Основи баз даних

Що таке Database DevOps?

Почнемо з визначення DevOps. DevOps — це набір методів, що поєднують розробку програмного забезпечення (dev!) та IT-операції (ops!) з метою швидшого надання більше функцій, виправлень і оновлень відповідно до бізнес-цілей. Database DevOps застосовує ті самі принципи, переконавшись, що код бази даних включено в той самий процес, що й код розробки.

Database DevOps допомагає командам визначити та оптимізувати процес розробки та випуску додатків, усунувши відоме вузьке місце: зміни коду бази даних.

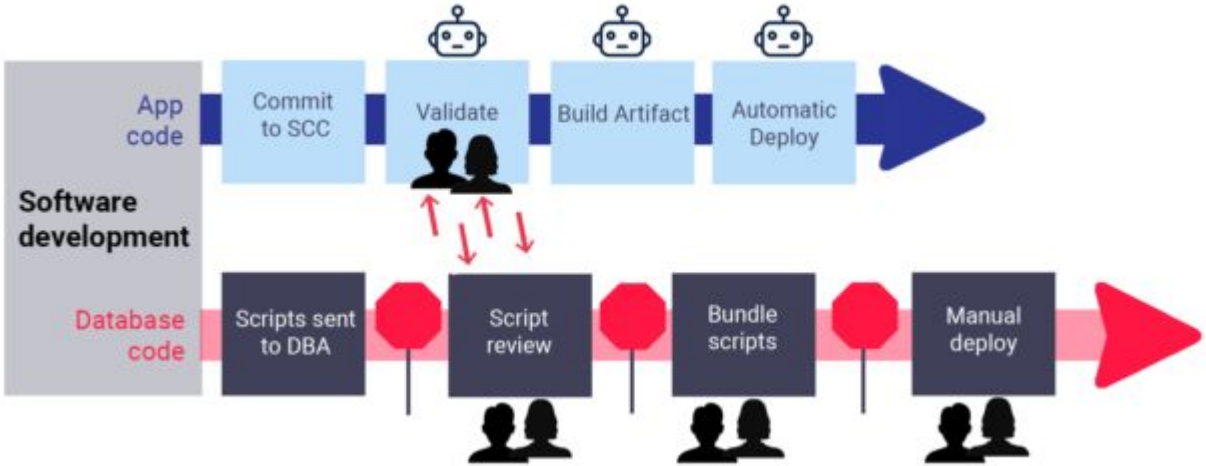
Обробка коду бази даних як коду програми

Все починається з розробників. Більшість змін коду програми, над якими працюють розробники програмного забезпечення, також впливають на код бази даних. Згідно з опитуванням стану розгортання баз даних у доставці додатків, 57% усіх змін додатків потребують відповідної зміни бази даних. Багато корпоративних розробників мають завдання внести ці зміни та написати деякий код SQL. Оскільки SQL не є фаворитом розробників і оскільки поганий код бази даних може завдати реальної шкоди, усі ці зміни зазвичай перевіряються досвідченими адміністраторами баз даних (DBA). Адміністратори баз даних перевіряють код бази даних, щоб переконатися, що зміни в базі даних не вплинуть на продуктивність програми, безпеку або інтеграцію даних.

На папері все це виглядає чудово, чи не так? Але є кілька речей, які відбуваються в реальному світі, і роблять цей процес не таким чудовим для більшості компаній.

Вузьке місце бази даних

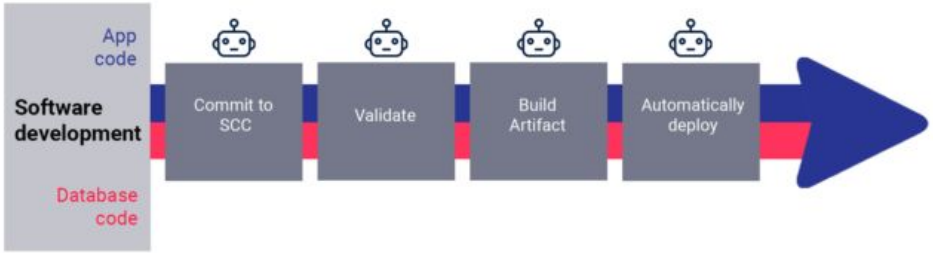
У звіті про стан розгортання баз даних у доставці додатків за 2019 рік було виявлено, що другий рік поспіль розгортання баз даних є вузьким місцем. 92% респондентів повідомили про труднощі з прискоренням розгортання баз даних.



Вузьке місце бази даних

Оскільки зміни бази даних відбуваються вручну, запити на перевірку коду бази даних часто є останньою справою, що затримує випуск. Зрозуміло, що розробники розчаровані, оскільки код, який вони написали кілька тижнів тому, все ще перевіряється. Увесь процес зміни бази даних — це просто блокувальник.

Тепер командам більше не потрібно чекати, поки адміністратори баз даних перевірять зміни до останнього етапу. Це не тільки можливо, але й необхідно зробити це раніше в процесі та запакувати весь код разом.



