машин между физическими хостами, избегая преждевременного перераспределения вычислительных мощностей, необходимость в которых обусловлена текущими показаниями мониторинга. Данный подход позволяет давать более объективную оценку текущей нагрузки на вычислительный кластер и предоставлять альтернативные схемы управления ресурсами с учётом наиболее вероятного изменения потребления ресурсов с течением времени [2, 3].

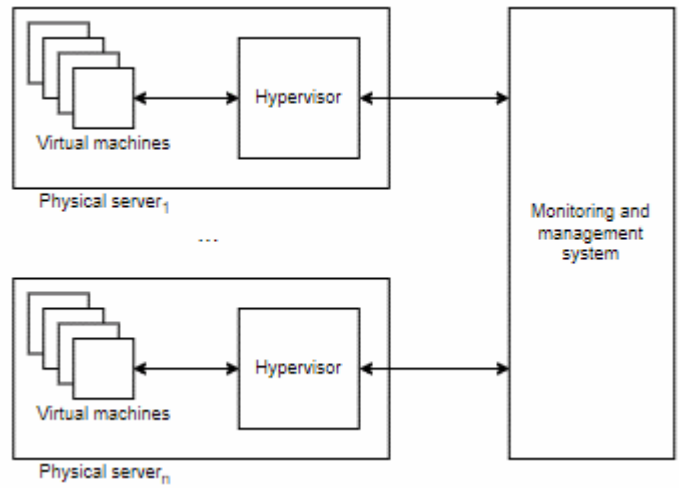

Рис. 1 – Общая схема облачного вычислительного кластера

Использование статистических предсказаний в процессе управления облачными ресурсами так же упрощает соблюдение соглашения об уровне предоставления услуг (англ. ServiceLevelAgreement, SLA). Статистическое планирование использования ресурсов позволяет предварительно резервировать дополнительные вычислительные мощности для поддержания метрик доступности и функционирования сервиса (время доступа, общая доступность и т.п.) на заявленном уровне [2].

Список использованных источников:

1. Ворожцов А.С., Тутова Н.В., Тутов А.В. Динамическое распределение вычислительных ресурсов центров обработки данных // T-COMM: Телекоммуникации и транспорт. – 2016. – Том 10. - №7. – С.47-51

2. Resource Central: Understanding and Predicting Workloads for Improved Resource Management in Large Cloud Platforms. [Электронный ресурс] – Режим доступа: https://www.microsoft.com/en-us/research/publication/resource-central-understanding-predicting-workloads-improved-resource-management-large-cloud-platforms/

3. Michael Borkowski, Stefan Schulte, Christoph Hochreiner Efficient Resource Management Technique for Performance Improvement in Cloud Computing // IEEE/ACM 9th International Conference on Utility and Cloud Computing. – 2016.

ДЕНОРМАЛИЗАЦИЯ КАК СРЕДСТВО УЛУЧШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ БАЗ ДАННЫХ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Хадарович П.В.

Медведев С.А. – к.т.н., доцент

В настоящее время количество высоконагруженных приложений неизменно увеличивается. Чтобы справиться с возрастающей нагрузкой, производится либо улучшение аппаратной части машины, обрабатывающей запросы пользователей, либо добавление нескольких новых машин для обработки запросов. Но применение некоторых методов оптимизации производительности баз данных, позволит улучшать аппаратную часть или внедрять новые машины значительно реже. Денормализация является одним из таких методов. При ее грамотном применении, можно существенно увеличить скоростьобработки данных, что в свою очередь, уменьшит время обработки запросов.

Денормализация – намеренное преобразованиеструктуры базы данных в состояние, которое не соответствует критериям нормализации баз данных. Архитектура базы данных, соответствующая критериям нормализации, способствует ее улучшенному пониманию и дальнейшему сопровождению. Но при больших объемах данных, данный подход может существенно снизить производительность обработки данных. В таких случаях, денормализация способна уменьшить время обработки данных.

