МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ

ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. А. И. ГЕРЦЕНА»



Основная профессиональная образовательная программа

Направление подготовки 09.03.01 Информатика и вычислительная техника

Направленность (профиль) «Технологии разработки программного обеспечения»

форма обучения – очная

**ОТЧЁТ**

по реализации проекта для дисциплины «Базы данных»  
по направлению “09.03.01 – Информатика и вычислительная техника”  
(профиль: “Технологии разработки программного обеспечения ”)

Преподаватель: к.ф-м.н., доцент кафедры ИТиЭО

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

(Жуков Н. Н.)

Преподаватель: ассистент кафедры ИТиЭО

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

(Иванова Е. А.)

Студенты 2 курса:

Лебедев Д. С. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Собинин Е. Я. \_\_\_\_\_\_\_\_\_\_\_\_

Таринская Т. Г. \_\_\_\_\_\_\_\_\_\_\_\_\_

Санкт-Петербург  
2022

| Оглавление |  |
| --- | --- |
| Ответственные | 3 |
| Предметная область | 3 |
| Ход выполнения нормализации | 4 |
| 1. Концептуальное проектирование | 4 |
| 1.1 Определение сущностей и их документирование | 4 |
| 1.2 Определение связей между сущностями | 5 |
| 1.3 Создание ER-модели предметной области | 5 |
| Логическое проектирование | 6 |
| 2.1 Выбор модели данных | 6 |
| 2.2 Определение набора таблиц исходя из ER-модели и их документирование | 6 |
| 2.3 Нормализация таблиц | 6 |
| 2.4 Определение атрибутов, их значений и ключей для сущностей | 7 |
| 2.5 Итоговая логическая модель данных | 12 |
| 3. Физическое проектирование | 13 |
| 3.1 Проектирование таблиц базы данных средствами выбранной СУБД | 13 |
| Запросы на создание таблиц | 13 |
| Запросы на добавление внешних ключей | 15 |
| Запросы на заполнение таблиц | 16 |
| Запросы на изменение таблиц | 19 |
| Запросы на вывод данных | 20 |
| Запросы на создание индексов | 20 |
| Создание триггера и функции | 21 |
| Использования NoSQL баз данных для моделирования выбранной  предметной области | 21 |

# 

# 

# Ответственные

Лебедев Д.С. – разработчик проекта. В обязанности Лебедева Д.С. входил процесс концептуального, логического проектирования, определение сущностей и их атрибутов, процесс нормализации базы данных. При выполнении данного задания были использованы знания по следующим формам нормализации: 1НФ-5НФ. Подготовка отчета и презентации.

Собинин Е.Я. – разработчик проекта. В обязанности Собинина Е.Я. входил процесс логического, физического проектирования. Написание запросов на создание бд, создание таблиц, установку связей, заполнение таблиц, вывод информации. Подготовка отчета и презентации.

Таринская Т. Г. – разработчик проекта. В обязанности Таринской Т. Г. входил процесс концептуального, логического, физического проектирования, определение сущностей и их атрибутов, процесс нормализации базы данных. При выполнении данного задания были использованы знания по следующим формам нормализации: 1НФ-5НФ. Подготовка отчета и презентации.

# Предметная область

База данных электронной библиотеки должна позволять выполнять основные функции, необходимые для читателей.

**База данных должна позволять решать следующие задачи**

* позволять вести учет книг (название, автор, тематика, издательство, серия персоналия, обложка, наличие, другие поля)
* позволять вести учет и управлять информацией о библиотеках (название библиотеки, адрес, координаты, телефоны, почта, социальные сети, график работы)
* позволять вести учет и управлять информацией о мероприятиях/событиях библиотек (дата, время, название мероприятия, описание мероприятия, возрастной рейтинг, телефон и имя ответственного, страница регистрации, постер мероприятия, категория мероприятия и т.д)
* создавать и управлять подборками книг (название подборки, аннотация подборки / краткое описание, полное описание подборки / текстовый материал с описанием книг, постер для подборки, книги, участвующие в подборке, статус подборки: опубликован или нет, составитель)
* учесть необходимость связывания описанных выше сущностей между собой;
* позволять вести учет новостного раздела (новости могут публиковать каждая конкретная библиотека, а также новость может относится ко всем библиотекам, объединенным в сеть)

# 

# Ход выполнения нормализации

**1. Концептуальное проектирование**

**1.1 Определение сущностей и их документирование**

- Сущность "Книги/books":

этот объект будет хранить информацию о книгах.