Основні виклики DevOps для баз даних

Дані свідчать про те, що більшість команд розробників не тільки думають про DevOps, але й активно впроваджують робочий процес DevOps у частини свого бізнесу.1 Компанії це розуміють. Вони хочуть інвестувати в практики DevOps. Останнє опитування галузі показує, що 52% організацій уже застосували підхід DevOps. Ще 30% планують прийняти підхід хоча б до деяких проектів. Це 82% опитаних організацій!

Потім ви подивіться на команду бази даних:

- 92% повідомляють, що важко прискорити процес розгортання бази даних2
- 91% доводиться кілька разів переробляти зміни бази даних, перш ніж вони будуть готові до розгортання у виробництві2
- 91% стикаються з проблемами прискорення розгортання бази даних2
- 80% погоджуються, що для впровадження змін коду бази даних потрібно більше часу, ніж для інших змін2

Основні виклики DevOps для баз даних

Легко припустити, що головною перешкодою є гроші. Здається, це не так. Згідно з опитуванням State of DevOps 2018 року, лише 10% респондентів назвали брак бюджету перешкодою для впровадження DevOps.1 Ось основні проблеми, які перешкоджають наскрізному DevOps у компаніях будь-якого розміру:

1. Надання даних для тестування програми

У вас є нова функція, яку потрібно розробити, і ви готові до роботи… за винятком того, що вам потрібні дані. Немає сенсу використовувати будь-які старі дані. Ви повинні бути якомога ближче до справжнього.

Використання робочих даних гарантує, що ваш код і дані працюють разом належним чином. Але більшість розробників не можуть (і не повинні) використовувати робочі дані. Принаймні, не раніше, ніж вони будуть зашифровані або замасковані. Нормативні вимоги та вимоги відповідності (а також базова повага до конфіденційності даних) вимагають від розробників кращих результатів. Життєво важливо, щоб усі конфіденційні дані були захищені в усіх середовищах.

Основні виклики DevOps для баз даних

1. Надання даних для тестування програми

Для багатьох розробників це відбувається так:

- Розробник подає заявку до відділу IT із переліком необхідних ресурсів (таких як доступ до бази даних, віртуальна машина з певною операційною системою та встановленими системними інструментами).
- Розробник зустрічається з IT, щоб обґрунтувати ресурси.
- Розробник чекає, поки хтось із IT запровадить конфігурацію.
- Розробник усуває проблеми з IT, тому що хтось зробив щось не так. (Наприклад, надано неправильні характеристики).
- Нарешті, через кілька днів або навіть тижнів, розробник має середовище, готове для тестування розробки.
- Через місяць, під час випуску, розробник виявляє, що сталася зміна бази даних і їхній код не працює.

Не дивно, чому так багато розробників вдаються до надання власних ресурсів з Інтернету або внутрішніх систем, чого саме IT-спеціалісти намагалися уникнути в першу чергу. Є також багато проблем, з якими стикаються адміністратори баз даних.

Основні виклики DevOps для баз даних

2. Постійність даних

У багатьох компаніях код програми знаходиться в конвеєрі CI/CD і підтримується за допомогою контролю версій. Коли розробник надсилає новий код, бац! Новий код може замінити старий код. Потім є база даних. Безперервне розгортання не таке просте для баз даних. Щоб сховище даних вважалося постійним, воно має писати в енергонезалежне сховище. Наприклад, якщо нова функція у вашій програмі вимагає зміни схеми, вам потрібно перенести дані до нової структури.

3. Дуже. Багато. Інструментів

Є дивовижні інструменти, які допоможуть спростити інтеграцію процесу DevOps. На жаль, з'явилось так багато інструментів, що стає дещо складно тримати їх рівномірно, переконатися, що вони добре поєднуються один з одним і є перспективними. Додавання нового інструменту сюди може порушити щось інше. Або, можливо, щось було неправильно інтегровано? Це може призвести до великого відтоку та невеликої роботи.

Основні виклики DevOps для баз даних

4. Люди

Найбільший виклик? Одним словом: люди. Команди розробників і операційні команди мають подолати абсолютно різні підходи до розвитку в цих багатофункціональних командах. Синхронізувати зміни програми та бази даних легше сказати, ніж зробити. Кожна компанія вже має процес злиття двох. Зазвичай це некрасиво, але до цього звикаєш. Це стає частиною вашої роботи. Думка про додавання нових інструментів і процесів до і без того складної ситуації здається неправильною. Таке відчуття, що це додає складності.

І це все; люди в цьому процесі вже напружені до межі зламу і ледве змушують свою каденцію випуску працювати як є. Важко уявити собі зміни. У цьому випадку почуття переважають над фактами.

Як подолати виклики DevOps бази даних

Доведено, що впровадження процесу Database DevOps прискорює цикли випуску та зменшує кількість помилок. Отже, що ваша команда може почати робити, щоб подолати ці дуже реальні виклики?

