PICQUERY: A High Level Query Language for Pictorial Database Management

THOMAS JOSEPH AND ALFONSO F. CARDENAS

Abstract—A reasonably comprehensive set of data accessing and manipulation operations that should be supported by a generalized pictorial database management system (PDBMS) is proposed. A corresponding high-level query language, PICQUERY, is presented and illustrated through examples. PICQUERY has been designed with a flavor similar to QBE as the highly nonprocedural and conversational language for the pictorial database management system PICDMS, designed and developed at UCLA. PICQUERY and a relational QBE-like language would form the language by which a user may access conventional relational databases and at the same time pictorial databases managed by PICDMS or other robust PDBMS. This language interface is part of an architecture toward data heterogeneity transparency over pictorial and nonpictorial databases.

Index Terms—Database management, heterogeneous data, pictorial data, pictorial data accessing and processing, pictorial query language.

I. INTRODUCTION

Most of the data management systems that have been implemented to manage pictorial information (digitized images, drawings, etc.) were developed for use in specific application areas such as geographical applications, military reconnaissance, and medical applications. Very few generalized systems have been developed. Consequently the access or query languages developed for these systems are also largely application specific.

In this paper, we identify and propose a standard set of data manipulation operations that should be supported by generalized pictorial database management systems (PDBMS), and develop a high level query language (PICQUERY) which carries out these operations. PICQUERY has been designed to reside as a major software layer above a robust PDBMS such as PICDMS (Picture Database Management System) [1], [2], which is a PDBMS designed and developed at UCLA and now commercially available. PICQUERY has a flavor similar to QBE [3], and to QPE (Query by Pictorial Example) [4].

Architecturally, this effort is part of the UCLA Heterogeneous Distributed DBMS project, whose interest is the development of data heterogeneity transparency over pictorial and nonpictorial or conventional data (e.g., relational). As such, PICQUERY is intended to be the inter-

Manuscript received May 29, 1987; revised October 30, 1987.
T. Joseph was with the Department of Computer Science, University of California, Los Angeles, CA 90024. He is now with First Interstate Services Corporation, Los Angeles, CA 90245.
A. F. Cardenas is with the Department of Computer Science, University of California, Los Angeles, CA 90024.
IEEE Log Number 8819883.

0098-5589/88/0500-0630$01.00 © 1988 IEEE

Fig. 1. Pictorial and nonpictorial database management architecture.

face through which the user may access conventional relational databases using QBE and at the same time pictorial databases managed by PICDMS (or other robust PDBMS) using PICQUERY (see Fig. 1). PICQUERY and QBE are seen as one single language by the user.

Section II summarizes the architecture and data manipulation capabilities of PICDMS. This section explains the motivation for the PICQUERY language and examines the architectural fit of PICQUERY with PICDMS.

Section III identifies the standard set of operations that should be supported by a PDBMS.

Section IV focuses on the unique capabilities of PICQUERY and explains how various types of queries can be formulated.

Section V concludes this work.

II. DATA MODEL AND MANIPULATION IN PICDMS

PICDMS [1], [2] uses the gridded data representation scheme as opposed to the topological or polygonal subdivision scheme. Hence, it provides the generality to deal with pictorial or image data. It has a unique dynamic stacked image logical data structure as illustrated in Fig. 2, which consists of two-dimensional variables or pictures registered to the same grid. All data values (one for each picture) for the same grid cell or pixel are stored in a conventional data record, but with the significant fact that the data record is of variable logical structure depending on the number of images stored. This allows flexibility and
efficiency in the addition and deletion of pictorial data to the system, which is a crucial requirement. A new image is added as a new attribute in the data record, and not as a new instance of a fixed logical record. Conventional DBMS do not support such schema dynamism. The PICDMS data dictionary keeps track of the current record structure(s).

