Clinical Data Retrieval:  
25 Years of Temporal Query Management at the University of Vienna Medical School

Wolfgang Dorda, Walter Gall, Georg Duftschmid

Section of Medical Information and Retrieval Systems  
Department of Medical Computer Sciences  
University of Vienna Medical School

Corresponding author:

Prof. Wolfgang Dorda  
Department of Medical Computer Sciences  
University of Vienna

General Hospital  
Spitalgasse 23  
A-1090 Vienna  
Austria  
Tel.: +43-1-40400 / 6699 (6698)  
Fax.: +43-1-40400 / 6697

E-mail: wolfgang.dorda@akh-wien.ac.at
Abstract

Objectives: Today, many clinical information systems include analysis components which allow clinicians to apply a selection of predefined statistical functions that satisfy typical cases. They are mostly too inflexible to handle complex, non-standard problems, however. The focus of this paper, therefore, is to present an approach that enables clinicians to autonomously create ad hoc queries including temporal relations in an interactive environment.

Methods: We developed the query language AMAS, which was specifically customized for users from the medical domain to flexibly retrieve and interpret temporal, clinical data. AMAS provides for a significant temporal expressiveness in data retrieval using time-stamped clinical databases and relies on an operator-operand concept for the specification of a query.

Results: Within the last 25 years, four different clinical retrieval systems have been implemented at the Department of Medical Computer Sciences, based on the AMAS query language. Currently, these systems allow access to the medical records of more than 2 million patients. Physicians of 46 different departments at the University of Vienna and Graz Medical Schools have made extensive use of these systems in the course of clinical research and patient care, executing more than 10,000 queries per year.

Conclusions: We discuss a list of 20 issues that represent the most essential lessons we have learned in the development of the four systems mentioned above. Amongst others, our experiences indicate that the operator-operand concept allows an intuitive specification of complex, temporal queries. Further, customization to different user classes, based on their statistical background, is essential.

Key words
Clinical Retrieval System, Temporal Querying, Time Relations, Temporal Operators
MESH: Information Storage and Retrieval; Medical Informatics Applications
1 Introduction

The use of clinical information systems has substantially increased over recent years, leading to the introduction of electronic management systems for medical records. These systems are designed to achieve the following objectives:

a) Support in patient care

Generating a rapid overview of past findings not only streamlines workflows but also facilitates the assessment of a patient’s clinical status. The possibility to effectively identify risk factors greatly improves the quality of patient care.

b) Retrieval of clinical data

Data retrieval by clinical information systems has far-reaching implications for statistical analysis in the scientific arena (e.g., retrospective studies) and lays the foundation for resource planning both within the hospital (capacity utilization statistics, etc.) and for healthcare planning on a large scale [1].

As clinicians increasingly rely on electronic data for medical quality assurance and statistical analysis, there is a growing need for appropriate tools that are effective in the hands of non-expert users. Experiences with clinical information systems show that retrieval tools have to provide users with the possibility of cross-patient searches purposes as well as fast access mechanisms.

Commercially available clinical information systems offer a range of predefined procedures that cover most common querying requirements. However, these modules tend to be too inflexible to allow the formulation of complex, non-standard operations. This is where customized tools come in that enable clinicians to define their own database queries in an intuitive environment. Surprisingly, however, the majority of commercially available clinical information systems offer no or only inadequate support for such ad hoc queries, so that their potential for data retrieval and interpretation remains seriously restricted.
**Temporal issues of data retrieval**

Diagnostic, therapeutic and laboratory data tend to be meaningful only when viewed against their temporal background. Therefore, it must be possible to retrieve data based on their temporal relationships, as in:

Example 1:

“Return all patients showing complication Y within 7 days after surgical procedure X”.

Figure 1 shows the application of the above query to an excerpt from a patient record.

![Figure 1](image)

Other query types required in a clinical retrieval system include:

Example 2:

“Return all patients with
diagnosis X AND
surgical procedure Y *during the same period of hospitalization* AND
complication Z *within 3 month after surgery*”

Example 3:

“Return all patients who experienced symptom X

*between 1.1.1999 and 31.12.1999
around 2 hours after lunch
for 30 minutes*”

While clinicians have a natural interest in solving these and similar problems, their attempts are usually doomed to failure, as currently available systems do not support these temporal functions in the hands of non-expert users.