Почніть з малого (але думайте про ВЕЛИКИЙ вплив)

Щоб подолати фактор страху перед змінами, найкраще почати з малого. (Бонусні бали, якщо ви знайдете програму, яка справляє великий вплив.) Знайдіть або створіть невелику багатофункціональну команду та доручіть їй відповідати за невеликий випуск. Спочатку нехай команда зробить це по-старому. Відстежуйте все.

Потім доручіть тій самій команді впоратися з кожним із зазначених вище завдань, починаючи з наведених нижче пропозицій. Нехай команда спробує новий метод для кількох випусків. Відстежуйте ті самі показники. Тепер ви будете озброєні даними про швидкість і гнучкість цього нового способу впровадження змін, який змусить вас здивувати.

Як подолати виклики DevOps бази даних

Інтелектуальне самообслуговування даних

Інструменти самообслуговування даних можуть допомогти IT-адміністраторам уникнути витоків даних і проблем із невідповідністю, оскільки вони не дозволяють розробникам отримувати доступ до систем, до яких вони не мають доступу, або передавати дані в недозволені місця. І, в ідеалі, процес набагато швидший і простіший, що розробники не мотивовані шукати тіньові IT-варіанти.

Для надання даних ми рекомендуємо використовувати такий інструмент, як Delphix. Уявіть собі світ, у якому ваші розробники можуть за лічені хвилини надавати дані, які базуються на реальних даних, але постійно маскуються з метою безпеки, у будь-яке середовище — будь то локальне чи хмарне. Це змінює правила гри.

Як подолати виклики DevOps бази даних

Розумна автоматизація баз даних

Говорячи про те, що змінило правила гри, давайте поговоримо про інтеграцію змін бази даних у ваш процес DevOps і використання безперервної доставки бази даних. Коли деякі люди чують «поводьтеся з кодом бази даних як з кодом програми», реакція здригається: «Ні! Це інше!» Це правда. Вони не однакові. Але можна перевести базу даних у систему керування джерелами та обробляти зміни бази даних подібно до змін програми. Існує кілька додаткових міркувань, як-от додавання сценарію міграції, щоб дозволити існуючим даним відповідати будь-якій зміненій схемі, наприклад. Але ідея полягає в тому, щоб автоматизувати ці зміни та помістити код бази даних у контроль джерела, де можлива безперервна доставка. Коли постійна інтеграція змін бази даних відбувається разом із кодом програми, ви на порядки прискорюєте доставку програми. (І це точніше. І ви щойно створили контрольний слід.)

Як подолати виклики DevOps бази даних

Нові ролі для підтримки нового процесу

Здається нерозумним, що реалізація абсолютно нового підходу до випусків додатків вимагає нового способу визначення ролей, але багато команд ігнорують цей крок. Немає універсального способу визначити, які ролі потрібні для вашої конкретної програми та галузі. Потрібно багато проб і помилок, щоб зрозуміти, що працює, а що ні. Сядьте зі своєю новоствореною командою та обговоріть, які завдання потрібні для випуску цієї конкретної програми. Потім сядьте та визначте, де їхні навички та знання відповідають вимогам, і визначте їх роль.

Як подолати виклики DevOps бази даних

Практикуйте емпатію

Зміни нелегкі. Це може здатися нерозумним, коли ви намагаєтесь досягти результатів швидше, але важливо приділити час, щоб вислухати та співчувати кожному в команді.

Переконайтеся, що ви запровадили систему, яка дозволяє кожному в команді висловлювати занепокоєння. Кожен у команді має розуміти, як працюватиме система, щоб вони розуміли, як висвітлювати проблеми, з якими вони стикаються, і що станеться далі, щоб мати шлях до вирішення проблем.

Деякі команди перевіряють стан здоров'я наприкінці кожного спринту, де члени команди можуть оцінити свої почуття щодо різних факторів, зокрема командної роботи та підтримки. Постійні негативні оцінки можуть свідчити про те, що існує системна проблема, яка потребує уваги. Менеджери також можуть спеціально запитати в режимі 1:1 про те, як працює чи не працює комунікація, щоб виявити шаблони.

6 проблем, які допомагає вирішити DevOps бази даних

У недавній статті на DevOps.com Хелен Біл обговорює шість поширених джерел конфліктів в IT, які DevOps може допомогти вирішити. Ці джерела конфлікту виникають, коли IT намагається сформувати нову поведінку та процеси, щоб задовольнити потреби бізнесу в швидшому виконанні завдань.

Проблема №1: Незаплановані запити до інфраструктури

«Мені потрібен сервер… зараз»

Оперативні групи мають повну планову та підготовлену роботу. Коли для розробки потрібні нові середовища, це додає незаплановану роботу та застає операції зненацька. Розробка розглядає це як навмисне нехтування при спробі задовольнити потреби бізнесу, щоб рухатися швидше, що зовсім не так. Пам’ятайте, що окрім такого типу незапланованої роботи, Operations також має справу з «неочікуваними системними збоями та дефектами».

