we find that the animals with code c1 and c2 are of the species S1 while the animal c3 is a member of the S2 species. The set for the association Assigned gives us information on which sector is prepared for the species considered in the SPECIES set. Sector S4 is not yet assigned (which is correct since SECTOR takes part in the Assigned association with cardinality (0, N). Similarly there are no animals of the species S3, S4 and S5.

Reverse modelling – step 2

(2 hours) By reverse modelling we mean the ability to read a data schema, in our case an ER schema, i.e. translating the schema into a natural language specification of the reality represented. Reading data schemas is necessary not only to know what db contain but also to understand how important syntactical rules of a data model are in order to precisely understand the reality concerned.

From our experience in teaching db in Computer Science university courses we know reverse modelling is always useful whenever a data model is introduced for all levels students. A fruitful exercise to begin with is to compare two or more simple data schemas such as those in figure 1 related to the same reality but expressing different user requirements for what shall go in the db i.e. expressing different models of the same reality. Schema in figure 1.A specifies that “there are sectors, each distinguished by a number. A sector can be acquired by the refuge but not yet assigned to any species or may be assigned to (possibly prepared for) more species.

Each one of the considered species has exactly one sector where it can be recovered”.

Of course there is not a unique correct translation, but we can detect unacceptable translations such as the following one for the data schema in figure 1.B: “The manager of the animals refuge decides a given number of sectors in its property and when an animal is registered it goes into one of these sectors” Questions about the schemas, such as: “Given the reality modelled in schema of figure 1.A), we maintain memory of each animal entering the refuge, giving it a name and a code?”, help the students in fixing the peculiarities of the schema. The answer to this question would be negative because only animals of species having assigned a sector can be registered.

Then the “final” data schema of the considered reality is shown and the students are driven to derive a natural language translation of it. This exercise allows a better understanding of the aspects we want to register in the db, then to judge whether different reverse models can be correct and to change the given schema for making it suitable to users’(in our case students’) specific requests.

From entities and relationships to tables – step 3

(2 hours) In this activity we introduce the relational data model and show the standard way to derive relational tables from an ER schema by driving the students to translate the given ER schema for a library into a relational schema. The following relations are some of the relations in the library relational schema.

LITERARY-WORK(Title, YearFirstPubl, OriginalLanguage*, Classification)
 WORK-AUTHOR(Title, AuthorFamName, AuthorName)
 AUTHOR(FamName, Name, Nation)
 COPY(Title, Where, Language, Publisher*, YearPubl*, YearBought*, Peculiarities)
 WHERE(House)
 FRIENDS(FamName, Name, Surname, Tel, Email)
 ON_LOAN(Title, Where, Date, FamName, Name/Surname)

The above relation WHERE(House) lists the houses where the library owner decides to keep each copy of a book. The relational data model is interesting because it is very simple as for data structures (a table is a set of tuples of simple data values each having a name) and also simple as for operators using tables to compose new tables. This simplicity in the powerful way of producing new information from what is in the tables makes it worth of being known by the students (also: makes it beautiful). It is a good example of a language invented by a computer scientist to build something in between the users and files that students use on their digital devices.

Querying tables is composing sets – step 4

(4 hours) Each table is a set of information and a query specifies a new set of tuples obtained by composing the sets of information explicit in the db. For specifying a query we can either define the characteristic property of the result set using a first order predicate calculus (the tuple relational calculus) or specify how to compose basic sets of data (the tables) into new sets of data using given operators (i.e. specify relational algebra expressions). In our project we use an Office type suite with the QueryByExample type interface to specify queries based on tuple relational calculus. Connections with logics are of course important for the school environment and emphasized in cooperation with professors of mathematics and/or philosophy. One of the queries proposed is the following: “If we have books having Spanish as their original language, find if and where we have copies of them in Spanish”. Different languages are shown for specifying the query in different ways as follows:

L1. $\{C(\text{Title}, \text{Where}) \mid \exists C \in \text{COPY} (\exists L \in \text{LITERARY_WORK} \&$

- (L.OriginalLanguage= 'Spanish') &(C.Title=L.Title)&(C.Language= 'Spanish'))
- L2. $\Pi_{\text{Title,Where}}((\sigma_{(C.\text{Language}='Spanish') \& (L.\text{OriginalLanguage}='Spanish')}$
 $(\text{COPY}) \mid > < \mid (C.\text{Title}=L.\text{Title}) \text{LITERARY_WORK}))$
- L3. SELECT C.Title, C.Where
 FROM COPY as C, LITERARY_WORK as L
 WHERE (C.Title=L.Title)&
 (L.OriginalLanguage='Spanish')&
 (C.Language=' Spanish')
- L4. The Office suite type user interface to a db, in our case LibreOffice, as from
<https://it.libreoffice.org/>

In L1 students use a logical language type they also see in mathematics and/or in philosophy lectures while with L2, L3 and L4 they experiment digital languages. We can suggest that L3 and L4 in the above specification are translated by a database management system (dbms) into L2 and this one is transformed into the following L2-a expression in order to have a smaller cardinality of the sets resulting from sub-expressions and thus a shorter computation time of the full expression.

