**АНОТАЦИЯ**

СОДЕРЖАНИЕ

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ 3](#_Toc480997628)

[ВВЕДЕНИЕ 4](#_Toc480997629)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc480997630)

[1.1. Определение проекта 5](#_Toc480997631)

[1.2. Жизненный цикл проекта 7](#_Toc480997632)

[1.3. Графовые базы данных 9](#_Toc480997633)

[1.3.1. Что такое граф? 10](#_Toc480997634)

[1.3.2. Графовая СУБД 11](#_Toc480997635)

[1.3.3. Механизмы вычисления графов 13](#_Toc480997636)

[1.3.4. Преимущества графовых баз данных 15](#_Toc480997637)

[1.4. Сравнение реляционных и графовых БД. SQL и NoSQL подход 16](#_Toc480997638)

# ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

ВКР – выпускная квалификационная работа

ЖЦП – жизненный цикл проекта

ЖЦ – жизненный цикл

ТЭО – технико-экономическое обоснование

RDF – Resource Description Framework – среда описания ресурса

БД – база данных

СУБД – система управления базами данных

OLTP – Online Transaction Processing – обработка транзакций в реальном времени

MapReduce – модель распределенных вычислений

NoSQL – Not only SQL – не только SQL

SQL – Structured Query Language – язык структурированных запросов

OLAP – Online Analytical Processing – интерактивная аналитическая обработка

SOR – System of Record – система записи

ПО – программное обеспечение

# ВВЕДЕНИЕ

# АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## Определение проекта

В современной литературе по управлению проектами можно выделить два основополагающих подхода к определению проекта: системный подход и деятельностный.

Системный подход характеризует проект как систему временных действий, направленных на достижение неповторимого, но в то же время конкретного результата.

Системный подход к определению проекта предопределяет основные его характеристики:

* разовость – все проекты представляют собой разовое явление. Они приходят и уходят, появляются и исчезают, оставляя после себя конкретные результаты, существенно отличаясь от наших повседневных обязанностей и деятельности,
* уникальность – нет двух идентичных проектов. Каждый из них, независимо от его результатов, в своей основе имеет что-то уникальное, характерное только для него,
* инновационность – в процессе реализации проекта всегда создается нечто новое. Изменения могут быть колоссальными или менее значительными,
* результативность – все проекты имеют вполне очевидные результаты. Это может быть новый дом, напечатанная книга, модифицированная структура предприятия, рабочее приложение. Все проекты нацелены на получение конкретных результатов, другими словами, они направлены на достижение целей,
* временная локализация – все проекты ограничены временными рамками. Проект – это создание чего-либо к назначенному сроку, он имеет планируемую дату завершения, после которой команда проектантов распускается.

Все перечисленные характеристики взаимосвязаны и задают определенные границы проекта, три его измерения, критерия, по которым можно оценить проект (рис.1).

