2021

Development Team Progress Report

ATENA VAHEDIAN (TEAM LEAD)
ADEYEMI OWOSENI
ASHLEY HUSCROFT
LEYSAN GILFANOVA
MAHMOUDREZA KARIMKHANINEJAD
MOHAMMAD MOSIHUZZAMAN

Summary

TechSkills Program provided an opportunity to work on an industry project called "WeVote". The project was kicked off on Oct 14, 2020, when the first meeting with the client was held.

On Oct 20, 2020, learners were placed in various teams, taking roles of the Data Cleaning, Development, and Quality Assurance. Team Members mentioned above are part of the Development Team for this project. From that point on, client meetings were held every week on Thursdays, to discuss project scope, updates, and challenges.

The Development team started working on the project on Oct 20, 2020, and has had regular weekly meetings to discuss updates and future plans. Documents that are created by the development team include Entity Relationship Diagram (ERD), Table Diagram, DB_Atributes, SQL Coding Standard, and Database Security and Backup Plan.

ERD

Creating an ERD was one of the first tasks that the Development team worked on. To create the first draft, the Development team researched on the internet to find proper data sources that could potentially be added to the 2017 election data, referenced by the client. Sources for community demographic information were discovered and shared with the client and suggested to the Data Cleaning team to use as a data source. Therefore, a primary understanding of the available data helped the team to create a preliminary draft of ERD. However, table normalization was not considered in that version. In the coming weeks, ERD was modified to have tables in their normal form. Also, a few meetings were held with the client to review and decide on what demographic information needed to be included in the database. ERD was first approved by the client on Nov 26, 2019, before starting the coding phase.

However, over the process of creating the database and running various test queries by the Quality Assurance team, a few changes were applied to the diagram structure. These changes were made to:

1. Clarify and correct the relationship between normalized and join tables.
2. Ensure correct query results based on the client's expectations.

The flow of this ERD is as following:

Starting from CandidateName table, a candidate with a specific candidate type and office type has a certain number of votes at various voting stations. Each voting station has a name, type, and information such as voter turnout and number of enumerated electors. Information about voting stations is joined in a table which is connected to the candidates' join table. Voting stations are then connected to communities with certain demographic data such as Age, Gender, Income, …. Each community relates to a ward in which certain candidates are elected. PoliticalScale table is the only stand-alone table in this diagram. This is because the assignment of candidates to these scales is supposed to be done by the dashboard user.

Coding Standard

The Coding Standard document was prepared by the Development team in the planning stage of the project. This document provides a guideline to the team members on best practices for coding of the database. This

especially useful when having multiple people working on the same database to keep the code structure consistent. The first version was finalized on Nov 11, 2019. However, going through the coding and database development phase, a few changes to the naming conventions had to be made. The naming conventions were updated to ensure that all the objects created in the database follow a certain standard that can distinguish them from the other objects.

Table Diagram

To make the development process easier, a Table Diagram was created based on the latest ERD, and Primary Key/Foreign Keys were set for every table in the database. Over the coding and quality assurance process, changes were made to the diagram to represent the correct relationship between the tables and assign proper Primary Keys and Foreign Keys.

It was discovered that some communities relate to more than one voting station and some voting stations represent more than one community. To accommodate this many-to-many relationship in the database, two Join tables were designed for:

1. Joining normalized tables
2. Including certain Foreign Keys in the tables, where having that Foreign Key in the normalized table would result in ID duplication.

For further clarification, each table is discussed in detail here.

1. VotingStationType Table

In the original dataset, various types of voting stations were included. However, not all of those could be related to certain communities. For example, advance and travel types of votes could not be pinned to any community. Therefore, it was decided by the client to include only Regular and Special voting results in the database, so that they could be related to communities and related demographic information. As a result, changes were made to the ERD and Table Diagram to account for the latest request from the client.

To simplify the table normalization process in the database, numeric IDs were assigned to each voting station type. These generated IDs were used as Primary Key in this table and Foreign Key in other tables.

2. VotingStationName Table

In the original dataset, each voting station has a name associated with an ID. In this table, a list of all the available voting stations with their corresponding IDs are presented. To establish a relationship between VotingStationType and VotingStationName table, VotingStationTypeIDs was used as Foreign key in the latter table.

