Database name: Mobile App Store Database

Description:

The App Store hosts many apps for mobile users, ranging from games, self-help, education, or other types of entertainment. The database contains many details about each app in the platform, including but not limited to categories, average rating, number of reviews, and size.

This database makes use of **DynamoDB**, a managed database that makes it possible to create indexes and choose which keys to use among the many attributes of the apps. This database is created for three possible clients: the end users, app developers, and the developers behind the app store.

Table Schema:

Appstore Database Table Schema

Indexes:

Category_Index - Global Secondary Index with Category as the hash key and Rating as the range key.

Developer_Index - Global Secondary Index with Developer as the hash key and Rating as the range key.

Access Patterns:

Appstore developers

1. The app store developers might want to change its top 10 apps to display in their home page. As such, there is a need to find the apps with the highest average rating out of each category.
    - We make use of the Category_Index to query. Specifically, we match the HASH Key with the input category and then we sort the RANGE Key (Rating) descendingly.
2. The app store team wants to determine what card type is the most popular among the users to generate a report to their sponsors.
    - In order to do this, we have to scan the whole table with the matching card type and then we return the resulting count.

Users

3. A user might be worried that their device doesn't have enough storage space, so they want to know what apps in a certain category are less than or equal to a specific size.
    - We make use of the Category_Index to query. We first match the user's inputted category with the HASH Key and then set a filter for the Size attribute, making it less than or equal to the user's input.
4. A user wants to leave reviews for an app.
    - The user first has to determine the app's ID to submit a review. In order to do so, we scan the table with the matching app name.
    - Once the user knows what the app ID is, a function is defined so that the user will just fill in the details of his review.
5. A user wants to install or purchase apps.
    - We make use of two functions that update the table for the installed and purchased apps.
6. A user wants to leave comments to a review.
    - We first return the reviews of the app for the user to note the review ID.
    - When the user chooses the review id, they can then add comments.
7. A user wants to add a new payment method to their account.
    - We will make use of a function that will add the payment method.

App developers

8. An up-and-coming developer wants to create an app that would trend and garner popularity. They want to research features on apps that are doing good as well as determine the competitiveness from other developers through different categories.
    - In the developer's research, he wants first to determine what apps have a number of installs that are higher than their target value. We make use of the Category_Index here first, matching the input category with the HASH Key and then making use of a filter for the Installs attribute.
    - After determining the developers for each app of interest, they would want to look more into what other apps each developer has. Here, we can make use of the Developer_Index, matching the input developer with the HASH Key.
9. An app developer wants to update their app's description.
    - We will first match the app id with the HASH key and then update the App_Description attribute.

Device manufacturers

10. A phone manufacturer with their in-house Operating System wants to determine how many apps are compatible with their OS to provide support in future updates. They also want to check how many models manufacturers have that host their OS.
    - We first scan the table, matching the OS of the apps with the manufacturer's input. This scan will return the count of apps with the OS.
    - Next, we query so that we could check how many models an OS hosts of a particular manufacturer's devices. We match the input OS with the HASH key and then we filter the expression, equating the Manufacturer attribute to the input.

Reference: Google Play Store Apps | Kaggle