**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)»**
**(МГТУ им. Н. Э. Баумана)**

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

# РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

## *К КУРСОВОЙ РАБОТЕ*

## *НА ТЕМУ:*

Разработка базы данных для АРМ разметчика параллельного корпуса технических текстов.

Студент ИУ7-64Б _____ К. А. Рунов
(Группа) (Подпись, дата) (И. О. Фамилия)

Руководитель курсовой работы _____ Ю. В. Строганов
(Подпись, дата) (И. О. Фамилия)

2024 г.

# СОДЕРЖАНИЕ

# ОПРЕДЕЛЕНИЯ

В настоящей расчетно-пояснительной записке применяют следующие термины с соответствующими определениями.

База данных — это некоторый набор перманентных (постоянно хранимых) данных, используемых прикладными программными системами какого-либо предприятия. [1, с. 51]

Домен — это подмножество значений некоторого типа данных, имеющих определенный смысл. [2]

Атрибут отношения — это пара вида <имя_атрибута, имя_домена>. Имена атрибутов должны быть уникальны в пределах отношения. [2]

Схема отношения — это именованное множество упорядоченных пар <имя_атрибута, имя_домена>. Степенью или «арностью» схемы отношения является мощность этого множества. [2]

Кортеж, соответствующий данной схеме отношения — это множество упорядоченных пар <имя_атрибута, значение_атрибута>, которое содержит одно вхождение каждого имени атрибута, принадлежащего схеме отношения. Значение атрибута должно быть допустимым значением домена, на котором определен данный атрибут. Степень кортежа совпадает со степенью соответствующей схемы отношения. [2]

Отношение, определенное на множестве из N доменов содержит две части: заголовок (схему отношения) и тело (множество из M кортежей). Значения N и M называются соответственно степенью и кардинальностью отношения. [2]

Потенциальный ключ — это непустое подмножество множества атрибутов схемы отношения, обладающее свойствами уникальности (в отношении нет двух различных кортежей с одинаковыми значениями потенциального ключа) и неизбыточности (никакое из собственных подмножеств множества потенциального ключа не обладает свойством уникальности). [2]

Первичный ключ — выбранный потенциальный ключ. [2]

Альтернативный ключ — потенциальный ключ, не являющийся первичным. [2]

Внешний ключ в отношении R2 — это непустое подмножество множества атрибутов FK этого отношения, такое, что:

а) существует отношение R1 с потенциальным ключом CK;

б) каждое значение внешнего ключа FK в текущем значении отношения R2 обязательно совпадает со значением ключа CK некоторого кортежа в текущем значении отношения R1. [2]

Реляционная база данных — это набор отношений, имена которых совпадают с именами схем отношений в схеме базы данных. [2]

Система управления базой данных (СУБД) представляет собой программное обеспечение, которое управляет всем доступом к базе данных. [1, с. 87]

# ВВЕДЕНИЕ

В современном мире немаловажное значение имеет корпусная лингвистика. Корпуса текстов находят применение в различных областях — в машинном переводе, в разработке словарей, в лингвистических исследованиях. Для того, чтобы из корпуса текстов можно было извлекать пользу, тексты в нем должны быть размечены. Существуют алгоритмы, позволяющие автоматически производить разметку, но для проверки ее корректности все равно требуется вмешательство человека.

На данный момент не существует открытых параллельных корпусов технических текстов. Также нет открытых информационных систем, позволяющих одновременно

— производить разметку текста в параллельном корпусе,

— производить поиск по параллельному корпусу,

— организовать удобную работу множества разметчиков.

Создание такой информационной системы позволит во многом автоматизировать рабочее место разметчиков параллельного корпуса.

Целью данной работы является разработка базы данных для автоматизации рабочего места разметчиков параллельного корпуса технических текстов.

Задачи курсового проекта:

— провести анализ предметной области параллельных корпусов текстов;

— спроектировать сущности базы данных и ограничения целостности АРМ разметчика корпуса технических текстов;

— выбрать средства реализации базы данных и приложения;

— разработать сущности базы данных и реализовать ограничения целостности базы данных;

— описать интерфейс доступа к базе данных;

— исследовать зависимость времени ответа от количества запросов в секунду.

# 1 Аналитический раздел

В данном разделе будет проведен анализ предметной области корпусов текстов, будет формализована задача и данные, описаны пользователи, которые будут взаимодействовать с проектируемым приложением к базе данных, рассмотрены существующие модели данных и будет выбрана одна из моделей для дальнейшей разработки базы данных на ее основе.

## 1.1 Анализ предметной области

Корпусная лингвистика — это раздел компьютерной лингвистики, который занимается разработкой принципов построения и использования корпусов текстов с помощью компьютерных технологий. Лингвистические корпуса представляют собой структурированные массивы данных, которые используются для изучения языковых единиц в текстах. В рамках корпуса существует поисковая система, которая позволяет находить необходимые языковые единицы и примеры их употребления благодаря технологии текстовой разметки. В свою очередь разметка может быть ручной и автоматической. [3]

### Виды корпусов текстов

В таблице 1 приведена классификация корпусов текстов по разным признакам.

| **Признак** | **Типы корпусов** |
|---|---|
| Цель | Многоцелевые, специализированные |
| Параллельность | Параллельные, сопоставимые |
| Динамичность | Динамические (мониторные), статические |
| Разметка | Размеченные, неразмеченные |
| Характер разметки | Морфологические, синтаксические, семантические, анафорические, просодические и т. д. |
| Объем текстов | Полнотекстовые, «фрагментнотекстовые» |

Таблица 1 – Классификация корпусов [4, с. 57]

Корпус технических текстов, для которого будет разрабатываться база

данных в настоящей работе, является специализированным, параллельным, многоязыковым, динамическим (будет постоянно пополняться).

## Параллельные корпусы

Параллельные корпусы — корпусы, представляющие собой множество текстов-оригиналов, написанных на каком-либо исходном языке, и текстов — переводов этих исходных текстов на один или несколько других языков [4, с. 61].

