Управление доступом на основе ролей в ClickHouse

Пример

Старый вариант (users.xml):

```
<yandex>
  <users>
    <alice>
      <profile>manager</profile>
      <networks>
        <ip>::/0</ip>
      </networks>
      <password_sha256_hex>
        …
      </password_sha256_hex>
      <allow_databases>
        <database>products</database>
        <database>sales</database>
      </allow_databases>
    </alice>
  </users>
  <profiles>
    <manager>
      <max_memory_usage>
        1000000000
      </max_memory_usage>
    </manager>
  </profiles>
</yandex>
```

Новый вариант (DCL):

```
CREATE ROLE manager
   SETTINGS max_memory_usage=1000000000;

GRANT ALL ON products.* TO manager;
GRANT ALL ON sales.* TO manager;
CREATE USER alice IDENTIFIED BY ‘password’;
GRANT manager TO alice;
```

Преимущества

- Гибкая настройка, можно указывать конкретные права (например, `GRANT SELECT`, `GRANT INSERT`)
- Можно ассоциировать права и настройки с ролью и потом назначить роль многим пользователям.
- Больше возможностей для задания политик защиты строк (*row-level security*) и настроек
- `ON CLUSTER`: права сразу для всего кластера
- Без редактирования *users.xml*

GRANT & REVOKE

GRANT ALL ON db.table TO john	Разрешить пользователю john выполнять любые действия с таблицей db.table
GRANT SELECT ON db.table TO john	Разрешить пользователю john выполнять SELECT из таблицы db.table
GRANT SELECT(x,y) ON db.table TO john	Разрешить пользователю john выполнять SELECT столбцов x, y из таблицы db.table
GRANT SELECT, INSERT, ALTER UPDATE ON db.table TO john, manager	Разрешить пользователю john и роли manager выполнять SELECT, INSERT и ALTER UPDATE для таблицы db.table
GRANT ALL ON db.* TO john	Разрешить пользователю john выполнять любые действия с базой данных db
GRANT ALL ON *.* TO john WITH GRANT OPTION	Разрешить пользователю john выполнять любые действия, включая передачу прав другим пользователям и ролям
GRANT manager TO john	Разрешить пользователю john разрешено использовать роль manager.
REVOKE SELECT(x,y) ON db.table FROM john	Отозвать разрешение пользователю john выполнять SELECT столбцов x, y из таблицы db.table
REVOKE ALL ON *.* FROM john	Отозвать все разрешения, данные пользователю john (кроме ролей)

Права доступа

ALL

SELECT | INSERT | KILL QUERY | OPTIMIZE | TRUNCATE | dictGet

ALTER

ALTER TABLE

ALTER UPDATE | ALTER DELETE | ALTER SETTINGS

ALTER COLUMN

- ALTER ADD COLUMN
- ALTER DROP COLUMN
- ALTER MODIFY COLUMN
- ALTER COMMENT COLUMN
- ALTER CLEAR COLUMN
- ALTER RENAME COLUMN

ALTER INDEX

- ALTER ORDER BY
- ALTER ADD INDEX
- ALTER DROP INDEX
- ALTER MATERIALIZE INDEX
- ALTER CLEAR INDEX

ALTER CONSTRAINT

- ALTER ADD CONSTRAINT
- ALTER DROP CONSTRAINT

ALTER VIEW

- ALTER VIEW REFRESH
- ALTER VIEW MODIFY QUERY

ALTER FETCH PARTITION | ALTER MOVE PARTITION

ALTER FREEZE PARTITION | ALTER TTL

CREATE

CREATE DATABASE

CREATE DICTIONARY

CREATE TABLE

- CREATE VIEW
- CREATE TEMPORARY TABLE

DROP

DROP DATABASE

DROP DICTIONARY

DROP TABLE

- DROP VIEW

INTROSPECTION

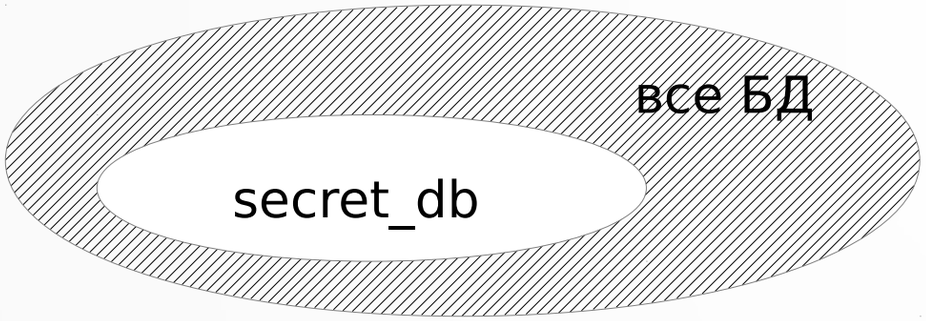
- addressToLine
- addressToSymbol
- demangle

