VoteX – Campaign Compass

Set Commands

Sanjay Ponnambalam N

RA2211026010136

Y1 – B. Tech CSE – AI & ML

Kirushakkarasu KT

RA2211026010134

Y1 – B. Tech CSE – AI & ML

1.SET Union - Retrieve the names of all voters and donors.

```
SELECT Name FROM Voter
UNION
SELECT Name FROM Donor;
```

2. Union All Retrieve the names of all voters and donors, including duplicates.

```
SELECT Name FROM Voter
UNION ALL
SELECT Name FROM Donor;
```

3.Intersect - Find the names of voters who are also donors.

```
SELECT Name FROM Voter
INTERSECT
SELECT Name FROM Donor;
```

4.Minus - Find the names of voters who are not donors.

```
SELECT Name FROM Voter
```

```
MINUS
SELECT Name FROM Donor;
```

5. Check if there are any donors who are also voters.

```
SELECT CandidateID, Name, PartyAffiliation
FROM Candidate outer
WHERE EXISTS
(SELECT 1
FROM Candidate inner
WHERE inner.CandidateID = outer.CandidateID
AND inner.Name <> outer.Name
AND inner.PartyAffiliation = outer.PartyAffiliation
AND inner.CampaignPlatform = outer.CampaignPlatform
AND inner.ContactDetails = outer.ContactDetails);
```

6. Not Exists --- Select candidate attributes where there does not exist another candidate with the same PartyAffiliation, CampaignPlatform, and ContactDetails but a different CandidateID

```
SELECT CandidateID, Name, PartyAffiliation, CampaignPlatform, ContactDetails
FROM Candidate outer
-- Check if there does not exist another candidate satisfying the specified conditions
WHERE NOT EXISTS
  (
  -- Subquery to find candidates with the same PartyAffiliation, CampaignPlatform, and ContactDetails but a different CandidateID
  SELECT 1
  FROM Candidate inner
  WHERE inner.PartyAffiliation = outer.PartyAffiliation
  AND inner.CampaignPlatform = outer.CampaignPlatform
  AND inner.ContactDetails = outer.ContactDetails
  AND inner.CandidateID <> outer.CandidateID
  );
```

6. In - Retrieve the names of donors who donated more than 5000 rupees.

SELECT Name FROM Donor WHERE DonationAmount > 5000;

7. Not In - Retrieve the names of donors who donated less than or equal to 5000 rupees.

SELECT Name FROM Donor WHERE DonationAmount <= 5000;

/////

8.Except - Find the names of donors who are not also volunteers.

SELECT Name FROM Donor

EXCEPT

SELECT Name FROM Volunteer;

9.Distinct - Retrieve unique names of donors.

SELECT DISTINCT Name FROM Donor;

10. Retrieve unique regions represented in the Constituency table.

SELECT DISTINCT Region

FROM Constituency;

11. Find Common Volunteers Between Different Campaigns:This query will identify volunteers who are assigned to multiple campaigns by finding the intersection of the sets of volunteers for different campaigns.

SELECT Name

FROM Volunteer

WHERE AssignedCampaignID IN (SELECT CampaignID FROM Campaign WHERE Description = 'Campaign 1')

INTERSECT

SELECT Name

FROM Volunteer

WHERE AssignedCampaignID IN (SELECT CampaignID FROM Campaign WHERE Description = 'Campaign 2');

12. Retrieve Names of Donors Who Are Also Volunteers: This query will retrieve the names of donors who are also volunteers by finding the intersection of the sets of names from the Donor and Volunteer tables.