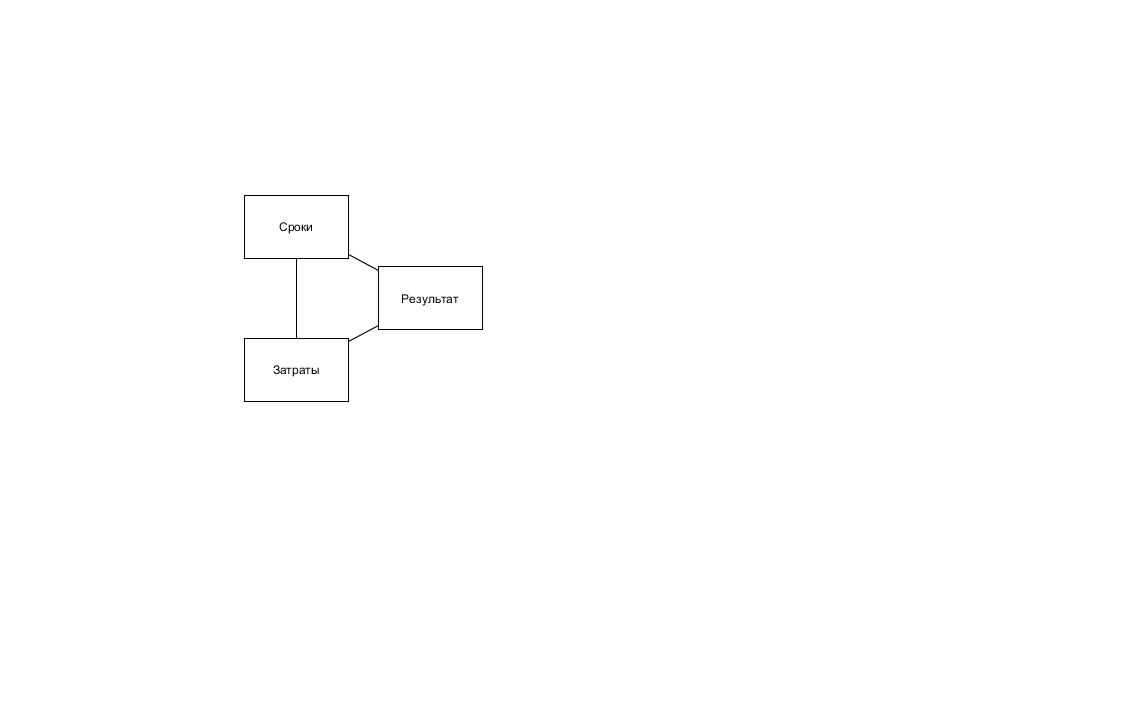


Рисунок 1. Схема измерения проекта

Планирование и реализация проекта всегда связаны с тремя главными вопросами:

* сколько времени это займет,
* сколько это будет стоить,
* совпадет ли конечный результат с тем, который планировался вначале.

Первый вопрос выводит на первый план проблему временных ограничений, установленных для реализации всего проекта и отдельных его этапов. Второй вопрос привлекает наше внимание к стоимости проекта, третий касается вопроса о результативности проектной деятельности.

Универсальность и многоаспектность проектной технологии определены разноуровневыми многослойными взаимодействиями и измерениями проекта. Измерения проекта – цели, время, стоимость – являются одновременно ограничениями проекта, задающими систему координат, в которой вынужден работать проект-менеджер. Сверхзадача проект-менеджера – найти оптимальное соотношение этих трех ограничений проекта, с которыми неразрывно связаны интересы участников проекта.

Второй подход – деятельностный – определяет проект как деятельность субъекта по переводу объекта из наличного состояния в состояние желаемого будущего, которое наиболее полно отвечает его представлениям, ожиданиям. [1]

Таким образом, проект в самом широком смысле может пониматься как творческая, разумная, целеполагающая деятельность субъекта. Проект - это то, что задумывается и после этого планируется, имеет определенные системы подходов с четко установленными временными рамками, капиталом и необходимым результатом. Каждый проект сам по себе уникален во всех, или практически во всех его аспектах.

## Жизненный цикл проекта

Для того что бы понять природу работы над проектом, необходимо описать его жизненный цикл, смысл которого вполне понятен. Проект имеет свое начало и свой конец, а также периоды роста, стабильности и спада. В простейшем случае ЖЦП включает в себя начальный, промежуточный и заключительный этапы, фазы.[2]

Фазы, этапы жизненного цикла проекта включают в себя выполнение основных мероприятий по проекту:

* разработка ТЭО и рабочего проекта,
* контрактная деятельность,
* организация и финансирование работ,
* создание новых технологий,
* планирование ресурсов и хода работ,
* закупка материалов и оборудования,
* выполнение работ и сдача готовых объектов.

Таким образом, исходя из основных мероприятий по проекту, строятся фазы ЖЦП (рис. 2).



Рисунок 2. Основные фазы жизненного цикла проекта

Фазы проекта – это отдельные части в рамках проекта, требующие дополнительного контроля для эффективного получения основного результата проекта, они обычно выполняются последовательно, но в некоторых проектных ситуациях могут перекрываться. Высокоуровневый характер фаз проекта превращает их в элемент жизненного цикла проекта. Фаза проекта не является группой процесса управления проектом.

