Synthetic variables

Columns that were never in the file Names _____

Superset, database `Dades classe`, schema `public`. You start from the table `games` and the script `games_clean.sql` (ask the teacher for it). Pairs: one clicks, one reads; swap at Task 3. Results here.

Reference · the receipt

On the receipt	What it is	In SQL / Superset
Line total (units × price)	A new value **for every row**, made from that row only.	**Calculated column**: `owners_min * price`
TOTAL at the bottom	One value **for the whole group**; the rows disappear.	**Metric**: `SUM(owners_min * price)`
"This line is 40% of your receipt"	A group value written **next to every row**; the rows stay.	**Window function**: `SUM(...) OVER ()`

A **synthetic variable** is a column that nobody collected: it is computed from the ones that were. Read `OVER ()` as "over all the rows, without collapsing them"; `PARTITION BY` is the `GROUP BY` of a window.

Part A · Assemble `games_clean` (8 min)

Step	Do this	You should see / write
A1	**SQL → SQL Lab**, schema `public`: paste the whole `games_clean.sql` and run it. Then run `SELECT COUNT(*) FROM games_clean;`	Rows: _____ `games` has 26 688. Which line of the script removed the rest? _____
A2	Read the last statement of the script (`ALTER TABLE ...`). It adds a column that was never in the file.	Its name: _____ Computed from: _____
A3	**Datasets → + Dataset** → Database `Dades classe`, Schema `public`, Table `games_clean` → **Create dataset** (the arrow next to **Create and explore dataset**).	`games_clean` is in your Datasets list. Do not start Part B without it.

Part B · Task 1 · A calculated column (6 min)

Step	Do this	You should see / write
1.1	**Datasets** → on the row `games_clean`, the pencil under **Actions** (**Edit**) → tab **Calculated columns** → **Add item**. Name: `est_revenue`. **SQL expression**: delete the sample text first, then `owners_min * price`. **Data type**: `NUMERIC`. **Save** → **OK**.	Which line of the receipt is this? _____

1.2 Check the same expression in SQL Lab:

```
SELECT game, owners_min, price, owners_min * price AS est_revenue
FROM games_clean ORDER BY est_revenue DESC LIMIT 5;
```

Top game: _____ `est_revenue`: _____ In words (millions? billions?): _____

Stuck?

- **`This database does not allow for DDL/DML...`**: call the teacher. It is one tick in the connection settings.
- **The script ran halfway**: run it again whole. It always rebuilds the table from zero.
- **`This table already has a dataset`**: it is already registered. Look in the Datasets list.
- Still stuck after two tries? Ask your neighbours, then the teacher.

Task 2 · Two metrics (7 min)

Step	Do this	You should see / write

2.1	Same editor, tab **Metrics** → **Add item**, twice.
	`total_est_revenue` = `SUM(owners_min * price)`
	`pct_free` = `100.0 * SUM(CASE WHEN is_free THEN 1 ELSE 0 END) / COUNT(*)`
	Save → **OK**.
		Predict before step 2.2: which `owners_tier` (Hit, Mid, Niche) has the highest share of free games? _____

2.2	In SQL Lab, compute both expressions per tier: write the query yourself, with `GROUP BY owners_tier`.	Hit: _____ · _____ % Mid: _____ · _____ % Niche: _____ · _____ %
2.3	Run `SELECT SUM(est_revenue) FROM games_clean;` It fails **on purpose**.	The error says: _____ You made `est_revenue` in Task 1. Where does it live? _____

Task 3 · Window functions (11 min)

3.1 Run it exactly as written: the share of games in each price band.

```sql
SELECT price_band, COUNT(*) AS games,
       ROUND(100.0 * COUNT(*) / SUM(COUNT(*)) OVER (), 1) AS pct_of_games
FROM games_clean GROUP BY price_band ORDER BY games DESC;
```

Biggest band: _____ with _____ % On the receipt, what is `SUM(COUNT(*)) OVER ()`? _____

3.2 Each game against the average price of its own year.

```sql
SELECT game, release_year, price,
       ROUND(AVG(price) OVER (PARTITION BY release_year), 2)         AS year_avg,
       ROUND(price - AVG(price) OVER (PARTITION BY release_year), 2) AS vs_year_avg
FROM games_clean ORDER BY owners_min DESC, number LIMIT 5;
```

Which of the five costs more than the average of its year? _____ By how much? _____

3.3 Adapt the query of 3.1: instead of the share of **games** per `price_band`, the share of **estimated revenue** per `owners_tier`.

The Hit tier is _____ % of the games and _____ % of the estimated revenue.

3.4 Look at the Niche row of 3.3. Did those games really earn that? Give one reason why `est_revenue` is **not** what a game really earned.

_____

_____

Done when Every blank is filled · `games_clean` is in your Datasets list. **Fast pair?** Click the dataset name `games_clean` to open it in Explore. Chart type **Table**; **Dimensions**: `owners_tier`; **Metrics**: pick `total_est_revenue` from the **Saved** tab; **Create chart**. Do not save it. Then try the same three variables on your own open dataset.

At home Finish the handout. One more window: with `RANK() OVER (PARTITION BY release_year ORDER BY owners_min DESC)`, find the three most-owned games of 2017.