

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)»
Московский техникум космического приборостроения

СПЕЦИАЛЬНОСТЬ: 09.02.07 Информационные системы и программирование
(квалификация «Администратор баз данных»)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по теме:

Разработка (подготовка) документации и отчётных форм для
внедрения программных средств платформы Amazon Web Services
интеграции распределённых приложений, миграции и передачи
данных

Руководитель разработки
от техникума

(подпись, дата)

АБВГДЕЖЗИ К
(Ф.И.О.)

Разработчик

(подпись, дата)

МАБВГД ЕЖЗИК
(Ф.И.О.)

Москва 20__

**«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
Московский техникум космического приборостроения**

УТВЕРЖДАЮ
Председатель ПЦК спец. 09.02.07
Н.А. Жилкина

«___» _____ 20__ г.

ЗАДАНИЕ

на выполнение курсового проекта

по профессиональному модулю «ПМ.04 Сопровождение и обслуживание программного
обеспечения компьютерных систем»

Студент НГААГНАНГАН ТМП-70
(фамилия, инициалы, индекс группы)

Руководитель СЕНКНЕОРПНГПИШ
(фамилия, инициалы)

График выполнения работы: 25% к 4 нед., 50% к 8 нед., 75% к 12 нед., 100% к 15 нед.

1 Тема курсового проекта

Разработка (подготовка) документации и отчётных форм для внедрения программных средств платформы Amazon Web Services интеграции распределённых приложений, миграции и передачи данных

2 Техническое задание

Разработать (подготовить) документацию и отчётные формы для внедрения программных средств платформы Amazon Web Services интеграции распределённых приложений, миграции и передачи данных

3 Оформление курсового проекта

3.1 Расчетно-пояснительная записка на ___ листах формата А4.

3.2 Перечень графического материала КП (плакаты, схемы, чертежи и т.п.) – схемы алгоритма

Дата выдачи задания «___» _____ 20__ г.

Руководитель курсового проекта _____ РОАОГШЕНП

СОДЕРЖАНИЕ

Введение.....	4
1. Постановка задачи.....	5
2. Теоретические основы	5
2.1 Облачные вычисления.....	5
2.2 Интеграция приложений в Amazon Web Services.....	5
2.2.1 Сервисы интеграции приложений Amazon Web Services.....	5
2.2.2 Примеры использования интеграции приложений	5
2.2.3 Перемещение эксабайтов данных с AWS Snowmobile	5
2.3 Миграция данных с помощью Amazon Web Services	5
2.3.1 Решения Amazon Web Services для миграции	5
2.4 Передача данных в Amazon Web Services	5
2.4.1 Перемещение эксабайтов данных с AWS Snowmobile	5
2.4.2 Работа с потоковыми данными в Amazon Kinesis Streams	5
3. Практическая реализация	5
3.2 Создание таблицы NoSQL и выполнение запросов к ней	5
3.3 Настройка документной базы данных	5
3.4 Хранение и извлечение файла с помощью Amazon S3	5
4. Тестирование.....	5
4.1 План тестирования.....	5
4.2 Тест-кейсы	5
4.3 Баг-репорты	5
4.4 Отчёт о тестировании	5
5. Результаты.....	5
5.1 Создание бесплатного аккаунта Amazon Web Services	5
5.2 Результаты создания таблицы NoSQL и выполнения запросов к ней.....	5
5.3 Результаты настройки документной базы данных	5
5.4 Результаты хранения и извлечения файла с помощью Amazon S3	5
5.5 Руководство пользователя	5
Заключение.....	5
Приложение А. Руководство пользователя по созданию таблицы NoSQL и выполнения запросов к ней.....	5

Приложение Б. Руководство пользователя по настройке документной базы данных	5
Приложение В. Руководство пользователя по хранению. и извлечению файла с помощью Amazon S3	5
Приложение Г. Отчёт о тестировании.....	5

ВВЕДЕНИЕ

В современных условиях (на текущем этапе развития науки и техники облачные вычисления предоставляют вычислительные службы (в том числе серверы, хранилища, базы данных, сети, программное обеспечение, инструменты аналитики и интеллектуального анализа) через Интернет («облако»). Такие службы ускоряют внедрение инноваций, повышают гибкость ресурсов и обеспечивают экономию благодаря высокой масштабируемости. Вы обычно платите только за облачные службы, которые позволяют сократить эксплуатационные расходы, а также повысить эффективность управления инфраструктурой и масштабирования по мере изменения потребностей бизнеса.

Amazon Web Services – самая крупная публичная облачная платформа в мире, обладающая широчайшим портфелем решений и сервисов, от построения отказоустойчивых инфраструктур и платформ для разработки, до сервисов машинного обучения, интернета вещей, аналитики больших данных и многого другого. В 2006 году Amazon Web Services (AWS) начали предлагать ИТ-услуги на рынке в виде веб-сервисов, которые сегодня известны как облачные вычисления. В этом облаке нам не нужно планировать серверы и другую ИТ-инфраструктуру, которая занимает много времени заранее. Вместо этого эти сервисы могут мгновенно раскручивать сотни или тысячи серверов за считанные минуты и быстрее доставлять результаты. Сегодня AWS предоставляет высоконадежную, масштабируемую и недорогую инфраструктурную платформу в облаке, которая обслуживает множество предприятий в 190 странах мира.

Amazon Web Services (AWS) – это самая распространенная в мире облачная платформа с широчайшими возможностями, предоставляющая более 175 полнофункциональных сервисов для центров обработки данных по всей планете. Миллионы клиентов, в том числе стартапы, ставшие лидерами по скорости роста, крупнейшие корпорации и передовые правительственные учреждения, используют AWS для снижения затрат, повышения гибкости и ускоренного внедрения инноваций.

1. ПОСТАНОВКА ЗАДАЧИ

Целью курсового проекта является разработка (подготовка) документации и отчётных форм для внедрения программных средств платформы Amazon Web Services интеграции распределённых приложений, миграции и передачи данных.

Характер разработки: прикладная квалификационная работа.

Основания для разработки: учебный план специальности, рабочая программа дисциплины, распоряжение по учебному заведению.

Требования к курсовому проекту: требования к проектируемой системе определены соответствующими стандартами ГОСТ 19 и ГОСТ 34; требования к структуре документов определены соответствующими стандартами ЕСКД и КСПД; требования к оформлению определены соответствующими методическими указаниями.

Порядок контроля и приёмки курсового проекта: контроль выполнения курсового проекта проводится руководителем поэтапно в соответствии с утвержденным графиком выполнения курсового проекта; на завершающем этапе руководитель осуществляет нормоконтроль представленной исполнителем документации и принимает решение о допуске (недопуске) курсового проекта к защите, при подготовке защиты курсового проекта исполнитель представляет документацию, делает краткое сообщение по теме разработки и демонстрирует реализацию.

При работе над курсовым проектом учитывать: степень соответствия представленной разработки требованиям технического задания, качество реализации, документации и доклада по теме проекта, соблюдение исполнителем графика выполнения курсового проекта.

Плановые сроки выполнения: __ семестр 20__ / __ учебного года:

Исходные данные:

- в курсовом проекте необходимо выполнить следующие этапы разработки, которые должны быть отражены в пояснительной записке: изучение необходимых теоретических сведений в соответствии с заданием, выявление действующих субъектов системы;
- курсовой проект основывается на фактических материалах, полученных в ходе учебной или практической профессиональной деятельности, с использованием результатов последних исследований, а также на результатах собственных исследований и экспертной оценки.
- в случае отсутствия практического материала по конкретной теме допускается смоделировать его самостоятельно с учётом соблюдения определённых пропорций между сферы деятельности.

Состав курсового проекта:

- Текстовые и графические документы;
- Программная и технологическая документация (при необходимости);
- Пояснительная (расчётно-пояснительная) записка;
- Обязательные структурные элементы: титульный лист (одна страница), содержание (от одной страницы), введение (от одной страницы), список литературы (от одной страницы), заключение (от одной страницы),
- Объём основной части работы: минимум два раздела с минимум двумя параграфами (от пяти страниц) в каждом разделе.

Содержание пояснительной записки:

- Титульный лист;
- Лист задания;
- Содержание;
- Введение;
- Исходные данные;
- Основной раздел (теоретическая, аналитическая и проектно-аналитическая часть);
- Заключение;
- Список использованной литературы;
- Приложения.

Детализированное техническое задание на курсовой проект:

- во введении обосновать актуальность исследуемой темы, степень её изученности, дать критический обзор работ, характеризующих сущность рассматриваемого вопроса, определить цель и задачи, описать применяемые методы исследования;
- в теоретической части изложить теоретические концептуальные основы исследуемой проблемы и собственное понимание существа вопросов исследуемой темы, сформировать своё аргументированное мнение по неоднозначным вопросам;
- в аналитической части изложить особенности функционирования объекта исследования, проанализировать и раскрыть особенности сбора, обработки и подготовки информации о процессах, целесообразно рассмотреть порядок формирования и представления информации об отдельных объектах исследования;
- в проектно-аналитической части разработать и внести предложения по рациональной методологии исследования объекта с использованием современных информационных технологий, доказать целесообразность и необходимость предлагаемых решений, спрогнозировать результаты от внедрения предложений;

- в заключении изложить основные результаты исследования, сформулировать выводы, предложения и перспективы дальнейшей разработки темы.

2. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ

2.1 Облачные вычисления

Под облачными вычислениями понимается доставка ИТ-ресурсов по требованию через Интернет с оплатой по факту использования. Покупать, размещать и обслуживать физические ЦОД и серверы не требуется. Вместо этого вы получаете доступ к технологическим сервисам: вычислительным сервисам, хранилищам и базам данных, которыми можно пользоваться по мере необходимости благодаря поставщику услуг (например, Amazon Web Services).

Облачные вычисления нужны организациям вне зависимости от типа, размера и отрасли. Облако можно использовать для самых разных целей, включая резервное копирование данных, аварийное восстановление, разработку и тестирование ПО, анализ больших данных, для систем электронной почты, виртуальных рабочих столов, а также интернет-приложений, ориентированных на клиентов. Так, компании в сфере здравоохранения используют облако, чтобы сделать разработку планов лечения для пациентов более индивидуальной. Компании в сфере финансовых услуг применяют облако в качестве основы для систем обнаружения и предотвращения случаев мошенничества в режиме реального времени. Разработчикам видеоигр облако необходимо, чтобы иметь возможность предоставлять онлайн-игры пользователям по всему миру.

Преимущества облачных вычислений

Гибкость. С облаком вы просто получаете доступ к множеству технологических возможностей, что позволяет быстрее внедрять инновации и реализовывать практически любые идеи. Можно быстро выполнить развертывание необходимых ресурсов для самых разных областей использования, от сервисов инфраструктуры (например, вычислительных сервисов, сервисов хранилищ и баз данных) до Интернета вещей (IoT), машинного обучения, озер данных, систем анализа и т. д. Развертывание технологических сервисов происходит за считанные минуты, и переход от идеи к ее воплощению теперь занимает гораздо меньше времени. Можно больше экспериментировать, смелее пробовать новые идеи для повышения качества обслуживания клиентов, а также увереннее трансформировать собственный бизнес.

Эластичность. С облачными вычислениями не обязательно выделять ресурсы заранее и в избыточном количестве на случай всплесков активности бизнеса. Вместо этого можно выделить столько ресурсов, сколько требуется в данный момент времени. Ресурсы в облаке

можно масштабировать в соответствии с потребностями бизнеса путем увеличения или уменьшения объема.

Сокращение затрат. С помощью облака вы можете превратить капитальные затраты (такие как ЦОД и физические серверы) в переменные расходы и платить только за используемые ИТ-ресурсы. Кроме того, переменные расходы становятся значительно ниже благодаря существенной экономии при увеличении масштабов работы.

