Revision	Description
4/4/2010	Original

SQL-Hero History Tracking and Region Differences (Compliance)

Introduction

What exactly is the history tracking that SQL-Hero provides and how does it contrast with say Microsoft's GDR for VSTS Database Edition? First, a tool such as GDR works on the premise that you will track all database changes in a repository *outside* of the actual physical database. Changes are effectively reconciled to and from the repository. There are a few reasons this opens up a "gap":

- It is possible to effectively bypass GDR by making changes in the physical database. It is true that a process should be in place to prevent this, but developers have been known to need to apply hot-fixes to production databases, sometimes in a hurry it has been observed that an audit trail of actual changes is an important element in *any* process to close this loop.
- As a SQL developer, I definitely need to work with a real database as I develop, in order to test and debug a variety of objects that operate in concert. There isn't integration within Visual Studio or SSMS that would capture *all* of my work as I develop and debug SQL. As such, one still faces a reconciliation step.
- Assuming the above, GDR will only contain a snapshot in time, namely the point at which I've consciously committed my work. It will *not* contain all changes I've ever made in a development database, unless I commit every single change as I work: unlikely given the extra overhead it would impose in my development process.
- GDR will not record all commands I issue against a database (including DML), automatically.

SQL-Hero differs in that it can automatically track all changes in the actual database: both DDL and DML (see the caveats on DML later). There are other side advantages to doing things in this way, including the fact we can associate names of developers with changes and have a more accurate picture of the sequence of change, even in the intermediate steps you would have during development.

Change Tracking

There are two ways that SQL-Hero can track all changes. One is by forcing all developers to use the SQL-Hero editor tool (note that there are ways to enforce this using SQL security). This is the only real option if you're using a database other than SQL Server 2005 or 2008. The other way is to apply a DDL trigger to the databases you want to track. You can do this from the "DB Settings" command (¹¹) on the History tool window:

🕐 SQL-Hero	
Editor Data Compare Schema Compare Differences	History Notifications Create Data Testing Region Diffs Tracing Monitori
Style: Details (with filters)	💽 😨 Search 🗙 Clear 😥 Show Criteria 🔌 Apply Label 🗊 DB Settings
	Search Criteria
Attribute	Value

This option is only available if your client install of SQL-Hero is pointed at a SQL-Hero application server. For details on what this means, consult the "Installing SQL-Hero" whitepaper.

With the database settings window, you will see all available databases which are designated as being "global." You control whether a connection is global on the Settings tool where all connections are managed:

R	U SQL-Hero										
/	Editor Data Compare Schema Compare Differences History Notifications Create Data Testing Region Diffs Tracing Monitoring Settings										
	💽 Active Connections 🎄 All SQL Servers 💋 Add Connection 🗼 Delete 👔 Manage Aliases 🙀 Update SQL-Hero Server 😝 Export 🇃 Import										
	Drag a column header here to group by that column.										
Е	Alias	Туре	Category	LogDDLOnI	LogAll	WarnExec	Global	Local Storage For	ConnString		
	AzureDemo	SQL Azure 10	Azure	N					Data Source=tcp:bpl		
	Demo08	SQL 2008	Demo	V					Data Source=(local)		
	Demo08LSF	SQL 2008	Demo					Demo08	Data Source=(local)		
	Finance-Development	SQL 2008							Data Source=(local)		
	ha sa a c	001.0005	11 - 1 - 1	-	-	-	-		D. C. // 15		

Back to the "DB Settings" window, we see all global connections, but not all databases can have DDL or DML tracking applied (i.e. tracked outside of SQL-Hero, as well). DDL tracking is only available for SQL 2005 and 2008. DML tracking will be available for SQL 2008 in a future release of SQL-Hero. In the example below, we have active DDL tracking on the Finance-Development database.