SOURCES

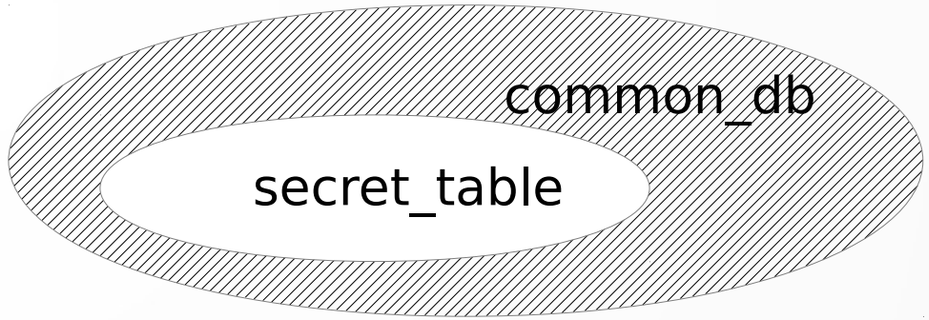
- FILE
- URL
- REMOTE
- MYSQL
- ODBC
- JDBC
- HDFS
- S3

Частичный отзыв
(partial revokes)

```
GRANT SELECT ON *.* TO alex;
REVOKE SELECT ON secret_db.* FROM alex;
```



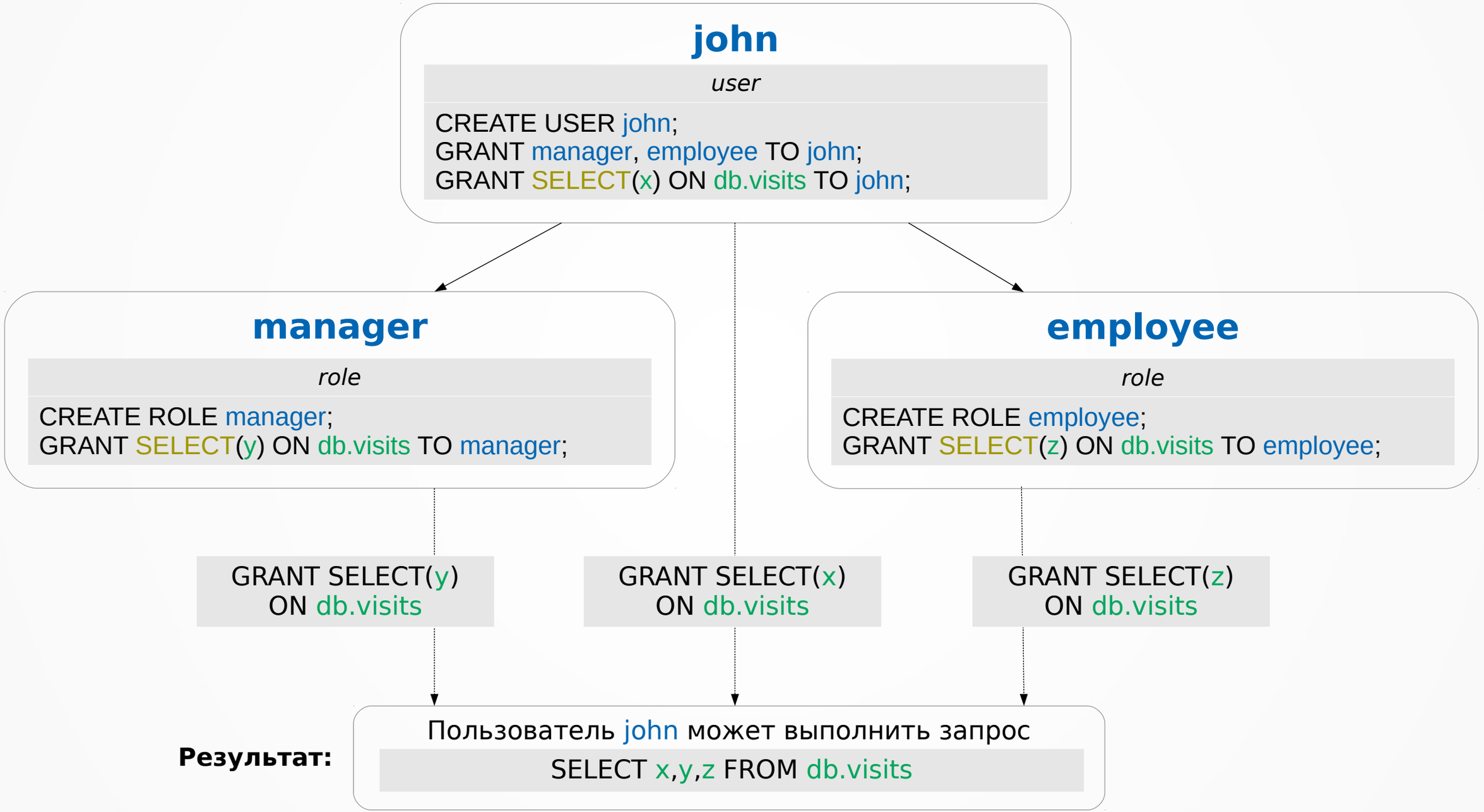

```
GRANT INSERT ON common_db.* TO alex;
REVOKE INSERT ON common_db.important_table FROM alex;
```



