Программный продукт "ПАРУС – Бюджет 8"

ПАРУС-PostgreSQL: Установка, конвертация, обновление

Руководство администратора

Москва 2024

Оглавление

Типовые сценарии создания/конвертации/обновления

Для **создания** новой БД – установить инсталлятор ParusPG и выполнить им создание БД (аналогично инсталлятору для СУБД Oracle).

При **конвертации** Oracle – PostgreSQL: после конвертации можно использовать инсталлятор ParusPG (рекомендуется) или сценарии _AfterConvert (если есть приложения, не поддерживаемые на сегодня инсталлятором ParusPG).

Для **обновления** имеющейся БД – установить инсталлятор ParusPG и выполнить им обновление (аналогично инсталлятору для СУБД Oracle).

<table>
<tr><td rowspan="2">Создание</td><td rowspan="2"></td><td colspan="2">Конвертация</td></tr>
<tr><td>Пользовательская
(инсталлятор ParusPG)</td><td>Служебная
(скрипты _AfterConvert)</td></tr>
<tr><td colspan="4">Установка PostgreSQL, инициализация и настройка кластера, создание служебных ролей</td></tr>
<tr><td colspan="4">Установка расширений PostgreSQL</td></tr>
<tr><td colspan="4">Создание базы данных PostgreSQL</td></tr>
<tr><td></td><td>Обновление</td><td></td><td></td></tr>
<tr><td colspan="2">Установка инсталлятора ParusPG</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>Подготовка скриптов _AfterConvert 1,2</td></tr>
<tr><td></td><td></td><td colspan="2">Установка конвертера (ppc.exe)</td></tr>
<tr><td></td><td></td><td colspan="2">Подготовка БД Oracle
(Стерилизация базы данных)</td></tr>
<tr><td></td><td></td><td colspan="2">Конвертация (PPCConvert.exe)</td></tr>
<tr><td>Создание БД инсталлятором ParusPG</td><td colspan="2">Выполнение скрипта _AfterConvert 1 и функции pg_temp.DO_AFTER_CONVERT
Обновление БД инсталлятором ParusPG</td><td>Выполнение скрипта _AfterConvert 1 и функции pg_temp.DO_AFTER_CONVERT
Выполнение скрипта _AfterConvert 2</td></tr>
</table>

В данном разделе рассмотрены вопросы установки и настройки PostgreSQL.

Процесс конвертации описан в главе "Конвертация Oracle-PostgreSQL", работа с инсталлятором для PostgreSQL – в главе "Инсталлятор для PostgreSQL. Создание и обновление БД".

Выбор дистрибутива PostgreSQL

Возможны следующие варианты:

- Оригинальная версия PostgreSQL Core (Версии для Windows поддерживаются EnterpriseDB или другими вендорами).
- Из репозитария операционной системы. Установка и обновление выполняется с помощью штатного менеджера пакетов (для Windows существует, например, неофициальный Chocolatey).
- Сторонний дистрибутив (Postgres Pro, EnterpriseDB, BigSQL)

Поддерживаемые версии СУБД PostgreSQL – 9.6.x и выше (на сегодня до 15.x).

Рекомендуется выбор 64-х разрядной СУБД PostgreSQL из состава дистрибутива (для Linux) или Postgres Pro (для Windows).

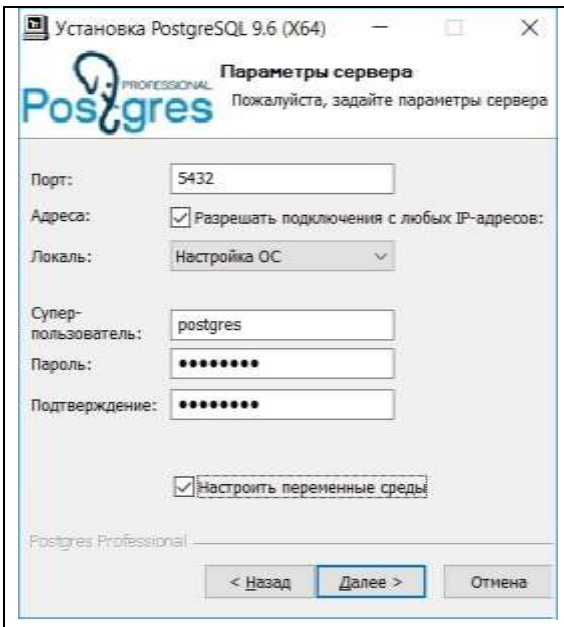
Установка PostgreSQL

Далее рассматривается установка СУБД PostgreSQL на примере Postgres Pro Standard 9.6 для Windows и различные варианты для Linux.

Перед установкой рекомендуется проверить, что языковые и региональные настройки по умолчанию соответствуют региону "Россия". Для Windows – "Панель управления - Региональные стандарты", для Linux – команда "locale" должна возвращать значения параметров "ru_RU.UTF-8". В противном случае потребуется задавать нужные значения вручную, например, при выполнении команд управления базой данных.

Установка на Windows

1. Скачать дистрибутив требуемой версии: https://postgrespro.ru/windows.
2. Запустить инсталлятор, например, PostgresPro_9.6.x.x_64bit_Setup.exe.
3. Указать место установки, например, C:\PostgreSQL\9.6.
4. Выбрать каталог данных – место, где будет развернут кластер (кластер – область хранения баз данных – аналог базы данных в Oracle), например, "C:\PostgreSQL\9.6\data" (см. Инициализация и настройка кластера). Рекомендуется выделить отдельный физический диск/массив.
5. Задать параметры сервера:



"Порт" – порт прослушивателя, который будет указываться в настройках клиентских соединений, в том числе Конвертера и WIN-клиента ПП "ПАРУС-Бюджет 8". Значение по умолчанию "5432", рекомендуется оставить без изменений, если нет особых причин изменить.

"Разрешить подключение с любых IP-адресов" – для ознакомления и демонстрации желательно включить. Будет разрешено подключение со всех компьютеров в сети и открыт порт в брандмауэре.

Важно! При реальной работе необходимо выполнить настройку доступа вручную в соответствии с требованиями безопасности. См. "Настройка доступа к PostgreSQL".

"Локаль" – **Важно!** – использовать "Настройка ОС" для ОС с русским языком интерфейса, для других – в том числе с MUI, выбрать "Russian, Russia".

"Супер-пользователь" – администратор БД (аналог SYS в Oracle), обычно "postgres".

"Настроить переменные среды" – задать системные (т.е. для всех пользователей) переменные для упрощения взаимодействия с БД (переменные PGDATA = C:\PostgreSQL\9.6\data, PGDATABASE = postgres и др. но не все возможные, а также в переменную PATH добавиться путь с исполняемыми файлами СУБД C:\PostgreSQL\9.6\bin). Если эти переменные заданы – их можно явно не указывать при выполнении команд, например, "psql.exe" будет эквивалентно "C:\PostgreSQL\9.6\bin\psql.exe -d postgres -U postgres -h localhost -p 5432".

Далее, в настройках производительности выбрать "Провести оптимизацию параметров" в соответствии с имеющимся объемом ОЗУ (при выборе "Использовать параметры по умолчанию" будет заданы минимальные необходимые значения).

По окончании установки будет создан сервис "postgresql-X64-9.6" с автоматическим типом запуска, который будет стартовать БД (кластер C:\PostgreSQL\9.6\data).

Не запрещается (но не рекомендуется) создавать более одного кластера на сервере. Если установлено более одной версии PostgreSQL – каждая работает независимо друг от друга.

Можно при необходимости переместить кластер в другой каталог (например, в Windows потребуется изменить в реестре параметры сервиса "HKLM\SYSTEM\CurrentControlSet\Services\postgresql-X64-9.6\ImagePath" и дать необходимые права на новый каталог "icacls *X:\PostgreSQL\data* /grant "NETWORK SERVICE":(OI)(CI)F /T").

Необходимо учесть размещение пользовательских табличных пространств и при необходимости пересоздать ссылки из каталога pg_tblspc на реальные каталоги табличных пространств (для Windows команда "mklink /d /j ссылка назначение").

Установка на Linux

Вариант 1. Установка PostgreSQL из дистрибутива

Установка выполняется с помощью штатного менеджера пакетов, например для Астра Линукс (Debian, Ubuntu):

```
sudo apt install postgresql postgresql-contrib postgresql-client postgresql-server-dev-all
```

Размещение кластера по умолчанию – /var/lib/pgsql/data, настройки – /etc/postgresql/9.6/main.

Вариант 2. Установка Postgres Pro из репозитариев

[Опция] Создать пользователя postgres:

```
useradd postgres
passwd postgres
```

Если пользователя нет, то он будет создан автоматически, но без права входа в систему и домашнего каталога (выполнять команды от его имени можно будет только с помощью sudo).

Подключить репозитарий для соответствующей ОС с помощью менеджера пакетов (инструкция по подключению расположена на странице загрузки), например для CentOS:

```
sudo rpm -ivh http://repo.postgrespro.ru/pgpro-9.6/keys/postgrespro-9.6.centos.pro.yum-1.0-1.noarch.rpm
или лучше так:
yum install http://repo.postgrespro.ru/pgpro-9.6/keys/postgrespro-9.6.centos.pro.yum-1.0-1.noarch.rpm
```

Перед установкой желательно проверить доступность пакетов. В отличии от "родных" пакетов из состава ОС, которые как правило начинаются с "postgre**sql**", пакеты Postgres Pro начинаются с "postgre**spro**":

```
yum search postgrespro*
```

Выполнить установку пакетов (для CentOS 7):

```
yum install postgrespro96-server postgrespro96-contrib postgrespro96-libs postgrespro96-devel
```

Названия пакетов могут различаться для разных версий и ОС. Обязательными для установки являются следующие пакеты:

server	Сервер
contrib	Дополнительные модули (расширения)
libs	Клиентские библиотеки (libpq)
server-dev	Разработка расширений сервера – потребуется при сборке сторонних расширений. Также, обычно, содержит полезные утилиты, например, pg_config.

В теории, при установке должны быть автоматически установлены все отсутствующие в системе пакеты, от которых зависят устанавливаемые. Если этого не происходит (например, не подключен нужный репозитарий), нужно отменить установку, установить требуемые пакеты и повторить установку. Например, для CentOS 7 необходимо установить libzstd из EPEL-репозитария.

```
yum install epel-release
yum install libzstd
```

К сожалению, списка требуемых предустановленных пакетов нет.

Если СУБД поставляется в виде файлов пакетов на внешнем носителе или сетевом ресурсе (например, версии Certified или Enterprise Postgres Pro), то установку можно выполнить, предварительно создав локальный репозитарий с помощью createrepo для yum или genpkglist для apt.

Каталог установки зависит от ОС (например, /usr/local/pgsql /var/lib/pgsql /usr/pgpro-9.6).

Вариант 3. Установка Postgres Pro из исходного кода

Если используемая ОС не поддерживается Postgres Pro, то можно выполнить установку из исходных текстов (пример для RFRemix 28 (Fedora)):

1. Создать пользователя postgres (см. выше).
2. Установить необходимые для сборки пакеты:

```
dnf install uuid uuid-devel bison flex flex-devel systemd-devel readline-devel zlib-devel libxml2-devel libxslt-devel
```

   К сожалению, списка требуемых установленных пакетов нет, только требования. Если в процессе сборки или установки возникает ошибка, связанная с отсутствием необходимого пакета, необходимо выполнить "откат" (make uninstall, make clean), установить пакет, повторить сборку и установку.

3. Скачать исходный код (см. ссылку соответствующей версии на странице загрузки), распаковать архив, перейти в папку:

```
su – postgres
wget http://repo.postgrespro.ru/pgpro-9.6/src/postgrespro-x.x.x.x.tar.bz2
bunzip2 postgrespro-x.x.x.x.tar.bz2
tar xf postgrespro-x.x.x.x.tar
cd postgrespro-x.x.x.x
```

4. Запустить скрипт конфигурирования и выполнить сборку:

```
./configure --with-uuid=ossp --with-libxml --with-libxslt --with-systemd --enable-nls=ru
make world
```

5. Проверить, с какими опциями собран сервер: pg_config --configure.

   Вместо библиотеки OSSP UUID можно использовать libuuid (ключ --with-uuid=e2fs).

6. Выполнить установку:

```
sudo make install-world
```

   По умолчанию каталог установки /usr/local/pgsql

Выше рассмотрена полная сборка (world) для того, чтобы "собрались" все расширения, в том числе необходимые.

Можно выполнить только сборку/установку сервера, а затем только необходимых расширений:

```
make            # сборка сервера
make install    # установка сервера
```

На сегодня обязательными (кроме всегда присутствующего plpgsql) являются расширения: dblink, pg_variables, uuid-ossp, xml2, hstore:

```
cd contrib          # войти в каталог с расширениями
cd dblink           # войти в каталог конкретного расширения, например, dblink
make                # выполнить сборку
sudo make install   # выполнить установку (результат: .../share/extension/dblink.control)
cd ../uuid-ossp     # войти в каталог следующего расширения
... и т. д.
```

По окончании сборки не стоит удалять папку сборки. В дальнейшем она может потребоваться для сборки расширений или удаления PostgreSQL.

Вариант 4. Установка PostgreSQL с сайта проекта postgresql.org

Рекомендуется, если желаемая версия PostgreSQL недоступна в дистрибутиве.

На странице загрузки выбрать соответствующую ОС и установить согласно приведенной инструкции.

Например, установить PostgreSQL 12 на "Астра Линукс Орел" (совместим с Debian 9 Stretch).

Установить необходимые пакеты sudo apt install ca-certificates gnupg
Импортировать ключ репозитория curl https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
Добавить репозиторий в доступные (создать файл /etc/apt/sources.list.d/pgdg.list) sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt stretch-pgdg main" > /etc/apt/sources.list.d/pgdg.list'
Обновить репозитории sudo apt update # Найти пакеты для соответствующей версии PostgreSQL sudo apt search postgresql | grep 12 # Установить необходимые пакеты sudo apt install postgresql-12 postgresql-12-client postgresql-12-contrib postgresql-server-dev-12 # Если не указывать номер версии, то будет установлена последняя

Инициализация и настройка кластера

Один экземпляр работающего сервера PostgreSQL обслуживает кластер, который состоит из набора баз данных.

Перед началом работы с PostgreSQL необходимо проинициализировать кластер – обязательно указать каталог, в котором будет размещен кластер (создать хранилище, область размещения), и при необходимости задать параметры. По умолчанию кластер инициализируется в кодировке utf8.

Как правило, дистрибутивы PostgreSQL для Windows инициализируют кластер при установке, для Linux – зависит от дистрибутива – инициализация может выполняется вручную после установки.

Инициализация кластера и кодировка БД

На сегодняшний день база данных для развертывания ПП "ПАРУС-Бюджет 8" должна использовать кодировку **WIN1251** по следующим причинам:

- Более медленная работа функций PostgreSQL со строками utf8;
- Полная совместимость с win-клиентом (ANSI).

При создании новой базы данных кодировка кластера будет использоваться по умолчанию.

Можно создать кластер в utf8, а базу в win1251 (кодировка utf8 может использоваться с любой локалью), или проинициализировать кластер в win1251.

Чтобы избежать проблем при работе со сторонними утилитами рекомендуется **создать кластер в кодировке utf8** (или использовать созданный при установке), а **базу данных для ПП "ПАРУС-Бюджет 8" создавать в кодировке win1251**.

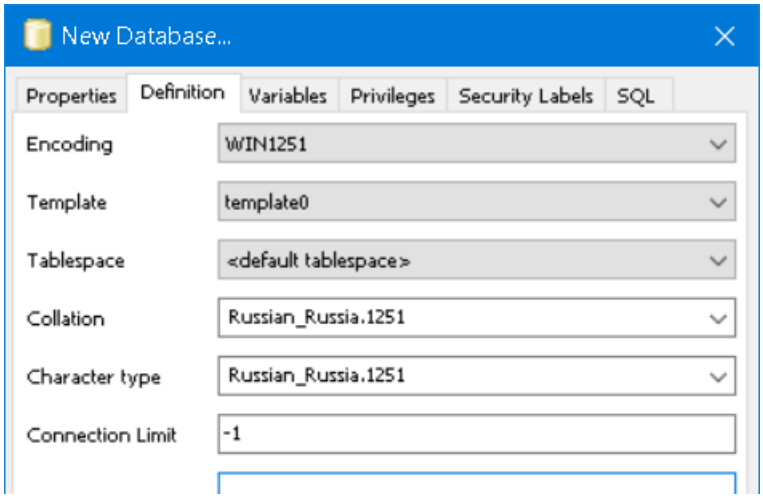
При создании кластера или БД необходимо также учитывать параметры LC_CTYPE (классификация символов) и LC_COLLATE (порядок сортировки строк).

Изменять языковые настройки кластера не рекомендуется без веских оснований (параметры LC_xxx в файле конфигурации postgresql.conf).

Windows

Вариант 1 (рекомендуется). Оставить кодировку кластера UTF-8, базу создавать из исходного шаблона (template0) с указанием кодировки WIN1251:

```
psql -c "CREATE DATABASE ppc1251 TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE =
'Russian_Russia.1251' LC_CTYPE = 'Russian_Russia.1251';"
```



Вариант 1А. Можно создать собственный шаблон (например, template_parus) и использовать его по умолчанию вместо template1:

```
CREATE DATABASE template_parus TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE =
'Russian_Russia.1251' LC_CTYPE = 'Russian_Russia.1251' IS_TEMPLATE=true;
-- здесь при желании добавить расширения (см. далее)
update pg_database set datistemplate=true where datname='template_parus';
update pg_database set datallowconn=true where datname='template_parus';
CREATE DATABASE ppc1251;
```

Linux

При установке PostgreSQL кластер может не инициализироваться.

Если кластер создан – можно оставить его в кодировке по умолчанию (обычно это UTF-8 или C), если не создан – рекомендуется создать его в кодировке UTF-8.

Перед инициализацией кластера или создания базы в кодировке WIN1251 потребуется:

- Настроить локаль.
- Настроить окружение пользователя-администратора postgres.

Настройка локали

Убедиться, что локаль WIN1251 установлена в ОС:

```
locale -a | grep ru_RU
> ru_RU.cp1251
```

Если локали 1251 нет – добавить (компиляция файла определений локали):

```
# Вариант 1:
sudo localedef -c -i ru_RU -f CP1251 ru_RU.CP1251
locale -a | grep ru_RU
> ru_RU.cp1251
```

```
# Вариант 2:
# Удалить комментарий со строки ru_RU.CP1251
sudo sed -i "s/# ru_RU.CP1251 CP1251/ru_RU.CP1251 CP1251/g" /etc/locale.gen
sudo locale-gen
> ru_RU.cp1251
# файл определений локали /etc/locale.gen может быть другим
```

Важно! Если кластер был проинициализирован и запущен, его необходимо перезапустить после добавления локали:

```
sudo systemctl restart postgresql
```

Пользователь-администратор (postgres)

Рекомендуется все действия с базами данных выполнять от имени специального пользователя postgres (кроме случаев, когда иное указывается специально). В зависимости от дистрибутива пользователь postgres может присутствовать в системе изначально, создан при установке PostgreSQL или создан Вами до его установки.

Проверка пользователя postgres:

```
cat /etc/passwd | grep postgres
postgres:x:46:46:PostgreSQL Server:/var/lib/pgsql:/dev/null
```

В целях безопасности у него может быть отключен вход в систему (/dev/null). Можно оставить эту настройку и выполнять все действия с помощью следующих конструкций:

"sudo su – postgres -c '<команда>'"

sudo -u postgres '<команда>'

или "включить" пользователя: "usermod -s /bin/bash postgres && sudo passwd postgres"

При настройке любого пользователя, администрирующего кластер, желательно задать каталог данных PGDATA и добавляется путь к исполняемым файлам, например:

```
su – postgres
echo "export PGDATA=/usr/local/pgsql/data" >> ~/.bash_profile
echo "export PATH=/usr/local/pgsql/bin:$PATH" >> ~/.bash_profile
```

Можно указать другие возможные переменные, например, PGDATABASE PGHOST и т.д. Если эти переменные заданы – их можно явно не указывать при выполнении команд, например, "psql " будет эквивалентно "/usr/local/pgsql/bin/psql -d postgres -U postgres -h localhost -p 5432".

Инициализация кластера (при необходимости, если кластер не создан при установке или создается вместо кластера по умолчанию) выполняется на сервере командой "initdb".

В некоторых дистрибутивах инициализация кластера предусмотрена специальным сценарием или командой (см. соответствующее руководство к ОС) – рекомендуется выполнить его, т.к. от параметров кластера могут зависеть другие компоненты системы. Например, в AltLinux кластер создается командой "/etc/init.d/postgresql initdb" от имени root.

При создании кластера:

Может потребоваться создать каталог для кластера и назначить владельца (дать права).

```
sudo mkdir /usr/local/pgsql/data
sudo chown postgres:postgres /usr/local/pgsql/data
chmod 0700 /usr/local/pgsql/data
```

Примеры инициализации кластера:

initdb -U postgres -W	Создание кластера в кодировке по умолчанию (обычно UTF-8 или C)
sudo su - postgres -c "/usr/lib/postgresql/9.6/bin/initdb --encoding='**UTF-8**' --lc-collate='**C**' --lc-ctype='ru_RU.CP1251' -D /trash/data"	Астра Линукс – создание отдельного кластера (ключ -D), с сортировкой 'C'

здесь "-U postgres" пользователь-администратор, "-W" задать пароль.

Если переменная PGDATA не задана, можно явно указать размещение с помощью ключа "-D".

Управление кластером

Для управления кластером в "ручном" режиме обычно используется команда "pg_ctl". Если задана переменная окружения "PGDATA", ключ "-D" можно не указывать.

pg_ctl start -D "/usr/local/pgsql/data" pg_ctl start	запуск кластера
pg_ctl stop -D "/usr/local/pgsql/data" pg_ctl stop	остановка кластера
pg_ctl stop -m i	принудительная остановка кластера – аналог "shutdown immediate" в Oracle
pg_ctl status	проверка состояния
pg_ctl restart	перезапуск кластера

Если в переменную окружения %PATH% не добавлен путь к pg_ctl, тогда необходимо указывать полный путь: **C:\PostgreSQL\9.6\bin**\pg_ctl.exe restart.

Для автоматического управления кластером обычно используется служба (сервис). В Windows он создается при установке.

