

Database Systems Design Project

Marist Connections

Daniel Cody

Professor Labouseur

18 November 2014

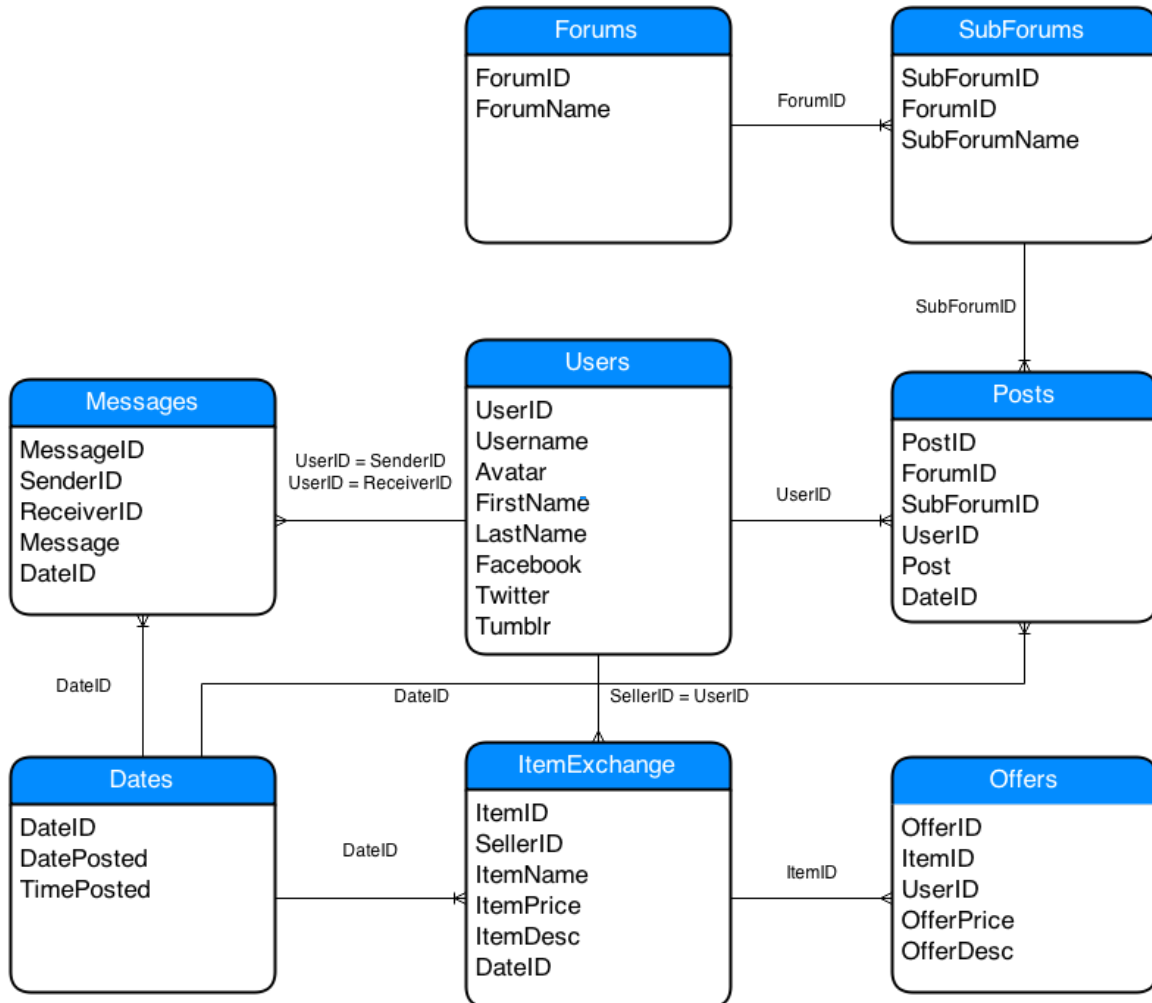
Table of Contents

Database Description	3
ER Diagram	4
Forums Table	5
SubForums Table	6
Date Table	7
Posts Table	8
Users Table	9
Messages Table	11
ItemExchange Table	12
Offers Table	13
Views	14
Queries	15
Store Procedures	16
Triggers	17
Security	18
Known Issues	19
Future Enhancements	20

Database Description:

My design project is to design a database for a Social Networking Website for Marist College. This website lets users sign up, connect their different social media accounts to their username. Then they are allowed to post on forums to ask about what is happening on their college campus. Users can message each other to meet people and communicate with people that they do not know. Also the users are also allowed to post items to be sold on their campus such as books, laptops, dorm room accessories, electronics, and more.

