INST733 Database Design, Section IM01
Project Report

Introduction

Embarking on the creation of a comprehensive database for the Institute of Applied Agriculture (IAA) at the University of Maryland is a transformative and indispensable endeavor. At the intersection of education, community-building, and strategic outreach, this initiative promises to revolutionize the way IAA engages with prospective students, current enrollees, and its alumni network. Imagine a centralized repository that not only facilitates seamless recruitment efforts by enabling personalized invitations to potential students but also empowers effective communication with current students through tailored correspondences. Moreover, the database serves as a link for alumni engagement, facilitating the organization of alumni events and fostering a sense of belonging and mentorship. As an information hub, this database would be the cornerstone of IAA's efforts, ensuring streamlined processes, enhanced student experiences, and a connected community that extends beyond the academic years. In the landscape of higher education, the creation of such a dynamic database for IAA emerges not just as a practical necessity but as a visionary step towards cultivating a thriving, interconnected, and forward-looking academic community.

This comprehensive database would seamlessly integrate information on prospective students, current students, and alumni, serving as a dynamic tool for IAA faculty and staff to enhance recruitment and retention strategies. By efficiently tracking prospective students, the IAA can streamline its outreach efforts, sending targeted Open House invitations to individuals who express interest at an earlier date in time. For current students, the database facilitates personalized correspondence, ensuring timely and relevant communication to support their academic journey. Alumni engagement is equally crucial, and the database enables the organization of events (such as Alumni Happy Hours), fostering a sense of community and networking among graduates. Furthermore, the database acts as a central hub for distributing INAG News to all stakeholders, ensuring consistent and timely updates on program developments and happenings. In essence, this database not only optimizes operational efficiency for IAA faculty and staff who use it but also strengthens its ability to cultivate a thriving and interconnected community of students and alumni.

