禪

طرحی جامع برای مدیریت تغییرات روی بانک اطلاعاتی

چگونه **DevOps** را برای پایگاههای داده پیادهسازی کنیم؟

code
plan
build
Dev
test
release
deploy
Ops
Operate
monitor

دواپس (DevOps) دیگر تنها مختص به ارتباط و همکاری و تقسیم وظایف بین گروههای توسعه نرمافزار و عملیات اجرایی یک سازمان نیست. به همین دلیل است که هر نوع عملیات دستی در چرخه تحویل شما باید مورد ارزیابی قرار گیرد تا مشخص شود که آیا میتوان آن را خودکارسازی کرد یا خیر . تغییرات پایگاه داده بهیقین یک فرآیند خستهکننده است و بهتر است که در پیادهسازی دواپس در نظر گرفته شود .

نکته قابل توجه اینکه باید بدانید، هیچ فرمول جادویی واحدی وجود ندارد که بتواند تمام مشکلاتتان را برطرف کند . همیشه هر راهکاری به این بستگی دارد که معماری شما تا چه اندازه پیچیده و بههمپیوسته است. هدف از این نوشتار این نیست که توضیح دهد چرا تغییرات پایگاه داده میتواند مشکل باشد، بلکه قصد دارد به شما یک دیدگاه متفاوت برای بهینهسازی تغییرات پایگاه داده بدهد و یک راهکار عملی از اینکه چگونه دواپس میتواند این فرآیند را تسهیل کند، ارائه دهد .

بهکارگیری پایگاه داده در **DevOps**

روش سنتی اعمال تغییرات روی یک پایگاه داده به این شکل آغاز میشود که توسعهدهندگان تغییراتی را که مایل به انجام آن هستند، روی فایلهایی با فرمت SQL مینویسند . سپس، این تغییرات توسط فردی که تجربه بیشتری در پایگاههای داده دارد، مورد بازبینی قرار میگیرد . معمولا یک مدیر پایگاه داده (DBA) که تمام مدت با پایگاه داده سروکار دارد، این بازبینی را انجام میدهد . این شخص بهتر از هر فرد دیگری با پیامدهای حاصل از اعمال تغییرات (نهتنها در وضعیت عملکرد، بلکه در ادغام دادهها) آشنا است. تا اینجا به نظر همهچیز درست و بدون نقص است، اما مشکل اینجا است که مدیر پایگاه داده

[illegible]

[illegible]

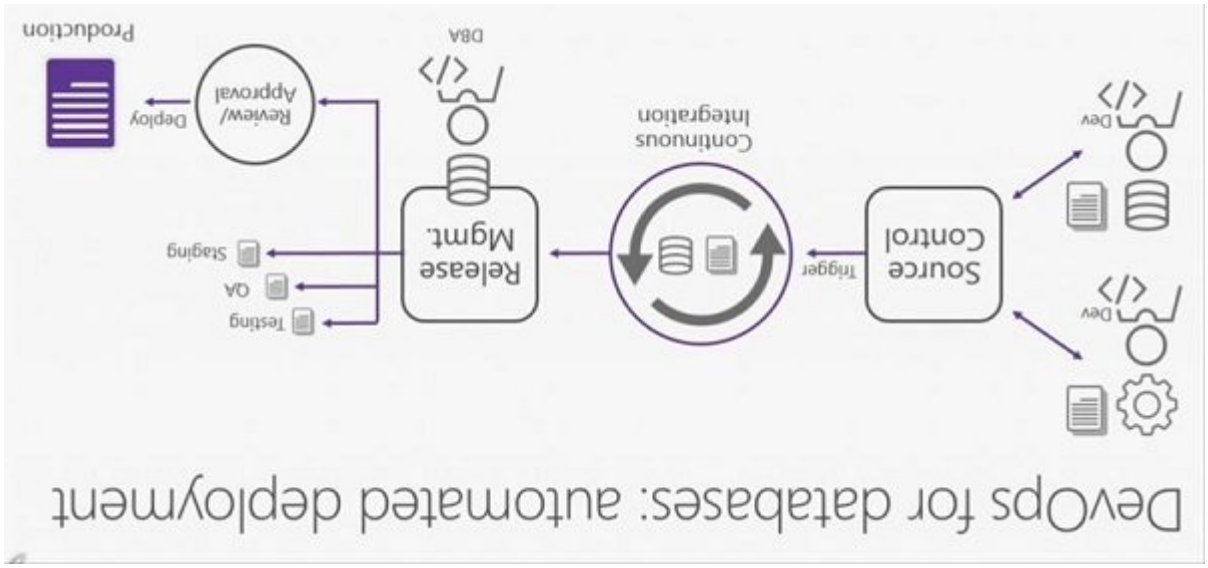


[illegible]

[illegible]

[illegible]

[illegible]