ER Diagram



Forums Table

Description:

This table is needed to keep track of the general forum topics such as classes, teachers, nightlife, intramurals and more.

Functional Dependency:

ForumID -> ForumName

Create Statement:

```
CREATE TABLE Forums (  
    ForumID serial NOT NULL,  
    ForumName text NOT NULL,  
    PRIMARY KEY(ForumID)  
);
```

Example Data:

ForumID	ForumName
1	Classes
2	Teachers
3	Intramurals

SubForums Table

Description:

This table is derived from the forums table, which based on the large forum topic such as teachers; you can select a specific forum on that topic such as Alan Labouseur or Benjamin Carle.

Functional Dependency:

SubForumID -> ForumID, SubForumName

Create Statement:

```
CREATE TABLE SubForums (
    SubForumID serial NOT NULL,
    ForumID integer NOT NULL REFERENCES Forums(ForumID),
    SubForumName text NOT NULL,
    PRIMARY KEY(SubForumID)
);
```

Example Data:

SubFormID	ForumID	SubForumName
1	1	Database Systems
2	1	Operating Systems
3	2	Alan Labouseur
4	2	Benjamin Carle
5	3	Soccer

Date Table

Description:

This table keeps track of all the different dates in which things happened by keeping an ID, which relates to a specific date and time

Functional Dependency:

DateID -> DatePosted, Time

Create Statement:

```
CREATE TABLE Dates(
    DateID serial NOT NULL,
    DatePosted date NOT NULL,
    Time integer NOT NULL,
    PRIMARY KEY(DateID)
);
```

Example Data:

Dates Table

DateID	DatePosted	Time
1	9/13/14	3:07 PM
2	9/14/14	5:21 PM
3	9/14/14	11:17 AM
4	9/14/14	12:30 PM
5	9/15/14	1:53 PM
6	9/20/14	3:07 PM
7	9/20/14	9:42 AM
8	9/22/14	8:01 AM
9	9/25/14	9:07 PM
10	9/29/14	3:21 PM
11	9/30/14	7:34 AM
12	10/1/14	2:42 PM
13	10/5/14	12:51 PM
14	10/6/14	11:00 AM

Posts Table

Description:

This table is used to keep track of the posts that are made on a specific Sub Forum.

Functional Dependency:

PostID -> ForumID, SubForumID, UserID, Post, Time

Create Statement:

```
CREATE TABLE Posts (
    PostID serial NOT NULL,
    ForumID integer NOT NULL REFERENCES Forums(ForumID),
    SubForumID integer NOT NULL REFERENCES SubForums(SubForumID),
    UserID integer NOT NULL REFERENCES Users(UserID),
    Post text NOT NULL,
    DateID integer NOT NULL REFERENCES Dates(DateID),
    PRIMARY KEY(PostID)
);
```

Example Data:

PostID	SubForumID	ForumID	Post	DateID
1	1	1	Great class, learn a ton	1
2	2	1	Hard but worth the work	2
3	2	1	Don't wait till last min for HW	3
4	3	2	Alan is the man, 007	4
5	4	2	Hard, but very rewarding	7
6	5	3	Anyone want to make a team?	10
7	4	2	Hard grader	11
8	5	3	I am down to make a team!	12
9	1	1	Love the design project	14

Users Table

Description:

This table is used to keep track of each individual user, their first and last name as well as their social media information so they can link their accounts to their social media accounts.