**Атрибуты:** код книги, название, автор, издательство, год издания, тематика, бумажная версия, электронная версия, наличие, код библотеки.

- Сущность "Библиотека/library":

этот объект будет хранить информацию о библиотеке в базе данных. **Атрибуты:** код библиотеки, название, адрес, график работы, координаты, телефон, почта, социальные сети.

- Сущность "Событие/event":

этот объект будет хранить информацию о событии в базе данных. **Атрибуты:** код события, название, ФИО ответственного, телефон ответственного, дата, время, возрастной рейтинг, категория, код библиотеки.

- Сущность "Каталог/catalog":

этот объект будет хранить информацию о каталоге с книгами.

**Атрибуты:** код каталога, аннотация, название, статус подборки, ключевые слова, составитель, код книги.

- Сущность "Новость/news":

этот объект будет хранить информацию о новостях, проходящие в библиотеке/-ах.

**Атрибуты:** код новости, код события, описание новости, время публикации.

# 

# 

# 1.2 Определение связей между сущностями

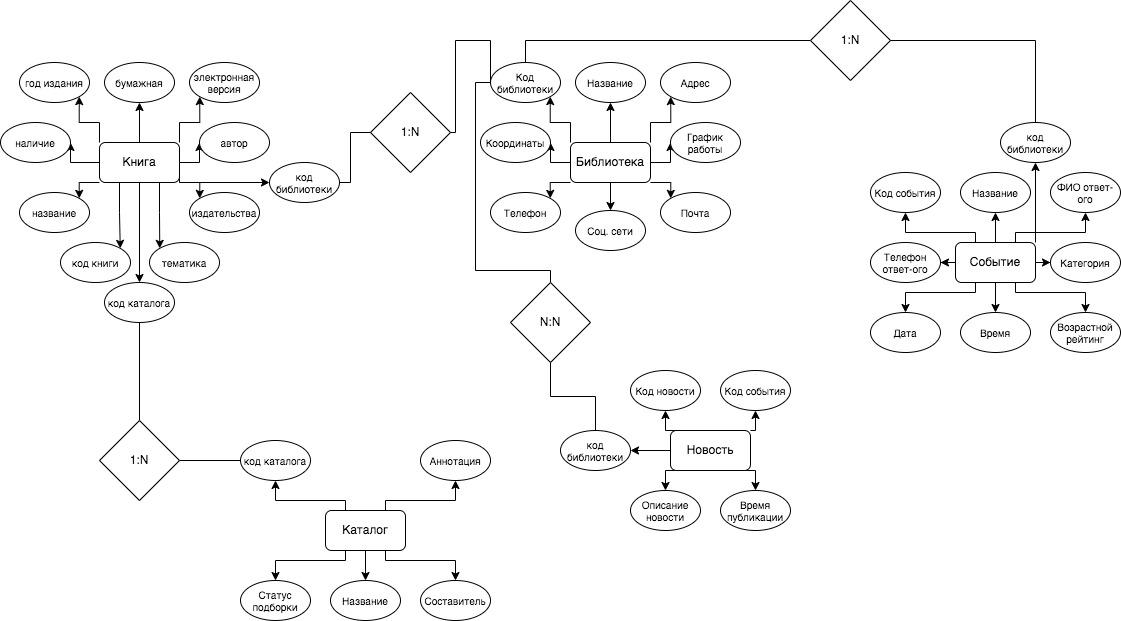
# 

- Связь "книги-библиотека"(1:N): одна книга относится к множеству библиотек;

- Связь "библиотека-событие"(1:N): в одной библиотеке может проходит как одно событие так и несколько;

- Связь "книги-каталог"(1:N): одна книга может содержаться в множестве каталогов;

- Связь "библиотека-новости"(N:N): множество библиотек могут публиковать множество новостей.



