Условие (входные данные)

Используется учебный набор данных объёмом **100 000 – 300 000 строк**, который **создаётся автоматически** средствами PostgreSQL.

Модель данных — журнал заказов интернет-магазина.

Поля таблицы `orders`:

- `order_id` — идентификатор заказа
- `order_date` — дата и время заказа
- `customer_id` — идентификатор клиента
- `product_id` — идентификатор товара
- `quantity` — количество товара
- `price` — цена за единицу
- `region` — регион покупателя

Задание (шаги)

Шаг 1. Создание таблицы

1. Создайте таблицу `orders`.
2. Выберите корректные типы данных.
3. Назначьте первичный ключ.

Шаг 2. Генерация и загрузка данных

1. Сгенерируйте **не менее 100 000 строк** с использованием:
   - `generate_series`,
   - встроенных функций PostgreSQL (`random`, `now()`).
2. Заполните таблицу случайными, но логически корректными данными.

Шаг 3. Базовый анализ данных

Напишите SQL-запросы для получения:

1. Общего количества заказов.
2. Общей выручки.
3. Топ-5 товаров по суммарной выручке.
4. Количества заказов по регионам.

Шаг 4. Индексация и оптимизация

1. Создайте индекс по `order_date`.
2. Создайте составной индекс по (`product_id`, `order_date`).
3. Повторно выполните запрос из пункта **3.3** и сравните:
   - время выполнения;
   - план запроса.

Шаг 5. Анализ производительности

1. Используйте `EXPLAIN ANALYZE` для одного агрегатного запроса.
2. Ответьте письменно:
   - используется ли индекс;
   - почему план запроса изменился или не изменился.

Подсказки по ключевым частям

- `generate_series(1, 100000)` — удобный способ генерации данных.
- Для дат используйте `TIMESTAMP`.
- Для анализа используйте `EXPLAIN ANALYZE`, а не просто `EXPLAIN`.
- Индексы ускоряют чтение, но замедляют вставку.
- Агрегации по большим таблицам часто используют `Seq Scan`.

Что проверить перед отправкой (чек-лист)

- [ ] Таблица `orders` создана
- [ ] В таблице ≥ 100 000 строк
- [ ] Все SQL-запросы выполняются без ошибок
- [ ] Индексы созданы корректно
- [ ] Приведён вывод `EXPLAIN ANALYZE`

- [ ] Даны текстовые выводы по оптимизации

Советы по улучшению работы

- Не используйте `SELECT *` в аналитических запросах.
- Проверяйте, какие индексы реально используются.
- Сравнивайте запросы **до и после** индексации.
- Используйте `ORDER BY` только при необходимости.
- Делайте краткие, но технически точные выводы.