МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Факультет информационных систем и технологий

Кафедра «Вычислительная техника»

К защите допустить

Зав. кафедрой ВТ

\_\_\_\_\_\_\_\_\_ К.В. Святов

« » июня 2022г.

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к выпускной квалификационной работе**

на тему: ***«*Автоматизация бизнес-процессов кафедры по реализации учебных планов вуза*»***

Направление:09.04.01 «Информатика и вычислительная техника»

Выполнил студент гр. ИВТАCмд-21 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кондратьев П.С.

Научный руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Негода В.Н., профессор

**Ульяновск – 2022**

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

09.04.01 «Информатика и вычислительная техника»

*(шифр и наименование направления)*

«Автоматизированные системы»

*(профиль направления подготовки)*

Квалификация *Магистр*

*(бакалавр/магистр/специалист)*

Факультет Информационных систем и технологий

*(наименование факультета, где осуществляется обучение по направлению/магистерской программе/специальности)*

**УТВЕРЖДАЮ**

**Заведующий кафедрой**

«Вычислительная техника»

(наименование кафедры)

*к.т.н., доцент Святов К.В.*

(ученая степень, ученое звание, Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

«\_\_\_\_» июня 2022г.

**ЗАДАНИЕ**

**на выполнение** *выпускной работы магистра*

(наименование выпускной квалификационной работы)

Студенту группы **ИВТАСмд-21** *очной* формы обучения

*Кондратьеву Павлу Сергеевичу*

(фамилия, имя, отчество)

1. Тема: «*Автоматизация бизнес-процессов кафедры по реализации учебных . планов вуза*» . (наименование ВКР)

утверждена приказом по Университету от «\_\_\_»\_\_\_\_\_\_\_\_\_202\_г. № \_\_\_\_\_

2. Срок сдачи студентом *выпускной работы магистра* «\_\_\_\_» июня 2022г.

(наименование ВКР)

3. Цели и задачи:

*Автоматизация бизнес-процессов кафедры по реализации учебных планов вуза.*

\* Название темы указывается в точном соответствии с приказом.

4. Исходные данные:

* *Расчет штатов ВТ;*
* *Нормы времени;*
* *Индивидуальный план преподавателя;*
* *Рабочие программы прошлых лет*
* *Система учебных планов ВУЗа*
* *Учебный план в формате plx*

5. Перечень вопросов, подлежащих разработке, или краткое содержание

*выпускной работы магистра*:

(наименование ВКР)

* *расширенная постановка задачи;*
* *проектирование системы;*
* *реализация системы.*

6. Перечень иллюстративного материала:

*Инфологическая модель, концепция архитектуры, онтологии в схематичном виде, функциональные компоненты, диаграмма вариантов использования, диаграмма прецедентов, ER-диаграмма, результаты работ экспериментов.*

Дата выдачи задания «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Руководитель профессор \_\_\_\_\_\_\_\_\_\_ / В.Н. Негода /

должность, учёная степень, ученое звание подпись инициалы, фамилия

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / подпись обучающегосяинициалы, фамилия

**Содержание**

[Аннотация 6](#_Toc105674528)

[Введение 7](#_Toc105674529)

[Глава 1. Анализ средств автоматизации управлением учебного процесса 11](#_Toc105674530)

[1.1. Функции организации учебного процесса 11](#_Toc105674531)

[1.2. Анализ автоматизируемых процессов 12](#_Toc105674532)

[1.3. Анализ известных средств поддержки управлением учебного процесса 14](#_Toc105674533)

[1.3.1 1С: Университет 15](#_Toc105674534)

[1.3.2 «Планы ВПО» ММИС Лаборатории 16](#_Toc105674535)

[1.3.3 КИС «Вектор» 18](#_Toc105674536)

[1.3.4 Система «Магеллан» для вузов 20](#_Toc105674537)

[1.3.5 Система учебных планов ВУЗа 22](#_Toc105674538)

[1.4. Особенности автоматизации процессов в современном вузе 26](#_Toc105674539)

[1.4. Анализ требований проектирования 28](#_Toc105674540)

[1.5. Формирование и анализ подхода к автоматизации учебных процессов, связанных с системой учебных планов вуза 29](#_Toc105674541)

[Глава 2. Разработка и анализ моделей подсистемы «Учебные планы» 32](#_Toc105674542)

[2.1. Построение логико-алгебраической модели АСУ «Учебные планы» 32](#_Toc105674543)

[2.2. Разработка инфологической модели множества учебных планов вуза 35](#_Toc105674544)

[2.3. Инфологическая модель учебного плана в системе «Планы ВПО» 37](#_Toc105674545)

[2.4. Алгоритмическая модель импорта данных из системы «Планы ВПО» в базу данных создаваемой подсистемы «Учебные планы» 38](#_Toc105674546)

[2.5. Архитектурная модель для индивидуального плана преподавателя в подсистеме «Учебные планы» 39](#_Toc105674547)

[2.6. Модель отчетов, составляемых на основе учебного плана 40](#_Toc105674548)

[2.7. Моделирование семантических отношений 44](#_Toc105674549)

[2.7.1 Описание онтологических отношений средствами алгебры конечных предикатов 46](#_Toc105674550)

[Глава 3. Разработка базовых проектных решений подсистемы 49](#_Toc105674551)

[3.1. Разработка диаграммы компонентов 49](#_Toc105674552)

[3.2. Разработка диаграмм вариантов использования 49](#_Toc105674553)

[3.3. Разработка диаграммы прецедентов индивидуального планапреподавателя 54](#_Toc105674554)

[3.4. Разработка схемы базы данных 54](#_Toc105674555)

[3.4.1 ER-диаграмма учебного плана в формате plx 54](#_Toc105674556)

[3.4.2 ER-диаграмма учебного плана 60](#_Toc105674557)

[Глава 4. Описание реализации подсистемы 62](#_Toc105674558)

[4.1. Процесс трансформации XML спецификаций учебных планов вуза в SQL базу данных 62](#_Toc105674559)

[4.2. Оценка эффекта от распараллеливания процесса ввода xml спецификаций учебных планов вуза в sql базу данных 63](#_Toc105674560)

[4.2.1. Организация экспериментов 63](#_Toc105674561)

[4.2.2. Проведение экспериментов и анализ результатов 63](#_Toc105674562)

[4.3. Экспериментальные исследования реализации учебных планов преподавателей вуза 67](#_Toc105674563)

[4.3.1. Формулирование гипотез 67](#_Toc105674564)

[4.3.2. Разработка планов экспериментов 69](#_Toc105674565)

[4.3.3. Формулирование гипотез 71](#_Toc105674566)

[4.4. Обзор реализации АСУ Индивидуальный план преподавателя подсистемы “Учебные планы” 77](#_Toc105674567)

[Заключение 81](#_Toc105674568)

[Список использованных источников 82](#_Toc105674569)

[Приложения 86](#_Toc105674570)

[Приложение А Parser XML спецификаций учебных планов вуза в SQL базу данных 86](#_Toc105674571)

[Приложение Б Plx спецификация учебного плана вуза 89](#_Toc105674572)

[Приложение В Шаблон индивидуального плана 92](#_Toc105674573)

# Аннотация

**к магистратской работе Кондратьева Павла Сергеевича на тему «Автоматизация бизнес-процессов кафедры по реализации учебных планов вуза»** Кафедра ВТ УлГТУ, 2022г.

Научный руководитель профессор, Негода В.Н.

**Работа** содержит 94 страницы, включающих, 37 иллюстраций, 3 приложения и список информационных источников из 62 наименования.

**Ключевые слова**: автоматизированная система, индивидуальный план преподавателя, автоматизация организационного управления учебного процесса, система автоматизации процессов, связанных с системой учебных планов вуза.

Представленная магистерская работа посвящена автоматизации бизнес-процессов кафедры по реализации учебных планов вуза.

Пояснительная записка содержит следующие разделы:

1. Анализ средств автоматизации управлением учебного процесса.
2. Разработка и анализ моделей подсистемы «Учебные планы». В данной главе рассматриваются описание задач системы, цели и задачи проекта, описываются инфологические, математическо-алгебраические модели.
3. Проектирование приложения. Эта глава включает в себя описание процесса функционирования системы, описание ER – диаграммы сущностей базы данных, компонентов, диаграммы деятельности.
4. В четвертой главе описан процесс реализация проектных решений.

# Введение

Сегодня, нет ни одной сферы человеческой деятельности, которую так или иначе не коснулась бы автоматизация. В настоящее время сотрудникам государственных учреждений необходимо работать с большим количеством документации.

Существуют различные формы отчетности, планы и другие документы, которые создаются на основе определенных стандартов.

Ежегодно руководство кафедры сталкивается с проблемой формирования плана нагрузки преподавателей.

В настоящий момент времени автоматизация учебных процессов, традиционно выполнявшихся вручную, является общепризнанной необходимостью. Это обусловлено рядом причин:

* Экономическим эффектом от снижения затрат на сбор и обработку информации в сравнении с ситуацией, когда эта же информация обрабатывается в “бумажном виде”;
* Возможность без особых затрат удовлетворять информационные потребности различных категорий пользователей, в том числе и не сотрудников университета;
* Объем дублируемой работы, затрачиваемой на обработку одной и той же информации, сводится к нулю;
* Снижается вероятность искажения данных.

Вузы России на по-разному решают вопрос об автоматизации учебных процессов. Они активно используются программный комплекс планов ВПО [1]. Работа с каждым учебным планов выполняется в среде MS Excel [2] или на базе различных платформ и технологий. Средства комплекса не предоставляет возможность выполнять обработку данных над всем массивом учебных планов, это важно для автоматизации различных процессов организационного управления:

* Мониторинг процессов формирования кафедры факультета;
* Сравнительный анализ степени соответствия различных планов’
* Требованиям стандартов и требований модификаций разделов планов разных наименований, направлений и профиля;
* Формирование многих рабочий документов по организации учебного процесса (индивидуальных планов преподавателя, расчасовок для разработки учебного расписания занятий).

Несмотря на то, что попытки решить данную проблему предпринимаются уже в течении нескольких лет, общепризнанное универсальное решение еще так и не было найдено.

**Целями и идеями проекта являются**

Целью данной работы является исследование и развитие подхода к автоматизации учебных процессов, базирующегося на технологии объектно-ориентированного моделирования и Интернет-технологии построения клиент-серверных систем.

В соответствии с поставленной целью решаются следующие задачи:

* Анализ существующих средств и технологий автоматизации учебных процессов;
* Исследование потребностей различных групп пользователей, чья деятельность связана с учебными планами;
* Разработка и исследование формальных моделей объектов и процессов, связанных с автоматизацией учебного процесса на основе учебных планов;
* Разработка объектно-ориентированных моделей как базовых проектных решений для системы «Учебные планы».

**Область исследования** – автоматизация организационного управления высшими учебными заведениями.

**Объект исследования** – автоматизация процессов, связанных с формированием и использованием учебных планов вуза.

**Предметом исследования** – методология автоматизации процессов, связанных с системой учебных планов вуза.

**Научно-методологическая новизна** заключается в следующем:

* Разработка нового сценария взаимодействия учебного плана;
* Развитие средств автоматизации организационного управления кафедры ВТ и УлГТУ;
* Автоматизировать процесс создания учебных планов вуза.
* Разработаны логико-алгебраические модели системы “Учебные планы”;
* Проанализированы бизнес-процессы, связанные с использованием учебных планов. Результаты анализа в отличие от аналогичных результатов отличают существенно более широким спектром акторов и охватываемых бизнес-процессов;
* Процесс трансформации XML спецификаций учебных планов вуза в SQL базу данных.

**Практическая ценность:**

* Разработаны объектно-ориентированные модели, представляющие систему согласованных проектных решений;
* Диаграммы вариантов использования различных групп пользователей;
* Диаграммы классов системы и ER-диаграммы для базы данных;
* Диаграммы последовательности действий для различных областей работы системы;
* Диаграмма распределения компонентов системы;
* Инфологической модели множества учебных планов вуза;
* Алгоритмическая модель импорта данных из системы «Планы ВПО» в базу данных создаваемой подсистемы «Учебные планы»;
* Описание онтологических отношений средствами алгебры конечных предикатов.

# Глава 1. Анализ средств автоматизации управлением учебного процесса

## 1.1. Функции организации учебного процесса

Функции организации учебного процесса состоят в создании требований, предъявляемых к информационной системе в образовательных учреждениях.

В Минобразовании РФ подготовлены специальные «Требования к отраслевой информационной системе сферы образования Российской Федерации» [3], которым должна удовлетворять информационная система в образовательных учреждениях. Также Минобразования РФ подготовлен стратегический документ – «Концепция создания интегрированной информационной системы Минобразования России» [4], которому по основным положениям должны соответствовать информационные системы вузов. Другая польза от использования этих документов заключается во многих детально описанных аспектах построения информационных систем в вузах.

Стоит понимать, что исполнение этих требований не являются обязательными для вузов (эти документы прежде всего разрабатывались для самого Минобразования РФ).

Система “Учебных планов” не должна обеспечивать автоматизацию абсолютно всех задач, использующих на учебные планы. Однако эти системы могут использовать систему “Учебные планы” как источник достоверных данных для своих нужд. Содержание ученых планов является структурой образующей основой для многих процессов управлением обучением, связь с которыми будут представлены ниже (1.2. Анализ автоматизируемых процессов).

Т. о. прежде всего к основным задачам системы относится реализация возможности сбора и хранения всей необходимой информации по учебным планам.

Кроме всего прочего в системе должно поддерживаться хранение и редактирование дополнительной справочной информации, например, такой как перечень кафедр, факультетов, дисциплин, необходимой для корректной работы с учебными планами.

Далее необходимо реализовать возможность генерации отчетов, базирующихся на учебных планах и поддержку различного рода нюансов в требованиях к таким отчетам.

Система должна автоматически контролировать за сбор сведений по учебным планам с кафедр университета и предоставлять отчеты о результатах.

Еще одной из обязательных задач системы является автоматическая проверка введенных планов на соответствие стандартам. Т. е. в системе должно поддерживаться хранение и редактирование сведений по стандартам, а также функции проверки планов на соответствие.

Таким образом, не взирая на степень вовлечения содержания учебного плана в реализацию тех или иных функций организации учебного процесса, все они обслуживают так или иначе реализацию совокупности учебных планов.

## 1.2. Анализ автоматизируемых процессов

Учебный план [5] – документ, определяющий перечень, последовательность и распределение по периодам обучения учебных курсов, предметов, дисциплин (модулей), практик, предусмотренных образовательной программой, временные затраты (трудоемкость) на их основе, а также виды учёной и самостоятельной деятельности, промежуточной и итоговой аттестации обучающихся.

Организация образовательного процесса в образовательном учреждении регламентируется учебным планом, годовым календарным учебным графиком и расписаниями занятий, разрабатываемыми и утверждаемыми образовательным учреждением самостоятельно.

Учебный план, по которому будет заниматься конкретный поток студентов, разрабатывается на кафедре. Это делается в соответствии со стандартами, утвержденными Министерством образования и науки Российской Федерации.

Федеральный государственный стандарт высшего образования [6] – совокупность требований, обязательных при реализации образовательной программы высшего образования по направлению подготовки образовательными учреждениями высшего образования.

Эти стандарты разрабатываются для каждого отдельного направления специальности и специализации. Они содержат требования по начиная от требований к списку дисциплин, которые должны изучаться в рамках данного направления и заканчивая требованиями на общее количество часов обучения по специальности за все учебные курсы.

После составления учебного плана кафедрой, он передается в учебную часть, где его проверяют и утверждают.

В соответствии с утвержденными учебными планами проводится расчет штатов [7] на кафедре. Он включает в себя список дисциплин, которые должна вести кафедра, а также некоторые виды специфических нагрузок, таких, например, как руководство кафедрой. Расчет штатов составляется на один учебный год на весенний и осенний семестры.

В соответствии с расчетом штатов, распределяются часы занятий и нагрузки для преподавателей кафедры, также в соответствии с расчетом штатов происходит распределение аудиторного фонда и т. п.

В роли основного документа преподавателя выступает индивидуальный план работы преподавателя [8], определяющим планируемые объемы и виды нагрузки преподавателей, а также основным отчетным документом, фиксирующим её фактическое выполнение по завершении учебного года.

Индивидуальный план работы преподавателя включает в себя следующие разделы:

* Повышение квалификации;
* План работы согласно учебной нагрузке;
* Учебно-методическую работу;
* Научно-исследовательскую работу;
* Организационно методическую работу;
* Воспитательную работу.

План составляется в двух экземплярах на бланках установленного образца, один из которых хранится в делах кафедры, а другой - у преподавателя.

В качестве базовых автоматизируемых процессов, которые автоматизируются, будем рассматривать только те, которые наиболее тесным образом связаны с содержанием учебных планов. К таковым относятся:

• Расчет штатов;

• Формирование расчасовок для разработки учебного расписания занятий;

• Требования к дисциплинам;

• Формирование индивидуальных планов работы преподавателей.

## 1.3. Анализ известных средств поддержки управлением учебного процесса

Автоматизация организационного управления учебным процессом [9] осуществляется в вузах страны и мира многие десятки лет. В России соответствующие средства чаще всего включаются в общевузовскую систему АСУ ВУЗ [10] – автоматизированную систему управления вуза.

В настоящее время имеет смысл выделить два вида систем: комплексные системы, поставляемые как программные продукты; отдельные разработки служб автоматизации вузов.

К комплексным системам относятся:

* 1С: Университет [11];
* «Планы ВПО» ММИС Лаборатории [12];
* КИС «Вектор» [13];
* Система «Магеллан» для вузов [14];
* Система учебных планов ВУЗа [15].

### 1.3.1 1С: Университет

Программный продукт "1С: Университет" разработан на технологической платформе "1С: Предприятие 8.3".

Наиболее широко в вузах РФ используется 1С: Университет. Рассмотрим структурно-функциональную организации этой системы более подробно, опираясь на [12].

Продукт охватывает все уровни деятельности основных подразделений учреждения высшего профессионального образования и интегрируется с типовыми решениями фирмы «1С» [16] для бухгалтерии и отдела кадров.

Решение позволяет автоматизировать учет, хранение, обработку и анализ информации об основных процессах высшего учебного заведения: поступление в вуз, обучение, оплата за обучение, выпуск и трудоустройство выпускников, расчет и распределение нагрузки профессорско-преподавательского состава, деятельность учебно-методических отделов и деканатов, поддержка ФГОС-3 [17] и уровневой системы подготовки (бакалавр, специалист, магистр) на уровне учебных планов и документов государственного образца об окончании вуза, формирование отчетности.

Решение может применяться для автоматизации рабочих мест сотрудников следующих структурных подразделений вуза:

* Приемная комиссия;
* Деканаты;
* Кафедры;
* Учебно-методический отдел;
* Бухгалтерия;
* Студенческий отдел кадров;
* Профсоюзный комитет.

По итогу Система «1С: Университет» включается в себя следующие компоненты:

* Планирование учебного процесса;
* Учет специализаций в документе «Учебный план»;
* Расчет и распределение нагрузки;
* Управление контингентом;
* Работа с приказами.

### 1.3.2 «Планы ВПО» ММИС Лаборатории

Информационная система «Планы» позволяет создать в рамках высшего учебного заведения единую систему автоматизированного планирования учебного процесса. Учебные планы (УП), создаваемые в ИС «Планы» полностью совместимы с форматом, используемым в процедуре государственной аккредитации.

Система также включает в себя комплект формализованных ФГОС, которые можно использовать для создания на их основе учебных планов и проверки качества УП.

Электронные макеты рабочих учебных планов содержат:

* График учебного процесса с нулевым курсом;
* Таблицу дисциплин с указанием распределения часов, зачетных единиц и форм контроля;
* Сводные данные по бюджету времени (в неделях, часах, зачетных единицах);
* Информацию о производственной, преддипломной и других видах практиках;
* Информацию о дипломных и курсовых работах (проектах);
* Информацию о государственных экзаменах;
* Нормативные показатели.

В электронных макетах УП реализован широкий набор инструментов, позволяющих составлять учебные планы и отслеживать качество конечного результата.

ПЛАНЫ ВПО – Планирование учебного процесса содержат модули:

1. Учебные планы ФГОС ВО;
2. Учебная нагрузка;
3. Рабочие программы дисциплин;
4. Семестровые графики.

