[Исходники баз данных](https://drive.google.com/file/d/0B1bbH3I82mAVeHR5RG9YQ2p1YlU/view?usp=sharing" \t "_blank)

Выполнять инструкции или осуществлять операции с объектами баз данных могут только авторизованные пользователи. Попытка выполнения любой из этих задач неавторизованным пользователем будет неудачной. Для выполнения задач, связанных с авторизацией, используются следующие три инструкции языка Transact-SQL: GRANT, DENY и REVOKE.

Прежде чем приступить к рассмотрению этих инструкций, будет полезным освежить наиболее важные факты о модели безопасности компонента Database Engine. Итак, эта модель разделяет мир сервера базы данных на принципалов безопасности и защищаемые объекты. Каждый защищаемый объект имеет связанные с ним разрешения, которые могут быть предоставлены принципалу. Принципалы, такие как отдельные лица, группы или приложения, могут обращаться к защищаемым объектам. *Защищаемые объекты* - это ресурсы, доступ к которым регулируется подсистемой авторизации. Существует три основных класса защищаемых объектов: сервер, база данных и схема, которые содержат другие защищаемые объекты, такие как регистрационные имена, пользователи базы данных, таблицы и хранимые процедуры.

Инструкция GRANT

***Инструкция GRANT*** предоставляет разрешения принципалам на защищаемые объекты. Эта инструкция имеет следующий синтаксис:

GRANT {ALL [PRIVILEGES]} | permission\_list

[ON [class::] securable] TO principal\_list [WITH GRANT OPTION]

[AS principal ]

[Соглашения по синтаксису](https://professorweb.ru/my/sql-server/2012/level1/1_0.php)

Предложение ALL означает, что указанному принципалу предоставляются все применимые к указанному защищаемому объекту разрешения. В параметре permission\_list указываются разрешаемые инструкции или объекты (разделенные запятыми), а в параметре class - класс или имя защищаемого объекта, для которого предоставляются разрешения. Предложение on securable указывает защищаемый объект, на который предоставляется разрешение. В параметре prinicpal\_list перечисляются все учетные записи (разделенные запятыми), которым предоставляются разрешения. Параметр principal и составляющие списка principal\_list могут быть учетной записью пользователя Windows, регистрационным именем или учетной записью пользователя, сопоставленной с сертификатом, регистрационным именем, сопоставленным с асимметричным ключом, пользователем базы данных, ролью базы данных или ролью приложения.

В таблице ниже приводятся разрешения и защищаемые объекты, к которым они применяются, а также краткое описание предоставляемых каждым разрешением возможностей:

|  |  |  |
| --- | --- | --- |
| Разрешения и соответствующие защищаемые объекты | | |
| **Разрешение** | **Применение** | **Описание** |
| *SELECT* | Таблицы и их столбцы, синонимы, представления и их столбцы, возвращающие табличные значения функции | Предоставляет возможность выборки (чтения) строк. Это разрешение можно ограничить одним или несколькими столбцами, перечислив требуемые столбцы. (Если список столбцов отсутствует, то разрешение применимо ко всем столбцам таблицы) |
| *INSERT* | Таблицы и их столбцы, синонимы, представления и их столбцы | Предоставляет возможность вставлять столбцы |
| *UPDATE* | Таблицы и их столбцы, синонимы, представления и их столбцы | Предоставляет возможность изменять значения столбцов. Это разрешение можно ограничить одним или несколькими столбцами, перечислив требуемые столбцы. (Если список столбцов отсутствует, то разрешение применимо ко всем столбцам таблицы) |
| *DELETE* | Таблицы и их столбцы, синонимы, представления и их столбцы | Предоставляет возможность удалять столбцы |
| *REFERENCES* | Определяемые пользователем функции (SQL и среды CLR), таблицы и их столбцы, синонимы, представления и их столбцы | Предоставляет возможность обращаться к столбцам внешнего ключа в родительской таблице, когда пользователь не имеет разрешения SELECT для этой таблицы |
| *EXECUTE* | Хранимые процедуры (SQL и среды CLR), определяемые пользователем функции (SQL и среды CLR), синонимы | Предоставляет возможность выполнять указанную хранимую процедуру или определенную пользователем функцию |
| *CONTROL* | Хранимые процедуры (SQL и среды CLR), определяемые пользователем функции (SQL и среды CLR), синонимы | Предоставляет возможности, подобные возможностям владельца; получатель имеет практически все разрешения, определенные для защищаемого объекта. Принципал, которому было предоставлено разрешение CONTROL, также имеет возможность предоставлять разрешения на данный защищаемый объект. Разрешение CONTROL на определенной области видимости неявно включает разрешение CONTROL для всех защищаемых объектов в этой области видимости |
| *ALTER* | Хранимые процедуры (SQL и среды CLR), определяемые пользователем функции (SQL и среды CLR), таблицы, представления | Предоставляет возможность изменять свойства (за исключением владения) защищаемых объектов. Когда это право предоставляется применимо к области, оно также предоставляет права на выполнение инструкций ALTER, CREATE и DROP на любых защищаемых объектах в данной области |
| *TAKE OWNERSHIP* | Хранимые процедуры (SQL и среды CLR), определяемые пользователем функции (SQL и среды CLR), таблицы, представления, синонимы | Предоставляет возможность становиться владельцем защищаемого объекта, для которого оно применяется |
| *VIEW DEFINITION* | Хранимые процедуры (SQL и среды CLR), определяемые пользователем функции (SQL и среды CLR), таблицы, представления, синонимы | Предоставляет получателю возможность просматривать метаданные защищаемого объекта |
| *CREATE (безопасность сервера)* | Нет данных | Предоставляет возможность создавать защищаемые объекты сервера |
| *CREATE (безопасность базы данных)* | Нет данных | Предоставляет возможность создавать защищаемые объекты базы данных |