Ĭ	Votifications - Monitored Databases								
	ок 🄇	Cance	el						
	DDL	DML	Database	Denial Conditions	Can DDL	Can DML			
►			AzureDemo						
			Demo08						
			Demo08LSF						
	2		Finance-Development	None					
			AzureDemo Demo08 Demo08LSF Finance-Development	None					

Optional denial conditions can also be applied on both SQL 2005 and 2008 databases (note that SQL Server 2008 out-of-the-box offers some similar capabilities – but 2005 does not):

Deny Change Rules								
🛃 ОК 💢 Р	Remove All Rule	s 🕥 Cancel						
The following	g conditions defi	ne what operations will be DENIED in the selected database.						
	User Names							
And/Or	Like/Not	Pattern						
`* □		fsmith						
* 🔽								
AND	O OR							
		Object Names						
And/Or	Like/Not	Pattern						
* 🔽	N							
C AND	6.00							
	• OR	N/A						
		Does not have permission to ALTER ANY LOGIN						
		Does not have permission to CREATE ANY DATABASE Does not have permission to ALTER TRACE						
		Does not have permission to CREATE TABLE						

Here we can build a set of conditions that will prevent DDL commands from being issued against the database, including checks against the user's name, object names, and checking against other privileges.

With DDL tracking in place, all changes - even those made outside of SQL-Hero - will be recorded in the repository. Change records do not appear immediately, however: the SQL-Hero Windows service works to aggregate changes, usually within one to two minutes. Note that enabling DDL tracking on a database adds a schema called "sqlhero" and a table called "sqlhero.ddllog".

Another option is to track changes made from the SQL-Hero editor, without requiring any additional schema objects. You can elect to log DDL commands:

Settings
ication Group

Or log all SQL statements, including DML:

R	🖞 SQL-Hero										
/	Editor Data Compare Schema Compare Differences History Notifications Create Data Testing Region Diffs Tracing Monitoring Settings										
	🛐 Active Connections 🍓 All SQL Servers 💋 Add Connection 🗶 Delete 🗊 Manage Aliases 😂 Update SQL-Hero Server 🔒 Export 🗃 Import										
	Drag a column header here to group by that column.										
	Alias	Туре	Category	LogDDLOnly		LogAll		WarnExec	Global	Local Storage For	Application Group
	AzureDemo	SQL Azure 10	Azure	V			Т		•		
	Demo08	SQL 2008	Demo	V					▼		
	Demo08LSF	SQL 2008	Demo	V					▼	Demo08	
	Finance-Development	SQL 2008		V							
	4		1			_			_	1	

In terms of what it means to "Log All," you can either log such changes in the SQL-Hero repository which makes even the DML commands available for searching on the History tool, or you can log them locally on your computer's file system (which makes them available for searching using the "Local Recorded SQL" tool described below), or using both techniques. The choice of how to log can be set using some Global Settings, available under Manage -> Global Settings on the Editor's toolbar menu:

🖞 SQL-Hero	
Editor Data Compare Schema Compare Differences Histo	ry Notifications Create Data
Category: RD Database: RD-Development	🔽 🍸 Filter 🚰 🕨 Execute
File ▼ Edit ▼ Tabs ▼ Actions ▼ Object ▼ Favorites ▼ Templates ▼	Manage - 👫 Find: count
Image: Stored Procedures Image: SqL (1) × Image: Image: SqL (1) ×	 Manage Users Tombstone Settings Source Control Settings Global Settings Manage UDP's Server Logs Recorded SQL History
🖞 Global Settings	×
Stored Procedure Test Requirements	
Must have test UDP and attempt to auto-generate	
Maximum wait (milliseconds) 8000	
Logging	
"Log All" writes to: Both Central Repository and Local File System	I
Build Settings Local File System (only) Both Central Repository and Local File System	ng: {0}
Enable "add dependencies" check-box on add to build window	

Now that you've collected some change records in the repository, how can you go about searching for them? This is where the History tool shines, with a wide range of search criteria:

🐙 SQL-Hero								
Editor Data Compare	Schema Compare Difference	s Hist	ory Notifications Crea	ate Data Testing Reg	ion Diffs	Tracing Monitorin	g Settings	
Style: Details (with filters)	 Max Objects: 10 	10 🔻	💈 Search 🏾 🗙 Clear 📢	Show Criteria 📐 Apply	/Label 🧃	📔 DB Settings		
	Search Criteria							
Attribute			Value		Val	ue End		
Database								
Object Name Equals								
Object Name Like								
Object Type								
Last Modified By User								
Any Modified By User								
Last Modified By Machine								
Any Modified By Machine								
Last Modified Between								
Any Modified Between			03/28/2010 12:00 AM					
Contains Text			validation					
Valerk Item Pange								
work item range								
						Search Results		
Database	Date	Text				User	Host Name	Name
Local-Test1	3/30/2010 20:08:54.190	ALTER TABLE BusinessUnit ADD RequiresCustomer bit NOT NULL DEFAULT (0) BUSINESSUNIT						