Модуль"Учебные планы ВО" является развитием электронных макетов учебных планов и предоставляет следующую функциональность:

* Проверка учебного плана на соответствие требованиями ФГОС-3, ФГОС3+ и ФГОС3++;
* Автоматизированное создание плана заочной формы обучения на базе очной формы;
* Настраиваемое количество семестров в учебном году (от 2-х до 4-х);

Автоматическое построение ученых планов в двух режимах: автоматическом (исходные данные - часы в неделю в семестрах изучения дисциплин), полуавтоматическом (исходные данные - часы в неделю и ЗЕТ в семестрах изучения дисциплин).

АС «Учебная нагрузка» [18] обеспечивает комплексный подход к формированию и распределению учебной нагрузки учреждений ВПО. Система рассчитана для работы в локальной сети и имеет три уровня доступа, которые определяют функционал доступный пользователям.

Программное обеспечение «Рабочие программы дисциплин» (далее ПО «РПД») входит в состав пакета "ПЛАНЫ" и предназначено для подготовки одноименных документов на основе рабочих учебных планов (РУП). Эти документы хранятся в базе данных и могут быть выведены в электронные или печатные формы с целью передачи в библиотечные фонды или для предоставления студентам. Они же могут использоваться для представления экспертам в области содержания образования при осуществлении процедур самоанализа или аккредитации.

Информационная система «Планы» позволяет создать в рамках высшего учебного заведения единую систему автоматизированного планирования учебного процесса. Учебные планы (УП), создаваемые в ИС «Планы» полностью совместимы с форматом, используемым в процедуре государственной аккредитации.

Система также включает в себя комплект формализованных ФГОС, которые можно использовать для создания на их основе учебных планов и проверки качества УП.

### 1.3.3 КИС «Вектор»

КИС “Вектор” была разработана Российским новым университетом (РосНОУ) [19].

В 2002 году руководством РосНОУ было принято решение о поэтапной разработке и вводе в эксплуатацию информационной системы вуза, получившей в дальнейшем название «Вектор».

Информационная система «Вектор» предназначена для организации и контроля процесса обучения высшего учебного заведения с разветвленной организационной структурой. ИС «Вектор» предоставляет возможность хранения и обработки информации по основным (в перспективе – по всем) бизнес-процессам [20] вуза в едином информационном пространстве. Система использует стандартизованный Windows-подобный интерфейс, что делает работу с ней интуитивно понятной. В ИС «Вектор» предусмотрено разграничение прав доступа участников учебного процесса к данным и функциям системы, их защита и безопасность. В соответствии с существующими в вузе бизнес-процессами в системе могут быть различные функциональные роли, для исполнения каждой из которых может быть назначено любое количество сотрудников.

Начиная с февраля 2004 года, в РосНОУ успешно функционирует система, которая обеспечивает:

* Автоматизацию большинства этапов управления делопроизводством и документооборотом;
* Повышение надежности и эффективности обработки информации;
* Осуществление оперативного, достоверного учета, анализа и контроля деятельности служб вуза;
* Повышение качества информационного обеспечения руководителей разных уровней;
* Оперативный обмен управленческой информацией между структурными подразделениями вуза.

С помощью информационной системы решаются следующие задачи:

* Сбор, передача, накопление и анализ информации от различных структурных подразделений вуза, факультетов, отделов, преподавателей и студентов;
* Накопление кадровой информации о студенте от поступления в вуз до его выпуска. При этом осуществляется оформление сопроводительной документации, выписок, справок, аналитической информации и т. п.;
* Мониторинг организации и проведение образовательного процесс в вузе;
* Учет и анализ результатов текущего контроля, промежуточной и итоговой аттестации студентов;
* Контроль за движением студенческого контингента, автоматизация процесса формирования приказа о зачислении студента в вуз, о переводе с курса на курс и т. п.;
* Автоматизация ведения бухгалтерского учета, контроль и учет оплаты обучения студентами, подготовка и оформление платежных поручений.

В состав ИС «Вектор» в настоящее время входят следующие модули: базовый (кадры, оплата, движение контингента, успеваемость, отчеты), «Вектор-фото», «Вектор-администратор» и «Учебные планы». Разрабатываются модули «Посещаемость» и «Расписание».

В конечном итоге, решение позволило университету улучшить качество образовательных услуг и расширить возможности студентов по освоению знаний. «Мы понимали – для того чтобы эффективно управлять университетом нужно научиться эффективно управлять информацией. ИС „Вектор“ была специально разработана для этого и полностью оправдала наши ожидания» заключает Владимир Алексеевич Зернов, ректор РосНОУ, профессор, д.т.н.

### 1.3.4 Система «Магеллан» для вузов

Система управления учебным процессом для вуза [21] – это программный продукт или программный комплекс, который предназначен для автоматизации всех бизнес-процессов образовательной организации. Наличие электронной системы – обязательное требование для прохождения проверки РОСОБРНАДЗОРА [22] образовательными учреждениями всех типов – от школы и учебного центра до передового вуза.

Система «Магеллан» (платная) – это возможность гибкой и простой настройки в соответствии с требованиями образовательного процесса, унифицированная архитектура и комплексный подход к автоматизации. По отзывам руководителей и сотрудников образовательных организаций России – именно такой продукт является сейчас наиболее востребованным.

Система «Магеллан» включает в себя более 17 программных модулей, охватывающих все участки образовательного процесса, личные кабинеты обучающегося, преподавателя и поступающего, а также множество других компонентов.

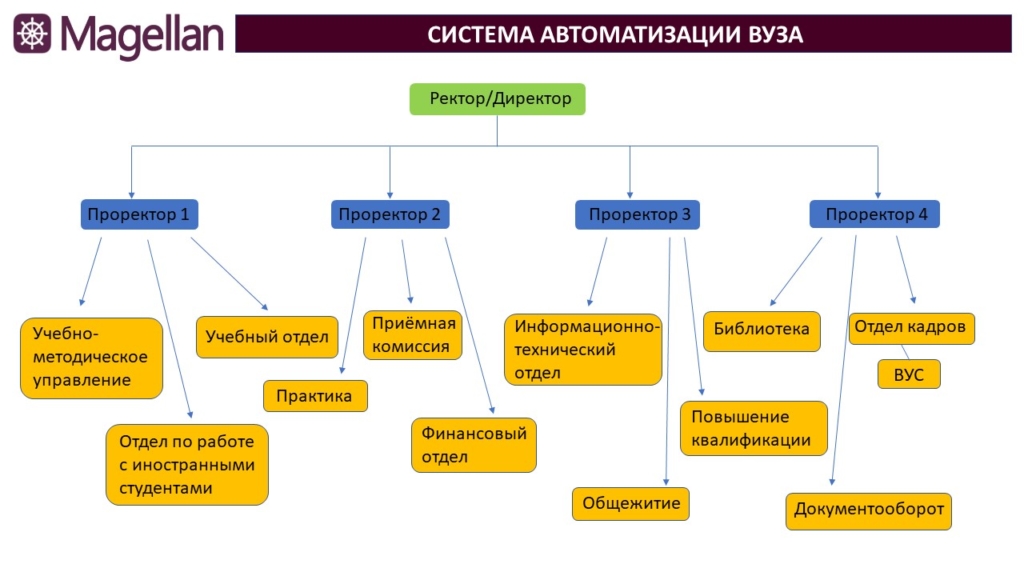


Рис.1.1. Объекты управления

Компоненты «Магеллана» управление учебным процессом:

* Приёмная комиссия;
* Отдел кадров;
* Деканат;
* Электронный журнал успеваемости;
* Расчёты с обучающимися;
* Учебно-методическое управление;
* Кафедра;
* Тематический план по дисциплине;
* Расписание.

Рассмотрим основные модули. Используя модуль «Деканат», вы сможете повысить эффективность работы сотрудников учебного отдела (деканата) по учёту личных дел обучающихся, а также их успеваемости, посещаемости, трудоустройства, контроля оплаты обучения, привести систему документооборота к единообразию и порядку, оперативно формировать сложнейшие отчёты, например, ВПО-1 [23].

Модуль «Кафедра» позволяет автоматически рассчитать и распределить учебную нагрузку по кафедрам и преподавателям. Данные для расчёта загружаются из учебных планов и РУПов, созданных в модуле «Учебно-методическое управление». Впоследствии преподаватели заполнят фактически выполненную нагрузку в своём личном кабинете, и вы с лёгкостью сможете проконтролировать её исполнение.

Модуль «Кафедра» позволяет:

* Распределить учебную работу между преподавателями кафедры на основе рассчитанной учебной нагрузки на кафедры;
* Вводить, хранить и анализировать информацию о выполнении учебной нагрузки преподавателями кафедры по неделям, месяцам, полугодию;
* Вводить, хранить и анализировать информацию о выполнении методической, научно-исследовательской и воспитательной (внеаудиторной нагрузки) работы преподавателями кафедры;
* Вводить фактически выполненную нагрузку онлайн через личный кабинет преподавателя;
* Формировать отчёты о выполнении нагрузки преподавателями кафедры по полугодиям и за учебный год. Регистрировать и печатать приказы, выписки и распоряжения по кафедре.

### 1.3.5 Система учебных планов ВУЗа

Является частью прототипа Автоматизации учебных процессов, связанных с системой учебных планов ВУЗа, написанной студенткой Карповой, кафедры «Вычислительная техника», в качестве выпускной работы.

АСУ учебные планы состоит из двух подсистем системы редактирования учебных планов, доступной только авторизированным пользователям и справочной системы учебных планов доступной без авторизации.

**Справочная подсистема**

На стартовой странице пользователь может выбрать факультет, учебные планы которого его интересуют. Также доступны ссылки на отчеты “Аудиторная нагрузка кафедр”, “ Раскладка аудиторной нагрузки кафедр”, “ Общий список дисциплин” и форма для получения списка кафедр.

После захода пользователя на страницу факультета, ему предоставляется список всех направлений, специальностей и специализаций факультета.

Для тех специальностей для которых имеются учебные планы выведен список планов со ссылками на отчеты в стандартном и расширенном виде, а также на объем занятий учебного плана. (Рис. 1. 2.)

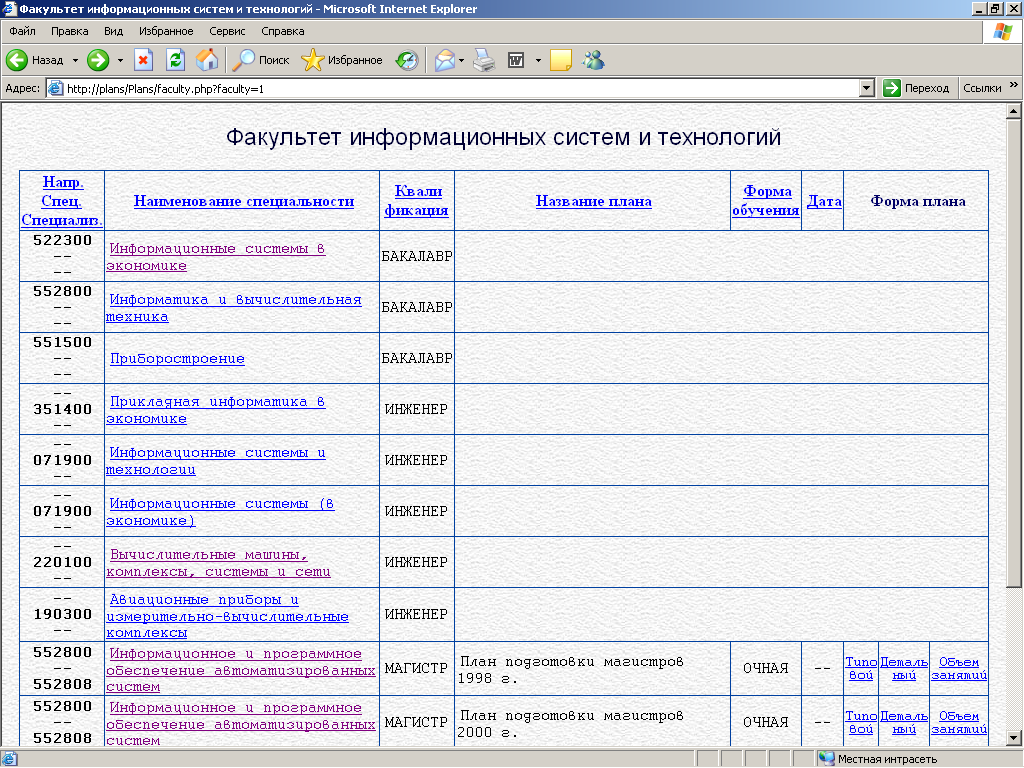


Рис. 1. 2. Страница факультета в справочной системе учебных планов

Название специальности является ссылкой ведущей на страницу специальности. Там можно посмотреть информацию о ее полном названии, ведущей кафедре, а также список потоков, существующих на данной специальности.

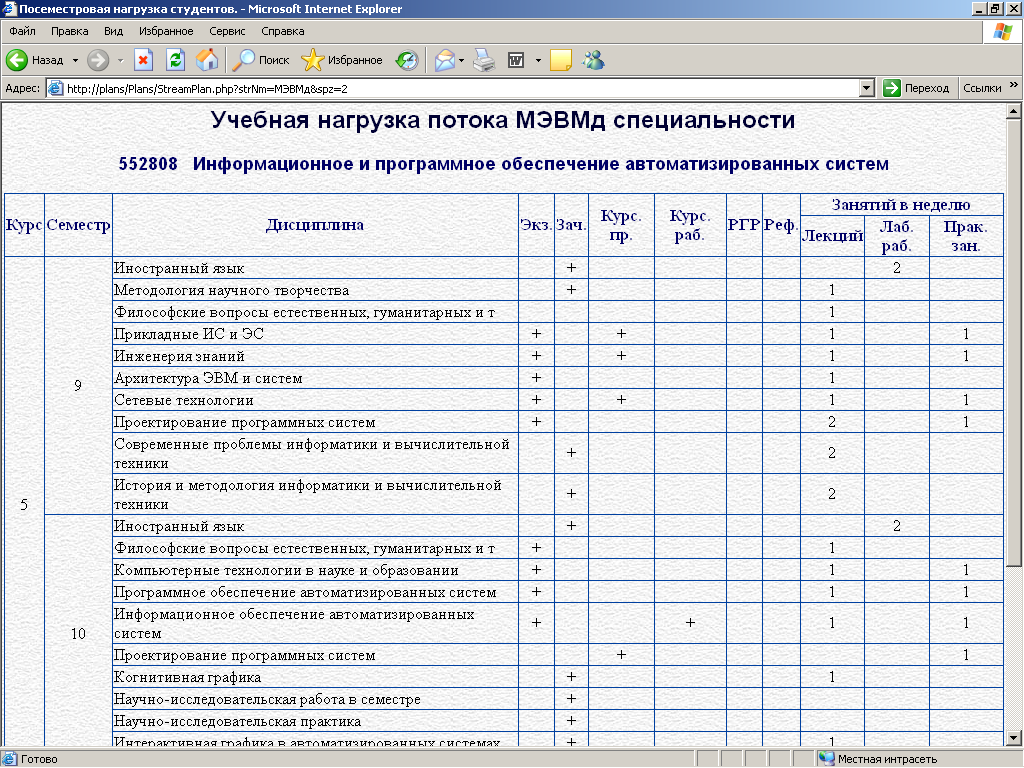


Рис. 1. 3. Отчет “Учебная нагрузка потока”.

Для каждого потока можно просмотреть отчет по учебной нагрузке. Он представляет собой список дисциплин за все курсы обучения потока, с указанием экзаменов, зачетов и других видов отчетности по дисциплине, а также количества лекционных и практических занятий и лабораторных работ в неделю (см. Рис. 1. 3.)

**Подсистема редактор учебных планов**

Здесь же можно проверить план на соответствие стандартам.

Кроме того, пользователи кафедра и учебная часть могут редактировать выбранный план.

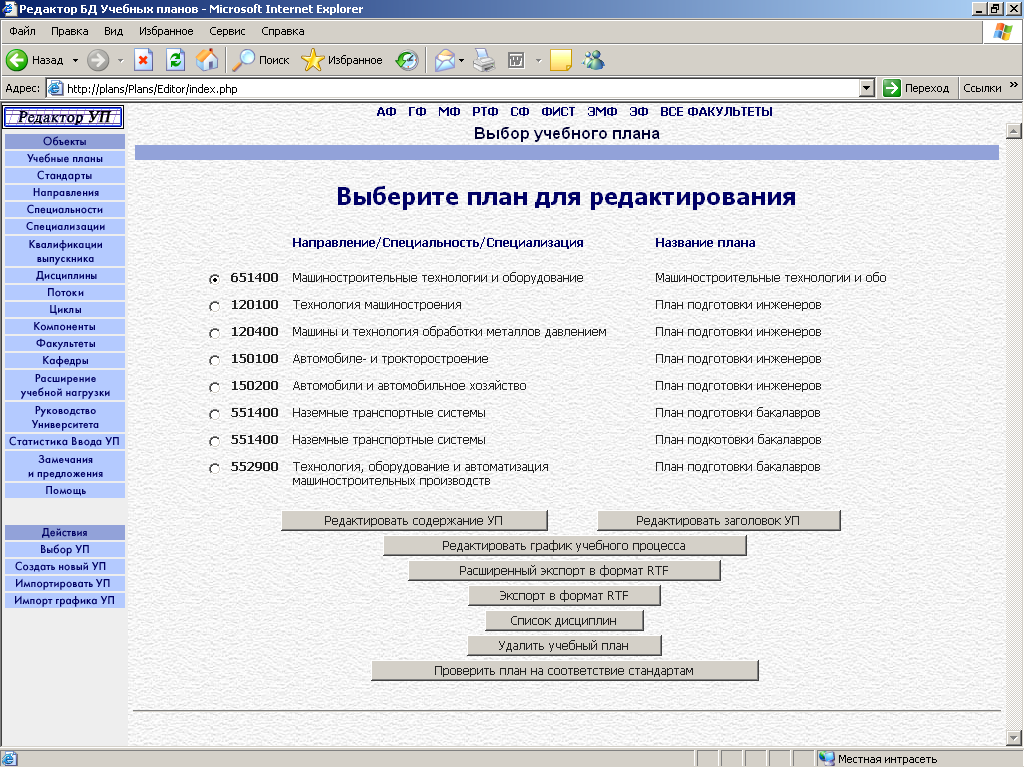


Рис. 1. 4. Стартовая страница редактора учебных планов

Редактор “Учебные планы” состоит из следующих модулей:

* Учебные планы;
* Стандарты;
* Направления;
* Специальности;
* Специализации;
* Квалификации выпускника;
* Дисциплины;
* Потоки;
* Циклы;
* Компоненты;
* Факультеты;
* Кафедры;
* Расширение учебной нагрузки;
* Руководство Университета;
* Статистика ввода учебных планов;
* Замечания и предложения.

Работа каждого модуля обеспечивается группой php-скриптов.

Доступ к модулям АСУ “Учебные планы” можно использую меню редактора расположенное слева. Также из этого подменю доступен раздел “Помощь”, содержащий краткую справку по системе “Учебные планы”.

Данная реализация не прошла временем и больший функционал ее отсутствует из ряда причин, к которым относится: старые библиотеки, которые уже не поддерживаются, неполной схемы базы данных и тд. Из доступного функционала работа способствующие остались простые select запросы по получению данных из бд, таких как справочники.

В конечном итоге, это решение послужили материалом для разработки системы и могут быть использованы в дальнейшем для разработки подобных систем и расширения функциональности уже существующей.

## 1.4. Особенности автоматизации процессов в современном вузе

Из предыдущего раздела видно, что большинство современных российских АСУ ВУЗов можно подразделить на две группы:

* Коммерческие комплексные системы управления вузом. Эти системы, как правило, охватывают весь комплекс задач связанных с работой вуза, включая финансы и материально техническое обеспечение. Камнями преткновения в вопросах внедрения таких систем является их рыночная стоимость и материально техническая база плюс дополнительное программное обеспечение, требуемое для внедрения такой системы. При рассмотрении вопроса о рыночной стоимости системы, может выясниться, что вуз либо не обладает достаточными средствами для закупки и внедрения такой системы, либо с большей готовностью расходует эти средства на разработку собственной АСУ, более гибко удовлетворяющей требованиям конкретного вуза;
* Некоммерческие АСУ, разрабатываемые силами самого вуза, на некоммерческой основе, для внутреннего использования. К плюсам таких решений можно отнести более низкую, по сравнению с коммерческими системами, стоимость, а также изначальное соответствие требованиям конкретного вуза, например, поддержка особенностей дистанционного образования, территориального распределения подразделений вуза, когда каждый филиал может иметь свою базу данных, и т.п. Минусы таких систем: 1) как правило, автоматизируются только отдельные функции и процессы 2) отсутствие комплексного сопровождения и трудности в развитии системы.