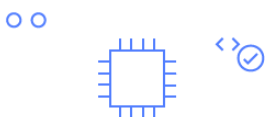
Глобальное развертывание за считанные минуты. С облаком можно расширить присутствие бизнеса в новые географические регионы и выполнить глобальное развертывание за считанные минуты. К примеру, глобальная инфраструктура AWS позволяет выполнить развертывание приложения в нескольких физических местоположениях всего за несколько щелчков мышью. Размещение приложений ближе к конечным пользователям позволяет повысить удобство работы и снизить задержку.

Типы облачных вычислений

Три основных модели облачных вычислений включают IaaS (инфраструктура как сервис), PaaS (платформа как сервис) и SaaS (программное обеспечение как сервис). Каждая модель предоставляет определенный уровень контроля, гибкости и возможностей управления: можно выбрать набор сервисов, который нужен именно вам.



Инфраструктура как сервис (IaaS). Модель IaaS включает в себя основные компоненты облачных ИТ. Она предоставляет доступ к сетевым возможностям, компьютерам (виртуальное или выделенное оборудование) и определенному объему хранилища. IaaS предполагает максимальный уровень гибкости и широчайшие возможности контроля ИТ-ресурсов. Эта модель более других похожа на существующие ИТ-ресурсы, привычные для большинства ИТ-отделов и разработчиков.



Платформа как сервис (PaaS). Модель PaaS не требует управления базовой инфраструктурой (чаще всего оборудованием и ОС) и позволяет сосредоточиться на развертывании приложений и управлении ими. Это повышает производительность работы, ведь больше не приходится беспокоиться о приобретении материально-технических ресурсов,

заниматься планированием мощности, обслуживанием ПО, установкой обновлений безопасности и выполнять другие трудоемкие задачи, необходимые для работы приложений.



Программное обеспечение как сервис (SaaS). Модель SaaS предоставляет собой готовый продукт, который запускается и управляется поставщиком сервиса. Чаще всего под SaaS-решениями понимают приложения для конечных пользователей (такие как веб-сайты электронной почты). При выборе этой модели не нужно беспокоиться о том, как происходит обслуживание сервиса и управление базовой инфраструктурой. Решить требуется лишь вопрос, как именно будет использоваться то или иное ПО.

Облачные сервисы

AWS предлагает больше сервисов (а также возможностей внутри этих сервисов), чем другие поставщики облачных решений, включая вычислительные сервисы, сервисы хранилищ, баз данных, сетевых конфигураций, озер данных, аналитики, машинного обучения, искусственного интеллекта, Интернета вещей (IoT) и т. д.

Облачные решения

AWS предоставляет объемное портфолио решений, которое помогает решать стандартные проблемы и быстрее разрабатывать свои продукты, используя платформу AWS. Каждое из решений AWS поставляется с подробной схемой архитектуры, руководством по развертыванию и инструкциями как для автоматического, так и для ручного развертывания.

Цены

AWS использует модель оплаты по факту использования.

Цены индивидуальны для каждого сервиса.

Продукты

AWS предлагает на выбор более 175 полнофункциональных сервисов, предназначенных для широкого спектра технологий, отраслей и сценариев использования.

2.2 Интеграция приложений в Amazon Web Services

Набор сервисов интеграции приложений в AWS обеспечивает взаимодействие между изолированными компонентами в микросервисах, распределенных системах и бессерверных приложениях. Нет необходимости выполнять рефакторинг всей архитектуры – изолирование приложений в любых масштабах позволяет снизить воздействие вносимых изменений, упрощая процесс обновления и ускоряя выпуск новых возможностей.

Использование сервисов интеграции приложений для объединения приложений избавляет от необходимости написания пользовательского кода для обеспечения их взаимодействия. Это позволяет уменьшить количество лишнего кода, который может повторяться в микросервисах и функциях.

Благодаря сервисам интеграции приложений вы можете переложить на AWS часть эксплуатационной ответственности и сосредоточиться на разработках и инновациях. Благодаря автоматическому масштабированию больше нет необходимости выделения серверов, управления ими и внесения исправлений.

Изолирование приложения с помощью сервисов интеграции приложений позволяет сохранить взаимодействие приложений, при этом сбой одного из сервисов или пиковая рабочая нагрузка никак не повлияют на работу других приложений.

Сервисы передачи сообщений с возможностью интеграции приложений могут адаптироваться к любой пропускной способности и использовать хранилища сообщений в зонах перекрестной доступности для обеспечения высокого уровня доступности и надежности.

Интеграция распределенных систем и бессерверных приложений с меньшим количеством кода

Набор сервисов интеграции приложений в AWS обеспечивает взаимодействие между изолированными компонентами в микросервисах, распределенных системах и бессерверных приложениях. Нет необходимости выполнять рефакторинг всей архитектуры – изолирование приложений в любых масштабах позволяет снизить воздействие вносимых изменений, упрощая процесс обновления и ускоряя выпуск новых возможностей.

Использование сервисов интеграции приложений для объединения приложений избавляет от необходимости написания пользовательского кода для обеспечения их взаимодействия. Это позволяет уменьшить количество лишнего кода, который может повторяться в микросервисах и функциях.

Благодаря сервисам интеграции приложений вы можете переложить на AWS часть эксплуатационной ответственности и сосредоточиться на разработках и инновациях.

Благодаря автоматическому масштабированию больше нет необходимости выделения серверов, управления ими и внесения исправлений.

Изолирование приложения с помощью сервисов интеграции приложений позволяет сохранить взаимодействие приложений, при этом сбой одного из сервисов или пиковая рабочая нагрузка никак не повлияют на работу других приложений.





Сервисы передачи сообщений с возможностью интеграции приложений могут адаптироваться к любой пропускной способности и использовать хранилища сообщений в зонах перекрестной доступности для обеспечения высокого уровня доступности и надежности.

2.2.1 Сервисы интеграции приложений Amazon Web Services

Передача сообщений	Надежная и высокая пропускная способность при обмене сообщениями по модели «издатель – подписчик» (Pub/Sub), SMS, электронная почта и мобильные push-уведомления Очередь сообщений для отправки, хранения и получения сообщений между компонентами приложений в любом объеме Брокер сообщений для Apache ActiveMQ и RabbitMQ, который упрощает миграцию данных и позволяет использовать гибридную архитектуру	Amazon Simple Notification Service (SNS)
Рабочие процессы	Объединение множества сервисов AWS в бессерверных рабочих процессах для быстрого создания и обновления приложений Масштабируйте Apache Airflow без необходимости выделения инфраструктуры и управления ей	AWS Step Functions
Управление API	Создание, публикация, обслуживание, мониторинг и обеспечение безопасности API в любом масштабе для бессерверных рабочих нагрузок и интернет-приложений Создание универсальных API для безопасного доступа к данным, их изменения и объединения из нескольких источников	Amazon API Gateway
Шина событий	Создание управляемой событиями архитектуры, объединяющей данные клиентских приложений, приложений SaaS и сервисов AWS	Amazon EventBridge
Интеграция API без	Автоматизация потока данных между приложениями SaaS и сервисами AWS	Amazon AppFlow

написания кода	практически при любом масштабе без необходимости писать код.	
-------------------	---	--

2.2.1 Примеры использования интеграции приложений

	Alpha Apps снижает затраты на доставку контента на 80 % благодаря использованию AWS Step Functions
	The Guardian News & Media автоматизирует доставку по подпискам с помощью AWS Step Functions
	Change Healthcare ежедневно обрабатывает миллионы конфиденциальных транзакций с помощью Amazon SNS и SQS
	redBus использует SNS и SQS для мониторинга, предупреждений и внутренней связи

2.2.1 Перемещение эксабайтов данных с AWS Snowmobile

AWS Snowmobile – сервис передачи эксабайтов данных, который способен быстро, безопасно и экономно перемещать экстремально большие объемы в AWS. Для перемещения данных используется укрепленный транспортный контейнер Snowmobile длиной 13,7 м, перевозимый при помощи грузового автомобиля с полуприцепом. В таком контейнере можно перемещать до 100 ПБ данных. Snowmobile позволяет без труда переместить большие массивы данных, например видеобиблиотеки и репозитории изображений, или выполнить миграцию всего ЦОД. Для всех данных используется 256-битное шифрование, а управлять ключами шифрования можно с помощью AWS Key Management Service (AWS KMS). В числе возможностей Snowmobile – GPS-отслеживание, сигнализация, круглосуточное видеонаблюдение и дополнительные автомобили службы безопасности для сопровождения контейнера во время перевозки.

2.3 Миграция данных с помощью Amazon Web Services

Благодаря AWS тысячи организаций успешно разместили в облаке объемные рабочие нагрузки. Среди них — GE, Coca-Cola Company, BP, Enel, Samsung, NewsCorp и Twenty-First

Century Fox. Этим компаниям удалось существенно снизить ИТ-затраты, а также повысить производительность, гибкость бизнеса и операционную отказоустойчивость.

Миграция в AWS — это любое перемещение рабочей нагрузки из локальной среды, хостинга или другого общедоступного облака. AWS помогает тысячам организаций переместить на свою платформу различные рабочие нагрузки, например веб-сайты, базы данных, хранилища, физические и виртуальные серверы или центры обработки данных целиком. Опираясь на многолетний опыт, мы разработали комплексный и проверенный подход к миграции рабочих нагрузок в AWS, чтобы вы могли быстро получить преимущества для собственного бизнеса.



Зачем переходить в облако AWS?

31 % - среднее сокращение затрат на инфраструктуру по сравнению с работой в локальной среде;

62 % - повышение эффективности управления ИТ-инфраструктурой по сравнению с работой в локальной среде.

Методика миграции

Для осуществления миграции сотен или тысяч рабочих нагрузок в облако необходим фазовый подход, включающий оценку, подготовку и планирование, миграцию и операции. Каждая новая фаза в этом подходе строится на предыдущей. В нормативном руководстве AWS представлены методы и рекомендации по прохождению каждого этапа миграции.

Инструменты и сервисы для миграции

VMware Cloud on AWS позволяет быстро переместить сотни приложений, виртуализированных на платформе vSphere, в облако AWS за считанные дни. При этом обеспечивается стабильность работы локальной среды.

Программа AWS по ускорению миграции

Программа AWS по ускорению миграции (MAP) предназначена для того, чтобы помочь организациям, желающим выполнить крупномасштабную миграцию, ускорить достижение их

бизнес-целей за счет объединения всех компонентов нашего механизма миграции с инвестициями для покрытия расходов на миграцию.

AWS Managed Services

AWS Managed Services предоставляет проверенную операционную среду корпоративного уровня, предназначенную для миграции производственной рабочей нагрузки в течение нескольких дней, а не месяцев. В целях выполнения требований безопасности и нормативно-правового соответствия AMS вводит в ваши приложения только необходимые модификации. После завершения миграции AMS несет ответственность за управление вашей облачной средой.

AWS Professional Services

Специалисты AWS Professional Services помогают автоматизировать и ускорить процесс миграции больших объемов рабочей нагрузки в облако AWS. Мы используем свои методики, инструменты и опыт для совместной работы с вашей командой и партнерской сетью AWS (APN), тем самым помогая клиентам ускорить миграцию данных и обеспечить ее надежность.

AWS Training and Certification



AWS Training and Certification позволяет разработать стратегию обучения, способную помочь вашей организации с выработкой правильных навыков работы в облаке. При участии профессионально подготовленного персонала организация сможет осуществить переход в облако на 80 % быстрее.

Миграция рабочих нагрузок с помощью CloudEndure



Сэкономьте время и средства за счет быстрой и надежной миграции баз данных в AWS

2.3.1 Решения Amazon Web Services для миграции

Комплексное решение для эффективной миграции данных в AWS и быстрого получения результатов для бизнеса. Благодаря AWS тысячи организаций успешно разместили в облаке объемные рабочие нагрузки. Среди них — GE, Coca-Cola Company, BP, Enel, Samsung, NewsCorp и Twenty-First Century Fox. Этим компаниям удалось существенно снизить ИТ-затраты, а также повысить производительность, гибкость бизнеса и операционную отказоустойчивость. Опираясь на многолетний опыт, мы разработали комплексный и проверенный подход к миграции рабочих нагрузок в AWS, чтобы вы могли быстро получить преимущества для собственного бизнеса.

