

# Redgate SQL Toolbelt Essentials - Hands-On Exercise

Welcome to the SQL Toolbelt Essentials practical exercise. This guide will walk you through exploring Redgate's database DevOps tools using sample databases.

## Getting Started

*The presenter will provide you with a Demo VM that has all required software pre-installed.*

*Want to try this on your own machine? Follow the [Setup Guide](#) to install SQL Server, SSMS, and the Redgate tools.*

### Step 1: Download the Exercise Files

1. Open a browser on the Demo VM and go to:

- o <https://github.com/MrTyRedgate/RGToolbeltEssentialsExercise>

2. Click the green **Code** button, then select **Download ZIP**

3. Create the folder `C:\Temp\ToolbeltEssentialsExercise\`

4. Copy the downloaded ZIP file into `C:\Temp\ToolbeltEssentialsExercise\`

5. Extract the contents of the ZIP file into this folder

### Step 2: Connect to SQL Server

1. Open **SQL Server Management Studio 18.10 (SSMS)**

2. In the Connect dialog:

- o **Server name:** REDGATE-DEMO\SQLEXPRESS
- o **Authentication:** Windows Authentication
- o Tick **Trust server certificate** (if applicable)
- o Click **Connect**

### Step 3: Run the Database Setup Script

1. In SSMS, go to **File > Open > File**

2. Navigate to `C:\Temp\ToolbeltEssentialsExercise\` and open `CreateSimpleDBDatabases.sql`

*Alternatively: Open a new query window and copy-paste the contents of the file*

3. Click **Execute** (or press F5)

4. Wait for the script to complete

5. Refresh the Databases folder in Object Explorer to see:

- o SimpleDB\_Dev1
- o SimpleDB\_Dev2
- o SimpleDB\_Test
- o SimpleDB\_Prod

6. **Optional:** To clean up the view, right-click on the **Databases** folder, select **Filter > Filter Settings**, set the Name filter to `Simple`, and click **OK**. This limits the view to only the newly created databases.

## Exercise Goals

By the end of this exercise, you will be familiar with:

### Primary Focus:

- **SQL Source Control** - Version control your database schema
- **SQL Compare** - Compare and synchronize database schemas between environments

### Secondary Tools:

- **Dependency Tracker** - Visualize object dependencies in your database
- **SQL Doc** - Generate documentation for your database schema

---

## Part 1: Explore the Tools

*Wait for the instructor before completing these exercises.*

### Exercise A: SQL Source Control - Initial Setup

**Objective:** Link a database to source control and commit the initial schema

1. In SSMS Object Explorer, right-click on `SimpleDB_Dev1`
2. Select **SQL Source Control > Link Database to Source Control...**
3. Choose your source control system (Git, TFS, SVN, etc.) or "Just let me try it out" for a Demo
4. Select a repository folder
5. Click **Link**
6. Observe how database objects appear as scripts in source control in the Commit tab
7. **Commit** all objects to version control as your initial baseline - think of a meaningful commit message (e.g., "Initial database schema")

---

### Exercise B: SQL Source Control - Making Changes

**Objective:** Make schema changes and commit them to source control

1. In SSMS, go to **File > Open > File** and open `Exercises.sql` from  
`C:\Temp\ToolbeltEssentialsExercise\`
2. Run the tasks in order (1, 2, 3) to make schema changes to `SimpleDB_Dev1`
3. Return to SQL Source Control in SSMS and use the Commit Tab
4. See the new changes appear (the `Socials` table, `ListSocials` stored procedure, and `WorkPhone` column)
5. Select and **Commit** all your changes to version control using a relevant commit message

---

### Exercise C: SQL Compare - Deploy to Test

**Objective:** Deploy your changes from Dev1 to Test

1. Open **SQL Compare** from the Start menu or SSMS Tools menu

2. In the comparison wizard:

- **Source:** Select **SQL Source Control**, then choose `SimpleDB_Dev1` with revision **Latest (HEAD)**
- **Target:** Select **Database**, choose server `REDGATE-DEMO\SQLEXPRESS`, tick **Trust certificate**, then select `SimpleDB_Test`

