**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное**

**учреждение**

**высшего профессионального образования**

**Национальный исследовательский университет**

**«Высшая школа экономики»**

**Московский институт электроники и математики**

**Национального исследовательского университета**

**«Высшая школа экономики»**

**Домашняя работа по курсу «Базы данных»**

**на тему:**

**«База данных производства фруктовой продукции».**

**Выполнил:**

студент группы

БИВ-135

Чибисов А.П.

**Проверила**:

к.т.н, доцент

Карпова И.П.

**Москва, 2016**

Оглавление

[1. Инфологическое проектирование 3](#_Toc449735429)

[1.1. Анализ предметной области 3](#_Toc449735430)

[1.2. Анализ информационных задач и круга пользователей системы 4](#_Toc449735431)

[2. Определение требований к операционной обстановке 4](#_Toc449735432)

[3. Выбор системы управления базой данных и других программных средств 5](#_Toc449735433)

[4. Логическое проектирование базы данных 6](#_Toc449735434)

[4.1. Преобразование ER-диаграммы в схему реляционной базы данных 6](#_Toc449735435)

[4.2. Составление реляционных отношений 6](#_Toc449735436)

[4.3. Нормализация реляционных отношений 9](#_Toc449735437)

[4.4. Определение дополнительных ограничений целостности 13](#_Toc449735438)

[4.5. Описание групп пользователей и прав доступа 14](#_Toc449735439)

[5. Реализация проекта базы данных 14](#_Toc449735440)

[5.1. Создание таблиц 14](#_Toc449735441)

[5.2. Создание представлений (готовых запросов) 17](#_Toc449735442)

[5.3. Создание функции 20](#_Toc449735443)

[5.4. Создание процедуры 21](#_Toc449735444)

[5.5. Создание триггеров 23](#_Toc449735445)

[5.6. Создание индексов 26](#_Toc449735446)

[5.7. Разработка стратегии резервного копирования 26](#_Toc449735447)

[6. Список литературы 26](#_Toc449735448)

# 1. Инфологическое проектирование

## **1.1. Анализ предметной области**

Особенности предметной области:

* Каждый поставщик может поставлять несколько поставок фруктов, у одной поставки должен быть только один поставщик.
* Одна поставка может входить в состав нескольких продукций, у одной продукции может быть в составе несколько поставок фруктов.
* Один отдел может производить несколько продукций, одна продукция может быть произведена только одним отделом.
* Каждый сотрудник работает в определенном отделе, в одном отделе может быть несколько сотрудников.
* Каждый отдел специализируется на выпуске определенного вида продукции (пр.: джем, варенье и т.п.).
* Зарплата сотрудника зависит только от занимаемой им должности.

Сущности предметной области:

1. **Поставщики.** Атрибуты: ид, название, адрес, телефон, контактное лицо.
2. **Поставки продуктов.** Атрибуты: ид, название, дата сбора, дата поставки, срок годности, вес, цена, страна происхождения.
3. **Продукция.** Атрибуты: ид, название, тип упаковки, дата производства, срок годности, масса брутто.
4. **Отделы.** Атрибуты: ид, название, телефон, специализация.
5. **Сотрудники.** Атрибуты: ид, ФИО, дата рождения, пол, паспортные данные, СНИЛС, ИНН, адрес, телефон, должность, заработная плата, образование.

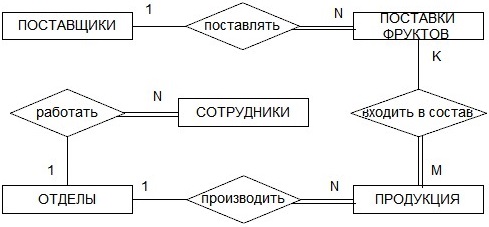


Рис. 1. ER–диаграмма «Производство фруктовой продукции»

## **1.2. Анализ информационных задач и круга пользователей системы**

Группы пользователей и их примерные задачи и запросы:

1. Руководители организации:

* придумывание новой продукции
* создание отделов
* создание/изменение должностей и их зарплат
* заключение договоров с поставщиками
* просмотр полных сведений о производстве

1. Руководители отделов с производством:

* изменение информации о продукции
* просмотр сведений о поставках
* просмотр сведений о сотрудниках своего отдела
* просмотр сведений о начальниках других отделов

1. Менеджеры производства:

* изменение единиц измерения
* изменение информаций о составе
* изменение списка типов упаковок

1. Рабочие отдела с производством:

* просмотр данных о других людях этого отдела

1. Сотрудники отдела кадров:

* прием/увольнение сотрудников
* изменение данных о сотрудниках

1. Бухгалтеры:

* получение ведомости на выплату зарплаты
* получение информации о закупленных поставках

1. Отдел закупок:

* ведение отчетности по поставкам.