SELECT Name

FROM Donor

INTERSECT

SELECT Name

FROM Volunteer;

13.Union of voter names and donor names

SELECT Name FROM Voter

UNION

SELECT Name FROM Donor;

14. Intersection of candidate names and volunteer names

SELECT Name FROM Candidate

INTERSECT

SELECT Name FROM Voter;

15.Union All of political party names and event names

SELECT Name FROM PoliticalParty

UNION ALL

SELECT Name FROM Event;

16.Except of fundraiser names and voter names

```
SELECT Name FROM Fundraiser
EXCEPT
SELECT Name FROM Voter;
```

17. Union of constituency names and polling station names

```
SELECT Name FROM Constituency
UNION
SELECT Name FROM PollingStation;
```

18.Intersection of event locations and polling station locations

```
SELECT Location FROM Event
INTERSECT
SELECT Location FROM PollingStation;
```

19. Union All of candidate names and volunteer names

```
SELECT Name FROM Candidate
UNION ALL
SELECT Name FROM Volunteer;
```

VoteX – Campaign Compass

Join Commands

Sanjay Ponnambalam N

RA2211026010136

Y1 – B. Tech CSE – AI & ML

Kirushakkarasu KT

RA2211026010134

Y1 – B. Tech CSE – AI & ML

1.Join Campaign with Candidate to get Campaign details along with the Candidate's name:

```
SELECT Campaign.*, Candidate.Name AS CandidateName
FROM Campaign
JOIN Candidate ON Campaign.CandidateID = Candidate.CandidateID;
```

2.Join Fundraiser with Campaign to get Fundraiser details along with the Campaign's start and end dates:

```
SELECT Fundraiser.*, Campaign.StartDate, Campaign.EndDate
FROM Fundraiser
JOIN Campaign ON Fundraiser.CampaignID = Campaign.CampaignID;
```

3.Join Event with Campaign to get Event details along with the Campaign's budget:

```
SELECT Event.*, Campaign.Budget
FROM Event
JOIN Campaign ON Event.CampaignID = Campaign.CampaignID;
```

4.Join Candidate with Constituency to get Candidate details along with the Constituency's region:

```
SELECT Candidate.*, Constituency.Region
FROM Candidate
```

```
JOIN Constituency ON Candidate.CandidateID = Constituency.ConstituencyID;
```

5.Join Candidate with Political Party to get Candidate details along with Party's ideology:

```
SELECT Candidate.*, PoliticalParty.Ideology
FROM Candidate
JOIN PoliticalParty ON Candidate.CandidateID = PoliticalParty.PartyID;
```

6.Join Voter with Constituency to get Voter details along with Constituency's name:

```
SELECT Voter.*, Constituency.Name AS ConstituencyName
FROM Voter
JOIN Constituency ON Voter.Address = Constituency.ConstituencyID;
```

7.Join Fundraiser with Political Party to get Fundraiser details along with Party's contact details:

```
SELECT Fundraiser.*, PoliticalParty.ContactDetails
FROM Fundraiser
JOIN Campaign ON Fundraiser.CampaignID = Campaign.CampaignID
JOIN Candidate ON Campaign.CandidateID = Candidate.CandidateID
JOIN PoliticalParty ON Candidate.PartyAffiliation = PoliticalParty.PartyID;
```

8.Join Fundraiser with Event to get Fundraiser details along with Event location and date:

```
SELECT Fundraiser.*, Event.Location, Event.EventDate
FROM Fundraiser
JOIN Event ON Fundraiser.CampaignID = Event.CampaignID;
```

9.Join Event with Political Party to get Event details along with Party's name and ideology:

```
SELECT Event.*, PoliticalParty.Name AS PartyName, PoliticalParty.Ideology
FROM Event
JOIN Campaign ON Event.CampaignID = Campaign.CampaignID
JOIN Candidate ON Campaign.CandidateID = Candidate.CandidateID
```

JOIN PoliticalParty ON Candidate.PartyAffiliation = PoliticalParty.PartyID;

10.Join Voter with Donor to get Voter details along with Donor's name and donation amount:

SELECT Voter.*, Donor.Name AS DonorName, Donor.DonationAmount

FROM Voter

JOIN Donor ON Voter.VoterID = Donor.DonorID;

11.Left Join: Join Event with Fundraiser to get Event details along with Fundraiser's name and funds raised:

SELECT Event.*, Fundraiser.Name AS FundraiserName, Fundraiser.FundsRaised

FROM Event

LEFT JOIN Fundraiser ON Event.CampaignID = Fundraiser.CampaignID;