**1.3 Создание ER-модели предметной области**

# 

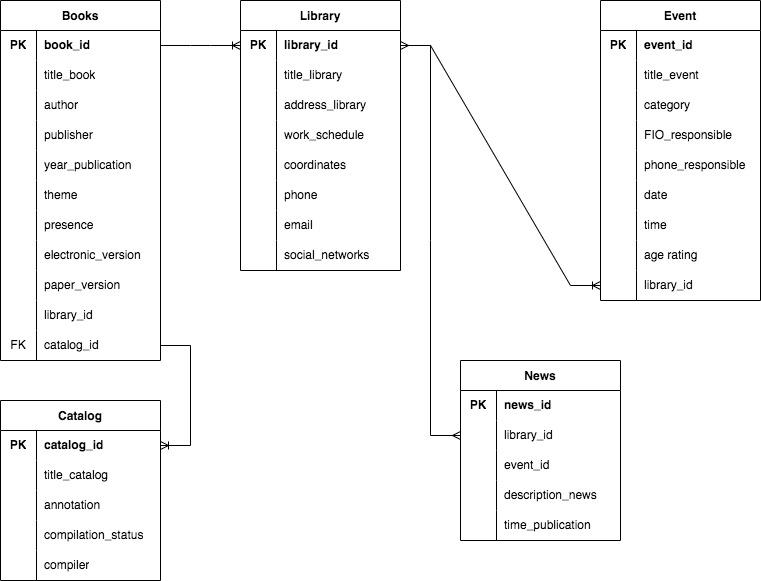
# 

**Логическое проектирование**

**2.1 Выбор модели данных**

Для визуализации данных в виде табличного представления была выбрана реляционная модель данных.

**2.2 Определение набора таблиц исходя из ER-модели и их документирование**

****

**2.3 Нормализация таблиц**

**1НФ:** Таблица находится в 1НФ, если все её поля содержат только простые неделимые значения.

Изменению подверглись следующие таблицы:

1. Books атрибуты:

author был вынесен в отдельную таблицу Author.

1. Library атрибуты

work\_schedule разделен на opening time и closing time.

1. Catalog и Event атрибуты:

responsible был вынесен в отдельную таблицу Responsible.

**2НФ:** Таблица находится в 2НФ, если она удовлетворяет требованиям 1НФ и не ключевые поля функционально полно зависят от первичного ключа. Данное требование выполняется для всех таблиц.

**3НФ:** Таблица находится в 3НФ если она удовлетворяет требованиям 2НФ и не содержит транзитивных зависимостей.

Транзитивная зависимость – зависимость между не ключевыми полями. Транзитивные зависимости были выявлены в следующий таблицах: Books (атрибут theme), Event (атрибуты category, age rating).

**НФБК:** Таблица находится в нормальной форме Бойса-Кодда тогда и только тогда, когда детерминанты всех её функциональных зависимостей являются потенциальными ключами.

Дополнительных зависимостей в структурах таблиц не найдено.

**4НФ:** Таблица находится в 4НФ, если она находится в НФБК и все нетривиальные многозначные зависимости фактически являются функциональными зависимостями от её потенциальных ключей.

Была обнаружена зависимость в таблице Library, для её устранения была создана дополнительная таблица Address в которую будут заноситься данные о улице, номере дома, координатах, следственно избавились от тривиальной многозначной зависимости и достигли 4НФ.

**5НФ:** Таблица находится в пятой нормальной форме тогда и только тогда, когда она находится в 4НФ и каждая нетривиальная зависимость определяется ее потенциальным ключом.

Нетривиальные зависимости не обнаружены, значит отношения приведены к 5НФ.

