МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

ЧЕРНИГОВСКИЙ НАЦИОНАЛЬНЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

КАФЕДРА ИНФОРМАЦИОННЫХ И КОМПЬЮТЕРНЫХ СИСТЕМ

**БД «УНИВЕРСИТЕТ»**

Курсовая работа по дисциплине   
«Организация баз данных»

|  |  |  |
| --- | --- | --- |
| Исполнитель  студент гр. КИ-122 |  | А.Ю. Магдыбор |
| Руководитель  ассистент |  | М.В. Харченко |

**2015**

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

на выполнение курсового проекта

по дисциплине «Организация баз данных»

**Магдыбор А.Ю., гр. КИ-122**

**Тема работы:** БД “Университет”

**Предполагаемые технические и эксплуатационные результаты работы:**

Данная работа представляет собой разработку полноценной БД, которая размещается на сервере PostgreSQL и используется при разработке информационно-компьютерной системы в архитектуре клиент-сервер.

Разработка БД должна включать:

1. построение концептуальной модели предметной области, содержащей не менее 4-х объектов, связанных между собой взаимосвязями типа 1:М и M:N, в виде диаграммы объектов языка UML и ее описание;
2. выполнение нормализации схемы БД (не менее 5-ти таблиц) с указанием:
   1. первичных и внешних ключей;
   2. типов данных атрибутов;
   3. индексов;
3. реализацию на уровне структуры БД средств обеспечения целостности данных:
   1. уникальность и обязательность ввода первичных ключей;
   2. поддержка целостности для внешних ключей (каскадное удаление и обновление);
   3. значения атрибутов по умолчанию и обязательность ввода значений атрибутов;
4. на уровне бизнес-логики обеспечение ограничений на значения данных атрибутов вида: "интервал", "перечислимое значение" и "сравнение значений двух атрибутов одной таблицы";
5. реализацию запросов к БД на поиск, обновление и удаление данных;
6. реализацию не мене 2-х хранимых процедур;
7. реализацию не мене 3-х триггеров для Insert, Update и Delete;
8. формирование не менее 2-х вариантов представлений;
9. реализацию назначение, проверку и отмену прав доступа.

Реализация пользовательского интерфейса предполагает обязательное наличие форм для просмотра, добавления, редактирования и удаления записей всех таблиц.

**Предполагаемая трудоёмкость работы:** 45-50 человеко-часов.

**Объем текстовой и графической документации:**

Пояснительная записка к проекту объёмом over9000 страниц печатного текста формата А4 и программная документация на систему объёмом over9000 страниц печатного текста формата А4. Объёмы текстовой информации могут быть скорректированы в процессе работы по согласованию с руководителем. Проект оформляется согласно СОККР-ИКС-2001.

**Внедрение и публикации:**

**Плановые сроки по этапам:**

1. Согласование задания 06.02.15
2. Анализ предметной области. Разработка схемы БД 20.02.15
3. Прототип пользовательского интерфейса 06.03.15
4. Классы домена (диаграмма классов + реализация на Java) 20.03.15
5. Классы для работы с БД. (диаграмма классов + реализация на Java) 10.04.15
6. WEB-интерфейс (HTML+CSS+JSP) 24.04.15
7. ANT скрипт сборки проекта. Демонстрация проекта 13.05.15
8. Оформление работы 25.05.15

Демонстрация работоспособности программного проекта до 13.05.15

**Плановый срок защиты проекта: 26.05.15**

Исполнитель работы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Магдыбор А.Ю.,

Руководитель работы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания: « » \_\_\_\_\_\_\_\_\_\_\_2015 г.

Реферат

Реферат

THE ABSTRACT

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 8](#_Toc410595277)

[1 АНАЛИЗ РЕШАЕМОЙ ЗАДАЧИ 9](#_Toc410595278)

[1.1 Анализ предметной области 9](#_Toc410595279)

[1.2 Цели и задачи системы 10](#_Toc410595280)

[1.3 Назначение системы 10](#_Toc410595281)

[1.4 Требования к системе 11](#_Toc410595282)

[2 Проектирование системы 12](#_Toc410595283)

[2.1 Выбор инструментальных средств разработки системы 12](#_Toc410595284)

[2.1.1 Сервер баз данных 12](#_Toc410595285)

[2.1.2 Технологии реализации системы 12](#_Toc410595286)

