Save the question, store the answer

Virtual datasets, `CREATE TABLE ... AS`, and an API

Names _____

Superset, database `Dades classe`, schema `public`, tables `pokemon` and `generations`. Today's analogy is a kitchen: the tables are the ingredients. Pairs: one types, the other writes here; swap at A2.

Reference

	Recipe	Tupperware
You keep	The question (the SQL).	The answer (the rows).
In Superset	A **virtual dataset**: a saved query.	A new table, made with `CREATE TABLE ... AS`.
It lives in	Superset: only the text of the query.	PostgreSQL: a real table, next to `pokemon`.
When you use it	It is cooked again, every time.	It is served as it was stored.

A **dataset** is what Superset can build charts from. A **physical dataset** points at a real table: a label on a jar, not a copy.

Part A · Save the question, store the answer (20 min)

A1 · The recipe. SQL → SQL Lab. Type and **Run** (it is Wednesday's JOIN):

```sql
SELECT p.id, p.pokemon, p.type_1, p.type_2, p.hp, p.attack, p.defense, p.speed,
       g.generation_id, g.region, g.release_year
FROM pokemon p
JOIN generations g ON p.generation_id = g.generation_id
```

Rows _____ `pokemon` has 949. Why not 949? _____

Arrow next to **Save** → **Save dataset** → **Save as new**, name `pokemon_with_region` → **Save & Explore**. The chart builder opens in a new browser tab: close it and open **Datasets**.

Type of `pokemon_with_region` _____ Type of `pokemon` and `generations` _____

A2 · The tupperware. New SQL Lab tab. First run the `SELECT` alone (from line 2). Then add line 1 and run it all.

```sql
CREATE TABLE pokemon_by_generation AS
SELECT g.generation_id, g.region, g.release_year,
       COUNT(p.id) AS n_pokemon,
       ROUND(AVG(p.attack)) AS avg_attack,
       MAX(p.speed) AS max_speed
FROM generations g
LEFT JOIN pokemon p ON p.generation_id = g.generation_id
GROUP BY g.generation_id, g.region, g.release_year;
```

The `SELECT` alone: rows _____ With line 1: what does SQL Lab show? _____

Where did the rows go? _____

Now run `SELECT * FROM pokemon_by_generation ORDER BY generation_id;` Rows _____

Most Pokémon: _____ with _____ `n_pokemon = 0`: _____ Its `avg_attack` says _____

Why is that cell not 0? _____

A3 · Label the jar. Superset has no dataset for the new table yet. **Datasets → + Dataset**: Database `Dades classe`, Schema `public`, Table `pokemon_by_generation`. Arrow next to **Create and explore dataset** → **Create dataset**.

Type of `pokemon_by_generation` in **Datasets** _____

Which one goes stale? (with the teacher, 5 min)

The teacher adds a ninth region (Paldea, 2022) to `generations` on the projector. Do not run it yourself: your tables must stay as they are. Write your prediction **before** it runs.

How many regions will it show?	I predict	It showed
The **table** `pokemon_by_generation` (the tupperware)		
The `SELECT` of A2 run again as a **query** (the recipe)		

Why? _____

Part B · From an API to a dataset (16 min)

An **API** is an address you ask, which answers with data instead of a web page. The answer comes in **JSON**: text made of names and values, where a value can hold more values inside. Superset cannot ask an API: you **extract**, then **load**.

B1 Open `https://pokeapi.co/api/v2/pokemon?limit=20` in the browser. No key, no login.

`count` says _____ Pokémon shown _____ Which field holds the address of the rest? _____

B2 Open `https://pokeapi.co/api/v2/pokemon/charizard`. In SQL Lab run `SELECT * FROM pokemon WHERE pokemon = 'charizard'`.

Height: API _____ table _____ Weight: API _____ table _____ Why? _____

How does each one store the two types? _____

B3 **Extract, flatten, load.** To **flatten** is to turn nested JSON into rows and columns. Download `api_to_csv.py` from the Resources page of the course site. In a terminal, in that folder: `python3 api_to_csv.py`. Open the `pokeapi_sample.csv` it writes. Then in Superset: **+ → Data → Upload CSV to database**, database `Dades classe`, schema `public`, table name `pokeapi_sample` → **Upload**. In SQL Lab: `SELECT type_1, COUNT(*) AS n FROM pokeapi_sample GROUP BY type_1 ORDER BY n DESC`

CSV: rows _____ columns _____ Top two types: _____ Type in **Datasets** _____

A new Pokémon comes out tomorrow. How do you get it into this table? _____

B4 **The API checklist.** Six questions before you build anything on an API. Fill in what you saw.

Ask	What to find out	PokéAPI
Address	The URL, and its **parameters** after the `?`.	
Key	Account or **API key** needed? A key is a password: never paste it anywhere.	
Limits	How many requests may you make?	
Pagination	Does one answer hold everything? Which field is next?	
Licence	May you reuse and publish the data? Who must you credit?	
Snapshot	Your table is a copy from one moment. Write down the date.	

Done when Every blank is filled · **Datasets** lists one **Virtual** and at least three **Physical** datasets. **Fast pair?** Repeat A1 and A2 on your own open dataset.

Stuck?

- **`relation "pokemon_by_generation" already exists`**: you ran it twice. Good news: the table is there.
- **`does not allow for DDL/DML`**: not your fault, call the teacher. **`Table already exists` on upload**: set **If table already exists** to **Replace**. **`pokeapi.co` does not load**: ask the teacher for the ready-made `pokeapi_sample.csv` and upload it.