A MULTIMEDIA TEMPORAL DATABASE MODEL

By

Nasrat M. Dermaika

A project submitted in partial fulfillment of the requirements for the degree of

M.S. Computer Science

Lebanese American University

2006
LEBANESE AMERICAN UNIVERSITY

Student Name: Nasrat Dernaika
Research Title: A Multimedia Temporal Database Model

We hereby approve the Thesis or Project of candidate for the Master of Computer Science (MS) Degree*

Advisor Signature: 
Dean Signature: 
Date: 20/4/2006

* We also certify that written approval has been obtained for any proprietary material contained there in.
Project Approval Form (Annex V)

Student Name: Nasrat Derailka
I.D. #: 199811110

Project Title: A Multimedia Temporal Database Model

Program: Master of Science in Computer Science

Division/Dept: Computer Science and Mathematics

School: Arts and Sciences

Approved by: Ramzi A. Haraty

Project Advisor:

Member: Faisal Abu Kizam

Date: April 20, 2006
To be signed by the student:

I grant to the LEBANESE AMERICAN UNIVERSITY the right to use this work, irrespective of any copyright, for the University’s own purpose without cost to the University or to its students, agents and employees. I further agree that the University may reproduce and provide single copies of the work, in any format other than in or from microforms, to the public for the cost of reproduction.
ABSTRACT

Multimedia databases are becoming an increasingly important area for research and application in the entire field of database systems, therefore multimedia databases is considered a new topic of research which is still very active and highly demanded. Temporal databases aroused long time ago, since users started requesting databases that support the time dimension feature, still this subject is under progress and research. This paper explores the nature and implications for Multimedia and Temporal database models with an overview about the nature if both models. Furthermore, this paper imposes a description and implementation for a new model; A Temporal Multimedia Database Model. Suggestions for a further work are proposed to cover the gaps discussed concerning this topic. The paper concludes with a brief discussion of the appropriateness and limitations of a temporal multimedia database model till our current time.
# TABLE OF CONTENTS

Chapter 1 – Introduction .................................................................................1
  Section 1.1 - Scope of the Project ...............................................................1

Chapter 2 – Literature Review ........................................................................2
  Section 2.1 – Literature Review for Temporal Databases .........................2
  Section 2.2 – Literature Review for Multimedia Databases .......................2

Chapter 3 - Temporal Databases ....................................................................4
  Section 3.1 Basic Concepts .........................................................................4
  Section 3.2 Different Types of Temporal Databases .....................................5
  Section 3.3 Temporal Relation .....................................................................5
  Section 3.4 Integrity Constraints ..................................................................6

Chapter 4 – Multimedia Database Model .......................................................9
  Section 4.1 Basic Concepts .........................................................................9
  Section 4.2 Different Types of Multimedia Databases ...............................10
  Section 4.3 Fast View on Oracle and Intermedia Package ...........................11

Chapter 5 - Temporal Multimedia Database Model .....................................17
  Section 5.1 Temporal Multimedia Databases .............................................17
  Section 5.2 Definition of Temporal Multimedia Relation ...........................17
  Section 5.3 Integrity Constraints ................................................................18

Chapter 6 – Model Implementation and Example .......................................19
  Section 6.1 TMMDB (Temporal Multimedia Database) ..............................19
  Section 6.2 TMMDB Functionality ..............................................................19
  Section 6.3 Example Queries .....................................................................20

Chapter 7 – Further Work and Conclusion ...................................................30
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 6.3.1: Logging to SQL Plus</td>
<td>20</td>
</tr>
<tr>
<td>Figure 6.3.2: Creating a virtual directory from where the images are imported</td>
<td>21</td>
</tr>
<tr>
<td>Figure 6.3.3: Creating a table CRIME_FILE</td>
<td>22</td>
</tr>
<tr>
<td>Figure 6.3.4: Schema for CRIME_FILE table</td>
<td>22</td>
</tr>
<tr>
<td>Figure 6.3.5: Inserting a crime file record into the CRIME_FILE table</td>
<td>24</td>
</tr>
<tr>
<td>Figure 6.3.6: Querying Media Data from the CRIME_FILE table</td>
<td>25</td>
</tr>
<tr>
<td>Figure 6.3.7: Exporting an image from the CRIME_FILE table to a file</td>
<td>26</td>
</tr>
<tr>
<td>Figure 6.3.8: Querying Media Data from the CRIME_FILE table with temporal restrictions</td>
<td>27</td>
</tr>
<tr>
<td>Figure 6.3.9: Querying Media Data from the CRIME_FILE table with temporal restrictions</td>
<td>28</td>
</tr>
<tr>
<td>Figure 6.3.10: Retrieving the insertion transaction time of the multimedia object using the getUpdateTime()</td>
<td>29</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table                                                                 Page
Table 3.1: A sample data of the crime files' temporal relation..................... 6
Table 3.2: A set of values reflecting the redundancy problem.......................... 7
Table 3.3: The replacement of the two tuples displaying redundancy by one......... 7
Table 3.4: A set of values reflecting the circumlocution problem...................... 7
Table 3.5: The replacement of the two tuples displaying circumlocution by one.... 7
Table 3.6: A set of values reflecting the contradiction problem........................ 8
Chapter 1