В Linux – при установке из репозитария – сервис создается для настроек по умолчанию (см. /usr/lib/systemd/system/*postgrespro-9.6*.service), при сборке – необходимо создать unit вручную:

```
sudo nano /usr/lib/systemd/system/postgresql.service
[Unit]
Description=Postgres Pro Standard database server
Documentation=man:postgres(1)

[Service]
Type=notify
User=postgres
ExecStart=/usr/local/pgsql/bin/postgres -D /usr/local/pgsql/data
ExecReload=/bin/kill -HUP $MAINPID
KillMode=mixed
KillSignal=SIGINT
TimeoutSec=0

[Install]
WantedBy=multi-user.target
```

Основные команды управления сервисом в Linux (с системой инициализации systemd):

sudo systemctl enable postgrespro-9.6 sudo systemctl disable postgrespro-9.6	включить автозапуск при загрузке отключить автозапуск
sudo systemctl start postgrespro-9.6 sudo systemctl stop postgrespro-9.6 sudo systemctl status postgrespro-9.6	запуск кластера остановка кластера проверка состояния кластера

Настройка параметров кластера

Параметры кластера расположены в конфигурационном файле postgresql.conf (основной файл настроек, который расположен в корневом каталоге кластера, например "/usr/local/pgsql/data/ postgresql.conf").

Рекомендуемые параметры кластера для выделенного сервера PostgreSQL (конкретные значения указаны для примера):

shared_buffers = 1GB max_wal_size = 1GB	Общий буфер сервера. Обычно 25-40% RAM. **Внимание!** PostgreSQL использует двойную буферизацию (дублирование) данных – собственный кэш shared_buffers и дисковый кэш ОС. Если объем shared_buffers недостаточен для хранения часто используемых данных, то они будут постоянно записываться и читаться с диска (через кэш ОС). Чем больше значение shared_buffers, тем меньше остается для дискового кэша, что может отрицательно сказаться на общей производительности. Оценочный объем файлового кэша ОС задается параметром effective_cache_size. Увеличивать значение до 40% можно, если рабочий набор данных помещается в оперативную память. Максимальный размер журнала пред-записи (WAL). При увеличении shared_buffers требуется соответственно увеличить max_wal_size, чтобы растянуть процесс записи большого объёма данных.
work_mem = 64MB	Память для сортировки результата запроса (ORDER BY, DISTINCT, merge joins, hash joins). Обычно выделяется 2-4% доступной памяти (без других приложений и shared_buffers, при памяти 1-4 ГБ рекомендуется устанавливать 32-128 MB). Если объем памяти недостаточен для сортировки результата, будут использоваться временные файлы. Слишком большой – может привести к своппингу. **Важно!** Это лимит на один узел плана запроса, а не на одно соединение, один запрос может использовать несколько work_mem.
maintenance_work_mem = 256MB	Память для работы команды VACUUM, ANALYZE, CREATE INDEX, ALTER TABLE ADD FOREIGN KEY. Вычисляется как 50% размера самой большой таблицы (постоянно используемой) или индекса, можно увеличивать до 25% RAM. При памяти 1-4 ГБ рекомендуется устанавливать 128-512 MB. Должно быть больше work_mem. Можно на время увеличить для импорта или конвертации.
temp_buffers = 32MB	Максимальное количество памяти, выделяемой КАЖДОЙ сессии для работы с временными таблицами. Можно увеличивать до 5% RAM. **Внимание!** Выделенная память не освобождается, пока сессия не завершится. Критически важный параметр при использовании HDD-дисков (не SSD).
wal_buffers = 16MB	Размер буфера журнала пред-записи (WAL), в который записываются данные до записи на диск. Увеличение значения может повысить производительность при большом количестве подключений.
huge_pages = try	Для Linux – Включить использование огромных страниц памяти (https://habrahabr.ru/post/228793/). В Windows поддержка добавлена с версии 11.

effective_cache_size = 4GB	Примерный объем файлового кэша операционной системы. Обычно это 50% доступной памяти (т.е. памяти, не занятой операционной системой и приложениями). Оптимизатор (планировщик запросов) использует эту оценку для построения плана каждого запроса. Если значение слишком низкое, оптимизатор может принять решение не использовать некоторые индексы. Более высокое значение рекомендуется устанавливать на выделенной только для PostgreSQL машине.
random_page_cost = 4 seq_page_cost = 1	Стоимость чтения одной страницы с диска при произвольном и последовательном доступе. Отношение зависит от типов используемых дисков (HDD, SSD, Raid, RAM).
max_locks_per_transaction = 1024	Количество блокировок объектов для каждой транзакции. **Параметр необходим для процесса конвертации**. Вычисляется как "количество ядер на компьютере, где работает конвертер" * 64
from_collapse_limit = 24 join_collapse_limit = 24	По сути, управляет планировщиком при выполнении вложенных запросов
fsync=off *full_page_writes=off* *synchronous_commit=off*	*ТОЛЬКО **ДЛЯ ТЕСТОВЫХ И ДЕМО БД**. Оптимизация, угрожающая стабильности*
max_connections = 500	Максимальное количество соединений с БД. Необходимо учитывать, что одно соединение с ПП "ПАРУС-Бюджет 8" обычно “потребляет” 3 физических соединения (одно дополнительное соединение для автономных транзакций, другое – для динамического SQL). Кроме того, часть соединений используется служебными процессами самого PostgreSQL и резервируется для администрирования (настройка superuser_reserved_connections).
lc_messages = 'en_US.UTF-8'	Опция, только при кодировке кластера UTF-8 : отключить русский язык в сообщениях об ошибках и логах (из-за кодировки базы WIN1251 некорректно выводятся сообщения). В ОС Linux должна быть установлена локаль "en_US": sudo localedef -c -i en_US -f UTF-8 en_US.UTF-8
escape_string_warning = on standard_conforming_strings = on	По умолчанию, с версии PostgreSQL 9.1, значение "on". Некоторые дистрибутивы, например, Астра Линукс, могут иметь значение "off". Необходимо проверить.
tcp_keepalives_idle = 600 tcp_keepalives_interval = 6 tcp_keepalives_count = 10	Только для Linux **при работе с ЦУД**-ом ("тонкий клиент"). Настройки устраняют проблемы с "зависанием" клиента в случае обрыва соединения между ЦУД-ом и СУБД и последующей попыткой повторного соединения.

При выполнении трудоемких однопользовательских операций (обновление, загрузка управляемых разделов, импорт и т.п.) можно **на время** изменить параметры в сторону увеличения, например, сделать work_mem = 256MB, temp_buffers = 128MB и maintenance_work_mem = 1GB, отключить fsync (если есть резервная копия БД).

Примеры просмотра текущих значений:

```
select current_setting('shared_buffers');
show shared_buffers;
select * from pg_settings order by name;
select * from pg_settings where name = 'shared_buffers';
select * from pg_settings where name in ('fsync', 'full_page_writes', 'synchronous_commit');
```

Основной процесс перечитывает файл конфигурации заново, получая сигнал SIGHUP, поэтому для вступления параметров в силу необходимо выполнить одно из действий:

- Перезапустить кластер.
- Запустить pg_ctl reload в командной строке.
- Вызвать SQL-функцию pg_reload_conf().

Примечание. В основном файле конфигурации postgresql.conf задаются значения, с которыми кластер стартует. Если параметры дублируются, то берется последнее значение (например, можно для удобства, дописать измененные значения в конец файла). После старта кластера значения параметров могут переопределяться значениями из файла postgresql.auto.conf. Этот файл не редактируется вручную, значения добавляются туда при выполнении команды "ALTER SYSTEM".

Настройка доступа

Важно!

Возможно, что после установки сервера БД PostgreSQL будет настроен только локальный доступ (на сервере) супер-пользователя, указанного при установке (обычно **postgres**).

Если пароль супер-пользователя не задавался при установке (initdb -W), то его нужно задать:

psql -h localhost -U postgres -c "ALTER ROLE postgres WITH PASSWORD 'password';"

Для того, чтобы другие пользователи могли работать с БД (в том числе по сети), необходимо выполнить соответствующую настройку в конфигурационных файлах (обычно расположены в корне каталога с данными кластера базы данных).

1. В конфигурационном файле postgresql.conf (основной файл настроек) изменить настройку **listen_addresses**, которая определяет на каких сетевых интерфейсах разрешать подключения (аналог настройки Oracle "LISTENER\ADDRESS" в файле listener.ora), например:

listen_addresses = ''	Пустой список – только доменные сокеты Unix (только Linux)
listen_addresses = '*'	Все доступные интерфейсы
listen_addresses = '0.0.0.0'	Все адреса IPv4 на всех интерфейсах
listen_addresses = 'localhost'	127.0.0.1 (для IPv4) и ::1 (для IPv6)
listen_addresses = '172.28.50.45, 192.168.50.2'	конкретные сетевые адреса (или имена)

2. В конфигурационный файл pg_hba.conf (файл аутентификации) добавить записи в соответствии с требуемыми типами подключений.

Для работы ПП "ПАРУС-Бюджет 8" нужно задать подключения по TCP/IPv4 с методом аутентификации по паролю (md5 или `scram-sha-256` для версий 12 и выше).

```
host    all     all     127.0.0.1/32    md5
host    all     all     0.0.0.0/0       md5
#host   all     all     192.168.1.0/24  md5
#host   all     all     172.28.0.0/16   md5
```

Методы trust и peer не поддерживаются, другие – пока не рассматриваются.

Обязательно требуется указать локальное подключение (127.0.0.1) с паролем – оно используется на сервере БД для поддержки автономных транзакций.

Общая информация по настройке доступа представлена в главе "Дополнения".

Роли

Для управления доступом к БД и разрешениями на объекты в ней используется концепция ролей. В зависимости от того, как роль настроена, ее можно рассматривать как пользователя или как группу. Любая роль может использоваться в качестве пользователя, группы, или того и другого.

Для создания роли используется команда "CREATE ROLE имя;", для удаления – "DROP ROLE имя;".

Примечания:

- В качестве имени роли желательно использовать идентификаторы, соответствующие правилам именования SQL – без специальных символов и несовпадающее с зарезервированными словами. В противном случае, необходимо указывать имя в двойных кавычках, но т.к. в стандарте разработки ПП "ПАРУС-Бюджет 8" по умолчанию не принято их использование, это может привести к ошибкам при работе.
- Создание/удаление роли можно выполнять непосредственно из ОС отдельными программами createuser и dropuser, которые входят в поставку PostgreSQL.

При создании роли в качестве параметров указываются ее атрибуты. Пользователи ПП "ПАРУС-Бюджет 8" для работы через WIN-клиент (включен флаг "Сеанс базы данных" в разделе "Пользователи") должны иметь атрибут LOGIN (разрешается вход на сервер) и PASSWORD (пароль необходимо указывать, т.к. действует аутентификация по паролю):

```
CREATE ROLE user1 LOGIN PASSWORD 'password';
```

или другой вариант (считается устаревшим):

```
CREATE USER user1 PASSWORD 'password';
```

(т.е., по сути, пользователь – это роль с правом входа)

Ограничения:

При создании пользователя необходимо учесть следующие особенности при работе с БД PostgreSQL:

1. По умолчанию имена пользователей соответствуют правилам именования SQL и прописываются строчными (маленькими) буквами, т.е. следующие команды создают одного и того же пользователя user1:

   CREATE ROLE user1 LOGIN PASSWORD 'password';

   CREATE ROLE **U**ser1 LOGIN PASSWORD 'password';

   CREATE ROLE **USER**1 LOGIN PASSWORD 'password';

2. Если требуется создать пользователя с именем, несоответствующим правилам SQL, необходимо использовать идентификаторы в кавычках (отделенные идентификаторы), например:

   CREATE ROLE "**U**ser1" LOGIN PASSWORD 'password';

   CREATE ROLE "**U**ser 1" LOGIN PASSWORD 'password';

   CREATE ROLE "**Гость**1" LOGIN PASSWORD 'password';

   Обратите внимание на то, что какое-нибудь приложение может работать некорректно.

3. Для работы с Windows-приложениями ПП "ПАРУС-Бюджет 8" (WIN-клиент и ЦУД-клиент) возможно создание пользователя с именем, состоящим из кириллицы, если все буквы прописные (большие), например:

   CREATE ROLE "**ГОСТЬ**1" LOGIN PASSWORD 'password';

Для веб-клиента эти ограничения не актуальны – имена пользователей задаются не в БД, а в служебной таблице USERLIST.

Для изменения атрибутов роли используется команда "ALTER ROLE", например:

ALTER ROLE user1 PASSWORD 'password1';	смена пароля
ALTER ROLE user1 NOLOGIN;	закрыть доступ на сервер
ALTER ROLE user1 IN DATABASE demo1 SET search_path TO parus, public;	установить пользователю user1 в БД demo1 переменную search_path

Для определения прав доступа на объекты в БД используется команда GRANT (для отзыва прав – REVOKE), например:

GRANT SELECT ON mytable TO PUBLIC;

GRANT ALL ON TABLESPACE parus_main TO parus;

Установка расширений

Расширения PostgreSQL предназначены для расширения функционала СУБД. В каждой базе по умолчанию обязательно есть только расширение plpgsql.

Обычно, расширения реализуются в виде нескольких файлов:

- SQL-скрипт, в котором объединены SQL-объекты (типы данных, функции и т.д.). Размещается в папке /share/extension/
- Динамически загружаемая библиотека, если она необходима (/lib/).
- Управляющий файл, в котором указывается имя скрипта, путь к библиотеке, версия и т.д. (/share/extension/).

Каждый дистрибутив PostgreSQL может включать свой набор расширений (помимо "базовых", включенных в PostgreSQL Core).

Функционал расширения становится доступен в БД только после его "**регистрации**" в БД (команда "CREATE EXTENSION").

> ***Примечание.*** Если добавить расширение (например, pg_variables) в шаблон template1, то оно будет автоматически регистрироваться при создании каждой новой БД:
>
> psql -U postgres -d template1 -c "CREATE EXTENSION pg_variables;"

Для работы ПП "ПАРУС-Бюджет 8" требуются следующие расширения (на **11.03.2024**, состав расширений и их версии могут меняться):

Расширение	Описание	Установка
dblink	Подключение к другим БД из сеанса текущей БД	PostgreSQL (входит в состав любого дистрибутива)
uuid-ossp	Генератор идентификаторов	PostgreSQL
xml2	Выполнение XPath-запросов и XSLT-преобразований	PostgreSQL
hstore	Хранения пар ключ/значение	PostgreSQL
pg_variables	Работа с переменными в текущей пользовательской сессии	Требуется версия 1.1 и выше Имеется в Postgres Pro, для СУБД других производителей необходимо установить расширение вручную
http	http-клиент	Устанавливается вручную Linux Windows
pgqrcode pgzip	Работа с QR-кодами и выгрузками архивов	Собственная разработка, поставляется вместе с конвертером Доступно **с релиза от 22.09.2021** Заменяет функционал, использующий Java (pljava)

Расширения могут регистрировать автоматически при выполнении каких-либо действий, например, импорт или конвертация. Во избежание проблем рекомендуется проверить регистрацию расширений на тестовой базе, которую затем удалить.

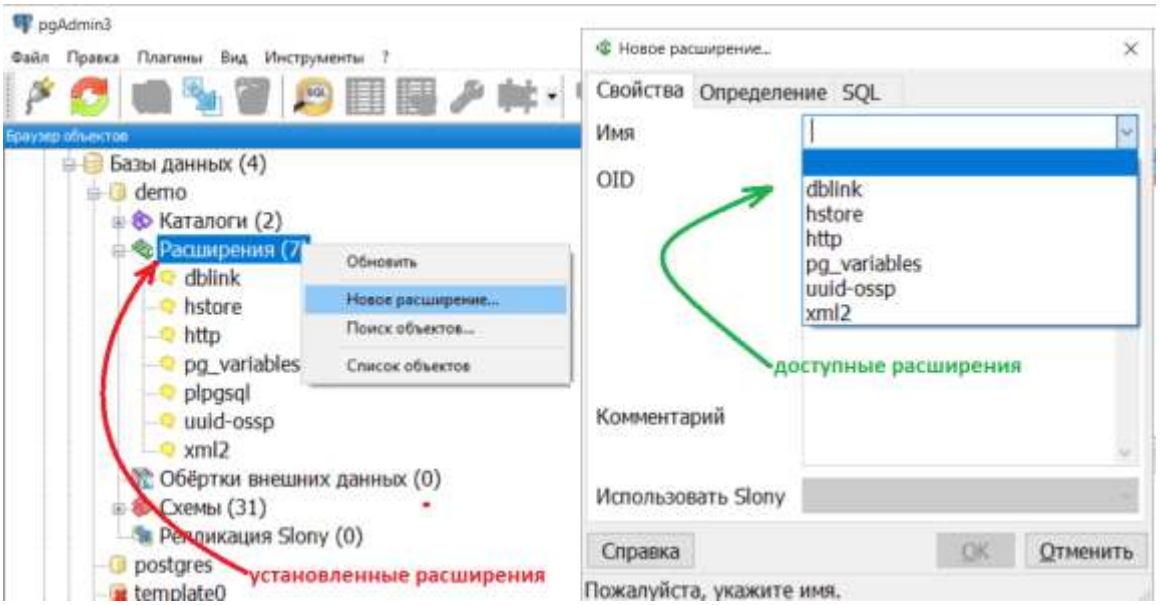
> Для обновления расширения необходимо скопировать соответствующие файлы поверх имеющихся и выполнить команду "ALTER EXTENSION <расширение> UPDATE TO '<версия>'; ". Например, в PostgresPro 9.6.8 доступна версия расширения pg_variables 1.0. Для обновления до версии 1.2 необходимо взять соответствующие файлы из версии PostgresPro 9.6.11 (или собрать), переписать старые файлы расширения новыми, и выполнить:
>
> psql -d ppc1 -c "ALTER EXTENSION pg_variables UPDATE TO '1.2';"

Получить список доступных расширений:

psql -U postgres -d pg0626 -c "select * from **pg_available_extensions;"**

Если installed_version <> null, то расширение установлено.

Для наглядности ниже показано управление расширениями с помощью утилиты администрирования pgAdmin:



Примечание. Сборка расширений под Windows – задача нетривиальная – зависит от версии Windows, версии и издателя PostgreSQL. Вариантов – множество. Под каждый вариант требуется развернуть среду разработки (или, как минимум, компилятор + зависимости), и собрать можно только конкретное расширение под ту же конфигурацию сервера БД (Windows + PostgreSQL). Рекомендуется использовать менее затратные варианты установки – подходящие сборки PostgreSQL, или установка на Linux.

pg_variables

Требуется версия 1.1 и выше.

Если установлена СУБД Postgres Pro версии 9.6.11 и выше, дополнительных действий не требуется, т.к. расширение pg_variables включено в поставку.

Для СУБД других производителей необходимо установить расширение дополнительно. См. https://github.com/postgrespro/pg_variables.

Установка на **Linux**:

При необходимости доставить пакет postgresql-**devel (**разработка расширений сервера):

```
dnf install postgresql-devel
```

Скачать, распаковать, собрать, установить:

```
su – postgres
wget https://github.com/postgrespro/pg_variables/archive/master.zip
unzip ~/master.zip
cd pg_variables-master
make USE_PGXS=1
sudo make USE_PGXS=1 install
```

Проверка (опция, в зависимости от версии, не все тесты могут проходить корректно):

```
make USE_PGXS=1 installcheck
```

При проверке будет создана БД, которую можно удалить:

```
psql -U postgres -c " DROP DATABASE contrib_regression;"
```

После сборки и установки рекомендуется не удалять каталог сборки для того, чтобы впоследствии осталась возможность удалить ПО не вручную, а автоматически с помощью команды "make unistall".

В **Windows** самый простой способ – установить где-либо Postgres Pro (обязательно той же разрядности и желательно той же версии) и скопировать файлы расширения в соответствующие папки.

Примечание. PostgrePro прекратил выпуск релизов (Standard и Enterprise) для Windows начиная с 15-ой версии. Остался "обычный" PostgreSQL для Windows (без pg_variables).

http

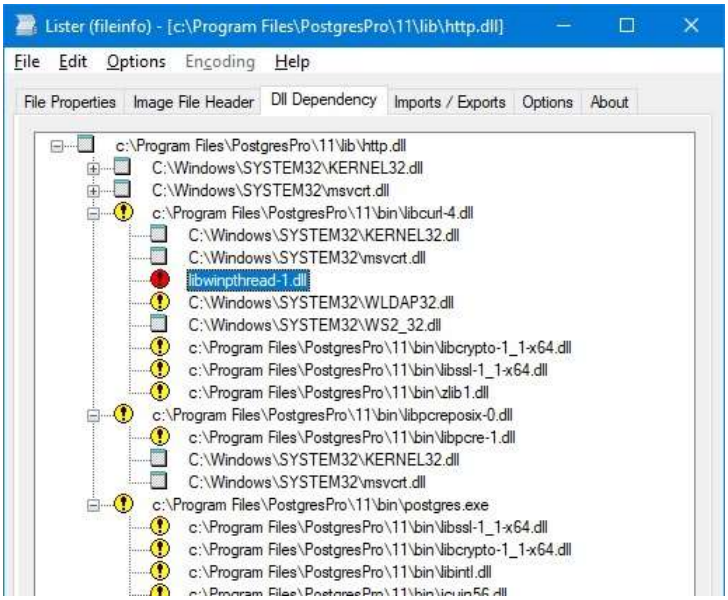
Расширение предназначено для взаимодействия с веб-сервисами/сайтами по протоколу http непосредственно из базы данных. Расширение является "оберткой" над утилитой curl (взаимодействует через библиотеку API libcurl).

Расширение устанавливается самостоятельно, на сегодняшний день не входит в состав дистрибутивов PostgreSQL.

Для **Windows** – скачать архив для соответствующей версии PostgreSQL и распаковать в соответствующие папки. В архив уже включена библиотека libcurl для Winsows.

> ***Примечание.***
>
> Если при регистрации расширения (create extension) возникает ошибка "*загрузить библиотеку http.dll не удалось: The specified module could not be found*", то это значит, что не установлены все зависимости для расширения.
>
> Например, для сборок PostgreSQL от EnterpriseDB и BigSQL потребуется установка OpenSSL для Windows (требуются библиотеки libeay32.dll, ssleay32.dll, libssl.dll и др.; при установке выбрать размещение библиотек в системный каталог). Для PostgreSQL версии <15 требуется OpenSSL 1.1, для версии >=15 - OpenSSL 3.
>
> В других сборках, например Postgres Pro, библиотеки OpenSSL присутствуют, но нет других библиотек.
>
> **Пример**. Установка библиотеки libwinpthread-1.dll (вариант).
>
> В файле README.txt из zip-архива с расширением определить компилятор, которым собрано расширение, например, "Compiled using mingw64-w64 GCC 8.1 chain (x86_64-win32-seh-rev1, Built by MinGW-W64 project)". Скачать соответствующий компилятор со страницы проекта MinGW-W64, в данном случае выбрать "MinGW-W64 GCC-8.1.0" ссылка на "x86_64-win32-seh". Из скачанного 7z-архива извлечь файл "\mingw64\bin\libwinpthread-1.dll" и скопировать его в каталог рядом с библиотекой libcurl-4.dll

Определить отсутствующие библиотеки можно, например, с помощью плагина Fileinfo для Total Commander (выбрать файл расширения http.dll – нажать просмотр F3 – вкладка "Dll dependency")



Для **Linux** – собирается вручную из исходников (для каждой ОС требуется собственная сборка, т. к. зависит от библиотеки libcurl, которая в каждом дистрибутиве своя).