Применение денормализации имеет смысл тогда, когда происходит потеря производительности по следующим причинам:

- в запросе присутствует слишком много операций объединения таблиц. В таких случаях, производится

объединение нескольких таблиц в одну, что позволяет сократить количество операций объединения [1];

- в запросе происходят сложные вычисления, возникающие, например, при использовании групповых и агрегатных функций. Чтобы избежать выполнения таких вычислений каждый раз при извлечении данных, применятся предварительное вычисление необходимых значений во время операции вставки новой записи в таблицу с последующим сохранением полученного значения в отдельной колонке, принадлежащей этой же таблице. Такой подход демонстрирует высокую производительность для данных, которыенеобходимо часто извлекать, но которые редко изменяются [2].

Основными недостатками денормализации являетсяпоявление избыточной информации. Также ее применение негативно сказывается для данных, которые часто обновляются. Кроме того, нельзя недооценивать тот факт, что денормализация ухудшает понимание модели данных и ее дальнейшее сопровождение.

Денормализация является очень действенным способом для улучшения производительности обработки данных. Она часто находит свое практическое применение в различных моделях данных. Но у нее есть свои недостатки. Прежде чем применять денормализацию, для начала нужно попробовать другие способы оптимизации производительности. И в том случае, когда они не приносят должного результата, стоит обратиться к денормализации.

Список использованных источников:
1. Grant Fritchey, Sql Server 2012 Query Performance Tuning – 470c.
2. https://habrahabr.ru/post/64524- электронный ресурс

АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Хильчук А.С.

Куликов С.С. – к.т.н., доцент

На данный момент во многих сферах общества используется множество программных продуктов. Также наблюдается тенденция к использованию мобильных приложений и отход от веб-вариантов приложений. Поэтому требования к стабильности и удобству использования этих приложений становятся выше.

Тестирование ПО – это процесс исследования, испытания программного продукта, по результатам которого можно выявить ситуации, в которых поведение программы является неправильным, нежелательным или не соответствующим спецификации. [1]

Для снижения затрат на проведение ручного тестирования внедряется её автоматизация. Автоматизированное тестирование ПО с это процесс верификации программного обеспечения, при котором основные функции и шаги теста, такие как запуск, инициализация, выполнение, анализ и выдача результата, выполняются автоматически, что помогает сократить время тестирования и упростить его процесс. [2]

Одной из главных задач при внедрении автоматизации тестирования на проекте по разработке мобильных приложений является выбор фреймворка для разработки тестов. Самым популярный из них – Appium [3], который является платформой для кроссплатформенной автоматизации тестирования с открытым исходным кодом. С помощью данного программного решения можно выполнять тестирование нативных, гибридных и мобильных веб-приложений на платформе iOS и Android. Управление элементами приложения и мобильным устройством осуществляется с использованием WebDriverJSONWireProtocol. Данный фреймворк является инструментом для тестирования по методу чёрного ящика, следовательно, нет необходимости в рекомпиляции приложения, а также в доступе к исходному коду. Данный фреймворк доступен для языков программирования Java, .Net (C#), Python, Ruby, JavaScript и PHP. К минусам можно отнести невозможность параллельного запуска тестов для платформы iOS, ограниченную поддержку жестов, сложность в настройке, поддержку версии Android только 4.1 и выше и достаточно низкую скорость работы.

К популярным фреймворкам по разработке тестов для мобильных приложений можно отнести Robotium [4]. Данная платформа с открытым исходным кодом позволяет тестировать нативные и гибридные Android-приложения по методу чёрного ящика, а значит для запуска теста, как и с Appium, от разработчика потребуется только исполняемый apk-файл. Но стоит учесть, что данная платформа подойдёт только для приложений, где переход к следующей итерации разработки не влечёт за собой сильного изменения пользовательского интерфейса, в противном случае разработчику тестов будет необходимо переписать вплоть до 90% кода тестов. Так же к минусам можно отнести поддержку только языка Java в качестве языка программирования тестов, а также поддержку только Android-приложений.

К группе часто используемых платформ для разработки автоматизированных тестов под мобильные приложения также относится Espresso от компании Google [5]. Основной API данного решения невелик, но так как это проект с открытым исходным кодом, код может быть расширен разработчиком тестов. Отличительной особенностью данной платформы является синхронизация с главным UI-потоком Android-приложения. Большинство фреймворков запускают тесты в отдельном потоке, вследствие чего возникают проблемы со скоростью и могут возникнуть критические ошибки в самих тестах вследствие обновления пользовательского интерфейса приложения. Большинство разработчиков игнорируют этот факт и ставят задержки в потоке