Т. к. все комплексные системы строятся на базе отдельных модулей, взаимодействующих между собой, для построения комплексной системы охватывающей автоматизацией все процессы вуза, то это позволит начинать автоматизацию отдельных наиболее актуальных комплексов процессов, а в последствии расширять и дополнять систему все новыми модулями без особых проблем для уже существующих АСУ.

Для информационных систем, обеспечивающих сопровождение активно потребляемого в различных приложениях информационного ресурса, причем, общедоступного широкому кругу акторов, характерно большое разнообразие функциональных возможностей и модулей программы, реализующих соответствующих функций. Система «Учебные планы» не только относится к таким информационным системам, но и имеет свойство активно эволюционировать. В этой связи возникает проблема обеспечения концептуальной целостности базовых проектных решений системы. Развиваемый в данной диссертации подход предполагает использование формальных моделей, обеспечивающих существенно более лаконичное представление информационных объектов и отношений между ними, нежели объектно-ориентированные модели. Кроме того, формальные модели в большей степени допускают применение автоматизированных методов верификации.

## 1.4. Анализ требований проектирования

Разрабатываемая подсистема автоматизации формирования индивидуальных планов является частью платформы, которая является частью автоматизированной системы поддержки рабочих программ [24].

Назначением данной подсистемы является работа с внешними данными, которые необходимы для создания «Рабочей программы». Внешние данные включают в себя рабочие планы дисциплин, расчеты штатов, нормы времени и т.д.

Являясь частью комплексной платформы, данная подсистема имеет общую цель системы – создание автоматизированной системы поддержки рабочих программ с возможностью использования готового шаблона и исходных данных по необходимому предмету.

Главная задача данной системы – снижение нагрузки сотрудника, отвечающего за обработку информации, а также минимизация возможности совершения ошибки при оформлении рабочей программы, так как имеется стандартизованный шаблон.

На данном этапе разработки выявлено 4 проблемы:

* При проектирование больших систем возникает проблема сильной связанности;
* Разделение программных модулей по типам (универсальные программные модули, программные модули слоя правил бизнеса, программные модули слоя документов, программные модули управления бизнес-процессом и интерфейсные программные модули);
* Обеспечение целостности больших информационных систем требует поддержания моделей системы в актуальном состоянии на всех этапах разработки;
* Формирование архитектуры системы как логической модели информационной системы, объединяющей модели бизнес-процессов, информационной модели и модели организационной структуры.

## 1.5. Формирование и анализ подхода к автоматизации учебных процессов, связанных с системой учебных планов вуза

Система “Учебные планы” строится в расчёте на получение следующих основных свойств:

* Общедоступность – возможность широкого доступа к ресурсам системы в том числе, людей, не участвующих напрямую в процессах, связанных с учебными планами
* Гибкость – возможность адаптации системы под конкретные требования образовательных стандартов и органов управления образованием, без необходимости изменения функциональности модулей и подсистем; гибкость дает возможность уменьшить затраты труда и риски порождения неустойчивого функционирования при модификации системы в связи с изменяющимися требованиями стандартов, а также требованиями по формам отчетности;
* Расширяемость – возможность дополнения системы новыми подсистемами и приложениями без изменения общей части системы; благодаря этому свойству мы получаем такие же выгоды, что и за счет гибкости, однако, здесь предусматривается изменение функциональности системы с минимальным риском порождения побочных эффектов;
* Понятность – это свойство обеспечивает возможность работы с системой без значительных затрат на предварительную подготовку и обучение пользователя; поскольку часть бизнес-процессов, базирующихся на учебных планах, объективно обладает значительной сложностью, то в развиваемом подходе реализуется разделение на 2 группы пользователей: представители широкой аудитории пользователей, потребляющие различные публикации учебных планов и информации, связанной с ними, и пользователи-менеджеры, осуществляющие формирование самих учебных планов и сопровождающей их информации; для первой категории интерфейс строится как интуитивно понятный и обеспечивается быстрый и легкий доступ к любой имеющийся для просмотра информации; для пользователей второй категории целесообразно пожертвовать интуитивной понятностью интерфейса в пользу обеспечения высоко-производительного труда;
* Невысокая ресурсоемкость – т. к. система предполагает большое количество пользователей, высокие требования к производительности машин пользователей не должны стать ограничивающим фактором при использовании системы.

В отличии от рассмотренных ранее систем разработка АСУ учебные планы в сетевой среде вуза дает следующие преимущества:

Возможность обеспечить информационные потребности пользователей, не имеющих непосредственного отношения к работе с учебными планами, начиная от преподавателей вуза и заканчивая абитуриентами. В среде Internet [25] удобно осуществлять взаимодействие между различными подсистемами, даже если они используют отличные друг от друга форматы данных, т. к. уже существуют технологии решающие эту проблему.

Т. о. использование Интернет-технологий позволяет построить систему, удовлетворяющую почти всем требованиям к автоматизированной системе управления вузом, перечисленным в разделе 1. 4. и индивидуальным требования к системе автоматизации процессов, связанных с учебными планами.

Система строится на основе концепции централизованных баз данных, несмотря на то, что сама по себе сетевая среда является распределенной. Распределенными в системе являются только клиентские станции.

Защита данных от несанкционированного доступа строится на основе отношения «Персона – Функции - Регионы данных» [26]. Благодаря этому удается обеспечить устойчивое функционирование системы в условиях, когда модификация данных доступна на уровне деканатов и кафедр. Уровень защищенности информации зависит в основном от дисциплины работников учебной части, деканатов и кафедр, которым предоставляются права модификации.

Многие бизнес-процессы, связанные с использованием учебных планов, предусматривают реализацию расчетных соотношений, имеющих свойство достаточно часто меняться. Традиционная реализация таких соотношений через математические выражения операторов языка программирования требует слишком частой модификации исходного кода. В предлагаемом подходе гибкость по отношению к нормативно-справочной информации процедурного характера достигается с помощью технологии XML [27].

Развиваемый в данной диссертации подход предполагает использование формальных моделей, обеспечивающих существенно более лаконичное представление информационных объектов и отношений между ними, нежели объектно-ориентированные модели. Кроме того, формальные модели в большей степени допускают применение автоматизированных методов верификации.

# Глава 2. Разработка и анализ моделей подсистемы «Учебные планы»

Ограниченность объема данной работы не позволяет представить весь спектр базовых моделей, связанных с реализацией учебных планах в разнообразных организационных процессах. В этой связи ниже рассмотрим только часть из них, включающую в себя следующее:

* Инфологическая модель множества учебных планов вуза;
* Инфологическая модель учебного плана в системе «Планы ВПО»;
* Алгоритмическая модель импорта данных из системы «Планы ВПО» в базу данных создаваемой подсистемы «Учебные планы»;
* Архитектурная модель подсистемы «Учебные планы».

## 2.1. Построение логико-алгебраической модели АСУ «Учебные планы»

Модели и моделирование играют чрезвычайно важную роль в деятельности человека. В сущности, всю совокупность знаний человека можно представить, как модель материального мира, в котором отражен сам субъект моделирования. Методы моделирования в настоящее время внедрились не только в технические области, но в сферы социально-экономические, сложные экономические, общественные, международных отношений, практически во все сферы человеческой деятельности. Это связано с необходимостью расширения и углубления знаний реального мира.

Под математической моделью [28] понимается приближенное описание какого-либо класса явлений внешнего мира, выраженное с помощью математической символики.

При разработке математической модели необходимо расчленить явление на элементарные процессы, выделить все действующие на объект внутренние и внешние факторы. Выяснить, какие из них существенны, а какие могут быть учтены лишь приблизительно.

Математическая модель - это не только уравнения математической задачи, но и дополнительные условия. устанавливающие границы их применимости. Все полученные с помощью этой модели теоретические результаты будут справедливы только в оговоренных рамках.

Система “Учебных планов” не должна обеспечивать автоматизацию абсолютно всех задач, использующих на учебные планы. Однако эти системы могут использовать систему “Учебные планы” как источник достоверных данных для своих нужд. Содержание ученых планов является структурой образующей основой для многих процессов управлением обучением.

Т. о. прежде всего к основным задачам системы относится реализация возможности сбора и хранения всей необходимой информации по учебным планам.

Кроме всего прочего в системе должно поддерживаться хранение и редактирование дополнительной справочной информации, например, такой как перечень кафедр, факультетов, дисциплин, необходимой для корректной работы с учебными планами.

Учебный план виде математической модели имеет вид:

*A=<department, plans, discipline, teacher>, где*

* department – множество кафедр;
* plans – множество планов;
* discipline – множество дисциплин;
* teacher – множество преподавателей.

В свою очередь splans (множество учебных планов) описывается:

*splans = {plan1, plan2, … plann}, где*

plani = {sdepartment.disci | sdepartment.disci ∈ sdiscipline}

*sdiscipline = {disci | i ∈ N*

График учебного процесса представляет собой распределение видов учебной нагрузки по неделям учебного курсам за все курсы обучения.

Таким образом график учебного процесса можно представить в виде отношения

GUP = {TL, Grade, NWeeks}, или

GUP *∈* TL x Grade x NWeeks, где

NWeeeks = {1...n} – номер недели.

TL = {ТО, К, Эк, УП, ПП, ДП, ИА, ВР, Сб, МД, НП, ТГФ, ТГИ, ТП, ГЭС} – вид занятий, можно представить его как ограниченное множество следующих значений:

* ТО – теоретическое обучение;
* К – каникулы;
* Эк – экзамены;
* УП – учебная практика;
* ПП – производственная практика;
* ДП – преддипломная практика;
* ИА – итоговая аттестация;
* ВР – выпускная работа, дипломное проектирование;
* Сб – сборы;
* МД – подготовка магистерской диссертации;
* НП – научно-исследовательская практика;
* ТГФ – гос. экзамен по философии + теоретическое обучение;
* ТГИ – гос. экзамен по иностранному языку теоретическое обучение;
* ТП – теоретическое обучение, сочетаемое с педагогической практикой;
* ГЭС – гос. экзамен по специальности;

Grade = {1…6} – курс, будем считать, что максимальный курс обучения 6 – для магистров, тогда курс может принимать значения от 1 до 6.

Весь график учебного процесса можно представить, как совокупность периодов обучения последовательно следующих друг за другом. Основываясь на этом положении, выведем следующие отношения:

FTL: TL x Grade -> SIW – период для конкретного вида занятий.

IWeeks = {<b, l>} |b, l *∈* NWeeks,b ≤ l – множество периодов недель учебного курса.

SIW = {siw| siw *∈* IWeeks} – период конкретного вида занятий представляет собой один из таких периодов.

Fiw: Grade x IWeeks -> TL – сопоставление номера курса и периода, дает вид занятий, соответствующий этому периоду.

SWeeks = {1…n} – множество недель в графике учебного процесса за весь период обучения.

Fswc: Grade x TL x SIW -> SWeek номера недель для конкретного вида обучения и курса

SWC = Fswc(crs, tl, siw) = ∑∑<bi, li> *∈* it, it *∈* crs (li-bi+1) – сумма недель конкретного вида обучения.

Из выше перечисленных моделей можно привести к примеру функцию выбора всех дисциплин, которые ведет кафедра. Эта функция является чаще всего использованной и имеет вид:

*fSelectDisc: splans x sdepartment => sdisc\*, где*

sdisc\* ∈ sdiscipline

Таким образом, не взирая на степень вовлечения содержания учебного плана в реализацию тех или иных функций организации учебного процесса, все они обслуживают так или иначе реализацию совокупности учебных планов.

## 2.2. Разработка инфологической модели множества учебных планов вуза

Онтология [29, 30, 31] определяет общий словарь для тех, кому нужно совместно использовать информацию в предметной области. Он включает машинно-интерпретируемые формулировки основных понятий предметной области и отношения между ними.

Задачи онтологии:

* Совместное использование людьми или программными агентами общего понимания структуры информации (основная цель разработки онтологии, предполагает совместное использование одной и той же базовой онтологии терминов);
* Возможность повторного использования знаний в предметной области (предполагает использование уже существующих онтологий для построения новой, возможно более сложной онтологии в определенной предметной области);
* Сделать допущения в предметной области явными (создание явных допущений в предметной области дает возможность легко изменить эти допущения при изменении знаний о предметной области);
* Отделение знаний в предметной области от оперативных знаний (если имеется общая задача конфигурирования продукта из его компонентов в соответствии с требуемой спецификацией, то можно создать систему, которая делает эту конфигурацию независимой от продукта и самих компонентов);
* Анализ знаний в предметной области (анализ знаний в предметной области возможен, когда имеется декларативная спецификация терминов).

Так как учебный план – это разбивка содержания образовательной программы по учебным курсам, по дисциплинам, по специальности. То организация образовательного процесса в образовательном учреждении регламентируется учебным планом, годовым календарным учебным графиком и расписаниями занятий, разрабатываемыми и утверждаемыми образовательным учреждением самостоятельно.

Разработка инфологической модели выполняется в режиме анализа Excel публикаций учебного плана. План содержит такое листы как:

* Нагрузка;
* Кафедры;
* Специальности;
* Направления;
* Дисциплины;
* Учебный план.

На основе анализа листов и взаимосвязи учебного плана, получаем следующую инфологическую модель, представлен на рис. 2.1. Инфологическая модель множества учебных планов вуза.

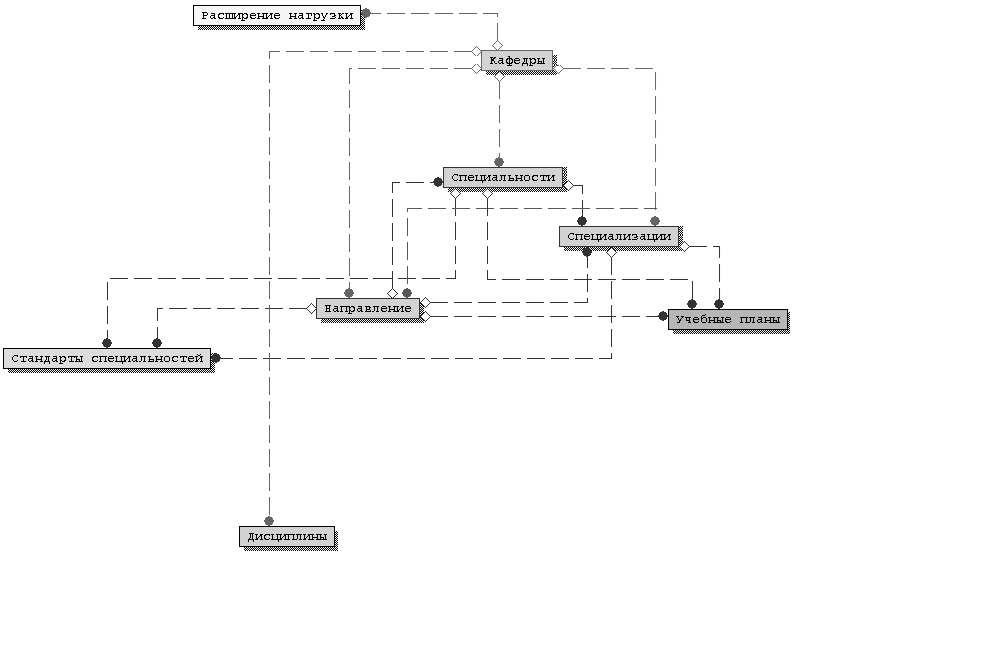


Рис. 2.1. Инфологическая модель множества учебных планов вуза

## 2.3. Инфологическая модель учебного плана в системе «Планы ВПО»

Общий вид схемы базы данных АСУ “Учебные планы” представлен на рис. 2.2.

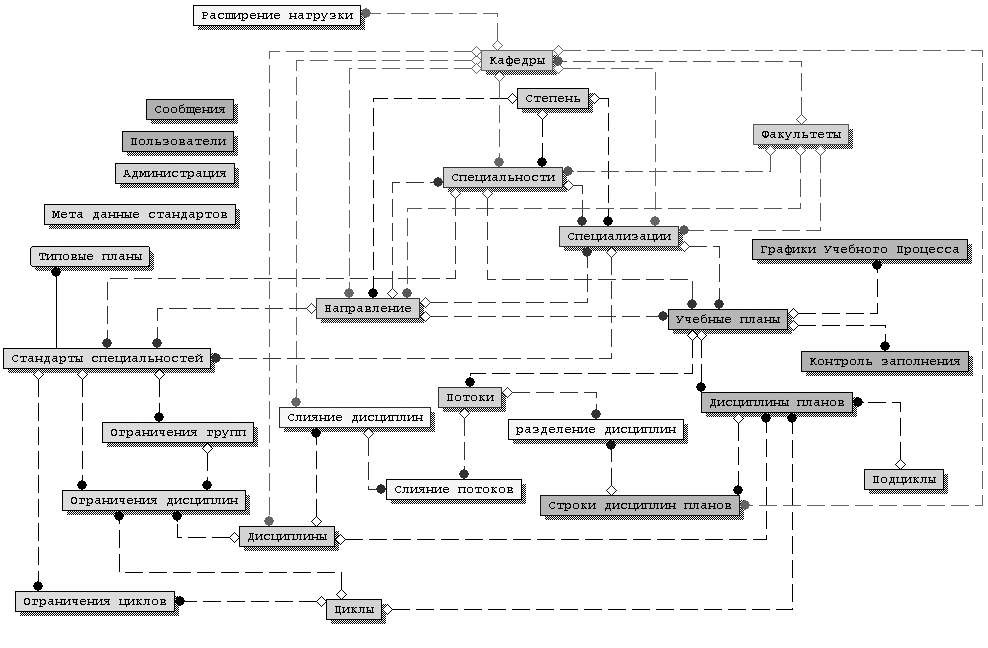


Рис. 2.2. Инфологическая модель учебного плана в системе «Планы ВПО»

## 2.4. Алгоритмическая модель импорта данных из системы «Планы ВПО» в базу данных создаваемой подсистемы «Учебные планы»

В вузах России в настоящее время активно используется программный комплекс «Планы ВПО». В этом комплексе спецификации учебных планов представлены в plx-файлах, которые содержат данные в формате XML. Работа с каждым учебным планом выполняется в среде MS Excel. Средства комплекса не предоставляют возможности выполнять обработку данных над всем массивом учебных планов. Это важно для автоматизации различных процессов организационного управления:

* мониторинга процессов их формирования кафедрами и факультетами;
* сравнительной оценки степени соответствия различных планов требованиям стандартов и требованиям унификации разделов планов разных направлений и профилей;
* организации такой публикации данных учебных планов, которая поддерживает сравнительный анализ абитуриентами;
* формирования многих рабочих документов по организации учебного процесса: индивидуальных планов преподавателей, расчасовок для составления расписания, информационных карт для организации зачетных и экзаменационных сессий.

Эффективная автоматизация перечисленных процессов требует ввода спецификаций планов в реляционные базы данных. В крупном вузе количество планов может составлять большие сотни, что определяет актуальность распараллеливания процессов ввода.

Алгоритмическая модель импорта данных из системы «Планы ВПО» в базу данных создаваемой подсистемы «Учебные планы», представлена ниже:

*A=<* *files, processor, multiprocessing, database>, где*

* files = {xml1, xml2…, xmln}, где xml *∈* Splans– множество учебных планов в формате plx (xml);
* processor = {tagList1, tagList2…, tagListn}, где tagList *∈* tableName– разбиение на множество столбцов, для записи сущностей базы данных;
* multiprocessing – параллельная обработка данных на разных ядрах процессора;
* database – CRUD операции с полученными данными.

## 2.5. Архитектурная модель для индивидуального плана преподавателя в подсистеме «Учебные планы»

Содержимое учебного плана можно представить, как множество дисциплин, сгруппированных по циклам и компонентам, каждая такая дисциплина будет характеризоваться кроме того нагрузкой, предусмотренной учебным планом. Ко всем выше сказанному добавим и индивидуальный план, который является нормативным документом, устанавливающим требования, порядок и правила составления индивидуального плана работы лиц профессорско-преподавательского состава вуза, занимающих штатные должности профессоров, доцентов, старших преподавателей, преподавателей и ассистентов кафедр на полных и неполных ставках.