The data access is through the use of a windowed raster scan control algorithm which runs a rectangular window over cells of the image stack in raster scan order (Fig. 3). The PICDMS data access/manipulation language (DML) is a command oriented procedural language. The syntax of a PICDMS user language command is as follows:

```
<Command> ::= <Command Name>
             | (<set of variable type statements>)
             | <set of data base assignment statements>,
             | FOR <set of data base grid cells>;
```

The command name specifies the action to be taken. The set of variable type statements define the variables to be used to perform the command. The set of database assignment statements indicates the operations to be carried out on each grid cell belonging to the set of grid cells indicated in the fourth and last part of the command. The seven major data manipulation command names used to carry out data base operations are: COMPUTE, LIST, DISTANCE, ADD, REPLACE, DELETE, and PRINT. A large variety of PDBMS operations can be performed by the use of these commands.

The PICDMS data model and access language have been designed to support the major pictorial data management needs as indicated in [1]. The fundamental operations projection, join, etc., of conventional DBMS do not constitute the proper set of operations for image access and processing, except for perhaps topological oriented image handling (as in [4]–[6]).

The PICDMS data manipulation language is procedural in nature in that the user specifies what is wanted as well as, to some extent, how to obtain it. As the operations become more complex, the queries in the PICDMS DML user language can be expected to become longer and more time consuming to formulate. (Nevertheless, with PICDMS DML any pictorial data management query not fulfilled by one command may be fulfilled by developing the proper procedure in the DML).

Consequently, we have designed and are currently implementing the PICQUERY language as the nonprocedural and highly conversational language for nonprogrammer end-users (oceanographers, military analysts, medical analysts, etc.) to access and process pictorial data in PICDMS. PICQUERY is the high level end-user interface software layer above the PICDMS DML (see Fig. 1).

We initially considered three types of languages for PICQUERY. A purely nonprocedural, menu driven query language is good for on-line access. However, when the number of possible operations is very large, as it is the case for PDBMS, the purely menu driven approach is cumbersome. Also, the purely menu driven approach is not entirely suitable for query formulation for PDBMS because of the large number of variable attributes that may be involved in a query. Similarly, a nonprocedural SQL-like query language will be difficult to use by end-users.

A tabular form of query formulation provides a very efficient and easy means of query formulation to the user. The tabular format provides the user brief, easy to understand guidance in query formulation. Query by Pictorial Example (QPE) developed by Chang and Fu [4]–[6] is a tabular query language for use with a PDBMS. The QPE approach adopts the QBE approach where the user formulates a query by entering an example of a possible answer in the appropriate location in a titled table, displayed on the screen by the query language interface. Each operation in a QPE like query language is specified by using one or more tables, where each table is displayed on the screen with some parts being supplied by the system and the other parts by the user.

QPE and other similar pictorial interfaces designed for use on top of a conventional DBMS environment are practically suitable only for topological structures, that is, for pictures with well delineated objects. On the other hand,
PICDMS and PICQUERY provide a native pictorial DBMS environment for arbitrary types of pictures, with the robust pictorial operations described in the following sections.

Thus, we propose PICQUERY as a new tabular query language beyond QPE for logically enhancing on-line access of PICDMS. PICQUERY has features of both a menu driven query language interface as well as that of a tabular query language. The PICQUERY language will be illustrated through examples in Section IV.

The PICQUERY language is being implemented as an interface to PICDMS. The PICQUERY queries may be translated into PICDMS DML to carry out the operations (see Fig. 1).

III. MAJOR PICTORIAL DATABASE MANAGEMENT SYSTEMS OPERATIONS

Many areas of discourse and specialized pictorial data management systems were examined to gather the common denominator of pictorial data accessing and manipulation operations required [1]. Fig. 4 shows a sample of various pictorial data. Three major application areas to illustrate the spectrum of such common operations that would have to be available in a generalized PDBMS are: geographical, industrial, and medical domains.

A PDBMS for geographic pictorial information systems needs to provide the following operations in the analysis of image data: capability of viewing images at different levels of detail (zooming), rotation of images by different angles, cross tabulation of data, identifying nearest object, statistical operations such as covariance measures or averages, geometrical or spatial operations, clustering and point classification, thresholding, and edge detection.

