2020 State of Database DevOps

Compliant Database DevOps

redgate

2020 State of Database DevOps

Contents

What is Database DevOps?

Donovan Brown

DevOps is the union of people, process, and products to enable continuous delivery of value to our end users. The most important word of this definition is value. To deliver value you must deploy all changes, including those to your database.

Foreword

Software development is changing rapidly and with this IT departments are changing everywhere. To win market share, all businesses, from small startups to large Enterprises, are increasing optimization in order to improve products and services for their customers. The most popular approach to increase this flow of value is DevOps.

DevOps initially found its way into organizations in the software development lifecycle for application components. DevOps has spread beyond the scope of applications into every aspect of infrastructure.

Databases, which were formerly seen as configurable repositories much like file shares, are now seen as stateful data, controlled by code. This evolving viewpoint is transformative: changes to databases are recognized as another code deployment which can and should be managed, tested, automated, and improved with the same robust, reliable methodologies applied to application code.

However, databases present unique challenges in using change methodology due to the very nature and value of data. In particular:

- Complex data recovery can create roadblocks to more modern change flow practices.
- If less traditional change methodologies are used unsuccessfully with databases, it causes significant financial vulnerability.
- Damage of business-critical data can impact complex data ecosystems long-term.
- Data loss can put the continued existence of the company at risk.

This results in database deployments requiring more care than standard change management to ensure the code that's deployed and data quality aren't impacted. The natural state of a database environment is growth, so as the surface area naturally spreads, the challenge of change management and releases becomes more daunting.

Developers and database specialists are often working across multiple platforms and technologies too. Responders to the 2020 State of Database DevOps survey report that their organizations most commonly use variations of Microsoft SQL Server, Oracle, MySQL / MariaDB, and PostgreSQL, in various on-premises and PaaS/IaaS cloud environments.

The findings from this vital report will help any team interested in adopting or improving their approach to Database DevOps, no matter what database platform or technologies they choose to work with.

In the report, you will learn multiple tactics to improve the quality of database code deployment, whether it is by changing the type of environment used for development, enhancing your code review practices, or improving your change management or approval practices.

If you have not yet launched your DevOps initiative, survey results will help you shape your business case with the decision makers on your team.

There's a significant chance your competitors have already adopted DevOps and are working to improve. It is not too late for you to begin: the important thing is to act before your company is unable to catch up.

Kellyn Pot'Vin-Gorman

Customer Success Engineer, Microsoft

Key findings

We are excited to share the findings from the fourth annual State of Database DevOps Survey. More than 2,000 people responded to the survey this year, from all industries and from around the world, doubling the number of participants from last year.

We see indications of increasing DevOps adoption in survey responses this year: 68% of responders are in some phase of DevOps adoption. 17% of respondents in Enterprises, which we define as organizations with 1,000 or more employees, reported that DevOps has been adopted across all projects in the organization - an impressive metric that shows DevOps is not only spreading, but is being fully adopted even by larger organizations.

In this report, we bring you insights which will help you inform the business cases you make for implementing or improving DevOps, a view of how the roles of Developers and DBAs continue to shift, and a deep dive into change management practices for database development.

Key findings in the report include:

- Slow development and release cycles and the inability to respond quickly to changing business requirements are now seen as the top two drawbacks of siloed database development.
- Frequent database deployments are increasing: 49% of respondents now report they deploy database changes to production weekly or more frequently.
- Frequent deployers who use version control report lower production defect rates.
- Respondents who report that it is easy to get a code review for database changes report lower production defect rates and lower lead time for change deployment.
- Although 38% of responders report the use of Change Approval Boards, we see no evidence that Change Approval Boards lower code defect rates, only that they increase lead time for changes.
- Respondents who reported that all or nearly all their database deployments take place with the system online also reported lower lead time for changes and lower defect rates.
- 60% of Enterprise respondents believe the move from traditional database development to a fully automated process for deployment can be achieved in a year or less. 66% in non-Enterprises believe this can be accomplished.

Making a successful business case for Database DevOps

Luke Poretta

Creating an effective business case for Database DevOps requires understanding what motivates and concerns your audience, as well as the benefits and hurdles of implementation. This report reveals insights in all these areas.

Motivations and benefits

When addressing reasons to make a change, consider how slow development and release cycles and the inability to respond quickly to changing business requirements impact your organization. These are the top two downsides respondents see in traditional siloed database development practices.

Effective business cases should also document the current speed of delivery of database changes in the organization, and track improvements seen in proof of concept or experimentation. This is the top driver overall for adoption of Database DevOps, and it is especially important to address for those in Financial Services, Media, and Retail.

Another important metric which will add value to your business case, especially in the Healthcare, Medical, and Pharmaceutical industries, is the amount of time Developers have for value-added work.

Concerns and hurdles

Business cases should propose how to overcome obstacles related both to training and upskilling and to disruption to workflows. It is most critical to help with upskilling just before and during a proof of concept, when concerns in this area are the strongest.

As you complete the proof of concept, your plan should be to continue to share skills more broadly in the organization, but increasingly focus on reducing disruption to workflows while enabling change. Expect that your team will need to address questions about how to synchronize application and database changes, how to overcome different approaches to application and database development, and how to protect and preserve business critical data.

Working in an Enterprise

Respondents in Enterprises were notably more likely to cite lack of alignment between development and operations as a top concern, compared to respondents in smaller organizations.

When writing a business case in an Enterprise, consider addressing concerns about how to work across entrenched silos of development and operations if there is a history of strong divisions in those areas. Additionally, the ability to more quickly respond to changing business requirements is a more significant motivation in these larger organizations.

Database DevOps adoption and its impacts

Database DevOps adoption and defect rates

We will be cautious in this report about making claims regarding overall DevOps adoption rates: our study has “DevOps” in the title, so our group of respondents may naturally be more likely to be DevOps adopters.

However, some trends may imply a growth in the popularity of DevOps, and at least indicate continuing popularity of these ways of working:

- 18% of survey respondents now say they have adopted DevOps across all projects, compared to 15% of respondents in 2019.
- 17% of respondents in Enterprises reported that DevOps has been adopted across all projects, an impressive statistic for these larger organizations.

We asked respondents, “Has your organization already adopted, or do you plan to adopt, a DevOps approach across any of your IT projects?” This year we added a new option so that those who are actively in the early stages of adoption and who are working on a proof of concept or experimenting with DevOps could identify themselves.

Has your organization adopted, or do you plan to adopt, DevOps across any of your IT projects?

Yes

36% Adopted across some projects

18% Adopted across all projects

14% In Proof of Concept or experimentation

Not yet

18% Planning to adopt within two years

13% No plans to adopt within two years

Non-adopters

In our very first survey, 20% of respondents said their organizations had no plans to adopt DevOps. That dropped to 15% of respondents in the 2019 study, and now in 2020 only 13% of respondents report that their organizations have no plans to adopt.

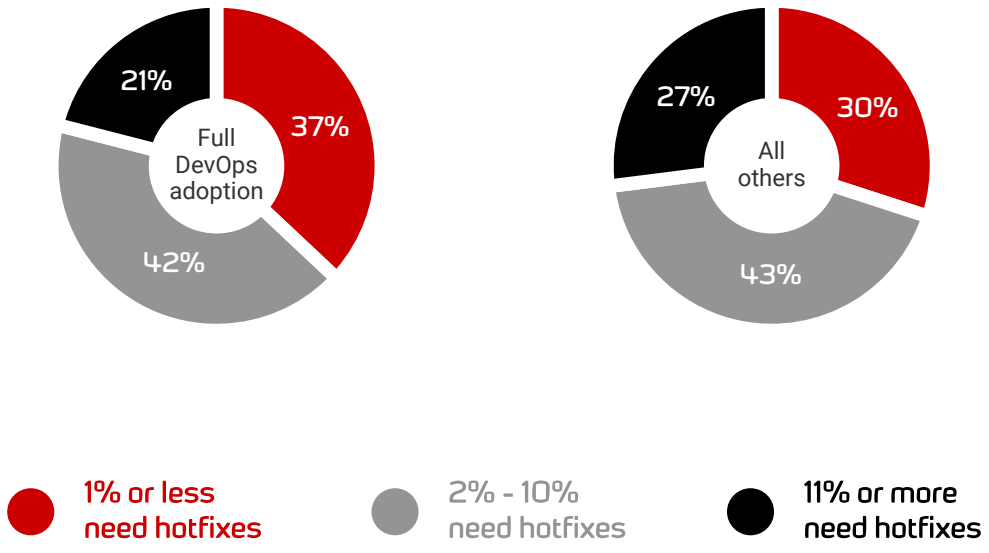
Industry sectors

Responders in the Financial Services and Insurance industries reported the highest levels of adoption this year, with more than 77% of respondents saying their organizations are either in proof of concepts (16%), have adopted across some projects (41%), or have adopted across all projects (20%). Overall adoption rates were near 70% for IT, Technology, Healthcare, Medical, Pharmaceuticals, Media, and Retail.

Enterprises

72% of those in Enterprises reported some level of DevOps experimentation or adoption for their organization, compared with 66% of those in non-Enterprises. 17% of those in Enterprises reported that DevOps has been adopted across all projects, compared to 18.5% of those in non-Enterprises – an impressive feat for the larger organizations.

How many deployments need hotfixes?



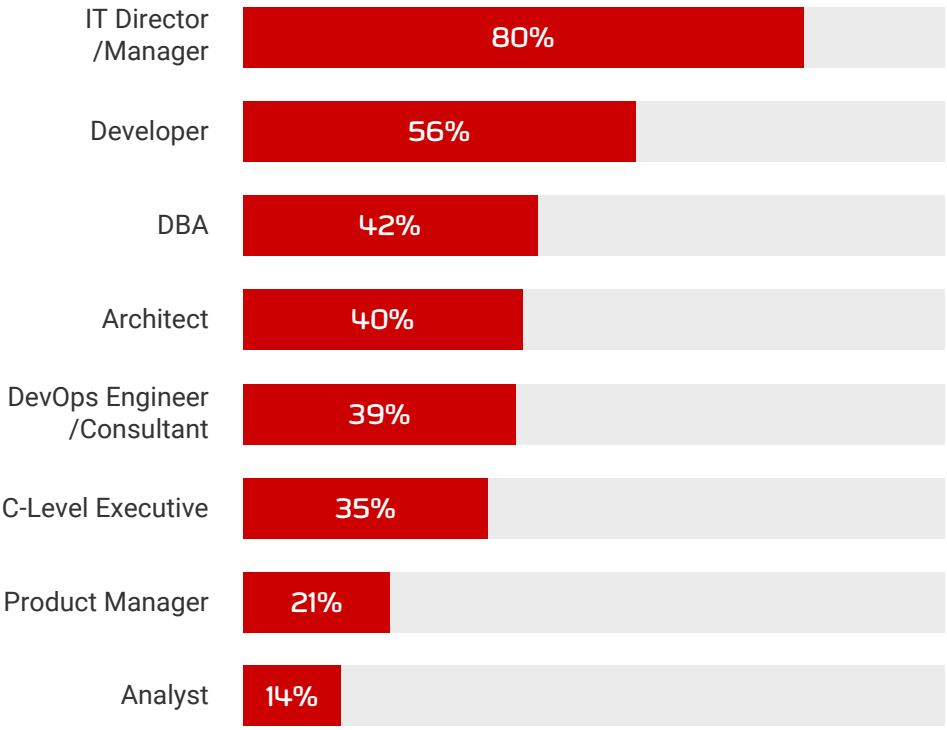
Full adopters report lower defect rates

37% of those who have adopted DevOps across all projects report that 1% or less of their deployments introduce code defects which require hotfixes, compared to 30% for all other groups.

Those who plan to adopt DevOps within two years, and those working on proof of concept or experiments are the most likely to report that more than 10% of deployments have defects which require hotfixes. This may indicate that DevOps is being implemented to reduce defects, or it may indicate increasing levels of awareness of tracking metrics related to code quality in these organizations.

Top decision makers for DevOps implementations

Who's involved in deciding to implement DevOps in your organization?



Respondents could give multiple responses to this question.

While the top four stakeholders involved in this decision were the same as in the 2019 survey, responses changed in an interesting way.

In the 2019 study, 67% cited the involvement or IT Directors / Managers, and this has risen to 80%, indicating that IT Directors / Managers are now nearly always at the heart of decisions about ways of working, like DevOps. Reported involvement for Developers, DBAs, and Architects also went up.