В таблице перечислены только наиболее важные разрешения. Так как модель безопасности компонента Database Engine является иерархической, то она содержит многие гранулярные разрешения, которые не отображены в списке. Описание этих разрешений смотрите в электронной документации.

Применение инструкции GRANT показано в примерах ниже. Для начала, в следующем примере показано использование разрешения CREATE:

USE SampleDb;

GRANT CREATE TABLE, CREATE PROCEDURE

TO Vasya, [ProfessorWeb\Alexandr];

В этом примере пользователям Vasya и [ProfessorWeb\Alexandr] дается право на выполнение инструкций языка Transact-SQL CREATE TABLE и CREATE PROCEDURE. (Как можно видеть в этом примере, инструкция GRANT для разрешения CREATE не включает параметр ON.)

В примере ниже пользователю Vasya предоставляется возможность для создания определяемых пользователем функций в базе данных SampleDb:

USE SampleDb;

GRANT CREATE FUNCTION TO Vasya;

Далее показано использование разрешения SELECT в инструкции GRANT:

USE SampleDb;

GRANT SELECT ON Employee

TO Vasya;

Здесь пользователь Vasya получает разрешение на чтение строк из таблицы Employee. Когда разрешение дается учетной записи пользователя Windows или регистрационному имени, это разрешение распространяется только на данную учетную запись (регистрационное имя). С другой стороны, разрешение, предоставленное группе или роли, распространяется на всех пользователей данной группы или роли.

В примере ниже показано использование разрешения UPDATE в инструкции GRANT:

USE SampleDb;

GRANT UPDATE ON Works\_on (EmpId, EnterDate)

TO Vasya;

После выполнения инструкции GRANT в этом примере ниже, пользователь Vasya может модифицировать значения столбцов Id и EnterDate таблицы Works\_on.

В следующем примере показано использование разрешения VIEW DEFINITION, которое предоставляет пользователям доступ для чтения метаданных:

USE SampleDb;

GRANT VIEW DEFINITION ON OBJECT::Employee TO Vasya;

GRANT VIEW DEFINITION ON SCHEMA::dbo TO Vasya;

Здесь показаны две инструкции для разрешений VIEW DEFINITION. Первая из них предоставляет пользователю Vasya разрешение на просмотр метаданных таблицы Employee базы данных SampleDb. (В предложении ON OBJECT указывается защищаемый объект базы данных. Посредством этого предложения можно предоставлять разрешения для работы с конкретными объектами, такими как таблицы, представления и хранимые процедуры.) Благодаря иерархической структуре защищаемых объектов, защищаемый объект более высокого уровня можно использовать, чтобы расширить область действия разрешения VIEW DEFINITION (или любого другого базового разрешения). Вторая инструкция GRANT в примере предоставляет пользователю Vasya доступ к метаданным всех объектов схемы dbo базы данных SampleDb.

В версиях до SQL Server 2005 было можно запрашивать информацию по всем объектам базы данных, даже если этими объектами владел другой пользователь. В последующих версиях разрешение VIEW DEFINITION позволяет предоставлять или запрещать доступ к разным частям метаданных, решая, таким образом, какую часть метаданных разрешать для просмотра другим пользователям.

