ELECTION / VOTING MANAGEMENT SYSTEM

1. Introduction

The Online Election Management System is a database-driven project designed to manage and conduct elections efficiently. The system stores information about voters, candidates, political parties, elections, and votes. It ensures data integrity using primary keys, foreign keys, unique constraints, and transaction management. The system prevents double voting and maintains accurate election results using SQL queries and views.

2. Objectives

- To design a relational database with proper normalization.
- To implement primary key and foreign key constraints.
- To prevent duplicate voting using unique constraints.
- To demonstrate JOIN operations (INNER, LEFT, RIGHT).
- To perform aggregate functions (COUNT, AVG).
- To implement subqueries and views.
- To demonstrate transaction management using COMMIT and ROLLBACK

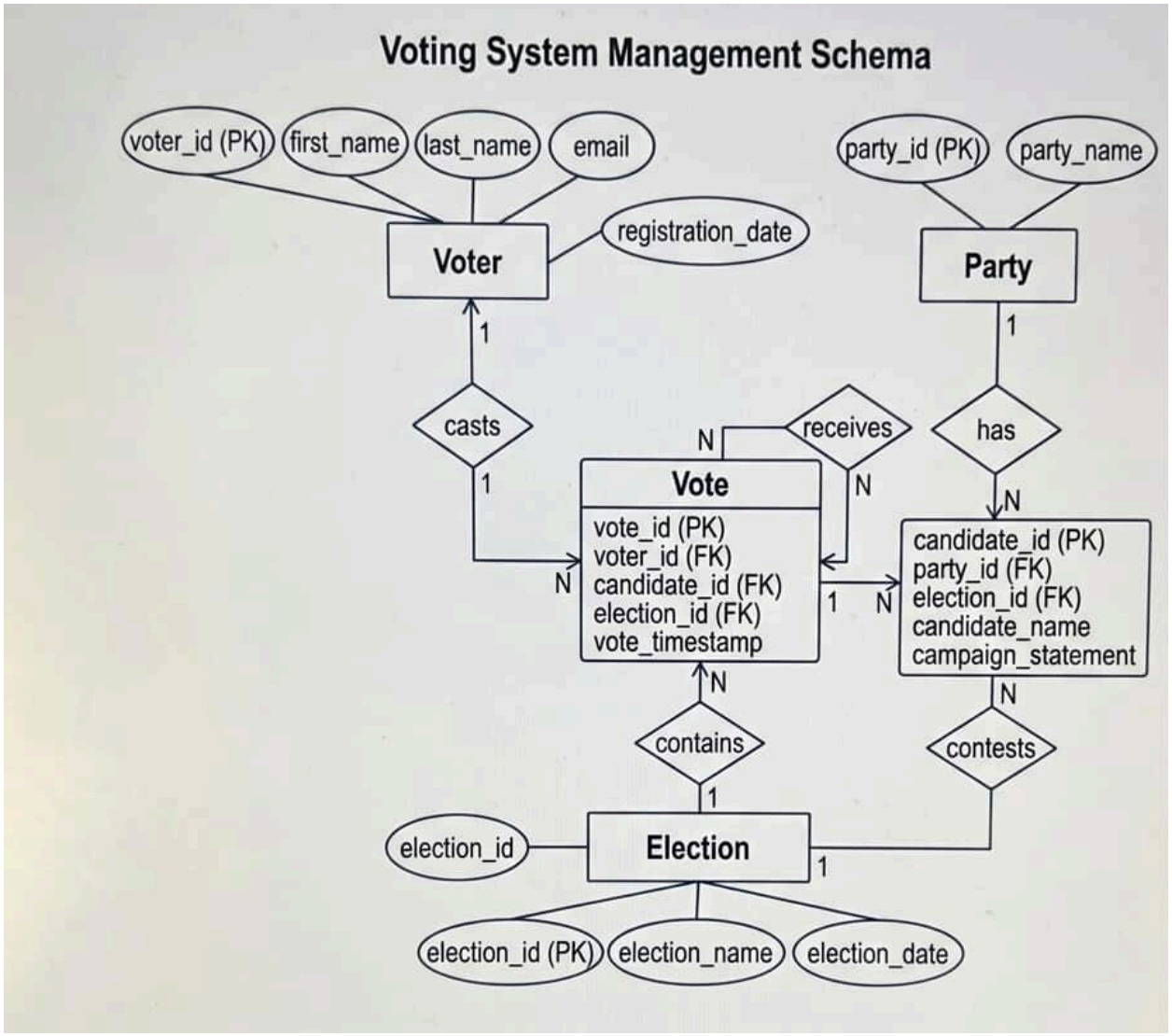
3. ENTITY LIST & DESCRIPTION

This project models a complete Election/Voting Management System with 5 entities:

#	Entity	Description	Key Attributes
1	Voters	Registered citizens/students who can cast votes	VoterID (PK), Email (UNIQUE), HasVoted (flag)
2	Candidates	Individuals contesting in elections	CandidateID (PK), PartyID (FK), Constituency
3	Elections	Election events with schedules, dates, and status	ElectionID (PK), Status, StartTime, EndTime