INTRODUCTION

Standard Relational Databases are databases that store textual data at a single moment of time. Temporal Databases are used to store time-varying data. Multimedia databases are used to store multimedia data like images, videos, and audios. Each type of these databases serves its own purpose. Our objective here is to develop a new data model that can be used to model temporal multimedia databases.

1.1 Scope of the Project

As we know relational databases store information about the real world and their design reflect real world entities. Relational databases support transactions on textual data only at the current time, where old data is deleted, updated, or ware-housed in new DBMSs. In temporal databases it is bit different, old historical data is kept; it is not deleted or updated. Therefore, temporal databases keep track of historical data. Past, present, and future data can be captured in these databases. Multimedia databases support the storage and retrieval of multimedia data, the multimedia database model is a new developed model, and thus it is totally different than relational databases in the way it is represented, and which we will discuss later in chapter 4.
Chapter 2

LITERATURE REVIEW

In this chapter we review the history of temporal databases, and then we review the history of multimedia databases over the past years.

2.1 Literature Review for Temporal Databases

Relational databases are used to keep data of the current time, where historical and future data are either updated or deleted. The situation in temporal databases is different, temporal databases are used for applications where storage and retrieval of past, present, and future data is required. For example, using relational databases for medical applications to keep records for patients and predict there a medical condition is not enough. Same case stands for advanced payroll applications, university student applications and banking applications. The use for relational model for such application will cause a high data redundancy problem. Here the demand for managing historical and time-varying data started to arise in 1970s in the area of medical information systems, and this interest increased in 1982. The studies done to develop a temporal data model followed two approaches [9]. The first approach is to extend the standard relational data model so that it supports time varying data, and the second approach is based on extending the snapshot model with time appearing as additional attributes. Users over the years begun to increasingly request for temporal database models, therefore, the time dimension has been added to many data models. These models include the entity-relationship model, semantic data models, knowledge based data models, relational data model, object-oriented data model, and deductive databases [9].

2.2 Literature Review for Multimedia Databases

Recently, the computer science society started recognizing there will be an increased demand for non-textual data to be processed. Examples of non relational data are image, audio, video, and document data. These multimedia data types are really challenging to handle through traditionally established areas in computer science. Writings and researches
multimedia databases in the field of computer science are still little, despite their interestingness and huge increased growth in the industry. Multimedia databases are totally different from classical relational databases having multimedia capabilities; this is because they fulfill unique requirements. These requirements are specific type of interactive multimedia, a relatively new class of applications that have also received scant coverage despite their critical importance in the World Wide Web. The term “multimedia” is just a part of a large category of abstract data, examples for that would include applications of X-rays, electro-cardiograms, and fingerprints. Such applications demand massive amounts of data storage. In reality, their storage and processing requirements are similar to those in multimedia applications. Therefore, multimedia databases can usually support most abstract data. In fact, the terms “multimedia” and “abstract data” are interchangeable [9].
Chapter 3

TEMPORAL DATABASE MODEL

Temporal data stored in temporal databases is different from the data stored in non-temporal databases. In temporal databases a time period is attached to the data, this time period expresses when the data stored was valid or saved in the database.

As we have said, standard relational databases consider the data stored in it to be valid and available at our present and current time (now). Therefore, they do not keep track of any past or future data. By adding a time period attribute to the data, it becomes possible to store information related to different database states at different time periods, meaning historical past and future data. Thus, temporal databases are used with environments where storage of historical data is required. Temporal databases do not only contain current data but they contain past and future data which are kept in the database contrary to the case in relational databases.

3.1 Basic Concepts

A standard relation is two-dimensional with attributes and records as dimensions. A temporal relation contains two additional, orthogonal time dimensions, namely valid time and transaction time. Valid time stands for the interval of time during which the record is active or valid. Transaction time records at what time facts are stored in the temporal relation. Valid and transaction time have precise, crisp definitions. If changes to the past are important, then valid time support is required. Transaction time is needed to roll back to a previous state in our database or to retrieve records processed or stored at a certain time.
3.2 Different Types of Temporal Databases

Temporal Databases can be divided into three types based on the two different types of time dimensions used in temporal databases.

