DBMS Project Report

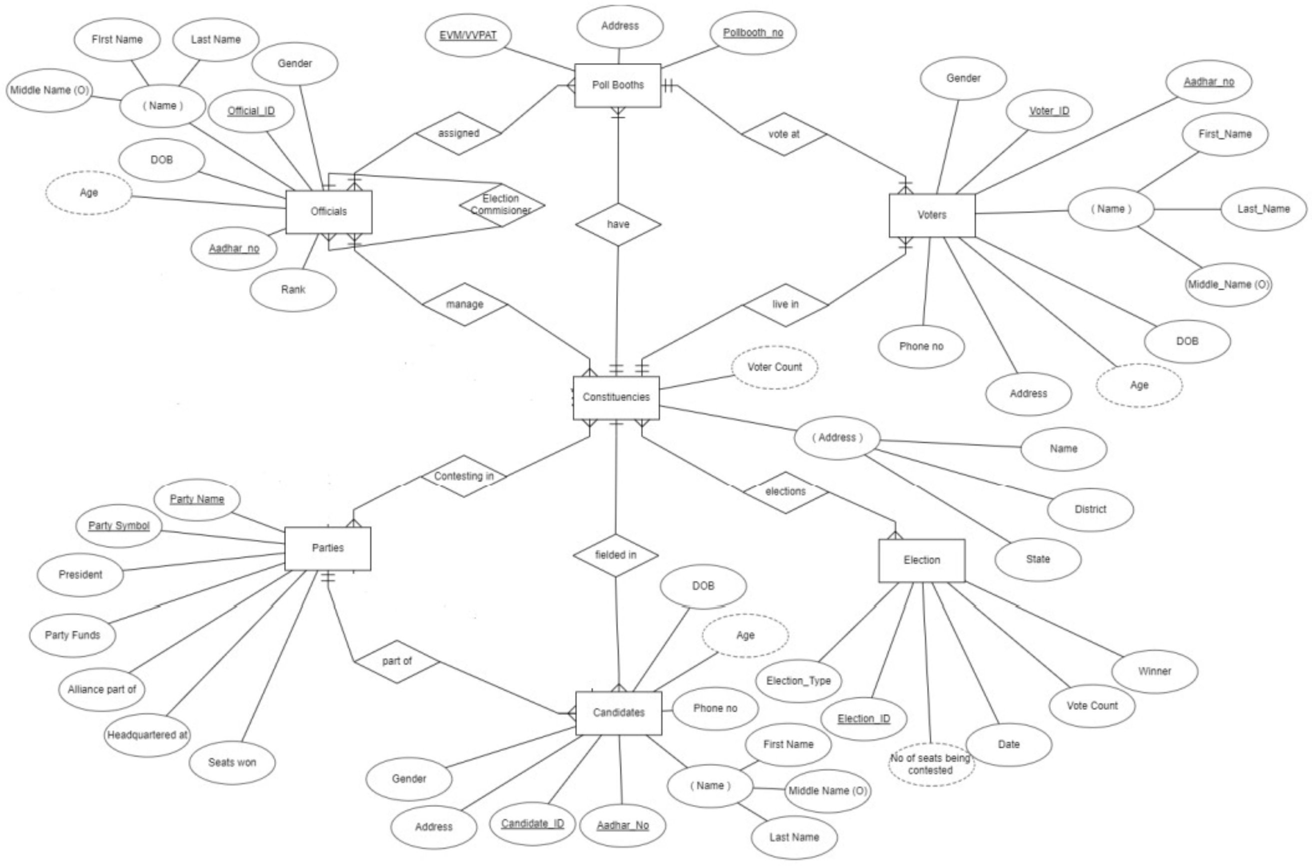
Election Management System

Team:

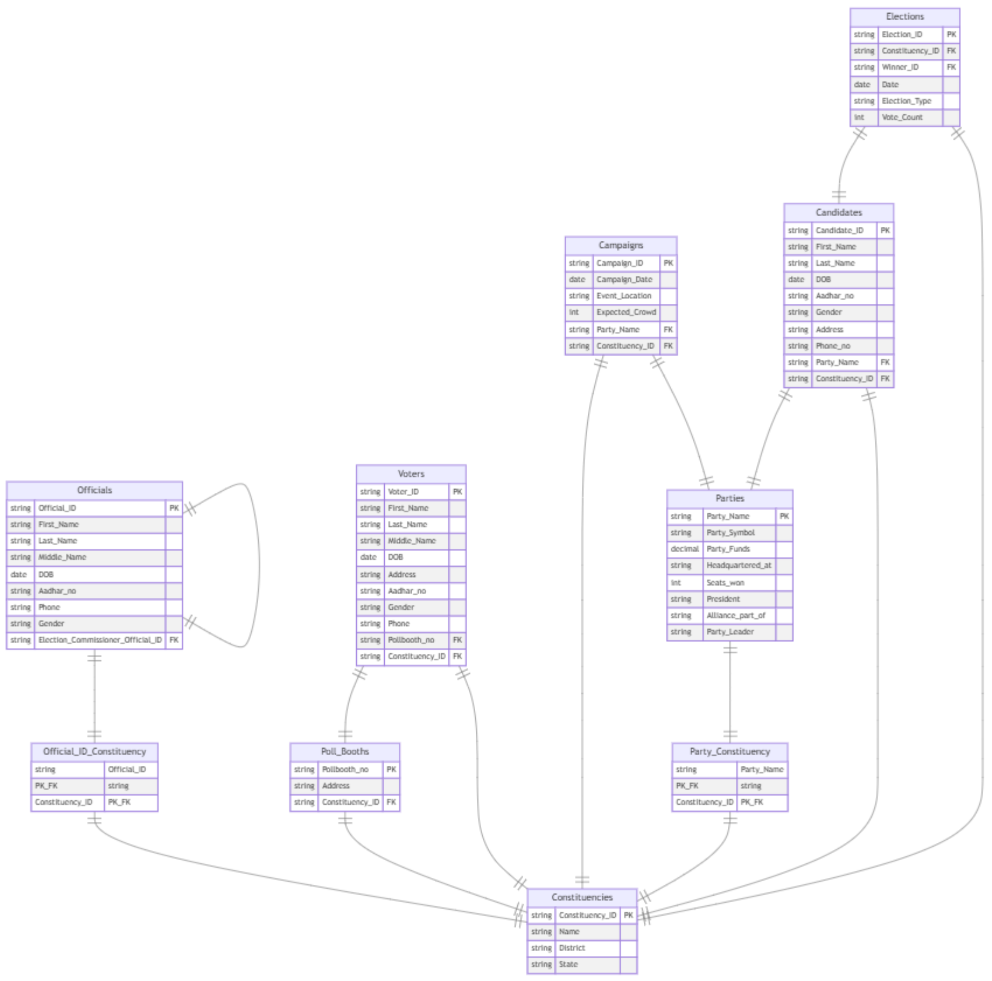
Manish P: PES1UG22CS332
Priyam R: PES1UG22CS453