4	Votes	Bridge/transaction table — records every vote cast	VoteID (PK), VoterID+CandidateID+ElectionID (FKs)
5	Parties	Political parties that candidates represent	PartyID (PK), PartyName, Symbol, Leader

4. ER DIAGRAM



5. Source Code

```
CREATE DATABASE ElectionDB;

USE ElectionDB;

CREATE TABLE Parties (

    PartyID INT AUTO_INCREMENT PRIMARY KEY,

    PartyName VARCHAR(100) NOT NULL UNIQUE,

    Symbol VARCHAR(50)  NOT NULL,

    Leader VARCHAR(100)

);


CREATE TABLE Voters (

    VoterID INT AUTO_INCREMENT PRIMARY KEY,

    Name VARCHAR(100) NOT NULL,

    Email VARCHAR(100) NOT NULL UNIQUE,

    Phone VARCHAR(15),

    Age INT CHECK (Age >= 18),

    Constituency VARCHAR(50)  NOT NULL,

    HasVoted TINYINT(1)   DEFAULT 0
```

```
);


CREATE TABLE Elections (
    ElectionID INT AUTO_INCREMENT PRIMARY KEY,
    Title VARCHAR(200) NOT NULL,
    ElectionDate DATE NOT NULL,
    StartTime DATETIME NOT NULL,
    EndTime DATETIME NOT NULL,
    Status VARCHAR(20)  DEFAULT 'Upcoming'
        CHECK (Status IN ('Upcoming','Ongoing','Completed'))
);


CREATE TABLE Candidates (
    CandidateID  INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    PartyID INT NOT NULL,
    Age INT NOT NULL,
    Constituency VARCHAR(50)  NOT NULL,
    Bio TEXT,
```

```
    FOREIGN KEY (PartyID) REFERENCES Parties(PartyID)
);

CREATE TABLE Votes (
    VoteID INT AUTO_INCREMENT PRIMARY KEY,
    VoterID INT NOT NULL,
    CandidateID INT NOT NULL,
    ElectionID  INT NOT NULL,
    VoteTime DATETIME DEFAULT NOW(),
    IPAddress VARCHAR(45),
    FOREIGN KEY (VoterID) REFERENCES Voters(VoterID),
    FOREIGN KEY (CandidateID) REFERENCES Candidates(CandidateID),
    FOREIGN KEY (ElectionID)  REFERENCES Elections(ElectionID),
    UNIQUE (VoterID, ElectionID)
);

INSERT INTO Parties (PartyName, Symbol, Leader) VALUES
    ('Progressive Party','Sun', 'Aarav Sharma'),
    ('National Alliance', 'Star', 'Sita Rana'),
```

```
    ('Green Future','Leaf','Bikram Thapa'),
    ('Unity Front','Hand','Priya Koirala'),
    ('Independent', 'Scale', NULL);

INSERT INTO Voters (Name, Email, Phone, Age, Constituency) VALUES
    ('Ram Bahadur', 'ram@email.com','9841000001', 25, 'Block A'),
    ('Sita Devi', 'sita@email.com','9841000002', 30, 'Block B'),
    ('Hari Prasad', 'hari@email.com','9841000003', 22, 'Block A'),
    ('Gita Kumari','gita@email.com','9841000004', 28, 'Block C'),
    ('Shyam Lal','shyam@email.com','9841000005', 35, 'Block B'),
    ('Kamala Thapa','kamala@email.com','9841000006', 19, 'Block A'),
    ('Raju KC','raju@email.com','9841000007', 45, 'Block C'),
    ('Bimala Rai','bimala@email.com','9841000008', 32, 'Block B');

INSERT INTO Elections (Title, ElectionDate, StartTime, EndTime, Status) VALUES
    ('Student Council President 2026','2026-03-10','2026-03-10 08:00:00','2026-03-10
17:00:00','Upcoming'),
    ('Class Representative Election', '2026-03-15','2026-03-15 09:00:00','2026-03-15
14:00:00','Upcoming'),
```

('Department Head Vote 2025','2025-11-05','2025-11-05 08:00:00','2025-11-05 16:00:00','Completed'),

('Sports Captain Election', '2026-01-20','2026-01-20 10:00:00','2026-01-20 13:00:00','Completed');

INSERT INTO Candidates (Name, PartyID, Age, Constituency, Bio) VALUES

('Anita Shrestha',1,26,'Block A','Experienced student leader'),

('Bikash Gurung',2, 29, 'Block A', 'Sports and culture advocate'),

('Chandra Magar',3, 31, 'Block B', 'Environment and sustainability'),

('Deepa Tamang',1, 24, 'Block B', 'Academic excellence champion'),

('Eshan Karki',4, 28, 'Block C', 'Youth empowerment initiative'),

('Fiona Rai',5, 33, 'Block C', 'Independent candidate'),

('Gopal Pant',2, 27, 'Block A', 'Infrastructure development'),

('Hira BK', 3, 22, 'Block B', 'First-time candidate');

