CSE 736: Advanced Topics in Database Systems

YINGJING YAN

INTRODUCTION

- Chabot is a picture retrieval system for a database that will eventually include over 500,000 digitized multi-resolution images.
- For retrieval, Chabot uses tools provided by POSTGRES, such as representation of complex data types, a rich query language, and extensible types and functions.

INTRODUCTION

- To implement retrieval from the current collection of 11,643 images, Chabot integrates the use of stored text and other data types with content-based analysis of the images to perform “concept queries”.
- The Chabot project was initiated at UC Berkeley to study storage and retrieval from a large collection of digitized images.

INTRODUCTION

- Requests vary from those where the ID number of the desired picture is already known, to very general requests for “scenic pictures” of California lakes and waterways.
- DWR keeps the slides that are requested most often in lighted display boxes for browsing; the rest of the collection is housed in archival containers and slide drawers.

INTRODUCTION

- While an attempt is made to annotate each image with as much descriptive information as possible, keyword indexing for an image collection has significant limitations. It may fail to handle problems such as non-specific request, inaccurate descriptions.

INTRODUCTION

- The Chabot project was initiated to replace the existing system with a better system that includes:
- An advanced relational database for images and data;
- Large-scale storage for images;
- On-line browsing and retrieval of images;
- A flexible, easy-to-use retrieval system;
- Retrieval of images by content.

System Motivation and Goals

- DWR needs a DBMS that can support a variety of complex data types including text, numerical data, relative and absolute time, and geographical location.
- Retrievals should be possible on any combination of the complex data types that are associated with the images, as well as on the content of the images themselves.

System Motivation and Goals

- (1) Scalability and Storage Concerns;
- (2) Simplicity of Use, Simplicity of Design;
- (3) Flexible Query Methods;
- (4) Querying by Image Content;
- (5) must integrate stored textual information with image content information.

Current Research

- The problem of how to store large numbers of digitized images and retrieve pictures from such a collection is an active area of research that overlaps many fields within computer science including graphics and image processing, information retrieval, and databases.

Description of Chabot

- Chabot includes a top-level user interface that handles both queries and updates to the database. Our querying mechanism retrieves images on the basis of stored textual data as well as on more complex relations among the stored data.

POSTGRES

- POSTGRES: To store the images and textual data, we are using POSTGRES. POSTGRES is particularly attractive for use with a database like Chabot; in addition to the standard relational database features, it provides features not found in traditional relational DBMS's, such as:

POSTGRES

- (1) Object-oriented properties;
- (2) Complex types;
- (3) User-defined indices;
- (4) User-defined functions.

Storage

- The storage solution is to use a two-level storage scheme. We use magnetic disk for storing the thumbnail images and text needed for browsing the database and we archive the large multi- resolution image files on a tertiary device, a Metrum VHS-tape jukebox.

Storage

- The Metrum holds 600 VHS tapes, each tape having a 14.5 GB capacity. With a total capacity of 10.8 TB, the Metrum is more than adequate as a repository for the DWR image library. The average time for the Metrum to find a tape, load it, and locate the required file is about 2 minutes - too slow for browsing a set of images but fast enough for filling a request from a DWR client once the desired image has been identified.