6 проблем, які допомагає вирішити DevOps бази даних

Проблема №1: Незаплановані запити до інфраструктури

Найважливіша порада Біл щодо DevOps тут полягає в тому, щоб запрошувати операційний відділ до співпраці якомога раніше в циклі розробки. Таким чином, операційний відділ краще розуміє роботу, яка виконується, і може планувати її у своїх робочих процесах.

Інша сфера, де DevOps може зменшити навантаження, це автоматизація. Це полегшує створення нової інфраструктури шляхом її автоматизації, створення або використання хмарної інфраструктури та інфраструктури віртуалізації.

6 проблем, які допомагає вирішити DevOps бази даних

Проблема №2 – Надмірний доступ до елементів виробництва

«Я хочу отримати доступ до виробничих систем»

Це трапляється, коли організація-розробник засмучена тим, що, на їхню думку, є недостатньою реакцією операцій. Хоча це здається швидким виправленням, воно збільшує ризик для виробничих систем, оскільки вводиться більше точок доступу. Підвищений ризик у виробництві призводить до більшої кількості незапланованої роботи для операцій. Більше роботи означає зниження здатності операцій підтримувати розвиток.

«DevOps не означає викинути весь процес із вікна та поставити під загрозу живе середовище», — пише Біл. «Це аж ніяк не є взаємовиключним з такими методологіями, як ITIL, але йдеться про кращу співпрацю людей для досягнення спільного бажаного результату». Щоб сприяти цій співпраці, Біл рекомендує, щоб операції передбачали розробку «визначення ваших процесів контролю змін і налаштування будь-яких систем». Працюйте над створенням деяких SLA разом як команда.

6 проблем, які допомагає вирішити DevOps бази даних

Проблема №3 – Відсутність ретроспективного аналізу несправностей

«Щось не так… чия це вина?» DevOps, як і Agile, — це зняття провини та зосередження на розробці рішень у команді. «Це означає, що жодного вказування пальцями та лайки один на одного», — пише Біл. DevOps може допомогти у двох сферах:

1. Налаштування та проведення бездоганних ретроспектив

2. Визначте способи за допомогою інструментарію, щоб або «запобігти невдачі, або вийти з ладу розумно».

6 проблем, які допомагає вирішити DevOps бази даних

Проблема №4 – забагато інструментів моніторингу DevOps

«Стільки сповіщень, вони безглузді»

DevOps виступає за створення циклів зворотного зв'язку від виробництва до розробки. Але під час роботи з клієнтами Біл виявила, що цей важливий відгук не втрачається через брак доступних інструментів DevOps для баз даних.

«Одна з проблем, з якою ми часто стикаємося, — пише Біл, — полягає в тому, що підприємства вже мають системи моніторингу. Їх багато. Моніторинг багатьох різних речей. Усе створює безліч попереджень».

Проблема, за словами Біл, полягає в тому, що ця велика кількість моніторингу призводить до «втоми тривоги». Є дані, але немає такого типу зворотного зв'язку, який необхідний для покращення, або критичний відгук губиться в морі інформації.

Порада Біл щодо цього полягає в тому, щоб об'єднати та раціоналізувати існуючі системи. «Витратьте трохи часу, щоб справді зрозуміти, які сповіщення важливі, і відповідно скорегувати свою політику», — пише вона. «Вважайте, що це постійне завдання для налаштування ваших систем у міру зміни ситуації».

6 проблем, які допомагає вирішити DevOps бази даних

Проблема №5 – Недостатньо часу на навчання DevOps

«Я не маю часу економити час»

Це тихий вбивця. Команди, які найбільше потребують використання DevOps, не мають часу думати про те, як вони можуть зробити щось краще, а тим більше зупинятися, щоб перевести дух. Збільшення очікувань впровадження інновацій у поєднанні з «крихкістю сучасної еволюції IT-інфраструктури» призводить до збільшення робочого навантаження в IT та великої кількості незапланованої роботи.

Ці 2 фактори часто є рушійними силами бізнесу, що стоять за впровадженням широко поширених показників DevOps для бази даних. За словами Біл, «DevOps розроблено для оптимізації продуктивності роботи на рівні культури, взаємодії та додатків». Це може допомогти послабити «тиск, який сьогодні зростає в багатьох IT-відділах».

6 проблем, які допомагає вирішити DevOps бази даних

Проблема №5 – Недостатньо часу на навчання DevOps

Її порада з цього приводу полягає в тому, щоб завчасно залучати виконавче спонсорство, щоб сприяти змінам на краще. Керівники можуть допомогти двома речами:

Запустіть пілот DevOps десь в організації, щоб почати зміни.

Виділіть трохи місця в бюджеті, щоб залучити зовнішнього консультанта для перевірки вашого «стану DevOps». Ймовірно, консультанти не скажуть вам нічого, чого ви ще не знаєте. Зовнішнє спостереження та документація все ще допомагають на початку побудови бізнес-кейсів.