The present work, therefore, explores the temporal aspects of clinical data retrieval and analysis. Our interest focuses on methods enabling medical personnel to create temporal que-
ries in an interactive environment, which naturally presupposes that the user interface must be suited for clinicians to independently define ad hoc queries.

In the following sections we shall first revisit some of the approaches to temporal data retrieval described in literature. But these temporal query languages and database management systems with sophisticated internal database representation of time (including time intervals) are not very adequate in a practical clinical environment, where only a simple time-stamp-based database architecture is suitable (as pointed out by Nigrin [2]).

The goal of our own query language AMAS described in the third chapter, is to achieve a significant breadth of temporal expressiveness using time-stamped clinical databases. AMAS is based on operands (corresponding to patient groups characterized by one or more clinical features) and operators (logical, comparison and arithmetic operators, as well as special aggregate and selection operators, see below) [3].

In this paper, we focus on the practical implementations of the AMAS language, resulting in a suite of data retrieval systems (described in chapter 4) developed for and successfully employed throughout the University of Vienna Medical School, which is one of Europe’s largest hospitals. By the help of these systems, clinicians were enabled to autonomously process thousands of clinical queries with respect to the course of disease. Motivated by short response times, they intensively used the system's functionality of interactive data retrieval and the associated new options of exploring clinical data. Looking back on 25 years of continuous operation, we shall review our experience with these systems and discuss our insights in chapter 5 “Evaluation and lessons learned”.

2 Related work: Time concepts and temporal query languages in medicine

What follows is a short discussion of temporal characteristics issues in the medical domain and an illustration how these are supported in various existing systems. Concepts of temporal reasoning and temporal data maintenance in medicine are described in more detail by Combi and Shahar [4].
2.1 Time Concepts and corresponding query languages

a) Time points versus time intervals

One recurrent topic of discussion concerns the issue of interval versus time point models. Most clinical databases and query languages do not support intervals as primitive elements but use them as constructs (e.g. Arden syntax [5]). Query languages that do support intervals (e.g. TimeLineSQL) are perforce associated with an interval-based design of the underlying database system [6], [7]. In this case, time points are represented by using identical START and END parameters.

b) Absolute versus relative temporal reference

Absolute temporal references are based on a timescale ("Symptoms started on 5.2.1994, 10:00 p.m."). They enforce an absolute order of elements and can be managed and queried with relative ease. Relative temporal references, by contrast, are based not on a timescale but on reference events ("Symptoms started during the holidays") and do not allow an immediate recognition of the element’s order. In medical contexts, relative temporal data frequently are more interesting than absolute ones.

Most clinical information systems store temporal data by means of absolute references. However, since clinically relevant questions often involve relative temporal components, they require algebraic translation to be specified. The TQuery language, for example, offers context-sensitive mechanisms to define relative temporal information [8]. It hereby enables end-users who are unaware of the absolute dates involved to incorporate clinically relevant reference points directly into their queries ("last visit", "last cycle of chemotherapy", etc.).

c) Time granularity

Sometimes it is necessary to merge heterogeneous clinical data that cover a variety of both explicit and implicit temporal information. Temporal data may be given in different levels of granularity (e.g. "The accident occurred in 1985" versus "The accident occurred on 12.7.1985 at 8:00 p.m.") and thereby cause both sequencing and semantic problems [9]. Handling these different granularities is an important tasks as it builds the foundation of temporal maintenance as well as temporal reasoning. The GCH-OSQL language, introduced by Combi [10], supports mixed temporal granularities. Combi further shows, how temporal granularity can be handled by object-oriented modelling and querying methods [11].
**d) Temporal uncertainty**

In the medical domain, one is frequently confronted with temporal uncertainties introduced by vague patient reports (e.g. „I first noticed the rash sometime around early August 1998“) or by integrating data with different time granularities, to name but two examples. Conversion from a lower to a higher granularity always creates temporal uncertainty. A temporal query system that provides extensive support for the temporal indeterminancies is the Tzolkin system [12]. As another example, in [13] a conceptual data model for the representation of the temporal aspects of clinical symptoms has been developed that includes their temporal uncertainty.