Industrial PDBMS, in contrast to geographic PDBMS, tend to deal with smaller volumes of pictorial data and these systems may have to carry out complex computations on the basis of image data. Operations such as rotation of images, zooming, edge detection, template matching, texture measures, calculation of geometric and statistical measures, etc., are relevant for these systems.

Medical PDBMS, such as those used for storing X-ray pictures, also deal with large amounts of image data. The operations relevant for these systems include enhancement, rotation of images, contour extraction, and segmentation for automatic and semiautomatic diagnosis.

A general PDBMS should provide as many of these data manipulation capabilities (shared across several areas of discourse) as possible. The various data manipulation capabilities that should be provided by a generalized PDBMS can be classified into six categories. Sections III-A—III-F shall now outline major operation included in the six categories.

A. Image Manipulation Operations

Image manipulation operations perform certain transformation on the image to provide a different perspective of the image:

1) Panning or shifting operation to view different sections of an image.
2) Rotation of images to give a different view of the image.
3) Zooming operations. There are three types of zooming operations [7]: vertical zooming, to enable the user to view and image at different levels of detail (microscopic or macroscopic); horizontal zooming, to enable the user to view parts of an image selected on the basis of a selection criterion whose value can be varied continuously; diagonal zooming, to enable the user to view parts of a picture A which is related to parts of another picture B selected on the basis of some selection criterion. Detailed examples of these are given in Section IV.
4) Superimposing of one picture over another (one can think of it as a "pictorial" join).
5) Masking certain areas of a picture on the basis of some criteria.
6) Color transformation, to display a picture using different color combinations.
7) Projection operation on an image.

B. Pattern Recognition Operations

These operations involve recognizing and drawing (establishing) a pattern or searching the pictorial database for objects that match an identified pattern:
1) Edge detection, to detect edges by measuring changes in light intensity along the picture.
2) Thresholding, to build a binary image which is white in the regions with light intensity less than a threshold limit and black elsewhere.
3) Contour drawing, to draw contour lines joining points on the picture associated with the same attribute value.
4) Similarity retrieval (or template matching) to identify or retrieve picture objects which are similar to a given picture using a certain similarity measure or which match certain template patterns. Similarity retrieval can be done on the basis of size, shape, texture, etc.
5) Establish the boundary or perimeter for a region.
6) Texture measure, to measure or quantify the texture of an image.
7) Clustering (or point classification), to cluster or group together objects or points which are close together in a picture.
8) Segmentation, to divide a picture on the basis of some criteria.
9) Interpolation, to interpolate scattered point values of any particular function.
10) Nearest neighbor, to identify and retrieve the nearest neighbor object (of a particular type).

C. Spatial or Geometric Operations

These operations deal with spatial relationships between objects within a picture. These operations often deal with computation of certain geometric attribute values. The objects referred to in classical spatial queries can be classified as points, lines, and regions. However, PICQUERY envisions that the user may define any arbitrary pattern in a picture as an object (e.g., river, tank, bad blood cell). Some of the spatial queries use set operators. The classical operations included in this category are as follows.

1) Distance operation (point to point, point to line, point to region, line to line, line to region, region to region).
2) Length, center, slope of a line.
3) Area, centroid, perimeter of a region.
4) Operation to find the portion of a given line object similar to another line object.
5) Operation to find the portion of a given region object similar to another region object.
6) Intersection of a point and a line, a point and a region, two lines, a line and a region, and two regions.
7) Union of two region objects.
8) Difference between two region objects.

D. Function Operations

These operations are used with variables associated with an image or a picture:

1) Common functions (minimum, maximum, total value, count).
2) Statistical functions (average, standard deviation, covariance, histogram, range values, cross tabulation).

E. User Defined Functions and Programming Language Interface

A programming language interface should be provided to enable the user to define new functions and formulate queries using these functions.

F. Input/Output Operations

The input/output operations consist of the following:

1) Print or list tabular image information on paper.
2) Output operation to display/print a picture or object within a picture on a display screen or get a hard copy output.

3) Coloring operation to color parts of a picture on the basis of some criteria.
4) Change or update operation to change values of some picture variables.
5) Store operation to store some new information associated with an image in the pictorial database.