12.Outer Join: Join Event with Fundraiser to get Event details along with Fundraiser's name and funds raised:

SELECT Event.*, Fundraiser.Name AS FundraiserName, Fundraiser.FundsRaised

FROM Event

LEFT JOIN Fundraiser ON Event.CampaignID = Fundraiser.CampaignID;

13.Right Outer Join: Join Event with Fundraiser to get Event details along with Fundraiser's name and funds raised:

SELECT Event.*, Fundraiser.Name AS FundraiserName, Fundraiser.FundsRaised

FROM Event

RIGHT JOIN Fundraiser ON Event.CampaignID = Fundraiser.CampaignID;

14.Right Outer Join: Join Event with Fundraiser to get Event details along with Fundraiser's name and funds raised:

```
SELECT Event.*, Fundraiser.Name AS FundraiserName, Fundraiser.FundsRaised

FROM Event

RIGHT JOIN Fundraiser ON Event.CampaignID = Fundraiser.CampaignID;
```

15.Retrieve Polling Station Details Along with the Name of the Constituency

```
SELECT PS.Name AS PollingStation_Name, PS.Location, PS.OpeningTime, PS.ClosingTime, C.Name AS Constituency_Name

FROM PollingStation PS

INNER JOIN Constituency C ON C.ConstituencyID = PS.ConstituencyID;
```

VoteX – Campaign Compass

View Commands

Sanjay Ponnambalam N

RA2211026010136

Y1 – B. Tech CSE – AI & ML

Kirushakkarasu KT

RA2211026010134

Y1 – B. Tech CSE – AI & ML

---

1. Create a view to retrieve polling station details with opening and closing times

```
CREATE VIEW PollingStationDetailsView AS
SELECT PS.Name AS PollingStation_Name, PS.Location, PS.OpeningTime, PS.ClosingTime
FROM PollingStation PS;
```

2. Create a view to retrieve campaigns along with their start and end dates

```
CREATE VIEW CampaignDatesView AS
SELECT Description AS Campaign_Description, StartDate, EndDate
FROM Campaign;
```

3. Create a view to list constituencies and their regions

```
CREATE VIEW ConstituencyRegionView AS
SELECT Name AS Constituency_Name, Region
FROM Constituency;
```

4. Create a view to list political parties along with their leaders

```
CREATE VIEW PoliticalPartyLeadersView AS
SELECT Name AS Party_Name, Leader
FROM PoliticalParty;
```

5.Create a view to retrieve candidate names and their party affiliations

```
CREATE VIEW CandidatePartyView AS
SELECT Name AS Candidate_Name, PartyAffiliation
FROM Candidate;
```

6.Create a view to list events along with their locations

```
CREATE VIEW EventLocationView AS
SELECT Name AS Event_Name, Location
FROM Event;
```

7.Create a view to retrieve events along with their corresponding campaign descriptions

```
CREATE VIEW EventCampaignDescriptionView AS
SELECT E.Name AS Event_Name, C.Description AS Campaign_Description
FROM Event E
INNER JOIN Campaign C ON E.CampaignID = C.CampaignID;
```

8. Create a view to list polling stations and their opening/closing times

```
CREATE VIEW PollingStationTimesView AS
SELECT Name AS PollingStation_Name, OpeningTime, ClosingTime
FROM PollingStation;
```

9.Create a view to retrieve donors along with their contact details

```
CREATE VIEW DonorContactDetailsView AS
SELECT Name AS Donor_Name, ContactDetails
FROM Donor;
```

10. This view provides details of all candidates.

```
CREATE VIEW CandidateDetails AS
SELECT CandidateID, Name, PartyAffiliation, CampaignPlatform, ContactDetails
FROM Candidate;
```

11. This view displays voter names and addresses.

```
CREATE VIEW VoterAddresses AS
SELECT VoterID, Name, Street, City, State, ZipCode, ContactDetails
FROM Voter;
```