**2.4 Определение атрибутов, их значений и ключей для сущностей**

| **Имя атрибута** | **Описание** | **Тип данных** | **Размер** | **NOTNULL** | **ОДЗ** |
| --- | --- | --- | --- | --- | --- |
| **Books** | | | | | |
| **book\_id** | Уникальный номер книги | int | - | NOT NULL |  |
| title\_book | Название книги | varchar | 100 | NOT NULL |  |
| author\_id | автор | int | - | NOT NULL | Не содержит цифр |
| publisher | издательство | varchar | 100 | NOT NULL | Не содержит цифр |
| year\_publication | год издания | int | - | NOT NULL |  |
| theme\_id | тематика | int | - | NOT NULL |  |
| presence | наличие | bool | - | NOT NULL | 1 and 0 |
| electronic\_version | электронная версия | bool | - | NOT NULL | 1 and 0 |
| link\_e\_version | ссылка на эл. версию | varchar | 200 |  |  |
| paper\_version | бумажная версия | bool | - | NOT NULL | 1 and 0 |
| library\_id | уникальный номер библиотеки | int | - | NOT NULL |  |
| catalog\_id | уникальный номер каталога | int | - | NOT NULL |  |
| **Author** | | | | | |
| **author\_id** | уникальный номер автора | int | - | NOT NULL |  |
| surname | фамилия | varchar | 30 | NOT NULL | Не содержит цифр |
| name | имя | varchar | 30 | NOT NULL | Не содержит цифр |
| patronymic | отчество | varchar | 30 |  | Не содержит цифр |
| **Theme** | | | | | |
| **theme\_id** | уникальный номер тематики | int | - | NOT NULL |  |
| title\_theme | название тематики | varchar | 30 | NOT NULL |  |
| **Library** | | | | | |
| **library\_id** | уникальный номер библиотеки | int | - | NOT NULL |  |
| title\_library | название библиотеки | varchar | 100 | NOT NULL |  |
| address\_id | адрес библиотеки | int | - | NOT NULL |  |
| opening time | время открытия | time | 6 | NOT NULL | с 8:00 до 22:00 |
| closing time | время закрытия | time | 6 | NOT NULL | с 8:00 до 22:00 |
| phone | телефон | varchar | 30 | NOT NULL |  |
| email | почта | varchar | 40 | NOT NULL |  |
| social\_networks | соц. сети | varchar | 100 | NOT NULL |  |
| **Address** | | | | | |
| **address\_id** | уникальный номер адреса | int | - |  |  |
| coordinates | координаты | varchar | 30 | NOT NULL |  |
| street | улица | varchar | 30 | NOT NULL |  |
| house\_number | номер дома | varchar | 10 | NOT NULL |  |
| **News** | | | | | |
| **news\_id** | уникальный номер новости | int | - | NOT NULL |  |
| library\_id | уникальный номер библиотеки | int | - | NOT NULL |  |
| event\_id | уникальный номер события | int | - | NOT NULL |  |
| description\_news | описание новости | varchar | 100 | NOT NULL |  |
| time\_publication | время публикации | timestamp | 30 | NOT NULL |  |
| **Catalog** | | | | | |
| **catalog\_id** | уникальный номер каталога | int | - | NOT NULL |  |
| title\_catalog | название каталога | varchar | 50 | NOT NULL |  |
| annotation | аннотация | varchar | 100 | NOT NULL |  |
| compilation\_status | статус подборки | bool | - | NOT NULL | 1 and 0 |
| responsible\_id | составитель | int | - | NOT NULL | Не содержит цифр |
| **Event** | | | | | |
| **event\_id** | уникальный номер события | int | - | NOT NULL |  |
| title\_event | название события | varchar | 100 | NOT NULL |  |
| category | категория | int | - | NOT NULL |  |
| responsible\_id | ФИО ответственного | int | - | NOT NULL | Не содержит цифр |
| date\_ | дата | date | 15 | NOT NULL |  |
| time\_ | время | time | 15 | NOT NULL |  |
| age\_rating | возрастной рейтинг | int | - | NOT NULL | Не содержит букв |
| library\_id | уникальный номер библиотеки | varchar | - | NOT NULL |  |
| **Category** | | | | | |
| **category\_id** | уникальный номер библиотеки | int | - | NOT NULL |  |
| title\_category | название категории | varchar | 20 | NOT NULL |  |
| **Age\_rating** | | | | | |
| **age\_id** | уникальный номер возраста | int | - | NOT NULL |  |
| age | возраст | varchar | 5 | NOT NULL |  |
| **Responsible** | | | | | |
| **responsible\_id** | уникальный номер ответственного/составителя | int | - | NOT NULL |  |
| surname | фамилия | varchar | 30 | NOT NULL | Не содержит цифр |
| name | имя | varchar | 30 | NOT NULL | Не содержит цифр |
| patronymic | отчество | varchar | 30 |  | Не содержит цифр |
| phone | телефон | varchar | 30 |  |  |

# 

# 2.5 Итоговая логическая модель данных

# 

# 

# 

**3. Физическое проектирование**

**3.1 Проектирование таблиц базы данных средствами выбранной СУБД**

Для реализации была выбрана Postgresql, в основном она была выбрана из-за возможности масштабирования – максимальный размер БД не ограничен, равно как и количество строк/индексов в таблице. Ещё один нюанс – обычно объём данных ограничен наибольшим возможным размером файлов в операционной системе. Постгрес умеет обходить это ограничение за счёт того, что способна хранить табличную информацию в множестве файлов меньшего объёма.