The above list is a reasonably comprehensive (although not complete) list of operations that may be expected of a very general PDBMS. It is apparent that conventional DBMS (relational, CODASYL, hierarchical) do not provide the proper set of fundamental operations to support properly the majority of the pictorial operations.

IV. THE PICQUERY LANGUAGE

The PICQUERY language commands may operate on the whole pictorial database or a set of picture-object identifiers. A picture is a distinctly identified, independent image stored in the pictorial database. An object is a named region, line, point, or part of a picture with which clearly defined features (variable values) are associated. Objects are identified to PICDMS/PICQUERY with the use of feature extraction techniques/packages [15]-[18] and database indexing (to identify picture-object relationships). If a picture or an object is identified in a command, the operation will be carried out on the identified picture-object set. If neither a picture nor an object is identified by the user in a command, the operation will be carried out on the entire pictorial database. Some commands require a picture-object set to be identified (e.g., ROTATE) whereas others may be carried out on the entire database (e.g., HORIZONTAL ZOOM). The syntax of the PICQUERY language is given in the Appendix.

PICQUERY is built on top of PICDMS which is grid-oriented. Pictorial databases structured in vector or polygon formats may be handled by PICQUERY/PICDMS if they are converted to grid format accepted by PICQUERY. Such converters [19] are available.

Since there are a large number of operations, the user will not be able to remember all the operations at any time. Hence, PICQUERY initially conveys to the user a list of the available operations. If the user types in "OPERATIONS P.", the system will print all the legal available operations with a brief explanation for each. If the user types in "IMAGE OPS P.", "PATTERN OPS P.", "FUNCTION OPS P.", or "I/O OPS P.", the system will list all the corresponding image manipulation operations.

Major elements of PICQUERY shall be now presented through a number of examples. A significant subset of the PICQUERY capability is illustrated. However, the following is not the complete definition of the language.

A. Image Manipulation Operations

1) Panning or Shifting Operations: A picture can be divided into a number of frames, each frame being the size of the display screen of the terminal. If a picture size is less than or equal to the screen size, the picture will be displayed at the center of the screen. When a picture is
viewed whose size is greater than the screen, initially the top left frame will be displayed. Then, the picture can be shifted vertically or horizontally using the shift operation. There is a second shift operation. This involves repositioning an object within a picture. For both these operations, the same table will be displayed.

Shift:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Horizontal Shift</th>
<th>Vertical Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC</td>
<td></td>
<td>-10</td>
<td>5</td>
</tr>
</tbody>
</table>

If the user wants to shift only the window over the picture, he/she should leave the object name column blank. The user has to enter the picture name in the first column, and the horizontal and vertical shifts in grid units in the third and fourth columns, respectively. A positive entry in the horizontal shift column indicates right shift and a negative entry indicates left shift. Similarly, a positive entry in the vertical shift column indicates shifting down whereas a negative entry indicates shifting up. If the user wants to carry out the second shift operation, he/she will have to fill in the column for object name. In that case, the identified object will be repositioned in the picture according to the entries in the last two columns. The entries in the table above indicates that the picture displayed is to be moved 10 grid units to the left and 5 grid units down.

2) Rotation Operation: As in the case of the shift operations, there are two rotation operations. The first rotation operation involves rotating a picture, whereas the second rotation operation deals with rotating and repositioning an object within a picture. For both these operations, the same table will be used. When the user types in "ROTAIE," the following table will be displayed.

Rotate:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Angle of Rotation From Vertical Axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC</td>
<td>STRIPE</td>
<td>-10°</td>
</tr>
</tbody>
</table>

Explaination: If the user wants to carry out the first rotate operation (i.e., rotation of the picture), he/she should leave the object name column blank. For the first operation, the user has to enter the picture name and the angle of rotation only. A positive value entered for angle indicates clockwise rotation relative to the vertical axis, and a negative value indicates anticlockwise rotation. If the second rotate operation is to be done, then the user has to enter the value for the object name column too. The entries made in the table shown above indicates an operation where the line object, STRIPE (belonging to picture, PIC) is to be rotated by 10° in the anticlockwise direction and repositioned within the picture.

