CollegeCrux

This document contains the analysis, design, planning, and implementation of the database for a new website, CollegeCrux.com

Brenden Bishop

Database Systems

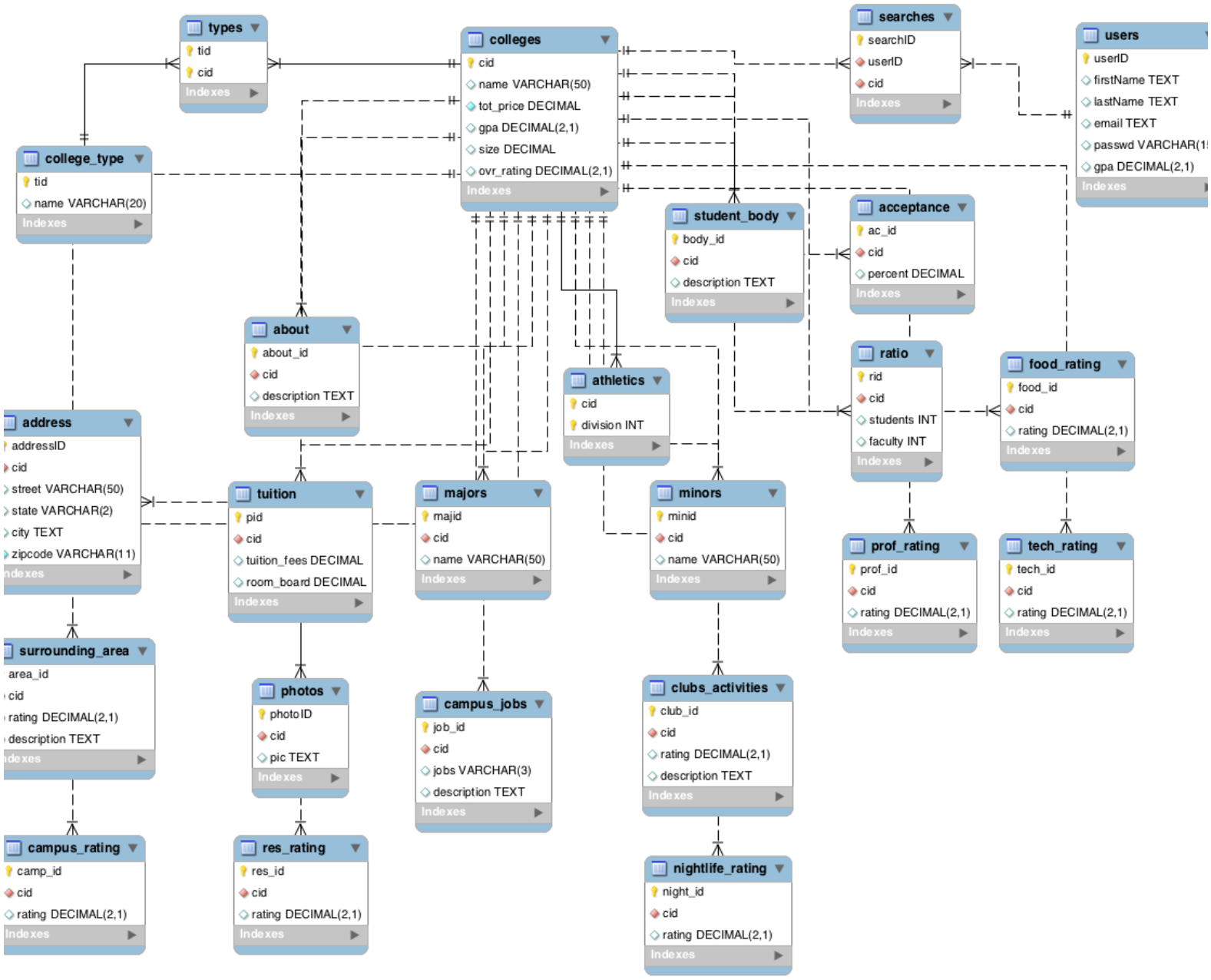
2013

Table of Contents

Executive Summary

The mission of this database is to provide a simple, efficient, informative website intended for high school students and their parents, for researching colleges. This will help students inform themselves and make a good decision on which college or university they will be attending. It will be one site to contain all colleges, not just the prominent ones. One of the strengths of the site will be our highly detailed information, about all types of colleges. Another is our accuracy, in which factual information comes directly from the colleges' websites themselves. The reviews and ratings will come from current students and alumni. All alumni and current students must have an account registered with their college email address to post only specifically to that college. This ensures security to maintain the integrity of the ratings and reviews. Another strong point will be the site's visual aesthetics of colleges and campuses. The site will be filled with college photos taken by the school and or students, so that prospective students can not only have detailed information but also, get a feel for the college and its campus. In addition, the site will focus on its ease of use; any parent or student can access the site for free without any account registration required and no limits. The point of the account is for students to save their colleges to help better narrow down and organize their choices. The college listings will also have direct links to the college's website to preserve accuracy and integrity, as well as, if available, direct links to apply to the college. The SQL code in this document will highlight the backend functionality of the site, some of the websites features will not be applicable from the database code as they will be done on the front end.

Entity Relationship Diagram



Tables: Create statements, functional dependencies, and Sample Data.

Colleges

Table:

