**Добавление нового пользователя**

В обозревателе объектов раскрываем вкладку «Безопасность» (Security), кликаем правой кнопкой мыши по вкладке «Имена входа» (Logins) и в контекстном меню выбираем «Создать имя входа…» (NewLogin…)

Откроется окно создания имени входа (Login — New). Теперь необходимо определиться с вариантом аутентификации нового пользователя. Возможны 2 варианта:

* Аутентификация с помощью пароля — Проверка подлинности SQL Server (SQL ServerAuthentication).
* Доступ для конкретного пользователя Windows — Проверка подлинности Windows (Windowsauthentication).

**Проверка подлинности SQL Server**

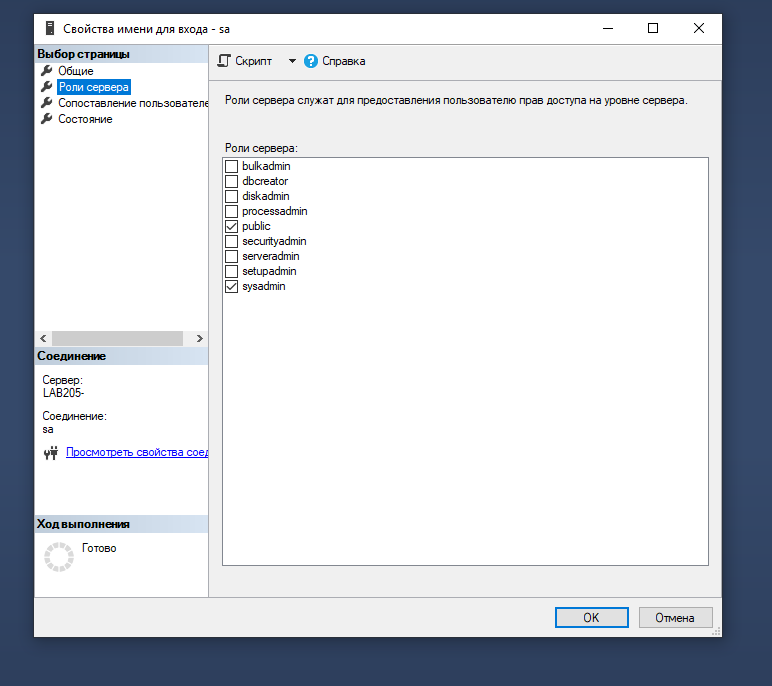
Для начала рассмотрим первый способ аутентификации. Например, создадим пользователя для работы сервера. Укажите имя входа (Loginname), выберете «Проверка подлинности SQL Server» (SQLServerAuthentication) и введите пароль (Password) пользователя (12345). Далее снимаем / отмечаем галочки у следующих параметров:

* Требовать использование политики паролей (Enforcepasswordpolicy)
* Задать срок окончания действия пароля (Enforcepasswordexpiration)
* Пользователь должен сменить пароль при следующем входе (Usermustchangepasswordatnextlogin)

Для данной задачи оставляем включенным только первый параметр.

Устанавливаем необходимые параметры и переходим на вкладку «Роли сервера» (ServerRoles).