**Запросы на создание таблиц**

Создание таблицы Books:

CREATE TABLE Books

(

book\_id SERIAL PRIMARY KEY NOT NULL,

title\_books varchar(100) NOT NULL,

presence bool NOT NULL,

year\_publicatin int NOT NULL,

paper\_version bool NOT NULL,

electronic\_version bool NOT NULL,

author int NOT NULL,

library\_id int NOT NULL,

publisher varchar(100) NOT NULL,

theme int NOT NULL

);

Создание таблицы Author:

CREATE TABLE Author

(

author\_id SERiAL PRIMARY KEY NOT NULL,

surname varchar(30) NOT NULL,

name varchar(30) NOT NULL,

patronymic varchar(30)

);

Создание таблицы Theme:

CREATE TABLE Theme

(

theme\_id SERiAL PRIMARY KEY NOT NULL,

title\_theme varchar(30) NOT NULL

);

Создание таблицы Library:

CREATE TABLE Library

(

library\_id SERIAL PRIMARY KEY NOT NULL,

title\_library varchar(100) NOT NULL,

address\_id int NOT NULL,

opening\_time time(6) NOT NULL,

closing\_time time(6) NOT NULL,

phone varchar(30) NOT NULL,

email varchar(40) NOT NULL,

social\_networks varchar(100) NOT NULL

);

Создание таблицы Address:

CREATE TABLE Address

(

address\_id SERIAL PRIMARY KEY NOT NULL,

coordinates varchar(30) NOT NULL,

street varchar(30) NOT NULL,

house\_number varchar(10) NOT NULL

);

Создание таблицы News:

CREATE TABLE News

(

news\_id SERIAL PRIMARY KEY NOT NULL,

library\_id int NOT NULL,

event\_id int NOT NULL,

description\_news varchar (100) NOT NULL,

time\_publication timestamp NOT NULL

);

Создание таблицы Catalog:

CREATE TABLE Catalog

(

catalog\_id SERIAL PRIMARY KEY NOT NULL,

book\_id int NOT NULL,

title\_catalog varchar (50) NOT NULL,

annotation varchar (100) NOT NULL,

compilation\_status bool NOT NULL,

responsible int NOT NULL

);

Создание таблицы Event:

CREATE TABLE Event

(

event\_id SERIAL PRIMARY KEY NOT NULL,

category varchar (20) NOT NULL,

responsible\_id int NOT NULL,

date\_ date NOT NULL,

time\_ time NOT NULL,

age\_id int NOT NULL,

library\_id int NOT NULL

);

Создание таблицы Category:

CREATE TABLE Category

(

category\_id SERIAL PRIMARY KEY NOT NULL,

title\_category varchar (20) NOT NULL

);

Создание таблицы Age\_rating:

CREATE TABLE Age\_rating

(

age\_id SERIAL PRIMARY KEY NOT NULL,

age varchar (5) NOT NULL

);

Создание таблицы Responsible:

CREATE TABLE Responsible

(

responsible\_id SERIAL PRIMARY KEY NOT NULL,

surname varchar(30) NOT NULL,

name varchar(30) NOT NULL,

patronymic varchar(30),

phone varchar(30) NOT NULL

);

**Запросы на добавление внешних ключей**

Внешние ключи для Books:

ALTER TABLE Books

ADD FOREIGN KEY (author\_id) REFERENCES Author(author\_id);

ALTER TABLE Books

ADD FOREIGN KEY (theme\_id) REFERENCES Theme(theme\_id);

ALTER TABLE Books

ADD FOREIGN KEY (library\_id) REFERENCES Library(library\_id);

Внешние ключи для Library:

ALTER TABLE Library

ADD FOREIGN KEY (address\_id) REFERENCES Address(address\_id);