При подготовке параллельных корпусов и разработке программ для их обработки, требуется выровнять тексты — установить соответствие между фрагментами текста оригинала и текста перевода. Для решения этой задачи существуют различные методы автоматического выравнивания текстов по предложениям, грамматическим конструкциям, терминам, словам и словосочетаниям. [4, с. 61]

Ниже приведен пример выравнивания текстов на уровне предложений.

| | | |
|---|---|---|
| 1 | THE PLAY — for which Briony had designed the posters, programs and tickets, constructed the sales booth out of a folding screen tipped on its side, and lined the collection box in red crepe paper — was written by her in a two-day tempest of composition, causing her to miss a breakfast and a lunch. | Пьеса, для которой Брайони рисовала афиши, делала программки и билеты, сооружала из ширмы кассовую будку и обклеивала коробку для денежных сборов гофрированной красной бумагой, была написана ею за два дня в порыве вдохновения, заставлявшего ее забывать даже о еде. |
| 2 | When the preparations were complete, she had nothing to do but contemplate her finished draft and wait for the appearance of her cousins from the distant north. | Когда приготовления закончились, ей не оставалось ничего, кроме как созерцать свое творение и ждать появления кузенов и кузины, которые должны были прибыть с далекого севера. |

Таблица 2 – Пример выравнивания текстов на уровне предложений [4, с. 62]

## Проблемы определения границ терминов

При попытке проведения автоматического выравнивания на уровне терминов, возникает ряд проблем. Среди них выделяют [3]:

— неправильное определение границ терминов-словосочетаний, состоящих из двух и более слов и составных терминов;

— распознавание составных терминов и терминов-словосочетаний, состоящих из двух и более слов; в частности, распознавание лексической единицы как части составного термина или как свободной лексической единицы;

— определение лексической единицы как термина в зависимости от контекста и тематики текста, в котором данная лексическая единица употребляется;

— объемные списки терминов-кандидатов, которые необходимо проверять вручную, поскольку частота не является достаточным критерием для оценки того, является ли выделенное слово термином или нет.

Точность определения границ термина при автоматической разметки является одной из основных лингвистических задач, а отсутствие на сегодняшний день веб-платформ для автоматической разметки русскоязычных текстов делает актуальным разработку таковой.

## Типы текстовых разметок

Существует множество типов текстовых разметок. Большинство современных корпусов относятся к корпусам морфологического или синтаксического вида [4, с. 56].

Далее подробнее будут рассмотрены структурная и семантическая разметки, поскольку они являются основными типами разметки в параллельном корпусе технических текстов.

## Структурная разметка

Структурная разметка предназначена для выделения структурных элементов текста (том, книга, часть, глава, действие, сноска, ремарка, стих, а также: абзац, предложение, словоформа и текстоформа — таблица, формула и др.) [5]. В контексте учебно-научных текстов структурная разметка использована для выделения названия статьи, авторов, оглавления, предисловия, введения и т.д.

На рисунке 1 представлена структурная схема элементов учебно-научного текста.

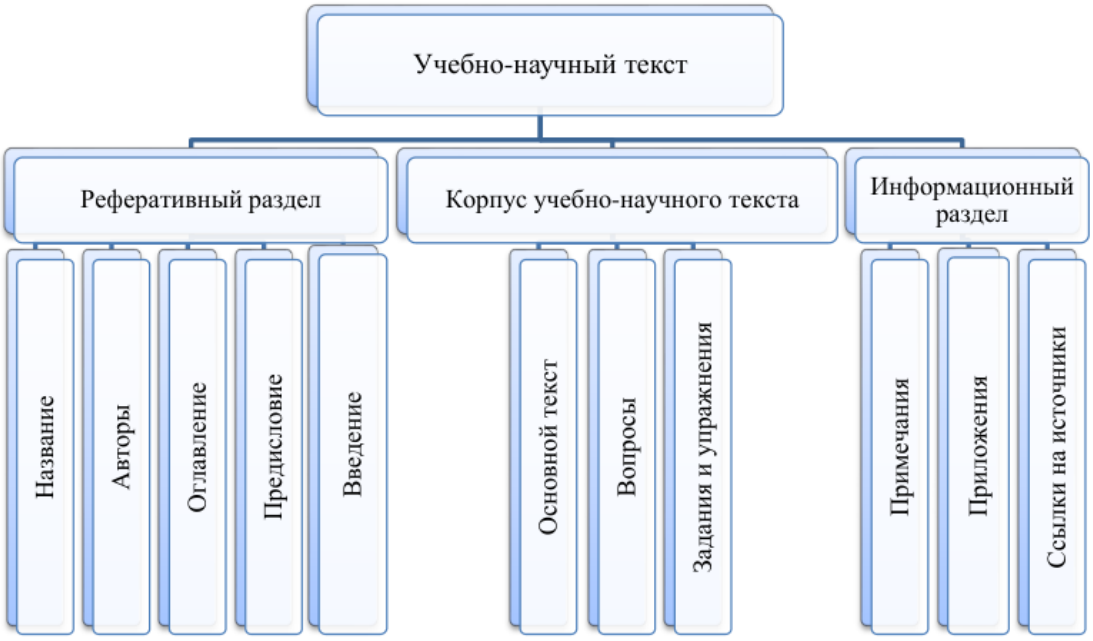


Рисунок 1 – Структурные элементы учебно-научных текстов [6]

## Семантическая разметка

Семантическая разметка помогает установить контекст высказывания и устранить двусмысленность. Основная цель семантической разметки — «формализировать» значения слов и сделать тексты пригодными для машинной обработки. [7]

Существует множество видов семантических разметок. Один из вариантов русской семантической разметки представлен на сайте Центра компьютерных и корпусных языковых исследований Ланкастерского университета [8, 4, с. 51]:

| | |
|---|---|
| A | Общие понятия |
| A1 | Общие понятия |
| A1.1.1 | Обычные действия / Изголовление ч.-л. |
| A1.1.1 | Повреждение и разрушение |
| A1.2 | Пригодность |
| A1.3 | Осторожность |
| ... | ..... |
| B | Тело человека |
| B1 | Анатомия и физиология |
| B2 | Здоровье и болезнь |
| ... | ..... |
| C | Искусства и ремесла |
| ... | ..... |

Таблица 3 – Русская семантическая разметка [8]

В научно-технических текстах для семантической разметки могут использоваться семантические падежи Ч. Филлмора [7].