В примере ниже показано использование разрешения CONTROL:

USE SampleDb;

GRANT CONTROL ON DATABASE::SampleDb TO Vasya;

Здесь пользователю Vasya предоставляются, по сути, все определенные права доступа к защищаемому объекту (в данном случае к базе данных SampleDb). Принципалу, которому предоставлено разрешение CONTROL для защищаемого объекта, также неявно предоставляется возможность самому предоставлять разрешения для данного объекта. Иными словами, разрешение CONTROL включает в себя предложение WITH GRANT OPTION. Разрешение CONTROL является самым высшим разрешением, применительно ко многим защищаемым объектам базы данных. Вследствие этого, разрешение CONTROL на уровне определенной области неявно включает разрешение CONTROL для всех защищаемых объектов в этой области. Поэтому разрешение CONTROL пользователя Vasya для базы данных SampleDb неявно включает в себя все разрешения к этой базе данных, а также все разрешения ко всем сборкам этой базы данных, все разрешения ко всем схемам и объектам базы данных.

По умолчанию, если пользователь A предоставляет разрешение пользователю B, то пользователь B может использовать разрешения при выполнении инструкций языка Transact-SQL в инструкции GRANT. **Предложение WITH GRANT OPTION** дает пользователю B дополнительную возможность предоставлять данное разрешение другим пользователям:

USE SampleDb;

GRANT SELECT ON Works\_on TO Vasya

WITH GRANT OPTION;

В этом примере пользователю Vasya предоставляется разрешение выполнять инструкцию SELECT для выборки строк из таблицы Works\_on, а также право самому предоставлять это разрешение другим пользователям базы данных SampleDb.

Инструкция DENY

***Инструкция DENY*** запрещает пользователю выполнять указанные действия на указанных объектах. Иными словами, эта инструкция удаляет существующие разрешения для учетной записи пользователя, а также предотвращает получение разрешений пользователем посредством его членства в группе или роли, которое он может получить в будущем. Эта инструкция имеет следующий синтаксис:

DENY {ALL [PRIVILEGES] } | permission\_list

[ON [class::] securable] TO principal\_list

[CASCADE] [ AS principal ]

[Соглашения по синтаксису](https://professorweb.ru/my/sql-server/2012/level1/1_0.php)

Все параметры инструкции DENY имеют точно такое же логическое значение, как и одноименные параметры инструкции GRANT. Инструкция DENY имеет дополнительный *параметр CASCADE*, в котором указывается, что разрешения, запрещенные пользователю A, будут также запрещены пользователям, которым он их предоставил. Если в инструкции DENY параметр CASCADE опущен, и при этом ранее были предоставлены разрешения для соответствующего объекта с использованием предложения WITH GRANT OPTION, исполнение инструкции DENY завершается ошибкой.

Инструкция DENY блокирует разрешения, полученные пользователем, группой или ролью посредством их членства в группе или роли. Это означает, что если член группы, которому запрещено разрешение, предоставленное для группы, то этот пользователь будет единственным из группы, кто не сможет использовать это разрешение. С другой стороны, если разрешение запрещено для всей группы, все члены этой группы не смогут пользоваться этим разрешением.

Инструкцию GRANT можно рассматривать как положительную авторизацию пользователя, а инструкцию DENY - как отрицательную. Обычно инструкция DENY используется для запрещения разрешений, уже предоставленных для группы (или роли), отдельным членам этой группы.

Использование инструкции DENY показано в примерах ниже. В следующем примере мы запрещаем пользователю Vasya два предоставленных ранее разрешения:

USE SampleDb;

DENY CREATE TABLE, CREATE PROCEDURE

TO Vasya;

Инструкция DENY в примере отменяет для пользователя Vasya ранее предоставленные ему разрешения на создание таблиц и процедур. В примере ниже показана негативная авторизация для некоторых пользователей базы данных SampleDb:

USE SampleDb;

GRANT SELECT ON Project TO PUBLIC;

DENY SELECT ON Project TO Vasya;

Вначале предоставляется разрешение на выборку всех строк из таблицы Project всем пользователям базы данных SampleDb. После этого это разрешение отменяется для пользователя Vasya.

Запрещение разрешений на более высоком уровне модели безопасности компонента Database Engine аннулирует разрешения, предоставленные на более низком уровне. Например, если разрешение SELECT запрещено на уровне базы данных SampleDb, и это разрешение предполагается для таблицы Employee, в результате чего разрешение SELECT будет запрещено для таблицы Employee так же, как и для всех других таблиц этой базы данных.