12.This view shows information about all campaigns.

```
CREATE VIEW CampaignInfo AS
SELECT CampaignID, StartDate, EndDate, Budget, Description
FROM Campaign;
```

13. This view provides details of all donors.

```
CREATE VIEW DonorDetails AS
SELECT DonorID, Name, ContactDetails, DonationAmount, DonationDate
FROM Donor;
```

14. This view lists contact details of volunteers.

```
CREATE VIEW VolunteerContacts AS
SELECT VolunteerID, Name, ContactDetails
FROM Volunteer;
```

15. This view displays information about all fundraisers.

```
CREATE VIEW FundraiserInfo AS
SELECT FundraiserID, Name, ContactDetails, CampaignID, FundsRaised, FundraiserDate
FROM Fundraiser;
```

16.This view provides details of all events.

```
CREATE VIEW EventDetails AS
SELECT EventID, Name, EventDate, Location, CampaignID
FROM Event;
```

17.This view lists contact details of political parties.

```
CREATE VIEW PartyContacts AS
SELECT PartyID, Name, Leader, Ideology, ContactDetails
FROM PoliticalParty;
```

18.This view displays information about all polling stations.

```
CREATE VIEW PollingStationInfo AS
SELECT PollingStationID, Name, Location, OpeningTime, ClosingTime, ConstituencyID
FROM PollingStation;
```

19. This view provides details of all constituencies.

```
CREATE VIEW ConstituencyDetails AS
SELECT ConstituencyID, Name, Region
FROM Constituency;
```

20.This view summarizes the total donations made by each donor.

```
CREATE VIEW DonationSummary AS
SELECT Donor.Name AS DonorName, SUM(DonationAmount) AS TotalDonation
FROM Donor
GROUP BY Donor.Name;
```

VoteX – Campaign Compass

Trigger Commands

Sanjay Ponnambalam N
RA2211026010136
Y1 – B. Tech CSE – AI & ML

Kirushakkarasu KT
RA2211026010134
Y1 – B. Tech CSE – AI & ML

1.Trigger to update constituency region when name changes

```
CREATE OR REPLACE TRIGGER UpdateConstituencyRegion
BEFORE UPDATE OF Name ON Constituency
FOR EACH ROW
BEGIN
   IF :OLD.Name <> :NEW.Name THEN
      UPDATE Constituency SET Region = :NEW.Region WHERE ConstituencyID = :NEW.ConstituencyID;
   END IF;
END;
/
```

2. Trigger to automatically update fundraiser date to current date

```
CREATE OR REPLACE TRIGGER UpdateFundraiserDate
BEFORE INSERT ON Fundraiser
FOR EACH ROW
BEGIN
   :NEW.FundraiserDate := SYSDATE;
END;
/
```

3. Trigger to enforce minimum donation amount for fundraisers

```
CREATE OR REPLACE TRIGGER CheckFundraiserDonation
BEFORE INSERT ON Fundraiser
FOR EACH ROW
BEGIN
   IF :NEW.FundsRaised < 1000 THEN
      RAISE_APPLICATION_ERROR(-20005, 'Minimum donation amount for fundraisers is 1000.');
   END IF;
END;
/
```

4.Trigger to automatically update volunteer contact details when inserted

```
CREATE OR REPLACE TRIGGER UpdateVolunteerContact
BEFORE INSERT ON Volunteer
FOR EACH ROW
BEGIN
   :NEW.ContactDetails := 'Contact details will be provided upon assignment.';
END;
/
```

5.Trigger to update campaign budget when fundraiser funds raised change

```
CREATE OR REPLACE TRIGGER UpdateCampaignBudget
AFTER INSERT OR UPDATE ON Fundraiser
FOR EACH ROW
BEGIN
   UPDATE Campaign
   SET Budget = Budget + :NEW.FundsRaised
   WHERE CampaignID = :NEW.CampaignID;
END;
```

