Social Unrest Database Construction

Overview

Purpose: Quantify social unrest in the United States (1950-2025) through a composite index measuring political violence, institutional trust, economic stress, external threats, and social fragmentation.

Author: Bruce Gavins

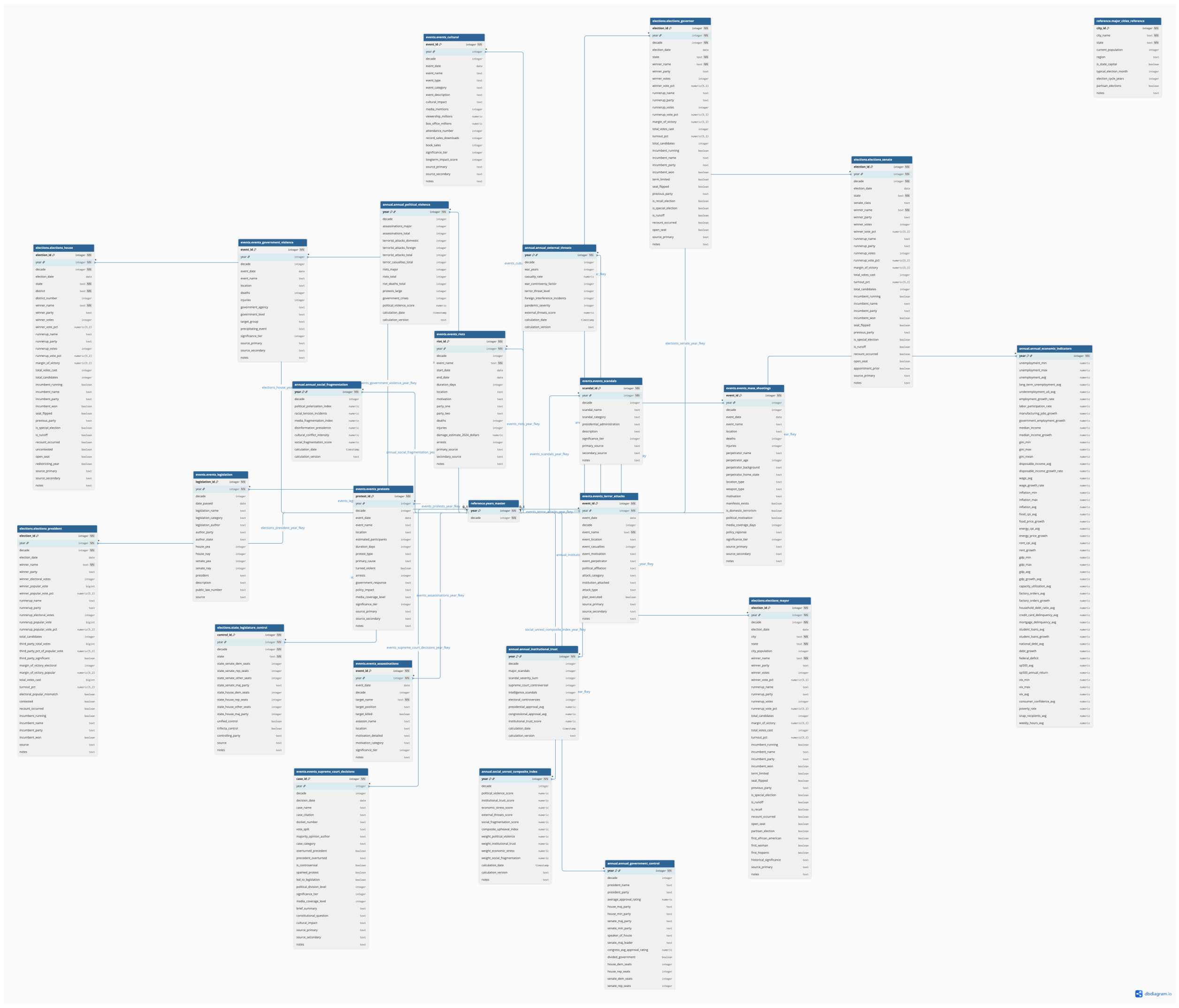
Date: 2025

Database: PostgreSQL 14+

Structure:

- reference schema: temporal and geographic foundation tables
- events schema: individual historical events (raw data)
- annual schema: yearly aggregated metrics and composite scores
- elections schema: election results at all government levels

Diagram



Schemas

1. Create schemas first to organize tables logically

```
CREATE SCHEMA IF NOT EXISTS reference;
COMMENT ON SCHEMA reference IS 'Foundation tables: temporal anchors an
d geographic references;

CREATE SCHEMA IF NOT EXISTS events;
```

```
COMMENT ON SCHEMA events IS 'Individual historical event recordeds (raw
data layer);

CREATE SCHEMA IF NOT EXISTS annual;
COMMENT ON SCHEMA annual is 'Yearly aggregated metrics and composite
unrest scores;

CREATE SCHEMA IF NOT EXISTS elections;
COMMENT ON SCHEMA elections IS 'Election results (presidential, congressi
onal, state, and local)';
```

Reference Tables

1. years_masters table: central temporal reference for all time -series data

```
CREATE TABLE reference.years_master (
      year INTEGER PRIMARY KEY,
      decade INTEGER NOT NULL,

      --Metadata
      created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

      --Constraints
      CONSTRAINT chk_year_range CHECK (year BETWEEN 1950 AND 2025),
      CONSTRAINT chk_decade_calc CHECK (decade = (year / 10) * 10)
);

COMMENT ON TABLE reference.years_masters IS 'Central temporal referenc
e: all time-series tables FK to this';
COMMENT ON COLUMN reference.years_master.year IS 'Primary temporal ke
y (1950-2025)';
COMMENT ON COLUMN reference.years_master.decade IS 'Computed decad
e grouping (1950, 1960, ..., 2020)';
```

```
-- Populate years 1950-2025
INSERT INTO reference.years_master (year, decade)
SELECT
    generate_series AS year,
    (generate_series / 10) * 10 AS decade
FROM generate_series(1950, 2025);
```