Перед сборкой необходимо убедиться, что установлены пакеты разработчика (dev или devel) для сервера PostgreSQL и libcurl (должны запускаться утилиты pg_config и curl-config).

```
pg_config   -- проверка установки пакета postgresql-server-dev (имя пакета может отличаться)
curl-config   -- проверка установки пакета libcurl-dev (имя пакета может отличаться)
wget https://github.com/pramsey/pgsql-http/archive/master.zip
unzip master.zip && cd pgsql-http-master
make
sudo make install
```

После установки необходимо зарегистрировать расширение в БД:

```
psql -d ppc1 -c "CREATE EXTENSION http;"
```

Проверка работы расширения:

```
SELECT urlencode('my special string''s & things?');
SELECT content FROM http_get('http://httpbin.org/ip');
```

Проверка работы расширения:

```
SELECT urlencode('my special string''s & things?');
SELECT content FROM http_get('http://httpbin.org/ip');
```

PGExtensions (pgzip, pgqrcode)

Собственная разработка для работы с QR-кодами и выгрузками zip-архивов.

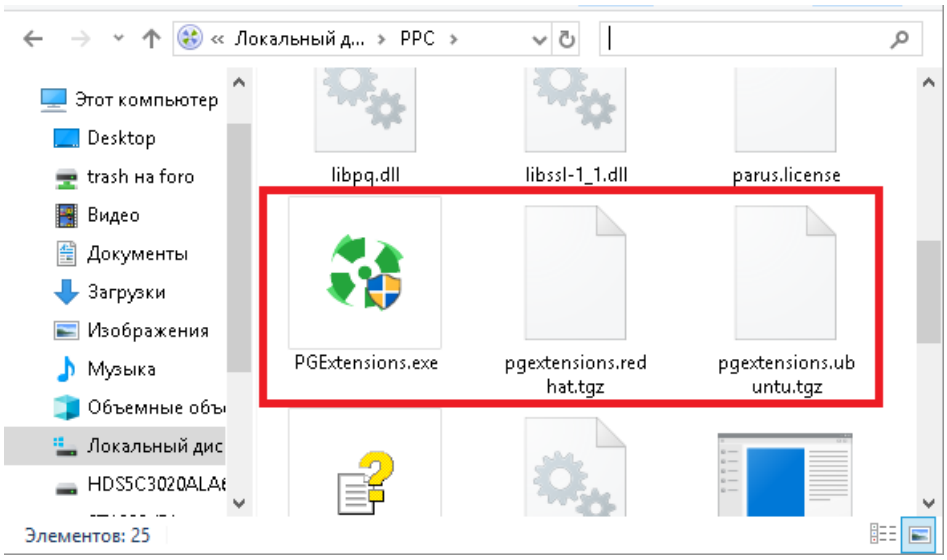
Доступно **с релиза от 22.09.2021**.

Заменяет функционал, использующий Java.

Поставляется вместе с Конвертером "Oracle-PostgreSQL" (входит в состав инсталлятора ppc.exe). После установки – размещен в папке с приложением.

Состоит из 3-х файлов, предназначенных для установки на разные ОС:

- PGExtensions.exe – Windows (32-х и 64-х разрядные версии).
- pgextensions.redhat.tgz – Linux RedHat-подобные ОС.
- pgextensions.ubuntu.tgz – Debian/Ubuntu/Астра.



Примечание. Можно извлечь требуемый файл без установки, с помощью архиватора 7-Zip:
WIN_CMD>"c:\Program Files\7-Zip\7z.exe" e x:\ppc.exe PGExtensions.exe

LINUX>7z e ~/ppc.exe pgextensions.redhat.tgz

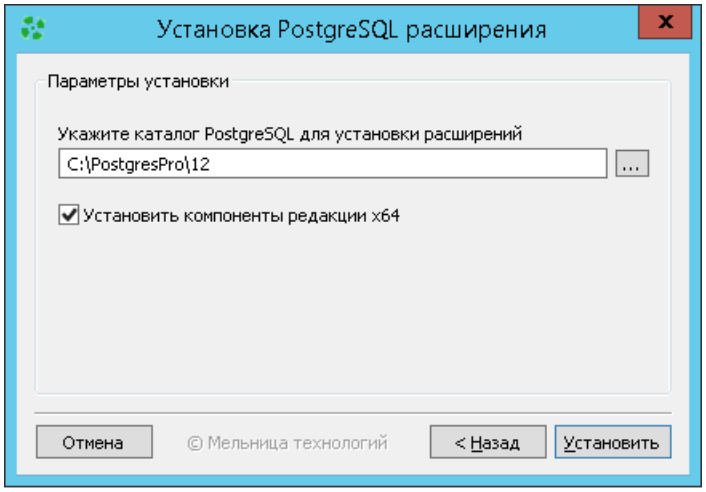
В PostgreSQL 13 добавлена возможность установки расширений непривилегированным пользователем. Для это в управляющем файле *.control добавлен параметр trusted. Этот параметр добавлен по умолчанию и для расширений pgzip и pgqrcode. Т.к. используется универсальный установщик для всех версий PostgreSQL, то для версий PostgreSQL < 13 необходимо удалить строку "**trusted=true**" из control-файлов (или после распаковки архива с файлами расширений или после установки непосредственно на месте, главное – перед выполнением "CREATE EXTENSION").

Пример правки (для Linux):

```
sed -i '/trusted = true/d' pgqrcode.control
sed -i '/trusted = true/d' pgzip.control
```

Установка на **Windows**

Запустить установщик PGExtensions.exe, подтвердить путь для установки (по умолчанию выбирается каталог установки сервера PostgreSQL), при необходимости изменить:



Установка на **Linux**

Установить пакет `libpng-dev`, если не установлен:

sudo apt install libpng-dev # Debian / Ubuntu / Астра
sudo yum install libpng-devel # RedHat

Выбрать подходящий для ОС архив (pgextensions.**redhat**.tgz или pgextensions.**ubuntu**.tgz), распаковать и запустить сценарий установки install.sh:

```
tar -xvzf pgextensions.redhat.tgz
sudo ./install.sh
```

Установщик попытается определить пути установки.

Если этого не произошло – необходимо задать значения вручную. Для определения значений можно воспользоваться утилитой pg_config с соответствующими ключами:

```
pg_config --pkglibdir
#pg_config --sharedir # потребуется добавить «/extension»
echo -n $(pg_config --sharedir) ; echo "/extension"
```

```
[parus@x8dtif ~]$ tar -xvzf /tmp/pgextensions.redhat.tgz
install.sh
pgqrcode--1.0.sql
pgqrcode.control
PGQRCode.so
pgzip--1.0.sql
pgzip.control
PGzip.so
[parus@x8dtif ~]$ pg_config --pkglibdir
/usr/pgpro-9.6/lib
[parus@x8dtif ~]$ pg_config --sharedir    + /extension
/usr/pgpro-9.6/share
[parus@x8dtif ~]$ sudo ./install.sh
[sudo] пароль для parus:
find: '/usr/pgsql-*': Нет такого файла или каталога
find: '/usr/pgsql-*': Нет такого файла или каталога
Specify the PostgreSQL directories where to install extensions:
  lib: /usr/pgpro-9.6/lib
  extension: /usr/pgpro-9.6/share/extension

Installing extensions pack...

«/home/parus/PGQRCode.so» -> «/usr/pgpro-9.6/lib/PGQRCode.so»
«/home/parus/pgqrcode.control» -> «/usr/pgpro-9.6/share/extension/pgqrcode.control»
«/home/parus/pgqrcode--1.0.sql» -> «/usr/pgpro-9.6/share/extension/pgqrcode--1.0.sql»
«/home/parus/PGzip.so» -> «/usr/pgpro-9.6/lib/PGzip.so»
«/home/parus/pgzip.control» -> «/usr/pgpro-9.6/share/extension/pgzip.control»
«/home/parus/pgzip--1.0.sql» -> «/usr/pgpro-9.6/share/extension/pgzip--1.0.sql»

Installation successful!
```

Примечания:

- Утилита pg_config обычно входит в состав пакета разработки (postgresql-dev) или клиентской (libpq-dev) СУБД и должна быть доступна пользователю, который ее выполняет.
- Найти каталог установки файлов расширений можно поиском аналогичных файлов для других расширений, например:

  `sudo find / -name pg_variables.so` # библиотеки
  `sudo find / -name pg_variables.control` # расширения
- Вместо выполнения сценария установки install.sh можно вручную скопировать файлы расширения в соответствующие каталоги, например:

  ```
  cp pgqrcode.control pgqrcode--1.0.sql /usr/pgpro-9.6/share/extension/
  cp PGQRCode.so /usr/pgpro-9.6/lib/
  ```

Проверка доступности расширений:

```
psql -U postgres -d demo
select * from pg_available_extensions where name in ('pgqrcode', 'pgzip');
```

Проверка работы расширений pgzip и pgqrcode:

```
psql -U postgres
create database test1;
\c test1

create extension if not exists pgzip;
select pgzip_crc('\xABCDEF');

create extension if not exists pgqrcode;
select pgqrcode_get_bitmap('Привет', 4, 0, 160);

\c postgres
drop database test1;
```

Создание БД

После инициализации кластера в нем уже будут созданы 3 базы данных:

- **postgres** – база данных суперпользователя (по умолчанию postgres). Для сторонних программ это база может выступать как служебная, поэтому не рекомендуется вносить в нее изменения;
- **template1** – шаблон (база данных), используемый по умолчанию при создании новых баз. В эту базу можно вносить изменения, которые должны присутствовать во ВСЕХ производных базах по умолчанию, например, устанавливать расширения или создавать какие-либо объекты;
- **template0** – исходный шаблон (база) – используется при создании баз с параметрами, отличными от параметров по умолчанию, заданных при инициализации кластера, например, локали. Базу НЕЛЬЗЯ изменять.

Новые базы данных создаются командой "CREATE DATABASE" после подключения к серверу, например, штатной утилитой "psql":

CMD>	psql -U postgres -h localhost -d postgres	Вход пользователем postgres (если БД не указана, вход выполняется в базу с именем пользователя)
PSQL#	CREATE DATABASE ***ppc1*** WITH OWNER = ***postgres*** TEMPLATE = ***template1*** ENCODING = 'WIN1251' LC_COLLATE = **'Russian_Russia.1251'** LC_CTYPE = **'Russian_Russia.1251'** TABLESPACE = **pg_default** CONNECTION LIMIT = **-1** ;	- создать БД с именем “ppc1” - владелец БД – по умолчанию – пользователь, выполняющий команду (должен иметь привилегию CREATEDB или SUPERUSER) - использовать шаблон (базы создаются копированием шаблона), по умолчанию "template1" - кодировка символов. Использовать WIN1251 - порядок сортировки текста. Использовать обязательно Russian_Russia.1251 (для Windows) или ru_RU.cp1251 или C (для Linux, см. Примечание ниже) - классификация символов. Использовать обязательно Russian_Russia.1251 (для Windows) или ru_RU.cp1251 (для linux) - табличное пространство – расположение в файловой системе. По умолчанию "pg_dafault" – "совпадает" с кластером - максимальное количество подключений к БД (по умолчанию "-1" – без ограничений, но необходимо учитывать настройку max_connections для кластера)

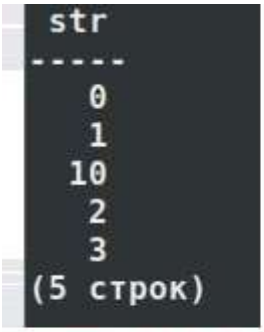
Примечания:

- Для ОС Linux существует несколько факторов, влияющих на то, какое правило сортировки LC_COLLATE выбрать – "ru_RU.cp1251" или "C" (версия СУБД PostgreSQL и параметры ее сборки, наличие и версия библиотеки ICU в ОС). Рекомендуется проверить правильность сортировки на тестовой базе данных в кодировке win1251, до создания базы для ПП "ПАРУС-Бюджет 8":

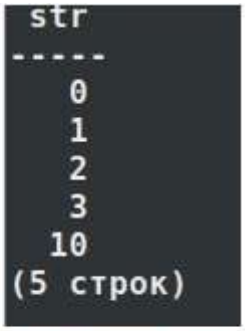
```
# Создать тестовую базу в кодировке 'WIN1251' с правилом сортировки 'ru_RU.cp1251'
psql -h localhost -U postgres -c "CREATE DATABASE test TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE =
'ru_RU.cp1251' LC_CTYPE = 'ru_RU.CP1251';"

#Подключиться к тестовой базе
psql -h localhost -U postgres -d test

#Выполнить запрос
```

```
select T.* from
(
select '  0' as STR
union all
select '  1' as STR
union all
select '  2' as STR
union all
select '  3' as STR
union all
select ' 10' as STR
) T
order by T.STR collate "ru_RU.cp1251";
```

```
select T.* from
(
select '  0' as STR
union all
select '  1' as STR
union all
select '  2' as STR
union all
select '  3' as STR
union all
select ' 10' as STR
) T
order by T.STR collate "C";
```



"Неправильная" сортировка



"Правильная" сортировка

```
# Удалить тестовую базу
\c postgres
drop database test;
\q
```

Если запрос с сортировкой `"ru_RU.cp1251"` выполняется с "правильной" сортировкой – базу для ПП "ПАРУС-Бюджет 8" создавать с LC_COLLATE = 'ru_RU.cp1251', если нет – LC_COLLATE = 'C'.

- Если выполнены все предварительные условия (пользователь ОС – это администратор БД, не менялись табличные пр-ва, локаль соответствует настройке ОС), то можно указать только имя БД при ее создании:

CREATE DATABASE ***ppc1*;**

Одну команду проще выполнять, передав ее psql с помощью значения ключа -c:

psql -h localhost -U postgres -c "CREATE DATABASE ***ppc1***;"

Если кластер создан в кодировке WIN1251, достаточно выполнить:

CREATE DATABASE ***ppc1;***

Будет использован шаблон "template1" с нужными языковыми параметрами.

Если кластер создан в кодировке UTF8, необходимо использовать шаблон "template0" и переопределить языковые параметры, например:

CREATE DATABASE ***ppc1*** TEMPLATE **template0** ENCODING = 'WIN1251' LC_COLLATE = '**Russian_Russia.1251**' LC_CTYPE = '**Russian_Russia.1251**';	Windows
CREATE DATABASE ***ppc1*** TEMPLATE **template0** ENCODING = 'WIN1251' LC_COLLATE = '**C**' LC_CTYPE = 'ru_RU.CP1251';	Linux LC_COLLATE зависит от СУБД
CREATE DATABASE ***ppc1*** TEMPLATE **template0** ENCODING = 'WIN1251' LC_COLLATE = '**ru_RU.CP1251**' LC_CTYPE = 'ru_RU.CP1251';	

См. также Удаление БД.

Проверка:>	psql	Логин пользователем postgres (если БД не указана, вход в базу выполняется с именем пользователя, адрес сервера – localhost, порт – 5432)
#	\l	Получить список баз в кластере
#	\c pg0626	Подключиться к БД *pg0626*
#	\dx	Получить список зарегистрированных расширений в БД
#	\q	Выход

Кластер и базы в кодировке WIN1251, установленные расширения:

```
[postgres@arc1 ~]$ psql
psql (9.6.11)
postgres=# \l
                                   Список баз данных
    Имя    | Владелец | Кодировка |  LC_COLLATE   |   LC_CTYPE    |     Права доступа
-----------+----------+-----------+---------------+---------------+-----------------------
 demo      | postgres | WIN1251   | ru_RU.CP1251  | ru_RU.CP1251  |
 postgres  | postgres | WIN1251   | ru_RU.CP1251  | ru_RU.CP1251  |
 template0 | postgres | WIN1251   | ru_RU.CP1251  | ru_RU.CP1251  | =c/postgres
           |          |           |               |               | postgres=CTc/postgres
 template1 | postgres | WIN1251   | ru_RU.CP1251  | ru_RU.CP1251  | =c/postgres
           |          |           |               |               | postgres=CTc/postgres
(4 строки)

postgres=# \c demo
Вы подключены к базе данных "demo" как пользователь "postgres".
demo=# \dx
                              Список установленных расширений
     Имя      | Версия |   Схема    |                                  Описание
--------------+--------+------------+-------------------------------------------------
 dblink       | 1.2    | public     | connect to other PostgreSQL databases from within
 hstore       | 1.4    | public     | data type for storing sets of (key, value) pairs
 http         | 1.3    | public     | HTTP client for PostgreSQL, allows web page retri
 pg_variables | 1.1    | public     | session variables with various types
 plpgsql      | 1.0    | pg_catalog | PL/pgSQL procedural language
 uuid-ossp    | 1.1    | public     | generate universally unique identifiers (UUIDs)
 xml2         | 1.1    | public     | XPath querying and XSLT
```

Кластер в кодировке UTF8, пользовательские БД – WIN1251:

```
[postgres@arch ~]$ psql
psql (9.6.11)
postgres=# \l
                                  Список баз данных
    Имя     | Владелец | Кодировка |  LC_COLLATE   |   LC_CTYPE    |      Права доступа
------------+----------+-----------+---------------+---------------+-----------------------
 demo       | postgres | WIN1251   | ru_RU.CP1251  | ru_RU.CP1251  |
 postgres   | postgres | UTF8      | ru_RU.UTF-8   | ru_RU.UTF-8   |
 ppc1       | postgres | WIN1251   | ru_RU.CP1251  | ru_RU.CP1251  |
 template0  | postgres | UTF8      | ru_RU.UTF-8   | ru_RU.UTF-8   | =c/postgres          +
            |          |           |               |               | postgres=CTc/postgres
 template1  | postgres | UTF8      | ru_RU.UTF-8   | ru_RU.UTF-8   | =c/postgres          +
            |          |           |               |               | postgres=CTc/postgres
```

Примечание. Для консоли Windows – установите в свойствах консоли равно-ширинный шрифт "Lucida Console" и измените кодовую страницу командой "chcp 1251" (или chcp 65001 для Windows 10), чтобы корректно отображались сообщения на русском языке.

Схемы и search_path

Схемы предназначены для **логического** размещения именованных объектов в базе данных.

При обращении к любому объекту, размещенному в БД, необходимо учитывать, в какой схеме он находится – можно явно указать имя схемы (<имя схемы>.<объект>) или добавить имя схемы в параметр search_path (<объект> – имя схемы не указывается).

Параметр "**search_path**":

- Перечисляет схемы, в которых выполняется поиск.
- Задает порядок поиска по схемам.

  (**Важно!** Схемы просматриваются в указанном порядке.)

Значение параметра "search_path" может задаваться на разных уровнях:

- Кластер – параметр "search_path" в конфигурационном файле **postgresql.conf**.
- База данных – "ALTER **DATABASE** dbname SET search_path=’schema1, schema2’;".
- Пользователь – "ALTER **ROLE** user1 SET search_path=’schema1, schema2’;" или "ALTER **ROLE** user1 **IN DATABASE** dbname SET search_path=’schema1, schema2’;".
- Текущая сессия – "**SET** search_path=’schema1, schema2’;".
- Конкретная функция – "ALTER **FUNCTION** func1() SET search_path=’schema1, schema2’;".

Значение более конкретного уровня перекрывает значения старшего.

Значение по умолчанию – "$user", public (т.е. ‘схема с именем текущего пользователя’, ‘схема по умолчанию’) – задается в postgresql.conf.

Во время сеанса пользователя его текущей схемой считается первая (из существующих) в пути поиска search_path (select current_schema();). Если при создании объектов (команда CREATE) не указывается схема, то считается, что объект создается в текущей схеме.

По умолчанию база содержит следующие схемы:

- **information_schema** – Информационная схема – содержит информацию об объектах в базе данных. Обычно используется для получения служебной информации. Схема отсутствует в пути поиска схем, поэтому ко всем объектам в ней нужно обращаться по полным именам.
- **pg_catalog** – Системный каталог – содержит *встроенные* типы, функции, операторы, таблицы самого PostgreSQL. Без указания в "search_path" схема неявно просматривается **ДО** всех схем, т.е. подразумевается "pg_catalog, schema1, schema2". Если схема указана в пути, она просматривается в заданном порядке.
- **public** – Схема по умолчанию – используется если отсутствует схема с именем текущего пользователя (это поведение по умолчанию, при значении search_path=‘"$user", public’). Все пользователи имеют права CREATE и USAGE в этой схеме. В PostgreSQL 15 и выше создавать (CREATE) объекты в схеме обычные пользователи не могут, только владелец базы или суперпользователь.
- **pg_temp** – Схема временных таблиц текущего сеанса (точнее, pg_temp это псевдоним временной схемы для текущего сеанса, полное имя состоит из префикса pg_temp и окончания в виде PID-а серверного процесса, например, pg_temp_33). Схема, если она отсутствует в search_path, будет просматриваться первой (даже перед pg_catalog). Просматривается только при поиске отношений (таблиц, представлений, последовательностей и т. д.) и типов данных, но никогда при поиске функций и операторов.
- **pg_toast** – Служебная схема для TOAST-таблиц, которая используется для хранения "больших" значений атрибутов типа TEXT, JSON и т.п. Если это временная таблица, то используется схема pg_toast_temp. Механизм работы TOAST незаметен для пользователя.

Остальные схемы создаются пользователями (личные схемы) для решения задач какого-либо приложения.

Для работы ПП "ПАРУС-Бюджет 8" потребуются создать следующие схемы (**с релиза от 01.2024**):

- Системная схема "**sys**" для "системных" объектов (в том числе объекты конвертера и объекты, эмулирующие работу с Oracle).
- Прикладная схема для объектов ПП "ПАРУС-Бюджет 8", имя может быть произвольное, обычно "**parus**" (аналог схемы parus в Oracle).

Использование табличных пространств

Табличные пространства предназначены для организации **физического** размещения файлов базы данных в файловой системе.

Конвертер/инсталлятор ПП "ПАРУС-Бюджет 8" поддерживает размещение данных по разным табличным пространствам.

При конвертации – если на сервере есть табличное пространство с именем, совпадающим с именем табличного пространства Oracle (регистр не важен), то таблица или индекс будет создан в этом табличном пространстве.

При создании БД с помощью инсталлятора – пространства выбираются на соответствующей вкладке.

По умолчанию (если не указано явно), объект создается в пространстве по умолчанию pg_default. Пространство по умолчанию может быть переопределено параметром default_tablespace.