```
DROP TABLE IF EXISTS colleges;
CREATE TABLE colleges (
  cid                   SERIAL not null,
  name                  VARCHAR(50),
  tot_price             NUMERIC not null,
  gpa                   NUMERIC(2, 1),
  size                  NUMERIC(5),
  ovr_rating            NUMERIC(2, 1),
 primary key(cid)
);
```

Functional Dependencies:

cid -> name
cid -> tot_price
cid -> gpa
cid -> size
cid -> ovr_rating

Sample Data:

cid	name	tot_price	gpa	size	ovr_rating
1	Marist College	42100	2.8	6303	3.5
2	Siena College	12495	2.7	3000	3
3	Stanford University	58846	3.5	19945	3.75
4	Yale University	55311	3.6	11250	0
5	Princeton University	53250	3.5	7912	0
6	Harvard University	53011	3.8	21000	0
7	Nassau Community College	4088	0	23000	0
8	Georgetown University	56362	3.4	7590	0

Addresses

Table:

```
DROP TABLE IF EXISTS address;
CREATE TABLE address (
  addressID   SERIAL not null,
  cid         SERIAL not null references colleges(cid),
  street      VARCHAR(50),
  state       VARCHAR(2),
  city        TEXT,
  zipcode     VARCHAR(11) not null,
 primary key(addressID)
);
```

Functional Dependencies:

addressID -> street
addressID -> state
addressID -> city
addressID -> zipcode

Sample Data:

addressID	cid	street	state	city	zipcode
1	1	3399 North Road	NY	Poughkeepsie	12601
2	2	515 Loudon Road	NY	Loudonville	12211
3	3	450 Serra Mall	CA	Stanford	94305
4	4	38 Hillhouse Avenue	CT	New Haven	06511
5	5	Princeton University	NJ	Princeton	08544
6	6	Harvard University	MA	Cambridge	02138
7	7	1 Education Drive	NY	Garden City	11530
8	8	3700 O St NW	DC	Washington	20057

CollegeType

Table:

```
DROP TABLE IF EXISTS college_type;
CREATE TABLE college_type (
  tid          SERIAL not null,
  name         varchar(20),
 primary key(tid)
);
```

Functional Dependencies:

tid -> name

Sample Data:

tid	name
1	Ivy
2	Tech
3	Private
4	Religious
5	State
6	Community

Types

Table:

```
DROP TABLE IF EXISTS types;
CREATE TABLE types (
     tid        serial not null references
                college_type(tid),
     cid        serial not null references
                colleges(cid),
 primary key(tid, cid)
);
```

Functional Dependencies:

None associate table.

