Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение

высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики управления и технологий

Ли Александр Андреевич БД-241м

## **Практическая работа 2. Изучение методов хранения данных на основе NoSQL Cassandra**

Направление подготовки/специальность

38.04.05 - Бизнес-информатика

Бизнес-аналитика и большие данные

(очная форма обучения)

ФИО преподавателя

Босенко Тимур Муртазович

Вариант 12

Москва

2025

Цель работы: получить практические навыки работы с базой данных Cassandra, изучив основные операции по управлению данными, включая создание и использование ключспейсов, таблиц, выполнение запросов CQL, а также работу с различными инструментами подключения и администрирования.

Ход работы

Запускаем Cassandra

**Sudo docker compose start**

Изображение выглядит как текст, снимок экрана, мультимедиа, гаджет

Автоматически созданное описание

Рис.1 Запуск кассандры

Проверяем запущенные сервисы Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рис.2 Проверка zeppelin

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рис.3 Проверка Cassandra web

Создали запись в кассандре

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рис.4 создание записи

Просматриваем существующие в настоящее время пространства ключей

**SELECT \* FROM system\_schema.keyspaces;**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рис.5 Пространства ключей

**CREATE KEYSPACE movies WITH replication =**

**{'class':'SimpleStrategy','replication\_factor':1};**

Изображение выглядит как текст, Шрифт, линия, снимок экрана

Автоматически созданное описание

Рис.6 создали новое пространство ключа

Создадим таблицы movie и actor

**DROP TABLE IF EXISTS movies.movie;**

**CREATE TABLE movies.movie (movie\_id int,**

**title text, // title**

**release\_year int, // year**

**running\_time int, // runtimes**

**languages set<text>, // language codes**

**genres set<text>, // genres**

**plot\_outline text, // plot outline**

**cover\_url text, // cover url**

**top250\_rank int, // top 250 rank**

**PRIMARY KEY (movie\_id)**

**);**

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Автоматически созданное описание

Рис.7 Создание таблицы movie2

**DROP TABLE IF EXISTS movies.actor;**

**CREATE TABLE movies.actor (actor\_id int,**

**name text, // name**

**headshot\_url text, // headshot**

**mini\_biography text, // mini biography**

**birth\_date text, // birth date**

**trade\_mark list<text>, // trade mark**

**PRIMARY KEY (actor\_id)**

**);**

Изображение выглядит как текст, снимок экрана, Шрифт, дисплей

Автоматически созданное описание

Рис.8 Создание таблицы actor

Добавим фильмы "Матрица" и "Криминальное чтиво"

// insert "The Matrix" - 0133093

INSERT INTO movies.movie (movie\_id, title, release\_year, running\_time, languages,

genres, plot\_outline, cover\_url, top250\_rank)

VALUES (0133093,

'The Matrix',

1999,

136,

{'en'},

{'Action', 'Sci-Fi'},

$$Thomas A. Anderson is a man living two lives. By day he is an average computer programmer and by night a hacker known as Neo. Neo has always questioned his reality, but the truth is far beyond his imagination. Neo finds himself targeted by the police when he is contacted by Morpheus, a legendary computer hacker branded a terrorist by the government. Morpheus awakens Neo to the real world, a ravaged wasteland where most of humanity have been captured by a race of machines that live off of the humans' body heat and electrochemical energy and who imprison their minds within an artificial reality known as the Matrix. As a rebel against the machines, Neo must return to the Matrix and confront the agents: super-powerful computer programs devoted to snuffing out Neo and the entire human rebellion$$,

'https://m.media-amazon.com/images/M/MV5BNzQzOTk3OTAtNDQ0Zi00ZTVkLWI0MTEtMDllZjNkYzNjNTc4L2ltYWdlXkEyXkFqcGdeQXVyNjU0OTQ0OTY@.\_V1\_SX101\_CR0,0,101,150\_.jpg',

19);

// insert "Pulp Fiction" - 0110912

INSERT INTO movies.movie (movie\_id, title, release\_year, running\_time, languages,

genres, plot\_outline, cover\_url, top250\_rank)

VALUES (0110912,

'Pulp Fiction',

1994,

154,

{'en', 'es', 'fr'},

{'Crime', 'Drama'},

$$Jules Winnfield (Samuel L. Jackson) and Vincent Vega (John Travolta) are two hit men who are out to retrieve a suitcase stolen from their employer, mob boss Marsellus Wallace (Ving Rhames). Wallace has also asked Vincent to take his wife Mia (Uma Thurman) out a few days later when Wallace himself will be out of town. Butch Coolidge (Bruce Willis) is an aging boxer who is paid by Wallace to lose his fight. The lives of these seemingly unrelated people are woven together comprising of a series of funny, bizarre and uncalled-for incidents.$$,