Пример:

```
mkdir -p /mnt/pg2/data2
chown postgres:postgres /mnt/pg2/data2
chmod -R 700 /mnt/pg2/data2
-- Windows: icacls X:\PostgreSQL\data /grant "NETWORK SERVICE":(OI)(CI)F /T
psql -d ppc1
CREATE TABLESPACE parus_main OWNER parus LOCATION '/mnt/pg2/data2';
```

Узнать имеющиеся табличные пространства можно запросами:

```
SELECT tablespace FROM pg_tables WHERE tablename = 'agnlist' AND schemaname = 'parus';
SELECT tablespace FROM pg_indexes WHERE indexname = 'i_agnlist_prnation_fk' AND schemaname = 'parus';

SELECT tablespace FROM pg_tables WHERE tablename = 'appmodules_ref$tmp' AND schemaname = 'parus';
SELECT * FROM pg_indexes where tablename like '%$tmp' order by indexname;
```

Внимание! В PostgreSQL для больших объектов (LOB) всегда используется табличное пространство по умолчанию (т. е. смысла в пространстве PARUS_LOB нет).

Дополнения

Экспорт / Импорт

Для переноса БД на другой сервер и/или создания резервной копии используется утилита pg_dump.

pg_dump выгружает только одну базу данных (или ее отдельные объекты, например, только данные). Для сохранения глобальных объектов, относящихся ко всему кластеру, например, пользователей, используется pg_dumpall.

pg_dump создает "согласованную" копию на момент подключения (неважно, что во время работы в БД вносятся изменения).

Дамп, созданный pg_dump, не зависит от операционной системы, архитектуры. Но может зависеть от версии и редакции СУБД.

Известные ограничения:

Из PostgreSQL 13 и ниже в PostgreSQL 14 и выше (и наоборот) Ошибка: *function array_cat(anyarray, anyarray) does not exist* Решение: изменился тип аргументов функций для работы с массивами с anyarray на anycompatiblearray. Необходимо пересоздать объекты, использующие эти функции (см. E.10.2. Migration to Version 14). При импорте – игнорировать ошибки, после импорта выполнить обновление инсталлятором Парус 8, пользовательские объекты при необходимости пересоздать вручную (см. лог ошибок импорта)
Из PostgreSQL 12 и выше в PostgreSQL 11 и ниже Ошибка: *unrecognized configuration parameter "default_table_access_method"* Решение (только для текстового дампа) – удалить "default_table_access_method" из дампа: `sed -i 's/SET default_table_access_method = heap;/-- SET default_table_access_method = heap;/' dump.psql`
Из защищенной СУБД Астра Линукс в "обычный" PostgreSQL При экспорте отключить экспорт меток безопасности `pg_dump --disable-macs --no-security-labels …`
Во время импорта "обнуляется" переменная (search_path=''), поэтому возможны ошибки при создании объектов, зависящих от размещения в схеме. Например, при создании функциональных индексов ("Ошибка: *отношение "options" не существует"*) Решение: создать объекты вручную после импорта, например: CREATE INDEX i_clnpersons_code ON clnpersons USING btree (f_clnpersons_format_code(company, code)); Имя объектов можно получить из лога ошибок

Возможны 2 варианта использования pg_dump:

1. Выгрузка данных в текстовый файл, по сути являющийся скриптом создания БД, который впоследствии выполняется на целевом кластере с помощью утилиты psql.
2. Выгрузка в архивном формате, для последующего восстановления с помощью утилиты pg_restore. Архивные форматы с последующим pg_restore – более универсальный способ, например, можно восстанавливать отдельные объекты, сжимать данные.

Далее рассматривается 1-ый вариант – создание резервной копии БД и ее восстановление.

Экспорт БД

Утилита pg_dump использует те же параметры управления подключением, что и утилита psql, т.е. для подключения необходимо указать пользователя, БД, пароль, адрес сервера, если они отсутствуют по умолчанию (например, пользователь) или не заданы через переменные окружения (например, PGPASSWORD) или отличаются от них.

Также требуется указать параметры формирования дампа, например, имя выходного файла.

Подробнее см. описание pg_dump.

Для выполнения резервного копирования БД с ПП "ПАРУС-Бюджет 8":

```
pg_dump -d demo -U parus -h 127.0.0.1 -p 5432 -f /tmp/demo1.psql -o
```

здесь,

"-d demo" – имя БД.

"-f /tmp/demo1.psql" – файл дампа (сценария).

"-o" (--oids) – выгружать идентификаторы объектов вместе с данными таблиц.

"-U postgres" – пользователь (владелец базы).

Другие полезные опции:

- no-tablespaces – выполнить экспорт без учета табличных пространств, при последующем импорте все объекты будут создаваться в табличном пространстве по умолчанию.
- schema=схема – выгружать только схему (или схемы) с объектами, содержащимися в указанной схеме.
- exclude-schema=схема – не выгружать указанную схему.

Для переноса пользователей в другой кластер (перенаправить вывод в файл /tmp/users.psql):

```
pg_dumpall -g -h 127.0.0.1 -p 5432 -U postgres > /tmp/users.psql
```

Из сценария потребуется вручную удалить пользователя postgres и тех пользователей, которые есть в целевом кластере.

> ***Примечание.*** Для переноса только пользователей достаточно выполнить команду "**pg_dumpall -r**" (переносить только роли), но так как в БД могут использоваться пользовательские табличные пространства и требуется специальное окружение пользователям parus и parus_web, то выполняется "**pg_dumpall -g**" – выгружать глобальные объекты (роли и табличные пространства).

Импорт БД

Здесь под импортом понимается выполнения сценария, подготовленного утилитой pg_dump (если использовались архивные форматы – нужно пользоваться утилитой pg_restore).

Перед выполнением импорта необходимо учитывать следующие обстоятельства:

1. Дамп (сценарий, архив), созданный утилитой pg_dump, содержит определения относительно шаблона template0. Это означает, что используемые расширения, процедуры и т. д., добавленные в базу через template1, также выгружаются. Они должны быть доступны в целевой базе, т. к. создаются командой "CREATE IF NOT EXSISTS".
2. Перед восстановлением дампа все пользователи, которые владели объектами или имели права на объекты в выгруженной базе данных, должны уже существовать.
3. В самом дампе задаются некоторые значения параметров, в том числе search_path (значение ‘’ – пустая строка). Поэтому в процессе импорта могут быть ошибки, связанные с вызовом объектов, выполнение которых зависит от текущей схемы или значения search_path. Например, создание "функциональных" индексов (индексы, при создании которых выполняется какая-либо функция).

4. Желательно создавать целевую базу с тем же именем, что и исходная, т.к. в дампе могут быть "ссылки" на имя БД, например, комментарий COMMENT ON DATABASE или обертки сторонних данных (dblink). Если "ссылок" нет – имя может быть произвольным. Впоследствии базу можно переименовать: ALTER DATABASE *demo* RENAME TO *новое_имя*; (после переименования необходимо выполнить пересоздание сервера автономных транзакций (см. ниже).

Рекомендуемый сценарий импорта:

Проверяется, что доступны требуемые расширения:

```
SELECT * FROM pg_available_extensions WHERE name IN ('hstore', 'dblink', 'xml2', 'uuid-ossp', 'pg_variables', 'http',
'pgzip', 'pgqrcode');
```

Создать пользователя владельца объектов ПП "ПАРУС-Бюджет 8" "parus" и публичную роль:

```
psql -U postgres -h 127.0.0.1 -d postgres
CREATE ROLE parus SUPERUSER LOGIN PASSWORD 'parusina' INHERIT;
CREATE ROLE parus_public NOLOGIN NOINHERIT;
GRANT parus_public TO parus;
```

Если роль уже создана – временно дать привилегию "superuser":

```
ALTER ROLE parus SUPERUSER;
```

Создать БД:

psql -U **parus** -h 127.0.0.1 -d **postgres**	
CREATE DATABASE demo TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE = 'Russian_Russia.1251' LC_CTYPE = 'ru_RU.CP1251';	Создать базу в кодировке WIN1251 (Windows)
CREATE DATABASE demo TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE = **'C'** LC_CTYPE = 'ru_RU.CP1251';	Создать базу в кодировке WIN1251 (Linux)

Примечание. Базу можно не создавать, если экспорт выполнен с ключом --create (pg_dump -C -d demo …). При импорте для подключения указать служебную БД postgres (psql -d **postgres …**).

При необходимости импортировать глобальные объекты (пользователи, их особенные права, табличные пространства):

```
psql -U postgres -d demo -f /tmp/users.psql > /tmp/imp_usr.log
```

Если используются параметры по умолчанию и пользователи не импортируются, этот сценарий можно не выполнять.

Выполнить импорт:

```
psql -U parus -h 127.0.0.1 -d demo -f /tmp/demo.psql > imp.log 2> imp.err
```

После импорта:

psql -U **parus** -h 127.0.0.1 -d **demo**
VACUUM ANALYZE; ALTER ROLE parus IN DATABASE demo SET search_path = parus, sys; SET search_path = parus, sys; REVOKE ALL ON SCHEMA public FROM public; GRANT USAGE ON SCHEMA public TO parus; GRANT USAGE ON SCHEMA public TO parus_public; GRANT ALL ON SCHEMA sys TO parus; GRANT USAGE ON SCHEMA sys TO parus_public; GRANT ALL ON SCHEMA parus TO parus; GRANT USAGE ON SCHEMA parus TO parus_public; Пересоздать сервис автономных транзакций: DROP FOREIGN DATA WRAPPER IF EXISTS parus_autonomous_transaction_service_fdw CASCADE; CREATE FOREIGN DATA WRAPPER parus_autonomous_transaction_service_fdw; CREATE SERVER parus_autonomous_transaction_service_server FOREIGN DATA WRAPPER parus_autonomous_transaction_service_fdw OPTIONS(host '127.0.0.1', port '5432', dbname '**demo**'); CREATE USER MAPPING FOR parus SERVER parus_autonomous_transaction_service_server OPTIONS(USER '**parus**', password '**parusina**');

Если при импорте будут ошибки вида "Ошибка: *отношение "options" не существует"* при создании индексов (см. лог ошибок imp.err), нужно "прогнать" создание индексов вручную, например:

psql -U **parus** -h 127.0.0.1 -d **demo**
CREATE INDEX i_clnpersons_code ON clnpersons USING btree (f_clnpersons_format_code(company, code)); CREATE INDEX i_clnpsdep_code ON clnpsdep USING btree (f_clnpsdep_format_code(company, code)); CREATE INDEX i_geografy_hname ON geografy USING btree (format_hier_name(NULL::numeric, "VERSION", fullname));

При необходимости, создать пользователей БД (если они не были созданы до импорта).

Имеющимся пользователям установить параметр search_path.

psql -U **postgres** -h 127.0.0.1 -d **postgres**
CREATE ROLE user1 LOGIN PASSWORD 'user1' INHERIT; GRANT parus_public TO user1;
ALTER **ROLE** user1 **IN DATABASE** demo SET **search_path** = parus, sys;

Примечание 1. Рекомендуется устанавливать параметр search_path именно "для роли в БД".

Примечание 2. Можно скопировать базу данных непосредственно с одного сервера на другой, например:

pg_dump -h host1 demo | psql -h host2 demo

Настройка доступа (pg_hba.conf)

Каждая запись файла pg_hba.conf состоит из нескольких полей, в которых указываются ключевые слова или конкретные значения:

TYPE	DATABASE	USER	ADDRESS	METHOD	options

Основные используемые значения:

TYPE	тип подключения	**local** – через доменные сокеты Unix **host** – подключение по TCP/IP **hostssl** – подключение по TCP/IP с шифрованием
DATABASE	база данных	*db1* – имя конкретной БД **all** – все БД *db1,db2,db3* – список из нескольких БД **sameuser** – имя БД совпадает с именем пользователя **samerole** – имя БД совпадает с именем роли, назначенной пользователю **@***имя_файла* – имя файла, в котором построчно перечислены БД
USER	пользователь / роль	*user1* – конкретный пользователь **all** – все пользователи *user1*, *user2*, *user3* – список пользователей **+***роль1* – пользователи, которым назначена роль "*роль1*" **@***имя_файла* – имя файла, в котором построчно перечислены пользователи
ADDRESS	клиентский адрес	*адрес1* – конкретный адрес или имя клиентской машины (имя будет разрешаться в IP-адрес стандартными средствами ОС) **all** – любой IP-адрес *172.28.0.0/16* – диапазон адресов (IPv4) в формате "начальный адрес диапазона/длина маски" *172.28.0.0 255.255.0.0* – диапазон адресов (IPv4) в формате "сеть маска" (это два поля) **samenet** – подсеть, к которой подключен сервер
METHOD	метод аутентификации	**reject** – запрет на подключение **trust** – безусловное подключение (вход под любым пользователем без пароля) **peer** – подключение по имени пользователя операционной системы (только для типа local) **md5** или **scram-sha-256** – шифрованный пароль **password** – нешифрованный пароль Другие методы см. здесь
options	параметры метода аутентификации	необязательные параметры выбранного метода аутентификации в формате имя=значение, например, при использовании файла сопоставления имен пользователей ОС и БД pg_ident.conf

Примеры:

local	all	postgres		peer	разрешить локальное подключение через Unix-сокет ко всем БД пользователю postgres (без пароля)
host	all	all	localhost	trust	разрешить локальное подключение через TCP IPv4 или IPv6 любому пользователю без пароля
host	postgres	user1	all	reject	запретить подключение к "postgres" пользователю "user1" со всех адресов
host	all	all	*172.28.0.0/16*	md5	разрешить подключение ко всем БД всем пользователям с адресов из подсети "*172.28.0.0/16*" при указании пароля
host	postgres	+admin	all	md5	разрешить подключение к "postgres" пользователям, с ролью "admin", со всех адресов при указании пароля

Важно!

- Порядок записей в pg_hba.conf имеет значение – записи обрабатываются последовательно до совпадения. Поэтому сначала должны располагаться записи с четкими критериями отбора (пользователь, база, адрес) и слабыми методами аутентификации, затем – записи с обобщенным отбором и "серьезными" методами.
- Проверкой через файл pg_hba.conf выполняется возможность соединения пользователя с БД, далее происходит проверка его привилегий для работы с ней, т.е. у пользователя минимум должна быть роль LOGIN (см. ниже).

Конвертация Oracle-PostgreSQL

Для миграции ПП "ПАРУС-Бюджет 8" на СУБД PostgreSQL разработано следующее ПО:

- Конвертер "Oracle-PostgreSQL" – предназначен для конвертации БД из Oracle в PostgreSQL. Описывается в этом документе.
- Инсталлятор ParusPG.exe – предназначен для создания новой БД и обновления имеющейся (аналогичен инсталлятору ParusBUDGET.msi для Oracle). Описание в главе "Инсталлятор для PostgreSQL. Создание и обновление БД".

Установка и подготовка самой СУБД PostgreSQL для работы с ПП "ПАРУС-Бюджет 8" описаны в главе "Установка PostgreSQL".

Процесс конвертации или создания БД позволяет развертывать ПП "ПАРУС-Бюджет 8" в том числе и на удаленных площадках, например, в облаке, когда отсутствует доступ к операционной системе и СУБД.

Для некоторых действий все равно потребуются права суперпользователя (владельцем площадки), их можно оформить в виде сценария.

Функционал доступен для релизов после 01.2024.

Внимание! Процесс конвертации находится в стадии разработки и окончательный алгоритм может отличаться от описываемого ниже.

Рекомендуемая последовательность действий при конвертации:

1. Установка конвертера.
2. Создание и подготовка целевой БД PostgreSQL (расширения, роли, схемы).
3. Подготовка исходной БД Oracle (стерилизация).
4. Конвертация.
5. Перенос/создание пользователей.
6. Выполнение сценария _AfterConvert_1_by_PARUS.pgsql и функции pg_temp.DO_AFTER_CONVERT.
7. Установка инсталлятора ParusPG.
8. Обновление БД инсталлятором ParusPG (или выполнение сценария _AfterConvert_2_by_PARUS.pgsql).

Установка конвертера

Системные требования

- Операционная система: MS Windows 7 и выше (рекомендуются редакции x64) или Windows Server 2008 R2 и выше.
- Клиент Oracle версии 11.2.0.4 или 10.2.0.5 **той же разрядности**, что и конвертер.
- Для работы с БД PostgreSQL и/или генерации клиента – требуется установленный клиент ПП "ПАРУС-Бюджет 8", желательно той же редакции, что и БД Oracle (ParusClient.msi или ParusBUDGET.msi).

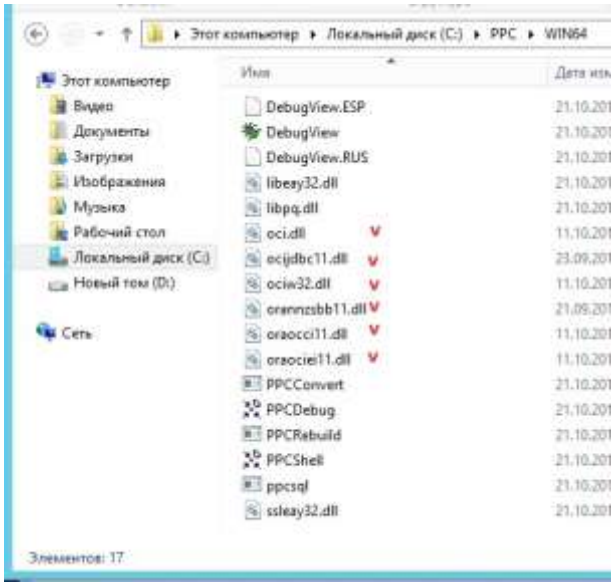
Все остальные требуемые компоненты включены в состав инсталлятора.

Примечание. Есть консольная 64-битная версия конвертера для ОС Linux.

32-х битный конвертер устанавливается всегда, т.к. его компоненты будут использоваться приложением ПП "ПАРУС-Бюджет 8" (32-бита) при работе с PostgreSQL после конвертации.

На 64-х разрядных ОС рекомендуется **НЕ выключать** флаг "Установить компоненты редакции x64" при установке и работать с редакцией x64.

Внимание! Для работы с БД Oracle 64-х разрядному конвертеру потребуется 64-х разрядный клиент Oracle.



Если 64-х разрядный клиент Oracle не установлен можно воспользоваться [Instant Client for Microsoft Windows (x64)](#).
Скачать [instantclient-basic-windows.x64-11.2.0.4.0.zip](#)
Распаковать dll-файлы из архива instantclient-basic-windows.x64-11.2.0.4.0.zip в папку, где установлены 64-х разрядные компоненты конвертера, в данном случае C:\PPC\WIN64 (dll-файлы от клиента должны находиться рядом с exe-файлами конвертера).
Можно создать здесь же файл локального именования tnsnames.ora для обращения к БД Oracle по псевдониму, или использовать метод Easy Connect, как будет показано ниже.
ОБЯЗАТЕЛЬНО при использовании Instant Client нужно задать переменную окружения NLS_LANG=AMERICAN_AMERICA.CL8MSWIN1251

Для установки рекомендуется использовать отдельную машину (кроме конвертации больших БД с ключом /copydata см. ниже).

При этом необходимо выполнить следующие системные требования:

- Свободное место на диске, где расположена папка %TEMP% – минимум 15ГБ.
- Объем памяти в зависимости от количества ядер процессора (без учета другого ПО):

Ядра CPU	ОЗУ, ГБ
2	4
4	6
8	10
16	18

Одноядерные системы не поддерживаются.

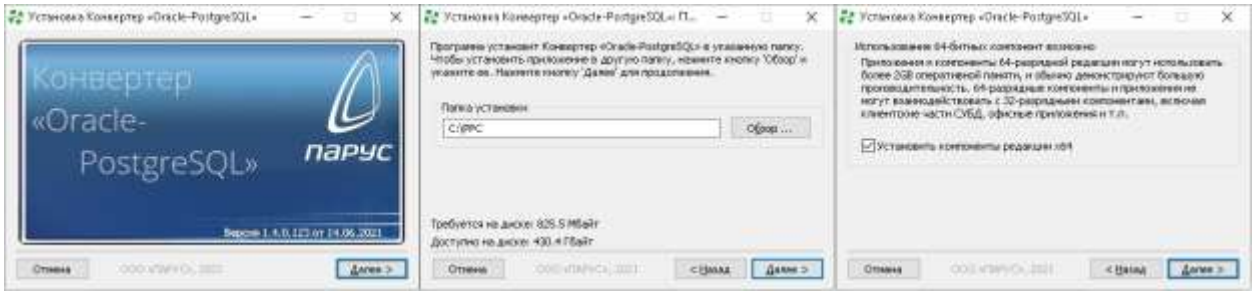
Поддерживаемые версии **СУБД** Oracle: 11.2.0.4 (версия Oracle 10 не поддерживается).

Рекомендуемые версии СУБД PostgreSQL 9.6 и выше:

- Linux – из состава дистрибутива.
- Windows – [PostgresPro](#).

Установка

Запустить инсталлятор ppc.exe, указать путь установки, например C:\PPC.



Подготовка к конвертации

Создание целевой БД PostgreSQL

Предполагается, что уже развернут кластер PostgreSQL, заданы требуемые параметры, настроен доступ. Подробнее см. главу "Установка PostgreSQL".

Команды выполняются последовательно с помощью штатной утилиты командной строки psql с любого рабочего места (рекомендуется, чтобы версия утилиты была не меньше версии БД на сервере). Можно воспользоваться сторонней средой администрирования/разработки для PostgreSQL, например, pgAdmin, DBeaver, datagrip, dbForge.

Создать пользователя кластера, который в дальнейшем будет администратором ПП "ПАРУС-Бюджет 8" (рекомендуется "parus"). При конвертации имя пользователя "parus" должно совпадать с аналогичным пользователем Oracle (владельцем схемы с объектами ПП "ПАРУС-Бюджет 8").

Дать привилегию "superuser" пользователю "parus" (привилегия выдается временно, на время конвертации, первого создания/обновления или импорта).

psql -h *host* **-U postgres**
CREATE ROLE *parus* **LOGIN PASSWORD '***parusina***' INHERIT;** **ALTER ROLE parus SUPERUSER;**

Создать групповую роль, например, "parus_public", в которую в дальнейшем будут включены все пользователи БД, которые работают с ПП "ПАРУС-Бюджет 8".

Включить пользователя "parus" в групповую роль "parus_public".

psql -h *host* **-d postgres -U parus**
CREATE ROLE parus_public NOLOGIN NOINHERIT; **GRANT parus_public TO parus;**

Создать целевую БД (например, demo).

Подключение здесь и далее (до указания другого) необходимо выполнять под пользователем "parus" для того, чтобы он стал владельцем объектов.