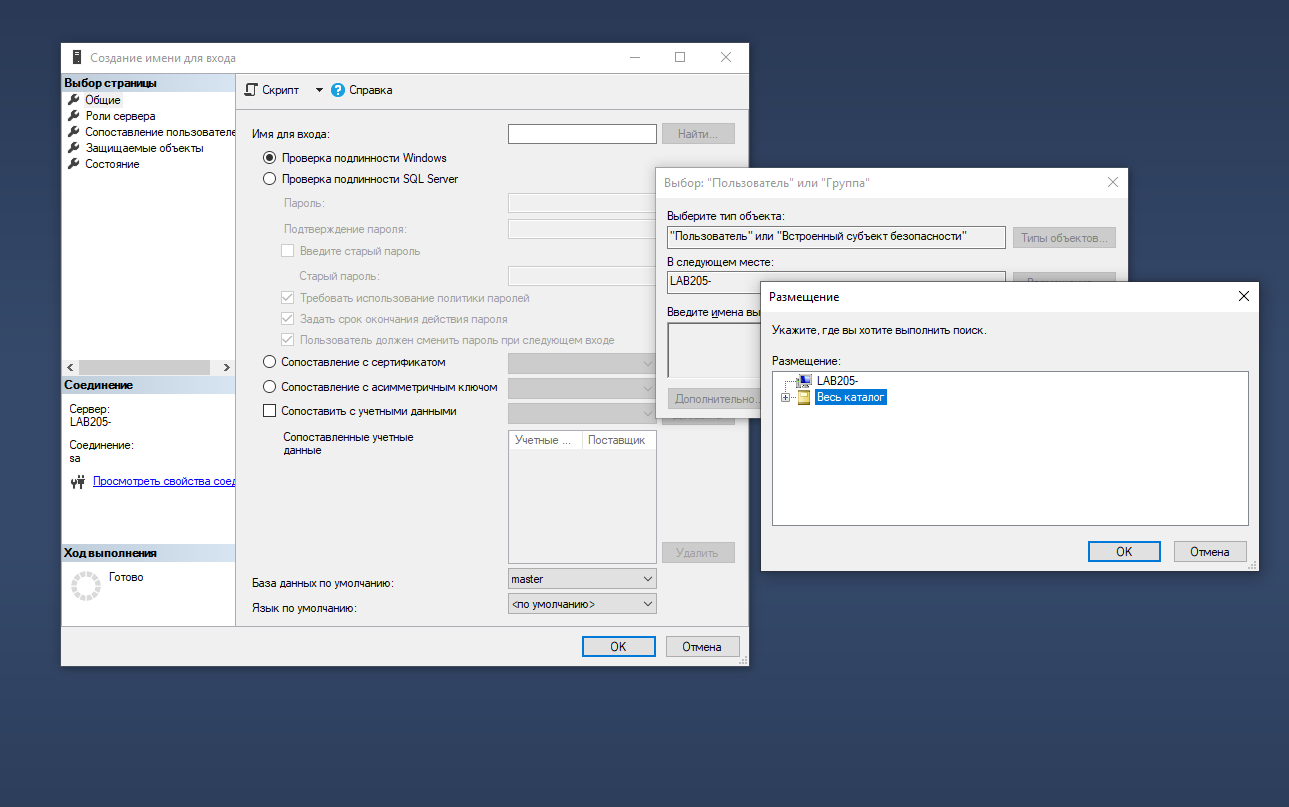
|  |  |
| --- | --- |
| **Предопределенная роль уровня сервера** | **Описание** |
| **sysadmin** | Члены предопределенной роли сервера **sysadmin** могут выполнять любые действия на сервере. |
| **serveradmin** | Члены предопределенной роли сервера **serveradmin** могут изменять параметры конфигурации на уровне сервера, а также выключать сервер. |
| **securityadmin** | Члены предопределенной роли сервера **securityadmin** управляют именами входа и их свойствами. Они могут предоставлять, запрещать и отменять разрешения на уровне сервера (инструкции GRANT, DENY и REVOKE). Они также могут предоставлять, запрещать и отменять разрешения на уровне базы данных (инструкции GRANT, DENY и REVOKE) при наличии доступа к базе данных. Кроме того, они могут сбрасывать пароли для имен входа SQL Server.   * *Примечание:* Возможность предоставления доступа к компоненте DatabaseEngine и настройки разрешений пользователей позволяет администратору безопасности назначать большинство разрешений сервера. Роль **securityadmin** должна считаться эквивалентной роли **sysadmin**. |
| **processadmin** | Члены предопределенной роли сервера **processadmin** могут завершать процессы, выполняемые на экземпляре SQL Server. |
| **setupadmin** | Члены предопределенной роли сервера **setupadmin** могут добавлять или удалять связанные серверы с помощью инструкций Transact-SQL. (При использовании ManagementStudio необходимо членство в роли **sysadmin**.) |
| **bulkadmin** | Члены предопределенной роли сервера **bulkadmin** могут выполнять инструкцию BULK INSERT. |
| **diskadmin** | Предопределенная роль сервера **diskadmin** используется для управления файлами на диске. |
| **dbcreator** | Члены предопределенной роли сервера **dbcreator** могут создавать, изменять, удалять и восстанавливать любые базы данных. |
| **public** | Каждое имя входа SQL Server принадлежит к роли сервера **public**. Если для участника на уровне сервера не были предоставлены или запрещены конкретные разрешения на защищаемый объект, то он наследует разрешения роли **public** на этот объект. Разрешения роли **public** следует назначать только тому объекту, который будет доступен всем пользователям. Нельзя изменить членство в роли **public**.   * *Примечание:* Роль **public** реализована не так, как другие роли. Однако разрешения для роли **public** могут быть назначены, запрещены или отозваны. |



После чего нажимаем «ОК» для сохранения выполненных действий.