Industry differences

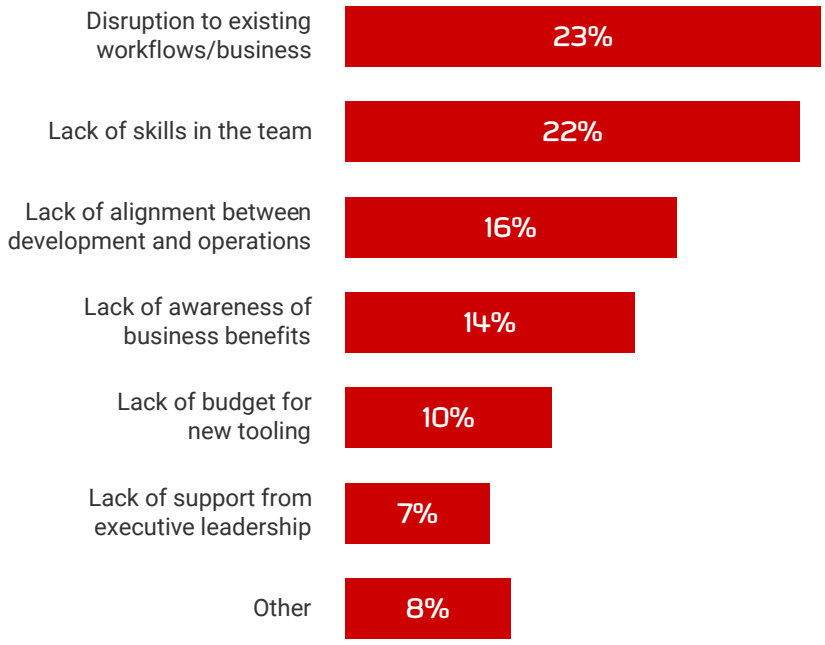
As in 2019, the ratios of stakeholders involved remains similar when analyzed by industry.

Enterprises

Respondents in Enterprises were somewhat more likely to report involvement of a DevOps Engineer / Consultant in the decision-making process, and slightly less likely to report involvement of Developers, as compared to non-Enterprises. This likely reflects different strategies taken in spreading cultural change in larger companies.

Primary obstacles to implementing DevOps

What was or what would be the main obstacle to implementing a DevOps approach in your organization?



We have seen some slight overall shifts in responses compared to the 2019 report:

- ***Disruption to existing workflows/business*** squeezed into the #1 spot, up from 21% in the 2019 report.
- ***Lack of appropriate skills in the team*** was selected by 22% overall in the 2019 report as well. We'll call this concern "upskilling" for brevity.
- Respondents were slightly more likely to be concerned about alignment between development and operations this year, and slightly less likely to be concerned about lack of awareness of benefits to the business.

We once again found that concerns varied between responder groups. We were able to get more granular insights in the 2020 study, as we are now able to segment those who are currently working through proof of concepts.

The early anticipators

Those who haven't adopted DevOps but are planning to adopt across some or all projects within two years were the most "across the board" with their answers.

This group was the most likely to answer that they see the biggest obstacle as being support from executive leadership, but that was still only 12% of the "early anticipator" group (compared to 6% citing executive support as their top concern across all responders).

Overall, this group is nearly equally concerned about upskilling (21%) and disruption of workflows (22%), their top two obstacles.

Those proving the concept

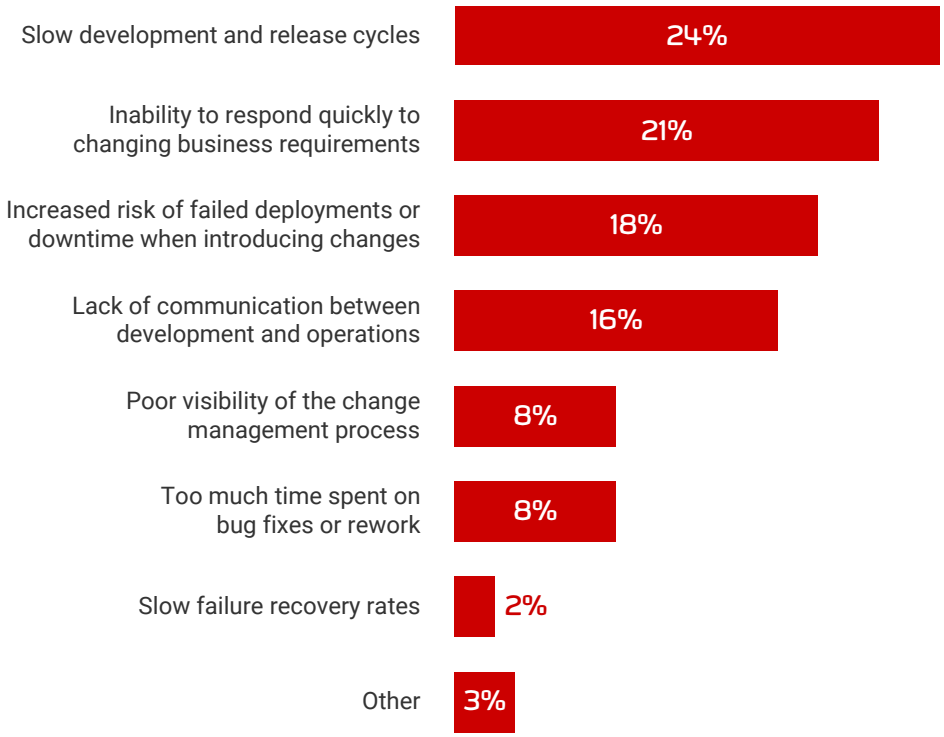
Our respondents who said they are currently experimenting with DevOps or working through proof of concept exercises were the most likely to be concerned with upskilling, with 30% citing it as their top concern.

Adopters across some or all projects

For respondents who have adopted DevOps across some projects, concerns about upskilling have dropped a bit: 26% cite it as their top concern. When we look at the respondents who have adopted DevOps across all projects, the number citing upskilling as a top concern goes down further to 23%.

Shifting views on the primary drawback of siloed database development

What do you consider the greatest drawback in traditional, siloed database development?



In the 2019 report, the top answer was ***increased risk of failed deployments***, with 23% of respondents selecting this option. Our prior studies saw a similar pattern. In the 2020 report, we see the risk of failed deployments falling to the third most popular answer.

The new top perceived drawbacks of siloed database development are ***slow development and release cycles*** and the ***inability to respond to quickly changing business requirements***. In previous years, these concerns were ranked #2 and #3, respectively.

This change in ranking of the concerns indicates an interesting mindset shift among the population responding to our survey: primary concerns have shifted away a bit from simply being able to release successfully and reliably and have moved toward being able to develop and release quickly in accord with business requirements.

Top concerns by industry segments

When grouping our respondents by industry, we see some variation between the top downside selected:

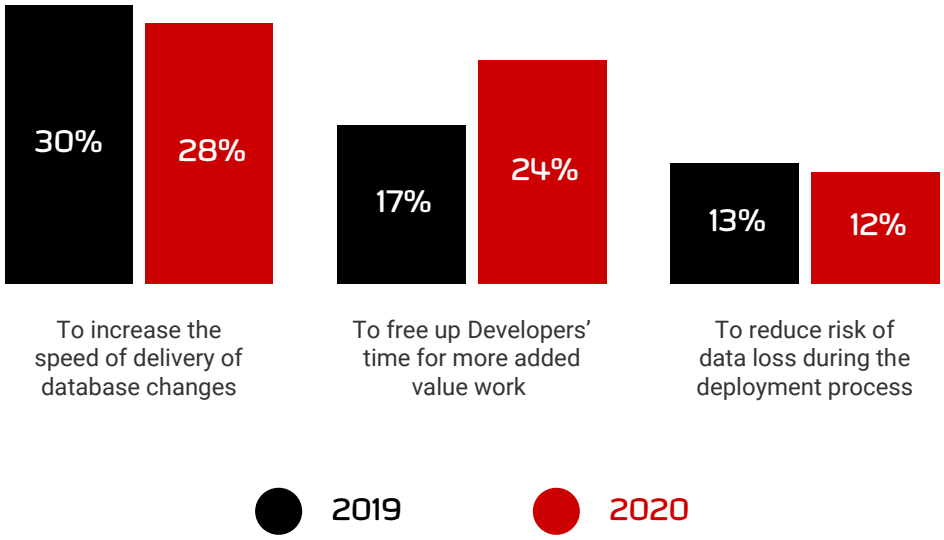
- 27% of those in IT & Technology and 24% of those in Financial Services selected ***slow development and release cycles*** as their top downside.
- 18% of those in Media & Retail and 25% of those in Government, Education, or Non-Profits selected ***inability to respond quickly to changing business requirements*** as their top downside.
- 24% of those in the Healthcare, Medical, or Pharmaceutical segment selected the ***increased risk of failed deployments and downtime*** as the top downside they saw.

Enterprises vs non-Enterprises

Those respondents who are in Enterprise organizations were more likely than those in smaller organizations to list ***inability to respond quickly to changing business requirements*** as the top downside they see in siloed database development practices.

Drivers for adopting Database DevOps

What would be the main driver for automating delivery of database changes as part of DevOps?



The top three answers selected by respondents follow the same pattern as those selected in our 2019 survey, but concern has risen notably in one area.
This change may reflect an increasing pressure or desire to deliver greater value to end users than respondents have felt in previous years, or it may reflect an increasing perception that DevOps is primarily a means of achieving the faster delivery of value.

Industries perceive the top drivers for adopting Database DevOps differently

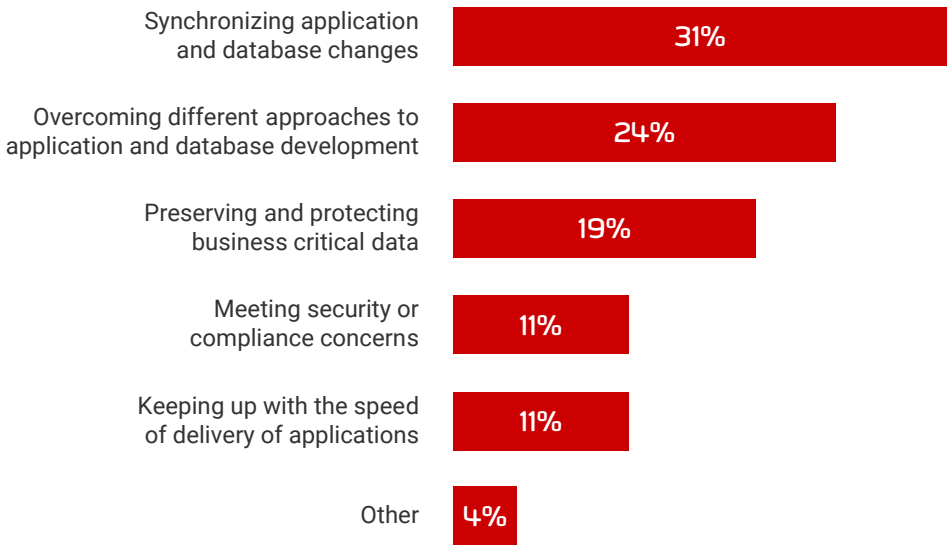
- 35% of those in Financial Services and 32% of those in Media or Retail selected ***increasing the speed of delivery of database changes*** as the top driver, showing that the ability to deliver changes quickly is notably more critical for these segments than the general population.
- 28% of those in the Healthcare, Medical, or Pharmaceutical industries selected ***freeing up developer time for value added work***, making it the top driver for this segment.

Leadership is even more concerned with speed of delivery

32% of those who identified themselves as Directors or Managers, and 38% of those who identified themselves as C-Level Executives, selected increasing the speed of delivery of database changes as the top driver for Database DevOps, as compared to the 28% of individual contributors who selected this as their top concern.

Challenges when integrating database changes into a DevOps process

What do you consider the greatest challenge in integrating database changes into a DevOps process?



Responses overall follow very similar patterns to what we saw from the 2019 study.

The top two selected answers are ***synchronizing application and database changes***, primarily a timing and technical concern, and ***overcoming different approaches to application and database development***, primarily a process and cultural concern.

We see some variations in the top perceived challenges by industry

- Those in the IT & Technology segment were the most concerned about ***synchronizing application and database changes***, with 35% of this group selecting it as their top concern.
- Those in Financial Services had a tie for their top concern: 31% selected ***synchronizing application and database changes***, while another 31% selected ***overcoming different approaches to application and database development***.
- The top challenge selected by respondents in the Healthcare, Medical, and Pharmaceutical industries was ***overcoming different approaches to application and database development*** (28%).

Company size is also a factor

Employees at Enterprises are almost equally concerned with ***overcoming different approaches to application and database development*** and ***synchronizing application and database changes***.

This perhaps illustrates the struggles that larger and more regulated companies specifically have in changing culture and entrenched processes.