Если кластер использует кодировку UTF-8, нужно использовать команду создания БД с указанием кодировки:

Windows:

"CREATE DATABASE demo TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE = 'Russian_Russia.1251' LC_CTYPE = 'Russian_Russia.1251';"

Linux (см. главу "Создание БД"):

CREATE DATABASE demo TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE = 'C' LC_CTYPE = 'ru_RU.CP1251';

psql -h *host* **-d postgres -U parus**
 # Кластер в кодировке WIN1251 **CREATE DATABASE** *demo***;** *# или кластер в кодировке UTF8 на Windows* **CREATE DATABASE** demo **TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE = 'Russian_Russia.1251' LC_CTYPE = 'Russian_Russia.1251';** # или кластер в кодировке UTF8 на Linux **CREATE DATABASE** demo **TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE = 'C' LC_CTYPE = 'ru_RU.CP1251';**

В перечисленных выше командах не важно к какой БД выполняется подключение, но обычно для этого используется БД, которую принято считать служебной (БД по умолчанию) – postgres.

Дальнейшие действия выполняются в созданной БД под созданным пользователем ("parus").

При необходимости, создать табличные пространства (см. главу "Использование табличных пространств").

Подготовка целевой БД PostgreSQL

Регистрация расширений.

Зарегистрировать расширения pg_variables (версии >=1.1) http pgzip pgqrcode.

psql -h *host* **-d demo -U parus**
CREATE EXTENSION IF NOT EXISTS dblink**;** **CREATE EXTENSION IF NOT EXISTS** "uuid-ossp"**;** **CREATE EXTENSION IF NOT EXISTS xml2;** **CREATE EXTENSION IF NOT EXISTS hstore;** **CREATE EXTENSION IF NOT EXISTS pg_variables;** **CREATE EXTENSION IF NOT EXISTS http;** **CREATE EXTENSION IF NOT EXISTS pgzip;** **CREATE EXTENSION IF NOT EXISTS pgqrcode;**

здесь:

-h host: адрес/имя сервера PostgreSQL, при локальном подключении можно не указывать.

-U <пользователь>: суперпользователь, который выполняет действия.

-d <БД>: база данных для подключения (если не указывается, то будет использована база по умолчанию, совпадающая с именем пользователя или заданная переменной PGDATABASE).

Связанные с расширениями объекты размещаются в схеме "public".

Проверка:

SELECT * FROM pg_available_extensions WHERE installed_version IS NOT NULL;	Список установленных расширений Требуются: dblink, pg_variables, uuid-ossp, hstore, xml2, http
SELECT * FROM pg_catalog.pg_roles WHERE rolname = 'parus';	Проверка пользователя parus
SELECT * FROM pg_settings WHERE name in ('max_locks_per_transaction', 'from_collapse_limit', 'join_collapse_limit');	Проверка параметров (см. главу "Установка PostgreSQL")

Публичная роль

Для схемы "public" забрать привилегии у псевдороли "public" и выдать соответствующие привилегии пользователю "parus" и групповой роли "parus_public".

```
REVOKE ALL ON SCHEMA public FROM public;
GRANT USAGE ON SCHEMA public TO parus;
GRANT USAGE ON SCHEMA public TO parus_public;
```

Системная схема sys

Создать системную схему "sys" и выдать соответствующие привилегии пользователю "parus" и групповой роли "parus_public".

```
CREATE SCHEMA sys;
GRANT ALL ON SCHEMA sys TO parus;
GRANT USAGE ON SCHEMA sys TO parus_public;
```

Имя системной схемы ("sys") фиксированное и не может быть другим.

В рамках одной базы данных может быть только одна схема системная схема.

В этой схеме будут размещаться данные конвертера и "системные объекты Oracle", которые используются в прикладном коде.

Прикладная схема

Создать прикладную схему, например, "parus" и выдать соответствующие привилегии пользователю "parus" и групповой роли "parus_public".

```
CREATE SCHEMA parus;
GRANT ALL ON SCHEMA parus TO parus;
GRANT USAGE ON SCHEMA parus TO parus_public;
```

Имя прикладной схемы может быть произвольным, например, "parus" или "application".

В этой схеме будут размещаться данные и объекты приложений ПП "ПАРУС-Бюджет 8".

Для пользователя "parus" установить путь поиска объектов – перечислить прикладную и системную схемы:

```
ALTER ROLE parus IN DATABASE demo SET search_path = parus, sys;
SET search_path = parus, sys; -- установка для текущего сеанса, чтобы не перелогиниваться
```

Рекомендуется задать путь поиска только для конкретной БД (опция "IN DATABASE"), а не во всем кластере, чтобы не влиять на другие установки ПП "ПАРУС-Бюджет 8".

Сервис автономных транзакций

Создать сервис автономных транзакций (ATS):

```
CREATE FOREIGN DATA WRAPPER parus_autonomous_transaction_service_fdw;

CREATE SERVER parus_autonomous_transaction_service_server
  FOREIGN DATA WRAPPER parus_autonomous_transaction_service_fdw
  OPTIONS(host '127.0.0.1', port '5432', dbname 'demo');

CREATE USER MAPPING FOR parus
  SERVER parus_autonomous_transaction_service_server
  OPTIONS(USER 'parus', password 'parusina');
```

Имена обертки данных сервиса автономных транзакций (fdw) и сервера сервиса автономных транзакций (server) фиксированные.

В опциях создания сервера рекомендуется указывать локальное соединение по протоколу IPv4.

При работе ПП "ПАРУС-Бюджет 8" для PostgreSQL используется локальное соединение, созданное безопасной функцией dblink_connect, которая требует аутентификацию. Поэтому необходимо согласование правил локального доступа к базе данных (настройки в файле pg_hba.conf) и параметров, используемых при создании сервера автономных транзакций.

В качестве параметров функции передаются логин, пароль пользователя-владельца схемы (обычно 'parus') и адрес локального подключения (по умолчанию – 'localhost').

В качестве адреса локального подключения могут быть использованы следующие значения:

• 127.0.0.1 – локальный адрес, протокол IPv4;

• ::1 – локальный адрес, протокол IPv6;

• localhost – имя локального адреса, в зависимости от ОС и сетевых настроек может разрешиться как через IPv4 (127.0.0.1), так и IPv6 (::1);

• /var/run/postgresql – unix-сокет. Подключение доступно только для Linux, конкретное значение определяется настройкой "unix_socket_directories" (show unix_socket_directories; select current_setting('unix_socket_directories');

Используемое в качестве параметра подключение должно быть описано в файле pg_hba.conf как безопасное (обычно md5 или scram-sha-256 для версий 12 и выше).

Примеры создания сервера для ATS с учётом правил доступа к базе данных:

psql -U postgres -d demo	Подключение к БД (по TCP/IP или local)
Создание сервера автономных транзакций	Строка в pg_hba.conf
CREATE SERVER … OPTIONS (host '***127.0.0.1***', port '***5432***', dbname '***demo***');	host all all 127.0.0.1/32 md5 или host demo all 127.0.0.1/32 md5 или host demo parus 127.0.0.1/32 md5
CREATE SERVER … OPTIONS (host '***localhost***', port '***5432***', dbname '***demo***');	Будет создано подключение с адресом 'localhost', поэтому, для гарантирования доступа: host all all 127.0.0.1/32 md5 host all all ::1/128 md5
CREATE SERVER … OPTIONS (dbname '***demo***');	Только для Linux: local all all md5

Подготовка исходной БД Oracle

Внимание! Перед внесением изменений в БД необходимо создать ее резервную копию или выполнить стерилизацию на копии БД.

Дальнейшие действия могут привести к безвозвратной потере данных!

После выполнения стерилизации приложения ПП "ПАРУС-Бюджет 8" работать не будут!

Для работы конвертера требуется служебный пользователь POSTGRES (и в БД Oracle и в ПП "ПАРУС-Бюджет 8"). Поэтому рекомендуется его создать (если этого не сделать – будет переименован пользователь PARUS):

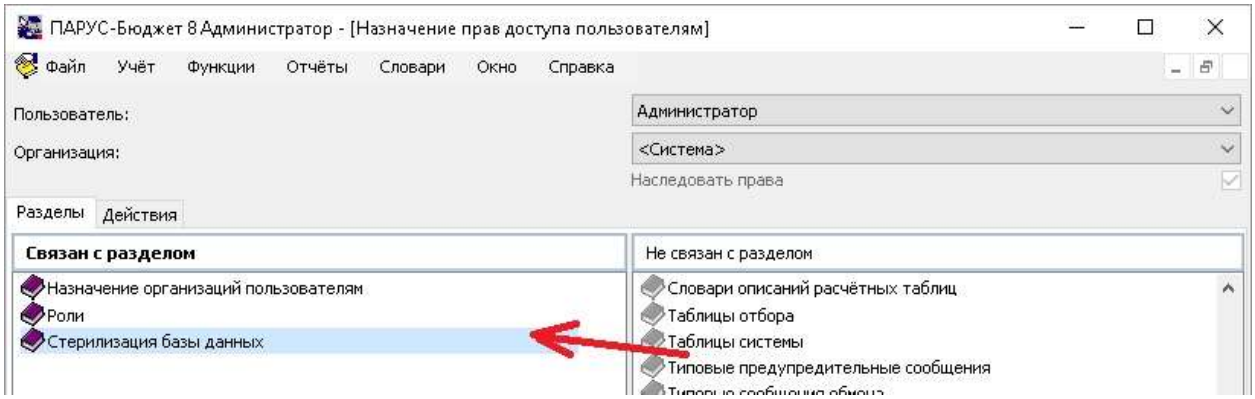
```
create user POSTGRES identified by postgres default tablespace PARUS_MAIN temporary tablespace TEMP;
grant CREATE SESSION to POSTGRES;
```

Далее в приложении "Администратор" в разделе "Пользователи" добавить пользователя. При этом требуется разрешить "Сеанс базы данных".

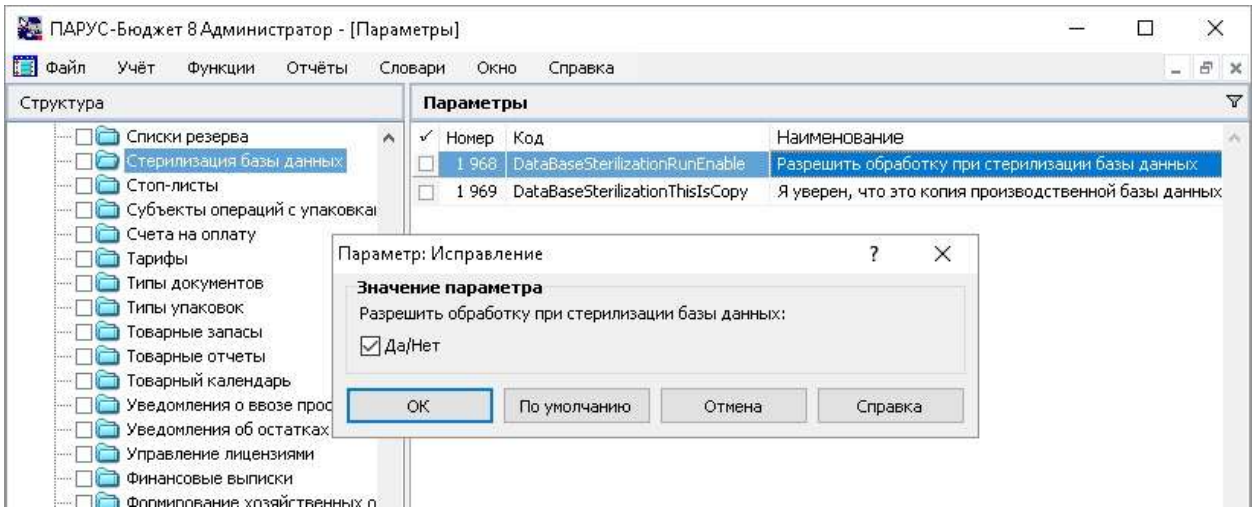
Никаких прав пользователю назначать нельзя!

Для подготовки БД к конвертации создан раздел "Функции \ Обслуживание \ Стерилизация базы данных" приложения "Администратор" (**с релиза 27.10.2021**).

Сделать раздел доступным пользователю-владельцу схемы (обычно PARUS) – дать права на раздел:



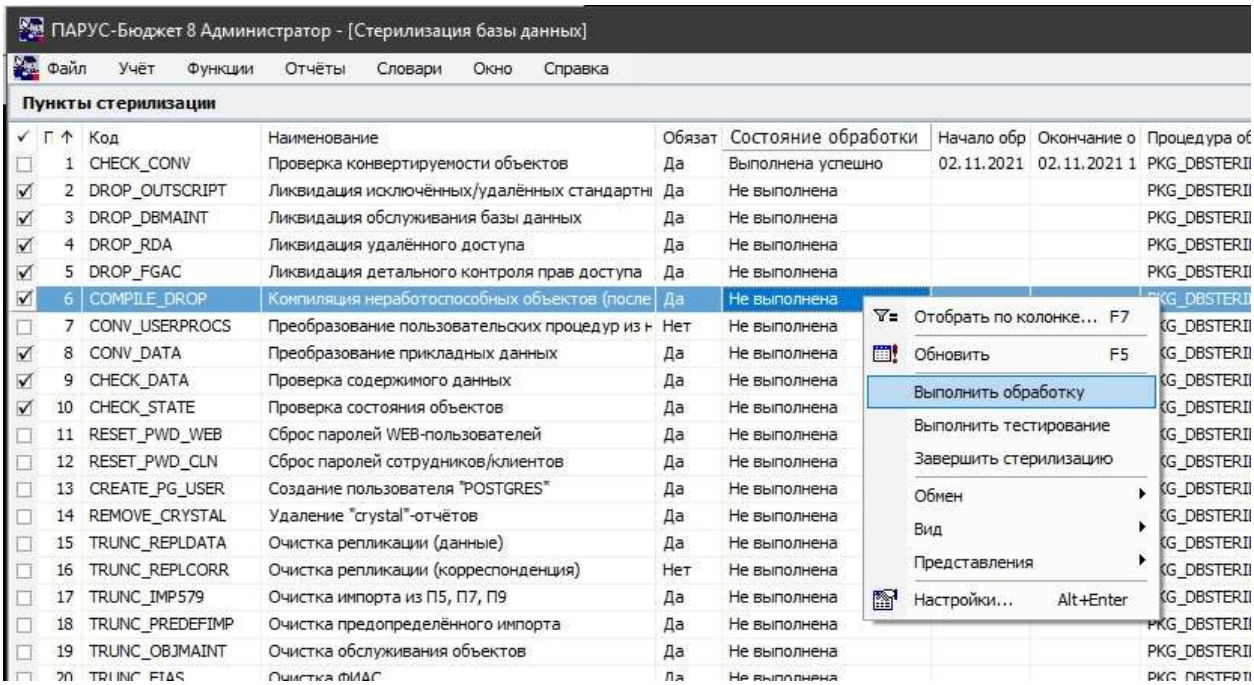
Включить 2 настройки (1968, 1969), позволяющие выполнять действия в разделе ("Файл \ Сервис \ Параметры"):



Раздел состоит из последовательности действий в предопределенном порядке.

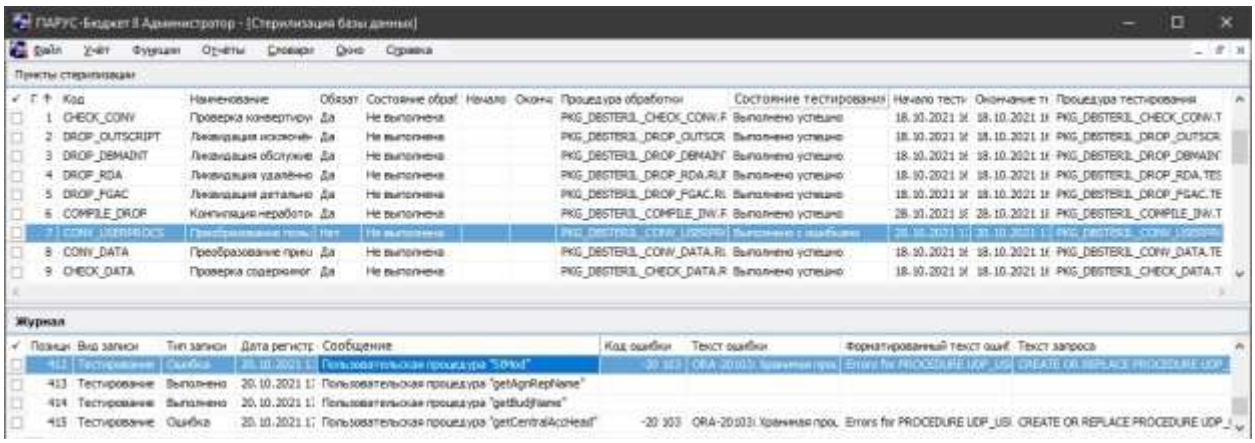
В разделе доступны следующие действия:

- "Выполнить тестирование" – выполнить действие (или выбранные действия последовательно) в безопасном режиме тестирования, без внесения изменений в БД.
- "Выполнить обработку" – выполнить действие (или выбранные действия последовательно). Каждое действие выполняется в случае, если все предыдущие обязательные действия находятся в состоянии обработки "Выполнено успешно".
- "Завершить стерилизацию" – действие выполняется однократно после выполнения всех обязательных обработок (удаляются объекты, относящиеся к стерилизации, выставляются необходимые настройки).



Какие именно действия выполняются для конкретного пункта можно посмотреть в теле пакета PKG_DBSTERIL_<Код> или в СХС (раздел "DataBaseSterilization \ DataBaseSterilizationRunTest").

В случае возникновения ошибок при тестировании / обработке, требуется их исправить и выполнить действие повторно. Текст запроса, приведшего к ошибке, и текст ошибки будут представлены в журнале.



Примечание. В случае ошибки *"ORA-01000: maximum open cursors exceeded"* нужно увеличить параметр open_cursors в init.ora.

Важно:

- Не выходите из приложения "Администратор" после обработки действия STOP_LICENSE (Остановка сервиса контроля лицензий). Дальнейшая работа через приложение будет невозможна.
- Если у пользователя владельца-схемы (parus) "отбиралась" роль DBA, то возможно потребуется восстановить минимально необходимые для работы ПП "ПАРУС-Бюджет 8" привилегии:

```
alter user PARUS quota unlimited on PARUS_MAIN;
alter user PARUS quota unlimited on PARUS_INDEX;
alter user PARUS quota unlimited on PARUS_LOB;
grant CREATE SESSION, ALTER SESSION, CREATE PROCEDURE, CREATE SEQUENCE, CREATE SNAPSHOT, CREATE PUBLIC
SYNONYM, DROP PUBLIC SYNONYM, CREATE VIEW, CREATE TABLE, CREATE TRIGGER, CREATE ANY CONTEXT to PARUS;
grant CREATE JOB to PARUS;
grant DROP ANY CONTEXT to PARUS;
grant SELECT on gv_$session to PARUS;
grant EXECUTE on dbms_pipe to PARUS;
grant EXECUTE on dbms_lock to PARUS;
grant CTXAPP to PARUS;
grant execute on CTX_DDL to PARUS;
```

Конвертер "Oracle-PostgreSQL"

Приложение " Конвертер "Oracle-PostgreSQL" предназначено для миграции ПП "ПАРУС-Бюджет 8" с СУБД Oracle на PostgreSQL без потери функциональности.

Приложение состоит из нескольких утилит:

- **Оболочка** (PPCShell.exe) – "стартовая страница" приложения для выбора дальнейших действий:

- **Конвертер** (PPCConvert.exe) – утилита конвертации объектов и переноса данных ПП "ПАРУС-Бюджет 8" из БД Oracle в PostgreSQL или создания/обновления БД PostgreSQL по исходным кодам для Oracle.
- **Коннектор** (PPCConnect.dll) – библиотека доступа клиента ПП "ПАРУС-Бюджет 8" к СУБД PostgreSQL (заменяет oci.dll).
- **Утилита форматирования** PL/pgSQL-кода (PPCFormat.exe) для повышения удобства чтения результирующего кода.
- **Утилита конвертации отдельных объектов** (PPCDebug.exe) для оперативной отладки и внесения отдельных изменений.
- **Служебная утилита** (PPCRebuild.exe) для работы с метаданными конвертера.
- **Отладочный монитор** (DebugView.exe) для локализации проблем. Отображает информацию о происходящих действиях в реальном времени.

Конвертация

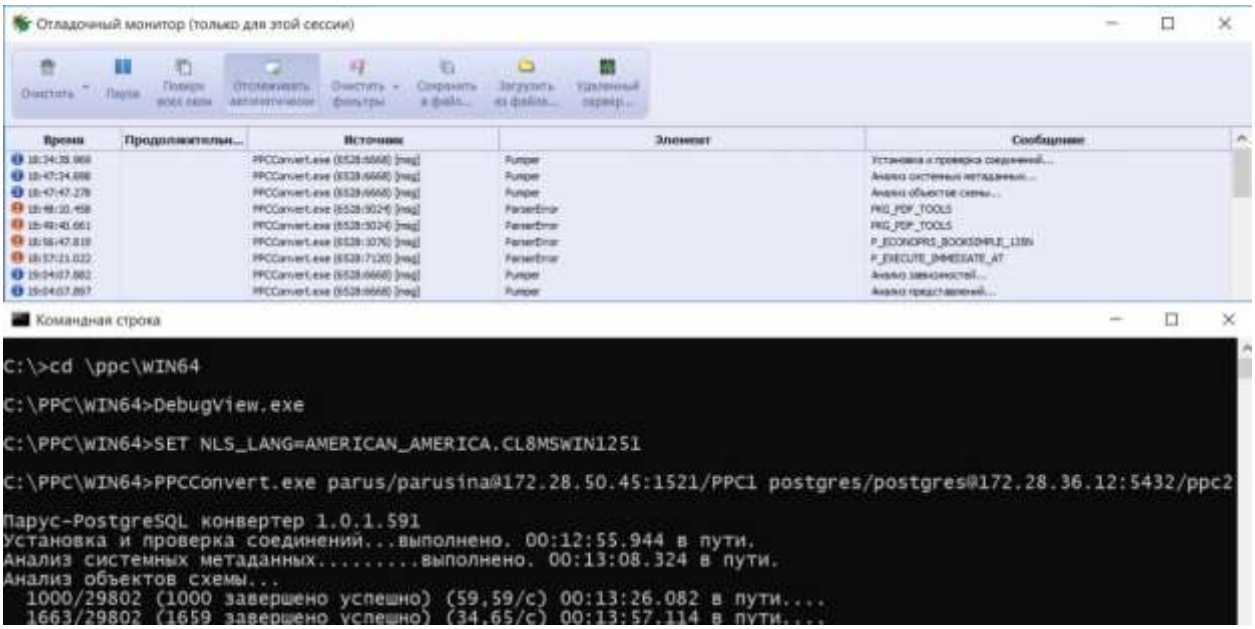
Существует несколько вариантов использования конвертера:

- **Конвертация** из БД Oracle в БД PostgreSQL.
- Отладка – Конвертация отдельных объектов.
- Создание сценариев **импорта** из БД Oracle в БД PostgreSQL (на выходе – скрипты создания БД, что удобно, если сервер PostgreSQL в данный момент недоступен). В качестве параметра целевой БД конвертеру указывается папка.
- Создание сценариев **создания/обновления** БД PostgreSQL из развернутого инсталлятора ПП "ПАРУС-Бюджет 8" для Oracle. В качестве параметра исходной базы указывается папка с инсталлятором (в ней должна быть папка \Scripts), в качестве целевого параметра возможно указание БД PostgreSQL или папки для сценариев импорта.

