DATA 503 Lab Handout
Grafana Dashboards on Railway (PostgreSQL 18)

Lucas P. Cordova, Ph.D. — Willamette University

Spring 2026

1 How to use this handout

Work through the steps in order. You may work **individually** or in **pairs**. If you hit a blocker you cannot clear quickly, pair up with a neighbor and compare connection strings, service names, and error messages before asking for help.

Materials on Canvas:

- `railway_dump.dump` — custom-format backup for `pg_restore` (built for PostgreSQL 18)
- `railway_dump.sql` — optional plain SQL you can run if your computer does not have `pg_restore`

2 Part A: Create a PostgreSQL 18 database on Railway

1. Go to `https://railway.app` and sign in.
2. Click **New Project** → **Empty Project** (or use a dedicated class project if instructed).
3. Click **Create** → **Database** → **Add PostgreSQL**.
4. Open the PostgreSQL service → **Settings**. If there is a **Version** or image tag option, select **PostgreSQL 18** (or the version your instructor specifies).
5. Wait until the database shows as deployed (healthy).
6. Open the PostgreSQL service → **Variables** (or **Connect**). Copy the **public** database URL Railway gives for connections from your own computer (often the variable `DATABASE_PUBLIC_URL`, or the URL under **Public network / TCP Proxy** on the Connect tab). You will need host, port, database name (often `railway`), user, and password from that URL.
7. Use this **public** URL for Beekeeper, pgAdmin, `psql`, and `pg_restore`. Private hostnames such as `*.railway.internal` do not work from your laptop.

3 Part B: Connect with Beekeeper Studio or pgAdmin

Beekeeper Studio

1. New connection → **Postgres**.

2. Paste the full **public** database URL if the app supports it, or copy host, port, user, password, and database name from that URL.
3. Test connection, then save.

pgAdmin

1. Register → **Server**. On the **Connection** tab, enter host, port, maintenance database (often `railway`), username, and password from the **public** database URL.
2. For connections through Railway's public endpoint, set SSL mode to **Require** unless Railway's Connect instructions say otherwise.

4 Part C: Load the class database from the dump

Option 1: `pg_restore` (preferred for `.dump` files)

You need the PostgreSQL **client tools** installed (version 18 preferred; older clients may fail against newer servers).

1. Download `railway_dump.dump` from Canvas to a folder you can find.
2. In a terminal, set your connection string (replace with your real URL from Railway):

   ```
   export DATABASE_PUBLIC_URL="postgresql://USER:PASSWORD@PUBLIC_HOST:PORT/railway"
   ```

3. Run (flags avoid ownership errors on hosted Postgres):

   ```
   pg_restore --verbose --no-owner --no-acl -d "$DATABASE_PUBLIC_URL" railway_dump.dump
   ```

4. Refresh Beekeeper or pgAdmin. You should see the class tables (for example census, survey, temperature tables) and the dashboard views (names starting with `v_`, etc.).

If `pg_restore` prints errors about existing objects, ask before re-running with destructive flags. Your instructor may give a fresh empty database or a revised dump.

Option 2: Plain SQL file (no `pg_restore`)

1. Download `railway_dump.sql` from Canvas.
2. In Beekeeper: open the file or paste contents, ensure you are connected to your Railway database, and execute the script.
3. In pgAdmin: use **Query Tool** → open file → Execute.
4. Alternatively with `psql`:

   ```
   psql "$DATABASE_PUBLIC_URL" -f railway_dump.sql
   ```

5 Part D: Add Grafana on Railway

Grafana needs its *own* configuration database so dashboards survive restarts. Your **pipeline** database holds the class data; a separate database (commonly named `grafana`) holds Grafana metadata.

1. In Beekeeper or pgAdmin, connected to your Postgres service, run:

   `CREATE DATABASE grafana;`

2. In the same Railway **project**, click **Create** → **Template** and search for **Grafana**. Deploy a well-maintained template (your instructor may name a specific one).

3. On the Grafana service, set environment variables (exact names can vary slightly by image; confirm with the template readme if something fails):

   Authentication

   - `GF_SECURITY_ADMIN_USER` = `admin`
   - `GF_SECURITY_ADMIN_PASSWORD` = a password you will remember

   Grafana config database (points at your Postgres service)

   - `GF_DATABASE_TYPE` = `postgres`
   - `GF_DATABASE_HOST` = host and port from the **same public** Postgres URL you use in Beekeeper (for example `short-name.proxy.rlwy.net:PORT`)
   - `GF_DATABASE_NAME` = `grafana`
   - `GF_DATABASE_USER` = often `postgres`
   - `GF_DATABASE_PASSWORD` = the Postgres password from Railway variables
   - `GF_DATABASE_SSL_MODE` = usually `require` for the public endpoint (follow the template readme if the deploy fails)

   Public URL

   - After Grafana gets a public URL, set `GF_SERVER_ROOT_URL` to that URL (including `https://`).

4. Redeploy Grafana if prompted. Enable **Serverless** on the Grafana service if you want it to sleep when idle (saves credits).

5. Log into Grafana from the public URL. Under **Connections** → **Data sources**, add **PostgreSQL** for your **class data** database:

   - Host / port: same **public** Postgres host and port as Beekeeper
   - Database: the database name that contains the restored tables (often `railway`)
   - User / password: prefer a **read-only** user your instructor helps you create; never paste admin passwords into shared screenshots
   - SSL: **require** (or whatever matches Railway's public Postgres settings)

6. Click **Save & test**. Fix host, database name, or SSL until the check passes.

6 Part E: Lab scenarios (build a small dashboard)

Create one dashboard with at least **four** panels. Use meaningful titles. Set a sensible refresh interval (for static class data, manual or 5–15 minutes is fine).

Scenario 1 — State population Bar chart: top 15 states by `total_pop` from `v_state_population_summary`. Sort descending.

Scenario 2 — Migration story Horizontal bar chart from `v_state_migration_rates` (for example top 20 by absolute migration rate). Use field overrides or value mappings so "Gaining" and "Losing" read clearly (color is optional).

Scenario 3 — Growth categories Pie or donut chart from `v_county_growth_summary` using `growth_category` and `num_counties`, or stat panels for `total_pop` per category.

Scenario 4 — Establishments per capita Bar chart from `v_state_estabs_per_capita` for `estabs_per_thous` (limit to top 15 states).

Scenario 5 — Seasons and stations Grouped bar chart from `v_seasonal_temperatures`: `station_name`, `season`, `avg_max_temp`.

Scenario 6 — Population tiers Table or bar chart from `v_population_tiers` comparing `tier`, `num_counties`, and `pct_of_total_pop`.

Scenario 7 — Pivot-style table Table panel: run the ice cream `crosstab()` query from lecture (offices as rows, flavors as columns). If the query fails, confirm `CREATE EXTENSION tablefunc;` was applied in your database.

Scenario 8 — Temperature pivot Table panel: run the median temperature by station and month `crosstab()` query from lecture.

7 What to turn in

Link to your dashboard in the Canvas assignment.