College Data Pipeline

Betsy Fridman, Alex Hill, Ellie Huang, Steve Wong

Project Overview

Build a unified, year-over-year relational database combining **College Scorecard** and **IPEDS** data to enable fast, historical queries on U.S. higher-education institutions.

Main Objectives

- Build a longitudinal SQL database integrating annual College Scorecard and IPEDS data.
- Link datasets using UNITID/OPEID crosswalks.
- Track year-over-year changes in admissions, tuition, outcomes, and institutional characteristics.
- Store geographic and Carnegie Classification info for historical analysis.
- Enable fast lookups and trend analyses for any institution and year.

About the Data

General Institution Information (location, institution type)

- Source: National Center for Education Statistics - Integrated Postsecondary Education Data System (IPEDS)
- Documentation: Click “Complete Data Files”, and then the year
- Upload frequency: 3 times a year (Fall, Winter, Spring), last update was on September 23, 2025

Scorecard Data: More Detailed Institution Information (financials, earnings)

- Source: U.S. Department of Education (College Scorecard)
- Documentation
- Upload frequency: Annually, last update was on November 17, 2025

Data to Store: IPEDS

- Institution ID (unique ID, primary key)
- Institution name
- Institution location data (address, city, state abbreviation, zip etc.)
- Carnegie classification
- Year (the end year for that academic period)

Data to Store: College Scorecard

- Institution ID (unique ID)
- Accreditation agency (grants formal recognition to educational institutions)
- Degree (predominant, highest)
- Control (private/public/nonprofit)
- Region
- Admission rate
- Tuition information (in-state, out-of-state, or program-specific, also revenue from tuition per student)
- Average faculty salary
- Default rates

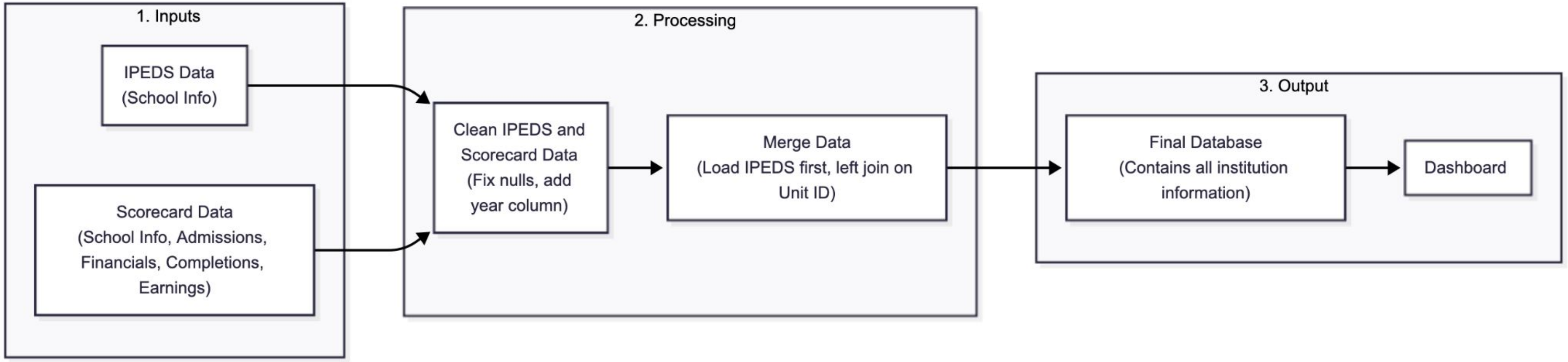
Schema Overview

1 main table

- **institution_ipeds_info** (parent table, every other table references this one) → institution-specific information that doesn't change very often

4 reference tables

- **institution_scorecard_info** → more institution info (Updated annually)
- **institution_financials** → yearly financial metrics
- **institution_admissions** → varies year to year, testing metrics
- **institution_completion** → graduation rates and demographic breakdowns/annual student outcomes


1. Inputs
IPEDS Data (School Info)
Scorecard Data (School Info, Admissions, Financials, Completions, Earnings)
2. Processing
Clean IPEDS and Scorecard Data (Fix nulls, add year column)
Merge Data (Load IPEDS first, left join on Unit ID)
3. Output
Final Database (Contains all institution information)
Dashboard

Set-up

1. Clone GitHub repository

2. Create local files (credentials_copy.py to store credentials)

3. Store data in root folder

Recommended Repo Structure

```
├── College_Scorecard_export (data zip
files)
├── README.md
├── credentials_copy.py
├── part_two.ipynb
├── load-ipeds.py
├── load-scorecard.py
```

Execute Order:

1. Create credentials_copy.py + upload sensitive user/database information
2. part_two.ipynb to create tables
3. load-ipeds.py FIRST (four times)
4. Then, load-scorecard.py. (four times as well)

Preprocessing IPEDS data

Column Selection & Validation: Selects columns (15 variables) format and creates a year variable, added as a new column.

Date & Time Handling: (To do) need to implement to ensure consistency across time zones, formatting, etc.

Numeric Data Cleaning: Replaces NA and nan with None across all 16 metric columns.

Preprocessing Scorecard data

Column Selection & Filtering: Selects required columns from scorecard data, then filters to only valid Unit IDs (valid = matches in IPEDS dataset, otherwise drop the row)

Ensure UNITID exists in scorecard data first.

Load script - IPEDS

Load CSV Input

- Takes file path from `sys.argv[1]`
- Reads the CSV into a pandas DataFrame
- Extracts the year from the filename and adds it as a `YEAR` column

Database Connection

- Uses credentials from `credentials_copy.py`
- Connects to the Postgres database with `psycopg2`

Insert/Update Logic (`insert_dataframe()`)

- Keeps only the columns needed for the target table
- For each row in the DataFrame:
  - If `UNITID` exists in the database: **update** the existing row
  - If `UNITID` does not exist: **insert** a new row
- Processes the data row-by-row until complete

Load script - Scorecard

Load and Prepare Data

- Takes CSV path from `sys.argv[1]` and loads it with `pd.read_csv()`
- Extracts YEAR from filename and adds **YEAR + 1** (Scorecard release lag)
- Splits the large Scorecard file into four cleaned dataframes

Database Prep

- Connects using credentials from `credentials_copy.py`
- Queries all valid UNITIDs from `institution_ipeds_info`
- Filters each Scorecard dataframe to only keep UNITIDs that already exist
- Prints counts of kept vs dropped rows

Insert Process (`insert_dataframe_strict()`)

- Takes a cleaned dataframe, target table name, and DB credentials, connect to database.
- Inserts rows **one by one**
  - If **any row fails**, the function **stops immediately** and performs a full **ROLLBACK**
  - If all rows succeed, performs a single **COMMIT**

Batching Strategy - (To do: look into faster processing)

- **Performance Optimization**: Executing 500 rows at once with executemany() is significantly faster than 500 individual execute() calls, reducing database round trips and network overhead
- **Transaction Efficiency**: Each batch commits as a unit, balancing between performance (larger batches) and recovery speed (smaller batches when errors occur)

Error Handling

- If a batch fails, print the error
  - Start inserting the rows in the batch one-by-one
  - Stop when the problematic row is found
  - Print the exact row and the reason for the error
  - Stop execution of program
- No data is inserted when an error occurs