Jan Pastucha 193662, Maksymilian Pek 193595

Scenarios of usage:

- Personalized workout recommendations – analyzing user workout patterns and recommend tailored workout plans
- Understanding membership and engagement – checking whether premium memberships correlate with higher engagement to enchance membership offerings
- Monitoring performance metrics – tracking user calories burned and workout intensity for personal trainers
- Seasonal trends in workout participation – identyifing trends in user engagement during different months to optimize class schedules and promotions

Explanation of decisions made

1. Types of data storing used
    a. Workout (ORC) - we used the ORC, because it is the best option for the fact table, which contains the most data in the database. It is highly compressed and has high storage efficiency, which allows for large-scale analytics. Columnar storage design also enables high-performance analytical queries.
    b. Date_data (Parquet) - we used the Parquet, because it supports efficient columnar storage, which is good for queries that filter, group, or join data on specific columns like year, month, or day. It ensures optimal compression and performance for analytics.
    c. Intensity (Parquet) - we used the Parquet here for the same reasons as in the date_data.

    d. Users (textfile) - we chose to use the simple textfile format for this table, because the table wouldn't be significantly impacted by lack of compression and indexing.
    e. Workout_type (textfile) - we used the textfile, because of the simplicity in storing arrays and maps. We didn't need the high compression nor index for this table, as we don't query that table too often.
    f. Location (Parquet) - we used the Parquet here for the same reasons as in the date_data and intensity tables.
2. Internal or external table
    a. Workout (internal) - we stored it as an internal table, because it is a core fact table – it is the most critical table in the schema.
    b. Date_data (internal) - we stored is as an internal table, because it includes only dates
    c. Intensity (internal) - we stored it as an internal table, because it contains workout-related data that is used in conjuction with the workout fact table.
    d. Users (internal) - we stored it as an internal table, because it contains personal data that is critical to the system's operations. By storing it as an internal table we ensure that the data is directly tied to the database schema.
    e. Workout_type (external) - we stored it as an external table, because it contains structured data that might be sourced or shared with external systems (like kilometers run from apps like Strava)
    f. Location (internal) - we stored it as an internal table, because it contains data that is directly tied to the internal structure of the database and does not require external managment.
3. Bucketing
    a. Year in date_data - the table is bucketed by year into 5 buckets to improve performance for queries grouped by or filtered by year.
4. Partitioning

a. Dynamic
    i. Level in intensity – we used dynamic partitioning in that column, because the number of different levels may change over time as new intensity categories are added (like very high, very low).
b. Static
    i. Gender in users – we decided to partition our data by gender to enchance query performance when filtering data by gender. The choice to use static partitioning instead of dynamic was due to fact that gender values are fixed – they will always be male or female

5. Complex types
    a. Array
        i. Equipment_used in workout_type – we made the equipment_used column as an array, because we wanted to store multiple equipments that user used during workout without having to clone the other data.
    b. Map
        i. Workout_details in workout_type – we made the workout_details column as a map, because it is designed to stored varied information about workouts, like sets, and reps done, or kilometers run.