1. Rollback Database:

Rollback Transaction Time database is a database that contains only one of the two orthogonal time dimensions, the transaction time. These databases support the recording of past and present data only. Every time a new record is inserted to the database the transaction time of its insertion is recorded.

2. Historical Databases:

Historical or Valid Time database is a temporal database that records only the valid time orthogonal time dimension. It supports the recording of past, present and future data, since the valid time depends on what the user believes. The user can believe that something would be in a certain interval in the future.

3. Bitemporal Databases:

The Bitemporal database is a database that contains both the two time orthogonal dimensions, the valid time and the transaction time. It supports the recording of past, present and future data. Not only it supports the recording of what we believe was true, is true, or will be true but also the recording of the time of when we did believe so.

3.3 Temporal Relation

A temporal relation is of the form:

\[ R(A_1,A_2,...,A_n | \tau^b) \]

It consists of a number of attribute values associated with a bitemporal timestamp value \( \tau^b \). The bitemporal timestamp value \( \tau^b \) is represented by the ordered pair \((c',c'')\) with \( c' \) representing the transaction time and \( c'' \) the valid time.
An example of a temporal relation would be the crime files' temporal relation presented in table 4.1. The valid time interval determines when the information stored in the tuple was valid. From “1953/3” till “1985/8” the crime file for “Burglary” was closed between 1953 and 1985.

Table 3.1: A sample data of the crime files’ temporal relation

<table>
<thead>
<tr>
<th>FILE ID</th>
<th>IMAGE</th>
<th>VIDEO</th>
<th>AUDIO</th>
<th>CRIME DESCRIPTION</th>
<th>FILE STATUS</th>
<th>ValidTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Burglary ....</td>
<td>CLOSED</td>
<td>[1953/3-1985/8]</td>
</tr>
<tr>
<td>101</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Killing...</td>
<td>OPENED</td>
<td>[1983/7-∞]</td>
</tr>
</tbody>
</table>

3.4 Integrity Constraints

Now that we have a good idea of temporal databases, we will talk next about the integrity constraints that we need to meet in order to make sure that all the instances in the database are meaningful. In temporal databases other than the primary and foreign key constraints we have two main constraints. The first constraint is used to solve the redundancy and circumlocution problems, and the second is used to solve the contradiction problem. So before talking about these constraints we need to talk about each of these problems.

**THE REDUNDANCY PROBLEM:** Let us take for example the crime file example once again, with the addition of the attribute valid time. In table 4.2 we present the crime file table along with some sample data. Let us assume that the primary key of this table is the crime file ID in conjunction with the valid time attribute. As we see that will not be enough to prevent the relation from containing the two tuples that are presented in the figure. These two tuples state the information about the crime file 100 twice, while we can simply state this information using only one tuple as is presented in table 4.3. This is what we refer to as the redundancy problem in temporal databases.
Table 3.2: A set of values reflecting the redundancy problem

<table>
<thead>
<tr>
<th>FILE ID</th>
<th>IMAGE</th>
<th>VIDEO</th>
<th>AUDIO</th>
<th>CRIME DESCRIPTION</th>
<th>FILE STATUS</th>
<th>ValidTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Burglary ....</td>
<td>CLOSED</td>
<td>[1953/3-1985/8]</td>
</tr>
<tr>
<td>100</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Killing...</td>
<td>CLOSED</td>
<td>[1983/7-00]</td>
</tr>
</tbody>
</table>

Table 3.3: The replacement of the two tuples displaying redundancy by one

<table>
<thead>
<tr>
<th>FILE ID</th>
<th>IMAGE</th>
<th>VIDEO</th>
<th>AUDIO</th>
<th>CRIME DESCRIPTION</th>
<th>FILE STATUS</th>
<th>ValidTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Burglary ....</td>
<td>CLOSED</td>
<td>[1953/3-00]</td>
</tr>
</tbody>
</table>

THE CIRCUMLOCTION PROBLEM: Once more we can not rely on the specified primary key on the crime file to prevent us from having the following sample tuples in the crime file table presented in table 4.4. Here again we can replace the two tuples by a single tuple as we see in table 4.5.