3) Zooming Operations: For each of three different zooming operations provided, there is a different tabular format for constructing queries. There are two types of vertical zoom operations. One vertical zoom operation enables the user to zoom in/out on a picture, whereas the other vertical zoom operation allows the user to do the same on an object within the picture (the other parts of the picture remaining at the same level of magnification as before). For both these operations, the same table is used to formulate queries. When the user first types in "VZOOM" then the following table will be displayed.

Vertical Zoom:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Degree of Magnification</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC</td>
<td>C1</td>
<td>3</td>
</tr>
</tbody>
</table>

The user has to enter only the first and the third columns for the first type of vertical zoom operation. For the second type of vertical zoom operation, the user has to fill in the object name column too. If the entry for the degree of magnification is greater than 1 then it is a zoom in operation, else it is a zoom out operation. The entry given in the table above indicates that the object C1 is to be magnified by a factor of 3 and displayed.

For the horizontal zoom operation, the user has to first type in "HZOOM". Then the following table will be displayed:

Horizontal Zoom:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Magnification</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC</td>
<td>UTAH</td>
<td></td>
</tr>
</tbody>
</table>

The horizontal zoom operation may be performed on a whole picture or on only an object within a picture. If the user wants to do the former he should not fill in the column for object name. If the user wants to do the latter he/she has to fill in the column for object name. The query filled in the table above reads as follows: "Identify and display the places in UTAH (in picture, PIC) where (elevation is greater than 1000 ft.) or (elevation is less than 1000 ft. and rainfall is greater than 3 inches per annum)." Observe the use of the field "Group" to be able to interpret the selection condition unambiguously. If the entry for "Group" in the second line in the table was 1 instead of 2, the query would have meant the following: "Identify and display the places in UTAH (in picture PIC) where (rainfall is greater than 3 inches per annum)."

For the diagonal zoom operation, the user first types in "DZOOM", to display the following format:

Diagonal Zoom:

<table>
<thead>
<tr>
<th>Picture 1</th>
<th>Object Name</th>
<th>Picture 2</th>
<th>Selection Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>MISSILE SITES</td>
<td>PIC2</td>
<td>Range .gt. 5000 .lt. 2500</td>
</tr>
</tbody>
</table>

Here, picture 1 and picture 2 are related pictures. Selection is first made on picture 2 and then those parts of picture 1 related to selected portions of picture 2 are identified and displayed. PIC2 contains prototypes of various kinds of missiles, whereas PIC1 contains missile sites. By this query, the user is trying to identify the points in picture PIC1 where missiles with range greater than 5000 miles and speed greater than 25,000 mph are deployed.

4) Superimposing: Superimpose:

<table>
<thead>
<tr>
<th>Picture 1</th>
<th>Object Name</th>
<th>Picture 2</th>
<th>Object Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>A1</td>
<td>PIC2</td>
<td>A2</td>
</tr>
</tbody>
</table>
Using this table, two pictures can be superimposed or two objects belonging to the same or different pictures can be superimposed. In the example query, the user wants to superimpose object A2 of PIC2 on object A1 of PIC1.

5) Color Transformation:
Color Transformation:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Color</th>
<th>Selection Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>UTAH</td>
<td>Blue</td>
<td>Elevation .gt. 1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Red</td>
<td>Elevation .le. 2000</td>
</tr>
</tbody>
</table>

The following coloring scheme for the region object UTAH is requested: blue for those portions with elevation greater than 1000 ft. and less than or equal to 2000 ft., and red for those portions with elevation greater than 2000 ft. If the column for object name is left unfilled, then the coloring operation will be carried out on the entire picture PIC.

B. Pattern Recognition Operations

1) Edge Detection Operation: When the user types in "EDGE DETECTION," the following table format will be displayed.