## 1.2 Существующие решения

В современном мире существует множество параллельных корпусов (Opus, Linguee, MyMemory, Glosbe, Reverso, TAUS Data Cloud и др.) [9], сервисов для автоматического выравнивания текстов (Hunalign, Euclid, Abbyy Aligner, Trados, Winalign, Wordfast tools, Giza++ и др.) [4, с. 62], служб, позволяющих создавать собственные корпусы и производить в них поиск (SketchEngine [10], NoSketchEngine [11]); инструментов для автоматического извлечения терминов (TerMine, TermExtraction, Terminology Extraction) [3] и прочих инструментов для работы с корпусами текстов (OpenCorpora [12]).

Но на данный момент не существует открытых параллельных корпусов технических текстов [13]. Также нет открытых информационных систем, позволяющих одновременно производить разметку текста в параллельном корпусе, производить поиск по параллельному корпусу и организовать удобную работу множества разметчиков.

## 1.3 Формализация задачи

В ходе выполнения курсовой работы необходимо спроектировать и разработать базу данных для хранения документов — технических текстов на разных языках, их разметок, и информации о пользователях базы данных.

Для взаимодействия с базой данных необходимо разработать интерфейс, предоставляющий возможности

— добавления новых документов в базу данных,

— добавления новых разметок в базу данных,

— произведения поиска по текстам и разметкам, хранящимся в базе данных.

## 1.4 Формализация данных

Разрабатываемая база данных должна хранить информацию о следующих сущностях:

— пользователь;

— документ;

— метаданные о документе — автор;

— задание на разметку;

— структурная разметка;

— терминологическая разметка.

На рисунке 2 представлена диаграмма сущностей разрабатываемой базы данных в нотации Чена.

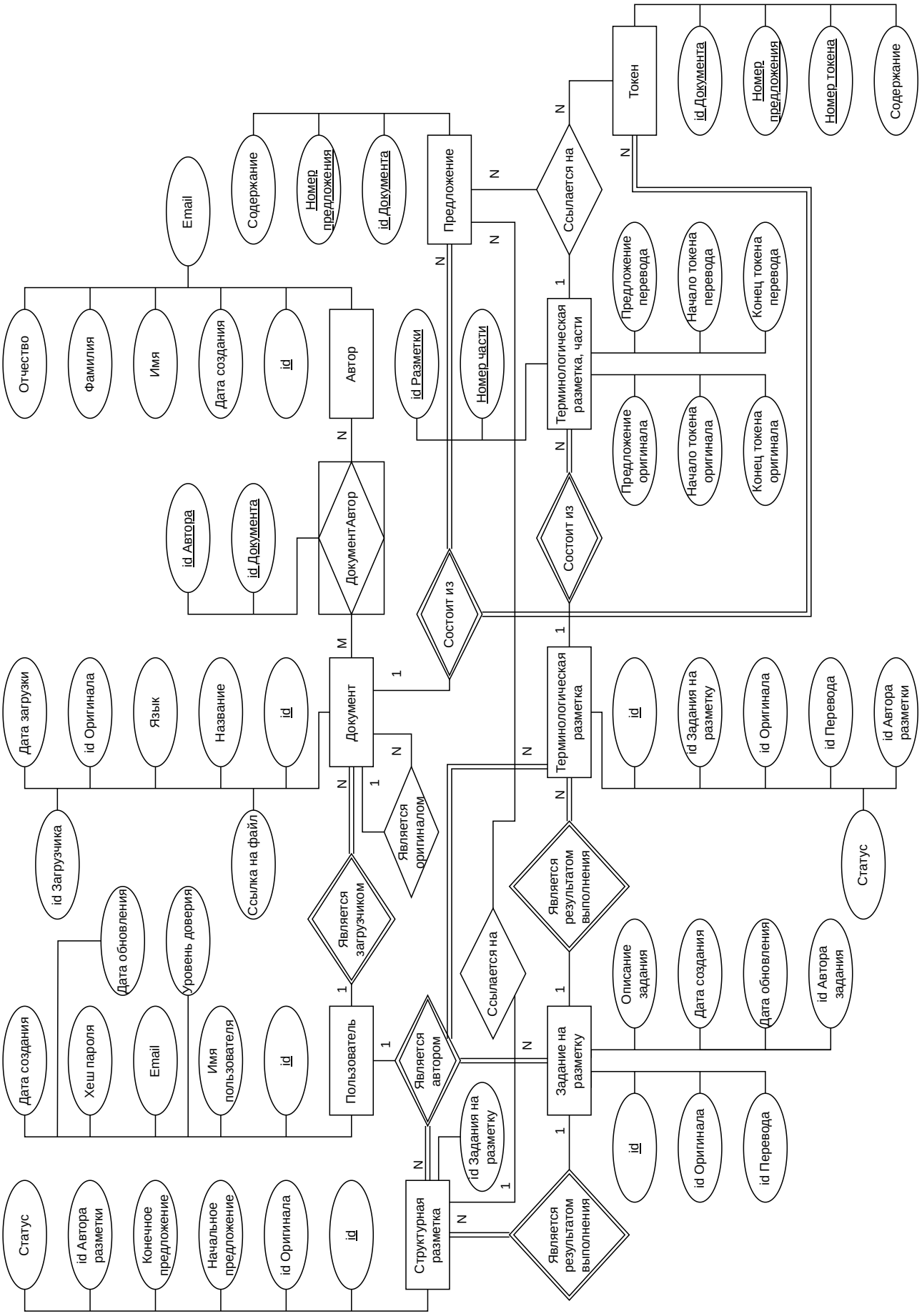


Рисунок 2 – ER-диаграмма в нотации Чена

## 1.5 Формализация и описание пользователей

Взаимодействовать с проектируемым приложением к базе данных будут три вида пользователей:

— администраторы;

— модераторы;

— пользователи.

