БЕЗОПАСНОСТЬ СИСТЕМ БАЗ ДАННЫХ

4. Управление доступом в СУБД

Управление доступом в СУБД

- Процедура и/а выполняет функцию глобального контроля прав пользователя по доступу в систему в целом.
- Однако использование даже самых надежных процедур и/а в большинстве случаев оказывается недостаточным для решения вопроса о допустимости выполнения любых операций с конкретными данными.
- В БД каждое групповое данное должно иметь связанный с ним (данным) ***набор ограничений доступа***.
- Процедуры регулирования допустимых действий пользователей по отношению к различным данным в БД называют ***авторизацией*** (от англ. authorization – разрешение, уполномочивание) или ***управлением доступом*** (или ***контролем доступа***).

Субъекты и объекты доступа

- При управлении доступом в СУБД различают два вида сущностей – ***активные*** и ***пассивные***. Активные сущности (***субъекты***) осуществляют доступ к пассивным сущностям – ***объектам***.
- Субъекты – это пользователи или приложения.
- ***Первичные объекты***, к которым регулируется доступ:
  - БД в целом;
  - отношения (таблицы);
  - представления;
  - процедуры.

Вторичные объекты доступа

- ***Вторичными объектами*** доступа могут быть:
  - формы;
  - отчеты;
  - запросы;
  - прикладные программы (ПП);
  - процедуры;
  - диаграммы и т.п.
- В большинстве типов СУБД возможность для субъекта пользоваться вторичными объектами определяется правами его доступа к соответствующим первичным объектам.

Требования по безопасности в реляционных СУБД

- Разные пользователи могут иметь доступ только к строго определенным совокупностям таблиц;
- Разные пользователи могут иметь право на строго определенный перечень операций над данными конкретных таблиц;
- Разные пользователи могут иметь только выборочный доступ к определенным элементам конкретных таблиц (отдельным строкам и столбцам);
- Доступ определенных пользователей к определенной совокупности таблиц может быть запрещен непосредственно (через запросы), но разрешен в диалоге с прикладной программой.

Схема доступа к данным в реляционных СУБД

- СУБД от имени конкретного субъекта выполняет конкретные операции над объектами БД в зависимости от прав, предоставленных субъекту на выполнение этих операций.
- Доступом к объектам БД можно управлять – разрешать доступ или защищать от доступа.
- Конкретный субъект обладает конкретными правами доступа к конкретному объекту.
- Совокупность операций, которые разрешено выполнять конкретному субъекту над конкретными объектами определяют ***привилегии*** (priveleges).

Категории пользователей СУБД

- В больших многопользовательских БД пользователи СУБД обычно разбиты на 3 категории:
  - *администратор сервера БД* - ведает установкой, конфигурированием сервера, регистрацией пользователей, групп, ролей и т. п.;
  - *администратор БД* - пользователь, создавший свою БД, и являющийся ее владельцем - предоставляет другим пользователям доступ к своей БД и к содержащимся в ней объектам, отвечает за ее сохранение и восстановление;
  - *прочие* (конечные) пользователи - оперируют данными БД, в рамках выделенных им привилегий.

Первоначальная учетная запись для администраторов БД

- Как правило, для администраторов БД сделана первоначальная учетная запись, чтобы сделать первоначальный вход в систему.
- Например,
  - в InterBase: SISDBA с masterkey;
  - в SQL-server: SA и пустой пароль;
  - в Oracle есть три начальных учетных записи: SIS, SYSTEM и MANAGER.

Категории пользователей в СУБД ЛИНТЕР

- Пример категорий пользователей в СУБД ЛИНТЕР:
  - *администратор базы данных* – категория DBA. Он управляет созданием БД, ее конфигурированием, регистрацией пользователей, групп, ролей, записью регистрационной информации и т.п.
  - *привилегированные пользователи БД* – категория RESOURCE. Это пользователи, которые имеют право на создание собственных объектов БД и управление привилегиями доступа к ним.
  - *пользователи БД* – категория CONNECT оперируют с объектами БД в рамках выделенных им привилегий доступа.

Виды привилегий в СУБД

- Привилегии в СУБД можно подразделить на две категории:
  - ***привилегии безопасности*** – позволяют выполнять административные действия в БД;
  - ***привилегии доступа*** – определяют права доступа субъектов к объектам БД.
- Привилегии безопасности всегда выделяются конкретному пользователю (а не группе, роли или всем) при его создании (оператором `CREATE USER`) или при изменения его привилегий (оператором `ALTER USER`).
- Набор привилегий безопасности определяется разработчиком СУБД.

Привилегии безопасности в СУБД INGRES

- `security` – право управлять всей безопасностью СУБД (админ сервера БД, а также лицо, отвечающее за ИБ);
- `createdb` – право на создание и удаление БД (адм. сервера БД и адм. отдельных БД);
- `operator` – право на выполнение действий оператора (адм. сервера БД и адм. отдельных БД, админ ОС);
- `maintain locations` – право на управление расположением БД на дисках (адм. сервера БД и адм. ОС);
- `trace` – право на изменение состояния флагов отладочной трассировки СУБД (адм. сервера БД и другие знающие пользователи при анализе сложных ситуаций).

Установление привилегий безопасности

- **Пример.** Заводится пользователь `Bill` с правом на создание БД и на выполнение функций оператора:

  ```
  CREATE USER Bill
  WITH PRIVILEGES createdb, operator;
  ```

  {создать пользователя `Bill` с привилегиями `createdb` и `operator`}.

- Для пользователя можно определить подразумеваемую группу:

  ```
  CREATE USER Mary
  WITH GROUP shoe_grp
  ```

  {создать пользователя `Mary` с подразумеваемой группой}.

Изменение привилегий безопасности

- Все последующие изменения привилегий безопасности выполняются посредством оператора `ALTER USER`. Оператор `DROP USER` позволяет удалить пользователя.
- Примеры:

```
ALTER USER Bill
DROP PRIVILEGES createdb;
```

{лишить пользователя `Bill` права на создание БД}.

```
DROP USER Bill;
```

{удалить пользователя `Bill`}.

Привилегии доступа

- В СУБД авторизация доступа к данным осуществляется с помощью привилегий доступа.
- Установление привилегий доступа зависит от вида управления доступом. Для ДУД – это задача владельца объекта. Для ПУД – это задача админа БД.
- Контроль привилегий доступа, независимо от вида управления доступом, обычно, возлагается на админа БД.
- Создавать и удалять именованные носители привилегий, а также изменять их характеристики может лишь пользователь с привилегией `security`. При этом необходимо иметь подключение к служебной БД, в которой хранятся сведения о субъектах и их привилегиях.