[2.2 Проектирование архитектуры системы 12](#_Toc410595287)

[2.2.1 Проектирование слоя бизнес-логики и бизнес-правил 12](#_Toc410595288)

[2.2.2 Проектирование слоя доступа к данным 13](#_Toc410595289)

[2.2.3 Проектирование слоя отображения 13](#_Toc410595290)

[3 Разработка 14](#_Toc410595291)

[3.1 Разработка базы данных системы 14](#_Toc410595292)

[3.1.1 Разработка схемы базы данных 14](#_Toc410595293)

[3.1.2 Обеспечение целостности данных 15](#_Toc410595294)

[3.1.3 Разработка базовых запросов 16](#_Toc410595295)

[3.1.4 Выбор индексов 21](#_Toc410595296)

[3.1.5 Разработка хранимых процедур и триггеров 22](#_Toc410595297)

[3.1.6 Разработка представлений 23](#_Toc410595298)

[3.1.7 Организация защиты данных 24](#_Toc410595299)

[3.1.8 Объектно-реляционное отображение 24](#_Toc410595300)

[3.2 Разработка модулей системы 25](#_Toc410595301)

[3.2.1 Разработка модулей слоя бизнес логики и бизнес правил 25](#_Toc410595302)

[3.2.2 Разработка модулей слоя доступа к данным 25](#_Toc410595303)

[3.2.3 Разработка модулей слоя отображения 26](#_Toc410595304)

[ВЫВОДЫ 27](#_Toc410595305)

[ПЕРЕЧЕНЬ ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 28](#_Toc410595306)

[ПРИЛОЖЕНИЯ 29](#_Toc410595307)

ВВЕДЕНИЕ

Актуальность темы, необходимость в создании программной системы (корпоративного приложения).

# АНАЛИЗ РЕШАЕМОЙ ЗАДАЧИ

## Анализ предметной области

Деревообрабатывающая промышленность является очень важной частью народного хозяйства Украины, предприятия занимающиеся заготовкой леса производят около 7% общего ВВП этой страны.

Типичное деревообрабатывающее предприятие закупает древесину у лесозаготовительных лесных хозяйств. Лесозаготовительные хозяйства зачастую поставляют древесину большими партиями, по несколько различных видов древесины за одну поставку.

Древесина бывает различных разнообразных типов в зависимости от различных древесных растений из которых она была получена. Основными классами древесины является хвойная, лиственная, а также экзотическая древесина, получаемая из деревьев растущих в Африке, Австралии и Южной Америке. Основное отличие хвойной от лиственной древесины это устойчивость к гниению, хвойная древесина зачастую менее устойчива к гниению, поэтому ее рекомендуется хранить отдельно от лиственной.

После получения древесина хранится на складе до начала ее обработки. Древесина обрабатывается в несколько этапов – сперва древесину вымачивают, а также обрабатывают антисептиком, что позволяет существенно увеличить ее прочность, а также увеличить сопротивляемость грибкам, насекомым и влаге. Следующим этапом является распиливание дерева и получение пиломатериалов. После распиливания дерева получают разнообразные пиломатериалы, такие как брус, брусок, доска, шпалы, обаполь и прочие. Также из отходов производства производят такую продукцию как шпон, фанеру, древесно-слоистый пластик (ДСП), древесно волокнистую плиту (ДВП) и прочие композитные материалы.

Следующим этапом деревообработки является хранение полученных пиломатериалов на складе. При правильном хранении древесина может хранится очень долго, для хранения и атмосферной сушки пиломатериалов существует ГОСТ 3808.1-80.

Готовые пиломатериалы очень широко применяются в народном хозяйстве – от создания мебели и построения жилья – до судостроения и авиастроения. Пиломатериалы зачастую покупают крупными партиями, в которые входят различные изделия.

В данной системе принимают участие такие объекты: изделие, лесничий, работник пилорамы, поставка. С данной информационной системой должны участвовать следующие объекты: изделие и древесина.

В системе ведутся учетные записи, поэтому по прохождению некоторого интервала времени создается рейтинг лесников по количеству их поставок. Это дает возможность проследить активность лесников и принять определенные меры.