Functional Dependency:

UserID → Username, Password, Avatar, FirstName, LastName, Facebook, Twitter, Tumblr,

Create Statement:

```
CREATE TABLE Users (  
    UserID serial NOT NULL,  
    Username text NOT NULL,  
    Password text NOT NULL,  
    Avatar text DEFAULT 'Avatar.png',  
    FirstName text NOT NULL,  
    LastName text NOT NULL,  
    Facebook text DEFAULT 'facebook.com',  
    Twitter text DEFAULT 'twitter.com',  
    Tumblr text DEFAULT 'tumblr.com',  
    PRIMARY KEY(UserID)  
);
```

Example Data:

UserID	Username	Password
1	dcods22	\$1\$O3JMY.Tw\$ewfdLnLjQ/ewfweF9.MTp3gHv/
2	btbishop	\$1\$O3JMY.Tw\$AdLnLjQ/5jXF9.MTp3gHv/
3	bigPhilly	\$1\$O3JMY.Tw\$AdLewfeLjQ/5jXF9.ewfgHv/
4	bicepsfordays	\$1\$O3JMY.Tw\$AntyLjQ/5jXF9tyntyp3gHv/

FirstName	LastName	Facebook	Twitter	Tumblr
Dan	Cody	Dan Cody	dcods22	Dan_Cody
Brenden	Bishop	Bren Bishop	bbishop93	BT_Bishop
Phillip	Picinic	Phil Picinic	bigPhil67	Big_Phil
Michael	Figguerito	Mike Figs	bigBiceps11	Mikey_Biceps

Messages Table

Description:

This table is used to keep track of the messages sent between users.

Functional Dependency:

MessageID -> SenderID, ReceiverID, Message, Time

Create Statement:

```
CREATE TABLE Messages (
    MessageID serial NOT NULL,
    SenderID integer NOT NULL,
    ReceiverID integer NOT NULL,
    Message text NOT NULL,
    DateID integer NOT NULL REFERENCES Dates(DateID),
    PRIMARY KEY(MessageID)
);
```

Example Data:

MessageID	SenderID	ReceiverID	Message	DateID
1	1	2	Wanna watch the MNF game?	5
2	2	1	Yea Ill be right over	6
3	3	1	Hows AI going?	8
4	4	2	Want to go to the caf?	9
5	1	3	Awful I dropped	13

ItemExchange Table

Description:

This table is used to keep track of the items that are posted in the item exchange to be sold between people on campus.

Functional Dependency:

ItemID -> SellerID, ItemName, ItemPrice, ItemDesc, DatePosted

Create Statement:

```
CREATE TABLE ItemExchange (
    ItemID serial NOT NULL,
    SellerID integer NOT NULL REFERENCES Users(UserID),
    ItemName text NOT NULL,
    ItemPrice integer NOT NULL,
    ItemDesc text NOT NULL,
    DateID integer NOT NULL REFERENCES Dates(DateID),
    PRIMARY KEY(ItemID)
);
```

Example Data:

ItemID	SellerID	ItemName	ItemPrice	ItemDesc
1	1	Samsung 32" TV	200	Just a year old
2	1	Lazy Boy	50	Slight rip in leather
3	2	Database Textbook	100	Barley Used, good read
4	4	Dumbbells	20	Very Heavy

Offers Table

Description:

This table is used to keep track of the offers that are posted on a specific Item in the exchange.

Functional Dependency:

OfferID -> ItemID, UserID, OfferPrice, OfferDesc

Create Statement:

```
CREATE TABLE Offers (
    OfferID serial NOT NULL,
    ItemID integer NOT NULL REFERENCES ItemExchange(ItemID),
    UserID integer NOT NULL REFERENCES Users(UserID),
    OfferPrice integer NOT NULL,
    OfferDesc text NOT NULL,
    PRIMARY KEY(OfferID)
);
```

Example Data:

OfferID	ItemID	UserID	OfferPrice	OfferDesc
1	1	3	125	Is it 1080p
2	1	2	150	Any Scratches?
3	4	1	15	How Heavy?

Views

Description:

Views are extremely useful to display certain pieces of the data in one format, where it is not stored in that way in the table. It is very helpful when trying to set permissions or only allowing certain people to have access to certain data. This could be very useful in a case of a school, where all the students information should not be available to everyone, but might be stored all in one table. Information such as social security numbers, and addresses are not important to teachers, but my contact info might be.