# 2. Определение требований к операционной обстановке

На основе результатов анализа предметной области рассчитаем приблизительный объём памяти, требуемый для хранения данных.

Ориентировочно:

* на производстве будет существовать примерно 15 отделов на которых будут трудиться примерно 1 000 сотрудников;
* будет выпускаться около 100 000 продукций в месяц;
* поставщиков будет порядка 1000 фирм, которые будут поставлять приблизительно 100 000 поставок в месяц;
* для связи поставок и поставщиков будет храниться информации о составе продукции примерно 100 000 \* 5, где 5 – примерное максимальное вхождение фруктов в состав.

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование** | **Примерный объем одной записи, КБ** | **Количество записей, шт** | **Общий объем, КБ** |
| Поставщики | 0,2 | 1 000 | 200 |
| Поставки фруктов | 0,25 | 100 000 | 25 000 |
| Сотрудники | 1 | 1 000 | 1 000 |
| Отделы | 0,1 | 15 | 1,5 |
| Продукция | 0,5 | 100 000 | 50 000 |
| Состав | 0,1 | 500 000 | 50 000 |
|  |  | **Всего** | 126201,5 |

Тогда объем памяти для хранения данных примерно составит:

Мд= 2 \* 126,3 = 252,6 МБ

Коэффициент 2 введён для того, чтобы учесть необходимость выделения памяти

под дополнительные структуры (например, индексы). Объём памяти будет

увеличиваться ежемесячно примерно на столько же при условии сохранения показателей выпуска.

Для хорошего функционирования нашей базы данных необходим немаленький объём памяти, растущий ежемесячно, это необходимо учитывать, при выборе сервера базы данных.

# 3. Выбор системы управления базой данных и других программных средств

Анализ информационных задач показывает, что для реализации требуемых функций подходят почти все СУБД для ПЭВМ (MS Access, Firebird, MySQL и др.). Все они поддерживают реляционную модель данных и предоставляют разнообразные возможности для работы с данными.

Для проектирования базы данных производства фруктовой продукции нами была выбрана свободная реляционная система управления базами данных MySQL, так как изучению именно этой системы была часть курса «Базы данных».

# 4. Логическое проектирование базы данных

## **4.1. Преобразование ER-диаграммы в схему реляционной базы данных**

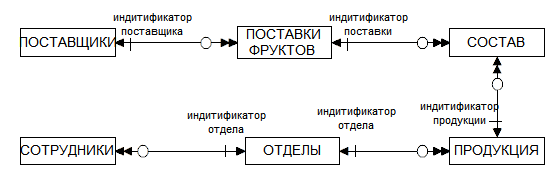


Рис. 2. Схема РБД, полученная из ER–диаграммы производства фруктовой продукции

## **4.2. Составление реляционных отношений**

Таблица 1. Поставщики (Providers)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_p | number(3) | ПК |
| название | name\_p | varchar(30) | обязательное поле |
| адрес | address\_p | varchar(100) | обязательное поле |
| телефон | phone\_p | varchar(10) | обязательное поле |
| контактное лицо | contname\_p | varchar(30) |  |

Таблица 2. Поставки фруктов (PostavkiFruits)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_pf | number(3) | ПК |
| название | name\_pf | varchar(30) | обязательное поле |
| дата сбора | datesbor\_pf | date | обязательное поле |
| срок годности | srok\_pf | date | обязательное поле |
| вес | weigth\_pf | number(10,6) | обязательное поле, >0 |
| единица измерения | edizm\_pf | varchar(20) | обязательное поле |
| стоимость | price\_pf | number(10,2) | обязательное поле, значение по умолчанию - 0 |
| страна происхождения | country\_pf | varchar(50) | обязательное поле |
| поставщик | postavshik\_pf | number(3) | Внк к Providers |
| дата поступления | datepost\_pf | date | обязательное поле, больше или равна даты сбора |

Таблица 3. Cостав (Sostav)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_s | number(3) | ПК |
| продукция | prod\_s | number(3) | Внк к Production |
| фрукт | fruct\_s | number(3) | Внк к PostavkiFruits |
| кол-во(масса) | kol\_s | number(10,6) | обязательное поле, >0 |
| единица измерения | edizm\_f | varchar(20) | обязательное поле |