In this example we've searched for any changes since 3/28/2010 and containing the word "validation". Notice that there are different report styles available. Here is a summary of the behavior of each:

Details (all versions)	Any change that met the criteria would qualify <i>all</i> object changes for a given object to be shown.
Details (with filters)	Only changes that fall within the search criteria are displayed.
Latest Version Only	Only the latest version of matched change records is returned, by object.
Object List (distinct)	A distinct list of object names is shown, based on criteria matches.
New Objects Only	An additional condition applied to the search criteria is that the matched change records must be for the first version of an object (i.e. if previous change records exist then no records are shown for the object).
Release Notes (comments)	An additional condition applied is that the matched change records must have an associated comment (as captured from the SQL-Hero Editor's comment prompting feature), and the output format includes these comments.

Note that the returned results includes a column called "Date" which represents the time the change was made as determined by the capturing process (be it SQL-Hero or the DDL trigger). If there is a large difference between this time and the time that the event was actually recorded in the repository, an additional column, "Recorded Date," is shown as well.

With results in the grid, the grid's context menu (right-click) allows you to do a number of things:

	Grid	Y	TODEX07 Ellements
₽.	View Version		Grouping
	Compare Versions		Hide Column
	Compare with Current		Unhide All
	Perform Search Using		Grid Layout
2	Select Object		Distinct Column Values
1	Add To Build	~	Auto-Fit Columns
	To Clipboard	B	View As Report
	To File 🕨 🕨		

Many of these operations are "standard" grid features that will not be covered here. Things specific to the history tool include the "View Version" command (¹/₂). This takes the currently selected change record's SQL text and places it into a new SQL-Hero Editor window, for the database it was associated with. This is especially useful if you're interested in reverting to a prior version of an object.

The "Compare Versions" command (^(E)) is only enabled if more than one result record is selected (you can multi-select by holding down Ctrl as you click on row selectors). It will use the SQL-Hero Text Differences tool to show deltas between the two change records. "Compare with Current" takes the selected change record and does a similar comparison, with the current version for the selected object.

"Perform Search Using" (a) takes the selected row's object name and reissues a search using the object name and database (and style "Details (all versions)"). This is useful if you locate an object change record and want to see *all* changes for the object, over time. "Select Object" (2) takes the selected row's object name and locates that object in the SQL-Hero Editor. The "Add to Build" command (2) will be covered in a different whitepaper on build management.

Note that the history search tool is accessible from a couple of different places in SQL-Hero. The most obvious place is on the Editor tool where one can simply right-click on any object, and request its full change history:

SQL-Hero		
Editor Data	Compare Schema Comp	are Differences History
Category: RD	▼ Database: RI	D-Development
File - Edit -	Tabs▼ Actions▼ Object▼	Favorites - Templates - Ma
🛃 🕺 👪 🏹) 🕮 😒 🏂 🖄 🖄 1	1 🛛 🕹 🔁 🔹 🕯
Stored Proce	edures SQL (1)) ×
	Refresh	Ctrl+R
	Create New	Ctrl+N
	🔀 Delete	Delete
	Move	Ctrl+Shift+E
	Copy	Ctrl+Shift+P
	🐗 Run	Ctrl+F5
	Compare	Ctrl+Shift+O
	Previous Object	t Alt+F11
	Locate Object (Pick Database) Ctrl +F2
	Advanced	•
	UDPs	Ctrl+F4
	Test Parameter	Values
	Copy, Test and	Rollback
	Add to Build	
	Show History	
	Show Testing H	istory

Another feature available related to change tracking is the ability to "label" objects in a database. If you've used some source control systems that support the concept of labeling, this is similar: you're essentially taking a current snapshot and marking it with a "name" which can be later used to recall the items as of when the label was created.

With SQL-Hero, this can be especially useful when you first start working with an established database. You will have no history for it to begin with, so an initial label can be useful as a starting point.

