

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
К ЛАБОРАТОРНЫМ РАБОТАМ ПО ДИСЦИПЛИНЕ
"ОРГАНИЗАЦИЯ БАЗ ДАННЫХ"
для студентов специальности

Утверждено
на заседании кафедры
ИСМ
Протокол N от

Цель и содержание лабораторных работ

Целью данных лабораторных работ является практическое применение знаний полученных при изучении курса организация баз данных.

Первая работа связана с моделированием предметной области. Необходимо изучить специфику выбранной предметной области, выделить основные ключевые сущности и связи между ними. Привести отношения к форме, не имеющей аномалий удаления, модификации и вставки.

Остальные работы связаны с построением базы данных используя стандартный язык запросов SQL. Это построение таблиц, манипулирование данными с помощью запросов и подзапросов. Написание функций и триггеров.

В работе используется сервер баз данных PostgreSQL.

Лабораторная работа №1 (МОДЕЛИРОВАНИЕ ПО)

Основные понятия

Примеры

Пример1

Задание

Разработать базу данных для автоматизации работы оптового склада. Система должна содержать информацию о местах хранения склада, о товаре, о покупках и продажах товара. Кроме того, структура базы данных должна предоставлять возможность хранить другую информацию, которая, по мнению студента, относится к данной предметной области и задачам, решаемым разрабатываемой системой.

Моделирование предметной области:

В результате проведенного анализа данной предметной области, были выявлены следующие ключевые сущности. Теперь определим типы основных сущностей.

Объект	Описание
МестаХранения	Места хранения товара, имеющиеся на складе. Это подразделения имеющие разное территориальное расположение и разных материально-ответственных лиц (МОЛ).
Товары	Товары и их характеристики
Контрагенты	Физические и юридические лица, которые могут покупать или поставлять товар.
Поставки	Информация о поступлениях товара.
Продажи	Информация о продажах товара.
Остатки	Информация об остатках товаров в местах хранения.

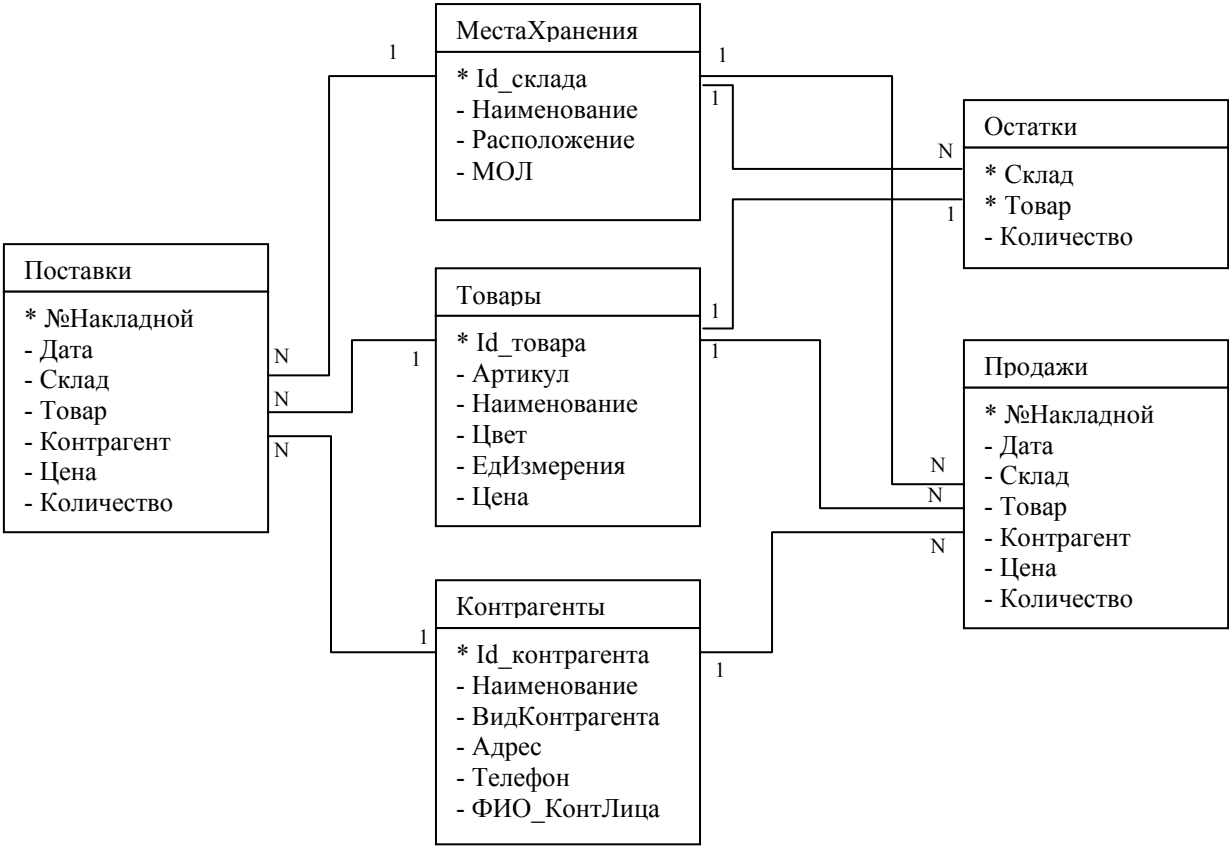
Определим типы связей между полученными сущностями:

Сущность 1	Степень связи	Сущность 2
Поставки	N:1	МестаХранения
Поставки	N:1	Товары
Поставки	N:1	Контрагенты
Продажи	Ny:1	МестаХранения
Продажи	Ny:1	Товары
Продажи	Ny:1	Контрагенты
Остатки	Ny:1	МестаХранения
Остатки	1y:1	Товары

Определение атрибутов:

Сущность	Атрибут	Тип данных
МестаХранения	Id склада	Ключевое поле, целый
	Наименование	Символьный
	Расположение	Символьный
	МОЛ	Символьный
Товары	Id товара	Ключевое поле, целый
	Артикул	Символьный
	Наименование	Символьный
	Цвет	Домен Цвета
	ЕдиницаИзмерения	Домен ЕдиницыИзмерения
	Цена	Вещественный
Контрагенты	Id контрагента	Ключевое поле, целый
	Наименование	Символьный
	ВидКонтрагента	Домен ВидыКонтрагентов
	Адрес	Символьный
	Телефон	Символьный
	ФИО_контактнЛица	Символьный
Поставки	№Накладной	Ключевое поле, целый
	Дата	Дата
	Склад	Целый (МестаХранения)
	Товар	Целый (Товары)
	Контрагент	Целый (Контрагенты)
	Цена	Вещественный
	Количество	Целый
Продажи	№Накладной	Ключевое поле, целый
	Дата	Дата
	Склад	Целый (МестаХранения)
	Товар	Целый (Товары)
	Контрагент	Целый (Контрагенты)
	Цена	Вещественный
	Количество	Целый
Остатки	Склад	Целый (МестаХранения)
	Товар	Целый (Товары)
	Количество	Целый

Информационная модель



Пример 2

Задание

Разработать базу данных для файлового хранилища интернет-проекта. Файловый архив содержит музыку, фильмы и книги в различных форматах. Для доступа к архиву необходима регистрация, скачать файл можно только за определенную плату. Кроме того, структура базы данных должна предоставлять возможность хранить другую информацию, которая, по мнению студента, относится к данной предметной области и задачам, решаемым разрабатываемой системой.

Моделирование предметной области:

В результате проведенного анализа данной предметной области, были выявлены следующие ключевые сущности. Теперь определим типы основных сущностей.

Объект	Описание
Файлы	Супертип. Общая информация о файле, независимо от того музыка это или фильм или книга.
Музыка	Подтип супертипа Файл. Информация о музыкальных файлах.
Фильмы	Подтип супертипа Файл. Информация о файлах с фильмами.
Книги	Подтип супертипа Файл. Информация о файлах с книгами.
Пользователи	Информация о зарегистрированных пользователях
Закачки	Информация о закачках файлов, производимых пользователями.

Определим типы связей между полученными сущностями:

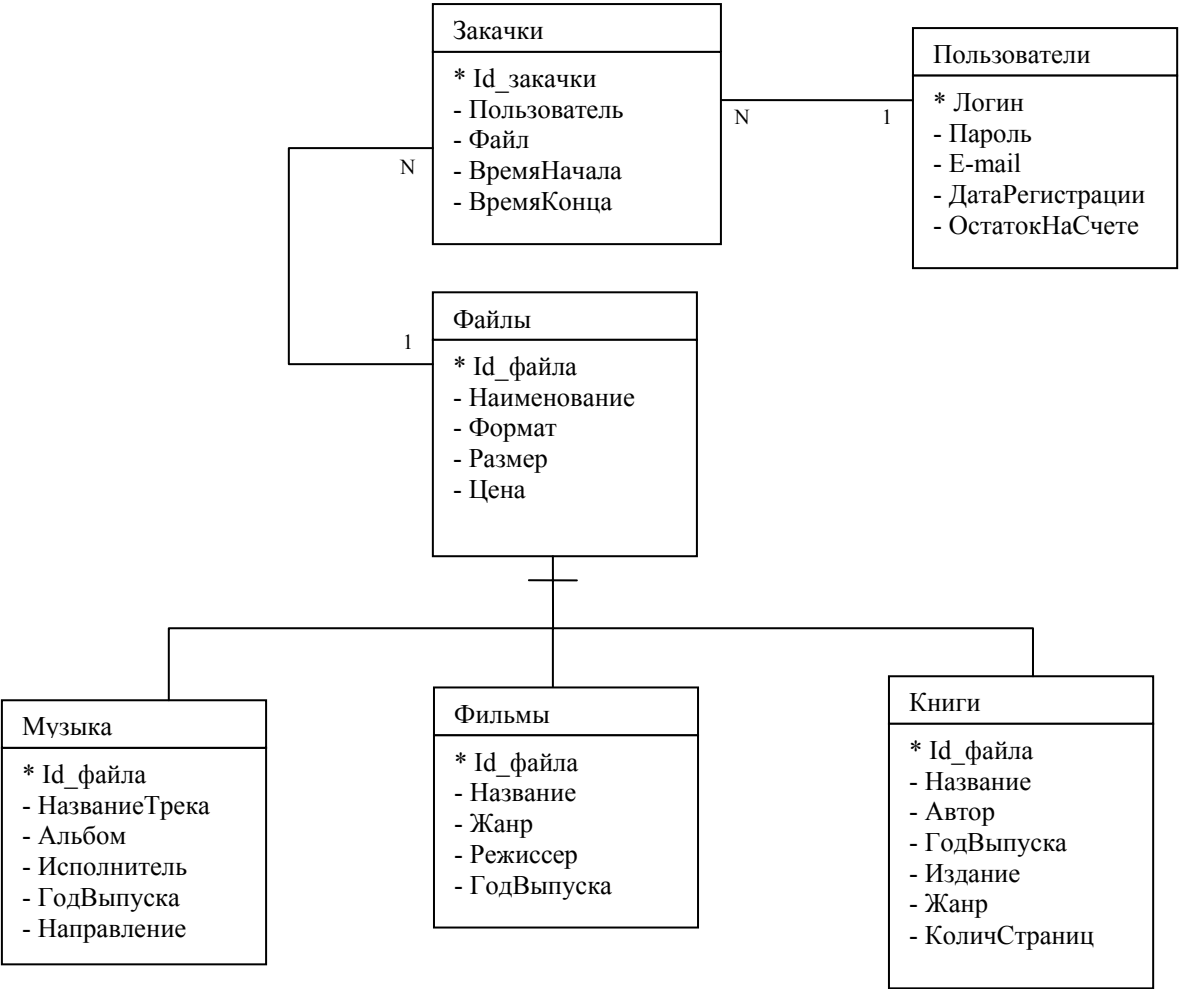
Сущность 1	Степень связи	Сущность 2
Файлы	Супертип-подтип	Музыка
Файлы	Супертип-подтип	Фильмы
Файлы	Супертип-подтип	Книги
Закачки	1y:1	Файлы
Закачки	Ny:1	Пользователи

Определение атрибутов:

