# Автоматизация работы оптовой фирмы.

В рамках данного цикла лабораторных работ необходимо автоматизировать работу оптовой фирмы. Для этого в рамках базы данных PostgreSQL необходимо создать объекты в схеме вашего пользователя (логин и пароль пользователя для доступа к базе данных студент должен получить у преподавателя) и написать клиентское приложение на базе компонентов ADO.NET или JDBC.

Оптовая фирма (далее организация) занимается оптовыми и мелкооптовыми поставками различных товаров в магазины Санкт-Петербурга. Требуется автоматизировать рабочее место менеджера по распределению товаров между двумя складами организации. Два склада несут функционально различную нагрузку в целях уменьшения арендной платы организации. Первый склад находится на территории Санкт-Петербурга и сильно меньше второго, который находится в значительном удалении. Первый склад исполняет функции КЭШа множества самых популярных товаров. Задачей менеджера по распределению является изучение спроса и выделения множества приоритетных товаров, которые будут завезены на первый склад. Менеджер должен иметь доступ к состоянию обоих складов, справочнику товаров и журналу реализации. При появлении заявки на определенный товар, менеджер заносит заявку в журнал и списывает необходимое количество товара с первого склада. В случае если на первом складе товара не достаточно – остаток списывается со второго. Такой вариант приносит лишние расходы фирме, так как в этом случае она занимается доставкой товара на территорию заказчика, чтобы не потерять клиента. Завоз товаров на первый склад производится ночью каждого дня в соответствии с приоритетами выставленными менеджером. Второй склад содержит достаточное число товаров, и его работа выходит за рамки данного проекта.

**База данных должна удовлетворять следующим требованиям:**

1. Контроль целостности данных, используя механизм связей
2. Операции модификации групп данных и данных в связанных таблицах должны быть выполнены в рамках транзакций.
3. Логика работы приложения должна контролироваться триггерами. В частности:
   * Триггер должен не позволять списать товар со второго склада при наличии товара на первом
   * Триггер должен контролировать, чтобы вводимая заявка не превышала суммарное количество товара на первом и втором складах
4. Все операции вычисления различных показателей (из требований к клиентскому приложению) должны реализовываться хранимыми процедурами.

**Требования к клиентскому приложению:**

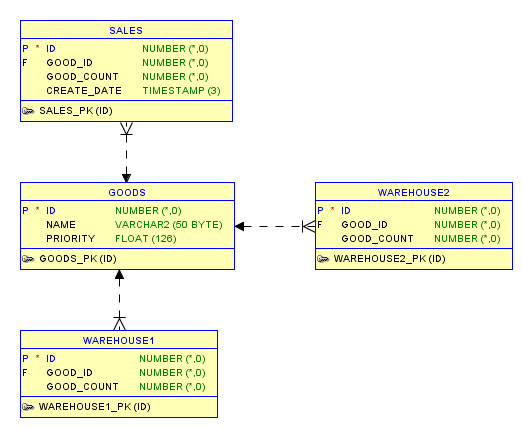
1. Необходимо реализовать интерфейсы для ввода, модификации и удаления

* Товаров, включая задание приоритета
* Заявки, с автоматическим списанием товара со складов

1. В главном окне приложения должен быть реализован журнал менеджера с просмотром количества товаров на складах.
2. Необходимо реализовать возможность просмотра менеджером следующих показателей:

* Пять самых популярных товаров.
* Изменение спроса данного товара за некоторый промежуток времени.
* Графическое отображение изменения спроса заданного товара

## Работа №1. Создание базы данных.



|  |  |  |
| --- | --- | --- |
| **Имя таблицы** | **Имя колонки** | **Расшифровка** |
| sales |  | Таблица заявок |
|  | id | Идентификатор заявки |
|  | good\_id | Идентификатор товара |
|  | good\_count | Количество товара |
|  | create\_date | Дата заявки |
| goods |  | Таблица товаров |
|  | id | Идентификатор товара |
|  | name | Наименование товара |
|  | priority | Приоритет товара |
| warehouse1, warehouse2 |  | Склад1, Склад2 |
|  | id | Идентификатор записи |
|  | good\_id | Идентификатор товара |
|  | good\_count | Количество товара на складе |

1. Запустить pgAdmin или psql.
2. Создать соединение, используя логин и пароль.
3. Изменить пароль вашего пользователя на свой, используя команду

ALTER USER *your*\_*user\_name* IDENTIFIED BY *new\_password*;

1. Создать необходимые таблицы.
2. Создать связи между таблицами:

|  |  |  |
| --- | --- | --- |
| **Название** | **Primary Key** | **Foreign Key** |
| fk\_sales\_goods | goods.id | sales.good\_id |
| fk\_warehouse1\_goods | goods.id | warehouse1.good\_id |
| fk\_warehouse2\_goods | goods.id | warehouse2.good\_id |

1. Создать Backup базы и запомнить место его расположения.
2. Удалить базу с сервера
3. Восстановить базу из Backup базы
4. Для всех таблиц реализовать автоматическое заполнение первичного ключа при вставке данных (автоинкремент).

### Правила выполнения работ

1. Засчитываются только целиком сделанные работы
2. Преподаватель оставляет за собой право на выдачу дополнительных заданий, в случае возникновения подозрения на стороннюю помощь при выполнении работы.
3. Посещения занятий ОБЯЗАТЕЛЬНО, кроме случаев, оговоренных лично с преподавателем.
4. Если студент сдает работу, то он должен знать, как он ее сделал.
5. Все работы можно делать дома, но это не является основанием для пропусков занятий.
6. Перед уходом студент должен удалить с сервера свою базу. Рекомендую не забывать делать Backup в осознанное место, что бы на следующем занятии не пришлось делать работу с начала.
7. К каждому заданию прилагается список операторов, ключевых слов.