Sample Data:

type_id	tid	cid
1	3	1
2	3	2
3	4	2
4	3	3
5	1	4
6	3	4
7	3	5
8	1	6
9	3	6
10	5	6
11	5	3
12	5	4
13	5	5
14	6	7
15	3	8
16	4	8

Ratio

Table:

```
DROP TABLE IF EXISTS ratio;
CREATE TABLE ratio (
  rid               SERIAL not null,
  cid               SERIAL not null references
                    colleges(cid),
  students          int,
  faculty           int,
 primary key(rid)
);
```

Functional Dependencies:

rid -> students
rid -> faculty

Sample Data:

rid	cid	students	faculty
1	1	5	1
2	2	14	1
3	3	5	1
4	4	5	1
5	5	6	1
6	6	7	1
7	7	12	1
8	8	11	1

Athletics

Table:

```
DROP TABLE IF EXISTS athletics;
CREATE TABLE athletics (
  cid              SERIAL not null references
                   colleges(cid),
  division         int,
 primary key(cid, division)
);
```

Functional Dependencies:

(cid, division) -> division

Sample Data:

aid	cid	division
1	1	1
2	2	1
3	3	1
4	4	1
5	5	1
6	6	1
7	7	3
8	8	1

Tuition

Table:

```
DROP TABLE IF EXISTS tuition;
CREATE TABLE tuition (
  pid                   SERIAL not null,
  cid                   SERIAL not null references
                        colleges(cid),
  tuition_fees          NUMERIC,
  room_board            NUMERIC,
 primary key(pid)
);
```

Functional Dependencies:

pid -> tuition_fees
addressID -> room_board

Sample Data:

pid	cid	tuition_fees	room_board
1	1	30,700	12,600
2	2	31,118	12,495
3	3	42,690	13,166
4	4	35,900	13,070
5	5	41,750	13,545

Photos

Table:

```
DROP TABLE IF EXISTS photos;
CREATE TABLE photos (
     photoID   SERIAL not null,
     cid       SERIAL not null references colleges(cid),
     pic       text,
primary key(photoID)
);
```

Functional Dependencies:

photoID -> pic

Sample Data:

photoID	cid	pic
1	1	http://www.maristmy575.com/wp-content/uploads/2010...
2	1	http://jobs.marist.edu/images/river.jpg
3	2	http://www.siena.edu/uploadedimages/home/About_Sie...
4	2	http://www.siena.edu/uploadedImages/Home/Student_L...
5	2	http://www.siena.edu/images/system/academic%20quag...
6	3	http://postdocs.stanford.edu/images/stanford_quad....
7	4	http://images.fastcompany.com/upload/yale%20law%20...
8	5	http://colleges.usnews.rankingsandreviews.com/img/...
9	6	http://www.filthylucre.com/wp-content/uploads/2009...
10	7	http://www.onlinecollegesdatabase.org/wp-content/u...
11	8	http://uadmissions.georgetown.edu/image/1242764887...

Users

Table:

```
DROP TABLE IF EXISTS users;
CREATE TABLE users (
     userID              SERIAL not null,
     firstName           TEXT,
     lastName            TEXT,
     email               TEXT,
     passwd              VARCHAR(15),
     gpa                 NUMERIC(2, 1),
primary key(userID)
);
```

Functional Dependencies:

userID -> firstName
userID -> lastName
userID -> email
userID -> passwd
userID -> gpa

Sample Data:

userID	firstName	lastName	email	password	gpa	access This is the level of the user. What they have access to add, update, review, etc.
1	Brenden	Bishop	brenden.b	16d7a4fca	3.4	1

Searches

Table:

```
DROP TABLE IF EXISTS searches;
CREATE TABLE searches (
     searchID       SERIAL not null,
     userID         SERIAL not null references
                    users(userID),
     cid            SERIAL not null references
colleges(cid),
primary key(searchID)
);
```

Functional Dependencies:

None associative table.

Sample Data:

No data yet since the website is not live. Basically this table is an associative table between the userID's and what college searches they have saved. Once they are logged into the site, they can then save certain college listings and or searches, once that happens this table will be populated with rows containing an auto incrementing search id, followed by the userID, and then which CollegeID he or she had saved. This will allow the users on the site to save their top choices and revisit them easily later on.

