Project 1 - Relational Database Design

Group C – David, Jas, George, & Harris

June 11th, 2014

Table of Contents

Introduction

The purpose of this project is to design an alumni database system based on the provided requirements. The system will keep track of an alumnus's involvement in clubs, events and donations.

Business Rules

- A member can be enrolled in zero to many clubs.
- A club has zero or more members.
- A member has zero or more interests.
- An interest can be shared by zero or more members.
- A member studies one or more subjects.
- A subject can be studied by zero to many members.
- A member can make zero to many alumni fee payments.
- An alumni fee payment belongs to one and only one member.
- Alumni fee must be paid for the member to stay as an active alumni.
- Membership start and end date need to be recorded for all current and past club enrolments.
- Members must pay fees on top of alumni fee to be enrolled in a club.
- Members can pay for zero to many clubs.
- A club can receive payments from zero to many members.
- Fees for clubs have fixed price but can be different for each club.
- Fee covers one year of membership.
- There are 3 states that a member can be in: Active, Holding and Former.
- If membership fee is up-to-date, then member status is Active.
- If membership fee is not up-to-date but is overdue for less than one year, then member status is Holding.
- If membership fee is not up-to-date and is overdue for more than one year, then member status is Former.
- An event can be associated with zero or one club.
- A club can be associated with zero to many events.
- An event includes one or more members.
- A member can be invited to zero to many events.
- Members are invited based on criterions including club memberships, interests, postal code, country, graduation year and subject studied.
- Members must pay fees on top of alumni fee to join an event.
- A member can pay for zero to many events.
- An event can receive payments from zero to many members.
- Fees for events have fixed price but can be different for each event.

- If a member did not pay for an event, he or she may receive a different invitation that offers the opportunity to pay associated club fee or event fee.
- A member can make zero to many donations.
- A donation comes from one and only one member.
- A donation can be for zero club (general donation) or one club (club specific donation).
- A donation can be "one-off" or regular and run for a given time.
- The type of donation needs to be recorded.

Process Description

1. We read over the description and identified a set of business rules that the system must conform to.

2. From the set of business rules, we agreed that the design can be split into three external models: club involvement, event involvement and donation.

3. Starting with club involvement, we identified a list of entities and their associated attributes.

4. Next we identified the relationships between the entities and their connectivity. We noted down the many to many relationships as we understand that bridge entities need to be introduced later to handle this type of relationship.

5. We identified the list of bridge entities that are necessary to implement the many to many relationships and their associated attributes.

6. At this point we have the external model for the club involvement part of the system.

7. We repeat steps 3 - 5 to create the external model for the event involvement part and donation part of the system.

8. We merge the three external model into the conceptual model.

9. We applied the process of Normalization to optimize the design.

Final Entities

HISTORIC_MEMBERSHIP (MEM_ID, CLUB_ID, START_DATE, END_DATE)

MEM_ID	CLUB_ID	START_DATE	END_DATE
1	1	1/1/2014	5/1/2014
4	1	1/3/2012	2/4/2013
2	2	1/1/2014	5/1/2014

ALUMNI (MEM_ID, F_NAME, L_NAME, POST_CODE, ALUM_STATUS)

MEM_ID	F_NAME	L_NAME	POST_CODE	ALUM_STATUS
1	Jas	Sohi	v6p4y1	Active
2	James	Brown	v5w3m6	Holding
3	John	Wayne	v7t3y8	Former
4	Jessica	Smith	v8q5t7	Active

DONATION (DONATION_ID, MEM_ID, CLUB_ID, DONATION_DATE, DONATION_AMT, START_DATE, END_DATE)

DONATION_ID	MEM_ID	CLUB_ID	DONATION_DATE	DONATION_AMT	START_DATE	END_DATE
1	1	1	9/9/2013	$ 5.00	9/9/2013	
2	1	2	9/9/2013	$ 34.00	9/9/2013	9/9/2014
3	1	4	9/9/2013	$ 69.00	9/9/2013	
4	2	4	8/1/2013	$ 83.00	8/1/2013	8/1/2014
5	3	3	8/25/2013	$ 93.00	8/25/2013	

CLUB_PAYMENT (PAYMENT_ID, CLUB_ID, MEM_ID, PAY_AMOUNT, PAY_DATE)

PAYMENT_ID	EVENT_ID	MEM_ID	PAY_AMOUNT	PAY_DATE
1	3	1	$55.45	4/3/2010
2	3	1	$35.55	1/3/2013
3	2	1	$25.00	5/18/2014

INVITATION (MEM_ID, EVENT_ID, INV_TYPE)

MEM_ID	EVENT_ID	INV_TYPE
4	1	Standard
4	2	Standard
3	2	Standard

CLUB (CLUB_ID, CLUB_TYPE)