3. RegularSpecialVotingStation_Result Table

This table joins information from three tables, VotingStationType, VotingStationName, CommunityInformation. Using this table, communities can be related to Voting stations. At the early stages of the coding, ward IDs were also used as a foreign key in this table. But the results of the test queries revealed that using ward IDs to relate communities and voting stations would generate duplicate records. Therefore, it was decided to use only community IDs as the Foreign Key in this table.

Also, it was suggested by the Quality Assurance team to only present the numeric data in the join tables. This would serve the purpose of creating a many-to-many relationship while maintaining the integrity of the database. Therefore, only VotingStationIDs, VotingStationTypeIDs and CommunityIDs are shown in this table. In the database, an identity column was generated for this table to generate a Primary Key.

4. VoterInformation Table

It was initially planned to include both EnumeratedElectorsNumbers and EnumeratedElectorsPercents in the VoterInformation table. It was then decided to eliminate the EnumeratedElectorsPercents column because of the mismatch between total population data and enumerated voter data. The total population is derived on a community boundary basis whereas the enumerated voter data is calculated on a voting station boundary basis. This was especially problematic in cases where voting stations were split between wards. Also, voter turnout values for four senior's homes were nullified, because they had over 100% turnout (possible erroneous enumerated voters count in the source data).

Also, it was planned to have the CommunityIDs as Foreign Key in VoterInformation table. However, due to the many-to-many relationship between VotingStationIDs and CommunityIDs, doing so would have resulted in having duplicated values in the two main columns of the table (EnumeratedElectorsPercents and VoterTurnout). Therefore, it was decided to use only VotingStationIDs as Foreign Key in this table. In the database, an identity column was generated for this table to generate a Primary Key.

5. OfficeType Table

This table lists the different types of offices that candidates run for in the Calgary municipal election. To simplify the table normalization process in the database, numeric IDs were assigned to each voting station type. These generated IDs were used as the Primary Key in this table and Foreign Key in other tables.

6. CandidateType Table

Some of the candidates may be running for the same office again or they might be the only candidates in that ward. That is why knowing if a candidate is incumbent or acclaimed is important. It was planned to have this table in a format where this status is defined by using a T/F value in Incumbent and acclaimed columns for every candidate. However, this turned out to be an inefficient way and not the best method to normalize the table. However, because the master tables were generated based on this structure before, the Development team needed to make changes to the database to accommodate the changes. As a result, this table was transformed to have two columns as candidate types and their corresponding IDs. Four candidate types were listed to cover all the scenarios:

1. A candidate is incumbent.
2. A candidate is acclaimed.
3. A candidate is neither incumbent nor acclaimed.
4. A candidate is both incumbent and acclaimed.

A corresponding ID was generated for each case to use as Foreign Key in other tables. However, the calculated IDs could not be used as Primary Keys, therefore, an identity column was generated for this table.

7. CandidateName Table

This table simply lists the candidates running for offices in the 2017 Calgary municipal election. The Primary Key is the IDs generated by the Data Cleaning team in the data preparation stage. To establish a relationship between this table and CandidateType table, CandidateTypeIDs was used as the Foreign Key.

8. Candidate_Result Table

This is another join table in the database. The table joins CandidateName and CandidateType tables and relates each candidate to a certain office and ward. Similar to the other join table, only numeric data was presented and an identity column was used as the Primary Key.

9. VoteCount Table

To include vote counts for every candidate at every voting station, a table called "VoteCount" was created. To generate a unique ID for each record, a new column called VotingStationCandidateIDs was added to the master data to combine voting station ID and candidate ID for every record. However, during database development, it was discovered that in 8 cases this combination results in the same ID (e.g. CandidateID=3, VotingStationID=1302 and VotingStationCandidateID=13023 and CandidateID=23, VotingStationID=130 and VotingStationCandidateID=13023). The primary approach that was taken by the Development team was to update this column in cases where it was repeated. It was brought up by the Quality Assurance team that, this could cause confusion. Therefore an identity column was used instead to generate unique IDs.

10. PoliticalScale Table

This table is a stand-alone table that lists 3 scales of political leaning. Candidates could be left-leaning, right-leaning, or center leaning. The intention is that every user of the dashboard would assign candidates to these categories based on their perspective.