## Цели и задачи системы

Целью системы управления пилорамой является автоматизация работы этой системы, а именно выдача и контроль заданий, учет лесников, закупаемой древесины и изделий. С помощью автоматизации регулируется вся работа заведения, составляется отчет о активности лесников, о выработке преподавателей, также составляется расписание работы всех участников данной системы. Благодаря ей упрощаеться как администрирование данным сервисом, так и взаимодействие с ним с клиентской стороны благодаря интуитивно понятному интерфейсу.

## Назначение системы

Данной системой будут пользоваться лесники, руководство и рабочие на пилораме. Она позволяет сделать улучшить процесс работы пилорамы, так как упрощается организация закупки древесины и производства изделий.

Система предназначена для автоматизации организации учебного процесса, а также для удобного взаимодействия преподавателей и студентов.

Лесникам предоставлены следующие возможности как:

* Ознакомится с закупаемой древесиной;
* Продать древесину;
* Зарегистрироватся в системе;
* Посмотреть прошлые поставки;

Предполагается, что с данным корпоративным приложением будут

работать сотрудники пилорамы, и они должны иметь возможность решать с его помощью следующие задачи:

* Сделать изделие;
* Посмотреть наявные изделия;
* Посмотреть наявную древесину;

Также в системе будет администратор, в обязанности которого входит следующее:

* Добавлять производимое изделие;
* Регистрировать пользователей;
* Добавлять закупаемую древесину;

На рисунке 1. 1 приведена диаграмма вариантов использования.

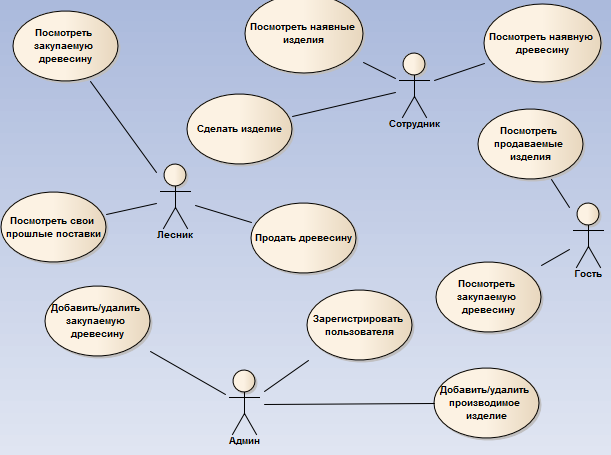


Рисунок 1 – Диаграмма вариантов использования

## Требования к системе

Разрабатываемая в рамках данной курсовой работы система «Пилорама» должна оперировать следующими объектами: лесничий, рабочий пилорамы, изделие, поставка, древесина.

Нужно разработать такую систему в которой бы и лесничему и было удобно выполнять свои функции. А именно должно быть удобно просматривать список закупаемых типов древесины, список поставок леснику, а также просматривать и изменять информацию о наявной древесине и производимых изделиях работникам пилорамы.

Система должна: предоставлять возможность руководству пилорамы управлять закупкой древесины и производством новых изделий. Они имеют право изменять, добавлять и удалять информацию в этих таблицах. Также руководство добавляет новых лесников и сотрудников в базу данных.

Лесники и простые рабочие не допускаються к администрированию информации. Лесники могут зарегистрироваться и просматривать необходимую им информацию: список закупаемой древесины и список их прошлых поставок, а также совершать новые поставки. Рабочие пилорамы могут изменять наявное количество древесины на складе, а также изменять количество наявных изделий определенного типа. Также система автоматически учитывает, что лесник, который продал 10 кубометров леса продает 11 кубометр бесплатно.

На объекты и правила взаимодействия между объектами в системе накладываются определенные ограничения, совокупность которых называется бизнес-логикой.

# Проектирование системы

## Выбор инструментальных средств разработки системы

### Сервер баз данных

Следует провести сравнение тех серверов баз данных, которые можно было бы

использовать для выполнения данной

работы. Из них выбрать какой-то один и

описать причину выбора (почему выбрали).

### Технологии реализации системы

Следует кратко описать используемые технологии и что они позволяют:

J2EE: Java Server Pages 2.069

Java Servlet 2.5

Java Server Pages Standard Tag Library

JPA

## Проектирование архитектуры системы

// раздел 1 методы

### Проектирование слоя бизнес-логики и бизнес-правил

##### Определение объектов системы

Описание классов домена, определение

полей и методов.