Внешние ключи для Catalog:

ALTER TABLE Catalog

ADD FOREIGN KEY (responsible\_id) REFERENCES Responsible(responsible\_id);

Внешние ключи для Event:

ALTER TABLE Event

ADD FOREIGN KEY (category\_id) REFERENCES Category(category\_id);

ALTER TABLE Event

ADD FOREIGN KEY (responsible\_id) REFERENCES Responsible(responsible\_id);

ALTER TABLE Event

ADD FOREIGN KEY (age\_id) REFERENCES Age\_rating(age\_id);

ALTER TABLE Event

ADD FOREIGN KEY (library\_id) REFERENCES Library(library\_id);

Внешние ключи для News:

ALTER TABLE News

ADD FOREIGN KEY (event\_id) REFERENCES Event(event\_id);

**Запросы на заполнение таблиц**

Таблица Author:

INSERT INTO Author(surname, name)

VALUES ('Сартр', 'Жан-Поль');

INSERT INTO Author(surname, name)

VALUES ('Кьеркегор', 'Сёрен');

INSERT INTO Author(surname, name)

VALUES ('Хайдеггер', 'Мартин');

INSERT INTO Author(surname, name, patronymic)

VALUES ('Государев', 'Илья', 'Борисович');

INSERT INTO Author(surname, name, patronymic)

VALUES ('Дугин', 'Александр', 'Гельевич');

Таблица Theme:

INSERT INTO theme (title\_theme)

VALUES ('Философия');

INSERT INTO theme (title\_theme)

VALUES ('Художественная литература');

INSERT INTO theme (title\_theme)

VALUES ('Информатика');

Таблица Address:

INSERT INTO Address(coordinates, street, house\_number)

VALUES ('60.036305, 30.323985', 'пр-т Энгельса', '111к1');

INSERT INTO Address(coordinates, street, house\_number)

VALUES ('59.936069, 30.321005', 'пр-т Невский', '20');

INSERT INTO Address(coordinates, street, house\_number)

VALUES ('59.928052, 30.312697', 'Гражданская', '13');

Таблица Responsible:

INSERT INTO Responsible(surname, name, patronymic, phone)

VALUES ('Лебедев', 'Дмитрий', 'Сергеевич', '+79998887332');

INSERT INTO Responsible(surname, name, patronymic, phone)

VALUES ('Собинин', 'Егор', 'Яковлевич', '+79532224554');

INSERT INTO Responsible(surname, name, patronymic, phone)

VALUES ('Жуков', 'Николай', 'Николаевич', '+79001114554');

Таблица Age\_rating:

INSERT INTO Age\_rating(age)

VALUES ('0+');

INSERT INTO Age\_rating(age)

VALUES ('3+');

INSERT INTO Age\_rating(age)

VALUES ('6+');

INSERT INTO Age\_rating(age)

VALUES ('12+');

INSERT INTO Age\_rating(age)

VALUES ('16+');

INSERT INTO Age\_rating(age)

VALUES ('18+');

Таблица Category:

INSERT INTO Category(title\_category)

VALUES ('Открытие');

INSERT INTO Category(title\_category)

VALUES ('Литературные чтения');

INSERT INTO Category(title\_category)

VALUES ('Встреча с автором');

INSERT INTO Category(title\_category)

VALUES ('Технические работы');

INSERT INTO Category(title\_category)

VALUES ('Другое');

Таблица Library:

INSERT INTO Library(title\_library, address\_id, opening\_time, closing\_time, phone, email, social\_networks)

VALUES ('Центральная Районная Библиотека "В Озерках"', '1', '11:00:00', '20:00:00', '88125105795', 'info@cbsvib.ru', 'https://www.cbsvib.ru/');

INSERT INTO Library(title\_library, address\_id, opening\_time, closing\_time, phone, email, social\_networks)

VALUES ('НЕВСКИЙ20', '2', '10:00:00', '21:00:00', '88124930010', 'info@bc-nevsky20.ru', 'http://bc-nevsky20.ru/')

INSERT INTO Library(title\_library, address\_id, opening\_time, closing\_time, phone, email, social\_networks)