Решения AWS для миграции

Наши механизмы миграции протекают с учетом особенностей, связанных с персоналом, обработкой, технологиями и финансами. Они учитываются в течение всего процесса миграции, что дает возможность гарантировать достижение желаемых результатов для бизнеса.

Методика миграции

Для осуществления миграции сотен или тысяч рабочих нагрузок в облако необходим фазовый подход, включающий оценку, подготовку и планирование, миграцию и операции. Каждая новая фаза в этом подходе строится на предыдущей. В нормативном руководстве AWS представлены методы и рекомендации по прохождению каждого этапа миграции.

2.4 Передача данных в Amazon Web Services

Потоковые данные — это данные, формируемые непрерывно тысячами источников, которые обычно отправляют записи данных одновременно и небольшими объемами (на уровне нескольких килобайтов). В состав потоковых данных входят различные виды данных, например файлы журналов, сформированных клиентами при использовании мобильных или интернет-приложений, покупки в интернет-магазинах, действия игроков в играх, информация из социальных сетей, финансовые торговые площадки и геопространственные сервисы, а также телеметрические данные, полученные от подключенных устройств или оборудования в ЦОД. Эти данные должны быть обработаны последовательно и инкрементно либо по каждой из записей, либо с использованием скользящего временного окна, после чего их можно использовать в различных аналитических задачах, включая корреляцию, агрегацию,

фильтрацию и шаблонизацию. Информация, полученная в результате подобного анализа, позволяет компаниям разобраться во многих аспектах своей деятельности, например в использовании сервисов (для задач учета/выставления счетов), активности серверов, навигации по веб-сайтам, геолокации устройств, людей или товаров, и в результате быстро реагировать на изменяющиеся условия. К примеру, компании могут отслеживать изменения общественного настроения в отношении своих торговых марок и продуктов за счет постоянного анализа потоков данных, получаемых из социальных медиа, и в случае необходимости обеспечивать своевременную реакцию.

Преимущества потоковой передачи данных

Обработка потоковых данных является предпочтительной для большинства сценариев использования, подразумевающих непрерывное формирование новых динамических данных. Обработка потоковых данных применима в большинстве отраслевых сегментов и случаев использования, подразумевающих обработку больших данных. Обычно компании начинают с простых задач, например со сбора данных системных журналов, или с элементарных вычислений, например с обновления минимумов и максимумов. Затем эти задачи трансформируются в более сложную обработку, происходящую в режиме, близком к реальному времени. Изначально приложения могут обрабатывать потоки данных с целью формирования простых отчетов и выполнения простых ответных действий, например активации сигнализации, когда значения ключевых параметров выйдут за указанные границы. В итоге этим приложениям приходится выполнять более сложные формы анализа данных, например применять алгоритмы машинного обучения и достигать более глубокого понимания ситуации на основании данных. С течением времени в этот процесс также добавляются комплексные потоковые алгоритмы обработки событий, например временные окна для определения последних популярных кинофильмов, предоставляя возможность получать более сложную аналитическую информацию.

Примеры потоковой передачи данных

- Датчики, используемые в транспортных средствах, промышленном оборудовании и сельскохозяйственной технике, отправляют данные в потоковое приложение. Приложение осуществляет мониторинг производительности, предупреждает возникновение возможных дефектов и автоматически заказывает необходимые запасные части для предотвращения простоя оборудования.

- Финансовое учреждение отслеживает изменения на фондовых биржах в режиме реального времени, вычисляет рисковую стоимость и автоматически выполняет ребалансировку портфеля ценных бумаг на основании изменений биржевого курса.
- Веб-сайт агентства недвижимости отслеживает набор данных, полученный с мобильных устройств клиентов, и предоставляет рекомендации по объектам недвижимости в режиме реального времени на основании данных геолокации.
- Гелиоэнергетическая компания должна предоставлять своим клиентам определенную проходную мощность, в противном случае ей придется платить штрафы. Она развернула приложение для обработки потоковых данных, которое осуществляет мониторинг всех используемых солнечных батарей и планирует необходимое обслуживание в режиме реального времени, что позволяет минимизировать периоды генерации низкой проходной мощности для каждой из батарей и избежать уплаты штрафов.
- Мультимедийный издатель осуществляет потоковую передачу миллиардов записей со своих онлайн-ресурсов, выполняет агрегацию и дополнение данных с учетом демографической информации о пользователях и оптимизирует размещение контента на веб-сайте, благодаря чему обеспечивается релевантность контента и повышается качество обслуживания посетителей.
- Компания-разработчик интернет-игр выполняет сбор потоковых данных о взаимодействиях игроков с играми и передает эти данные на игровую платформу. Затем выполняется анализ этих данных в режиме реального времени, в результате которого формируются стимулы и динамические эффекты для повышения вовлеченности игроков.

Сравнение пакетной обработки и потоковой обработки

Перед началом работы с потоковой передачей данных стоит сравнить *потоковую обработку* с *пакетной обработкой* и выявить различия. *Пакетная обработка* может использоваться для выполнения вычислений, связанных с обязательными запросами в различных наборах данных. Обычно при расчете результатов подобной обработки используются все входящие в пакет данные, благодаря чему достигается глубокий анализ наборов больших данных. В качестве примера платформ, поддерживающих пакетные задания, можно привести системы, использующие MapReduce, например Amazon EMR. В то же время *потоковая обработка* требует подачи последовательностей данных и инкрементного обновления метрик, отчетов и итоговой статистики в ответ на каждую поступающую запись данных. Этот тип обработки лучше всего подходит для мониторинга в режиме реального времени и функций ответа.

	Пакетная обработка	Потоковая обработка
Перечень данных	Запросы ко всем или большей части данных в наборе или же их обработка.	Запросы или обработка данных в пределах скользящего временного окна или самой последней записи данных.
Размер данных	Большие пакеты данных.	Отдельные записи или микропакеты из нескольких записей.
Производительность	Задержки от нескольких минут до нескольких часов.	Требуется задержка в пределах нескольких секунд или миллисекунд.
Анализ	Комплексная аналитика.	Простые функции ответа, агрегации данных или динамических метрик.

Многие организации выстраивают гибридные модели за счет комбинации двух подходов и поддерживают операции как на уровне реального времени, так и на пакетном уровне. Сначала данные обрабатываются с помощью платформы потоковых данных, например Amazon Kinesis, с целью извлечения важной информации в режиме реального времени, а затем размещаются в хранилище, например Amazon S3, где преобразуются и загружаются для решения различных задач пакетной обработки.

Возможные проблемы при работе с потоковыми данными

Обработка потоковых данных требует использования двух уровней: уровня хранилища и уровня обработки. Уровень хранилища должен поддерживать очередность записей и строгую непротиворечивость для обеспечения быстрых, экономичных и воспроизводимых операций записи и чтения больших потоков данных. Уровень обработки отвечает за потребление данных, расположенных на уровне хранилища, выполнение вычислений с использованием этих данных и уведомление уровня хранилища о том, какие данные можно удалить за ненадобностью. Кроме того, необходимо предусмотреть масштабируемость, надежность данных и отказоустойчивость как на уровне хранилища, так и на уровне обработки. В результате появилось множество платформ, предоставляющих необходимую инфраструктуру для создания приложений обработки потоковых данных, включая Amazon Kinesis Streams, Amazon Kinesis Firehose, Apache Kafka, Apache Flume, Apache Spark Streaming и Apache Storm.

Работа с потоковыми данными в AWS

Amazon Web Services (AWS) предлагает различные варианты работы с потоковыми данными. Вы можете воспользоваться управляемыми сервисами потоковых данных,

предлагаемых Amazon Kinesis, или выполнить развертывание в Amazon EC2 и использовать собственное решение для работы с потоковыми данными в облаке.

Сервис Amazon Kinesis – это платформа для работы с потоковыми данными в AWS. Она предлагает мощные сервисы, которые упрощают загрузку и анализ потоковых данных, а также позволяет создавать свои собственные настраиваемые приложения для решения специфических задач, возникающих при обработке потоковых данных. Платформа предлагает два сервиса: Amazon Kinesis Firehose и Amazon Kinesis Streams. Кроме того, вы можете использовать другие платформы потоковых данных, например Apache Kafka, Apache Flume, Apache Spark Streaming и Apache Storm, в Amazon EC2 и Amazon EMR.

Сервис Amazon Kinesis Streams позволяет создавать настраиваемые приложения для обработки или анализа данных в режиме реального времени для решения узкоспециальных задач. Он может непрерывно захватывать и сохранять данные со скоростью нескольких терабайт в час из сотен тысяч источников. У вас есть возможность создавать приложения, потребляющие данные из Amazon Kinesis Streams, для работы панелей управления в режиме реального времени, выдачи оповещений, реализации динамического ценообразования, проведения рекламных кампаний и т. д. Amazon Kinesis Streams поддерживает платформы потоковой обработки, выбранные пользователем, включая Kinesis Client Library (KCL), Apache Storm и Apache Spark Streaming.

Сервис Amazon Kinesis Firehose – это самый простой способ загрузки потоковых данных в AWS. Этот инструмент позволяет захватывать и автоматически загружать потоковые данные в Amazon S3 и Amazon Redshift, а затем выполнять анализ с помощью имеющихся средств бизнес-аналитики и информационных панелей практически в режиме реального времени. Он позволяет быстро реализовать подход ELT и воспользоваться преимуществами от использования потоковых данных.

Прочие потоковые решения в Amazon EC2

Пользователи могут установить платформы потоковых данных в Amazon EC2 и Amazon EMR по собственному усмотрению, а также создать собственные уровни хранилища и обработки. Создавая собственное решение для обработки потоковых данных в Amazon EC2 и Amazon EMR, можно избежать сложностей при выделении инфраструктуры и получить доступ к разнообразным вариантам хранения и обработки потоковых данных. В число

вариантов уровня хранилища потоковых данных входят Apache Kafka и Apache Flume. В число вариантов уровня обработки потоковых данных входят Apache Spark Streaming и Apache Storm.

2.4.1 Перемещение эксабайтов данных с AWS Snowmobile

AWS Snowmobile – сервис передачи эксабайтов данных, который способен быстро, безопасно и экономно перемещать экстремально большие объемы в AWS. Для перемещения данных используется укрепленный транспортный контейнер Snowmobile длиной 13,7 м, перевозимый при помощи грузового автомобиля с полуприцепом. В таком контейнере можно перемещать до 100 ПБ данных. Snowmobile позволяет без труда переместить большие массивы данных, например видеобиблиотеки и репозитории изображений, или выполнить миграцию всего ЦОД. Для всех данных используется 256-битное шифрование, а управлять ключами шифрования можно с помощью AWS Key Management Service (AWS KMS). В числе возможностей Snowmobile – GPS-отслеживание, сигнализация, круглосуточное видеонаблюдение и дополнительные автомобили службы безопасности для сопровождения контейнера во время перевозки.

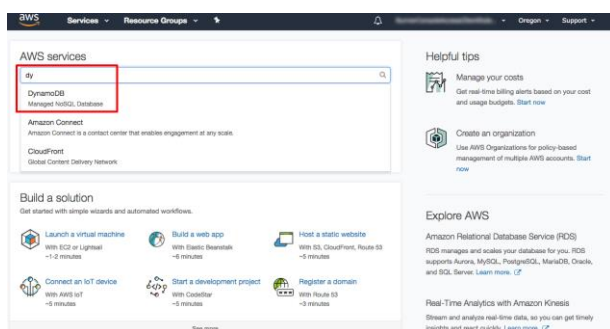
3. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ

3.2 Создание таблицы NoSQL и выполнение запросов к ней с помощью Amazon DynamoDB

В результате можно создать простую таблицу, добавлять данные, сканировать и запрашивать данные, удалять их, а также удалять всю таблицу с помощью консоли DynamoDB. DynamoDB – это полностью управляемая база данных NoSQL, которая поддерживает как документно-ориентированную модель, так и хранилища типа «ключ-значение». Благодаря гибкой модели данных, стабильной производительности и автоматическому масштабированию пропускной способности этот сервис является отличной платформой для мобильных и интернет-приложений, игр, рекламы, IoT и многих других приложений. Все действия доступны на уровне бесплатного пользования.