Database Design and Implementation

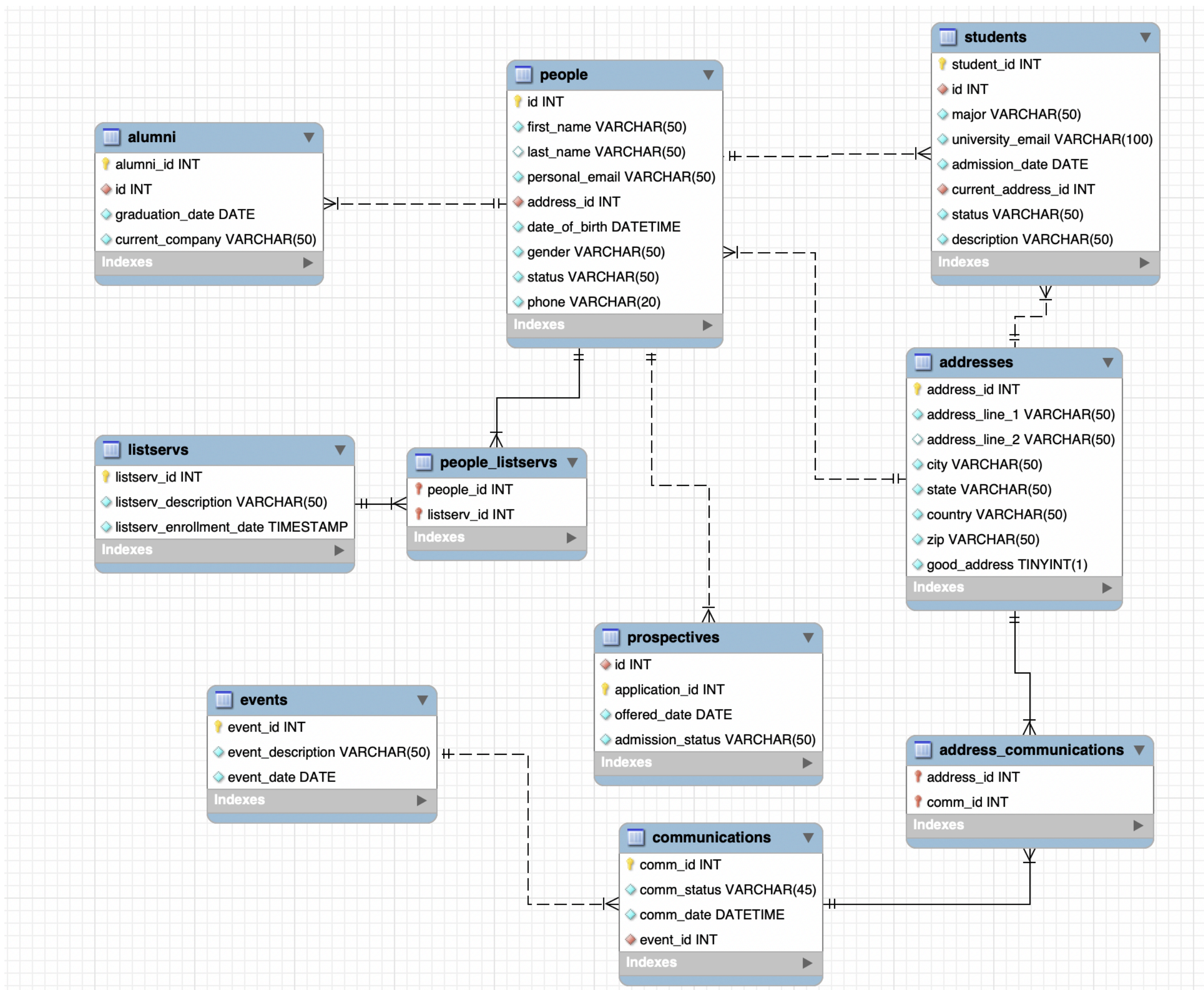

Figure 1 IAA Student Database ERD.

Once we had the ERD established, we could begin obtaining the sample data that would make its way into the database. It was important to grasp how the data would all fit together so that we were efficient and non-duplicative. We utilized Mockaroo (2024) to compile realistic sample data which proved to be surprisingly tedious due to the amount of cleanup required, our journal entries recount some examples. As we imported the data, there was still some fine-tuning required that was previously missed. Below are two examples of sample data.

id	first_name	last_name	personal_email	date_of_birth	address_id	gender	status	phone
1	Papagena	Bansal	pbansal0@disqus.com	11/28/1992	66	Female	Alumni	(315) 6398358
2	Tasia	Strickler	tstrickler1@purevolume.com	7/18/2003	22	Female	Student	(887) 9070558
3	Ronica	Fedorchenko	rfedorchenko2@wp.com	6/21/1992	96	Agender	Student	(650) 1419599
4	Katerina	Franchyonok	kfranchyonok3@dyndns.org	10/21/1989	9	Female	Alumni	(909) 7386595
5	Karyn	Cuthbertson	kcuthbertson4@japanpost.jp	5/28/1989	5	Polygender	Prospective Student	(160) 8626723
6	Hank	Pinn	hpinn5@ezinearticles.com	4/27/1993	46	Male	Prospective Student	(667) 3332383
7	Padraic	Jorck	pjorck6@vinaora.com	10/8/2003	55	Male	Prospective Student	(759) 9041790
8	Aldus	Fairburne	afairburne7@phoca.cz	7/1/1997	19	Male	Alumni	(847) 7197471
9	Viviene	Mudie	vmudie8@elegantthemes.com	3/24/1996	56	Female	Prospective Student	(187) 9057638
10	Reube	Koles	rkoles9@drupal.org	11/4/2001	1	Male	Alumni	(546) 3673103

Figure 2 Example of People Sample Data.

address_id	address_line_1	address_line_2	city	state	country	good_address
45	9 Brentwood Court	Suite 40	Newton	Massachusetts	USA	FALSE
250	14 Steensland Hill	Suite 38	Los Angeles	California	USA	FALSE
121	9961 Truax Terrace	Suite 38	New Castle	Pennsylvania	USA	TRUE
279	067 Summerview Avenue	PO Box 71111	Houston	Texas	USA	TRUE
87	11 Maryland Parkway	Apt 493	Salt Lake City	Utah	USA	FALSE
35	876 Wayridge Point	20th Floor	Peoria	Illinois	USA	FALSE
1	6435 Anhalt Alley	3rd Floor	Fort Worth	Texas	USA	TRUE
27	0650 Westport Court	Room 225	Tallahassee	Florida	USA	FALSE
8	119 Huxley Junction	Suite 26	Washington	District of Columbia	USA	TRUE

Figure 3 Example of Address Sample Data.

To demonstrate some CRUD operations, these are the IAA student database sample queries and the general design factors. For each query, we were able to create a view except for one in which we created a stored procedure.

- Create a dashboard that depicts the number of students who are currently studying and dropped out of university from each major.

  COUNT Status = "Student" and "DropOut"

- Who should be invited to an Alumni Happy Hour in Texas?

  Status = "Alumni" JOIN Addresses WHERE Good Address = "TRUE" and State = "Texas"

- Who should receive the most recent copy of INAG News?

  Status = "Prospective" and "Current" and "Alumni" JOIN Addresses WHERE Good Address = "TRUE" and COUNT Number of Rows.

- What type of listservs are the alumni, prospectives, and students currently enrolled right now?

  Status = "Prospective" and "Current" and "Alumni" JOIN People_Listserv

- What is the latest communication date of each event type?

  Event Description JOIN Communications to Fetch MAX(Communication Date)

- How many communications were sent for each event? How many have been received and how many are yet to be sent?