'https://m.media-amazon.com/images/M/MV5BNGNhMDIzZTUtNTBlZi00MTRlLWFjM2ItYzViMjE3YzI5MjljXkEyXkFqcGdeQXVyNzkwMjQ5NzM@.\_V1\_SY150\_CR1,0,101,150\_.jpg',

8);

// insert "Speed" - 0111257

INSERT INTO movies.movie (movie\_id, title, release\_year, running\_time, languages,

genres, plot\_outline, cover\_url, top250\_rank)

VALUES (0111257,

'Speed',

1994,

116,

{'en'},

{'Action', 'Adventure', 'Crime', 'Thriller'},

$$Bomber extortionist's elevator plan backfires, so he rigs a bomb to a LA city bus. The stipulation is: once armed, the bus must stay above 50 mph to keep from exploding. Also if LAPD Officer tries to unload any passengers off, Payne will detonate it. Joe Morton co-stars as Jack's superior, and Jeff Daniels supports Jack helping him try to defuse the bomb.$$,

'https://m.media-amazon.com/images/M/MV5BYjc0MjYyN2EtZGRhMy00NzJiLWI2Y2QtYzhiYTU3NzAxNzg4XkEyXkFqcGdeQXVyMTQxNzMzNDI@.\_V1\_SY150\_CR0,0,101,150\_.jpg',

null);

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.9 Добавили фильмы

Добавили актеров "Bruce Willis", "John Travolta", "Sandra Bullock", "Samuel L. Jackson", "Uma Thurman" и "Quentin Tarantino"

// insert "Bruce Willis" - 0000246

INSERT INTO movies.actor (actor\_id, name, headshot\_url, birth\_date, trade\_mark)

VALUES (0000246,

'Bruce Willis',

'https://m.media-amazon.com/images/M/MV5BMjA0MjMzMTE5OF5BMl5BanBnXkFtZTcwMzQ2ODE3Mw@@.\_V1\_UY98\_CR8,0,67,98\_AL\_.jpg',

'1955-03-19',

['Frequently plays a man who suffered a tragedy, had lost something or had a crisis of confidence or conscience.',

'Frequently plays likeable wisecracking heroes with a moral centre',

'Headlines action-adventures, often playing a policeman, hitman or someone in the military',

'Often plays men who get caught up in situations far beyond their control',

'Sardonic one-liners',

'Shaven head',

'Distinctive, gravelly voice',

'Smirky grin.',

'Known for playing cynical anti-heroes with unhappy personal lives']);

// insert "John Travolta" - 0000237

INSERT INTO movies.actor (actor\_id, name, headshot\_url, birth\_date, trade\_mark)

VALUES (0000237,

'John Travolta',

'https://m.media-amazon.com/images/M/MV5BMTUwNjQ0ODkxN15BMl5BanBnXkFtZTcwMDc5NjQwNw@@.\_V1\_UY98\_CR3,0,67,98\_AL\_.jpg',

'1954-02-18',

['Cleft chin and razor-sharp cheekbones',

'Often works some sort of dance into his roles',

'New Jersey accent',

'Black hair and blue eyes']);

// insert "Sandra Bullock" - 0000113

INSERT INTO movies.actor (actor\_id, name, headshot\_url, birth\_date, trade\_mark)

VALUES (0000113,

'Sandra Bullock',

'https://m.media-amazon.com/images/M/MV5BMTI5NDY5NjU3NF5BMl5BanBnXkFtZTcwMzQ0MTMyMw@@.\_V1\_UX67\_CR0,0,67,98\_AL\_.jpg',

'1964-07-26',

null);

// insert "Samuel L. Jackson" - 0000168

INSERT INTO movies.actor (actor\_id, name, headshot\_url, birth\_date, trade\_mark)

VALUES (0000168,

'Samuel L. Jackson',

'https://m.media-amazon.com/images/M/MV5BMTQ1NTQwMTYxNl5BMl5BanBnXkFtZTYwMjA1MzY1.\_V1\_UX67\_CR0,0,67,98\_AL\_.jpg',

'1948-12-21',

['Deep authoritative voice',

'Rebellious characters who are disliked or considered strange by others in the story',

'Often plays police officers or government officials. Both prone to intimidation or violence',

'Often plays very wise and intelligent characters with great capacities for violence',

'Frequently plays tough characters who swear a lot',

'Frequent swearing',

'Often sports a moustache or goatee in his films',

'Shaven head',

'Kangol hats',

'Often plays hotheaded characters with a fiery temper',

'Often shouts the word "motherf\*\*\*\*\*" at some point in a film',

'Frequently cast by Quentin Tarantino']);

