

1 Введение. Liquibase - это кроссплатформенная java-based open source система для управления миграциями баз данных. Это своего рода git для БД и реализует она приблизительно те же принципы.



Применительно к нашему проекту использование Liquibase позволит очень быстро вносить изменения в структуру БД, не затрачивая время на внесение данных вручную. Один человек работает с базой, вносит корректировки и затем делает файл changelog (если сравнивать с гитом, то это аналог коммита). Этот файл хранит в себе описание всех изменений и может быть передан другим пользователям, чтобы они могли на своих локальных компьютерах применить изменения и получить такую же точную копию БД, как и у автора изменений. Пользователь, получив changelog, выполняет скрипт, который делает всю работу за пользователя – подключается к БД, находит предыдущие изменения, создает все необходимые таблицы, наполняет их данными. Еще один плюс использования Liquibase заключается в возможности откатиться назад, отменить изменения, сделанные пользователем. Кроме того, в БД, с которой будет общаться Liquibase, создаются две дополнительные таблицы, хранящие всю информацию о внесенных изменениях: кто, когда и какие изменения вносил. Это все логируется в самой базе данных, и потом всегда можно поднять историю.

Каждый файл changelog может хранить в себе как одно, так и несколько изменений (они называются changeset). Каждое такое изменение имеет уникальные атрибуты id и author – это все для того, чтобы по их хэш сумме можно было точно установить, какой changeset применялся уже, а какой нет.

2 Установки и настройка. В первую очередь нам необходимо установить и настроить Liquibase. Идем на сайт <http://www.liquibase.org/> и качаем последнюю версию Liquibase:

На момент написания руководства актуальной была версия 3.5.3, ее и качаем:

Liquibase Downloads

Current Liquibase Core Release: 3.5.3 (Oct 13, 2016)

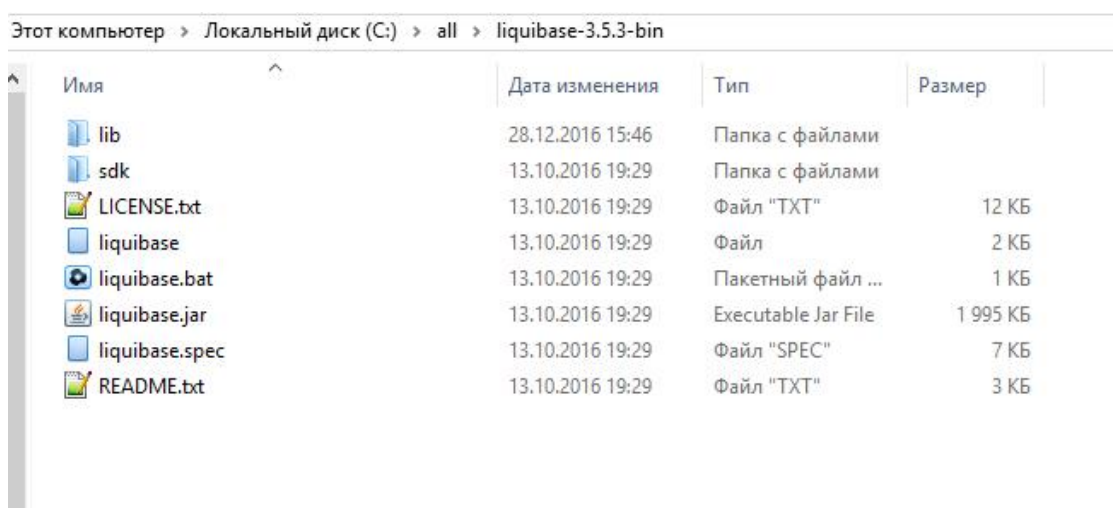
	Liquibase	Datical DB
Database Source Control/Versioning	✓	✓
Database Schema Management	✓	✓
Stored Logic Support		✓
Full ALM Workflow Support		✓
Deployment Simulation and Error Reporting		✓
Standards & Compliance Enforcement		✓
Automated Reporting backed by Complete Audit Database		✓
Full Commercial Support & Training		✓