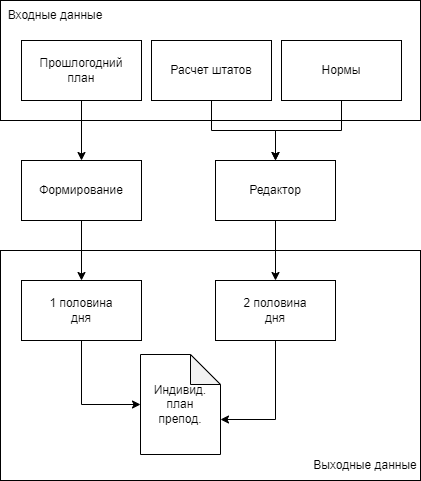


Рисунок 2.3. Концепция архитектуры для индивидуального плана преподавателя в подсистеме «Учебные планы»

## 2.6. Модель отчетов, составляемых на основе учебного плана

Основываясь непосредственно на данных учебного плана можно получить следующие виды отчетов:

* Список учебных планов (общий, или для опр. факультета);
* Список дисциплин учебного плана (выписка в диплом);
* Учебный план (расширенная, стандартная формы);
* Список направлений, специальностей, специализаций (общий, для факультета);
* Квалификации выпускника;
* Полный список дисциплин;
* Список дисциплин кафедры;
* Список потоков;
* Список циклов;
* Список компонентов;
* Список факультетов;
* Список кафедр;
* Информационная карта зачетов и экзаменов;
* Учебная нагрузка;
* Объем занятий;
* Расчет штатов.

Большинство вышеперечисленных отчетов не требует каких-либо дополнительных доработок модели учебных планов. Т. к. по сути, они являются публикациями уже существующих данных. Например, для составления отчета “Список учебных планов” требуется просто вывести все существующие учебные планы.

Составление отчета “Учебный план” требует более сложных выборок и операций над данными, т. к. требуется сопоставить данные графика учебного процесса, дисциплин, нагрузки по дисциплинам, провести некоторые вычисления величин, требуемых для включения в отчет.

Составление отчета “Расчет штатов” требует учета некоторых дополнительных сущностей к уже представленной модели, рассмотрим их здесь подробнее.

Расчет штатов – это отчет следующего вида:

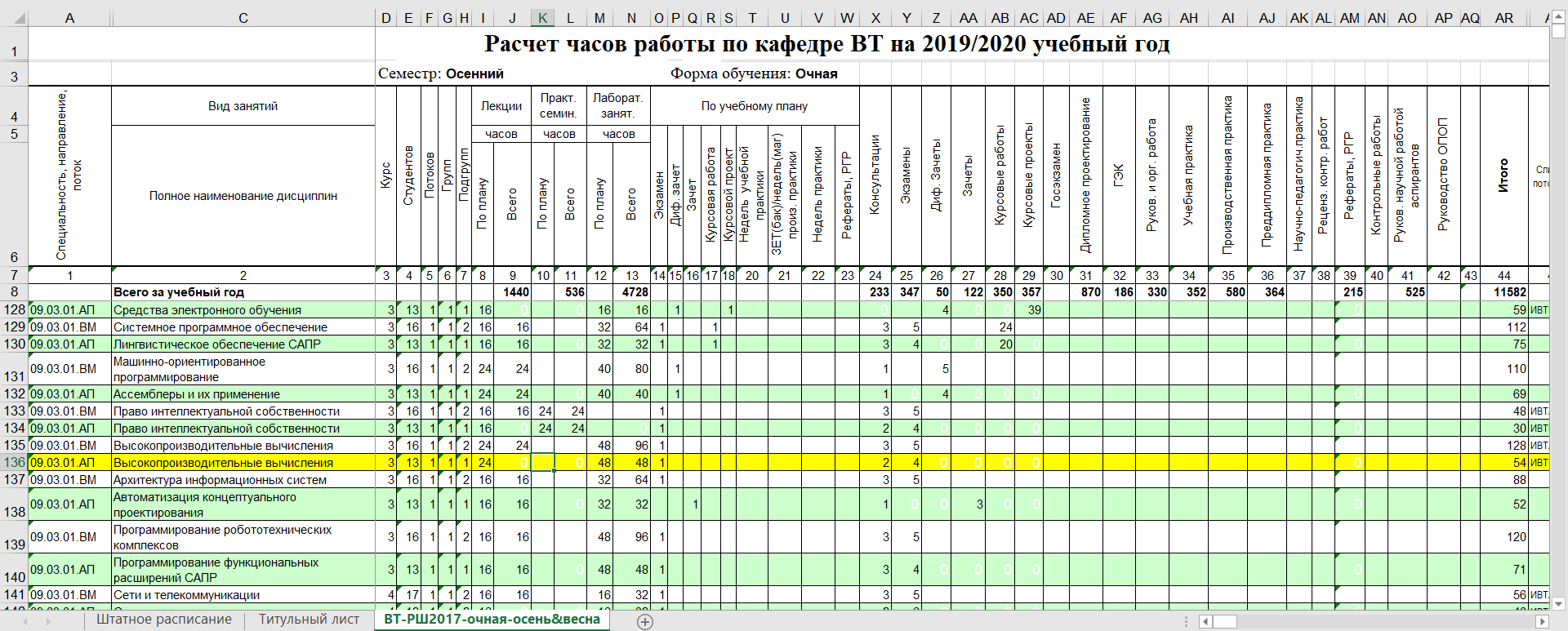


Рис. 2. 4. Выходная форма системы “Системы расчет штатов”

Таким образом, кроме всего прочего для составления расчета штатов необходимы еще и некоторые минимальные данные по контингенту учащихся:

Streams = {StreamName, Kurs, GroupCount, StdCount, PlanCode}

* StreamName – название потока;
* Kurs – курс;
* GroupCount – количество групп;
* StdCount – количество студентов;
* PlanCode – ссылка на план.

При расчете нагрузок по дисциплинам используются различные величины, перечислим основные:

* term – номер семестра;
* kurs – номер курса;
* lec – часы лекций;
* lab – часы лабораторных работ;
* pract – часы практик;
* ex – экзамен;
* test – зачет;
* kp – курсовой проект;
* kw – курсовая работа;
* rgr – расчетно-графическая работа;
* ref – реферат;
* wc – количество недель обучения.

Кроме того, этот список может быть дополнен в последствии.

Т. о. результирующую сущность расчет штатов можно представить следующим образом:

StaffCalculation = {DepartCode, StreamList, Variables}

StreamList = {StreamCode1, StreamCode2, StreamCode3,… StreamCodeN} – список потоков участвующих в расчете штатов;

* StreamCodei – ссылка на соответствующий поток;
* Variables = {var1…} – Неограниченное множество расчетных величин.

Множество неограниченное т. к. заранее не известно, какие именно величины может потребоваться вычислить, при составлении отчета.

Т. е. должна быть предусмотрена возможность, дополнить отчет, новой расчетной или постоянно величиной нагрузки.

Постоянные нагрузки для выбранной кафедры можно представить в виде следующей сущности:

Expansion = {DepartCode, ExpansionName, Semestr, Hours}, где

* ExpansionName – название расширения нагрузки;
* Semestr – номер семестра к которому оно относится (1 или 2 – соответственно для осеннего или весеннего семестров);
* Hours – величина нагрузки в часах.

Эта сущность позволяет учесть такие нагрузки, как, например, руководство кафедрой и т. п. Характерной особенностью всех этих нагрузок является, то, что они не зависят не от каких других величин, таких, например, контингент учащихся, количество недель обучения и т. п.

Грубо говоря, все данные нагрузки – это просто числа, которые будут учитываться в расчете штатов.

Более сложный случай представляют собой величины, которые в свою очередь являются производными от других величин. Например, количество часов на руководство курсовым проектом, по выделенной дисциплине зависит от того, предусмотрен ли курсовой проект учебным планом, и контингента потока, занимающегося по данному учебному плану.

Такую расчетную величину удобно представить в виде:

ValueI = F (const1 ... constn, value1... valuen) – т. е. она представляет собой некоторую функцию от множества констант и множества уже определенных расчетных величин.

Множество операций в данном функции можно ограничить операциями сложения вычитаний умножения и деления.

Следует также отметить что например

* Values1 = F (const1 .. constn)
* Values2 = F (const1 .. constn, value1)

После того как мы зададим формулу расчета, какой-либо величины, мы сможем использовать эту величину при расчете последующих величин наряду с константами.

## 2.7. Моделирование семантических отношений

Онтология [32] – это подробная спецификация структуры определенной проблемной области, формальное явное описание понятий в рассматриваемой предметной области (классов (иногда их называют понятиями)), свойств каждого понятия, описывающих различные свойства и атрибуты понятия (слотов (иногда их называют ролями или свойствами)), и ограничений, наложенных на слоты (фацетов (иногда их называют ограничениями ролей)).

Онтология включает в себя словарь (т.е. список логических констант и предикатных символов) для описания предметной области и набор логических высказываний, формулирующих существующие в данной проблемной области ограничения и определяющих интерпретацию словаря.

Онтология предлагает словарь для представления и обмена знаниями по интересующей проблеме и набор связей и свойств, которые определены между имеющимися в ее словаре неделимыми сущностями.

На практике разработка онтологии включает:

* Определение классов в онтологии;
* Расположение классов в таксономическую иерархию (подкласс – надкласс);
* Определение слотов и описание допускаемых значений этих слотов;
* Заполнение значений слотов экземпляров.

**Модель онтологии**

Формально онтология определяется как:

* X - конечное множество понятий (концептов) предметной области;
* R - конечное множество отношений между понятиями;
* F - конечное множество функций интерпретации, заданных на концептах и/или отношениях.

При R=0 и F=0 онтология трансформируется в простой словарь.

**Модель онтологической системы**

* O – онтология верхнего уровня (содержит общие понятия и отношения, не зависящие от предметной области - «объект», «свойство», «значение» и т.п.);
* P – множество предметных онтологий и онтологий задач предметной области (с учетом предпочтений пользователя);
* M – модель машины вывода данной онтологической системы (например, для изменения критериев релевантности поиска или критериев формирования репозитория).

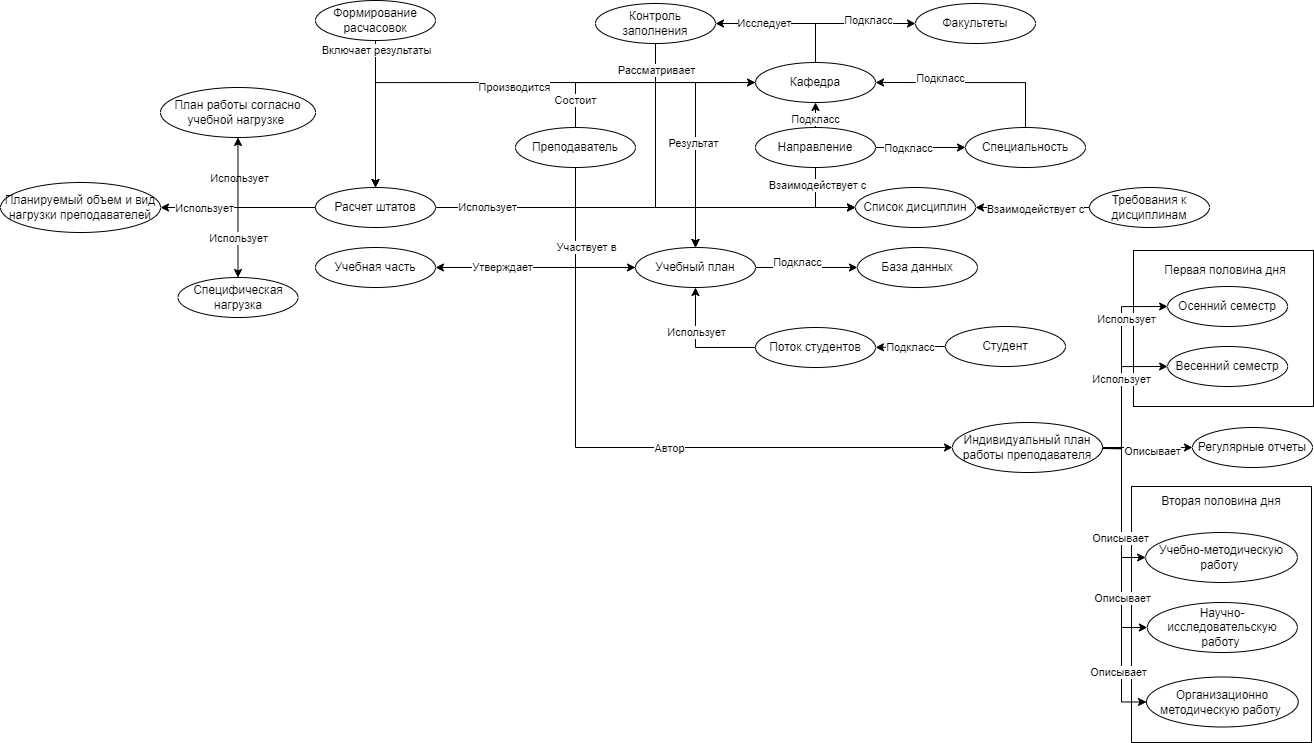


Рис. 2.5. Онтологии в схематичном виде

### 2.7.1 Описание онтологических отношений средствами алгебры конечных предикатов

Между словами в словосочетаниях существуют различные типы отношений. Отношения иерархии как разновидность сильных парадигматических отношений соответствуют отношениям подчинения или отношениям типа «выше-ниже». В пределах иерархических отношений различают отношения типа «род-вид» и «целое-часть» [33].

Отношение «род-вид» является одним из важнейших видов связей между понятиями. При этом родовым (подчиняющим) называется понятие, выражающее существенные признаки класса предметов, являющихся видами этого рода.

Соответственно видовым (подчиненным) называется понятие, которое отображает существенные признаки класса предметов, являющегося видом какого-либо рода.

В качестве математического аппарата для моделирования семантики свободных словосочетаний с родовидовыми отношениями предлагается использовать алгебру конечных предикатов (АКП). Это универсальный математический аппарат для описания детерминированных, дискретных и конечных информационных объектов. Этот аппарат позволяет описывать различные языковые закономерности в рамках единой математической модели.

Словоформы принимают свои сочетания в контексте. Контекстом в нашем случае служит словосочетание в определенной предметной области. Поэтому значение словоформы определяется лишь теми признаками, которые характеризуют ее связь со значениями слов в контексте, а также с объектами предметной области. Семантическая сочетаемость словоформ зависит от знаний предметной области и от родо-видовой структуры [34].

Слова, связанные родо-видовыми отношениями, представляют собой лексико-семантические классы с иерархической структурой, описываемой в виде родовидовых деревьев. Родо-видовое дерево представляет собой иерархию элементов, называемых узлами. В роли узлов выступают словоформы. Пример родовидового дерева представлен на рис. 2.6.

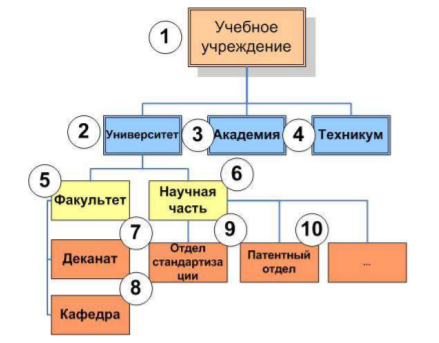


Рис. 2.6. Родо-видовое дерево лексико-семантического класса существительных

В зависимости от контекста родо-видовое дерево с этими же словоформами будет иметь другой вид. Для приведенного дерева (рис. 2.6) введем переменные и области их допустимых значений. Пусть

Уравнение, описывающее дерево, имеет следующий вид:

Решение данного уравнения соответствует узлам родо-видового дерева.

# Глава 3. Разработка базовых проектных решений подсистемы

## 3.1. Разработка диаграммы компонентов

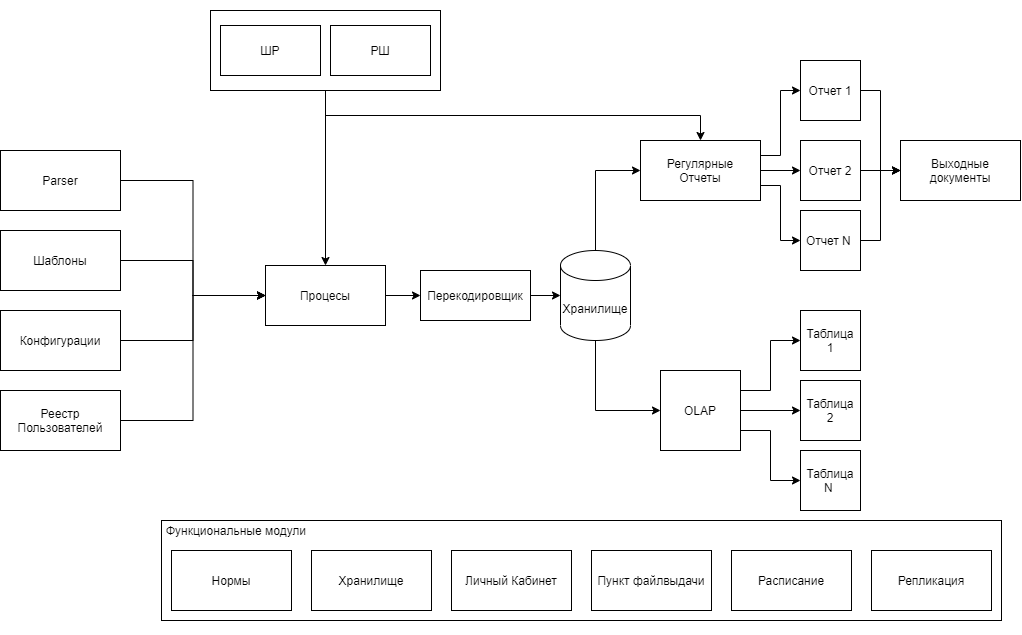


Рисунок 3.1. Функциональные компоненты учебного плана

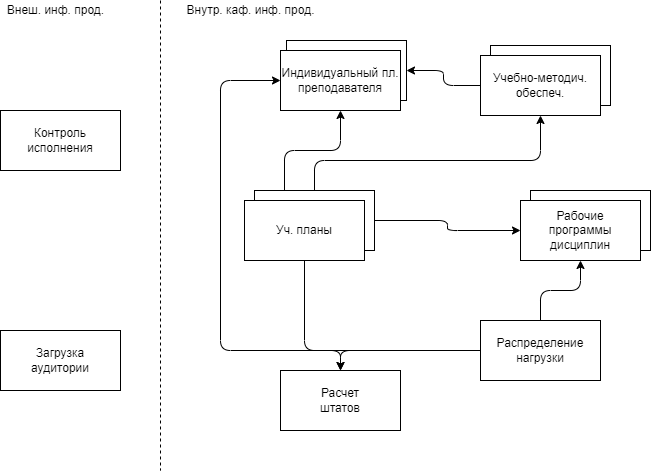


Рисунок 3.2. Внутренние и внешние компоненты информационной программы

## 3.2. Разработка диаграмм вариантов использования

Рассмотрим подробнее варианты использования системы АСУ “Учебные планы”. Т.к. система предназначена для использования различными группами пользователей и с различными целями, то следует рассмотреть варианты использования системы для каждой из этих групп пользователей отдельно.

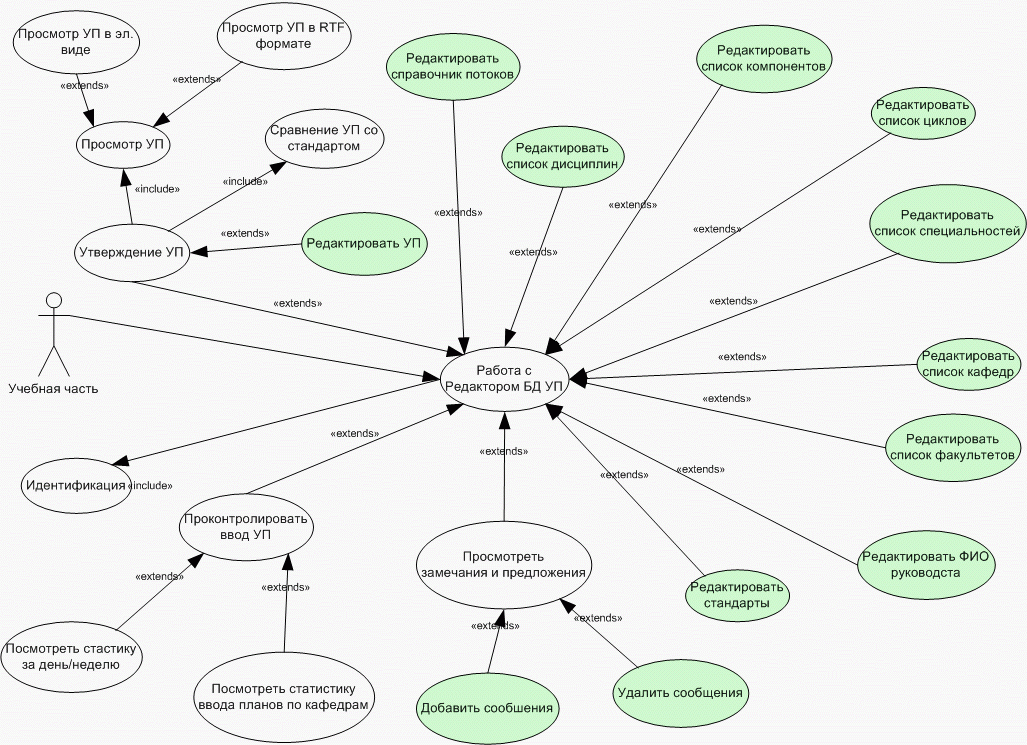


Рис. 3. 3. Диаграмма вариантов использования пользователя «Учебная часть»

Рассмотрим диаграмму вариантов использования для пользователя “Учебная часть”.