Edge Detection:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Edge Name</th>
<th>Selection Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td></td>
<td>E1</td>
<td>Elevation .gt. 128</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Elevation .le. 2000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Elevation .lt. 54</td>
</tr>
</tbody>
</table>

In this operation, the edges are detected on the basis of changes in light intensity between neighboring cells. So, when the user enters a spectral band variable in the column for "variable," the relative value of the spectral band variable for a cell (compared to that of the 3 neighboring cells) is considered for carrying out the operation. In this example the edge E1 is to be created and displayed if the relative band 7 value is less than 128 or elevation is less than or equal to 2000 ft. and the relative band 7 value is greater than 64.

2) Contour Drawing:
Contour Drawing:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td></td>
<td>Elevation</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Elevation</td>
<td>2000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Elevation</td>
<td>3000</td>
</tr>
</tbody>
</table>

The contour drawing operation can also be done either on the whole picture or an object within a picture. If the operation is to be done on the whole picture, then the column for object name should be left unfilled. In the example given above, the user is requesting contours to be drawn for elevation equal to 1000 ft., 2000 ft., and 3000 ft.

3) Similarity Retrieval:
Similarity Retrieval:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Basis</th>
<th>Picture 2</th>
<th>Object Name</th>
<th>Object Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>CI</td>
<td>Shape</td>
<td>PIC2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Similarity retrieval operations can be done on the basis of size, shape, texture, and any other user-defined, application dependant basis. The column for "Basis" in the table is for the user to indicate on what basis the operation is to be carried out. The purpose of this operation is to retrieve objects in "Picture 2" similar to the object identified in "Picture 1." A "P." entry in any column indicates that the values for the column is to be printed after the operation is carried out. In this example, the user is requesting identification and retrieval of objects in picture PIC2 similar to object CI in picture PIC1. As a result of this operation, the system will fill in the names of retrieved objects in the last two columns of the table.

4) Segmentation Operation:
Segmentation:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Segment Name</th>
<th>Selection Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>SEG1</td>
<td>Rainfall .le.</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>SEG2</td>
<td>Rainfall .le.</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>SEG3</td>
<td>Rainfall .le.</td>
<td>20</td>
</tr>
</tbody>
</table>

The user can use this operation to segment a whole picture or an object within a picture. If the user wants to segment an object within a picture, then the user has to fill in the column for object name. In this example, the user wants to divide the picture, PIC into 3 segments according to values for rainfall. SEG1 segment consists of areas with rainfall less than or equal to 5 inches per annum. SEG2 segment consists of areas with rainfall between 5 and 10 inches per annum and SEG3 segment consists of areas with rainfall greater than 10 inches per annum.

C. Spatial or Geometric Operations

1) Distance: When user types in "DISTANCE," the system responds with the following table.

Distance:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object 1 Name</th>
<th>Object 2 Name</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>MAINE</td>
<td>NEVADA</td>
<td>P.</td>
</tr>
</tbody>
</table>

The minimum distance between the two objects will be printed in the column entitled "Distance."

2) Length of a Line:

Length:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Line Name</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>NILE</td>
<td>P.</td>
</tr>
</tbody>
</table>

This example requests the length of the line NILE in picture PIC.

3) Area of a Region:

Area:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Region Name</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>EGYPT</td>
<td>P.</td>
</tr>
</tbody>
</table>

This example requests the area of the region EGYPT in picture PIC.

4) Operation to Find the Portion of a Line Similar to Another Line:
Line Similarity:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Line Name 1</th>
<th>Picture</th>
<th>Line Name 2</th>
<th>Basis</th>
<th>Name for Similar Portion</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>NILE</td>
<td>PIC2</td>
<td>RHINE</td>
<td>C11</td>
<td>RHILE</td>
</tr>
</tbody>
</table>

This operation involves finding the portions of line 1 similar to line 2. In this example, line RHINE is compared to line NILE. The portion in RHINE similar to NILE is to be identified and named RHILE. If the operation is successful, RHILE will be highlighted on the screen or printout for the user.

5) Operation to Find a Portion of a Given Region Object Similar to Another Region Object:

Region Similarity:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Region Name 1</th>
<th>Picture</th>
<th>Region Name 2</th>
<th>Basis</th>
<th>Name for Similar Portion</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>EGYPY</td>
<td>PIC2</td>
<td>FRANCE</td>
<td>T</td>
<td>TEMPERATURE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Here the portion in FRANCE with similar temperatures to portions in EGYPT is to be identified and named FRAPT. If the operation is successful, FRAPT will be highlighted on the screen for the user.

