Министерство образования и науки РФ

Санкт-Петербургский Политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа программной инженерии

ОТЧЕТ

по лабораторной работе вариант №3

Проектирование аналитической схемы базы данных

по дисциплине «Системы управления базы данных»

|  |  |
| --- | --- |
| Выполнил  студент гр. 5130904/10104  Преподаватель | Баженов Кирилл Максимович.  Прокофьев О. В. |

Санкт-Петербург

2024

**Практическое задание №3.1 Проектирование аналитической схемы**

**базы данных**

**Постановка задачи:**

Седьмое практическое задание связано с проектированием схемы базы данных для аналитики.

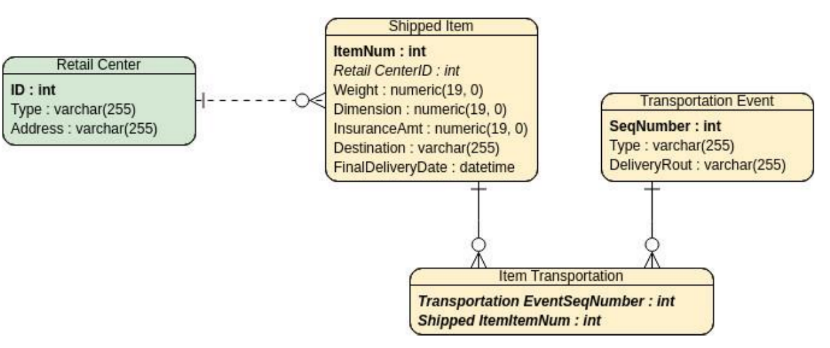
Будем исходить из того, что приложение, для которого была сделана база данных в задании стала

очень популярной и по ней каждый день можно собирать большой объем статистической

информации. Результатом данного практического задания являются: скрипты создания базы

данных, хранимая процедура (генератор) для ее заполнения, анализ плана выполнения запроса.

Пример используемой ER диаграммы (система доставки UPS)



Требования к БД

* Одна таблица (Item transportation) должна содержать не меньше 10 млн. записей, которые со временем теряют актуальность.
* Другая таблица (Shipped Item), связанная с первой, должна содержать не меньше 1 млн.

записей.

* Остальные таблицы должны содержать по 5-10 записей.
* В одной из таблиц с количеством записей 1 млн. должна быть колонка с текстом, по которой будет необходимо настроить полнотекстовый поиск.

Практическая часть включает:

1) наполнение таблицы, для этого нужно написать хранимую процедуру - генератор на языке

plpython3u, которая использует словари (для строковых типов), случайные значения (для

строковых, числовых типов).

2) оценку скорости выполнения запросов.

Для этого могут быть использованы механизмы секционирования, наследования и индексов.

Необходимо подготовить два запроса:

* Запрос к одной таблице, содержащий фильтрацию по нескольким полям.
* Запрос к нескольким связанным таблицам, содержащий фильтрацию по нескольким полям.

3) оценка скорости полнотекстового поиска (использование функций to\_tsvector to\_tsquery)

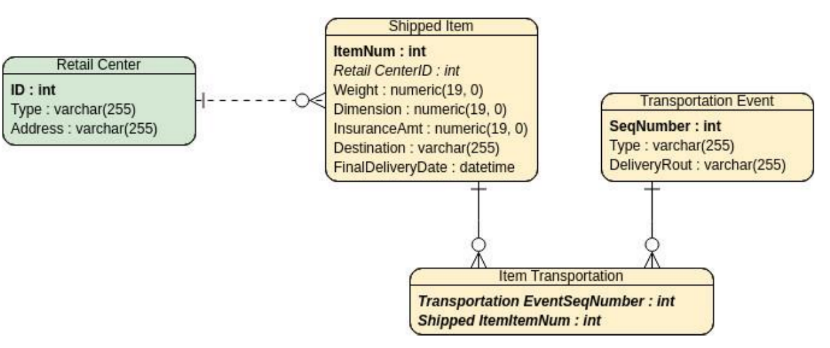
Для каждого из этих запросов необходимо провести следующие шаги:

* Получить план выполнения запроса без использования индексов (удаление индекса или

отключение его использования в плане запроса).