// insert "Uma Thurman" - 0000235

INSERT INTO movies.actor (actor\_id, name, headshot\_url, birth\_date, trade\_mark)

VALUES (0000235,

'Uma Thurman',

'https://m.media-amazon.com/images/M/MV5BMjMxNzk1MTQyMl5BMl5BanBnXkFtZTgwMDIzMDEyMTE@.\_V1\_UX67\_CR0,0,67,98\_AL\_.jpg',

'1970-04-29',

['Long blond hair and blue eyes', 'Statuesque, model-like figure']);

// insert "Keanu Reeves" - 0000206

INSERT INTO movies.actor (actor\_id, name, headshot\_url, birth\_date, trade\_mark)

VALUES (0000206,

'Keanu Reeves',

'https://m.media-amazon.com/images/M/MV5BNjUxNDcwMTg4Ml5BMl5BanBnXkFtZTcwMjU4NDYyOA@@.\_V1\_UY98\_CR4,0,67,98\_AL\_.jpg',

'1964-09-02',

['Intense contemplative gaze',

'Deep husky voice',

'Known for playing stoic reserved characters']);

// insert "Quentin Tarantino" - 0000233

INSERT INTO movies.actor (actor\_id, name, headshot\_url, birth\_date, trade\_mark)

VALUES (0000233,

'Quentin Tarantino',

'https://m.media-amazon.com/images/M/MV5BMTgyMjI3ODA3Nl5BMl5BanBnXkFtZTcwNzY2MDYxOQ@@.\_V1\_UX67\_CR0,0,67,98\_AL\_.jpg',

'1963-03-27',

['Lead characters usually drive General Motors vehicles, particularly Chevrolet and Cadillac, such as Jules 1974 Nova and Vincents 1960s Malibu.',

'Briefcases and suitcases play an important role in Pulp Fiction (1994), Reservoir Dogs (1992), Jackie Brown (1997), True Romance (1993) and Kill Bill: Vol. 2 (2004).',

'Makes references to cult movies and television',

'Frequently works with Harvey Keitel, Tim Roth, Michael Madsen, Uma Thurman, Michael Bowen, Samuel L. Jackson, Michael Parks and Christoph Waltz.',

'His films usually have a shot from inside an automobile trunk',

'He always has a Dutch element in his films: The opening tune, Little Green Bag, in Reservoir Dogs (1992) was performed by George Baker Selection and written by Jan Gerbrand Visser and Benjamino Bouwens who are all Dutch. The character Freddy Newandyke, played by Tim Roth is a direct translation to a typical Dutch last name, Nieuwendijk. The code name of Tim Roth is Mr. Orange, the royal color of Holland and the last name of the royal family. The Amsterdam conversation in Pulp Fiction (1994), Vincent Vega smokes from a Dutch tobacco shag (Drum), the mentioning of Rutger Hauer in Jackie Brown (1997), the brides name is Beatrix, the name of the Royal Dutch Queen.',

'[The Mexican Standoff] All his movies (including True Romance (1993), which he only wrote and did not direct) feature a scene in which three or more characters are pointing guns at each other at the same time.',

'Often uses an unconventional storytelling device in his films, such as retrospect (Reservoir Dogs (1992)), non-linear (Pulp Fiction (1994)), or "chapter" format (Kill Bill: Vol. 1 (2003)).',

'His films will often include one long, unbroken take where a character is followed around somewhere.']);

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Рис.10 Добавили актеров

Создадим таблицы фильмы по актерам и актеры по фильмам

DROP TABLE IF EXISTS movies.movies\_by\_actor;

CREATE TABLE movies.movies\_by\_actor (actor\_id int,

movie\_id int,

title text,

PRIMARY KEY (actor\_id, movie\_id)

);

DROP TABLE IF EXISTS movies.actors\_by\_movie;

CREATE TABLE movies.actors\_by\_movie (movie\_id int,

title text STATIC,

actor\_id int,

name text,

PRIMARY KEY (movie\_id, actor\_id)

);

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рис.11 Создание таблиц

Добавим несколько фильмов для данных актеров.