Для создания таблицы NoSQL и выполнения запросов к ней требуется аккаунт. Уровень бесплатного пользования AWS – это 25 ГБ хранилища и до 200 миллионов запросов в месяц для Amazon DynamoDB.

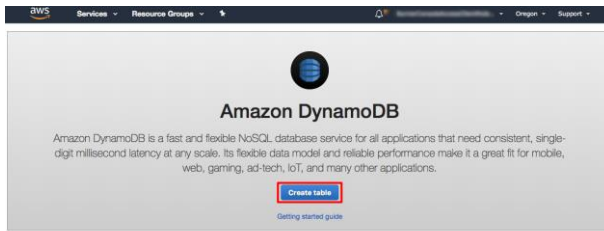
Откройте Консоль управления AWS, оставив открытым данное пошаговое руководство. Когда экран загрузится, начните вводить *DynamoDB* в строке поиска и откройте консоль DynamoDB.



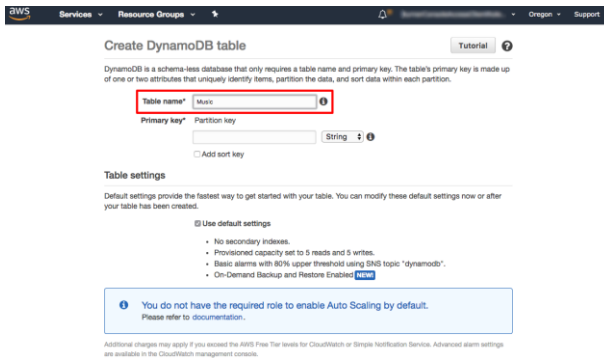
Шаг 1. Создание таблицы NoSQL

На данном шаге вы создадите таблицу с помощью консоли DynamoDB.

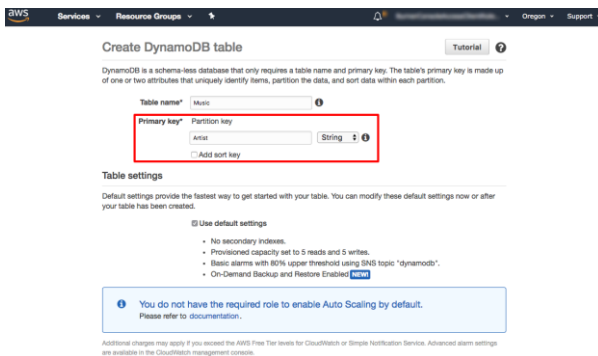
а) На консоли DynamoDB выберите Create table.



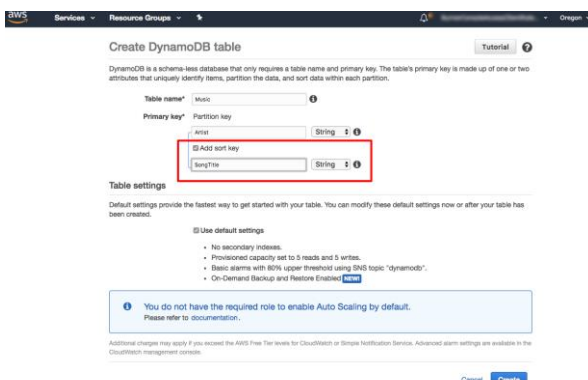
б) В качестве примера использования для этого руководства мы используем музыкальную библиотеку. В поле Table name введите *Music*.



в) Ключ секции служит для распределения данных между разделами с целью обеспечения масштабируемости. Важно выбрать для ключа атрибут с большим диапазоном значений, который с большей вероятностью обеспечит равномерно распределенную схему доступа. Введите *Artist* в поле Partition key.



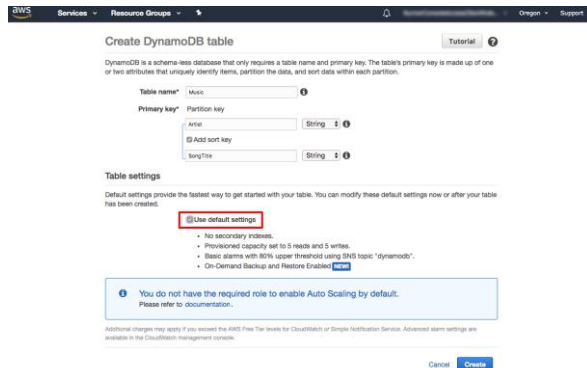
г) Поскольку у каждого исполнителя может быть много композиций, с помощью ключа сортировки можно легко сортировать данные. Установите флажок Add sort key. Введите *songTitle* в поле Add sort key.



д) Затем для таблицы потребуется включить функцию Auto Scaling в DynamoDB.

Auto Scaling в DynamoDB меняет количество выделенных для таблицы ресурсов чтения и записи в зависимости от количества запросов. С помощью роли AWS Identity and Access Management (AWS IAM) под названием *DynamoDBAutoscaleRole* сервис DynamoDB будет управлять автоматическим масштабированием от вашего имени. DynamoDB создает для вас эту роль при первом включении Auto Scaling в аккаунте.

Укажите DynamoDB на необходимость создания этой роли, сняв флажок Use default settings.



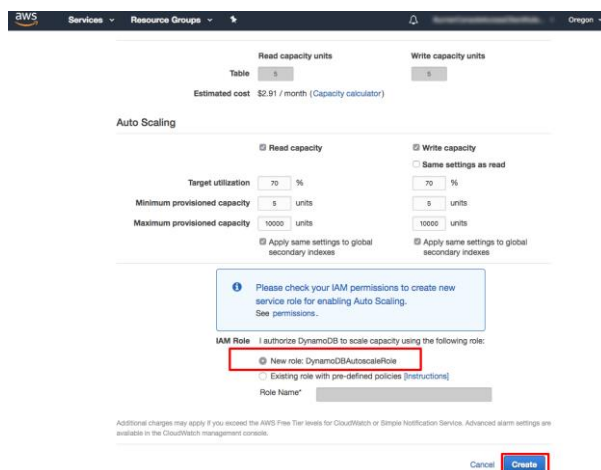
е) Прокрутите экран вниз через пункты Secondary indexes, Provisioned capacity и Auto Scaling до кнопки Create. В рамках данного руководства эти настройки изменяться не будут.

Обратите внимание, что в разделе Auto Scaling DynamoDB создаст для вас роль *DynamoDBAutoscaleRole*.

Теперь выберите Create.

Когда таблица Music будет готова к использованию, она отобразится в списке таблиц с флажком .

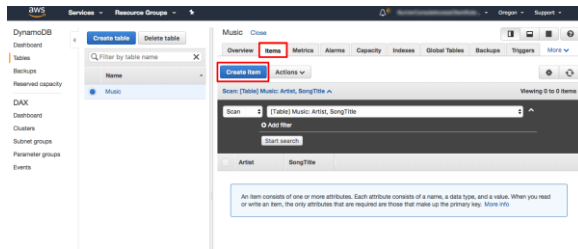
Итак, создали таблицу NoSQL с помощью консоли DynamoDB.



Шаг 2. Добавление данных в таблицу NoSQL

На этом шаге мы добавим данные в новую таблицу DynamoDB.

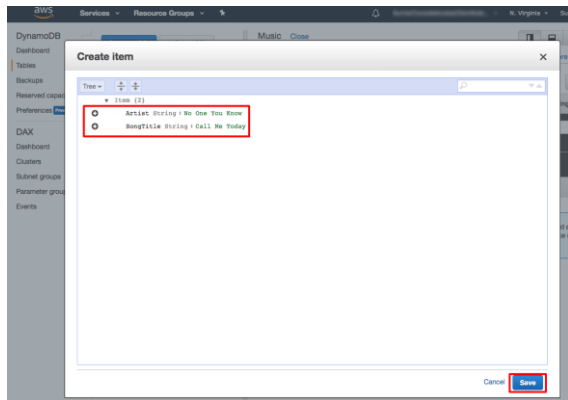
а) Перейдите на вкладку Items. На вкладке Items выберите Create item .



б) В окне ввода данных введите следующее:

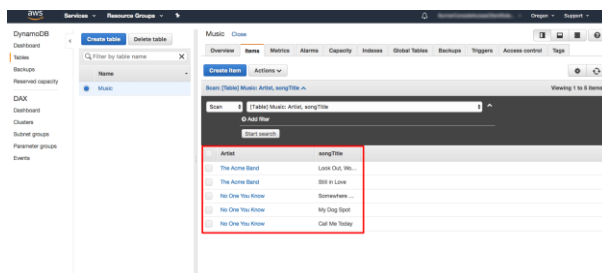
- Для атрибута Artist введите значение *No One You Know*.
- Для атрибута songTitle введите *Call Me Today*.

Чтобы сохранить элемент, нажмите кнопку Save.



в) Повторите процедуру, чтобы добавить еще несколько элементов в таблицу *Music*:

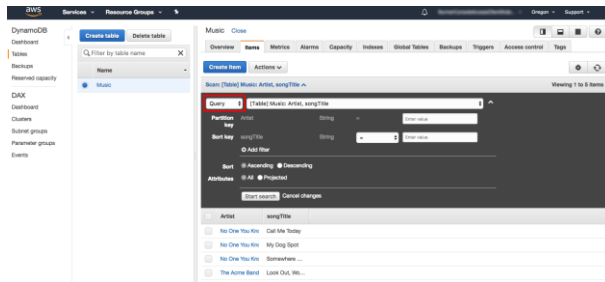
- Artist: *No One You Know*; songTitle: *My Dog Spot*
- Artist: *No One You Know*; songTitle: *Somewhere Down The Road*
- Artist: *The Acme Band*; songTitle: *Still in Love*
- Artist: *The Acme Band*; songTitle: *Look Out, World*



Шаг 3. Запрос к таблице NoSQL

На этом шаге выполняется поиск данных в таблице с использованием операций запросов. В DynamoDB операции запросов являются эффективным инструментом и используют ключи для поиска данных. Операции сканирования охватывают всю таблицу.

а) В раскрывающемся списке в темно-сером баннере над элементами измените значение Scan на Query.

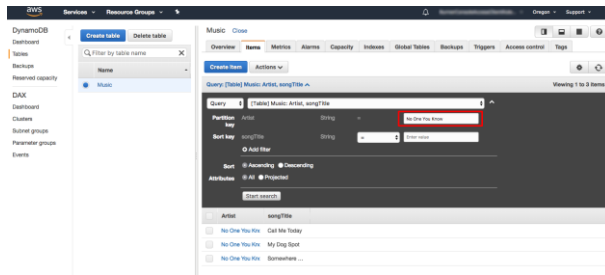


б) Можно по-разному использовать консоль для отправки запросов в таблицу *Music*. Для первого запроса выполните следующее действие:

- В поле Artist введите *No One You Know* и нажмите кнопку Start search. На экране появятся все песни исполнителя *No One You Know*.

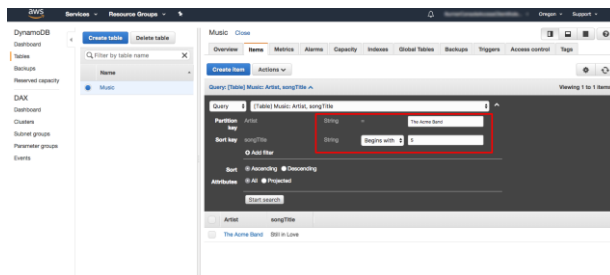
Попробуйте другой запрос:

- В поле Artist введите *The Acme Band* и нажмите кнопку Start search. На экране появятся все песни исполнителя *The Acme Band*.