3. Click **Compare Now**

4. Review the differences - you should see the changes you made in Exercise B

5. Select all the new objects to deploy

6. Click **Deploy** and then **Deploy using SQL Compare** and Next using other defaults

7. Review the script and deploy the changes

8. Now repeat the process to deploy those changes to `SimpleDB_Prod` but beware! **NB** Did you notice anything about Prod that was concerning? You should have spotted the Drift, don't Deploy to Prod Skip to ## Bonus Exercise 2 to fix the drift first then return to this step

---

## Exercise D: Dependency Tracker (Secondary)

**Objective:** Visualize database object dependencies

1. Open **Dependency Tracker** from the Start menu

2. Connect to `SimpleDB_Dev1`

3. Explore the dependency graph for:

- `Sales.Orders` table - see related views, stored procedures, and foreign keys
  - `Sales.CustomerOrdersView` - see which tables it depends on
- 

## Exercise E: SQL Doc (Secondary)

**Objective:** Generate database documentation

1. Open **SQL Doc** from the Start menu

2. Create a new project and connect to `SimpleDB_Test`

3. Select all database objects to document

4. Choose output format (HTML, PDF, or Word)

5. Generate documentation

6. Review the output - tables, relationships, stored procedures are all documented

---

## Bonus Exercise 1: Link Dev2 to the Same Repository

**Objective:** Link a second database to an existing source control repository and sync changes

1. In SSMS Object Explorer, right-click on `SimpleDB_Dev2`

2. Select **SQL Source Control > Link Database to Source Control...**

3. Link it to the **same repository folder** you used for `SimpleDB_Dev1` which is an **existing** repository
  4. Once linked, go to the **Get Latest** tab
  5. Pull the latest changes from source control to update `SimpleDB_Dev2` with the schema changes from Dev1
  6. Verify that Dev2 now has the `Socials` table, `ListSocials` procedure, and `WorkPhone` column
- 

## Bonus Exercise 2: Rescue Prod Drift into Source Control

**Objective:** Bring untracked production changes back under version control

In Exercise C, you may have noticed Prod has some unexpected differences. This simulates a common real-world scenario where someone made "emergency" changes directly to production without going through source control.

1. Open **SQL Compare**
  2. Set up a **reverse comparison**:
    - **Source:** `SimpleDB_Prod` database
    - **Target:** `SimpleDB_Dev1` database
  3. Click **Compare Now**
  4. Identify the drift - you should see:
    - `Customers.Customer` has an extra column (`LastLoginDate`)
    - `Inventory.TempFlightCache` is an extra table
  5. **Decide what to do:**
    - Is the `LastLoginDate` column valuable? (Yes - security team needs it)
    - Is `TempFlightCache` needed? (No - it's leftover from an old report)
  6. Select only `Customers.Customer` and deploy to Dev1
    - You will have noticed that column `WorkPhone` would have been dropped from Dev1 by SQL Compare so you need to add it back, use the `Exercises.sql` to add just `WorkPhone` back. Select the code block for this and execute, taking care not to rerun the whole exercises script.
  7. Return to SQL Source Control and **commit** the rescued change
  8. Now your source control reflects the legitimate production change, and you can clean up the unnecessary `TempFlightCache` table from Prod later
  9. Now go and finish off your Prod deployment as the Drift is repaired
- 

## Database Schema Overview

Each sample database contains:

Schema	Objects
<b>Customers</b>	<code>Customer</code> , <code>LoyaltyProgram</code> , <code>CustomerFeedback</code> tables + views

<b>Inventory</b>	Flight, FlightRoute, MaintenanceLog tables + views
<b>Sales</b>	Orders, DiscountCode, OrderAuditLog tables + views + stored procedures

**Sample Stored Procedures:**

- `Sales.GetCustomerFlightHistory` - View customer's order history
  - `Sales.UpdateOrderStatus` - Update an order's status
  - `Sales.ApplyDiscount` - Apply discount codes to orders
  - `Inventory.UpdateAvailableSeats` - Manage flight seat inventory
  - `Customers.RecordFeedback` - Record customer feedback
- 

## Quick Reference

Tool	Purpose	Access
SQL Source Control	Version control for databases	SSMS > Right-click database
SQL Compare	Schema comparison & sync	Start Menu or SSMS Tools
Dependency Tracker	Visualize object relationships	Start Menu
SQL Doc	Generate documentation	Start Menu

---

*Happy exploring! Ask questions if you get stuck.*