6 проблем, які допомагає вирішити DevOps бази даних

Проблема №6. Адміністратор бази даних є вузьким місцем або перешкодою для DevOps

«Наш герой – вузьке місце…»

Іноді одна людина стає вузьким місцем у спробах швидше доставити, але не з тих причин, які ви можете припустити. «Як правило, трапляється ось що: одного разу інженер вирішує, що створення серверів — це виснажливий ручний процес, і що вони можуть написати швидкий сценарій, щоб зробити це трохи швидше. Наступного дня вони пишуть інший. І ще один».

Така поведінка - це добре. Це особа, яка бере на себе ініціативу покращити робочий процес шляхом автоматизації різних аспектів процесу. Але, за словами Біл, «проблема полягає в тому, що навіть незважаючи на те, що монстр сценаріїв працює досить добре більшу частину часу, коли він виходить з ладу, ніхто, крім його творця, практично не може усунути несправність». Це, як ви, напевно, можете припустити, не має належного масштабу. Він представляє спотворену картину рівня автоматизації в організації. Він також представляє окрему людину як єдину точку відмови для всієї системи.

6 проблем, які допомагає вирішити DevOps бази даних

Проблема №6. Адміністратор бази даних є вузьким місцем або перешкодою для DevOps

Ці героїчні люди почали автоматизувати завдання, щоб зберегти власний розум. Замість того, щоб робити це командною проблемою чи частиною процесу, ці особи беруть на себе вирішення проблеми, ймовірно, щоб уникнути бюрократії. Якщо культура в організації є більш сприйнятливою або відкритою для вивчення нових технік і методів, вона може запропонувати цим особам інші альтернативи для вивчення, окрім самостійного виправлення.

Інший варіант – усунути цих талановитих людей із щоденного процесу. Ви визнали, що це одні з найталановитіших людей в організації. Усунення їх від повсякденної роботи може зосередити їхній талант та інновації на оптимізації всього процесу або розробці нових. Їх загальний підхід може допомогти підняти всю систему.

4 причини, чому DevOps ігнорує спеціалістів із обробки даних

Уникайте групи даних на свій страх і ризик. Занадто часто ми бачили, як бази даних DevOps застосовуються в компаніях без взаємодії з групою даних. Ці зусилля не принесуть тих переваг, яких спочатку прагнув DevOps.

Простіше кажучи, зграя може рухатися так само швидко, як її найповільніший член. Якщо ви подивитесь на те, як подорожують вовки, ви помітите, що найповільніші члени знаходяться спереду, найшвидші та найсильніші – ззаду.

Але ми знову і знову бачимо, як команди DevOps повністю ігнорують спеціалістів із обробки даних. Від оцінки інструментів до вдосконалення процесів, ті, хто закликає до впровадження DevOps, регулярно ігнорують спеціалістів із обробки даних і не можуть їх прийняти. Іронія не втрачена для нас, оскільки ми бачимо рух, побудований на інклюзивності, ігноруючи ключовий компонент успіху компаній. Ось 4 причини, чому DevOps ігнорує спеціалістів із обробки даних.

4 причини, чому DevOps ігнорує спеціалістів із обробки даних

1. **Їх легко ігнорувати.**
   Більшість компаній можуть розгортати зміни баз даних у своїх базах даних для розробників і тестів. Був час, коли ця відповідальність належала Data Group. Але з впровадженням Agile компанії поступилися відповідальністю за впровадження змін у базі даних команді розробників і тестувальників. Таким чином, розробникам і тестувальникам не потрібно взаємодіяти з групою даних. Це призвело до того, що Data Group була повністю виключена з архітектурних рішень даних, оскільки вони повинні просто прийняти зміни відповідно до вимог команди розробників. Якщо група даних все-таки робить проблему через неправильні зміни схеми бази даних («Що! Це індекс із вісьмома стовпцями?!?!?»), команди випуску використовують обмеження терміну, щоб змусити групу даних все одно внести зміни. Тому не потрібно нічого робити, крім як ігнорувати групу даних. Подібність із Дугласом Адамсом «Поле чужих проблем» неймовірна.

4 причини, чому DevOps ігнорує спеціалістів із обробки даних

2. **Дані важкі.**
   Концепція стану дуже складна для команд розробників. Одним із 12 факторів для додатків є додатки без стану. Таким чином, введення ідеї обробки стану для рівня стійкості часто відкидається з рук. Як було сказано раніше, це чиясь проблема, тож навіщо взагалі її вирішувати. Якщо хтось інший несе відповідальність і проблема складна, немає причин її вирішувати. Зрештою, «це операційна проблема».