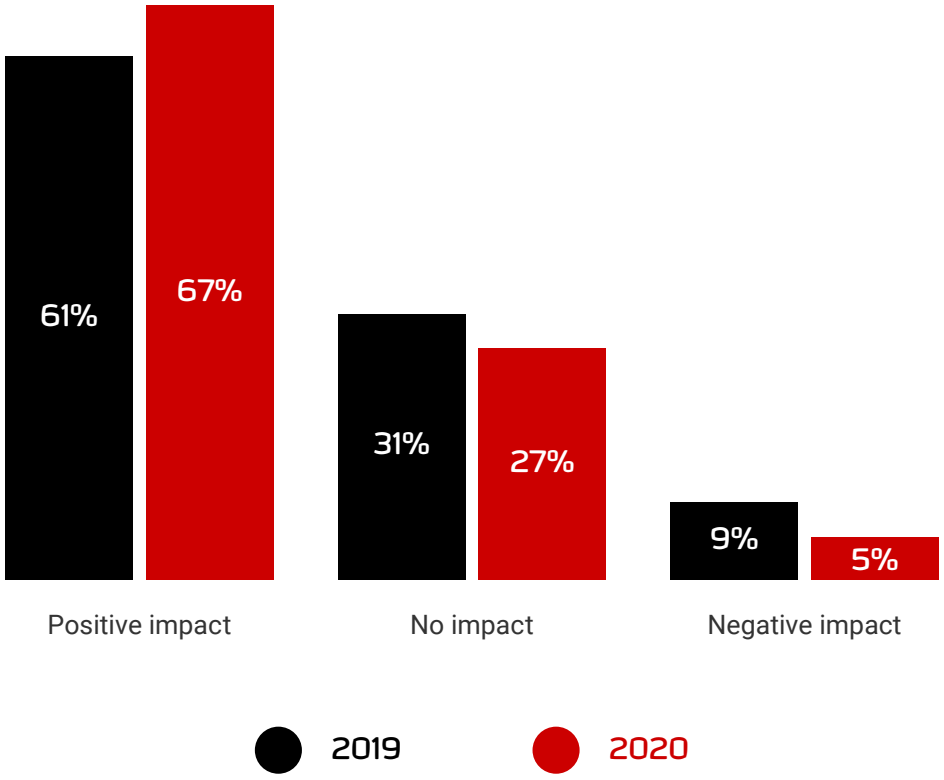
Insights from those who have adopted across all projects

Those who have adopted Database DevOps across all projects were 6% **more** likely to select ***keeping up with the speed of delivery of applications*** and 6% **less** likely to select ***overcoming different approaches to application and database development*** than other segments. This may indicate that some full adopters have found that the technical difficulty of database changes remains a challenge.

However, 34% of the full adopter segment selected ***synchronizing application and database changes*** as the top challenge, making it the top concern for this group as well as for the overall population of survey takers.

Database DevOps is perceived to have an increasingly positive impact on compliance

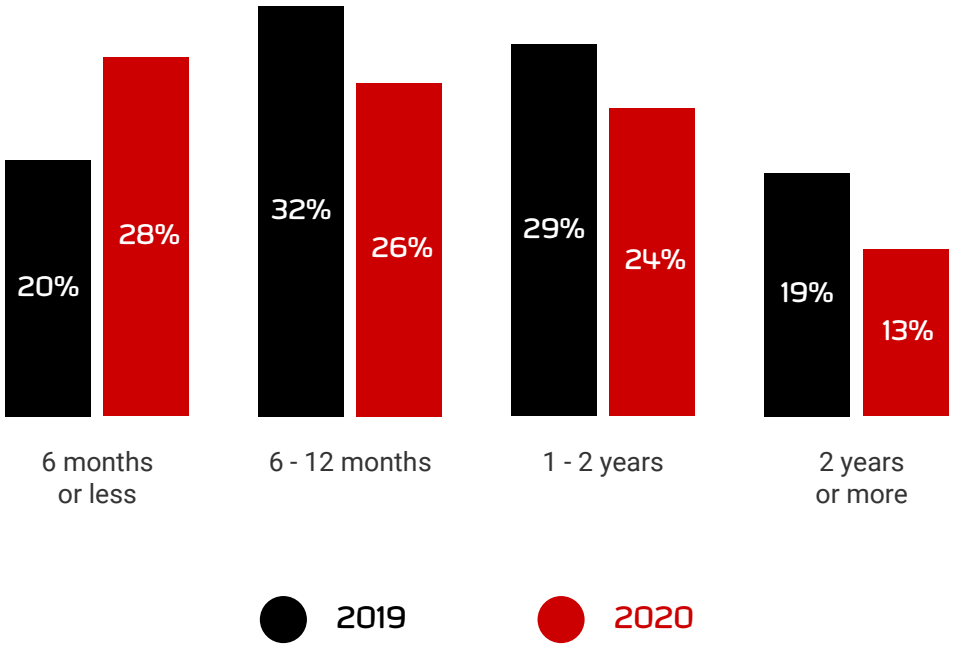
What impact do you think a DevOps approach to the database would have on regulatory and compliance requirements?



Overall responses follow a similar pattern to 2019, but more people see a positive impact. A similar pattern is seen across all verticals and for managers and individual contributors.

Estimated time to implement a fully automated process for deploying database changes

How long do you estimate it would take your team to move from traditional database development to a fully automated process for deploying database changes?



Responses are more optimistic this year, with a notably higher percentage of people reporting that this can be done in either less than six months, or within 6 - 12 months as compared to responses from last year.

Speed estimates vary by industry

- 22% of those within the Government, Education, and Non-Profit sectors believe that this move would take more than two years – the highest of any group.
- 71% of those in Financial Services or Insurance and 70% of those in Media or Retail believe that this move can be done in a year or less, the highest of any segments.

Enterprises

Those at Enterprises are only somewhat more pessimistic than those in smaller organizations. 60% of Enterprise respondents believe the move from traditional database developments to a fully automated process for deployment is possible in a year or less, compared to 66% in non-Enterprises.

The way Developers and DBAs work continues to shift

Team integration between Developers and DBAs

How closely are Developers and DBAs integrated in the teams you work with each week?

12% **Poor** - DBAs are completely siloed, unaware of what's happening in development, and deploy changes to production in isolation

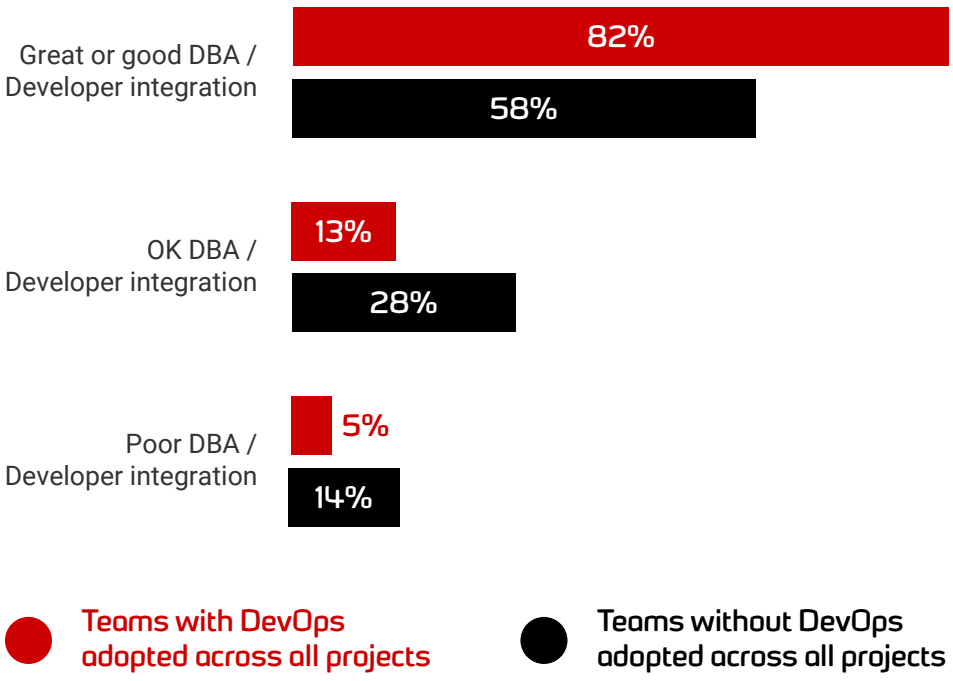
23% **Great** - all work together effectively as one team, using standardized processes and tools

25% **OK** - DBAs aren't really involved and only work with development when it's time for production deployment

40% **Good** - teams collaborate, with DBAs involved on an ad hoc basis when needed

The responses were broadly similar to prior surveys. The total number of respondents who say there is either good or great integration between these teams remained steady at 62%.

How closely are Developers and DBAs integrated in the teams you work with each week?



DevOps adoption status and team integration

As in 2019, teams who have adopted DevOps across all projects reported the strongest integration between these teams. In the 2020 survey, 43% of responders in this segment reported the two groups work effectively as one team, with 82% citing either good or great integration.

Teams who have no plans to adopt DevOps have 27% who report great collaboration between Developers and DBAs. This group also has the highest proportion of respondents who cite poor integration, with 19% of responders in this segment reporting that DBAs are completely siloed, unaware of what's happening in development, and deploying database changes to production in isolation.

Organization size and team integration

The proportion reporting good integration between teams is roughly similar overall when comparing those in Enterprise organizations with others.

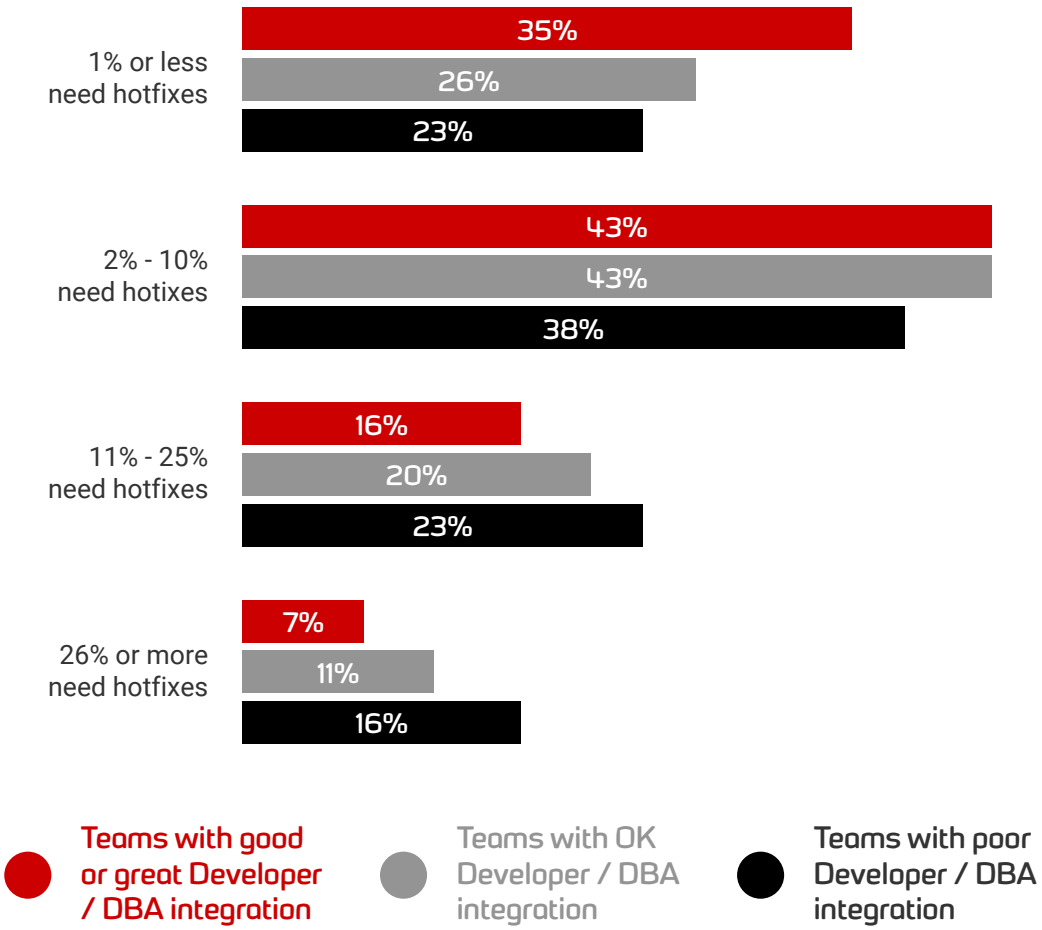
Respondents are most likely to report great integration between teams at non-Enterprises, and there is a sweet spot at 100 people or fewer in the organization, where 38% report great collaboration.

Interestingly, more people report poor integration between development and DBA teams at organizations with 5,000-10,000 employees than those at more than 10,000 – perhaps showing that at a certain scale companies are solving some of the problems of collaboration in a large organization.

Team integration and code quality

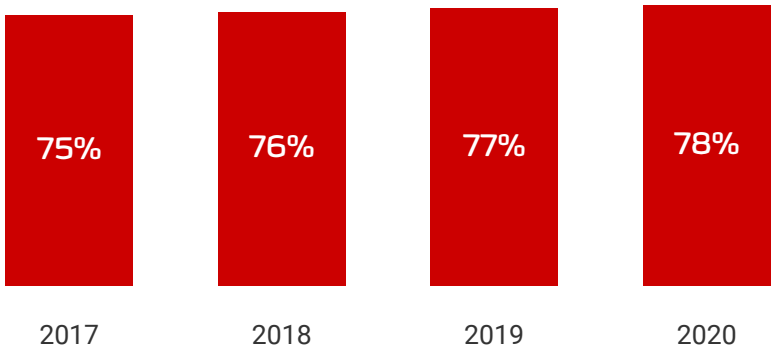
Those who report good or great team integration between Developers and DBAs report a lower percentage of deployments which require hotfixes or repair than other groups.

How many deployments need hotfixes?



Developers increasingly work across databases and applications

Percentage of companies where Developers are responsible for both application and database development



We have seen a small but consistent increase in the percentage of Developers working across applications and databases over the history of the survey.

Roles by industry

Respondents in the Healthcare, Medical, and Pharmaceutical industries reported the highest percentages of specialist Database Developers, with 28% of respondents in that group saying their team has dedicated Database Developers.