2. Major Cities Reference: metadata for tracked mayoral elections

```
CREATE TABLE reference.major_cities_reference (
    city_id SERIAL PRIMARY KEY,
    city_name TEXT NOT NULL,
    state TEXT NOT NULL,

    -- Geographic metadata
    region TEXT,
    is_state_capital BOOLEAN DEFAULT FALSE,
    current_population INTEGER,

    -- Election metadata
    typical_election_month INTEGER,
    election_cycle_years INTEGER,
    partisan_elections BOOLEAN DEFAULT TRUE,

    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT unique_city UNIQUE(city_name, state),
    CONSTRAINT chk_election_month CHECK (typical_election_month BETWEE
N 1 AND 12),
    CONSTRAINT chk_election_cycle CHECK (election_cycle_years IN (2, 4))
);
```

```
COMMENT ON TABLE reference.major_cities_reference IS 'Metadata for 50 m
ajor US cities tracked for mayoral elections';
COMMENT ON COLUMN reference.major_cities_reference.region IS 'Geograp
hic region: Northeast, South, Midwest, West';

-- Load city data
INSERT INTO reference.major_cities_reference (city_name, state, region, is_st
ate_capital, partisan_elections) VALUES
-- Northeast
('New York', 'NY', 'Northeast', FALSE, TRUE),
('Philadelphia', 'PA', 'Northeast', FALSE, TRUE),
('Boston', 'MA', 'Northeast', TRUE, TRUE),
('Newark', 'NJ', 'Northeast', FALSE, TRUE),
('Pittsburgh', 'PA', 'Northeast', FALSE, TRUE),
('Buffalo', 'NY', 'Northeast', FALSE, TRUE),
('Hartford', 'CT', 'Northeast', TRUE, TRUE),
('Manchester', 'NH', 'Northeast', FALSE, TRUE),

-- Midwest
('Chicago', 'IL', 'Midwest', FALSE, TRUE),
('Columbus', 'OH', 'Midwest', TRUE, TRUE),
('Indianapolis', 'IN', 'Midwest', TRUE, TRUE),
('Detroit', 'MI', 'Midwest', FALSE, TRUE),
('Milwaukee', 'WI', 'Midwest', FALSE, TRUE),
('Kansas City', 'MO', 'Midwest', FALSE, TRUE),
('Minneapolis', 'MN', 'Midwest', FALSE, TRUE),
('Cleveland', 'OH', 'Midwest', FALSE, TRUE),
('Cincinnati', 'OH', 'Midwest', FALSE, TRUE),
('St. Louis', 'MO', 'Midwest', FALSE, TRUE),
('Akron', 'OH', 'Midwest', FALSE, TRUE),
('Des Moines', 'IA', 'Midwest', TRUE, TRUE),
('Topeka', 'KS', 'Midwest', TRUE, TRUE),
('Lansing', 'MI', 'Midwest', TRUE, TRUE),

-- South
```

```
('Houston', 'TX', 'South', FALSE, TRUE),
('Charlotte', 'NC', 'South', FALSE, TRUE),
('Dallas', 'TX', 'South', FALSE, TRUE),
('Washington', 'DC', 'South', TRUE, TRUE),
('Miami', 'FL', 'South', FALSE, TRUE),
('Baltimore', 'MD', 'South', FALSE, TRUE),
('Atlanta', 'GA', 'South', TRUE, TRUE),
('Tampa', 'FL', 'South', FALSE, TRUE),
('New Orleans', 'LA', 'South', FALSE, TRUE),
('Orlando', 'FL', 'South', FALSE, TRUE),
('Richmond', 'VA', 'South', TRUE, TRUE),
('Little Rock', 'AR', 'South', TRUE, TRUE),
('Birmingham', 'AL', 'South', FALSE, TRUE),
('Mobile', 'AL', 'South', FALSE, TRUE),
('Jackson', 'MS', 'South', TRUE, TRUE),
('Nashville', 'TN', 'South', TRUE, TRUE),

-- West
('Los Angeles', 'CA', 'West', FALSE, TRUE),
('Phoenix', 'AZ', 'West', TRUE, TRUE),
('San Diego', 'CA', 'West', FALSE, TRUE),
('San Francisco', 'CA', 'West', FALSE, TRUE),
('Seattle', 'WA', 'West', FALSE, TRUE),
('Denver', 'CO', 'West', TRUE, TRUE),
('Oakland', 'CA', 'West', FALSE, TRUE),
('Sacramento', 'CA', 'West', TRUE, TRUE),
('Las Vegas', 'NV', 'West', FALSE, TRUE),
('Oklahoma City', 'OK', 'West', TRUE, TRUE),
('Honolulu', 'HI', 'West', TRUE, FALSE),
('Salt Lake City', 'UT', 'West', TRUE, TRUE);
```

Events Tables (Raw time-series data)

Design decision: separate tables per event type rather than single unified table

Reasoning:

1. Avoids sparse tables with excessive NULL values
2. Each event type has unique attributes (riots have durations, assassinations do not; assassinations have specific targets, riots do not)
3. Easier to maintain and validate type-specific constraints
4. Views provide unified access when needed

1. Assassinations table

```
CREATE TABLE events.events_assassinations (
    event_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER,
    event_date DATE,

    -- Target information
    target_name TEXT NOT NULL,
    target_position TEXT,
    target_killed BOOLEAN DEFAULT FALSE,

    -- Perpetrator
    assassin_name TEXT,

    -- Context
    location TEXT,
    motivation_detailed TEXT,
    motivation_category TEXT,

    -- Significance
    significance_tier INTEGER,

    -- Documentation
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    -- Constraints
    CONSTRAINT fk_assassination_year FOREIGN KEY (year) REFERENCES refe
rence.years_master(year),
    CONSTRAINT chk_significance CHECK (significance_tier BETWEEN 1 AND
5)
);

COMMENT ON TABLE events.events_assassinations IS 'Political assassination
s and attempts (presidents, leaders, activists)';
COMMENT ON COLUMN events.events_assassinations.significance_tier IS '1=
Minor, 5=Transformative (e.g., JFK, MLK)';
```

2. Terror attacks table

```
CREATE TABLE events.events_terror_attacks (
    event_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER,
    event_date DATE,

    -- Event details
    event_name TEXT NOT NULL,
    event_location TEXT,
    event_casualties INTEGER DEFAULT 0,

    -- Perpetrator
    event_perpetrator TEXT,
    political_affiliation TEXT,
    event_motivation TEXT,

    -- Classification
    attack_category TEXT,
    institution_attacked TEXT,
    attack_type TEXT,
    plan_executed BOOLEAN,

    -- Documentation
    source_primary TEXT,
    source_secondary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_terror_year FOREIGN KEY (year) REFERENCES reference.y
ears_master(year),
    CONSTRAINT chk_casualties CHECK (event_casualties >= 0)
);

COMMENT ON TABLE events.events_terror_attacks IS 'Domestic terrorist atta
cks with political motivation';
```