Table 3.4: A set of values reflecting the circumlocution problem

<table>
<thead>
<tr>
<th>FILE ID</th>
<th>IMAGE</th>
<th>VIDEO</th>
<th>AUDIO</th>
<th>CRIME DESCRIPTION</th>
<th>FILE STATUS</th>
<th>ValidTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Burglary ....</td>
<td>CLOSED</td>
<td>[1953/3-1985/8]</td>
</tr>
<tr>
<td>100</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Killing...</td>
<td>CLOSED</td>
<td>[1983/7-00]</td>
</tr>
</tbody>
</table>

Table 3.5: The replacement of the two tuples displaying circumlocution by one

<table>
<thead>
<tr>
<th>FILE ID</th>
<th>IMAGE</th>
<th>VIDEO</th>
<th>AUDIO</th>
<th>CRIME DESCRIPTION</th>
<th>FILE STATUS</th>
<th>ValidTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Burglary ....</td>
<td>CLOSED</td>
<td>[1953/3-00]</td>
</tr>
</tbody>
</table>

CONSTRAINT A: To fix both the problem of redundancy and the problem of circumlocution we specify the following constraint on temporal databases.
• If at any time a relation contains two distinct tuples that are identical except for their valid time values i1 and i2, then i1 merges i2 must be false. (merges is the logical or of overlaps and meets).

THE CONTRADICTION PROBLEM: Even with the previously stated constraint and the primary constraint there is still a problem that we might come across in temporal databases. Let us take for example the sample values of the crime file table presented in table 4.6. Here the crime file has ID number 100 on the date 1985/7 for example would have both the states “CLOSED” and “OPENED”. It is impossible for a crime file to have two different states at the same time because this would create a contradiction.

Table 4.6: A set of values reflecting the contradiction problem

<table>
<thead>
<tr>
<th>FILE ID</th>
<th>IMAGE</th>
<th>VIDEO</th>
<th>AUDIO</th>
<th>CRIME DESCRIPTION</th>
<th>FILE STATUS</th>
<th>ValidTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Burglary ....</td>
<td>CLOSED</td>
<td>[1953/3-1985/8]</td>
</tr>
<tr>
<td>100</td>
<td>XXX</td>
<td>VVV</td>
<td>AAA</td>
<td>Killing...</td>
<td>OPENED</td>
<td>[1983/7-∞]</td>
</tr>
</tbody>
</table>

In order to avoid contradiction in temporal databases, we enforce the following new constraint stated next.

CONSTRAINT B: If at any given time a relation containing two tuples that have the same apparent primary key value (the non valid time attributes of the primary key) but differ on the values of their non-key attributes then their valid time values i1 and i2 must be such that i1 overlaps i2 is false.
Chapter 4

MULTIMEDIA DATABASE MODEL

Traditional database systems can only store text data and any multimedia information would have to be stored outside of the database and referenced to. While this was a solution that was followed before the advent of multimedia database systems, they have many drawbacks in typical database issues like indexing, retrieval etc. which always caused problems. The Multimedia databases are used in every walk of life for various applications like video libraries, digital archives, music databases etc.

4.1 Basic Concepts

From its name we can deduce, multimedia comprises many kinds of media like images, videos, audios, graphics, and animations. It can also include MRI data and other abstract data types [4]. Many database management systems support the storage of abstract data types in binary large objects, this type is commonly known as BLOB; Recent database management systems support abstract data types stored as objects (ANSI SQL3 now supports this). The multimedia databases are of prominence in the world of computers today and more so for the flexibility and convenience they offer in representing various forms of objects that we come across in our everyday lives [2]. The forms of storing multimedia content historically have been many and include examples like VHS tapes, audio tapes etc. While these were very useful and served their purpose well, the impact of computing has been such that it has necessitated storing multimedia on computers [1]. This has necessitated utilization of different forms for storing multimedia and they include:

- Image Data: Images are very commonly found in multimedia databases and their applications cover simple figures, icons, medical images like X-rays etc.
- Video Files: These have become very important with the advent of technologies like distribution of video etc. It is now more convenient than ever to store a home video on a personal computer.
- Audio files: These files are being used extensively to store as well as distribute music and are even shared over the internet.
- Document Data: These are the traditional text files where information is stored in the form of text. These files are still in use and have changed in terms of the capability of storage size.

4.2 Different Types of Multimedia Databases

Multimedia data can be classified into two main types: a) non-continuous media and b) continuous media. With continuous media, there are temporal constraints which would require synchronization between the different media objects [2]. Most of these media types are stored using a separate storage server that would meet the real time requirements, whereas non-continuous data are stored in the database with metadata about the files on the continuous data. Hence, it can be summarized that in order to store multimedia objects we need to represent the monomedia objects, the spatial and temporal relationship between them and finally we need to represent the document structure that deals with the logical structure and presentation structure of those objects [1].