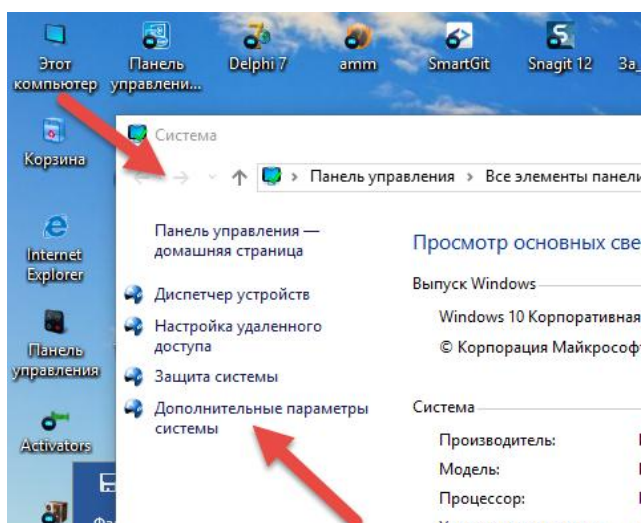
[Download liquibase-3.5.3-bin.zip](#)
[Download liquibase-3.5.3-bin.tar.gz](#)

[Learn more about Datical DB >>](#)

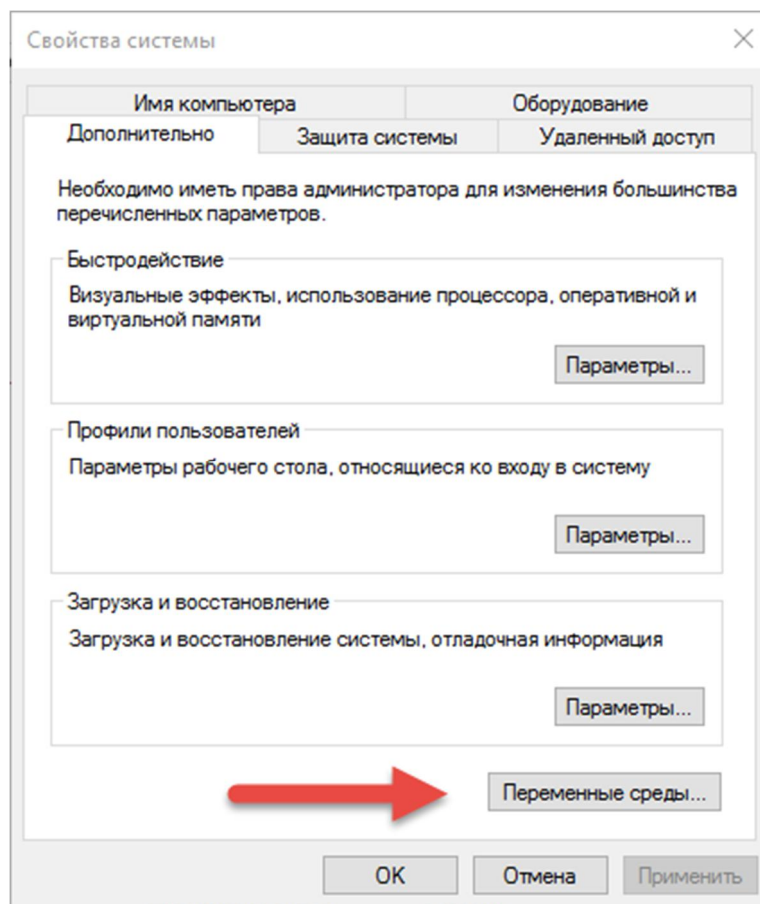
Распаковываем содержимое архива в отдельную папку. У меня это папка **C:\all\liquibase-3.5.3-bin** (очень далеко не прячьте, т.к. к этой папке еще прописывать путь в path):