Сущность	Атрибут	Тип данных
Файлы	Id файла	Ключевое поле, целый
	Наименование	Символьный
	ПолныйПуть	Символьный
	Формат	Символьный
	Размер	Вещественный
	Цена	Вещественный
Музыка	НазваниеТрека	Символьный
	Альбом	Символьный
	Исполнитель	Символьный
	ГодВыпуска	Символьный
	Направление	Символьный
Фильмы	Название	Символьный
	Жанр	Символьный
	Режиссер	Символьный
	ГодВыпуска	Символьный
Книги	Название	Символьный
	Автор	Символьный
	ГодВыпуска	Символьный

	Издание	Символьный
	Жанр	Символьный
	КоличествоСтраниц	Целый
Пользователи	Логин	Ключевое поле, символьный
	Пароль	Символьный
	E-mail	Символьный
	ДатаРегистрации	Дата
	ОстатокНаСчете	Вещественный
Загрузки	Id загрузки	Ключевое поле, целый
	Пользователь	Символьный (Пользователи)
	Файл	Целый (Файлы)
	ВремяНачала	Дата/время
	ВремяКонца	Дата/время

Информационная модель



Лабораторная работа №2 (SQL запросы для создания таблиц) Основные понятия

Примеры

Напишите SQL запросы для создания таблиц, которые были спроектированы в лабораторной работе №1.

Пример 1

Создание таблиц.

Создание таблицы МестаХранения:

```
create table МестаХранения(  
Id_склада int primary key, /*Целое число, первичный ключ*/  
Наименование char(30), /*Массив символов длины 30*/  
Расположение char(50), /*Массив символов длины 50*/  
МОЛ char(30)); /*Массив символов длины 30*/
```

Создание доменов Цвета и ЕдиницыИзмерения:

```
create domain Цвета char(20) /*Домен типа массив символов длины 20*/  
default 'без цвета' /*Значение по умолчанию*/  
check(value in('красный', 'черный', 'белый', 'серебристый', 'без цвета')); /*В домен входит  
список перечисленных значений*/
```

```
create domain ЕдиницыИзмерения char(10) /*Домен типа массив символов длины 10*/  
default 'шт' /*Значение по умолчанию*/  
check(value in('кг', 'м', 'л', 'шт')); /*В домен входит список перечисленных значений*/
```

Создание таблицы Товары:

```
create table Товары(  
Id_товара int primary key, /*Целое число, первичный ключ*/  
Артикул char(20), /*Массив символов длины 20*/  
Наименование char(50), /*Массив символов длины 50*/  
Цвет Цвета, /*Массив типа домен Цвета */  
ЕдИзмерения ЕдиницыИзмерения, /*Массив типа домен ЕдиницыИзмерения*/  
Цена decimal(10,2)); /*Фиксированное число с плавающей точкой, 2 знака после запятой*/
```

Создание домена ВидыКонтрагентов:

```
create domain ВидыКонтрагентов char(20) /*Домен типа массив символов длины 20*/  
default 'Юридическое лицо' /*Значение по умолчанию*/  
check(value in('Юридическое лицо', 'Физическое лицо')); /*В домен входит список  
перечисленных значений*/
```

Создание таблицы Контрагенты:

```
create table Контрагенты(  
Id_контрагента int primary key, /*Целое число, первичный ключ*/  
Наименование char(50), /*Массив символов длины 50*/
```


ВидКонтрагента ВидыКонтрагентов, /*Массив типа домен ВидыКонтрагентов*/
Адрес char(50), /*Массив символов длины 50*/
Телефон char(30), /*Массив символов длины 30*/
ФИО_контактнЛица char(30)); /*Массив символов длины 30*/

Создание таблицы Поставки:

```
create table Поставки(  
№Накладной int primary key, /*Целое число, первичный ключ*/  
Дата date default current_date, /*Дата, по умолчанию проставляется текущая*/  
Склад int references МестаХранения(Id_склада), /*Целое число, ограничение,  
определяющее, что значение этого поля может быть только такое, которое есть в таблице  
МестаХранения в поле Id_склада*/  
Товар int references Товары(Id_товара), /*Целое число, ограничение, определяющее, что  
значение этого поля может быть только такое, которое есть в таблице Товары в поле  
Id_товара*/  
Контрагент int references Контрагенты(Id_контрагента), /*Целое число, ограничение,  
определяющее, что значение этого поля может быть только такое, которое есть в таблице  
Контрагенты в поле Id_контрагента*/  
Цена decimal(10,2), /*Фиксированное число с плавающей точкой, 2 знака после запятой*/  
Количество int check(Количество>0)); /*Целое число, проверка, число должно быть >0*/
```

Создание таблицы Продажи:

```
create table Продажи(  
№Накладной int primary key, /*Целое число, первичный ключ*/  
Дата date default current_date, /*Дата, по умолчанию проставляется текущая*/  
Склад int references МестаХранения(Id_склада), /*Целое число, ограничение,  
определяющее, что значение этого поля может быть только такое, которое есть в таблице  
МестаХранения в поле Id_склада*/  
Товар int references Товары(Id_товара), /*Целое число, ограничение, определяющее, что  
значение этого поля может быть только такое, которое есть в таблице Товары в поле  
Id_товара*/  
Контрагент int references Контрагенты(Id_контрагента), /*Целое число, ограничение,  
определяющее, что значение этого поля может быть только такое, которое есть в таблице  
Контрагенты в поле Id_контрагента*/  
Цена decimal(10,2), /*Фиксированное число с плавающей точкой, 2 знака после запятой*/  
Количество int check(Количество>0)); /*Целое число, проверка, число должно быть >0*/
```

Создание таблицы Остатки:

```
create table Остатки(  
Склад int references МестаХранения(Id_склада), /*Целое число, ограничение,  
определяющее, что значение этого поля может быть только такое, которое есть в таблице  
МестаХранения в поле Id_склада*/  
Товар int references Товары(Id_товара), /*Целое число, ограничение, определяющее, что  
значение этого поля может быть только такое, которое есть в таблице Товары в поле  
Id_товара*/  
Количество int check(Количество>=0), /*Целое число, проверка, число должно быть >=0*/  
primary key(Склад, Товар)); /*Первичный ключ из двух полей Склад и Товар*/
```