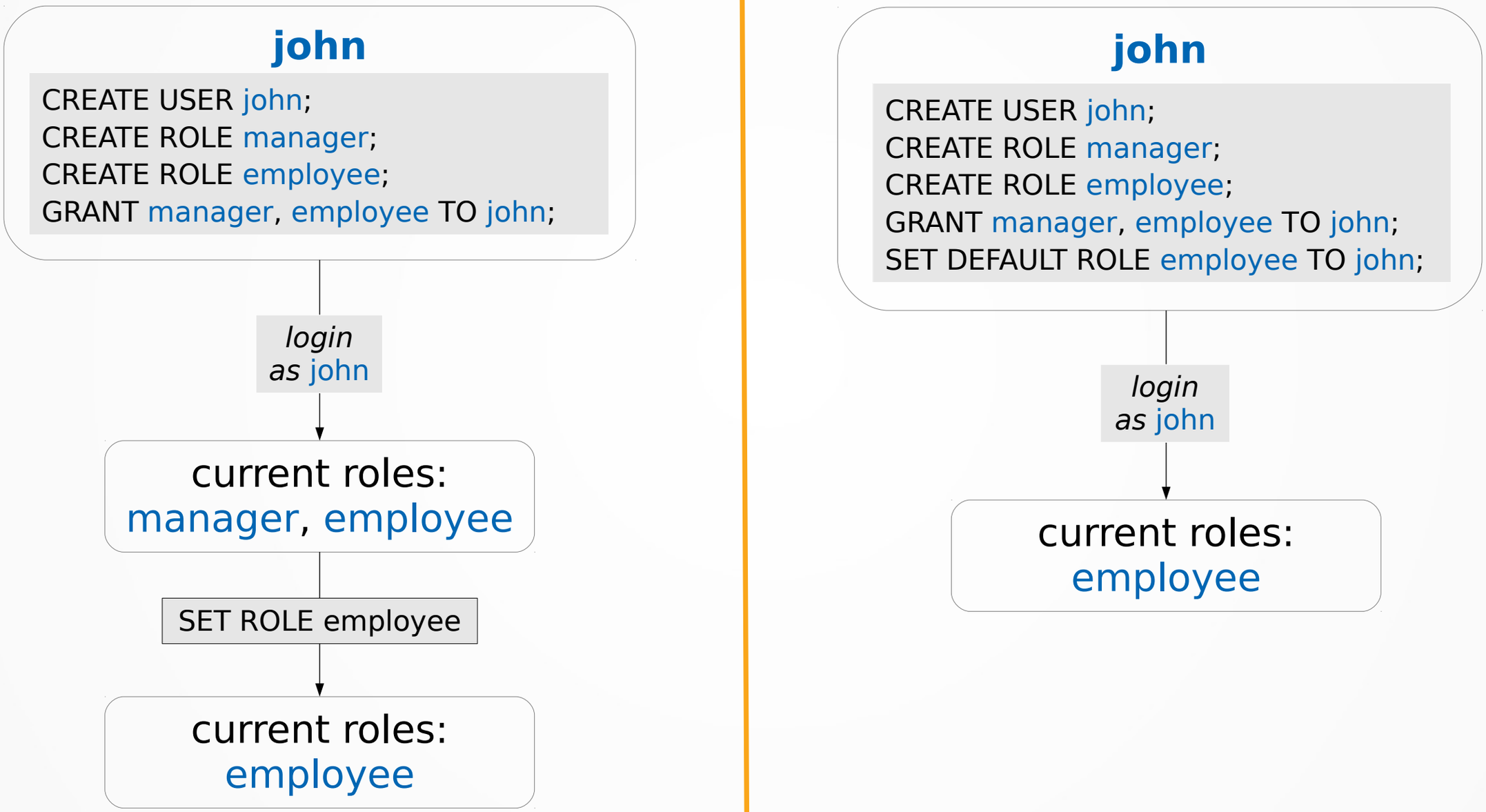
`SHOW GRANTS FOR alex;` – узнать права alex →вывод:

GRANT INSERT ON common_db TO alex;
REVOKE INSERT ON common_db.secret_table FROM alex;

Права, полученные от ролей, объединяются:



SET ROLE & SET DEFAULT ROLE



Настройки

Задача: задать для пользователя alex max_memory_usage = 1000000000 и запретить ему менять эту переменную.

Решение:

Старый вариант (users.xml):

```
<yandex>
  <users>
    <alex>
      <profile>profile1</profile>
      …
    </alex>
  </users>
  <profiles>
    <profile1>
      <max_memory_usage>
        1000000000
      </max_memory_usage>
      <constraints>
        <max_memory_usage>
          <readonly/>
        </max_memory_usage>
      </constraints>
    </profile1>
  </profiles>
</yandex>
```

Новый вариант (DCL):

(1)

```
CREATE USER alex SETTINGS
    max_memory_usage=1000000000 READONLY;
```

(2)

```
CREATE SETTINGS PROFILE profile1 SETTINGS
    max_memory_usage=1000000000 READONLY;

CREATE USER alex SETTINGS PROFILE profile1;
```

(3)

```
CREATE ROLE role1 SETTINGS
    max_memory_usage=1000000000 READONLY;

CREATE USER alex;

GRANT role1 TO alex;
```

Политики защиты строк

(Row-level security)

Старый вариант (users.xml):

```
<yandex>
  <users>
    <alice>
      …
      <allow_databases>
        <database>salaries</database>
      </allow_databases>
      <databases>
        <db>
          <salaries>
            <filter>level &lt; 5</filter>
          </salaries>
        </db>
      </databases>
    </alice>
  </users>
</yandex>
```

Новый вариант (DCL):

```
CREATE USER alice;
GRANT SELECT ON db.salaries TO alice;
CREATE POLICY filter ON db.salaries
    FOR SELECT USING level<5 TO alice;
```

Комбинирование политик защиты строк

С фильтром WHERE:

```
CREATE USER alex;
GRANT SELECT ON db.salaries TO alex;
CREATE POLICY filter ON db.salaries FOR SELECT USING level<5 TO alex;


-- login as alex;


SELECT * FROM db.salaries WHERE id<1000;
-- фактически используется условие level<5 AND id<1000
```

Комбинирование политик защиты строк

Друг с другом:

```
CREATE ROLE role1;
CREATE ROLE role2;
GRANT role1, role2 TO alex;
CREATE POLICY filter1 ON db.protected_data FOR SELECT USING a=1 TO role1;
CREATE POLICY filter2 ON db.protected_data FOR SELECT USING b=2 TO role2;

-- login as alex;

SELECT * FROM db.protected_data; -- ???
```

Решение: [AS { PERMISSIVE | RESTRICTIVE }]

```
CREATE POLICY filter_name ON db.table [AS { PERMISSIVE | RESTRICTIVE }]
FOR SELECT USING filter_condition TO role;
```

Правило: **Строка видима, если выполнено условие**

any_of(permissive_filters) AND all_of(restrictive_filters)

