Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Курганский государственный университет»

Кафедра программного обеспечения
автоматизированных систем

В. К. Волк

ЗАЩИТА ДАННЫХ В РЕЛЯЦИОННЫХ СУБД

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

ДЛЯ СТУДЕНТОВ
ИТ-СПЕЦИАЛЬНОСТЕЙ

Курган 2026

Кафедра: Программное обеспечение автоматизированных систем

Составил: доцент В. К. Волк

Печатается в соответствии с планом издания, утвержденным методическим советом университета «17» декабря 2025 года.

Утверждены на заседании кафедры «07» апреля 2026 года.

Аннотация

В методическом пособии приведены практические и контрольные задания различных уровней сложности, примеры выполнения этих заданий и ссылки на информационные источники по пяти основным аспектам управления доступом к объектам баз данных: анализ требований нормативных документов к защищенности автоматизированных систем, дискреционная и мандатная модели управления доступом к данным, безопасность уровня строк таблиц, хеширование, динамическое маскирование и шифрование данных.

Пособие может быть рекомендовано преподавателям при проведении лабораторно-практических занятий и контрольных работ, планировании самостоятельной работы студентов.

Выполнение заданий потребует от студентов определенной подготовки в области проектирования и программирования баз данных: концептуальная (ER) и логическая (реляционная) модель данных и ее реализация (физическая модель реляционной базы данных), основы SQL (DDL, DML, элементы процедурного SQL-программирования). Желательно также наличие у студентов практического опыта работы в среде одного из промышленных серверов баз данных.

Содержание

Общие методические рекомендации

Методы защиты информации, реализуемые на уровне серверов баз данных, изучаются студентами ИТ-специальностей в рамках профильных дисциплин «Администрирование программных систем» (01.05.01 – Фундаментальная математика и механика, 09.03.03 – Прикладная информатика, 09.03.04 – Программная инженерия) и «Безопасность систем баз данных» (10.05.03 – Информационная безопасность автоматизированных систем).

В методическом пособии представлены шесть тем тематического плана дисциплин, практическая часть которых нацелена на получение студентами навыков администрирования подсистем информационной безопасности на уровне серверов баз данных и освоение соответствующих программных инструментов. В процессе выполнения практических заданий исследуются:

- требования международных и российских нормативных документов к степени защищенности автоматизированных систем и к компетентности специалистов по информационной безопасности, проектирующих и обслуживающих такие системы;
- дискреционная и мандатная модели управления доступом;
- состав объектов и субъектов доступа уровня сервера и базы данных, SQL-средства их создания/модификации, ролевая модель управления доступом и соответствующие DCL-команды;
- хеширование данных как средство идентификации субъектов доступа;
- динамическое маскирование данных;
- шифрование данных.

Каждая тема содержит краткую аннотацию, вопросы для подготовки к контрольному тестированию и практические задания различных уровней сложности:

- *начальный уровень* (с весьма низким порогом вхождения) рассчитан на студентов, не заинтересованных в глубоком изучении дисциплины; выполнение студентом заданий этого уровня (при успешном прохождении контрольного тестирования) гарантирует ему получение оценки «удовлетворительно» на зачете или экзамене;
- выполнение более сложных заданий *базового уровня* гарантирует студенту получение оценки «хорошо».
- для получения оценки «отлично» студент должен выполнить задания *продвинутого уровня*.

Практические задания выполняются студентами индивидуально, их количество, состав и уровни сложности определяются преподавателем в соответствии с бюджетом времени, выделенным на проведение аудиторных занятий и самостоятельную работу студентов.

В таблицах 1 и 2 приведены рекомендации по количеству контрольных и практических заданий различных уровней сложности и балльным оценкам за их выполнение (согласно действующему в университете «Положению о балльно-рейтинговой системе контроля и оценки академической активности обучающихся»).

Таблица 1 – Рекомендации по количеству контрольных заданий и балльным оценкам результатов контрольного тестирования

Рубежный контроль	Темы дисциплины	Количество контрольных заданий	Баллов за одно задание	Всего баллов
Рубеж № 1	Тема № 1	10	0,5	0…5
Рубеж № 2	Тема № 2	10	0,5	0…5
Рубеж № 3	Темы № 3 – № 6	10	0,5	0…5

Таблица 2 – Рекомендации по количеству практических заданий различных уровней сложности и балльным оценкам результатов их выполнения

Темы дисциплины	Индивидуальные практические задания								
	Начальный уровень сложности			Базовый уровень сложности			Продвинутый уровень сложности		
	Всего заданий	Балл	Всего баллов	Всего заданий	Балл	Всего баллов	Всего заданий	Балл	Всего баллов
Тема № 1. Требования нормативных документов	17	1,5	26	-	-	-	-	-	-
Тема № 2. Управление доступом к данным	7	1	7	5	2	10	1	3	3
Тема № 3. Безопасность уровня строк (RLS)	-	-	-	2	2	4	1	3	3
Тема № 4. Хеширование данных	-	-	-	-	-	-	1	3	3
Тема № 5. Динамическое маскирование данных (DDM)	4	1	4	1	2	2	-	-	-
Тема № 6. Шифрование данных	-	-	-	-	-	-	2	3	6
Всего заданий / баллов	28		**37**	8		**16**	5		**15**
Дополнительные баллы			**11**			**15**			**20**

Практические задания начального уровня сложности являются обязательными для всех студентов, а задания базового и продвинутого уровней являются опциональными – решение о необходимости их выполнения принимается персонально для каждого студента с учетом результатов выполнения им предшествующих заданий.

Требования к содержанию и оформлению отчетов

Задания по вводной теме № 1 носят реферативный характер и требуют проведения анализа содержания соответствующих нормативных документов. Отчет о выполнении задания по этой теме оформляется в форме реферата и должен содержать основные положения анализируемого нормативного документа (со ссылками на соответствующие разделы документа).

Задания по остальным темам носят экспериментально-исследовательский характер и выполняются в среде MS SQL Server (рекомендуется использовать версию Standard Developer, поддерживающую операции шифрования данных). Отчет о выполнении заданий по этим темам должен содержать формулировку задания, описание методики проведения экспериментов, краткое описание используемых программных инструментов, иллюстрации процесса проведенных экспериментов и выводы по их результатам.

Тема 1. Анализ требований нормативных документов

Рассматриваются положения отдельных разделов нормативных документов, в которых определены требования к степени защищенности автоматизированных систем и к компетентности специалистов по информационной безопасности, проектирующих и обслуживающих такие системы.

Международный образовательный стандарт SWEBOK-v4.0 («Свод знаний по программной инженерии») [1.1]: область знаний «Безопасность программного обеспечения», в которой определены знания по защите данных и управлению доступом к данным.

ГОСТ 34.602-2020 «Информационные технологии. Техническое задание на разработку автоматизированной системы» [1.2]: раздел «Общие технические требования к АС», подраздел «Требования к защите информации».

Международный классификатор профессий ISCO-08 [1.3]: группа ISCO-08: 252 – «Специалисты по базам данных и сетям», подгруппа 252.29.6 – «Администратор безопасности ИКТ» (оперативное создание, использование и обслуживание процедур резервного копирования и восстановления баз данных; разработка и использование политик и процедур доступа к базам данных; обеспечение контроля безопасности и целостности данных).

Профессиональные стандарты РФ ОПД 06: Связь, информационные и коммуникационные технологии [1.4]: 06.11 – «Администратор баз данных», 06.032 – «Специалист по безопасности КС и сетей», 06.33 – «Специалист по защите информации в АС» (трудовые функции специалистов различный квалификационных уровней).

Руководящий материал ФСТЭК России «Автоматизированные системы. Защита от несанкционированного доступа к информации. Классификация автоматизированных систем и требования по защите информации» [1.5]. Согласно этому руководящему материалу[1], определено 3 группы и 9 классов защищенности автоматизированных систем (таблица 1.1) по трем определяющим признакам (критериям) такой классификации:

- по наличию в системе информации различных уровней конфиденциальности и степени ее конфиденциальности (5 категорий, от А до Д);
- по режиму доступа к информации (коллективный, индивидуальный);
- по различию полномочий субъектов доступа к информации (3 группы).

Таблица 1.1 – Классы защищенности автоматизированных систем

<table>
<tr><td colspan="2" rowspan="3">Режим доступа к информации</td><td rowspan="3">Группа</td><td colspan="5">Степень конфиденциальности информации</td></tr>
<tr><td colspan="3">Государственная тайна</td><td rowspan="2">Служебная тайна</td><td rowspan="2">Персональные данные</td></tr>
<tr><td>Особая важность</td><td>Сов. секретно</td><td>Секретно</td></tr>
<tr><td colspan="2"></td><td>А</td><td>Б</td><td>В</td><td>Г</td><td>Д</td></tr>
<tr><td rowspan="2">Многопользовательский</td><td>Различные права доступа</td><td>1</td><td>Класс 1А</td><td>Класс 1Б</td><td>Класс 1В</td><td>Класс 1Г</td><td>Класс 1Д</td></tr>
<tr><td>Равные права доступа</td><td>2</td><td colspan="3">Класс 2А</td><td colspan="2">Класс 2Б</td></tr>
<tr><td colspan="2">Однопользовательский</td><td>3</td><td colspan="3">Класс 3А</td><td colspan="2">Класс 3Б</td></tr>
</table>

Для каждого класса защищенности определены минимальные требования к защите информации, реализация которых должна осуществляться в рамках системы защиты информации от несанкционированного доступа (СЗИ НСД), состоящей из четырех подсистем:

- подсистема управления доступом;
- подсистема обеспечения целостности;
- подсистема регистрации и учета
- криптографическая подсистема.

В таблице 1.2 в качестве примера приведены требования к защите информации, предъявляемые каждой из этих четырех подсистем СЗИ НСД, для автоматизированных систем, отнесенных к первой классификационной группе (многопользовательские системы с различными правами доступа к информации различной степени конфиденциальности).

[1] Имеются и другие официальные документы, регламентирующие требования к защищенности автоматизированных систем: например, приказ ФСТЭК РФ №117 [1.6], в котором классифицируются угрозы и определяется требования к инфраструктуре, составу мероприятий, мерам и средствам защиты информации в государственных и иных информационных системах государственных органов, предприятий и учреждений, или семейство стандартов ГОСТ Р ИСО/МЭК 15408 «Методы и средства обеспечения безопасности. Критерии безопасности ИТ» [1.7], в которых приведен каталог компонентов безопасности и определена их иерархическая структура.

Таблица 1.2 – Требования к защите информации (АС группы 1)[2]

Требования	Классы защищенности				
	1Д	**1Г**	**1В**	**1Б**	**1А**
Подсистема управления доступом					
Идентификация, проверка подлинности и контроль доступа субъектов к томам, каталогам, файлам, записям и полям записей	-	+	+	+	+
Подсистема регистрации и учета					
Регистрация доступа программ субъектов к программам, томам, каталогам, файлам, записям и полям записей	-	+	+	+	+
Регистрация изменений полномочий субъектов доступа	-	-	+	+	+
Регистрация создаваемых защищаемых объектов доступа	-	-	+	+	+
Криптографическая подсистема					
Шифрование конфиденциальной информации	-	-	-	+	+
Шифрование информации, принадлежащей различным субъектам доступа (группам субъектов), на разных ключах	-	-	-	-	+
Использование аттестованных (сертифицированных) криптографических средств	-	-	-	+	+
Подсистема обеспечения целостности					
Обеспечение целостности программных средств и обрабатываемой информации	+	+	+	+	+
Наличие администратора (службы) защиты информации	-	-	+	+	+
Периодическое тестирование и наличие средств восстановления средств защиты информации	+	+	+	+	+
Использование сертифицированных средств защиты	-	-	+	+	+

1.1 Контрольные вопросы

SWEBOK

1) Как определены в «Руководстве по SWEBOK»:
   а) цели разработки SWEBOK?
   б) целевая аудитория SWEBOK?
   в) варианты использования SWEBOK?
2) Как связана *Программная инженерия* со следующими *дисциплинами*:
   а) *Архитектура ПО?*
   б) *Информатика?*
   в) *Информационная безопасность*?
   г) *Искусственный интеллект и машинное обучение*?
   д) *Компьютерные сети и коммуникации*?
   е) *Проектирование ПО*?
   ж) *Управление данными*?

[2] В таблице оставлены только те из регламентированных требований, которые могут быть выполнены на уровне серверов баз данных.

ГОСТ 34602

3) Какие *разделы* должны быть включены в состав документа «Техническое задание»?
4) Какие *подразделы* должны быть включены в состав раздела «Требования к АС» документа «Техническое задание»?
5) В каком подразделе раздела «Требования к АС» документа «Техническое задание» указываются требования к защите информации от несанкционированного доступа?

Руководящий материал ФСТЭК.

Классификация автоматизированных систем
и требования по защите информации

6) Перечислите *этапы классификации* АС по уровням защищенности.
7) Какие *исходные данные* рекомендуется использовать при классификации АС по уровням защищенности?
8) Какие *определяющие признаки* рекомендуется использовать при классификации АС по уровням защищенности?
9) Перечислите *группы защищенности* АС и *классы защищенности* каждой группы. АС какого класса будут наименее и наиболее защищенными?
10) Какие *режимы доступа* пользователей к информации определены для АС первой, второй и третьей *групп защищенности*?
11) Какие *подсистемы* системы защиты информации от несанкционированного доступа (СЗИ НСД) могут предъявлять требования к проектируемой АС?
12) Какие *требования* к защите информации в АС могут предъявлять следующие подсистемы СЗИ НСД?
    а) подсистема управления доступом;
    б) подсистема регистрации и учета;
    в) криптографическая подсистема;
    г) подсистема обеспечения целостности.
13) К какой классификационной группе относятся многопользовательские системы, в которых хранится и обрабатывается информация различной степени конфиденциальности, и пользователи системы могут иметь разные права доступа к информации?
14) К каким классам защищенности могут быть отнесены системы, в которых хранится и/или обрабатывается информация, составляющая государственную тайну?
15) К каким классам защищенности могут быть отнесены системы, в которых хранится и/или обрабатывается информация, составляющая служебную тайну?