L2-a $\Pi_{\text{Title,Where}}((\sigma_{(C.\text{Language}='Spanish') (COPY)) \mid > < \mid (C.\text{Title}=L.\text{Title})}$
 $(\sigma_{(L.\text{OriginalLanguage}='Spanish') (LITERARY_WORK))$

Every day students use normal files while operating on their computers: we can discuss how the relational algebra operators might be implemented using these or similar files or having different structures in order to have bigger efficiency in answering to queries. During this query-step we shall also consider problems that cannot be solved on the given data schema and to the related db because we do not have the needed data. The relation ON_LOAN registers one loan only for each book. The query: “which is the book we have rented most often?” cannot be answered in our db because we do not have the data to find out the answer. The query is an ill posed problem with respect to the db we have. Obviously a correct interpretation of existing data and of their metadata (i.e. data schemata) is important to derive correct information from the data we have and to avoid ill posed queries.

User requirements analysis – step 5

(2 hours) In db design user requirements analysis is a time expensive activity. In this project we only discuss examples because our goal is not to teach students how to design, for which we have no time nor we think db design of first interest for these students. We rather aim at showing through an example how a designer shall analyze the description of the reality the users ask to consider in the db for detecting ambiguities and inaccuracies. This activity may show to some teachers and to students that the exegetic analysis on living and lost languages documents, peculiar to their Lyceum, is also needed in Informatics, as perhaps few of them would expect. Thus, this becomes another activity showing that Informatics profits from attitudes that students are expected to develop in secondary schools focusing in humanities.

Abstraction levels in db design and in db management systems – step 6

(4 hours) In this conclusive step we point out critical aspects of the activities concerning db design and of a dbms that we have not yet considered though they are unavoidable for they guarantee the data safety, particularly privacy, concurrency, efficiency and recovery so that real db applications can work properly. Some fundamental ideas in dbms are considered such as the transaction concept described as one of the key elements in Informatics by MacCormick, in his 2011 book.

Most relevant aspect of this conclusive step is going back to our motivations leading to choose db design and querying techniques as an experience to propose to a Lyceum. On one side these activities are important to grasp the attention of the students interested in learning computer science aspects that can be useful for their future and to make them find that these aspects are nearer than they would expect to their attitudes and competences. But we shall not limit this experience to learning how to use a tool such as a dbms. The described db activities allow us to emphasize one of the important principles of Informatics as a science: the different abstraction levels through which a complex problem is solved. During the last activity we particularly focus on the different levels of a db design (user, conceptual, logical and physical levels) and on the abstraction levels of a db system architecture. By showing how in db design and in a db system the complexity of the problem is hidden through levels of abstraction we give a concrete example of how, in general, most computer application are build: by layers of abstraction, hiding away details not proper of a level and trying to reduce and factoring out details so that the programmer/user can focus on a few concepts at a time (from Wikipedia, Abstraction (computer science)). We sketch how abstraction layers in db design and db systems are composed into a hierarchy of levels, each layer encapsulating and addressing a different part of the needs of the design or of the system, thereby reducing the complexity of the associated solutions particularly for the end users.

CONCLUSION

We have been proposing different informatics activities in different schools for years. Educational robotics has been offered in Italy from kindergarten and primary schools as describe by different authors. Examples of activities are in the 2008 paper by De Michele, Demo and Siega. Recently, activities resulting from the experiences introducing to programming were shown during the Teachers for Teachers (T4T) workshops from 27th to 29th September 2012. T4T was a three days hands-on meetings where teachers showed to other teachers, from all types and levels of schools, a number of programming activities developed with their students during the previous three years using different programming languages such as EasyLogo by Lubo Salanci (2010) and well known Scratch, S4A (Scratch for Arduino) and Python. For this last language T4T has been likely the first Italian experience using the Guzdial and Ericson's approach, from their 2010 book. Yet, forcing Informatics to be only or mainly programming is dangerous because many still consider programming as equal to coding. I.e. we risk reducing it to technicalities that are not interesting for the education and can even be boring for students and teachers.

Still, the question: "what kind of informatics shall be present in schools, particularly in secondary schools" has no satisfactory global answer and the research is in progress. In the activity here presented students learn to interact with a simple db. They also learn that some activities of a computer scientist, for example of a db designer, require the kind of abilities they acquire during their Lyceum years. They see that a computer scientist is an inventor of languages, of software architectures built by levels of abstractions, each level solving its problems, i.e. having its proper functionalities.