6) Set Operations: This is a group of three operations: Union, Intersection, and Difference

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object 1 Name</th>
<th>Picture</th>
<th>Object 2 Name</th>
<th>Function</th>
<th>Basis</th>
<th>Name of Result Object</th>
<th>Result Null</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>C1</td>
<td>PIC2</td>
<td>C2</td>
<td>INTERSECTION</td>
<td>TEXTURE</td>
<td>C3</td>
<td>P</td>
</tr>
</tbody>
</table>

In this example, those portions in objects C1 (of PIC 1), and C2 (of PIC 2) with the same texture measure are to be recognized and identified as C3. After the operation is over, the system will print "yes" or "no" to indicate whether there is any result or not. If the result is non-null, then C3 (the result object) will be highlighted on the screen or printout.

In the following example, the area covered by objects R1 and R2 (of PIC1) will be identified as R3 and highlighted on the screen.

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object 1 Name</th>
<th>Picture</th>
<th>Object 2 Name</th>
<th>Function</th>
<th>Basis</th>
<th>Name of Result Object</th>
<th>Result Null</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC1</td>
<td>R1</td>
<td>PIC2</td>
<td>R2</td>
<td>UNION</td>
<td>AREA</td>
<td>R3</td>
<td>F</td>
</tr>
</tbody>
</table>

D. Function Operations

1) Generally Used Functions: When the user types in 'GENERAL-FUNCTION' the following table will be displayed.

General Functions:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Variable</th>
<th>Function</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC</td>
<td>UTAH</td>
<td>Rainfall</td>
<td>MINIMUM</td>
<td>P.</td>
</tr>
</tbody>
</table>

Possible functions are: MINIMUM, MAXIMUM, TOTAL, COUNT, AVERAGE, and STANDARD DEVIATION. After the operation is carried out, the value computed will be printed in the last column. In this example the user wants to know the minimum rainfall pertaining to the region UTAH.

2) Histogram:

Histogram:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Variable</th>
<th>Value Range</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC</td>
<td>UTAH</td>
<td>Temperature</td>
<td>30 to 40</td>
<td>40</td>
</tr>
</tbody>
</table>

The number of points whose variable value falls within the value range will be printed in the frequency column. The user is requesting the frequency histogram on the basis of temperature values for the region UTAH in picture PIC.

3) Range Values:

Range Values:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Variable</th>
<th>Range Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC</td>
<td>UTAH</td>
<td>Elevation</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>P.</td>
</tr>
</tbody>
</table>

Here, the range values are printed in the last two columns after the operation is carried out. The user wants to know the range of values for elevation for the region UTAH in picture PIC.

4) Crosstabulation:

Crosstabulate:

<table>
<thead>
<tr>
<th>Picture</th>
<th>Object Name</th>
<th>Crosstab Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC</td>
<td>Population</td>
<td>F. Census Tract</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>School</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Franks</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Edith</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sarah</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Johnson</td>
</tr>
</tbody>
</table>

This operation will calculate and print the crosstabulation variable values for the new segmentation variable groups based on the figures for the old segmentation variable groups. In this example, the user is trying to estimate school district population from census tract population, given a picture of school districts and one of census tracts.

E. User Defined Functions

The programming language interface in PICQUERY allows the user to define his/her own functions to query the database. For this, the user will have to carry out the following tasks: 1) define the function to PICQUERY by writing a procedure in PICDMS DML and/or PICQUERY and a programming language like Pascal or C; 2) design and store a tabular format (to be associated with the query for this function) into PICQUERY. This feature allows for the extension of PICQUERY capabilities to satisfy the needs of various users.