16) К каким классам защищенности могут быть отнесены системы, в которых хранятся и/или обрабатываются персональные данные?
17) Какие требования предъявляет подсистема управления доступом к автоматизированным системам:
    а) 1-й классификационной группы?
    б) 2-й классификационной группы?
    в) 3-й классификационной группы?
18) Какие требования предъявляет криптографическая подсистема к автоматизированным системам:
    а) 1-й классификационной группы?
    б) 2-й классификационной группы?
    в) 3-й классификационной группы?
19) Какие из требований (см. п. 18 и 19) могут быть выполнены на уровне сервера баз данных?

Профессиональные стандарты

20) Для чего и как применяются профессиональные стандарты?
21) Дайте определения следующим терминам, используемым в российских профессиональных стандартах:
    а) обобщенная трудовая функция;
    б) трудовая функция;
    в) трудовые действия;
    г) квалификационный уровень.
22) Какие категории требований к ИТ-специалистам предъявляются российскими профессиональными стандартами?
23) Как взаимосвязаны профессиональные и образовательные стандарты?
24) Какие ИТ-профессии, представленные в реестре российских профессиональных стандартов ОПД 06, требуют выполнения трудовых функций по защите информации?
25) Какие ИТ-профессии представлены в следующих группах международного классификатора профессий ISCO-08?
    а) группа 25 – Специалисты-профессионалы по ИКТ;
    б) группа 35 – Специалисты-техники по ИКТ.

1.2 Практические задания

Задания *начального уровня сложности*

Задание 1.1. Приведите перечень *обобщенных трудовых функций* и соответствующих им *квалификационных уровней*, определенных профессиональным стандартом РФ для одной из следующих ИТ-профессий:

а) 06.011 – Администратор баз данных;

б) 06.030 – Специалист по защите информации в ТКС и сетях;
в) 06.032 – Специалист по безопасности КС и сетей;
г) 06.033 – Специалист по защите информации в АС;
д) 06.034 – Специалист по технической защите информации.

Задание 1.2. Сравните перечни ИТ-профессий следующих категорий, определенные российскими (ОПД 06) и международными (ISCO-08:13, ISCO-08:25 и ISCO-08:35) профессиональными стандартами. Приведите коды и наименования соответствующих профессий.

а) разработчики ПО;
б) руководители и менеджеры;
в) администраторы;
г) аналитики;
д) специалисты по информационной безопасности.

Задание 1.3. Определите *класс защищенности* следующих сервисов автоматизированных систем (согласно руководящему документу ФСТЭК) и обоснуйте правильность своего решения:

Сервис	Варианты использования
а) управление книжным фондом публичной библиотеки	- поиск книг по классификатору жанров; - поиск книг и журнальных статей по их авторам; - редактирование классификатора жанров печатной продукции; - регистрация новых поступлений и списания элементов книжного фонда
б) мониторинг продаж торговой площадки	- поиск товаров по иерархическому классификатору; - поиск товаров по наименованиям и производителям; - просмотр и редактирование прайс-листа; - редактирование классификатора товаров; - регистрация операций поставки и отпуска товаров; - анализ складских запасов по категориям и производителям товаров
в) мониторинг академической успеваемости студентов образовательной организации	- просмотр расписаний экзаменационных сессий; - регистрация и просмотр персональных результатов экзаменационных сессий; - формирование и просмотр статистических отчетов по результатам экзаменационных сессий (средние баллы студенческих групп по учебным дисциплинам)
г) управление сервером электронной почты	- регистрация и просмотр реквизитов клиентов; - просмотр списка корреспондентов клиента; - просмотр корреспонденции клиента; - анализ активности клиента
д) управление лечебно-профилактическим учреждением (ЛПУ)	- регистрация пациентов ЛПУ; - просмотр расписаний работы врачей и очередей приема пациентов врачами, запись пациентов на прием; - регистрация и просмотр результатов приема пациентов врачами; - формирование и просмотр статистических отчетов

Сервис	Варианты использования
е) управление клиентской службой коммерческого банка	- регистрация и просмотр реквизитов клиентов банка; - просмотр текущих остатков денежных средств на счетах клиентов банка; - анализ операционной активности клиента банка; - формирование и просмотр статистических отчетов по результатам работы с клиентами банка
ж) управление агентурной сетью государственной разведывательной службы	- регистрация и просмотр реквизитов агентов (имена, адреса, пароли, явки и пр.); - просмотр переписки агентов со своими кураторами (задания и отчеты об исполнении); - просмотр отчетов кураторов секретных агентов; - формирование и просмотр аналитических отчетов о результатах работы агентурной сети

1.3. Информационные источники

1.1. Guide to the Software Engineering Body of Knowledge v4.0 // IEEE Computer Society Ed. Hironori Washizaki, Waseda University, 2024. URL: https://ieeecs-media.computer.org/media/education/swebok/swebok-v4.pdf

1.2. ГОСТ 34.602-2020. Информационные технологии. Комплекс стандартов на автоматизированные системы. Техническое задание на разработку автоматизированной системы. URL: https://docs.cntd.ru/document/1200181804.

1.3. Международный стандарт классификации профессий ISCO-08. URL: https://www.ilo.org/sites/default/files/wcmsp5/groups/public/@europe/@ro-geneva/@sro-moscow/documents/publication/wcms_306603.pdf

1.4. Профессиональные стандарты РФ. ОПД 06: Связь, информационные и коммуникационные технологии.
URL: https://classinform.ru/profstandarty/06-sviaz-informatcionnye-i-kommunikatcionnye-tekhnologii.html

1.5. Автоматизированные системы. Защита от несанкционированного доступа к информации. Классификация автоматизированных систем и требования по защите информации. – Руководящий документ ФСТЭК.

1.6. Приказ ФСТЭК России от 11.04.2025 № 117 «Об утверждении Требований о защите информации, содержащейся в государственных информационных системах, иных информационных системах государственных органов, государственных унитарных предприятий, государственных учреждений».

1.7. Семейство стандартов ГОСТ Р ИСО/МЭК 15408 «Методы и средства обеспечения безопасности. Критерии безопасности ИТ». Часть 1 – «Введение и общая модель». Часть 2 – «Функциональные компоненты безопасности».

Тема 2. Модели управления доступом к данным

2.1 Требования CIA

Информационная безопасность включает три базовых компонента – *конфиденциальность*, *целостность* и *доступность*, называемые *триадой CIA* (*Confidentiality*, *Integrity*, *Availability*). Конфиденциальность – это способность защитить данные от лиц, не имеющих к ним прав доступа, доступность – это возможность предоставить санкционированный доступ к данным, а целостность – это предотвращение несанкционированного изменения данных с возможностью выполнения отката таких изменений в случае нарушения целостности. Обеспечение доступа к информации легальным пользователям и защита конфиденциальной информации от несанкционированного доступа – две взаимосвязанные задачи, для решения которых могут использоваться различные методы и средства, определяемые требованиями к уровню защищенности системы.

Для реализации требований CIA используется стратегия *глубокой многоуровневой защиты* (рисунок 2.1), согласно которой на каждом ее уровне размещаются соответствующие средства защиты и применяются защитные меры, затрудняющие несанкционированные проникновения и атаки.

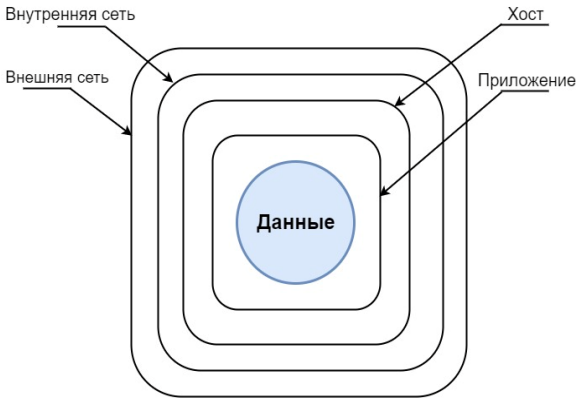


Рисунок 2.1 – Многоуровневая защита данных

Самый глубокий уровень защиты – это *уровень данных*, который представлен как минимум тремя подуровнями: файловая система, сервер баз данных и собственно база данных. Предметом изучения дисциплины являются два последних подуровня.

2.2 Дискреционная модель

Большинство коммерческих серверов баз данных используют технологию *дискреционного управления доступам* (discretionary access control), обеспечивающую *логическую защиту* данных путем разграничения прав доступа к поименованным объектам данных логического уровня (базам данных, таблицам, хра-

нимым представлениям, функциям и процедурам) со стороны субъектов доступа (пользователей и их групп, внешних приложений, хранимых функций и процедур).

Информация о субъектах доступа и логических объектах данных хранится в системных каталогах баз данных. Например, MS SQL Server хранит информацию о пользователях и их ролях в системной таблице SysUsers, информацию о таблицах и их столбцах – в таблицах SysObjects и SysColumns пользовательской базы данных, а информацию об учетных записях (именах входа) пользователей – в таблице SysLogins системной базы данных Master. Соединением (join) системных представлений sys.schemas, sys.objects, sys.database_principals и sys.database_permissions можно получить информацию о разрешениях доступа к объектов схем базы данных, предоставленных или отклоненных субъектам доступа.

В такой системе хранения данных выдача субъекту разрешения (permission) доступа к объекту данных означает установление связи между экземплярами субъектов и объектов доступа, а определение типа разрешения доступа означает присвоение соответствующего значения атрибуту такой связи. Таким образом, разрешение доступа также является логическим объектом базы данных, что ограничивает возможности использования технологий дискреционного управления доступом в информационных системах повышенного уровня защищенности.

Язык SQL предоставляет функционально полный набор средств дискреционного управления доступом, состав и синтаксис которых могут несущественно отличаться в различных реализациях. Ниже приведен краткий обзор таких языковых средств.

Категории разрешений доступа:

CREATE / ALTER – разрешение создания / модификации объектов доступа: баз данных, таблиц, представлений, процедур, функций и др.

SELECT – разрешение выборки (чтения) строк или отдельных столбцов таблицы или представления.

INSERT – разрешение вставлять строки в таблицу или представление.

UPDATE – разрешение изменять данные в таблице или ее отдельных столбцах.

DELETE – разрешение удалять строки из таблицы или представления.

REFERENCES – разрешение ссылаться на указанный объект (создавать внешние ключи в подчиненных таблицах без разрешения доступа к главным таблицам).

EXECUTE – разрешение выполнения хранимых процедур и функций.

SQL-операторы управления разрешениями:

CREATE USER … , CREATE ROLE … – создание субъектов доступа соответствующих типов.

GRANT <разрешение> ON <объект> TO <субъект> [WITH GRANT OPTION] – предоставление субъекту указанного *разрешения* доступа к объекту (опционально – с правом передачи полученных разрешений другим субъектам).

DENY <разрешение> ON <объект> TO <субъект> – запрет субъекту указанного *разрешения* доступа к объекту.

REVOKE <разрешение> ON <объект> TO <субъект> – отмена действие выполненных ранее операторов GRANT или DENY (откат до предыдущего состояния разрешений доступа).

Дополнительно к перечисленным выше SQL-средствам прямого управления доступом, серверы баз данных предоставляют администраторам программные компоненты, а также неязыковые средства экранного интерфейса.

Дискреционная защита информации удовлетворяет потребностям большинства коммерческих приложений, однако для систем повышенного уровня защищенности ее возможностей оказывается явно недостаточно. Перечислим основные проблемы защиты конфиденциальной информации, которые либо не решаются, либо решаются лишь частично в рамках технологии дискреционного (логического) управления доступом.

Во-первых, дискреционная защита не позволяет надежно разграничить доступ *к различным строкам* одной таблицы[3]. Частичное решение проблемы – запрет доступа к таблице и разрешение доступа к отдельным представлениям, созданным на базе этой таблицы и содержащим различные условия выборки строк. Слабость такого решения очевидна: невозможно разграничить доступ к нескольким строкам таблицы, удовлетворяющим одному условию выборки.

Во-вторых, разграничение доступа *к различным столбцам* одной таблицы также создает определенные проблемы. Защита, основанная на создании представления, не содержащего информации «секретных» столбцов таблицы, легко обходится средствами SQL. Более радикальное решение предполагает модификацию схемы базы данных: защищаемые столбцы таблицы выделяются в отдельную таблицу, связанную с исходной таблицей, в которой остается только общедоступная информация. Такое решение может оказаться достаточно надежным, однако оно приведет к снижению производительности системы (еще раз вспомним процедурные планы выполнения SQL-запросов с соединением таблиц).

[3] В теме № 3 будет рассмотрено применение технологии RLS как одного из возможных решений этой проблемы.

Третий, возможно, главный недостаток дискреционного управления доступом – это отсутствие средств защиты от несанкционированного *распространения* конфиденциальной информации: ничто не препятствует пользователю, легально получившему доступ (с разрешением чтения SELECT) к таблице с конфиденциальной информацией, сделать эту информацию доступной другим пользователям путем вставки (INSERT) этой информации в другую (например, временную) общедоступную таблицу.

