# Assignment 1 Grading Rubric

## Overview

This rubric ensures consistency between students and graders. Assignment 1 focuses on creating tables, loading the Reddit dataset, and correctly applying constraints. Grading is fully automated based on your assignment1.sh and .sql scripts.

## Grading Breakdown (100 points total)

1. Database Creation & Normal Insertion – 40 points

- Table creation (20 pts):

All required tables must be created with correct names, schema, lowercase attribute names, primary keys, foreign keys, and necessary constraints.

- Data loading (10 pts):

All CSV files (authors, subreddits, submissions, comments) must load successfully into their respective tables.

  - Row counts should match the raw datasets.
- Integrity checks (10 pts):
  - Primary key and foreign key constraints must be valid.
  - Only the five relationships in the provided figure will be graded.

2. Optimized Data Insertion – 10 points

  - Efficient insertion of all entries using **COPY** or **pg_bulkload** (only one method is required).
  - To earn full credit, your loading process must complete within ~300 seconds in the grading environment.
  - Reference performance: ~100 seconds using pg_bulkload.

3. Query Results – 50 points

The five required queries are equally weighted (10 points each). For full credit, your queries must produce the exact output schema and semantics described below. Your queries.sql must save each query result into a table named query1, query2, …, query5.

- Query 1 (10 pts):
  - Return the total number of comments authored by the user `xymemez`.
  - Output columns: count of comments.

- Query 2 (10 pts):

    - Return the total number of subreddits for each subreddit type.
    - Output columns: subreddit type, subreddit count.

- Query 3 (10 pts):

    - Return the top 10 subreddits arranged by the number of comments. For each, calculate the average score of comments, rounded to 2 decimal places.
    - Output columns: name, comments count, average score.

- Query 4 (10 pts):

    - Return users with average karma > 1,000,000. Display their name, link_karma, comment_karma, and a computed label column where label = 1 if link_karma >= comment_karma, else 0. Sort by average karma (descending), break ties alphabetically by name.
    - Output columns: name, link karma, comment karma, label.

- Query 5 (10 pts):

    - For user [deleted_user], return the count of comments grouped by subreddit type, across all subreddits where this user has commented.
    - Output columns: sr type, comments num.

### Grading Notes for Queries

- Each query must be correct and reproducible on the full dataset.
- Column names and table names **must match exactly** as specified.
- Queries will be checked by autograder, so formatting precision matters.
- Use CREATE TABLE queryX AS … to save results.
- You may create helper views or tables, but only query1–query5 will be graded.

## Testing Guidelines

Students should verify correctness by checking:

- All four required tables are created.
- Row counts for authors, subreddits, comments, and submissions match the assignment description.
- Primary key and foreign key constraints are present and enforced.
- assignment1.sh successfully executes without manual intervention.
- assignment1.sh calls the required .sql files but does not contain database creation.

## Submission Requirements

- Submit a zip file named Assignment-1.zip containing:
    - **assignment1.sh** (entry point script)
    - **.sql** file(s)
    - **README.md** (optional notes to graders)
- assignment1.sh must:
    - Call the .sql file to create the table(s)
    - Load the data using either:
        - COPY (from an additional .sql file), **or**
        - pg_bulkload
    - Generate five tables, namely, query1, query2, …, query5 respectively for each query.
- Assume CSV files are already in the same folder (./filename.csv)

## Policies & Notes

- No late submissions unless under documented emergency.
- Plagiarism will result in course failure; anti-plagiarism tools will be used.
- The grading environment uses **PostgreSQL 14** with **pg_bulkload pre-installed**.
- Scripts are executed under the postgres user with database name 'postgres'.
- After discussion with the professor, the autograding scripts themselves will not be released.