VALUES ('Библиотека мастеров интеллектуального труда – БМИТ', '3', '10:00:00', '18:00:00', '89313515575', 'info@inlibrary.ru', 'http://inlibrary.ru/')

Таблица Event:

INSERT INTO Event(title\_event, category\_id, responsible\_id, date\_, time\_, age\_id, library\_id)

VALUES ('Открытие библиотеки на Энгельса', '1', '1', '2022-06-01', '12:00:00', '1', '1');

INSERT INTO Event(title\_event, category\_id, responsible\_id, date\_, time\_, age\_id, library\_id)

VALUES ('Встреча с А.Г. Дугиным', '3', '2', '2022-07-01', '12:00:00', '5', '2');

INSERT INTO Event(title\_event, category\_id, responsible\_id, date\_, time\_, age\_id, library\_id)

VALUES ('Технические работы в библеотеке на Невском', '4', '3', '2022-06-24', '15:30:00', '1', '2');

INSERT INTO Event(title\_event, category\_id, responsible\_id, date\_, time\_, age\_id, library\_id)

VALUES ('Выступление местных поэтов', '2', '1', '2022-08-1', '16:00:00', '6', '3');

INSERT INTO Event(title\_event, category\_id, responsible\_id, date\_, time\_, age\_id, library\_id)

VALUES ('Музыкальный вечер', '5', '2', '2022-09-01', '18:00:00', '2', '1');

Таблица Books:

INSERT INTO books(title\_books, presence, year\_publicatin, paper\_version, electronic\_version, link\_e\_versin, author\_id, library\_id, publisher, theme\_id)

VALUES ('Введение в веб-разработку на языке JavaScript. Учебное пособие', '1', '2022', '1', '1', 'https://e.lanbook.com/book/206588', '4', '1', 'Лань', '3')

INSERT INTO books(title\_books, presence, year\_publicatin, paper\_version, electronic\_version, link\_e\_versin, author\_id, library\_id, publisher, theme\_id)

VALUES ('Тошнота', '0', '2020', '1', '1', 'https://libbox.ru/book/toshnota', '1', '2', 'АСТ', '2')

INSERT INTO books(title\_books, presence, year\_publicatin, paper\_version, electronic\_version, link\_e\_versin, author\_id, library\_id, publisher, theme\_id)

VALUES ('Дневник обольстителя', '1', '2021', '1', '1', 'https://librebook.me/dnevnik\_obolstitelia', '2', '3', 'Рипол Классик', '1')

INSERT INTO books(title\_books, presence, year\_publicatin, paper\_version, electronic\_version, author\_id, library\_id, publisher, theme\_id)

VALUES ('Что такое метафизика?', '1', '2020', '1', '0', '3', '1', 'Академический проект', '1');

INSERT INTO books(title\_books, presence, year\_publicatin, paper\_version, electronic\_version, author\_id, library\_id, publisher, theme\_id)

VALUES ('Археомодерн', '0', '2022', '0', '0', '5', '2', 'Академический проект', '1');

Таблица News:

INSERT INTO news(library\_ID, event\_id, description\_news, time\_publication)

VALUES ('1', '1', 'Открытие новой библиотеки. На откртытии будет присутсвовать глава администрации района и директор библиотеки.', '2022-05-22 19:10:25');

INSERT INTO news(library\_ID, event\_id, description\_news, time\_publication)

VALUES ('2', '2', 'Встреча с российским философом Александром Гельевичем Дугиным. На встрече будет проводиться автограф-сессия и конференция, на которой можно задать интересующие вопросы', '2022-06-22 19:10:25');

INSERT INTO news(library\_ID, event\_id, description\_news, time\_publication)

VALUES ('2', '3', 'Закрытие библиотеки на ремонт с 24 июня 2022. Библиотека работает до 15:30.', '2022-06-01 19:10:25');

INSERT INTO news(library\_ID, event\_id, description\_news, time\_publication)

VALUES ('3', '4', 'Проведение литературного вечера 1 августа 2022. На вечере будут выступать местные поэты. Можно будет услышать стихи русской классики и авторские произведения.', '2022-06-15 19:10:25');

INSERT INTO news(library\_ID, event\_id, description\_news, time\_publication)

VALUES ('1', '5', 'Музыкальный вечер в честь начала учебного года. Ожидается выступление ансамблей из музыкального колледжа', '2022-08-01 19:10:25');