в) Попробуйте другой запрос, но на этот раз сузьте результаты поиска:

- В поле Artist введите *The Acme Band*.
- В поле songTitle выберите Begins with из раскрывающегося списка и введите *S*.
- Нажмите Start search. Отобразится только композиция Still in Love в исполнении *The Acme Band*.

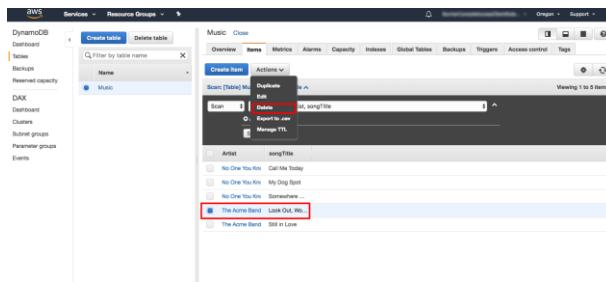


Шаг 4. Удаление существующего элемента

На этом шаге мы удалим элемент из таблицы DynamoDB.

а) Вновь измените раскрывающийся список Query на Scan.

Установите флажок рядом с *The Acme Band*. В раскрывающемся списке Actions выберите Delete. Консоль предложит подтвердить операцию удаления. Выберите Delete, чтобы соответствующий элемент был удален.

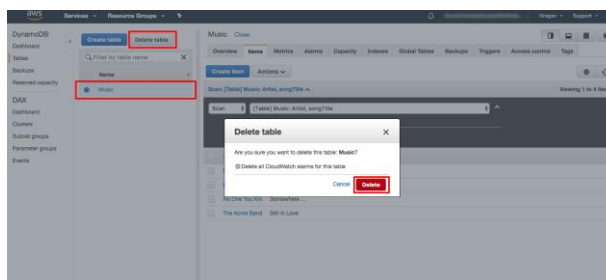


Шаг 5. Удаление таблицы NoSQL

На этом шаге мы удалим таблицу DynamoDB.

а) Можно легко удалить таблицу из консоли DynamoDB. Рекомендуется удалять таблицы, которыми вы больше не пользуетесь, чтобы не платить за соответствующие ресурсы.

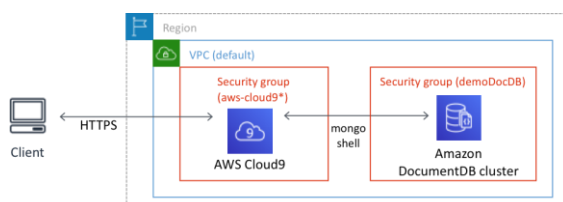
- На консоли DynamoDB выберите параметр рядом с таблицей Music и выберите Delete table.
- В диалоговом окне подтверждения выберите Delete.



3.2 Настройка документной базы данных с Amazon DocumentDB (с поддержкой MongoDB) и AWS Cloud9

Сервис Amazon DocumentDB (с поддержкой совместимости с MongoDB) – это быстрая, масштабируемая, высокодоступная и полностью управляемая документная база данных, которая поддерживает рабочие нагрузки MongoDB, позволяет хранить и индексировать данные JSON, а также запрашивать их. Описано подключение к кластеру Amazon DocumentDB из среды AWS Cloud9 через оболочку Mongo, а также создание несколько запросов.

На схеме ниже показана финальная архитектура:

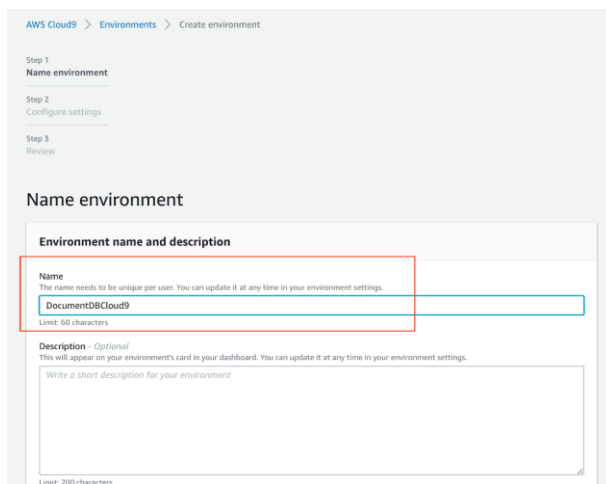


1. Создание среды AWS Cloud9

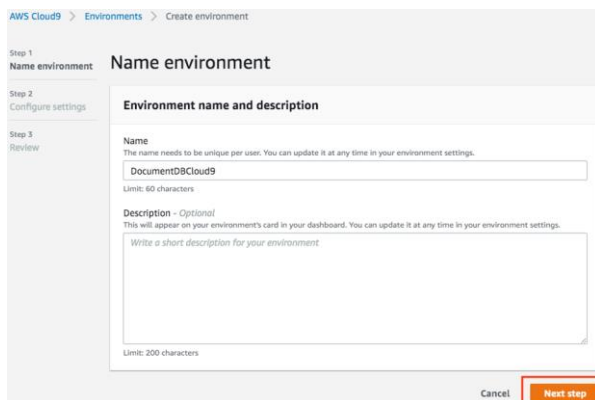
1.1 В Консоли управления AWS перейдите в консоль AWS Cloud9 и нажмите кнопку Создать среду.



1.2 Введите название DocumentDBCloud9.



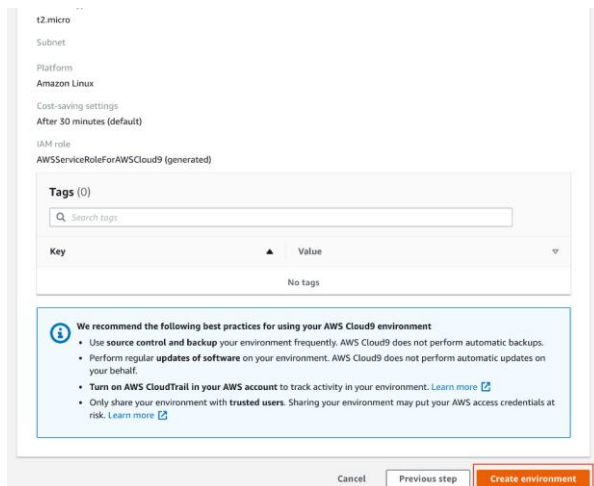
1.3 Нажмите кнопку Следующий шаг.



1.4 В разделе Настройки конфигурации примите все параметры по умолчанию.

1.5 Нажмите кнопку Следующий шаг.

1.6 В разделе Обзор, нажмите кнопку Создать среду.



1.7 Создание среды AWS Cloud9 может занять до трех минут.

2. Создание группы безопасности

2.1 В консоли управления Amazon EC2 в разделе Сеть и безопасность выберите Группы безопасности.

2.2 Нажмите Создать группу безопасности.

2.3 В поле Название группы безопасности введите *demoDocDB*.

2.4 В поле Описание опишите группу.

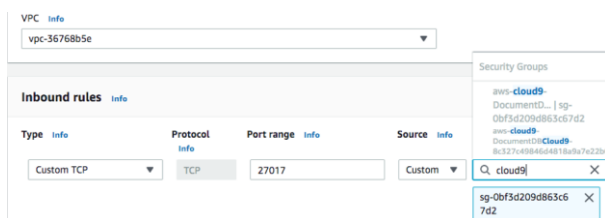
2.5 В поле VPC примите использование вашего частного виртуального облака по умолчанию.

2.6 В разделе Правила для входящих подключений нажмите Добавить правило.

2.7 В поле Тип выберите Настраиваемое правило TCP.

2.8 В поле Диапазон портов введите 27017.

2.9 Исходная группа безопасности – это группа для среды AWS Cloud9, которую вы только что создали. Под информацией о параметре Источник выберите стандартное значение Настраиваемый. Введите cloud9 в поле, расположенное рядом со значением Настраиваемый, чтобы увидеть список всех доступных групп безопасности.



2.10 Выберите группу безопасности *aws-cloud9-`<environment name>`*.

2.11 Примите все параметры по умолчанию и нажмите кнопку Создать группу безопасности.

Правила для исходящих подключений менять не требуется.

На скриншоте показаны группы безопасности, которые вы создали на этом этапе. Также вы можете видеть группу AWS Cloud9, которая появилась после создания среды AWS Cloud9.

Security group ID	Security group name	VPC ID	Description	Owner	Int
sg-00c3061f2d431b172	demoDocDB	vpc-36768b5e	My description.	arn:aws:iam::111111111111:role/AmazonDocDBRole	1
sg-0bf3d209d83c57d2	aws-cloud9-Document...	vpc-36768b5e	Security group for AW...	arn:aws:iam::111111111111:role/AmazonDocDBRole	2
sg-c51741ae	default	vpc-36768b5e	default VPC security gr...	arn:aws:iam::111111111111:role/AmazonDocDBRole	1

3. Создание кластера Amazon DocumentDB

3.1. В консоли управления Amazon DocumentDB в разделе Кластеры нажмите кнопку Создать.



3.2. На странице Создание кластера Amazon DocumentDB выберите db.t3.medium в поле Класс инстанса, а затем укажите 1 в поле Количество инстансов. Это позволит сократить затраты.

Instance class **Info**

- db.t3.medium
2 vCPUs, 4GB RAM
- db.r5.large
2 vCPUs, 16GB RAM
- db.r5.xlarge
4 vCPUs, 32GB RAM
- db.r5.2xlarge
8 vCPUs, 64GB RAM
- db.r5.4xlarge
16 vCPUs, 128GB RAM
- db.r5.12xlarge
48 vCPUs, 384GB RAM
- db.r5.24xlarge
96 vCPUs, 768GB RAM
- db.r4.large
2 vCPUs, 15.25GB RAM
- db.r4.xlarge
4 vCPUs, 30.5GB RAM
- db.r4.2xlarge
8 vCPUs, 61GB RAM
- db.r4.4xlarge
16 vCPUs, 122GB RAM
- db.r4.8xlarge
32 vCPUs, 244GB RAM
- db.r4.16xlarge
64 vCPUs, 488GB RAM
- db.t3.medium
2 vCPUs, 4GB RAM

Amazon DocumentDB requires permissions to manage AWS resources on your behalf. By clicking Create cluster, you grant permission for Amazon DocumentDB to create a service-linked role in AWS IAM that contains the required permissions.

☐ Show advanced settings Cancel Create cluster

3.3. Другие настройки оставьте по умолчанию.

3.4 В разделе Аутентификация введите имя пользователя и пароль.

Authentication

Master username **Info**
Specify an alphanumeric string that defines the login ID for the master user.
demoUser
Master username must start with a letter and contain 1 to 63 characters

Master password **Info** Confirm master password **Info**
Master password must be at least eight characters long and cannot contain a / (slash), " (double quote) or @ (at symbol).

3.5 Активируйте параметр Показывать расширенные настройки.

☒ Show advanced settings Cancel Create cluster

3.6 В разделе Настройки сети выберите demoDocDB в поле Группы безопасности VPC.

VPC security groups
A security group acts as a virtual firewall for your instance to control inbound and outbound traffic.

Select VPC security groups

demoDocDB (VPC) × default (VPC) ×

3.7 Нажмите кнопку Создать кластер.

Сервис Amazon DocumentDB запустил процесс создания кластера, который может занять несколько минут. Вы можете подключиться к кластеру, когда у кластера и инстанса будет

отображаться состояние «Доступно». В то время как Amazon DocumentDB создает кластер, вы можете завершить оставшиеся шаги, чтобы подключиться к кластеру.

4. Установка оболочки Mongo

4.1 Если ваша среда AWS Cloud9 открыта, перейдите к шагу 3.

4.2 В консоли управления AWS Cloud9 выберите DocumentDBCloud9 в разделе Ваши среды.

4.3 Выберите Открыть IDE.