/

6.Trigger to ensure events cannot be scheduled on weekends

```
CREATE OR REPLACE TRIGGER CheckEventWeekend
BEFORE INSERT OR UPDATE ON Event
FOR EACH ROW
BEGIN
   IF TO_CHAR(:NEW.EventDate, 'D') IN (1, 7) THEN
      RAISE_APPLICATION_ERROR(-20008, 'Events cannot be scheduled on weekends.');
   END IF;
END;
/
```

7.Trigger to enforce maximum length for political party names

```
CREATE OR REPLACE TRIGGER CheckPoliticalPartyNameLength
BEFORE INSERT OR UPDATE ON PoliticalParty
FOR EACH ROW
BEGIN
   IF LENGTH(:NEW.Name) > 50 THEN
      RAISE_APPLICATION_ERROR(-20010, 'Political party name cannot exceed 50 characters.');
   END IF;
END;
/
```

8.Trigger to update voter contact details when voter address changes

```
CREATE OR REPLACE TRIGGER UpdateVoterContactDetails
BEFORE UPDATE OF Street, City, State, ZipCode ON Voter
FOR EACH ROW
BEGIN
```

```
    :NEW.ContactDetails := :NEW.ContactDetails || ' - Updated Address';
END;
/
```

9.Trigger to restrict insertion of polling stations outside working hours

```
CREATE OR REPLACE TRIGGER CheckPollingStationHours
BEFORE INSERT ON PollingStation
FOR EACH ROW
BEGIN
    IF TO_CHAR(:NEW.OpeningTime, 'HH24:MI') < '08:00' OR TO_CHAR(:NEW.ClosingTime, 'HH24:MI')
> '18:00' THEN
        RAISE_APPLICATION_ERROR(-20011, 'Polling stations must operate between 8:00 AM and 6:00
PM.');
    END IF;
END;
/
```

10. Trigger to update constituency region when name changes

```
CREATE OR REPLACE TRIGGER UpdateConstituencyRegion
BEFORE UPDATE OF Name ON Constituency
FOR EACH ROW
BEGIN
    IF :OLD.Name <> :NEW.Name THEN
        UPDATE Constituency SET Region = :NEW.Region WHERE ConstituencyID = :NEW.ConstituencyID;
    END IF;
END;
/
```

11.Trigger to update candidate contact details when party affiliation changes

```
CREATE OR REPLACE TRIGGER UpdateCandidateContact
```

```
BEFORE UPDATE OF PartyAffiliation ON Candidate
FOR EACH ROW
BEGIN
  IF :OLD.PartyAffiliation <> :NEW.PartyAffiliation THEN
    UPDATE Candidate SET ContactDetails = :NEW.ContactDetails WHERE CandidateID =
:NEW.CandidateID;
  END IF;
END;
/
```

12.Trigger to insert a default campaign platform for a new candidate

```
CREATE OR REPLACE TRIGGER InsertDefaultCampaignPlatform
BEFORE INSERT ON Candidate
FOR EACH ROW
BEGIN
  IF :NEW.CampaignPlatform IS NULL THEN
    :NEW.CampaignPlatform := 'Default Campaign Platform';
  END IF;
END;
/
```

13.Trigger to insert a default description for a new campaign

```
CREATE OR REPLACE TRIGGER InsertDefaultDescription
BEFORE INSERT ON Campaign
FOR EACH ROW
BEGIN
  IF :NEW.Description IS NULL THEN
    :NEW.Description := 'Default Campaign Description';
  END IF;
END;
/
```

14. Trigger to update volunteer contact details when name changes

```
CREATE OR REPLACE TRIGGER UpdateVolunteerContact
BEFORE UPDATE OF Name ON Volunteer
FOR EACH ROW
BEGIN
   IF :OLD.Name <> :NEW.Name THEN
      UPDATE Volunteer SET ContactDetails = :NEW.ContactDetails WHERE VolunteerID =
:NEW.VolunteerID;
   END IF;
END;
/
```

15.Trigger to insert a default location for a new event

```
CREATE OR REPLACE TRIGGER InsertDefaultLocation
BEFORE INSERT ON Event
FOR EACH ROW
BEGIN
   IF :NEW.Location IS NULL THEN
      :NEW.Location := 'Default Event Location';
   END IF;
END;
/
```

