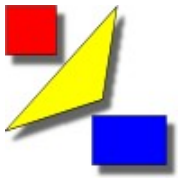
Schema Version Control for Oracle (SVCO)

Version: 3.0.0

End-User Documentation



SUMsoft Solutions

Introduction

What Is Version control or Revision control?

Revision control (also known as version control (system) (VCS), source control or (source) code management (SCM)) is the management of multiple revisions of the same unit of information. Version control is the process of managing and tracking changes, and is essential regardless of the size of your organization or the type of software you develop.
Generally speaking, version control tools provide these features:

- **Central repository**
  Source control tools provide a central repository to host all files, including source code, executables, documentations, and other files such as images. Project team members can access the central location from their own workstation. This central repository serves as the "official" copy of the project files.

- **Change tracking**
  Version control tools keep track of the changes made in a file when it's versioned. This allows developers to reconstruct earlier versions of the file, which is useful for recovering earlier work or to get to an earlier release.

- **File difference comparison**
  Version control tools allow users to compare different versions of the file in its repository, or to compare the file in repository against the (local) file that the user is currently working on.

- **History**
  Developers can examine the revision history for each file, including the comments made by the developer when revision was created.

- **Label files**
  Version control tools allow developer to mark a set of files with a label. This is used to identify all files in a release. Users can retrieve the whole set by the label name.

Using SVCO

The great advantage of SVCO is that it's fully integrated solution. The SVCO track all schema objects versions fully **automatically**. The repository and all SVCO functionality persist inside Oracle. You don't need any extra version control system and you should not give up to your favourite Oracle development tools. On the other side you could easily integrate SVCO in your own development process.
After installation you have a new user SVCO and two roles to regulate SVCO functionality and repository access. *SVCO user* is the central repository of our system. This user or schema will store all information about schema objects and their differences. Normally you will never work as *SVCO user* because all SVCO functionality is granted to both roles.

SVCO roles concept

There are 2 roles available to regulate SVCO functionality and repository access. Initially both roles are granted to SYS and SVCO users only. The role SVCO_REP_OPERATOR gives you possibility to execute fast every procedure or function from VCONTROL package. All users which get SVCO_REP_OPERATOR granted also have the read only access to the repository options and repository views. Such users are able to work only with they own objects from the repository. The second role SVCO_REP_ADMIN includes all privileges of SVCO_REP_OPERATOR role. With SVCO_REP_ADMIN role you could also execute any procedures and functions from VCONTROL package. As well you have full access to the repository options. But the

main difference is that with SVCO_REP_ADMIN role you could manage objects versions in any schemas.

Major SVCO functionality

As soon as you install SVCO on your Oracle database server and specify “Include objects” SVCO will track the objects changes fully **automatically.** Each object in the repository is held as a set of snapshots, called *object versions*, of its state at certain points in time.

Creating schema objects versions manually

To create schema objects versions manually just call procedure *Repository_Sync* from VCONTROL package.

Generating DDL script from object version

After at least one object version is exists you could generate DDL script from it. Procedure *Get_Object_DDL* from VCONTROL package performs this operation.

Delete the object version

To delete the object version you should call procedure *Delete_Object_Version* from VCONTROL package.

Tag object version

You could create a new tag (Procedure Create_New_Tag from VCONTROL package) in the repository to label needed object versions with this tag (Procedure Tag_Object_Version from VCONTROL package). To generate DDL script for objects with the same tag use procedure Get_Objects_DDL_By_Tag from VCONTROL package. Tagged object version could be untagged (remove tag) with Untag_Object_Version procedure from VCONTROL package. It's also possible to delete existing tag from the repository with help of Delete_Tag procedure from VCONTROL package.

Enabling / disabling

Sometimes you need to disable / enable SVCO, for example before / after performing big import operation on versioned objects. To disable SVCO execute procedure *Disable_Version_Control* from VCONTROL package, to enable - *Enable_Version_Control*.