##### Проектирование бизнес-правил

Словесное описание алгоритмов, схемы

алгоритмов. Приветствуется использование

либо диаграммы деятельности (Activity),

либо диаграммы состояний

(State).

### Проектирование слоя доступа к данным

##### Проектирование фабрики DAO

Описание структуры DAO. В ходе

выполнения курсовой работы нужно было

разобраться с готовой реализацией данного

слоя и переделать на свою реализацию. В

данном пункте необходимо описать то, что

получилось и привести диаграмму классов.

##### Проектирование интерфейсов DAO

Диаграмма классов. Описание каждого

интерфейса и его назначение.

### Проектирование слоя отображения

Подробное

описание

нижеследующих

пунктов. Вынести диаграммы, которые

сдавались как этап курсовой работы.

##### Проектирование веб-страниц

##### Навигация между страницами

# Разработка

Разработка – это центральный раздел, в котором описывается вся основная работа. Происходит выбор модели данных, разрабатывается логическая структура данных и модулей системы.

## Разработка базы данных системы

### Разработка баз данных — процесс создания схемы [базы данных](http://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) и определения необходимых ограничений [целостности](http://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D0%BE%D1%81%D1%82%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85).

### Разработка схемы базы данных

Исходя из анализа и проектирования слоя бизнес-логики и правил была построена структура БД. На рисунке 3.2 приведена концептуальная модель базы данных, а на рисунке 3.3 – физическая модель базы данных.

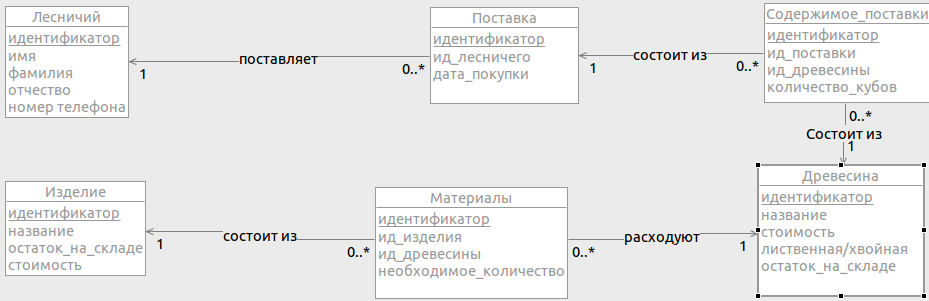


Рисунок 3.1 – Концептуальная модель базы данных

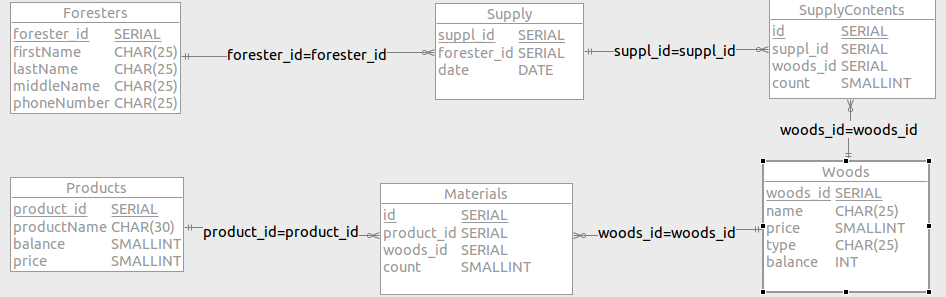


Рисунок 3.2 – Физическая модель базы данных

В базе данных программной системы содержится вся информация относительно её объектов, а именно о :

- леснике;

- поставке;

- древесине;

- содержимом поставки;

- материале изделия;

- изделии;

Между объектами установлены такие связи:

* ЛЕСНИЧИЙ(поставляет) ПОСТАВКА(степень связи N:1);
* ПОСТАВКА(состоит из) ДРЕВЕСИНА(степень связи M:N);
* ИЗДЕЛИЕ(состоит из) ДРЕВЕСИНА(степень связи M:N);

В каждой таблице уникальный первичный ключ является суррогатным. Это дополнительное служебное поле, добавленное к уже имеющимся информационным полям таблицы, единственное предназначение которого — служить первичным ключом. Значения этого поля не образуется на основе каких-либо других данных из БД, а генерируются искусственно. Главное достоинство суррогатного ключа состоит в том, что он никогда не изменяется, поскольку не является информативным полем таблицы (не несёт никакой информации об описываемом записью объекте). Использовать суррогатный ключ имеет смысл в случае, когда возможны изменения полей, составляющих (естественный) первичный ключ (в особенности если этот ключ — составной). В этом случае возникает проблема так называемых "каскадных изменений". При использовании же суррогатного ключа в качестве первичного, изменять его не придётся. Также, при выполнении запросов, использующих суррогатные ключи, сравнение полей будет происходить быстрее, в особенности, если естественный первичный ключ представляет собой строку.