4 причини, чому DevOps ігнорує спеціалістів із обробки даних

3. **Фахівці з обробки даних мають інші цілі, ніж розробники.**
   У групи розвитку є стимул щось змінити. Оперативна група стимулюється підтримувати стабільність. Зміни є ворогом стабільності. Таким чином, ми застосували DevOps, щоб пом'якшити конфлікт і знайти спосіб, щоб обидві команди досягли своїх цілей. У Data Group дещо інші цілі. Вони також прагнуть стабільності, але цілісність і безпека даних також є найважливішими. Пам' ятайте, що найціннішим активом компанії в наборі технологій є дані. Подивіться на приклад програми для обміну фотографіями: неважливо, що ви можете наклеїти на свої фотографії собачі вуха, якщо програма продовжує забувати, з ким ви підключені, і ви не можете поділитися фотографією. Функції важливі, але лише за наявності даних для їх підтримки. Звичайно, люди, які створюють ці функції, можуть заперечити це. Таким чином, ми маємо різницю в цінностях, яка призводить до уникнення.

4 причини, чому DevOps ігнорує спеціалістів із обробки даних

4. **Фахівці з обробки даних надто часто кажуть «Ні».**
   Нарешті, спеціалісти з обробки даних дійсно мають заслужену репутацію доктора Ні. У минулому адміністратори баз даних були винні у використанні даних як клубу Біллі, щоб добитися свого. У результаті вони взяли на себе всю відповідальність за зміни виробничої бази даних. Це включає в себе всю славу і всі головні болі; сьогодні це все головний біль, біль і ламання рук. Через довгу історію доручення їм бути остаточними арбітрами розгортання виробничих баз даних, і часто будучи незадоволеними змінами в схемі та логіці, створеними розробкою, вони змушують решту компанії висловлювати своє розчарування короткими словами: «Добре. Тоді зробіть це самі». Через такий тривалий час вимоги бути єдиною точкою, через яку протікають усі зміни, Data Group не лише спричинила зростання їх робочого навантаження разом із випуском додатків (експоненціально!), тепер, коли вони страждають від високих вимог, решта вовчої зграї не дуже хоче допомогти.

Ми можемо це виправити

Ці виклики не є непереборними. Ми можемо вирішити ці проблеми, усунувши першопричину: група даних не є частиною наших нових команд розробки. Зараз ми розробляємо міжфункціональні команди; у нас більше немає окремих команд UX і Business Logic. Таким чином, нам потрібно застосувати розподілене керування даними та перемістити деяких із наших спеціалістів із обробки даних у ці Групи розробки.

Звичайно, нам все ще потрібні наші спеціалісти з обробки даних, які розуміють проблеми зберігання, продуктивності та високої доступності баз даних. Але великий відсоток наших фахівців із обробки даних потребує порівняння з колегами з системного адміністрування та відображення їхньої кар'єри, прийнявши інфраструктуру як код.

Застосувавши цей новий підхід із об'єднанням і розподілом обов'язків, ми можемо швидше просунути наш пакет до кінцевої мети, яка полягає у швидшому випуску програмного забезпечення та задоволених клієнтів.

Чому IT-директори повинні звернути увагу на DevOps

Загроза номер один для заходів Agile або цифрової трансформації вашої організації полягає в тому, наскільки добре ваша команда керує процесом випуску бази даних. Коли випуски програмного забезпечення відбувалися лише один або два рази на рік, перегляди вручну для змін бази даних мали сенс. Це вимагало ресурсів DBA, щоб також переглянути поточний стан бази даних і зрозуміти зміни в контексті. «Чи збирається новий індекс створювати занадто багато в таблиці? Чи стовпці в цьому зовнішньому ключі вже мають індекси?» це запитання, на які потрібно відповісти перед розгортанням зміни бази даних.

Зараз, коли кількість програм та їх випусків зростає, DevOps баз даних і цифрова трансформація знаходяться під загрозою через схильний до помилок, виснажливого та повільного процесу.

Ось п’ять причин, чому IT-директори повинні дбати про випуски баз даних

1. Ваша компанія розробляє програмне забезпечення.

Як сказав нам Марк Андрісен у 2011 році, програмне забезпечення поїдає світ. Оскільки всі компанії залучатимуть своїх клієнтів за допомогою привабливих програм, компанії з найкращим програмним забезпеченням будуть переможцями. Перевірений спосіб удосконалення програмного забезпечення – це ранній і частий випуск. Однак якщо ваша організація загрузла в застарілих ручних змінах, ваш бізнес ніколи не досягне свого повного потенціалу як компанії-розробника програмного забезпечення. Коли ви чуєте заяву «Тут це робиться не так», вам потрібно негайно запитати, чому це так, і допомогти своїй команді зрозуміти, що старі упередження повинні формувати те, як вони досягають своїх цілей. Настав час розбити скло.

Ось п’ять причин, чому IT-директори повинні дбати про випуски баз даних

2. Якщо ви спочатку не вирішите проблему з базою даних, ви витрачаєте гроші на інші вдосконалення.

Знову і знову ми чуємо, як технологічні лідери кажуть, що їм слід було вирішити проблему розгортання бази даних, перш ніж братися за легку проблему автоматизації випуску програм. Стадо може рухатися лише так швидко, як його найшвидший член. Незалежно від того, скільки грошей і часу ви інвестуєте в DevOps, зусилля будуть витрачені даремно, доки ви не прискорите випуски своєї бази даних. Ми називаємо це «розривом швидкості», коли темпи випуску додатків перевищують випуски баз даних. Мені подобається думати про це як про встановлення коліс на Тесла; просто немає сенсу використовувати старі та нові методи випуску програмного забезпечення разом.