Таблица 4. Продукция (Production)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_prod | number(3) | ПК |
| название | name\_prod | varchar(50) | обязательное поле |
| тип упаковки | type\_prod | varchar(30) | обязательное поле |
| дата производства | date\_prod | date | обязательное поле |
| срок годности | srok\_prod | date | обязательное поле, больше даты производства |
| произведено отделом | otd\_prod | number(3) | Внк к Otdels |
| масса брутто | mas\_prod | number(10,6) | обязательное поле, >0 |
| единица измерения | edizm\_prod | varchar(20) | обязательное поле |

Таблица 5. Отделы (Otdels)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_o | number(3) | ПК |
| название | name\_o | varchar(30) | обязательное поле |
| телефон | phone\_o | varchar(10) | обязательное поле |
| специализация | spec\_o | varchar(20) | обязательное поле |

Таблица 6. Работники (Employees)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_e | number(3) | ПК |
| ФИО | name\_e | varchar(65) | обязательное поле |
| дата рождения | born\_e | date | обязательное поле |
| пол | sex\_e | char(1) | обязательное поле |
| паспортные данные | passport\_e | varchar(50) | обязательное поле |
| ИНН | inn\_e | char(12) | обязательное поле |
| СНИЛС | snils\_e | char(14) | обязательное поле |
| адрес | address\_e | varchar(100) | обязательное поле |
| телефон | phone\_e | varchar(10) |  |
| отдел | otdel\_e | number(3) | Внк к Otdels |
| должность | dolzn\_e | varchar(30) | обязательное поле |
| заработная плата | sal\_e | number(15) | обязательное поле |
| образование | obraz\_e | varchar(30) | обязательное поле |

## **4.3. Нормализация реляционных отношений**

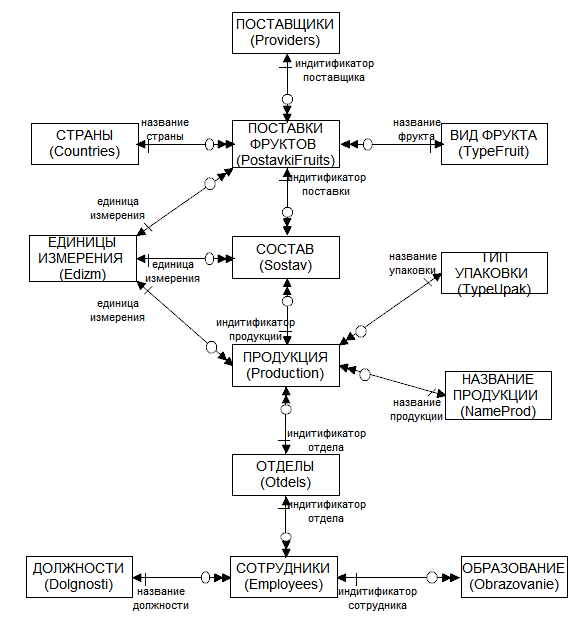
****

Рис. 3. Окончательная схема БД производства фруктовой продукции

Таблица 1.1 Поставщики (Providers)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_p | number(3) | ПК |
| название | name\_p | varchar(30) | обязательное поле |
| адрес | address\_p | varchar(100) | обязательное поле |
| телефон | phone\_p | varchar(10) | обязательное поле |
| контактное лицо | contname\_p | varchar(30) |  |

Таблица 2.1 Поставки фруктов (PostavkiFruits)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_pf | number(3) | ПК |
| название | name\_pf | varchar(30) | Внк к TypeFruit |
| дата сбора | datesbor\_pf | date | обязательное поле |
| срок годности | srok\_pf | date | обязательное поле, больше даты сбора |
| вес | weigth\_pf | number(10,6) | обязательное поле, >0 |
| единица измерения | edizm\_pf | varchar(20) | Внк к Edizm |
| стоимость | price\_pf | number(10,2) | обязательное поле, значение по умолчанию - 0 |
| страна происхождения | country\_pf | varchar(50) | Внк к Countries |
| поставщик | postavshik\_pf | number(3) | Внк к Providers |
| дата поступления | datepost\_pf | date | обязательное поле, больше или равна даты сбора |

Таблица 2.2. Виды фруктов (TypeFruit)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Название фрукта | name\_f | varchar(30) | ПК |

Таблица 2.3. Страны (Countries)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Название страны | name\_count | varchar(50) | ПК |

Таблица 2.4. Единицы измерения(Edizm)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| единица измерения | name\_ed | varchar(20) | ПК |
| система мер | sistmer\_ed | varchar(20) | обязательное поле |
| коэффициент перевода в кг | kg\_ed | number(10,6) | обязательно поле, >0 |