И последнее – дискреционная защита не позволяет физически изолировать технического специалиста, выполняющего функции администратора базы данных, от управления конфиденциальными данными (в том числе и от ознакомления с ними), что требует повышения меры ответственности администратора и вряд ли соответствует политике информационной безопасности, реализуемой в хорошо защищенной системе.

Как видим, дискреционная логическая защита является довольно слабой, и основная причина такой слабости заключается в том, что информация о защите данных хранится отдельно от самих защищаемых данных. Существенно более высокий уровень защищенности информации обеспечивает так называемая *мандатная* защита, реализуемая на уровне физической модели базы данных.

2.3 Мандатная модель

Мандатное управление доступом (*mandatory access control*) основано на использовании «меток безопасности», присваиваемых экземплярам защищаемых объектов базы данных, и «мандатов», выдаваемых субъектам и разрешающих им доступ к объектам с определенными «метками безопасности». Классическая модель системы мандатного управления доступом впервые была описана Дэвидом Беллом и Леонардом Лападулой (рисунок 2.2).

Согласно этой модели, каждому объекту доступа присваивается определенный *уровень ценности* (*WAL – Write Access Level*), ограничивающий возможность его модификации, и *уровень конфиденциальности* (*RAL – Read Access Level*), ограничивающий возможность просмотра этого объекта.

WAL- и *RAL*-уровни присваиваются также и *субъектам* доступа: RAL-уровень субъекта устанавливается равным максимальному RAL-уровню объекта, доступному для прочтения этим субъектом, а WAL-уровень субъекта устанавливается равным минимальному RAL-уровню объекта, доступному для модификации этим субъектом.

Субъект не получит разрешения на чтение объекта доступа, *RAL*-уровень которого выше его собственного *RAL*-уровня, тем самым конфиденциальная информация будет защищена от несанкционированного прочтения.

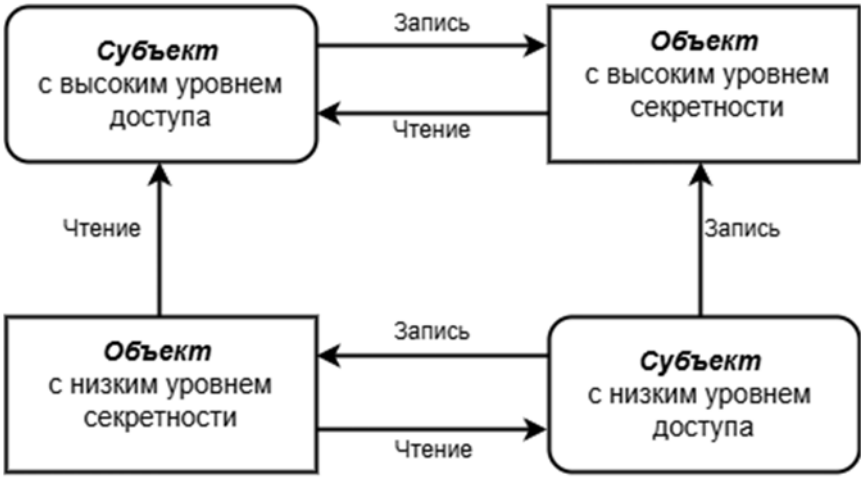


Рисунок 2.2 – Диаграмма разрешений модели мандатной защиты

WAL-уровень субъекта доступа ограничивает его возможности по понижению уровня конфиденциальности объекта: субъект не получит разрешения на модификацию (изменение, удаление или вставку) объекта доступа, *RAL*-уровень которого выше *WAL*-уровня самого этого субъекта. Иными словами, пользователь не сможет сделать доступную ему информацию менее конфиденциальной, чем задано ее *RAL*-уровнем.

2.4 Архитектура системы управления доступом MS SQL-Server

MS SQL-Server поддерживает двухуровневую архитектуру системы дискреционного управления доступом (рисунок 2.3).

Уровень сервера баз данных

Субъектами доступа уровня сервера являются *учетные записи* (имена входа, *logins*) пользователей и *фиксированные серверные роли*, используемые для группировки учетных записей. На этом уровне производится *идентификация* и *аутентификация* учетных записей и их распределение по серверным ролям, в результате чего учетные записи получают соответствующие глобальные разрешения (permissions) выполнения операций с базами данных.

Уровень базы данных

На уровне базы данных производится *авторизация* субъектов доступа, в результате которой они получают соответствующие разрешения доступа к объектам этой базы данных.

Субъекты доступа этого уровня – это *пользователи базы данных,* каждый из которых ассоциирован с определенной учетной записью сервера, и *роли базы данных*, используемые для группировки субъектов доступа. На уровне базы данных поддерживаются роли трех категорий:

- *фиксированные роли*, через членство в которых субъекты доступа получают глобальные разрешения ко всем объектам базы данных;

- создаваемые администратором *пользовательские роли*, члены которых наследуют разрешения, определенные администратором для соответствующих ролей;

- *роли приложений*, через которые прикладные программы получают доступ к компонентам базы данных.

Объекты доступа уровня базы данных – это схемы, таблицы (и их отдельные столбцы), а также хранимые в базе данных представления (view), процедуры и функции.

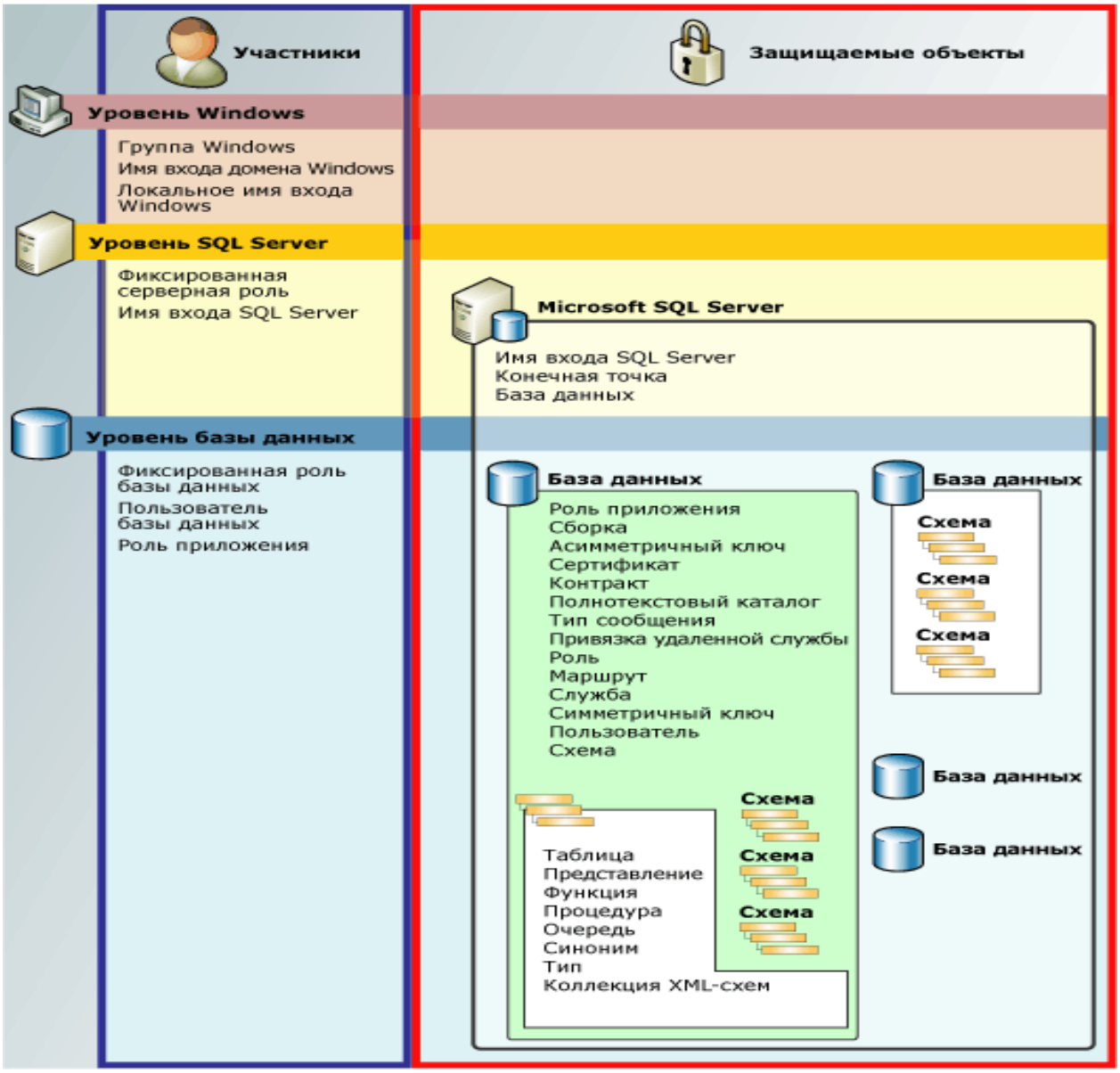


Рисунок 2.3 – Иерархия субъектов и объектов доступа MS SQL-Server

В базе данных могут быть созданы одна или несколько *схем* – логических контейнеров, используемых для группировки объектов базы данных (таблиц, представлений, процедур, функций) с целью упрощения процессов управления разрешениями доступа субъектов к таким объектам.

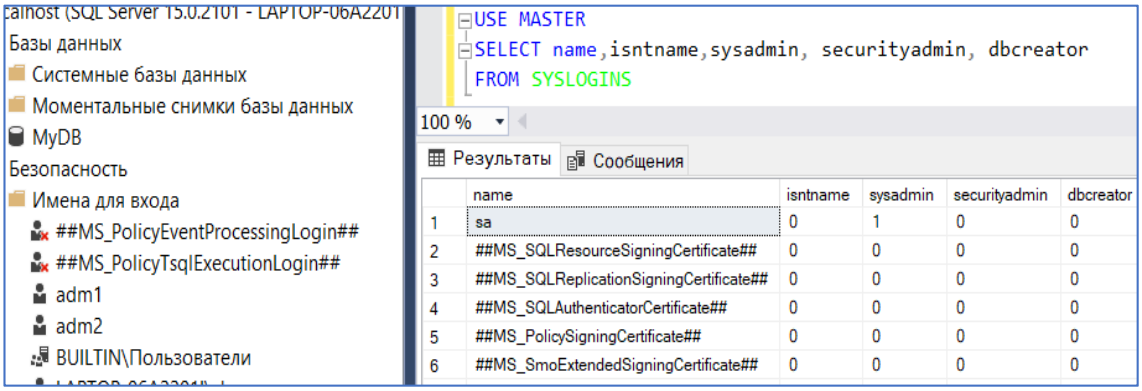
У каждой схемы базы данных имеется пользователь-владелец, который наделен правом предоставления субъектам разрешений доступа к объектам этой схемы. По умолчанию в базе данных создается схема, владельцем которой является владелец этой базы данных – пользователь dbo (data base owner), а также схемы, владельцами которых являются каждая из фиксированных ролей, пользователи sys, INFORMATION_SCHEMA и guest. В базе данных могут быть созданы и другие схемы, владельцами которых могут быть назначены пользователи, пользовательские роли или роли приложений.

2.5 Примеры

Рассмотренные ниже примеры иллюстрируют возможности просмотра и управления субъектами доступа.

Пример 2.1. Просмотр списка учетных записей

Список учетных записей с указанием их членства в фиксированных ролях сервера получен в результате выборки данных из системной таблицы SysLogins системной базы данных Master.