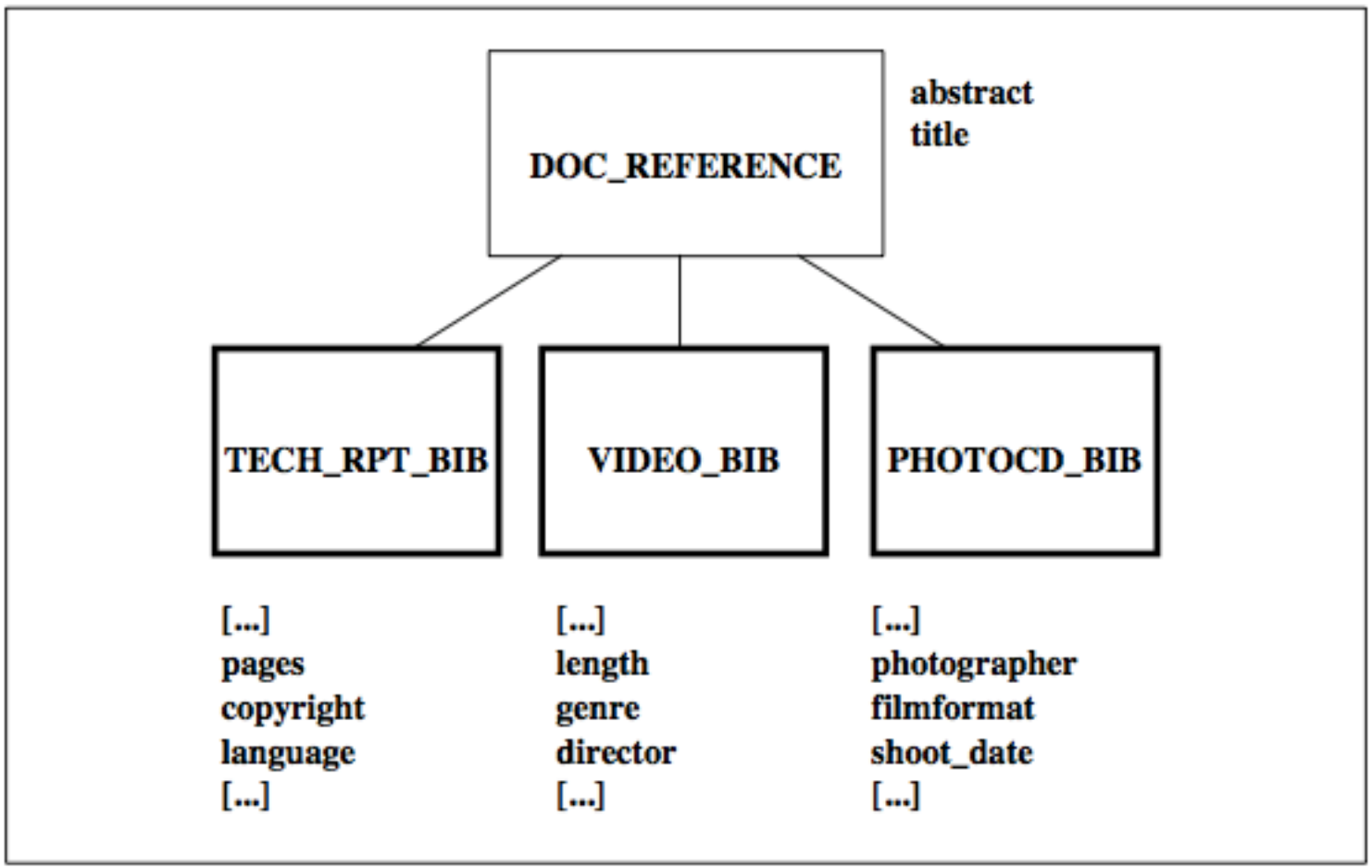
The Schema

- The schema for the Chabot project was designed to fit with those of other research projects in progress at Berkeley -- a collection of technical reports and a video library.
- The image class in their database is called PHOTOCD_BIB, for “Photo-CD Bibliography”, which inherits the attributes “title” and “abstract” from the DOC_REFERENCE class, which is shared by the technical report and video object classes.

The Schema

- As shown below, the PHOTOCD_BIB class contains “bibliographical” information about the image object, such as the ID number, the name of the photographer, the film format, the date the photo was taken, and so on. A complete list of attributes for the PHOTOCD_BIB class is shown in Table 1 below.

The Schema



Schema for technical report, video, and photo-cd classes

The Schema

- Most of the attributes for the image class are stored as text strings; there are two fields that have type abstime, the "shoot_date" of the photo and the "entry_date" that the information was entered into the database. These allow us to perform time-relative searches, for example, "Find all shots of Lake Tahoe that were taken after January 1, 1994."