Привилегии доступа

- Набор привилегий можно определить для конкретного пользователя, группы, роли или всех пользователей.
- Объектом защиты может быть БД, таблица, представление, хранимая процедура и т.д.
- Субъектом защиты может быть пользователь, группа пользователей, роль, хранимая процедура (если это предусмотрено реализацией СУБД).
- Привилегии устанавливаются и отменяются операторами языка SQL: `GRANT` (передать) и `REVOKE` (отобрать).
- Оператор `GRANT` указывает конкретного пользователя, который получает конкретные привилегии доступа к указанной таблице.

Установление привилегий

- Конкретный пользователь СУБД опознается по уникальному идентификатору (user-id). Любое действие над БД, любой оператор языка SQL всегда выполняется не анонимно, а от имени конкретного пользователя.
- Идентификатор пользователя определяет набор доступных объектов БД для конкретного физического лица или группы лиц. Для того, чтобы связать идентификатор пользователя с конкретным оператором SQL, в большинстве СУБД используется сеанс работы с БД, в начале которого пользователь сообщает СУБД свой идентификатор и пароль. Все операции над БД, которые будут выполнены после этого, СУБД свяжет с этим конкретным пользователем.

Управление привилегиями

- **Примеры.**
- Назначение привилегии субъекту:

```
GRANT привилегия [ON объект] TO субъект
WITH GRANT OPTION]
```

- Отмена привилегии субъекту:

```
REVOKE привилегия [ON объект] FROM субъект
```

- Назначение привилегии всем пользователям системы :

```
GRANT привилегия [ON объект] TO PUBLIC
```

- Отмена привилегии всем пользователям системы:

```
REVOKE привилегия [ON объект] FROM PUBLIC
```

Управление привилегиями

- Оператор `GRANT` может содержать необязательную часть, принципиально важную для защиты СУБД. Имеется в виду конструкция

```
GRANT ...
...
WITH GRANT OPTION.
```

- Подобный оператор `GRANT` передает не только указанные в нем привилегии, но и права на их дальнейшую передачу. Очевидно, что использование конструкции `WITH GRANT OPTION` ведет к децентрализации контроля над привилегиями и содержит потенциальную угрозу безопасности данных.

Группы

- Для облегчения процесса администрирования большого количества пользователей их объединяют в ***группы*.**
- Применяются три способа сочетания индивидуальных и групповых прав в доступах пользователей:
  - Один и тот же идентификатор используется для доступа к БД целой группы физических лиц.
  - Конкретному физическому лицу присваивается уникальный идентификатор.
  - Одновременно поддерживается идентификатор пользователя и идентификатор группы пользователей.

Группы

- Когда пользователь тем или иным способом инициирует сеанс работы с БД, он может указать, от имени какой из своих групп он выступает. Кроме того, для пользователя обычно определяют подразумеваемую группу.
- Совокупность всех пользователей именуется как `PUBLIC`. Придание привилегий `PUBLIC` – удобный способ задать подразумеваемые права доступа.
- Прежде чем присваивать привилегии группам и ролям, их (группы и роли) необходимо создать.

Создание и изменение групп

- Группы создаются с помощью оператора CRFATE GROUP:

```
CREATE GROUP sales
WITH USERS = (Bill, Mary, Joe);
```

{Создать группу sales с пользователями Bill, Mary, Joe}.

- Для изменения группы служит оператор ALTER GROUP:

```
ALTER GROUP sales
ADD USERS (Austin);
```

{Добавить в группу sales пользователя Austin}

```
ALTER GROUP sales
DROP USERS (Joe);
```

{Удалить из группы sales пользователя Joe}.

Удаление групп

- Оператор `DROP GROUP` позволяет удалять группы, правда только после того, как опустошен список членов группы:

```
ALTER GROUP sales
DROP all;
{Опустошить список членов группы sales}
DROP GROUP sales;
{Удалить группу sales}.
```

Роли в СУБД

- Одна из проблем защиты данных в СУБД возникает по той причине, что с БД работают как прикладные программы (ПП), так и пользователи, которые их запускают.
- Запуск ПП пользователями с различными правами доступа к данным, может приводить к нарушению безопасности в БД.
- Решение этой проблемы заключается в том, чтобы ПП были приданы свои, не связанные с конкретным пользователем, привилегии доступа к объектам БД, которые она обрабатывает.
- При этом админу не нужно каждый раз при запуске пользователями ПП заботиться о соответствии привилегий пользователей и ПП.

Роли в СУБД

- В современных СУБД это решение обеспечивается механизмом ***ролей*** (role). При этом роль выступает в качестве промежуточной сущности между пользователями и их привилегиями. Роль реализуется как отдельный именованный объект, хранящийся в БД.
- Роль связывается с конкретной ПП для придания последней привилегий доступа к БД, таблицам, представлениям и процедурам БД.
- Роль создается и удаляется администратором БД.
- С ролью не ассоциируют перечень допустимых пользователей, - вместо этого роли защищают паролями, если это предусмотрено производителем СУБД.

Роли в СУБД

- Роли также удобно использовать, когда тот или иной набор привилегий необходимо выдать (или отобрать) целой группе пользователей. С одной стороны, это облегчает администратору управление привилегиями, с другой - вносит определенный порядок в случае необходимости изменить набор привилегий для группы пользователей сразу.
- В целом же, механизм ролей позволяет обеспечить приемлемую безопасность систем БД при малых трудозатратах на выдачу и отбор привилегий пользователям.

Создание и удаление ролей

Создание роли с паролем:

```
CREATE ROLE dly sales
WITH PASSWORD = 'e11332'
```

- Оператор `ALTER ROLE` служит для изменения пароля роли:

```
ALTER ROLE dly_sales
WITH PASSWORD = 'h94pq861'
```

- Оператор `DROP ROLE` позволяет удалить ранее созданную роль:

```
DROP ROLE dly_sales.
```

Предоставление роли привилегий

- Как только роль создана, ей можно предоставить привилегии доступа к объектам БД. Если субъект – пользователь, привилегии назначаются явно. Если субъект – роль, то для управления привилегиями используются предложения:

  ```
  GRANT ROLE имя_роли [ON объект] TO
  субъект [WITH GRANT OPTION]
  ```

- **Пример:**
- `GRANT ROLE salesClerk`
- `TO Mary;`
- {Предоставить пользователю `Mary` привилегию роли `salesClerk`}.