The Election Management System (EMS) is designed to centralize and streamline the management of elections, covering all election-related tasks such as voter registration, candidate management, election setup, polling booth allocation, voting, and result generation. This SRS describes the full system's requirements, functionality, and constraints, ensuring transparency, security, and efficiency in the electoral process.

ER Diagram:



Relational Schema:


Elections
string Election_ID PK
string Constituency_ID FK
string Winner_ID FK
date Date
string Election_Type
int Vote_Count
Campaigns
string Campaign_ID PK
date Campaign_Date
string Event_Location
int Expected_Crowd
string Party_Name FK
string Constituency_ID FK
Candidates
string Candidate_ID PK
string First_Name
string Last_Name
date DOB
string Aadhar_no
string Gender
string Address
string Phone_no
string Party_Name FK
string Constituency_ID FK
Officials
string Official_ID PK
string First_Name
string Last_Name
string Middle_Name
date DOB
string Aadhar_no
string Phone
string Gender
string Election_Commissioner_Official_ID FK
Voters
string Voter_ID PK
string First_Name
string Last_Name
string Middle_Name
date DOB
string Address
string Aadhar_no
string Gender
string Phone
string Pollbooth_no FK
string Constituency_ID FK
Parties
string Party_Name PK
string Party_Symbol
decimal Party_Funds
string Headquartered_at
int Seats_won
string President
string Alliance_part_of
string Party_Leader
Official_ID_Constituency
string Official_ID
PK_FK string
Constituency_ID PK_FK
Poll_Booths
string Pollbooth_no PK
string Address
string Constituency_ID FK
Party_Constituency
string Party_Name
PK_FK string
Constituency_ID PK_FK
Constituencies
string Constituency_ID PK
string Name
string District
string State

List of Triggers Used

1. **voter_age**
   Trigger to calculate the age of a voter before inserting into the voter table.
2. **voter_age_on_update**
   Trigger to update the voter's age if their date of birth changes during an update.
3. **prevent_delete_constituency**
   Trigger to prevent deletion of a constituency if voters are still associated with it.
4. **prevent_delete_pollbooth**
   Trigger to prevent deletion of a poll booth if voters are still associated with it.
5. **update_seats**
   Trigger to update the number of seats in an election when a new entry is added to the election_cons table.
6. **set_official_id**
   Trigger to set the official ID before inserting into the official table.
7. **off_age**
   Trigger to calculate the age of an official before inserting into the official table.
8. **official_age_on_update**
   Trigger to update an official's age if their date of birth changes during an update.
9. **set_candidate_id**
   Trigger to set the candidate ID before inserting into the candidate table.
10. **cand_age**
    Trigger to calculate the age of a candidate before inserting into the candidate table.
11. **increment_party_member_count**

    Trigger to increase the party member count when a candidate registers
12. **add_gender**

    Trigger to auto increment the gender count

Functions Used:

1. **malecount:**
   Used to calculate number of male voters in a constituency
2. **femalecount:**
   Used to calculate number of female voters in a constituency

List of Procedures Used:

1. **getconsdets:**
   This procedure retrieves detailed information about each constituency, including the number of male and female voters and the count of polling booths.
2. **getvoterdets:**
   This procedure retrieves the details of all voters.
3. **getcanddets:**
   This procedure retrieves details about candidates running for election.
4. **getpartydets:**
   This procedure retrieves details about political parties.
5. **getofficialdets:**
   This procedure retrieves details about officials working in the election process.

List of Nested Queries Used:

1. Query to fetch constituency name

List of joint operations:

1. create procedure getconsinfo(in input_voter_id varchar(20)) begin select c.constituency_name, c.state, c.voter_count, concat(coalesce(cand.first_name,''),' ',coalesce(cand.middle_name,''),' ',coalesce(cand.last_name,'')) as candidate_name, cand.age,p.party_name,p.party_symbol from constituency c left join candidate cand on cand.cons_fight = c.constituency_name left join party p on cand.party_rep = p.party_name where cand.cons_fight in (select constituency_name from voter where voter_id = input_voter_id);
   This is a left joint used inside a procedure. The left joint clause is used to combine rows from the constituency table (c) and the candidate table (cand):

2. create procedure getconsdets() begin select c.constituency_name, malecount(c.constituency_name) as male_count, femalecount(c.constituency_name) AS female_count, count(distinct pb.poll_booth_id) as poll_booth_count from constituency c left join poll_booth pb on c.constituency_id = pb.constituency_id group by c.constituency_name;