Majors

Table:

```
DROP TABLE IF EXISTS majors;
CREATE TABLE majors (
     majid          SERIAL not null,
     cid            SERIAL not null references
                    colleges(cid),
     name           VARCHAR(50),
primary key(majid)
);
```

Functional Dependencies:

majid -> name

Sample Data:

majid	cid	name
1	1	Accounting
2	1	American Studies
3	1	Applied Mathematics
4	1	Athletic Training
5	1	Biochemistry (B.A.)
6	1	Biochemistry (B.S.)
7	1	Biology
8	1	Biology Education
9	1	Biomedical Sciences
10	1	Business Administration
11	1	Finance
12	1	Human Resource Management
13	1	International Business
14	1	Marketing
15	1	Chemistry (B.A.)
16	1	Chemistry (B.S.)
17	1	Communication
18	1	Advertising
19	1	Communication Studies
20	1	Journalism
21	1	Public Relations
22	1	Sports Communication
23	1	Computer Science
24	1	Computer Science/Game Design
25	1	Criminal Justice
26	1	Digital Media
27	1	Economics
28	1	Education
29	1	Adolescence Education (Grades 7-12)
30	1	Childhood/Special Education (Grades 1-6)

Minors

Table:

```
DROP TABLE IF EXISTS minors;
CREATE TABLE minors (
     minid                SERIAL not null,
     cid            SERIAL not null references
colleges(cid),
     name           VARCHAR(50),
primary key(minid)
);
```

Functional Dependencies:

minid -> name

Sample Data:

minid	cid	name
1	1	Accounting
2	1	Advertising
3	1	African Diaspora Studies
4	1	American Studies
5	1	Anthropology
6	1	Art (Studio)
7	1	Art History
8	1	Biology
9	1	Business
10	1	Catholic Studies
11	1	Chemistry
12	1	Cinema Studies
13	1	Communication Studies
14	1	Computer Science
15	1	Criminal Justice
16	1	Digital Video Production
17	1	Economics
18	1	English - Literature
19	1	English - Theatre
20	1	English - Writing
21	1	Environmental Policy
22	1	Environmental Science
23	1	Environmental Studies
24	1	Fashion Merchandising
25	1	French
26	1	General Communication
27	1	Global Studies
28	1	History
29	1	Hudson River Valley Regional Studies
30	1	Information Systems

Professor Rating

Table:

```
DROP TABLE IF EXISTS prof_rating;
CREATE TABLE prof_rating (
     prof_id            SERIAL not null,
     cid                SERIAL not null references
                        colleges(cid),
     rating             NUMERIC(2, 1),
primary key(prof_id)
);
```

Functional Dependencies:

prof_id -> rating

Sample Data:

prof_id	cid	rating
1	1	3.5
2	2	2.9
3	3	4.1
4	4	3.9
5	5	4.3
6	6	4.5
7	7	2.3
8	8	3.7

Residence Rating

Table:

```
DROP TABLE IF EXISTS res_rating;
CREATE TABLE res_rating (
     res_id         SERIAL not null,
     cid            SERIAL not null references
                    colleges(cid),
     rating         NUMERIC(2, 1),
primary key(res_id)
);
```

Functional Dependencies:

res_id -> rating

Sample Data:

res_id	cid	rating
1	1	4
2	2	2.9
3	3	4.9
4	4	4.1
5	5	3.9
6	6	4.5
7	7	1
8	8	4.2

Nightlife Rating

Table:

```
DROP TABLE IF EXISTS nightlife_rating;
CREATE TABLE nightlife_rating (
     night_id       SERIAL not null,
     cid            SERIAL not null references
                    colleges(cid),
     rating         NUMERIC(2, 1),
primary key(night_id)
);
```

Functional Dependencies:

night_id -> rating

Sample Data:

night_id	cid	rating
1	1	4.5
2	2	2.7
3	3	4.8
4	4	4.1
5	5	3.7
6	6	3.2
7	7	3.3
8	8	4.6

Food Rating

Table:

```
DROP TABLE IF EXISTS food_rating;
CREATE TABLE food_rating (
     food_id        SERIAL not null,
     cid            SERIAL not null references
                    colleges(cid),
     rating         NUMERIC(2, 1),
primary key(food_id)
);
```