Изъятие привилегий роли

- Для изъятия привилегий роли используется предложение:

```
REVOKE ROLE salesClerk
FROM Mary
```

- {Изъять привилегию роли `salesClerk` у пользователя `Mary`}.

Представления

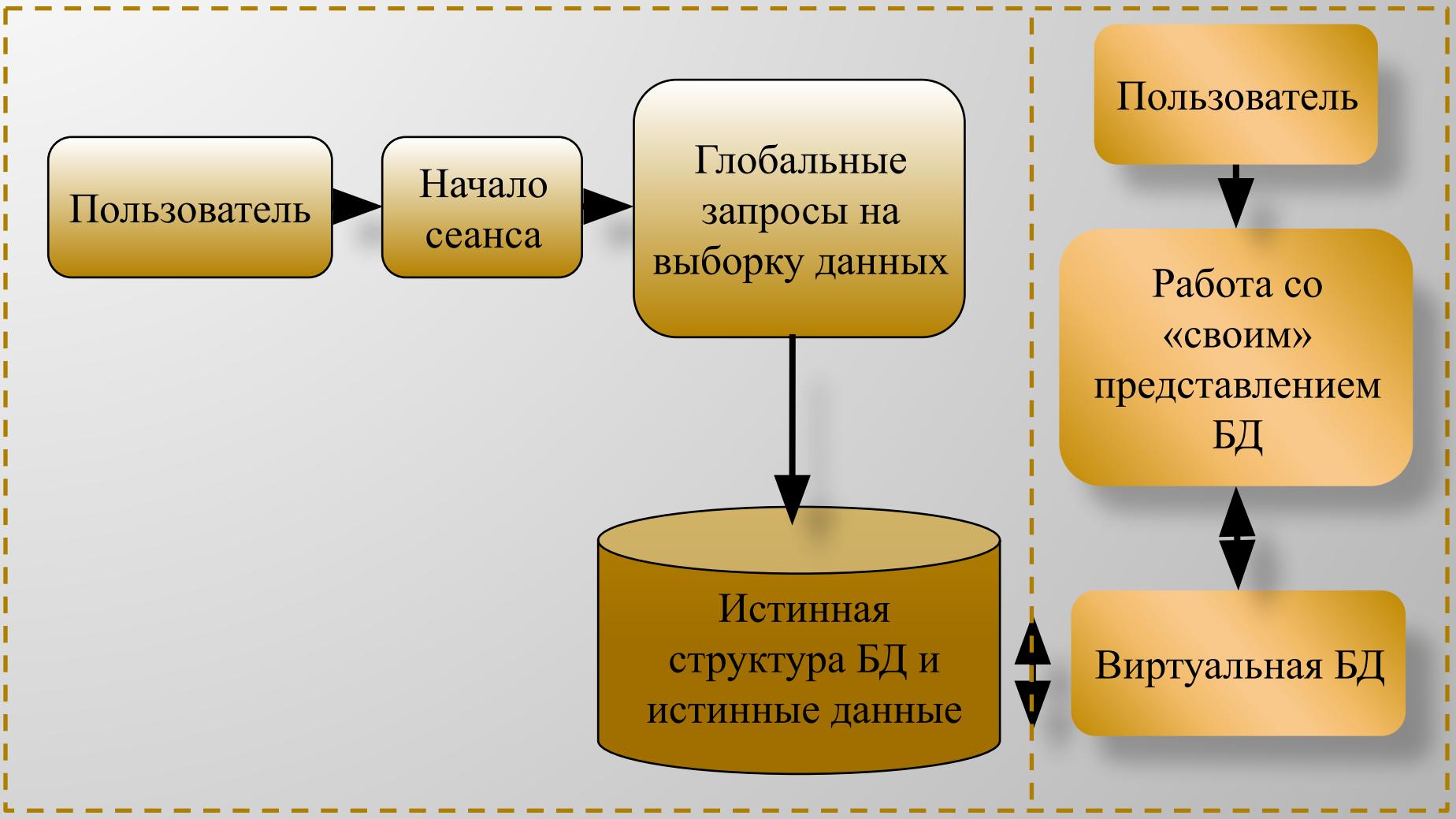
- При использовании дискреционной модели получается, что доступ регулируется на уровне именованных объектов, а не на уровне собственно хранящихся в них данных.
- Так, в реляционной СУБД объектом будет, например, именованное таблица, а субъектом - зарегистрированный пользователь.
- В этом случае нельзя ограничить доступ только к части информации, хранящейся в таблице.
- Частично проблему регулирования доступа к данным внутри объектов решают ***представления*** (Views).

Представления

Представлением (Views) называется сохраняемый в БД авторизованный глобальный запрос на выборку данных.

- Результатом запроса на выборку является набор данных, представляющий временную таблицу, с которой можно работать как с обычной реляционной таблицей.
- В результате серии таких запросов для пользователя создается некая ***виртуальная*** БД со «своей» схемой данных и «своими» данными.
- Представления позволяют сделать видимыми для субъектов только определенные столбцы базовых таблиц (реализовать проекцию) или отобрать определенные строки (реализовать селекцию).

Схема техники представлений



Представления как защита данных в СУБД

- Авторизованный характер запросов, формирующих представления, позволяет предоставить конкретному пользователю только те данные и только в том виде, которые ему необходимы для решения его задач.
- Исключается возможность доступа, просмотра и изменения других данных, не имеющих отношения к данному пользователю.
- С помощью представления администратор БД защищает таблицы от НСД и снабжает каждого пользователя своим видением БД, когда недоступные пользователю объекты как бы не существуют.

Обращение к представлению

- К представлению можно обращаться точно так же, как и к таблицам, за исключением операций модификации данных, поскольку некоторые типы представлений являются не модифицируемыми.
- Возможность модификации представлений определяется производителем СУБД.
- Особенности выполнения операций модификации данных при использовании представлений связаны с ограниченным видением пользователей реальных таблиц, которое в общем случае не позволяет им объективно в рамках выделенных им полномочий проводить операции модификации.

Хранение представлений

- В конкретных реализациях представление обычно хранится как текст, описывающий запрос выборки, а не собственно выборку данных.
- Собственно выборка создается динамически на момент выполнения предложения SQL, использующего представление.
- Таким образом, при использовании представлений данные БД находятся там же где и находились до создания представления.