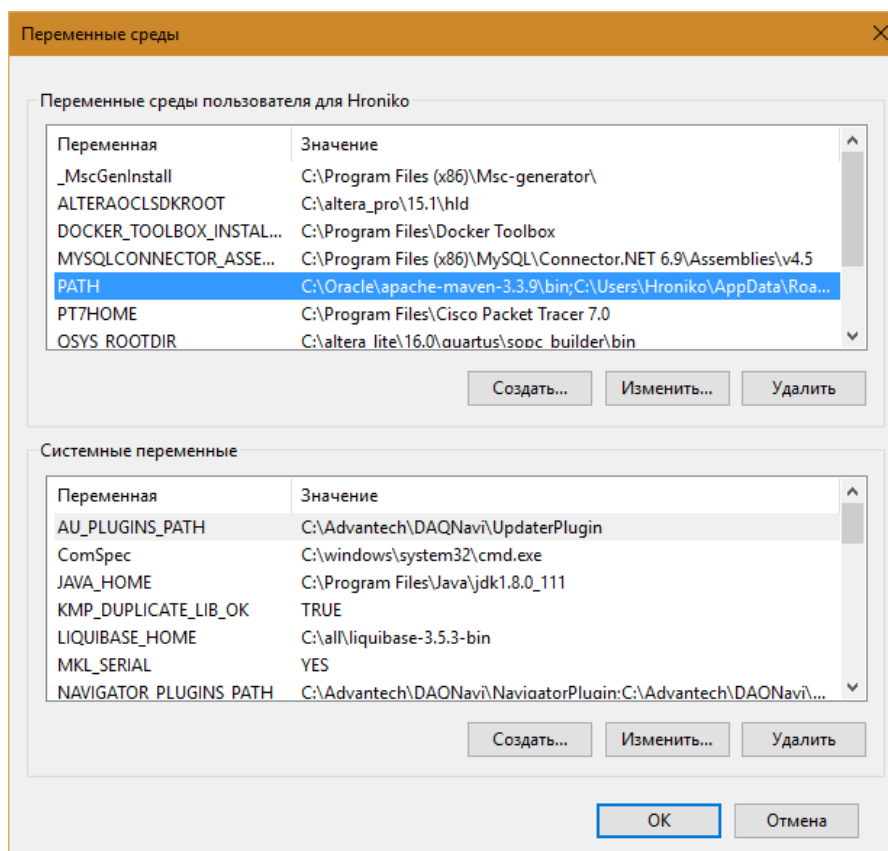
Заходим на рабочий стол, правой кнопкой по ярлыку **Мой компьютер** (Этот компьютер – в Win10), выбираем **Свойства**, затем **Дополнительные параметры системы**:



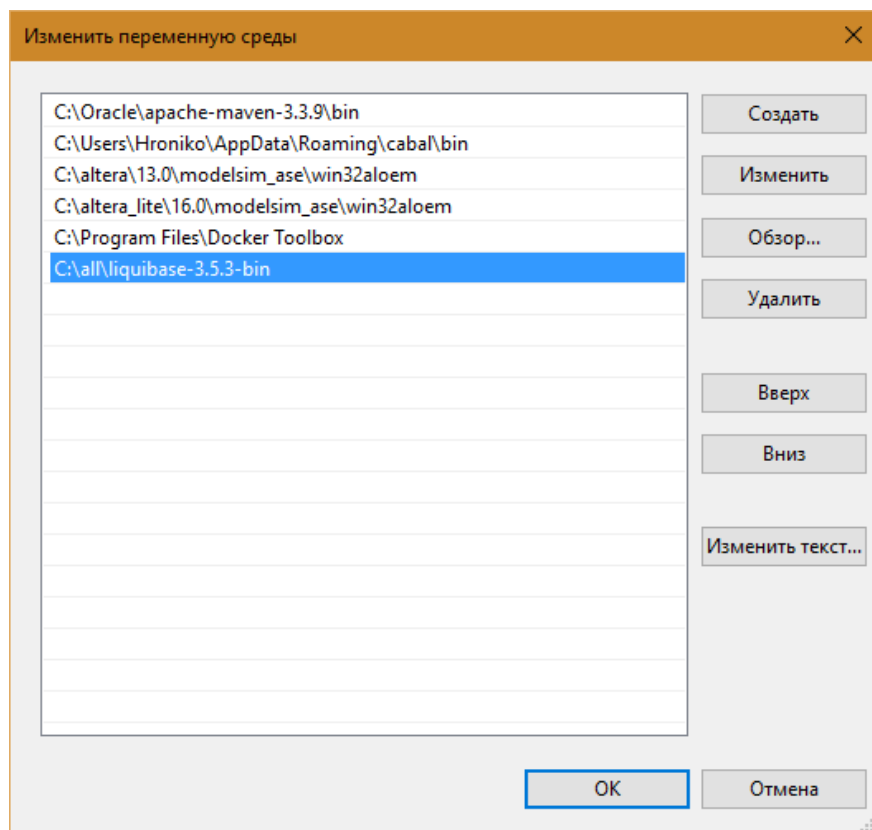
В открывшемся окне жмем кнопку «**Переменные среды**»:



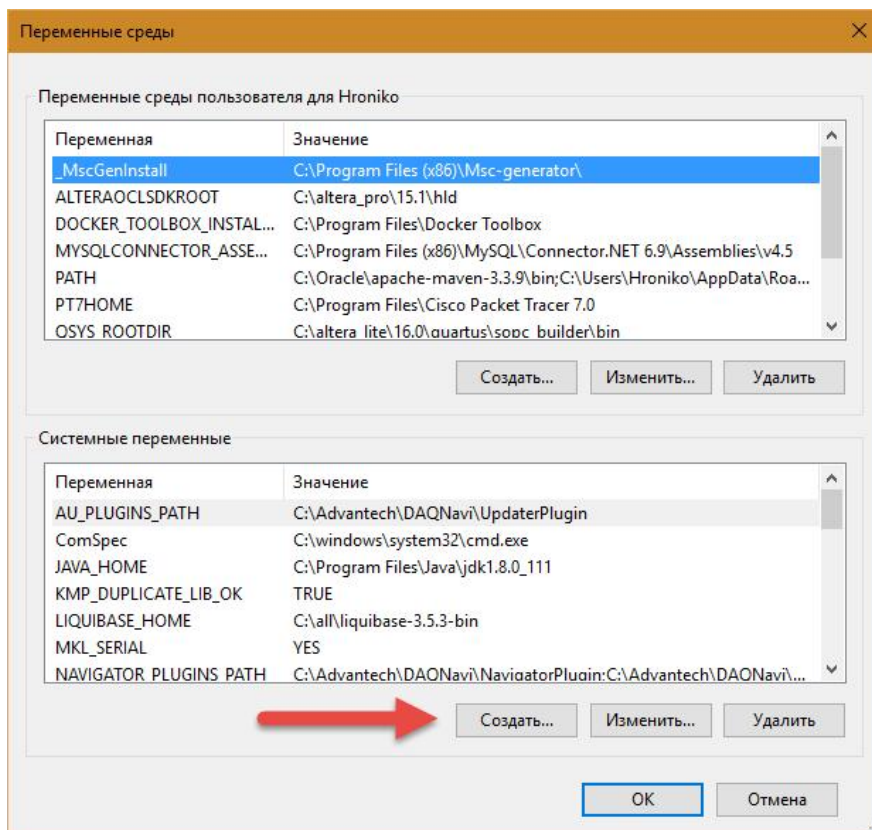
В верхней области окна находим переменную **PATH**, выделяем ее и жмем кнопку «**Изменить**»:



Добавляем полный путь к распакованной папке Liquibase:

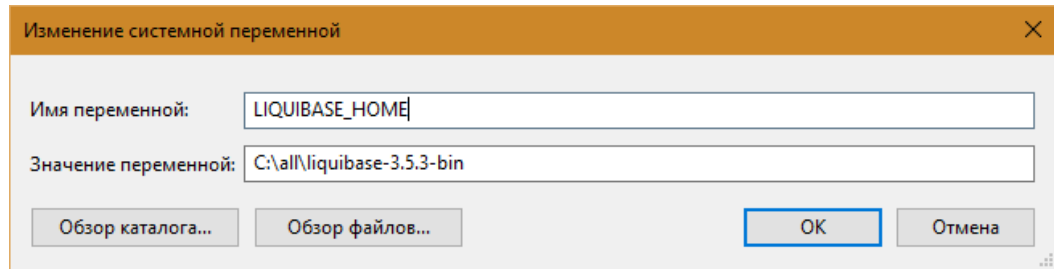


В нижней части окна нажимаем «Создать»



В качестве имени переменной вписываем **LIQUIBASE_HOME**

В качестве значения переменной указываем полный путь **C:\all\liquibase-3.5.3-bin**



Закрываем все окна и запускаем командную строку. Выполняем команду **liquibase --version**