3. Riots table

```
CREATE TABLE events.events_riots (
    riot_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER,

    -- Event details
    event_name TEXT NOT NULL,
    start_date DATE,
    end_date DATE,
    duration_days INTEGER,
    location TEXT,

    -- Parties involved
    motivation TEXT,
    party_one TEXT,
    party_two TEXT,

    -- Impact metrics
    deaths INTEGER,
    injuries INTEGER,
    damage_estimate_2024_dollars DECIMAL,
    arrests INTEGER,

    -- Documentation
    primary_source TEXT,
    secondary_source TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_riot_year FOREIGN KEY (year) REFERENCES reference.yea
rs_master(year),
    CONSTRAINT chk_duration CHECK (duration_days >= 0),
    CONSTRAINT chk_date_order CHECK (end_date >= start_date)
);

COMMENT ON TABLE events.events_riots IS 'Major civil unrest events (Watts,
Detroit, post-MLK riots, etc.)';
```

4. Scandals table

```
CREATE TABLE events.events_scandals (
    scandal_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER,
    -- Note: No event_date - scandals typically span time periods

    scandal_name TEXT,
    scandal_category TEXT,
    presidential_administration TEXT,
    description TEXT,

    -- Significance
    significance_tier INTEGER,

    -- Documentation
    primary_source TEXT,
    secondary_source TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_scandal_year FOREIGN KEY (year) REFERENCES referenc
e.years_master(year),
    CONSTRAINT chk_significance CHECK (significance_tier BETWEEN 1 AND
5)
);

COMMENT ON TABLE events.events_scandals IS 'Major political scandals (Wa
tergate, Iran-Contra, etc.)';
COMMENT ON COLUMN events.events_scandals.year IS 'Year scandal beca
me public (not when underlying actions occurred)';
```

5. Legislation table (major federal legislation passed)

```
CREATE TABLE events.events_legislation (
    legislation_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER,
    date_passed DATE,

    -- Bill details
    legislation_name TEXT,
    legislation_category TEXT,
    public_law_number TEXT,

    -- Author information
    legislation_author TEXT,
    author_party TEXT,
    author_state TEXT,

    -- Vote tallies
    house_yea INTEGER,
    house_nay INTEGER,
    senate_yea INTEGER,
    senate_nay INTEGER,

    -- Context
    president TEXT,
    description TEXT,

    -- Documentation
    source TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_legislation_year FOREIGN KEY (year) REFERENCES referen
ce.years_master(year),
    CONSTRAINT chk_house_votes CHECK (house_yea >= 0 AND house_nay >
= 0),
    CONSTRAINT chk_senate_votes CHECK (senate_yea >= 0 AND senate_nay
>= 0)
);

COMMENT ON TABLE events.events_legislation IS 'Major federal legislation (C
ivil Rights Act, Patriot Act, etc.)';
```

6. Government violence tables (use of government/military force against civilians)

```
CREATE TABLE events.events_government_violence (
    event_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER NOT NULL,
    event_date DATE,

    event_name TEXT,
    location TEXT,

    -- Victims
    deaths INTEGER,
    injuries INTEGER,
    target_group TEXT,

    -- Perpetrator details
    government_agency TEXT,
    government_level TEXT,

    -- Context
    precipitating_event TEXT,
    significance_tier INTEGER,

    -- Documentation
    source_primary TEXT,
    source_secondary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_gov_violence_year FOREIGN KEY (year) REFERENCES refer
ence.years_master(year),
    CONSTRAINT chk_gov_level CHECK (government_level IN ('Federal', 'State',
'Local')),
    CONSTRAINT chk_casualties CHECK (deaths >= 0 AND injuries >= 0)
);

COMMENT ON TABLE events.events_government_violence IS 'Government us
e of force against civilians (Kent State, Bonus Army, etc.)';
```

7. Protests table (major demonstrations and movements)

```
CREATE TABLE events.events_protests (
    protest_id SERIAL PRIMARY KEY,
    year INTEGER,
    decade INTEGER,
    event_date DATE,

    event_name TEXT,
    location TEXT,

    -- Scale
    estimated_participants INTEGER,
    duration_days INTEGER,

    -- Characteristics
    protest_type TEXT,
    primary_cause TEXT,

    -- Outcome
    turned_violent BOOLEAN,
    arrests INTEGER,
    government_response TEXT,
    policy_impact TEXT,

    -- Significance
    media_coverage_level TEXT,
    significance_tier INTEGER,
    -- Documentation
    source_primary TEXT,
    source_secondary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_protest_year FOREIGN KEY (year) REFERENCES reference.
years_master(year),
    CONSTRAINT chk_participants CHECK (estimated_participants >= 0)
);

COMMENT ON TABLE events.events_protests IS 'Major protests and demonst
rations (March on Washington, anti-war protests, etc.)';
```

8. Mass shootings table

```
CREATE TABLE events.events_mass_shootings (
    event_id SERIAL PRIMARY KEY,
    year INTEGER,
    decade INTEGER,
    event_date DATE,

    event_name TEXT,
    location TEXT,

    -- Casualties
    deaths INTEGER,
    injuries INTEGER,

    -- Perpetrator
    perpetrator_name TEXT,
    perpetrator_age INTEGER,
    perpetrator_background TEXT,
    perpetrator_home_state TEXT,

    -- Context
    location_type TEXT,
    weapon_type TEXT,
    motivation TEXT,
    manifesto_exists BOOLEAN,

    -- Classification
    is_domestic_terrorism BOOLEAN,
    political_motivation BOOLEAN,

    -- Significance
    media_coverage_days INTEGER,
    policy_response TEXT,
    significance_tier INTEGER,

    -- Documentation
    source_primary TEXT,
    source_secondary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_shooting_year FOREIGN KEY (year) REFERENCES referenc
e.years_master(year),
    CONSTRAINT chk_casualties CHECK (deaths >= 0 AND injuries >= 0)
);

COMMENT ON TABLE events.events_mass_shootings IS 'Mass shooting event
s (4+ casualties)';
```

9. Cultural events table

```
CREATE TABLE events.events_cultural (
    event_id SERIAL PRIMARY KEY,
    year INTEGER,
    decade INTEGER,
    event_date DATE,

    event_name TEXT,
    event_type TEXT,
    event_category TEXT,
    event_description TEXT,
    cultural_impact TEXT,

    -- Quantifiable metrics
    media_mentions INTEGER,
    viewership_millions DECIMAL,
    box_office_millions DECIMAL,
    attendance_number INTEGER,
    record_sales_downloads INTEGER,
    book_sales INTEGER,

    -- Significance
    significance_tier INTEGER,
    longterm_impact_score INTEGER,

    -- Documentation
    source_primary TEXT,
    source_secondary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_cultural_year FOREIGN KEY (year) REFERENCES referenc
e.years_master(year),
    CONSTRAINT chk_impact CHECK (longterm_impact_score BETWEEN 1 AN
D 10)
);
```

```
COMMENT ON TABLE events.events_cultural IS 'Major cultural moments and
media events shaping public consciousness';
```