Здесь рассматривается вариант конвертации. В связи с тем, что выпущен инсталлятор ПП "ПАРУС-Бюджет 8" для PostgreSQL (ParusPG), использование вариантов с сценариями нецелесообразно.

Внимание! Перед запуском конвертера убедитесь, что во временной папке %TEMP% нет каталогов и файлов вида DMLxxxx, которые могли остаться от выполнявшихся ранее конвертаций. Это может привести к следующей ошибке: *"Exception: Temporary folder is inaccessible. Shared access error"*.

Удобней выполнять конвертацию из командной строки (**PPCConvert.exe**), оболочку (PPCShell.exe) использовать для конвертации не рекомендуется.



Сначала запускается отладчик. Полученную информацию в дальнейшем можно использовать для анализа ошибок.

Затем выполняется команда конвертации:

```
cd PPC\WIN64
SET NLS_LANG=AMERICAN_AMERICA.CL8MSWIN1251
DebugView.exe -filters 1;3;5
PPCConvert.exe parus/parusina@ora_server:1521/orcl1 parus/parusina@pg_server:5432/demo /format
/comments /j 8 /publicschema sys /targetschema parus /noschemaspec /publicrole parus_public
```

Параметры запуска конвертера:

- parus/parusina@ora_server:1521/orcl1 – строка подключения к Oracle в формате Easy connect (user/password@сервер:порт/sid), где parus/parusina – логин и пароль владельца схемы, где установлен ПП "ПАРУС-Бюджет 8". Если используется метод локального именования (TNS), достаточно указать псевдоним из tnsnames.ora;
- parus/parusina@pg_server:5432/demo – строка подключения к PostgreSQL, где parus/parusina - логин и пароль суперпользователя, владельца БД "demo".

Установка переменной окружения NLS_LANG обязательна при использовании Oracle Instance Client.

Далее следуют ключи запуска.

Важно! Не меняйте ключи и их значения по своему усмотрению! Конвертер – это универсальное решение, но для работы ПП "ПАРУС-Бюджет 8" требуются конкретные настройки. По сути, для изменения доступны только:

- **имя** прикладной схемы (targetschema);
- **имя** публичной роли (publicrole).

Дополнительно можно управлять:

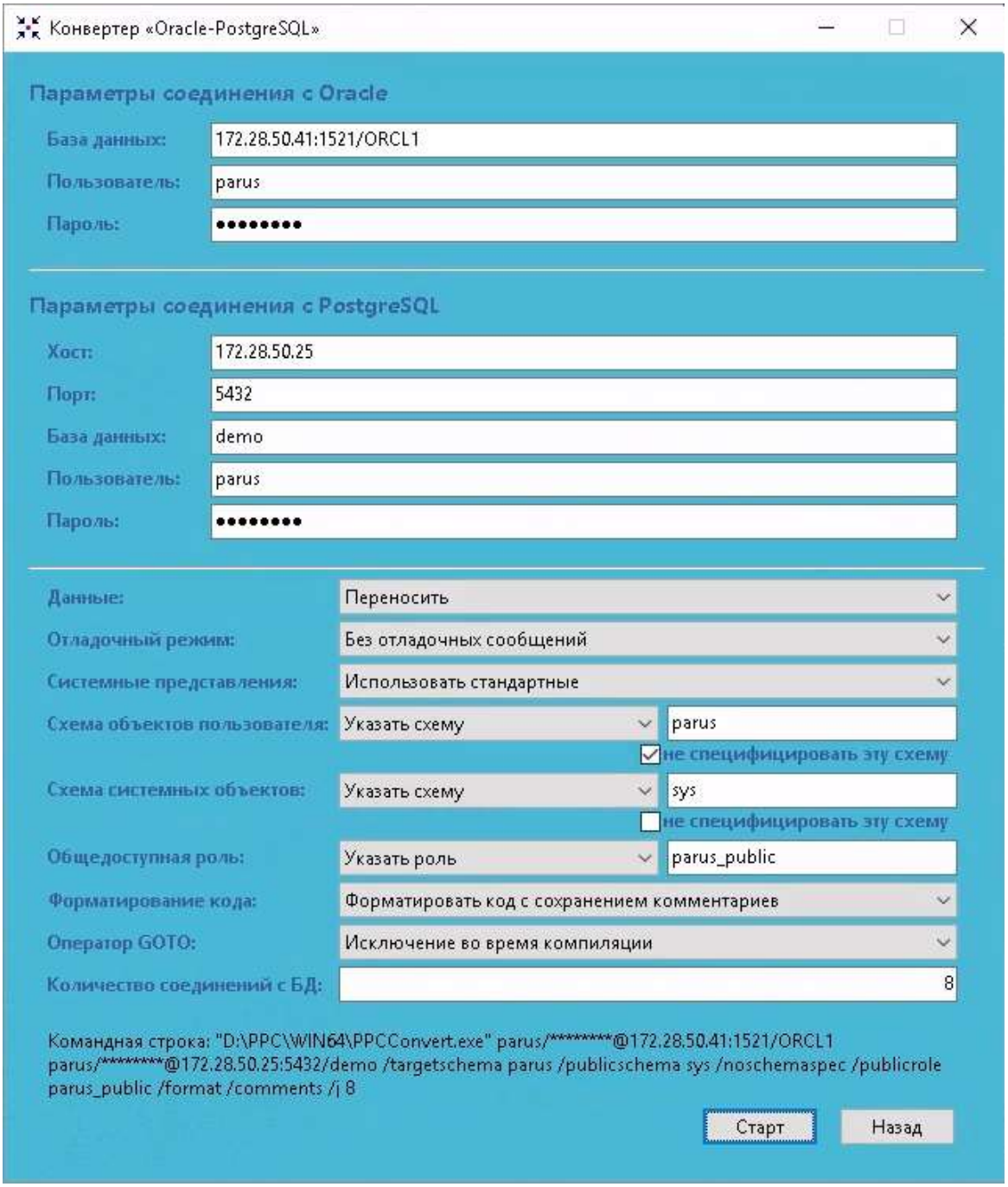
- количеством соединений с БД (j, если указано);
- переносом комментариев в телах функций и триггеров (comments);
- ошибками при конвертации объектов с оператором GOTO (goto);
- копированием таблиц для больших БД (copydata).

Ключи запуска конвертера и оболочки

Ключ	Назначение	Описание
Конвертер		
/j <N>	включить	Количество соединений конвертера с БД PostgreSQL, где <N>-целое число >= 2 Значение по умолчанию (если не задано) определяется автоматически по количеству ядер в системе, в которой запущен конвертер Рекомендуется устанавливать значение равное количеству ядер/потоков **на сервере БД** PostgreSQL
/publicschema sys	включить	Указание ранее созданной схемы для системных объектов обязательно, имя схемы “sys” – фиксированное Отсутствие флага /nopublicschemaspec (не специфицировать схему) указывает на то, что в коде конвертируемых объектов имя схемы будет указываться перед их именем, например, sys.substr, а не substr
/targetschema parus /noschemaspec	включить	Указание ранее созданной схемы объектов пользователя (прикладная или целевая схема) Флаг /noschemaspec (не специфицировать схему) указывает на то, что в коде конвертируемых объектов имя схемы не будет указываться перед их именем, например, aglinst, а не parus.agnlist
/publicrole parus_public	включить	Указание ранее созданной публичной (общедоступной) роли
/format	включить	Форматирование результирующего кода 1. Форматировать код во время генерации. ppcconvert будет форматировать тела функций "на лету". Использование этого режима замедлит конвертацию. 2. В состав инсталлятора включена утилита ppcformat, которую можно "натравить" на БД, и она отформатирует все функции в БД: ppcformat postgres/pass@server/database
/comments	включить	Сохранение комментариев в телах функций и триггеров Работает только с ключом /format
/debug	отладка	Конвертация в отладочном режиме В логе показывается каждый вызов каждой функции со всеми значениями всех параметров Это помогает при отладке, но замедляет работу
/copydata	для больших БД на Windows	Конвертации огромных баз (от 20-30 Гб) В обычном режиме для переноса данных используется оператор INSERT, и перенос происходит строка за строкой В режиме "/copydata" будет формироваться csv-файл, который будет загружаться в БД с помощью оператора COPY, что гораздо более эффективно. При этом надо иметь в виду следующие обстоятельства: - Необходимо, чтобы целевая БД и конвертер работали на одной машине (соответственно, это Windows, готовую БД потом можно отнести на Linux) - Таблицы либо загрузятся целиком (оператор COPY выполнится), либо нет. Сообщение об ошибке оператора COPY ляжет в лог, и с этим нужно будет разбираться. - Сами csv-файлы будут создаваться в папке %TEMP%, и там должно быть достаточно места.

Дополнительные ключи		
/goto	включить/ тестирование	Т.к. конвертация объектов с оператором GOTO не поддерживается, данный параметр определяет поведение конвертера: - без ключа (по умолчанию) – исключение поднимается во время компиляции, ошибка попадает в лог, объект не конвертируется - с ключом – выражение <goto mylabel> конвертируется в выражение <raise exception '%', 'GOTO statement is not supported.'> Исключение будет подниматься во время выполнения
/nodata	отладка/ тестирование	Не переносить данные, ограничиться конвертацией объектов
/nosysviews	не использовать	Не создавать "системные представления Oracle"
/save	отладка	Формируются файлы в папке %ProgramData%\Techmill\PPCConvert: storage.bin – хранилище документов script.sql – полный скрипт создания БД postgres, который содержит (в правильных местах) ссылки на файлы data*.sql и converter.sql; этот скрипт можно использовать, в том числе, и для поиска converter.sql – скрипт загрузки метаданных конвертера data*.sql – скрипты загрузки данных Скрипты никак не привязаны к имени БД. То есть можно взять "с собой" файлы script.sql, converter.sql и data*.sql, отнести их куда угодно, и там прогнать script.sql. Будет создана абсолютно работоспособная БД postgres.
Оболочка (см. Работа с БД PostgreSQL)		
/keep_temporary_objects	не использовать	Удаление временных объектов после закрытия соединения По умолчанию удаление происходит сразу после использования
/no_autonomous_transactions	не использовать	Сейчас можно организовать вызов "автономных" функций в том же соединении, указав при запуске p8application.exe ключ /no_autonomous_transactions. Проблема с автономными транзакциями. insert into selectlist ....; ... commit; ... autonomous_procedure(....); Запрещается явное управление транзакциями, поэтому автономная транзакция не видит изменений вызывающей транзакции
/debug	отладка	Работа в отладочном режиме Помогает при отладке, но замедляет работу

Соответствующая команда в оболочке (PPCShell.exe) будет выглядеть так:



После конвертации

Краткая последовательность действий:

- (Опция) Перенос или создание пользователей БД.
- Выполнение сценария _AfterConvert_1_by_PARUS.pgsql и выполнение функции pg_temp.DO_AFTER_CONVERT.
- Обновление БД инсталлятором ParusPG или выполнение сценария _AfterConvert_2_by_PARUS.pgsql.

Пользователи БД

Пользователи кластера PostgreSQL потребуются для:

- служебных ролей (фоновые и анонимные пользователи);
- пользователей ПП "ПАРУС-Бюджет 8" работающих через WIN-клиент.

Пользователь-администратор ПП "ПАРУС-Бюджет 8" (обычно "parus") был создан ранее, до создания БД и конвертации.

Для пользователей ПП "ПАРУС-Бюджет 8", работающих только в веб-приложениях (Парус-Онлайн, Веб-своды, Личный кабинет), создание соответствующих пользователей БД не требуется.

Вариант 1. Создание пользователей вручную:

Создать (CREATE) пользователя или обновить (ALTER), назначить ему групповую роль (parus_public), задать путь поиска search_path:

```
CREATE ROLE user1 LOGIN PASSWORD 'passwd' INHERIT;
GRANT parus_public TO user1;
ALTER ROLE user1 IN DATABASE demo SET search_path = parus, sys;
```

Вариант 2. Генерация сценария создания пользователей.

Доступно, если при стерилизации БД Oracle выполнялось действие "Выгрузка пользователей базы".

Выполняется запрос, который генерирует команды создания пользователей. Эти команды потом выполняются отдельно или в виде sql-скрипта (см. Вариант 2 в user.txt). Вывод запроса можно записать в файл.

```
psql -U parus -d demo --quiet --tuples-only --output=/tmp/users.psql
```

```
select t.statement
    from
    (...
;
\q
```

```
psql -U parus -d demo -f /tmp/users.psql
```

Вариант 3. Генерация функции создания пользователей и ее выполнение.

Доступно, если при стерилизации БД Oracle выполнялось действие "Выгрузка пользователей базы".

Выполняется запрос, который генерирует функцию создания пользователей (pg_temp.create_other_users). Функция потом выполняются в этом же сеансе (см. Вариант 3 в user.txt).

```
psql -U parus -d demo
```

```
create or replace function pg_temp.create_other_users
   (...
;
select pg_temp.create_other_users('parus_public','parus');
\q
```

_AfterConvert_1_by_PARUS.pgsql

Сценарий создает необходимые "системные" объекты, отсутствующие в исходной БД Oracle, также выполняет ряд проверок и настроек.

Сценарий доступен в каталоге соответствующего инсталлятора:
ftp.parus.ru/master_disk/PARUS_8/БЮДЖЕТ/Парус_8_5_6_1_XXXX/SERVICE/Parus8.PostgreSQL/
или генерируется самостоятельно (см. далее)

Выполнение:

```
psql -U parus -d demo
\i <путь_к_сценарию>\_AfterConvert_1_by_PARUS.pgsql
select pg_temp.DO_AFTER_CONVERT('parus','parus_public','parus');
\q
```

Здесь параметры: 'parus' – <владелец БД>, 'parus_public'- публичная роль>, 'parus' – <прикладная схема>.

Важно! Выполнить 1-й сценарий постконвертации и функцию pg_temp.DO_AFTER_CONVERT необходимо в одном сеансе! Функция доступна в схеме pg_temp только для этого сеанса.

Обновление инсталлятором ParusPG / _AfterConvert_2_by_PARUS.pgsql

Требуется установить инсталлятор и выполнить обновление.

Описание см. в главе "Инсталлятор PostgreSQL".

На сегодняшний день инсталлятор ПП "ПАРУС-Бюджет 8" для PostgreSQL (ParusPG) поддерживает не все модули. Перечень модулей и сервисов, которые можно создать/обновить данным инсталлятором можно найти для соответствующего релиза в файле SERVICE\Parus8.PostgeSQL\Readme.txt.

Если использование инсталлятора нежелательно или в лицензии имеются не поддерживаемые на данный момент приложения, можно выполнить сценарий _AfterConvert_2_by_PARUS.pgsql от имени пользователя-владельца схемы (parus).

```
psql -h host -U parus -d ppc1 -f "_AfterConvert_2_by_PARUS.pgsql"
```

Содержимое сценария зависит от состава приложений, поэтому генерируется пользователем.

Сценарии _After_Convert (опция)

Сценарии 1 и 2 генерируются из СХС (для соответствующего релиза ПП "ПАРУС-Бюджет 8").

Сценарий 1 – универсальный, поставляется с инсталлятором.

Сценарии 2 – зависит от состава лицензии, генерируется пользователем.

Подготовка сценариев

_AfterConvert_1_by_PARUS.pgsql – сценарий проверки/настройки системного окружения в PostgreSQL:

- Настроить соединение с СХС (файл oreader.ini).
- Выполнить _AfterConvert_1_by_PARUS.cmd.
- На выходе – Сценарий 1: _AfterConvert_1_by_PARUS.pgsql.

_AfterConvert_2_by_PARUS.pgsql – сценарий замещения "пустых" объектов кодом psql:

- Редактировать _AfterConvert_2_by_PARUS.ini – подставить в "APP.APPCODE in (...)" список кодов приложений из лицензии или Oracle БД (SELECT APPCODE FROM APPLIST;).
- Выполнить _AfterConvert_2_by_PARUS.cmd.
- На выходе – Сценарий 2: _AfterConvert_2_by_PARUS.pgsql.

Важно! Утилита oreader.exe генерирует сценарии в кодировке win1251.

При необходимости для перекодирования в UTF-8:

Windows:

воспользоваться каким-либо сторонним перекодировщиком или редактором, например, Notepad++

Linux:

```
iconv -f CP1251 -t utf-8 _AfterConvert_1_by_POSTGRES.pgsql -o _AfterConvert_1_by_POSTGRES_utf8.pgsql
```

Выполнение сценариев _AfterConvert

При "ручном" выполнении сценариев лучше воспользоваться сторонним приложением, например, pgAdmin, чтобы не зависеть от кодировки файлов.

Подключиться к БД соответствующим пользователем, открыть "Инструмент запросов", скопировать текст сценария в окно запросов и выполнить.

При выполнении из командной строки/консоли необходимо соответствие кодировки файлов и настроек локали. Например, в командной строке Windows – сменить текущую кодовую страницу (по умолчанию 866) и использовать файлы с кодировкой win1251:

```
chcp 1251
psql -h host -U parus -d demo
   \i _AfterConvert\_AfterConvert_1_by_POSTGRES.pgsql
   select pg_temp.DO_AFTER_CONVERT('parus','parus_public','parus');
   \q
psql -h host -U parus -d demo -f _AfterConvert\_AfterConvert_2_by_PARUS.pgsql
```

В консоли Linux (по умолчанию utf8) нужно использовать файлы …_utf8.pgsql:

При выполнении сценариев сообщения типа "ЗАМЕЧАНИЕ:" можно игнорировать.

Работа с БД PostgreSQL

Подготовка

Примечания:

- Пользователь создается в кластере, а в не отдельной базе (как временное решение в PostgreSQL реализована возможность "привязки" пользователя к базе, если создавать его в формате "имя_пользователя@база_данных" и включить параметр "db_user_namespace=on").
- В PostgreSQL нет пользователей и групп как таковых, есть роли.
- Экспорт/Импорт пользователей из кластера в кластер:

  ```
  pg_dumpall -g > users.psql
  psql -U postgres -f users.psql
  ```

В состав конвертера входит коннектор (PPCConnect.dll) для взаимодействия win-клиента ПП "ПАРУС-Бюджет 8" с БД (вместо oci.dll). Он работает аналогично инстант-клиенту Oracle, поэтому необходимо задать пользовательскую или системную переменную окружения **NLS_LANG**=AMERICAN_AMERICA.CL8MSWIN1251.

Версия коннектора должна совпадать с версией конвертера, которым "сделана" БД.

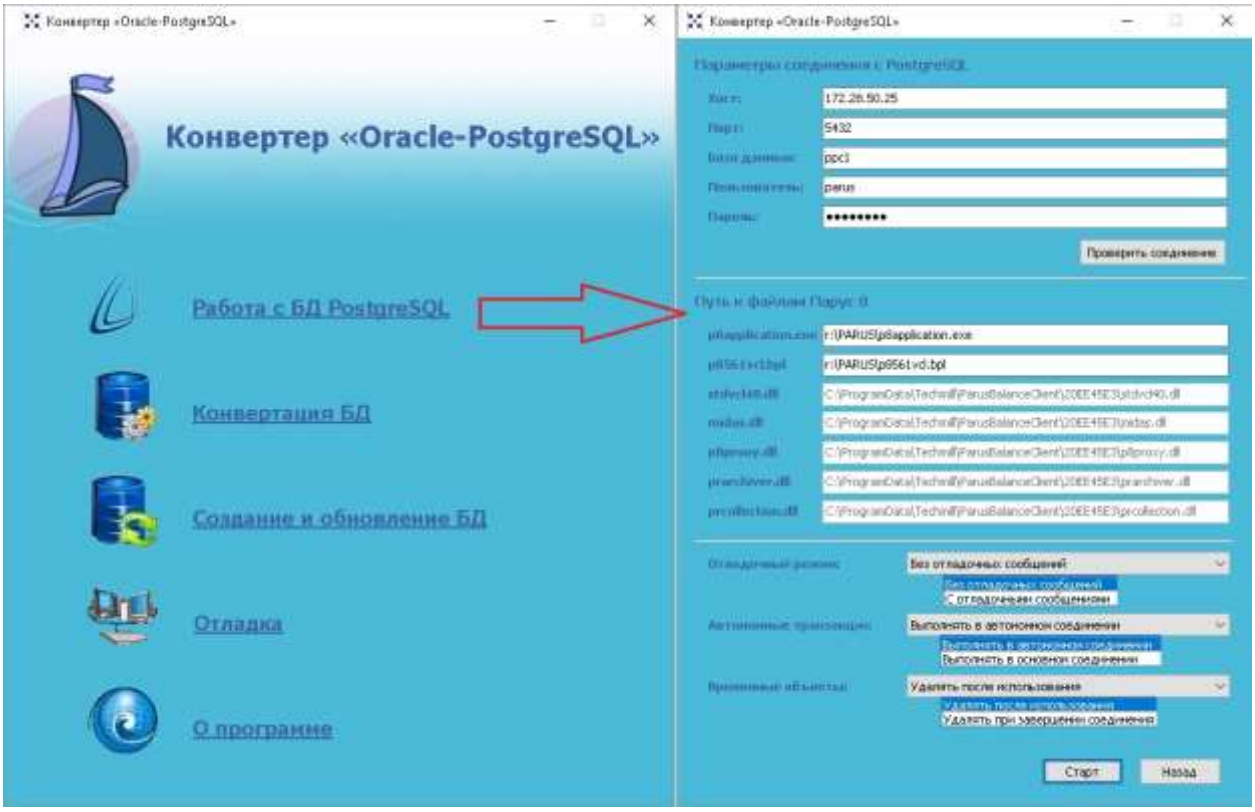
На компьютере (для первичного запуска), должны быть установлены:

- Конвертер (достаточно версии x86).
- Клиент ПП "ПАРУС-Бюджет 8" редакции не выше редакции ПП в БД (клиент поддерживает автоматическое обновление).

Работа

Запустить оболочку PPCShell.exe и выбрать пункт "Работа с БД PostgreSQL".

Указать в "Параметрах соединения" адрес (из этих данных будет сформирован файл tnsnames.ora) и пользователя. Поля "Путь к файлам Парус 8", при установленном клиенте ПП "ПАРУС-Бюджет 8" будут заполнены автоматически (можно менять).



После первого запуска в каталоге "%LOCALAPPDATA%\PPC" будет скомпонован клиент, который можно использовать в дальнейшем автономно, без конвертера, например, на другой машине. Если на ней ранее не был установлен Win-клиент ПП "ПАРУС-Бюджет 8", достаточно задать переменную NLS_LANG, установить вручную MSXML 4.0 Service Pack 3 и при необходимости изменить адрес в файле tnsnames.ora.