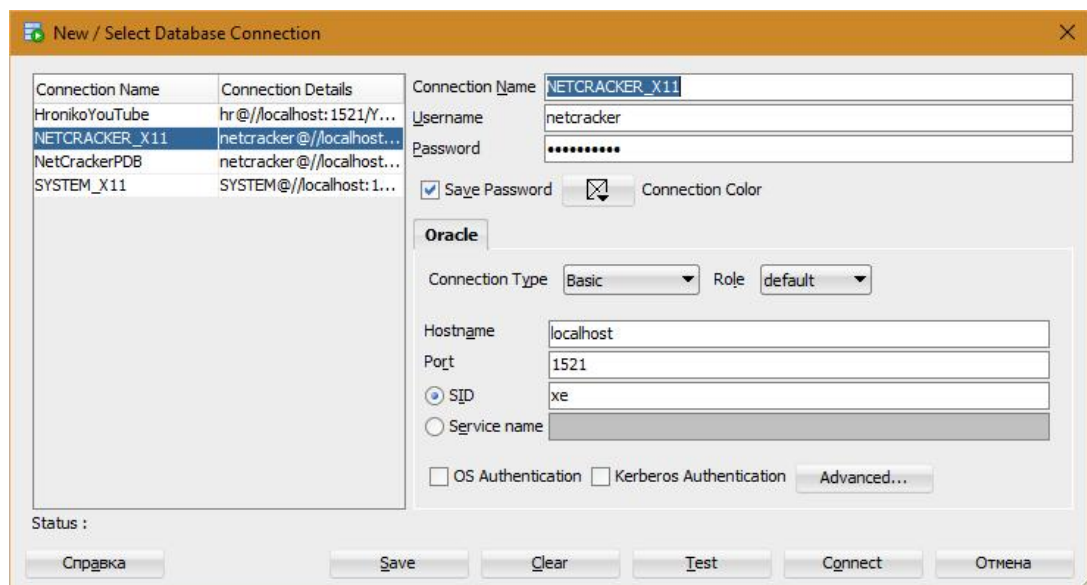
В ответ должно появиться сообщение о версии установленной программы:

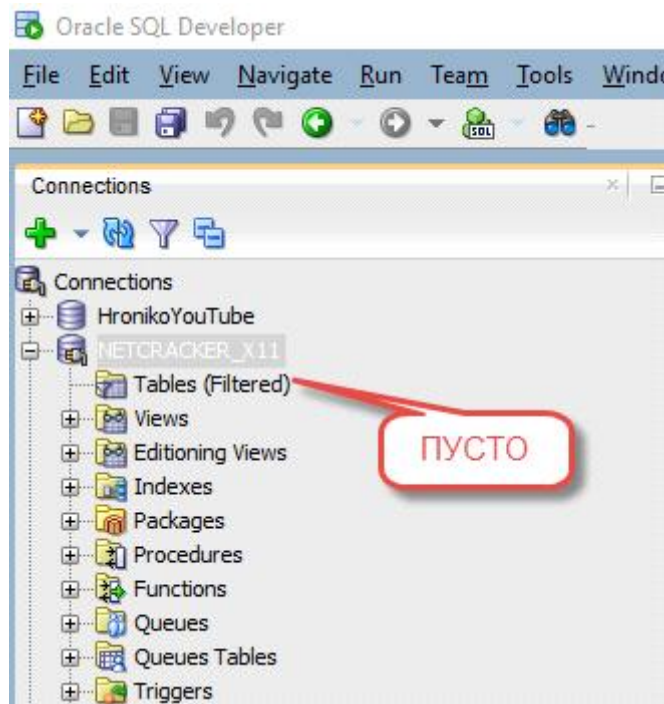


Если все сделали правильно, то с настройками переменных сред покончено.

3 Использование. Теперь, когда у нас есть установленная Liquibase, необходимо убедиться, что на компьютере установлена БД Оракл 11 версии. Я буду рассматривать работу применительно к версии Express Edition, стандартная база XE и пользователь **netcracker** с паролем **netcracker**, слушающая порт 1521 для подключения клиентов.