10. Supremer court decisions table

```
CREATE TABLE events.events_supreme_court_decisions (
    case_id SERIAL PRIMARY KEY,
    year INTEGER,
    decade INTEGER,
    decision_date DATE,

    -- Case details
    case_name TEXT,
    case_citation TEXT,
    docket_number TEXT,
    vote_split TEXT,
    majority_opinion_author TEXT,

    -- Classification
    case_category TEXT,
    constitutional_question TEXT,

    -- Precedent
    overturned_precedent BOOLEAN DEFAULT FALSE,
    precedent_overturned TEXT,

    -- Impact
    is_controversial BOOLEAN,
    sparked_protest BOOLEAN,
    led_to_legislation BOOLEAN,
    political_division_level INTEGER,
    cultural_impact TEXT,

    -- Significance
    significance_tier INTEGER,
    media_coverage_level INTEGER,

    -- Documentation
    brief_summary TEXT,
    source_primary TEXT,
    source_secondary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_scotus_year FOREIGN KEY (year) REFERENCES reference.
years_master(year),
    CONSTRAINT chk_division CHECK (political_division_level BETWEEN 1 AND
5),
    CONSTRAINT chk_coverage CHECK (media_coverage_level BETWEEN 1 AN
D 4)
);

COMMENT ON TABLE events.events_supreme_court_decisions IS 'Landmark
Supreme Court decisions (Brown v. Board, Roe v. Wade, etc.)';
```

Annual Aggregate Tables

These tables roll up event data into yearly metrics for composite index calculation. Populated via aggregation queries or ETL processes.

- Also includes yearly economic indicators

1. Economic indicators table

```sql
CREATE TABLE annual.annual_economic_indicators (
    year INTEGER PRIMARY KEY,

    -- Employment
    unemployment_min DECIMAL,
    unemployment_max DECIMAL,
    unemployment_avg DECIMAL,
    long_term_unemployment_avg DECIMAL,
    underemployment_u6_avg DECIMAL,
    employment_growth_rate DECIMAL,
    labor_participation_rate DECIMAL,
    manufacturing_jobs_growth DECIMAL,
    government_employment_growth DECIMAL,

    -- Income & inequality
    median_income DECIMAL,
    median_income_growth DECIMAL,
    gini_min DECIMAL,
    gini_max DECIMAL,
    gini_mean DECIMAL,
    disposable_income_avg DECIMAL,
    disposable_income_growth_rate DECIMAL,
    wage_avg DECIMAL,
    wage_growth_rate DECIMAL,

    -- Prices & inflation
    inflation_min DECIMAL,
    inflation_max DECIMAL,
    inflation_avg DECIMAL,
    food_cpi_avg DECIMAL,
    food_price_growth DECIMAL,
    energy_cpi_avg DECIMAL,
    energy_price_growth DECIMAL,
    rent_cpi_avg DECIMAL,
    rent_growth DECIMAL,
    -- GDP & production
    gdp_min DECIMAL,
    gdp_max DECIMAL,
    gdp_avg DECIMAL,
    gdp_growth_avg DECIMAL,
    capacity_utilization_avg DECIMAL,
    factory_orders_avg DECIMAL,
    factory_orders_growth DECIMAL,

    -- Debt & financial stress
    household_debt_ratio_avg DECIMAL,
    credit_card_delinquency_avg DECIMAL,
    mortgage_delinquency_avg DECIMAL,
    student_loans_avg DECIMAL,
    student_loans_growth DECIMAL,
    national_debt_avg DECIMAL,
    debt_growth DECIMAL,
    federal_deficit DECIMAL,

    -- Market & sentiment
    sp500_avg DECIMAL,
    sp500_annual_return DECIMAL,
    vix_min DECIMAL,
    vix_max DECIMAL,
    vix_avg DECIMAL,
    consumer_confidence_avg DECIMAL,

    -- Social indicators
    poverty_rate DECIMAL,
    snap_recipients_avg DECIMAL,
    weekly_hours_avg DECIMAL,

    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_economic_year FOREIGN KEY (year) REFERENCES referen
```

```
ce.years_master(year)
);

COMMENT ON TABLE annual.annual_economic_indicators IS 'Comprehensive
economic metrics aggregated by year (52 indicators)';
```

2. Government control table: political power distribution data

```
CREATE TABLE annual.annual_government_control (
    year INTEGER PRIMARY KEY,
    decade INTEGER,

    -- Executive
    president_name TEXT,
    president_party TEXT,
    average_approval_rating DECIMAL,

    -- Legislative control
    house_maj_party TEXT,
    house_min_party TEXT,
    house_dem_seats INTEGER,
    house_rep_seats INTEGER,

    senate_maj_party TEXT,
    senate_min_party TEXT,
    senate_dem_seats INTEGER,
    senate_rep_seats INTEGER,

    -- Leadership
    speaker_of_house TEXT,
    senate_maj_leader TEXT,

    -- Approval & division
    congress_avg_approval_rating DECIMAL,
    divided_government BOOLEAN,

    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_gov_control_year FOREIGN KEY (year) REFERENCES refere
nce.years_master(year)
);

COMMENT ON TABLE annual.annual_government_control IS 'Yearly snapshot
of political control across branches';
```

3. Political violence table: aggregated violence metrics

```
CREATE TABLE annual.annual_political_violence (
    year INTEGER PRIMARY KEY,
    decade INTEGER,

    -- Event counts
    assassinations_major INTEGER DEFAULT 0,
    assassinations_total INTEGER DEFAULT 0,

    terrorist_attacks_domestic INTEGER DEFAULT 0,
    terrorist_attacks_foreign INTEGER DEFAULT 0,
    terrorist_attacks_total INTEGER DEFAULT 0,
    terror_casualties_total INTEGER DEFAULT 0,

    riots_major INTEGER DEFAULT 0,
    riots_total INTEGER DEFAULT 0,
    riot_deaths_total INTEGER DEFAULT 0,

    protests_large INTEGER DEFAULT 0,
    government_crises INTEGER DEFAULT 0,

    -- Composite score
    political_violence_score DECIMAL,

    -- Metadata
    calculation_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    calculation_version TEXT DEFAULT '1.0',

    CONSTRAINT fk_political_violence_year FOREIGN KEY (year) REFERENCES
reference.years_master(year)
);

COMMENT ON TABLE annual.annual_political_violence IS 'Yearly aggregation
of political violence events';
COMMENT ON COLUMN annual.annual_political_violence.political_violence_s
core IS 'Calculated score (0-100): 20A + 15T + 10R + 5P + 8G, capped at 100';
```