**Проверка подлинности Windows**

Теперь добавим администратора SQL Server, выбрав его из текущих пользователей Windows. Для этого создайте нового пользователя и способ аутентификации укажите «Проверка подлинности Windows» (Windowsauthentication). Далее, чтобы ввести имя входа, нажмем «Найти» (Search…), затем «Дополнительно» (Advanced…), далее Размещение и выбираем весь каталог, в следующем окне «Поиск» (FindNow) и выбрав необходимого пользователя из списка, закроем все окна нажав на «ОК».



Устанавливаем необходимые параметры и переходим на вкладку «Роли сервера» (ServerRoles).

Перечень ролей БД:

Public – минимальные права доступа к БД (на просмотр)

Db\_owner– может выполнять любые действия с БД

Db\_accessadmin– добавляет и удаляет пользователей БД

Db\_sequrityadmin– управляет ролями в Бди разрешениями на запуск команд и работу с объектами БД

Db\_ddladmin– добавляет, изменяет и удаляет объекты БД

Db\_backupoperator– осуществляет резервное копирования БД

Db\_datareader– может просматривать все данные в каждой таблице в БД

Db\_datawriter- может добавлять, удалять и изменять данные в каждой таблице в БД

Db\_denydatareader– запрет на просмотр всех данных в каждой таблице в БД

Db\_denydatawriter- запрет на добавление, удаление и изменение всех данных в каждой таблице в БД

