Project 2B - Databases

30 Possible Points

3/16/2022

Attempt 1

IN PROGRESS
Next Up: Submit Assignment

Add Comment 0

Unlimited Attempts Allowed

Details

Due: Wednesday, March 16
Points: 30 points
Deliverables: A ZIP file of your project
Grading: 1 point for each test passed (10 base tests + 20 instructor tests)
Resources:

- build.gradle (https://drive.google.com/file/d/1wzmUQfzkC7hsCW39z0ubvdZGVjlg_U7x/view?usp=sharing) (subject to minor tweaks)
- DreamRepository (https://drive.google.com/file/d/1kaNbd_YWYHNp2vUj2euQVunAIEN3_d5V/view?usp=sharing)
- BaseDreamCatcherListDetailTest (https://drive.google.com/file/d/1JadeB-0NwFjOrVLPwu5tG38yWF2jLOxW/view?usp=sharing)

Additional Code

In addition to the files provided above, please copy-paste the code below into appropriate Kotlin files. These are **annotated** model classes that you will use in your project.

Dream.kt

```
@Entity(tableName = "dream")
data class Dream(
    @PrimaryKey val id: UUID = UUID.randomUUID(),
    var title: String = "",
    var date: Date = Date(),
    var isFulfilled: Boolean = false,
    var isDeferred: Boolean = false
)
```

DreamEntry.kt

```
@Entity(tableName= "dream_entry",
```

Submit Assignment

```
    indices = [Index(name = "dream_entry_i1",  value = ["dreamId"])])
data class DreamEntry(
    @PrimaryKey val id: UUID = UUID.randomUUID(),
    val date: Date = Date(),
    val text: String = "",
    val kind: DreamEntryKind = DreamEntryKind.REFLECTION,
    val dreamId: UUID
)
```

DreamEntryKind.kt

```
enum class DreamEntryKind {
    CONCEIVED, DEFERRED, FULFILLED, REFLECTION
}
```

DreamWithEntries.kt

```
data class DreamWithEntries(
    @Embedded var dream: Dream,
    @Relation(
        parentColumn = "id",
        entityColumn = "dreamId"
    ) var dreamEntries: List<DreamEntry>
)
```

You will also need a DAO class as was given in criminal intent. Since the DreamWithEntries DAOs are a bit complicated, we have given them below. However, please note that you must implement the DAOs for Dream and DreamEntry on your own.

```
@Dao
interface DreamDao {

    // ----------------------------------------------------------
    // Dream functions
    // ----------------------------------------------------------

    // TODO Implement getters / add / and update methods for Dream

    // ----------------------------------------------------------
    // DreamEntry functions
    // ----------------------------------------------------------

    // TODO Implement getter / delete / add / and update methods for DreamEntry

    // ----------------------------------------------------------
    // DreamWithEntries functions
    // ----------------------------------------------------------

    @Query("SELECT * FROM dream WHERE id=(:dreamId)")
    fun getDreamWithEntries(dreamId: UUID): LiveData<DreamWithEntries>

    @Transaction
    fun updateDreamWithEntries(dreamWithEntries: DreamWithEntries) {
        val theDream = dreamWithEntries.dream
        val theEntries = dreamWithEntries.dreamEntries
        updateDream(dreamWithEntries.dream)
        deleteDreamEntries(theDream.id)
```

Submit Assignment

```
    fun addDreamWithEntries(dreamWithEntries: DreamWithEntries) {
        addDream(dreamWithEntries.dream)
        dreamWithEntries.dreamEntries.forEach { e -> addDreamEntry(e) }
    }

    // ----------------------------------------------------------
    // Clear Dream and Entries (used for testing purposes)
    // ----------------------------------------------------------

    @Query("DELETE FROM dream")
    fun deleteAllDreamsInDatabase()

    @Query("DELETE FROM dream_entry")
    fun deleteAllDreamEntriesInDatabase()

}
```

The annotations are used by the Room component library to help create a database. These model classes are more complex than those you see in Chapter 11 of BNR. That's because we have two tables in our project instead of one, and there is a many-to-one relationship between them. **Please do not modify any code we give to you unless we explicitly tell you that you can.**

Overview

This project extends Project 2A by adding a database. In fact, the base test we give you for this project are exactly the same as those we gave you for Project 2A. However, please make sure you save an independent copy of Project 2A before you start this one. You should save copies of all your projects in case we have problems grading them. For Project 2B, you will need to complete reading Chapters 11 and 12 of BNR.

Choose a submission type

Upload

More

Canvas Files

Choose a file to upload

File permitted: ZIP

Submit Assignment

Submit Assignment