Заполнение таблиц данными:

```
INSERT INTO "МестаХранения" ("id_склада", "Наименование", "Расположение", "МОЛ")  
VALUES (1, 'Склад - Косметика', 'ул.Толбухина 230', 'Иванов П.Р.');
```

```
INSERT INTO "МестаХранения" ("id_склада", "Наименование", "Расположение", "МОЛ")  
VALUES (2, 'Склад - Продукты', 'пр. Мира 8', 'Петров И.Т.');
```

```
INSERT INTO "МестаХранения" ("id_склада", "Наименование", "Расположение", "МОЛ")  
VALUES (3, 'Склад - Мебель', 'ул. М.Жукова 2', 'Сидоров М.Р.');
```

```
INSERT INTO "Контрагенты" ("id_контрагента", "Наименование", "ВидКонтрагента",  
"Адрес", "Телефон", "ФИО_контактнЛица") VALUES (1, 'ЧП "Алиса"', 'Юридическое  
лицо', 'ул. Героев Сталинграда 45', '554676', 'Васильев Петр Иванович');
```

```
INSERT INTO "Контрагенты" ("id_контрагента", "Наименование", "ВидКонтрагента",  
"Адрес", "Телефон", "ФИО_контактнЛица") VALUES (2, 'Новиков И.Н.', 'Физическое  
лицо', 'ул. Греческая 9 кв 5', '345678', 'Новиков Иван Николаевич');
```

```
INSERT INTO "Контрагенты" ("id_контрагента", "Наименование", "ВидКонтрагента",  
"Адрес", "Телефон", "ФИО_контактнЛица") VALUES (3, 'ОАО "Одескабель"',  
'Юридическое лицо', 'Николаевская дорога 144', '234567', 'Максимов Сергей Васильевич');
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (1, 'МОЛ123', 'Молоко в пакетах 1л', 'без цвета', 'шт', 3.80);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (2, 'МОЛ124', 'Молоко', 'без цвета', 'л', 3.20);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (3, 'M567', 'Масло сливочное, 1 сорт', 'без цвета', 'кг', 5.50);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (4, 'M6778', 'Масло растительное, высший сорт, 1л', 'без цвета',  
'шт', 6.70);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (5, 'X687897', 'Хлеб обеденный', 'без цвета', 'шт', 2.40);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (6, 'D6888', 'Диван двуспальный, производитель "Арт",  
'красный', 'шт', 1000.00);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (7, 'Г6677', 'Гарнитур кухонный', 'серебристый', 'шт', 1500.00);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (8, 'C4444', 'Столик журнальный', 'черный', 'шт', 700.00);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (9, 'K5777', 'Крем "Нежность" для лица, 75г.', 'без цвета', 'шт',  
10.40);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (10, 'K3434', 'Крем "Аленка" для рук, 50г.', 'без цвета', 'шт', 5.80);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (11, '5555', 'Шампунь "Shamtu", 250мл.', 'без цвета', 'шт', 6.70);
```

```
INSERT INTO "Товары" ("id_товара", "Артикул", "Наименование", "Цвет",  
"ЕдИзмерения") VALUES (12, '9999', 'Мыло хозяйственное 72%', 'без цвета', 'шт', 2.30);
```

```
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",  
"Цена", "Количество") VALUES (1, '2007-08-20', 1, 9, 1, 10.20, 10);
```

```
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",  
"Цена", "Количество") VALUES (2, '2007-08-20', 1, 10, 1, 9.70, 20);
```

```

INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (3, '2007-08-20', 1, 12, 2, 7.60, 15);
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (4, '2007-08-21', 1, 11, 2, 6.40, 10);
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (5, '2007-08-21', 2, 1, 1, 5.30, 20);
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (6, '2007-08-22', 2, 2, 2, 5.40, 20);
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (7, '2007-08-22', 2, 3, 2, 4.70, 14);
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (8, '2007-08-22', 2, 4, 2, 1.30, 24);
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (9, '2007-08-22', 2, 5, 2, 2.20, 10);
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (10, '2007-08-23', 3, 6, 1, 3000.00, 4);
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (11, '2007-08-23', 3, 7, 1, 2500.00, 2);
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (12, '2007-08-23', 3, 8, 1, 1500.00, 3);

```

```

INSERT INTO "Продажи" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (1, '2007-08-25', 1, 9, 2, 12.20, 1);
INSERT INTO "Продажи" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (2, '2007-08-25', 1, 10, 1, 11.70, 1);
INSERT INTO "Продажи" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (3, '2007-08-25', 2, 5, 3, 2.60, 2);
INSERT INTO "Продажи" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (4, '2007-08-25', 3, 6, 3, 3500.00, 1);
INSERT INTO "Продажи" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (5, '2007-08-25', 3, 7, 3, 4500.00, 1);
INSERT INTO "Продажи" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (6, '2007-08-26', 3, 8, 3, 2000.00, 1);

```

В результате получили такие таблицы:

Места хранения:

	id_склада [PK] integer	Наименование character(30)	Расположение character(50)	МОЛ character(30)
1	1	Склад - Косметика	ул.Толбухина 230	Иванов П.Р.
2	2	Склад - Продукты	пр. Мира 8	Петров И.Т.
3	3	Склад - Мебель	ул. М.Жукова 2	Сидоров М.Р.
*				

Контрагенты:

	id_конт [PK] integer	Наименование character(50)	ВидКонтрагента "ВидыКонтрагента"	Адрес character(50)	Телефон character(10)	ФИО_контактного_лица character(30)
1	1	ЧП "Алиса"	Юридическое лицо	ул. Героев Сталинграда	554676	Васильев Петр Иванович
2	2	Новиков И.Н.	Физическое лицо	ул. Греческая 9 кв 5	345678	Новиков Иван Николаевич
3	3	ОАО "Одескабель"	Юридическое лицо	Николаевская дорога 144	234567	Максимов Сергей Васильевич
*						

Товары:

	id_товара [PK] integer	Артикул character(20)	Наименование character(50)	Цвет "Цвет"	Единица измерения "Единицы"	Цена numeric(10,2)
1	1	МОЛ123	Молоко в пакетах 1л	без цвета	шт	3.80
2	2	МОЛ124	Молоко	без цвета	л	3.20
3	3	M567	Масло сливочное, 1 сорт	без цвета	кг	5.50
4	4	M6778	Масло растительное, высший сорт, 1л	без цвета	шт	6.70
5	5	X687897	Хлеб обеденный	без цвета	шт	2.40
6	6	D6888	Диван двуспальный, производитель "Арт"	красный	шт	1000.00
7	7	G6677	Гарнитур кухонный	серебристый	шт	1500.00
8	8	C4444	Столик журнальный	черный	шт	700.00
9	9	K5777	Крем "Нежность" для лица, 75г.	без цвета	шт	10.40
10	10	K3434	Крем "Аленка" для рук, 50г.	без цвета	шт	5.80
11	11	5555	Шампунь "Shamtu", 250мл.	без цвета	шт	6.70
12	12	9999	Мыло хозяйственное 72%	без цвета	шт	2.30
*						