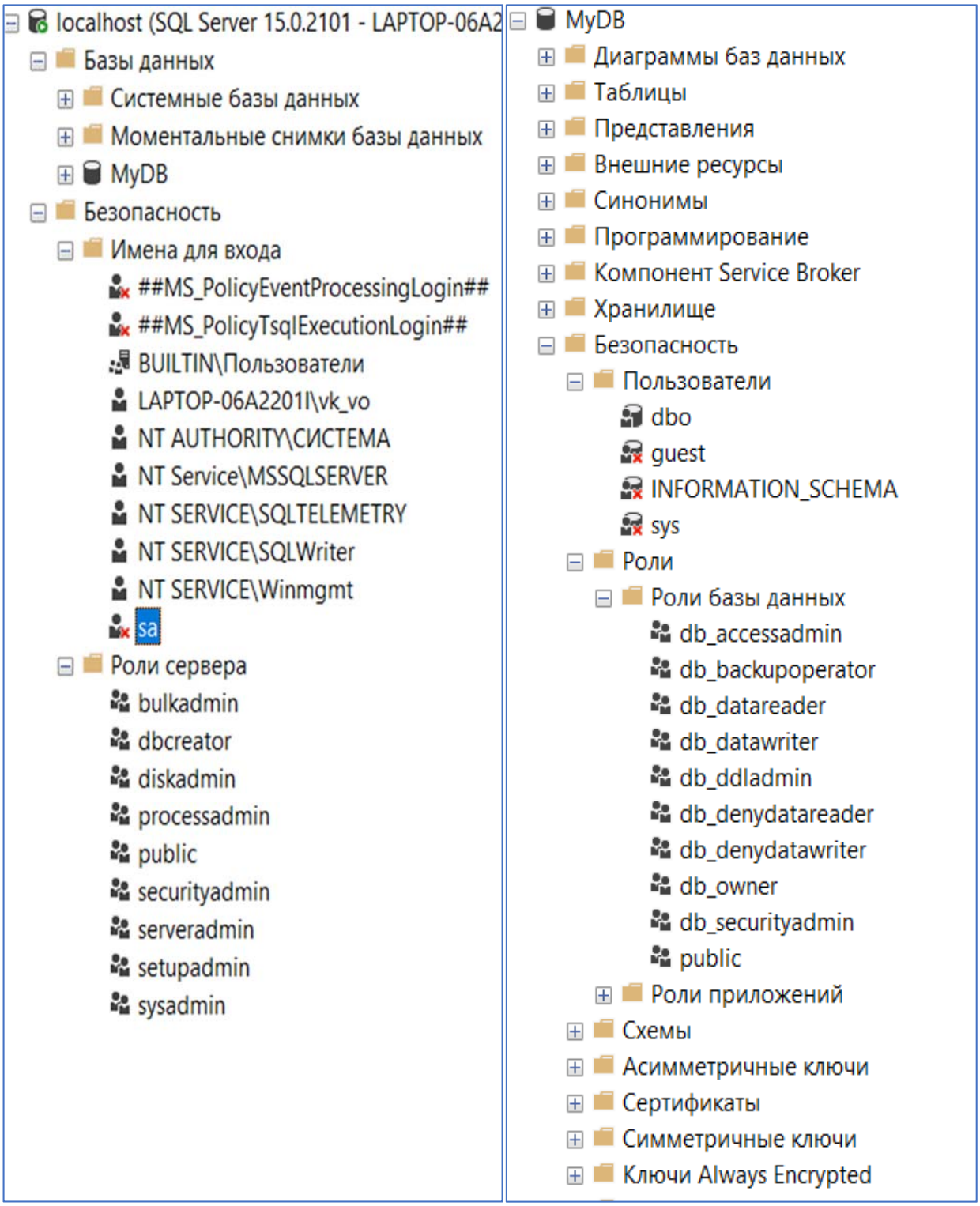
Пример 2.2. Просмотр списка субъектов доступа (пользователей и ролей базы данных) выборкой данных из системной таблицы SysUsers

```
USE MyDB
SELECT uid, name, sid, issqluser, issqlrole FROM SysUsers
```

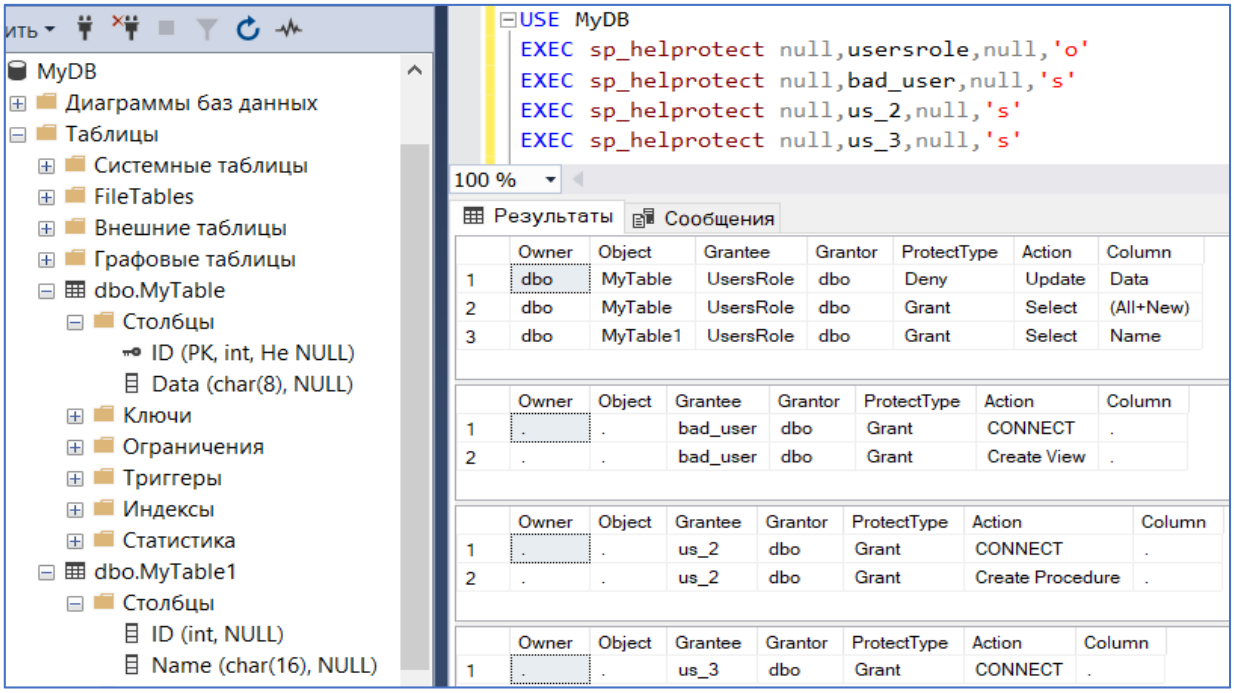
Результаты | Сообщения

uid	name	sid	issqluser	issqlrole
0	public	0x0105000000000009040000083741B0...	0	1
1	dbo	0x0105000000000005150000000FCC2F7...	0	0
2	guest	0x00	1	0
3	INFORMATION_SCHEMA	NULL	1	0
4	sys	NULL	1	0
5	bad_user	0x6D65C7F93BBB3F41A29636BCA1F3B9...	1	0
6	us_2	0xE1CD5A11ED36144D9788888CA3DDD...	1	0
7	us_3	0x507514059E56A34E8983CDEB11E37B...	1	0
8	UsersRole	0x0105000000000009040000005066A27...	0	1

Пример 2.3. Использование обозревателя объектов MS SQL Server Management Studio для просмотра состава субъектов доступа уровня сервера и базы данных

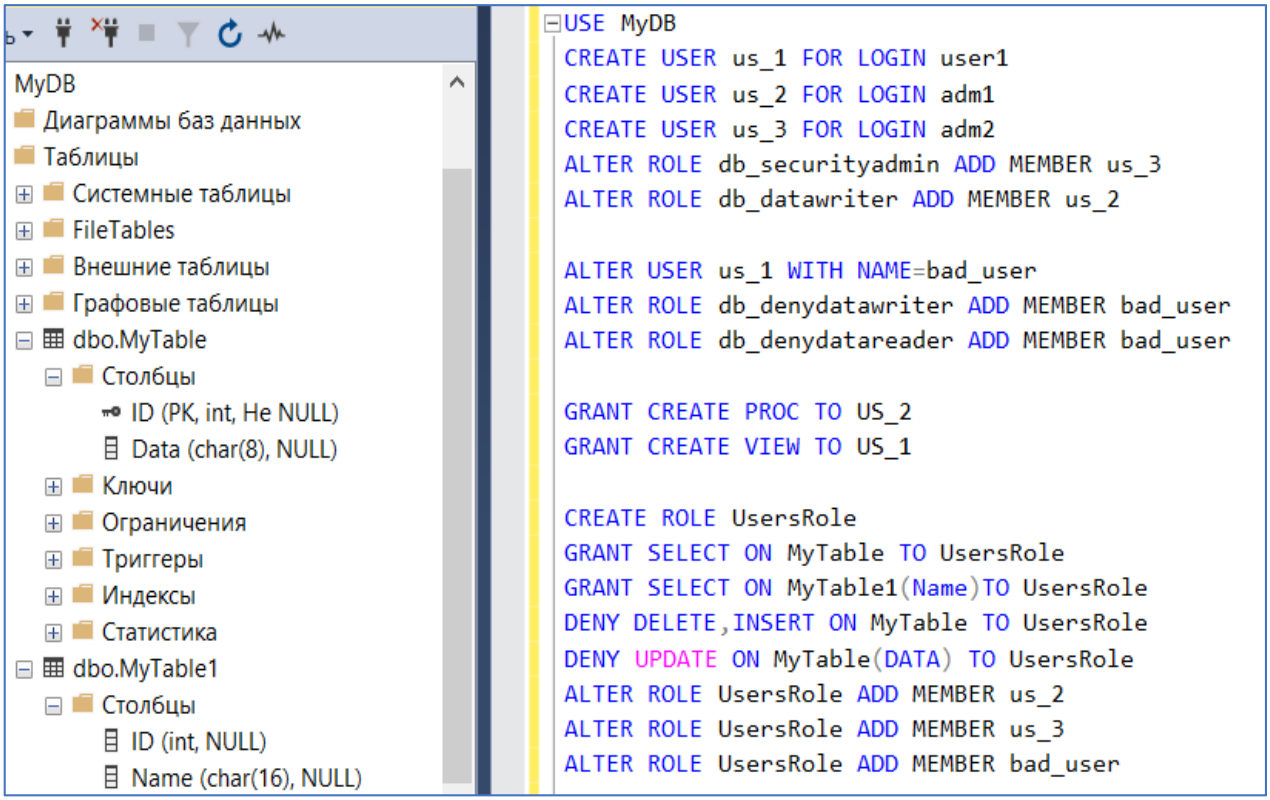


Пример 2.4. Просмотр разрешений доступа субъектов к объектам базы данных с использованием системной процедуры sp_helprotect



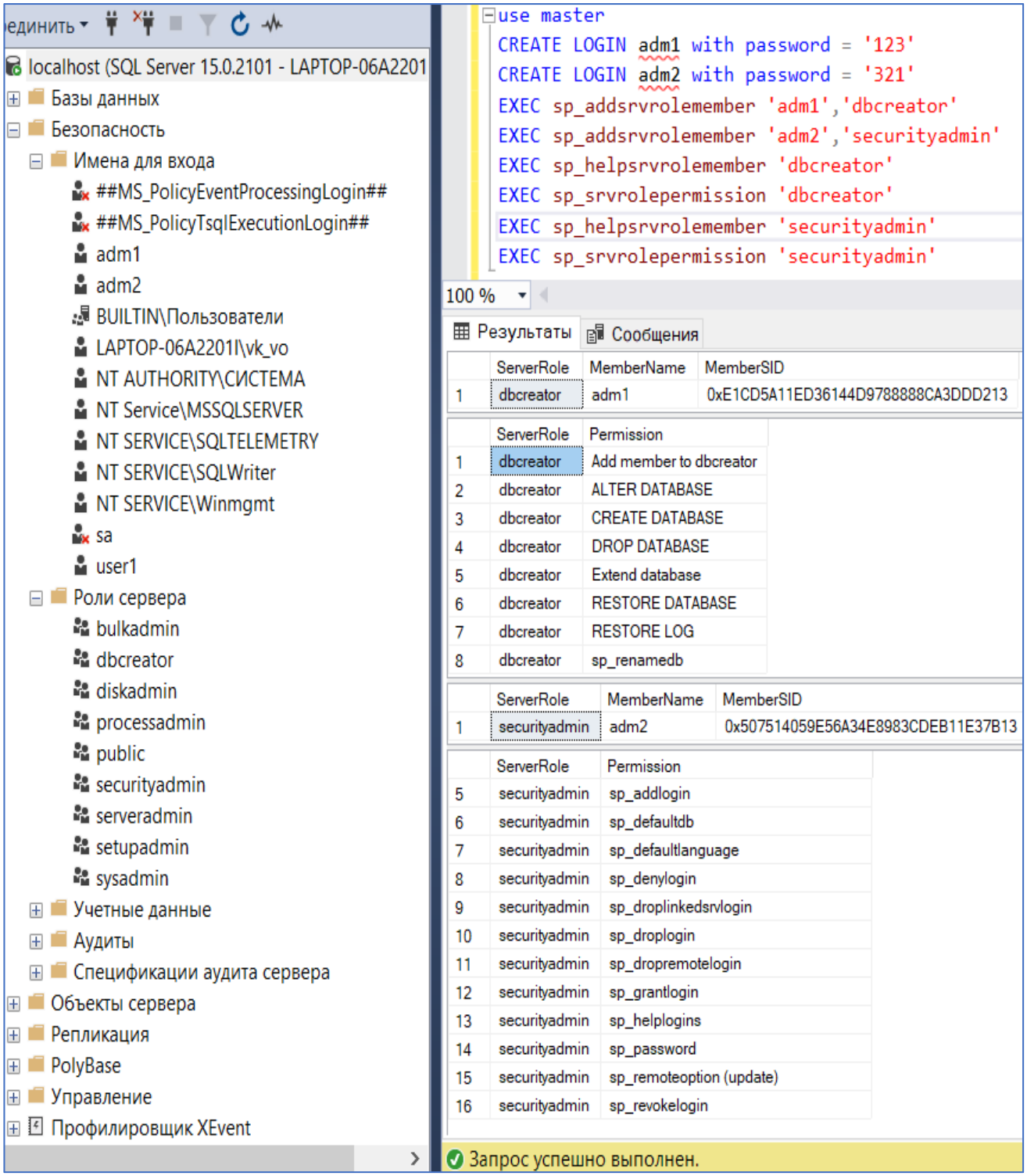
Более детальная информация о разрешениях доступа может быть получена с использованием хранимых системных представлений sys.server_permissions, sys.server_principals, sys.database_permissions и sys.database_principals.

Пример 2.5. Управление пользователями, ролями и разрешениями доступа



Пример 2.6. Управление учетными записями и ролями сервера

Системные хранимые процедуры *sp_addsrvrolemember* и *sp_helpsrvrolemember* использованы, соответственно, для включения учетных записей в серверные роли и для просмотра списка членов таких ролей, а хранимая процедура *sp_addsrvrolepermission* – для просмотра разрешений одной из серверных ролей.