The Schema

attribute	type	description
abstract	text	abstract (for documents)
title	text	title (of document)
comments	text	comments
disknum	text	Photo-CD number
imgnum	integer	image number on CD
id	text	DWR ID number
doc_type	text	nature, art, legal, etc.
copyright	text	copyright information
indexer	text	person creating db entry
organization	text	who commissioned photo
category	text	DWR category - "SWP", etc.
subject	text	DWR subject - "The Delta"
location	text	one of 9 California regions
description	text	a description of the image
job_req_num	text	DWR job request ID
photographer	text	photographer
filmformat	text	"35 mm slide"
perspective	char16	aerial - ground - close-up
color	char	C (color) B (black & white)
orientation	char	H (horizontal) V (vertical)
histogram	text	color histogram
entry_date	abstime	date of db entry
shoot_date	abstime	date photo was taken
oid	oid	POSTGRES object ID

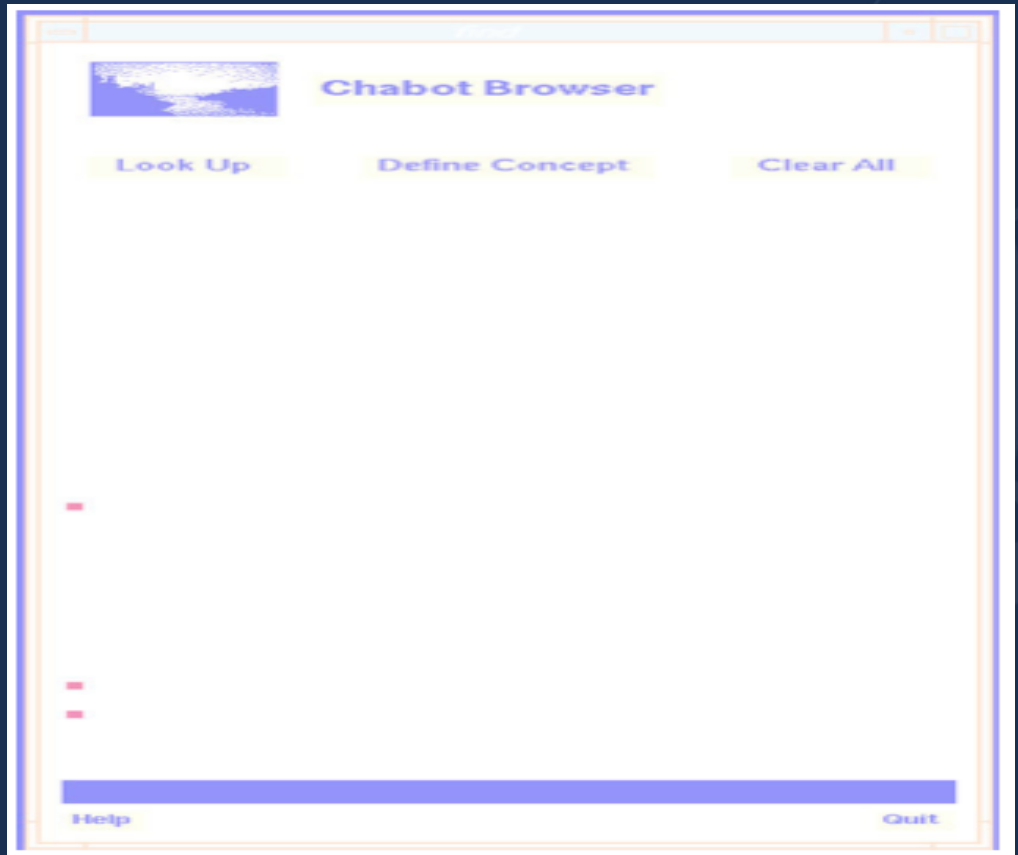
Table 1: Attributes for the PHOTOCD_BIB class

The User Interface

- The interface for Chabot is designed to prevent accidental corruption of data while browsing the database; the main screen gives the user three options: *find, edit, and load.*
- The database can be modified only via the edit and load screens and user authorization for these screens is required. The find screen is for running queries and for browsing the database.

The User Interface

- An example of the current implementation for the find window appears below.



the *find* window

The User Interface

- For example, using the search criteria from the find screen shown, the Postquel query would be

  :retrieve (q.all) from q in PHOTOCD_BIB where q.shoot_date>”Jan 1 1994” and q.location~”2” and MeetsCriteria (“SomeOrange”,q.histogram)

MeetsCriteria

- To implement concept queries, we use two capabilities that POSTGRES provides: storage of pre-computed content information about each image (a color histogram) as one of the attributes in the database, and the ability to define functions that can be called at run-time as part of the regular querying mechanism to analyze this stored information.