Кроме самого система должна также содержать некоторые дополнительные объекты с информацией необходимой для корректного функционирования системы. Это справочники кафедр, факультетов, специальностей, дисциплин, потоков, циклов, компонентов, справочники стандартов учебных планов и руководства университета.

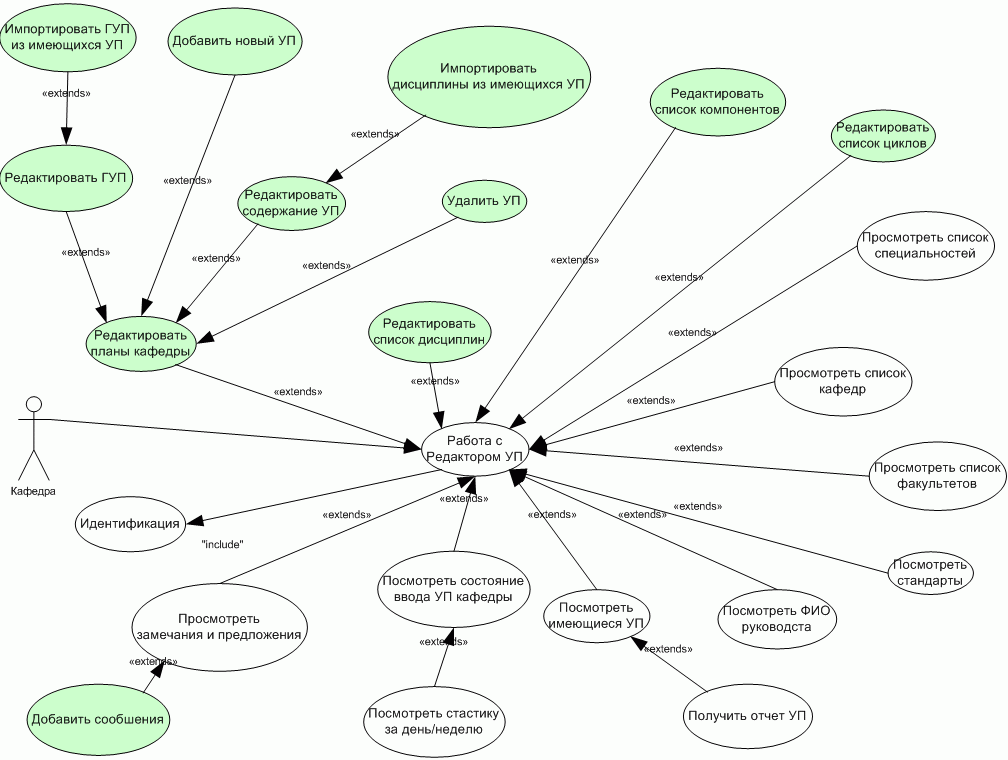


Рис. 3. 4. Диаграмма вариантов использования пользователя «Кафедра»

На плечи учебной части ложится задача контролировать обстоятельность этих дополнительных справочников

Учебная часть должна контролировать ввод учебных планов кафедрами, проверять их на соответствие стандартам и утверждать соответствующие УП.

Также учебная часть должна следить за полнотой и адекватностью стандартов, на соответствие которым проверяются введенные учебные планы.

Статистика ввода учебных планов необходима, чтобы контролировать процесс заполнения базы данных учебными планами. Она позволяет максимально быстро получить сведения о степени полноты базы данных, не прибегая к просмотру всех планов “вручную”.

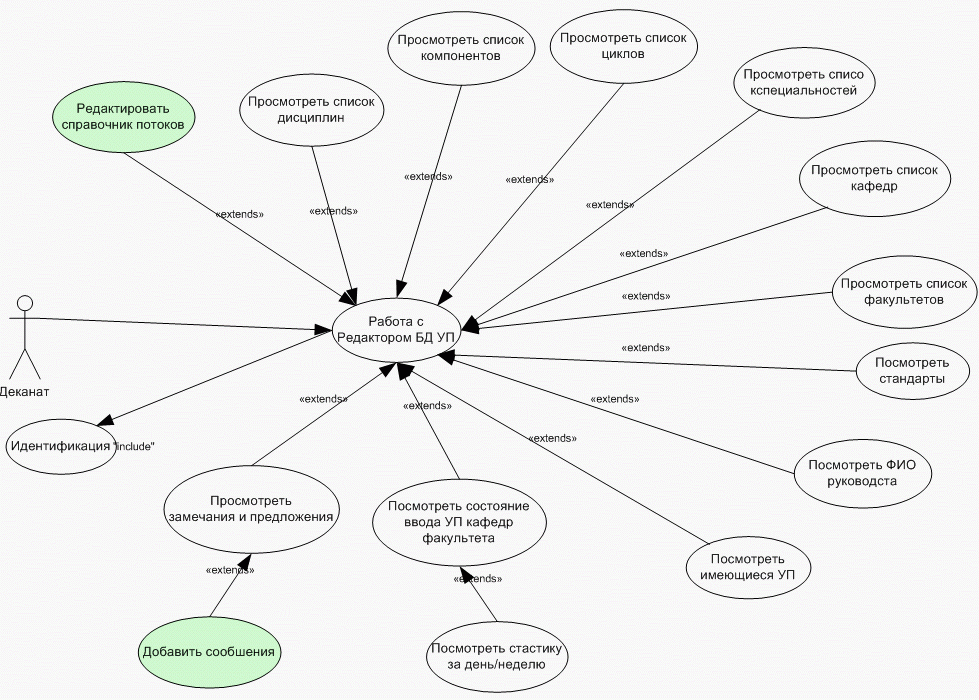


Рис. 3. 5. Диаграмма вариантов использования пользователя «Деканат»

Справочник учебных планов, нужно дополнить интерфейсом для добавления и просмотра замечаний. Это пользователям системы оперативно доводить до администратора замечания по работе АСУ. Эта возможность будет особенно полезна на этапе тестирования системы.

Пользователь “Кафедра” (Рис. 3. 4.) обладает более ограниченными возможностями по сравнению с учебной частью. Он обладает возможностью редактировать создавать и удалять планы по специальностям, относящимся к его кафедре.

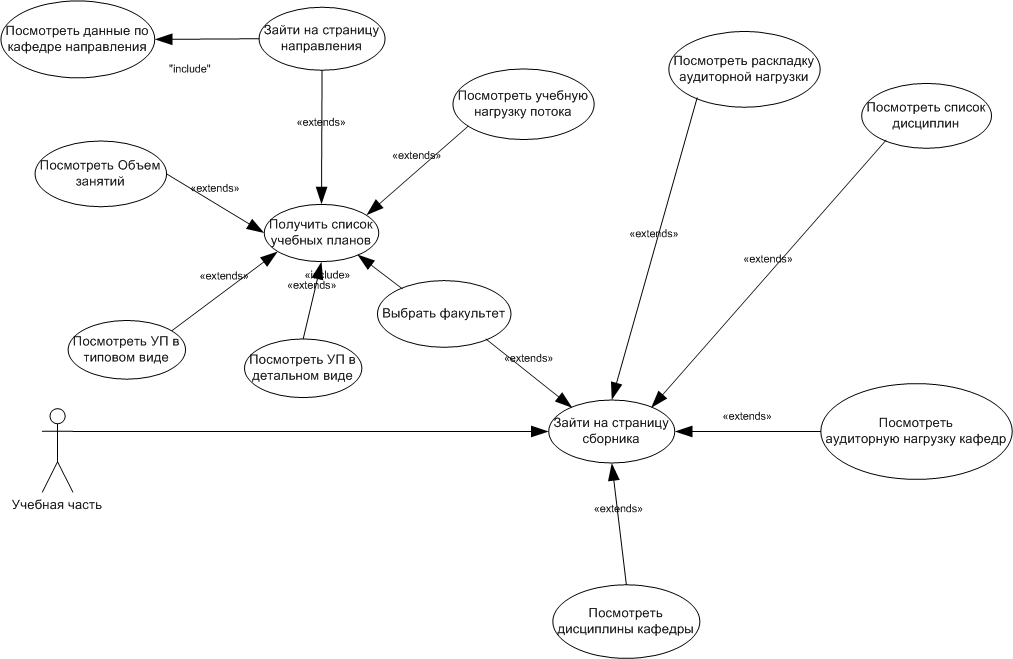


Рис. 3. 6. Диаграмма вариантов использования для незарегистрированного пользователя

Кроме того, пользователь может редактировать справочники дисциплин, циклов и компонентов. Т. к. эта информация может изменяться по мере заполнения учебных планов. Однако кафедра имеет ограниченные возможности на редактирование этих справочников. Например, пользователь может удалять или редактировать только те дисциплины, которые относятся к его кафедре.

Пользователь “Деканат” (Рис. 3. 5.) может редактировать только справочник потоков, куда он может заносить данные о потоках своего факультета.

Пользователи деканат и кафедра имеют возможность на просмотр всех справочников, включая те, правами, на редактирование которых они не обладают, а также оставлять свои замечания и предложения.

Последний вариант использования системы – это вариант для незарегистрированных пользователей (Рис. 3. 6.)

Этот вариант представляет использование системы пользователем, не имеющим прямого отношения к работе над учебными планами, но желающим получить по ним какую-либо информацию. Этот пользователь никак не может влиять на состояние базы данных учебных планов, но зато может получать различные виды отчетов по информации, содержащейся в ней.

## 3.3. Разработка диаграммы прецедентов индивидуального планапреподавателя

Рассмотрим диаграмму прецедентов индивидуального плана преподавателя.

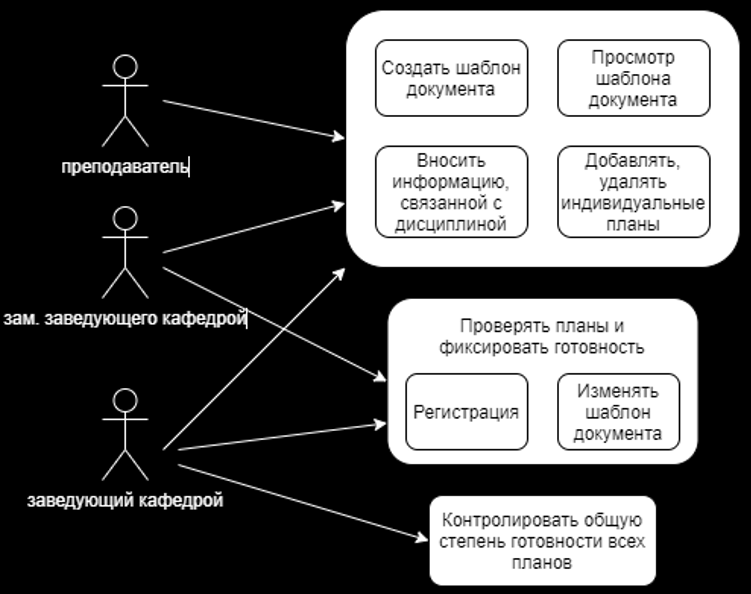


Рис. 3. 7. Диаграмма прецедентов индивидуального плана преподавателя

## 3.4. Разработка схемы базы данных

### 3.4.1 ER-диаграмма учебного плана в формате plx

Рассмотрим основную часть схемы plx подробнее. Центральной частью модели базы данных является сущность “Планы”. На физическом уровне ей соответствует таблица содержащая список кафедр, факультетов и основные атрибуты планов (форма обучения, срок обучения составитель и тд.).

Из таблиц, с которыми она связана следует отметить также таблицы “Факультеты” и “ПланыСтроки”.

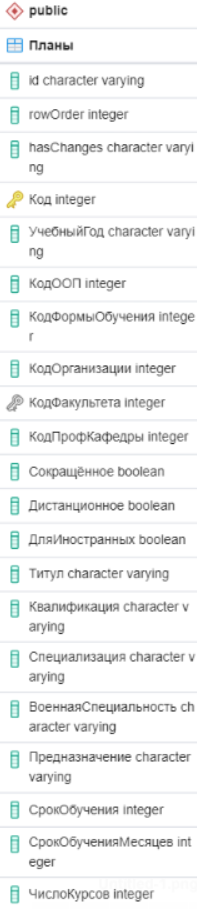


Рис. 3.8. Физическая модель таблицы “Планы”

”ПланыСтроки” – содержат список дисциплин учебного плана. В этой таблице хранится информация определяющая основные свойства дисциплины. Это код плана, к которому дисциплина относится, код дисциплины, код ведущей кафедры (использующийся при составлении отчетов), общее количество часов и т. д. Эта информация не зависит от вида учебной нагрузки или ведущей кафедры.

Вторая таблица “Факультеты” содержит список всех факультетов университета с указанием названия, сокращенного названия факультета, а также ФИО и подписи декана.

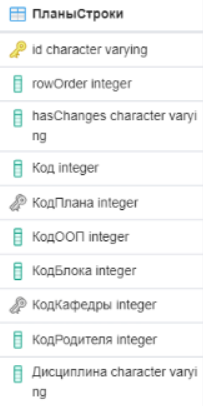


Рис. 3.9. Физическая модель таблицы “ПланыСтроки”

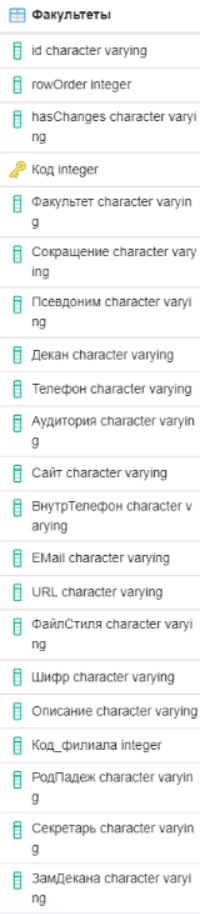


Рис. 3.10. Физическая модель таблицы “Факультеты”

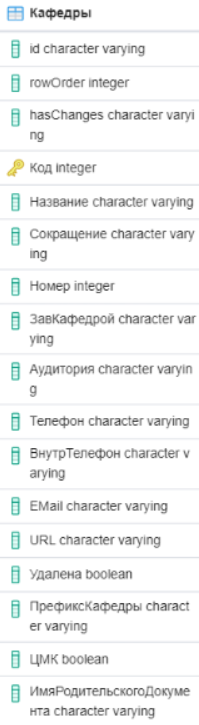


Рис. 3.11. Физическая модель таблицы “Кафедры”

Эта таблица содержит все необходимые сведения по кафедре (название кафедры, факультет к которому она относится и т. д.). Она имеет ключ “Код”, который используется в списке дисциплин учебного плана и списке нагрузок по учебному плану для отнесения всей дисциплины или конкретного вида учебной нагрузки к данной кафедре.

Из данных входных файлов (учебные планы в формате plx) были выявлены соответствующие поля и по ним были созданы таблицы в бд с соответствующими признаками.

Были выявлены следующие таблицы: ПланыНовыеЧасы, Уровень\_образования, псСтандарты, СправочникВидыПрактик, Компетенции, псГруппы, псТипыХарактеристикФункции, псОбобщенныеФункции, Заезды, ПланыРазбиения, Факультеты, ДолжностныеЛица, ПланыКомпетенцииДисциплины, СправочникТипаЧасов, Документ, ПланыПрофСтандарты, ПланыГрафикиЯчейки, diffgram, ПараметрыПлана, псФункции, СкрытьВидыРабот, КратностьЧасов, Филиалы, ПланыЦиклы, before, СправочникТипБлока, ДолжЛица\_Планы, СправочникБазы, ПланыНовыеГрафики, ПланыКомпетенции, ПланыКомпетенцииПрофСтандарты, ЗаездыНормы, СправочникВидОбъекта, ПланыКонтингент, СправочникТипаРабот, ПрограммаПодготовки, ФормаОбучения, псХарактеристикиФункций, ВидыДеятельности, ПланыСтроки, Кафедры, ПланыПрофили, ВыбранныеВидыРабот, ООП, УровеньОбразования, СправочникВидыРабот, СправочникТипОбъекта, ПланыВидыДеятельности, Планы.

Выше были описаны основные сущности данного файла, все остальные 46 таблиц будут представлены в **Приложении А:** Plx спецификация учебного плана вуза.

### 3.4.2 ER-диаграмма учебного плана

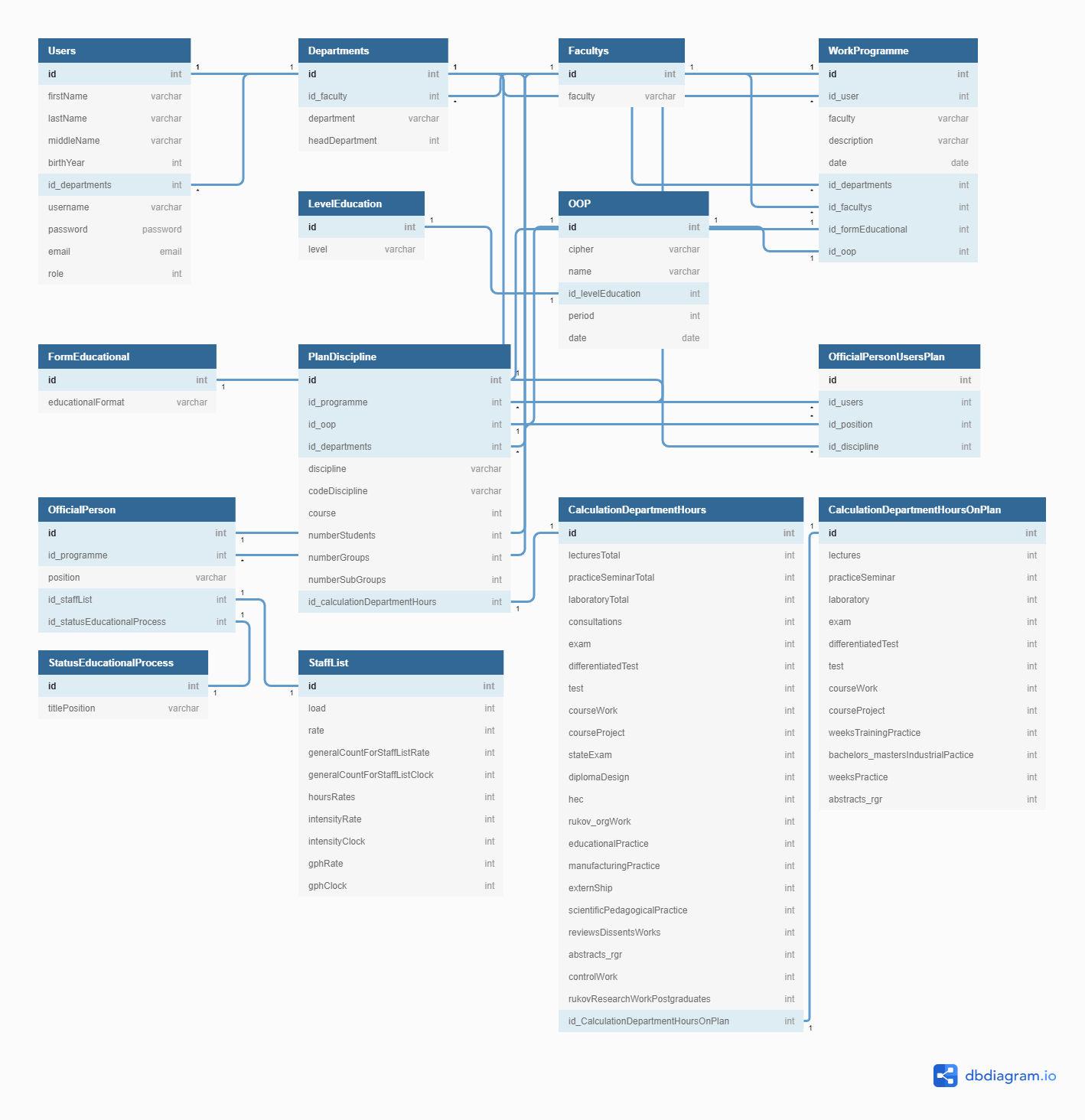


Рис. 3.12. Физическая модель базы “Индивидуальные планы преподавателя”

Выше были показаны основные сущности разрабатываемой системы, подробнее о них рассмотрим ниже.

