**Документация к функции createOrder**

Функция createOrder предназначена для добавления нового заказа на покупку билетов в базу данных. Включает проверку уникальности события, уникальности штрихкода и бронирование заказа через внешний API (с использованием моков). Если бронирование и подтверждение успешны, заказ сохраняется в базе данных.

**Параметры функции**

* **$event\_id** (int) — уникальный идентификатор события.
* **$event\_date** (string) — дата и время события в формате Y-m-d H:i:s.
* **$ticket\_adult\_price** (int) — цена билета для взрослого на момент покупки.
* **$ticket\_adult\_quantity** (int) — количество взрослых билетов.
* **$ticket\_kid\_price** (int) — цена детского билета.
* **$ticket\_kid\_quantity** (int) — количество детских билетов.

**Процесс выполнения**

1. **Подключение к базе данных**:
   * Устанавливается соединение с MySQL.
   * В случае ошибки подключение завершается.
2. **Проверка на уникальность event\_id**:
   * Функция проверяет, существует ли уже заказ с таким event\_id.
   * Если заказ существует, возвращается сообщение об ошибке без добавления нового заказа.
3. **Генерация уникального штрих-кода (barcode)**:
   * Генерируется случайное число от 10000000 до 99999999.
   * Проверяется уникальность barcode в базе данных, чтобы избежать дублирования.
   * В случае дублирования генерация повторяется, пока не будет найден уникальный штрих-код.
4. **Попытка бронирования через мок API**:
   * Отправляется запрос на бронирование с использованием функции mockBookingAPI.
   * Если ответ API содержит ошибку barcode already exists, генерируется новый barcode.
   * Если бронирование завершилось ошибкой (кроме barcode already exists), функция возвращает сообщение об ошибке.
   * Если сгенерировался уникальный штрихкод, сохраняем его в переменную и идём дальше.
5. **Подтверждение бронирования через мок API**:
   * Если бронирование успешно, отправляется запрос на подтверждение с помощью mockApprovalAPI.
   * В случае успешного подтверждения рассчитывается общая стоимость заказа, после чего он сохраняется в базе данных.
6. **Сохранение заказа в базу данных**:
   * Данные заказа сохраняются в таблице nevatrip\_order с использованием подготовленного запроса.
   * Вставляется информация о event\_id, event\_date, ticket\_adult\_price, ticket\_adult\_quantity, ticket\_kid\_price, ticket\_kid\_quantity, barcode, equal\_price, created (текущая дата и время).
   * Если все шаги выполнены, функция возвращает строку с сообщением об успешном создании заказа и его штрихкодом.

**Возвращаемое значение**

* **string** — сообщение о результатах выполнения:
  + Успех: "Заказ успешно создан с штрихкодом: [barcode]"
  + Ошибка: "Ошибка при бронировании: ..." или "Ошибка подтверждения заказа: ...".

**Описание вспомогательных функций**

**generateUniqueBarcode**

Функция для генерации уникального штрих-кода. Возвращает строку с barcode, которая гарантированно уникальна в базе данных.

**mockBookingAPI**

Эмулятор внешнего API для бронирования заказа. Принимает массив данных и возвращает случайный ответ:

* Успешный ответ: ['message' => 'order successfully booked']
* Ошибки: ['error' => 'barcode already exists']

**mockApprovalAPI**

Эмулятор внешнего API для подтверждения заказа. Принимает массив данных и возвращает случайный ответ:

* Успешный ответ: ['message' => 'order successfully approved']
* Возможные ошибки: ['error' => 'event cancelled'], ['error' => 'no tickets'], ['error' => 'no seats'], ['error' => 'fan removed']

**Документация к структуре базы данных для заказов и билетов**

Эта структура базы данных разделяет информацию о заказах и билетах на две связанные таблицы: orders и tickets. Она поддерживает нормализацию данных, позволяя каждому билету иметь уникальный штрих-код и различные типы билетов для одного заказа.

**Таблица orders**

Таблица orders хранит общую информацию о каждом заказе.

* **id** (INT, AUTO\_INCREMENT, PRIMARY KEY): Уникальный идентификатор заказа.
* **event\_id** (INT, NOT NULL): Уникальный идентификатор события, на которое был сделан заказ.
* **event\_date** (DATETIME, NOT NULL): Дата и время, когда состоится событие.
* **created** (DATETIME, DEFAULT CURRENT\_TIMESTAMP): Дата и время создания заказа.

**Назначение**

Таблица orders используется для хранения базовых данных о заказе, включая идентификатор события, дату события и время создания заказа. Она обеспечивает уникальность каждого заказа и связывает его с конкретным событием.

**Таблица tickets**

Таблица tickets содержит информацию о каждом отдельном билете, связанном с определенным заказом.

* **id** (INT, AUTO\_INCREMENT, PRIMARY KEY): Уникальный идентификатор билета.
* **order\_id** (INT, NOT NULL): Внешний ключ, ссылающийся на id в таблице orders, указывающий на заказ, к которому относится билет.
* **ticket\_type** (ENUM('adult', 'kid', 'discount', 'group'), NOT NULL): Тип билета, который может быть взрослым (adult), детским (kid), льготным (discount), или групповым (group). Дополнительные типы билетов могут быть добавлены в будущем.
* **price** (INT, NOT NULL): Цена билета на момент заказа.
* **barcode** (VARCHAR(120), UNIQUE, NOT NULL): Уникальный штрих-код для каждого отдельного билета, обеспечивающий идентификацию при проверке билетов.

**Назначение**

Таблица tickets содержит детализированную информацию о каждом билете. Она поддерживает разные типы билетов для одного заказа, позволяя хранить и управлять ими по отдельности. Уникальность штрих-кода barcode гарантирует, что каждый билет идентифицируется и проверяется отдельно.

**Связи**

* **Внешний ключ order\_id** связывает tickets с orders, устанавливая отношение "один ко многим" между заказом и билетами, то есть один заказ может содержать несколько билетов.

**Примеры использования**

1. **Создание нового заказа** При создании заказа данные записываются в таблицу orders с указанием event\_id и event\_date.
2. **Добавление билетов к заказу** Для каждого билета, входящего в заказ, создается отдельная запись в таблице tickets с указанием order\_id, типа билета ticket\_type, его цены и уникального штрих-кода.
3. **Проверка билетов** Каждый билет в таблице tickets имеет уникальный штрих-код barcode, который может быть использован для проверки и допуска на мероприятие.