См. "Ключи запуска конвертера и оболочки".

Парус-Онлайн

Для работы через веб-интерфейс необходима установка приложения "**Парус-Онлайн 2**" (см. инструкции "СИС_Онлайн_Установка_Web2.pdf").

У анонимного пользователя должен быть задан параметр search_path:

```
ALTER ROLE parus_web IN DATABASE demo SET search_path = parus, public;
```

Единственное отличие в настройке от Oracle – указать тип соединения kind="postgre" в \Config\auth.config (по умолчанию используется kind="oracle").

```
<connection name="PG" connectionString="host=172.28.50.1;port=5432;database=pg1101;user id=parus_web;password=parus_web;unicode=true;pooling=true;max pool size=20;min pool size=20;Default Command Timeout=0" kind="postgre"/>
```

Рекомендуется выставлять значения min pool size по максимальному значению (равному max pool size).

Для работы сервисов необходимы фоновые пользователи, у которых также необходимо задать параметр search_path:

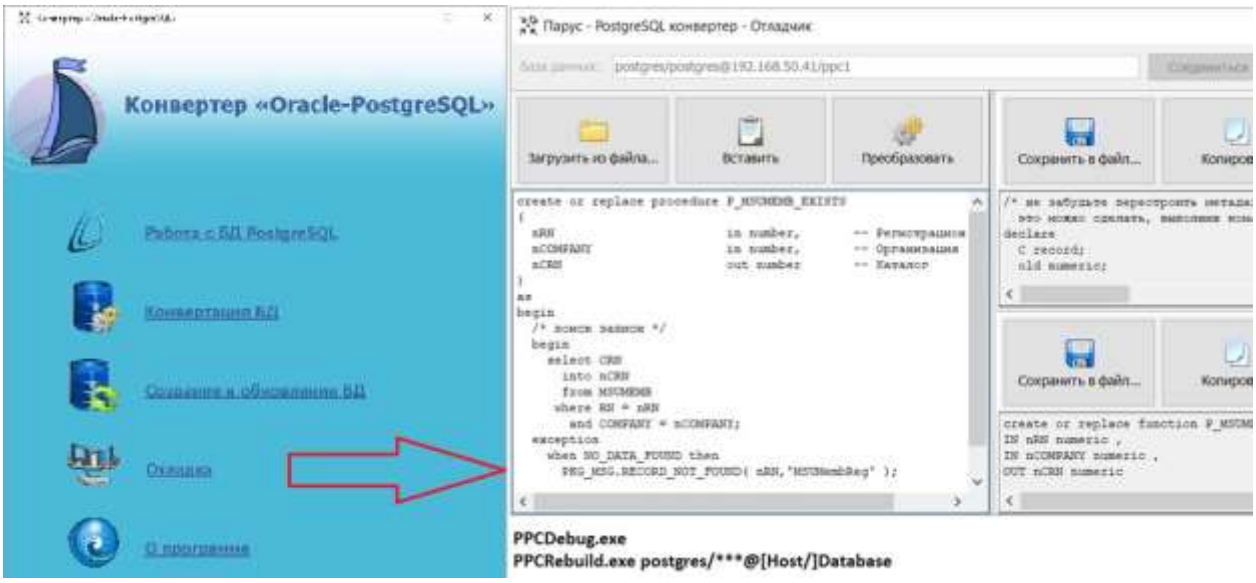
```
ALTER ROLE parus_rpt IN DATABASE demo SET search_path = parus, public;
```

При описании соединения с БД пул не используется – один экземпляр каждого сервиса использует одно соединение:

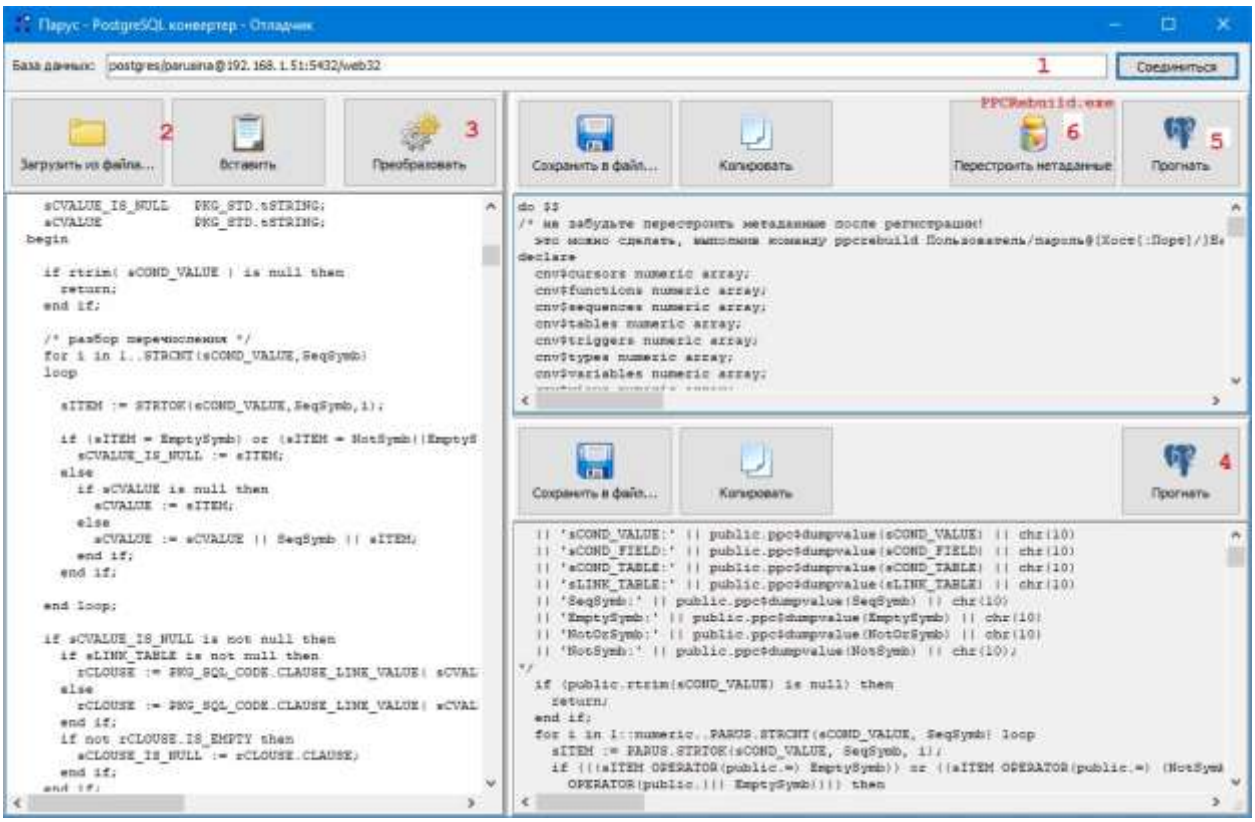
```
"ConnectionString": "host=127.0.0.1;port=5432;database=demo;user id=parus_rpt;password=parus_rpt;unicode=true; pooling=false;Default Command Timeout=0"
```

Приложения

Конвертация отдельных объектов (отладчик)



1. Соединение с БД.
2. Вставка/Загрузка исходного текста объекта (из Oracle).
3. Конвертация.
4. Создать новый объект (объекты) в целевой PostgreSQL-базе.
5. Регистрировать новый объект (объекты).
6. Выполнить ppcrebuild.



Ограничения PL/SQL кода и данных

Проверки на несовместимость кода и данных может быть выполнена в разделе "Стерилизация базы данных".

Данные и параметры функций

Не поддерживаются:

Некоторые типы данных, не используемые в ПП "ПАРУС-Бюджет 8", например, FLOAT.
Проверка:
`select DISTINCT DATA_TYPE from USER_TAB_COLUMNS`
`WHERE DATA_TYPE not in ('BLOB', 'CHAR', 'CLOB', 'DATE', 'LONG', 'LONG RAW', 'NUMBER', 'RAW', 'TIMESTAMP(6)', 'TIMESTAMP(6) WITH LOCAL TIME ZONE', 'VARCHAR2');`

Использование зарезервированных или ключевых слов в Oracle и/или в PostgreSQL в качестве пользовательских идентификаторов. Например, имя колонки "DATE", пользователь базы данных "ALL". По стандарту такие идентификаторы обычно экранируются кавычками, а это не принято в коде ПП "ПАРУС-Бюджет 8".

Несоответствие данных объявленному типу. Например, в таблице FILELINKS (Присоединенные документы) в "Текстовом файле" хранится "Двоичный тип содержимого" (в поле CLOB хранятся двоичные данные).
Ошибка: Data error – Truncating zero-symbol in string.
Поиск в таблице FILELINKS CLOB-ов, содержащих NULL:
`select CODE, RN, cast(substr(cdata,1,10) as varchar2(10))`
`from FILELINKS`
`where cdata is not null and instr(cdata,chr(0)) <> 0;`

Не ANSI-строки в текстовых полях (varchar). Как правило, такие данные попадают через сторонние приложения, например, значения PASSWORD и LOGIN для web-приложения "Парус-Консультант" в таблицах clnclients и clnpersons.
`update parus.clnclients set "PASSWORD" = null;`
`update parus.clnpersons set "PASSWORD" = null, "LOGIN" = null;`

Даты по условиям: меньше "01/01/0100 12:00:00.000 AM" и больше "12/31/9999 11:59:59.999 PM".
Будет ошибка, например: *Invalid date/time value 10-7-12 0:0:0*
Для поиска записей можно выполнить блок (данные сохраняются в таблице trace_table):

```
declare
  sSQL  PKG_STD.tSQL;
begin
  for rec in
    (select TABLE_NAME, COLUMN_NAME from user_tab_columns where DATA_TYPE =
'DATE' order by TABLE_NAME, COLUMN_NAME)
  loop
    sSQL := 'begin '
      || ' for rec1 in (select "' || rec.Column_Name  || '" WW from ' || rec.TABLE_NAME
      || ' where "' || rec.Column_Name || '" is not null and  "' || rec.Column_Name
      || '" < TO_DATE(''01.01.0100'', ''DD.MM.YYYY'')) loop PKG_TRACE.REGISTER('' ||
rec.TABLE_NAME
      || '', ''|| rec.COLUMN_NAME || '', d2s(rec1.WW));  end loop; end;';
begin
   execute immediate sSQL;
exception
  when others then
    PKG_TRACE.REGISTER(sSQL);
end;
  end loop;
end;
```

Просмотр данных: select DATA, data1, DATA2 from trace_table order by rn desc;
(Удалить данные: delete from trace_table;)

Далее необходимо исправить записи, добавить, например, 2000 лет
Пример для таблицы DOCINPT поля IN_DATE и TO_DATE:
alter TABLE DOCINPT disable all triggers;
update DOCINPT set IN_DATE = add_months(IN_DATE, 12 * 2000) where IN_DATE < TO_DATE('01.01.2000', 'DD.MM.YYYY');
alter TABLE DOCINPT enable all triggers;

Не переносятся значения паролей, хранимые в виде хэша.

Это касается паролей пользователей Oracle и пароли веб-доступа.

Функции

Оператор GOTO не поддерживается

Псевдостолбец "rownum" поддерживается только для оператора SELECT

Условная компиляция не поддерживаются
Нужно вычистить директивы условной компиляции ($IF not DBMS_DB_VERSION.ver_le_10 $THEN).

Поддержка join – без указания типа трактуется как inner join
… from acatalog A join companies c on A.company = c.rn
=>
… from acatalog A **inner** join companies c on A.company = c.rn

Использование в конструкциях NVL и COALESCE параметров разных типов:
SELECT last_name, NVL(**TO_CHAR(commission_pct), 0**) …

SELECT last_name, NVL(**TO_CHAR(commission_pct), 'Not Applicable'**) …

"Короткие замыкания"
Exception: Short circuit with type declaration "t_cursor" is not allowed.

```
function Get_Head_DocsBank_853_854_SP
   …
```

```
   v_num_ver VARCHAR2(1000);
   …
/* здесь можно декларировать новые процедуры, они будут видеть переменную v_num_ver. Это
"обычное замыкание". Количество уровней вложенности не ограничено. */
   begin
    if psBUDG_LEVEL is null or length (psBUDG_LEVEL) > 1 then raise_application_error(-20102,
'Неверно заполнено поле "Уровень бюджета"'); end if;
   …
    declare
     v_id NUMBER(17);
    …
      procedure set_serial ...
/* а здесь уже нет! Это "короткое замыкание". Такое не поддерживается, будет ошибка */
    …
    begin

    end;
   …
   end;
```

RANGE PRECEDING поддерживается только с UNBOUNDED

Системные объекты Oracle

Поддерживаются в объеме, необходимом для конвертации и, если они используются в объектах ПП "ПАРУС-Бюджет 8". Поэтому у владельца схемы (обычно PARUS) должны отсутствовать права DBA.

Системные представления Oracle разделены на следующие группы:

1. "Ненужные". Например, "v_$temporary_lobs". Они сконвертированы в "пустое представление" чтобы работали ссылки типа v_$temporary_lobs%rowtype или v_$temporary_lobs.nocache_lobs%type.

   Функционал, который что-то делает с временными BLOB-ми, найдет это представление пустым.

2. Метаданные. Например, "all_tables". Они конвертированы как представления над метаданными конвертера (ppccnvtables). Работают ссылки типа и код, который решит сначала проверить наличие таблицы, например, "AGNLIST", а потом сделать что-либо.

3. Значимые представления, (например, "gv_$nls_valid_values" или "gv_$session") – эмулируются, т.к. в ПП "ПАРУС-Бюджет 8" есть объекты, зависимые от них и использующие их данные.

4. Алиасы и псевдоалиасы.

   CREATE OR REPLACE VIEW "v_$session" AS SELECT * FROM "gv_$session";

   или

   CREATE OR REPLACE VIEW user_tables AS SELECT * from "all_tables";

Инсталлятор для PostgreSQL. Создание и обновление БД

Инсталлятор ParusPG.exe – предназначен для создания новой БД и обновления имеющейся (аналогичен инсталлятору ParusBUDGET.msi для Oracle).

Также инсталлятор используется для подготовки сконвертированной базы данных для работы с ПП "ПАРУС-Бюджет 8" (выполнить "обновление" вместо скриптов _AfterConvert_2) (подробнее см. главу "После конвертации").

Процесс конвертации описан в части "Конвертация Oracle-PostgreSQL".

Установка и подготовка самой СУБД PostgreSQL для работы с ПП "ПАРУС-Бюджет 8" описаны в части "Установка PostgreSQL".

Установка инсталлятора

Системные требования

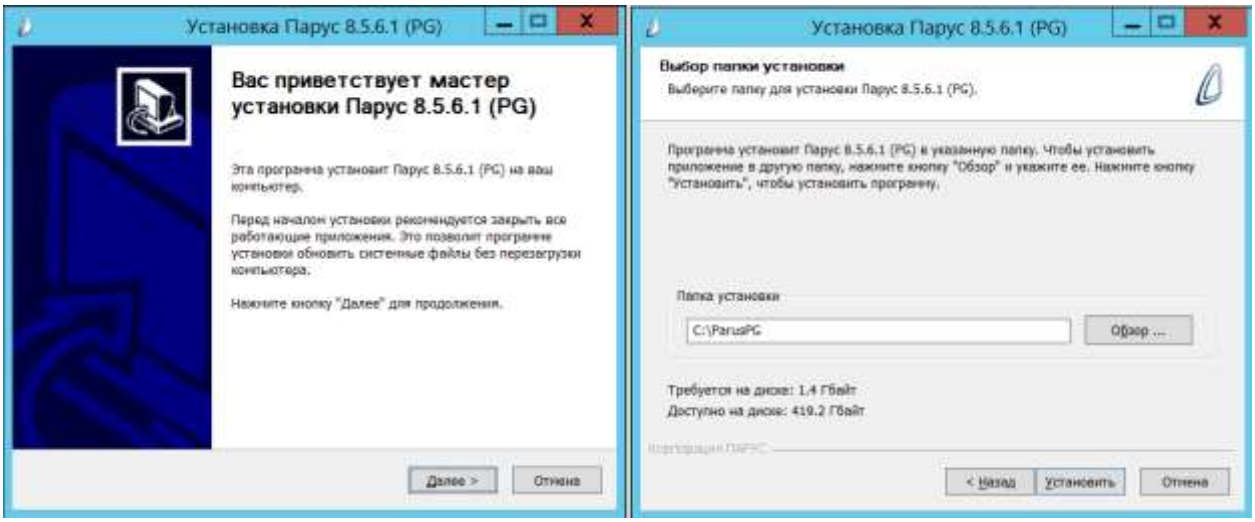
- Операционная система: MS Windows 7 и выше (рекомендуются редакции x64) или Windows Server 2008 R2 и выше, Microsoft .NET 4.5.
- Операционная система Linux, Mono 5 и выше (с WinForms при наличии GUI).

Все остальные требуемые компоненты включены в состав инсталлятора.

Поддерживается работа в консольном режиме и через графический интерфейс пользователя.

Установка на Windows

Запустить инсталлятор ParusPG.exe, указать путь установки, например C:\ParusPG.



При инсталляции не создаются ярлыки, пункты главного меню и записи в базе MS Installer. Для удаления инсталлятора достаточно удалить папку с содержимым. Для обновления – установить в ту же папку (рекомендуется предварительно удалить старые файлы, кроме Config).

Установка на Linux

Установить mono версии 5.x или выше. Рекомендуется установка из дистрибутива или, для установки более новой версии – из репозиториев проекта Mono.

Установить (если не установлен) архиватор 7-zip.

Создать целевой каталог для инсталлятора. Можно установить инсталлятор только для текущего пользователя – тогда каталог создать в домашнем каталоге пользователя, или для всех пользователей – в этом случае обычно используется каталог "/opt" (каталог для приложений, не поддерживающих структуру файловой системы Linux).

Например, для Debian-based дистрибутивов (Debian/Ubuntu/Астра):

```
sudo apt install mono-complete
sudo apt install p7zip-full
sudo mkdir /opt/ParusPG
sudo chown <user>:<group> /opt/ParusPG
# или mkdir ~/ParusPG
7z x -o/opt/ParusPG /tmp/ParusPG.exe
cd /opt/ParusPG
```

Примечание. Вместо распаковки 7z-архиватором можно скопировать каталог с установленным приложением из Windows-системы.

Для запуска инсталлятора в графическом режиме выполняется команда:

```
mono PostgresUpdaterUi.exe
```

Для запуска инсталлятора в консольном режиме (требуется рабочий конфигурационный файл PostgresUpdaterUi.exe.Config, размещенный рядом с PostgresUpdaterUi.exe):

```
mono PostgresUpdaterUi.exe console ParusPassword=parus PostgresPassword=passwd
```

Подготовка БД к установке ПП "Парус-Бюджет 8"

Предполагается, что уже выполнены все действия по подготовке, а именно:

- Установлен PostgreSQL, инициализирован кластер.
- Настроены параметры кластера (доступ в файле pg_hba.conf, параметры в postgresql.conf).
- Установлены расширения, не входящие в стандартную поставку PostgreSQL: http, pg_variables, pgzip, pgqrcode.

Рекомендуется следовать описанной ниже последовательности действий.

Подключиться к PostgreSQL под суперпользователем, в качестве базы можно использовать любую (обычно – служебная "postgres").

Создать пользователя-владельца объектов ПАРУС-Бюджет 8 (обычно, "parus").

Привилегия "superuser" пользователю "parus" дается на время создания БД (конвертации или импорта).

Создать групповую роль (имя может быть любое).

Включить пользователя "parus" в групповую роль "parus_public".

```
psql -U postgres -d postgres ...
CREATE ROLE parus SUPERUSER LOGIN PASSWORD 'parusina' INHERIT;
CREATE ROLE parus_public NOLOGIN NOINHERIT;
GRANT parus_public TO parus;
```

Если пользователь уже существует, дать ему права суперпользователя:

ALTER ROLE **parus SUPERUSER**;

Подключиться к PostgreSQL под пользователем "parus", в качестве базы можно использовать любую (обычно – служебная "postgres").

Создать базу данных, например "demo". Подробнее см. часть "Установка PostgreSQL".

```
psql -U parus -d postgres ...
-- Linux
CREATE DATABASE demo TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE = 'C' LC_CTYPE = 'ru_RU.CP1251';
-- Windows
CREATE DATABASE demo TEMPLATE template0 ENCODING = 'WIN1251' LC_COLLATE = 'Russian_Russia.1251'
LC_CTYPE = 'Russian_Russia.1251';
```

Подключиться к созданной БД под пользователем "parus".

Зарегистрировать расширения (объекты расширений будут установлены в схему "public").

```
psql -U parus -d demo ...
CREATE EXTENSION IF NOT EXISTS "uuid-ossp";
CREATE EXTENSION IF NOT EXISTS "hstore";
CREATE EXTENSION IF NOT EXISTS "dblink";
CREATE EXTENSION IF NOT EXISTS "xml2";
CREATE EXTENSION IF NOT EXISTS "pg_variables";
CREATE EXTENSION IF NOT EXISTS "http";
CREATE EXTENSION IF NOT EXISTS "pgqrcode";
CREATE EXTENSION IF NOT EXISTS "pgzip";
```

Привилегии для схемы "public".

Для схемы "public" забрать привилегии у групповой роли "public" и выдать соответствующие привилегии пользователю "parus" и групповой роли "parus_public".

```
REVOKE ALL ON SCHEMA public FROM public;
GRANT USAGE ON SCHEMA public TO parus;
GRANT USAGE ON SCHEMA public TO parus_public;
```

Создать системную схему "sys" и выдать соответствующие привилегии пользователю "parus" и групповой роли "parus_public".

```
CREATE SCHEMA sys;
GRANT ALL ON SCHEMA sys TO parus;
GRANT USAGE ON SCHEMA sys TO parus_public;
```

Создать прикладную схему "parus" (имя может быть любое) и выдать соответствующие привилегии пользователю "parus" и групповой роли "parus_public".

```
CREATE SCHEMA parus;
GRANT ALL ON SCHEMA parus TO parus;
GRANT USAGE ON SCHEMA parus TO parus_public;
```

Для пользователя "parus" установить путь поиска объектов (см. главу "Схемы и search_path").

```
ALTER ROLE parus IN DATABASE demo SET search_path = parus, sys;
SET search_path = parus, sys;
```

Команда "SET search_path" устанавливает параметр в текущем сеансе (в противном случае необходимо перелогиниться).