SVCO repository options

Certain behaviour of SVCO could be managed with repository options. All such options are located in the table T_OPTIONS.

OPTION NAME	***DEFAULT VALUE***	***COMMENTS***
COMPRESS	TRUE	If TRUE, compress all objects data in the repository to save disk space.
COMPRESS_QUALITY	6	Speed versus efficiency of resulting compressed output. Valid values are the range 1..9, with a default value of 6. 1=fastest compression, 9=slowest compression and best compressed size.
ENCRYPT	FALSE	If TRUE, encrypt all objects data in the repository.
INDEX_SEGMENT_ATTRIBUTES	TRUE	If TRUE, emit segment attributes (physical attributes, storage attributes, tablespace, logging.
INDEX_STORAGE	TRUE	If TRUE, emit storage clause. (Ignored if INDEX_SEGMENT_ATTRIBUTES is FALSE.)
INDEX_TABLESPACE	TRUE	If TRUE, emit tablespace. (Ignored if INDEX_SEGMENT_ATTRIBUTES is FALSE.)

SEQ_IGNORE_NEXTVAL	TRUE	if TRUE, ignore last sequence value (NEXTVAL)
SYSTEM_GENERATED	FALSE	If TRUE, select indexes or triggers even if they are system-generated. If FALSE, omit system-generated indexes or triggers. Defaults to FALSE.
TABLE_CONSTRAINTS	TRUE	If TRUE, emit all non-referential table constraints.
TABLE_OID	FALSE	If TRUE, emit the OID clause for object tables.
TABLE_REF_CONSTRAINTS	TRUE	If TRUE, emit all referential constraints (foreign keys)
TABLE_SEGMENT_ATTRIBUTES	TRUE	If TRUE, emit segment attributes (physical attributes, storage attributes, tablespace, logging.
TABLE_STORAGE	TRUE	If TRUE, emit storage clause. (Ignored if TABLE_SEGMENT_ATTRIBUTES is FALSE.)
TABLE_TABLESPACE	TRUE	If TRUE, emit tablespace. (Ignored if TABLE_SEGMENT_ATTRIBUTES is FALSE.)
TYPE_OID	FALSE	If TRUE, emit the OID clause.
VALID	FALSE	If TRUE, proceed only with objects in STATUS=VALID and objects without STATUS.
VIEW_FORCE	TRUE	If TRUE, use the FORCE keyword in the CREATE VIEW statement.

Include objects to version control

Only objects listed in the table T_INCLUDE_OBJECTS will be under SVCO version control. The access to the table T_INCLUDE_OBJECTS for SVCO users is realized via V_INCLUDE_OBJECTS view.

VIEW COLUMN	*COMMENTS*
SCHEMA	Schema (user owner) name
OBJECT_NAME	Wildcards are allowed. Use # to escape wildcards e.g. T#_%

Exclude objects from version control

Objects listed in the table T_IGNORE_OBJECTS will be excluded from SVCO version control. The access to the table T_IGNORE_OBJECTS for SVCO users is realized via V_IGNORE_OBJECTS view.

VIEW COLUMN	*COMMENTS*
SCHEMA	Schema (user owner) name
OBJECT_NAME	Wildcards are allowed. Use # to escape wildcards e.g. T#_%

Repository tags

All repository tags are listed in T_TAG table. Use corresponding procedures for tag manipulation instead of this table.

TABLE COLUMN	*COMMENTS*
TAG_ID	Primary key
NAME	Tag name
COMMENTS	Comments for the tag
CREATED_AT	Date and time of tag creation

SVCO repository views

View V_OBJECT shows all objects located in SVCO repository.

VIEW COLUMN	*COMMENTS*
SCHEMA	Schema (user owner) name
OBJECT	The name of versioned object
OBJECT_TYPE	SVCO object type. See “Supported object types”
DEPENDENT_SCHEMA	Schema (user owner) name on which the versioned object is depend on

DEPENDENT_OBJECT	Object name on which the versioned object is depend on
DEPENDENT_OBJECT_TYPE	SVCO object type on which the versioned object is depend on

View V_OBJECT_VERSION consists of all object versions.