DevOps for databases: automated deployment
Source Control
Trigger
Release Mgmt.
Continuous Integration
Review/ Approval
Deploy
Production
Dev
DBA
Testing
QA
Staging

[illegible]: (1[illegible])

[illegible]

فقدان یک استاندارد واحد بین موتورهای پایگاه داده

تغییرات پایگاه داده یک استاندارد سازگار ندارد، زیرا هرکدام از موتورهای آن به روش متفاوتی برای مدیریت کردنشان نیاز دارند. از سویی تاثیر این تغییرات از موتوری به موتور دیگر تفاوت دارد. برای مثال، میزان تاثیر ایندکسها در SQL Server بهاندازه ایندکسها در Oracle یا MySQL نیست. زبان پرسوجوی ساختیافته (SQL) ممکن است تنها چیزی باشد که بین موتورهای جستوجو مشترک است. اما حتی در این زمینه هم این فرامین نتایج متفاوتی را ارائه میکنند. اما در آینده ممکن است ما شرایط راحتتری در این صنعت داشته باشیم، زیرا روش تعامل خود با پایگاههای داده را استانداردسازی خواهیم کرد. اما تا آن زمان برای چگونگی نحوه تغییرات موتور جستجوی خود برنامهریزی کنید. برای سهولت در انجام این کار میتوانید از فریم ورکهای ORM (سرنام Object-Relational Mapping) یا سایر ابزارهای موجود در این زمینه استفاده کنید.

معماریهای بههمپیوسته

اغلب اوقات مشکلات به وجود آمده در پایگاههای داده بهواسطه نحوه معماری سیستم به وجود میآید. وقتی شما یک معماری بههمپیوسته را که در مرکز آنیک پایگاه داده قرار دارد، در اختیار دارید با مشکلات اساسی بیشتری نسبت به تغییرات پایگاه داده در پیادهسازی دواپس برخورد میکنید. این روزها با متداول شدن سیستمهای توزیعشده، الگوهای معماری مثل میکرو سرویسها وجود دارد که با اختصاص یک پایگاه داده به هر میکروسرویس چنین مشکلاتی را برطرف کرده است. وقتی شما از پایگاه داده تنها برای اهداف ذخیرهسازی استفاده میکنید، تغییرات سادهتر خواهد شد. مطمئنا شما دادهها را برای تجزیهوتحلیل آنها ذخیره میکنید. به همین دلیل، در بسیاری از پروژهها دادهها به بخش تحلیل داده منتقل میشوند که بهندرت به تغییر نیاز پیدا میکنند.

فقدان فرهنگ درست برای پیش بردن درست چرخه تولید

رها کردن بازبینی تغییرات پایگاه داده در پایان چرخه کاری نشانهای از ضعف ارتباط بین گروهها است. شاید به این دلیل که گروهها اهداف مشترکی در سر ندارند یا مردم تصور میکنند در این زمینه به کمک احتیاج ندارند. شما دیگر نباید منتظر بمانید تا مسئولان پایگاه داده در مرحله نهایی به بازبینی تغییرات بپردازند، بلکه آنها باید هر زمان که لازم باشد در این فرآیند حضورداشته باشند. با گذشت زمان، توسعهدهندگان، واحد عملیات اجرایی و مسئولان پایگاه داده در مورد چگونگی تغییرات در پایگاه داده به توافق خواهند رسید و هر چه کل گروه بیشتر در فرآیند بازبینی مشارکت داشته باشند، انجام آن راحتتر و دقیقتر خواهد بود. زمانیکه یک همکاری برنامهریزیشده بین گروههای مختلف وجود داشته باشد، نتایج مثبتی نیز در پی خواهد داشت. در نتیجه شما باید آن را بهعنوان یک هدف اصلی در نظر گرفته و سایرین را نیز از آن آگاه سازید.

Developer Cloud Service
GIT | Hudson CI
Masking & Subsetting
CLOUD CONDUCTOR
ORCHESTRATES
V 1.0
Prod DB
API first
Test Master
code push to master
triggers build
V 1.1
Integration
pass
QA / UAT
Deploy
thin cloned instances
fail
Live deploy to production with blue green deployment for app and DB
Instantiate from backup
Backup
Develops on a fully orchestrated stack
t0
tn

مطلب پیشنهادی

DevOps

نگاه

دِواپس؛ چالمان هنوز خوب نیست!

راهکارهای فنی برای پایگاه داده

ما متوجه شدیم که چگونه پایگاههای داده میتوانند به یک مشکل ویژه در دواپس (**DevOps**) تبدیل شوند و چگونه میتوان اوضاع را سروسامان داد . اما چند راهکار فنی نیز وجود دارد که پیادهسازی دواپس (**DevOps**) در تغییرات پایگاه داده را تقویت میکند .