4. Institutional trust metrics: trust erosion metrics

```
CREATE TABLE annual.annual_institutional_trust (
    year INTEGER PRIMARY KEY,
    decade INTEGER,

    -- Event counts
    major_scandals INTEGER DEFAULT 0,
    scandal_severity_sum INTEGER DEFAULT 0,
    supreme_court_controversial INTEGER DEFAULT 0,
    intelligence_scandals INTEGER DEFAULT 0,
    electoral_controversies INTEGER DEFAULT 0,

    -- Approval metrics
    presidential_approval_avg DECIMAL,
    congressional_approval_avg DECIMAL,

    -- Composite score
    institutional_trust_score DECIMAL,
    -- Metadata
    calculation_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    calculation_version TEXT DEFAULT '1.0',

    CONSTRAINT fk_institutional_trust_year FOREIGN KEY (year) REFERENCES reference.years_master(year)
);

COMMENT ON TABLE annual.annual_institutional_trust IS 'Yearly metrics of institutional trust erosion';
```

5. External threats: wars, terrorism, election interference, state meddling

```
CREATE TABLE annual.annual_external_threats (
    year INTEGER PRIMARY KEY,
    decade INTEGER,

    -- War metrics
    war_years INTEGER DEFAULT 0,
    casualty_rate DECIMAL,
    war_controversy_factor INTEGER,

    -- Other threats
    terror_threat_level INTEGER,
    foreign_interference_incidents INTEGER DEFAULT 0,
    pandemic_severity INTEGER DEFAULT 0,

    -- Composite score
    external_threats_score DECIMAL,

    -- Metadata
    calculation_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    calculation_version TEXT DEFAULT '1.0',
    CONSTRAINT fk_external_threats_year FOREIGN KEY (year) REFERENCES r
eference.years_master(year)
);

COMMENT ON TABLE annual.annual_external_threats IS 'Yearly metrics of ext
ernal threats (war, terrorism, pandemics)';
```

6. Social fragmentation: polarization, division, tribalism

```
CREATE TABLE annual.annual_social_fragmentation (
    year INTEGER PRIMARY KEY,
    decade INTEGER,

    -- Fragmentation metrics
    political_polarization_index DECIMAL,
    racial_tension_incidents DECIMAL,
    media_fragmentation_index DECIMAL,
    disinformation_prevalence DECIMAL,
    cultural_conflict_intensity DECIMAL,

    -- Composite score
    social_fragmentation_score DECIMAL,

    -- Metadata
    calculation_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    calculation_version TEXT DEFAULT '1.0',

    CONSTRAINT fk_social_fragmentation_year FOREIGN KEY (year) REFEREN
CES reference.years_master(year)
);
```

```
COMMENT ON TABLE annual.annual_social_fragmentation IS 'Yearly metrics o
f social division and polarization';
```

7. Composite Index: final unrest scores

```
CREATE TABLE annual.social_unrest_composite_index (
    year INTEGER PRIMARY KEY,
    decade INTEGER,

    -- Component scores (0-100 each)
    political_violence_score DECIMAL,
    institutional_trust_score DECIMAL,
    economic_stress_score DECIMAL,
    external_threats_score DECIMAL,
    social_fragmentation_score DECIMAL,

    -- Final composite index (0-100)
    composite_upheaval_index DECIMAL,

    -- Component weights (must sum to 1.0)
    weight_political_violence DECIMAL DEFAULT 0.25,
    weight_institutional_trust DECIMAL DEFAULT 0.20,
    weight_economic_stress DECIMAL DEFAULT 0.15,
    weight_external_threats DECIMAL DEFAULT 0.20,
    weight_social_fragmentation DECIMAL DEFAULT 0.20,

    -- Metadata
    calculation_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    calculation_version TEXT DEFAULT '1.0',
    notes TEXT,

    CONSTRAINT fk_composite_year FOREIGN KEY (year) REFERENCES refere
nce.years_master(year),
    CONSTRAINT chk_weights_sum CHECK (
        ABS((weight_political_violence + weight_institutional_trust +
             weight_economic_stress + weight_external_threats +
             weight_social_fragmentation) - 1.0) < 0.001
    )
);

COMMENT ON TABLE annual.social_unrest_composite_index IS 'Final weighte
d composite upheaval index combining all 5 components';
COMMENT ON COLUMN annual.social_unrest_composite_index.composite_u
pheaval_index IS 'Weighted average: 0.25*PV + 0.20*IT + 0.15*ES + 0.20*ET
+ 0.20*SF';
```

Election Results Tables

Covers most levels of American politics, from presidential to local elections.

1. Presidential elections table

```
CREATE TABLE elections.elections_president (
    election_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER NOT NULL,
    election_date DATE,

    -- Winner details
    winner_name TEXT NOT NULL,
    winner_party TEXT,
    winner_electoral_votes INTEGER,
    winner_popular_vote BIGINT,
    winner_popular_vote_pct DECIMAL(5,2),
    -- Runner-up details
    runnerup_name TEXT,
    runnerup_party TEXT,
    runnerup_electoral_votes INTEGER,
    runnerup_popular_vote BIGINT,
    runnerup_popular_vote_pct DECIMAL(5,2),

    -- Additional candidates
    total_candidates INTEGER,
    third_party_total_votes BIGINT,
    third_party_pct_of_popular_vote DECIMAL(5,2),
    third_party_significant BOOLEAN,

    -- Race characteristics
    margin_of_victory_electoral INTEGER,
    margin_of_victory_popular DECIMAL(5,2),
    total_votes_cast BIGINT,
    turnout_pct DECIMAL(5,2),

    -- Special circumstances
    electoral_popular_mismatch BOOLEAN DEFAULT FALSE,
    contested BOOLEAN DEFAULT FALSE,
    recount_occurred BOOLEAN DEFAULT FALSE,

    -- Incumbent information
    incumbent_running BOOLEAN,
    incumbent_name TEXT,
    incumbent_party TEXT,
    incumbent_won BOOLEAN,

    -- Documentation
    source TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_pres_year FOREIGN KEY (year) REFERENCES reference.ye
```

```
ars_master(year),
    CONSTRAINT chk_pres_percentages CHECK (
        winner_popular_vote_pct BETWEEN 0 AND 100 AND
        runnerup_popular_vote_pct BETWEEN 0 AND 100
    )
);

COMMENT ON TABLE elections.elections_president IS 'Presidential election r
esults (1952-present)';
```