Инструкция REVOKE

***Инструкция REVOKE*** удаляет предоставленное или запрещенное ранее разрешение. Эта инструкция имеет следующий синтаксис:

REVOKE [GRANT OPTION FOR]

{ [ALL [PRIVILEGES] ] | permission\_list ]}

[ON [class:: ] securable ]

FROM principal\_list [CASCADE] [ AS principal ]

[Соглашения по синтаксису](https://professorweb.ru/my/sql-server/2012/level1/1_0.php)

Единственным новым параметром инструкции REVOKE является *параметр GRANT OPTION FOR*. Все другие параметры этой инструкции имеют точно такое же логическое значение, как и одноименные параметры инструкций GRANT или DENY. Параметр GRANT OPTION FOR используется для отмены эффекта предложения WITH GRANT OPTION в соответствующей инструкции GRANT. Это означает, что пользователь все еще будет иметь предоставленные ранее разрешения, но больше не сможет предоставлять их другим пользователям.

Инструкция REVOKE отменяет как "позитивные" разрешения, предоставленные инструкцией GRANT, так и "негативные" разрешения, предоставленные инструкцией DENY. Таким образом, его функцией является нейтрализация указанных ранее разрешений (позитивных и негативных).

Использование инструкции REVOKE показано в примере ниже:

USE SampleDb;

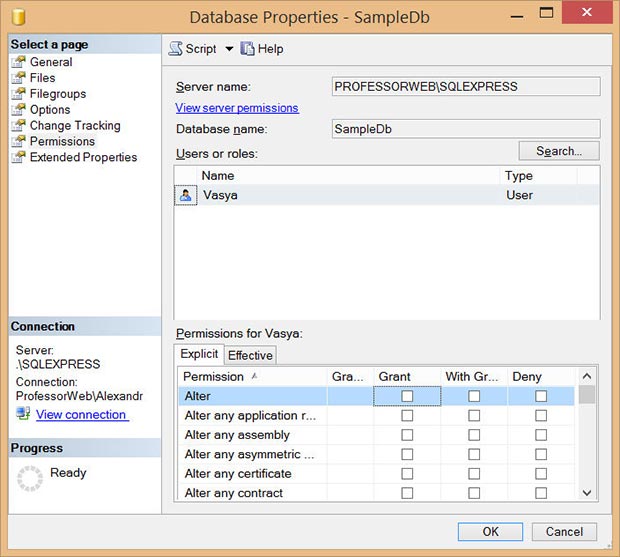
REVOKE SELECT ON Project FROM public;

Инструкция REVOKE в этом примере отменяет предоставленное разрешение выборки данных для роли public. При этом существующее запрещение разрешения этой инструкции для пользователя Vasya не удаляется, поскольку разрешения, предоставленные или запрещенные явно членам группы (или роли), не затрагиваются удалением этих разрешений (положительных или отрицательных) для данной группы.

Управление разрешениями с помощью среды Management Studio

Пользователи базы данных могут выполнять действия, на которые им были предоставлены разрешения. Разрешения на определенные действия установлены в значение g (от GRANT) в столбце state в представлении просмотра каталога sys.database\_permissions. Негативная запись в таблице не дает возможность пользователям выполнять деятельность. Значение d (от DENY) в этом столбце state аннулирует разрешение, предоставленное пользователю явно или неявно посредством предоставления его роли, к которой он принадлежит. Таким образом, пользователь не может выполнять данное действие в любом случае. И последнее возможное значение r (от REVOKE) в столбце state означает, что пользователь не имеет никаких явных разрешений, но может выполнять действие, если роли, к которой он принадлежит, предоставлено соответствующее разрешение.