Поставки:

	№Накладной [PK] integer	Дата date	Склад integer	Товар integer	Контрагент integer	Цена numeric(10,2)	Количество integer
1	1	2007-08-20	1	9	1	10.20	10
2	2	2007-08-20	1	10	1	9.70	20
3	3	2007-08-20	1	12	2	7.60	15
4	4	2007-08-21	1	11	2	6.40	10
5	5	2007-08-21	2	1	1	5.30	20
6	6	2007-08-22	2	2	2	5.40	20
7	7	2007-08-22	2	3	2	4.70	14
8	8	2007-08-22	2	4	2	1.30	24
9	9	2007-08-22	2	5	2	2.20	10
10	10	2007-08-23	3	6	1	3000	4
11	11	2007-08-23	3	7	1	2500	2
12	12	2007-08-23	3	8	1	1500	3
*							

Продажи:

	№Накладной [PK] integer	Дата date	Склад integer	Товар integer	Контрагент integer	Цена numeric(10,2)	Количество integer
1	1	2007-08-25	1	9	2	12.20	1
2	2	2007-08-25	1	10	1	11.70	1
3	3	2007-08-25	2	5	3	2.60	2
4	4	2007-08-25	3	6	3	3500.00	1
5	5	2007-08-25	3	7	3	4500.00	1
6	6	2007-08-26	3	8	3	2000.00	1
*							

Дополнение

Рассмотрим таблицу поставки из примера 1. Обратим внимание на поле НомерНакладной, это поле является первичным ключом, каждой новой накладной присваивается следующий по порядку номер. Для того чтобы пользователю не нужно было следить за нумерацией накладных, в PostgreSQL используются последовательности. В других СУБД этот механизм часто называется счетчиком.

Последовательность – это объект базы данных, который фактически представляет собой автоматически увеличивающееся число.

Опишем последовательность для таблицы Поставки.
create sequence НомерНакладной;

Затем, при добавлении строки в таблицу:

```
INSERT INTO "Поставки" ("№Накладной", "Дата", "Склад", "Товар", "Контрагент",
"Цена", "Количество") VALUES (nextval('НомерНакладной'), '2007-08-20', 1, 9, 1, 10.20,
10);
```

Функция nextval увеличивает текущее значение заданной последовательности и возвращает новое значение в виде величины типа integer.

Для большей надежности использование последовательности НомерНакладной в качестве счетчика в таблице Поставки можно задать на этапе определения таблицы:

```
create table Поставки(
№Накладной int primary key default nextval('НомерНакладной'),
Дата date default current_date,
Склад int references МестаХранения(Id_склада),
Товар int references Товары(Id_товара),
Контрагент int references Контрагенты(Id_контрагента),
Цена decimal(10,2),
Количество int check(Количество>0));
```

Также для этих целей можно использовать тип данных serial.

Для более подробной информации о последовательностях смотрите документацию.

Пример 2

В этом примере показана связь супертип-подтип. В PostgreSQL для организации связи такого типа существует инструмент, называемый наследование. Таблица может наследовать некоторые атрибуты и их свойства от одной или нескольких других таблиц. Производная таблица создается командой Create table, в которую включается секция inherits, выглядит это так:

```
Create table ПроизводнаяТаблица  
(Определение) inherits (БазоваяТаблица) ;
```

В предлагаемом примере базовой таблицей является таблица Файлы, производными таблицами Музыка, Фильмы и Книги. Каждая из производных таблиц обладает своими уникальными свойствами, но у них также много общего, объединенного в таблице Файлы. К сожалению, данный механизм в PostgreSQL реализован не полностью, ограничения таблиц не наследуются. В результате со стороны СУБД нет полного контроля над целостностью данных. Скорее всего, в ближайших версиях PostgreSQL эта проблема будет решена, но пока, в данном примере реализуем этот контроль другим путем. В данной схеме данных существует две проблемы: первая, это то, что мы можем добавить в таблицы Файлы, Музыка и Книги одинаковые значения поля «Id_файла», т.е. ограничение primary key не наследуется; вторая проблема состоит в том, что мы не можем в таблице Загрузки для поля «Файл» создать ограничение references Файлы(Id_файла). Первая проблема решается путем создания последовательности id_файла, которая будет использоваться как счетчик для поля «Id_файла» всех трех производных таблиц Фильмы, Музыка, Книги. Это должно гарантировать неповторяющиеся значения в этом поле. Вторую проблему решим при помощи добавления триггера КонтрольЦелостности в лабораторной работе №5.

Создание таблиц.

Создание таблицы Файлы:

```
create sequence id_файла; /*Создание последовательности id_файла*/
```

```
create table Файлы(  
Id_файла int primary key default nextval('id_файла'), /*Целое число, первичный ключ, по  
умолчанию значение поля равно следующему элементу последовательности id_файла*/  
Наименование char(50) unique, /*Массив символов длины 50, значение уникально*/  
Размер int, /*Целое число*/  
Цена decimal(10,2), /*Фиксированное число с плавающей точкой, 2 знака после запятой*/  
Формат char(4)); /*Массив символов длины 4*/
```

Создание таблицы Музыка:

```
create table Музыка(  
НаименованиеТрека char(50), /*Массив символов длины 50*/  
Альбом char(50), /*Массив символов длины 50*/  
Исполнитель char(50), /*Массив символов длины 50*/  
ГодВыпуска char(4), /*Массив символов длины 4*/  
Направление char(50)) /*Массив символов длины 50*/  
inherits (Файлы); /*Данная таблица наследует поля таблицы Файлы*/
```

