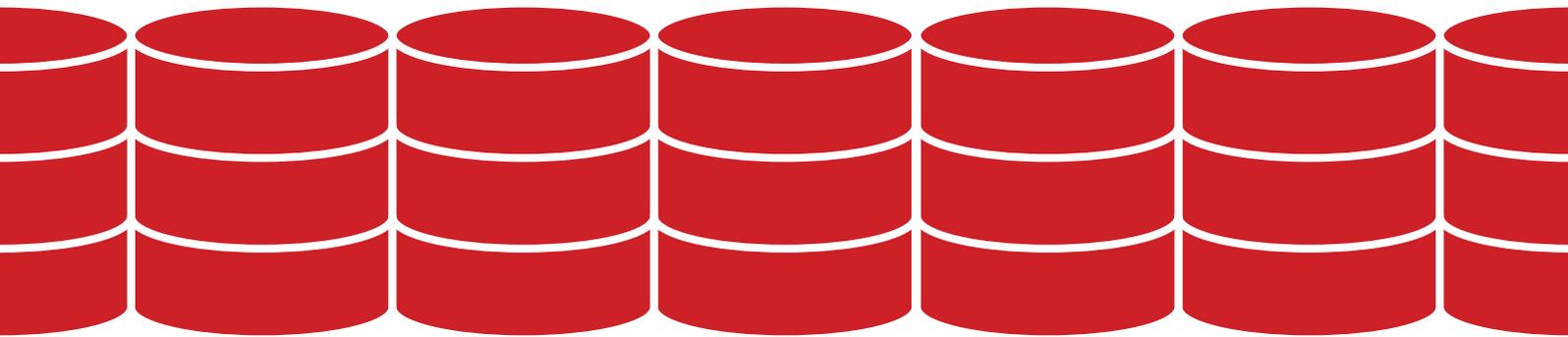


WHITEPAPER

Improving database development



redgate
ingeniously simple

Introduction

At Red Gate, we believe in creating simple, usable tools that address the problems of software developers and technology businesses. In considering the entire database development process we observed a number of costly, time-consuming, and highly obstructive problems that keep developers and businesses from working efficiently.

This paper gives an overview of the problems of database development, particularly those of change management and migration. It outlines the typical Develop, Test, Deploy structure of database development, proposes improvements at each stage of the process, and introduces a source control solution that finally places databases on a par with application development.

These recommendations are particularly relevant to development teams and businesses seeking to increase development efficiency, for instance to eliminate a backlog of high-priority, mission-critical development work. Crucially for these teams, the paper describes a complete database development package that integrates with existing working practices.

The development cycle

The standard database development process has three core stages:

1. Develop

The development team updates the database.

Ideally, changes are source controlled. This enables tracking, and eases continuous integration. However, source control for databases is not yet widely adopted.

2. Test

Migration scripts are created to move development changes to realistic testing and staging environments.

The QA or Testing team ensure there are no bugs or regressions.

3. Deploy

If testing is successful, migration scripts are created to deploy the changes to a staging environment, and then to production servers.

These final deployment scripts are typically scrutinized and ultimately run by a DBA.

The database development cycle is similar to that for application code, but with some crucial difficulties. For instance, there is often no source control; versioning is cumbersome; dependency chains can be complex and opaque; and the very nature of database code makes deployment a significant bottleneck.

Each stage in the cycle typically involves substantial manual scripting and a change management overhead.

Each stage also has inefficiencies, and can be automated or made simpler, reducing errors and easing coordination.

This whitepaper seeks to address these issues, and offer solutions using the Red Gate SQL Developer Bundle.

The problem of change management

Even in a small project, where a single developer is modifying a simple database, changes can introduce bugs and regressions. The more complex the development effort, the more likely it is that changes will have unforeseen consequences.

Source control is the solution typically adopted in application development. It allows teams and managers to see what was changed, when, and by whom; as well as offering a sandbox, versioning, and rollback.

When more than one developer is making modifications to a database, the whole team has all of these benefits. Sandboxes, in particular, are important, as they allow an individual developer to explore potentially breaking changes without disrupting the work of the team.

However, team development introduces the possibility of conflicts. Source control systems make it easier to find and resolve them.

The difficulty of database source control

In application development, source code is kept under in source control so that each revision to a file is retained.