**e) Qualitative versus quantitative relationships**

Relationships between two events can be specified either in qualitative ("Symptoms started after the vacation") or in quantitative ("Symptoms started two to three weeks after the vacation") terms. A key approach to qualitative temporal relationships - alongside the time point algebra [14] - is Allen’s interval algebra based on 13 different relations [15]. Temporal constraints imposed by quantitative temporal relationships are usually specified as intervals to express an uncertain distance between two events.

### 2.2 Temporal Medical Query languages

**a) Extended SQL-Languages**

Despite considerable research efforts in the domain of temporal database management over the past 20 years [16], only few generalized temporal systems have been implemented. Most existing temporal query languages are extended standard languages (e.g. SQL → HSQL). The following sample query in TSQL [17] gives an impression of how SQL is temporally extended:

Example:

*Return the last admission date of all patients, who were admitted within the last 5 years*

```sql
SELECT (Patient-ID, TIME-START) LAST
FROM Admission
TIME-SLICE year [NOW – 5, NOW]
```
A consensus temporal extension of SQL-92, designated TSQL2, was introduced in 1994 [18]. Up to 1998, many of the temporal concepts from TSQL2 have been incorporated into SQL3 [19]. Because of different circumstances, SQL vendors were not enthusiastic about this development and there were disagreements about the incorporated temporal capabilities. A new standard, the temporal extension of SQL3 (SQL:1999) is expected for 2002 [20].

In chapter 2.1. we gave some examples of the use of such temporal-SQL-extensions in the clinical context. But while temporal SQL query languages encompass powerful vocabularies enabling expert users to formulate complex queries, they are, like most underlying database query languages, too abstract and formal to be employed by clinical end-users.

b) Query languages for the Clinicians

Different systems have been developed to access clinical databases by means of temporal queries. In the following, we give a short overview of several interesting approaches:

A useful approach for an end-user query system was presented by Banhart and Klaeren [21]. In an effort to define object-oriented user views of complex relational data structures, they developed a query generator embedded in an object-oriented data model that is visualized by directed graphs. Temporal relationships are flagged with a set of operators to search for first or last instances, for example. This concept greatly facilitates ad hoc queries.

Nigrin et al. [2] shows that temporal queries of time-stamp based documents can be very expressive. Using their “Dxtractor” system, the clinical user can first define groups of patients, who show specific clinical findings. These groups can then be linked by logical operands (AND, OR, NOT). Further, temporal operators such as BEFORE, AFTER, EQUALS, EARLIST and LATEST can be used. By means of the BY and WITHIN-descriptors, the lower and upper bound on the temporal relationship between two points events can be specified as well. The process of linking individual patient groups through operators creates new patient groups. A problem hereby is that the temporal information of both operands of a logic AND operation, for example, is taken over to the result set. In this context, Nigrin states that “temporal information contained in result sets, even of a simple Boolean AND operation, is not meaningful”. Consequently, the Dxtractor system only allows the formulation of the temporal relation between two operands, as the result set does not contain meaningful, temporal information. Such result sets may only be processed further by non-temporal operators.
Systems, which support temporal intervals, were among others developed by the Stanford group. They have done important work in the tasks of temporal abstraction (e.g., within the RÉSUMÉ system [22]) and temporal maintenance (e.g., within the Chronus system [23]). Additionally, they combine these both functions within the Tzolkin module [24] and call the whole architecture a *temporal mediator* [25].

The state of the art of temporal query management in the medical domain may be summarized as follows:

There are some interesting suggestions and prototypes for temporal expansions of database query languages, which are SQL expansions mostly. However, these temporal query languages are only commercially available in a very restricted extent. In addition, they are too complicatedly to be applied by clinicians and therefore not meaningfully usable. User-friendly clinical query systems thus continue to represent a considerable challenge for developers.