Enterprises and specialization

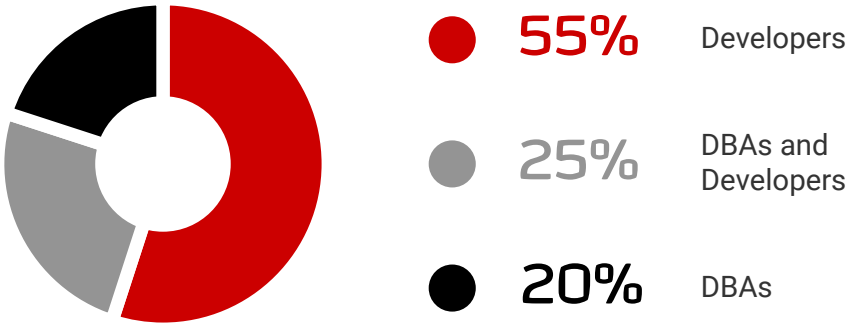
25% of respondents in Enterprises reported that their team has dedicated Database Developers, as compared to 19% in smaller organizations.

Takeaways

While there are still positions for specialist Database Developers, the number of these specialists appears to be declining rather than increasing.

Who authors and deploys database changes?

Who authors deployment scripts?

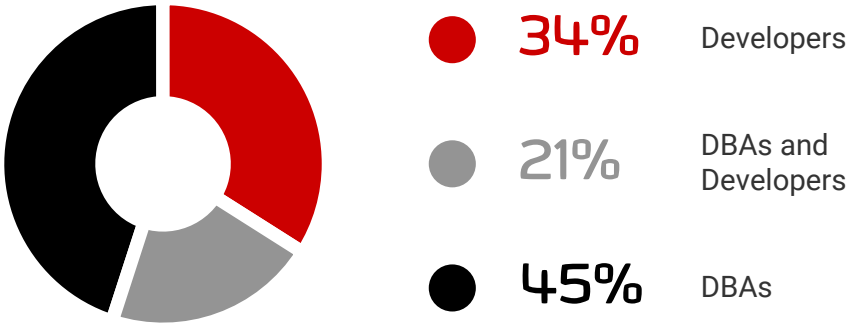


80% of respondents reported that Developers are involved in authoring database changes in some way, up from 75% in the 2019 study.

DevOps adopters are less likely to have DBA involvement in authoring changes

Those who have not adopted DevOps and have no plans to adopt within the next two years were the most likely to have database change scripts authored by DBAs, with only 48% of respondents in this category reporting that only Developers author their change scripts.

Who deploys to production?



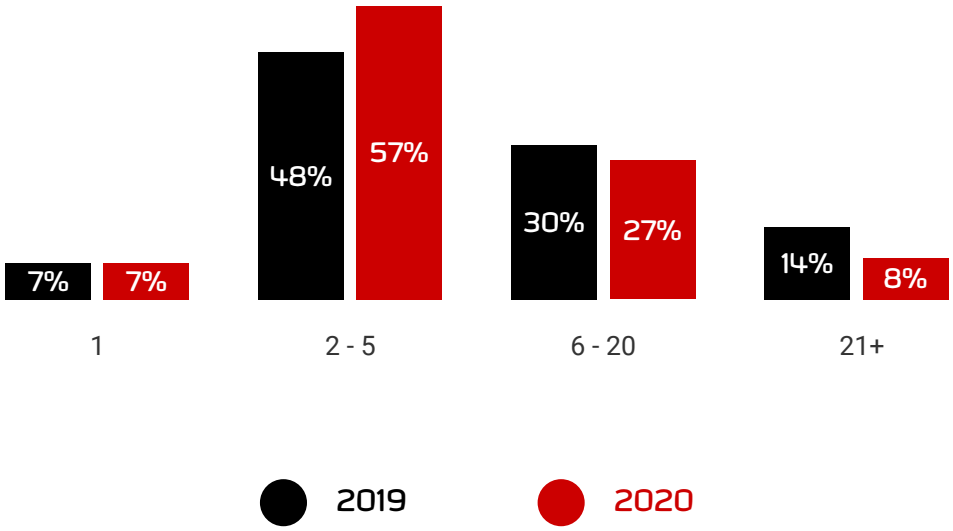
Industry shifts

Looking at responses by industry, some interesting patterns are visible. In the Healthcare sector, 59% of respondents reported at least some Developer involvement in production deployments, higher than the overall respondent average of 55%.

Only 48% of respondents in the Financial Services sector reported Developer involvement in production deployments, suggesting that divisions between roles remain more prevalent in some sectors than others.

Team sizes for database development

On average, for a given database, how many people work on its codebase?



Industry insights

Financial Services has the highest proportion of teams where more than 20 people work on a single database codebase (12%), with Healthcare, Medical, and Pharmaceutical close behind (11%), and Media and Retail a near third (10%).

Enterprises

Respondents in Enterprises were more likely to report a larger team working on a single database codebase. 11% of Enterprise respondents report more than 20 people working on a single database's codebase on average, compared to 6.5% for non-Enterprises.

Defect rates

Respondents in smaller teams were more likely to report that their deployments are unlikely to introduce defects requiring hotfixes.

For teams of 1-5 people, 75% reported that 1% or less of deployments required hotfixes, compared to 71% of teams sized 6-20 and 69% of teams with 50 or more contributors to the codebase.

Lead time

65% of people on teams smaller than five report a lead time of one week or shorter, compared to 60% of those on teams sized 6-20, and 56% of those on teams of 50 or more.

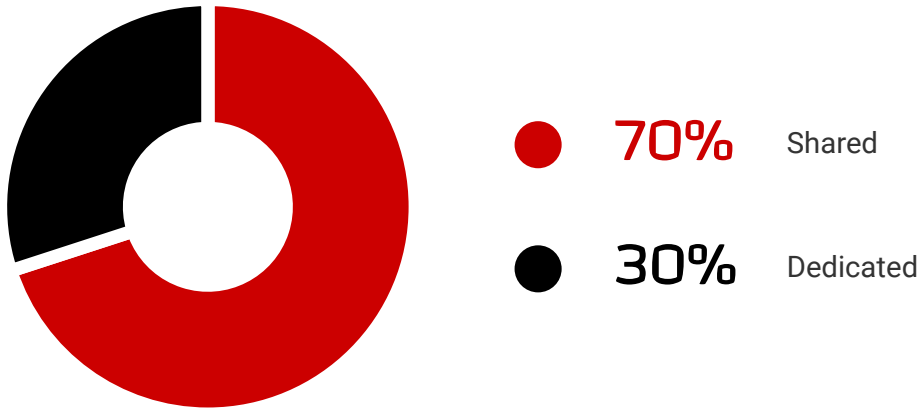
Shared vs dedicated databases for development

We modified the way we asked this question in an important way this year. In prior years, we have asked respondents if they use shared database environments, dedicated database environments, or a combination of both.

This year, we instructed the person to select either the shared model or the dedicated model only, and to "think of the project with the greatest business value to your organization which has active database development".

We made this change to find which practices are in place for the most critical databases, where having dedicated environments would make the most impact.

Do you use shared or dedicated database environments?



Industry insights

- The responders most likely to have dedicated development environments for high-value databases work in Media or Retail (36%), IT or Technology (35%), and Financial Services (29%).
- Responders working in the Industrial or Manufacturing segment are the least likely to have dedicated development environments (20%).

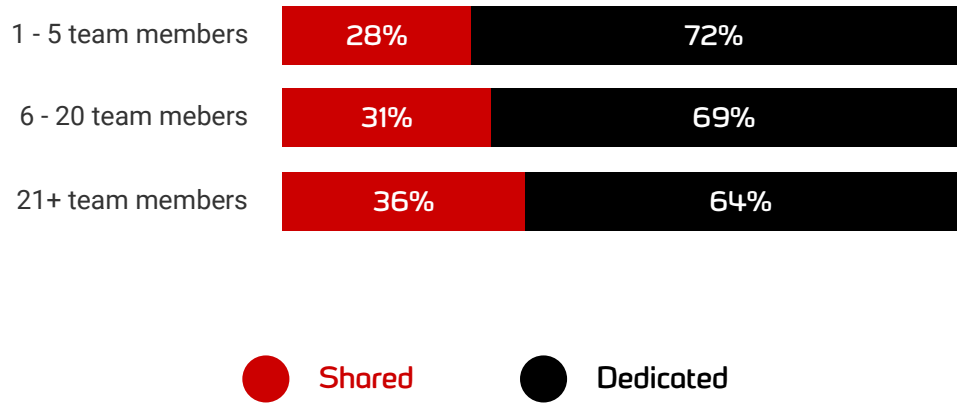
Enterprises

Although one might think that Enterprises would be more likely to have resources to support dedicated development environments, Enterprises lag behind. Only 26% of respondents in Enterprises report using dedicated database environments for their most business-critical database, whereas 32% in non-Enterprises have this practice.

Team sizes

As teams grow larger, they are more likely to use dedicated environments.

Do you use shared or dedicated database environments, grouped by team size?



This pattern very likely relates to the fact that using a shared development environment becomes increasingly confusing and more likely to cause errors as the number of people working on a codebase grows.

Code quality

Teams using dedicated environments reported a lower percentage of production deployments which cause defects that require hotfixes:

- 32% of those using dedicated environments report 1% or less of deployments require hotfixes, compared to 29% of those using a shared environment.
- 26% of those using shared environments report that more than 10% of deployments require hotfixes, compared to 22% of those using dedicated environments.

This may indicate that dedicated environments help create higher quality code or prevent frequent accidents during changes. It may also be the case that organizations which are likely to provide dedicated database environments are likely to follow other practices which also contribute to higher quality code.

De-identification and masking of production datasets in development environments

62% of responders told us that data in their development and test environments is a copy of production.

For those using copies of production data for development who reported masking status, 58% reported that all sensitive data is de-identified or masked and 42% say that at least some sensitive data is not masked.

Enterprises

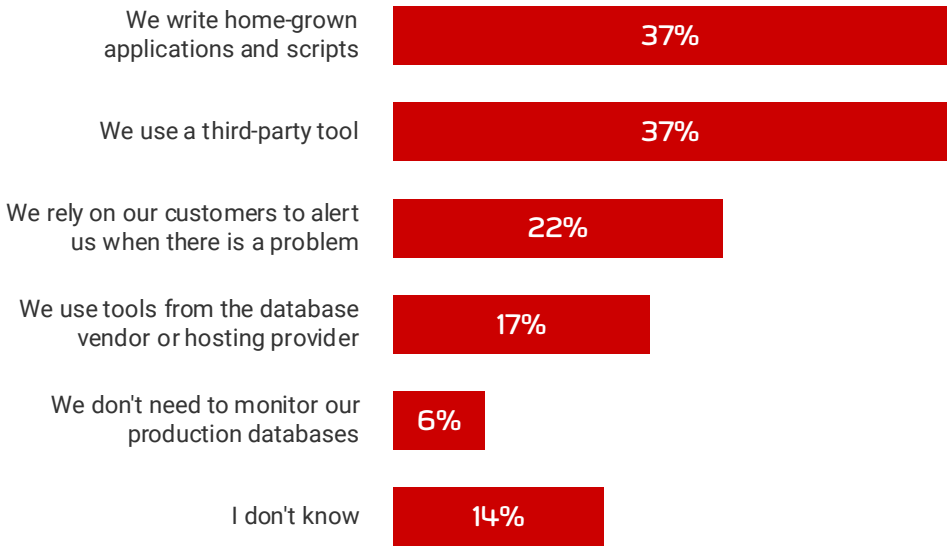
Enterprises appear to have larger amounts of un-masked data in development and test environments. For those who report using copies of production data, 48% of those in Enterprises reported that at least some sensitive has not been masked, compared to 37% of those in non-Enterprises.

While it might seem that Enterprises would be more able to afford solutions for data masking that would scale across teams, the slower moving nature of Enterprises may be responsible for them lagging in this area.

Monitoring methodologies

We asked survey takers how their production databases were currently monitored for availability, performance, and other issues. It is common for multiple approaches to be taken to monitoring, so multiple responses were allowed for this question.

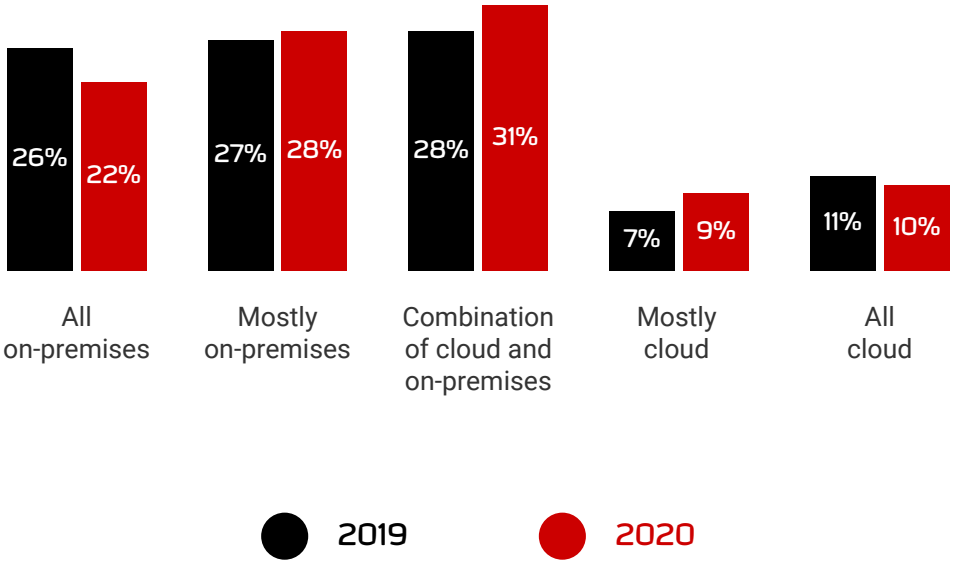
How do you monitor your production databases for availability, performance, and other issues?