Выполним подключение к БД при помощи SQL Developer и убедимся в отсутствии таблиц в базе:





При использовании Liquibase изменения структуры базы данных можно хранить в changelog файлах в форматах XML, YAML, JSON или SQL. Я использовал SQL формат, так как Костя уже подготовил SQL скрипты для создания структуры EAV модели в БД, наполнения БД и удаления таблиц, и мне не потребовалось писать все ченджсеты с нуля:

Имя	Дата изменения	Тип	Размер
ncc_drop_all.sql	27.12.2016 11:30	SQL Text File	1 КБ
ncc_fill_tab.sql	27.12.2016 14:29	SQL Text File	9 КБ
ncc2_create.sql	27.12.2016 14:25	SQL Text File	3 КБ

На основе Костиных файлов я сделал два changelog файла: один для создания и наполнения базы, второй для удаления всех данных (можно было обойтись и откатом, конечно). Ниже показана структура папки с подготовленными файлами для создания таблиц и наполнения данными:

Имя	Дата изменения	Тип	Размер
autoupdate.bat	29.12.2016 12:39	Пакетный файл ...	1 КБ
createLiquibaseScript.sql	28.12.2016 23:55	SQL Text File	12 КБ
Liquibase.properties	29.12.2016 11:24	Файл "PROPERTIES"	1 КБ
ojdbc6-12.1.0.2.jar	28.12.2016 21:33	Executable Jar File	3 606 КБ

Файл **autoupdate.bat** представляет собой обычный батник, выполняющий команду **liquibase update**

По этой команде Liquibase просматривает структуру файлов в текущей папке и ищет файл **Liquibase.properties** – в нем я указал настройки для команды update, чтобы не писать все многочисленные параметры все время:

driver: oracle.jdbc.OracleDriver ---- тип драйвера БД

classpath: ojdbc6-12.1.0.2.jar ---- путь к драйверу, я положил его в текущую папку

databaseClass: liquibase.database.core.OracleDatabase ---- класс Liquibase

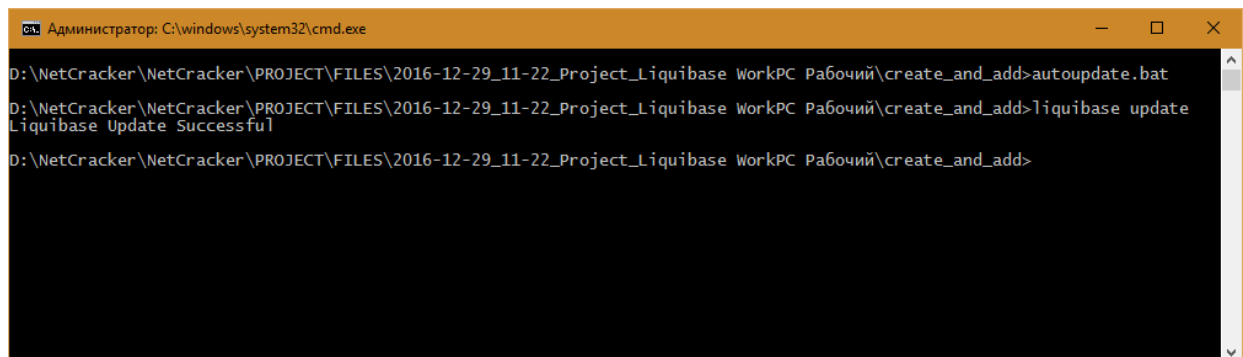
url: jdbc:oracle:thin:@localhost:1521:XE ---- адрес к серверу, у нас локальный сервер Оракл

changeLogFile: createLiquibaseScript.sql ---- путь к ченжлогу

username: netcracker ---- имя пользователя БД

password: netcracker ---- пароль пользователя БД

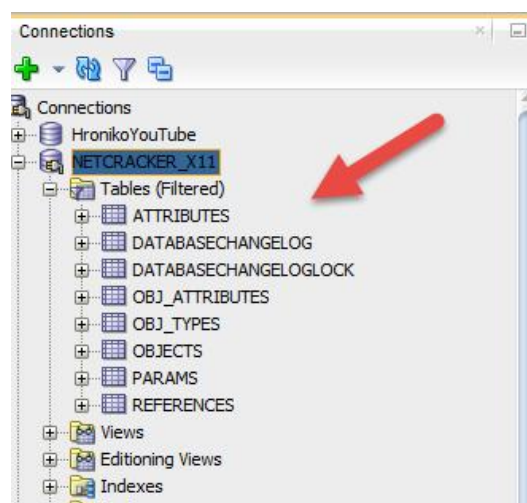
Запускаем командную строку, переходим в папку со скриптом и выполняем bat-файл:



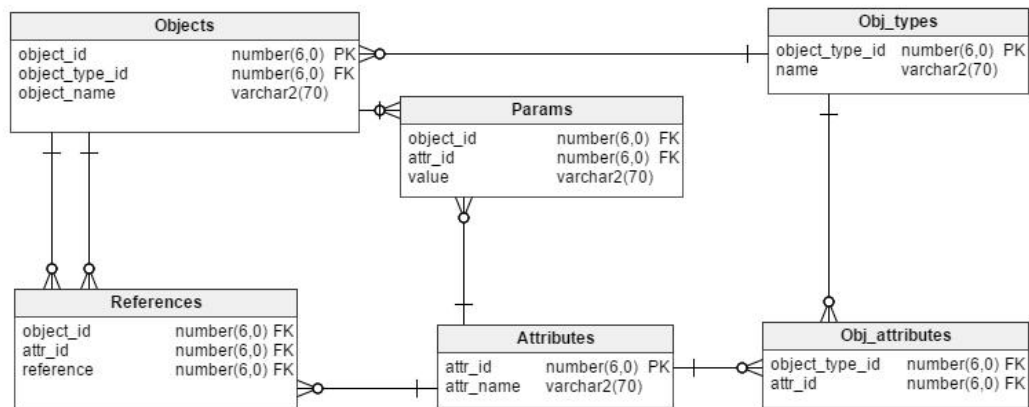
```
Администратор: C:\windows\system32\cmd.exe
D:\NetCracker\NetCracker\PROJECT\FILES\2016-12-29_11-22_Project_Liquibase WorkPC Рабочий\create_and_add>autoupdate.bat
D:\NetCracker\NetCracker\PROJECT\FILES\2016-12-29_11-22_Project_Liquibase WorkPC Рабочий\create_and_add>liquibase update
Liquibase Update Successful
D:\NetCracker\NetCracker\PROJECT\FILES\2016-12-29_11-22_Project_Liquibase WorkPC Рабочий\create_and_add>
```

Бат-файл запустит Liquibase, она отработает update, найдет файл **Liquibase.properties**, извлечет из него все настройки, подключится к базе данных, подгрузит ченджлог, пройдет по всем ченджсетам и применит к базе те из них, о которых нет информации в логе базы. При успешном выполнении работы Liquibase будет выведено сообщение **Liquibase Update Successful**.

Посмотрим, что произошло с базой. Отключимся и заново подключимся к базе через SQL Developer и посмотрим на структуру таблиц:



Легко заметить, что Liquibase корректно отработал и создал полную структуру EAV модели такого вида:



Выполним простой запрос и убедимся в наличие данных в таблице:

Worksheet Query Builder

```
SELECT * FROM ATTRIBUTES;
```

Script Output x Query Result x

SQL | All Rows Fetched: 17 in 0,03 seconds

ATTR_ID	ATTR_NAME
1	1 name
2	2 birth_date
3	3 country
4	4 sex
5	5 e-mail
6	6 additional_field
7	7 picture
8	8 login
9	9 password
10	10 friends
11	11 task_id
12	12 events
13	101 time_start
14	102 time_end
15	103 duration
16	104 task_comment
17	105 priority

И, кроме того, были созданы две вспомогательные таблицы **DATABASECHANGELOG** и **DATABASECHANGELOGLOCK**:

Worksheet Query Builder

```
SELECT * FROM DATABASECHANGELOG;
```

Query Result x

SQL | All Rows Fetched: 2 in 0,011 seconds

ID	AUTHOR	FILENAME	DATEEXECUTED	ORDEREXECUTED	EXECTYPE	MD5SUM
1 1	Hroniko	createLiquibaseScript.sql	29.12.16 15:30:03,617000000	1	EXECUTED	7:d2cd560f
2 2	Hroniko	createLiquibaseScript.sql	29.12.16 15:30:03,807000000	2	EXECUTED	7:64ba839e

Все работает))