4.4 Вставьте следующий код в открытую командную строку, чтобы создать файл репозитория:

```
echo -e "[mongodb-org-3.6]
Repository\nbaseurl=https://repo.mongodb.org/yum/amazon/2013.03/mongodb-
org/3.6/x86_64\npgpgcheck=1 \nenabled=1 \ngpgkey=https://www.mongodb.org/static/pgp/server-
3.6.asc" | sudo tee /etc/yum.repos.d/mongodb-org-3.6.repo
```

4.5 По завершении установите оболочку Mongo с помощью этого кода:

```
sudo yum install -y mongodb-org-shell
```

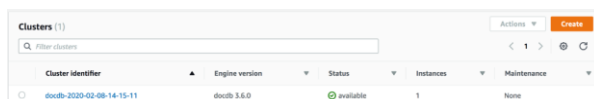
4.6 Чтобы шифровать данные при передаче, скачайте сертификат ЦС для Amazon DocumentDB. Используйте для этого следующий код:

```
wget https://s3.amazonaws.com/rds-downloads/rds-combined-ca-bundle.pem
```

4.7 Теперь вы можете подключиться к кластеру Amazon DocumentDB.

5. Подключение к кластеру Amazon DocumentDB

5.1 Найдите свой кластер в консоли управления Amazon DocumentDB в разделе Кластеры. В этом пособии для примера используется кластер docdb-2020-02-08-14-15-11.



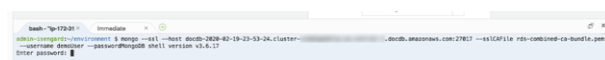
5.2 Выберите кластер, который вы создали, нажав на идентификатор кластера (docdb-2020-02-08-14-15-11 в этом примере).

5.3 Скопируйте строку подключения, указанную под Подключение к кластеру с помощью оболочки Mongo.

Не заполняйте поле `<insertYourPassword>`, чтобы ввести пароль при подключении с помощью оболочки Mongo. Так вам не придется предоставлять пароль открытым текстом в незашифрованном виде.



5.4 Строка подключения выглядит как код, представленный на скриншоте.



5.5 При вводе пароля вы должны увидеть подсказку `rs0:PRIMARY>`. Это означает, что вы успешно подключились к своему кластеру Amazon DocumentDB.

Если у вас возникнут вопросы или проблемы, посетите страницу Поиск и устранение неполадок, связанных с Amazon DocumentDB.

6. Вставка данных и запросы к ним

6.1 После подключения к кластеру вы можете создать несколько запросов, чтобы ознакомиться с работой документной базы данных.

Чтобы вставить документ, введите следующий код:

```
db.collection.insert({"hello": "DocumentDB"})
```

Вы получите такой результат:

```
WriteResult({ "nInserted" : 1 })
```

6.2 Вы можете прочитать созданный документ с помощью команды `findOne()`, поскольку она возвращает только один документ. Используйте для этого следующий код:

```
db.collection.findOne()
```

Вы получите такой результат:

```
{ "_id" : ObjectId("5e401fe56056fda7321fbd67"), "hello" : "DocumentDB" }
```

6.3 Чтобы создать несколько запросов, воспользуйтесь примером с профилями для разработчиков игр. Сначала введите несколько записей в набор соответствующих профилей. Используйте для этого следующий код:

```
db.profiles.insertMany([
  { "_id" : 1, "name" : "Tim", "status": "active", "level": 12, "score": 202 },
  { "_id" : 2, "name" : "Justin", "status": "inactive", "level": 2, "score": 9 },
  { "_id" : 3, "name" : "Beth", "status": "active", "level": 7, "score": 87 },
  { "_id" : 4, "name" : "Jesse", "status": "active", "level": 3, "score": 27 }
])
```

Вы получите такой результат:

```
{ "acknowledged" : true, "insertedIds" : [ 1, 2, 3, 4 ] }
```

6.4 Используйте команду `find()`, чтобы получить все документы из набора профилей. Используйте для этого следующий код:

```
db.profiles.find()
```

Вы получите такой результат:

```
{ "_id" : 1, "name" : "Tim", "status" : "active", "level" : 12, "score" : 202 }
{ "_id" : 2, "name" : "Justin", "status" : "inactive", "level" : 2, "score" : 9 }
{ "_id" : 3, "name" : "Beth", "status" : "active", "level" : 7, "score" : 87 }
```

```
{ "_id" : 4, "name" : "Jesse", "status" : "active", "level" : 3,
```

6.5 Создайте запрос для получения одного документа с помощью фильтра. Используйте следующий код:

```
db.profiles.find({name: "Jesse"})
```

Вы получите такой результат:

```
{ "_id" : 4, "name" : "Jesse", "status" : "active", "level" : 3, "score" : 27 }
```

6.6 Обычный способ, которым пользуются разработчики игр, заключается в поиске профиля определенного пользователя и повышении значений в его профиле. Например, вы хотите провести акцию для самых лучших активных геймеров. Если геймер заполняет опрос, вы добавляете ему 10 баллов.

Для этого используйте команду `findAndModify`. Допустим, Тим получил и заполнил опрос. Чтобы добавить баллы для профиля Тима, введите следующий код:

```
db.profiles.findAndModify({  
  query: { name: "Tim", status: "active"},  
  update: { $inc: { score: 10 } }  
})
```

Вы получите такой результат:

```
{  
  "_id" : 1,  
  "name" : "Tim",  
  "status" : "active",  
  "level" : 12,  
  "score" : 202  
}
```

6.7 Вы можете проверить правильность результата с помощью следующего запроса:

```
db.profiles.find({name: "Tim"})
```

Вы получите такой результат:

```
{ "_id" : 1, "name" : "Tim", "status" : "active", "level" : 12, "score" : 212 }
```

7. Очистка данных

По завершении обзора выключите кластер Amazon DocumentDB или удалите кластер, чтобы не платить за его использование в будущем.

По умолчанию после 30 минут неактивного состояния среда AWS Cloud9 останавливает работу соответствующих инстансов EC2, что также позволяет экономить средства.

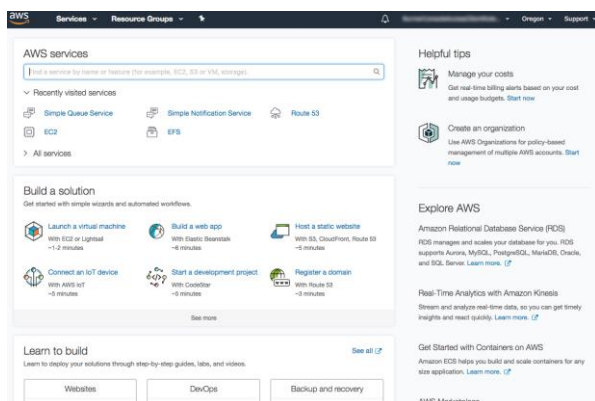
3.4 Хранение и извлечение файла с помощью Amazon S3

Amazon Simple Storage Solution (S3) – это сервис, позволяющий хранить большие объемы данных (называемые *объектами*). С помощью этого пособия вы научитесь создавать корзины Amazon S3, а также загружать в них файлы, извлекать и удалять их.

Действия, описываемые в данном пособии, можно выполнить в рамках уровня бесплатного пользования AWS. Для хранения файлов в AWS требуется аккаунт. Уровень бесплатного пользования AWS включает в себя хранилище объемом 5 ГБ, 20 000 запросов Get и 2000 запросов Put для сервиса Amazon S3.

Шаг 1. Вход в консоль Amazon S3

Щелкните здесь, и в новом окне браузера откроется Консоль управления AWS. При этом руководство останется открытым. Когда стартовый экран загрузится, введите имя пользователя и пароль, чтобы начать работу. Введите S3 в строке поиска и выберите S3, чтобы открыть консоль сервиса.

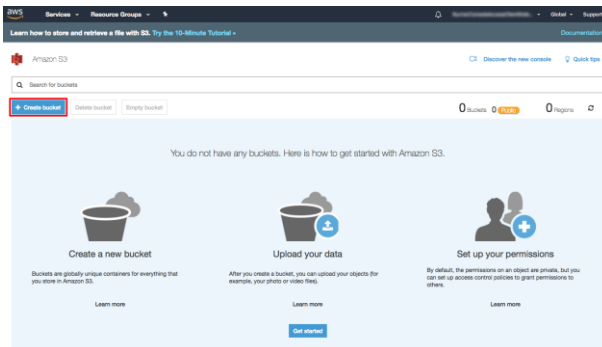


Шаг 2. Создание корзины S3

На этом шаге мы создадим *корзину* Amazon S3. Корзина – это контейнер для хранения файлов.

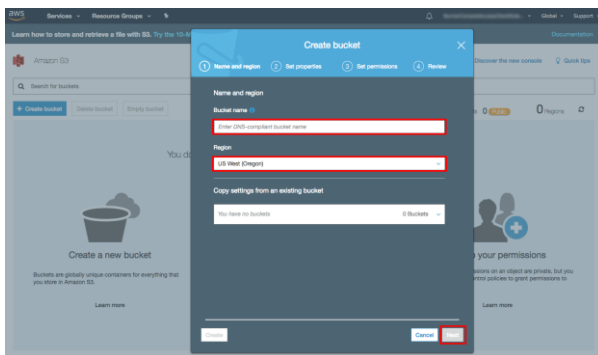
а. На панели S3 нажмите Создать корзину.

Если вы создаете корзину впервые, откроется экран, показанный на следующем изображении. Если ранее вы уже создавали корзины S3, в панели управления S3 будут показаны все имеющиеся корзины.



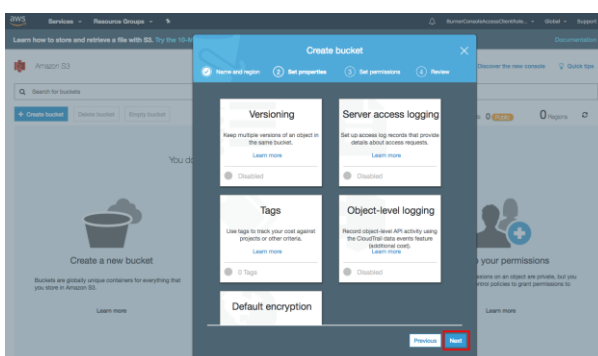
б. Введите имя корзины. Все корзины в Amazon S3 должны иметь уникальные имена. Существует ряд других ограничений для имен корзин S3. Выберите регион, в котором требуется создать корзину.

Нажмите Далее.

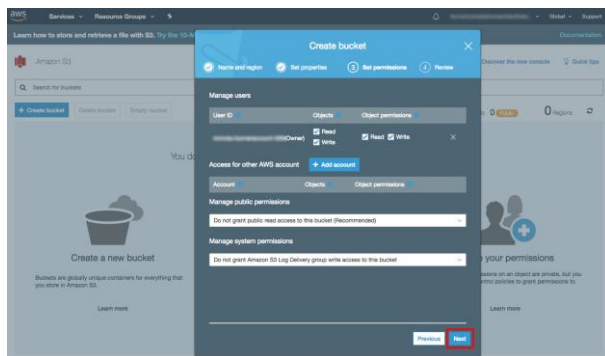


с. Для корзин S3 доступна масса полезных дополнительных возможностей, включая управление версиями, ведение журнала доступа к серверу, теги, ведение журнала на уровне объектов и шифрование по умолчанию. В этом пособии они не будут использоваться.

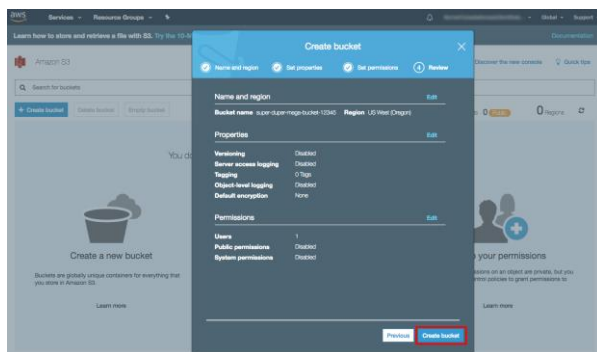
Нажмите Далее.