Notably, 22% of all responders indicated that their organization relies on customers to alert them when there is a problem.

Cloud adoption

Where are your production databases hosted?



We see a small drop in the number of responders who report being a combination of cloud and on-premises, and a slight rise in those who say they are all on-premises between last year's survey and this year's survey.

DevOps adopters

Those who have adopted DevOps across some or all projects or who are experimenting with DevOps were less likely than non-adopters to say that their databases are hosted entirely on-premises. 23% of adopters reported this vs 36% of those who have no plans to adopt. 13% of DevOps adopters responded that their databases are entirely hosted in the cloud, compared to 5% of those who reported no plans to adopt.

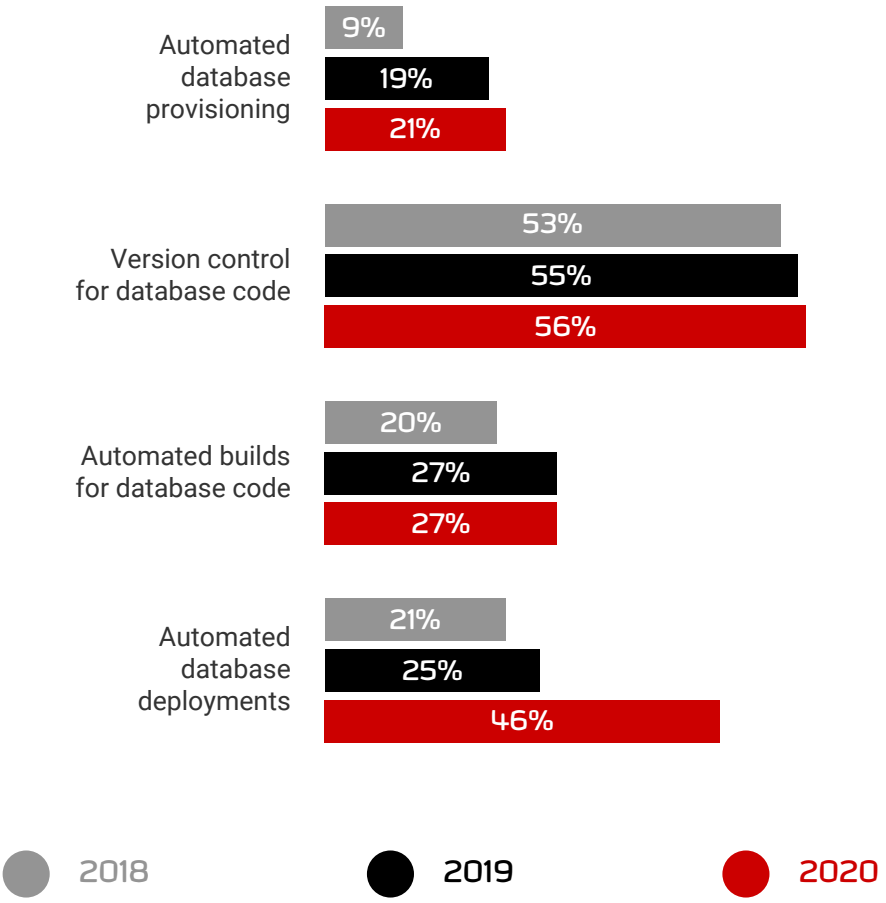
Enterprises

35% of responders in Enterprises reported that their databases are hosted mostly on-premises, compared with only 22% of those in smaller organizations. However, 31% of those in Enterprises reported that they use a combination of cloud and on-premises databases, compared to only 26% of those in smaller organizations. Enterprises were no more likely to report hosting databases entirely on-premises than other responders.

Shifts in key practices for database development

We asked respondents about software development practices. Here are the broad high-level trends for four key practices over the last three years.

Which, if any, of these practices are already in place for your application or database development?



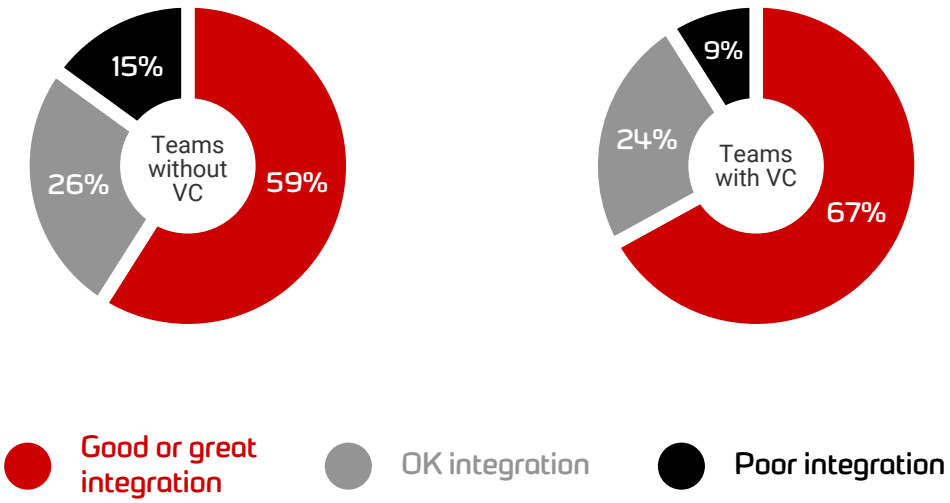
Automated provisioning

We continue to see an increase in the number of respondents reporting the use of automated provisioning for databases. 19% report that they are using some form of automated provisioning for a non-production environment.

Version control

We see a very modest growth in the number of respondents who report the use of version control for databases this year. We were interested in digging into the topic of version control for databases specifically to identify what benefits or downsides are associated with the use of version control.

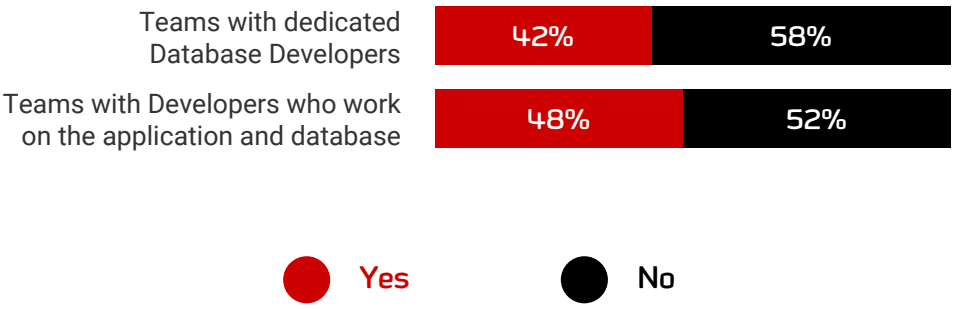
How closely are Developers and DBAs integrated in the teams you work with each week?



Team integration

Those who reported database code is in version control were more likely to report good or great team integration, and less likely to report poor integration. This may indicate that practices associated with version control promote more effective collaboration or help build trust between these groups.

Is your database under version control?



Who champions version control?

Those with dedicated Database Developers were less likely to report that they have databases in version control, signaling that dedicated Database Developers aren't all championing this practice.

Which teams are most likely to use version control for databases?

Respondents who reported that both DBAs and Developers author deployment scripts were the most likely to say that they have the database in version control. This supports the notion that version control can enable collaboration, even across team silos.

Continuous integration

While 27% of respondents report using automated builds to validate database code, only 15% report performing unit tests on database code. This difference may be related in part to organizations with broad existing test coverage for applications who choose to cover databases tests by automated testing via the application.

Orchestration tooling for continuous integration

In our 2019 survey, 16% of respondents reported using custom or home-grown tooling to orchestrate builds for database code. That number has fallen to 9%, indicating a greater adoption of vendor tooling for continuous integration.

The top five vendor continuous integration tools used by responders are Azure DevOps Services, Azure DevOps Server / TFS, Jenkins, GitHub Actions, and GitLab.

Automated deployments

Overall, 46% of respondents are performing some form of deployment automation to at least one environment, a significant jump from previous years.

33% of respondents are doing semi-automated deployments to production environments. Only 16% of respondents report that they have fully automated deployments to production databases. Note that respondents could indicate that they were doing both fully and semi-automated production database deployments, as their processes might vary for different databases.

Orchestration tooling for automated deployments

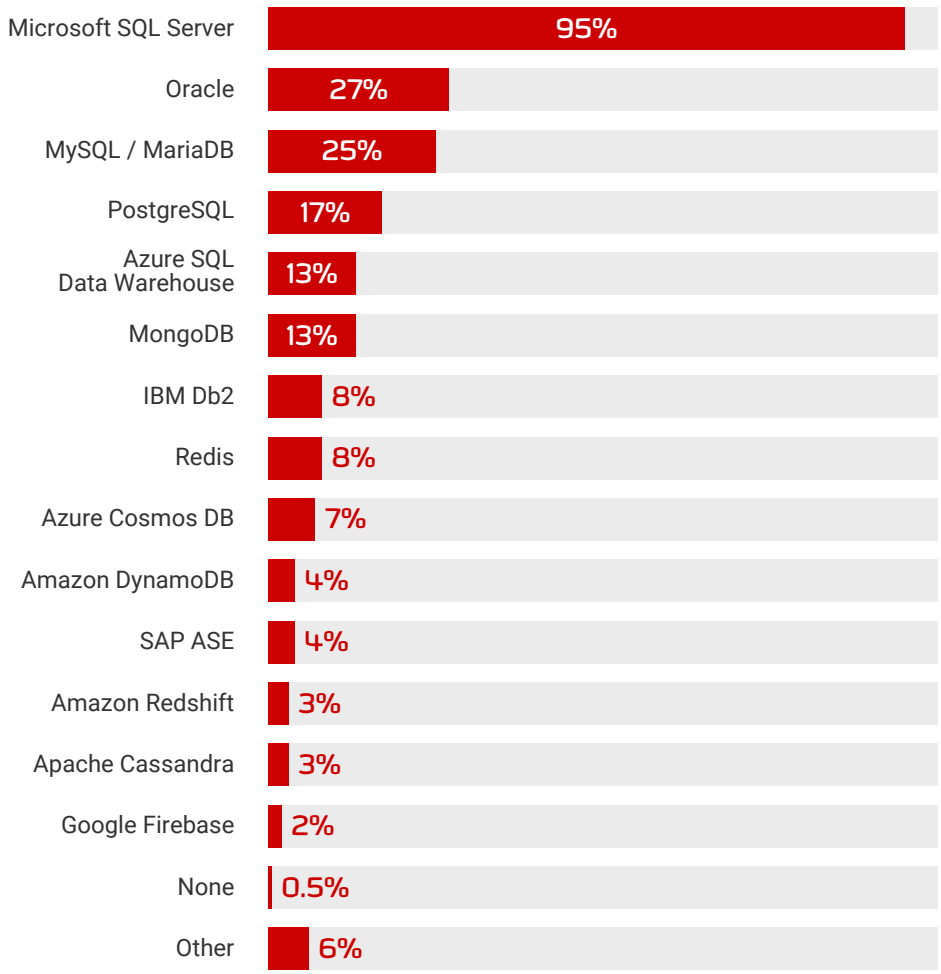
Here, we see a parallel pattern to what we observed in the orchestration tooling for continuous integration. In the 2019 survey, 16% of respondents reported using a custom or home-grown solution for automated deployments. This number also fell to 9% in the 2020 survey.

The top five vendor automated deployment tools used by responders are Azure DevOps Services, Azure DevOps Server / TFS, Jenkins, Octopus, and GitHub Actions.

Database systems in use

We asked which database system(s) are in use by our respondents, who could select multiple responses. We instructed respondents to include PaaS equivalents as part of the originating database system for the purposes of simplicity.

Which database system do you use?



The importance of deployment frequency

Steve Jones

One of the main things that DevOps style software development tries to do is increase the number of deployments for your software, as well as reduce the time between code commit and code deployment to production. These metrics are often tracked in various research items, as well as in organizations, and the highest performing companies usually excel in these areas.

Why is this important? I would argue that the longer code changes live in a version control system, but are undeployed, the riskier it is to deploy these changes and the greater the effort to test the entire system. This is because we are increasing the surface area of changes that need to be tested, and we are increasing the chance that this work might not meet the needs of our customer.

For databases, the deployment frequency is both more and less important. On one hand, we want to have good data modeling and add appropriate schema changes that meet our goals. This means we want to take the time to examine the requirements and appropriately normalize schemas, set efficient data types, and so on. This should slow the deployment.

On the other hand, the database provides the schema that the application builds on, and having this available early is helpful for Developers, ETL processes, report writers, and more. They will build their changes on top of the application, and we want to ensure these schema changes are available to them.