Создание таблицы Фильмы:

```
create table Фильмы(  
Название char(50), /*Массив символов длины 50*/  
Жанр char(50), /*Массив символов длины 50*/  
Режиссер char(50), /*Массив символов длины 50*/  
ГодВыпуска char(4)) /*Массив символов длины 4*/  
inherits (Файлы); /*Данная таблица наследует поля таблицы Файлы*/
```

Создание таблицы Книги:

```
create table Книги(  
Название char(50), /*Массив символов длины 50*/  
Автор char(50), /*Массив символов длины 50*/  
Издание char(50), /*Массив символов длины 50*/  
ГодВыпуска char(4), /*Массив символов длины 4*/  
КоличествоСтр int, /*Целое число*/  
Жанр char(50)) /*Массив символов длины 50*/  
inherits (Файлы); /*Данная таблица наследует поля таблицы Файлы*/
```

Создание таблицы Пользователи:

```
create table Пользователи(  
Логин char(10) primary key, /*Массив символов длины 10, первичный ключ*/  
Пароль char(10), /*Массив символов длины 10*/  
e_mail char(50) check(e_mail like '%@%'), /*Массив символов длины 50, проверка на  
присутствие символа @ в строке*/  
Остаток decimal(10,2), /*Фиксированное число с плавающей точкой, 2 знака после  
запятой*/  
ДатаРегистрации date); /*Дата*/
```

Создание таблицы Закачки:

```
create sequence закачки; /*Создание последовательности закачки*/
```

```
create table Закачки(  
Id_закачки int primary key default nextval('закачки'), /*Целое число, первичный ключ, по  
умолчанию значение поля равно следующему элементу последовательности закачки*/  
Пользователь char(10) references Пользователи(Логин), /*Строка символов длины 10,  
ограничение, определяющее, что значение этого поля может быть только такое, которое  
есть в таблице Пользователи в поле Логин*/  
Файл int, /* Целое число, здесь должно было быть references Файлы(Id_файла), но пока  
оставим эту запись до выхода следующих версий PostgreSQL*/  
ВремяНачала timestamp, ), /*Дата и время*/  
ВремяКонца timestamp); /*Дата и время*/
```

Заполнение таблиц:

```
INSERT INTO "Фильмы" ("Наименование", "Размер", "Цена", "Формат", "Название",  
"Жанр", "Режиссер", "ГодВыпуска")  
VALUES('Movies\ПятыйЭлемент', 700, 20.00, 'avi', 'Пятый элемент', 'Фантастика,  
Приключения', 'Люк Бессон', '1997');  
INSERT INTO "Фильмы" ("Наименование", "Размер", "Цена", "Формат", "Название",  
"Жанр", "Режиссер", "ГодВыпуска")
```

```
VALUES('Movies\\ДжейнОстин', 693, 15.00, 'avi', 'Джейн Остин', 'Мелодрама, Драма',
'Джулиан Джаррольд', '2007');
INSERT INTO "Фильмы" ("Наименование", "Размер", "Цена", "Формат", "Название",
"Жанр", "Режиссер", "ГодВыпуска")
VALUES('Movies\\ПротивостояниеГигантам', 680, 18.00, 'avi', 'Противостояние гигантам',
'Драма', 'Алекс Кендрик', '2006');
INSERT INTO "Фильмы" ("Наименование", "Размер", "Цена", "Формат", "Название",
"Жанр", "Режиссер", "ГодВыпуска")
VALUES('Movies\\Matrix', 693, 15.00, 'avi', 'Матрица', 'Фентези, Фантастика', 'Энди
Вачовски и Ларри Вачовски', '2004');
```

```
INSERT INTO "Музыка" ("Наименование", "Размер", "Цена", "Формат",
"НаименованиеТрека", "Альбом", "Исполнитель", "ГодВыпуска", "Направление")
VALUES('Music\\treck2', 6, 1.50, 'mp3', 'How High', 'Confessions On A Dance Floor',
'Madonna', '2005', 'Pop');
INSERT INTO "Музыка" ("Наименование", "Размер", "Цена", "Формат",
"НаименованиеТрека", "Альбом", "Исполнитель", "ГодВыпуска", "Направление")
VALUES('Music\\treck6', 4, 1.50, 'mp3', 'Hung Up', 'Confessions On A Dance Floor', 'Madonna',
'2005', 'Pop');
INSERT INTO "Музыка" ("Наименование", "Размер", "Цена", "Формат",
"НаименованиеТрека", "Альбом", "Исполнитель", "ГодВыпуска", "Направление")
VALUES('Music\\treck8', 5, 1.50, 'mp3', 'Push', 'Confessions On A Dance Floor', 'Madonna',
'2005', 'Pop');
INSERT INTO "Музыка" ("Наименование", "Размер", "Цена", "Формат",
"НаименованиеТрека", "Альбом", "Исполнитель", "ГодВыпуска", "Направление")
VALUES('Music\\IWBL', 3, 1.80, 'mp3', 'I wanna be loved', 'Have a nice day', 'Bon Jovi', '2005',
'Rock');
INSERT INTO "Музыка" ("Наименование", "Размер", "Цена", "Формат",
"НаименованиеТрека", "Альбом", "Исполнитель", "ГодВыпуска", "Направление")
VALUES('Music\\Билан_трек2', 4, 1.50, 'mp3', 'Ты должна быть рядом', 'На берегу неба',
'Дима Билан', '2004', 'Pop');
```

```
INSERT INTO "Книги" ("Наименование", "Размер", "Цена", "Формат", "Название",
"Автор", "Издание", "ГодВыпуска", "КоличествоСтр", "Жанр")
VALUES('Books\\ТиПрПБД', 7, 25.00, 'djvu', 'Теория и практика построения баз данных',
'Д.Кренке', 'Питер', 400, '2003', 'Информатика');
INSERT INTO "Книги" ("Наименование", "Размер", "Цена", "Формат", "Название",
"Автор", "Издание", "ГодВыпуска", "КоличествоСтр", "Жанр")
VALUES('Books\\ПроектирБД', 6, 15.00, 'djvu', 'Проектирование реляционных баз данных',
'Джен Л.Харрингтон', 'Лори', 241, '2006', 'Информатика');
INSERT INTO "Книги" ("Наименование", "Размер", "Цена", "Формат", "Название",
"Автор", "Издание", "ГодВыпуска", "КоличествоСтр", "Жанр")
VALUES('Books\\ПостгресДляПрофессионалов', 8, 21.00, 'pdf', 'PostgreSQL. Для
профессионалов', 'Дж.Уорслей, Дж.Дрейк', 'Питер', 498, '2003', 'Информатика');
```

```
INSERT INTO "Пользователи" ("Логин", "Пароль", "e_mail", "Остаток",
"ДатаРегистрации")
VALUES('Petya', '12345', 'petya@mail.ru', 200.00, '2007-10-22');
```



```
INSERT INTO "Пользователи" ("Логин", "Пароль", "е_mail", "Остаток",  
"ДатаРегистрации")  
VALUES('Vasya', '54321', 'Vasya@mail.ru', 100.00, '2007-10-24');  
INSERT INTO "Пользователи" ("Логин", "Пароль", "е_mail", "Остаток",  
"ДатаРегистрации")  
VALUES('Slava', '12345', 'slava@mail.ru', 120.00, '2007-10-30');
```

```
INSERT INTO "Закачки" ("Пользователь", "Файл", "ВремяНачала", "ВремяКонца")  
VALUES('Petya', 2, '2007-12-03 11:27:23.809', '2007-12-03 11:35:20.000');  
INSERT INTO "Закачки" ("Пользователь", "Файл", "ВремяНачала", "ВремяКонца")  
VALUES('Slava', 8, '2007-12-03 11:27:23.809', '2007-12-03 11:50:13.609');
```