2. Senate elections table

```
CREATE TABLE elections.elections_senate (
    election_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER NOT NULL,
    election_date DATE,
    state TEXT NOT NULL,
    senate_class TEXT,

    -- Winner details
    winner_name TEXT NOT NULL,
    winner_party TEXT,
    winner_votes INTEGER,
    winner_vote_pct DECIMAL(5,2),

    -- Runner-up details
    runnerup_name TEXT,
    runnerup_party TEXT,
    runnerup_votes INTEGER,
    runnerup_vote_pct DECIMAL(5,2),

    -- Race characteristics
    margin_of_victory DECIMAL(5,2),
    total_votes_cast INTEGER,
    turnout_pct DECIMAL(5,2),
    total_candidates INTEGER,

    -- Incumbent information
    incumbent_running BOOLEAN,
    incumbent_name TEXT,
    incumbent_party TEXT,
    incumbent_won BOOLEAN,

    -- Seat changes
    seat_flipped BOOLEAN DEFAULT FALSE,
    previous_party TEXT,

    -- Special circumstances
    is_special_election BOOLEAN DEFAULT FALSE,
    is_runoff BOOLEAN DEFAULT FALSE,
    recount_occurred BOOLEAN DEFAULT FALSE,

    -- Context
    open_seat BOOLEAN DEFAULT FALSE,
    appointment_prior BOOLEAN DEFAULT FALSE,

    -- Documentation
    source_primary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_senate_year FOREIGN KEY (year) REFERENCES reference.
years_master(year),
    CONSTRAINT unique_senate_election UNIQUE(year, state, is_special_electi
on)
);
```

```
COMMENT ON TABLE elections.elections_senate IS 'U.S. Senate election resu
lts by state';
```

3. House elections table

```
CREATE TABLE elections.elections_house (
    election_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER NOT NULL,
    election_date DATE,

    -- Geographic
    state TEXT NOT NULL,
    district TEXT NOT NULL,
    district_number INTEGER,

    -- Winner details
    winner_name TEXT NOT NULL,
    winner_party TEXT,
    winner_votes INTEGER,
    winner_vote_pct DECIMAL(5,2),

    -- Runner-up details
    runnerup_name TEXT,
    runnerup_party TEXT,
    runnerup_votes INTEGER,
    runnerup_vote_pct DECIMAL(5,2),

    -- Race characteristics
    margin_of_victory DECIMAL(5,2),
    total_votes_cast INTEGER,
    total_candidates INTEGER,

    -- Incumbent information
    incumbent_running BOOLEAN,
    incumbent_name TEXT,
    incumbent_party TEXT,
    incumbent_won BOOLEAN,

    -- Seat changes
    seat_flipped BOOLEAN DEFAULT FALSE,
    previous_party TEXT,

    -- Special circumstances
    is_special_election BOOLEAN DEFAULT FALSE,
    is_runoff BOOLEAN DEFAULT FALSE,
    recount_occurred BOOLEAN DEFAULT FALSE,
    uncontested BOOLEAN DEFAULT FALSE,

    -- Context
    open_seat BOOLEAN DEFAULT FALSE,
    redistricting_year BOOLEAN DEFAULT FALSE,

    -- Documentation
    source_primary TEXT,
    source_secondary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_house_year FOREIGN KEY (year) REFERENCES reference.
years_master(year),
    CONSTRAINT unique_house_election UNIQUE(year, state, district, is_specia
l_election)
);

COMMENT ON TABLE elections.elections_house IS 'U.S. House of Representa
tives election results by district';
```

4. Gubernatorial elections table

```
CREATE TABLE elections.elections_governor (
    election_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER NOT NULL,
    election_date DATE,
    state TEXT NOT NULL,

    -- Winner details
    winner_name TEXT NOT NULL,
    winner_party TEXT,
    winner_votes INTEGER,
    winner_vote_pct DECIMAL(5,2),

    -- Runner-up details
    runnerup_name TEXT,
    runnerup_party TEXT,
    runnerup_votes INTEGER,
    runnerup_vote_pct DECIMAL(5,2),

    -- Race characteristics
    margin_of_victory DECIMAL(5,2),
    total_votes_cast INTEGER,
    turnout_pct DECIMAL(5,2),
    total_candidates INTEGER,

    -- Incumbent information
    incumbent_running BOOLEAN,
    incumbent_name TEXT,
    incumbent_party TEXT,
    incumbent_won BOOLEAN,
    term_limited BOOLEAN DEFAULT FALSE,

    -- Seat changes
    seat_flipped BOOLEAN DEFAULT FALSE,
    previous_party TEXT,

    -- Special circumstances
    is_recall_election BOOLEAN DEFAULT FALSE,
    is_special_election BOOLEAN DEFAULT FALSE,
    is_runoff BOOLEAN DEFAULT FALSE,
    recount_occurred BOOLEAN DEFAULT FALSE,

    -- Context
    open_seat BOOLEAN DEFAULT FALSE,

    -- Documentation
    source_primary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_gov_year FOREIGN KEY (year) REFERENCES reference.yea
rs_master(year),
    CONSTRAINT unique_governor_election UNIQUE(year, state, is_special_ele
ction)
);

COMMENT ON TABLE elections.elections_governor IS 'Gubernatorial election
results by state';
```

5. Mayoral elections (50 top cities by population, historical significance, cultural significance)

```
CREATE TABLE elections.elections_mayor (
    election_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER NOT NULL,
    election_date DATE,
    city TEXT NOT NULL,
    state TEXT NOT NULL,
    city_population INTEGER,

    -- Winner details
    winner_name TEXT NOT NULL,
    winner_party TEXT,
    winner_votes INTEGER,
    winner_vote_pct DECIMAL(5,2),

    -- Runner-up details
    runnerup_name TEXT,
    runnerup_party TEXT,
    runnerup_votes INTEGER,
    runnerup_vote_pct DECIMAL(5,2),

    -- Additional candidates
    total_candidates INTEGER,

    -- Race characteristics
    margin_of_victory DECIMAL(5,2),
    total_votes_cast INTEGER,
    turnout_pct DECIMAL(5,2),

    -- Incumbent information
    incumbent_running BOOLEAN,
    incumbent_name TEXT,
    incumbent_party TEXT,
    incumbent_won BOOLEAN,
    term_limited BOOLEAN DEFAULT FALSE,

    -- Seat changes
    seat_flipped BOOLEAN DEFAULT FALSE,
    previous_party TEXT,

    -- Special circumstances
```

```
    is_special_election BOOLEAN DEFAULT FALSE,
    is_runoff BOOLEAN DEFAULT FALSE,
    is_recall BOOLEAN DEFAULT FALSE,
    recount_occurred BOOLEAN DEFAULT FALSE,

    -- Context
    open_seat BOOLEAN DEFAULT FALSE,
    partisan_election BOOLEAN DEFAULT TRUE,

    -- Historical significance
    first_african_american BOOLEAN DEFAULT FALSE,
    first_woman BOOLEAN DEFAULT FALSE,
    first_hispanic BOOLEAN DEFAULT FALSE,
    historical_significance TEXT,

    -- Documentation
    source_primary TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_mayor_year FOREIGN KEY (year) REFERENCES reference.
years_master(year),
    CONSTRAINT unique_mayor_election UNIQUE(year, city, is_special_electio
n)
);

COMMENT ON TABLE elections.elections_mayor IS 'Mayoral election results f
or 50 major U.S. cities';
```