Таблица 3.1 Состав (Sostav)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_s | number(3) | ПК |
| продукция | prod\_s | number(3) | Внк к Producion |
| фрукт | fruct\_s | number(3) | Внк к PostavkiFruits |
| кол-во(масса) | mas\_s | number(10,6) | Обязательное поле, >0 |
| единица измерения | edizm\_s | varchar(20) | Внк к Edizm |

Таблица 4.1 Продукция (Production)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_prod | number(3) | ПК |
| название | name\_prod | varchar(50) | Внк к NameProd |
| тип упаковки | type\_prod | varchar(30) | Внк к TypesUpak |
| дата производства | date\_prod | date | обязательное поле |
| срок годности | srok\_prod | date | обязательное поле, больше даты производства |
| произведено отделом | otd\_prod | number(3) | Внк к Otdels |
| масса брутто | mas\_prod | number(10,6) | обязательное поле, >0 |
| единица измерения | edizm\_prod | varchar(20) | Внк к Edizm |

Таблица 4.2. Названия продукции (NameProd)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| название продукции | name\_pr | varchar(50) | ПК |

Таблица 4.3. Типы упаковки (TypeUpak)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| название упаковки | name\_t | varchar(30) | ПК |
| вес упаковки | weigth\_t | number(10,6) | обязательное поле, >0 |

Таблица 5.1 Отделы (Otdels)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_o | number(3) | ПК |
| название | name\_o | varchar(30) | обязательное поле |
| телефон | phone\_o | varchar(10) | обязательное поле |
| специализация | spec\_o | varchar(20) | обязательное поле |

Таблица 6.1 Работники (Employees)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id | id\_e | number(3) | ПК |
| фамилия | fname\_e | varchar(25) | обязательное поле |
| имя, отчество | lname\_e | varchar(40) | обязательное поле |
| дата рождения | born\_e | date | обязательное поле |
| пол | sex\_e | char(1) | обязательное поле |
| серия и номер паспорта | sernum\_e | char(10) | обязательное уникальное поле |
| когда выдан паспорт | date\_e | date | обязательное поле |
| кем выдан паспорт | given\_e | varchar(50) | обязательное поле |
| ИНН | inn\_e | char(12) | обязательное уникальное поле |
| СНИЛС | snils\_e | char(14) | обязательное уникальное поле |
| адрес | address\_e | varchar(100) | обязательное поле |
| телефон | phone\_e | varchar(10) |  |
| отдел | otdel\_e | number(3) | Внк к Otdels |
| должность | dolzn\_e | varchar(30) | Внк к Dolgnosti |

Таблица 6.2. Должности (Dolgnosti)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| название должности | name\_dol | varchar(30) | ПК |
| заработная плата | sal\_dol | number(10,2) | обязательное поле, > 6250 руб. |

Таблица 6.3. Образоание (Obrazovanie)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| id работника | id\_obr | number(3) | Внк к Employees |
| тип образования | type\_obr | number(15) | обязательное поле |
| специальность | spec\_obr | varchar(30) |  |
| дата окончания учебного заведения | date\_obr | date | обязательное поле |
| номер диплома/аттестата | numdip\_obr | number(15) |  |

## **4.4. Определение дополнительных ограничений целостности**

* + - 1. Атрибут пол (sex\_e) может принимать одно из двух значений: 'ж','м'.
      2. Поля серия и номер паспорта, ИНН (sernum\_e, inn\_e, snils\_e) соответственно могут состоять только из цифр.
      3. Атрибут тип образования (type\_obr) может принимать одно из следующих значений: 'начальное', 'среднее', 'средне-специальное', 'высшее'.
      4. Атрибут масса упаковки (typeupak.weight\_t) должен указываться в кг.
      5. Масса нетто продукции(mas\_prod) должна быть равна суммарной массе() входящих поставок плюс масса упаковки(weight\_t). Все величины должны суммироваться и складываться в кг.