MeetsCriteria

- The function “MeetsCriteria” is the underlying mechanism that is used to perform concept queries. The example above shows how MeetsCriteria is used within a query. It takes two arguments: a color criterion such as “Some Orange” and a color histogram.
- The user selects a color criterion from a menu on the *find* screen, and a call to MeetsCriteria is incorporated into the query using the selected color.

MeetsCriteria

- For the histograms, we have experimented with quantizing the colors in our images to a very small number so that run-time analysis is speeded up. We have found that quantizing to as few as 20 colors allows us to find the predominant colors in a picture for the “Mostly” queries while still providing a glimpse of the minor colors for the “Some” queries.

MeetsCriteria

- POSTGRES's query optimization facility is used to minimize the search set of histograms. The function returns true if the histogram meets the criterion, false if it does not. Although the method for finding histograms that meet the criterion varies according to which color is being checked, in general the algorithm employs two metrics: compliance and count.

MeetsCriteria

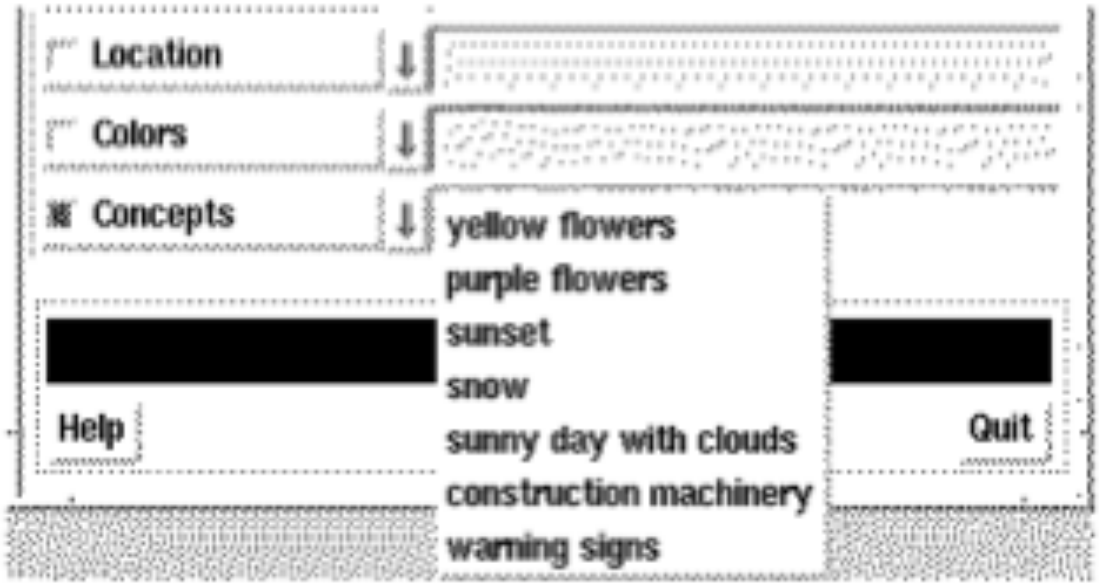
- The former count is used when we are looking for “Some” colors; in the “Some Yellow” example, we get a true result if just one or two of the twenty colors in the histogram qualify as “yellow”. We use the total pixel count for the “Mostly” matches: more than 50% of the total pixels of an image must be “red” in order for the image to meet the “Mostly Red” criterion.

Concept Queries

- In addition to using color directly for content analysis, users can compose higher level content-based queries to the database that embody contextual information such as “sunset” and “snow”. These queries are called concept queries.

Concept Queries

- The *Concepts* selection on the *find* screen of the interface lists the concept queries that are available, each of which has been previously defined by the user:



Concept Queries

- Selecting a concept from the pull-down menu generates a Postquel query that incorporates a combination of search criteria that satisfy the concept. Typically MeetsCriteria is used in these queries for color analysis in combination with some other textual criteria.