To create a label, use the "Apply Label" command (S) on the History tool:

🐙 SQL-Hero											
Editor	Data Compare	Schema Compare	Differences	History	Notificatio	ns Cr	eate Data	Testing	Region Diffs	Tracing	M
Style: Deta	ils (all versions)	✓ Max	Objects: 100	-	Search 🔀	Clear	📡 Show Cr	riteria 🔊	Apply Label	间 DB Settin	igs
										Search	Crite

This brings up a dialog where you must enter a name for your label, plus you can tweak which objects will be included in the label operation:

📐 Label ()bjects										
This proc backgrou added to t	ess may take s nd. All databas he history log,	ome time and will e objects that me labelled with the i	be will therefore be exe et the selected criteria v dentifier you provide be	ecuted in the vill be low.							
	Database:	RD-Developmen	ıt 💌	[
	Label:	Baseline 4/3/10									
ALL Stored User f	ALL User functions (table) Check constraints Stored procedures User functions (multi) Defaults User functions (scalar) Views Foreign keys										
	N		Ourse/Sahama	Ture							
	CustomerTyp	e	dbo	U							
	PK_Custome	er958B614C0	dbo	РК							
	up_SomeCha	ngeProc	dbo	Р							
	up_SomeSlov	vProc	dbo	Р							
4 select	ed		<u>O</u> K	<u>C</u> ancel							

When you select OK, the labeling operation will start in the background. Progress will be reported on the status bar. Search results after the label operation, searching on the object sp_SomeSlowProc could look like this:

					S	earch Results
Database	Action	Name	Comment	Version	Date	Text
RD-Development	L	up_SomeSlowProc	LABEL: Baseline 4/3/10	4	4/3/2010 22:33:36.600	/****** Object: Stor
RD-Development	U	up_SomeSlowProc		4	4/3/2010 21:38:18.407	ALTER PROCEDU
RD-Development	U	up_SomeSlowProc		3	4/3/2010 21:37:10.503	ALTER PROCEDU
RD-Development	U	up_SomeSlowProc		2	4/3/2010 21:36:43.863	ALTER PROCEDU
RD-Development	I	up_SomeSlowProc	Created	1	4/3/2010 15:58:10.347	CREATE PROCED

Notice that the action is "L" for labeling. (It is "U" for an update, "I" for an insert/create.) The comment field shows the label name – but also "Created" because that was the comment the user entered when prompted to do so, when the object was first created.

Local Recorded SQL

SQL-Hero version 0.9.6 introduces a new feature which lets anyone track all commands issued against *any* database: whether it's a global connection or not in SQL-Hero. Simply toggle the "Record SQL" command ([•]) to "on," and then all commands (DDL and DML) issued in all SQL-Hero Editor windows will be recorded.

Editor Data Compare Schema Compare Differen	ices History Notifications Create Data Testing Region Diffs Tracing I
Category: SQLHero Database: Local-SQLHero	▼ V Filter 🈁 🕨 Execute (F5) Name Filter: 🔽 C
File • Edit • Tabs • Actions • Object • Favorites •	Templates • Manage • 🏦 Find: count • Replace:
2 🕺 🕲 👔 😫 😒 💁 🗊 🐿 💱 🍫 🐑 🖱	N 🖄 📲 🖼 😨 🗐 🐒 🖺 😂 🐼 🕞 🗚 🎲 🖉 💽 🖄 🧐
Stored Procedures GodeXSpecific.up_AppSecurityPrivilege_GetAll Result CodeXSpecific.up CodeList BvKev	SQL (1) × Record SQL

To go back and review what was captured, use the Manage -> Recorded SQL History option, available on the toolbar menu:

Editor Data Compare Schema Compare Differences Histo	ory	Notifications Create Data
Category: SQLHero Database: Local-SQLHero	•	🍸 Filter 🔭 🕨 Execute
File ▼ Edit ▼ Tabs ▼ Actions ▼ Object ▼ Favorites ▼ Templates ▼	Man	age 🕶 👫 Find: count
🛛 🖬 📭 😭 🗩 🎜 🖓 🖄 😵 😓 🐂 🕺 🦓 🚽	22	Manage Users
	A	Tombstone Settings
E B. CodeXSpecific.up_AppSecurityPrivilege_GetAll	8	Source Control Settings
	2	Global Settings
El···suu CodeXSpecific.up_CodeList_d Fl···suu CodeXSpecific.up CodeList ForList	Ð	Manage UDP's
	jII.	Server Logs
⊡ CodeXSpecific.up_CodeList_u	8	Recorded SQL History
U CodeXSpecific.up_MachineParameter_ByKey	_	

This brings up a new tab in the editor space which summarizes captured commands, by date and database.