## **4.5. Описание групп пользователей и прав доступа**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Таблицы |  | | Группы пользователей (роли) | | | | |
| Руководители организации | Руководители  отделов с  производством | | Менеджеры производства | Сотрудники отд. кадров | Бухгалтеры | Отдел  закупок |
| Поставщики | SIUD |  | |  |  |  | S |
| Поставки фруктов | S | S | | S |  | S | SIUD |
| Виды фруктов | S |  | |  |  |  | SIUD |
| Страны | S |  | |  |  |  | SIUD |
| Единицы измерения | S | S | | SIUD |  | S | S |
| Состав | S | S | | SUID |  | S |  |
| Продукция | S | SUID | | S |  | S |  |
| Название продукции | SIUD | S | |  |  |  |  |
| Типы упаковки | S | S | | SUID |  |  |  |
| Отдел | SIUD | S | | S | S | S |  |
| Работники | S |  | |  | SIUD | S |  |
| Должности | SIUD |  | |  |  | S |  |
| Образование | S |  | |  | SIUD | S |  |

Где:

S – чтение данных (select);

I – добавление данных (insert);

U – модификация данных (update);

D – удаление данных(delete).

# 5. Реализация проекта базы данных

## **5.1. Создание таблиц**

1. Поставщики (Providers):

create table providers

(

id\_p numeric(3) primary key not null,

name\_p varchar(30) not null,

address\_p varchar(100) not null,

phone\_p varchar(10) not null,

contname\_p varchar(30)

);

2. Поставки фруктов (PostavkiFruits):

create table postavkifruits

(

id\_pf numeric(3) primary key not null,

name\_pf varchar(30) references typefruit,

datesbor\_pf date not null,

srok\_pf date not null,

weigth\_pf numeric(10,6) not null check,

edizm\_pf varchar(20) references edizm,

price\_pf numeric(10,2) not null,

country\_pf varchar(50) references countries,

postavshik\_pf numeric(3) references providers,

datepost\_pf date not null

);

3. Виды фруктов (TypeFruit):

create table typefruit

(

name\_f varchar(30) primary key not null

);

4. Страны (Countries):

create table countries

(

name\_count varchar(50) primary key not null

);

5. Единицы измерения(Edizm):

create table edizm

(

name\_ed varchar(20) primary key not null,

sistmer\_ed varchar(20) not null,

kg\_ed numeric(10,6) not null

);

6. Состав (Sostav):

create table sostav

(

id\_s numeric(3) primary key not null,

prod\_s numeric(3) references production,

fruct\_s numeric(3) references postavkifruits,

mas\_s numeric(10,6) not null,

edizm\_s varchar(20) references edizm

);

7. Продукция (Production):

create table production

(

id\_prod numeric(3) primary key not null,

name\_prod varchar(50) references nameprod,

type\_prod varchar(30) references typesupak,

date\_prod date not null,

srok\_prod date not null,

otd\_prod numeric(3) references otdels,

mas\_prod numeric(10,6) not null,

edizm\_prod varchar(20) references edizm

);

8. Названия продукции (NameProd):

create table nameprod

(

name\_pr varchar(50) primary key not null

);

9. Типы упаковки (TypeUpak):

create table typeupak

(

name\_t varchar(30) primary key not null,

weigth\_t numeric(10,6) not null

);

10. Отделы (Otdels):

create table otdels

(

id\_o numeric(3) primary key not null,

name\_o varchar(30) not null,

phone\_o varchar(10) not null,

spec\_o varchar(20) not null

);

11. Работники (Employees):

create table employees

(

id\_e numeric(3) primary key not null,

fname\_e varchar(25) not null,

lname\_e varchar(40) not null,

born\_e date not null,

sex\_e char(1) not null,

sernum\_e char(10) not null unique,

date\_e date not null,

given\_e varchar(50) not null,

inn\_e char(12) not null unique,

snils\_e char(14) not null unique,

address\_e varchar(100) not null,

phone\_e varchar(10),

otdel\_e numeric(3) references otdels,

dolgn\_e varchar(30) references dolgnosti,

login\_e varchar(15)

);

12. Должности (Dolgnosti):

create table dolgnosti

(

name\_dol varchar(30) primary key not null,

sal\_dol numeric(10,2) not null

);

13. Образоание (Obrazovanie):

create table obrazovanie

(

id\_obr numeric(3) references employees,

type\_obr varchar(15) not null,

spec\_obr varchar(30),

date\_obr date not null,

numdip\_obr numeric(15)

);

## **5.2. Создание представлений (готовых запросов)**

1. Данные о участниках отдела, где работает сотрудник:

create or replace view my\_kollegi( Familiya, Name, Dolgnost, Tel) as

select e1.fname\_e, e1.lname\_e, e1.dolgn\_e, e1.phone\_e

from employees e1

where exists (select \*

from employees e

where e.otdel\_e = e1.otdel\_e and e.login\_e = user());