Лабораторная работа №3 (ЗАПРОСЫ)

Основные понятия

Примеры

Напишите SQL запросы для манипулирования данными в таблицах, созданных в лабораторной работе №2.

Пример 1

1. У одного из контрагентов изменился номер телефона и адрес, составить SQL запрос для соответствующих изменений в таблицах.

```
update Контрагенты set Адрес='ул. Греческая 30, кв 4', Телефон='221234' where  
Наименование = 'ЧП "Алиса"'
```

2. Создайте представление, содержащее информацию о контрагентах, являющихся юридическими лицами.

```
create view ЮрЛица as  
select * from Контрагенты where ВидКонтрагента='Юридическое лицо'
```

3. Сколько наименований товаров содержит справочник Товары, у которых единица измерения - штуки.

```
select count(Наименование) from Товары where ЕдИзмерения='шт'
```

4. Какая выручка была получена от продаж в августе 2007 г.

```
select sum(Цена*Количество) from Продажи where Дата>='2007-08-01' and Дата<='2007-08-31'
```

5. Какая выручка была получена в августе 2007 года, за каждый день в отдельности.

```
select Дата, sum(Цена*Количество) from Продажи where Дата>='2007-08-01' and  
Дата<='2007-08-31'  
group by Дата
```

```
select Дата, sum(Цена*Количество) from Продажи  
group by Дата having Дата>='2007-08-01' and Дата<='2007-08-31'
```

6. Показать продаваемые товары.

```
select distinct Т.Наименование from Товары Т, Продажи П where П.Товар=Т.Id_товара
```

7. Выберите из базы данных информацию о поставках, произошедших 22 августа 2007 г. Создайте представление, в котором необходимо отобразить номер накладной, склад на который пришел товар, наименование товара, цена, количество, сумма.

```
create view ПриходТовара as
select П.№Накладной, МХ.Наименование as Склад, Т.Наименование as Товар, П.Цена,
П.Количество, П.Цена*П.Количество as Сумма from МестаХранения МХ, Товары Т,
Поставки П
where П.Склад=МХ.Id_склада and П.Товар=Т.Id_товара and П.Дата='2007-08-22'
```

8. Вывести контрагентов, которые являются одновременно и покупателями и продавцами.

```
select distinct К.Наименование from Контрагенты К, Поставки Пост, Продажи Пр
where К.Id_контрагента=Пост.Контрагент and К.Id_контрагента=Пр.Контрагент and
Пост.Контрагент=Пр.Контрагент
```

Пример 2

1. Пользователь Slava положил на свой счет 50 грн.

```
update Пользователи set Остаток=Остаток+50 where Логин = 'Slava'
```

2. Найти наименования файлов, которые скачал пользователь с логином Petya

```
select Пользователь, Наименование from Закачки, Файлы where
Закачки.Файл=Файлы.id_файла and Пользователь='Petya'
```

3. Сколько суммарно места занимают в базе файлы каждого формата, отсортировать в порядке возрастания занимаемого объема.

```
select Формат, sum(Размер) from Файлы group by Формат order by sum(Размер)
```

4. Создать представление Отчет для отображения логина пользователя, его электронного почтового адреса, имени файла, который он закачивал и стоимости закладки этого файла.

```
create view Отчет as
select П.Логин, П.e_mail, Ф.Наименование, Ф.Цена from Пользователи П, Закачки З,
Файлы Ф where П.Логин=З.Пользователь and З.Файл=Ф.id_файла
```

5. Найти общее время всех закачек

```
select sum(ВремяКонца-ВремяНачала) from Закачки
```

6. Найти общее время закачек каждого пользователя

```
select Пользователь, sum(ВремяКонца-ВремяНачала) from Закачки group by Пользователь
```

7. Вывести пользователей остаток на счету у которых 0 и которые зарегистрировались до начала 2007 года

```
select Логин from Пользователи where ДатаРегистрации<'2007-01-01' and Остаток=0
```

8. Максимальный файл по размеру, который закачал каждый пользователь

```
select З.Пользователь, max(Ф.Размер) from Файлы Ф, Загрузки З where Ф.id_файла=З.Файл
group by З.Пользователь
```

Лабораторная работа №4 (ПОДЗАПРОСЫ)

Основные понятия

Примеры

Напишите SQL запросы с использованием подзапросов для манипулирования данными в таблицах, созданных в лабораторной работе №2.