INSERT INTO Votes (VoterID, CandidateID, ElectionID, IPAddress) VALUES

(1, 1, 3, '192.168.1.10'),

(2, 3, 3, '192.168.1.11'),

(3, 2, 3, '192.168.1.12'),

(4, 5, 3, '192.168.1.13'),

```
    (5, 1, 3, '192.168.1.14'),
    (6, 4, 4, '192.168.1.15'),
    (7, 6, 4, '192.168.1.16'),
    (8, 5, 4, '192.168.1.17');

UPDATE Voters SET HasVoted = 1 WHERE VoterID IN (1,2,3,4,5,6,7,8);

SELECT v.Name AS Voter, c.Name AS Candidate,
       p.PartyName, e.Title AS Election, vt.VoteTime
FROM Votes vt
INNER JOIN Voters v ON vt.VoterID = v.VoterID
INNER JOIN Candidates c ON vt.CandidateID = c.CandidateID
INNER JOIN Elections e ON vt.ElectionID  = e.ElectionID
INNER JOIN Parties p ON c.PartyID = p.PartyID;

SELECT v.VoterID, v.Name, v.Constituency,
       COALESCE(c.Name, 'Did Not Vote') AS CandidateVotedFor
FROM Voters v
LEFT JOIN Votes      vt ON v.VoterID      = vt.VoterID
```

```
LEFT JOIN Candidates c  ON vt.CandidateID = c.CandidateID;


SELECT c.Name AS Candidate, p.PartyName,
    COUNT(vt.VoteID) AS TotalVotes
FROM Votes vt
RIGHT JOIN Candidates c ON vt.CandidateID = c.CandidateID
RIGHT JOIN Parties    p ON c.PartyID      = p.PartyID
GROUP BY c.CandidateID, c.Name, p.PartyName;


SELECT c.Name AS Candidate, p.PartyName,
    COUNT(vt.VoteID) AS TotalVotes
FROM Candidates c
LEFT JOIN Votes   vt ON c.CandidateID = vt.CandidateID
LEFT JOIN Parties p  ON c.PartyID     = p.PartyID
GROUP BY c.CandidateID, c.Name, p.PartyName
ORDER BY TotalVotes DESC;


SELECT p.PartyName,
    COUNT(vt.VoteID) AS TotalVotes,
```

```
    ROUND(AVG(COUNT(vt.VoteID)) OVER (), 2) AS AvgVotesAllParties
FROM Parties p
LEFT JOIN Candidates c  ON p.PartyID     = c.PartyID
LEFT JOIN Votes      vt ON c.CandidateID = vt.CandidateID
GROUP BY p.PartyID, p.PartyName;


SELECT c.Name AS Candidate, p.PartyName,
    COUNT(vt.VoteID) AS TotalVotes
FROM Candidates c
JOIN Votes   vt ON c.CandidateID = vt.CandidateID
JOIN Parties p  ON c.PartyID     = p.PartyID
GROUP BY c.CandidateID, c.Name, p.PartyName
HAVING COUNT(vt.VoteID) > (
  SELECT AVG(VoteCount) FROM (
    SELECT COUNT(VoteID) AS VoteCount
    FROM Votes
    GROUP BY CandidateID
  ) AS AvgTable
);
```

```
CREATE VIEW ElectionResults AS
    SELECT e.Title AS Election,
           c.Name AS Candidate,
           p.PartyName,
           c.Constituency,
           COUNT(vt.VoteID) AS TotalVotes
    FROM Elections e
    LEFT JOIN Votes      vt ON e.ElectionID  = vt.ElectionID
    LEFT JOIN Candidates c  ON vt.CandidateID = c.CandidateID
    LEFT JOIN Parties    p  ON c.PartyID      = p.PartyID
    GROUP BY e.ElectionID, e.Title, c.CandidateID,
             c.Name, p.PartyName, c.Constituency;

SELECT * FROM ElectionResults ORDER BY Election, TotalVotes DESC;

START TRANSACTION;
INSERT INTO Votes (VoterID, CandidateID, ElectionID, IPAddress)
VALUES (3, 2, 1, '192.168.1.20');
```

```
UPDATE Voters SET HasVoted = 1 WHERE VoterID = 3;

COMMIT;

-- If any error occurs, rollback everything:

-- ROLLBACK;

SELECT * FROM Votes     WHERE VoterID = 3;

SELECT HasVoted FROM Voters WHERE VoterID = 3;
```

6. Conclusion

The Online Election Management System successfully demonstrates the core concepts of DBMS including database design, normalization, primary and foreign keys, SQL implementation, joins, aggregate functions, subqueries, views, and transaction control. The system ensures secure and efficient vote management while maintaining data integrity and preventing duplicate voting. This project fulfills all required DBMS practical requirements.