Структура фаз позволяет разделить проект на логические подгруппы для более легкого управления, планирования и контроля. Количество фаз, необходимость в них и степень налагаемого контроля зависит от размера фаз, сложности и потенциального влияния на проект.[3]

Один из важных моментов, характеризующий ЖЦП, является нарастание трудозатрат по фазам жизненного цикла. Типичный вид трудозатрат показан на рисунке 3, будем считать, что на каждую фазу дается одинаковое количество времени.[4]

Рисунок 3. Диаграмма распределение трудозатрат по фазам ЖЦ

Проект часто начинается с идеи, которая появляется у одного человека. Постепенно, по мере формулирования, анализа и оценки этой идеи, привлекаются дополнительные специалисты. Еще больше участников требуется на фазе разработки проекта. Пик трудозатрат приходится на фазу реализации проекта.

На последнем этапе происходит постепенное высвобождение участников проектной команды. Следует помнить, что проект должен иметь четкое окончание во времени, после которого все работы по проекту закрываются, и на проект перестают тратиться ресурсы.[5]

## Графовые базы данных

Графовая база данных – NoSQL решение, разновидность баз данных с реализацией сетевой модели в виде графа и его обобщений с использованием графовой СУБД, которая дает новые возможности для работы со связанными данными. Графовую модель данных обычно рассматривают как обобщение RDF – модели или сетевой модели данных.[6]

### Что такое граф?

Граф – совокупность точек, соединенных линиями. Точки называются вершинами, или узлами, а линии – ребрами, или дугами.

Степенью входа вершины является количество входящих в нее ребер, степенью выхода – количество исходящих ребер.

Граф, содержащий ребра между всеми парами вершин, является полным. Встречаются такие графы, ребрам которых поставлено в соответствие конкретное числовое значение, они называются взвешенными графами, а эти значения – весом ребра. Когда у ребра оба конца совпадают, то есть оно выходит из вершины и входит в нее, то такое ребро называется петлей (рис.4).[7]