### 3 The Query Language “AMAS”

#### 3.1 Basic Data Model

In the clinical routine, most medical events are associated with absolute temporal references. Accordingly, in our basic data model, time-stamps are linked to the findings and integrated in the retrieval process using the AMAS time logic.

Additionally, the AMAS language uses a data model that includes various time attributes with different semantics (see Figure 2), all of which can be referenced within a query:

**Birthdate of patient:** Used to calculate the age of the patient at the time of the finding.

**Date of case:** Each medical problem for which a patient receives treatment in the hospital will generate a new case associated with a date.

**Date of document:** For each case, several documents may be created, each of which is associated with a date.
**Temporal variable values:** Each document can contain variable values of different types, such as DATE, TIME or TIMESTAMP.

---

**Figure 2**

### 3.2 Time Operators

Our query language AMAS is based on logical operands and operators for data selection. Operands may be single variables or partial query results and may be further constrained and linked by unary and binary operators. Several operators are provided to specify temporal constraints. What follows is an overview of the various types of time-related AMAS operators (compare Table 1):

<table>
<thead>
<tr>
<th>a) Binary operators</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>a.1) Binary Operators with temporal conditions:</strong> The logical operator “AND”(^1), and all binary arithmetic and comparison operators can be extended by temporal conditions. These binary temporal operators are specified by an operator symbol, a temporal grouping specification and optional arguments.</td>
</tr>
</tbody>
</table>

**Example:**

```
“surgery X AND within 7 days complication Y”
```

<table>
<thead>
<tr>
<th>operator symbol:</th>
<th>&amp; days 0:7</th>
</tr>
</thead>
<tbody>
<tr>
<td>grouping specification:</td>
<td>days</td>
</tr>
<tr>
<td>argument:</td>
<td>0:7</td>
</tr>
</tbody>
</table>

**a.2) Binary Temporal operator TIME DIFFERENCE:** calculates the difference in various granularities (e.g., day, month, minute) between two time variables.

---

\(^1\) The logical "OR" does not allow a meaningful definition of temporal relations.
b) Unary operators

b.1) The unary selection operator consists of the symbol “$”, a grouping specification, and one or more arguments.

Example: ”Data of the last patient visit”
$ visit last
operator symbol: $
grouping specification: visit
argument: last

There exist three types of specifications:

- Selection: The unary operator TEMPORAL ORDER selects all data of the grouping (patient visits, documents, ...) according to their temporal sequence (last, first, ...) or further constrains grouped data.

  The unary operator DATE SELECTION selects temporal variables by specifying temporal constraints, taking into account time granularities (“Return all data referring to February 2000”).

- Functions: They aggregate the data of the grouping to a single function (like mean, median,...). The result is the function-value.

- Condition: Selects all the input-data of the grouping, if the specified condition is satisfied (e.g.: “Return all patients’ data with an average systolic blood pressure higher than 160 - over a period of two hours”).

b.2) The unary constant operators enable comparisons or arithmetic calculations of the input data with a constant value. As an example, the unary constant operator +5 increases each input value by 5.

3.3 Logical Queries

The temporal operators described above can be interactively incorporated into any database query within the AMAS retrieval systems in order to define ad hoc queries. The following example shows the query resulting from example 2, presented in the introduction, as formu-
lated within ArchiMed, the latest member in the family of AMAS retrieval systems. The query falls into three operands that are linked by two binary temporal operators.

<table>
<thead>
<tr>
<th>Operand 1</th>
<th>Diagnosis X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operator</td>
<td>AND “same period of hospitalisation”</td>
</tr>
<tr>
<td>Operand 2</td>
<td>Surgical procedure Y</td>
</tr>
<tr>
<td>Operator</td>
<td>AND “month difference 0:3”</td>
</tr>
<tr>
<td>Operand 3</td>
<td>Complication Z</td>
</tr>
</tbody>
</table>

To avoid bothering clinicians with syntactical issues, the AMAS systems feature user-friendly interfaces that allow queries to be formulated in an intuitive manner with minimized visibility of technical details. All operators can be retrieved from selection lists. Figure 3 shows, how the binary temporal operator “Difference in month” can easily be formulated in ArchiMed [26].

