# **About Project**

**Project:**

Business intelligence analysis of mobile phones marketing with training dataset.

**We have:**

A csv file with mobile phones information of price and characteristics.

**Task:**

1. Create a Data Base and ERD consisting of 4 tables and having connections with each other. (in PostgreSQL)
2. Make data analysis, using data-science and ML tools to analyze and make predictions, so that it has profit for business.
3. Having made your research and predictions, connect to Superset BI and create a useful dashboard for business.

**Goal of the project is to:**

1. Import csv file to PostgreSQL.
2. Divide mobile dataset into 2 tables, 1st mobile and 2nd brand with one-to-many connection.
3. Figure out the expense, business\_value of each mobile phone.
4. Search particular amount of user\_amount for brands.
5. Predict other user\_amout and replace null values using the most suitable algorithm and making algorithm better in performance.
6. With the found expense and predicted\_user\_amount, using price column calculate income, revenue and loss.
7. Create separate table mobile\_analysis for expense, business\_value and battery\_hour that we have and connect it to mobile table with one-to-one connection.
8. Create separate table brand\_analysis for predicted\_user\_amount, income, revenue and loss, after, connect it to brand table with one-to-one connection.
9. Create a dashboard with the final Data Base.

# **SQL - Data Base manipulation**

**CREATE TABLES**

*CREATE TABLE mobile (*

*id SERIAL PRIMARY KEY,*

*name VARCHAR,*

*brand VARCHAR,*

*model VARCHAR,*

*battery\_capacity\_mah DECIMAL,*

*screen\_size\_inches DECIMAL,*

*touchscreen VARCHAR,*

*resolution\_x DECIMAL,*

*resolution\_y DECIMAL,*

*processor DECIMAL,*

*ram\_mb DECIMAL,*

*internal\_storage\_gb DECIMAL,*

*rear\_camera DECIMAL,*

*front\_camera DECIMAL,*

*operating\_system VARCHAR,*

*wifi VARCHAR,*

*bluetooth VARCHAR,*

*gps VARCHAR,*

*number\_of\_sims VARCHAR,*

*three\_g VARCHAR,*

*four\_g\_lte VARCHAR,*

*price DECIMAL*

*);*

*CREATE TABLE mobile\_analysis (*

*id SERIAL PRIMARY KEY,*

*expense DECIMAL,*

*business\_value DECIMAL,*

*battery\_hour DECIMAL,*

*mobile\_id INTEGER UNIQUE,*

*CONSTRAINT fk\_mobile\_id FOREIGN KEY (mobile\_id) REFERENCES mobile (id) ON DELETE CASCADE*

*);*

*CREATE TABLE brand (*

*id SERIAL PRIMARY KEY,*

*name VARCHAR,*

*users\_amount DECIMAL*

*);*

*CREATE TABLE brand\_analysis (*

*id SERIAL PRIMARY KEY,*

*predicted\_users\_amount DECIMAL,*

*income DECIMAL,*

*revenue DECIMAL,*

*loss DECIMAL,*

*brand\_id INTEGER UNIQUE,*

*CONSTRAINT fk\_brand\_id FOREIGN KEY (brand\_id) REFERENCES brand (id) ON DELETE CASCADE*

*);*

**COPY CSV FILE TO POSTGRESQL TABLE**

*\COPY mobile FROM 'C:\Users\user\...\mobile.csv' WITH (*

*DELIMITER ',',*

*FORMAT csv,*

*HEADER true,*

*ENCODING 'UTF8'*

*);*

**ONE TO MANY CONNECTION OF MOBILE AND BRAND**

My task is to connect 2 tables – brand and mobile. I want to add brand\_id column as foreign key to mobile table. After compare name column of brand table with brand column of mobile and set mobile.brand\_id = brand.id where mobile.brand = brand.name then I can delete mobile.brand as I will have mobile.brand\_id.

1. *ALTER TABLE mobile*

*ADD COLUMN brand\_id INTEGER;*

1. *UPDATE mobile*

*SET brand\_id = brand.id FROM brand WHERE mobile.brand = brand.name;*

1. *ALTER TABLE mobile*

*ADD CONSTRAINT fk\_mobile\_brand FOREIGN KEY (brand\_id) REFERENCES brand(id) ON DELETE SET NULL;*

1. *ALTER TABLE mobile DROP COLUMN brand;*

Data for users\_amount was taken from google statistics websites.

# **Libraries and methods used in the project**

**Third-party libraries:**

1. pandas==2.2.3
2. numpy==2.0.2
3. matplotlib==3.9.2
4. seaborn==0.13.2
5. python-dotenv== 1.0.1
6. sqlalchemy==2.0.36
7. psycopg2==2.9.10
8. scikit-learn==1.5.2

**Built-in library:**

* os (not a third-party library)

**Methods:**

*pandas*

1. pd.read\_sql()
2. df.head()
3. df.columns
4. df.dtypes
5. df.drop()
6. df.describe()
7. df.groupby().mean()
8. df.isnull().sum()
9. df.apply()
10. df.corr()
11. df.index
12. df.mul()
13. df.min()
14. df.max()
15. df.loc()
16. df.iloc()
17. df.fillna(0).astype(float).round()
18. df.rename()
19. pd.NA
20. df.copy()
21. df.to\_sql()
22. pd.concat()
23. df.reset\_index()
24. pd.merge()
25. df.notna(), df.isna()
26. df.shape[0]
27. pd.DataFrame()
28. df.sort\_values()

*numpy*

1. np.where()
2. np.mean()
3. np.std()

*matplotlib*

1. plt.show()

*seaborn*

1. sns.heatmap()

*python-dotenv*

1. load\_dotenv()

*sqlalchemy*

1. create\_engine()

*psycopg2*

*scikit-learn*

1. from sklearn.model\_selection import train\_test\_split
2. from sklearn.preprocessing import MinMaxScaler
3. from sklearn.metrics import r2\_score, mean\_squared\_errorfrom sklearn.linear\_model import LinearRegression
4. from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
5. from sklearn.svm import SVR
6. from sklearn.neighbors import KNeighborsRegressor
7. from sklearn.tree import DecisionTreeRegressor
8. from sklearn.model\_selection import cross\_val\_score
9. from sklearn.model\_selection import RandomizedSearchCV

*os*

1. os.getenv()