CLUB_ID	CLUB_TYPE
1	French Club
2	Reading Club
3	Chess Club

EVENT (EVENT_ID, EVENT_DATE, EVENT_NAME, EVENT_LOCATION)

EVENT_ID	EVENT_DATE	EVENT_NAME	EVENT_LOCATION
1	5/14/2014	Volleyball Finals	BCIT Gymnasium
2	3/7/2013	Chess Club Fundraiser	The Roxy
3	2/2/2012	Book Club Winter Meetup	VPL Downtown

ALUMNI_PAYMENT (PAYMENT_ID, MEM_ID, PAY_DATE)

PAYMENT_ID	MEM_ID	PAY_DATE
1	4	5/3/2010
2	1	7/3/2013
3	1	5/3/2014

CURRENT_MEMBERSHIP (MEM_ID, CLUB_ID, START_DATE, END_DATE, CLUB_STATUS)

MEM_ID	CLUB_ID	START_DATE	END_DATE	CLUB_STATUS
1	4	1/1/2014		Active
4	2	1/3/2012		Active
2	5	1/1/2014		Active

FEE (FEE_ID, CLUB_ID, FEE_AMOUNT, EVENT_ID)

FEE_ID	CLUB_ID	EVENT_ID	FEE_AMOUNT
1	1	1	$50.00
2	2	3	$45.00
3	2	1	$23.00
4	2	2	$14.50

EVENT_PAYMENT (PAYMENT_ID, EVENT_ID, MEM_ID, PAY_AMOUNT, PAY_DATE)

PAYMENT_ID	EVENT_ID	MEM_ID	PAY_AMOUNT	PAYMENT_DATE
1	3	1	$55.45	5/3/2010
2	4	1	$35.55	5/3/2013
3	2	1	$25.00	5/3/2014

Assumptions

Here are the assumptions we made for our model:

- It is possible for the same alumni to quit and re-join the same club multiple times. There is no restriction on the number of times that an alumni can enroll in a club.

- Payment to club covers membership for a year starting with the day the payment is made. In an ideal situation member will pay on the same day that their club membership starts. If that is not the case (e.g. member paid in advance for a future membership), application must bear the responsibility to identify if a fee is overdue as the database system cannot provide that information.

Limitations

Here are some limitations of our model:

- An alumni can only have one interest and one subject. In theory the system can store multiple interests and subjects if the fields are multi-valued - but it's not best practice.

- Fees cannot be changed. The system records the exact payment amount for all payments so it does allow partial payments (e.g. member making a deposit). If a fee is increased, subsequent queries will show that all previous payments are not made in full.

- If a club member currently in holding state pays to renew the membership, the system will not be able to track the time period when the member is in holding state.

- A donation can only be attributed to one alumni. It is not possible to record more than one alumni making the same donation. A workaround is to have one record for each alumni, storing the amount they contributed in the overall donation.

Conclusions/Observations

Following are some reflections we get from completing the project:

- Business rules dictate design - it is imperative to establish a set of correct business rules before proceeding into design. Having clear and concise business rules simplifies the design process. The opposite also holds true.

- It is much easier to first work with parts of the overall system (i.e. external model) and merge the parts together later than to come up with the conceptual model directly.

- Controlled data redundancy, in addition to foreign keys, can be beneficial as it allows easier and faster querying. There is often a tradeoff between space usage and performance.

- Sometimes it can be beneficial to store records of the same nature (i.e. containing the same attributes) in two different tables. An example in this project would be the CURRENT_MEMBERSHIP and HISTORIC_MEMBERSHIP table. Although theoretically they can be merged into one table, having them separate allows faster query in certain situations (e.g. if user just wants to query about current members).

- Surrogate keys are very useful when an entity instance can only be identified by a composite primary key with a large number of primary key elements.

Appendix 1 - ERD – Alumni Database System

Appendix 2 – Sample Reports

Here are some reports that could be queried based on our model:

1. Financial Report - The Accounts Receivable department of the University would need is a list of all Active Alumni who have not paid their club fees in full so they could follow up accordingly.

MEM_ID	ALUM_STATUS	CLUB_FEES PAID
36	Active	No
4	Active	No
2	Active	No
5	Active	No
3	Active	No
54	Active	No
55	Active	No

2. Club Popularity – A report that counts the total number of active members of each club and ranks clubs based on popularity.

TOTAL # OF ACTIVE MEMBERS	CLUB_ID
487	3
390	8
388	2
323	5
232	1
101	9
89	11
67	7
34	4
33	18
20	6
10	12

3. Event Size – A report that lists the number of people that attended the top 7 Events.

NUMBER OF ATTENDEES	EVENT_ID
300	5
279	4
256	9
244	2
200	1
88	3
45	8