Concept Queries

- For example, when “sunset” is chosen from the *Concepts menu, the following query is sent to the database:*
  retrieve (q.all) from q in PHOTOCD_BIB where q.description ~ “sunset” or MeetsCriteria(“MostlyRed”,q.histogram) or MeetsCriteria (“MostlyOrange”,q.histogram)

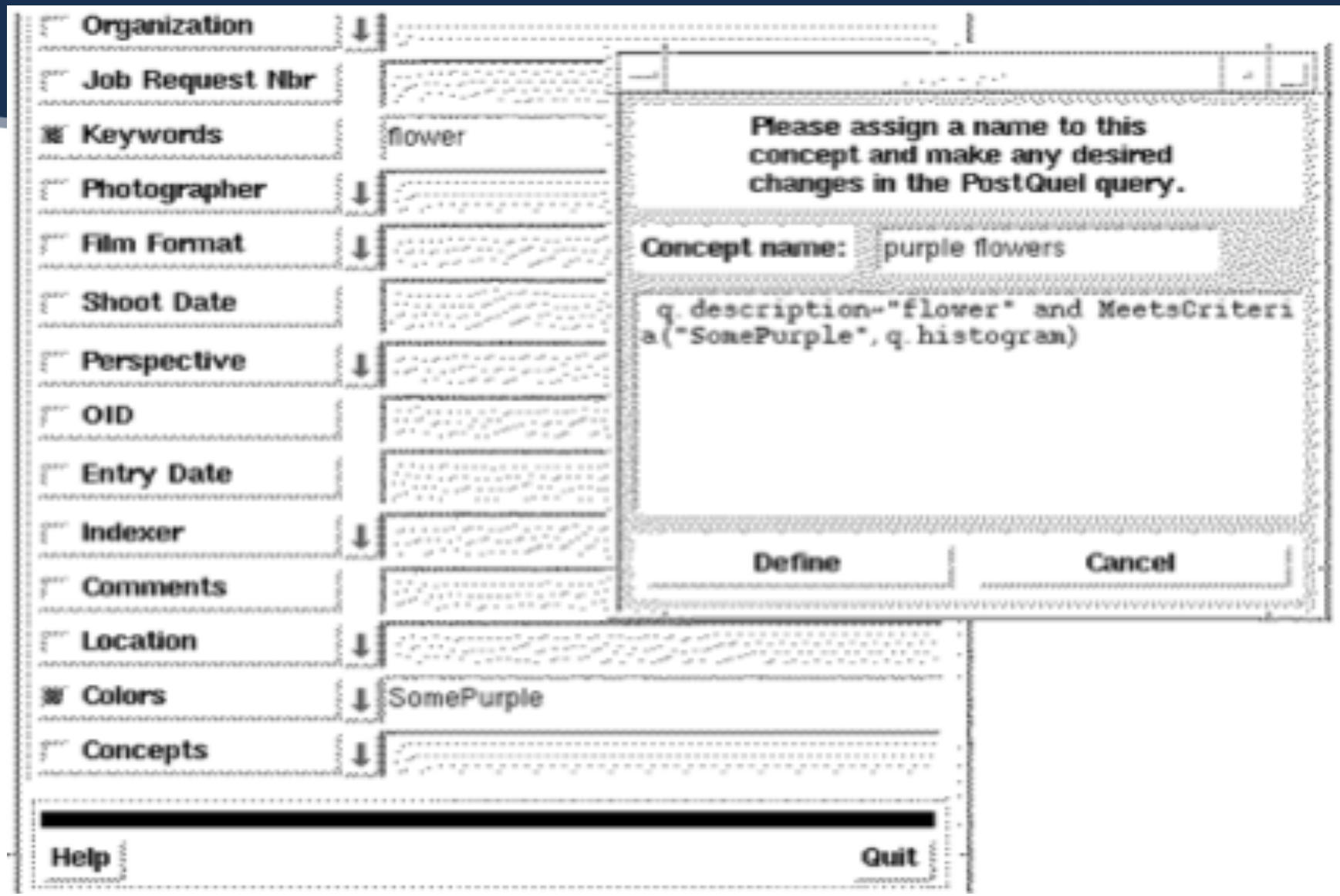
Concept Queries

- In this case, the user has defined the concept “sunset” as including images that have the stored keyword “sunset” associated with them, or images that have red or orange as their predominant color. Concept queries can be used in conjunction with other criteria.