Compliant Database DevOps requires that we balance these two items in our work.

We should perform data modeling, but not get caught up in delaying code changes because of it. After all, quick changes mean that we can refactor or alter these schema changes in the future if necessary. We also want to learn the patterns for data modeling in our particular problem domain and ensure we can more quickly implement similar schema changes in the future.

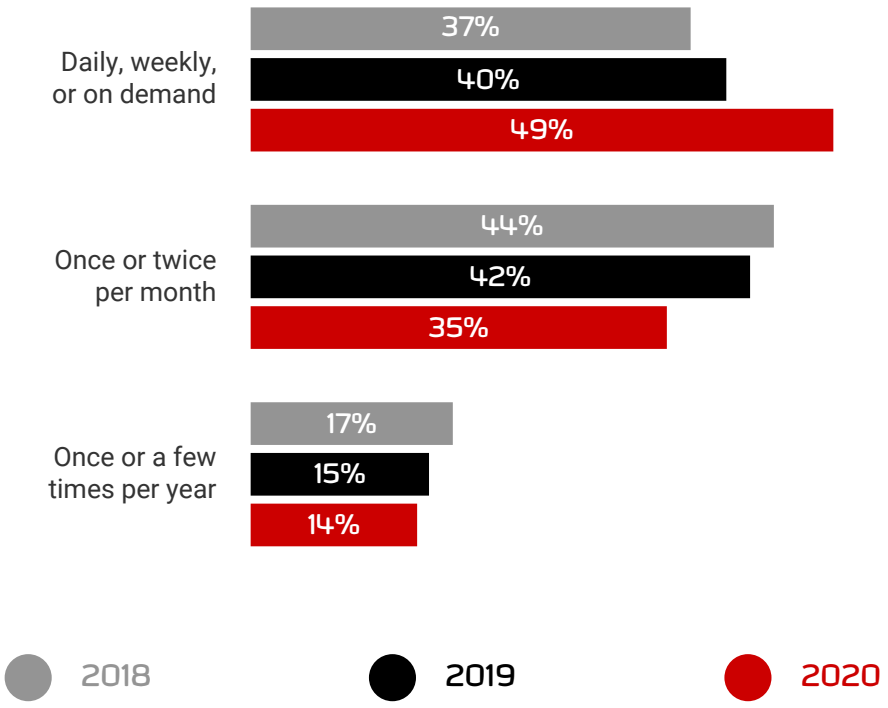
A more regular deployment frequency also means that we are adept at implementing changes without causing unnecessary downtime or disruption to the application, which allows us great flexibility and confidence that we can deliver those changes to live environments and allow Developers to build on the new functionality.

This is one of the top metrics that indicates a software development pipeline is agile and able to meet the demands of its customers.

Database deployment frequency is increasing

Data and analysis

How frequent are product deployments, on average?



We have seen a significant jump in the rate of frequent deployments. While previously 40% or less of respondents indicated that they deploy weekly or more frequently, in the 2020 survey that number has risen to 49%.

Which industry deploys the most frequently?

54% of respondents in the Financial Services sector reported deploying database changes to production at least once per week, the most of any industry sector. Media and Retail were close, with 53% reporting weekly or more frequent deployments. 51% of respondents in the healthcare sector reported this deployment frequency.

Lead time from authoring to deployment

We also asked respondents how long it takes, on average, for changes to make their way to production once they are code complete. This lead time is important, as long delays between writing code and deploying it can make it more difficult to support and troubleshoot if any issues arise.

Frequent deployments do not always indicate a short lead time for changes.

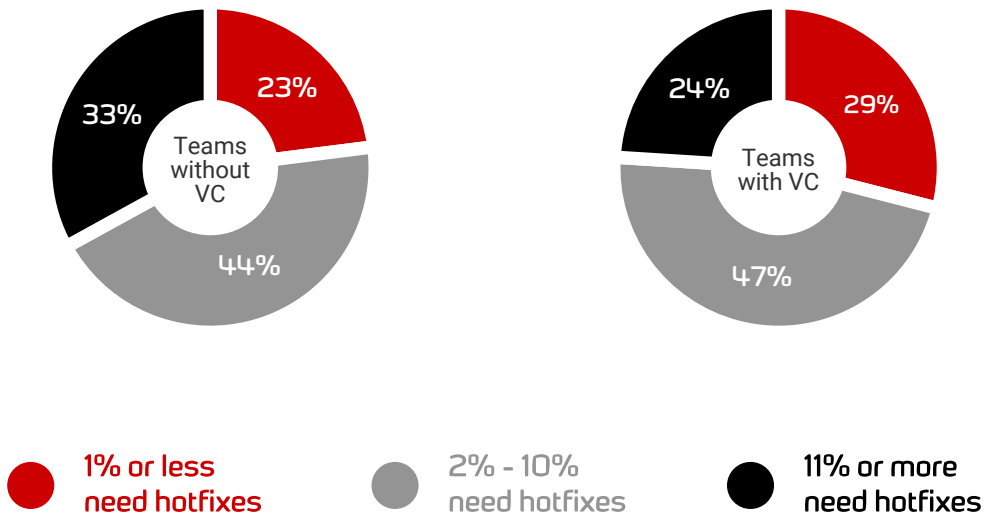
For those who deploy weekly or more often, 77% of respondents reported that the lead time for changes was one week or less, compared to only 51% of those who deploy on a monthly basis.

How do teams who deploy database changes to production weekly or more frequently work?

Interestingly, only 47% of those who say they deploy weekly or more frequently say they are using version control for the database. In other words, the majority of those deploying database changes frequently are making changes to the databases directly, and not storing those changes in version control.

Looking more deeply into this data, those deploying databases weekly or more frequently who DO use version control for database code are more likely to report that 1% or less of deployments contain defects that require hotfixing. Those who DO NOT use version control are notably more likely to report that more than 10% of deployments contain defects that require hotfixing.

How many deployments need hotfixes?



Summing up, while use of version control for database code is not required for frequent deployments, those who use version control report lower defect rates.

Change management and Change Approval Boards

In this year's survey, we studied the topic of change management, asking survey takers a series of questions about how database changes are proposed, assessed, and reviewed.

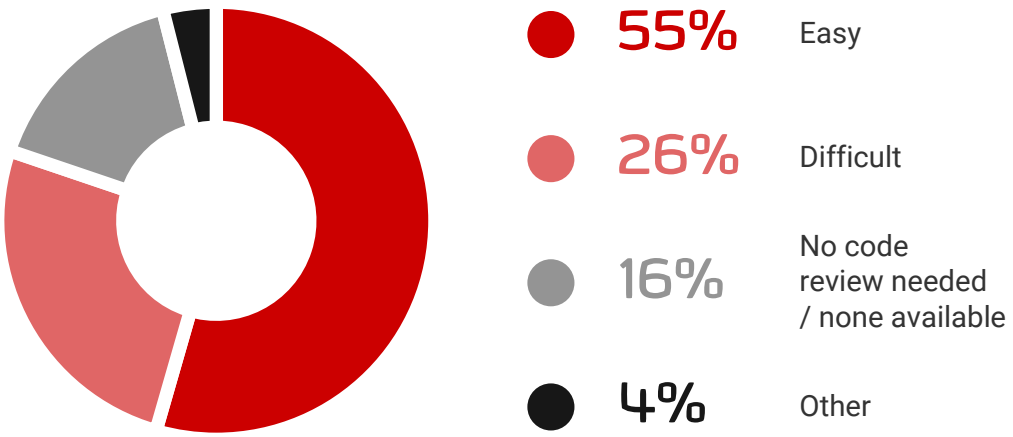
We find that respondents who can easily get code reviews for database changes report lower production defect rates and lower lead time for changes to be deployed to production. Conversely, although 38% of responders report the use of Change Approval Boards as part of their requirements for database deployments, we see no evidence that Change Approval Boards impact code defects, only that they increase lead time for changes.

We also find that the majority of respondents (55%) now perform most or all of their deployments for their most business-critical databases while the system is online and serving users. The group of users who reported that all or nearly all of their deployments take place with the system online also reported lower lead time for changes and lower defect rates.

Ease of code review

We asked survey takers how easily team members are able to get a code review for database code changes from someone with expertise in database coding and deployment. We asked them to think specifically of reviews early in the software development lifecycle, such as prior to committing code or prior to merging code from a feature branch.

How easily can team members get a code review for database code changes from someone with expertise in database coding and deployment?

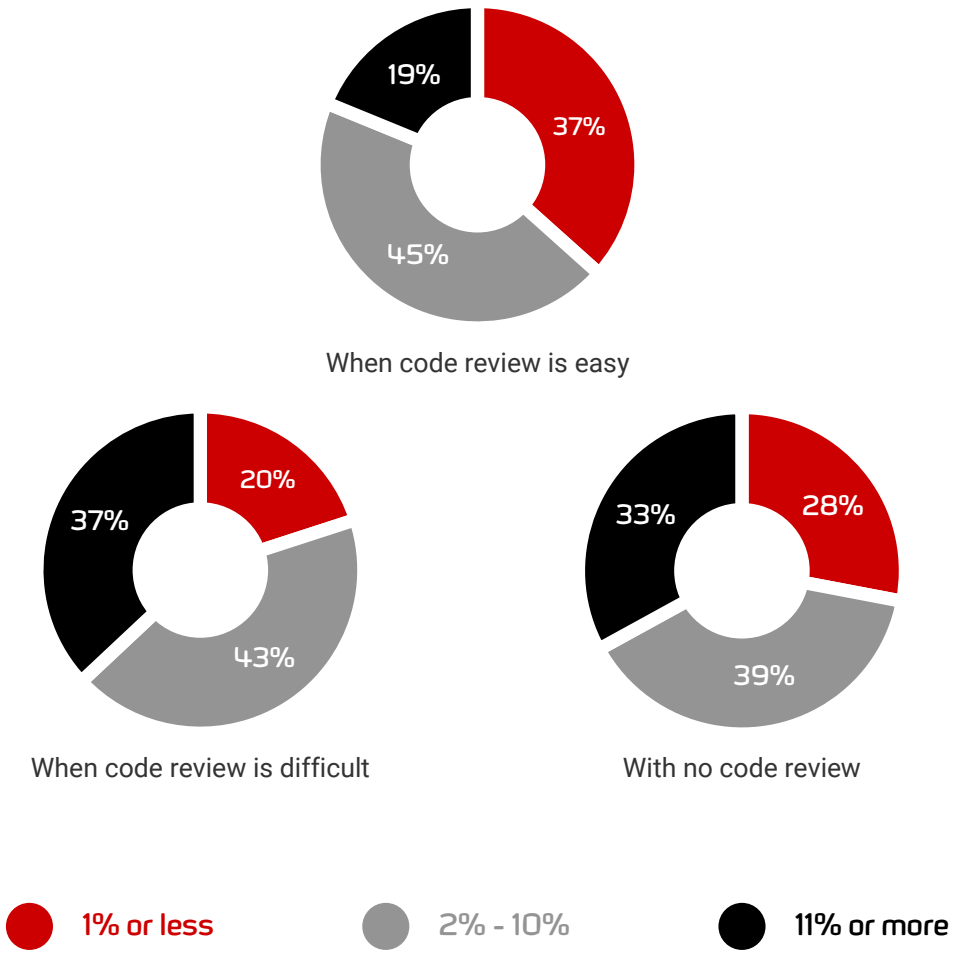


While for most respondents it is easy to get one, 26% of respondents said it is either somewhat or very difficult to get a code review for database changes. A further 16% reported that no code reviews are done for database changes early in the software development cycle – half reported that this is by choice, and half reported that this is due to no available resources to perform the code review.

Code quality

Those who reported that it's easy to get a code review were much more likely to report that 1% or less of production database deployments cause defects which require hotfixing.

How many deployments need hotfixes?



Those who find it difficult to get a review don't appear to fare much better overall than those with no code review, perhaps because when it's difficult to get a review the reviewer may either be hurried or not very familiar with the domain or code area they are reviewing.

Lead time

81% of those who reported that it is easy to get a code review indicated lead times for deployment of database changes to production were one week or less, compared to 63% of those who reported it is difficult to get a code review.

DevOps adoption

DevOps adopters are notably more likely to report that it's easy to get a code review.

- 72% of those who have adopted DevOps across all projects say it is easy to get a code review.
- 61% of those who have adopted DevOps across some projects say it is easy to get a code review.
- 47% of those who have not adopted say it is easy to get a code review.

Enterprises

While one might expect that it would be easier to get a code review in a larger organization, we do not see data to suggest that it is either easier or more difficult based on organization size.