**Figure 3**

The system automatically executes this query by performing the following actions in the background:

- Retrieve all patients suffering on “diagnosis X”
- Retrieve all patients with ”surgical procedure Y”
- Execute the operator module “AND (same period of hospitalisation)”: To be able to meaningfully process the time information in the result set, the time information is transferred only from one operand. As an example, only the attributes of the second operand are taken in the result set for the logical AND operator in the default case [3]. Consequently, when linking the operands X and Y, only the timestamps of the second operand Y are passed on. Hereby, the result set can be meaningfully linked with the third operand Z with respect to the temporal relationship “and within 3 months”. In this manner, temporal sequences between any number of operands can be specified. Thus, AMAS allows the user to formulate entire temporal sequences including any number of individual events.
- Retrieve all patients with ”complication Z”
- Execute the operator module “AND (month difference 0:3)”
Technically speaking, each operator consists of two program modules, one for formulating and one for generating queries. The generation modules are successively invoked by a query generator which compiles the formulated query into an SQL statement. The fact that query formulation is dissociated from query generation ensures that it can easily be customized, for example by setting up multiple interfaces optimized for different user requirements.

Advantages of the above approach based on operands and operators are (i) its reusability, in that all elements of a query and its results can be saved and reused, and (ii) its expandability, in that new operators can easily be added to the system.

### 4 Evolution of the AMAS-Systems

For more than 25 years, the systems of the AMAS family [26], [27], [28], [29], [30], [31], described in the following have enabled clinicians to exploit their data from the clinical routine for explorative statistical purposes. The main objective in developing these various systems has always been to put theoretical results from the domain of temporal reasoning to good use in a practical clinical environment.

**a) Batch-WAMAS**

The AMAS language was originally developed for WAMIS, a clinical information system that was introduced at the University of Vienna Medical School in the early seventies [32]. This system relies on updates of patient records at varying time intervals, depending on the collection of data during clinical routine.

In the mid-seventies, the requirement to make these medical data available for scientific research by rendering them usable for explorative statistical analysis led to the introduction of the WAMAS [27] temporal retrieval system. Although at the time it was still state of the art to define query conditions on punch cards, the early WAMAS system already provided the option of defining queries that included temporal synchronizations of clinical data (e.g., “Search for all patients showing complication B within X days after intervention A”).

The system was in routine daily use at the University of Vienna Medical School all the way up from 1976 to the early 1990s and provided the basis for countless scientific studies.
b) Interactive-WAMAS

In 1979, the WAMAS system was redesigned and brought to a basis of interactive query management [28]. This step was made possible by the ongoing technological progress in electronic data processing, which did not equally cover all areas, however: Rather than using a relational database, which was not commercially available at the time, it was necessary to build an index to the data within the file system and update it on a daily basis.

In the course of the system’s redesign also the expressiveness of the temporal synchronization operators was extended: Besides operators using absolute time differences (“up to 7 days after the event”) it was now also possible to consider the individual medical context by specifying patient-specific intervals, (e.g., “Search for all patients with Diagnosis A AND complication B during the same period of hospitalisation”).

The system provided effective mechanisms for user guidance, so that clinicians could independently conduct queries and thereby gain a more profound access to their data. This simplified access to formulating temporal conditions was widely accepted by clinical users.

c) WAREL

When commercial SQL databases became available in the 1980s, the IBM DB2-based WAREL system was introduced [29], [30]. This system serves as a clinical data warehouse by collecting medical data from two clinical documentation systems (WAMIS and KIS, the Hospital Information System of Vienna’s general hospital) and one laboratory information system. WAREL’s enhanced functionality included synchronization intervals not only for logical but also for comparative and arithmetic operators. The flexibility of the system was increased by specifying the operators in a “knowledge base”: This means that all attributes of the operators like priority, valency, necessary input parameters, sorting after attributes, and result attributes are managed in tables. New operators - even if they were not contained in SQL (e.g., operator LINEARLY DECLINING in temporal aggregates) - could be added by simply entering the required characteristics. The system also offers, via “semantic query checks”, intelligent assistance in formulating queries and complex text analysis for text retrieval.

