**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 4](#_Toc201310080)

[ГЛАВА 1. ОБЗОР ИНСТРУМЕНТОВ 6](#_Toc201310081)

[ГЛАВА 2. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ АНАЛИЗА С ИСПОЛЬЗОВАНИЕМ PANDAS И POLARS 8](#_Toc201310082)

[ГЛАВА 3. ВЫВОДЫ И РЕКОМЕНДАЦИИ 11](#_Toc201310083)

[ЗАКЛЮЧЕНИЕ 12](#_Toc201310084)

[Список литературы 13](#_Toc201310085)

# **ВВЕДЕНИЕ**

В условиях стремительного роста объёмов цифровой информации анализ данных становится ключевым элементом в принятии решений в науке, экономике и управлении. По прогнозам аналитического агентства IDC, к 2025 году ежегодный объём создаваемой и реплицируемой информации превысит 180 зеттабайт [1]. Это требует не только мощной инфраструктуры хранения, но и эффективных инструментов обработки и анализа структурированных данных.

В среде программирования Python одной из наиболее популярных библиотек для работы с табличными данными является **Pandas**. Благодаря интуитивно понятному синтаксису, широкому функционалу и глубокой интеграции с другими библиотеками (такими как NumPy, Matplotlib, Scikit-learn), она получила широкое распространение в исследовательской и прикладной аналитике. Однако, несмотря на зрелость и богатую экосистему, архитектура Pandas имеет ряд ограничений: все операции выполняются в одном потоке, а данные целиком загружаются в оперативную память, что затрудняет обработку больших объёмов.

В ответ на эти вызовы появляются новые библиотеки, ориентированные на производительность и масштабируемость. Одной из таких является **Polars**, реализованная на языке Rust. Она поддерживает многопоточную и ленивую обработку, использует колонно-ориентированное хранение, а также интегрируется с форматом Apache Arrow, обеспечивая высокую скорость выполнения операций и эффективное использование ресурсов системы [2].

Актуальность исследования заключается в необходимости осознанного выбора инструментов анализа данных с учётом специфики задачи, объёма информации и доступных вычислительных ресурсов. Объективное сравнение возможностей Pandas и Polars позволяет определить оптимальные условия их использования и повысить эффективность аналитических процессов.

Цель работы — провести сравнительный анализ библиотек Pandas и Polars на примере типовой задачи фильтрации табличных данных.

Задачи**:**

* исследовать архитектурные и функциональные особенности Pandas и Polars;
* реализовать одинаковую задачу анализа с использованием обеих библиотек;
* сравнить производительность при одинаковых условиях;
* сформулировать практические выводы и рекомендации по выбору инструмента.

Объект исследования — библиотеки Pandas и Polars. Предмет исследования — производительность и применимость указанных библиотек в задачах аналитики.

Курсовая работа включает три главы. Первая посвящена теоретическому обзору инструментов. Вторая содержит практическую реализацию и измерения производительности. В третьей главе сформулированы выводы и рекомендации.

# **ГЛАВА 1. ОБЗОР ИНСТРУМЕНТОВ**

## **1.1 Pandas**

Библиотека Pandas — фундаментальный инструмент анализа табличных данных в Python. Название происходит от «panel data» — многомерных панельных структур, используемых в экономике и статистике. С момента своего появления в 2008 году Pandas зарекомендовала себя как надёжное и универсальное средство обработки данных.

Основные структуры — Series (одномерный массив) и DataFrame (двумерная таблица). Они позволяют удобно представлять и анализировать данные, выполнять фильтрацию, агрегацию, преобразования типов, объединение таблиц, анализ временных рядов.

Ключевые преимущества Pandas:

* простой и логичный синтаксис;
* широкий набор встроенных методов (фильтрация, группировка, агрегация, сводные таблицы и др.);
* поддержка большого числа форматов (CSV, Excel, JSON, SQL и др.);
* интеграция с NumPy, Matplotlib, Seaborn и другими библиотеками;
* активное сообщество и хорошая документация.