д. Для корзины S3 можно установить набор разрешений. Оставьте значения по умолчанию и нажмите Далее.



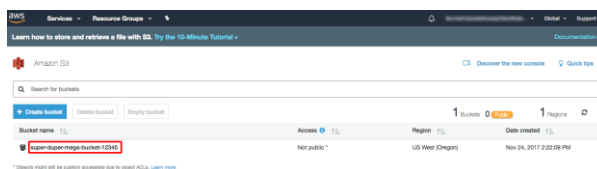
е. Проверьте параметры и нажмите Создать корзину.



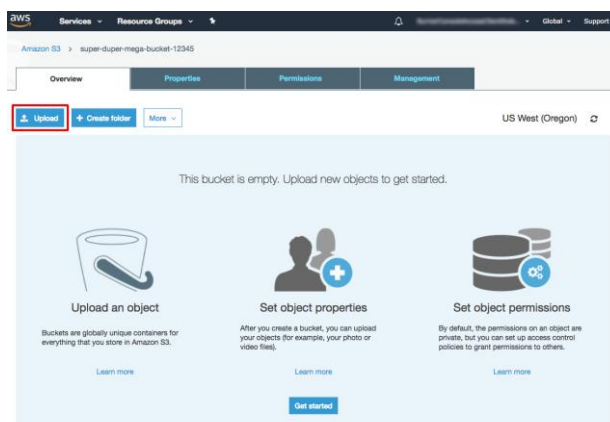
Шаг 3. Отправка файла

На этом шаге отправим файл в новую корзину Amazon S3.

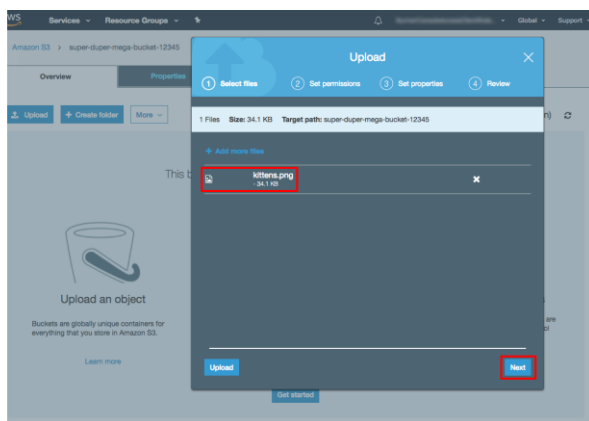
а. Созданная корзина появится в консоли S3. Чтобы перейти к корзине, нажмите ее название.



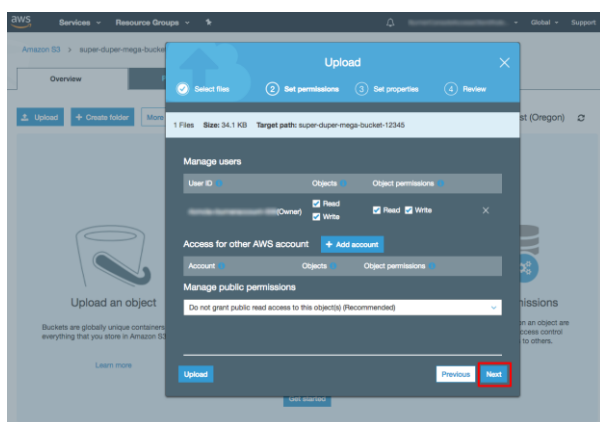
б. Откроется главная страница корзины. Нажмите Отправить.



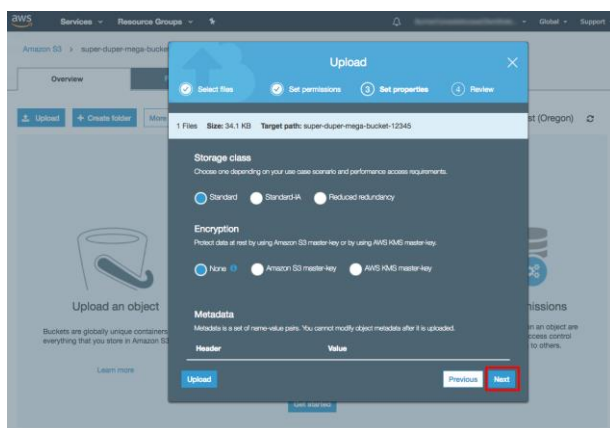
с. Чтобы выбрать файл для отправки, нажмите Добавить файлы и выберите образец файла для сохранения ИЛИ перетащите файл в область отправки. После выбора файла для отправки нажмите Далее.



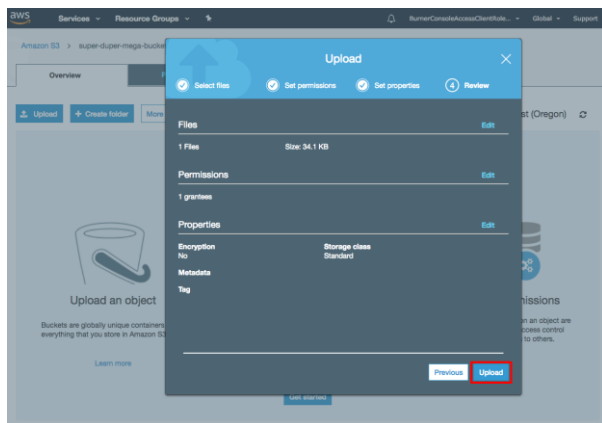
д. Для объекта можно установить набор разрешений. Для целей данного пособия мы оставим значения по умолчанию. Нажмите Далее.



е. Для объекта можно установить набор привилегий, такие как класс хранения, параметры шифрования и метаданные. Оставьте значения по умолчанию и нажмите Далее.



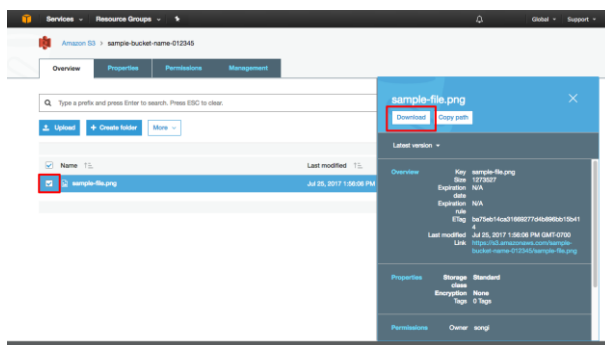
ф. Проверьте параметры и нажмите Отправить. Объект появится на главном экране корзины.



Шаг 4. Извлечение объекта

На этом шаге мы скачаем файл из корзины Amazon S3.

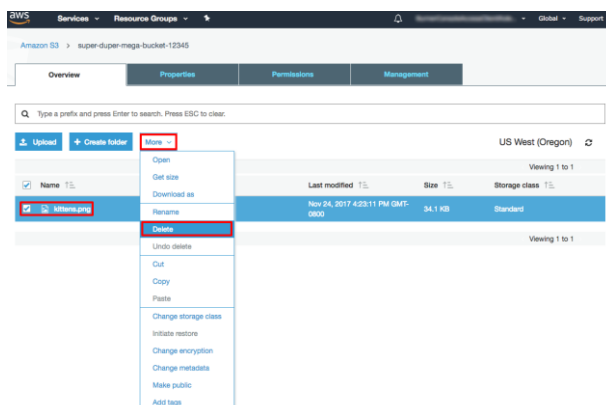
а. Поставьте галочку в поле напротив файла, который хотите скачать из корзины, и нажмите Скачать.



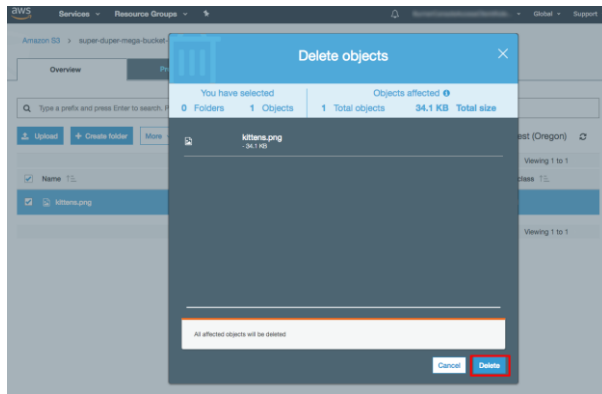
Шаг 5. Удаление объекта и корзины

Объект и корзину можно легко удалить с помощью консоли Amazon S3. Рекомендуется всегда удалять объекты, которые больше не используются, чтобы не платить за их хранение.

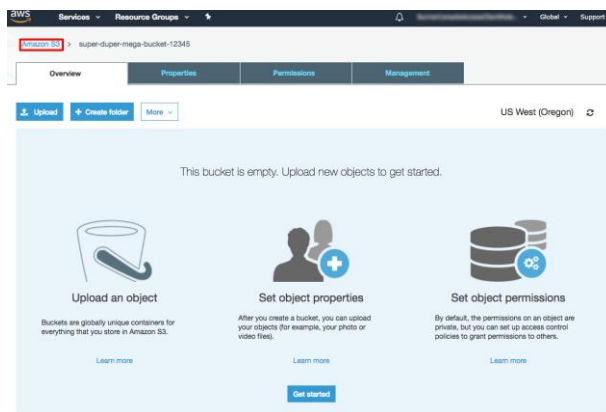
а. Сначала удалим объект. Поставьте галочку в поле напротив файла, который необходимо удалить, и выберите Дополнительно > Удалить.



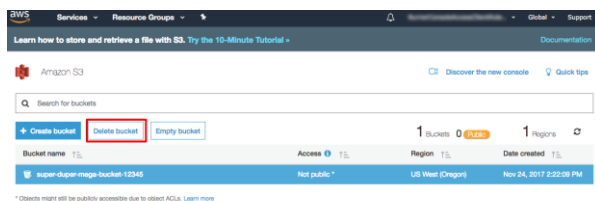
б. Проверьте, что выбрали нужный объект, и подтвердите удаление. Нажмите Удалить.



с. Нажмите на Amazon S3 для просмотра всех корзин в данном регионе.



д. Нажмите справа от имени созданной корзины, чтобы выбрать ее, а затем нажмите Удалить. Введите имя корзины и нажмите Подтвердить.



5. ТЕСТИРОВАНИЕ

4.1 План тестирования

План тестирования (тест-план) – это документ, описывающий весь объем работ по тестированию, начиная с описания тестируемых объектов, стратегии, расписания, критериев начала и окончания тестирования, до необходимого в процессе работы оборудования, специальных знаний, а также оценки рисков с вариантами их разрешения. Тест-план является важной составляющей любого грамотно-организованного процесса тестирования, так как содержит в себе всю необходимую информацию, описывающую данный процесс.

Условия для плана тестирования:

- возможность решения задач по тестированию;
- построение стратегии тестирования;
- возможность вести учёт всех требуемых ресурсов, как технических, так и человеческих;
- планирование использования ресурсов на тестирование;
- просчёт рисков, возможных при проведении тестирования.