16. Trigger to update fundraiser date when funds raised changes

```
CREATE OR REPLACE TRIGGER UpdateFundraiserDate
BEFORE UPDATE OF FundsRaised ON Fundraiser
FOR EACH ROW
BEGIN
   IF :OLD.FundsRaised <> :NEW.FundsRaised THEN
```

```
    UPDATE Fundraiser SET FundraiserDate = SYSDATE WHERE FundraiserID = :NEW.FundraiserID;
  END IF;
END;
/
```

17.Trigger to insert a default leader for a new political party

```
CREATE OR REPLACE TRIGGER InsertDefaultLeader
BEFORE INSERT ON PoliticalParty
FOR EACH ROW
BEGIN
  IF :NEW.Leader IS NULL THEN
    :NEW.Leader := 'Unknown Leader';
  END IF;
END;
/
```

18. Trigger to update polling station opening time when closing time changes

```
CREATE OR REPLACE TRIGGER UpdateOpeningTime
BEFORE UPDATE OF ClosingTime ON PollingStation
FOR EACH ROW
BEGIN
  IF :OLD.ClosingTime <> :NEW.ClosingTime THEN
    UPDATE PollingStation SET OpeningTime = :NEW.OpeningTime WHERE PollingStationID =
:NEW.PollingStationID;
  END IF;
END;
/
```

19. Trigger to delete associated events when a campaign is deleted

```
CREATE OR REPLACE TRIGGER DeleteAssociatedEvents
```

```
BEFORE DELETE ON Campaign
FOR EACH ROW
BEGIN
   DELETE FROM Event WHERE Event.CampaignID = :OLD.CampaignID;
END;
/
```

20. Trigger to insert a default location for a new event

```
CREATE OR REPLACE TRIGGER InsertDefaultLocation
BEFORE INSERT ON Event
FOR EACH ROW
BEGIN
   IF :NEW.Location IS NULL THEN
      :NEW.Location := 'Default Event Location';
   END IF;
END;
/
```

VoteX – Campaign Compass

Cursor Commands

Sanjay Ponnambalam N

RA2211026010136

Y1 – B. Tech CSE – AI & ML

Kirushakkarasu KT

RA2211026010134

Y1 – B. Tech CSE – AI & ML

---

1. Cursor to retrieve candidate names and their campaign platforms

```
DECLARE
   CURSOR CandidateCursor IS
      SELECT Name, CampaignPlatform
      FROM Candidate;
BEGIN
   FOR CandidateRec IN CandidateCursor LOOP
      DBMS_OUTPUT.PUT_LINE('Candidate: ' || CandidateRec.Name || ', Campaign Platform: ' ||
CandidateRec.CampaignPlatform);
   END LOOP;
END;
/
```

2. Cursor to calculate total funds raised by fundraisers

```
DECLARE
   TotalFunds NUMBER := 0;
BEGIN
   FOR FundraiserRec IN (SELECT FundsRaised FROM Fundraiser) LOOP
      TotalFunds := TotalFunds + FundraiserRec.FundsRaised;
```

```
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Total Funds Raised: ' || TotalFunds);
END;
/
```

3. Cursor to list event names and their locations

```
DECLARE
    CURSOR EventCursor IS
        SELECT Name, Location
        FROM Event;
BEGIN
    FOR EventRec IN EventCursor LOOP
        DBMS_OUTPUT.PUT_LINE('Event: ' || EventRec.Name || ', Location: ' || EventRec.Location);
    END LOOP;
END;
/
```

4.Cursor to display voter names and addresses

```
DECLARE
    CURSOR VoterCursor IS
        SELECT Name, Address
        FROM Voter;
BEGIN
    FOR VoterRec IN VoterCursor LOOP
        DBMS_OUTPUT.PUT_LINE('Voter: ' || VoterRec.Name || ', Address: ' || VoterRec.Address);
    END LOOP;
END;
/
```