Администратор имеет полный доступ к данным: может добавлять, удалять тексты; добавлять, удалять, изменять разметки.

Модераторы производят проверку разметки.

Пользователи имеют возможность осуществлять разметку, которая после своего создания должна пройти модерацию перед публикацией. Также пользователи могут искать нужные разметку и текст среди уже существующих.

На рисунке 3 приведена диаграмма вариантов использования проектируемого приложения к базе данных.

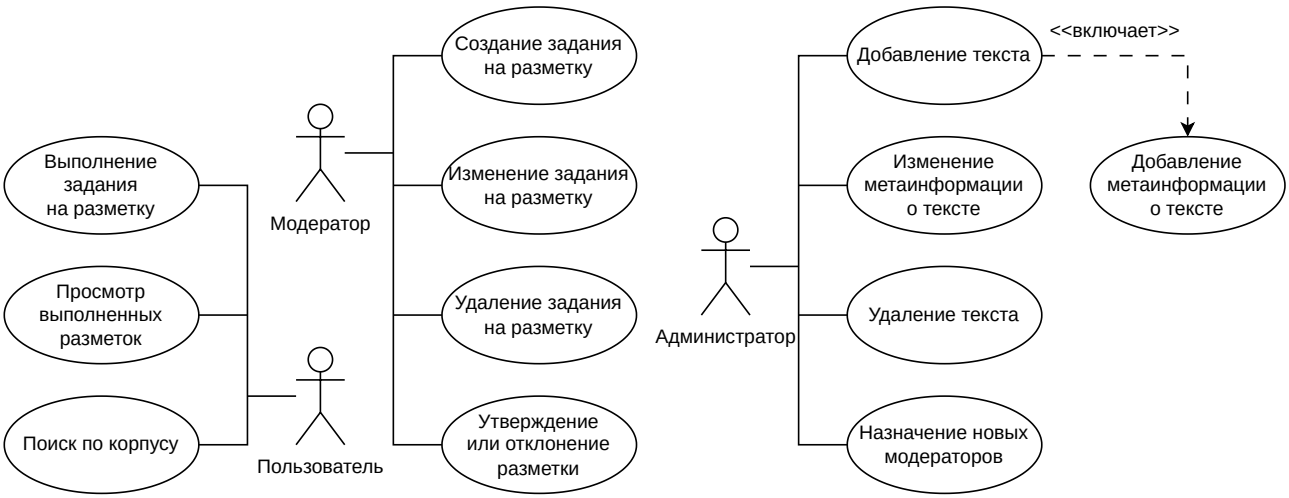


Рисунок 3 – Диаграмма вариантов использования

## 1.6 Модели данных

Модель данных — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Упомянутые объекты позволяют моделировать структуру данных, а операторы — поведение данных. [1]

Дейт выделяет три составляющих моделей данных: структурную, манипуляционную и целостную. [14]

### 1.6.1 Реляционная

Структурная часть реляционной модели описывает, из каких объектов состоит реляционная модель. Постулируется, что основной структурой данных, используемой в реляционной модели, являются нормализованные «n-арные» отношения. [1, 2]

В реляционной модели каждый кортеж любого отношения должен отличаться от другого кортежа этого отношения. [1, 2]

Манипуляционная часть реляционной модели описывает два эквивалентных способа манипулирования реляционными данными — реляционную алгебру и реляционное исчисление. [2]

В целостной части реляционной модели фиксируются два базовых требования целостности, которые должны выполняться для любых отношений в любых реляционных базах данных. Это целостность сущностей и целостность ссылок. [2]

Поддержание целостности сущностей обеспечивается средствами СУБД и осуществляется с помощью двух ограничений:

1) при добавлении записей в таблицу проверяется уникальность их первичных ключей,

2) не позволяется изменение значений атрибутов, входящих в первичный ключ. [2]

Требование целостности ссылок состоит в следующем: Для каждого значения внешнего ключа, появляющегося в дочернем отношении, в роди-

тельском отношении должен найтись кортеж с таким же значением первичного ключа. [2]

### 1.6.2 Инвертированные списки

Модель инвертированных списков похожа на реляционную, но на более низком уровне абстракции — хранимые таблицы, а также некоторые пути доступа к этим хранимым таблицам (в частности, некоторые индексы) напрямую доступны пользователю. Как и реляционная база данных, база данных инвертированных списков содержит множество «таблиц» или файлов, и эти «таблицы» или файлы разделены на строки (записи) и колонки (поля), как и в реляционном случае. [14, с. 388]

Однако, существуют и некоторые значительные различия:

1) Строки таблицы инвертированного списка упорядочены в физической последовательности, независимой от индексов. Поля внутри строк также упорядочены слева направо.

2) Определяется общий порядок для базы данных, где строки одной таблицы могут предшествовать строкам другой или чередоваться в определенном порядке.

3) Можно задать любое количество ключей поиска для таблицы, что позволяет как прямой, так и последовательный доступ на основе этих ключей. Индексы не являются прозрачными для пользователя, но их поддержка осуществляется СУБД.

В общем случае операторы манипуляции данными в модели инвертированных списков делятся на две широкие категории:

1) Операторы, которые устанавливают адресуемость к какой-либо записи в базе данных (операторы поиска, нахождения или определения местоположения).

2) Операторы, которые работают с записью по ранее установленному адресу.

Операторы поиска, в свою очередь, делятся на две подкатегории:

а) Операторы, которые находят запись «из ниоткуда» — то есть операторы прямого поиска.