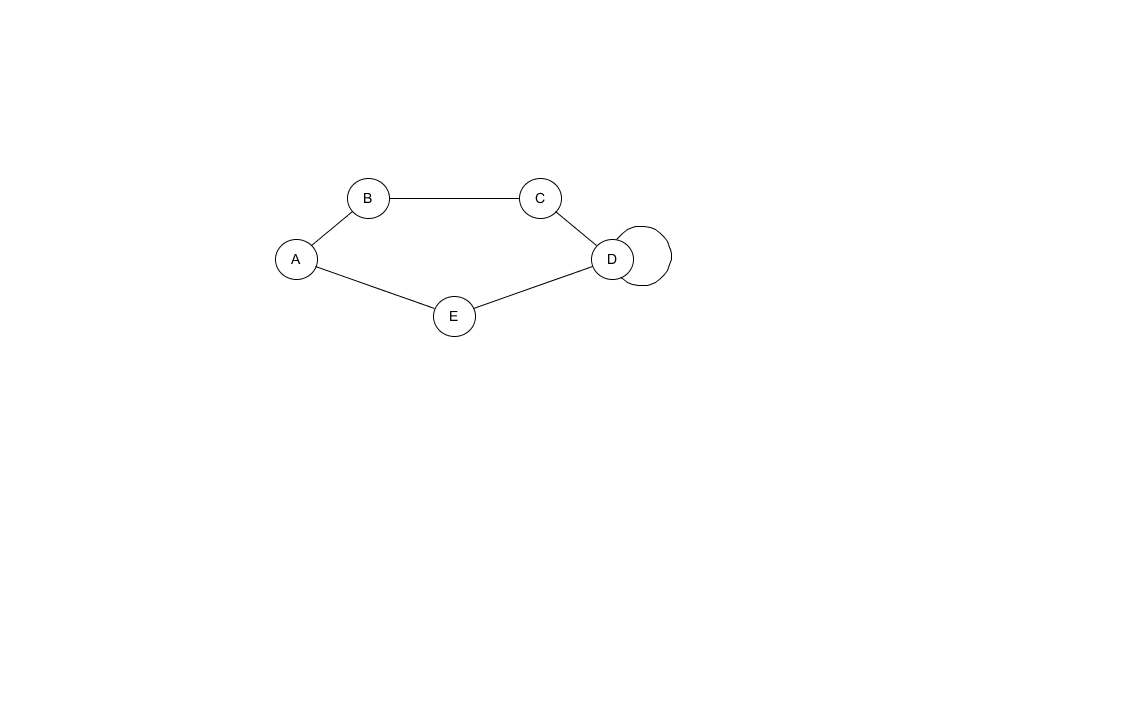


Рисунок 4. Пример простейшего графа с петлей

Также, граф имеют свою классификацию:

* связный – между любой парой вершин существует как минимум один путь,
* несвязный – существует хотя бы одна вершина, не связанная с другими,
* ориентированный – ребра графа являются направленными, то есть существует только одно доступное направление между двумя связанными вершинами,
* неориентированный – по каждому из ребер можно осуществлять переход в обоих направлениях,
* смешанный – характерен наличием как ориентированных, так и неориентированный ребер.[8]

Граф – это универсальная и выразительная структура, которая позволяет моделировать всевозможные сценарии, от постройки атомной станции до строительства системы дорог, от поставок продуктов питания до историй болезни населения, он чрезвычайно полезен при анализе самых разнообразных наборов данных в таких областях, как наука, государственное управление и бизнес.

### Графовая СУБД

Система управления графовыми базами данных (графовые базы данных) поддерживает методы создания, чтения, изменения, и удаления, основанные на графовой модели данных. Графовые базы данных, как правило, поддерживают систему транзакций реального времени (OLTP). Соответственно, они оптимизированы для выполнения транзакций и спроектированы с учетом транзакционной целостности и оперативности.

Имеются две особенности графовых баз данных, которые необходимо учитывать при рассмотрении применяемой ими технологии:

1. принцип хранения. Некоторые графовые базы данных используют специализированные хранилища графов, предназначенные и оптимизированные для хранения и обработки именно графов. Но такую технологию хранения используют не все графовые базы данных. Некоторые сериализуют графы и размещают их в реляционной, объектно-ориентированной или какой-то другой базе данных или хранилище;
2. порядок обработки. Некоторые определения требуют, чтобы графовая база данных использовала смежность без индексов, то есть физическое соединение друг с другом.

Взаимосвязи в графовой модели данных являются гражданами первого сорта. Здесь к ним относятся не так, как в других системах управления базами данных, где для отображения взаимосвязей применяются такие механизмы, как внешние ключи или внешние операции, например MapReduce. Собирая абстракции узлов и взаимосвязей в связанные структуры, графовая база данных позволяет строить модели любой сложности, лучше всего отражающие предметную область. Полученные модели проще и в то же время нагляднее, чем те, что создаются с помощью традиционных реляционных баз данных или других NOSQL-хранилищ.[9]

Самые популярные на сегодняшний день графовые СУБД:

* Neo4j,
* HyperGraphDB,
* ArangoDB,
* FlockDB,
* Giraph,
* OrientDB,
* Infinite Graph.

Фундаментальная модель данных графовых баз очень простая: узлы, соединенные ребрами (дугами). Помимо этой существенной характеристики, существуют много вариаций в моделях данных – в частности, в том, какие механизмы используются для хранения вершин и ребер. Например, база FlockDB, представляет собой простую совокупность узлов и ребер без какого-либо механизма для дополнительных атрибутов, Neo4j позволяет присоединять Java – объекты в качестве свойств узлов и ребер в неструктурированном виде, а Infinite Graph хранит Java-объекты, являющиеся экземплярами подклассов таких встроенных типов, как узлы и ребра.

На рисунке 5 показан пример модели графовой базы данных с маленькими узлами и многочисленными связями между ними.