However, database development is generally done by working directly with a database, rather than with script files. DML and DDL queries modify the current state of a database, so there are no files being changed. There is therefore no source code to control.

Most existing methods for source controlling databases are workarounds. Source code is created artificially by scripting out database objects, or storing migration scripts - and this sits outside the database development cycle.

Within the development workflow, scripts may be specifically created for migration and deployment, and those scripts can be checked into a source control repository. However, deployment scripts do not represent a practical history of incremental changes, and creating them regularly complicates the development process.

Mature development projects will already employ a source control system. Finding a simple way to record database modifications in that system solves the change management problem by taking greater advantage of your existing infrastructure. So there is no need for an additional investment.

Improving database development

With Red Gate tools, the database development process becomes simpler and more unified.

Database changes are source controlled alongside application code, where both can be versioned and become part of a continuous integration process. Tests can be run, and readily automated, as can schema documentation, and eventual deployment.

The result is that higher quality development becomes easier, and requires less developer time.

Improving the development environment

SQL Server Management Studio is central to the SQL Server development and administration process. It is the industry standard IDE.

There are alternatives - notably Visual Studio, used for developing application code - but they are not so closely tailored to database development.

As SQL Server Management Studio is the preferred and recommended IDE, it makes sense to introduce productivity and process enhancements directly into that environment. Switching between tools is disruptive, and purchasing a new IDE is disruptive and costly.

The Red Gate SQL Developer Bundle integrates directly with SQL Server Management Studio, to address two problems: individual productivity, and team processes.

Productivity improvements

The majority of database development time is, unsurprisingly, spent writing and maintaining SQL code, so this is the most obvious productivity bottleneck to address.

SQL Prompt is a code auto-completion and layout tool. This makes writing SQL faster and more reliable.

SQL Prompt also helps developers with code maintenance. It lets you automate table splits, renames, and other common, time-consuming refactorings.

A modern database may contain hundreds if not thousands of objects, so writing code is only part of the productivity problem. Even with easily adopted naming conventions, just finding a specific object, or an object that references another can take time. **SQL Search** searches for SQL code across one or more databases, making it easier to work with large, complex schemas.

Process improvements

The cornerstone of Red Gate's database development process recommendations is the introduction of source control.

Once source control is in place, it is easier to adopt continuous integration and unit testing. These are established best practice for application code, but have so far not been readily available for database development.

SQL Compare Pro and **SQL Data Compare Pro** simplify migration and deployment, which in turn enables automation. Automated deployments, and the realistic data generation provided by **SQL Data Generator** make it easy to adopt database unit testing, as well as performance and integration tests.

Red Gate's source control solution

SQL Source Control is Red Gate's source control solution. It integrates your existing source control system with the database development workflow.

SQL Source Control is an add-in for SQL Server Management Studio. It maintains a folder of scripts representing the structure of a database in source control.

Database source control can therefore easily be adhered to, as the day-to-day steps such as committing changes have been simplified and brought into the existing workflow.

Advantages of SQL Source Control

The benefits of database source control are the same as the benefits of source control for application code. You can audit changes, seeing when, by whom, and why they were made. You can isolate those changes, and where necessary, roll them back. Each developer has a safe sandbox, changes are merged back into the central repository, and batches of changes can be collated into labeled versions.

SQL Source Control is designed to eliminate the extra overhead of creating code to source control. Because it sits within SQL Server Management Studio and rapidly detects schema changes, committing those changes is simple and quick.

The source control repository can therefore represent "the truth" - the latest version of the database incorporating all changes. Because SQL Source Control makes it easy to update a database with changes from source control, and commit changes to source control, it eliminates the need to create change scripts during development. On its own, this saves time and eases change management.

Source control that is simple and unobtrusive is more likely to be used.

Source control with a low barrier to entry is easier to adopt.

Usable, accessible source control that sits within the existing development workflow is a cornerstone of good change management.

Conclusion

Red Gate's SQL Developer Bundle presents an alternative to the traditional scenario in which database development is cumbersome, challenging, and out of step with application code.

Adopting it improves productivity, change management, testing, and deployment, across the whole database development life cycle.

You can download the SQL Developer Bundle, with all the tools described in this paper, for a free 14 day trial here:

www.red-gate.com/sql-developer-bundle