## Работа №2. Язык SQL-DML

### Выборка данных

* однотабличная выборка
  1. Вывести товары, упорядочив в алфавитном порядке по наименованию и в обратном порядке по приоритету
  2. Посчитать количество товаров в заявках за заданную дату
* выборка с подзапросами
  1. Вывести все заявки на товары с приоритетом меньше заданного
  2. Вывести наименования на товары, отсутствующие на «складе 1»
* соединение таблиц (join)
  1. Вывести заявки и наименования товаров, включая товары, на которые не было заявок
  2. Вывести количество товаров, на которые не было заявок за заданную дату
* для реализации проекта
  1. Вывести суммарное количество заданного товара на первом и втором складах.
  2. Пять самых популярных товаров за заданный промежуток времени, упорядочив в порядке уменьшения спроса.

Hints: select, count, join, where, in, exists, order by, group by, having

### Вставка данных

* однотабличная вставка
  1. Добавить новый товар
  2. Добавить заявку на товар из п1
* многотабличная вставка в рамках транзакции
  1. Добавить в рамках транзакции новый товар и некоторое количество его завести на «склад 2» и «склад 1»
  2. Добавить в рамках транзакции новый товар и некоторое количество его завести на «склад 2» и «склад 1». В случае, если суммарное число товаров на складе 1 превышает некоторую заданную величину – транзакцию откатить.

Hints: insert, where, in, exists, commit, rollback

### Удаление данных

* удаление по фильтру и удаление из связанных таблиц
  1. Удалить заявки, у которых дата меньше заданной
  2. Удалить товары, не имеющие заявок
  3. Удалить товар и заявки на него
* удаление в рамках транзакции
  1. Удалить в рамках транзакции товар со «склада 1» с наименьшим приоритетом
  2. то же, что и п1, но транзакцию откатить

Hints: delete, where, in, exists, commit, rollback

### Модификация данных

* модификация по фильтру
  1. Изменить количество товаров с заданным наименованием
  2. Списать 5 единиц заданного товара со «склада 1»
  3. Списать 5 единиц заданного товара со «склада 1». Если количество товара на этом складе меньше 5, не достающее число списать со второго склада.
* модификация в рамках транзакции
  1. В рамках транзакции поменять заданный товар во всех заявках на другой и удалить его.
  2. то же, что и п1, но транзакцию откатить

Hints: update, where, in, exists, commit, rollback

### Дополнительные задания

## Работа №3. Представления, хр. процедуры, триггера и курсоры

### Представления

1. Создать представление, отображающее все товары, число которых на складе1 менее некоторого числа
2. Создать представление, отображающее 5 самых популярных товаров за заданный месяц.

Hints: select, where, count, max, group by, having, like, create view, drop view

### Хранимые процедуры

* без параметров

1. Создать хранимую процедуру, выводящую список товаров для перевоза и его количество согласно текущему состоянию приоритетов.

* с входными параметрами

1. Создать хранимую процедуру с параметром количество перевозимого товара за ближайший рейс и выводящую все товары, которые необходимо привезти, и их количество.
2. Создать хранимую процедуру, имеющую два параметра «товар1» и «товар2». Она должна возвращать даты, спрос в которые на «товар1» был больше чем на «товар2». Если в какой-либо день один из товаров не продавался, такой день не рассматривается.

* с выходными параметрами

1. Создать хранимую процедуру с входными параметрами, задающими интервал времени, и выходным – идентификатором товара. Процедура должна возвращать товар с максимальным приростом спроса.
2. Создать хранимую процедуру с входными параметрами, задающими интервал времени, и выходным параметром – товар, с минимальным числом продаж за заданный период времени.

Hints: select, where, count, max, group by, having, create procedure, drop procedure

### Триггера

* Триггера на вставку

1. Создать триггер, который не позволяет добавить заявку на товар, число которого на обоих складах меньше указанного в заявке
2. Создать триггер, который не позволяет добавить заявку c числом товара меньше 1.

* Триггера на модификацию

1. Создать триггер, который не позволяет уменьшить число товара на складе 2 при наличии этого товара на складе 1.

* Триггера на удаление

1. Создать триггер, который при удалении товара в случае наличия на него ссылок откатывает транзакцию

Hints: select, where, in, exists, join, commit, rollback, create trigger, drop trigger

### Курсоры

* Хранимая процедура для прогноза спроса на заданный товар

Необходимо реализовать хранимую процедуру, рассчитывающую прогнозируемый спрос на 7 дней на некоторый товар. Хранимая процедура должна иметь два входных параметра задающие интервал времени для анализа изменения спроса и параметр, задающий анализируемый товар.

Предлагаемый алгоритм: с помощью курсора формируем временную таблицу, содержащую номер дня в рассматриваемом интервале и число хранящегося товара. Максимальный номер сохраняем в переменной.

Организуем курсор, перебирающий последовательно строки временной таблицы, упорядоченные по номеру точки, и усредняющий попарно соседние точки (за каждую итерацию точек становится на 1 меньше). Полученное среднее значение каждой пары, заменяет значение минимальной по номеру точки. Последняя точка удаляется.

Работа курсора повторяется до тех пор, пока в таблице не останется 2 точки. Будем считать, что разница спроса в этих двух точках равна разнице спроса между прогнозируемым на следующий день спросом и спросом на заданный товар последнего дня. Таким образом, мы можем получить величину прогнозируемого спроса. (экстраполяция методом скользящего среднего)

Hints: CURSOR, %NOTFOUND, FETCH