The experience described in this paper has been proposed in two different classical secondary schools and is under development as an activity both for introducing students to some Informatics education, where usually there is none, and also for discussing with teachers, while activities are developed, whether the proposal has the connections we hypothesize with the attitudes of a classical type of school. A questionnaire in this sense is also designed for the students and will be distributed during the last meetings. Yet we trust in somehow positive answers since a group of teachers has supported our proposal or at least has been curious and is attending the activities and waiting for the conclusion.

A similar experience would be interesting also for university courses related to Medicine, Law, Ancient Languages, Foreign languages or Translation. Indeed, this project has been well accepted since the beginning because a number of professors and the schools principals feel its achievements are important for the education of their students mainly choosing the university courses just named as their careers.

Nevertheless it would be positive to propose the experience also in technical and vocational secondary schools. In technical schools with Informatics specialization the described activities would supplement strictly technical db aspects already covered and refine them with the aspects related to logics and to the attention for user requirements. It would represent a time for reflecting and see from different perspectives the several characters of a db course that in these schools are sometimes neglected to consider mainly the technical perspectives.

In the technical or vocational schools not having a computer science specialization, a similar experience shall be introduced particularly to enrich what is done about databases in basic certification type of courses, popular in secondary schools as an optional achievement.

In conclusion, though this experience is felt particularly near to the attitudes of students in Lyceum type of secondary schools, we are proposing it to all types of secondary schools because an introduction to querying databases and reading data schemas to know what a db contains, shall be part of Informatics Education for all in the next future. For this reason we are teaching a course having about the same organization of the project here summarized both to future teachers of Informatics in secondary higher schools and to future teachers of Technologies in secondary lower schools.

ACKNOWLEDGMENTS

Thanks to the working group "Informatics in schools" of the Informatics department of the author's University: its members are active toward introducing Informatics principles in schools and in discussing which computer science subjects other than programming are most suited for the different types of schools. Thanks to future teachers of Informatics in high secondary schools currently attending a course about the activities here described.

Many thanks to the not many professors of the classical secondary schools who are working with us cooperating in a new project with problems not yet solved.

REFERENCES

- Antonisch, P., K. (2006). Databases as a Tool of General Education, R.T. Mittermeier (Ed), In *Proceedings of the ISSEP Conference*, LNCS 4226, pp 59-70.
- Atzeni, P., et all. (1999). *Database Systems. Concepts, Languages and Architectures*, McGraw-Hill.
- Baruck, S. (1985). *L'âge du capitaine. De l'erreur en mathématiques*. Seuil, Paris.
- Brinda, T.; Puhlmann, H.; Schulte, C. (2009). Bridging ICT and CS - Educational Standards for Computer Science in Lower Secondary Education. IN *Proceedings of the 14th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, ITiCSE 2009*. URL: <http://www.inf.fu-berlin.de/inst/ag-ddi/docs/bridgingICTandCS.pdf>
- De Michele, M.S., Demo, G. B. and Siega, S. (2008). A Piedmont SchoolNet for a K-12 Mini-Robots Programming Project: Experiences in Primary Schools, in *Proceedings. Terecop Workshop, First SIMPAR Conference*.
- Europe Open Data. (2013). <http://open-data.europa.eu/open-data/>
- Guzdial, M., Ericson, B. (2010). *Introduction to Computing and Programming in Python, A Multimedia Approach*, 2/E, Pearson.
- LibreOffice (2013). <http://www.libreoffice.org/>
- MacCormick, J. (2011). *Nine Algorithms That Changed the Future*, Princeton University Press.

- Obama, B. (2009) 2009 Memorandum,
http://www.whitehouse.gov/the_press_office/TransparencyandOpenGovernment
- Salanci, L. (2010). EasyLogo – discovering basic programming concepts in a constructive manner, *Proceedings Constructionism 2010*, Paris.
- Stroedter, C. (2012). Data modeling and database systems as part of general education in CSE, *Proceedings WiPSCE*, Hamburg.
- T4T-Teachers for teachers workshop. (2012): <http://www.di.unito.it/T4T/2012/> and <http://orientamento.educ.di.unito.it/login/index.php>
- Wilson, C., Sudol, L.A., Stephenson, C. and Stehlik, M, (2010). *Running on Empty: the Failure to Teach k-12 Computer Science in the Digital Age*, ACM & CSTA Report, <http://www.acm.org/Runningonempty/>.

Biography



G. Barbara Demo coordinates the working group Informatics and schools of the GRIN, Italian association of the researchers and professors of Computer Science in Science University Schools. She has been working with teachers and students in all types and levels of schools in order to promote Informatics as a discipline.

Copyright

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/3.0/>