d) ArchiMed

In the 1990s, the AMAS language was implemented in the client-server application ArchiMed [31]. This system, which is being employed by the University of Vienna Medical
School and by ten departments of the university clinic in the city of Graz, features a graphical user interface and provides further user support enhancements. Aside from search queries, the ArchiMed system is also suited for rule-based screening based on defined trigger events. This means that patient records are continuously checked for predefined temporal conditions that will trigger specific activities such as alert messages being displayed, print jobs being started, or alert reports being updated. For example, we may want to define an alert message for patients with condition X diagnosed within the last two years showing a greater than 5% rise in laboratory parameter Y within three days. In this way, the query language AMAS also lays the foundation for rule-based screening systems capable of reflecting temporal developments.

5 Evaluation and Lessons learned

The AMAS query language was developed specifically for users from the clinical practice to retrieve and interpret clinical data. Clinicians are hardly versed in database management, so it is not an easy task for them to formulate logical conditions. Translating a clinical query into logical expressions is a complex procedure that requires appropriate training. The intuitive and user-friendly interface of the AMAS systems greatly facilitates this type of work in the hands of non-expert users. Even the use of complementary temporal conditions does not seem to add an extra dimension to the complexity of formulating queries. These conditions have been used extensively, and they have ensured that countless clinical queries could be processed in a meaningful manner. Defining the temporal sequence of the various operands is not too difficult a task when users can visualize this sequence in a “natural” order. Therefore, clinical users at the Vienna and Graz university clinics have made extensive use of the AMAS systems to retrieve data reflecting the clinical course of their patients (table 2).

Our practical experience based on the extensive use of the AMAS temporal query language by clinicians can be summarized as follows:

**General requirements:**

1. **Authorization Levels** — There must be a way to define which users, or user groups, are authorized to access specific functions (e.g., number of returned patients, anonymous sta-
2. **AD-HOC QUERIES** — A great many clinical queries cannot be foreseen as they only arise in a specific context, so it must be possible to formulate *ad-hoc* queries based on any of the available variables.

3. **INTERACTIVITY** — The system must offer short response times so that clinicians can familiarize themselves and analyze their clinical data in a truly interactive environment. Any delayed (e.g., overnight) or low-priority processing of queries will be regarded as inadequate.

4. **SAVING OF QUERIES** — The option to save and re-execute logical queries or to retrieve previously selected patient groups or clinical data at a later time has been used extensively.

**Usability:**

5. **INTUITIVE INTERFACE** — Clinicians need to gain a good overview of their clinical data. Since the use of retrieval systems is not part of their regular activities, it is extremely important that the user interface is optimized for intuitiveness. In the ArchiMed system, all specifications for a query can be chosen from predefined selection lists.

6. **DEFAULT ASSUMPTIONS** — Typical query formulations are simplified in the AMAS systems by numerous default assumptions, which can be output along with the queries. Clinicians feel that this “annotated” representation is extremely helpful when it comes to formulating more complex queries.

7. **SYNCHRONIZATION OPERATORS** — In order to facilitate temporal queries, the various operators (binary, logical or comparative) were extended to “temporal synchronization operators”, such that clinicians can formulate synchronization conditions by just adding them to the operators. Placing these temporal operators between the two relevant operands has proved to be a very intuitive way of formulating temporal conditions in the clinical practice. Users get a realistic look and feel of temporal relationships, as the temporal sequence of the various operands is defined in a “natural” order.

8. **SEMANTIC CHECKS** — Another very helpful feature in formulating queries is the “semantic check” of temporal conditions. For example, the system recognizes impossible queries.
such as “All patients whose mean cholesterol level while in hospital was greater than 120 and who had complication X three days later”. As the period of hospitalization can no longer be linked to a specific date, the subsequent condition “three days later” does not make sense. The system recognizes such semantic errors and supports users by pointing out the sources of error with great accuracy.

9. **User Support** — In addition to the advanced user interface and semantic checks, the great acceptance of the AMAS systems among clinical users is also due to the high availability of training courses, help lines, user manuals and application-oriented sample collections.

