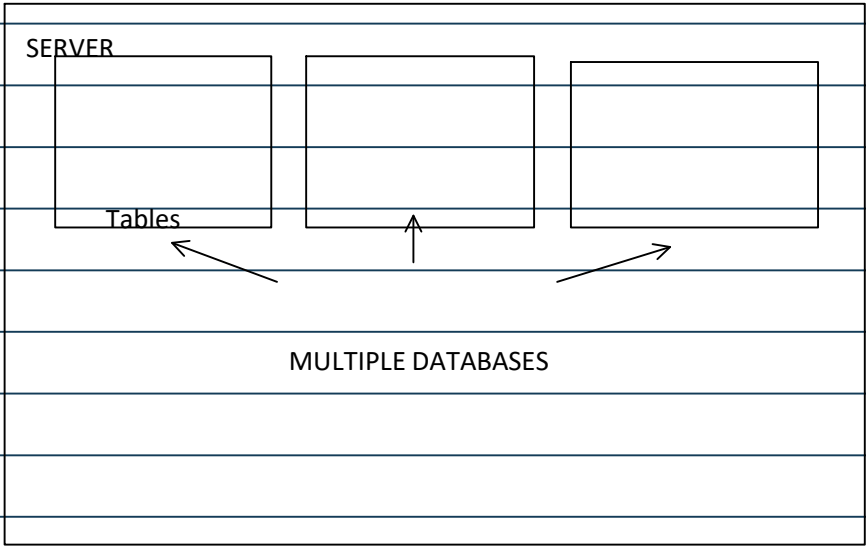


OLTP (Database)(Relational DB) -----PIPELINE-----> OLAP (Data Warehouse)(Column DB)



```
[83] import pymysql
      from sqlalchemy import create_engine

[82] ! pip install pymysql

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pymysql
  Downloading PyMySQL-1.0.2-py3-none-any.whl (43 kB)
    _____ 43.8/43.8 KB 2.8 MB/s eta 0:00:00
Installing collected packages: pymysql
Successfully installed pymysql-1.0.2

[84] mycursor = conn.cursor()

mycursor.execute('CREATE DATABASE dream11')
conn.commit()
```

LOADING PROCESS

```
[84] mycursor = conn.cursor()

[85] mycursor.execute('CREATE DATABASE dream11')
      conn.commit()

engine = create_engine("mysql+pymysql://admin:911Pentagon@database-olap.codzmntflx6t.ap-northeast-1.rds.amazonaws.com/dream11")
# {root}:{password}@{url}/{database}
export_df.to_sql('batter_points', con = engine)
```

pandas.read_sql_query

pandas.read_sql_query(sql, con, index_col=None, coerce_float=True, params=None, parse_dates=None, chunksize=None, dtype=None, dtype_backend=_NoDefault.no_default) [\[source\]](#)

Read SQL query into a DataFrame.

Returns a DataFrame corresponding to the result set of the query string. Optionally provide an *index_col* parameter to use one of the columns as the index, otherwise default integer index will be used.

pandas.DataFrame.to_sql

DataFrame.to_sql(name, con, *, schema=None, if_exists='fail', index=True, index_label=None, chunksize=None, dtype=None, method=None) [\[source\]](#)

Write records stored in a DataFrame to a SQL database.

Databases supported by SQLAlchemy [\[1\]](#) are supported. Tables can be newly created, appended to, or overwritten.

Using environment variables to securely code with usernames and passwords how?