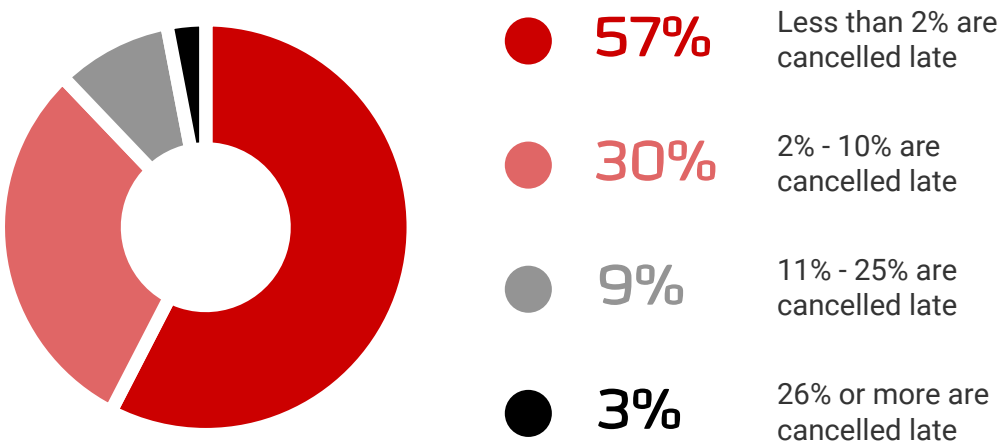
Deployment cancellation rates

We asked respondents, “What percentage of database deployments are canceled or returned to development by a change-approver who has rejected the change late in the software development lifecycle?”

We clarified that by "late in the software development lifecycle", we mean near the point where the change would be deployed to the production database.

We asked this question because deployments which are interrupted at a very late stage often have a negative effect on the software development process. They can either cause re-work for teams if the deployment is sent back for changes after nearly making it to production or impact the deployment pipeline for other changes.

How many database deployments are canceled or returned to development by a change approver who has rejected the change late in the software development lifecycle?



Most respondents (57%), indicated that this never, or almost never, happens. However, a significant number of respondents do see deployments which are cancelled very late in the deployment process.

Industry

Respondents in the Financial Services and Insurance segment were most likely to report some percentage of deployments are cancelled late, with 50% of the respondents indicating that this happens in 2% or more of deployments.

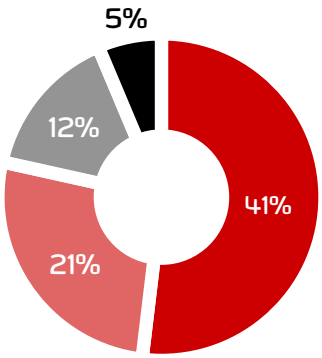
Relationship with code review

Those who report higher levels of deployments cancelled late in the software development cycle are less likely to say that it's easy to get a code review – particularly those for whom more than 10% of deployments are cancelled late.

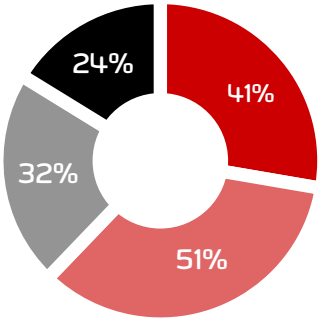
Code quality

Respondents who reported greater frequencies of deployments being cancelled late in the software development cycle were also more likely to report a higher percentage of deployments to production which cause defects that later require hotfixes.

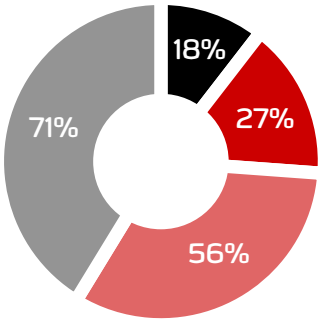
How many of your database deployments are cancelled late?



Teams whose deployments need hotfixes less than 2% of the time



Teams whose deployments need hotfixes 2 - 10% of the time



Teams whose deployments need hotfixes more than 10% of the time

- Less than 2% are cancelled late
- 2% - 10% are cancelled late
- 11% - 25% are cancelled late
- 26% or more are cancelled late

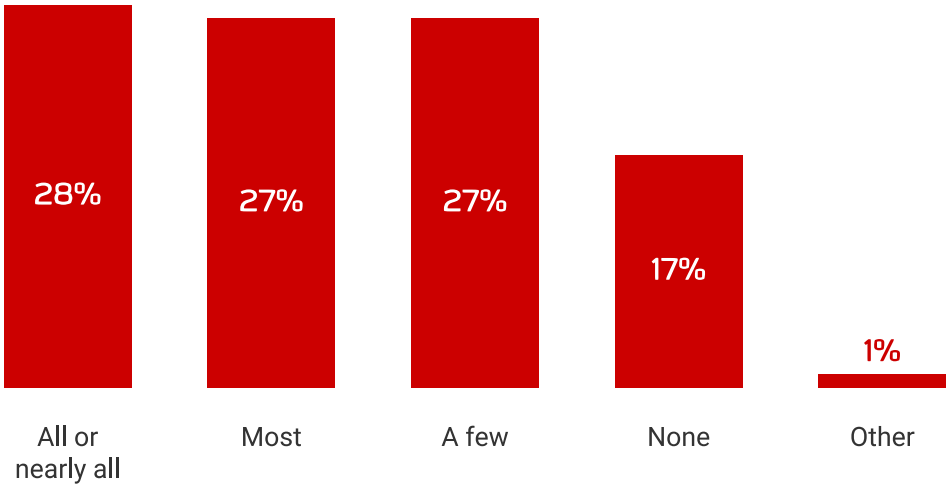
Interpretation

Together, these results suggest that making it easy for team members to get a quality code review early in the software development cycle, and taking other steps to ensure that requirements for deployment can be clearly met, produce database code with fewer defects.

Deployment windows and system status during deployments

We asked respondents, “Which of the following best describes the system status or application status when your team deploys database changes to production?” For this question, we asked them to consider the database which has the greatest business value, and which remains under active development.

How many of your database deployments take place when the system is live for users?



For their most business-critical databases, most responders (55%) deploy most or all changes when the system is online. Only 17% always deploy when the system is offline.

Defect rates

Those who reported that all or nearly all of their deployments take place with the system online were most likely to report that 1% of deployments or less require hotfixes. 75% of this group reported that 10% or less of their deployments cause defects which require hotfixes – the best of any group.

Lead time

Those whose deployments all take place with the system online are far more likely to report a short lead time for changes (one day or less).

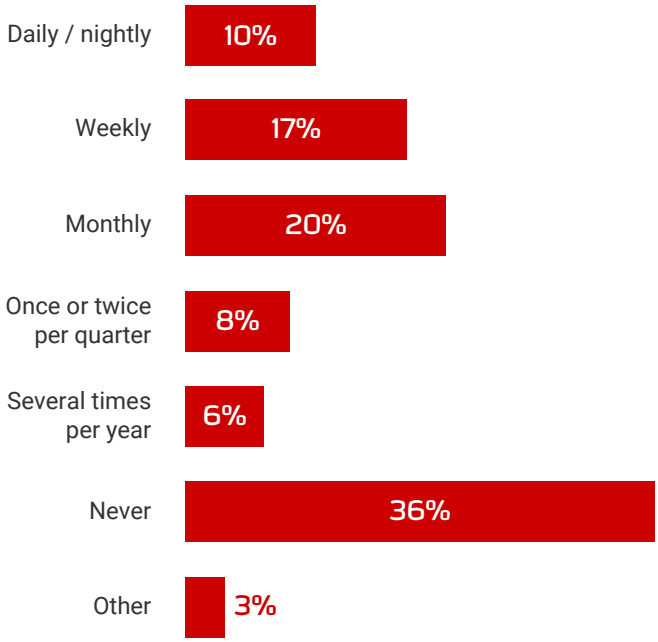
Interpretation

These findings may indicate that the consistent practice of designing changes for a critical database while it is online results in higher quality code which can be shipped quickly.

Scheduled downtime window frequency

We asked respondents whether, for the database with the highest business value they work with, their team has scheduled maintenance windows (planned downtime) which they may use for deployments.

How often you have regularly scheduled downtime for your database?



The majority of those who have a nightly or weekly downtime deploy while the system is live

53% of those who indicated they have a daily/nightly or weekly downtime window also indicated that they deploy most or all of their deployments while the system is live for users.

Interpretation

One might assume that whenever a downtime window is available for database deployments, a team would choose to take advantage of it for the purposes of simplicity. There are several reasons why this is not the case. Deployments during live business hours often provide better support when a deployment has unintended consequences, as more staff are available to respond during working hours.

This practice also encourages teams to build incremental, backwards compatible changes and to leverage functionality such as feature flags which minimize the impact of any single deployment.

Changing thoughts on addressing code defects in production

Kathi Kellenberger

Despite all the testing and checks that happen in the pipeline, sometimes deployments do not go as expected. When only software such as services and executables are involved, it's easy to roll back to the previous configuration by just replacing the changes with what was there before.

Database changes, however, are not easy to roll back.

A database is more than a set of tables and stored procedures. Databases contain the organization's actual data that may even be changing during deployments, and rolling back could cause data loss or corruption.

I remember years ago working on massive rollouts that took entire weekends, with the application shut down the entire time. The rollback plan for the database was to restore a backup, and, luckily, that rarely happened. In a DevOps organization with frequent deployments, restoring a backup is not usually possible.

Another option is to write rollback scripts for each change, but that would be very tedious. The best advice is to "roll forward" instead, dealing with issues if they come up and making sure that the fixes are part of the normal process.

Rollback approaches

We asked, “What best describes your team's approach to unexpected database code defects in production which are caused by deployments?”

How do you approach approach unexpected database code defects in production which are caused by deployments?

32%	We take a backup or snapshot of production prior to a deployment and restore this environment if there’s a problem.
28%	We triage, assess, and make a plan for defects caused by a deployment if and when they are found in production (this is sometimes called a roll forward approach).
21%	We use a pre-prepared formal rollback plan or down / undo script to revert the change.
14%	We take a backup or snapshot of production prior to a deployment and restore this environment to another location if there’s a problem, so we can compare and fix the environment manually.
4%	Other

Approaches vary by industry

- The industry segments most likely to use a prepared, formal rollback plan or down/undo scripts are Media and Retail (27% use them) and Financial Services (25%). The Healthcare, Medical, or Pharmaceutical segment is the least likely to use this approach (17%).
- The industry segments most likely to use a “roll forward” approach are Energy and Utilities (32%) and Healthcare, Medical, or Pharmaceutical (31%).

Enterprises

Respondents in Enterprises are notably more likely to use a prepared rollback or down script (27%) when compared to those in smaller organizations (17%). Enterprise respondents were also less likely than non-Enterprise respondents to use a roll-forward approach (26% vs 31%).

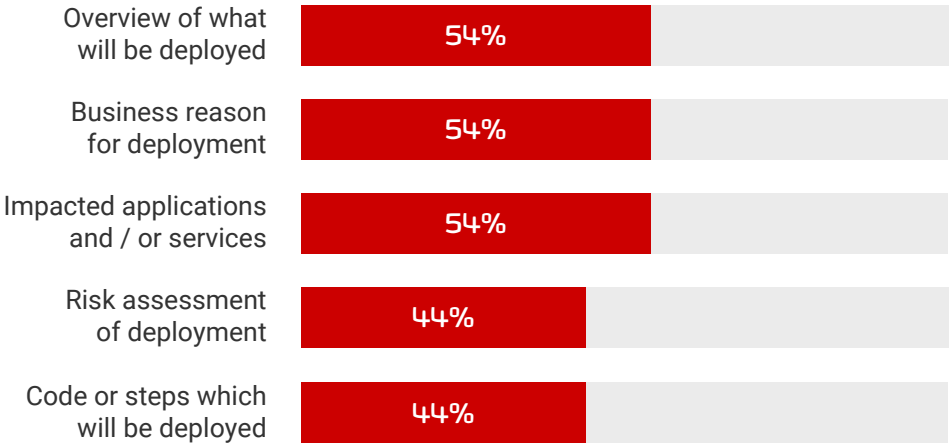
Percentage of deployments causing defects which require hotfixes

Defect rates were similar for respondent groups, whether they were using a backup approach, a roll forward approach, or a prepared rollback approach. In other words, writing rollback plans or down scripts isn't associated with a lower need to hotfix.

Change requirements

We asked survey takers, "Which of the following are you required to provide for change review/approval of production database deployments?" The top five results were:

Which of these must you provide for change review or approval of production database deployments?



Number of requirements

Respondents were able to choose multiple responses to this question, as change approval processes often have multiple requirements. 47% of those who need to provide justifications for review or approval are required to submit three or fewer items on our list.

If not a Change Approval Board, then what?

Grant Fritchey

One of the interesting pieces of information from the survey this year was the number of Enterprises that have some type of Change Approval Board. Further, those with a Change Approval Board have a longer lead time between code completion and deployment, yet, based on the responses, aren't getting any lower code defects. So, what gives?

I worked at a very large global Enterprise for a number of years. When I started there, they had approval boards for just about everything except oxygen use. My first experience with one of the boards was when I attempted to bring in, believe it or not, Redgate tools. I was told that since the tool wouldn't have a broad base of use, we couldn't bring it in. Needless to say, I fought that, successfully. Then, as we began to automate our deployment processes, we ran into the approval board for deployments and change.