Привилегии доступа к таблицам

- Привилегии доступа имеют свои особенности для определенных видов объектов, к которым они относятся.
- Привилегии доступа к таблицам позволяют реализовать следующие основные требования по безопасности:
  - разные пользователи могут иметь доступ только к строго определенным совокупностям таблиц;
  - разные пользователи могут иметь право на строго определенный перечень операций над данными конкретных таблиц (создание, чтение, модификация, удаление);

Привилегии доступа к таблицам

- разные пользователи могут иметь только выборочный доступ к определенным элементам конкретных таблиц (отдельным строкам и отдельным столбцам);
- доступ определенных пользователей к определенной совокупности таблиц может быть запрещен непосредственно (через запросы), но разрешен в диалоге с прикладной программой.

- При управлении доступом к таблицам набор привилегий в реализации СУБД определяется производителем.

Привилегии доступа к таблицам

- В качестве минимального набора выступают привилегии выборки и модификации данных:

  `SELECT` – привилегия на выборку данных;
  `INSERT` – привилегия на добавление данных;
  `DELETE` – привилегия на удаление данных;
  `UPDATE` – привилегия на обновление данных

- Дополнительные привилегии изменения структуры таблиц:

  `ALTER` – изменение физической/логической структуры базовой таблицы (изменение размеров и числа файлов таблицы, введение дополнительного столбца и т.п.);

  `INDEX` – создание/удаление индексов на столбцы таблицы;

  `ALL` – все возможные действия над таблицей.

Привилегий доступа к таблицам

- Другие разновидности привилегий:
  - `CONTROL` (IBM DB2) - комплексная привилегия управления структурой таблицы;
  - `REFERENCES` - привилегия создания внешних ключей;
  - `RUNSTAT` - выполнение сбора статистической информации по таблице

и другие.

Примеры предоставления привилегий доступа к таблицам

- Пример предоставления пользователю `Adrian` право на выборку данных из таблицы продавцов:

```
GRANT SELECT
ON salespeople
TO Adrian.
```

- Привилегии можно перечислять через запятую, например:

```
GRANT SELECT, INSERT, DELETE
ON sales
TO Joe, Adrian, Thomas
```

{Предоставить пользователям `Joe, Adrian` и `Thomas` право на выборку, добавление и удаление данных из таблицы `sales`}.

Примеры предоставления привилегий доступа к таблицам

- Если необходимо выделить привилегию не на всю таблицу, а лишь на отдельные столбцы, используют следующий синтаксис:

```
GRANT SELECT (name, address)
ON sales
TO Adrian
```

{Предоставить пользователю `Adrian` право на выборку данных из столбцов `name, address` таблицы `sales`}.

Примеры предоставления привилегий доступа к таблицам

- Пользователям могут выделяться разные привилегии к разным столбцам одной таблицы, например:

```
GRANT SELECT,
UPDATE (empname, empaddress),
DELETE (admission)
ON employee
TO Bill, Maru
```

{Предоставить пользователям `Bill` и `Maru` право на выборку из всей таблицы `employee`, на изменение столбцов `empname` и `empaddress` таблицы `employee` и на удаление столбца `admission` (допуск) той же таблицы `employee`}.

Примеры предоставления привилегий доступа к таблицам

- Обобщающий аргумент `all` заменяет в команде `GRANT` перечень всех возможных привилегий работы с таблицей.

```
GRANT all
ON salesPeople
TO Diane
```

- Аргумент `PUBLIC` имеет аналогичный смысл, но только в отношении пользователей, например:

```
GRANT SELECT
ON phonelist
TO PUBLIC
```

Примеры предоставления привилегий доступа к таблицам

- Иногда возникает ситуация, когда пользователя необходимо не только наделить привилегиями, но и дать возможность передавать эти привилегии другим пользователям. В этом случае используют команду

```
GRANT SELECT
ON salesPeople
TO Adrian
WITH GRANT OPTION
```

{Пользователю `Adrian` разрешено передавать привилегию `SELECT` к таблице продавцов `salesPeople` третьим лицам}.

Примеры предоставления привилегий доступа к таблицам

- Теперь `Adrian` может ввести, например, команду:

```
GRANT SELECT
ON salesPeoples
TO Stephen
WITH GRANT OPTION
```

{Пользователю `Stephen` разрешено передавать привилегию `SELECT` к таблице продавцов `salesPeople` третьим лицам}.

- Понятно, что такой мощной командой нужно пользоваться очень аккуратно, т.к., например, в рассмотренном выше примере количество пользователей, имеющих доступ к таблице, будет расти в геометрической прогрессии.

Пример лишения привилегий доступа к таблицам

- Лишение пользователя привилегий осуществляется командой `REVOKE`. Ее синтаксис аналогичен синтаксису команды `GRANT`, но имеет обратный смысл.

```
REVOKE INSERT
ON salesPeople
FROM Adrian
```

{Лишить пользователя `Adrian` права на добавление данных в таблицу продавцов `salesPeople`}.

Лишение привилегий доступа к таблицам

- Возникает вопрос: кто имеет право отменять привилегии, и что делать в том случае, если пользователь успел передать свои привилегии?
- Это не стандартная ситуация, поэтому нет никаких авторитетных ответов на эти вопросы.
- Наиболее общий подход в такой ситуации следующий: привилегии отменяются пользователем, который их предоставил, и отмена будет каскадироваться, то есть она будет автоматически распространяться на всех пользователям получивших от него эту привилегию.

Использование представлений для фильтрации привилегий

- Всякий раз, когда пользователю выделяется привилегия к базовой таблице, она автоматически распространяется на все строки, а при использовании возможных исключений `UPDATE` и `REFERENCES`, и на все столбцы таблицы.
- Действия привилегий по отношению к базовой таблице можно сделать более точными, используя представления.
- Сначала создается представление, которое ссылается на основную таблицу, а затем уже выделяется привилегия доступа к этому представлению, а не к самой таблице.
- Это значительно улучшает базисные возможности команды GRANT.

Использование представлений для фильтрации привилегий

- Таким образом представления позволяют сделать видимыми для субъектов только определенные столбцы или определенные строки базовых таблиц.
- Сконструировав подходящие представления, администратор БД, по существу, защищает таблицы от несанкционированного доступа и снабжает каждого пользователя своим видением БД, когда недоступные для него объекты как бы не существуют.