Ось п’ять причин, чому IT-директори повинні дбати про випуски баз даних

3. Компанії, які повністю використовують DevOps, перевершують S&P 500.

Найкраща частина звіту про стан DevOps – це докази того, що компанії, які запровадили DevOps, перевершили S&P 500 за трирічний період. Як зазначено вище, ви повинні прийняти повну стратегію DevOps і включити базу даних, щоб повністю реалізувати потенціал вашої компанії. Компанії DevOps більш спритні та конкурентоспроможні на ринку. Вони можуть швидше скористатися можливостями.

Ось п’ять причин, чому IT-директори повинні дбати про випуски баз даних

4. Ваші конкуренти вирішують цю проблему і перемагають вас.

Опитування, проведене Liquibase (Datical) і журналом CIO, показало, що швидкість випуску баз даних найбільше блокує цифрову трансформацію. 91% опитаних адміністраторів баз даних заявили, що їхній поточний процес випуску баз даних уповільнює випуск програм. У ваших нових конкурентів цієї проблеми немає. Крім того, ваші старі існуючі конкуренти так само обізнані про цю проблему, як і ви. З конкуренцією з боку стартапів, які намагаються забрати ваш бізнес із високою рентабельністю, а старі існуючі конкуренти вирішують цю проблему, ви не можете дозволити собі зволікати.

Ось п’ять причин, чому IT-директори повинні дбати про випуски баз даних

5. Ви збережете своїх співробітників і підвищите їх кваліфікацію.

Найкращі та найталановитіші адміністратори баз даних із багаторічним досвідом роботи втомилися від роботи у вихідні. Це безпосередньо впливає на якість їхнього життя і є абсолютно непотрібним. Оскільки економіка США заграє з повною зайнятістю для висококваліфікованих працівників, IT-директори не можуть дозволити собі втрачати своїх найкращих працівників на користь конкурентів, які пропонують кращі робочі умови. Ми не говоримо про стільці Aeron і столи для настільного футболу; ми говоримо про фортепіанні концерти, ігри малої ліги, пропущені побачення, тому що члени вашої команди чекають випуску бази даних. Якщо IT-директори не вирішать цю проблему, ваші співробітники знайдуть іншу роботу, де це не проблема.

Найкращі практики для Database DevOps

Незважаючи на те, що база даних створює кілька унікальних проблем, нею слід керувати за допомогою тих самих основних протоколів DevOps, які забезпечують безпечний і надійний вихідний код, завдання, конфігурацію, збірку та керування розгортанням. Однак часто базою даних нехтують, оскільки вона складніша. На відміну від інших програмних компонентів або скомпільованого коду, ви не можете просто скопіювати зміни бази даних від розробки до тестування та виробництва. Оскільки база даних є сховищем вашого найціннішого активу — ваших даних, точне її збереження є обов’язковим для безперервної доставки.

Ось найкращі практики, які ми знайшли для команд, які хочуть інтегрувати зміни та розгортання бази даних у загальний процес, щоб покращити якість випуску та пришвидшити випуск.

Найкращі практики для Database DevOps

Використовуйте той самий процес для доставки всього коду

Оскільки у вас, імовірно, уже є інструменти для переміщення коду програми, дуже просто використовувати ті самі інструменти та процеси для коду бази даних. Перевірте весь код у систему керування джерелами та використовуйте CI/CD однаково для коду програми та коду бази даних. Код бази даних вимагає додаткової уваги.

Вносьте невеликі поетапні зміни в базу даних

Якщо DevOps чогось навчив нас, так це тому, що невеликі зміни кращі. Особливо це стосується баз даних. Перехід до фази розгортання випуску програмного забезпечення та усвідомлення того, що в базі даних є погані зміни, які потрібно знайти та переробити адміністратору баз даних, зупиняє все. Це навіть не найгірший сценарій. Якщо запровадити погану зміну бази даних, це може змусити вашу команду вивести все з ладу, а потім витратити наступні кілька днів, щоб привести все в порядок. Ключовим є використання інструменту, який дозволяє невеликі зміни, які можна відстежувати.

Найкращі практики для Database DevOps

Спростіть об’єднання змін бази даних із кількох джерел

У сучасних організаціях дані рідко зберігаються в одному місці. Крім того, найкраще бути проактивним, оскільки трапляються такі речі, як злиття та поглинання компаній. Розумно планувати заздалегідь і налаштовувати процес із самого початку, щоб спростити об’єднання змін із різних джерел.

Найкращі практики для Database DevOps

Увімкніть швидкі цикли зворотного зв'язку

Важливою частиною створення хорошого програмного забезпечення є хороше спілкування. Цикли зворотного зв’язку – це механізми, які використовуються для перевірки та отримання відгуків про процес розробки програмного забезпечення, і важливо включити цикли зворотного зв’язку бази даних, якщо ви цього ще не зробили. Мета полягає в тому, щоб отримати позитивний і негативний відгук і швидко реагувати на нього для покращення процесу.