  Event Description JOIN Communications to Fetch COUNT(Communication ID)

View Name	Req. A	Req. B	Req. C	Req. D	Req. E
student_summary_view	**X**	**X**	**X**		
Stored Procedure: GetAlumniByState("State")	**X**	**X**			
INAG_address_details	**X**	**X**			**X**
enrollment_summary_view	**X**		**X**	**X**	
event_lastest_communication	**X**		**X**		
event_communications_view	**X**		**X**		

Changes from Original Design

From the original design, we drastically changed the database schema, including tables, fields, data types, and relationships between tables, for improved organization and efficiency. We have merged all the address tables into one master table. Additionally, we have merged a majority of the common fields in students, alumni, prospective, and dropouts into a single table called people. We kept the students, alumni, and prospectives tables which record any additional information such as admission status and offered date in the prospectives table.

Issues and Solutions Experienced During Development

Developing a robust database involves navigating various complexities, from comprehending the intricacies of the data to structuring it into coherent tables and establishing meaningful relationships between them. Our first hurdle was the challenge of truly understanding the data at hand, which required a deep dive into its nuances and dependencies. Additionally, determining how to represent this data in a relational database best was daunting, as it involved making decisions about table structures that accurately reflect real-world entities and their attributes. Establishing the relationships between tables, particularly in scenarios involving many-to-many relationships, can be complex and prone to errors if not handled meticulously. Creating linking tables specifically designed to manage many-to-many relationships offered clarity and efficiency, ensuring that data relationships are accurately represented and easily navigable. Our team also revised the schema to adopt a simpler or more logical approach that better aligns with the data's nature. We tried to streamline the database development process by implementing these solutions, resulting in a more robust and user-friendly system for storing and accessing data.

Lessons Learned

The journal entries document our progression in creating a database for the IAA program at UMD. Lessons learned include the importance of maintaining project scope, as advised by Professor Duffy to focus on a specific program rather than the entire university to avoid overcomplication and confidentiality issues by using realistic sample data instead of real data. We identified key questions the database should answer which guided the inclusion and exclusion of information. The process of compiling sample data highlighted the need for attention to detail, ensuring data accuracy. Feedback from Professor Diker prompted further refinement, suggesting consolidation of tables and the addition of linking tables for improved organization and efficiency. These lessons underscore the significance of thorough planning, iterative refinement, and responsiveness to feedback in our database development project.

Journal Entries for Reference

- ***2/15:*** *Our team has decided to create a database for a program (IAA) at UMD that tracks the students as they begin as prospective students, through matriculation and graduation. In discussing the idea with Professor Duffy, she recommended that we keep the idea within the confines of a specific program as opposed to the whole university to avoid overcomplication and use realistic sample data as opposed to real data to avoid confidentiality issues.*
- ***2/24:*** *We began writing the project proposal and talking through the scope to determine what questions we wanted the database to answer which in turn determined what*

information did and did not need to be included. For example, the IAA does not sponsor education visas so by default all students either have to be US citizens or have an existing visa so citizenship information is unavailable in our database.

- ***3/16:*** *Review project proposal feedback; update our tables to include primary/foreign key indicators and include references.*
- ***4/04:*** *Draft ERD and receive feedback from Srikanth about the overall structure and data types. Based on the draft ERD, we could begin utilizing Mockaroo (2024) to compile realistic sample data. Compiling the sample data was surprisingly tedious; for example, we needed to ensure that if a student was born in 1998, their college graduation date wasn't in 2000 when they were just two years old.*
- ***05/02:*** *With our ERD in its final form, we shared it with the class and received valuable feedback from Professor Diker that it may be beneficial to condense our prospective, current, and alumni tables into one "people" table and have an "id" that could correspond with the other tables. Additionally, he recommended that we add a couple of linking tables.*
- ***05/04:*** *As we built the MySQL database, we made several edits. Two examples are the inclusion of dropout status and the reason for dropout. In addition, if a student ID is shared across prospective, current, and alumni statuses, they are mutually excluded.*
- ***05/06:*** *Executing CRUD operations was based on realistic use cases. At times, our queries were too simple for the scope of the project so we had to be creative in crafting more complex queries.*

Potential Future Work

If we had more time, we would have liked to expand the functionality and features of the IAA student database that we created. However, we recognize that there would be some limitations to utilizing MySQL. This semester we briefly discussed the availability of other options to build databases including NoSQL and document databases. Both of these are useful for storing unstructured or semi-structured data, such as student essays, research papers, or course materials, MongoDB or Couchbase are examples of document databases. MySQL may face scalability challenges in handling large datasets or high concurrency levels, potentially leading to performance issues or increased hardware costs. NoSQL databases, particularly those with distributed and horizontally scalable architectures, are better suited for handling massive amounts of data. Lastly, MySQL's relational model may not easily support complex relationships between different entities within the student ecosystem, such as clubs, courses, or advisors, whereas NoSQL databases can handle these relationships more flexibly.

References

Kleppmann, Martin. (2017) Designing Data Intensive Applications. ISBN: 978-1449373320. O'Reilly Media.

Mockaroo, LLC. (2024). Random Data Generator. Mockaroo: https://www.mockaroo.com/

Murach, Joel. (2019). Murach's MySQL (3rd Ed). ISBN: 978-1943872367. Mike Murach & Associates.