Пример 2.7. Просмотр и создание схем базы данных

Список схем базы данных доступен в обозревателе объектов SQL Server Management Studio: базы данных / <база данных> / безопасность / схема. Этот же список можно получить обращением к системному представлению sys.schemas: SELECT * FROM sys.schemas.

Для создания схем в базе данных необходимо иметь в этой базе разрешение CREATE SCHEMA. Для указания другого пользователя базы данных в качестве владельца схемы создающий схему должен иметь разрешение IMPERSONATE для этого пользователя. Если владельцем создаваемой схемы назначается роль базы данных, создающий схему должен или быть членом этой роли, или иметь разрешение ALTER на эту роль.

Схема базы данных может быть создана средствами обозревателя объектов SQL Server Management Studio: базы данных / <база данных> / безопасность / схема / создать. В диалоговом окне указать имя схемы и имя ее владельца (выбрать пользователя или роль базы данных), далее перейти к страницам «Разрешения» и «Расширенные свойства».

Следующим SQL-кодом в активной базе данных создается схема new_schema, принадлежащая пользователю user_1 и содержащая таблицы table_1 и table_2. Пользователям user_2 и user_3 соответственно разрешается и запрещается чтение объектов этой схемы.

```
CREATE SCHEMA new_schema
   AUTHORIZATION user_1;
GO
CREATE TABLE table_1 (… , … , …)
CREATE TABLE table_2 (… , … , …)
);
GO
GRANT SELECT
  ON SCHEMA:: new_schema TO user_2;
GO
DENY SELECT
  ON SCHEMA:: new_schema TO user_3;
GO
```

2.6 Контрольные вопросы

1) Опишите основные черты и применяемые методы дискреционной защиты информации в реляционных базах данных. Почему такая защита получила название *логической*?
2) Верно ли утверждение о том, что при дискреционном управлении доступом информация о защите данных хранится отдельно от самих защищаемых данных?
3) Позволяет ли дискреционное управление доступом изолировать администратора базы данных от хранимой в ней конфиденциальной информации?
4) Как организовано хранение информации о субъектах и объектах доступа в MS SQL Server?
5) Предложите несколько способов защиты от несанкционированного чтения/модификации отдельных строк таблицы. Оцените их надежность и предложите способы обхода такой защиты.
6) Предложите несколько способов защиты от несанкционированного чтения/модификации отдельных столбцов таблицы. Оцените их надежность и предложите способы обхода такой защиты.
7) Опишите основные черты и применяемые методы мандатной защиты информации. Почему такая защита получила название *физической*?
8) Верно ли утверждение о том, что при мандатном управлении доступом информация, защищающая данные, хранится отдельно от самих защищаемых данных?
9) Определите понятие «метка безопасности» и опишите информационную структуру меток безопасности объекта и субъекта доступа.
10) Опишите классическую модель мандатного управления доступом (модель Белла – Лападулы):
    - может ли субъект доступа читать объект, уровень конфиденциальности которого выше его собственного уровня?
    - может ли субъект доступа читать объект, уровень конфиденциальности которого ниже его собственного уровня?
    - может ли субъект доступа модифицировать объект, уровень конфиденциальности которого выше его собственного уровня?
    - может ли субъект доступа модифицировать объект, уровень конфиденциальности которого ниже его собственного уровня?

11) Что определяют «уровень ценности» WAL (Write Access Level) и «уровень конфиденциальности» RAL (Read Access Level) объекта доступа?

12) В каком отношении находятся WAL- и RAL-уровни объекта и субъекта доступа?

13) Какие задачи защиты данных решаются на уровне сервера и на уровне базы данных?

14) Какие субъекты и объекты доступа определены на уровне сервера и на уровне базы данных?

15) Какое место занимает схема базы данных в системе разграничения доступа субъектов к объектам базы данных?

16) Какие разрешения должен иметь пользователь базы данных, создающий схему в этой базе данных?

17) Может ли пользователь, создающий схему базы данных, предоставить право владения этой схемы другому пользователю?

18) Какие категории субъектов доступа уровня базы данных могут быть владельцами схем этой базы данных?

19) Определите понятия «ролевое управление доступом», «роль сервера» и «роль базы данных». Может ли одна роль быть членом другой роли?

20) Сравните понятия «глобальные разрешения доступа» и «локальные разрешения доступа». Как можно выдать субъектам такие разрешения?

21) Сравните понятия «явные разрешения доступа» и «косвенные разрешения доступа». Как можно выдать субъектам такие разрешения?

22) Что означают разрешения типов ALL, SELECT, UPDATE, DELETE, INSERT и EXECUTE. Каким субъектам и на какие объекты доступа могут быть выданы такие разрешения?

23) Какие SQL-команды используются для управления разрешениями доступа?

24) В каких элементах системного каталога базы данных сохраняется информация о выданных разрешениях?

25) В каких элементах системного каталога базы данных сохраняется информация о созданных в этой базе данных схемах?

2.7 Практические задания

Задание 2.1. Управление субъектами и объектами доступа уровня сервера баз данных (MS SQL Server):

Задания *начального уровня* сложности

а) Перечислите субъекты и объекты доступа уровня сервера и дайте каждому из них краткую характеристику.
б) Прямым доступом к системному каталогу определите состав учетных записей и баз данных, управляемых сервером.
в) Определите состав разрешений доступа, получаемых учетными записями через их членство в следующих серверных ролях: sysadmin, diskadmin, securityadmin, dbcreator, public.

Задания *базового уровня* сложности

г) Используя SQL-средства и системные процедуры, создайте несколько учетных записей, поместите их в серверные роли (по своему усмотрению) и экспериментально подтвердите факт получения ими соответствующих разрешений доступа.
д) Допускается ли членство учетной записи одновременно в нескольких серверных ролях? Ответ подтвердите экспериментом.
е) Допускается ли членство серверной роли в другой серверной роли? Ответ подтвердите экспериментом.
ж) Получает ли учетная запись через членство в серверной роли разрешение включать другие учетные записи в эту роль и/или в какие-нибудь другие серверные роли? Ответ подтвердите экспериментально.

Задание 2.2. Управление субъектами и объектами доступа уровня базы данных.

Задания *начального уровня* сложности

а) Перечислите субъекты и объекты доступа уровня базы данных и дайте каждому из них краткую характеристику.
б) Прямым доступом к системному каталогу базы данных определите состав объектов и субъектов доступа этой базы данных.
в) Прямым доступом к системному каталогу базы данных определите состав схем этой базы данных.
г) Напишите три хранимых представления для выборки из системной таблицы Sysusers (или из представления Sys.Sysusers) информации о пользователях, пользовательских и фиксированных ролях базы данных.

д) Определите состав разрешений доступа, получаемых пользователями базы данных через их членство в следующих фиксированных ролях базы данных:

- db_datareader; db_datawrier;
- db_denydatareader; db_denydatawriter;
- db_ddladmin; db_owner; public.

Для выполнения последующих заданий потребуется:

- *создать базу данных с несколькими таблицами произвольной структуры (не менее двух столбцов) и заполнить таблицы произвольными данными;*
- *создать несколько хранимых представлений для чтения данных из соединенных (Join) таблиц;*
- *создать несколько хранимых процедур для чтения/модификации данных в созданных таблицах;*
- *используя SQL-средства и соответствующие системные процедуры, создать в этой базе данных несколько пользователей и несколько пользовательских ролей;*
- *экспериментально подтвердить правильность своих ответов на поставленные вопросы.*

е) Администраторами сервера и пользовательской базы данных реализован следующий типовой сценарий:

- в контексте системной БД «Master» создана учетная запись:

```
USE Master
Create LOGIN …
```

- в контексте пользовательской БД (например, Users_DB) для этой учетной записи создан пользователь:

```
Create User Us1 …
```

и этому пользователю передано управление:

```
Execute as Us1
Select user_name()
```

Подтвердите экспериментом свои ответы на следующие вопросы:

- в каких ролях базы данных Users_DB пользователь Us1 получил членство?
- какие разрешения доступа к объектам базы данных Users_DB получил пользователь Us1?
- как можно выдать пользователю Us1 следующие разрешения:
  - чтение/удаление/модификацию строк одной из таблиц базы данных?

- чтение/удаление/модификацию отдельных столбцов в строках одной из таблиц базы данных?
- чтение/удаление/модификацию данных всех таблиц базы данных?

Задания *базового уровня* сложности:

ж) Поместите пользователей в фиксированные роли базы данных (по своему усмотрению). Какие разрешения доступа получены пользователями соответствующих ролей?

з) Допускается ли членство пользователя одновременно в нескольких фиксированных ролях базы данных?

и) Допускается ли членство фиксированной роли в другой фиксированной роли базы данных?

к) Получает ли пользователь через членство в фиксированной роли разрешение включать других пользователей в эту роль?

л) Допускается ли членство пользователя одновременно в нескольких пользовательских ролях базы данных?

м) Допускается ли членство пользовательской роли в другой пользовательской роли базы данных?

н) Допускается ли членство пользовательской роли в фиксированной роли базы данных?

о) Получает ли пользователь через членство в пользовательской роли разрешение включать других пользователей в эту роль?

Задание 2.3. Управление разрешениями доступа к объектам БД.

а) Перечислите категории разрешений доступа (MS SQL Server) и дайте характеристику каждому из них.

б) Используя SQL-операторы Grant, Deny, Revoke, установите пользователю следующие разрешения доступа:
   - право / запрет чтения всей таблицы и её отдельных столбцов;
   - право / запрет удаления строк таблицы;
   - право / запрет вставки строк таблицы;
   - право / запрет выполнения хранимой процедуры;
   - право / запрет создания / модификации процедуры / представления;
   - отмените ранее установленные разрешения.

Задание 2.4. При разработке системы разграничения доступа пользователей к объектам базы данных используется модель Белла – Лападулы, иллюстрация которой приведена на рисунке 2.2.

- сформируйте схему реляционной базы данных, поддерживающей классификатор RAL- и WAL-уровней для объектов и субъектов доступа к

данным: субъекты доступа – пользователи, объекты доступа – таблицы базы данных, количество RAL- и WAL-уровней может изменяться в разумном диапазоне;

- приведите пример заполнения таблиц базы данных, реализующих как сам классификатор уровней доступа, так и ассоциации с ним пользователей и таблиц БД;
- разработайте хранимые процедуры, иллюстрирующие ограничения доступа в соответствии с требованиями модели Белла – Лападулы и доказывающие правильность реализации этих требований средствами реляционной модели данных.

Задания *продвинутого уровня* сложности:

Задание 2.5. Анализ иерархии разрешений доступа к данным. Экспериментально подтвердите, опровергните или уточните формулировки следующих гипотез:

Гипотеза № 1. Запрет доступа к объекту имеет более высокий приоритет по сравнению с предоставлением доступа. Рассмотрите одну из трех ситуаций:

а) персональное (явное) и косвенное (через членство в пользовательской роли) предоставление разрешений;

б) локальное (явное или косвенное) и глобальное (через членство в фиксированной роли базы данных) предоставление разрешений;

в) разрешение, установленное для таблицы в целом, и разрешение, установленное для ее отдельных столбцов.

Гипотеза № 2. Использование хранимых представлений позволяет обойти запрет доступа к таблицам базы данных. Рассмотрите одну из трех ситуаций:

а) пользователю явно установлен запрет (Deny Select) доступа к таблице, но установлено разрешение (Grant Select) доступа к представлению, в котором присутствует оператор чтения (Select) таблицы;

б) пользователю явно установлен запрет (Deny Select) доступа к таблице, но установлено разрешение (Grant Create) создания представления, в котором присутствует оператор чтения (Select) таблицы;

в) пользователю явно установлен запрет (Deny Select) доступа к таблице, но установлено разрешение (Grant Create) создания таблиц и представлений, в том числе представления, в котором присутствует оператор чтения (Select) этой таблицы и оператор копирования (Insert Into) данных в другую таблицу.

Гипотеза № 3. Использование хранимых процедур позволяет обойти запрет доступа к таблицам базы данных. Рассмотрите одну из трех ситуаций:

а) пользователю явно установлен запрет (Deny Select) доступа к таблице, но установлено разрешение (Grant Execute) доступа к хранимой процедуре, в которой присутствует оператор чтения (Select) таблицы;

б) пользователю явно установлен запрет (Deny Select) доступа к таблице, но установлено разрешение (Grant Create) создания хранимых процедур (в том числе и таких, в которых присутствует оператор чтения (Select) этой таблицы);

в) пользователю явно установлен запрет (Deny Select) доступа к таблице, но установлено разрешение (Grant Create) создания таблиц и хранимых процедур (в том числе и таких, в которых присутствует оператор чтения (Select) этой таблицы и оператор копирования (Insert Into) данных в другую таблицу).

Задание 2.6. Используя информацию открытых источников, рассмотрите, опишите и попытайтесь реализовать следующие сценарии атак проникновения на *MS SQL Server* потенциальных нарушителей информационной безопасности:

а) легальный («обычный») пользователь *MS SQL Server* нелегально получает доступ к расширенной хранимой процедуре (eXtended Procedure) xp_cmdshell и с помощью этой процедуры повышает свой статус до администратора с соответствующими разрешениями.

б) программист клиентского приложения, легально работающего с сервером баз данных, атакует *MS SQL Server* путем использования техники SQL-инъекций.

2.8 Информационные источники

2.1 Создание схем баз данных. URL: https://learn.microsoft.com/ru-ru/sql/relational-databases/security/authentication-access/create-a-database-schema?view=sql-server-ver17

2.2 Sys.database_permissions (Transact-SQL) URL: https://learn.microsoft.com/en-us/sql/relational-databases/system-catalog-views/sys-database-permissions-transact-sql.