4.3 Representation of Multimedia in Databases

Multimedia data can be represented in several ways in a database. Multimedia data can be relational data in a relational database where the data representing the image is textual; this type of data is represented in binary form. Therefore, multimedia data in relational databases possess BLOB and CLOB types. Multimedia data can be also represented as a pointer to an external multimedia file, which will not be saved inside the database. Instead, its external address or location is stored and which references this external multimedia file. Last, multimedia data are represented in the form of objects implemented in Object Oriented Databases (OOD), therefore multimedia data can be managed as easily as standard attribute data. The last way of storing multimedia data will be the case in our project and will be explained in more details in the next chapter. In our project, multimedia data will be stored in an Oracle database using the interMedia package or cartridge, which includes a class library that defines all types of multimedia objects [4].
4.4 Fast view on Oracle and interMedia package

4.4.1 Introduction to interMedia

*interMedia* is new a feature supported by Oracle. This feature allows us to store, manipulate, and retrieve geographic location information, images, audios, videos, and other heterogeneous media data with other enterprise information. Oracle *interMedia* extends the Oracle DBMS' reliability, availability, and data management to multimedia content in Internet, electronic commerce, and media-rich applications as well as online Internet-based services [4].

4.4.2 Object Relational Technology

The DBMS we are using in our project is Oracle 10g. Oracle 10g is an *object relational* database management system. This means that in addition to its traditional role in managing relational and textual data safely and efficiently, it provides support for the definition of object types, including the data associated with objects and the methods that can be performed on them. This great new mechanism, which supports a complete object orientation, includes support for binary large objects (BLOB). This is to provide a good means to add multimedia objects in our case and which is complex to do. This means adding complex object to our database like digitized audios, images, and videos. Different object relational types exist in Oracle; for audio data we have the *ORDAudio* object type, heterogeneous data have the *ORDDoc*, image data have the *ORDImage*, and video have the *ORDVideo* object relational type. And all of the four object relational data types store data source in an object relational type called as *ORDSource* [4].
4.4.3 Multimedia Content Management

Oracle interMedia package is capable of storing, retrieving, managing, and manipulating multimedia data managed by Oracle database management system. Therefore, the Oracle interMedia package supports multimedia storage, retrieval, and management of the following:

- Binary large objects (BLOBs) stored locally in Oracle DBMS. The objects contain audio, image, video data, and other heterogeneous media data.
- File based large objects, known by BFILEs. These are stored locally in file systems within the operating system. They also contain audio, image, video data, or other heterogeneous media data.
- URLs that also contain audio, image, video or other heterogeneous media data. This media data can be stored on any HTTP server like Oracle Internet Application Server, Netscape Application Server, Microsoft Internet Information Server, and Apache HTTPD server etc ...
- Streams of audio and video data that could be stored on a specific media

Multimedia applications have many requirements that are common. Object types in interMedia package support application requirements that are common. Moreover, they can be extended to address application-specific requirements. With the interMedia package, managing multimedia data becomes easy as if we are managing attribute data in standard or relational databases [4].

InterMedia package in Oracle allows us to access applications through different interfaces (relational and object interfaces). Interaction between database applications (in C++, Java and traditional 3GLs) and intermedia is also supported. Furthermore, interaction between PL/SQL and OCI (Oracle Call Interface), and interMedia is also supported through a class library interface [4].

InterMedia supports storing well known file formats that includes desktop publishing image, and streaming audio and video formats in Oracle databases. InterMedia package
facilitates adding audios, images, videos, and other heterogeneous media objects to existing tables. It also allows insertion and retrieval for multimedia data. This provides database designers and administrators with the ability to extend their existing database applications with multimedia data or to design and build new end-user multimedia database applications from the ground up. Database designers and developers can use the functions provided by InterMedia to build their multimedia applications [4].

The object types that Oracle interMedia use to describe multimedia data are really similar to Java or C++ classes. A part of these object types are the following: ORDAudio, ORDDoc, ORDImage, and ORDVideo. Any of these object types consists of attributes, including metadata and the media data, and methods. As we have mentioned above, media data consists of the actual audio, image, video, or other heterogeneous data. Metadata is information about this data. For example, object length, compression type, and format all form constituents of metadata. Methods are functions that perform or act on the object itself such as getContent() and setProperties(), which are methods of all object types that we have discussed above.

As we have seen, a common media data storage model exists for InterMedia objects. The media data component of these objects can be saved inside the database in the form of binary large object under transaction control or can be saved outside the database, without transaction control. In the latter part, a pointer is stored in the database under transaction control, and the data is saved in the form of:

- External binary files (BFILE) which we already discussed above.
- An HTTP server based URL as the “Oracle HTTP Server”.
- A user-defined source on a specialized media data server or other server.

A convenient method for managing large and huge repositories of media data that resides in the form of flat files on a read only media, is storing this data outside the database.
This external data can be imported and converted into binary large objects at any time for processing and transaction control.

