Jennifer Haliewicz – Data Portfolio

Work Sample: Cleaning & Normalizing a Mock Donor Database

Project Overview

This work sample demonstrates my ability to clean and normalize messy donor data, design a proper schema, and implement SQL solutions to enforce data integrity and consistency.

Skills Highlighted

- Data Cleaning & Normalization
- SQL / Database Design
- Data Quality Management
- Schema Design & Relationships
- Portfolio-ready Documentation

1. Generate Mock Data with Common Data-Quality Issues

I created a synthetic donor dataset containing realistic problems often encountered in nonprofit CRMs, including:

- Duplicate donor records
- Non-uniform data entry, such as inconsistent address/phone formats, varying capitalization of names, and mixed date formats
- Structural issue: the database creates a new donor record for each donation instead of maintaining a one-to-many relationship between donors and donations

2. Cleaning and Normalizing

a. Schema Design

Created a normalized database structure:

- **Donors Table:** Stores unique donor information.
- **Donations Table:** Stores donation records linked to donors via `donor_id`.

```
CREATE TABLE donors (
    donor_id SERIAL PRIMARY KEY,
    first_name VARCHAR(225) NOT NULL,
    last_name VARCHAR(225) NOT NULL,
    email VARCHAR(225),
    address VARCHAR(225) NOT NULL,
    city VARCHAR(225) NOT NULL,
    phone VARCHAR(20),
    state VARCHAR(225) NOT NULL
);

CREATE TABLE donations (
    donation_id SERIAL PRIMARY KEY,
    donor_id INT NOT NULL,
    donation_amount DECIMAL(10,2),
    donation_date DATE,
    campaign VARCHAR(225),
    donation_type VARCHAR(225),
    FOREIGN KEY (donor_id) REFERENCES donors(donor_id)
);
```

b. Populating the Donors Table

To populate donors while avoiding duplicates:

```
INSERT INTO donors (first_name, last_name, email, address, city, phone, state)
SELECT first_name, last_name, email, address, city, phone, state
FROM (
    SELECT *,
           ROW_NUMBER() OVER (PARTITION BY address ORDER BY donation_date DESC) AS rn
    FROM donors_donations
) AS ranked
WHERE rn = 1;
```

c. Linking Donations to Donors

```
ALTER TABLE donors_donations
ADD COLUMN donor_id INT;

UPDATE donors_donations dd
SET donor_id = d.donor_id
FROM donors d
WHERE dd.first_name = d.first_name
  AND dd.last_name = d.last_name
  AND dd.email = d.email;
```

d. Cleaning and Standardizing Dates

```
UPDATE donors_donations
SET donation_date =
    CASE
        WHEN donation_date ~ '^\d{2}/\d{2}/\d{4}$' THEN TO_DATE(REPLACE(donation_date,'/','-'),
'MM-DD-YYYY')
        WHEN donation_date ~ '^\d{2}-[A-Za-z]{3}-\d{4}$' THEN TO_DATE(donation_date, 'DD-Mon-
YYYY')
        ELSE donation_date::DATE
    END;
```

e. Populating the Donations Table

```
INSERT INTO donations (donor_id, donation_amount, donation_date, campaign, donation_type)
SELECT donor_id, donation_amount, donation_date, campaign, donation_type
FROM donors_donations;
```

f. Standardizing Address Abbreviations

```
UPDATE donors
SET address = REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(
               REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(
               REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(address,
               'Drive', 'Dr'),
               'Road', 'Rd'),
               'Street', 'St'),
               'Avenue', 'Ave'),
               'Trail', 'Trl'),
               'Expressway', 'Expy'),
               'Extension', 'Ext'),
```

```
            'Parkway', 'Pkwy'),
            'Station', 'Sta'),
            'Loop', 'Loop'),
            'Causeway', 'Cswy'),
            'Throughway', 'Thwy'),
            'Place', 'Pl'),
            'Bridge', 'Brg'),
            'Mount', 'Mt'),
            'Mountain', 'Mtn'),
            'Terrace', 'Ter'),
            'Village', 'Vlg'),
            'Haven', 'Hvn'),
            'Island', 'Is'),
            'Motorway', 'Mtwy');
```

g. Formatting Phone Numbers

```
UPDATE donors
SET phone = CASE
    WHEN phone ~ '^\d{10}$' THEN
        REGEXP_REPLACE(phone, '(\d{3})(\d{3})(\d{4})', '(\1) \2-\3')
    WHEN phone ~ '^\d{7}$' THEN
        REGEXP_REPLACE(phone, '(\d{3})(\d{4})', '\1-\2')
    ELSE
        phone
END;
```

h. Normalizing Name Capitalization

```
UPDATE donors
SET
    first_name = INITCAP(LOWER(first_name)),
    last_name = INITCAP(LOWER(last_name));
```

3. Next Steps / Best Practices

- Use appropriate data types for dates and numeric fields.
- Enforce consistent formats for phone numbers, postal codes, and other fields using constraints or regex.
- Implement stored procedures or triggers to automate data cleaning on new entries.

Database Schema

Normalized tables with auto-incremented IDs:

Donors Table

Column Name	Data Type	Key / Notes
donor_id	SERIAL	Primary Key, Auto-increment
first_name	VARCHAR(225)	Not Null
last_name	VARCHAR(225)	Not Null
email	VARCHAR(225)	
address	VARCHAR(225)	Not Null
city	VARCHAR(225)	Not Null
phone	VARCHAR(20)	
state	VARCHAR(225)	Not Null

Donations Table

Column Name	Data Type	Key / Notes
donation_id	SERIAL	Primary Key, Auto-increment
donor_id	INT	Foreign Key → donors(donor_id)
donation_amount	DECIMAL(10,2)	
donation_date	DATE	
campaign	VARCHAR(225)	
donation_type	VARCHAR(225)	

Relationships

Each donor can have multiple donations (1-to-many relationship):