Предоставление привилегий доступа к представлениям

- **Пример.** Пусть пользователю `Thomas` необходимо предоставить возможность видеть только столбцы `name` и `address` базовой таблицы продавцов `salesPeople`.

```
CREATE VIEW thomasView
AS SELECT name, address
FROM salesPeople.
```

- Теперь пользователю `Thomas` выделяется привилегия `SELECT` к представлению `thomasView`, а не к самой таблице

```
GRANT SELECT
ON thomasView
TO Thomas.
```

Доступ к представлениям с фильтрацией строк

- Пользователю Adrian предоставить привилегию UPDATE в таблице заказчиков customers, размещенных в Лондоне:

```
CREATE VIEW londoncust
AS SELECT *
FROM customers
WHERE city = 'London'
WITH CHECK OPTION,
```

затем необходимо передать пользователю Adrian привилегию UPDATE к созданному представлению:

```
GRANT UPDATE
ON Londoncust
TO Adrian.
```

Доступ к представлениям с фильтрацией столбцов и строк

- Фильтрацию данных в представлении можно распространить на конкретные столбцы и строки исходной таблицы.
- Пусть всем пользователям необходимо предоставить возможность видеть в исходной таблице сотрудников `employee` только два столбца: фамилию `name` и отдел `dept` и только строки, относящиеся к обувному отделу `shoe`.
- Сначала создаем соответствующее представление:

```
CREATE VIEW empview AS (условие)
SELECT name, dept
FROM employee
WHERE dept = 'shoe'.
```

Доступ к представлениям с фильтрацией столбцов и строк

- Затем предоставляется всем право на выборку из созданного представления:

```
GRANT SELECT
ON empview
TO PUBLIC.
```

- Если при доступе к представлению `empview`, пытаться запросить сведения, относящиеся к отделу игрушек `toy`:

```
SELECT *
FROM empview
WHERE dept = 'toy',
```

то ответ будет из нуля строк, а не код ответа.

Привилегии пользователя при создании представления

- При создании представления пользователь должен иметь привилегию `SELECT` ко всем базовым таблицам, на которые имеются ссылки в создаваемом представлении.
- Если представление – модифицируемое, то пользователь должен иметь привилегии `INSERT`, `UPDATE`, и `DELETE` к базовым таблицам, причем эти привилегии будут автоматически передаваться и к представлению.
- Если пользователь не имеет привилегий на модификацию в базовых таблицах, то он не сможет получить их и в представлениях, которые он создал, даже если сами эти представления – модифицируемые.
- Привилегия `REFERENCES` при создании представлений никогда не используется.

Представления с агрегатными функциями

- Другая возможность состоит в том, чтобы предлагать пользователям доступ к уже извлеченным данным, а не к фактическим значениям в исходной таблице.
- Реализовать эту возможность позволяют агрегатные функции.
- С помощью агрегатных функций можно создавать представление, которое реализует:
  - счет COUNT,
  - среднее значение AVG,
  - общее количество SUM,
  - и др.

Представления с агрегатными функциями

Пример.

```
CREATE VIEW datetotals
AS SELECT odate, COUNT (*), SUM (amt),
AVG (amt)
FROM Orders
GROUP BY odate.
```

- Теперь можно передать пользователю `Diane` привилегию `SELECT` в представлении `datetotals`:

```
GRANT SELECT
ON datetotals
TO Diane.
```

Привилегии доступа к процедурам

- По отношению к *процедуре* можно предоставить лишь право на ее выполнение. При этом не нужно заботиться о выделении прав доступа к объектам, обрабатываемым процедурой – их наличие не обязательно.
- Таким образом процедуры БД являются удобным средством предоставления контролируемого доступа для выполнения строго определенных действий с данными.
- Пример предоставления права выполнения (`EXECUTE`) процедуры `integ check` всем пользователям:

```
GRANT  EXECUTE
ON PROCEDURE integ_check
TO PUBLIC.
```

Привилегии доступа к процедурам

- По умолчанию все пользователи имеют право создавать процедуры в БД. Если бы они при этом автоматически получали права на их выполнение, то смогли бы осуществить, по существу, любую операцию с данными, поскольку выполнение процедуры не требует прав доступа к обрабатываемым объектам.
- Для исключения этой возможности предоставление права на выполнение процедуры, возможна только администратором БД.
- Видно, насколько осторожно нужно относиться к предоставлению привилегий выполнения по отношению к новым, непроверенным процедурам.

Права доступа к БД

- Права доступа к БД как к единому целому может предоставлять только администратор БД или администратор сервера БД. Эти «права» на самом деле устанавливают ряд ограничений на использование БД, т. е. по сути, являются запретительными.
- Так, в СУБД INGRES в правах доступа к БД используются следующие виды ограничений:
  - `QUERY_IO_LIMIT` – предполагаемое максимальное число операций ввода/вывода на один запрос;
  - `QUERY_ROW_LIMIT` – предполагаемое максимальное число строк, возвращаемых одним запросом;

Права доступа к БД

- ограничение на использование ресурсов процессора;
- ограничение на число запрашиваемых страниц;
- управление доступом к БД;
- управление созданием таблиц;
- управление созданием процедур;
- управление использованием опции `u` (подключение от имени другого пользователя).

- По умолчанию в правах пользователя нет этих ограничений.
- Привилегии `QUERY_IO_LIMIT` и `QUERY_ROW_LIMIT` пользователей проверяются на основании оценок, выдаваемых оптимизатором запросов .

Примеры операторов доступа к БД

- Примеры операторов управления доступом к БД, как к единому целому:

```
GRANT NOCREATE_TABLE, NOCREATE_PROCEDURE
ON DATABASE new_accts
TO GROUP clerks;
```

{Запретить членам группы `clerks` создавать таблицы и процедуры в базе данных `new_accts`};

```
GRANT QUERY_ROW_LIMIT 250
ON DATABASE customers
TO Mary.
```

{Разрешить пользователю `Mary` получать в БД клиентов `customers` не более 250 строк на один запрос}

Права доступа к серверу БД

- Права доступа к серверу БД распространяются на все БД, обслуживаемые данным сервером. Набор этих прав тот же, что и для отдельных БД.
- Пример оператора, предоставляющего обладателям роли *creators* право на создание таблиц во всех БД:

```
GRANT CREATE TABLE
ON CURRENT INSTALLATION
TU ROLE creators;
```

- Привилегии, явно определенные для отдельных БД, имеют приоритет над привилегиями сервера.