V. Conclusions

The motivation and justification for a highly interactive user language and environment for the increase and challenge of generalized pictorial data management has been presented. PICQUERY and a relational language like QB would form the language by which a user may access conventional relational databases and at the same time pictorial databases managed by PICDMS (or other robust PDBMSs) (see Fig. 1). A reasonably comprehensive set of pictorial data accessing and manipulation operations required of a PDBMS has been presented. The PICQUERY language designed provides support for the majority of these operations with few command entries. It is apparent that conventional PDBMS's (based on conventional relational, CODASYL, hierarchical DBMS) do not
provide the proper set of fundamental operations to support properly the majority of pictorial operations.

A significant subset of PICQUERY has been illustrated through examples. The full definition of the language has not been included. It is clear, however, that the PICQUERY approach is such that it is a rather open ended language environment which can be extended to accommodate additional or new functions that may be of particular interest to specific areas. In fact, the architecture of PICDMS, of the procedural oriented PICDMS DML and of PICQUERY is such that they provide a fundamental shell on which further language operations for pictorial data management needs may be easily built.

Some of the operations cited in Section III and illustrated in Section IV with PICQUERY can be done fully graphically with a mouse device and without keyboard intervention by means of a graphical interface such as Apple McIntosh, Microsoft Windows, or Digital Research’s GEM. Examples are the movement of the window over a large picture, the rotation or shifting of objects within a picture, etc. This possibility is being currently addressed.

Group Identifier identifies the set of condition lines that are part of the same complex condition. For example, if the selection condition is (A or B or C) and (D), Condition lines A, B, and C will have the same group identifier; whereas condition line D will have a different Group Identifier. The use of the Group Identifier removes any ambiguity in formulating selection conditions.

---

**APPENDIX A**

**PICQUERY SYNTAX**

The PICQUERY Syntax is defined as follows.

```plaintext
<Command> :: = <Command Name>
<Set of Picture-Object Identifiers> :: = <Picture Identifier> <Object Identifier> ...
<Set of Functional Variable Identifiers/Values> :: = <Functional Variable Identifier> <Functional Variable Values> ...

Functional Variable values can be the result of an operation too.
<Resultant Object Identifiers> :: = <Object Identifier> <Presence of Resultant Object>
<Set of Selection Conditions> :: = <Selection Condition line> ...
<Selection Condition line> :: = <Variable Identifier> <Relation Operator> <Variable Values> <Logical Operator> <Group Identifier>
```
REFERENCES


Thomas Joseph received the B.Tech. degree in mechanical engineering from the Indian Institute of Technology, Madras, India, the M.B.A. degree from the Indian Institute of Management, Calcutta, India, and the M.S. degree in computer science from the University of California, Los Angeles.

He is currently working as a consultant at First Interstate Services Corporation in Los Angeles. He is involved in the design and implementation of new advanced bank credit card systems. While at UCLA, he was involved in research leading to the development of higher level accessing of pictorial data.

Alfonso F. Cardenas received the B.S. degree from San Diego State University, San Diego, CA, and the M.S. and Ph.D. degrees in computer science, from the University of California, Los Angeles.

He is currently a Professor in the Department of Computer Science, School of Engineering and Applied Science, at UCLA, and a consultant in computer science and management. He has been a consultant to both users of database technology and to vendors of database management technology. He has offered seminars or consulted for IBM Corporation, the Rand Corporation, Arthur Young and Company, TTI/Citcorp, Systems Engineering Laboratories, Petroleos Mexicanos, Jet Propulsion Laboratory, Goodyear Atomic Corporation, General Dynamics, Hughes Aircraft, and other major organizations. He was visiting Scientist/Consultant to IBM Corporation in 1972-1975, on future database and applications generation technology. He is member of the board of directors of two companies in the computer field. He has led seminars and has spoken at conferences all around the world. He is author of the book Data Base Management Systems (Boston: MA: Allyn and Bacon, 1979 and 1984), coeditor and coauthor of Computer Science (New York: Wiley, 1971), and author of many other publications.

Dr. Cardenas is the current president of the Very Large Data Base Endowment, which conducts the annual international conference on very large databases. He is listed in Who's Who in Computers and Data Processing, Dictionary of International Biography, and American Men of Science. He is a member of the Association for Computing Machinery and the Society for Information Management.