Рисунок 5. Пример модели графовой базы данных

Работая с этой моделью, мы имеем возможность задавать вопросы вроде “найти книгу в категории “Базы данных”, написанную кем-то, чей друг мне нравится”.[10]

### Механизмы вычисления графов

Такие механизмы позволяют выполнять глобальные графовые вычислительные алгоритмы для больших наборов данных. Они предназначены для решения таких задач, как идентификация кластеров данных или получение ответов на такие вопросы, как: “Сколько всего взаимосвязей, сколько их в среднем, полна ли социальная сеть?”

Из-за своей направленности на глобальные запросы механизмы вычисления графов, как правило, оптимизированы для сканирования и пакетной обработки больших объёмов информации, и в этом отношении они похожи на другие технологии пакетного анализа, такие как интеллектуальный анализ данных или аналитическая обработка в реальном времени (OLAP), используемые в реляционном мире.[11]

Некоторые механизмы вычисления включают в себя и средства хранения графов, а другие (большинство) заботятся только об обработке данных, получаемых из внешнего источника, а затем возвращают результаты для сохранения в другом месте. Рисунок 6 демонстрирует типовую архитектуру развертывания механизмов вычисления графов.



Рисунок 6. Типовая архитектура развертывания механизмов вычисления графов

Схема включает в себя систему записи (SOR) базы данных со свойствами OLTP (к примеру, Oracle или Neo4j), которая обслуживает запросы и отвечает на запросы, поступающие от приложения и, в конечном счете, от пользователей. Периодический задания на извлечения, преобразования и загрузку данных перемещают данные из системы записи базы данных в механизм вычисления графов для выполнения анализа и автономных запросов.[12]

### Преимущества графовых баз данных

Практический любою деятельность, будь то план маршрута похода в магазин или же разработку атомной электростанции можно представить в виде графа но, к сожалению, мы живем в мире, где правят жесткие правила и стандарты, ограниченные бюджеты, установленные сроки для выполнения задач или реализацию проектов. Предоставляемый графовыми базами данных новый способ моделирования данных сам по себе не дает достаточного основания для замены давно устоявшихся и понятных платформ обработки данных. Этот способ должен давать незамедлительную и очень значительную практическую пользу. Мотивация перехода на графовые базы данных заключается в том, что при определенной модели данных такой переход будет давать существенное увеличение производительности на один и более порядков.

Ощутимый прирост производительности при использовании графовых баз данных достигается, если работа ведется с взаимосвязанными данными, по сравнению с теми же реляционными базами данных или NoSQL-хранилищами. В отличие от реляционных БД, где учет взаимосвязей большого объёма данных ощутимо ухудшает производительность запросов, производительность графовых БД при росте объёма данных остается неизменной. Это связана с тем фактом, что запросы в графовой БД локализуются в определенной части графа. В итоге время выполнения каждого запроса зависит только от размера части графа, в которой происходит поиск, а не от общего его размера.

По мимо роста в производительности, графовые базы данных предоставляют очень гибкую модель данных и способ развертывания, который соответствует современным способам развертывания ПО. Структура данных должна соответствовать изменяющимся потребностям, а не навязываться заранее и оставаться неизменной. В графовых БД эта задача легко решается, графовая модель данных учитывает потребности бизнеса, что и дает ей возможность изменяться со скоростью изменения самого бизнеса.

Возможность расширения означает, что можно добавлять и дополнять новые виды взаимосвязей, новые узлы, новые метки, а также новые подграфы в существующую структуру и что самое приятное – это происходит без нарушения существующих запросов и функционала приложения. Такая возможность положительно влияет на производительность процесса разработки и снижает риски для проекта. Благодаря гибкости графовой БД, нет необходимости заранее моделировать задачу в мельчайших подробностях, что очень неудобно, поскольку бизнес-требования очень быстро меняются. Способность графов к расширению также позволяет уменьшить количество миграций, это снижает нагрузку при обслуживании данных и уменьшает риск потерь.[9, 10]

## Сравнение реляционных и графовых БД. SQL и NoSQL подход