Чек-лист плана тестирования:

- скорость загрузки страниц (интерфейса ПО);
- адаптивность;
- язык и система исчислений, формат времени представлены в привычном формате;
- удобный шрифт и качество контента;
- основной функционал понятный и его не нужно искать;
- отсутствие горизонтальной полосы прокрутки;
- однообразие интерфейса, есть карта сайта;
- присутствие информации о компании и есть рабочее лого;
- нет раздражительных элементов для пользователей (например, реклама со звуком);
- страница 404 сообщает время и причину ошибки, содержит информацию об основных разделах и контактах;
- удобство пользования функционалом;
- удобство пользования функционалом:

- мгновенный скроллинг (кнопки вверх/вниз на длинных страницах);
- варианты выбора динамически изменяются (при начале ввода есть список доступных результатов, также есть возможность не выбирать из списка, если нет подходящего варианта);
- всё, что можно сделать программно, делается;
- присутствует удобный поиск на сайте или в приложении;
- удобство кнопок (нажимается сама кнопка, а не текст на ней или близлежащая область);
- отсутствуют яркие вызывающие цвета и цветной текст;
- цвет зависит от восприятия подсознательного значения (красный – ошибка, зеленый – всё хорошо);
- оптимальный размер активных элементов (ссылки, баннеры, кнопки);
- пояснения всплывают, где это необходимо (например, почему кнопка/ссылка неактивна, не нужно их совсем скрывать);
- строка поиска, контакты, подзаголовки на странице расположены по схеме в виде буквы F (так скользит взгляд пользователя);
- оформление ссылок в однообразных цветах и стиле по всему приложению;
- единственное поле поиска, помощь при отсутствии результатов;
- возможна авторизация через социальные сети;
- есть удобная обратная связь с подтверждением вопроса, отправкой ответа;
- модуль поддержки с человеком, который также отображает время ожидания ответа;
- разбивка на тематические категории.
- обязательные для ввода поля регистрации пользователя выделены, минимальное, но необходимое количество данных;
- простота заполнения полей регистрации пользователя (использование подсказок);
- проверка валидности данных ввода по заполнению до отправки формы;
- информационное сообщение об ошибке (также выделение нужного поля);
- функция просмотра пароля;
- подтверждение регистрации;
- есть возможность отказаться от рассылок.
- единообразие в размерах однотипных форм и цветов;
- полезность (изображение с информацией или элемент дизайна);
- высокое качество.

Тестирование проводилось параллельно с практической реализацией.

4.2 Тест-кейс

Тест-кейс № 1 приведён в таблице № 1.

Таблица № 1

Название	Тест доступности сервиса
Функция	Соединение
Действие: открыть сайт https://	Ожидаемый результат: сайт открыт и доступен. Фактический результат: сайт открыт и доступен
Действие: возврат на шаг назад	Ожидаемый результат: отсутствие ошибки соединения. Фактический результат: отсутствие ошибки соединения
Результат	Тест пройден

Тест-кейс № 2 приведён в таблице № 2.

Таблица № 2

Название	
Функция	
Действие	
Действие	
Результат	

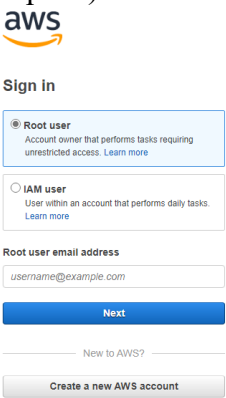
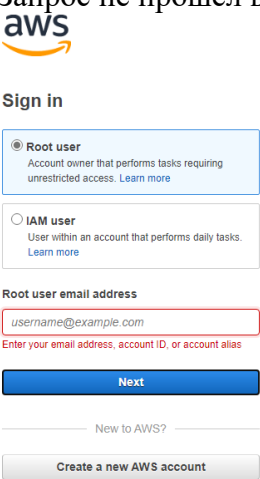
4.3 Баг-репорты

Баг-репорты отсутствуют, т. к. отсутствуют ошибки.

или

Баг-репорт № 1:

Краткое описание	Авторизация на главной странице с использованием кириллицы работает не правильно
Проект	https://aws.amazon.com/
Компонент приложения	Страница авторизации
Номер версии	0.001
Важность: S1 Блокирующая (Blocker) S2 Критическая (Critical) S3 Значительная (Major) S4 Незначительная (Minor)	S3 Значительная (Major)

S5 Тривиальная (Trivial)	
Приоритет: P1 Высокий (High) P2 Средний (Medium) P3 Низкий (Low)	P2 Средний (Medium)
Статус	Новая
Автор	Ф. И. О.
Назначен на	Должность «Программист»
Шаги воспроизведения	<p>Открываем главную страницу сайта https://aws.amazon.com/ => слева страницы находим раздел: «авторизация»(см. копию экрана):</p>  <p>Введите логин (например, «user»). Нажмите кнопку «Вход»</p>
Фактический результат	<p>Запрос не прошел валидацию. (смотри копию экрана):</p> 
Ожидаемый результат	Поиск прошел успешно, описание требуемого показано верно.

4.4 Отчёт о тестировании

В отчёте о тестировании суммированы цели и результаты тестирования (описанные в плане тестирования).

Отчёт о тестировании имеет следующую структуру.

- Обозначение продукта тестирования;
- Вычислительные системы, использованные при тестировании (технические средства, программные средства и их конфигурация);
- Дата окончания тестирования;
- Результаты тестирования описания продукта, документации, программ и данных;
- Перечень несоответствий требованиям, либо перечень несоответствий рекомендациям, либо перечень не учтенных в продукте рекомендаций, либо формулировка того, что продукт не был протестирован на соответствие рекомендациям;
- Формулировка, что результаты тестирования относятся только к протестированным компонентам продукта;
- Формулировка, что полная копия отчета о тестировании не может быть изготовлена без письменного разрешения тетировщиков.

Отчёт о тестировании приведён в приложении Д.

5. РЕЗУЛЬТАТЫ

Если сервис недоступен – раздел исключается.

5.1 Создание бесплатного аккаунта Amazon Web Services

Результаты создания бесплатного аккаунта Amazon Web Services с копиями экрана приведены в приложении А. Если сервис недоступен или аккаунт не требуется – пункт исключается.



Sign in

☒ **Root user**

Account owner that performs tasks requiring unrestricted access. [Learn more](#)

☐ **IAM user**

User within an account that performs daily tasks. [Learn more](#)

Root user email address

username@example.com

Next

— New to AWS? —

Create a new AWS account

5.2 Результаты создания таблицы NoSQL и выполнения запросов к ней

Результаты создания таблицы NoSQL и выполнения запросов к ней с копиями экрана приведены в приложении Б. Если сервис недоступен – пункт исключается.

5.3 Результаты настройки документной базы данных

Результаты создания таблицы NoSQL и выполнения запросов к ней с копиями экрана. Если сервис недоступен – пункт исключается.

5.4 Результаты хранения и извлечения файла с помощью Amazon S3

Результаты хранения и извлечения файла с помощью Amazon S3 с копиями экрана. Если сервис недоступен – пункт исключается.

5.5 Руководство пользователя

Разработанное (разработанные) Руководство (руководства) пользователя.

№ п/п	Наименование руководства пользователя	Приложение
1.	Руководство пользователя по созданию таблицы NoSQL и выполнения запросов к ней	А
2.	Руководство пользователя по настройке документной базы данных	Б
3.	Руководство пользователя по хранению. и извлечению файла с помощью Amazon S	В

ЗАКЛЮЧЕНИЕ

Целью разработки данного курсового проекта являлась разработка (подготовка) документации и отчётных форм для внедрения программных средств платформы Amazon Web Services интеграции распределённых приложений, миграции и передачи данных.

В ходе написания данной курсовой работы были получены практические навыки в области разработки (подготовки) документации и отчётных форм для внедрения программных средств платформы Amazon Web Services интеграции распределённых приложений, миграции и передачи данных.

Опыт, полученный при написании работы, подтвердил необходимость в правильной выработке требований к разработке (подготовке) документации и отчётных форм для внедрения программных средств платформы Amazon Web Services интеграции распределённых приложений, миграции и передачи данных.

В рамках курсового проектирования получен опыт создания среды AWS Cloud9 для работы с Amazon DocumentDB. Установлена оболочка Mongo, создан кластер Amazon DocumentDB, и подключение к нему, отправлено несколько запросов, вставлены данные и созданы запросы к документам JSON с помощью Amazon DocumentDB.

Создана резервная копия файла в облаке. Для этого создана корзина Amazon S3 и отправлен в неё файл как объект S3. Amazon S3 обеспечивает надёжность хранения объектов на уровне 99,999999999 %, что позволяет гарантировать доступ к ним в любой момент времени. Также извлечена сохранённая копия файла и удалена корзина.

Созданы таблицы, запрошены таблицы и элементы, управление которыми осуществлялось из консоли управления AWS.

Заявленные цели полностью достигнуты, однако, возможный недостаток, не влияющий на итоговый результат, проявляется в виде недочёта оформления пояснительной записки.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 «Юрайт», 2020
- 2 «Юрайт», 2020
- 3 «Amazon», 2020
- 4
- 5 <https://>
- 6 <https://>
- 7 <https://>
- 8 <https://>
- 9

ПРИЛОЖЕНИЕ А

(обязательное)

Руководство пользователя по созданию таблицы NoSQL и
выполнения запросов к ней

ПРИЛОЖЕНИЕ Б

(обязательное)

Руководство пользователя по настройке документной базы данных

ПРИЛОЖЕНИЕ В

(обязательное)

Руководство пользователя по хранению. и извлечению файла с
помощью Amazon S3

ПРИЛОЖЕНИЕ Г

(обязательное)

Отчёт о тестировании

Наименование подсистемы

Сервис Amazon S3 облачной платформы Amazon Web Services.

Краткая характеристика

Сервис. Amazon S3 – это безопасное, надежное облачное хранилище с большими возможностями масштабирования для объектов, составляющих статический веб-сайт. Примеры объектов, которые можно хранить: HTML-страницы, файлы CSS, изображения, видеофайлы и объекты JavaScript. Простой веб-интерфейс упрощает использование объектного хранилища Amazon S3 и обеспечивает передачу данных на хранение и их извлечение из любой точки Интернета. Это означает, что надежный доступ к сохраненным данным обеспечен для всех конечных пользователей.

Вычислительные системы, использованные при тестировании

Компьютер _____. Операционная система _____. Программное обеспечение _____ (например, браузер, Office).

Дата тестирования

14 декабря 2020 г.

Состав рабочей группы

1. Ф. И. О. (студент (студентка) группы _____ - ___, специальность – «_____»)

Выявленные в ходе тестирования ошибки (проблемы, замечания)

Ошибок (проблем, замечаний) не выявлено.

или

№ п/п	Ошибка	Описание	Способ устранения	Ошибка устранена (да/нет)
1	Инстанс перестал отвечать	Причины подобных проблем, как правило, зависят от особенностей внутренней конфигурации инстанса	Процедура восстановления инстанса, который перестал отвечать на запросы, зависит от типа инстанса (на основе EBS или хранилища инстансов). Сначала изучите информацию, которую инстанс выводит в консоль, чтобы определить, почему перезагрузка повлияла на инстанс. Иногда информации, выведенной в консоль, достаточно, чтобы определить причину ошибки инстанса. При работе в Консоли управления AWS требуется сделать следующее выберите нужный инстанс; для просмотра связанной информации в меню «Instance Actions» выберите пункт «View System Log». Если вы используете инструменты API Amazon EC2 выполните команду <code>ec2-get-console-output</code> .	
2	Потеряны данные	Чаще всего восстановить данные в хранилище инстансов невозможно.	В некоторых случаях служба поддержки AWS Support может восстановить часть данных при условии, что работа инстанса не была завершена и базовое оборудование в порядке. Процесс восстановления данных не гарантирует положительного результата и может занять несколько дней, поэтому не следует полагаться на восстановление данных службой поддержки AWS Support как на основную стратегию резервного копирования.	
3	Ошибки корневого тома	При переходе инстанса в спящий режим сохраняются данные оперативной памяти. В случае остановки инстанс отключается, а	Иногда ошибки корневого тома для некорректно загружающегося инстанса на основе EBS можно исправить вручную. Ручное исправление ошибок является сложным процессом и не рекомендуется для пользователей, не обладающих опытом системного администрирования. Примеры действий, которые пользователи	

№ п/п	Ошибка	Описание	Способ устранения	Ошибка устранена (да/нет)
		данные оперативной памяти удаляются. В обоих случаях данные из корневого тома EBS и любых подключенных томов данных EBS сохраняются.	предпринимают на основе анализа данных консоли после ошибки инстанса, включают выполнение команды fsck на соответствующем томе, отключение модуля SELinux и исправление ошибок в файле fstab.	
4				
5				

Результаты тестирования относятся только к протестированным компонентам продукта.

Полная копия отчета о тестировании не может быть изготовлена без письменного разрешения тетировщиков.