Ограничения:

* данные целиком загружаются в оперативную память, что затрудняет работу с крупными файлами;
* все операции выполняются последовательно в одном потоке;
* при больших объёмах возможно замедление и высокая нагрузка на систему.

Таким образом, Pandas — оптимальный выбор для задач умеренного объёма, прототипирования, обучения и построения аналитических отчётов.

## 1.2 Polars

Polars — высокопроизводительная библиотека, разработанная на языке Rust с фокусом на параллельную и ленивую обработку данных. Это современный инструмент, ориентированный на скорость, эффективность и масштабируемость.

Особенности архитектуры:

* колонно-ориентированная модель (Columnar Storage), ускоряющая операции агрегации, фильтрации и выборки;
* автоматическое распределение вычислений по ядрам процессора (многопоточность);
* ленивые вычисления (*lazy evaluation*), при которых операции оптимизируются и исполняются как единый план.

Преимущества Polars:

* высокая производительность (в 5–10 раз выше, чем у Pandas при больших данных);
* низкое потребление памяти за счёт Apache Arrow;
* выразительный и краткий синтаксис;
* возможность обработки данных, не помещающихся в оперативную память.

Ограничения:

* менее развитая экосистема;
* отличия в синтаксисе требуют адаптации;
* для некоторых операций (многотабличные объединения, сложные индексы) требуется иная логика, чем в Pandas.

Polars особенно эффективен в случаях обработки миллионов строк, при построении отчётных пайплайнов и в условиях ограниченных ресурсов.

## 1.3 Сравнительный анализ Pandas и Polars

Обе библиотеки решают сходные задачи — фильтрация, агрегация, группировка, работа с временными данными. Однако их архитектура и принципы исполнения существенно различаются.

Основные отличия представлены в таблице:я

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | Критерий | Pandas | Polars | | --- | --- | --- | | Язык реализации | Python (с C/NumPy) | Rust (с API для Python) | | Архитектура | Строко-ориентированная, однопоточная | Колонно-ориентированная, многопоточная | | Модель вычислений | Немедленная (eager execution) | Ленивые вычисления (lazy execution) | | Производительность | Высокая при малом объёме | Очень высокая при любом объёме | | Использование памяти | Среднее или высокое | Эффективное (Apache Arrow) | | Масштабируемость | Ограничена оперативной памятью | Поддержка out-of-core обработки | | Форматы файлов | CSV, Excel, JSON, SQL, HDF5 | CSV, Parquet, JSON, IPC | | Порог входа | Низкий | Средний | | Интеграция с библиотеками | NumPy, Matplotlib, Seaborn, Scikit-learn | PyArrow, Plotly, Scikit-learn (в процессе) | | Поддержка временных рядов | Полная | Ограничена, но развивается | |

# ГЛАВА 2. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ АНАЛИЗА С ИСПОЛЬЗОВАНИЕМ PANDAS И POLARS

## 2.1 Описание используемого датасета

Для проведения практического эксперимента использован датасет winners.csv, содержащий структурированную информацию о победителях автоспортивных соревнований. Каждая запись включает такие атрибуты, как дата гонки, название этапа, трасса, имя победителя и его команда. Данные представлены в виде классической таблицы, что делает их подходящими для демонстрации возможностей инструментов анализа.

Сценарий эксперимента включает:

* фильтрацию строк по определённому году (например, 2000);
* выборку записей с упоминанием конкретного гонщика (например, Schumacher);
* сравнительный анализ времени выполнения операций с использованием Pandas и Polars.

Таким образом, задача имитирует типичный аналитический запрос: быстрая выборка информации по заданным условиям из большого массива данных.

## 2.2 Реализация с использованием библиотеки Pandas

…

Комментарии:

* Pandas использует строковые методы для фильтрации значений, что удобно при работе с датами, хранящимися в текстовом формате.
* Поиск осуществляется методом str.contains, обеспечивающим гибкий поиск по подстроке.
* Все операции выполняются последовательно, и весь датасет загружается в оперативную память.