2.3 Sys.fn_builtin_permissions (Transact-SQL) URL: https://docs.microsoft.com/en-us/sql/relational-databases/system-functions/sys-fn-builtin-permissions-transact-sql.

Тема 3. Безопасность уровня строк таблиц (RLS)

3.1 Типовые этапы подготовки и использования RLS

В ситуации, когда при проектировании базы данных было принято решение о централизованном (и единообразном) хранении какой-либо информации в одной таблице, бывает необходимо разграничить доступ к её отдельным строкам со стороны различных пользователей. Например, клиенты компании должны иметь доступ только к своим контрактам, студенты – только к своим оценкам, а врачи – к информации о своих пациентах.

Проблема в том, что строка таблицы не является объектом логической модели данных – строки в таблицах не упорядочены, они не имеют ни имен, ни порядковых номеров. Одно из возможных решений – применение технологии RLS (Row Level Security), которая позволяет использовать мета-данные о пользователях базы данных для разграничения их доступа к различным строкам таблицы. Для реализации RLS необходимо создать *предикат безопасности*, задающий условие доступности строки таблицы, и *политику безопасности*, связывающую предикат безопасности с целевой таблицей и включающую (или отключающую) режим RLS.

Предикат безопасности определяется как встроенная TVF-функция, вычисляющая значение логического выражения, операндами которого могут быть входные параметры функции и мета-данные пользователя базы данных. Предикат безопасности может использоваться в качестве *предиката фильтрации* строк (при их выборке) или *предиката блокировки* строк (при их модификации). После создания соответствующей *политики безопасности* для целевой таблицы предикат безопасности будет автоматически применяться к каждой строке этой таблицы при каждой попытке доступа.

Типовая последовательность этапов подготовки к реализации RLS (для случая использовании предиката фильтрации):

Шаг 1. Определить в базе данных целевую таблицу, доступ к строкам которой требуется разграничить (например, таблицу *Сотрудники*, в которой столбец *Имя_Сотрудника* будет использоваться в предикате фильтрации для сравнения с именем пользователя).

Шаг 2. Создать в базе данных пользователей (CREATE USER), для которых требуется разграничить доступ к строкам целевой таблицы.

Шаг 3. Создать TVF-функцию (CREATE FUNCTION), в которой должен быть определен предикат фильтрации (логическое выражение, единичное значение которого определит доступные строки целевой таблицы).

Шаг 4. Выдать пользователям разрешение GRANT SELECT на целевую таблицу и TVF-функцию предиката фильтрации.

Шаг 5. Создать политику безопасности (CREATE SECURITY POLICY), в которой должны быть заданы:

- TVF-функция, используемая в качестве предиката фильтрации (ADD FILTER PREDICATE), с указанием имени столбца целевой таблицы в качестве входного параметра (например столбца, в котором хранится имя сотрудника);
- целевая таблица (ON <таблица>);
- режим включения политики безопасности (WITH (STATE = ON)).

Сформированная политика безопасности будет автоматически применяться к целевой таблице до тех пор, пока она не будет отключена (ALTER SECURITY POLICY … WITH (STATE = OFF)).

Детальное описание, примеры и условия применения RLS приведены в соответствующем разделе технической документации [3.1].

3.2 Примеры

Пример 3.1. RLS-доступ к данным успеваемости студентов

В базе данных ведется учет успеваемости студентов по нескольким учебным дисциплинам в соответствии со следующей бизнес-моделью: студент изучает несколько дисциплин (и несколько студентов изучают одну дисциплину) под руководством нескольких преподавателей (не более одного преподавателя на одну дисциплину, но несколько дисциплин у одного преподавателя), у одного родителя может несколько студентов. Оценки студентов по изучаемым ими дисциплинам регистрируются в единой таблице Tbl_All_Grades(Студент, Дисциплина, Оценка, Преподаватель, Родитель_студента).

Следует отметить некоторую условность рассматриваемого примера – такую слабо нормализованную таблицу вряд ли стоит использовать в реальной базе данных. Будем считать, что эта таблица регулярно формируется соответствующей выборкой из множества связанных таблиц нормализованной базы данных.

В базе данных определены пользователи четырех категорий: менеджеры, преподаватели, студенты и родители студентов, при этом имя пользователя (за исключением пользователя «Менеджер») совпадает с одним из значений столбцов Студент, Преподаватель или Родитель_студента.

Студент должен иметь доступ только к своим оценкам, родитель – только к оценкам своих детей, преподаватель – только к оценкам по своим дисциплинам, а менеджер – ко всем строкам таблицы.

Этапы подготовки к реализации RLS (для случая использовании предиката фильтрации).

Шаг 1. Создаем в базе данных RLS-test_Studies целевую таблицу Tbl_All_Grades(Студент, Дисциплина, Оценка, Преподаватель, Родитель_студента) и заполняем ее данными в соответствии с описанной выше бизнес-моделью (в качестве примера: восемь студентов – детей четырех родителей, получили различные оценки по восьми дисциплинам у четырех преподавателей).

Шаг 2. Создаем 17 пользователей (CREATE USER <имя> WITHOUT LOGIN):

- восемь студентов с соответствующими именами (Student_one, … , Student_eught);
- четыре преподавателя с соответствующими именами (Teacher_one, … , Teacher _four);
- четыре родителя с соответствующими именами (Parent_one, … , Parent _four);
- один менеджер с именем Manager.

Шаг 3. Создаем предикат фильтрации:

```
CREATE FUNCTION fn_RLS_Predicate(@Person_name AS NVARCHAR(32))
RETURNS TABLE
WITH SCHEMABINDING AS
RETURN SELECT 1 AS fn_securityPredicate_result
WHERE @Person_name = USER_NAME() OR USER_NAME()='Manager';
```

Функция возвратит единицу, если имя пользователя совпадает со значением переданного в функцию параметра, или если запрос к целевой таблице выполняется от имени пользователя Manager.

В примере использована системная функция USER_NAME(), возвращающая имя пользователя (строкового типа), под которым он зарегистрирован в системном каталоге базы данных. Аналогичный результат будет получен при использовании системных переменных USER, CURRENT_USER и SESSION_USER. Системная функция USER_ID() возвращает идентификатор пользователя.

Для проверки работоспособности предиката фильтрации можно использовать следующий SQL-код:

```
execute as user = 'Manager'
select * from fn_RLS_Predicate('1234'); (возвращает 1)
execute as user = 'Student_one'
select * from fn_RLS_Predicate('Student_one'); (возвращает 1)
execute as user = 'Student_one'
select * from fn_RLS_Predicate('Student_two'); (возвращает Null)
```

Шаг 4. Выдаем разрешения GRANT SELECT на целевую таблицу Tbl_All_Grades и TVF-функцию предиката фильтрации fn_RLS_Predicate всем 17 пользователям.

Шаг 5. Создаем три политики безопасности – по одной для каждой категории пользователей:

```
CREATE SECURITY POLICY StudentsFilter
ADD FILTER PREDICATE fn_RLS_Predicate(Студент)
ON Tbl_All_Grades WITH (STATE = ON);
CREATE SECURITY POLICY TeacherFilter
ADD FILTER PREDICATE fn_RLS_Predicate(Преподаватель)
ON Tbl_All_Grades WITH (STATE = OFF);
CREATE SECURITY POLICY ParentsFilter
ADD FILTER PREDICATE fn_RLS_Predicate(Родитель_студента)
ON Tbl_All_Grades WITH (STATE = OFF);
```

Для проверки работоспособности сформированной политики безопасности выполним от имени каждого из пользователей выборку данных из целевой таблицы select * from Tbl_All_Grades:

execute as user = 'Manager' – возвращает все строки целевой таблицы;

execute as user = 'Student_one' – возвращает только строки с оценками пользователя Student_one;

execute as user = 'Student_five' – возвращает только строки с оценками пользователя Student_five.

Выключение политики безопасности уровня строк таблицы:

```
ALTER SECURITY POLICY StudentsFilter WITH (STATE = OFF)
```

Пример 3.2. Разграничение доступа к строкам таблицы успеваемости студентов через хранимое представление (без применения RLS)

Согласно бизнес-модели, описанной в предыдущем примере, создана нормализованная база данных, состоящая из пяти соответственно взаимосвязанных таблиц: Students, Parents, Techers, Disciplines и Grades, заполненных аналогично: восемь студентов – детей четырех родителей, получили различные оценки по восьми дисциплинам у четырех преподавателей.

Создано хранимое представление, формирующее выборку всех оценок, полученных всеми студентами по всем дисциплинам у всех преподавателей:

```
CREATE VIEW All_Grades_view AS
  SELECT S.Student as  Студент,  D.Discipline as Дисциплина,
    G.Grade as Оценка,  T.Teacher as Преподаватель,
    P.Parent as [Родитель Студента]
  FROM (((Parents P JOIN Students S ON P.ParentID=S.ParentID)
    JOIN Grades G ON S.StudentID=G.StudentID)
    JOIN Disciplines D ON G.DisciplineID=D.DisciplineID)
    JOIN Teachers T ON T.TeacherID=D.TeacherID
```

По аналогии с предыдущим примером в базе данных созданы 17 пользователей (один менеджер, восемь студентов, четыре преподавателя и четыре родителя), каждому из которых выдано разрешение GRANT SELECT на представление All_Grades.

Возможность разграничения доступа пользователей к строкам хранимого представления иллюстрируется следующим SQL-кодом:

```
EXECUTE AS user='Student_one';
SELECT * from All_Grades_view
WHERE Студент = USER_NAME() OR USER_NAME()='Manager';
REVERT;
EXECUTE AS user='Teacher_one';
SELECT * from All_Grades_view
WHERE Преподаватель = USER_NAME() OR USER_NAME()='Manager';
REVERT;
EXECUTE AS user='Parent_one';
SELECT * from All_Grades_view
WHERE [Родитель Студента] = USER_NAME() OR USER_NAME()='Manager';
REVERT;
EXECUTE AS user='Manager';
SELECT * from All_Grades
WHERE Студент = USER_NAME() OR USER_NAME()='Manager';
REVERT;
```

Приведенный пример вряд ли имеет практическую ценность, так как все пользователи получают право чтения хранимого представления, и ничто не мешает любому из них выполнить Select * from All_Grades.

Пример 3.3. Разграничение доступа к строкам таблицы успеваемости студентов через хранимую процедуру (без применения RLS).

В базе данных, рассмотренной в предыдущем примере, дополнительно создана следующая хранимая процедура, в которой организована фильтрация строк, выбираемых из хранимого представления по условию совпадения имени пользователя и имени персоны (студента, его родителя или преподавателя изучаемой студентом дисциплины) со значением соответствующего столбца (за исключением пользователя Manager, для которого выборка производится без фильтрации строк).

```
CREATE PROCEDURE [dbo].[All_Grades_proc] AS
BEGIN
  DECLARE @username nvarchar(32)
  SET @username = user_name()
  IF @username ='Manager' select * from All_Grades_view
ELSE
  IF @username in (select Студент from All_Grades_view)
      select * from All_Grades_view  where Студент =  @username
ELSE
  IF @username in (select Преподаватель from All_Grades_view)
      SELECT * FROM All_Grades_view
      where Преподаватель = @username
ELSE
  IF @username IN (SELECT [Родитель Студента]
                        FROM All_Grades_view)
      SELECT * FROM All_Grades_view
      WHERE [Родитель Студента]= @username
END
```

Каждому пользователю выдано разрешение GRANT EXECUTE на выполнение этой процедуры.

Для проверки рассмотренного выше метода фильтрации строк хранимого представления через доступ к нему из хранимой процедуры можно использовать следующий SQL-код:

```
use [RLS-test_Studies]
-- выборка данных об успеваемости студента "student_one"
EXECUTE AS user = 'stuent_one'
EXEC All_Grades_proc
REVERT;
-- выборка данных об успеваемости студентов
-- по дисциплинам преподавателя "teacher_four"
EXECUTE AS user = 'teache_four'
EXEC All_Grades_proc
REVERT;
-- выборка данных об успеваемости студентов –
-- детей родителя "Parent_two"
EXECUTE AS USER = 'parent_two'
EXEC All_Grades_proc
REVERT;
-- выборка данных об успеваемости всех студентов
-- по всем дисциплинам
EXECUTE AS user = 'manager'
EXEC All_Grades_proc
REVERT;
```

3.3 Контрольные вопросы

1) Какие задачи защиты данных решаются средствами RLS и в каких ситуациях RLS будет эффективной?
2) Определите базовые понятия RLS:
   - предикат безопасности;
   - предикат фильтрации;
   - предикат блокировки;
   - политика безопасности.
3) Как включается и отключается политика безопасности?

3.4 Практические задания

Задания *базового уровня* сложности

Задание 3.1 Опишите типовую последовательность этапов подготовки к реализации RLS для случая использовании предиката фильтрации.

Задание 3.2 Приведите пример SQL-кода для создания *предиката безопасности* уровня строк таблиц.

Задание 3.3 Приведите пример SQL-кода для создания *политики безопасности* уровня строк таблиц.

Задания *продвинутого уровня* сложности

Задание 3.4 Реализуйте стандартными средствами RLS разграничение доступа врачей и медицинских сестер к информации о пациентах отделения лечебного учреждения: заведующий отделением имеет доступ к информации обо всех пациентах, лечащие врачи и медицинские сестры – только к информации о пациентах, которых они лечат.

Задание 3.5 Выполните задание 3.4 без применения средств RLS.