| **Название встроенной процедуры** | **Описание** |
| --- | --- |
| sp\_grantlogin | – позволяет использовать пользователей или группы ОС для соединения с Microsoft SQL Server™ , используя Windows Authentication.  Этот пример позволяет пользователю WindowsNTCorporate\BobJсоединяться сSQLServer.  EXEC sp\_grantlogin 'Corporate\BobJ' |
| sp\_defaultdb | Изменяет для пользователя БД по умолчанию  Этот пример настраивает БД по умолчанию pubs для пользователя Victoria.  EXEC sp\_defaultdb 'Victoria', 'pubs' |
| sp\_grantdbaccess | Добавляет учетную запись из раздела security в текущую БД, для учетных записей MicrosoftWindows также дает разрешение на доступ к текущей БД.  Синтаксис:  EXEC sp\_grantdbaccess [@loginame =] 'login'     [,[@name\_in\_db =] 'name\_in\_db' [OUTPUT]]  Этот пример добавляет учетную запись Corporate\GeorgeWв текущую БД и присваивает псевдоним внутри БДGeorgie.  EXEC sp\_grantdbaccess 'Corporate\GeorgeW', 'Georgie' |
| sp\_revokedbaccess | Удаляет информацию об учетной записи из текущей БД.  Синтаксис:  EXEC sp\_revokedbaccess [ @name\_in\_db = ] 'name'  Этот пример удаляет учетную запись Corporate\GeorgeWиз текущей БД.  EXEC sp\_revokedbaccess 'Corporate\GeorgeW' |
| sp\_addrole | Создает новую роль в текущей БД.  Этот пример создает новую роль в текущей БД с названием Managers.  EXEC sp\_addrole 'Managers' |
| sp\_addrolemember | В текущей БД назначает роль конкретному пользователю.  Пример A.  Этот пример добавляет учетную запись Corporate\JeffL из Windows NT в БД Sales как пользователя Jeff. Jeff затем получает роль Sales\_Managers в БД Sales.  USE Sales --сделать текущей БД Sales  GO –выполнить команду, а потом запустить следующую  EXEC sp\_grantdbaccess 'Corporate\JeffL', 'Jeff'  GO  EXEC sp\_addrolemember 'Sales\_Managers', 'Jeff'  Пример B.  Этот пример добавляет пользователя SQLServerс именемMichaelк ролиEngineeringв текущей БД.  EXEC sp\_addrolemember 'Engineering', 'Michael' |
| sp\_helprotect | Показывает список привилегий, ассоциированных с ролью. |
| sp\_helprolemember | Показывает список пользователей БД, входящих в указанную роль |
| sp\_addsrvrolemember | Присвоение встроенной серверной роли для существующей учетной записи  sp\_addsrvrolemember [ @loginame = ] 'login'   , [ @rolename = ] 'role'  Например:  sp\_addsrvrolemember 'Admin\_DB', 'sysadmin' |
| sp\_dropsrvrolemember | Удаление встроенной серверной роли для учетной записи или группы  sp\_dropsrvrolemember [ @loginame = ] 'login' , [ @rolename = ] 'role'  Например:  sp\_dropsrvrolemember 'Admin\_DB' , 'sysadmin' |
| sp\_helpsrvrole | Описание только встроенных ролей в SQL Server  sp\_helpsrvrole [ [ @srvrolename = ] 'role' ]  Например:  sp\_helpsrvrole 'sysadmin' |
| sp\_helpsrvrolemember | Возвращает список ролей и учетных записей, которым присвоены эти роли  sp\_helpsrvrolemember [ [ @srvrolename = ] 'role' ]  Например:  sp\_helpsrvrolemember 'sysadmin' |
| sp\_srvrolepermission | Возвращает список ролей и разрешений, которые присвоены этим ролям  sp\_srvrolepermission [[@srvrolename =] 'role']  Например:  sp\_srvrolepermission 'sysadmin' |
| sp\_addlogin  sp\_adduser | Создание новой учетной записи в SQL Server в разделе Sequrity:  sp\_addlogin [ @loginame = ] 'login'     [ , [ @passwd = ] 'password' ]     [ , [ @defdb = ] 'database' ]     [ , [ @deflanguage = ] 'language' ]     [ , [ @sid = ] sid ]     [ , [ @encryptopt = ] 'encryption\_option' ]  Например:  sp\_addlogin 'login1',sysname, 'DB\_Books'  Создает пользователя в SQL Server без PUBLIC в БД ' DB\_Books'.  Нужно еще использовать  sp\_adduser [ @loginame = ] 'login'     [ , [ @name\_in\_db = ] 'user' ]     [ , [ @grpname = ] 'group' ],  Пример:  Создана база данных DB\_Books. В ней создан пользователь Admin\_DB с серверной ролью sysadmin, с ролью в БД db\_owner.  Cоздать в QueryAnalyzer нового пользователя с именем Public\_ и паролем Public\_1 (пароль не должен совпадать с именем пользователя) с помощью следующих команд (не забудьте нажать F5 для запуска команд на выполнение):  Exec sp\_addlogin 'Public\_','Public\_1', 'DB\_Books'  use DB\_Books  Execsp\_adduser 'Public\_','Public\_'  В БД DB\_Books создан пользователь Public\_ с ролью в БД DB\_Books public. |
| Deny (отрицание) | Этот пример запрещает несколько системных привилегий для нескольких пользователей.  Пользователи не могут использовать системные привилегии CREATEDATABASEorCREATETABLE, если они не наделены имим через командуGRANT.  Пример:  DENY CREATE DATABASE, CREATE TABLE  TO Mary, John, [Corporate\BobJ]  DENY SELECT, INSERT, UPDATE, DELETE ON authors TO Mary, John, Tom |
| Grant (предоставлять) | This example grants multiple statement permissions to the users Mary and John, and the Corporate\BobJ Windows NT group.  GRANT CREATE DATABASE, CREATE TABLE  TO Mary, John, [Corporate\BobJ]  Назначение разрешения на выборку (SELECT) для ролиPUBLICв таблицеAuthors:  GRANT SELECT ON Authors TO public |
| Revoke (отменять) | This example revokes multiple statement permissions from multiple users.  REVOKE CREATE TABLE, CREATE DEFAULT  FROM Mary, John  This example removes the denied permission from Mary and, through the SELECT permissions applied to the Budget role, allows Mary to use the SELECT statement on the table.  REVOKE SELECT ON Budget\_Data TO Mary |

Создание ролей программно

Для упрощения управления правами доступа в системе создаются роли, которые затем можно назначать группе пользователей.