Example:

CREATE OR REPLACE VIEW PostInfo

AS

```
SELECT f.ForumName, s.SubForumName, u.UserName, p.Post, p.DatePosted
FROM Users u
INNER JOIN Posts p ON p.UserID = u.UserID
INNER JOIN Subforums s ON p.SubForumID = s.SubForumID
INNER JOIN Forums F ON s.ForumID = f.ForumID;
```

Queries

Description:

Queries are used to change, update, or select information from a database. These statements can be written to select data from multiple tables, and also can have different statements attached to it, which will select different data. Set operations, joins, keywords, and mathematic functions can be applied to each query, which will return a special set of data.

Examples:

```
SELECT f.forumName, s.subForumName FROM
```

```
subforums s INNER JOIN forums f ON f.forumID = s.forumID;
```

ForumName	SubForumName
Classes	Database Systems
Classes	Operating Systems
Teachers	Alan Labouseur
Teachers	Benjamin Carle
Intramurals	Soccer

```
SELECT * FROM Posts
```

```
WHERE subForumID IN
```

```
(SELECT subForumID FROM SubForums
```

```
WHERE SubForumName = 'Alan Labouseur');
```

PostID	SubForumID	ForumID	Post	DateID
4	3	2	Alan is the man, 007	4

Store Procedures

Description:

A store procedure is a compilation of SQL statements that are put together into one function to do a specific task.

Example:

```
CREATE OR REPLACE FUNCTION MostUsedForum() RETURNS  
TABLE (forumName text, count integer) as $$
```

```
    SELECT forumID, max(num)  
  
    FROM  
  
        (SELECT forumID, COUNT(forumID) as num  
  
        FROM subForums GROUP BY forumID)  
  
    subForums  
  
    GROUP BY forumID  
  
    ORDER BY MAX DESC  
  
    LIMIT 1;
```

```
$$ language 'sql' $$
```


Triggers

Description:

A trigger can be used to do an function upon another event on a table.

Example:

```
CREATE TRIGGER UpdateOnOffer ON offers FOR INSERT AS
BEGIN
    SELECT @itemID =
        (SELECT itemID FROM offers
         ORDER BY itemID DESC LIMIT 1);
    SELECT @buyerID =
        (SELECT userID FROM offers
         ORDER BY itemID DESC LIMIT 1);
    SELECT @sellerID =
        (SELECT sellerID FROM ItemExchange
         WHERE itemID = @itemID);
    SELECT @dateNum =
        (SELECT dateID FROM offers
         ORDER BY itemID DESC LIMIT 1);
    INSERT INTO messages
        (SenderID, ReceiverID, Message, DateID)
        VALUES @buyerID, @sellerID, "You Received an offer on your item",
        @dateNum;
END;
```

Security

There is a decent amount of security already placed on this database. The relationships between tables are enforced by the create statements. This helps enforce insert, update and delete anomalies.

In addition the passwords in the Users table are not stored in plain text they are encrypted with an MD5 crypt hash. Storing passwords in plain text can create major issues if your database is ever breached, which is clearly an issue

Also roles can be created that allow only certain users to do certain things, such as a campus representative can only edit the content on the forums.

Example:

```
CREATE ROLE db_admin  
  
GRANT SELECT, INSERT, DELETE  
  
ON Forums, SubForums, Posts, Users, Messages, ItemExchange, Offers, Dates  
  
TO db_admin;
```

```
CREATE ROLE campus_rep  
  
GRANT SELECT, INSERT, DELETE  
  
ON Forums, SubForums, Posts  
  
TO campus_rep;
```

Known Issues

A known issue to this point with the database is, the handling of group messages since there only exists the ability to have a direct message to a user since there is only a sender and a receiver of a message

Another known issue with the database is the ability to link more social media accounts, such as Google+ or Foursquare. The database currently is only setup to have a link to your Facebook, Twitter, and Tumblr accounts.

Future Enhancements

Some future enhancements to the website could be to include more of a specific teacher rating system instead of a forum for it. This would entail having a teacher's table, which could also track the classes they teach. Then the users could rate the specific teachers as well as the classes they took.

Another future enhancement could be to also include a more specific Item Exchange to the website, where the items could be sorted into types such as books, furniture, electronics and more. This would add a table for the item type to the database, which would make the website searchable by each category.