6. State legislature tables (aggregate)

   Aggregate rather than by individual election because:

   1. size of data would be far larger than other collections

2. different states, districts have differing elections processes and rules, making consistent columnar structure difficult
3. aggregate is appropriate enough for research and project purpose as it only needs to show political power by year and trends over time

```
CREATE TABLE elections.state_legislature_control (
    control_id SERIAL PRIMARY KEY,
    year INTEGER NOT NULL,
    decade INTEGER NOT NULL,
    state TEXT NOT NULL,

    -- State senate
    state_senate_dem_seats INTEGER,
    state_senate_rep_seats INTEGER,
    state_senate_other_seats INTEGER,
    state_senate_maj_party TEXT,

    -- State house
    state_house_dem_seats INTEGER,
    state_house_rep_seats INTEGER,
    state_house_other_seats INTEGER,
    state_house_maj_party TEXT,

    -- Overall legislature control
    unified_control BOOLEAN,
    trifecta_control BOOLEAN,
    controlling_party TEXT,

    -- Documentation
    source TEXT,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_legislature_year FOREIGN KEY (year) REFERENCES refere
nce.years_master(year),
    CONSTRAINT unique_state_year UNIQUE(year, state)
);

COMMENT ON TABLE elections.state_legislature_control IS 'Aggregate state l
egislative control by year (avoids tracking 7000+ individual races)';
```

Indexes

- performance optimization for common query patterns

1. Reference table indexes

```
CREATE INDEX idx_years_decade ON reference.years_master(decade);

CREATE INDEX idx_cities_region ON reference.major_cities_reference(region);
CREATE INDEX idx_cities_state ON reference.major_cities_reference(state);
```

2. Events table indexes

```
-- Assassinations
CREATE INDEX idx_assassinations_year ON events.events_assassinations(yea
r);
CREATE INDEX idx_assassinations_decade ON events.events_assassinations
(decade);
CREATE INDEX idx_assassinations_date ON events.events_assassinations(eve
nt_date);

-- Terror attacks
```

```
CREATE INDEX idx_terror_year ON events.events_terror_attacks(year);
CREATE INDEX idx_terror_decade ON events.events_terror_attacks(decade);
CREATE INDEX idx_terror_date ON events.events_terror_attacks(event_date);

-- Riots
CREATE INDEX idx_riots_year ON events.events_riots(year);
CREATE INDEX idx_riots_decade ON events.events_riots(decade);
CREATE INDEX idx_riots_start_date ON events.events_riots(start_date);

-- Scandals
CREATE INDEX idx_scandals_year ON events.events_scandals(year);
CREATE INDEX idx_scandals_decade ON events.events_scandals(decade);
CREATE INDEX idx_scandals_admin ON events.events_scandals(presidential_a
dministration);

-- Legislation
CREATE INDEX idx_legislation_year ON events.events_legislation(year);
CREATE INDEX idx_legislation_decade ON events.events_legislation(decade);
CREATE INDEX idx_legislation_date ON events.events_legislation(date_passe
d);

-- Government violence
CREATE INDEX idx_gov_violence_year ON events.events_government_violence
(year);
CREATE INDEX idx_gov_violence_decade ON events.events_government_viole
nce(decade);

-- Protests
CREATE INDEX idx_protests_year ON events.events_protests(year);
CREATE INDEX idx_protests_decade ON events.events_protests(decade);

-- Mass shootings
CREATE INDEX idx_shootings_year ON events.events_mass_shootings(year);
CREATE INDEX idx_shootings_decade ON events.events_mass_shootings(dec
ade);
```

```
-- Cultural events
CREATE INDEX idx_cultural_year ON events.events_cultural(year);
CREATE INDEX idx_cultural_decade ON events.events_cultural(decade);

-- Supreme Court
CREATE INDEX idx_scotus_year ON events.events_supreme_court_decisions(y
ear);
CREATE INDEX idx_scotus_decade ON events.events_supreme_court_decision
s(decade);
```

3. Annual table indexes

```
CREATE INDEX idx_political_violence_decade ON annual.annual_political_viole
nce(decade);
CREATE INDEX idx_institutional_trust_decade ON annual.annual_institutional_t
rust(decade);
CREATE INDEX idx_external_threats_decade ON annual.annual_external_threat
s(decade);
CREATE INDEX idx_social_frag_decade ON annual.annual_social_fragmentatio
n(decade);
CREATE INDEX idx_composite_decade ON annual.social_unrest_composite_in
dex(decade);
```

4. Election table indexes

```
CREATE INDEX idx_pres_year ON elections.elections_president(year);
CREATE INDEX idx_senate_year_state ON elections.elections_senate(year, stat
e);
CREATE INDEX idx_house_year_state ON elections.elections_house(year, stat
e);
CREATE INDEX idx_gov_year_state ON elections.elections_governor(year, stat
e);
CREATE INDEX idx_mayor_year_city ON elections.elections_mayor(year, city);
```