Users – это пользователь, зарегистрировавшийся в системе, который может просматривать или создавать свои шаблоны ИП. Предполагается, что пользователь, не авторизованный в системе не имеет доступ к системе.

Departments и Facultys – таблицы с названиями кафедр и факультетов, которые используются при заполнении титульного листа ИП.

PlanDiscipline – одна из главных таблиц описывающая дисциплины, которые соответствуют текущему пользователю (преподавателю). Данная таблица используется в связке с 3 таблицами – CalculationDepartmentHours, StaffList, OfficialPerson. За каждой дисциплиной закреплена рабочая нагрузка, которая описывается в отдельной таблице CalculationDepartmentHours. В ней указываются часы на проведение лекция, практик, лабораторный занятий, экзаменов, зачетов и тд.

У каждого преподавателя есть своя должность в вузе и к каждой должности закрепляется объем рабочей нагрузки, которая учитывается в штатном расписании кафедры, помимо должности у пользователя есть статус в учебном процессе:

* Штатный;
* Внутр.совм.;
* Внешний совм.;
* Работодатель;
* Договор ГПХ.

Все выше перечисленные должности описываются в таблице StaffList, которая в свою очередь связана с промежуточной таблицей OfficialPerson с помощью, которой каждый пользователь может получить данные о текущей программе для закрепленный за преподавателем дисциплин.

# Глава 4. Описание реализации подсистемы

## 4.1. Процесс трансформации XML спецификаций учебных планов вуза в SQL базу данных

В задаче трансформации XML спецификаций в SQL базу данных, стоит анализа разметки и создания парсинга данных по записи в базу данных.

XML или расширяемый язык разметки, который обычно используется для структурирования, хранения и передачи данных между системами. Хотя он и не так распространен, как раньше, он все еще используется в таких службах, как RSS [35] и SOAP [36], а также для структурирования файлов. Поскольку Python [37] является популярным языком для Интернета и анализа данных, вполне вероятно, что в какой-то момент вам понадобится читать или записывать XML-данные, и в этом случае вам повезло.

Эффективная автоматизация перечисленных процессов требует ввода спецификаций планов в реляционные базы данных. В крупном вузе количество планов может составлять большие сотни, что определяет взятие решения, что нужно использовать распараллеливания данных процессов ввода.

Модуль DOM [38]– это упрощенная реализация объектной модели документа. DOM – это интерфейс прикладного программирования, который рассматривает XML как древовидную структуру, где каждый узел в дереве является объектом. Таким образом, использование этого модуля требует, чтобы мы были знакомы с его функциями. Модуль ElementTree [39] предоставляет более «питонический» интерфейс для работы с XMl и является хорошим вариантом для тех, кто не знаком с DOM.

Ниже описываются результаты экспериментального исследования инерционности параллельных реализаций функций ввода XML спецификаций в SQL базу данных.

## 4.2. Оценка эффекта от распараллеливания процесса ввода xml спецификаций учебных планов вуза в sql базу данных

В задаче экспериментирования с процедурами параллельного вычисления [40] имеется два критериальных параметра: время вычисления и степень ускорения, определяемая отношением времени последовательной обработки ко времени параллельной обработки.

### 4.2.1. Организация экспериментов

Значения критериальных параметров зависят от двух факторов: размера рабочей нагрузки на процедуру и количества параллельных ветвей.

Для эксперимента на языке Python был написан парсер xml(plx) файлов, который анализирует содержимое спецификаций учебных планов и записывает обработанные данные в таблицы базы данных.

Эксперименты организуются как двухфакторные: первым фактором является число параллельно работающих потоков, вторым – размер рабочей нагрузки. Варьирование размера рабочей нагрузки обеспечивается через изменение количества XML-файлов.

Для создания достаточно большого числа файлов создан генератор, который формирует заданное число экземпляров учебных планов путем создания копий двух реальных учебных планов.

План эксперимента представляет собой сочетание всех значений, указанных выше факторов. Число потоков берется из множества {1,2,3,4}, а число файлов из множества {10, 25, 50, 100, 150}.

### 4.2.2. Проведение экспериментов и анализ результатов

Для организации параллелизма на Python была использована библиотека multiprocessing [41], для замера времени – функция time(), которая возвращает время в секундах с начала эпохи как число с плавающей запятой.

На основе средств библиотеке multiprocessing процессы порождаются путём создания объекта Process и последующего вызова его метода start().

Последовательность операций, выполняемых в одной итерации программы проведения экспериментов, такова:

* Проверка файлов на существования их в базе данных;
* Разбиение множества файлов рабочей нагрузки на части, каждая из которых реализуется в одном потоке;
* Порождение процессов и их запуск с фиксацией стартового времени;
* В каждом потоке в объекты Python-программы вводятся данные из xml;
* (plx) файлов соответствующей части рабочей нагрузки и затем выводятся через SQL-запросы в базу данных с последующей фиксацией времени завершения.

В экспериментах использовался компьютер на базе процессора I5 – 5700U (2,5 ГГц, 2 ядра, 4 потока), имеющий оперативную память 8 GB.

В первую серию экспериментов были вовлечены файлы с размером 829 КБ. Результаты измерения времени вычисления работы программы при 153 различных сочетаниях значений факторов для этой серии экспериментов приведены в таблице 4.1.

График зависимости коэффициента ускорения от числа потоков при различных значениях рабочей нагрузки приведен на рисунке 4.1.

Таблица 4.1. Результаты замеров времени (ceк) вычисления

при размере одного файла 829 КБ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Потоки** | **10** | **25** | **50** | **100** | **150** |
| **1** | 7,2677302 | 18,5514128 | 36,7133557 | 82,0052709 | 110,6746280 |
| **2** | 5,5745096 | 11,8321518 | 27,2187612 | 43,9257984 | 65,4179015 |
| **3** | 4,2266352 | 9,4364001 | 18,4481730 | 35,6156911 | 54,6968827 |
| **4** | 3,6308805 | 9,6784853 | 15,7115521 | 31,4356100 | 44,7100415 |

Рис. 4.1. Графики коэффициентов ускорения при варьировании числа потоков

Вторая серия экспериментов проведена с файлом, имеющим размер 3082КБ и ее результаты представлены в таблице 4.2 и на рис. 4.2.

Таблица 4.2. Результаты замеров времени (ceк) вычисления

при размере одного файла 3082КБ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Потоки** | **10** | **25** | **50** | **100** | **150** |
| **1** | 30,3956904 | 76,0451130 | 152,6322774 | 312,3548293 | 449,4782640 |
| **2** | 17,9967038 | 43,4448020 | 85,6357100 | 181,7410304 | 272,0168783 |
| **3** | 16,0176477 | 38,7935459 | 80,5575785 | 152,7809765 | 235,8286857 |
| **4** | 13,2225451 | 35,3233423 | 62,3538620 | 131,6907467 | 198,1465356 |

Рис. 4.2. Графики коэффициентов ускорения при варьировании числа потоков

Из полученных результатов видно, что время, затраченное на трансформацию множества plx-файлов учебных планов в единую SQL базу данных (рис. 3.6. ER-диаграмма базы данных), пред сильно зависит от размера файла, однако определяется не только им. Эффект от распараллеливания довольно заметен, несмотря на то, что значительную долю затрат занимает собственно ввод из файлов. Эффект от распараллеливания для четырех потоков достигает 2,5. Учитывая, что в используемой платформе всего 2 ядра и дополнительные два потока в режиме hyperthreading [42] разделяют ресурсы этих двух ядер, ускорение можно считать существенным.

Экспериментальные исследования показали целесообразность распараллеливания процессов трансформации plx-файлов учебных планов в SQL базу данных даже в среде бюджетного персонального компьютера, не имеющего достаточно большой кэш-памяти и расслоения данных в дисковой RAID системе. Коэффициент ускорения для четырех потоков в двухъядерной архитектуре лежит в пределах между 2 и 2,5.

За счет такого ускорения экономия времени для массива из 150 учебных планов размером в 3 МБ составляет более 4 минут. Это означает, что распараллеливание можно рекомендовать в сетевых сервисах, обрабатывающих большие массивы plx-файлов. Например, в условиях глобальной проверки, касающейся многих вузов, либо в одном крупном вузе с числом учебных планов в несколько тысяч.

Проведенные эксперименты позволили выявить потенциал улучшений исследуемой программы. Этот потенциал касается повышения рациональности распределения рабочей нагрузки на потоки, а также разделения фаз ввода-вывода и фаз обработки таким образом, чтобы уменьшить затраты времени на ожидание в очереди доступа к разделяемым ресурсам.

## 4.3. Экспериментальные исследования реализации учебных планов преподавателей вуза

### 4.3.1. Формулирование гипотез

В рамках данной работы будет проводиться эксперимент над функцией выбора всех дисциплин, которые ведет кафедра. Эта функция имеет вид:

fSelectDisc: splans x sdepartment => sdisc\*, где

sdisc\* ∈ sdiscipline

Таким образом, не взирая на степень вовлечения содержания учебного плана в реализацию тех или иных функций организации учебного процесса, все они обслуживают так или иначе реализацию совокупности учебных планов.

Перед проведением тестирования сформулируем гипотезы [43], которые и будут проверяться при проведении экспериментов.

**Гипотеза 1:** Очевидно, что при росте количества одновременных соединений и количество потоков, используемых нагрузочной утилитой, TPS [44] будет линейно расти до определенного порога. Чтобы преодолеть данных порог можно использовать индексы.

**Гипотеза 2:** Метрика TPS имеет обратную зависимость от количества строк в таблице.

**Гипотеза 3:** По сколько в запросе выполняет join для двух таблиц, то можем сделать индексы на поля, к которым делается join. Это существенно ускорит select запрос.

**Гипотеза 4:** Добавление индексов к полям join, замедлит обновление и вставку строк в таблицу, поскольку каждый раз нужно будет перестаивать этот самый индекс.

Сейчас в качестве СУБД в проекте используется PostgresSQL [45]. В таком случае может быть несколько вариантов для оптимизации. Например, кеширование результатов некоторых запросов, добавление дополнительных индексов, нормализация таблиц, введение репликации и т. д. Но вероятность того, что проблема производительности будет именно в базе данных очень мала, так как в данный момент в прототипе совершается довольно мало транзакций и не выполняется никаких тяжелых запросов.

Так же есть вероятность, что мы упремся в оперативную память. Вероятность этого так же мала, но, если это произойдет можно либо нарастить память на сервере, либо не хранить данные в оперативной памяти, а вынести их в БД или другое хранилище.

Возможно проблема с производительностью будет связана с ограничениями пропускного канала сети. Если такая проблема действительно возникнет, то с ней нужно будет разбираться отдельно, так как на ширину пропускного канала мы повлиять не сможем. Но такая проблема так же скорее все не возникнет, так как сейчас в проекте происходит довольно мало сетевых взаимодействий.

Наиболее вероятно, что узким местом станет процессор, так как именно на него приходится большая часть нагрузки при обработке запросов. В таком случае можно масштабировать систему вертикально или горизонтально. При вертикальном масштабировании системы мы просто поставим на сервере процессор с большим количеством ядер. Но при вертикальном масштабировании мы рано или поздно упремся в потолок по производительности, так как очевидно нельзя до бесконечности наращивать мощности одного сервера. Тогда будем масштабироваться горизонтально и распределим нагрузку между несколькими серверами.

### 4.3.2. Разработка планов экспериментов

**Эксперимент 1: исследование зависимости TPS от рабочей нагрузки**

В данном эксперименте проверяются первую гипотезу, а именно: метрика TPS будет линейно расти до определенного порога, при возрастании нагрузки.

Факторами в данном эксперименте будут количество одновременных соединений и количество потоков, проверяемый параметр TPS – количество транзакций в секунду.

Теперь выберем запрос, который будет проверяться на нагрузке. Реально один из наиболее популярных запросов на кафедре является выборка дисциплин, которые ведутся на кафедре.

Распишем требования к проведению эксперимента:

* Необходимо запускать один и тот же скрипт несколько раз, так как при последующих запусках может отработать кэш, что может заметно повлиять на результаты тестирования;
* Для большей точности измерения TPS каждый нагрузочный тест должен продолжаться не менее 30 секунд;
* Для проведения экспериментов необходимо развернуть тестовое окружение. Тестировать производительность на локальной машине нельзя, так как большую часть ресурсов процессора и оперативной памяти будут занимать сторонние процессы, соответственно результат нагрузочного тестирования будет не репрезентативным. Для проведения стрессового тестирования имеет смысл развернуть отдельный сервер, на котором будет работать только тестируемое приложение. Проще всего для этих целей будет развернуть облачный сервер.

Напишем формулу для вычисления целевого параметра TPS. Из названия метрики очевидно, что она рассчитывается по формуле:

TPS =

Теперь выразим формулой наши предположения относительно результатов эксперимента.

RPS = , где

То есть по нашим предположениям TPS линейно зависит от количества потоков и соединений, обрабатываемых в одном потоке.

**Эксперимент 2: исследование зависимости TPS от количества строк в базе данных**

В данном эксперименте проверяются вторую гипотезу, а именно: метрика TPS имеет обратную зависимость от количества строк в таблице.

Факторами в данном эксперименте будут количество строк в таблицах “Кафедры” и “ПланыСтроки”. Проверяемый параметр TPS – количество транзакций в секунду.

Запрос, который будет проверяться на нагрузке – выборка дисциплин, которые ведутся на кафедре.

Теперь выразим формулой наши предположения относительно результатов эксперимента.

1. TPS =

То есть по нашим предположениям TPS линейно зависит от количества ядер процессора.

**Эксперимент 3: исследование зависимости TPS от наличия индекса**

В данном эксперименте проверяются третью и четверную гипотезу, а именно: ускорение выборки и замедление вставки при наличии индекса.

Факторами в данном эксперименте будет индекс.

Запрос, который будет проверяться на нагрузке – выборка дисциплин, которые ведутся на кафедре.

Проверяемый параметр TPS – количество транзакций в секунду.

### 4.3.3. Формулирование гипотез

Для проведения экспериментов была развернута тестовая бд, в среде Postgres, с инструментом стресс-теста – pgbench [46]. Таблицы в базе данных были заполнены данными из файлов plx, содержимое которого состоит из спецификаций учебных планов.

Как тестирующая утилита была взята pgbench – это простая программа для запуска тестов производительности Postgres. Она многократно выполняет одну последовательность команд, возможно в параллельных сеансах базы данных, а затем вычисляет среднюю скорость транзакций (число транзакций в секунду). По умолчанию pgbench тестирует сценарий, примерно соответствующий TPC-B, который состоит из пяти команд SELECT, UPDATE и INSERT в одной транзакции. Однако можно легко протестировать и другие сценарии, написав собственные скрипты транзакций.

1. Теперь можно приступить непосредственно к тестированию. Команда для запуска такого теста будет выглядеть следующим образом:
2. *sudo -u postgres pgbench -f script.sql postgres*
3. В файле script.sql храниться наш запрос, а postgres – это название бд с которой мы работает в данный момент.
4. Будем запускать один и тот же скрипт несколько раз, так как при последующих запусках может отработать кэш, что заметно повлияет на результаты тестирования.
5. Перейдем к **перовому** экспериментальному тестированию.
6. *sudo -u postgres pgbench -c -j -f script.sql postgres , где*
7. -с – количество смоделированных клиентов, то есть количество параллельных сессий базы данных. По умолчанию 1.
8. -j – количество рабочих нитей в pgbench. Использование более чем одного потока может быть полезно на многопроцессорных машинах. Клиенты распределены между доступными потоками как можно более равномерно. По умолчанию 1.

После проведения первого эксперимента зафиксируем количество соединений (50) и потоков (4).

*sudo -u postgres pgbench -P 10 -T 30 -c 50 -j 4 -f script.sql postgres, где*

-P – Выводить отчёт о прогрессе через заданное число секунд (сек). Выдаваемый отчёт включает время, прошедшее с момента запуска, скорость (в TPS) с момента предыдущего отчёта, а также среднее время ожидания транзакций и стандартное отклонение. В режиме ограничения скорости (-R) время ожидания вычисляется относительно назначенного времени запуска транзакции, а не фактического времени её начала, так что оно включает и среднее время отставания от графика.

-T – Выполнять тест с ограничением по времени (в секундах), а не по числу транзакций для каждого клиента. Параметры -t и -T являются взаимоисключающими.

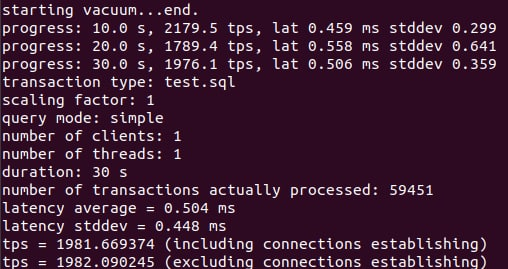


Рис. 4.3. Первый эксперимент, зависимость TPS от рабочей нагрузки

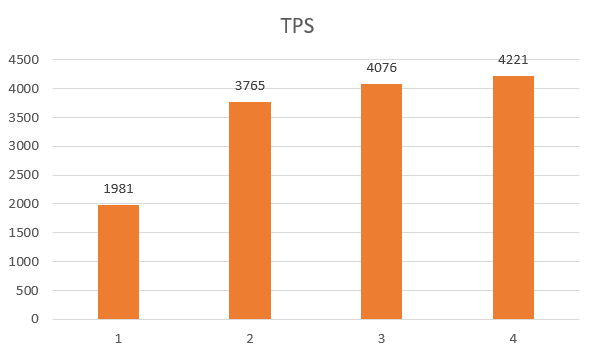


Рис. 4.4. Первый эксперимент, зависимость TPS от рабочей нагрузки

Из графика мы видим, что первые два столбца имеют линейную зависимость от рабочей нагрузки. При 3-х и более соединений, мы видим, что возрастания TPS не происходит, это говорит нам о том, что мы достигли порогового значения.

Перейдем ко **второму** экспериментальному тестированию.

*sudo -u postgres pgbench -P 10 -T 30 -c 50 -j 4 -f test.sql postgres*

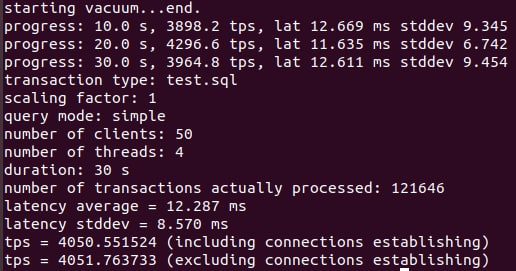


Рис. 4.5. Второй эксперимент

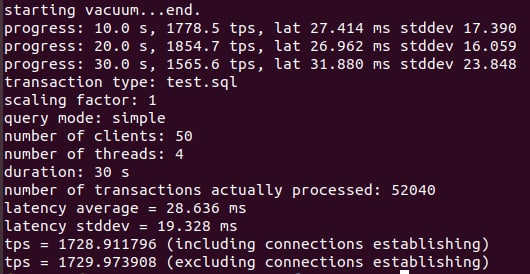


Рис. 4.6. Второй эксперимент, удвоили кол-во данных в таблицах

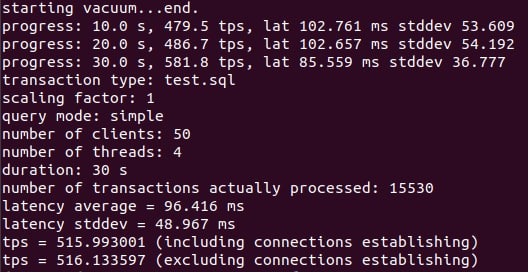


Рис.4.7. Второй эксперимент, учетверили кол-во данных

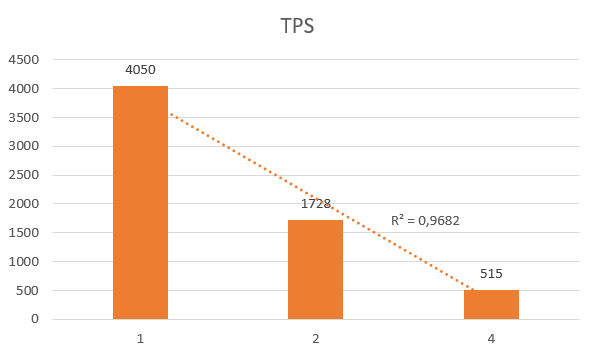


Рис. 4.8. Второй эксперимент – исследование зависимости TPS от количества строк в базе данных

Из гистограммы видно, что с увеличением объема данных линейно падает TPS. Построив линию тренда, мы убедились то, что стремиться к 1, а это значит, что зависимость действительно линейна.