**// Фильмы для актера "Bruce Willis"**

**INSERT INTO movies.movies\_by\_actor (actor\_id, movie\_id, title)**

**VALUES (0000246, 0110912, 'Pulp Fiction');**

**INSERT INTO movies.movies\_by\_actor (actor\_id, movie\_id, title)**

**VALUES (0000246, 1606378, 'A Good Day to Die Hard');**

**INSERT INTO movies.movies\_by\_actor (actor\_id, movie\_id, title)**

**VALUES (0000246, 0217869, 'Unbreakable');**

**INSERT INTO movies.movies\_by\_actor (actor\_id, movie\_id, title)**

**VALUES (0000246, 0377917, 'The Fifth Element');**

**INSERT INTO movies.movies\_by\_actor (actor\_id, movie\_id, title)**

**VALUES (0000246, 0112864, 'Die Hard: With a Vengeance');**

**// Фильмы для актера "Keanu Reeves"**

**INSERT INTO movies.movies\_by\_actor (actor\_id, movie\_id, title)**

**VALUES (0000206, 0133093, 'The Matrix');**

**INSERT INTO movies.movies\_by\_actor (actor\_id, movie\_id, title)**

**VALUES (0000206, 0234215, 'The Matrix Reloaded');**

**INSERT INTO movies.movies\_by\_actor (actor\_id, movie\_id, title)**

**VALUES (0000206, 0111257, 'Speed');**

**// Фильмы для актера "Sandra Bullock"**

**INSERT INTO movies.movies\_by\_actor (actor\_id, movie\_id, title)**

**VALUES (0000113, 0111257, 'Speed');**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис.12 Добавление данных актеров

Добавим данные в таблицу actors\_by\_movie

**// Актеры для фильма "The Matrix"**

**INSERT INTO movies.actors\_by\_movie (movie\_id, title, actor\_id, name)**

**VALUES (0133093, 'The Matrix', 0000206, 'Keanu Reeves');**

**INSERT INTO movies.actors\_by\_movie (movie\_id, title, actor\_id, name)**

**VALUES (0133093, 'The Matrix', 0000401, 'Laurence Fishburne');**

**INSERT INTO movies.actors\_by\_movie (movie\_id, title, actor\_id, name)**

**VALUES (0133093, 'The Matrix', 0005251, 'Carrie-Anne Moss');**

**INSERT INTO movies.actors\_by\_movie (movie\_id, title, actor\_id, name)**

**VALUES (0133093, 'The Matrix', 0915989, 'Hugo Weaving');**

**// Актеры для фильма "Pulp Fiction"**

**INSERT INTO movies.actors\_by\_movie (movie\_id, title, actor\_id, name)**

**VALUES (0110912, 'Pulp Fiction', 0000237, 'John Travolta');**

**INSERT INTO movies.actors\_by\_movie (movie\_id, title, actor\_id, name)**

**VALUES (0110912, 'Pulp Fiction', 0000168, 'Samuel L. Jackson');**

**INSERT INTO movies.actors\_by\_movie (movie\_id, title, actor\_id, name)**

**VALUES (0110912, 'Pulp Fiction', 0000246, 'Bruce Willis');**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описаниерис.13 добавление данных в actors\_by\_movie

Добавим рейтинги для фильма "Криминальное чтиво" используя столбцы-счетчики

**DROP TABLE IF EXISTS movies.rating\_by\_movie;**

**CREATE TABLE movies.rating\_by\_movie (movie\_id int,**

**one\_star counter,**

**two\_star counter,**

**three\_star counter,**

**four\_star counter,**

**five\_star counter,**

**PRIMARY KEY (movie\_id)**

**);**

**UPDATE movies.rating\_by\_movie**

**SET five\_star = five\_star + 1**

**WHERE movie\_id = 0110912;**

**UPDATE movies.rating\_by\_movie**

**SET four\_star = four\_star + 1**

**WHERE movie\_id = 0110912;**

**UPDATE movies.rating\_by\_movie**

**SET five\_star = five\_star + 1**

**WHERE movie\_id = 0110912;**

**UPDATE movies.rating\_by\_movie**

**SET five\_star = five\_star + 1**

**WHERE movie\_id = 0110912;**

**UPDATE movies.rating\_by\_movie**

**SET two\_star = two\_star + 1**

**WHERE movie\_id = 0110912;**

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рис.14 Добавление рейтингов

Проверка текущих рейтингов криминального чтива

**SELECT \* FROM movies.rating\_by\_movie WHERE movie\_id = 0110912;**

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рис.15 Проверка рейтингов криминального чтива

Cоздадим таблицу, которая подсчитывает количество просмотров каждого фильма, разделенных на зрителей-мужчин и женщин, и по месяцам

**DROP TABLE IF EXISTS movies.movie\_viewed\_by\_time;**

**CREATE TABLE movies.movie\_viewed\_by\_time (movie\_id int,**

**year int,**

**month int,**

**male counter,**

**female counter,**

**PRIMARY KEY (movie\_id, year, month)**

**) WITH CLUSTERING ORDER BY (year DESC, month DESC);**

Изображение выглядит как текст, Шрифт, линия, снимок экрана

Автоматически созданное описание

Рис.16 таблицы подсчитывает количество просмотров каждого фильма

Добавим несколько примеров значений в новую таблицу.