10. **Customized User Interfaces** — Our retrieval systems are customized for two different user groups:

   *Clinical users.* The AMAS data retrieval systems were specifically designed for clinical users to independently formulate queries via advanced prompting routines and comprehensive help functions. With the help of numerous default settings, queries can be formulated quickly and with relative ease using just a few mouse clicks. Furthermore, the system design was optimized and complex processing routines instituted to speed up interactive querying.

   *Experts.* Special requirements can be handled by using optional specifications and changing the default operator settings. Furthermore, the system environment offers excellent support for incorporating SAS programs into the interactive queries via user exits, enabling expert users who are familiar with the SAS language to introduce system enhancements for special queries in the form of predefined process routines.

**Patient selection:**

11. **Logical Temporal Conditions** — Patients are selected via their clinical data by formulating logical conditions. Since clinical data can only be adequately interpreted in their temporal context, clinical users must be able to include temporal conditions for patient selection. Furthermore, there is a need for complex temporal operators beyond the scope of standard SQL (e.g., “continuously falling”).

12. **Formation of Temporal Data Groups** — Temporal references used in data group formation may include dates or administrative criteria (e.g., period of hospitalization) or data
pertaining to the clinical context (e.g., treatment cycle) to formulate selections such as “First document of the day”.

13. Moving time windows — In AMAS it is also possible to formulate temporal group clauses for aggregate functions, e.g. to select “all patients with a median value greater than X, where the median is calculated from all data of three consecutive days”.

14. Text retrieval — The AMAS language was not least developed for text retrieval and includes an extensive range of search modes covering all types of clinical texts. It allows text comparison based on “string” and “lexical base” (see [33]) and also addresses factors like word order. These features too are being routinely and successfully used throughout the University of Vienna Medical School. The research into the “lexical base” search turned out to require too many resources, considering the complexities of the German language, and was therefore not developed further.

15. Expressiveness of the retrieval language — The AMAS language as summarized in Table 1 satisfies all querying requirements in our clinical environment. There has been no more need to expand the syntax over the past few years.

Information on selected patients and further processing of their data:

16. Hit lists and patient lists — Clinicians first need an overview of hits, i.e. of selected patients in terms of how often the conditions formulated in the query were met. The AMAS systems employ various counting methods in returning the number of patients, visits or documents meeting the query conditions. They also include the option of automatically calculating age and sex distributions, which has proved to be a very useful feature. Furthermore, there must be a way for authorized users to generate lists with the names of patients or address labels, etc.

17. Clinical patient data — Users naturally have the option to generate reports from the clinical data of selected patients, and they can select patients from a returned list and then interactively switch to the clinical information system. This window opens a more detailed view of the clinical data on record for a patient and thus greatly facilitates in-depth study and follow-up documentation of specific cases.
18. **DATA COLLECTION FOR STATISTICAL ANALYSIS** — Before statistical indicators such as histograms, x/y plots or correlation coefficients can be calculated, the retrieval system has to collect the relevant values for the selected patients. The requirement that these values must be selectable from temporal sequences (e.g., “first” or “last”) implies that the same operators as presented in Table 1 must be available to confine the statistical analysis to specific values, as in “first value after hospitalization”. Furthermore, temporally related (“synchronized”) data must be selectable for multivariate analysis, which was achieved by developing a special **JOIN** operator combining multiple variables, as in “minimum time difference, but only up to a difference of X days”.

19. **EXECUTION OF STATISTICAL ANALYSIS** — As another popular feature, clinical users can invoke common statistics functions (e.g., standard tests, descriptive or survival statistics) right from within the retrieval system. The data are internally handed over to a statistics system (SAS) that will run the appropriate procedure and display the results. Expert users can go even further and use SAS commands to execute highly specialized statistical procedures from within the system. The systems also features an export function for statistics software packages.

20. **STATISTICAL EXPERT SYSTEM** — Attempts to include user guidance for explorative statistical data analysis in terms of a “statistical expert system” have been moderately successful at best. Although a system designed to support clinicians in selecting appropriate statistical methods was developed [34] and extensively used for some time, access to this functionality has since sharply decreased. Similar observations have been reported for other “statistical expert systems”.