5. Cursor to retrieve political party names and their leaders

```
DECLARE
  CURSOR PartyCursor IS
    SELECT Name, Leader
    FROM PoliticalParty;
BEGIN
  FOR PartyRec IN PartyCursor LOOP
    DBMS_OUTPUT.PUT_LINE('Party: ' || PartyRec.Name || ', Leader: ' || PartyRec.Leader);
  END LOOP;
END;
/
```

6.Cursor to display volunteer names and contact details

```
DECLARE
  CURSOR VolunteerCursor IS
    SELECT Name, ContactDetails
    FROM Volunteer;
BEGIN
  FOR VolunteerRec IN VolunteerCursor LOOP
    DBMS_OUTPUT.PUT_LINE('Volunteer: ' || VolunteerRec.Name || ', Contact: ' ||
VolunteerRec.ContactDetails);
  END LOOP;
END;
/
```

7.Cursor to retrieve constituency names and their regions

```
DECLARE
  CURSOR ConstituencyCursor IS
```

```
    SELECT Name, Region
    FROM Constituency;
BEGIN
  FOR ConstituencyRec IN ConstituencyCursor LOOP
    DBMS_OUTPUT.PUT_LINE('Constituency: ' || ConstituencyRec.Name || ', Region: ' ||
ConstituencyRec.Region);
  END LOOP;
END;
/
```

8. Cursor to list polling station names and their locations

```
DECLARE
  CURSOR PollingStationCursor IS
    SELECT Name, Location
    FROM PollingStation;
BEGIN
  FOR PollingStationRec IN PollingStationCursor LOOP
    DBMS_OUTPUT.PUT_LINE('Polling Station: ' || PollingStationRec.Name || ', Location: ' ||
PollingStationRec.Location);
  END LOOP;
END;
/
```

9. Cursor to display campaign IDs and their descriptions

```
DECLARE
  CURSOR CampaignCursor IS
    SELECT CampaignID, Description
    FROM Campaign;
BEGIN
```

```
  FOR CampaignRec IN CampaignCursor LOOP
    DBMS_OUTPUT.PUT_LINE('Campaign ID: ' || CampaignRec.CampaignID || ', Description: ' ||
CampaignRec.Description);
  END LOOP;
END;
/
```

10.Cursor to retrieve event names and their dates

```
DECLARE
  CURSOR EventCursor IS
    SELECT Name, EventDate
    FROM Event;
BEGIN
  FOR EventRec IN EventCursor LOOP
    DBMS_OUTPUT.PUT_LINE('Event: ' || EventRec.Name || ', Date: ' || EventRec.EventDate);
  END LOOP;
END;
/
```

11.number of polling stations

```
DECLARE
  totalPollingStations NUMBER := 0;
BEGIN
  -- Cursor to retrieve the count of polling stations
  FOR PollingStationRec IN (SELECT COUNT(*) AS StationCount FROM PollingStation) LOOP
    totalPollingStations := PollingStationRec.StationCount;
  END LOOP;
```

12. Display the total number of polling stations

```
  DBMS_OUTPUT.PUT_LINE('Total number of polling stations: ' || totalPollingStations);
END;
/
```

13. Cursor to list fundraiser names and their campaigns

```
DECLARE
  CURSOR FundraiserCampaignCursor IS
    SELECT f.Name AS FundraiserName, c.Description AS CampaignDescription
    FROM Fundraiser f
    JOIN Campaign c ON f.CampaignID = c.CampaignID;
BEGIN
  FOR FundraiserCampaignRec IN FundraiserCampaignCursor LOOP
    DBMS_OUTPUT.PUT_LINE('Fundraiser: ' || FundraiserCampaignRec.FundraiserName || ', Campaign: ' || FundraiserCampaignRec.CampaignDescription);
  END LOOP;
END;
/
```

14.Cursor to display political party names and their ideologies

```
DECLARE
  CURSOR PartyIdeologyCursor IS
    SELECT Name, Ideology
    FROM PoliticalParty;
BEGIN
  FOR PartyIdeologyRec IN PartyIdeologyCursor LOOP
    DBMS_OUTPUT.PUT_LINE('Party: ' || PartyIdeologyRec.Name || ', Ideology: ' || PartyIdeologyRec.Ideology);
  END LOOP;
END;
```