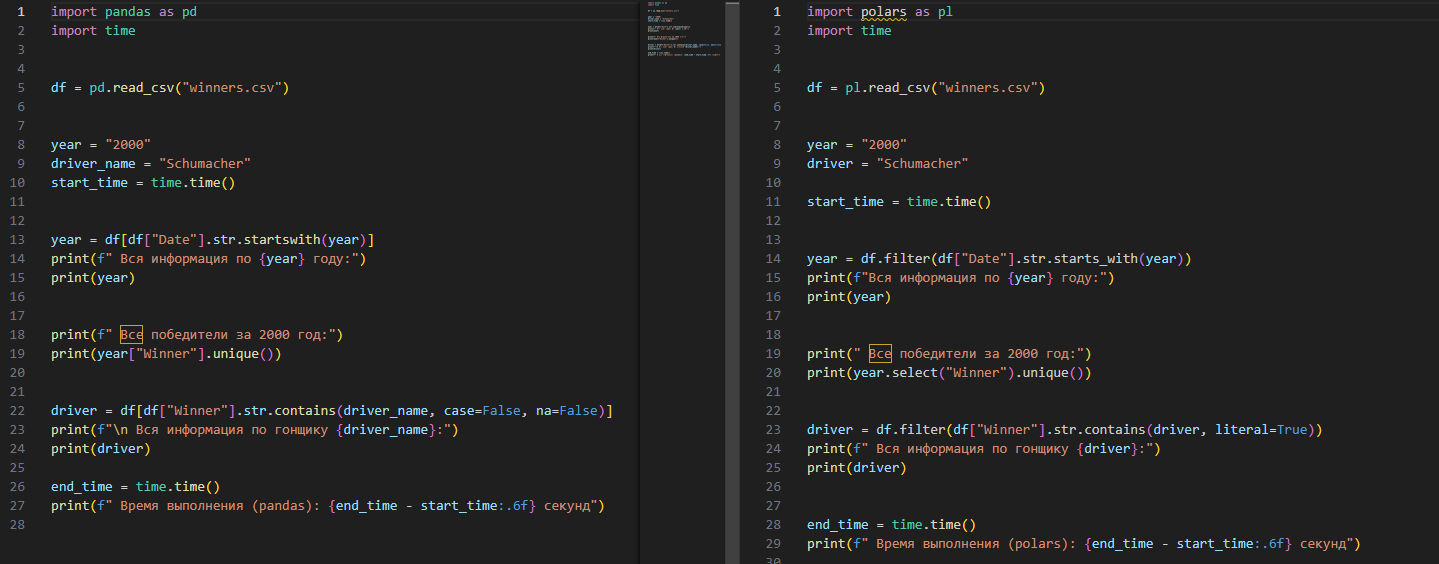
## 2.3 Реализация с использованием библиотеки Polars

…

Комментарии:

* Синтаксис Polars ориентирован на выражения (pl.col(...)), что позволяет формировать цепочки фильтраций и трансформаций.
* Поддержка многопоточности и оптимизированное использование памяти ускоряют выполнение операций.
* Благодаря колонно-ориентированной архитектуре Polars обрабатывает только те столбцы, которые необходимы в запросе.

## 2.4 Сравнительный анализ результатов

****

Результаты выполнения обеих реализаций показали следующее:

* Обе библиотеки корректно выполняют фильтрацию и выборку данных по заданным условиям.
* Визуальный результат (набор строк) идентичен, что подтверждает эквивалентность логики.
* Время выполнения операций в Polars оказалось заметно ниже, особенно при увеличении объёма строк.
* Использование Polars позволяет снизить нагрузку на память, поскольку операции исполняются в колонно-ориентированном формате и оптимизируются заранее.

## 2.5 Выводы по главе 2

Сравнительный эксперимент подтвердил теоретические преимущества Polars в задачах, связанных с фильтрацией и обработкой больших наборов данных. Несмотря на схожесть синтаксиса, подход к исполнению в Pandas и Polars отличается коренным образом: первый — это последовательная обработка в памяти, второй — оптимизированный многопоточный расчёт.

