Synchro entre la base locale et l'API

Dans le fichier pubspec.yaml verifie que les dépendances sont à jour

```
dependencies:
  sqflite: ^2.4.1
  path_provider: ^2.0.2
```

Votre fichier dao.dart doit ressembler à cela :

```
import 'package:sqflite/sqflite.dart';
import 'package:path/path.dart';
import 'dart:async';

class DatabaseHelper {
  static Database? _database;

  // Accès à la base de données SQLite
  Future<Database> get database async {
    if (_database != null) return _database!;
    _database = await _initDB();
    return _database!;
  }

  // Initialisation de la base de données SQLite
  _initDB() async {
    var databasesPath = await getDatabasesPath();
    String path = join(databasesPath, 'app_database.db');
    return await openDatabase(path, version: 1, onCreate: _onCreate);
  }

  // Création des tables
  _onCreate(Database db, int version) async {
    await db.execute('''
    CREATE TABLE categorie (
      id INTEGER PRIMARY KEY AUTOINCREMENT,
      libelle TEXT NOT NULL
    )
    ''');
    await db.execute('''
    CREATE TABLE auteur (
      id INTEGER PRIMARY KEY AUTOINCREMENT,
      nom TEXT NOT NULL,
      prenoms TEXT NOT NULL,
      email TEXT
    )
    ''');
    await db.execute('''
    CREATE TABLE livre (
      id INTEGER PRIMARY KEY AUTOINCREMENT,
      libelle TEXT NOT NULL,
      description TEXT,
      auteur_id INTEGER NOT NULL,
      categorie_id INTEGER NOT NULL
    )
    ''');
  }
```

```
  // Insérer une catégorie dans la base de données locale
  Future<void> insertCategorie(Map<String, dynamic> categorie) async {
    final db = await database;
    await db.insert('categorie', categorie, conflictAlgorithm:
ConflictAlgorithm.replace);
  }

  // Récupérer toutes les catégories
  Future<List<Map<String, dynamic>>> getCategories() async {
    final db = await database;
    return db.query('categorie');
  }

  // Insérer un auteur
  Future<void> insertAuteur(Map<String, dynamic> auteur) async {
    final db = await database;
    await db.insert('auteur', auteur, conflictAlgorithm:
ConflictAlgorithm.replace);
  }

  // Récupérer tous les auteurs
  Future<List<Map<String, dynamic>>> getAuteurs() async {
    final db = await database;
    return db.query('auteur');
  }

  // Insérer un livre
  Future<void> insertLivre(Map<String, dynamic> livre) async {
    final db = await database;
    await db.insert('livre', livre, conflictAlgorithm: ConflictAlgorithm.replace);
  }

  // Récupérer tous les livres
  Future<List<Map<String, dynamic>>> getLivres() async {
    final db = await database;
    return db.query('livre');
  }
}
```

Synchronisation Manuelle

La version2.4.1 de sqflite, permet de synchroniser manuellement les données entre la base locale et l'API Spring Boot via des appels HTTP.

Exemple de fonction de synchronisation pour récupérer les catégories depuis l'API et les insérer dans la base de données locale :

```
import 'package:http/http.dart' as http;
import 'dart:convert';

// Synchronisation des catégories entre l'API et la base locale
Future<void> syncCategoriesWithAPI() async {
  final db = await DatabaseHelper().database;

  // 1. Récupérer les catégories depuis l'API
  var url = Uri.parse('http://localhost:8080/api/categorie');
```

```
  var response = await http.get(url);

  if (response.statusCode == 200) {
    List<dynamic> categories = json.decode(response.body);

    // 2. Insérer les catégories récupérées dans la base locale
    for (var categorie in categories) {
      await db.insert('categorie', {
        'id': categorie['id'],
        'libelle': categorie['libelle'],
      }, conflictAlgorithm: ConflictAlgorithm.replace);
    }
  } else {
    print('Erreur lors de la récupération des catégories');
  }
}

// Synchronisation des catégories locales avec l'API
Future<void> pushCategoriesToAPI() async {
  final db = await DatabaseHelper().database;

  // 1. Récupérer les catégories depuis la base locale
  List<Map<String, dynamic>> categories = await db.query('categorie');

  // 2. Pour chaque catégorie locale, envoyer une requête POST vers l'API
  for (var categorie in categories) {
    var url = Uri.parse('http://localhost:8080/api/categorie');
    var response = await http.post(
      url,
      headers: {"Content-Type": "application/json"},
      body: json.encode({
        'id': categorie['id'],
        'libelle': categorie['libelle'],
      }),
    );
    if (response.statusCode == 200) {
      print('Catégorie synchronisée');
    } else {
      print('Erreur lors de la synchronisation de la catégorie');
    }
  }
}
```

Synchronisation Automatique avec un Timer

Il est possible de mettre en place une synchronisation automatique en utilisant un Timer pour appeler régulièrement les fonctions de synchronisation.

Ci-dessous un exemple pour démarrer une synchronisation automatique toutes les 15 minutes :

```
import 'dart:async';

Timer? syncTimer;

void startAutoSync() {
  syncTimer = Timer.periodic(Duration(minutes: 15), (timer) {
    syncCategoriesWithAPI();  // Appeler la fonction de synchronisation ici
```

```
    pushCategoriesToAPI();    // Si tu veux pousser les données locales vers l'API
  });
}

void stopAutoSync() {
  syncTimer?.cancel();
}
```

Gestion des conflits

Lors de la synchronisation bidirectionnelle, il est impératif de gérer les conflits entre les données locales et celles de l'API. Voici quelques stratégies possibles :

- **Timestamps** : Utiliser un champ timestamp pour chaque entrée, ce qui permet de savoir quelle version des données est la plus récente.
- **Flags de synchronisation** : Ajouter un champ is_synced pour marquer si une donnée a été synchronisée avec l'API.