SQL Provisioner Plugin

The *SQL Provisioner* exports records to a SQL database.

Capabilities

Provisionable Objects	• People • Groups • Reference Data (see below)
Provisioning Eligibility	• Eligible: Full record written • Ineligible: Core record written • Deleted: No record written
Provisioning Keys	✗ Not Supported
Bi-directional Reprovision All	✗ Not Supported

Configuration

1. SQL Provisioner is part of the *SqlConnector* Registry Plugin, which must be activated.
2. Create a new, empty database where the Provisioner will write to. This database is the *Target Database*.
    a. The Target Database must be available when the configuration is saved (below), as the plugin will attempt to apply the database schema and sync reference data.
3. If there isn't already a Server defined in the Servers Registry for the desired Target Database, add a new one.
    a. The *Plugin* for the Server is `CoreServer.SqlServers`.
    b. Select the appropriate RDBMS type. Currently, only *Postgres*, *MariaDB*, and *MySQL* are fully supported. Other options are supported by the underlying framework but are not regularly tested.
4. When adding a new Provisioning Target, the Plugin is `SqlConnector.SqlProvisioners`.
5. After creating the Provisioning Target, select the appropriate target Server, as added above.
6. The provisioner is also configured with a *Table Name Prefix*, which is prepended to the Target Database table names (as defined below). Absent a use case to change it, it is recommended that the default value be left in place. The Table Name Prefix must be alphanumeric and end in an underscore (_).

Using *Table Name Prefix* to provision multiple SQL Provisioner's worth of data to the same Target Database is currently supported, but not recommended. The underlying framework no longer supports table prefixes, resulting in a variety of unsupported workarounds to implement this functionality. It is unclear if this capability will be supportable in the long term.

Database Permissions

SQL Provisioner requires permissions to create and alter tables, and manage all rows within those tables.

Sample Postgres GRANT

```
SQL> GRANT CREATE, SELECT, INSERT, UPDATE, DELETE, TRUNCATE ON ALL TABLES IN SCHEMA reporting_database TO
user_role;
```

Target Database Schema

Reference Data

In addition to the various operational records (People, Groups) the SQL Provisioner will write some *Reference Data* to the Target Database. Reference Data can be used to understand foreign keys in the provisioned operational data. Reference Data is fully provisioned when the plugin is initially configured, and then updated whenever there are changes to the Reference Data. The tables provisioned as Reference Data are available in the Table Inventory, below.

Note that while Reference Data is updated in real time, there is no further event issued on these changes (ie: there will be no change reflected on the Primary Objects that link to the Reference Data via foreign keys).

Table Inventory

The Target Database Schema is a subset of the regular database schema, and only reflects records that are ordinarily eligible for provisioning. Table names are prefixed with `sp_` (unless configured otherwise) to distinguish them from operational tables, and also to avoid conflicting with existing code (technical: to avoid auto-binding to existing models).

Table	Type	Derived From	Available Since
sp_ad_hoc_attributes	Operational	ad_hoc_attributes	v5.0.0
sp_addresses	Operational	addresses	v5.0.0
sp_cous	Reference	cous	v5.0.0
sp_email_addresses	Operational	email_addresses	v5.0.0
sp_external_identities	Operational	external_identities	v5.0.0
sp_external_identity_roles	Operational	external_identity_roles	v5.0.0
sp_group_members	Operational	group_members	v5.0.0
sp_group_owners	Operational		v5.0.0 *Provisional*
sp_groups	Operational	groups	v5.0.0
sp_identifiers	Operational	identifiers	v5.0.0
sp_names	Operational	names	v5.0.0
sp_people	Operational	people	v5.0.0
sp_person_roles	Operational	person_roles	v5.0.0
sp_pronouns	Operational	pronouns	v5.0.0
sp_telephone_numbers	Operational	telephone_numbers	v5.0.0
sp_types	Reference	types	v5.0.0
sp_urls	Operational	urls	v5.0.0

⚠ The Target Database Schema will not automatically be removed if the plugin configuration is deleted. Drop the schema manually if it is no longer required.

Manually Updating the Target Database Schema and Reference Data

The Target Database Schema and Reference Data are automatically updated whenever the SqlProvisioner configuration is saved (PAR-SqlProvisioner-1, PAR-SqlProvisioner-2). It is also possible to manually perform either operation by viewing the plugin configuration and clicking the appropriate button.

Resyncing Reference Data will also resync all Groups, since these are effectively Reference Data as well (PAR-SqlProvisioner-3).

Deleting the SqlProvisioner will *not* delete either the target database schema or the reference data (PAR-SqlProvisioner-4). These tables must be deleted manually (if desired).

Plugin Application Rules

1. When the SQL Provisioner configuration is saved, the database schema will be applied to the Target Database.
2. When the SQL Provisioner configuration is saved, the reference data will be synced to the Target Database.
3. When Reference Data is resynced, Groups are also resynced.
4. When the SQL Provisioner is deleted, neither the database schema nor reference data is touched.

See Also

- Registry Table: sql_provisioners