4.4.4 Multimedia Methods and Object Types

Oracle interMedia supplies the developer with ORDAudio, ORDDoc, ORDImage, and ORDVide object types and methods. These object types with their methods enable the database application developer to do the following:

- Modify the time an object was updated last time
- Manipulate the location of multimedia data
- Extract fields from multimedia data
- Retrieve and manipulate multimedia data from Oracle interMedia, Web servers etc...
- Perform minimal set of manipulation operations on multimedia data (ORDImage only)
- Perform file operations on the source and metadata extraction in XML formats (ORDAudio, ORDDoc, and ORDVide only)

4.4.6 How Multimedia Data is Stored?

In interMedia, multimedia data can be stored as an internal source within Oracle database, under transactional control as binary large objects (BLOB). It can also be stored as an external reference for digitized multimedia data in the form system-specific BFILE in a local file-system, as a URL on HTTP servers or as a user defined source on other servers.

Despite that these external storage means are remarkably efficient for integrating pre-existing sets of multimedia data within Oracle databases, as we have mentioned before the multimedia data will not be under transactional control. Binary large objects are stored in the database table spaces in a very efficient way that helps optimize space and access time to the database. Binary large objects whose size is less than 4 Kilobytes are stored inline, on the
other hand large BLOBs whose size is more than 4 Kilobytes may not be stored inline with other raw data. Therefore this depends on the BLOB size, in which a locator which is a pointer to the actual location of the BLOB value. The former is stored in the row while the latter is stored in other table spaces. Thus, when selecting a BLOB we are actually selecting the locator instead of the actual or real value. The good thing about this design is that we can store multiple BLOB locators in the same row. We might want to store a small video clip about a certain surgical operation with an audio recording that contains a description for that operation with image snapshots from the operation and those who are concerned with doing that operations like doctors, professors, and so on [4].

4.4.7 Querying Multimedia Data

In order to query multimedia data we should first have the interMedia cartridge installed within Oracle. Then we store our multimedia data within the Oracle database. Once stored in the right way, data can be retrieved and queried using various alphanumeric columns or object attributes of the table to find the row that holds our desired data. For instance, we might select a short video clip about a certain crime that occurred (from the CRIME_FILE table). The collection of multimedia data can be in this table or other tables in the database. This data can be linked to a set of attributes that describe the related content. Multimedia data content can be described with textual components and numeric attributes such as dates and ID numbers as it is the case in our project, even it can be described by other multimedia attributes. Within Oracle, data attributes can be located in the same table as the object type with objects also containing the metadata. Otherwise, the application developer or designer can define composite object types that include one of the interMedia object types combined with other attributes of different types [4].

4.4.8 How to Access Multimedia Data?

Multimedia data is accessed using applications like PL/SQL, Oracle Call Interface, and Java. This is done through object relational types enhanced by the intermedia package (e.g. ORDAudio, ORDDoc, ORDImage, and ORDVideo). To access the attributes within
each object we use the dot syntax notation. Consider a variable $V$ and data attribute $DA$, the syntax will be as following:

$V.DA$

For invoking methods of multimedia objects, the syntax is also the dot notation. For a variable $V$, function $F$ and parameters ($P1, P2, \ldots, Pn$), the example will be as following:

$V. F (P1, P2 \ldots)$

There we can see that a complete set of media attribute methods are provided for accessing attributes for each media type.
Chapter 5

TEMPORAL MULTIMEDIA DATABASE MODEL

So far we have seen the temporal model and the multimedia model. The Temporal Multimedia Database Model combines both concepts of temporal and multimedia models. Temporal multimedia databases meet the two requirements. They are concerned with keeping the historical multimedia data; past, current, and future. They have both the characteristics of a temporal database and those of a multimedia database. In this chapter, we present the definition for a temporal multimedia database model.

5.1 Temporal Multimedia Databases

As previously mentioned, we have two types of time dimensions that can be used in a temporal database the valid-time that records the validity of facts, and the transaction time which records the time when the facts were stored in the database. In our model, we restrict the discussion to the recording of the validity of facts. Therefore, the time dimension that we will use in our model is the valid time. To extend the multimedia database model to be a temporal multimedia database model, we add to our schema two attributes that define the validity of our record Time Valid Start and Time Valid End [6].

5.2 Definition of Temporal Multimedia Relation

A multimedia temporal relation is of the form:
\[ R(A_1, A_2, ..., A_n, O_1, O_2, ..., O_n\mid \bar{t}) \]
It consists of a number of attribute values associated with a bitemporal timestamp value \( \bar{t} \). \( A_1 \) to \( A_n \) represent a number of relational or textual attributes; they could have attributes as primary keys or foreign keys depending on the database schema design. \( O_1 \) to \( O_n \) represent a number of object-type attributes where each of these could be a representation for an image,
video, audio, or document object. These can’t be primary keys or referenced as foreign keys. In intermedia they are of the types ordImage, ordVideo, ordAudio etc…

The bitemporal timestamp value $t^0$ is represented by the ordered pair $(c',c')$ with $c'$ representing the transaction time and $c'$ the valid time. An example of a multimedia temporal relation will be conveyed in next chapter we will see a table whose attributes are a combination of relational attributes and multimedia object attributes [6].