Concept Queries

- Users can define new concepts and add them to the *Concepts* menu by first selecting criteria on the find screen that should be included in the new concept. Clicking on the "*Define Concept*" button on the find screen brings up a dialog box prompting the user for the name of the new concept, as illustrated below.

Concept Queries



Concept Queries

- The Postquel query can be edited, after which the user presses the “Define” button to register the new concept. The query is written to a file in the user’s home directory, so that the new concept is immediately available, and future invocations of the browser will include it as well.

Concept Queries

- The editing capability can also be used to add postquel constructs that may not be otherwise available, such as disjunctive conjunctions. The user can edit the concept file, make copies of the file available to other users, and incorporate others' concepts in the file.

Multi-Level Aggregation

- To test our content analysis, we measured the *recall* and *precision* of some of the concept queries that are shown in the User Interface section. Recall is the proportion of relevant materials retrieved, while precision quantifies the proportion of retrieved materials that are relevant to the search.

Testing

- For each concept query, we identified by hand all the images in the collection that we thought should be included in the result set. We then tried various implementations of the concept using different combinations of content-based and stored textual data. We measured *recall* and *precision* for each implementation.

Testing

- Table 2 shows the results from one of the test queries that is representative of our findings, the concept “yellow flowers”. For this concept, we first identified 22 pictures in the collection that were relevant; we then implemented the “yellow flowers” function in seven different ways using different combinations of search criteria.

Testing

- As shown below, queries 1-3 used keyword search only, queries 4 and 5 used only content-based information, and queries 6 and 7 used a combination of keyword and content-based data.

Testing

#	keywords	color content	retrieved	relevant	recall	precision
1	"flower"	-	55	13	59.1	23.6
2	"yellow"	-	11	5	22.7	45.4
3	"flower" and "yellow"	-	5	4	18.1	80.0
4	-	SomeYellow (2)	235	16	72.7	6.8
5	-	SomeYellow(1)	377	22	100	5.8
6	"flower"	and SomeYellow (2)	7	7	31.8	100
7	"flower"	and SomeYellow(1)	15	14	63.6	93.3

Table 2: Query "Find yellow flowers" (relevant images = 22)

Testing

- In this test, two different methods for finding yellow were tried. SomeYellow (2) means there were at least two yellow colors in a 20-element histogram.
- SomeYellow (1) means that only one yellow color is needed for the picture to be counted as having “some yellow”.

Testing

- As shown for query 5, pictures can be retrieved with 100% recall if the color definition is broad enough, but the precision is too low: the 377 images retrieved from query 5 would require the user to browse nineteen screens of thumbnails (each screen displays 20 images) to find the pictures of yellow flowers. Using the coarse definition for yellow in conjunction with the keyword “flower” gives the best result: query 7 has a recall of 63.6% with a very high precision of 93%.

Testing

- A good deal of experimentation was necessary to find the right combination of color content and keywords.
- Thus, the success of the concepts that users define will depend to some degree on their familiarity with the images in the collection.
- In summary, we found that retrieving images on keywords alone or on content alone produced unsatisfactory results.

~~Conclusions and Future Work~~

- Implementations are underway to include techniques to improve the color analysis, other content analysis techniques besides color, such as texture, shape, and line.
- Since so many of our retrievals are based on the stored textual data rather than on the images, some information retrieval techniques will be included, such as the use of a thesaurus and dictionary.

Conclusions and Future Work

- One plan is to integrate the Chabot schema with those of the geographical and environmental datasets of other research projects at UC Berkeley such as satellite imagery, aerial photography, and environmental reports.

Conclusions and Future Work

- One planned enhancement is to generate longitude and latitude coordinates for the DWR images using GIPSY, a system for extracting these coordinates from textual place names. Using this technique, a location name like “El Cerrito” that is attached to one of the DWR images can be associated with spatial data that contains longitudinal coordinates.