### Обеспечение целостности данных

Обеспечение целостности представляют собой утверждения о допустимых значениях отдельных информационных единиц и связях между ними.

Обеспечение целостности могут относиться к разным информационным объектам: атрибутам, кортежам, отношениям, связям между ними и тому подобное.

Для полей (атрибутов) используются следующие виды ограничений:

* тип и формат поля;
* задание диапазона значений;
* недопустимость пустого поля;
* задание домена;
* проверка на уникальность значения какого-либо поля. Ограничение

позволяет избежать записей-дубликатов.

Ограничения целостности нашей базе данных описаны в таблицах 3.1-3.6.

Таблица 3.1 – Описание таблицы «Лесничий»

|  |  |  |  |
| --- | --- | --- | --- |
| Поле | Описание | Тип данных | Ограничения |
| forester\_id | Идентификационный код лесника | SERIAL | Первичный ключ |
| FirstName | Имя лесника | CHAR(25) | Обязательное для ввода |
| LastName | Фамилия лесничего | CHAR(25) | Обязательное для ввода |
| PhoneNumber | Телефон лесника | CHAR(25) |  |
| MiddleName | Отчество лесника | CHAR(25) | Обязательное для ввода |

Таблица 3.2 – Описание таблицы «Поставки»

|  |  |  |  |
| --- | --- | --- | --- |
| Поле | Описание | Тип данных | Ограничения |
| suppl\_id | Идентификатор поставки | SERIAL | Первичный ключ |
| forester\_id | Идентификатор лесника | SERIAL | внешний ключ, запрет обновления, запрет удаления;  обязательное для ввода. |
| date | Дата поставки | DATE | Default: CURRENT\_DATE |

Таблица 3.3 – Описание таблицы «Содержимое поставки»

|  |  |  |  |
| --- | --- | --- | --- |
| Поле | Описание | Тип данных | Ограничения |
| id | Идентификатор | SERIAL | Первичный ключ |
| suppl\_id | Идентификатор поставки | SERIAL | внешний ключ, запрет обновления, запрет удаления;  обязательное для ввода. |
| woods\_id | Идентификатор древесины | SERIAL | внешний ключ, запрет обновления, запрет удаления;  обязательное для ввода. |
| Count | Количество кубометров данной древесины в поставке | SMALLINT | Ограничение:  count > 0 and  count < 1000  NOT NULL |

Таблица 3.4 – Описание таблицы «Древесина»

|  |  |  |  |
| --- | --- | --- | --- |
| Поле | Описание | Тип данных | Ограничения |
| woods\_id | Идентификатор | SERIAL | Первичный ключ |
| name | Название вида древесины | CHAR(25) | Обязательно ко вводу, уникальное |
| Type | Тип древесины | CHAR(25) | Обязательно ко вводу, type = 'хвойная' OR type = 'лиственная' |
| Price | Cтоимость одного кубометра данного типа древесины | SMALLINT | Обязательно ко вводу, price > 0  NOT NULL |
| balance | Количество кубометров данной древесины на складе | INT | Ограничение:  balance > 0 and  balance < 1000  NOT NULL |

Таблица 3.5 – Описание таблицы «Материалы»

|  |  |  |  |
| --- | --- | --- | --- |
| Поле | Описание | Тип данных | Ограничения |
| id | Идентификатор | SERIAL | Первичный ключ |
| product\_id | Идентификатор изделия | SERIAL | внешний ключ, запрет обновления, запрет удаления;  обязательное для ввода. |
| woods\_id | Идентификатор древесины | SERIAL | внешний ключ, запрет обновления, запрет удаления;  обязательное для ввода. |
| count | Количество кубометров данной древесины необходимой для производства изделия | SMALLINT | Ограничение:  count > 0 and  count < 1000  NOT NULL |