1. Перейдем ко **третьему** экспериментальному тестированию.
2. *sudo -u postgres pgbench -P 10 -T 30 -c 50 -j 4 -f test.sql postgres*

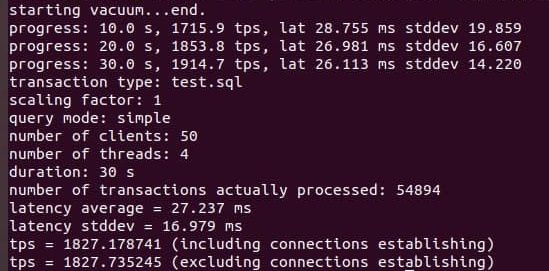


Рис. 4.9. Третий эксперимент, добавили индексы

При добавлении индексов на поля, которые используется в тестовом запросе, мы видим значительное увеличение значения TPS. Проверить то, что используются индексы, можно при использовании explain, который показывает план запроса.



Рис.4.10. План запроса без индексов

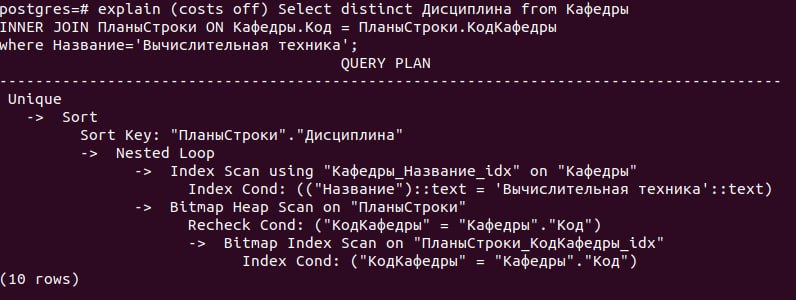


Рис.4.11. План запроса с индексами

## 4.4. Обзор реализации АСУ Индивидуальный план преподавателя подсистемы “Учебные планы”

Индивидуальный план является нормативным документом, устанавливающим требования, порядок и правила составления индивидуального плана работы лиц профессорско-преподавательского состава вуза, занимающих штатные должности профессоров, доцентов, старших преподавателей, преподавателей и ассистентов кафедр на полных и неполных ставках. Стандарт устанавливает нормы времени для расчета учебной работы кафедр и преподавателя, примерные нормы времени для расчета трудоемкости учебнометодических, научно-исследовательских и научно-методических, организационно-методических и других видов работ.

Индивидуальный план работы преподавателя охватывает все виды поручений, выполняемых преподавателем в учебном году, и является основным документом, конкретизирующим должностные обязанности преподавателя из раздела «Права и обязанности работника» трудового договора, заключенного между преподавателем и университетом.

Он ежегодно составляется на предстоящий учебный год и оформляется на типовом бланке установленного образца (Форма ИП представлена в **Приложение В:** Шаблон индивидуального плана).

Индивидуальный план преподавателя является основным документом, регламентирующим планирование и выполнение:

* Аудиторная нагрузка;
* Учебно-методической работы;
* Научно-исследовательской работы;
* Учебно-воспитательной и организационно-методической работы;
* Сводная таблица за год.

Объем всех поручений преподавателя исчисляется в часах.

WEB-версия редактора может претерпеть некоторые изменения на основе полученных функциональных моделей. На данном этапе разработке редактор выглядит как меню, с семантическое выбором поле в индивидуальном плане. Для каждой смысловой части было создано краткое введение для текущего шага редактора, которое появляется только при первом посещении шага редактора. Ниже, на рисунках приведены примеры модулей отображение.

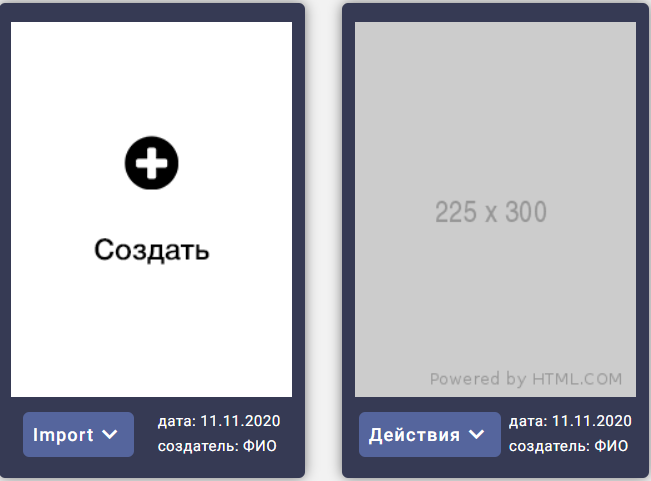


Рис. 4.12. Модуль редактора титульного листа

Также для каждой из модулей было создано краткое введение для текущего шага редактора, которое появляется только при первом посещении шага редактора.

Пример интро для Второй половины дня:

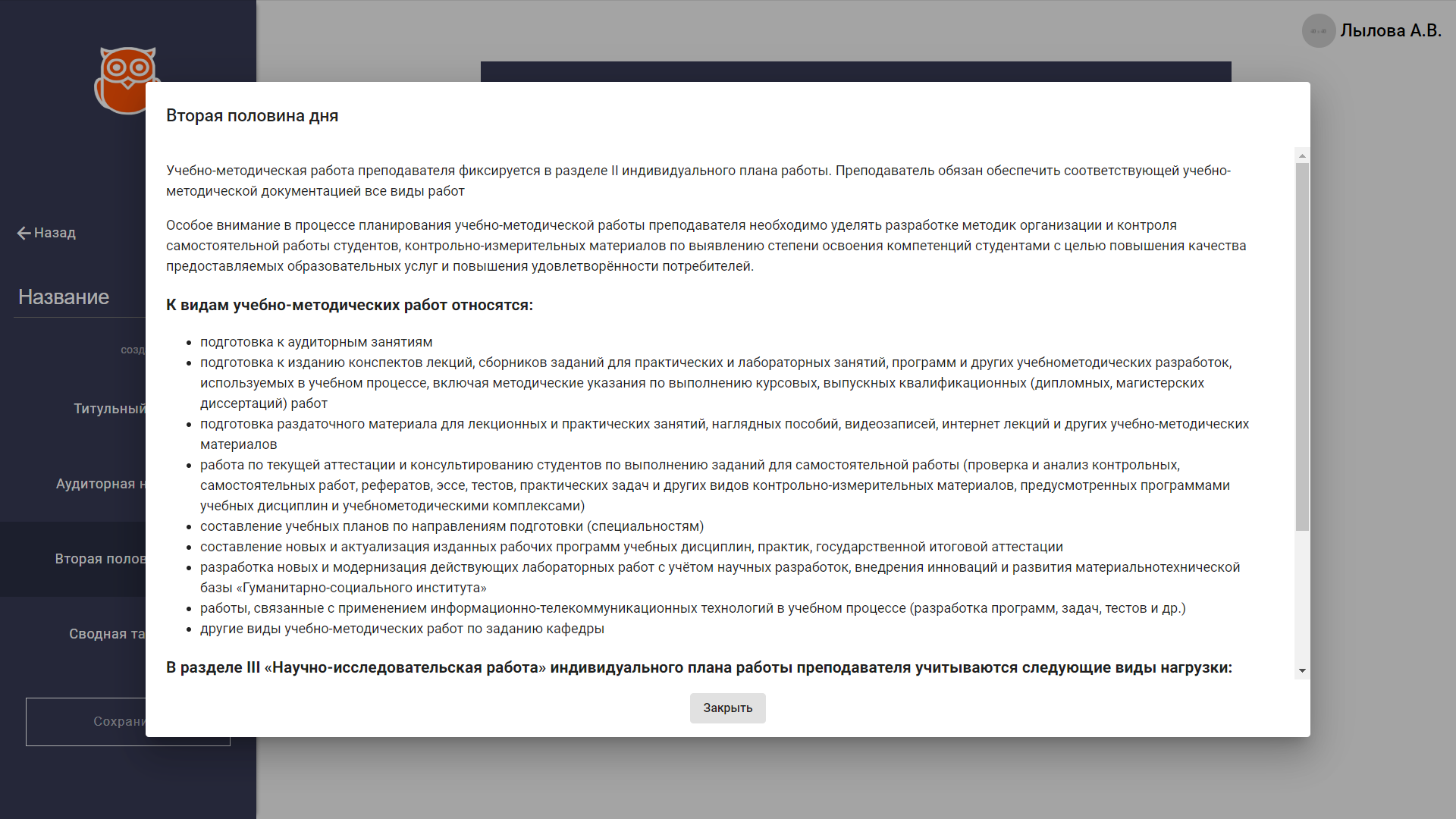


Рис. 4.13. Введение в модуль редактора второй половины дня

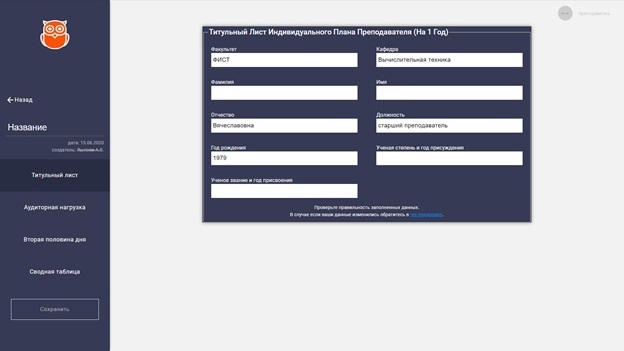


Рис. 4.14. Модуль редактора титульного листа

Данный титульный лист оформлен в виде простой таблицы, в которой авто-подставляются данные с БД. Преподаватель на данном шаге проверяет правильность заполнения личных данных. Также данная таблица легко изменяемая и не требует использования лишних строк html кода для создания дополнительных или же удаления полей.

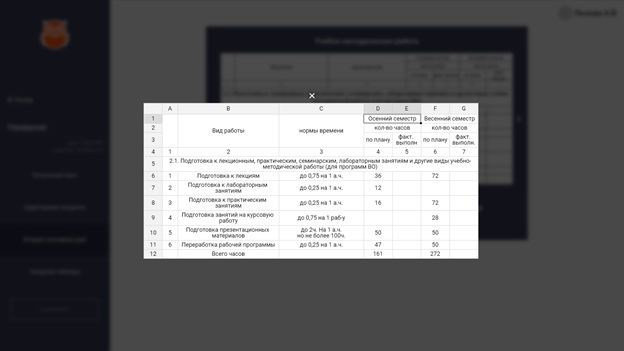


Рис. 4.15. Модуль модального редактора таблицы

В данном компоненте выбирается нужная таблица для редактирования после чего при нажатии на превью таблицы открывается модальное окно с редактором таблицы.

Данное модально окно подгружает выбранную таблицу из шаблона, в данном примере используется шаг второй половины дня, в которой данные задаются самим преподавателя, в магистратуре планируется реализовать контроллер для автоматизирования заполнения второй половины дня.

# Заключение

В результате проведенной работы был проведен анализ учебных процессов, базирующихся на учебных планах ВУЗа.

Был разработан ряд моделей, описывающий систему автоматизации. Эти модели послужили материалом для разработки системы и могут быть использованы в дальнейшем для разработки подобных систем и расширения функциональности уже существующей.

При этом было исследована предметная область «Автоматизированных рабочих мест» и «Автоматизированных систем». Была разработана система автоматизации процессов связанных с системой учебных планов вуза.

# Список использованных источников

1. Информационная система «Планы». URL: <https://www.mmis.ru/programs/plany>
2. Программа для работы с электронными таблицами. URL: <https://www.microsoft.com/ru-ru/microsoft-365/excel>
3. Требования к отраслевой информационной системе сферы образования Российской Федерации. URL: <http://www.consultant.ru/document/cons_doc_LAW_140174/07b2a433c35195c28c56a221a970dc3ce333fade/>
4. Концепция создания интегрированной информационной системы Минобразования России. URL: <https://edu.ru/documents/view/10436/>
5. Учебный план вузов. URL: <https://disshelp.ru/blog/uchebnyj-plan-v-vuze-eto/#Структура_учебного_плана>
6. Федеральный государственный стандарт высшего образования. URL: <http://www.consultant.ru/document/cons_doc_LAW_142304/>
7. Расчет штатов – Режим доступа: электронная таблица excel
8. Индивидуальный план преподавателя на 18-19 год – Режим доступа: ксерокопия
9. Сравнение Системы управления образовательным процессом. URL: <https://soware.ru/categories/education-management-systems>
10. АСУ ВУЗ. URL: <https://vuz.osvita.net/ru/>
11. Программный продукт "1С: Университет". URL: <https://solutions.1c.ru/catalog/university/features>
12. Информационная система «Планы». Сайт ММИС-Лаборатории. URL: <https://www.mmis.ru/programs/plany>
13. Система «Вектор». URL: <http://www.snhpro.ru/about/Vector.pdf>
14. «Магеллан» - комплексная информационная система для университета, института, академии, курсов повышения квалификации. URL: <https://magellanius.ru/>
15. Автоматизация учебных процессов связанных с си-темой учебных планов ВУЗа. ВКР. Карпова.
16. Фирма "1С". URL: <https://1c.ru/>
17. ФГОС. URL: <https://fgos.ru/>
18. АС «Учебная нагрузка». URL: <https://www.mmis.ru/programs/nagruzka>
19. РосНОУ. URL: <https://rosnou.ru/>
20. Анализ и управление бизнеспроцессами. URL: <https://books.ifmo.ru/file/pdf/2017.pdf>
21. Система управления учебным процессом для вуза. URL: <https://dspace.tltsu.ru/bitstream/123456789/273/1/Гущина%201-06-13.pdf>
22. Федеральная служба по надзору в сфере образования и науки. URL: <https://obrnadzor.gov.ru/>
23. Форма N ВПО-1 «Сведения об организации, осуществляющей образовательную деятельность по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры». URL: <https://minobrnauki.gov.ru/action/stat/highed/>
24. Рабочие программы СПО. URL: <https://mosmetod.ru/files/metod/spo/ych_disceplin.pdf>
25. Internet. URL: <https://trends.rbc.ru/trends/industry/6223a37a9a79472fe845c934>
26. Защита информации от несанкционированного доступа. URL: <https://searchinform.ru/services/outsource-ib/zaschita-informatsii/ot-nesanktsionirovannogo-dostupa/>
27. XML. URL: <https://support.microsoft.com/ru-ru/office/xml-для-начинающих-a87d234d-4c2e-4409-9cbc-45e4eb857d44>
28. Математическое моделирование. URL: <https://moodle.kstu.ru/pluginfile.php/402080/mod_resource/content/1/Лекция1.pdf>
29. Синицын С.В., Налютин Н.В. Верификация программного обеспечения - М.: Московский инженерно-физический институт (государственный университет), 2006. - 157с.
30. Соловьев В.Д, Добров Б.В. Онтологии и тезаурусы - Казань, Москва: Учебное пособие, 2006. - 157 с.
31. Карпов В.Э. Онтологии. - 33 с.
32. ОНТОЛОГИЯ. Основные понятия, модели, классификация. URL: <http://mei06.narod.ru/sem6/pz/shpora/lec7.htm>
33. Замаруева И. В. Математические модели семантики свободных словосочетаний с родо-видовими компонентами и их применение в АИС. – Дис… кан. техн. наук. – Харьков : ХТУРЭ, 1990. – 170 с.
34. Шаронова Н.В. Математические модели суффиксального словообразования и их использование в системах автоматической обработки текстов на русском языке. – Дис… кан. техн. наук. – Харьков : ХТУРЭ, 1984. – 217 с.
35. RSS. URL: <https://ru.wikipedia.org/wiki/RSS>
36. SOAP. URL: <https://habr.com/ru/post/591573/>
37. Python. URL: <https://www.python.org/>
38. Объектная модель документа: что такое DOM и чем не является? URL : <https://proglib.io/p/what-is-dom>
39. The ElementTree XML. URL: <https://docs.python.org/3/library/xml.etree.elementtree.html>
40. Гафаров Ф.М. Г12 Параллельные вычисления: учеб. пособие / Ф.М. Гафаров, А.Ф. Галимянов. – Казань: Изд-во Казан. ун-та, 2018. – 149 с.
41. Многопоточность в Python. URL: <https://docs-python.ru/tutorial/mnogopotochnost-python/>
42. Hyper-threading. URL: <https://ru.wikipedia.org/wiki/Hyper-threading>
43. Семь видов гипотез. URL: <https://habr.com/ru/company/hopox/blog/299882/>
44. Как измерять производительность блокчейн сетей. Основные метрики. URL: <https://habr.com/ru/post/473846/>
45. PostgresSQL. URL: <https://www.postgresql.org/>
46. Pgbench. URL: <https://postgrespro.ru/docs/postgresql/12/pgbench>
47. Кублашвили, О. В. Автоматизированные системы документационного обеспечения управления : учеб. пособие / О. В. Кублашвили. – Москва : МГУП, 2009. – 106 с.;
48. Кудрявцев, А. В., Система автоматизированной генерации рабочих программ дисциплин на основе сетевой базы данных / Кудрявцев, А. В. // ИКТ В ОБРАЗОВАНИИ – 2009. С. 68 – 73
49. Чистая архитектура. Искусство разработки программного обеспечения. — СПб.: Питер, 2018. — 352 с.: ил. — (Серия «Библиотека программиста»). ISBN 978-5-4461-0772-8
50. Бурков В. Н., ИриковВА. Модели и методы управления организационными системами. М.: Наука, 1994.
51. Гвишиани Д.М. Организация и управление. М.: Наука, 1972.
52. ГраберМартин. Введение в SQL. М.: Издательство "Лори", 1996.
53. Глушков В.М. Введение в АСУ. Киев, "Техника", 1972.
54. Дейт К. Введение в системы баз данных. М.: Диалектика, 1998.
55. Зиглер К. Методы проектирования программных систем. М.: Мир, 1985.
56. Шрейдер Ю.А., Шаров А.А. Системы и модели. М.: Радио и связь, 1982.
57. Шорин В.Г. Введение в автоматизированные системы управления. -М.: Знание, 1974.
58. Цыгичко В.Н., Клоков В.В. Основные принципы описания сложных организационных систем // Диалектика и системный анализ. М.: Наука, 1986.
59. Оценка компаний: Анализ и прогнозирование с использованием отчетности. URL: <http://mdk-arbat.ru/book/7890>
60. jExcel CE is a JavaScript plugin. URL: <https://github.com/paulhodel/jexcel>
61. Зачем нам UML? Или как сохранить себе нервы и время. URL: <https://habr.com/ru/post/458680/>
62. ПОЛОЖЕНИЕ ОБ ИНДИВИДУАЛЬНЫХ ПЛАНАХ РАБОТЫ ПРЕПОДАВАТЕЛЕЙ. URL: http://www.rzgmu.ru/images/upload/subdivisions/296\_06102017.pdf

# Приложения

## Приложение А Parser XML спецификаций учебных планов вуза в SQL базу данных

import xml.etree.ElementTree as ET

import configparser

import psycopg2

import datetime

import os

import multiprocessing

import time

request = {}

def chunks(lst, chunk\_count):

listChunk = []

chunk\_size = len(lst) // chunk\_count

for i in range(0, chunk\_size \* chunk\_count, chunk\_size):

listChunk.append(lst[i:i+chunk\_size])

i = 0

for j in range(chunk\_size \* chunk\_count, len(lst)):

listChunk[i].append(lst[j])

i += 1

return listChunk

def files(path):

for file in os.listdir(path):

if os.path.isfile(os.path.join(path, file)):

yield file

def connect(conf):

conn = None

try:

print("Connecting to the PostgreSQL database...")

conn = psycopg2.connect(

user = conf['DATABASE']['USERNAME'],

password = conf['DATABASE']['PASSWORD'],

host = conf['DATABASE']['HOST'],

port = conf['DATABASE']['PORT'],

database = conf['DATABASE']['DB']