Functional Dependencies:

food_id -> rating

Sample Data:

food_id	cid	rating
1	1	2.5
2	2	3
3	3	2.9
4	4	4.3
5	5	3.9
6	6	4.1
7	7	2.9
8	8	3.6

Campus Rating

Table:

```
DROP TABLE IF EXISTS campus_rating;
CREATE TABLE campus_rating (
     camp_id            SERIAL not null,
     cid                SERIAL not null references
                        colleges(cid),
     rating             NUMERIC(2, 1),
primary key(camp_id)
);
```

Functional Dependencies:

camp_id -> rating

Sample Data:

camp_id	cid	rating
1	1	5
2	2	3.3
3	3	4.9
4	4	4.1
5	5	4.6
6	6	3.8
7	7	2.1
8	8	4

Technology Rating

Table:

```
DROP TABLE IF EXISTS tech_rating;
CREATE TABLE tech_rating (
     tech_id            SERIAL not null,
     cid                SERIAL not null references
                        colleges(cid),
     rating             NUMERIC(2, 1),
primary key(tech_id)
);
```

Functional Dependencies:

tech_id -> rating

Sample Data:

tech_id	cid	rating
1	1	4.7
2	2	3.9
3	3	4.8
4	4	4.1
5	5	3.9
6	6	4.3
7	7	2.2
8	8	3.7

Acceptance

Table:

```
DROP TABLE IF EXISTS acceptance;
CREATE TABLE acceptance (
     ac_id              SERIAL not null,
     cid                SERIAL not null references
                        colleges(cid),
     percent            NUMERIC(4, 1),
primary key(ac_id)
);
```

Functional Dependencies:

ac_id -> percent

Sample Data:

ac_id	cid	percent
1	1	14.3
2	2	16.4
3	3	8
4	4	11
5	5	8
6	6	6
7	7	100
8	8	8

Surrounding Area

Table:

```
DROP TABLE IF EXISTS surrounding_area;
CREATE TABLE surrounding_area (
     area_id            SERIAL not null,
     cid                SERIAL not null references
                        colleges(cid),
     rating             NUMERIC(2, 1),
     description        TEXT,
primary key(area_id)
);
```

Functional Dependencies:

area_id -> rating
area_id -> description

Sample Data:

area_id	cid	rating	description
1	1	2.5	The City of Poughkeepsie is on the western edge of...
2	2	0	Siena College, located in Loudonville, at the hear...
3	3	0	The Stanford campus offers a wealth of sightseeing...
4	4	0	Yale University is located in the heart of downtow...
5	5	0	The Princeton area, which has a population of appr...
6	6	0	The local scene Cambridge and Boston are famous...
7	7	0	Nassau Community College, located in Garden City, ...
8	8	0	The campus is nestled in the historic Georgetown n...

Campus Jobs

Table:

```
DROP TABLE IF EXISTS campus_jobs;
CREATE TABLE campus_jobs (
     job_id              SERIAL not null,
     cid                 SERIAL not null references
                         colleges(cid),
     jobs                VARCHAR(3),
     description         TEXT,
primary key(job_id)
);
```

Functional Dependencies:

job_id -> jobs
job_id -> description

Sample Data:

job_id	cid	jobs	description
1	1	Yes	Residential Networking, Help Desk, Telecommunicait...
2	2	Yes	There are jobs on campus.
3	3	Yes	There are jobs on campus.
4	4	Yes	There are jobs on campus.
5	5	Yes	There are jobs on campus.
6	6	Yes	HelpDesk, Fitness, Tutoring, etc...
7	7	Yes	There are jobs on campus.
8	8	Yes	There are jobs on campus.

Clubs/Activities

Table:

```
DROP TABLE IF EXISTS clubs_activities;
CREATE TABLE clubs_activities (
     club_id            SERIAL not null,
     cid                SERIAL not null references
                        colleges(cid),
     rating             NUMERIC(2, 1),
     description        TEXT,
primary key(club_id)
);
```

Functional Dependencies:

club_id -> rating
club_id -> description

Sample Data:

club_id	cid	rating	description
1	1	3.5	Some of the clubs offered, this is not including i...
2	2	0	Accounting Students Association Amercian Market...
3	3	0	Challenge for Charity C4C is a nonprofit organiza...
4	4	0	A total list of the clubs can be found here: https...
5	5	0	Student organizations are created and run by stude...
6	6	0	The Office of Student Life integrates the academic...
7	7	0	You've heard the advice before -- get involved, jo...
8	8	0	Georgetown encourages students to actively engage ...