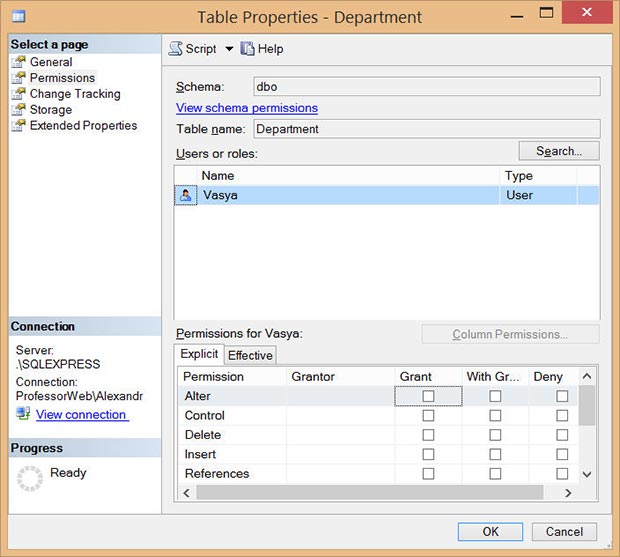
Для управления разрешениями с помощью среды Management Studio разверните сервер, а затем папку "Databases". Щелкните правой кнопкой мыши требуемую базу данных и в контекстном меню выберите пункт Properties. В открывшемся диалоговом окне свойств базы данных Database Properties - SampleDb выберите страницу Permissions, после чего нажмите кнопку Search, чтобы выбрать пользователей, которым предоставить или запретить разрешения. В открывшемся диалоговом окне нажмите кнопку Browse, в диалоговом окне Browse for Objects выберите требуемых пользователей или роли (например, guest и public) и нажмите кнопки ОК соответствующих окон, чтобы добавить выбранных пользователей (роли).



Выбранные таким образом пользователи будут отображены в окне свойств базы данных в поле Users or roles (Пользователи и роли).

Теперь для выбранных пользователей можно разрешить или запретить выполнение определенных действий. В частности, для предоставления разрешения, для требуемого действия установите флажок в столбце Grant, а для запрещения действия установите для него флажок в столбце Deny. Установленный флажок в столбце With Grant означает, что получатель разрешения также имеет право предоставлять его другим пользователям. Отсутствие установленных флажков в столбце разрешений или запрещений означает, что для данного пользователя нет никаких явных разрешений или запрещений на выполнение соответствующих действий.

Предоставление и запрещение разрешений пользователям или ролям для выполнения действий с индивидуальными таблицами базы данных посредством среды Management Studio осуществляется точно так же, как и для всей базы данных: в обозревателе объектов Object Explorer разворачивается вся иерархия папок сервера вплоть до требуемой таблицы, открывается окно свойств этой таблицы, выбираются и добавляются в поле Users or roles требуемые пользователи, после чего им предоставляются или запрещаются требуемые разрешения на исполнение определенных действий установкой соответствующих флажков:



Использование ограничений через представления

Наложение ограничения на использование определенных столбцов и/или строк означает, что механизм представления обеспечивает управление доступом к данным. Например, если таблица Employee содержит также столбец с информацией о ставке сотрудников, тогда доступ к этому столбцу можно ограничить, используя представление, которое отображает все столбцы таблицы за исключением столбца ставок. Подобным образом разрешение на такую ограниченную выборку данных из таблицы можно предоставить всем пользователям базы данных, используя данное представление, тогда как только небольшое число пользователей (имеющих повышенные права) будут иметь право на доступ ко всем данным таблицы.

В следующих трех примерах показано использование представлений для ограничения доступа к данным. В следующем примере мы добавляем ограничение доступа к таблице Project ее двумя столбцами:

USE SampleDb;

GO

CREATE VIEW view\_without\_budget

AS SELECT Number, ProjectName

FROM Project;

Как показано в примере ниже, представление view\_without позволяет разделить пользователей на две группы: тех, кто может просматривать бюджеты всех проектов, и тех, для которых доступ к столбцу Budget таблицы Projects не предоставляется.

USE SampleDb;

GO

ALTER TABLE Employee

ADD user\_name CHAR(60) DEFAULT SYSTEM\_USER;

GO

CREATE VIEW view\_my\_rows

AS SELECT Id, FirstName, LastName, DepartamentNumber

FROM Employee

WHERE user\_name = SYSTEM\_USER;

В этом примере модифицируется схема таблицы Employee, добавляя в нее новый столбец user\_name. Теперь при каждой вставке строки в эту таблицу в столбец user\_name вставляется регистрационное имя пользователя, осуществляющего эту вставку. Создав соответствующие представления, пользователи могут с помощью этих представлений выбирать из таблицы только те строки, которые они сами вставили.

USE SampleDb;

GO

CREATE VIEW view\_analyst

AS SELECT Employee.Id, FirstName, LastName

FROM Employee, Works\_on

WHERE Employee.Id = Works\_on.EmpId

AND Job = 'Аналитик';

Представление view\_analyst в примере представляет горизонтальное и вертикальное подмножество таблицы Employee. Иными словами, оно ограничивает доступ к определенным строкам и столбцам этой таблицы.