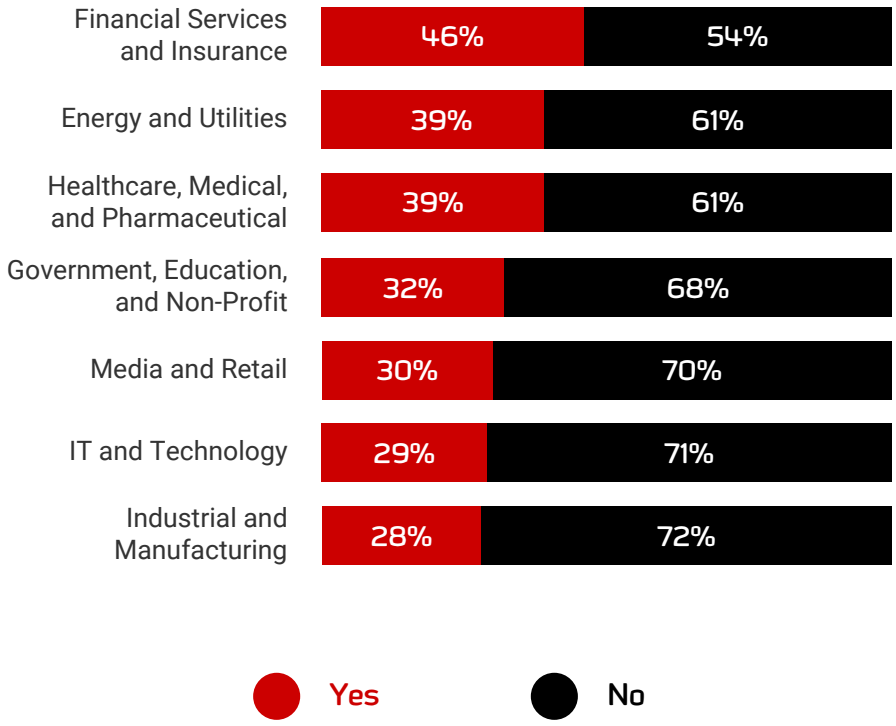
The key to either eliminating the board, or getting them to change how they operate, turned out to be information and education. First, we provided a detailed analysis of how we were testing our software releases. Then, we educated the board members on how exactly we were automating things and the level of documentation we could provide.

Not only were we able to show successful testing, but we could also show exactly what changes we were making and when. That helped to reassure the board and, instead of a blocker, they became participants in the process.

Change Approval Boards (CABs)

38% of responders overall reported that approval for database deployments is required from a Change Approval Board.

Must a Change Approval Board approve your changes?



Code quality

The use of a Change Approval Board isn't clearly correlated with a lower defect rate. Comparing responses between those who use Change Approval Boards and those who say they have no change approval process at all, those lacking a change process are more likely to say that 1% or fewer releases need hotfixes, while those with a board are more likely to say that 11-25% of changes need hotfixes -- no clear trend emerges.

Enterprises and team sizes

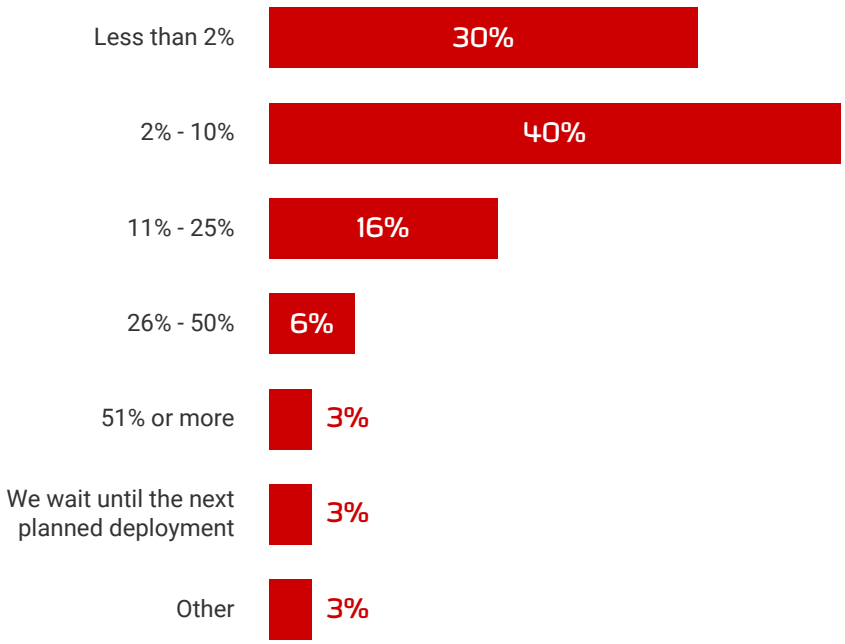
Respondents working with larger development teams and Enterprises are more likely to use a Change Approval Board.

Lead time

Respondents using a Change Approval Board report higher lead times for change deployments than those who do not use a Change Approval Board.

How many deployments cause defects which require hotfixes?

How many of your database deployments need hotfixing, or coding and deploying an immediate fix to an urgent defect introduced by the change?

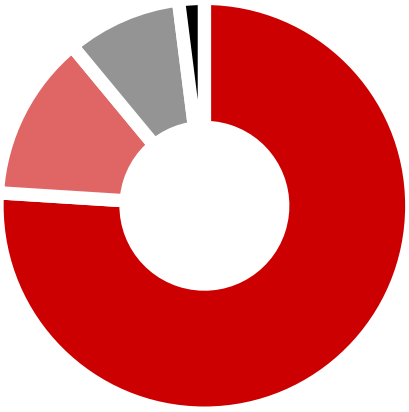


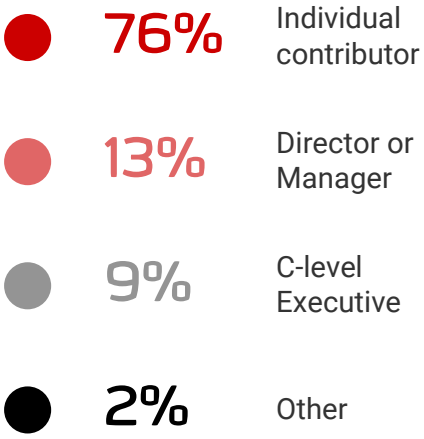
We clarified that the hotfix might occur because a rollback plan did not work as intended, the decision was made not to execute a rollback plan, or because they practice a roll forward approach.

Overall, most respondents report that 2-10% of code deployments for databases cause urgent defects. Broadly similar trends are seen across different industries, as well as when comparing Enterprises with non-Enterprises. No single segment among these groups is notably more or less likely to report code defects than the others. To err is human.

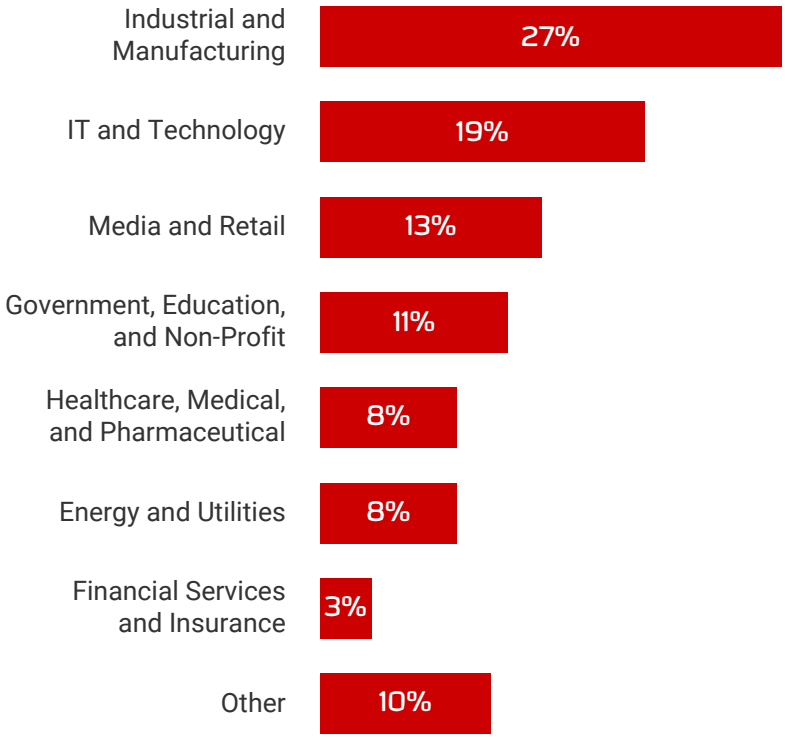
Survey demographics

Seniority of respondents

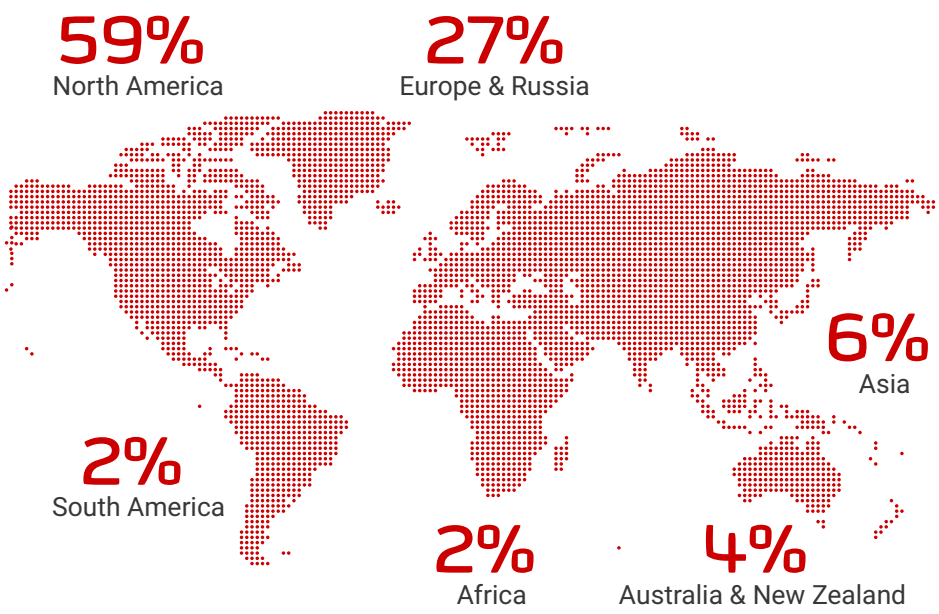




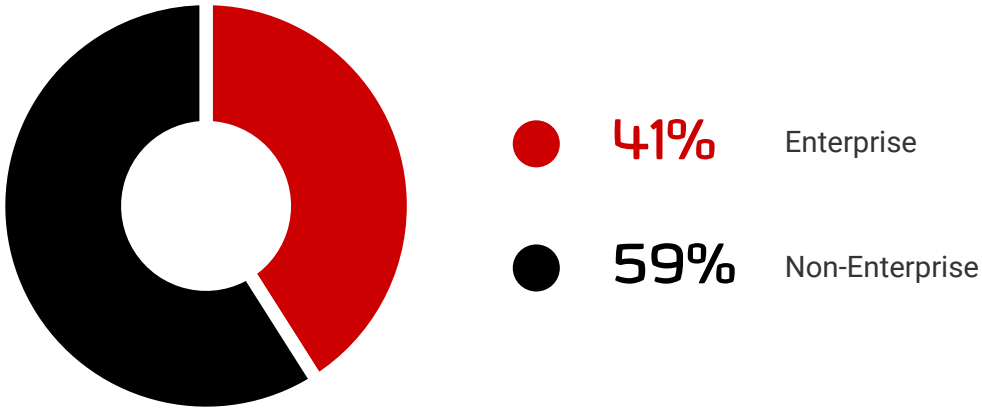
Industry of respondents



Region of respondents



Size of respondents' organization



Contributors

Kendra Little is the primary author of this year's report. Kendra is a DevOps Advocate at Redgate and a Microsoft Certified Master. She has trained IT leaders, Developers, and DBAs around the world to optimize development and operations for business-critical databases.

Kellyn Pot'Vin-Gorman authored our foreword. Kellyn is a Customer Success Engineer at Microsoft Corporation, specializing in Oracle On Azure, Data Platform, and automation. She is a proud member of Oak Table and an Oracle ACE Director Alumnus and former Idera ACE with over 20 years of extensive experience in database administration, database group management, and cloud migrations.

Donovan Brown contributed his definition of DevOps. Donovan is a Principal Cloud Advocate Manager of the Methods and Practices Organization in DevRel at Microsoft.

Luke Poretta, who contributed suggestions on making an effective business case for DevOps, is an Account Executive at Redgate. Luke has strong experience working with clients developing business cases and conceptual domain knowledge in the areas of DevOps & CI/CD, specifically around Microsoft SQL Server and Oracle database platforms.

Grant Fritchey, a DevOps Advocate at Redgate, shared his experiences moving beyond Change Approval Boards at an Enterprise organization. Grant has more than thirty years' experience in IT working in development and database administration. He has authored multiple books on the Microsoft Data Platform.

Steve Jones shared insights on the importance of deployment frequency. Steve is a DevOps Advocate at Redgate with over 25 years of experience building and managing database software. Steve is a founder of SQLServerCentral, the world's largest SQL Server online community and has been recognized as an MVP by Microsoft for the last 11 years.

Kathi Kellenberger wrote about changing approaches to rollbacks. Kathi is the editor of Simple Talk, is a DevOps Advocate at Redgate, and a Microsoft Data Platform MVP. She has worked with SQL Server for over 20 years and has authored, co-authored, or tech edited over a dozen technical books.

Tom Russell supported the charitable matching commitment for this year's survey and his team brought this visual publication to life. Tom is the Head of Brand at Redgate.