Schpt SQL Quick Find:		🔎 💽 Sh	ow Last 1000 🔸 🗕
Period / Database /			
Period : A. Today (1 item)			
Database : Local-SQLHero (l item)	No. of Concession, name	
Added On 7	# Lines	Object Name	SQL
4/1/2010 07:46:03.000	1	ContainerType	SELECT * FROM ContainerType

From here you can search for string matches (Object Name and/or SQL), or script the text back out to an Editor window, for the same database the SQL was captured against. Note that information captured in the local recorded history is not necessarily stored in the central repository and as such, the tool window will not find commands captured only in the repository.

Region Differences and Compliance

Consider the scenario where you have three different databases to support an application: a development database, a staging database, and a production database. Imagine a user has reported a problem in the staging database, and a developer has gone in to do investigation. As part of the debugging process, they change an existing object with the intent of reversing their change when complete. During their work they create some new debugging objects, modify some data – and eventually resolve the issue. Let's suppose they forget to reverse the change that was intended to assist in debugging. Is there a way to find out quickly if anything's wrong, before a user does?

With SQL-Hero, there is more than one way to uncover a problem like this. One way is using automated unit testing. That technique is covered in the whitepaper on unit and performance testing. Another way is using the Region Differences tool.

The premise here is that you often have an assumed sequence through which objects are moved – typically from development, to staging, to production. If you see a change that's in staging but not in development, you have a *compliance issue*. Or if you have an object that's in production and not in staging, that's another kind of *compliance issue*. The Region Differences tool consumes the change tracking data in the SQL-Hero repository and reports these compliance issues.

In the example below, we're comparing the databases RD-Development and RD-QA. You can list two or more databases, in the box highlighted in green, provide optional filter criteria in the box highlighted in red, and then click on Refresh, highlighted in blue:

1	K 2 0	L-Hero									
	E	ditor	Data C	ompare	Schema Compare	Differences	History Notifications	Create Data Testin	g Region Diffs	Tracing Monite	oring Settings 🗸 🗸
	Repo	rt Type	Current	t Differenc	es 🔻 🕏 Ref	fresh 🔽 Tog	gle Selection For Current (Only For All Selected	🍸 Filter 😪 Pu	blish 📕	
			Туре	Schema	Name	From DB	From Last Mod	From Mod Machine	From Mod By	To DB	
	►		Р	dbo	up_SomeChangeProc	RD-Developm	ent 4/2/2010 23:44:03.340	CODEX07	CODEX07\Administra	tor RD-QA 4/2	Enter the list of database aliases (comma separated or on new lines),
			Р	dbo	up_SomeNewProc	RD-Developm	ent			RD-QA 4/2	in the expected sequence of object promotion (e.g. "Development, QA,
											Production"):
											RD-Development,RD-QA
											RD-Development
											RD-QA
											Only include objects last changed by me
											marked to ignore
											Objects changed between
											3/27/2010 8:34:07 AM
											and:

I	Туре	Schema	Name	From DB	From Last Mod	From Mod Machine	From Mod By	To DB	To Last Mod	To Mod Machine	To Mod/Assigned	Reason
1	Ρ	dbo	up_SomeChangeProc	RD-Development	4/2/2010 23:44:03.340	CODEX07	CODEX07\Administrator	RD-QA	4/2/2010 23:44:49.677	CODEX07	CODEX07\Administrator	
T	Р	dbo	up_SomeNewProc	RD-Development				RD-QA	4/2/2010 23:44:39.887	CODEX07	CODEX07\Administrator	

The way to read these result is that the stored procedure up_SomeChangeProc is newer in the RD-QA database than in the RD-Development database (compare the Last Mod dates). The procedure up_SomeNewProc is in RD-QA and not in RD-Development.