2. Данные о других начальниках отделов, для начальника отдела:

create or replace view other\_nach( Num\_otdel,Familiya, Name, Tel) as

select e1.otdel\_e, e1.fname\_e, e1.lname\_e, e1.phone\_e/\*исключить некоторые поля\*/

from employees e1

where exists (select \*

from employees e

where e.otdel\_e <> e1.otdel\_e and e.login\_e = user()

and e.dolgn\_e = e1.dolgn\_e);

3. Поставщики, от которых не было поставок фруктов:

create or replace view no\_postavok(ID, NAME\_Provider) as

select p.id\_p, p.name\_p from providers p

where not exists (select \* from postavkifruits pf

where p.id\_p = pf.postavshik\_pf);

4. Продукция, в состав которой входит несколько фруктов и их количество:

create or replace view no\_single\_prod(ID, NAME\_Postavki, kol\_vo) as

select prod.id\_prod, prod.name\_prod, (select count(\*)

from sostav s

where prod.id\_prod = s.prod\_s) as cnt

from production prod

having cnt > 1;

5. Количество продукции по категории и названию:

create or replace view cnt\_production(kateg, name, kol\_vo) as

select o.spec\_o, prod.name\_prod, count(\*)

from production prod, otdels o

where prod.otd\_prod = o.id\_o

group by o.spec\_o, prod.name\_prod;

6. Статистика по странам стоимости поставок фруктов:

create or replace view sum\_postavki(country, cost) as

select pf.country\_pf , sum(pf.price\_pf)

from postavkifruits pf

group by pf.country\_pf;

7. Поставщики, поставляющие фрукты из разных стран:

create or replace view many\_countries(ID, Name, Phone, kolvo\_country) as

select p.id\_p, p.name\_p, p.phone\_p,

(select count(distinct pf.country\_pf)

from postavkifruits pf

where p.id\_p = pf.postavshik\_pf) as cnt\_country

from providers p

having cnt\_country > 1;

8. Сколько дней проходит с момента поставки фруктов до производства продукции:

create or replace view dneysr (ID\_prod, avg\_day) as

select prod.id\_prod, avg(datediff(prod.date\_prod, spf.d))

from production prod, (select

s.prod\_s as pr, pf.datepost\_pf as d

from sostav s, postavkifruits pf

where

s.fruct\_s = pf.id\_pf

) as spf

where spf.pr = prod.id\_prod

group by prod.id\_prod;

9. Самые дешёвые поставщики за 1 кг одной и той же поставки фруктов:

create or replace view cheaper\_postavki (id\_postav, name, postavshik, min\_price) as

select pf0.id\_pf, pf0.name\_pf, pf0.postavshik\_pf, mn.min\_pf

from postavkifruits pf0, (

select pf.name\_pf as name, min(pf.price\_pf / to\_kg(pf.weigth\_pf, pf.edizm\_pf)) as min\_pf

from postavkifruits pf

group by pf.name\_pf) as mn

where mn.name = pf0.name\_pf

and pf0.price\_pf / to\_kg(pf0.weigth\_pf, pf0.edizm\_pf) - mn.min\_pf <= 0.000001;

10. Остаток поставок (неиспользуемые для производства поставки):

create or replace view ostatki\_postavok(ID, Name, Weight) as

select p0.id\_pf, p0.name\_pf, to\_kg(p0.weigth\_pf, p0.edizm\_pf)

from postavkifruits p0

where not exists (

select \*

from sostav s

where s.fruct\_s = p0.id\_pf

)

union

select p0.id\_pf, p0.name\_pf, to\_kg(p0.weigth\_pf, p0.edizm\_pf) - p1.sum\_

from postavkifruits p0,

(select p.id\_pf as id\_pf, sum(to\_kg(s.mas\_s, s.edizm\_s)) as sum\_

from sostav s, postavkifruits p

where s.fruct\_s = p.id\_pf

group by p.id\_pf

) as p1

where p0.id\_pf = p1.id\_pf

and p1.sum\_ < to\_kg(p0.weigth\_pf, p0.edizm\_pf);

Примечание: в представлениях 9-10 используется функция to\_kg.