Datical має автоматизовані контури зворотного зв’язку, вбудовані безпосередньо в систему. Механізм правил бази даних перевіряє код бази даних. Якщо код не проходить перевірку правил, відгук надсилається безпосередньо розробнику для виправлення. Крім того, Datical моделює зміни бази даних перед розгортанням. Якщо щось не так, розробники негайно отримують сповіщення, щоб виправити це, перш ніж розгортаються будь-які погані зміни бази даних.

Найкращі практики для Database DevOps

Використовуйте керування версіями змін бази даних, щоб мати детальний контроль над функціями

Уявіть собі світ, де ви можете бачити всі зміни, внесені у ваші бази даних, на одній інформаційній панелі. Багато компаній впроваджують багато змін у базу даних одночасно, і це спричиняє проблеми, коли щось йде не так. Ніхто не знає, яка зміна спричинила проблему. Крім того, якщо ваша компанія вирішить, що функція не буде розгортатися з певним випуском, ви можете легко побачити, які зміни бази даних пов’язані з цією функцією.

Найкращі практики для Database DevOps

Тестуйте рано і часто

Безперервне тестування стало найкращою практикою для інтеграції тестування протягом життєвого циклу розробки. Gartner прогнозує, що до 2020 року ініціативи DevOps допоможуть 50% підприємств запровадити постійне тестування. Безперервне тестування покладається на такі інструменти, як Jenkins, для перевірки вихідного коду, паралельного запуску тестів і сповіщення розробників про відхилення збірки. Це визначає пріоритетність дійових елементів, щоб команди могли зосередитися на завданнях, які мають найбільший вплив на бізнес-пріоритети.

Найкращі практики для Database DevOps

Створіть один раз, розгорніть багато (використовуючи артефакти)

Більшість програмних процесів використовують щось на зразок наступного:

- Перевірте код програми в системі керування джерелами (наприклад, Git)
- Створіть артефакт за допомогою інструменту CI (наприклад, Jenkins)
- Збережіть в репозиторії (наприклад, Artifactory)
- Розгорніть за допомогою інструменту CD (наприклад, Jenkins)

Найкраща практика Database DevOps виглядатиме приблизно так:

- Перевірте код програми та код бази даних у системі керування джерелами (наприклад, Git)
- Виконайте перевірки правил зміни бази даних (автоматизовано за допомогою Datical)
- Створіть артефакт за допомогою інструменту CI (наприклад, Jenkins)
- Збережіть в репозиторії (наприклад, Artifactory)
- Симулюйте зміни для забезпечення ідеального стану бази даних (автоматизовано за допомогою Datical)
- Розгорніть за допомогою інструменту CD (наприклад, Jenkins)

Найкращі практики для Database DevOps

Забезпечте виробниче середовище для розробки змін

Щоб переконатися, що зміни коду додатка та коду бази даних добре відтворюються під час розгортання на виробничій платформі, найкраще розробити та протестувати середовище, яке максимально наближено до вашого виробничого середовища.

Чим більше схоже на виробництво ваше середовище тестування та постановки, тим впевненіше ви можете бути, аж до моменту випуску «запусти й забудь» включно.

З огляду на це, ваше середовище розробки не обов’язково має бути таким самим, як робоче. Наприклад, ви точно не повинні використовувати реальні виробничі дані. Має набагато більше сенсу використовувати такий інструмент, як Delphix.

Найкращі практики для Database DevOps

Інструментуйте процес для видимості, послідовності та надійності

Важливо налаштувати весь процес, щоб ви могли легко отримувати дані для пошуку проблем і вести об’єктивний, бездоганний шлях до вдосконалення. Він має бути значущим і легко візуалізованим для виявлення проблем і тенденцій. Дані мають бути прозорими та доступними кожному.

Кінцевою метою є створення програм, які збирають дані про бізнес, програми, бази даних та інфраструктуру. Для бази даних DevOps Liquibase збирає детальну інформацію про кожне розгортання бази даних, яке вона виконує, і автоматично реєструє її в централізованій базі даних, усуваючи людські помилки, властиві ручному введенню.

Клієнти Liquibase Business і Enterprise отримують легкий доступ за вимогою до інформації про розгортання бази даних за допомогою консолі керування розгортанням бази даних, що дозволяє легко стежити за результатами розгортання. Зацікавлені сторони в усьому підприємстві можуть швидко бачити статус будь-яких змін бази даних, помилок, попереджень і порушень правил.

Підсумок

Ви можете надсилати код настільки швидко, наскільки ви можете його розгорнути. Автоматизація змін бази даних і розгортання є критично важливою. Це усуває людські помилки та прискорює процес. Якщо після розгортання вам потрібно швидко виправити помилку, важлива можливість перевірити її та виконати швидке нове розгортання.

Питання та відповіді