### 6 Conclusion

Clinical information systems have become an essential tool to support medical research within the last decade. However, many system’s abilities in the domain of data retrieval are still limited by the fact that they do not provide flexible mechanisms for the statistical analysis of patient data. Most systems offer a number of predefined queries, but only few of them support *ad hoc* queries.
Clinical data can only be meaningfully analyzed when temporal relations are taken into account. Today, most retrieval systems are based on extensions of the query language SQL and therefore require intuitive interfaces to be accepted by clinical end-users. Aside from individual prototypes, there exist only few clinical retrieval systems at present that permit *ad hoc* queries by the clinicians themselves reflecting temporal relationships.

This was the rationale for developing the AMAS query language and associated systems presented in this work. Being optimized for clinical requirements by enabling *ad hoc* queries reflecting the clinical course of patients, they have been used routinely at the Vienna and Graz university clinics for many years. The experiences, we gained from 25 years of cooperation with clinicians in the area of medical retrieval systems and in the advancement of the AMAS language, are compiled in the section “Evaluation and lessons learned”.

**Acknowledgments**

We wish to thank Dr. P. Sachs and DI E. Eigenbauer for the successful cooperation in the area of clinical data retrieval, as well as DI Oeser for the valuable suggestions he made in many interesting discussions and M. Mayrhofer for her support in preparing the manuscript.
References


Fig. 1 The occurrence of complication Y is relevant at timepoint $t_1(Y)$ but not at timepoint $t_2(Y)$, which is not within 7 days after the time of surgery $t(X)$. 
Fig. 2  Time Attributes in the AMAS data model
Fig. 3  The window specifies the binary operator MONTH DIFFERENCE.
Table 1
Overview of the AMAS operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Operator type</th>
<th>specifications</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Binary operators:</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No temporal spec.</td>
<td>logical or</td>
<td>!</td>
<td>OP_A ! OP_B</td>
</tr>
<tr>
<td>Binary Temporal operators</td>
<td>logical “and”</td>
<td>&amp;</td>
<td>OP_A &amp; days 0:7 OP_B</td>
</tr>
<tr>
<td></td>
<td>logical “and not”</td>
<td>~</td>
<td>OP_A ~ days 0:7 OP_B</td>
</tr>
<tr>
<td></td>
<td>comparison</td>
<td>&gt;, &lt;</td>
<td>OP_A &lt; hours -3:3 OP_B</td>
</tr>
<tr>
<td></td>
<td>arithmetic operators</td>
<td>+, -, *, /</td>
<td>OP_A / visit 0:0 OP_B</td>
</tr>
</tbody>
</table>
| Time difference           | -Time                          | Complication –Time Surgical interven- |}

| **Unary operators:**      |                                |                                  |                                              |
|---------------------------|                                |                                  |                                              |
| Selection operator $      | Selection                      | first, second, ..., last time interval | $visit last document                       |
| Function                  | mean, median, standard dev., etc.|                                  | $pat 14 days mean                           |
| Condition                 | log. expression                |                                  | $pat 1 year median 0:10 $pat 3 days continuously falling |
| **constant operators**    |                                |                                  |                                              |
| $                           | comparison operators           | constant value                   | > 30                                         |
|                            | arithmetic operators           | constant value                   | + 3                                          |
Table 2
Characteristics of the retrieval systems of the AMAS-family

<table>
<thead>
<tr>
<th></th>
<th>University Clinics of Vienna</th>
<th>University Clinics of Graz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Departments using query language AMAS</td>
<td>40</td>
<td>6</td>
</tr>
<tr>
<td>Patients</td>
<td>2,200,000</td>
<td>14,000</td>
</tr>
<tr>
<td>Documents</td>
<td>54,000,000</td>
<td>64,000</td>
</tr>
<tr>
<td>Values</td>
<td>750,000,000</td>
<td>1,591,000</td>
</tr>
<tr>
<td>Queries / year</td>
<td>10,000</td>
<td>1,000</td>
</tr>
</tbody>
</table>