Removing Analytic Plugins from StoredIQ AppStack and Gateway

1. Introduction

The StoredIQ Administrative GUI supports the deletion of Analytic Plugins (formerly called Cartridges) and Step-up actions referring to these. Nevertheless, this GUI-based deletion is restricted and applies only to those plug-ins which have not been applied in document indexing.

StoredIQ maintains a history of the application of an Analytics Plugin to a given volume, where this so-called cartridge history records all applications of all different versions of such plug-ins. The reason for this is that document indexing is a non-incremental process: In order to retain the results of previous indexation steps, such as harvests of other plugin applications, all these steps have to be repeated when applying a new plugin version. In this way, the cartridge history helps to maintain recall levels across different indexation steps.

This document explains how to remove Analytics Plug-ins from the StoredIQ Fixpack 7.6.0.19 application stacks and gateways even if these plug-ins have been applied already. Please note that, due to the non-incremental indexation, this should be done only for plugin versions which are clearly obsolete, in the sense that the search terms they deliver will not be missed in StoredIQ applications such as searching, filtering or Insights.

2. Removing from the Gateway

Via ssh or similar protocols, login into the gateway's server's console as root. From the `/root` directory, issue the commands:

```
psql -U dfuser -d dfdata
dfdata=# select model_id,name,model_path from
application_schema.cartridge_model;
```

(We show a standard psql prompt in order to distinguish these commands from Linux commands). This will give you a table with columns model_id,name(of the Analytic plugin), and model_pathi.e. the path where the plugin is being installed. Note that the plugin name is the internal name which you would specify in the plugin's `cartridge.properties` file.

Leave the psql command line interface with the `\q` command. Delete the directory of the Analytic plugin to be deleted through

```
rm -rf /deepfs/cartridge/<cartNumber>/
```

where `<cartNumber>` is the number of this plugin in the `model_path` cell. Then remove the entire row for this plugin through

```
psql -U dfuser -d dfdata
delete from application_schema.cartridge_model where model_id =
'<model_id>';
```

3. Removing from the Appstack

Via ssh or similar protocols, login into the appstack's server's console as root. Work through the following procedure from the `/root` directory.

First go to the enamel database:

/siq/bin/psql -U postgres -d enamel

From the cartridgemodel table, get the cartridge_id for the plugin to be deleted, record it and delete the plugin's entry in this table:

```
enamel=#select * from cartridgemodel;
enamel=#delete from cartridgemodel where id='<cartridge_id>';
```

(We show the both the select and the delete statement, so as to encourage controlling the delete operation). From the `cartridgehistory` table, get the `cartridgehistory_id` for the plugin to be deleted, record it and delete the plugin's entry in this table:

```
enamel=#select * from cartridgehistory;
enamel=#delete from cartridgehistory where
cartridge_id='<cartridge_id>';
```

From the `stepup_analytics_execution` table, get the row which contains the `cartridgehistory_id` retrieved in the previous step, and delete the respective entry in this table:

```
enamel=#SELECT * FROM stepup_analytics_execution WHERE
'<cartridgehistory_id>' = ANY(cartridge_history_ids );
enamel=#update stepup_analytics_execution SET
cartridge_history_ids = array_remove(cartridge_history_ids,
'<cartridgehistory_id>');
```

Note that this will leave an empty list in those cases where there is only one `cartridgehistory_id` in the list. In these singleton cases, the better option is to remove the singleton row right away:

```
enamel=#delete FROM stepup_analytics_execution WHERE
'<cartridgehistory_id>' = ANY(cartridge_history_ids );
```

From the `action` table, get the `action_id` for the step-up analytics action which contains your analytics plugin (such an action must exist if the cartridge has already been applied). Record this id.

```
enamel=#select * from action;
```

Before you can delete the action object, you have to delete two objects bearing a foreign key relationship to it. First, delete the execution of the action:

```
enamel=#select * from execution where action_id = '<action_id>';
enamel=#delete from execution where action_id = '<action_id>';
```

Then, delete the stepup analytics action:

```
enamel=#select * from stepup_analytics_action where action_id =
'<action_id>';
enamel=#delete from stepup_analytics_action where action_id =
'<action_id>';
```

Now you can delete the action:

```
enamel=#delete from action where id = '<action_id>';
```

Switch to the docket database which refers to some of the objects on the enamel database:

```
\q
/siq/bin/psql -U postgres -d docket
```

Delete docket's step-up analytics action:

```
docket=#select * from entity_enamel_action where entity_id =
'<action_id>';
docket=#delete from entity_enamel_action where entity_id =
'<action_id>';
```

Delete docket's `cartridgehistory` reference:

```
docket=#select * from entity_enamel_cartridgehistory where
entity_id = '<cartridgehistory_id>';
docket=#delete from entity_enamel_cartridgehistory where entity_id
= '<cartridgehistory_id>;
```

Delete docket's `cartridgemodel` reference:

```
docket=#select * from entity_enamel_cartridgemodel where entity_id
= '<cartridge_id>';
docket=#delete from entity_enamel_cartridgemodel where entity_id =
'<cartridge_id>;
```

Now you can restart the gateway and appstack services. When reviewing the Admin GUI, both the Analytics-plugin to be deleted and any step-up action referring to it should no longer be listed.