б) Операторы, которые находят запись в терминах ее позиции относительно какого-либо ранее установленного адреса — то есть операторы относительного поиска. [14, с. 389]

Модель инвертированных списков не поддерживает общие правила целостности. Большинство ограничений должны обеспечиваться пользователем, включая ссылочную целостность. [14, с. 391]

### 1.6.3 Иерархическая

Иерархическая база данных состоит из упорядоченной коллекции экземпляров одного типа дерева.

Тип дерева включает один корневой тип записи и упорядоченную коллекцию зависимых (нижнего уровня) типов поддеревьев. Тип поддерева, в свою очередь, также состоит из одного типа записи — корневого типа поддерева — и упорядоченной коллекции зависимых типов поддеревьев нижнего уровня. Таким образом, весь тип дерева представляет собой иерархическую структуру типов записей. [14, с. 406]

Основное различие между иерархической и реляционной структурой заключается в следующем: в иерархической базе данных информация, которая в реляционной базе данных представлена внешними ключами, представлена связями «родитель-ребенок». [14, с. 408]

Язык манипулирования данными в иерархической базе данных включает операторы для работы с данными в виде деревьев:

— Нахождение конкретного дерева в базе данных.

— Переход от одного дерева к следующему в иерархической последовательности.

— Перемещение от записи к записи внутри дерева по иерархическим путям.

— Вставка новой записи в дерево.

— Удаление указанной записи.

Эти операторы, как правило, работают на уровне записей (манипулируют отдельными записями (или строками) в базе данных, а не группами записей или целыми наборами данных). [14, с. 410]

Иерархическая модель автоматически поддерживает ссылочную целостность по правилу: ни один потомок не может существовать без своего родителя. Если удаляется родитель, система удаляет все поддерево. Потомок не может быть вставлен без существующего родителя. Это эквивалентно соблюдению правил внешнего ключа в реляционной модели. [14, с. 411]

### 1.6.4 Сетевая

Сетевая база данных состоит из набора записей и связей — точнее, из любого числа экземпляров нескольких типов записей и связей. Каждый тип связи включает два типа записей: родительский и дочерний. Каждый экземпляр связи состоит из одного экземпляра родительского типа записи и упорядоченного набора экземпляров дочернего типа записи. Для конкретного типа связи L с родительским типом записи P и дочерним типом записи C справедливо следующее:

— Каждый экземпляр P является родителем в одном экземпляре L.

— Каждый экземпляр C является дочерней записью в не более чем одном экземпляре L.

Таким образом, сетевую структуру данных можно рассматривать как расширенную форму иерархической структуры данных. Основное различие между ними заключается в следующем: в иерархической структуре у дочерней записи есть ровно один родитель; в сетевой структуре у дочерней записи может быть любое количество родителей. [14, с. 447]

Язык манипулирования данными в сетевой базе данных состоит из набора операторов для работы с данными, представленными в виде записей и связей. Примеры таких операторов включают следующее:

— Оператор для нахождения конкретной записи по значению какого-либо поля в этой записи;

— Оператор для перехода от родителя к его первому ребенку в какой-либо связи;

— Оператор для перехода от одного ребенка к следующему в какой-либо связи;

— Оператор для перехода от ребенка к родителю в какой-либо связи;

— Оператор для создания новой записи;

— Оператор для удаления существующей записи;

— И так далее.

Эти операторы обычно так же, как и в инвертированных списках, и в иерархической модели, работают на уровне записей.

Сетевая модель, как и иерархическая, включает встроенную поддержку ссылочной целостности через связи. Например, можно задать правило, что дочерняя запись не может быть вставлена без существующего родителя. [14, с. 452]

### 1.6.5 Постреляционная

Постреляционная модель расширяет реляционную модель. Она устраняет ограничение на атомарность данных, позволяя использовать поля с многозначными значениями. Значения этих полей могут содержать подзначения, и такой набор значений рассматривается как отдельная таблица, встроенная в основную таблицу. [2]

### 1.6.6 Выбор модели данных

Как видно по рисунку 2, соблюдения строгой иерархии на диаграмме сущностей разрабатываемой базы данных не наблюдается. В случае сетевой модели возникают проблемы с выбором «главного» элемента, с которого будет осуществляться обход, и с обработкой нескольких путей, ведущих от главного элемента к искомому в случае наличия связи «многие-ко-многим». Описанная ранее формализация данных лучше всего укладывается в рамки реляционной модели, поэтому в основе разрабатываемой базы данных будет

использоваться именно она. Помимо прочего, у реляционной модели есть еще ряд преимуществ — современные ее реализации могут автоматически обеспечивать целостность данных и оптимизировать запросы, а также поддерживают транзакционность.

## 1.7 Вывод

В данном разделе был проведен анализ предметной области корпусов текстов, была формализована задача и данные, описаны пользователи, которые будут взаимодействовать с проектируемым приложением к базе данных, рассмотрены существующие модели данных, и была выбрана реляционная модель.

# 2 Конструкторский раздел

## 2.1 Проектирование базы данных

На рисунке 4 представлена схема проектируемой базы данных.