5.3 Integrity Constraints

Integrity constraints in multimedia databases are not a challenging problem. Integrity constraints in multimedia databases are restricted to the problem of redundancy and content-based retrieval. Content-based retrieval is not our issue in this paper, but concerning the redundancy problem this can be easily solved by normalizing the database schema and this must be done because the redundancy of multimedia data is disastrous on contrary to redundancy of relational data [6]. This is because as we know, multimedia data require huge spaces to store. As for entity and referential integrity problems these don not arise in multimedia databases, since the representation of multimedia attributes (which are objects in our project) can not be primary keys or a reference for other keys (foreign keys). Therefore, the integrity constraints in Temporal Multimedia databases are the same as the integrity constraints of Temporal Databases mentioned above in chapter 3.
Chapter 6

MODEL IMPLEMENTATION AND EXAMPLE

In this chapter we will see an implemented prototype that demonstrates the implementation of a Temporal Multimedia Database by combining the techniques and concepts of both multimedia and temporal database models.

6.1 TMMDB (Temporal Multimedia Database)

Many commercial database management systems as Oracle, Sybase, Informix do not support temporal constraints; Therefore, are non-temporal DBMS. They might support data types to store data like events, but they do not enhance the developer neither with a temporal query language, nor with a temporal data definition language, nor with a temporal manipulation language. TimeDB is a commercial product provided by TimeConsult and that supports the manipulation and management of temporal data with an enhanced temporal query language, temporal data definition language, and temporal manipulation language.

6.2 TMMDB Functionalities

TMMDB stands for a temporal multimedia database model. We have developed a small prototype that shows how multimedia and temporal data could be stored simultaneously, giving the end-user the ability to retrieve, update, and insert such type of data. In our module we use the PL/SQL to execute all these operations. The user therefore, can retrieve multimedia data according to the time flags specified in SQL statements. Note that it's important to mention that the query language we are using doesn't support temporal constraints. The reason we did not use or integrate TimeDB (supports ATSQL2) with multimedia data is that TimeDB can only be integrated with relational databases. What happens is that TimeDB converts its temporal query language to a standard SQL query language and since we are using PL/SQL with defined multimedia objects (interMedia Library), we can't integrated TimeDB to query such type of databases.
Query 3: Exporting an image from the CRIME_FILE table to a file

Query Description:

This query does the opposite of the INSERT query. As we have seen the INSERT query uploads or inserts a multimedia file to the database. In this query we download a multimedia file from the database to a virtual directory path that we create. The query is obvious! We select the file we want to download referencing it to a multimedia object that we declare above. In this case it is the img object variable that references an image file. We download the file using the export() method. In the export() method we pass the virtual directory name and the multimedia file name that will be downloaded as parameters. As we said the procedure should be in the scope between BEGIN and END. No COMMIT here since we are not executing an update query like INSERT or UPDATE.
Query 4: Querying Media Data from the CRIME_FILE table with temporal restrictions – Querying the valid crime files.

Query Description:
This query is very similar to query 2. It queries the media data for an image, audio, or video. For example, image height and width or number of frames in a video clip, as we have seen before in query 2. But the main difference is that it uses temporal constraints. For example is the following query we retrieve the media data for multimedia file in CRIME_FILE table where the Time Validity ends after the current date or where the Time Validity is specified as Null meaning infinity.
Figure 6.3.8: Querying Media Data from the CRIME_FILE table with temporal restrictions

```sql
SELECT t.id_id, t.image.getHeight() height, t.image.getWidth() width, t.uid.getMimeType() mimetype, t.uid.getMimeType() mimetypeuid, t.aud.getMimeType() mimetypeaud, t.TUE FROM CRIME_FILE t WHERE TUE=sysdate OR TUE=NULL;
```

<table>
<thead>
<tr>
<th>ID</th>
<th>HEIGHT</th>
<th>WIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>1575</td>
<td>1200</td>
<td>1600</td>
</tr>
<tr>
<td>image/jpeg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>video/mpeg</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>HEIGHT</th>
<th>WIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>02-JUL-07</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>HEIGHT</th>
<th>WIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>02-JUL-07</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>HEIGHT</th>
<th>WIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>1580</td>
<td>1200</td>
<td>1600</td>
</tr>
<tr>
<td>image/jpeg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>video/mpeg</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>HEIGHT</th>
<th>WIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>audio/mpeg</td>
<td>02-JUL-07</td>
<td></td>
</tr>
</tbody>
</table>

SQL>
Query 5: Querying Media Data from the CRIME_FILE table with temporal restrictions –
Querying the crime files that were valid between '12-DEC-1994' and '12-DEC-2002'.