Таблица 3.6 – Описание таблицы «Изделия»

|  |  |  |  |
| --- | --- | --- | --- |
| Поле | Описание | Тип данных | Ограничения |
| product\_id | Идентификатор | SERIAL | Первичный ключ |
| name | Название вида изделия | CHAR(25) | Обязательно ко вводу, уникальное |
| Price | Cтоимость одного изделия | SMALLINT | Обязательно ко вводу, price > 0  NOT NULL |
| balance | Количество изделий на складе | SMALLINT | Ограничение:  balance > 0 and  balance < 1000  NOT NULL |

### Физическое проектирование базы данных

**DROP** **INDEX** IX\_Woods\_1;

**DROP** **INDEX** IX\_Product\_1;

**DROP** **INDEX** IX\_Foresters\_1;

**DROP** **INDEX** IX\_Foresters\_2;

**DROP** **TABLE** Materials;

**DROP** **TABLE** SupplyContents;

**DROP** **TABLE** Supply;

**DROP** **TABLE** Foresters;

**DROP** **TABLE** Products;

**DROP** **TABLE** Woods;

**CREATE** **TABLE** Woods (

woods\_id SERIAL **NOT** **NULL**

, name **CHAR**(25) **NOT** **NULL** **CONSTRAINT** UQ\_Woods\_1 **UNIQUE**

, price **SMALLINT** **DEFAULT** '0' **NOT** **NULL**

, **type** **CHAR**(25) **NOT** **NULL**

, balance **INT** **DEFAULT** 0

, **PRIMARY** **KEY** (woods\_id)

);

**CREATE** **TABLE** Products (

product\_id SERIAL **NOT** **NULL**

, productName **CHAR**(30) **NOT** **NULL** **CONSTRAINT** UQ\_Product\_1 **UNIQUE**

, balance **SMALLINT** **DEFAULT** '0'

, price **SMALLINT** **DEFAULT** '0' **NOT** **NULL**

, **PRIMARY** **KEY** (product\_id)

);

**CREATE** **TABLE** Foresters (

forester\_id SERIAL **NOT** **NULL**

, firstName **CHAR**(25) **NOT** **NULL**

, lastName **CHAR**(25) **NOT** **NULL**

, middleName **CHAR**(25) **NOT** **NULL**

, phoneNumber **CHAR**(25) **DEFAULT** '111-111-11'

, **PRIMARY** **KEY** (forester\_id)

);

**CREATE** **INDEX** IX\_Foresters\_1 **ON** Foresters (firstName, lastName, middleName);

**CREATE** **INDEX** IX\_Foresters\_2 **ON** Foresters (lastName);

**CREATE** **TABLE** Supply (

suppl\_id SERIAL **NOT** **NULL**

, forester\_id SERIAL **NOT** **NULL**

, **date** **DATE** **DEFAULT** CURRENT\_DATE **NOT** **NULL**

, **PRIMARY** **KEY** (suppl\_id)

, **CONSTRAINT** FK\_Supply\_1 **FOREIGN** **KEY** (forester\_id)

**REFERENCES** Foresters (forester\_id) **ON** **DELETE** **RESTRICT** **ON** **UPDATE** **RESTRICT**

);

**CREATE** **TABLE** SupplyContents (

id SERIAL **NOT** **NULL**

, suppl\_id SERIAL **NOT** **NULL**

, woods\_id SERIAL **NOT** **NULL**

, count **SMALLINT** **DEFAULT** '0' **NOT** **NULL**

, **PRIMARY** **KEY** (id)

, **CONSTRAINT** "FK\_Содержимое\_поставки\_1" **FOREIGN** **KEY** (suppl\_id)

**REFERENCES** Supply (suppl\_id) **ON** **DELETE** **RESTRICT** **ON** **UPDATE** **RESTRICT**

, **CONSTRAINT** "FK\_Содержимое\_поставки\_2" **FOREIGN** **KEY** (woods\_id)

**REFERENCES** Woods (woods\_id) **ON** **DELETE** **RESTRICT** **ON** **UPDATE** **RESTRICT**

);

**CREATE** **TABLE** Materials (

id SERIAL **NOT** **NULL**

, product\_id SERIAL **NOT** **NULL**

, woods\_id SERIAL **NOT** **NULL**

, count **SMALLINT** **DEFAULT** '0' **NOT** **NULL**

, **PRIMARY** **KEY** (id)