Задание 3.6 Реализуйте стандартными средствами RLS разграничение доступа сотрудников компании к информации о клиентах: начальник отдела по работе с клиентами имеет доступ к информации обо всех клиентах, менеджеры – только к информации о клиентах, которых они обслуживают.

Задание 3.7 Выполните задание3.6 без применения средств RLS.

Задание 3.8 Реализуйте стандартными средствами RLS разграничение доступа руководителей компании к информации о подчиненных им сотрудниках: директор компании имеет доступ к информации обо всех сотрудниках, начальники отделов – только к информации о сотрудниках своих отделов, а сотрудники (в том числе и начальники) – только к своей персональной информации.

Задание 3.9 Выполните задание 3.8 без применения средств RLS.

3.5. Информационные источники

3.1. Безопасность на уровне строк. URL - https://learn.microsoft.com/ru-ru/sql/relational-databases/security/row-level-security

Тема 4. Хеширование данных

4.1 Хеш-функции и алгоритмы хеширования

Хеширование *данных* – это их обработка специальной *хеш-функцией*[4], обладающей следующими (полезными для задач идентификации и индексации) свойствами:

– хеш-функция является *детерминированной*, то есть возвращает всегда одинаковый результат (*хеш*) для одинаковых входных значений ключа (блока хешируемых данных);

– идеальная хеш-функция осуществляет *преобразование без коллизий*, то есть гарантирует крайне маловероятную возможность получения одинаковых хешей для различных ключей;

– даже при минимальном изменении значения ключа (например, при изменении только одного бита в 32-байтовом блоке данных, как показано в одном из приведенных ниже примеров) хеш-функция возвращает радикально отличающийся хеш;

– хеш-функция *не является вычислительно сложной*, что существенно в задачах индексации и поиска индексированных данных;

– хеш-функция всегда *возвращает значение постоянной длины* (например, 256 бит для SHA-256), независимо от длины входного ключа, что позволяет строить экономичные индексные структуры для данных большой длины;

– хеш-функция *необратима*, то есть нельзя получить ключ из его хеша.

Результат хеширования объекта данных называют его *хеш-значением* (*hash-value*), или просто *хешем* (*hash*). Программные приложения могут использовать различные хеш-функции для различных целей: например, для идентификации, аутентификации и хранения паролей, контроля целостности данных, сравнения с сохраненными хешами вредоносных файлов, для верификации распределенных транзакций и др.

СУБД используют хеширование для целей идентификации объектов хранения и субъектов доступа к данным, а также для индексации данных (как правило, для индексации таблиц по столбцам длинных строковых типов, когда классические B^+-tree-индексы оказываются малоэффективными).

MS SQL Server использует функцию HASHBYTES() с алгоритмами MD2, MD4, MD5, SHA, SHA1, SHA2-256 или SHA2-512. Начиная с версии SQL Server 2016, все алгоритмы, кроме двух последних, считаются устаревшими – они могут использоваться, но вызовут диагностическое сообщение.

---

[4] Свое название эти функции получили от англ. *hash* – *перемешивать*, другое их название – «функции свертки». Существует множество различных хеш-функций (алгоритмов хеширования), специально разработанных для различных областей и целей их использования, идентификация и индексация в базах данных – одна из таких областей.

4.2 Примеры использования Хеш-функций

Пример 4.1. Хеширование значения переменной строкового типа

```
DECLARE @Hash_Input NVARCHAR(32);
SET @Hash_Input_1 = CONVERT(NVARCHAR(32),'qwertyasdfgh1234');
SET @Hash_Input_2 = CONVERT(NVARCHAR(32),'Qwertyasdfgh1234');
SELECT HASHBYTES('SHA2_256', @Hash_Input_1);
SELECT HASHBYTES('SHA2_256', @Hash_Input_1);
SELECT HASHBYTES('SHA2_256', @Hash_Input_2);
```

Результат:

```
0x5DA5DD4D8DBED56397435D8E7B058AB464A977C6B6B6AC03E0E22ED449A614FF
0x5DA5DD4D8DBED56397435D8E7B058AB464A977C6B6B6AC03E0E22ED449A614FF
0x5413DCD9DF34877561B7813CA1B04BCD84CD6A91C514DB473CADA14E8D4319FF
```

Пример 4.2. Выборка хеш-значения столбца таблицы

```
CREATE TABLE HashTable_1 (c1 NVARCHAR(32));
INSERT INTO HashTable_1 VALUES ('Input_data_one');
INSERT INTO HashTable_1 VALUES ('Input_data_two');
SELECT c1 FROM HashTable_1;
SELECT HASHBYTES('SHA2_256', c1) as 'hash_c1' FROM HashTable_1;
```

Результат:

c1
Input_data_one
Input_data_two

hash_c1
0x91C51D98FE200E103E86771BB63C20CD937CF8E937EB8C91119EAAC87C2A3B63
0xB2018A442C09B175BEFC28F8451C6D7B0383962629D0A50F5F6C2F165FD73A63

Пример 4.3. *Вставка хеш-значения в таблицу*

```sql
CREATE TABLE HashTable (c1 NVARCHAR(32), hash_c1 NVARCHAR(32));
DECLARE @Hash_Input_one NVARCHAR(32);
DECLARE @Hash_Input_two NVARCHAR(32);
SET @Hash_Input_one = CONVERT(NVARCHAR(32),'Input_data_one');
SET @Hash_Input_two = CONVERT(NVARCHAR(32),'Input_data_two');
INSERT INTO HashTable(c1,hash_c1)
 VALUES (@Hash_Input_one,HASHBYTES('SHA2_256',@Hash_Input_one));
INSERT INTO HashTable(c1,hash_c1)
 VALUES (@Hash_Input_two,HASHBYTES('SHA2_256',@Hash_Input_two));
SELECT c1, hash_c1 FROM HashTable;
```

Результат:

c1	hash_c1
Input_data_one	양頰⍰□壤▯氳솔粓⍰⍰醌鷔줵⍰挻
Input_data_two	⍰绝ব疱⍰⍰▯笄范[illegible]켑ঁ江Ω흚挻

4.3 Контрольные вопросы

1) Что называют *хеш-функциями* и каковы их основные свойства?
2) Что называют *хеш-значением* (или просто *хешем*)?
3) Что представляют собой входные данные хеш-функций и результаты их преобразования?
4) Как влияет длина входного параметра хеш-функции на длину генерируемого хеш-значения?
5) Что называют *коллизиями* при использовании хеш-функций?
6) В каких областях и с какими целями используются методы хеширования данных?
7) Приведите примеры используемых хеш-функций и дайте им краткую характеристику.

4.4 Практические задания

Задания *продвинутого уровня* сложности

Задание 4.1 Используя информацию открытых источников, дайте характеристику одному из алгоритмов хеширования, опишите этот алгоритм и приведите области его применения:

- MD2, MD4, MD5;
- SHA, SHA_1;
- SHA_2-256, SHA_2-512;
- SHA_3;
- любой другой алгоритм хеширования.

Задание 4.2 Используя приведенные выше примеры, подготовьте хранимую SQL-процедуру для моделирования процесса идентификации и аутентификации пользователя согласно следующему сценарию:

- при создании пользователя его учетные данные (*имя входа* и *пароль* в строковом формате) сохраняются в проверочной таблице базы данных в хешированном виде;
- при вызове процедуры ей передаются *имя входа* и *пароль* в качестве входных параметров;
- процедура хеширует значения этих параметров и сравнивает их хеши с данными, хранимыми в проверочной таблице:
  - если хеш-значение полученного *имени входа* отсутствует в проверочной таблице, процедура выдает сообщение об ошибке идентификации и заканчивает работу;
  - если хеш-значение полученного *пароля* не соответствует хеш-значению полученного *имени входа* в проверочной таблице, процедура выдает сообщение об ошибке аутентификации и заканчивает работу;
  - в противном случае процедура выдает сообщение об успешной аутентификации и заканчивает работу.

4.5 Информационные источники

4.1 Функция HASHBYTES(). URL: https://learn.microsoft.com/ru-ru/sql/t-sql/functions/hashbytes-transact-sql?view=sql-server-ver17

Тема 5. Динамическое маскирование данных

Dynamic Data Masking (DDM) позволяет скрыть от непривилегированных пользователей реальные значения конфиденциальных данных, но при этом, в отличие от шифрования данных, сами хранимые данные не модифицируются, а только «маскируются» при их выводе. Реальные данные, полученные выборкой из таблицы, представляются пользователю замаскированными в соответствии с определенными для этих данных «масками».

5.1 DDM-функции и примеры их использования

Пример 5.1. Маскирование данных столбцов таблиц

Столбец таблицы можно замаскировать, если при создании (CREATE) или модификации (ALTER) схемы таблицы применить к нему функцию, задающую правило маскирования. Всего имеется пять типов масок (SQL Server 2022) и, соответственно, пять функций маскирования (таблица 5.1).

Таблица 5.1 – Функции маскирования данных

Функция	Правило маскирования	Примеры использования
default()	Полное маскирование столбца: XXX…X – для строк; число 0 – для чисел; 1900-01-01 и 00:00:00,0000000 – для дата-временных типов; цифра «0» (ASCII) – для бинарных типов	CREATE TABLE: … <имя столбца> varchar(32) MASKED WITH (FUNCTION = ‘default()’) ALTER TABLE: ALTER COLUMN <имя столбца> ADD MASKED WITH (FUNCTION = ‘default()’)
email()	Маскируются XXX@XXX все символы адреса кроме первого символа и суффикса .com	CREATE TABLE: … <имя столбца> varchar(32) MASKED WITH (FUNCTION = ‘email()’) ALTER TABLE: ALTER COLUMN <имя столбца> ADD MASKED WITH (FUNCTION = ‘email()’)
random()	Данные любого числового типа маскируются случайным значением в заданном диапазоне	CREATE TABLE: … <имя столбца> bigint MASKED WITH (FUNCTION = ‘random (1,1000)’) ALTER TABLE: ALTER COLUMN month ADD MASKED WITH (FUNCTION = ‘random (1,12)’)
partial()	Часть исходной строки (кроме префикса и суффикса заданных размеров) маскируется пользовательской строкой заполнения	CREATE TABLE: … <имя столбца> varchar(32) MASKED WITH (FUNCTION = ‘partial (3,’qwerty’,2)’) ALTER TABLE: ALTER COLUMN name ADD MASKED WITH (FUNCTION = ‘partial (3,’xxxx’,0)’)
datetime()	Маскируется указанная часть данных дата-временных типов (Y, M, D, h, m, s)	CREATE TABLE: … Birthday datetime MASKED WITH (FUNCTION = ‘datetime (“Y“)’) ALTER TABLE: ALTER COLUMN Birthday ADD MASKED WITH (FUNCTION = ‘datetime (“M”)‘)

Пример 5.2. Просмотр информации о замаскированных столбцах

Для просмотра списка замаскированных столбцов таблиц базы данных с указанием имен соответствующих функций маскирования можно использовать системные представления sys.masked_columns и sys.tables:

```
SELECT tbl.name as table_name,
        c.name as column_name,
        c.is_masked,
        c.masking_function
FROM sys.masked_columns AS c
        JOIN sys.tables AS tbl  ON c.object_id = tbl.object_id
WHERE is_masked = 1;
```

Созданную маску можно удалить:

```
ALTER TABLE <имя таблицы>

        ALTER COLUMN <имя столбца> DROP MASKED;
```

Пример 5.3. Выдача и отзыв разрешения UNMASK

По умолчанию пользователи (или пользовательские роли), имеющие разрешение типа SELECT на таблицу с маскированными столбцами, при просмотре строк таблицы будут видеть в этих столбцах замаскированные данные. Для того чтобы пользователь (или роль) мог просматривать замаскированные данные в немаскированном виде, он должен получить разрешение UNMASK на объект соответствующего уровня (базу данных, схему базы данных, таблицу или столбец таблицы):

```
GRANT UNMASK ON <таблица> (<столбец>) TO <пользователь/роль>;

GRANT UNMASK ON <таблица> TO <пользователь/роль>;

GRANT UNMASK ON SCHEMA:: <схема> TO <пользователь/роль>;

GRANT UNMASK TO <имя базы данных>;
```

Отзыв разрешения типа UNMASK:

```
REVOKE UNMASK
ON <таблица> (<столбец>) FROM <пользователь/роль>;

REVOKE UNMASK ON <таблица> FROM <пользователь/роль>;

REVOKE UNMASK ON SCHEMA:: <схема> FROM <пользователь/роль>;

REVOKE UNMASK FROM <имя базы данных>;
```

5.2 Контрольные вопросы

1) Какие задачи защиты данных решаются средствами DDM и в каких ситуациях DDM будет эффективным?
2) Какие алгоритмы и ключи шифрования применяются в DDM?
3) Пользователь имеет разрешение типа SELECT на таблицу с маскированными столбцами. Будет ли такой пользователь при просмотре строк таблицы видеть исходные данные замаскированных столбцов?
4) Пользователь имеет разрешение типа SELECT на маскированные столбцы таблицы. Будет ли такой пользователь при просмотре строк таблицы видеть исходные данные замаскированных столбцов?
5) Для чего используются разрешение типа UNMASK?
6) Когда будет целесообразным использование DDM-функций default(), email(), random(), partial()?

5.3 Практические задания

Задания *начального уровня* сложности

Задание 5.1. Приведите примеры маскировки и демаскировки столбца таблицы пользовательской базы данных и экспериментально подтвердите полученные результаты.

Задание 5.2. Приведите пример просмотра списка замаскированных столбцов таблиц пользовательской базы данных.