**// Pulp Fiction Views 2019/03**

**UPDATE movies.movie\_viewed\_by\_time**

**SET male = male + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 03;**

**UPDATE movies.movie\_viewed\_by\_time**

**SET male = male + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 03;**

**UPDATE movies.movie\_viewed\_by\_time**

**SET female = female + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 03;**

**// Pulp Fiction Views 2019/04**

**UPDATE movies.movie\_viewed\_by\_time**

**SET female = female + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 04;**

**UPDATE movies.movie\_viewed\_by\_time**

**SET male = male + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 04;**

**UPDATE movies.movie\_viewed\_by\_time**

**SET female = female + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 04;**

**UPDATE movies.movie\_viewed\_by\_time**

**SET female = female + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 04;**

**// Pulp Fiction Views 2019/05**

**UPDATE movies.movie\_viewed\_by\_time**

**SET male = male + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 05;**

**UPDATE movies.movie\_viewed\_by\_time**

**SET male = male + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 05;**

**UPDATE movies.movie\_viewed\_by\_time**

**SET female = female + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 05;**

**UPDATE movies.movie\_viewed\_by\_time**

**SET female = female + 1**

**WHERE movie\_id = 0110912 AND year = 2019 and month = 05;**

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.17 добавлений новых значений

Выведем просмотры фильма "Криминальное чтиво" за все время

SELECT \*

FROM movies.movie\_viewed\_by\_time

WHERE movie\_id = 0110912;

Выведем просмотры фильма "Криминальное чтиво" за один месяц

SELECT \*

FROM movies.movie\_viewed\_by\_time

WHERE movie\_id = 0110912 AND year = 2019 AND month = 05;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.18просмотры Криминального чтива

Выведем просмотры фильма "Криминальное чтиво" за месяц с января по май

**SELECT \***

**FROM movies.movie\_viewed\_by\_time**

**WHERE movie\_id = 0110912 AND year = 2019 AND month >= 01 AND month <= 5;**

Изображение выглядит как текст, линия, число, Шрифт

Автоматически созданное описание

Рис.19 Просмотры с января по май

Индивидуальное задание

Создайте ключспейс universities с репликацией SimpleStrategy и коэффициентом репликации 1.

**CREATE KEYSPACE universities**

**WITH replication = {'class': 'SimpleStrategy', 'replication\_factor': 1};**

**DESCRIBE KEYSPACES;**

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.20 создание ключспейс universities

**DESCRIBE KEYSPACE universities;**

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рис.21 информация о ключспейс universities

Создайте таблицу professors в ключспейсе universities с полями professor\_id (int), name (text), department (text), email (text), office (text) и первичным ключом professor\_id.

**CREATE TABLE universities.professors (**

**professor\_id INT PRIMARY KEY,**

**name TEXT,**

**department TEXT,**

**email TEXT,**

**office TEXT**);

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рис.22 Создание таблицы professors

Вставьте три профессора в таблицу professors.

**INSERT INTO universities.professors (professor\_id, name, department, email, office)**

**VALUES (1, ‘Bosenko T M', 'Big data, ' BosenkoTM@gmail.com', 'Room 101');**

**INSERT INTO universities.professors (professor\_id, name, department, email, office)**

**VALUES (2, 'Sahnyk P A', AI, ' SahnykPA@gmail.com', 'Room 202');**

**INSERT INTO universities.professors (professor\_id, name, department, email, office)**

**VALUES (3, 'Frolov Y V', Economic science', ' FrolovYV@gmail.com', 'Room 303');**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис.23 добавление информации о профессорах

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.24 добавление информации о профессорах

Выберите всех профессоров из таблицы professors.

**SELECT \* FROM universities.professors;**

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.25 вывод всех профессоров

Обновите поле office профессора с professor\_id = 3.

**UPDATE universities.professors**

**SET office = 'Room 303'**

**WHERE professor\_id = 3;**

Изображение выглядит как текст, число, Шрифт, линия

Автоматически созданное описание

Рис.26 обновление поля третьего профессора

заключение

После выполнения лабораторной работы, были получены практические навыки работы с базой данных Cassandra, изучены основные операции по управлению данными, включая создание и использование ключспейсов, таблиц, выполненены запросов CQL, а также работа с различными инструментами подключения и администрирования.