About

Table:

```
DROP TABLE IF EXISTS about;
CREATE TABLE about (
     about_id           SERIAL not null,
     cid                SERIAL not null references
                        colleges(cid),
     description        TEXT,
primary key(about_id)
);
```

Functional Dependencies:

about_id -> description

Sample Data:

about_id	cid	description
1	1	Marist College, recognized for excellence by U.S. ...
2	2	Founded in 1937, Siena College is a private, Catho...
3	3	Stanford University is one of the world's leading ...
4	4	Founded in 1701, Yale University is the third olde...
5	5	Princeton University is a vibrant community of sch...
6	6	Founded in 1636, Harvard University is the first i...
7	7	A great career. A new direction. The excellent pre...
8	8	Georgetown University is a private Jesuit research...

Student Body

Table:

```
DROP TABLE IF EXISTS student_body;
CREATE TABLE student_body (
     body_id            SERIAL not null,
     cid                SERIAL not null references
                        colleges(cid),
     description        TEXT,
primary key(body_id)
);
```

Functional Dependencies:

body_id -> descrption

Sample Data:

body_id	cid	description
1	1	Life at Marist outside the classroom offers a weal...
2	2	Over 60 organizations featuring service, student g...
5	3	Stanford is a residential teaching and research un...
6	4	The richness of life at Yale extends far beyond th...
7	5	A vast range of cultural, educational, athletic an...
8	6	Harvard University has around 20,000 students acro...
9	7	Expect a great community! After all, community is ...
10	8	With more than 6,300 undergraduates from all 50 st...

Views

Views can be useful because they provide a simplified, easy to access version of some useful, popular, or prominent queries. For example, when searching for colleges one of the main characteristics people factor is the price of the school. This view will show all of the colleges that have a total cost of under $50k per year. This includes tuition and room and board. These types of views will be used in the website and how we query certain results, users will be able to choose the conditions of the search.

For example:

```
CREATE VIEW CollegesUnder50k as
(SELECT name, tot_price
FROM colleges
WHERE tot_price BETWEEN 0 and 50000);
```

An example of a more advanced view may consist of trying to find a college that is of type private, has a totprice of under 50k, is Division 1, and has jobs on campus.

```
CREATE VIEW college_jobs as
(SELECT colleges.name, colleges.tot_price,
athletics.division, campus_jobs.jobs
     FROM colleges
     JOIN colleges
     ON colleges.tot_price < 50000
     JOIN types
     ON types.cid = colleges.cid
     AND tid = '3'
     JOIN campus_jobs
     ON colleges.cid = campus_jobs.cid
     AND campus_jobs.jobs = 'yes'
     JOIN athletics
     ON campus_jobs.cid = athletics.cid
     AND athletics.division = '1')
GROUP BY cid;
```

Reports and Queries

Here are some select statements that commonly used reports provided by the database.

Colleges under $55k total price with a campus rating of at least 3.0 and a picture of the campus.

This report shows what colleges fit the criteria above provided by the user.

```
SELECT colleges.name, campus_rating.rating, photos.pic
FROM colleges, campus_rating, photos
WHERE cid in (SELECT campus_rating.cid
            FROM campus_rating
            WHERE rating > 3.0);
```

Another potentially common report would be a list of all of the Division 1 State Schools with Computer Science as a major.

```
SELECT colleges.name, colleges.tot_price, colleges.gpa,
colleges.size, colleges.ovr_rating
FROM colleges, majors, athletics, types
WHERE colleges.cid
IN (SELECT athletics.cid
     FROM athletics
     WHERE division = '1'
     AND types.cid IN (SELECT types.cid
                 FROM types, majors
                 WHERE tid = '5'
                 AND majors.name = 'Computer Science);
```

Stored Procedures

Stored procedures are very efficient because they reduce the number of round trips between application and database server. All SQL statements are wrapped inside a function stored in the database server so the application only has to issue a function call to get the result back instead of sending multiple statements and wait for the result between each. They also, increase the application performance because user-defined functions are pre-complied and stored in the database server. In addition, once a function is developed, any application can use it.