Задание 5.3. Приведите примеры выдачи и отзыва разрешений пользователю базы данных на просмотр замаскированного столбца таблицы, всех замаскированных столбцов таблицы, всех таблиц базы данных, содержащих замаскированные столбцы.

Задание 5.4. Приведите примеры выдачи и отзыва разрешений группе пользователей базы данных на просмотр замаскированного столбца таблицы, всех замаскированных столбцов таблицы, всех таблиц базы данных, содержащих замаскированные столбцы.

Задания *базового уровня* сложности

Задание 5.5. Экспериментально исследуйте приоритетность иерархии разрешений на просмотр замаскированных данных.

5.4 Информационные источники

5.1 Динамическое маскирование данных. URL: https://learn.microsoft.com/ru-ru/sql/relational-databases/security/dynamic-data-masking?view=sql-server-ver17

Тема 6. Шифрование данных

Шифрование не решает проблем ограничения доступа к данным, но если злоумышленник все же смог преодолеть установленные защитные ограничения и получил несанкционированный доступ к объекту базы данных, то шифрование исключит (или крайне затруднит) для него возможность интерпретации нелегально полученных конфиденциальных данных. При этом для легальных пользователей конфиденциальные данные будут оставаться и доступными, и интерпретируемыми.

Принимая решение о необходимости шифрования данных, следует понимать, что шифрование может приводить к потере производительности, при этом зашифрованные данные (независимо от типа исходных незашифрованных данных) будут храниться как varbinary(), что исключает возможности их сортировки, фильтрации и индексации.

6.1 Иерархия средств шифрования

MS SQL Server поддерживает иерархическую инфраструктуру управления шифрованием (рисунок 6.1): симметричные и асимметричные ключи различных уровней, сертификаты, подписи.

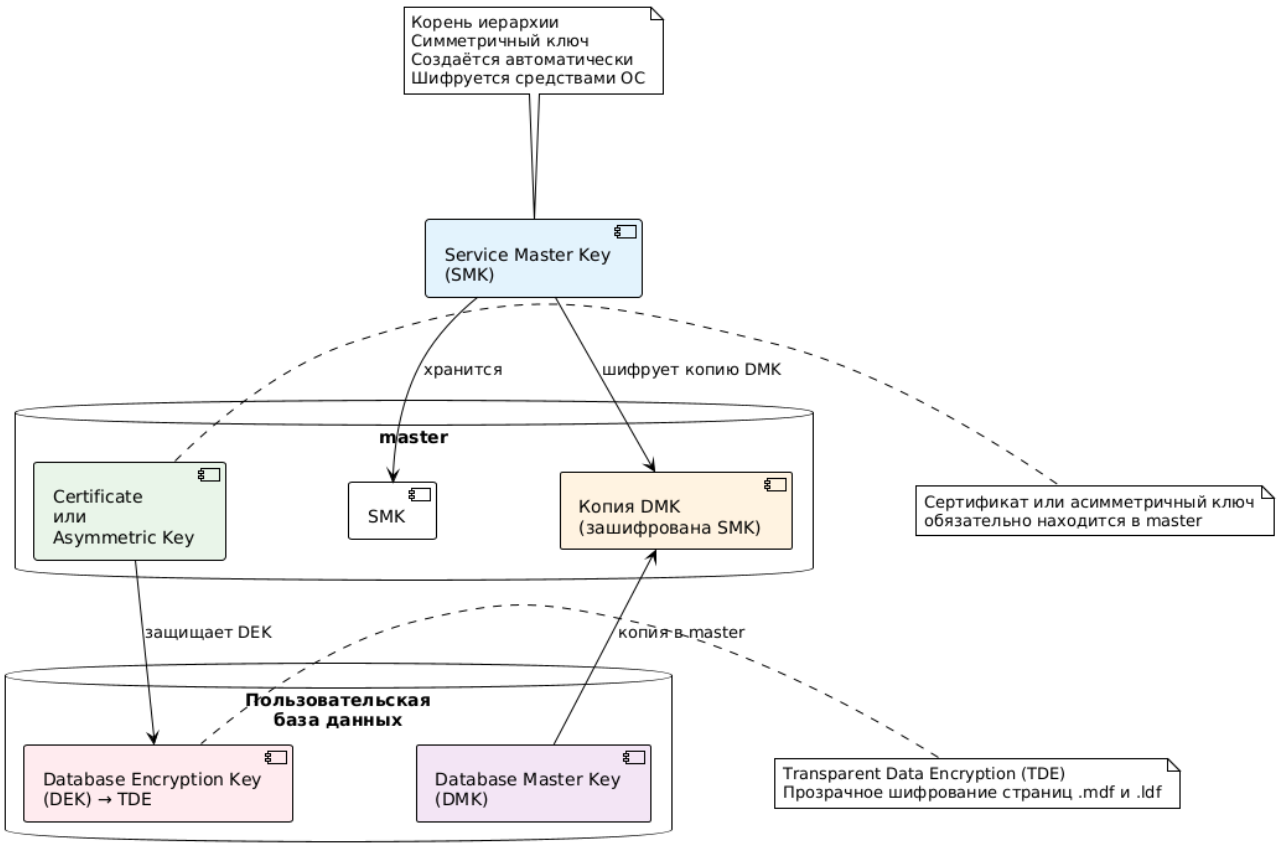


Рисунок 6.1 – Иерархия средств шифрования данных

Уровень 0. Корнем иерархии является *главный ключ службы* (Service Master Key, SMK) – симметричный ключ, создаваемый автоматически при первом запуске экземпляра MS SQL Server. SMK хранится в системной базе данных Master, шифруется средствами ОС (Windows Data Protection API) и используется для шифрования главного ключа базы данных.

Уровень 1. *Главный ключ базы данных* (Database Master Key, DMK) – это симметричный ключ (AES-256), защищённый паролем пользователя и дополнительной копией через SMK. DMK автоматически создается в системных базах данных и должен быть создан «вручную» в каждой пользовательской базе данных:

```
USE My_DB
CREATE MASTER KEY ENCRYPTION
BY PASSWORD='Super_Strong_Passwrd;
```

DMK хранится в базе данных, в которой создан, а его копия – в системной базе данных Master. Основная задача DMK – защита сертификатов, симметричных и асимметричных ключей, используемых в базе данных.

Уровень 2. Сертификаты (X.509) и асимметричные ключи (алгоритмы шифрования RSA-2048/3072/4096, ECDSA). Используются для защиты ключей шифрования нижележащих уровней (DEK для TDE).

```
CREATE CERTIFICATE Cert_Enc WITH SUBJECT='Certificate for Encryption';
CREATE ASYMMETRIC KEY AsymKey_RSA WITH ALGORITHM = RSA_2048;
```

Уровень 3. Симметричные ключи шифрования (алгоритмы шифрования AES-128/192/256) – основной инструмент шифрования/дешифрования конфиденциальных данных столбцов таблиц базы данных.

```
CREATE SYMMETRIC KEY SymKey_AES
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE Cert_Enc;
```

Уровень 4. *Ключ шифрования базы данных* (Database Encryption Key, DEK) используется для *прозрачного шифрования* (Transparent Data Encryption, TDE) базы данных на файловом уровне (называемого также *неактивным шифрованием*). TDE шифрует ввод-вывод в реальном времени: каждая файловая страница data-файлов и log-файлов базы данных автоматически шифруется при записи из буфера ОЗУ в соответствующий файл и автоматически дешифруется при загрузке в буфер ОЗУ из файла.

DEK – это симметричный ключ, он создается и хранится в пользовательской базе данных и может быть защищен сертификатом или асимметричным ключом. Сертификат, используемый для защиты DEK, создается и хранится в системной БД MASTER.

```
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_256
ENCRYPTION BY SERVER CERTIFICATE Cert_Enc;
```

6.2 Функции шифрования / дешифрования

MS SQL Server предоставляет функции (таблица 6.1) для реализации шифрования и дешифрования данных на каждом уровне иерархии.

Таблица 6.1 – Функции шифрования данных

Симметричное шифрование / дешифрование	Асимметричное шифрование / дешифрование
encryptbykey() decryptbykey() encryptbypassphrase() decryptbypassphrase() key_id(), key_guid() key_name() decryptbykeyautoasymkey() symkeyproperty() DecryptByKeyAutoCert()	encryptbyasymkey() decryptbyasymkey() encryptbycert() decryptbycert() asymkeyproperty() asymkey_id()
Управление подписями	Копирование сертификатов
signbyasymkey() verifysignedbyasymkey() signbycert() verifysignedbycert() is_objectsigned()	certencoded() certprivatekey()

6.3 Примеры

Пример 6.1. Создание главного ключа базы данных (DMK)

Для создания DMK требуется выполнить следующий SQL-код в контексте пользовательской базы данных:

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = ‘any password’;
```

Пример 6.2. Управление ключами прозрачного шифрования базы данных.

```
Создание DEK:
  CREATE DATABASE ENCRYPTION KEY
     WITH ALGORITHM = { AES_128 | AES_192 | AES_256 }
  ENCRYPTION BY SERVER   {
      CERTIFICATE Encryptor_Name |  ASYMMETRIC KEY Encryptor_Name
   } ;
Удаление DEK:       DROP DATABASE ENCRYPTION KEY
Изменение DEK:     ALTER DATABASE ENCRYPTION KEY
```

Пример 6.3. Управление режимом прозрачного шифрования БД

Для управления режимами TDE (включением и выключением, приостановкой и возобновлением) используется следующая SQL-команда:

```
ALTER DATABASE
SET ENCRYPTION { ON | OFF | SUSPEND | RESUME }
```

Для просмотра информации о режиме TDE можно использовать следующие системные хранимые представления: sys.databases, sys.certificates и sys.dm_database_encryption_keys.

Пример 6.4. Шифрование данных столбцов таблиц

Шифрование столбцов таблицы симметричным ключом:

```
CREATE CERTIFICATE Cert_Name WITH SUBJECT = ‘Sert_owner’;
CREATE SYMMETRIC KEY Key_Name WITH ALGORITHM = AES_256
      ENCRYPRION BY CERTIFICATE Cert_Name;
CREATE TABLE Table_with_secret_data (
      Id INT IDENTITY PRIMARY REY, Name VARCHAR(128),
      Secret_data VARCHAR(MAX));
OPEN SYMMETRIC KEY Key_Name DECRYPTION BY CERTIFICATE Cert_Name;
```

```
INSERT INTO Table_with_secret_data (Id, Name, Secret_data)
  VALUES
  (name_one, EncryptByRey(Key_GUID(Key_Name,' secret_data_one'))),
  (name_two, EncryptByRey(Key_GUID(Key_Name,' secret_data_two'))),
   …
   …
   (name_six, EncryptByRey(Key_GUID(Key_Name,' secret_data_six')));
CLOSE SYMMETRIC KEY Key_Name;
```

Просмотр строк таблицы
(информация зашифрованного столбца недоступна):

```
SELECT name, Secret_data FROM Table_with_secret_data;
```

Просмотр строк таблицы
с дешифрованием данных зашифрованного столбца:

```
OPEN SYMMETRIC KEY Key_Name DECRYPTION BY CERTIFICATE Cert_Name;
SELECT name, DecryptByKey(Secret_data)
    FROM Table_with_secret_data;
CLOSE SYMMETRIC KEY Key_Name;
```

6.4 Контрольные вопросы

1) Дайте определение *сертификата* (как оно трактуется в MS SQL Server). Для чего используются сертификаты? Приведите примеры их создания и использования средствами SQL.
2) Дайте определения симметричного и асимметричного шифрования. Как они используются в инфраструктуре шифрования MS SQL Server? Приведите примеры.
3) Какие алгоритмы шифрования используются в инфраструктуре шифрования MS SQL Server (2022)? Приведите примеры.
4) Дайте определение *главного ключа базы данных* (DMK).
5) Что называют *прозрачным* (или *неактивным*) *шифрованием* (Transparent Data Encryption, TDE)?
6) Дайте определение *ключа шифрования базы данных* (DEK). Приведите примеры его создания, изменения и использования.

6.5 Практические задания

Задания *продвинутого уровня сложности*

Задание 6.1 Рассмотрите схему иерархической инфраструктуры управления шифрованием, реализованную в MS SQL Server, и опишите состав задач, методов и инструментов, используемых на каждом уровне иерархии.

Задание 6.2 Приведите примеры использования элементов системного каталога базы данных для просмотра информации о созданных сертификатах и ключах шифрования.

Задание 6.3 Приведите пример создания и использования симметричного ключа для шифрования / дешифрования нескольких столбцов таблицы.

Задание 6.4 Приведите пример создания и использования асимметричного ключа для шифрования / дешифрования нескольких столбцов таблицы.

Задание 6.5 Приведите примеры включения и отключения режима TDE в пользовательской базе данных, экспериментально подтвердите факт шифрования data-файла базы данных.

6.6. Информационные источники

6.1 Андресс Дж. Защита данных. От авторизации до аудита. Санкт-Петербург : Питер, 2021. 272 с.: ил.

6.2 Иерархия средств шифрования. URL: https://learn.microsoft.com/ru-ru/sql/relational-databases/security/encryption/encryption-hierarchy

Волк Владимир Константинович

ЗАЩИТА ДАННЫХ В РЕЛЯЦИОННЫХ СУБД

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

ДЛЯ СТУДЕНТОВ
ИТ-СПЕЦИАЛЬНОСТЕЙ

Редактор Н. М. Быкова

Библиотечно-издательский центр КГУ
640020, г. Курган, ул. Советская, 63/4.
Курганский государственный университет