11. NewWard Table

The ward boundaries have been updated recently in Calgary, meaning that some communities have moved to another ward since the last election was held in 2017. The client wanted to include both old and new ward IDs in the database and this introduced some challenges to the table structures. The relationship between old and new ward IDs is not one-to-one, therefore having both columns in the ward table means that neither one of the columns has the condition to become the Primary Key. This was a critical problem because ward ID is used in other tables as Foreign Key. The final decision was to use a table called NewWard to hold only NewWardIDs as Primary Key there and Foreign Key in other tables.

12. WardChange Table

At first, the Development team wanted to use both old and new ward IDs as Composite Primary Key and therefore use both as Foreign Keys in the other tables. However, having both the old and new ward IDs as Foreign Key in CommunityInformation table resulted in having repeated community IDs (Primary Key in the CommunityInformation table). To resolve this challenge, a table called WardChange was created to have both new and old ward IDs.

13. CommunityInformation Table

This table is the main link between community demographics and election results. 198 communities across Calgary have been listed in this table. The community IDs were generated by the Data Cleaning team during the data preparation phase. The community IDs for Woodbine, Woodlands, and Seton were inserted incorrectly in the master table. These IDs were corrected inside the database before normalized tables were generated. In order to connect this table to election results, NewWardIDs were used as Foreign Key in this table.

14. HomeOwnership Table

The HomeOwnership table presents the number and percentage of homeowners and tenants in each community. The Primary key is a database generated identity column and community IDs are used as Foreign Key.

15. AgeGroup Table

This table presents the number and percentage of community residents in different age groups starting from 20 years old to older than 80 years. The Primary key is a database generated identity column and community IDs are used as Foreign Key.

16. MedianIncome Table

This table presents the number and percentage of community residents that fall in different income tax brackets. The Primary key is a database generated identity column and community IDs are used as Foreign Key.

17. Citizenship Table

The Citizenship table presents the number and percentage of community residents who were Canadian citizens or noncitizens at the time of the Calgary Civic Census 2016. The Primary key is a database generated identity column and community IDs are used as Foreign Key.

18. Family Table

This table provides information about what portion of the community households have children. Similar to the other demographic tables in the database, the Primary key is a database generated identity column and community IDs are used as Foreign Key.

19. Gender Table

The number and percentage of female and male residents in each community ids presented in this table. The Primary key is a database generated identity column and community IDs are used as Foreign Key.

DB Attributes

This document was created by the Development team at the early stages of the project to list all the tables, their corresponding attributes, data type and expected range of the values. This was used as a reference for the Data Cleaning team to prepare the master files according to the data types and structure that is most suited for the database development.

The final design of the database has 19 tables including two join tables.

Security and Backup Plan

The purpose of the database security and backup plan for this project is to detail the Development teams' recommendations to safeguard the database against potential threats to its integrity. The need to create robust security and backup plan cannot be overemphasized as every database will encounter threats in both the design stage and after its creation.

Additional Database Objects

In order to improve the performance of the database, additional objects were added to improve the database performance and secure it from unwanted changes. Below is a list of these objects and their functionality:

1. Constraints

In addition to the Foreign Key and Primary Key constraints added during the database normalization process, the Development team decided to add Not Null and Check constraints to ensure data integrity in the database. For instance, a Check constraint was added to all the numeric demographic attributes to ensure nonnegative numbers are used in the tables. Also, the Not Null constraint was added to CandidateNames, CommunityNames and OfficeTypes since these are identified as crucial information to this database.

2. Views

Another type of object that was added to the database is the view object. In order to maintain the integrity of the database and keep the underlying tables safe, views were created for all the offices (Mayor, Councillor, Public School Trustee and Separate School Trustee). These views summarized all the information about the candidates running for each office, the community, ward and voting stations they were running in, and their vote counts. Two additional views were also created to summarize community demographic data in numbers and percentages.

3. Stored Procedures

To easily find and remove duplicates from each table, stored procedures were created to simplify the process. The primary decision was to have each stored procedures designed in a way that it would take a table name and give the user an option of providing one or two column names as input to perform each task. However, it was discovered that the code does not operate as expected when less than two column names are inputted. Therefore, the Development team generated a separate stored procedure for each action to take only one column name as input.

Also, it was discovered that more than 100 null rows existed in the master table. To simplify the removal of these records in the development phase and in the future, the Development team generated another stored procedure to delete all the null records from a selected table.