Ключевые итоги:

* **Pandas** остаётся удобным и универсальным инструментом для анализа данных малого и среднего объёма, особенно на этапах прототипирования и визуализации.
* **Polars** демонстрирует более высокую скорость выполнения и лучше масштабируется, что делает её актуальной в условиях реального промышленного использования.
* Обе библиотеки заслуживают внимания и могут применяться в зависимости от особенностей задачи и ограничений вычислительной среды.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы проведено комплексное сравнение двух ключевых инструментов анализа табличных данных в языке программирования Python — Pandas и Polars. Исследование включало изучение архитектурных и функциональных особенностей, практическую реализацию типовой задачи фильтрации, а также оценку производительности на основании объективных замеров.

Pandas зарекомендовала себя как удобный и надёжный инструмент с широкой экосистемой и поддержкой, подходящий для анализа небольших и средних объёмов информации. Её преимущество — в доступности, простоте освоения и тесной интеграции с другими библиотеками экосистемы Python.

Polars, в свою очередь, проявила себя как высокопроизводительное и эффективное решение, особенно при работе с крупными массивами данных. За счёт многопоточности, ленивой модели вычислений и колонно-ориентированного формата хранения библиотека обеспечивает значительное преимущество по скорости и экономии ресурсов.

Таким образом, исследование подтвердило необходимость осознанного подхода к выбору инструментов аналитики. Ни одна из библиотек не является универсально лучшей — они решают разные классы задач и дополняют друг друга. Умение сочетать удобство Pandas и производительность Polars — важный навык современного аналитика.

# Список литературы

1. Pandas documentation. Официальный сайт [Электронный ресурс]. – Режим доступа: https://pandas.pydata.org/docs/, свободный. – (Дата обращения: 19.06.2025).
2. Polars User Guide: The Polars Book. Официальная документация [Электронный ресурс]. – Режим доступа: https://pola-rs.github.io/polars-book/, свободный. – (Дата обращения: 19.06.2025).
3. Burchell J. Polars vs. pandas: What’s the Difference? / JetBrains PyCharm Blog [Электронный ресурс]. – Опубликовано: 04.07.2024. – URL: https://blog.jetbrains.com/pycharm/2024/07/polars-vs-pandas/ – (Дата обращения: 19.06.2025).
4. Trivedi A. Pandas vs Polars: A Comprehensive Comparison / Analytics Vidhya [Электронный ресурс]. – Опубликовано: 06.08.2024. – URL: https://www.analyticsvidhya.com/blog/2024/08/pandas-vs-polars/ – (Дата обращения: 19.06.2025).
5. ML Journey. Polars vs Pandas Performance Comparison [Электронный ресурс]. – Опубликовано: 19.03.2025. – URL: https://mljourney.com/polars-vs-pandas-performance-comparison/ – (Дата обращения: 19.06.2025).
6. Statology. Pandas vs Polars: Performance Benchmarks for Common Data Operations [Электронный ресурс]. – Опубликовано: 04.11.2024. – URL: https://www.statology.org/pandas-vs-polars-performance-benchmarks-for-common-data-operations/ – (Дата обращения: 19.06.2025).
7. Markaicode. Pandas vs Polars: DataFrame Library Speed Comparison for Large Datasets [Электронный ресурс]. – Опубликовано: 27.05.2025. – URL: https://markaicode.com/pandas-vs-polars-speed-comparison-large-datasets/ – (Дата обращения: 19.06.2025).
8. GeeksforGeeks. Introduction to Pandas in Python [Электронный ресурс]. – URL: https://www.geeksforgeeks.org/pandas/introduction-to-pandas-in-python/ – (Дата обращения: 19.06.2025).
9. GeeksforGeeks. Mastering Polars: High-Efficiency Data Analysis [Электронный ресурс]. – URL: https://www.geeksforgeeks.org/mastering-polars-high-efficiency-data-analysis-and-manipulation/ – (Дата обращения: 19.06.2025).
10. GitHub. Polars: Fast DataFrames in Rust and Python [Электронный ресурс]. – URL: https://github.com/pola-rs/polars/ – (Дата обращения: 19.06.2025).
11. Studytonight. Features and Advantages of Pandas [Электронный ресурс]. – URL: https://www.studytonight.com/pandas/features-of-pandas – (Дата обращения: 19.06.2025).