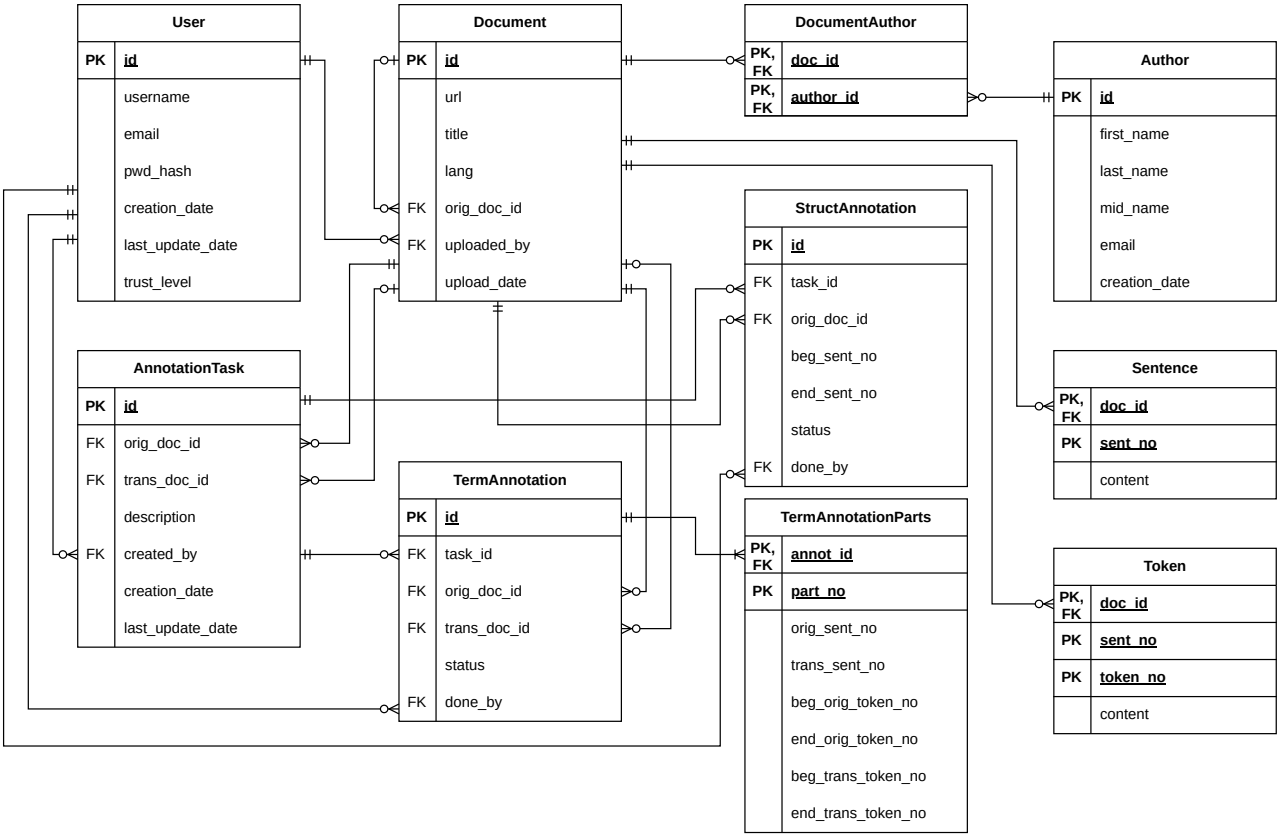


Рисунок 4 – ER-диаграмма проектируемой базы данных в нотации Мартина

## 2.2 Описание сущностей

В данном подразделе будут описаны десять сущностей проектируемой базы данных: User, Document, DocumentAuthor, Author, AnnotationTask, StructAnnotation, TermAnnotation, TermAnnotationParts, Sentence, Token.

### User

Сущность User формализует пользователя базы данных и имеет следующие поля:

— id — идентификатор пользователя;

— username — имя пользователя;

— email — электронная почта пользователя;

— pwd_hash — хеш пароля пользователя;

— trust_level — уровень доверия пользователя (зависит от качества и количества совершенных разметок);

— creation_date — дата регистрации пользователя;

— last_update_date — дата последнего обновления информации о пользователе.

### Document

Сущность Document формализует документ, хранимый в корпусе, и имеет следующие поля:

— id — идентификатор документа;

— url — ссылка на документ (в сети или файловой системе);

— title — название документа / статьи;

— lang — язык документа;

— orig_doc_id — ссылка на оригинальный документ, в случае, если данный документ является переводом;

— uploaded_by — ссылка на пользователя, загрузившего документ в систему;

— upload_date — дата загрузки документа в систему.

### DocumentAuthor

Сущность DocumentAuthor формализует связь «многие-ко-многим» документов и авторов:

— doc_id — идентификатор документа;

— author_id — идентификатор автора документа.

### Author

Сущность Author формализует автора документов:

— id — идентификатор автора;

— first_name — имя автора;

— last_name — фамилия автора;

— mid_name — отчество автора;

— email — электронная почта автора;

— creation_date — дата появления сущности в системе.

### AnnotationTask

Сущность AnnotationTask формализует задание на разметку:

— id — идентификатор задания на разметку;

— orig_doc_id — ссылка на оригинальный документ, участвующий в разметке;

— trans_doc_id — ссылка на перевод документа, участвующий в разметке;

— description — описание задания на разметку;

— created_by — ссылка на пользователя, создавшего задание на разметку;

— creation_date — дата создания задания на разметку;

— last_update_date — дата последнего обновления задания на разметку.

### StructAnnotation

Сущность StructAnnotation формализует структурную разметку:

— id — идентификатор структурной разметки;

— task_id — ссылка на задание на разметку, результатом выполнения которого данная разметка является;

— orig_doc_id — ссылка на документ, для которого выполняется структурная разметка;

— beg_sent_no — номер начального предложения разметки;

— end_sent_no — номер последнего предложения разметки;

— status — статус разметки (утверждена модератором или нет);

— done_by — ссылка на пользователя, выполнившего разметку.

### TermAnnotation

Сущность TermAnnotation формализует терминологическую разметку:

— id — идентификатор терминологической разметки;

— task_id — ссылка на задание на разметку, результатом выполнения которого данная разметка является;

— orig_doc_id — ссылка на оригинальный документ, для которого выполняется терминологическая разметка;

— trans_doc_id — ссылка на перевод документа, для которого выполняется терминологическая разметка;

— status — статус разметки (утверждена модератором или нет);

— done_by — ссылка на пользователя, выполнившего разметку.

### TermAnnotationParts

Сущность TermAnnotationParts формализует составляющие терминологической разметки — определяет границы терминов, входящих в разметку, и устанавливает их соответствие в оригинальном документе и его переводе:

— annot_id — ссылка на идентификатор терминологической разметки;

— part_no — номер части, термина, входящего в терминологическую разметку;

— orig_sent_no — номер предложения в оригинальном документе;

— trans_sent_no — номер предложения в документе-переводе;

— beg_orig_token_no — номер токена, являющегося началом термина в оригинальном документе;

— end_orig_token_no — номер токена, являющегося концом термина в оригинальном документе;

— beg_trans_token_no — номер токена, являющегося началом термина в документе-переводе;

— end_trans_token_no — номер токена, являющегося концом термина в документе-переводе.