)

except (Exception, psycopg2.DatabaseError) as error:

raise SystemExit(error)

print("Connection successful")

return conn

def insert(connection, table, fields, value):

try:

cursor = connection.cursor()

insertQuery = f"INSERT INTO {table} ({fields}) VALUES ({value})"

cursor.execute(insertQuery)

connection.commit()

cursor.close()

except (Exception) as error:

raise SystemExit("Ошибка при работе с PostgreSQL", error)

def realName(elem):

lbracket = elem.find('{')

rbracket = elem.find('}')

if(lbracket != -1 and rbracket != -1):

return elem[rbracket + 1 : ]

else:

return elem

def main(calc, proc, config, listFiles):

print(f"Запускаем поток № {proc}")

connection = connect(config)

with open("log", "a", encoding="utf-8") as file:

for fileName in listFiles:

tree = ET.parse("files/" + fileName)

tagList = []

for elem in tree.iter():

tagList.append(elem.tag)

tagList = list(set(tagList))

for elem in tagList:

request[realName(elem)] = []

for tag in tagList:

for item in tree.iter(tag):

tableName = realName(item.tag)

tempColumns = []

tempValues = []

for key in item.attrib:

tempColumns.append(f"\"{realName(key)}\"")

tempValues.append(f"'{item.attrib[key]}'")

if("Документ" != tableName):

tempColumns.append(f"\"ИмяРодительскогоДокумента\"")

tempValues.append(f"'{fileName}'")

request[tableName].append({"columns": tempColumns, "values": tempValues})

for table in request:

for data in request[table]:

insert(connection, f"\"{table}\"", ', '.join(data['columns']), ', '.join(data['values']))

# file.write(f"Записи успешно добавлены ​​в базу данных '{config['DATABASE']['DB']}' из файла {fileName} \n")

connection.close()

print("Соединение с PostgreSQL закрыто")

file.close()

request.clear()

print(f"{calc} циклов вычислений закончены, по файлам {listFiles}. Процессор № {proc}")

def processesed(procs, config, listFiles):

# procs - количество ядер

processes = []

# делим вычисления на количество ядер

for proc in range(procs):

p = multiprocessing.Process(target=main, args=(len(listFiles[proc]), proc, config, listFiles))

processes.append(p)

p.start()

# Ждем, пока все ядра завершат свою работу.

for p in processes:

p.join()

if \_\_name\_\_ == "\_\_main\_\_":

config = configparser.ConfigParser()

config.read("conf.ini")

filesDirectory = list(files("files"))

n\_proc = 1

start = time.time()

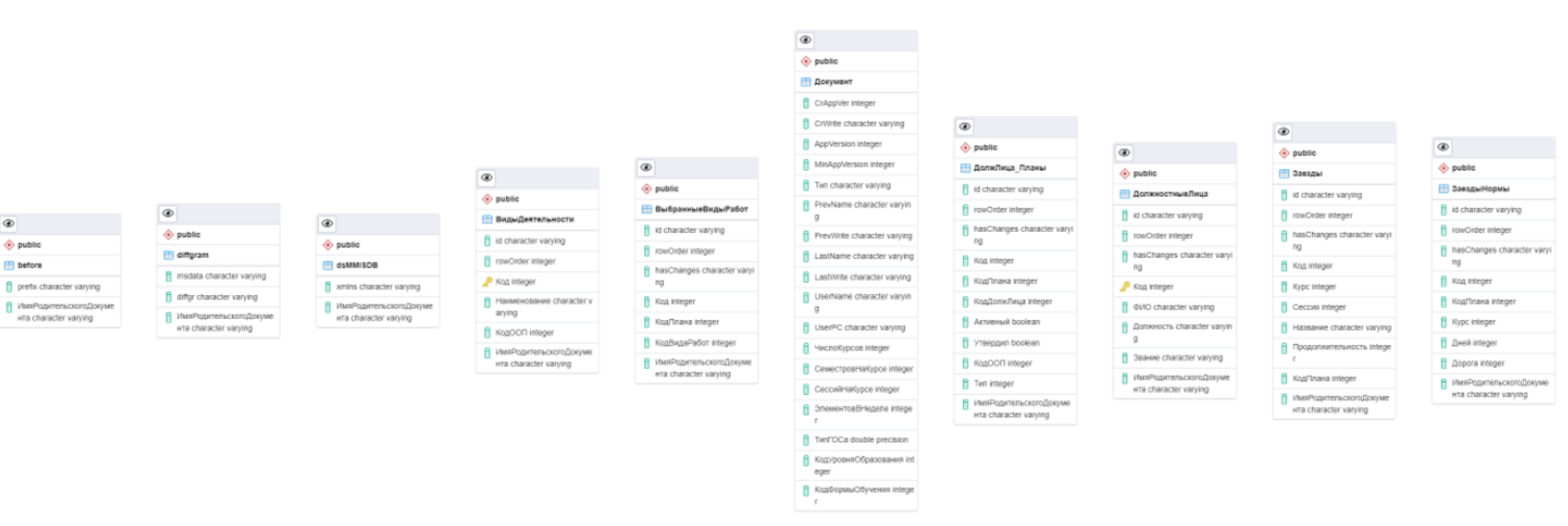
processesed(n\_proc, config, filesDirectory)

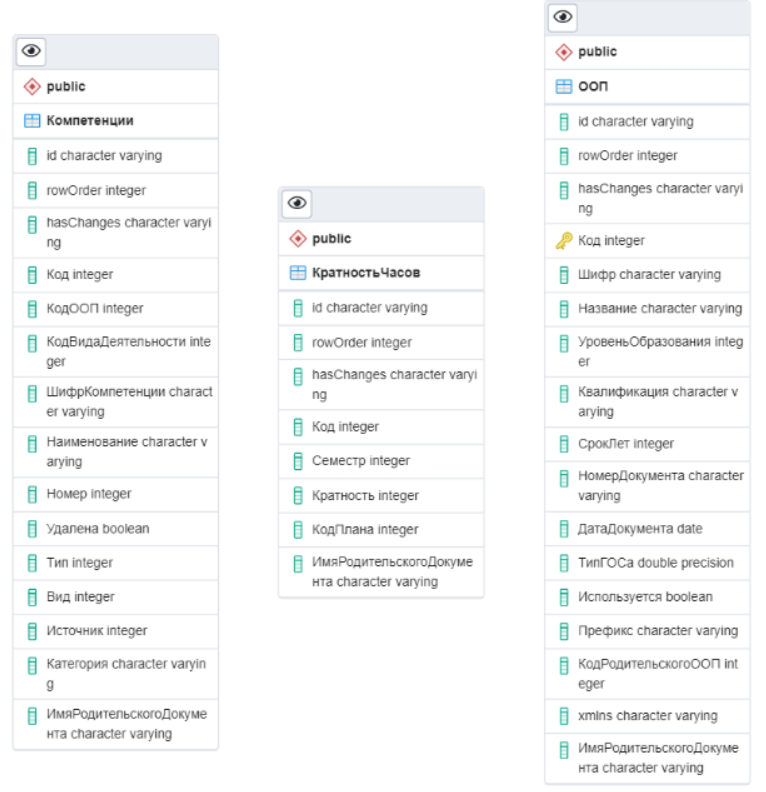
end = time.time()

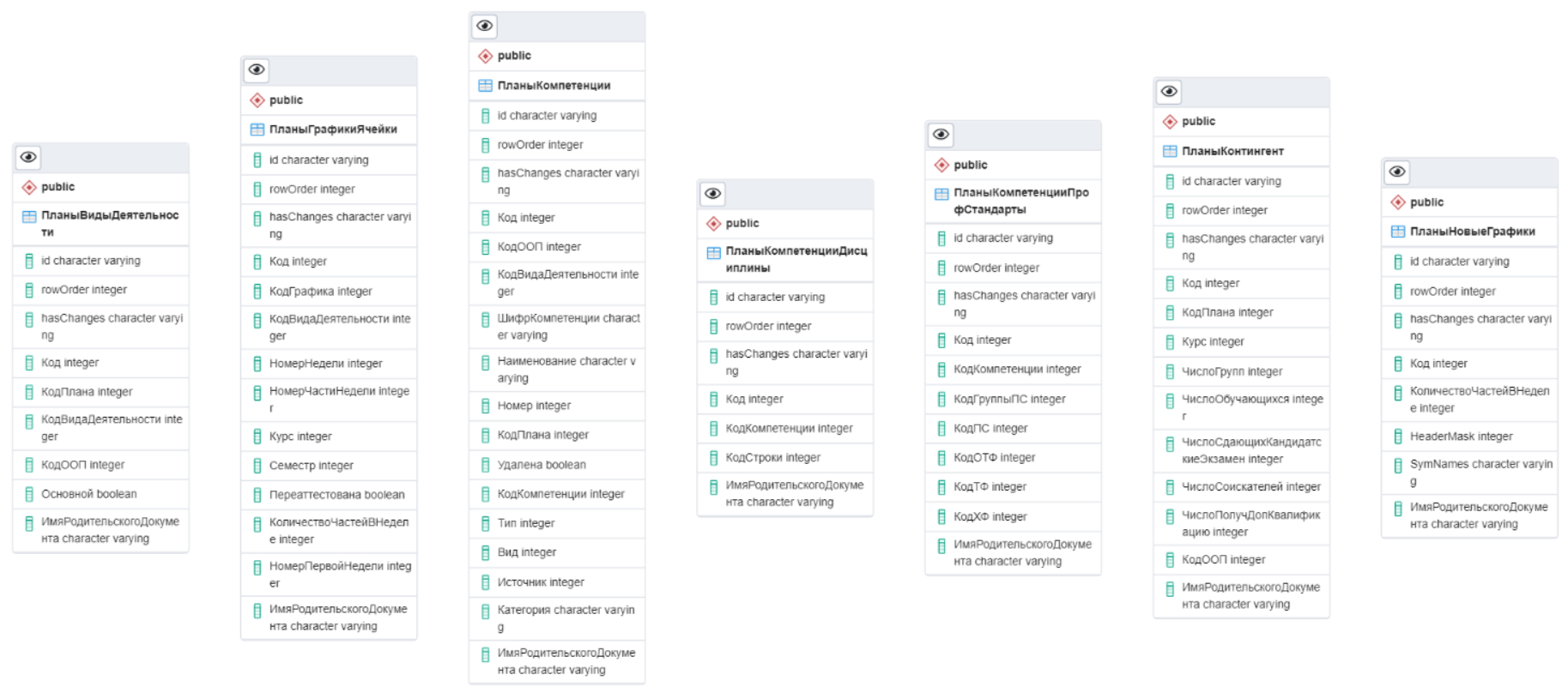
print(f"Всего {n\_proc} ядер в процессоре")

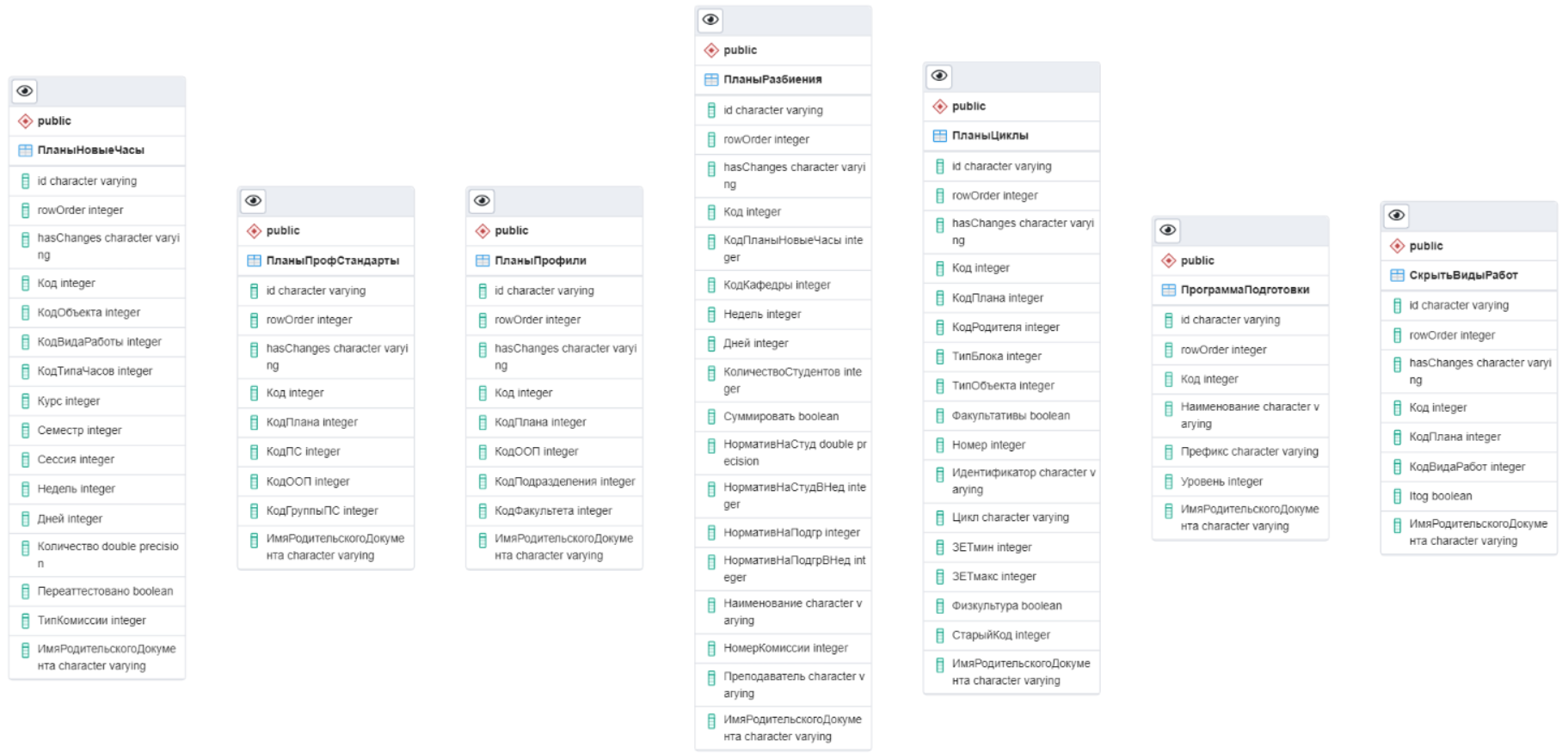
print(f"Итого: ", end - start)

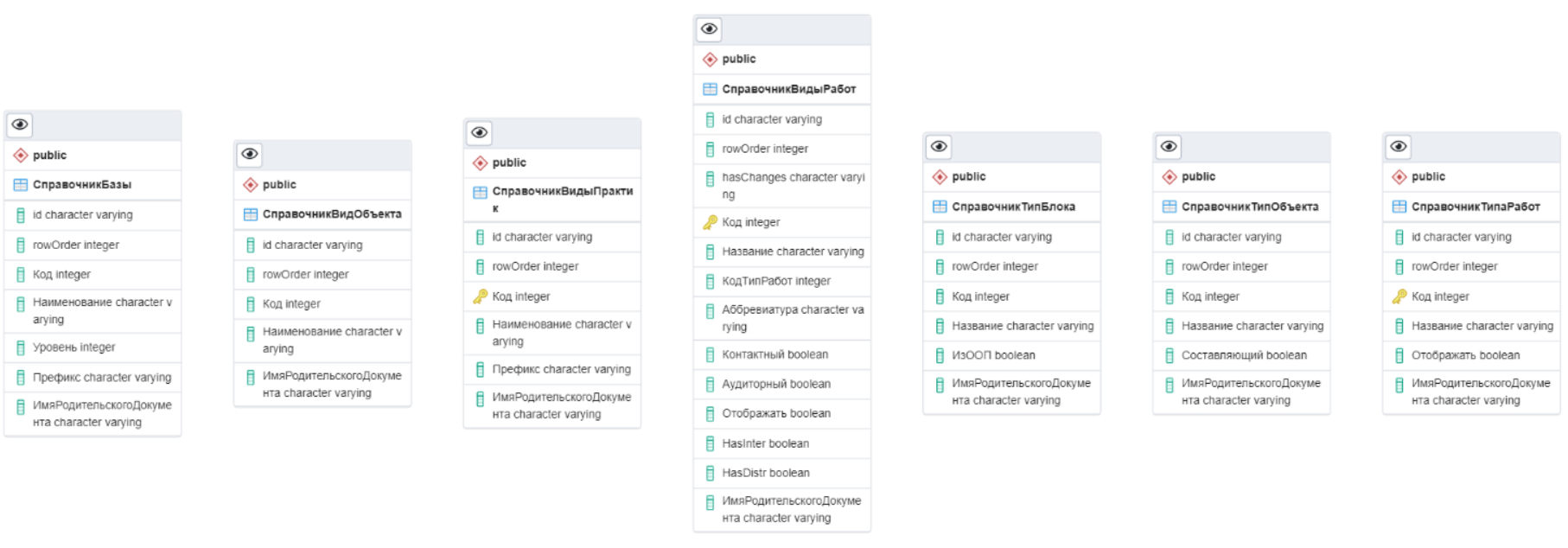
## Приложение Б Plx спецификация учебного плана вуза



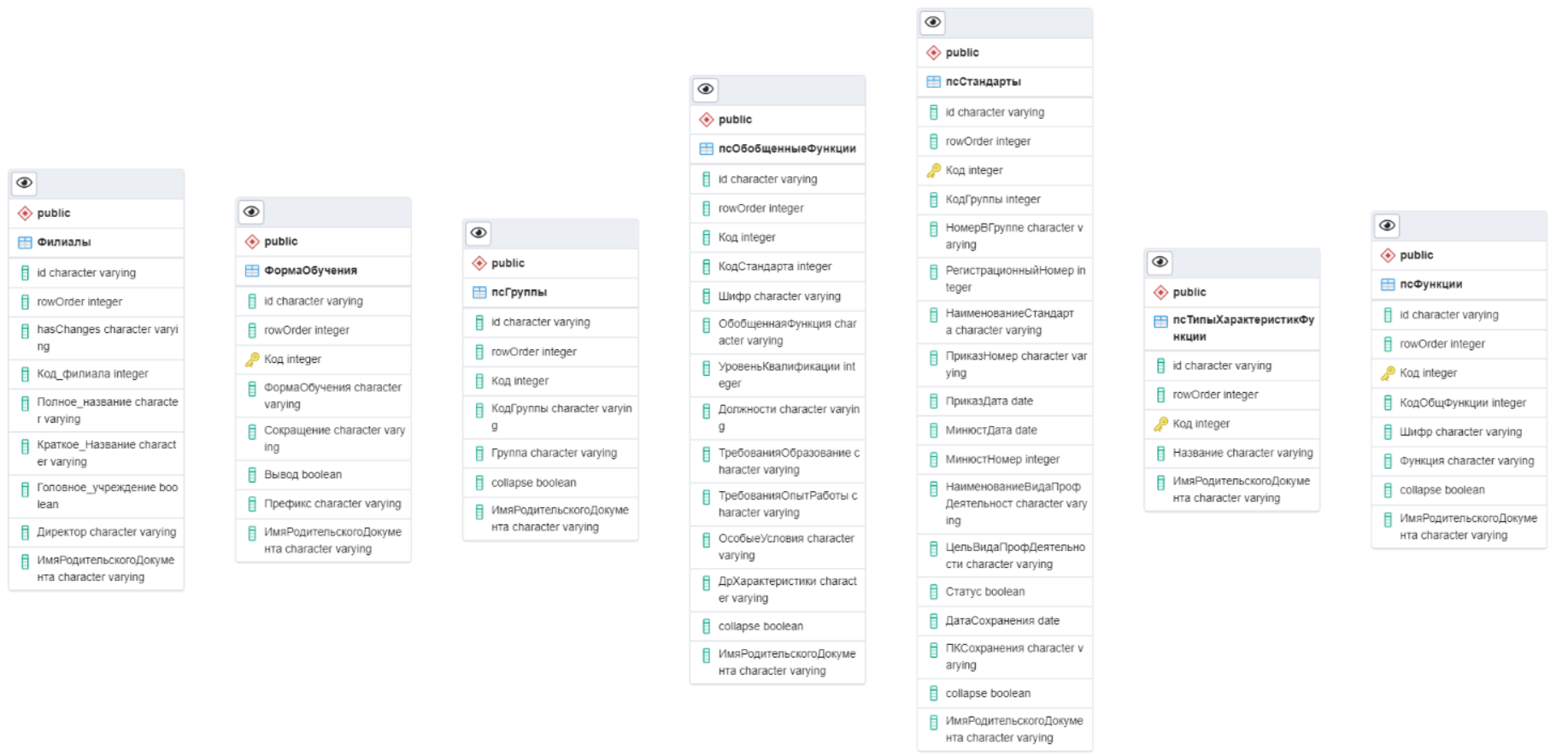












## Приложение В Шаблон индивидуального плана