/

15. Cursor to calculate total funds raised by each fundraiser

```
DECLARE
   CURSOR FundraiserFundsCursor IS
      SELECT FundraiserID, SUM(FundsRaised) AS TotalFunds
      FROM Fundraiser
      GROUP BY FundraiserID;
BEGIN
   FOR FundraiserFundsRec IN FundraiserFundsCursor LOOP
      DBMS_OUTPUT.PUT_LINE('Fundraiser ID: ' || FundraiserFundsRec.FundraiserID || ', Total Funds Raised: ' || FundraiserFundsRec.TotalFunds);
   END LOOP;
END;
/
```

16. Cursor to display campaign IDs and their start dates

```
DECLARE
   CURSOR CampaignStartDateCursor IS
      SELECT CampaignID, StartDate
      FROM Campaign;
BEGIN
   FOR CampaignStartDateRec IN CampaignStartDateCursor LOOP
      DBMS_OUTPUT.PUT_LINE('Campaign ID: ' || CampaignStartDateRec.CampaignID || ', Start Date: ' || CampaignStartDateRec.StartDate);
   END LOOP;
END;
/
```

17.Cursor to retrieve Constituency details

```
DECLARE
  CURSOR ConstituencyCursor IS
    SELECT ConstituencyID, Name, Region
    FROM Constituency;
BEGIN
  -- Loop through each row in the cursor
  FOR ConstituencyRec IN ConstituencyCursor LOOP
    -- Output Constituency details
    DBMS_OUTPUT.PUT_LINE('Constituency ID: ' || ConstituencyRec.ConstituencyID || ', Name: ' ||
ConstituencyRec.Name || ', Region: ' || ConstituencyRec.Region);
  END LOOP;
END;
/
```

18.Cursor to retrieve Volunteer information

```
DECLARE
  CURSOR VolunteerCursor IS
    SELECT VolunteerID, Name, ContactDetails
    FROM Volunteer;
BEGIN
  -- Loop through each row in the cursor
  FOR VolunteerRec IN VolunteerCursor LOOP
    -- Output Volunteer information
    DBMS_OUTPUT.PUT_LINE('Volunteer ID: ' || VolunteerRec.VolunteerID || ', Name: ' ||
VolunteerRec.Name || ', Contact Details: ' || VolunteerRec.ContactDetails);
  END LOOP;
END;
/
```

19.Cursor to retrieve Fundraiser details

```
DECLARE
   CURSOR FundraiserCursor IS
      SELECT FundraiserID, Name, ContactDetails, CampaignID, FundsRaised, FundraiserDate
      FROM Fundraiser;
BEGIN
   -- Loop through each row in the cursor
   FOR FundraiserRec IN FundraiserCursor LOOP
      -- Output Fundraiser details
      DBMS_OUTPUT.PUT_LINE('Fundraiser ID: ' || FundraiserRec.FundraiserID || ', Name: ' ||
FundraiserRec.Name || ', Contact Details: ' || FundraiserRec.ContactDetails || ', Campaign ID: ' ||
FundraiserRec.CampaignID || ', Funds Raised: ' || FundraiserRec.FundsRaised || ', Fundraiser Date: '
|| FundraiserRec.FundraiserDate);
   END LOOP;
END;
/
```

20. Cursor to retrieve Event information

```
DECLARE
   CURSOR EventCursor IS
      SELECT EventID, Name, EventDate, Location, CampaignID
      FROM Event;
BEGIN
   -- Loop through each row in the cursor
   FOR EventRec IN EventCursor LOOP
      -- Output Event details
      DBMS_OUTPUT.PUT_LINE('Event ID: ' || EventRec.EventID || ', Name: ' || EventRec.Name || ',
Event Date: ' || EventRec.EventDate || ', Location: ' || EventRec.Location || ', Campaign ID: ' ||
EventRec.CampaignID);
   END LOOP;
END;
/
```