### Sentence

Сущность Sentence формализует предложение текста документа:

— doc_id — ссылка на документ, к которому относится предложение;

— sent_no — номер предложения в документе;

— content — текст предложения.

### Token

Сущность Token формализует токен, получаемый в процессе токенизации текста документа:

— doc_id — ссылка на документ, в котором содержится предложение, которому принадлежит токен;

— sent_no — номер предложения в документе, которому принадлежит токен;

— token_no — номер токена в предложении;

— content — текст токена.

## 2.3 Описание ограничений целостности

В данном подразделе в таблицах 4 – 13 будут описаны проектируемые ограничения целостности базы данных в контексте выбранной, реляционной, модели.

Таблица 4 – Ограничение целостности User

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| id | UUID | первичный ключ |
| username | строка | уникальный, не пустое значение |
| email | строка | не пустое значение |
| pwd_hash | строка | не пустое значение |
| trust_level | вещественное число | не пустое значение, по умолчанию 0 |
| creation_date | временная метка | не пустое значение |
| last_update_date | временная метка | не пустое значение |

Таблица 5 – Ограничение целостности Document

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| id | UUID | первичный ключ |
| url | строка | не пустое значение |
| title | строка | не пустое значение |
| lang | строка | не пустое значение |
| orig_doc_id | UUID | внешний ключ на поле id таблицы Document |
| uploaded_by | UUID | внешний ключ на поле id таблицы User, не пустое значение |
| upload_date | временная метка | не пустое значение |

Таблица 6 – Ограничение целостности DocumentAuthor

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| doc_id | UUID | первичный ключ, внешний ключ на поле id таблицы Document |
| author_id | UUID | первичный ключ, внешний ключ на поле id таблицы Author |

Таблица 7 – Ограничение целостности Author

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| id | UUID | первичный ключ |
| first_name | строка | — |
| last_name | строка | — |
| mid_name | строка | — |
| email | строка | — |
| creation_date | временная метка | не пустое значение |

Таблица 8 – Ограничение целостности AnnotationTask

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| id | UUID | первичный ключ |
| orig_doc_id | UUID | внешний ключ на поле id таблицы Document, не пустое значение |
| trans_doc_id | UUID | внешний ключ на поле id таблицы Document |
| description | строка | не пустое значение |
| created_by | UUID | внешний ключ на поле id таблицы User |
| creation_date | временная метка | не пустое значение |
| last_update_date | временная метка | не пустое значение |

Таблица 9 – Ограничение целостности StructAnnotation

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| id | UUID | первичный ключ |
| task_id | UUID | внешний ключ на поле id таблицы AnnotationTask, не пустое значение |
| orig_doc_id | UUID | внешний ключ на поле id таблицы Document, не пустое значение |
| beg_sent_no | целое число | не пустое значение |
| end_sent_no | целое число | не пустое значение |
| status | строка | — |
| done_by | UUID | внешний ключ на поле id таблицы User, не пустое значение |

Таблица 10 – Ограничение целостности TermAnnotation

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| id | UUID | первичный ключ |
| task_id | UUID | внешний ключ на поле id таблицы AnnotationTask, не пустое значение |
| orig_doc_id | UUID | внешний ключ на поле id таблицы Document, не пустое значение |
| trans_doc_id | UUID | внешний ключ на поле id таблицы Document |
| status | строка | — |
| done_by | UUID | внешний ключ на поле id таблицы User, не пустое значение |

Таблица 11 – Ограничение целостности TermAnnotationParts

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| annot_id | UUID | первичный ключ, внешний ключ на поле id таблицы TermAnnotation |
| part_no | целое число | первичный ключ |
| orig_sent_no | целое число | не пустое значение |
| trans_sent_no | целое число | не пустое значение |
| beg_orig_token_no | целое число | не пустое значение |
| end_orig_token_no | целое число | не пустое значение |
| beg_trans_token_no | целое число | не пустое значение |
| end_trans_token_no | целое число | не пустое значение |

Таблица 12 – Ограничение целостности Sentence

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| doc_id | UUID | первичный ключ, внешний ключ на поле id таблицы Document |
| sent_no | целое число | первичный ключ |
| content | строка | не пустое значение |

Таблица 13 – Ограничение целостности Token

| **Поле** | **Тип** | **Ограничение** |
|---|---|---|
| doc_id | UUID | первичный ключ, внешний ключ на поле id таблицы Document |
| sent_no | целое число | первичный ключ |
| token_no | целое число | первичный ключ |
| content | строка | не пустое значение |

## 2.4 Описание функций, процедур и триггеров

Чем больше пользователь производит корректных разметок, тем сильнее модераторы могут его разметкам «доверять». Рейтинг доверия пользователей можно вычислять разными способами. Одним из примитивных способов вычисления рейтинга доверия пользователя может быть следующий: увеличивать рейтинг доверия пользователя на пять за каждую утвержденную разметку, и уменьшать на четыре — за каждую отклоненную. Пересчет рейтинга доверия пользователей можно производить автоматически, после проверки модератором его разметки, с помощью триггера. Алгоритм работы триггера, срабатывающего в ответ на обновление статуса разметки, приведен на рисунке 5 ниже.