Query Description:
This query is similar to query 4. But we are retrieving multimedia data with different conditions this time.

```
SELECT t.id id,  
  t.image.getHeight() height,  
  t.image.getWidth() width,  
  t.image.getMimeType() minetype,  
  t.vid.getMimeType() minetypevid,  
  t.aud.getMimeType() minetypaud ,  
  TUE  
FROM CRIME_FILE t  
WHERE TUE<='12-DEC-2001' AND TUE>='12-DEC-1992';
```

<table>
<thead>
<tr>
<th>ID</th>
<th>HEIGHT</th>
<th>WIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINETYPE</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINETYPEVID</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINETYPEAUD</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TUE</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1570</td>
<td>610</td>
<td>1062</td>
</tr>
<tr>
<td>image/jpeg</td>
<td>video/npeg</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>HEIGHT</th>
<th>WIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINETYPE</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINETYPEVID</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINETYPEAUD</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TUE</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>audio/npeg</td>
<td>27-JAN-00</td>
<td></td>
</tr>
</tbody>
</table>
Query 6: Retrieving the insertion transaction time of the multimedia object using the `getUpdateTime()`

Query Description:
Usually when inserting an intermedia multimedia object into our database we use the `setUpdateTime()` to set the record transaction or insertion time. Some objects automatically call the `setUpdateTime()` time method. But what if we want to know or retrieve the multimedia object insertion or update time? These multimedia objects support a method called `getUpdateTime()`. In the following query, we use the “SET SERVEROUTPUT ON” to enable us to display an output on the screen using the “DBMS_OUTPUT.PUT_LINE()”.

Figure 6.3.10: Retrieving the insertion transaction time of the multimedia object using the `getUpdateTime()`

```sql
SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
2  img ORDImage;
3  v ORDVideo;
4  a ORDAudio;
5  BEGIN
6  SELECT t.image,
7      t.vid,
8      t.aud
9  INTO img,
10     v,
11     a
12  FROM CRIME.FILE t
13  WHERE t.id=1570;
14  -- printing the time and date output ...
15  DBMS_OUTPUT.PUT_LINE(TO_CHAR(img.getUpdateTime(),'MM-DD-YYYY HH24:MI:SS'));
16  DBMS_OUTPUT.PUT_LINE(TO_CHAR(v.getUpdateTime(),'MM-DD-YYYY HH24:MI:SS'));
17  DBMS_OUTPUT.PUT_LINE(TO_CHAR(a.getUpdateTime(),'MM-DD-YYYY HH24:MI:SS'));
18  COMMIT;
19  END;
20 /
02-09-2006 14:12:41
02-09-2006 14:12:43
02-09-2006 14:12:44

PL/SQL procedure successfully completed.
SQL>
```
Chapter 7

FURTHER WORK AND CONCLUSION

Multimedia Database Systems involve the interplay of dynamic (or time-sensitive) objects such as audio, video and static objects such as text, images etc. Multimedia systems have grown tremendously in the past years especially on the World Wide Web, where access to online news, music videos, weather etc. is available just by a single mouse-click. As and when people become more dependable on the web to access information, managing such large chunks of data can become a formidable task. Think about the bandwidth consumption and the I/O storage that we would be aiming at, as and when this data multiplies. Factors like these could impede the system's performance consequently affecting the user's information needs. Hence we need a better way of managing the data that is stored in these Multimedia Database Systems. This can be achieved through effective buffer management and data integration. There needs to be an effective query method that ties up with the spatial and temporal relations between the multimedia objects. Since the temporal objects are time sensitive, there needs to be an efficient way of synchronizing these objects via continuous displays. Multimedia Database Systems employ object-oriented databases that provide better complex object support, integration and concurrency control as opposed to relational databases.

Further work, is still needed to support temporal multimedia databases. This is because we have seen that commercial products established till now do not support features of temporal multimedia databases [6]. Products that support manipulation and management of temporal data just work with relational databases. Therefore, if our database was relational the problem of enforcing the temporal constraints to our databases would have been much easier. But since Oracle is a non temporal DBMS and our database is object-oriented, we were not able to find an existing solution that supports the manipulation of multimedia and temporal data. So progress is still needed to enhance the developer with DBMS that
supports the manipulation of temporal multimedia data with temporal multimedia query language and multimedia temporal definition language. Or we can extend a product like TimeDB, to support Object-Oriented Databases (which include multimedia) as it is the case in relational databases with a temporal query language, a temporal data definition language, temporal manipulation language and temporal constraints.
REFERENCES