Here is an example of a stored procedure that lets the user choose the type of college and it will return a table listing the colleges ordered by their overall rating.

```
CREATE FUNCTION bestCollegeType(tid int) returns
table(name varchar, ovr_rating numeric) as $$
    SELECT name, ovr_rating FROM colleges, types
     WHERE tid=$1
     ORDER BY ovr_rating desc;

$$ language 'sql';
```

Here is another example showing the smallest college size depending on the college type the user specifies.

```
CREATE FUNCTION bestCollegeType(size int) returns
table(name varchar, size int) as $$
    SELECT name, size FROM colleges, types
     WHERE tid=$1
     ORDER BY size asc;

$$ language 'sql';
```

Triggers

Triggers are useful because they can be used for the college total prices. Lets say at the start of a new semester a college decides to raise tuition by a specific amount. A trigger can be used to simply add that amount to the current tuition so that periodic increments in tuition or room and board or taxes etc. will not have to be calculated manually and then add the total back into the database.

For example:

```
CREATE TRIGGER update_totprice
AFTER UPDATE OF tot_price ON colleges
FOR EACH ROW
EXECUTE PROCEDURE update_price();
Trigger Function:

CREATE FUNCTION update_price()
RETURNS 'TRIGGER'
AS $BODY$
BEGIN
IF (TG_OP = 'UPDATE') THEN
UPDATE colleges SET tot_price = old.tot_price +
new.tot_price
WHERE OLD.cid = NEW.cid;
END IF;
RETURN NULL;
END;

$$ LANGUAGE plpgsql;
```

Security

There are a few security precautions and features that are already implemented and there are future ones to be implemented on the website as well. Mainly the security deals with user accounts, which includes prospective students, alumni, and or colleges. All users will be allowed to save their searches however where it lies different is in the access level. If a user has access level of 1 it is a normal user, if it has access level of 2, it means they are a current student and or alumni. Depending on what the email address is of the level 2, they can post reviews to college listings ONLY if the @edu email matches the one of the college they are looking to post on. This preserves the integrity of the reviews for the colleges. Finally, a level 3 access is a college and once logged in, they will have the ability to add/update a college, once again only if it matches their @edu email.

The registration page is not active yet, however it will have the user be able to choose, prospective student, current student/alumni, or college representative. That will then be passed into a contact form to administrators of the database who will then verify the status of which they have chosen, based on the @edu email address and communication with the colleges for their representatives.

The last form of security comes with the user accounts. The passwords are inserted using the MD5 function provided by PHP this encrypts the data to a length of 40 random characters of letters and numbers. This will be applied through the PHP inserts and then in addition that string will be encrypted using a Salt Key. A salt key is a random string of charcters and integers that is used to encrypt hash character of a string differently. The salt key is user defined and therefore never the same and is difficult to decrypt.

All of these security features and precautions will be implemented upon full completion and future enhancements of the CollegeCrux database.

Implementation Notes/ Known Problems/ Future Enhancements

As far as the implementation, I would have liked for the database to have had a more tables and constraints particularly in the user accounts and reviewing portion of the website. The only problem is that the reviews, comments, and messaging system has not been planned or implemented yet and is a future implementation.

It would have been nice to be able to incorporate all necessary into this version of the database, however, with the abundance of categories taken into account, having everything would have been far complex and time consuming. On the time provided I believe this version does provide usefulness and paves the way for the future as well as providing a good basis to start from.

The future intention is to not only have listings for colleges but also for banking/loaning institutions and or brokers, as well as textbook trading and selling internally within our site. Another possible enhancement is Facebook integration. See where you're friends applied, and or are looking, see who went where, and who's going where? The banking listings can be applied in the very same manner the colleges have been and will offer prospective students and parents and insight into how they can fund their money for their education. To be able to research the best college fit and the best funding/savings program for that particular user, all in one place, is a niche that has yet to be addressed. No more searching individually for schools back and forth using Google, and then researching good bank accounts/loan/savings programs for the student. It can all be done and done accurately and efficiently in one place, CollegeCrux.com. To see the test/beta website please visit http://www.collegecrux.com/test.php