Таблица Catalog:

INSERT INTO catalog(title\_catalog, annotation, compilation\_status, responsible\_id)

VALUES ('А подумать?!', 'Данный каталог предлагает к прочтению философские труды разных эпох', '1', '2');

INSERT INTO Catalog(title\_catalog, annotation, compilation\_status, responsible\_id)

VALUES ('Мне чуть больше двадцати. Я хочу войти в ИТ!', 'Все, что нужно разработчику', '0', '3');

**Запросы на изменение таблиц**

Удалим значение NOT NULL

ALTER TABLE books

ALTER COLUMN link\_e\_versin DROP NOT NULL;

Изменим длину строки

TABLE news

Alter COLUMN description\_news TYPE varchar(300);

--добавим и удалим столбцы из таблиц

--т.к. добавлять список в ячейку не есть хорошо, изменим структуру БД

--добавить catalog\_id в book

--указать внешний ключ

--удалить book\_id из catalog

--вписать нужные id из catalog в book

ALTER TABLE books ADD COLUMN catalog\_id int;

ALTER TABLE catalog DROP COLUMN book\_id;

ALTER TABLE Books

ADD FOREIGN KEY (catalog\_id) REFERENCES Catalog(catalog\_id);

UPDATE books

SET catalog\_id = 1

WHERE theme\_id = 1;

UPDATE books

SET catalog\_id = 2

WHERE theme\_id = 3;

**Запросы на вывод данных**

Возможное отображение информации для пользователя:

SELECT Books.title\_books , Author.Name, Author.Surname, Books.year\_publicatin, Books.publisher, Catalog.title\_catalog

FROM (Books INNER join Author on Books.author\_id = Author.author\_id) INNER JOIN Catalog

ON Books.catalog\_id = Catalog.catalog\_id

ORDER BY Books.title\_books;

SELECT Event.title\_event, Library.title\_library, News.description\_news, News.time\_publication

FROM (News INNER JOIN Event on News.event\_id = Event.event\_id) INNER JOIN Library

ON News.library\_id = Library.library\_id

ORDER BY News.time\_publication;

SELECT Library.title\_library, Address.street, Address.house\_number, Library.opening\_time, Library.closing\_time, Library.phone

FROM Library INNER JOIN Address

ON Library.address\_id = Address.address\_id;

**Запросы на создание индексов**

Индекс для названий книг

CREATE INDEX idx\_lib\_\_title on Books(title\_books);

Индекс для названий событий

CREATE INDEX idx\_lib\_\_event on Event(title\_event);

Индекс для поиска по авторам

CREATE INDEX idx\_lib\_name\_surname on Author (name, surname);

Такой индекс нам не подходит, поэтому удалим его и сделаем GIN

DROP INDEX idx\_lib\_name\_surname

Подключаем расширение:

CREATE EXTENSION pg\_trgm;

Делаем запрос:

CREATE INDEX trgm\_idx\_author\_surname\_name ON Author USING gin (surname gin\_trgm\_ops, name gin\_trgm\_ops);

**Создание триггера и функции**

Делаем таблицу для триггера:

CREATE TABLE IF NOT EXISTS trigger\_table\_resp(

name varchar(50),

surname varchar (50),

date DATE

);

Создаем функцию:

CREATE OR REPLACE FUNCTION afterinsert()

RETURNS trigger AS

$$

BEGIN

INSERT INTO trigger\_table\_resp(name, surname, date)

VALUES (NEW.name, NEW.surname, current\_date);

RETURN NEW;

END;

$$

LANGUAGE 'plpgsql';

CREATE TRIGGER after\_insert

AFTER INSERT

ON Responsible

FOR EACH ROW

EXECUTE PROCEDURE afterinsert();

**Использования NoSQL баз данных для моделирования выбранной**

**предметной области**

Если структура данных полностью рандомная - NoSQL подходит.

Если там есть хоть какие-то намеки на однородность - появится необходимость миграции данных, и индексации, а со всем этим NoSQL справляется сильно хуже, чем SQL. Так как например, если есть динамические структуры и их необходимо подключить/добавить в реляционную базу - у Postgresql есть jsonb, который достаточно хорошо выполняет данное действие.