Отмена привилегий доступа

- Для отмены привилегий, выданных ранее (как разрешительных, так и запретительных), служит оператор `REVOKE`. Синтаксически он аналогичен оператору `GRANT` с заменой слова `TO` на `FROM`.
- Пример:

```
REVOKE QUERY ROW LIMIT
ON DATABASE employee
FROM ROLE review emp.
```

- Приведенный оператор отменяет ограничение на число результирующих строк в одном запросе, если ранее оно было наложено на роль `review emp`.

Иерархия прав доступа

- Иерархия различных прав доступа задается, обычно, производителем СУБД. Так, в СУБД INGRES средства управления доступом позволяют реализовать следующие ограничения доступа:
  - ***операционные ограничения*** (за счет прав доступа `INSERT, UPDATE, DELETE`);
  - ***ограничения по значениям*** (за счет механизма представлений);
  - ***ограничения на ресурсы*** (за счет привилегий доступа к БД).
- Последний вид ограничений во многих СУБД, в частности, в СУБД ЛИНТЕР отсутствует, вследствие отсутствия доступа к БД в целом.

Иерархия прав доступа

- При обработке запроса СУБД сначала проверяет операционные ограничения. Если они оказываются нарушенными, запрос отвергается с выдачей соответствующей диагностики. Нарушение ограничений на значения влияет только на количество возвращаемых результирующих строк. Никакой диагностики при этом не выдается. После учета двух предыдущих ограничений, запрос поступает на обработку оптимизатору запросов для проверки ограничений на ресурсы. Если тот обнаружит превышение этих ограничений, запрос будет отвергнут с выдачей соответствующей диагностики.

Иерархия прав доступа

В соответствии с принятой в СУБД системой привилегий каждый пользователь помимо собственных, имеет привилегии `PUBLIC`, может входить в различные группы и запускать приложения с определенными ролями. Возникает вопрос: как соотносятся между собой права, предоставленные различным именованным носителям привилегий?

В СУБД INGRES иерархия авторизации привилегий выглядит следующим образом:

- роль (высший приоритет);
- пользователь;
- группа;
- `PUBLIC` (низший приоритет).

Иерархия прав доступа

- Для каждого объекта, к которому осуществляется доступ, СУБД пытается отыскать в иерархии привилегию, относящуюся к запрашиваемому виду доступа; (`SELECT`, `EXECUTE` и т.п.).
- Например, при попытке доступа к таблице с целью обновления (`UPDATE`), СУБД последовательно проверяет привилегии роли, пользователя, группы и всех пользователей (`PUBLIC`). Если хотя бы на одном уровне иерархии привилегия `UPDATE` имеется, запрос передается для дальнейшей обработки. В противном случае используется подразумеваемое право, доступа, которое предписывает отвергнуть запрос.

Трактовка ограничений на ресурсы

- Пример для СУБД INGRES. Пусть на 4-х уровнях иерархии прописаны свои ограничения на число результирующих строк запроса (привилегия `QUERY_ROW_LIMIT`):
  - роль – 1700;
  - пользователь – 1500;
  - группа – 2000;
  - `PUBLIC` – 1000.
- Если пользователь в начале сеанса задал и роль, и группу, будет использовано ограничение по роли. Если бы привилегия для роли отсутствовала или роль не была бы задана, он смог бы получать 1500 строк и т.д. Если бы привилегия `QUERY ROW LIMIT` вообще не была бы специфицирована, СУБД по умолчанию вообще не наложила бы ограничений на число строк.

Информация о привилегиях

- Важно не только давать и отбирать привилегии, но и иметь информацию о том, какими правами доступа обладает каждый из субъектов. Такая информация в том или ином виде хранится в специальной служебной БД СУБД. Так, в СУБД INGRES подобные данные хранятся в таблицах специальной БД `iidbdb`.
- Необходимую информацию о правах доступа, относящихся к текущему подключению можно получить с помощью функции `dbmsinfo`, а также путем анализа содержимого БД `iidbdb`. Можно узнать имена действующих группы и роли, значения количественных ограничений, наличие привилегий для создания таблиц и процедур и т.п.

Информация о привилегиях

- Другие таблицы `iiusergroup`, `iirole` и `iidbprivileges` БД `iidbdb` содержат, соответственно, список групп и их состав, перечень ролей вместе с зашифрованными паролями и сведения о привилегиях доступа к БД.
- Так, таблица `iiusergroup` состоит из трех столбцов:
  - `groupid` – имя группы;
  - `groupmem` – имя члена группы;
  - `reserve` – резервный столбец.
- Средствами SQL из этих таблиц можно извлечь всю необходимую информацию.

Соотношение прав доступа СУБД и ОС

- Данные, обрабатываемые средствами СУБД, располагаются в файлах и/или логических томах ОС. Соответственно, и доступ к этим данным возможен как с помощью СУБД, так и посредством утилит ОС. Это создает угрозу несанкционированного вмешательства в БД через утилиты ОС.
- Чтобы средствами ОС нельзя было скомпрометировать БД и сопутствующую информацию, например журналы транзакций, должен быть установлен максимально жесткий режим доступа к соответствующим файлам и томам.

Соотношение прав доступа СУБД и ОС

- Данные из БД могут экспортироваться в файлы ОС или другие хранилища. Возможен и обратный процесс импорта данных. Необходимо следить за тем, чтобы подобные операции не понижали уровень защищенности системы.
- Сделать это не просто. Во-первых, ОС может не обеспечивать должной защиты. Во-вторых, даже для развитых ОС необходимо соответствие между механизмами защиты СУБД и ОС. При большом числе пользователей данная задача становится очень сложной. Поэтому практичное решение сводится к административному контролю за экспортом/импортом информации.

Соотношение прав доступа СУБД и ОС

- Исходные тексты сложных процедур БД, также могут храниться в файлах ОС, и потенциально возможно их нелегальное изменение, способное привести к нарушениям ИБ.
- СУБД и ОС предлагают во многом сходные средства защиты данных. Более того, многие СУБД просто полагаются на ОС. Но даже для таких «дружественных» сервисов проведение согласованной политики безопасности является достаточно сложным делом.
- Поэтому на практике приходится всячески ограничивать информационный обмен между СУБД и ОС.

Мандатная защита в СУБД

- Современные АИС обеспечивают также принудительный контроль доступа (mandatory access control). Он основан на отказе от понятия владельца данных и опирается на, так называемые, ***метки безопасности*** (security labels), которые присваиваются данным при их создании и служат для классификации данных по уровням конфиденциальности.
- Метки содержатся также и на официальных разрешениях (допусках) пользователей для обращения к данным соответствующего уровня конфиденциальности.
- При мандатной защите конкретный пользователь может оперировать только с данными, расположенными на уровне конфиденциальности, который соответствует его статусу.