, **CONSTRAINT** FK\_Materials\_1 **FOREIGN** **KEY** (product\_id)

**REFERENCES** Products (product\_id) **ON** **DELETE** **RESTRICT** **ON** **UPDATE** **RESTRICT**

, **CONSTRAINT** FK\_Materials\_2 **FOREIGN** **KEY** (woods\_id)

**REFERENCES** Woods (woods\_id) **ON** **DELETE** **RESTRICT** **ON** **UPDATE** **RESTRICT**

);

Триггер для добавления:

При добавлении записей в таблицу «Содержимое поставки» должна выполнятся проверка есть ли в данной поставке древесина иного класса, если есть то отображается сообщение об ошибке ввода.

Триггер для изменения:

При увеличении значения в столбце «количество» в таблице «Изделия» должно происходить уменьшения количества древесины, указанной в таблице «Материалы», в таблице «Древесина» пропорционально изменению количества изделий.

Триггер для удаления:

При удалении из таблицы «Поставки» необходимо также удалить все записи в таблице «Содержимое поставки» которые ссылаются на данную таблицу.

Хранимые процедуры:

1) Процедура findMostUsedWoods

Входные параметры:

Выходные параметры: таблица со столбцами woods\_id, Woods.name, count.

Процедура находит наиболее часто используемую древесину.

2) Процедура findMostActiveForester

Входные параметры: –

Выходные параметры: таблица со столбцами forester\_id, Forester.lastName, count.

Процедура находит всех лесников и количество их поставок в порядке убывания.

### Разработка базовых запросов

Определить и дать полное описание базовых запросов на выборку, обновление и удаление данных. Привести их листинги на SQL.

### Выбор индексов

Для удобного использования системы стоит использовать поля fio (Ф.И.О) и id\_tabel (табельный номер) из таблицы Teach (Преподаватель). Так будет удобно осуществлять поиск преподавателя, для проведения нужных модификаций в таблице.

**DROP** **INDEX** indTeach;

Также для быстрого и точного поиска студента был создан индекс indStud. В этом индексе объеденены такие поля как id\_z(№зачетки)и fio (Ф.И.О) из таблицы Student (Студент).

**DROP** **INDEX** indStud;

### Разработка хранимых процедур и триггеров

Дать полное описание хранимых процедур и

триггеров БД. Привести реализацию на SQL.

**3.1.6.1 Описание хранимых процедур**

1. Процедура teach.

Входные параметры: нет

Выходные параметры: таблица со столбцами ФИО и стаж\_работы данного преподавателя.

Процедура осуществляет выбор 10 преподавателей, у которых стаж работы самый большой.

СREATE OR REPLACE FUNCTION teach(OUT "fio" character varying, OUT storage\_life int)

RETURNS SETOF record AS

$BODY$

select teach. fio, max(teach. storage\_life)

from teach

order by max desc

limit 10;

$BODY$

LANGUAGE 'sql' VOLATILE

COST 100

ROWS 1000;

ALTER FUNCTION teach() OWNER TO postgres;

2) Процедура findStudent

Входные параметры: №зачетки

Выходные параметры: таблица со столбцами id\_z и fio.

Процедура осуществляет поиск студента по заданному №зачетки .

СREATE OR REPLACE FUNCTION findStudent(IN id\_z serial, OUT fio character varying)

RETURNS SETOF record AS

$BODY$

select student. fio

from student

where $1=id\_z

$BODY$

LANGUAGE 'sql' VOLATILE

COST 100

ROWS 1000;

ALTER FUNCTION findStudent() OWNER TO postgres;

Вызов SELECT \* FROM Student('125515');

#### Описание триггеров

1. AFTER INSERT ON Student: при добавлении новой группы, курсу присвоить значение 1 . Реализация триггера на SQL:

CREATE OR REPLACE FUNCTION trig\_kurs()

RETURNS trigger AS

$BODY$BEGIN

NEW.course=’1’;

RETURN NEW;

END; $BODY$

LANGUAGE 'plpgsql' VOLATILE

COST 100;

ALTER FUNCTION trig\_kurse () OWNER TO postgres;

2) INSERT/ UPDATE INTO student

Если средний балл в таблице student >=4.5, то обновить поле stip, а именно увеличить значение на 20%.