VIEW COLUMN	***COMMENTS***
SCHEMA	Schema (user owner) name
OBJECT	The name of versioned object
OBJECT_TYPE	SVCO object type. See “Supported object types”
VERSION_ID	Object version number. First object version receive 1
DB_USER	Object version creator details: Database username
OS_USER	Object version creator details: Operating system username of the client process
MACHINE	Object version creator details: Name of the host machine
DDL_TIME	When the object version was created
IP_ADDRESS	Object version creator details: IP address of the machine from which the client is connected
MODULE	Object version creator details: The application name (module) set through the DBMS_APPLICATION_INFO package or OCI

View V_OBJECT_DROPPED shows all dropped objects located in SVCO repository.

VIEW COLUMN	***COMMENTS***
DROPPED_OBJECT_ID	Internal dropped object ID
SCHEMA	Schema (user owner) name
OBJECT	The name of versioned object
OBJECT_TYPE	SVCO object type. See “Supported object types”
DB_USER	Object dropper details:

	Database username
OS_USER	Object dropper details: Operating system username of the client process
MACHINE	Object dropper details: Name of the host machine
DROPPED_TIME	When the object was dropped
IP_ADDRESS	Object dropper details: IP address of the machine from which the client is connected
MODULE	Object dropper details: The application name (module) set through the DBMS_APPLICATION_INFO package or OCI
DEPENDENT_OBJECT_ID	Dropped object ID on which the versioned object is depend on
DEPENDENT_SCHEMA	Schema (user owner) name on which the versioned object is depend on
DEPENDENT_OBJECT	Object name on which the versioned object is depend on
DEPENDENT_OBJECT_TYPE	SVCO object type on which the versioned object is depend on

View V_OBJECT_VERSION_DROPPED consists of all object versions of dropped object.

VIEW COLUMN	***COMMENTS***
DROPPED_OBJECT_ID	Internal dropped object ID
SCHEMA	Schema (user owner) name
OBJECT	The name of versioned object
OBJECT_TYPE	SVCO object type. See “Supported object types”
VERSION_ID	Object version number. First object version receive 1
DB_USER	Object version creator details: Database username
OS_USER	Object version creator details: Operating system username of the client process
MACHINE	Object version creator details:

	Name of the host machine
DDL_TIME	When the object version was created
IP_ADDRESS	Object version creator details: IP address of the machine from which the client is connected
MODULE	Object version creator details: The application name (module) set through the DBMS_APPLICATION_INFO package or OCI

View V_OBJECT_VERSION_TAG consists of object versions and their tags.

VIEW COLUMN	*COMMENTS*
SCHEMA	Schema (user owner) name
OBJECT	The name of versioned object
OBJECT_TYPE	SVCO object type. See “Supported object types”
VERSION_ID	Object version number. First object version receive 1
TAG	Tag name

PL/SQL packages

There are two packages VCONTROL and VCONTROL_ADMIN for end user operations, consider to read documentation inside package specifications. In the package EXCEPTIONS_ listed all custom defined exceptions. Other packages are for internal SVCO functionality. Please do not try to modify or recompile any SVCO objects.

Supported object types

Current version of SVCO supports the following Oracle schema object types:

- LIBRARY
- TYPE (spec and body)
- DB_LINK
- CLUSTER
- TABLE

- TRIGGER
- INDEX
- INDEXTYPE
- FUNCTION
- PROCEDURE
- VIEW
- MATERIALIZED VIEW
- MATERIALIZED VIEW LOG
- SEQUENCE
- SYNONYM
- PACKAGE (spec and body)
- COMMENT
- OBJECT_GRANT

Other object types are planed for the next SVCO release.

We would appreciate any comments and suggestions concerning the product.