Права доступа к представлениям:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Представления |  | Группы пользователей (роли) | | | | | | |
| Руководители орган. | | Руководители  отделов с произв-ом | Менеджеры произв-ва | Рабочие отдела с произв-ом | Сотрудники отд. кадров | Бухгалтеры | Отдел  закупок |
| Данные об участниках отдела, где работает сотрудник  My\_kollegi | S | | S | S | S | S | S | S |
| Данные о других начальниках отделов, для начальника отдела  Other\_nach |  | | S |  |  |  |  |  |
| Поставщики, от которых не было поставок фруктов  no\_postavok | S | |  | S |  |  |  |  |
| Продукция, в состав которой входит несколько фруктов и их кол-во  no\_single\_prod | S | | S | S |  |  |  |  |
| Количество продукции по категории и названию  cnt\_production | S | | S | S |  |  |  |  |
| Статистика по странам стоимости поставок фруктов  sum\_postavki | S | | S | S |  |  |  | S |
| Поставщики, поставляющие фрукты из разных стран  many\_countries | S | |  | S |  |  | S | S |
| Сколько дней проходит с момента поставки фруктов до производства продукции  dneysr | S | |  | S |  |  |  |  |
| Самые дешевые поставщики за 1 кг одной и той же поставки фруктов  cheaper\_postavki | S | | S | S |  |  |  | S |
| Остаток поставок  ostatki\_postavok | S | | S | S |  | S | S | S |

## **5.3. Создание функции**

Функция перевода из одних единиц измерения в кг:

delimiter //

create function to\_kg (weigth\_ numeric(10,6), ed varchar(20))

returns numeric(10,6)

begin

declare perem numeric(10,6) ;

set perem = -1;

select edizm.kg\_ed into perem from edizm where ed = edizm.name\_ed;

if (perem = -1) then

signal sqlstate '45000' set

message\_text = 'нет такой ед.изм';

end if;

return perem\*weigth\_;

end; //

delimiter ;

## **5.4. Создание процедуры**

Создание этикетки для всех продукций с указанием даты по текущую дату, названия, состава, массы брутто, массы нетто:

delimiter //

create procedure etiketka(d date)

begin

declare mas numeric(10,6) default 0;

declare cnt numeric(5) default 0;

declare kat varchar(20);

declare etik text;

declare l\_eof bool default false;

declare otd\_prod1 numeric(3);

declare name\_prod1 varchar(50);

declare id\_prod1 numeric(3);

declare mas\_prod1 numeric(10,6);

declare date\_prod1 date;

declare edizm\_prod1 varchar(20);

declare crs1 cursor for select otd\_prod, name\_prod, id\_prod, mas\_prod, date\_prod, edizm\_prod

from production

where date\_prod between d and now();

declare continue handler for not found set l\_eof = true;

open crs1;

l1: loop

set etik = '';

fetch crs1 into otd\_prod1, name\_prod1, id\_prod1, mas\_prod1, date\_prod1, edizm\_prod1;

select o.spec\_o into kat from otdels o where o.id\_o = otd\_prod1;

set etik = concat\_ws(' ',etik,'-=Etiketka=- ', kat , name\_prod1);

set etik = concat\_ws('\n',etik, '-=Sostav:=-');

set mas = 0;

if l\_eof then

leave l1;

end if;

set cnt = cnt + 1;

begin

declare weigth\_pf1 numeric(10,6);

declare edizm\_s1 varchar(20);

declare name\_pf1 varchar(30);

declare crs2 cursor for select weigth\_pf, edizm\_s, name\_pf

from sostav s, postavkifruits pf

where id\_prod1 = s.prod\_s and pf.id\_pf = s.fruct\_s;

open crs2;

l2: loop

fetch crs2 into weigth\_pf1, edizm\_s1, name\_pf1;

if l\_eof then

leave l2;

end if;

set mas = mas + to\_kg(weigth\_pf1, edizm\_s1);

set etik = concat\_ws('\n',etik, name\_pf1);

end loop l2;

close crs2;

set l\_eof = false;

end;

set mas\_prod1 = to\_kg(mas\_prod1,edizm\_prod1);

set etik = concat\_ws('\n',etik, 'Massa netto kg:', mas\_prod1 - mas);

set etik = concat\_ws('\n',etik, 'Massa brytto kg:',mas\_prod1);

set etik = concat\_ws('\n',etik, 'Goden do:',date\_prod1);

select etik;

end loop l1;

set etik = concat\_ws('\_','After date: ', d, 'have', cnt, 'prodaction');

select etik;

close crs1;

end; //

delimiter ;

## **5.5. Создание триггеров**

1. Проверка наличия уже такого состава на тот же продукт и поставку:

delimiter //

create trigger sost

before

insert

on sostav

for each row

begin

DECLARE cnt1 numeric;

DECLARE cnt2 numeric;