مدیریت تغییرات پایگاه داده با **Migration**

Migration، اسکریپتهای از پیش آمادهشدهای هستند که تغییرات پایگاه داده را که بهطور مطلوب پیادهسازی شده است، در خود جای دادهاند . پس تفاوتی نمیکند که شما چندبار این اسکریپت را اجرا کرده باشید، این تغییرات تنها یکبار اعمال خواهد شد . ضمن اینکه بهتر است این اسکریپتها را بهصورت نسخهبندی شده کنترل کنید تا بتوانید وضعیت تغییرات را تحت نظر داشته باشید و سادهتر این تغییرات را مدیریت کنید . بهعبارتدیگر، اسکریپتهایی که برای اعمال تغییرات پایگاه داده به وجود میآورید در قالب کدها هستند. شما میتوانید یک Migration یکسان را در محیطهای مختلف به اجرا گذارید و نتایج یکسانی به دست آورید. شروع کار نیز معمولا در محیط محلی یا همان کامپیوتر توسعهدهنده انجام خواهد شد.

تمرین در یک محیط شبیه محصول نهایی

آزمایش، یکی دیگر از راهکارهای فنی است که با وجود پیادهسازی راحت به کمی نظم و انضباط نیاز دارد . شما باید قبل از اعمال تغییرات در یک محیط محصول نهایی آن را آزمایش کنید . اگر جدول داده شما خیلی بزرگ است (آنقدر بزرگ که تکرار کردن آن در یک محیط متفاوت از محصول نهایی هزینهبر خواهد بود) ، اطمینان حاصل کنید که لااقل این تغییرات را با بخش زیادی از این داده شبیهسازی کنید . این کار به

شما کمک میکند مطمئن شوید این تغییرات زیاد زمانبر نخواهد بود و جدول شما را برای مدت طولانی مسدود نخواهد کرد . کانتینرها روش خوبی برای تمرین هستند . آنها ساده و ارزان هستند و اگر مشکلی ایجاد شود میتوانید همهچیز را دور بریزید و از ابتدا شروع کنید .

ابزار اتوماسیون پایگاه داده

نمیشود در مورد پایگاههای داده صحبت کرد و به چند ابزار مرتبط اشارهای نکرد . ابزارهای زیادی برای این منظور وجود دارد و هر از گاهی نمونههای جدیدی از آن منتشر میشود . در ادامه چند نمونه از معروفترین آنها آورده شده است :

- Liquibase (رایگان)
- Datical (نسخه غیر رایگان Liquibase)
- Redgate (Microsoft Stack)
- Delphix (بیشتر از تنها تغییرات پایگاه داده)
- DBmaestro (در واقع بهعنوان دواپس (**DevOps**) برای پایگاههای داده فروخته میشود)

علاوه بر ابزارهایی برای موتورهای پایگاه داده، فریم ورکهایی نیز وجود دارد که از migration پشتیبانی میکنند :

- Entity Framework
- GORM
- Compose
- Hibernate

تغییرات پایگاه داده را به مراحل عقبتر منتقل کنید

همانگونه که مشاهده کردید، هیچ فرمول جادویی وجود ندارد که شما بتوانید با استفاده از آن دواپس (**DevOps**) را برای پایگاههای داده پیادهسازی کنید . عوامل بسیار زیادی در این امر دخیل هستند، اما نخستین قدم این است که تردید و تنبلی را کنار گذاشته و به اوضاع سروسامان دهید . وقتی بحث بر سر دادهها است، تغییر را با آغوش باز بپذیرید، اگر چه این کار هراسانگیز است، سعی کنید همهچیز را از فرآیند تا معماری تا حد ممکن ساده نگه دارید . تغییرات در پایگاه داده بهخودیخود کار دشواری نیست، مشکل اینجا است که آسیبرسان به تمام یا بخشی از دادهها همیشه بهطور بالقوه وجود دارد . تکرار و ثبات، دو نکته کلیدی در موفقیت شما محسوب میشود، به همین دلیل است که قبل از تولید محصول نهایی به تمرین نیاز دارید .

نویسنده :
محسن آقاجانی
دسته بندی:
کارگاه
تاریخ انتشار :
11:20 - 21/08/1397
برچسب:
دواپس - پایگاه داده - Devops - مدیر پایگاه داده - اتوماسیون - بانک اطلاعاتی

نشانی
□□□□:https://www.shabakeh-mag.com/workshop/13981/%DA%86%DA%AF%D9%88%D9%86%D9%87-devops-%D8%B1%D8%A7-%D8%A8%D8%B1%D8%A7%DB%8C-%D9%BE%D8%A7%DB%8C%DA%AF%D8%A7%D9%87%E2%80%8C%D9%87%D8%A7%DB%8C-%D8%AF%D8%A7%D8%AF%D9%87-%D9%BE%DB%8C%D8%A7%D8%AF%D9%87%E2%80%8C%D8%B3%D8%A7%D8%B2%DB%8C-%DA%A9%D9%86%DB%8C%D9%85%D8%9F