Метки безопасности

- Мандатная защита, позволяет связать метки безопасности с каждой строкой любой таблицы в БД. Любой пользователь может потребовать в своем запросе отобразить любую таблицу из БД, однако увидит он только те строки, метки безопасности которых не превышают его статус.
- СУБД проверяет статус пользователя и, в ответ на его запрос, возвращает только те строки таблицы, которые удовлетворяют запросу и соответствуют этому статусу.
- Метки безопасности неизменны на всем протяжении существования объекта защиты, уничтожаются только вместе с ним и территориально (на диске) располагаются тоже вместе с ним, а не в системном каталоге, как это происходит при логической защите.

Метки безопасности

- Средства мандатной защиты предоставляются специальными (trusted) версиями СУБД, которые не позволяют игнорировать метки безопасности при получении доступа к информации.
- Такие реализации СУБД, как правило, представляют собой комплекс средств как на машине-сервере, так и на машине-клиенте, при этом возможно использование специальной защищенной версии ОС. Кроме разграничения доступа к информации посредством меток конфиденциальности, защищенные СУБД предоставляют средства слежения за доступом субъектов к объектам защиты (аудит).

Метки безопасности

- Использование СУБД с возможностями мандатной защиты позволяет разграничить доступ собственно к данным, хранящимся в информационной системе, от доступа к именованным объектам данных.
- Существуют реализации, позволяющие разграничивать доступ вплоть до конкретного значения конкретного атрибута в конкретной строке конкретной таблицы.
- Обычно сама метка представляет собой целый набор значений, отражающих, например, уровень защищенности устройства, на котором хранится таблица, уровень защищенности самой таблицы, уровень защищенности атрибута и уровень защищенности кортежа.

СУБД INGRES/Enhanced Security

- Механизм меток безопасности был впервые реализован в версии СУБД INGRES/Enhanced Security (INGRES с повышенной безопасностью).
- В этой СУБД к каждой реляционной таблице неявно добавляется столбец, содержащий метки безопасности строк таблицы. Каждая метка состоит из трех компонентов:
  - уровень секретности;
  - категория;
  - область.
- Компонент «уровень секретности» зависит от приложения.
- Например, для военных ведомств США это 4 уровня секретности от «совершенно секретно» до «несекретно».

СУБД INGRES/Enhanced Security

- Понятие «категория» позволяет разделить данные на тематические отсеки. В коммерческих приложениях категориями могут быть «финансы», «кадры», «материальные ценности» и т.п. В военной сфере категориями могут быть рода войск, типы вооружений и военной техники и т.п.
- Компонент «область» является еще одним средством деления информации на отсеки. Он может действительно иметь географический смысл (например, страна, регион). В коммерческих приложениях «область» может обозначать тип местности (городская, сельская), городские районы, филиалы, торговые точки и т.п. В военных приложениях – это - военные округа, места дислокации частей и т.п.

СУБД INGRES/Enhanced Security

- Каждый пользователь INGRES/Enhanced Security имеет статус, определяемый присвоенной ему меткой безопасности пользователя (МБП). Он может получить доступ к данным с меткой безопасности данных (МБД), если одновременно удовлетворяются следующие 3 условия:
  - уровень секретности в МБП должен быть не ниже уровня секретности в МБД;
  - набор категорий, заданных в МБД, должен целиком содержаться в МБП;
  - набор областей, заданных в МБП, должен целиком содержаться в МБД.

СУБД INGRES/Enhanced Security

Пример 1

МБД				МБП				Доступ
Уровень секрет-и	Категория	Область		Уровень секрет-ти	Категория	Область		
Секретно	Финансы	Россия СНГ		Сов. секретно	Финансы и кадры	Россия	→	Разрешен
Секретно	Финансы	Россия СНГ		Сов. секретно	Кадры	Россия	→	Запрещен

СУБД INGRES/Enhanced Security

Пример 2

Метка данных		
Уровень секретности	Категория	Область
Сов. секретно	Ракетное оружие	Россия, Украина
Сов. секретно	Ракетное оружие	Россия

Метка пользователя			Доступ
Уровень секретности	Категория	Область	
Сов. секретно	Ядерное и ракетное оружие	Россия	→ Разрешен
Сов. секретно	Ядерное и ракетное оружие	Россия, Украина	→ Запрещен

СУБД INGRES/Enhanced Security

- Для совместимости с обычными версиями СУБД, столбец с метками безопасности не показывается.
- Механизм меток безопасности не отменяет, а дополняет произвольное управление доступом. Пользователи по-прежнему могут оперировать данными в рамках своих привилегий, но даже при наличии привилегии SELECT им будет доступна только часть данных.
- При добавлении или изменении строк они наследуют МБП, инициировавшего операцию. Таким образом, даже если авторизованный пользователь перепишет секретную информацию в общедоступную таблицу, менее благонадежные пользователи не смогут ее прочитать.

СУБД INGRES/Enhanced Security

- Специальная привилегия `DOWNGRADE` позволяет изменять метки безопасности, ассоциированные с данными. Это необходимо для коррекции меток, по тем или иным причинам оказавшихся неправильными.
- СУБД INCRES/Enhanced Security допускает не только скрытое, но и явное включение меток безопасности в реляционные таблицы. Появился новый тип данных, security label, поддерживающий соответствующие операции сравнения.
- INCRES/Enhanced Security – первая СУБД, получившая сертификат по классу безопасности B1. В настоящее время появились и другие СУБД с мандатной защитой.

СУБД «Линтер»

- Отечественным примером является СУБД с меточной безопасностью является СУБД «Линтер», которая получила признание в силовых структурах, как единственная СУБД, имеющая сертификат по второму классу защиты от НСД, что соответствует классу В3 оранжевой книги.
- Мандатная защита ЛИНТЕР также опирается на механизм меток безопасности, однако, по сравнению с СУБД INCRES/Enhanced Security ЛИНТЕР имеет свои специфические особенности, которые позволяют существенно более строго реализовать контроль управления доступом к конфиденциальным данным.

СУБД «Линтер»

1. В СУБД «Линтер» все перечисленные объекты (независимо от их иерархии в БД) разбиваются на группы принадлежности.
   - Объект может принадлежать только одной из групп (это может быть, например, разбиение по отделам организации).
   - Группы принадлежности напрямую связаны с группами субъектов. Субъект вправе видеть только данные своей группы, если между группами субъектов не установлены отношения доверия.