Создадим для нашего примера роли библиотекаря (LIBRAR) и читателя (READER).

Пример создания роли библиотекаря:

USE Demo\_DataBase --сделать текущей БД Demo\_DataBase

EXEC sp\_addrole 'LIBRAR'

Повторный запуск тех же команд сгенерирует ошибки типа «В БД уже существует роль LIBRAR».

Пример создания роли читателя:

USEDemo\_DataBase--сделать текущей БДDemo\_DataBase

EXEC sp\_addrole 'READER'

Библиотекарь должен обладать правами на чтение, удаление, изменение, добавление во все таблицы БД Demo\_DataBase, а также должен иметь возможность запускать на исполнение процедуры и функции БДDemo\_DataBase. Поэтому роли библиотекаря из системных привилегий назначаемEXECUTE, а из привилегий доступа к объектам назначаем DELETE, INSERT, UPDATE, SELECT.

Читатель должен обладать правами на чтение из таблиц ADMIN\_BOOKS.AUTHORS, ADMIN\_BOOKS.BOOKS, ADMIN\_BOOKS.PUBLISHING\_HOUSE. Поэтому роли читателя из привилегий доступа к объектам назначаем SELECT.

Оператор представления привилегий

Синтаксис:

grant<привилегия>, ...

ON < объект >, …

TO <имя>

[with grant option];

Атрибут with grant option дает право пользователю самому раздавать права, которые он получил.

С помощью оператора grant для каждого пользователя формируется список привилегий, привилегии управляют работой сервера данных с точки зрения защиты данных. Выполнению каждой транзакции предшествует проверка привилегий пользователя, сеанс которого породил транзакцию.

Пользователь, предоставивший привилегию другому называется грантор (grantor — предоставитель). Привилегия является предоставляемой, если право на нее можно предоставить другим пользователям.

PUBLIC— имя роли, которую получает пользователь при добавлении в список пользователей конкретной БД, включает в себя минимальный набор прав на чтение данных из таблиц и представлений в БД.

Роль библиотекаря названа LIBRAR. Операторы назначения прав доступа для этой роли представлены ниже:

GRANT DELETE, INSERT, UPDATE, SELECT ON BOOKS TO LIBRAR

GRANT DELETE, INSERT, UPDATE, SELECT ON AUTHORS TO LIBRAR

GRANT DELETE, INSERT, UPDATE, SELECT ON DELIVERIES TO LIBRAR

GRANT EXECUTE TO LIBRAR

Роль читателя названа READER. Операторы назначения прав доступа для этой роли представлены ниже:

GRANT SELECT ON BOOKS TO READER

GRANT SELECT ON AUTHORS TO READER

GRANT SELECT ON PUBLISHING\_HOUSE TO READER

Создание пользователей с определенной ролью

Пример создания библиотекаря Ivanov\_Lib и присвоения ему роли:

Exec sp\_addlogin 'Ivanov\_Lib','Ivanov', 'Demo\_DataBase'

use Demo\_DataBase

Exec sp\_adduser 'Ivanov\_Lib','Ivanov\_Lib'

EXEC sp\_addrolemember 'LIBRAR', 'Ivanov\_Lib'

Пример создания читателя Petrov\_Read и присвоения роли:

Exec sp\_addlogin 'Petrov\_Read','Petrov', 'Demo\_DataBase'

use Demo\_DataBase

Exec sp\_adduser 'Petrov\_Read','Petrov\_Read'

EXEC sp\_addrolemember 'READER', 'Petrov\_Read'

Оператор отмены привилегий

Синтаксис отмены привилегий:

Revoke[with grant option]

< привилегии >,…

ON < объект >,…

FROM <имя\_пользователя>;

Предложение with grant optionсохраняет за пользователем перечисленные привилегии, но отменяет его право передавать их кому-либо другому.

**Пример:**

REVOKE SELECT ON AUTHORS FROM READER

Оператор изымания роли у пользователя:

Revoke <список ролей> from <список пользователей>.

Пример:

use Demo\_DataBase

EXEC sp\_droprolemember 'READER', 'Petrov\_Read'