Комбинирование политик защиты строк

Правило: **Строка видима, если выполнено условие**
any_of(permissive_filters) AND all_of(restrictive_filters)

Ответ:

```
CREATE ROLE role1;
CREATE ROLE role2;
GRANT role1, role2 TO alex;
CREATE POLICY filter1 ON db.protected_data
    FOR SELECT USING a=1 TO role1; -- permissive
CREATE POLICY filter2 ON db.protected_data
    FOR SELECT USING b=2 TO role2; -- permissive


-- login as alex;


SELECT * FROM db.protected_data;
-- фактически используется условие a = 1 OR b = 2
```

Альтернативный вариант:

```
CREATE ROLE role1;
CREATE ROLE role2;
GRANT role1, role2 TO alex;
CREATE POLICY filter1 ON db.protected_data
    FOR SELECT USING a=1 TO role1; -- permissive
CREATE POLICY filter2 ON db.protected_data
    AS RESTRICTIVE FOR SELECT USING b=2 TO role2;


-- login as alex;


SELECT * FROM db.protected_data;
-- фактически используется условие a = 1 AND b = 2
```

ON CLUSTER

кластер:

```
<yandex>
  <remote_servers>
    <my_cluster>
      <shard>
        <replica>
          <host>node_1</host>
          <port>9000</port>
        </replica>
      </shard>
      <shard>
        <replica>
          <host>node_2</host>
          <port>9000</port>
        </replica>
      </shard>
    </my_cluster>
  </remote_servers>
</yandex>
```

управление доступом:

```
CREATE ROLE manager ON CLUSTER ‘my_cluster’
                              SETTINGS  max_memory_usage=1000000000;

GRANT ON CLUSTER ‘my_cluster’ ALL ON products.* TO manager;
GRANT ON CLUSTER ‘my_cluster’ ALL ON sales.* TO manager ;

CREATE USER alice ON CLUSTER ‘my_cluster’
                             IDENTIFIED BY ‘password’;

GRANT ON CLUSTER ‘my_cluster’ manager TO alice;

ALTER USER alice ON CLUSTER ‘my_cluster’
                             IDENTIFIED BY ‘another_password’;
```

Хранение параметров управления доступом

конфигурационный файл сервера:

```
<yandex>
  <access_control_path>/var/lib/clickhouse/access/</access_control_path>
</yandex>
```

```
CREATE USER kevin SETTINGS readonly=1;
```

/var/lib/clickhouse/access/

```
e4375cac-afb7-11ea-b3de-0242ac130004.sql:
ATTACH USER kevin SETTINGS readonly=1;


users.list:
kevin→e4375cac-afb7-11ea-b3de-0242ac130004
```

```
ALTER USER kevin IDENTIFIED BY ‘abc’;
```

/var/lib/clickhouse/access/

```
e4375cac-afb7-11ea-b3de-0242ac130004.sql:
ATTACH USER kevin IDENTIFIED WITH sha256_hash BY
'BA7816BF8F01CFEA414140DE5DAE2223B00361A396177A9CB410FF61F20015AD'
SETTINGS readonly=1;


users.list:
kevin→e4375cac-afb7-11ea-b3de-0242ac130004
```

Хранение параметров управления доступом

```
CREATE USER kevin IDENTIFIED BY ‘abc’
                  SETTINGS readonly=1;

CREATE ROLE assistant;
GRANT SELECT ON *.* TO assistant;

GRANT assistant TO kevin;
```


/var/lib/clickhouse/access/

```
e4375cac-afb7-11ea-b3de-0242ac130004.sql:
ATTACH USER kevin IDENTIFIED WITH sha256_hash BY
'BA7816BF8F01CFEA414140DE5DAE2223B00361A396177A9CB410FF61F20015AD'
SETTINGS readonly=1;

ATTACH GRANT ID(‘95297098-afba-11ea-b3de-0242ac130004’)
TO kevin;


95297098-afba-11ea-b3de-0242ac130004.sql:
ATTACH ROLE assistant;
ATTACH GRANT SELECT ON *.* TO assistant;


users.list:
kevin→e4375cac-afb7-11ea-b3de-0242ac130004


roles.list:
assistant→95297098-afba-11ea-b3de-0242ac130004
```

Планы

- Авторизация через *LDAP/Kerberos*, *OAuth*
- Хранение пользователей и ролей в *ZooKeeper*