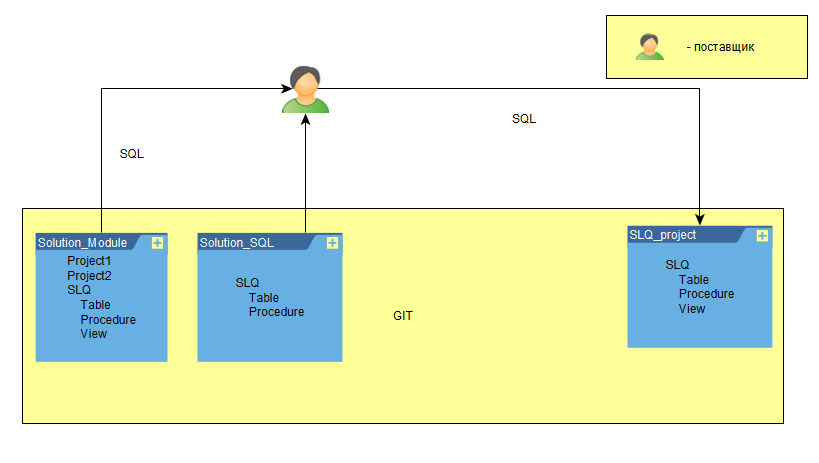
Описание проблемы: на текущий момент в отделе разработки не регламентировано хранение SQL объектов в системе управления версиями гит. Разработчики могут не создавать проекты в гите. Таким образом теряется контроль над версионностью, что приводит к ошибкам во время использования неактуальных скриптов.

Предлагаемое решение:

На данный момент существует негласное соглашение о ведении SQL в гит, которого стараются придерживаться разработчики.

**Первый вариант** — это лишь доработанное соглашение, которое было ранее:

Имеется проект со всеми SQL объектами. Пушить в данную ветку может только поставщик, которому необходимо актуализировать во время поставки данные с проекта разработчика в данный репозиторий. Со стороны разработчика в АРМе необходимо хранить SQL-объекты прямо в проектах модулей в папке или решении “SQL”. Если задача касается не модуля, то необходимо заводить отдельный проект с названием в формате “\*\_SQL”, где \* - любое название модуля латинскими буквами. Схема работы поставщика с подобной структурой.



Как по такой схеме должна происходить поставка:

1. Разработчик присылает поставщику название проекта на гит с SQL-объетами. А также имена изменившихся объектов. Данная ветка должна быть master-веткой, иначе считается, что разработчик не готов ставить данную задачу.
2. Поставщик копирует все объекты в SQL\_project и производит поставку.

При отмене поставки данной задачи:

1. Разработчик отменяет все изменения в своем проекте в ветке мастер;
2. Поставщик отменяет коммит с задачей в SQL\_project

Плюсы и минусы данной структуры:

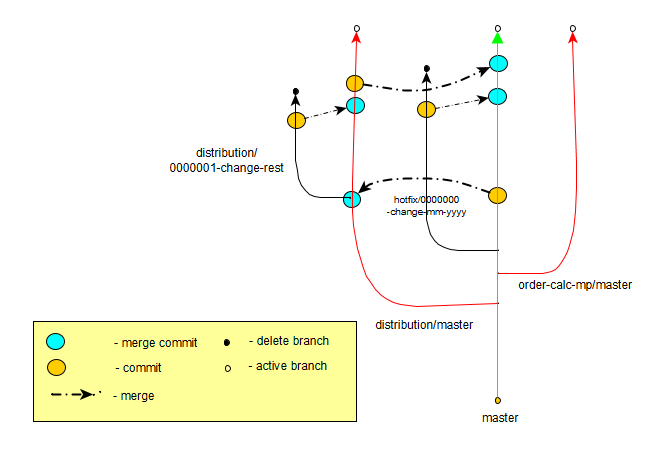
* Плюсы:
  + Наличие SQL кода сразу в проекте модуля.
* Минусы:
  + SQL код дублируется по факту в двух разных проектах: для текущей задачи, для всех объектов SQL. Может привести к путанице.
  + Отсутствие полноценного merge-запроса. На поставщика возлагается задача вставить всё то, что сделал разработчик.
  + Разработчику необходимо описать в письме поставщику: какие объекты он затронул, чтобы правильно произошёл перенос в SQL\_project.
  + Нет возможности найти какой-либо объект, если он не задокументирован на портале. Gitlab не позволяет искать по проектам.

**Второй вариант** - вместе с возможностями гит и воссоздание новой модели ветвления для SQL проекта.

Данный вариант подразумевает хранение всей SQL иерархии в одном git репозитории. Все манипуляции с внедрением/созданием новых задач производятся с помощью системы ветвления.

Основная ветка master хранит в себе боевую версию SQL объектов. Пушить в эту ветку может только поставщик (остальные лишаются такого права на уровне управления ветками). Затем от мастера идут ветки проектов (order-calc-mp/master, distribution/master) и ветки которые могут не относиться к каким-либо проектам (hotfix/0000000-change-mm-yyyy). Все ветки, которые дорабатывают или изменяют уже имеющиеся проекты создаются от master проектов (order-calc-mp/master => order-calc-mp/0000001-change-rest).

Схема ветвления:



Как по такой схеме должна происходить поставка:

1. Разработчик присылает поставщику название ветки с задачей.
2. Поставщик делает merge сначала в master ветки инструмента, а затем уже с данных мастер веток в основную ветку.

При отмене поставки данной задачи:

1. Поставщик отменяет merge commin из основной мастер ветки и из мастер веток инструментов.

Плюсы и минусы данной структуры:

* Плюсы:
  + Полный контроль версий и возможность поиска необходимых объектов в одном репозитории;
  + SQL код не просто вставляется и отправляется, а делается merge commit, с помощью которого можно отследить все зависимости;
* Минусы:
  + Отсутствие SQL кода в проекте модуля. Для удобства необходимо связь прописывать в документации.

Скрипты наката\отката. Поставка в бой