DECLARE cnt3 numeric;

select count(\*) into cnt1

from postavkifruits as pf

where new.fruct\_s = pf.id\_pf;

select count(\*) into cnt2

from production as p

where new.prod\_s = p.id\_prod;

select count(\*) into cnt3

from sostav as s

where new.fruct\_s = s.fruct\_s and new.prod\_s = s.prod\_s;

if not(cnt1 > 0 and cnt2 > 0 and cnt3 = 0) then

signal sqlstate '45000' set

message\_text = 'такая запись уже есть(обновляйте значение массы)';

end if;

end; //

delimiter ;

2. Проверка корректности ввода данных работников:

delimiter //

create trigger proverka before insert on employees

for each row

begin

if((year(curdate()) - year(new.born\_e) -

(date\_format(curdate(),'00-%m-%d') < date\_format(new.born\_e,'00-%m-%d'))) < 16) then

signal sqlstate '45000' set

message\_text = 'Человеку меньше 16 лет';

end if;

if ((year(new.date\_e) - year(new.born\_e) -

(date\_format(new.date\_e,'00-%m-%d') < date\_format(new.born\_e,'00-%m-%d'))) < 14) then

signal sqlstate '45000' set

message\_text = 'Неверная дата выдачи паспорта';

end if;

set new.sex\_e := lower(new.sex\_e);

set new.fname\_e := upper(new.fname\_e);

set new.lname\_e := upper(new.lname\_e);

end; //

delimiter ;

3. Проверка корректности ввода данных поставок фруктов:

delimiter //

create trigger prov\_postavki before insert on postavkifruits

for each row

begin

if((datesbor\_pf >= now()) or (srok\_pf < datesbor\_pf) or (datepost\_pf <= datesbor\_pf and datepost\_pf >= now())) then

signal sqlstate '45000' set

message\_text = 'Неверно введена дата';

end if;

if(weigth\_pf <= 0) then

signal sqlstate '45000' set

message\_text = 'Неверно введен вес';

end if;

if(price\_pf < 0) then

signal sqlstate '45000' set

message\_text = 'Неверно введена цена';

end if;

end; //

delimiter ;

4. Проверка корректности ввода данных единиц измерения:

delimiter //

create trigger prov\_edizm before insert on edizm

for each row

begin

if(kg\_ed <= 0) then

signal sqlstate '45000' set

message\_text = 'Неверно введен коэффициент перевода в кг';

end if;

end; //

delimiter ;

5. Проверка корректности ввода данных состава:

delimiter //

create trigger prov\_sostav before insert on sostav

for each row

begin

if(mas\_s <= 0) then

signal sqlstate '45000' set

message\_text = 'Неверно введена масса';

end if;

end; //

delimiter ;

6. Проверка корректности ввода данных типы упаковки:

delimiter //

create trigger prov\_typeupak before insert on typeupak

for each row

begin

if(weigth\_t <= 0) then

signal sqlstate '45000' set

message\_text = 'Неверно введен вес';

end if;

end; //

delimiter ;

7. Проверка корректности ввода данных продукции:

delimiter //

create trigger prov\_product before insert on production

for each row

begin

if((date\_prod > now()) or (srok\_prod <= date\_prod)) then

signal sqlstate '45000' set

message\_text = 'Неверно введена дата';

end if;

if(mas\_prod <= 0) then

signal sqlstate '45000' set

message\_text = 'Неверно введена масса';

end if;

end; //

delimiter ;

8. Проверка корректности ввода данных должностей:

delimiter //

create trigger prov\_dolgnost before insert on dolgnosti

for each row

begin

if(sal\_dol<6250) then

signal sqlstate '45000' set

message\_text = 'Неверно введена заработная плата';

end if;

end; //

delimiter ;

## **5.6. Создание индексов**

Для повышения эффективности работы с данными нами не создавались индексы, кроме автоматических индексов первичных ключей и уникальных полей таблиц. Другие индексы создаются администратором базы данных после анализа запросов пользователей.

## **5.7. Разработка стратегии резервного копирования**

Интенсивность обновления разработанной базы данных производства фруктовой продукции довольно низкая, поэтому для обеспечения сохранности вполне достаточно проводить полное резервное копирование базы данных раз в день. Для разработанной базы данных необходимо держать сервер включенным круглосуточно, так как менеджеры производства должны иметь круглосуточный доступ к базе данных, ведь производство идёт без остановки.

# 6. Список литературы

1. Карпова И.П. «Проектирование реляционных баз данных: Метод. указания к курсовому проектированию по курсу «Базы данных». – М., МИЭМ, 2010.

2. Карпова И.П. «Учебное пособие по курсу «Базы данных». – М., РИО МГИЭМ, 2009.

3. Справочное руководство по MySQL, издание 306, 2010.