Создать сервис автономных транзакций (см. главу "Сервис автономных транзакций").

```
DROP FOREIGN DATA WRAPPER IF EXISTS parus_autonomous_transaction_service_fdw CASCADE;
CREATE FOREIGN DATA WRAPPER parus_autonomous_transaction_service_fdw;
CREATE SERVER parus_autonomous_transaction_service_server
  FOREIGN DATA WRAPPER parus_autonomous_transaction_service_fdw
  OPTIONS(host '127.0.0.1', port '5432', dbname 'demo');
CREATE USER MAPPING FOR parus
  SERVER parus_autonomous_transaction_service_server
  OPTIONS(USER 'parus',password 'passwd');
```

Выполнить сценарий "superuser_before.sql" из поставки инсталлятора.

```
psql -U parus -d demo ... -f "Scripts/superuser_before.sql"
```

Забрать привилегию "superuser" у пользователя "parus".

```
ALTER ROLE parus NOSUPERUSER;
```

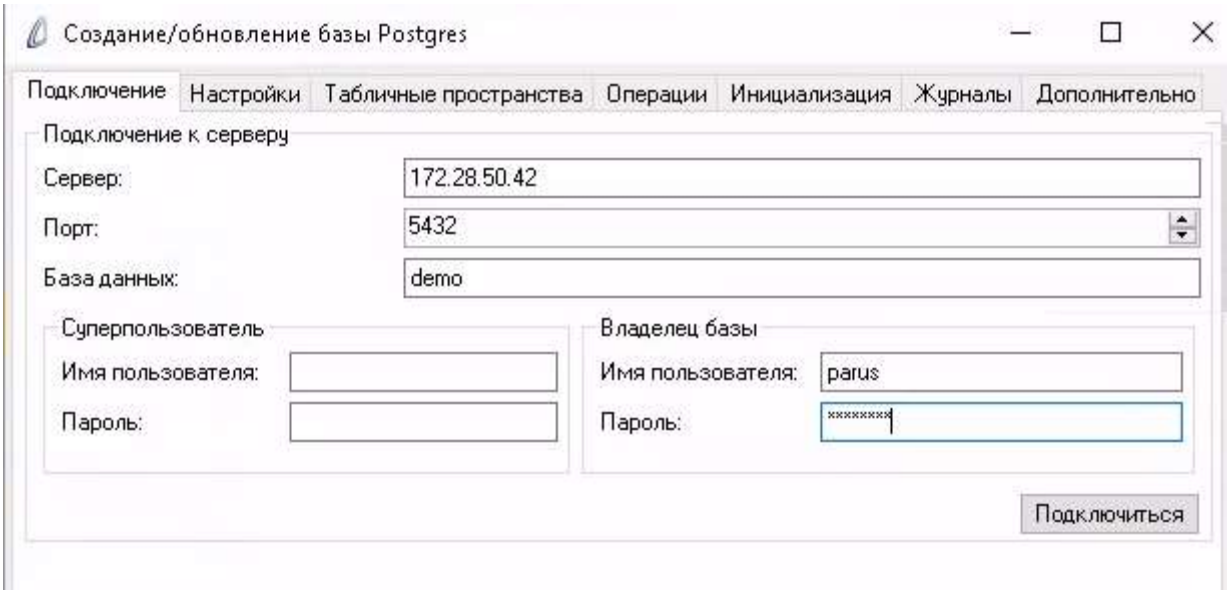
Если отсутствуют права суперпользователя (например, при установке у облачного провайдера), то необходимо предоставить перечисленные выше команды администратору сервера СУБД, например, в виде скрипта.

Выполнение создания/обновления

Запустите инсталлятор PostgresUpdaterUi.exe.

Обязательные для заполнения поля будут выделены желтым цветом.

Если рядом с PostgresUpdaterUi.exe имеется конфигурационный файл PostgresUpdaterUi.exe.Config, то из него будут считаны значения параметров.



Подключение к серверу:

- Сервер, порт – адрес кластера PostgreSQL. Настройка удаленного и локального доступа к кластеру и базам данных выполняется в файлах конфигурации (pg_hba.conf и postgresql.conf).
- База данных – целевая база данных в кодировке WIN1251.
- Суперпользователь (обычно "postgres"). Можно не задавать, если были выполнены все описанные в разделе "Подготовка БД к установке".
- Владелец базы ПП "ПАРУС-Бюджет 8" (обычно "parus").

При заполнении перечисленных выше полей становится доступной кнопка "Подключиться".

После подключения становятся доступны остальные закладки ("Настройки", "Табличные пространства" и т.д.).

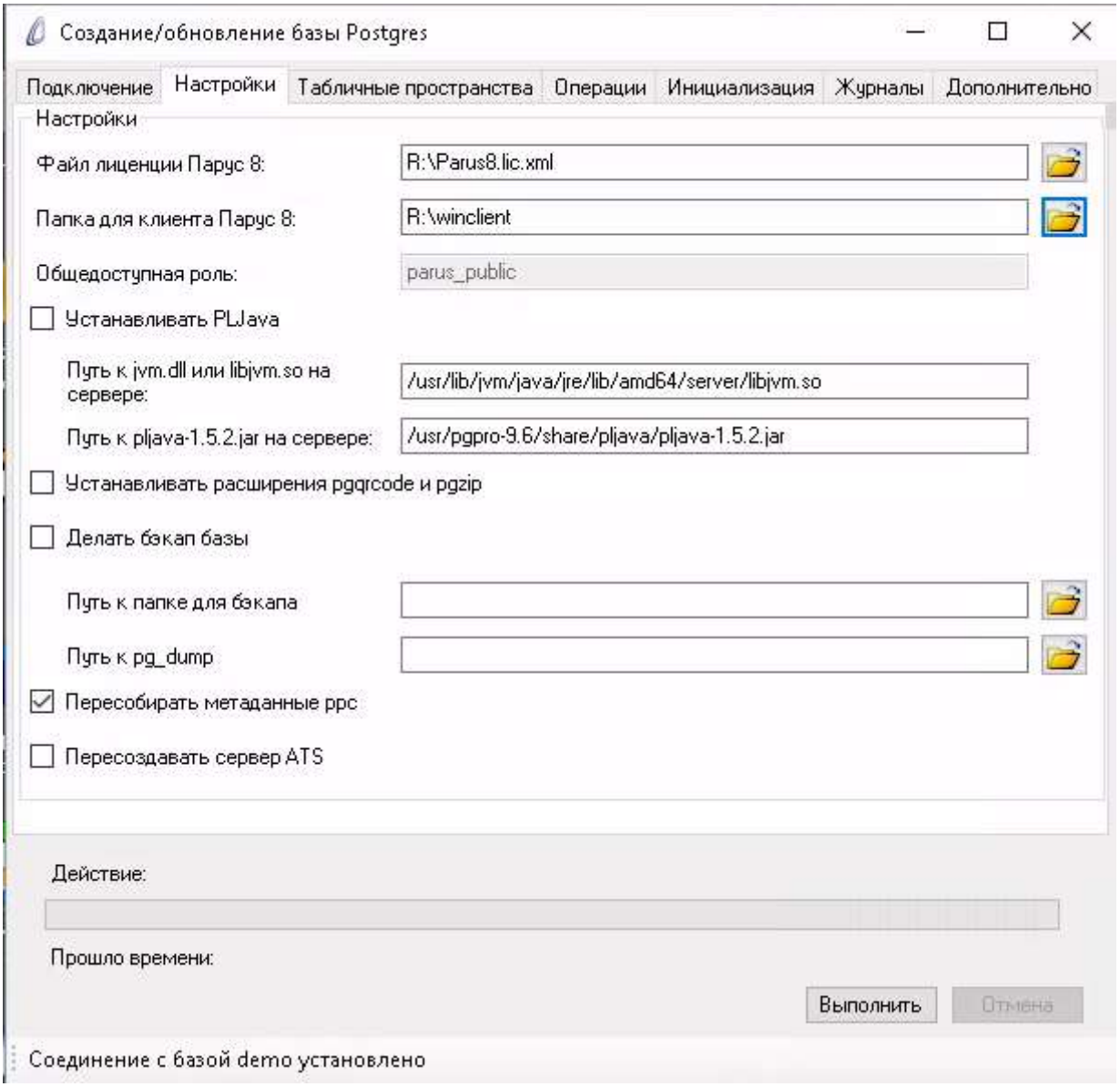
На вкладке "Настройки":

- **Файл лицензий** – поле обязательно при создании базы. При обновлении – опция указывается, если нужно обновить лицензию.
- **Папка для клиента** ПП "ПАРУС-Бюджет 8" – место размещения скомпонованного win-клиента для работы с БД (приложение p8application.exe со своими библиотеками; коннектор, соответствующий версии БД; файл настроек соединения с БД tnsnames.ora).

  Важно! Коннектор работает как Oracle Instant Client, поэтому на клиентском рабочем месте ОБЯЗАТЕЛЬНО при использовании win-клиента нужно задавать переменную окружения NLS_LANG=AMERICAN_AMERICA.CL8MSWIN1251.
- **Общедоступная роль** – имя общедоступной роли ("parus_public"), если роль существует, то она будет определена автоматически.
- **Устанавливать PLJava** – Устаревшая опция настройки расширения pljava. Присутствует для совместимости с ранними версиями (до появления расширений pgzip и pgqrcode) и возможности поддержки пользовательского кода, написанного на java.
- **Устанавливать расширения pgzip и pgqrcode** – выполнить команду "CREATE EXTENSION" для этих расширений.

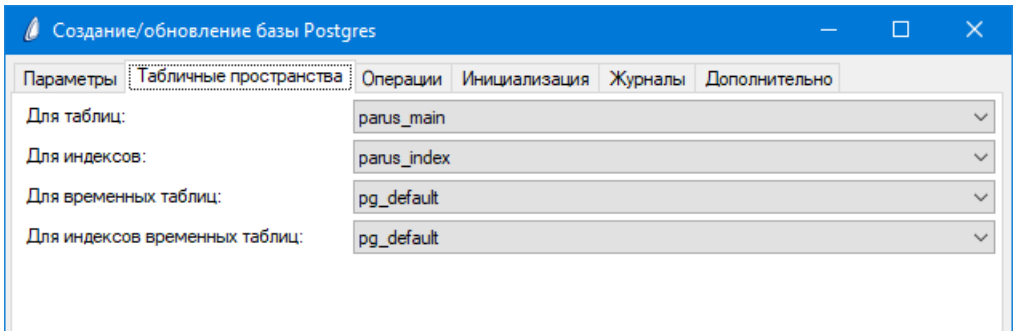
  Важно! Никакие библиотеки, скрипты и управляемые файлы на сервер БД не устанавливаются! Только регистрация в БД.
- **Делать бэкап базы** – создать резервную копию (локальный дамп) БД с помощью утилиты pg_dump перед выполнением обновления (должен быть установлен PostgreSQL-клиент).
- **Пересобирать метаданные ppc** – аналог действия "Пересборка метаданных" на вкладке "Дополнительно" (см. ниже), но выполняется не в ручном режиме, а после создания/обновления.

- **Пересоздать сервер ATS** – см. главу "Сервис автономных транзакций".

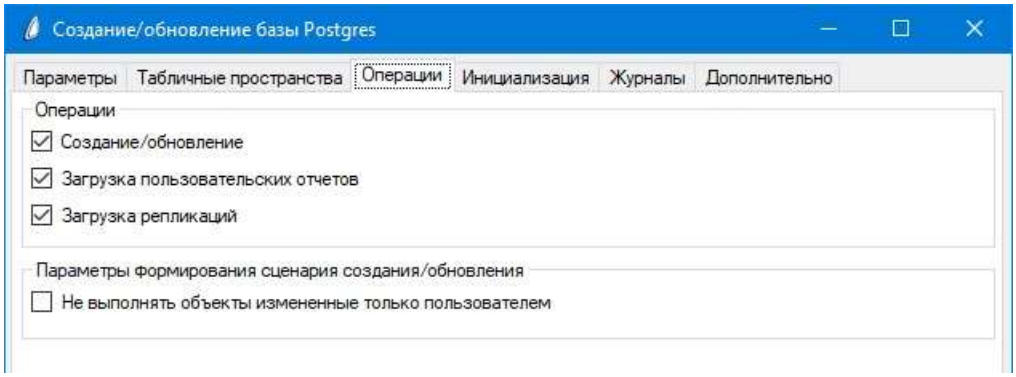


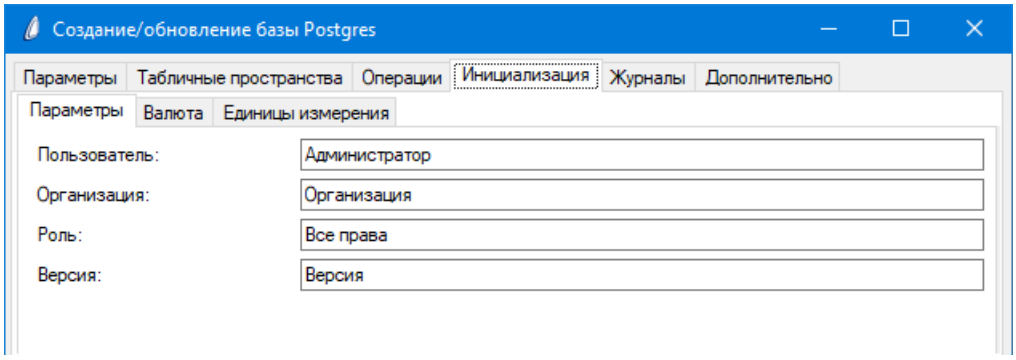
- **Табличные пространства** – для отдельного размещения различных типов объектов ПП "ПАРУС-Бюджет 8" можно выбрать пространства, отличные от пространства по умолчанию (pg_default).

  Важно! Использование пользовательских пространств (с целью увеличения производительности) имеет смысл только при их размещении на физическом диске, отличном от диска, на котором размещено пространство pg_default.



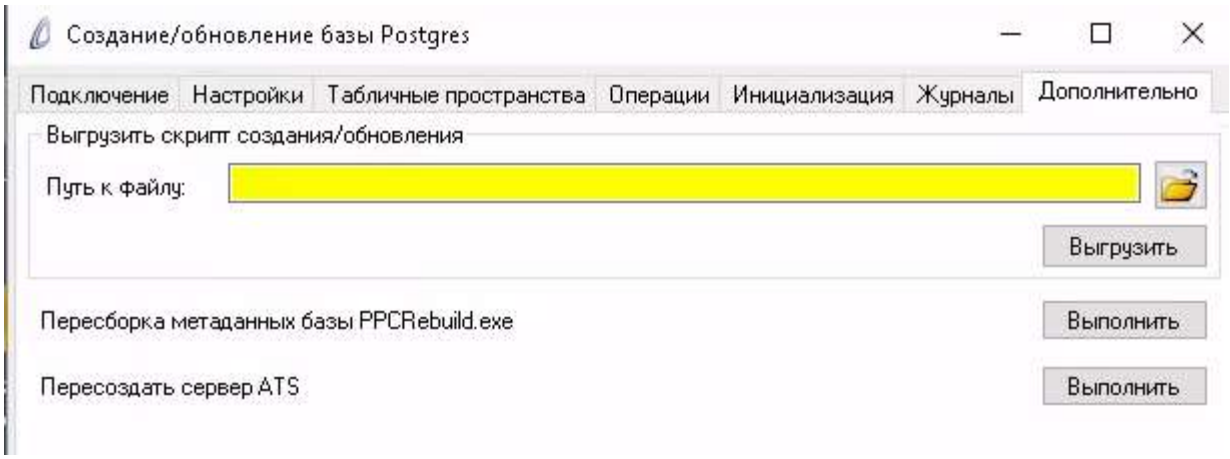
Другие параметры аналогичны созданию/обновлению в СУБД Oracle:





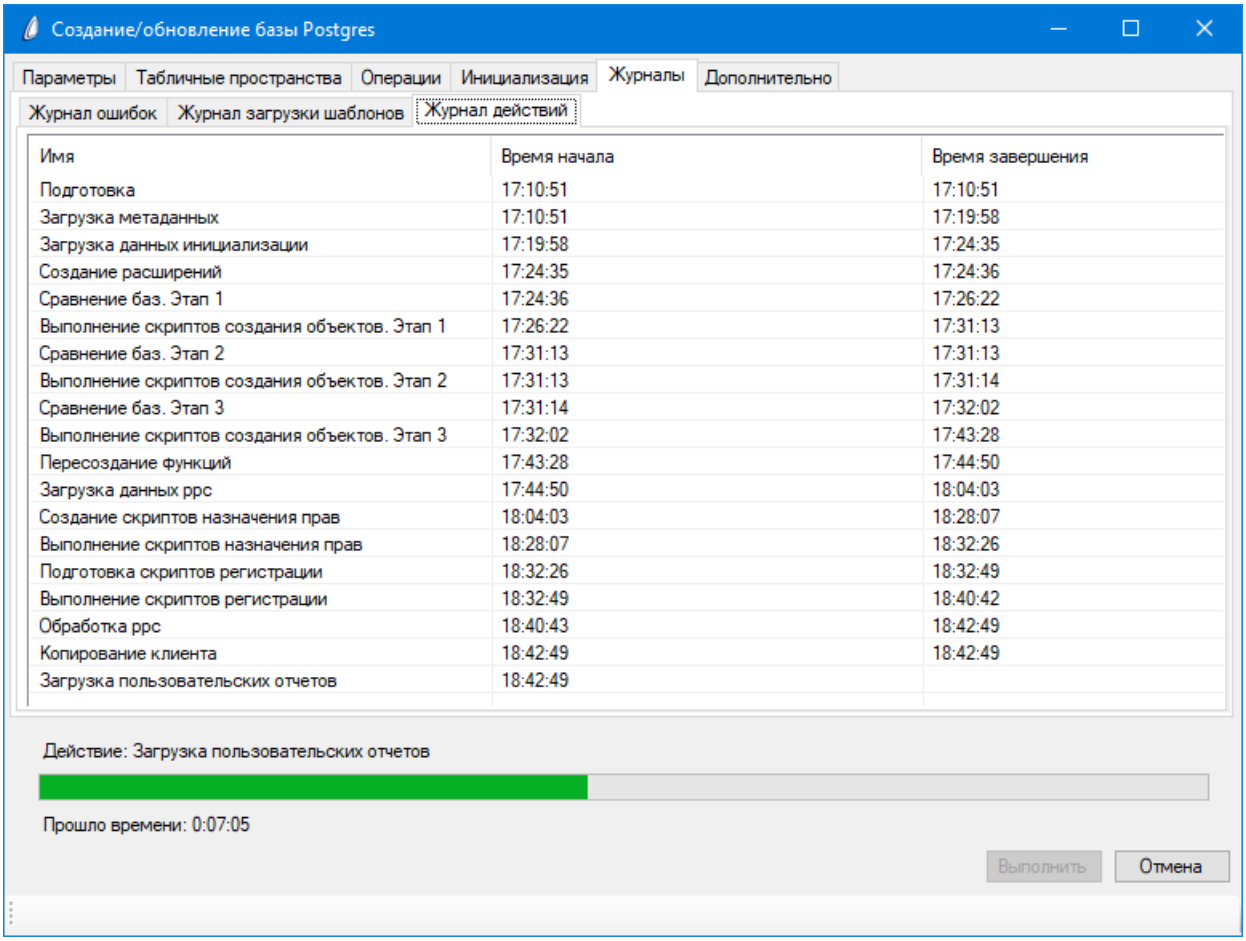
Опции на вкладке "Дополнительно":

- "Выгрузить скрипт" – можно после создания/обновления "Выгрузить" в указанный файл сценарий выполнения.
- "Пересборка метаданных" – требуется после выполнения ОБНОВЛЕНИЯ, если ранее, при конвертации, из Oracle-базы были перенесены **пользовательские объекты** (информация о них хранится в метаданных конвертера, которые заменяются при обновлении, пере-сборка их "восстанавливает"). Сейчас это действие можно задать при выполнении обновления на вкладке "Настройки" – "Пересобирать метаданные ppc".
- Пересоздать сервер ATS – см. главу "Сервис автономных транзакций". Действие можно задать при выполнении обновления на вкладке “Настройки”.



После задания параметров – нажать кнопку "Выполнить" в главном окне.

За ходом выполнения удобнее наблюдать на вкладке "Журналы".



Ошибки, возникшие при создании/обновлении, регистрируются в журнале ошибок (см. соответствующую вкладку).

Работа в консольном режиме

Для автоматизации процесса обновления доступен консольный режим работы приложения.

Все настройки должны находиться в конфигурационном файле **PostgresUpdaterUi.exe.Config**, который необходимо подготовить отдельно перед выполнением создания/обновления.

Пример конфигурационного файла

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings file="">
    <clear />
    <add key="LicenseFilePath" value="" />
    <add key="ClientPath" value="R:\demo12" />
    <add key="NoRepl" value="False" />
    <add key="CheckReplications" value="True" />
    <add key="CheckUserReports" value="True" />
    <add key="CheckUpdate" value="True" />
    <add key="CheckBackup" value="False" />
    <add key="InitUser" value="Администратор" />
    <add key="InitOrg" value="Организация" />
    <add key="InitRole" value="Все права" />
    <add key="InitVersion" value="Версия" />
    <add key="InitCurCode" value="643" />
    <add key="InitCurIso" value="RUB" />
    <add key="InitCurName" value="Российские рубли" />
    <add key="InitCurBase1" value="рубль" />
    <add key="InitCurBase2" value="рубля" />
    <add key="InitCurBase10" value="рублей" />
    <add key="InitCurSub1" value="копейка" />
    <add key="InitCurSub2" value="копейки" />
    <add key="InitCurSub10" value="копеек" />
    <add key="UnlScript" value="" />
    <add key="TsTables" value="pg_default" />
    <add key="TextTsIndexes" value="pg_default" />
    <add key="TsTablesTemp" value="pg_default" />
    <add key="TsIndexesTemp" value="pg_default" />
    <add key="Server" value="172.28.3.190" />
    <add key="Port" value="5432" />
    <add key="Base" value="small" />
    <add key="ParusUser" value="parus" />
    <add key="PostgresUser" value="" />
    <add key="CheckPlJava" value="False" />
    <add key="JrePath" value="/usr/lib/jvm/java/jre/lib/amd64/server/libjvm.so" />
    <add key="PostgresPath" value="/usr/pgpro-9.6/share/pljava/pljava-1.5.2.jar" />
    <add key="BackupPath" value="" />
    <add key="PgdumpPath" value="" />
    <add key="MetadataRebuild" value="True" />
    <add key="InstallQrZip" value="False" />
    <add key="PublicRole" value="parus_public" />
  </appSettings>
</configuration>
```

Подготовленный файл копируется в каталог приложения.

При запуске указываются пароли суперпользователя и владельца схемы:

```
PostgresUpdaterUi.exe console ParusPassword=parus PostgresPassword=pass
```

Если требуется суперпользователь – необходимо добавить ключ PostgresPassword=pass

Для **Linux**:

```
mono PostgresUpdaterUi.exe console ParusPassword=parus PostgresPassword=pass
```