* Получить статистику (IO и Time) выполнения запроса без использования индексов.
* Создать нужные индексы, позволяющие ускорить запрос.
* Получить план выполнения запроса с использованием индексов и сравнить с первоначальным планом.
* Получить статистику выполнения запроса с использованием индексов и сравнить с первоначальной статистикой.
* Оценить эффективность выполнения оптимизированного запроса.

**Выбранная ER диаграмма**

****

**Скрипт DDL аналитической БД**

create table retail\_center (  
 id serial primary key,  
 type text,  
 address text  
);  
  
create table shipped\_item(  
 item\_num int primary key,  
 retail\_center\_id int references retail\_center(id),  
 weight numeric(19, 0),  
 dimension numeric(19, 0),  
 insurance\_amt numeric(19, 0),  
 destination text,  
 final\_delivery\_date timestamp  
);  
  
create table transportation\_event (  
 seq\_number int primary key,  
 type text,  
 delivery\_rout text  
);  
  
create table item\_transportation(  
 transportation\_event\_seq\_number int references transportation\_event(seq\_number),  
 shipped\_item\_item\_num int references shipped\_item(item\_num)  
);

**Скрипт для заполнения баз случайными данными**

CREATE EXTENSION IF NOT EXISTS plpython3u;do$$from datetime import datetime, timezoneimport randomimport stringimport decimaldef rand\_str(length): # choose from all lowercase letter letters = string.ascii\_lowercase result\_str = ''.join(random.choice(letters) for i in range(length)) return result\_strdef rt\_data(rows): data = [] for i in range (rows): data.append((rand\_str(5), rand\_str(5))) return datadef tr\_data(rows): data = [] for i in range (rows): data.append((str(i+1), rand\_str(5), rand\_str(5))) return datadef sp\_data(rows): data = [] retail\_ids = [row["id"] for row in plpy.execute("SELECT id FROM retail\_center")] event\_ids = [row["seq\_number"] for row in plpy.execute("SELECT seq\_number FROM transportation\_event")] for i in range (rows): data.append((str(i+1), str(random.choice(retail\_ids)), str(decimal.Decimal(random.randrange(155, 389))/100), str(random.uniform(1.25, 100.25)), str(random.uniform(1.25, 100.25)), rand\_str(random.randint(2,15)), str(datetime.now(timezone.utc)))) return datadef item\_data(rows): data = [] item\_ids = [row["item\_num"] for row in plpy.execute("SELECT item\_num from shipped\_item")] event\_ids = [row["seq\_number"] for row in plpy.execute("SELECT seq\_number FROM transportation\_event")] for i in range (rows): data.append((str(random.choice(event\_ids)), str(random.choice(item\_ids)))) return datadef write\_to\_csv(random\_data, csv\_name): with open(csv\_name, 'w+') as f: for record in random\_data: f.write(','.join(record) + '\n')rt\_data = rt\_data(10)write\_to\_csv(rt\_data, 'retail.csv')plpy.execute("COPY retail\_center(type, address) FROM 'retail.csv' CSV")tr\_data = tr\_data(10)write\_to\_csv(tr\_data, 'transport.csv')plpy.execute("COPY transportation\_event(seq\_number, type, delivery\_rout) FROM 'transport.csv' CSV")s\_data = sp\_data(1000000)write\_to\_csv(s\_data, 'shipped.csv')plpy.execute("COPY shipped\_item(item\_num, retail\_center\_id, weight, dimension,"+ "insurance\_amt, destination, final\_delivery\_date) FROM 'shipped.csv' CSV")it = item\_data(5000000)write\_to\_csv(it, 'item.csv')plpy.execute("COPY item\_transportation(transportation\_event\_seq\_number,shipped\_item\_item\_num) FROM 'item.csv' CSV")$$ language plpython3u

**Анализ выполнения запросов**

**1) Запрос к одной таблице, содержащей фильтрацию по нескольким полям**

* Получить план выполнения запроса без использования индексов (удаление индекса или

отключение его использования в плане запроса).

Seq Scan on shipped\_item (cost=0.00..23849.00 rows=246694 width=28) (actual time=0.017..362.852 rows=245527 loops=1)

Filter: ((dimension < '50'::numeric) AND (insurance\_amt > '50'::numeric))

Rows Removed by Filter: 754473

* Получить статистику (IO и Time) выполнения запроса без использования индексов.

Buffers: shared hit=8849

Planning Time: 0.207 ms

Execution Time: 443.729 ms

* Создать нужные индексы, позволяющие ускорить запрос.