Names of who modified it last and the machine from where it was modified are included. Note that the machine name is included to cover the case where SQL Authentication may be used - otherwise you could see a SQL login name that doesn't tell you who really made the change. Note that as part of SQL-Hero user management, you can relate machine names to users, allowing SQL-Hero to come up with a user's name even if SQL Authentication is being used. User management is found on the Editor tool under the Manage -> Manage Users toolbar menu:

SQL-Hero						
Editor	Data Compare	Schema Compare	Differences	History	Notifications	Create Data
Category:	RD	Database: RD-D	evelopment		🖌 🍸 Filter 🈤	Execute
File ▼ E	dit• Tabs• Act	ions • Object • F	avorites 🔹 Temp	olates 🕶 Mai	nage 🔹 🐴 Fir	nd: count
2 SOD	V3 (BL 52 53	5) 🖄 🖆 🛠		💥 🛓 😫	Manage Users	
E. kult Store	ad Procedures			A	Tombstone Se	ttings

Note the "Default Machine":

2	User Maintenance					
6	🚽 OK 🕤 Cancel 👌 Ap	ply Credentials				
				Users	I	
	Username	IsActive	ls Admin	E-Mail	SMS E-Mail	Default Machine
	CODEX07\Administrator			joelc@codexframe		-
	dsfsdf	◄				
	fsmith		Π			-

This association to a user, and ultimately a user's e-mail address is important for another feature that the Region Difference's tool provides: the ability to schedule region difference checking and deliver the results via e-mail. This option is available on the Region Diffs toolbar:

1.2	K SQL-Hero	R.												
	Editor	Data C	ompare	Schema Compare	Differ	ences His	story	Notifications	Create Data	Testing	Region D	Diffs Traci	ng N	1oni
	Report Type	Curren	t Differen	ces 👻 🕏	Refresh		e Select	ion For Current C	Only For All	Selected	Y Filter	Rublis		
		Tune	Schema	Name	-	From DR		From Last Mod	Erom Mod	Machine	Erom M	lod By	To DR	1

Below is an example where we've set up a scheduled region difference compare between RD-Development and RD-QA for all objects changed in the last 40 days. "Replace Existing" causes any outstanding region difference records to be removed, as opposed to accumulating a growing list. Optional name filters can be applied (here, we're filtering out objects that are in the "Debug" schema). Scheduling information is also supplied here.

Region Difference Schedules									
🛃 OK 🕤 Cancel Apply	/ 🗙 Delete 🟚	Refresh							
Alias Sequence List	Days Since Change	Replace Existing	Include Name	Exclude Name	Next Run	Frequency	Status	Updated By	Updated On
RD-Development,RD-QA	40			^\[?debug\]?\.	04/02/2010 12:00 AM	Weekly	Scheduled	CODEX07\Administrator	04/03/2010
*		N							

This scheduled task essentially goes out and "publishes" region difference records. These can then be picked up by the SQL-Hero notification system. There is a separate whitepaper that covers notifications, but to close the loop on region differences, note that it is rather easy to set up a notification that would deliver an e-mail to all people who are involved in a compliance issue (or a fixed distribution list). Here's an example of what one of those e-mails can look like:

Respond			A	tions	Junk E-mail	Options	194	Find	OneNote			
From: Notifications@5Q/bero.null Sent: Sat 4,3/2010.832 To: jpek@codesfinemenk.com Sent: Sat 4,3/2010.832												Sat 4/3/2010 8:32 AM
Suppress region Uniference - Computing										Ĩ		
	Туре	Databa	se Schema	Name	Event Date	Processed			Failure / Message	User	Machine	e Object Type
	Region Difference	RD-QA	dbo	up_SomeChangeProc	4/2/2010 11:44:49 PM	4/3/2010 8:26:55 AM	Object is n CODEX07	ewer in 'RD \Administra	-QA' (modified by CODEX07\Administrator, CODEX07), than in 'RD-Development' (modified by ttor, CODEX07)	CODEX07\Administrator	CODEX07	Stored Procedure
				up_SomeNewProc	4/2/2010 11:44:39 PM	4/3/2010 8:26:55 AM	Object exi	ts only in 'F	2D-QA' (modified by CODEX07\Administrator, CODEX07), not in 'RD-Development'.	CODEX07\Administrator	CODEX07	Stored Procedure
Times are shown using GMT-7												