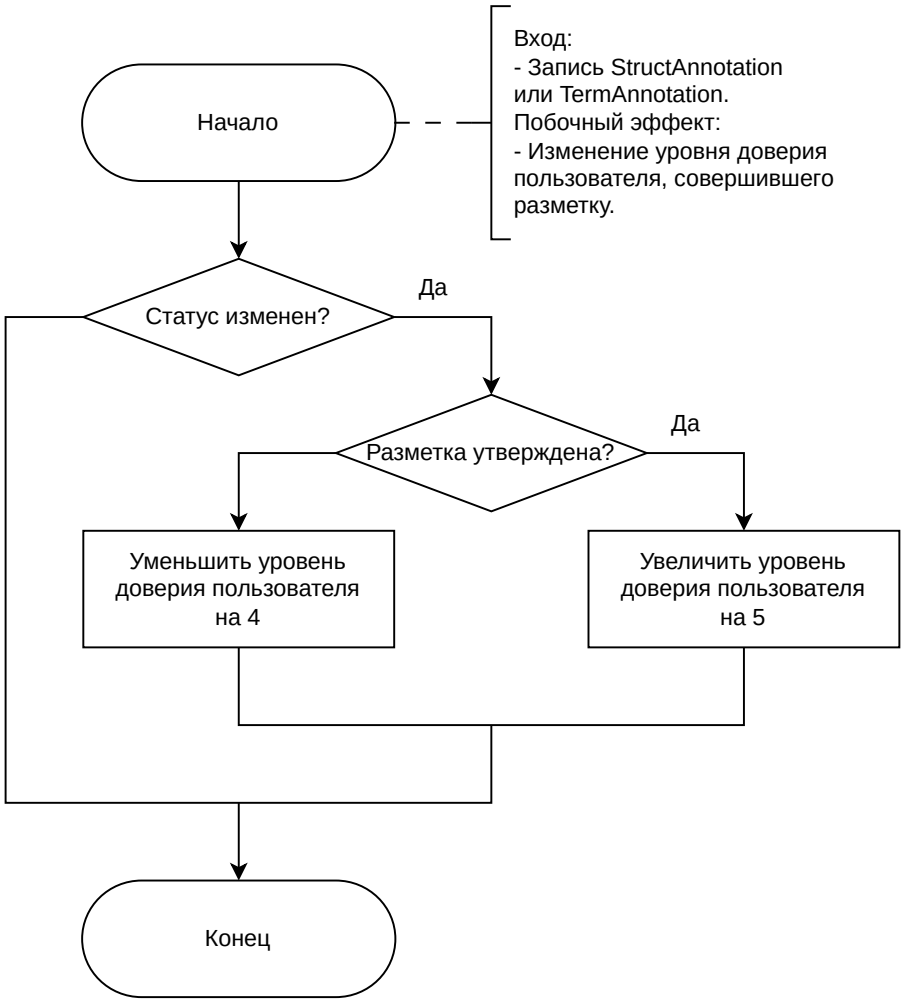


Рисунок 5 – Схема алгоритма работы триггера, отвечающего за пересчет уровня доверия пользователей

## 2.5 Описание ролевой модели

## 2.6 Вывод

# 3 Технологический раздел

## 3.1 Выбор средств реализации

## 3.2 Описание реализаций

### 3.2.1 Сущности базы данных

### 3.2.2 Ограничения целостности базы данных

### 3.2.3 Ролевая модель на уровне базы данных

### 3.2.4 Функции, процедуры и триггеры

### 3.2.5 Тестирование

### 3.2.6 Интерфейс доступа к базе данных

## 3.3 Вывод

# 4 Исследовательский раздел

## 4.1 Технические характеристики

## 4.2 Описание исследования

## 4.3 Проведение исследования

## 4.4 Вывод

# ЗАКЛЮЧЕНИЕ

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ


1. Дейт К. Дж. Введение в системы баз данных, 8-е издание.: Пер. с англ. — М.: Издательский дом «Вильямс», 2005. — 1328 с.: ил. — Парал. тит. англ.

2. Гаврилова Ю.М. Конспект лекций по курсу «Базы Данных». 2023.

3. Бутенко Ю.И., Сулейманова Э.В. Терминологическая разметка научно-технических текстов в специальном корпусе // Проблемы лингвистики и лингводидактики в неязыковом вузе: 5-я Международная научно-практическая конференция. 2022. Т. 1. С. 329–337.

4. Захаров В.П., Богданова С.Ю. Корпусная лингвистика: учебник. 3-е изд., перераб. — СПб.: Изд-во С.-Петерб. ун-та, 2020. — 234 с.

5. Лесников В.С. Виды разметок текстовых корпусов русского языка // Научно-техническая информация. Сер. 2. Информационные процессы и системы. 2019. № 9. С. 27–30.

6. Бутенко Ю.И. Модель учебно-научного текста для разметки корпуса научно-технических текстов // Экономика. Информатика. 2021. Т. 48, № 1. С. 123–129.

7. Бутенко Ю.И., Попова Н.М. Особенности семантической разметки в корпусе научно-технических текстов // Проблемы лингвистики и лингводидактики в неязыковом вузе: 5-я Международная научно-практическая конференция. 2022. Т. 1. С. 324–328.

8. University Centre for Computer Corpus Research on Language [Электронный ресурс]. – URL: https://ucrel.lancs.ac.uk/usas (дата обращения: 20.05.2024).

9. Бутенко Ю.И., Киселёва А.Д. Анализ современных корпусов параллельных текстов // Актуальные проблемы лингвистики и лингводидактики в неязыковом вузе: 4-я Международная научно-практическая конференция. 2020. Т. 1. С. 238–242.

10. Sketch Engine: Create and search a text corpus [Электронный ресурс]. – URL: https://www.sketchengine.eu (дата обращения: 20.05.2024).

11. NoSketch Engine [Электронный ресурс]. – URL: https://nlp.fi.muni.cz/trac/noske (дата обращения: 20.05.2024).

12. OpenCorpora — открытый корпус [Электронный ресурс]. – URL: https://opencorpora.org (дата обращения: 25.03.2024).

13. Бутенко Ю.И., Строганов Ю.В., Бабаджанян Р.В. Исследовательский прототип параллельного корпуса научно-технических текстов // Актуальные проблемы лингвистики и лингводидактики в неязыковом вузе: 4-я Международная научно-практическая конференция. 2020. Т. 1. С. 205–209.

14. Date C. J. Relational Database Writings, 1991-1994. — Addison-Wesley Publishing Company., 1995 — 542 с.