Пример 1

1. Кто из контрагентов совершил покупку на самую большую сумму.

```
select К.Наименование, П.Цена*П.Количество from Контрагенты К, Продажи П where
К.Id_контрагента=П.Контрагент and П.Цена*П.Количество=(select max(Цена*Количество)
from Продажи)
```

2. Кто из контрагентов принес большую выручку.

```
select max(Сумма) from (select Контрагент, sum(Цена*Количество) as Сумма from
Продажи group by Контрагент) ПЗ
```

3. Показать непроданные товары.

```
select Тов.Наименование from Товары Тов where Id_товара not in(select distinct Товар from
Продажи)
```

Пример 2

1. Найти пользователя, который закачал файл максимального размера из всех закаченных файлов.

```
select З.Пользователь, max(Ф.Размер) from Файлы Ф, Загрузки З where Ф.id_файла=З.Файл
group by З.Пользователь having max(Ф.Размер)=(select max(ПФ.Размер) from (select
З.Пользователь, max(Ф.Размер) from Файлы Ф, Загрузки З where Ф.id_файла=З.Файл group
by З.Пользователь) ПФ)
```

2. Найти пользователей, которые не закатывали файлы из раздела Музыка

```
select З.Пользователь from Загрузки З, Файлы Ф where З.Файл=id_файла and З.Файл not in
(select id_файла from Музыка)
```

3. Найти файлы, которые ни разу не закатывались.

```
select Наименование from Файлы where id_файла not in(select Файл from Загрузки)
```

Лабораторная работа №5 (ФУНКЦИИ И ТРИГГЕРЫ)

Основные понятия

Примеры

Функции и триггеры.

Пример 1

Написать триггеры для поддержания актуального количества товара на складах в таблице Остатки.

Триггер, который при добавлении записи в таблицу Поставки, проверяет, есть ли запись с таким складом и товаром в таблице остатки, если нет, то добавляет ее со значением Количество = 0, затем увеличивает Количество в остатках на количество в поставке товара.

```
CREATE FUNCTION ПриходТовара()
RETURNS trigger
AS 'DECLARE /*Блок объявления переменных*/
    rec record;
BEGIN /*Начало основного программного блока функции*/
    select into rec * /*выбрать записи в переменную rec*/
    from Остатки /*из таблицы Остатки*/
    where new.Товар=Товар; /*где товар такой же как в Поставке*/
    if not found /*если не найдено не одной записи удовлетворяющей условию*/
    then insert into Остатки values (new.Склад, new.Товар, 0); /*то вставить
соответствующую строку в таблицу Остатки*/
    end if;
    update Остатки set Количество=Количество+new.Количество where new.Товар=Товар
and new.Склад=Склад; /*увеличить Количество в строке с соответствующим Товаром
и Складом на количество товара, указанное в поставке*/
    RETURN new; /*Вернуть новую запись для таблицы Поставки*/
END;' /*Конец основного программного блока функции*/
LANGUAGE 'plpgsql';
```

```
CREATE TRIGGER ПриходТовара /*Создать триггер*/
AFTER INSERT /*вызываемый после операции insert*/
ON Поставки /*для таблицы поставки*/
FOR EACH ROW /*для каждого кортежа*/
EXECUTE PROCEDURE ПриходТовара(); /*вызвать процедуру ПриходТовара()*/
```

Триггер, который при добавлении записи в таблицу Продажи, уменьшает количество товара в остатках на величину занесенную в продаже.

```
CREATE FUNCTION ПродажаТовара()
RETURNS trigger
AS 'BEGIN /*Начало основного программного блока функции*/
    update Остатки
    set Количество=Количество-new.Количество
    where new.Товар=Товар and new.Склад=Склад;
    RETURN new;
```

```
END;'  
LANGUAGE 'plpgsql';
```

```
CREATE TRIGGER ПродажаТовара /*Создать триггер*/  
AFTER INSERT /*вызываемый после операции insert*/  
ON Продажи /*для таблицы продажи*/  
FOR EACH ROW /*для каждого кортежа*/  
EXECUTE PROCEDURE ПродажаТовара(); /*вызвать процедуру ПродажаТовара()*/
```

Функция подсчитывает стоимость товаров на всех складах

```
CREATE FUNCTION ПодсчетСтоимостиТоваров()  
RETURNS float  
AS ' DECLARE /*Блок объявления переменных*/  
    Сумма float;  
    rec RECORD;  
    BEGIN /*Начало основного программного блока функции*/  
        Сумма:=0;  
        FOR rec IN SELECT * FROM Остатки LOOP  
            Сумма:=Сумма+rec.Количество*(Select Цена from Товары where  
id_товара=rec.Товар);  
        END LOOP;  
        RETURN Сумма;  
    END;'  
LANGUAGE 'plpgsql';
```

Функция подсчитывает стоимость товаров на конкретном складе

```
CREATE FUNCTION ПодсчетСтоимостиТоваров(int)  
RETURNS float  
AS ' DECLARE /*Блок объявления переменных*/  
    Сумма float;  
    rec record;  
    BEGIN /*Начало основного программного блока функции*/  
        Сумма:=0;  
        FOR rec IN SELECT * FROM Остатки where Склад=$1 LOOP  
            Сумма:=Сумма+rec.Количество*(Select Цена from Товары where  
id_товара=rec.Товар);  
        END LOOP;  
        RETURN Сумма;  
    END;'  
LANGUAGE 'plpgsql';
```

Пример 2

Написать триггер для организации контроля целостности данных. При записи кортежа в таблицу Закачки, проверяем, есть ли файл с таким id_файла в таблице Файлы, если есть, то кортеж благополучно записывается в таблицу Закачки, если его там нет, то система выдаст ошибку "File not found"

```
CREATE FUNCTION КонтрольЦелостности()
RETURNS trigger
AS 'DECLARE
    rec record;
BEGIN /*Начало основного программного блока функции*/
    select into rec *
    from Файлы where id_файла=new.Файл;
    if not found
        THEN RAISE EXCEPTION "File not found";
    else
        RETURN new;
    end if;

END;'
LANGUAGE 'plpgsql';
```

```
CREATE TRIGGER КонтрольЦелостности
BEFORE INSERT
ON Закачки
FOR EACH ROW
EXECUTE PROCEDURE КонтрольЦелостности();
```

Написать функцию ИзменениеЦены, входным параметром является процент, на который нужно увеличить цену. Функция изменяет поле Цена во всех строках таблицы Файлы.

```
CREATE OR REPLACE FUNCTION ИзменениеЦены(int)
RETURNS int
AS 'BEGIN /*Начало основного программного блока функции*/
    UPDATE Файлы SET Цена=Цена*(1.0+cast($1 as float)/100);
    RETURN 1;
END;'
LANGUAGE 'plpgsql';
```