СУБД «Линтер»

2. В СУБД «Линтер» все объекты выстроены в иерархию по уровням конфиденциальности и по уровням ценности или важности.
   - Уровень конфиденциальности разбивает объекты по доступности на чтение. Пользователь с более низким уровнем доступа не будет знать даже о существовании объектов с более высоким уровнем конфиденциальности.
   - Уровень ценности, напротив, разбивает данные (объекты) по важности, ограничивая возможность их удаления и модификации.

СУБД «Линтер»

- Метка объекта включает:
  - группу субъекта, который внес данный объект;
  - уровень доступа на чтение – RAL (Read Access Level);
  - уровень доступа на запись – WAL (Write Access Level).
- Метка субъекта выглядит аналогично:
  - группу, к которой принадлежит субъект;
  - RAL-уровень субъекта, который представляет собой максимальный RAL-уровень доступной субъекту;
  - WAL-уровень субъекта, то есть минимальный RAL-уровень объекта, который может быть создан этим субъектом.

СУБД «Линтер»

- Все пользователи БД считаются разбитыми на непересекающиеся группы. Группа описывает область доступных пользователю данных. Для каждой группы существует админ группы (DBA), созданный админом системы. При этом пользователи одной группы не видят данных, принадлежащих пользователям другой группы.
- В системе реализовано понятие «уровень доверия между группами». Уровни доверия не могут быть вложенными. Группа представляет собой число в диапазоне [1-250]. Группа 0 – группа админ системы. Только он может создать пользователя в группе, отличной от своей. Все данные, созданные от имени пользователя, помечаются его группой.

СУБД «Линтер»

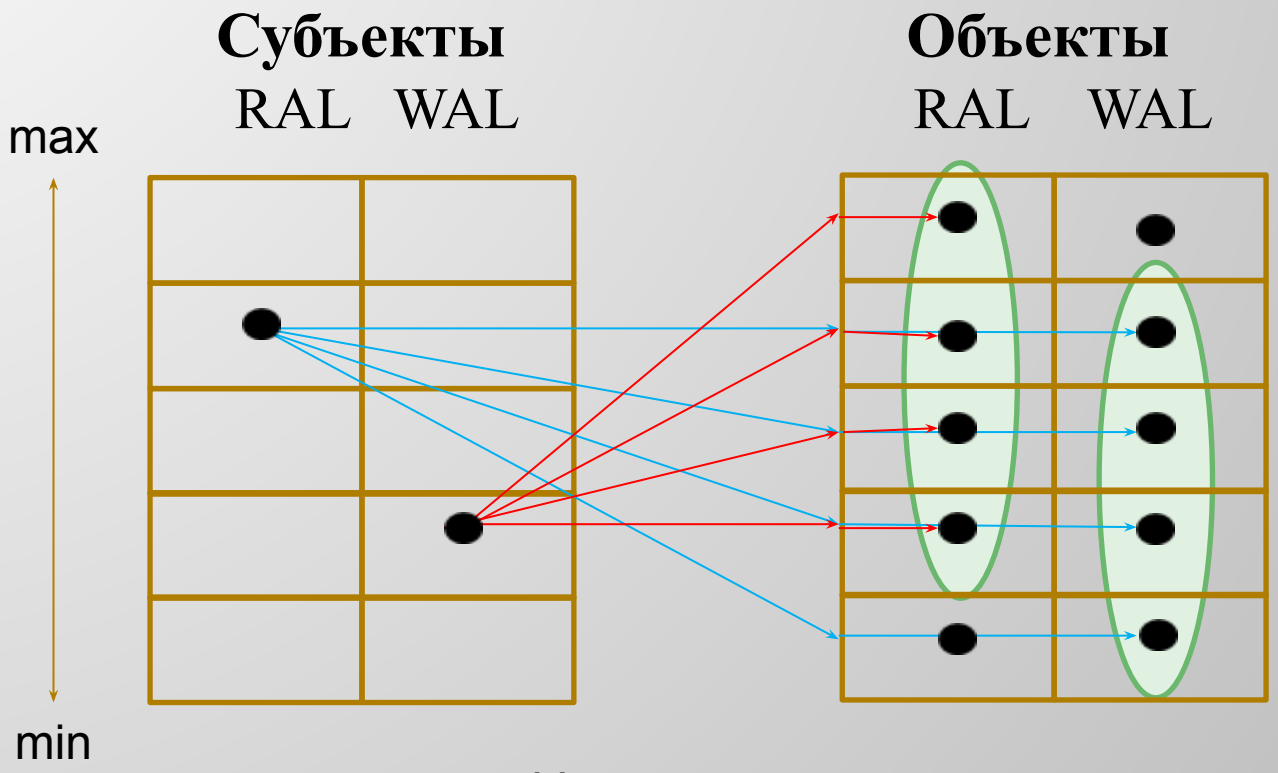
- Уровни доступа вводятся для проверки прав на осуществление чтения-записи информации.
- Вводятся следующие уровни доступа:

1. Для пользователя (субъекта):
   - RAL – уровень доступа; пользователь может получать (читать) информацию, RAL-уровень которой не выше его собственного уровня доступа;
   - WAL – уровень доверия на понижение уровня конфиденциальности; пользователь не может вносить информацию с уровнем доступа (RAL-уровнем) более низким, чем данный WAL-уровень пользователя.

СУБД «Линтер»

2. Для информации:
   - RAL – уровень чтения; пользователь может получать (читать) информацию, RAL-уровень которой не выше его собственного RAL-уровня (может читать менее конфиденциальные данные);
   - WAL – уровень ценности или уровень доступа на запись (модификацию, удаление); пользователь может модифицировать (удалять) информацию, WAL-уровень которой не выше его RAL-уровня.

СУБД «Линтер»


Субъекты
RAL WAL
Объекты
RAL WAL
max
min
Уровень доступа на чтение
Уровень доступа на запись

СУБД «Линтер»

- Создать пользователя с произвольными уровнями может только администратор системы. Остальные администраторы (DBA) могут создавать пользователей (или изменять их уровень) лишь в пределах отведенных им уровней.
- Пользователь может принудительно пометить вводимые данные, указав в списке атрибутов уровни доступа для соотв. записей и полей (для операций INSERT или UPDATE).
- По умолчанию вносимые данные наследуют уровни пользователя, вносящего/изменяющего данные.
- Уровни, как и группы, нельзя использовать в случае, если они не созданы специальными запросами.