View Tables

- unified access to related data across schemas

1. Unified Events View

```
CREATE VIEW events.events_all AS
SELECT
    'Assassination'::TEXT as event_type,
    event_id,
    year,
    decade,
    event_date,
    target_name as event_name,
    notes as event_summary,
    'events.events_assassinations'::TEXT as source_table
FROM events.events_assassinations

UNION ALL

SELECT
    'Terror Attack'::TEXT,
    event_id,
    year,
    decade,
    event_date,
    event_name,
    notes,
    'events.events_terror_attacks'::TEXT
FROM events.events_terror_attacks

UNION ALL

SELECT
    'Riot'::TEXT,
    riot_id,
    year,
    decade,
    start_date as event_date,
    event_name,
    notes,
    'events.events_riots'::TEXT
FROM events.events_riots

UNION ALL

SELECT
    'Scandal'::TEXT,
    scandal_id,
    year,
    decade,
    NULL::DATE as event_date,  -- Scandals span time periods
    scandal_name,
    description,
    'events.events_scandals'::TEXT
FROM events.events_scandals

UNION ALL

SELECT
    'Mass Shooting'::TEXT,
    event_id,
    year,
    decade,
    event_date,
    event_name,
    notes,
    'events.events_mass_shootings'::TEXT
FROM events.events_mass_shootings

UNION ALL

SELECT
    'Government Violence'::TEXT,
    event_id,
    year,
    decade,
    event_date,
    event_name,
    notes,
    'events.events_government_violence'::TEXT
FROM events.events_government_violence

UNION ALL

SELECT
    'Protest'::TEXT,
    protest_id,
    year,
    decade,
    event_date,
    event_name,
    notes,
    'events.events_protests'::TEXT
FROM events.events_protests

UNION ALL

SELECT
```

```
    'Cultural Event'::TEXT,
    event_id,
    year,
    decade,
    event_date,
    event_name,
    event_description,
    'events.events_cultural'::TEXT
FROM events.events_cultural

UNION ALL

SELECT
    'Legislation'::TEXT,
    legislation_id,
    year,
    decade,
    date_passed,
    legislation_name,
    description,
    'events.events_legislation'::TEXT
FROM events.events_legislation

UNION ALL

SELECT
    'Supreme Court'::TEXT,
    case_id,
    year,
    decade,
    decision_date,
    case_name,
    brief_summary,
    'events.events_supreme_court_decisions'::TEXT
FROM events.events_supreme_court_decisions
```

```
ORDER BY event_date DESC NULLS LAST;

COMMENT ON VIEW events.events_all IS 'Unified view of all historical events
across all event tables';
```

2. Unified Elections View

```
CREATE VIEW elections.elections_all AS
SELECT
    'Presidential'::TEXT as election_type,
    'Federal'::TEXT as election_level,
    election_id,
    year,
    decade,
    election_date,
    NULL::TEXT as state,
    NULL::TEXT as district,
    winner_name,
    winner_party,
    winner_popular_vote as winner_votes,
    winner_popular_vote_pct as winner_vote_pct,
    runnerup_name,
    runnerup_party,
    runnerup_popular_vote as runnerup_votes,
    runnerup_popular_vote_pct as runnerup_vote_pct,
    margin_of_victory_popular as margin_of_victory,
    total_votes_cast,
    turnout_pct,
    incumbent_running,
    incumbent_name,
    incumbent_party,
    incumbent_won,
    FALSE::BOOLEAN as seat_flipped,
    contested as is_contested,
    FALSE::BOOLEAN as is_special_election,
    FALSE::BOOLEAN as is_runoff,
    recount_occurred,
    'elections.elections_president'::TEXT as source_table,
    notes
FROM elections.elections_president

UNION ALL

SELECT
    'Senate'::TEXT,
    'Federal'::TEXT,
    election_id,
    year,
    decade,
    election_date,
    state,
    NULL::TEXT,
    winner_name,
    winner_party,
    winner_votes::BIGINT,
    winner_vote_pct,
    runnerup_name,
    runnerup_party,
    runnerup_votes::BIGINT,
    runnerup_vote_pct,
    margin_of_victory,
    total_votes_cast::BIGINT,
    turnout_pct,
    incumbent_running,
    incumbent_name,
    incumbent_party,
    incumbent_won,
    seat_flipped,
    FALSE::BOOLEAN,
    is_special_election,
    is_runoff,
    recount_occurred,
    'elections.elections_senate'::TEXT,
    notes
FROM elections.elections_senate

UNION ALL

SELECT
    'House'::TEXT,
    'Federal'::TEXT,
    election_id,
    year,
    decade,
    election_date,
    state,
    district,
    winner_name,
    winner_party,
    winner_votes::BIGINT,
    winner_vote_pct,
    runnerup_name,
    runnerup_party,
    runnerup_votes::BIGINT,
    runnerup_vote_pct,
    margin_of_victory,
    total_votes_cast::BIGINT,
    NULL::DECIMAL as turnout_pct,
    incumbent_running,
    incumbent_name,
    incumbent_party,
    incumbent_won,
    seat_flipped,
    FALSE::BOOLEAN,
    is_special_election,
    is_runoff,
    recount_occurred,
    'elections.elections_house'::TEXT,
    notes
FROM elections.elections_house

UNION ALL

SELECT
    'Governor'::TEXT,
    'State'::TEXT,
    election_id,
    year,
    decade,
    election_date,
    state,
    NULL::TEXT,
    winner_name,
    winner_party,
    winner_votes::BIGINT,
    winner_vote_pct,
    runnerup_name,
    runnerup_party,
    runnerup_votes::BIGINT,
    runnerup_vote_pct,
    margin_of_victory,
    total_votes_cast::BIGINT,
    turnout_pct,
    incumbent_running,
    incumbent_name,
    incumbent_party,
    incumbent_won,
    seat_flipped,
    is_recall_election,
    is_special_election,
    is_runoff,
    recount_occurred,
    'elections.elections_governor'::TEXT,
    notes
FROM elections.elections_governor

UNION ALL

SELECT
    'Mayor'::TEXT,
    'Local'::TEXT,
    election_id,
    year,
    decade,
    election_date,
    state,
    city as district,
    winner_name,
    winner_party,
    winner_votes::BIGINT,
    winner_vote_pct,
    runnerup_name,
    runnerup_party,
    runnerup_votes::BIGINT,
    runnerup_vote_pct,
    margin_of_victory,
    total_votes_cast::BIGINT,
    turnout_pct,
    incumbent_running,
    incumbent_name,
    incumbent_party,
    incumbent_won,
    seat_flipped,
    is_recall,
    is_special_election,
    is_runoff,
    recount_occurred,
    'elections.elections_mayor'::TEXT,
    notes
FROM elections.elections_mayor;

COMMENT ON VIEW elections.elections_all IS 'Unified view of all election res
ults across all levels of government';
```

3. Mayoral Elections with city context View

```
CREATE VIEW elections.mayoral_elections_context AS
SELECT
    e.*,
    c.region,
    c.is_state_capital,
    c.current_population as current_city_population
FROM elections.elections_mayor e
LEFT JOIN reference.major_cities_reference c
    ON e.city = c.city_name AND e.state = c.state;

COMMENT ON VIEW elections.mayoral_elections_context IS 'Mayoral election
s enriched with city metadata from reference table';
```