CREATE OR REPLACE FUNCTION () stud\_trig RETURNS trigger AS

$$BEGIN

IF NEW.sr\_ball >=4.5 Then

NEW.stip= stip\*0.2;

END IF;

RETURN NEW;

LANGUAGE 'plpgsql' VOLATILE

COST 100;

ALTER FUNCTION stud\_trig () OWNER TO postgres;

1. AFTER INSERT ON Student: при добавлении нового студента группы, увеличить количество студентов на 1 . Реализация триггера на SQL:

CREATE OR REPLACE FUNCTION trig\_cnt()

RETURNS trigger AS

$BODY$BEGIN

NEW.cnt\_stud=cnt\_stud+1;

RETURN NEW;

END; $BODY$

LANGUAGE 'plpgsql' VOLATILE

COST 100;

ALTER FUNCTION trig\_cnt() OWNER TO postgres;

### Разработка представлений

Дать полное описание представлений БД.

Привести реализацию на SQL.

### Организация защиты данных

Пользователями БД в приведенном примере могут быть: администратор БД, сотрудник пилорамы, лесник и гость. Опишем для каждой группы пользователей права доступа к каждой таблице и к каждому полю (атрибуту). Администратор БД имеет доступ ко всем данным (по записи), может изменять структуру базы данных и связи между отношениями, устанавливает права доступа для всех остальных групп.

Лесник имеет возможность добавлять и просматривать информацию о поставках, увеличивать количество древесины, а также смотреть закупаемую древесину и продаваемые материалы

Сотрудник может просматривать наявную древесину, а также уменьшать ее количество, также он может просматривать изготовляемые изделия и увеличивать их количество.

Гость на веб-сайте может просматривать закупаемую древесину, а также просматривать изготовляемые изделия.

Перечисленные права доступа удобно представить в виде таблицы 3.2.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Роль | Сотрудник | Лесник | Поставка | Содержимое поставки | Древесина | Изделия | Материалы |
| Администраор | S,U,D,I | S,U,D,I | S,U,D,I | S,U,D,I | S,U,D,I | S,U,D,I | S,U,D,I |
| Лесник | - | - | S,I | S,I | S,U | S | - |
| Сотрудник | - | - | - | - | S,U | S,U | - |
| Гость | - | - | - | - | S | S | - |

В таблице S –select, U –update, D –delete, I –insert.

### Объектно-реляционное отображение

Следует описать основные структуры

используемые в файлах sql-маппинга:

собственно описание отображения объектов

на таблицы, входные-выходные параметры

при описании SQL запросов. Полный

листинг файлов маппинга вынести в

приложение.

При разработке программы осуществляется доступ к данным, поэтому для упрощения разработки такой программы и для увеличения эффективности и скорости работы с полученными данными использовалось объектно-реляционное отображение и его реализация на OpenJPA.

Системы управления реляционными базами данных показывают хорошую производительность на глобальных запросах, которые затрагивают большой участок базы данных, но объектно-ориентированный доступ более эффективен при работе с малыми объёмами данных, так как это позволяет сократить семантический провал между объектной и реляционной формами данных.

Все системы ORM обычно проявляют себя в том или ином виде, уменьшая в некотором роде возможность игнорирования базы данных. Более того, слой транзакций может быть медленным и неэффективным (особенно в терминах сгенерированного SQL). Но ORM избавляет программиста от написания большого количества кода, часто однообразного и подверженного ошибкам, тем самым значительно повышая скорость разработки. Кроме того, большинство современных реализаций ORM позволяют программисту при необходимости самому жёстко задать код SQL-запросов, который будет использоваться при тех или иных действиях (сохранение в базу данных, загрузка, поиск и.т.д.) с постоянным объектом.

## Разработка модулей системы

В разделе проектирования приводятся диаграммы классов, а в разделах

разработки приводятся классы реализации

на Java и их описание.

### Разработка модулей слоя бизнес логики и бизнес правил

Код реализации ограничений целостности

БД на Java с подробными комментариями.

Код

Реализациивызовов хранимых процедур на Java с подробными комментариями.

### Разработка модулей слоя доступа к данным

// тут

### Разработка модулей слоя отображения

// и тут будет неведомая хуйня

ВЫВОДЫ

Написать, что было сделано в результате

выполнения курсовой работы, что не

удалось

сделать,

как

можно

усовершенствовать приложение.

ПЕРЕЧЕНЬ ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Минимум 10 источников

ПРИЛОЖЕНИЯ