

**СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ «КАТРАПС» (СУБД
«КАТРАПС») ВЕРСИИ 1.10.11**

РУКОВОДСТВО АДМИНИСТРАТОРА ИС

Листов 12

МОСКВА

2025

Оглавление

Работа с СУБД «КАТРАПС».....	3
Вход в СУБД «КАТРАПС».....	3
Создание схемы базы данных.....	3
Извлечение данных.....	6
Добавление данных.....	7
Изменение данных.....	8
Удаление данных.....	8
Управление пользователями СУБД «КАТРАПС».....	9
Создание пользователей и предоставление прав доступа.....	9
Создание учетной записи.....	9
Просмотр существующих пользователей и их привилегий.....	10
Смена пароля.....	10
Смена пароля пользователю root после установки.....	10
Блокировка учетной записи.....	10
Срок действия пароля пользователя.....	11
Системные переменные.....	11
Ограничение срока действия пароля пользователя.....	11

Работа с СУБД «КАТРАПС»

Вход в СУБД «КАТРАПС»

СУБД «КАТРАПС» — это система баз данных, сервер баз данных. Для взаимодействия с сервером используется клиентская программа. Клиент по умолчанию поставляется с СУБД «КАТРАПС» под названием `mysqli`. С помощью этого клиента можно либо вводить запросы из командной строки, либо переключаться в терминал, то есть в режим монитора.

Для использования режима монитора в командной строке необходимо ввести следующее:

```
mysqli -u //user_name// -p -h //ip_address//
```

Где:

`user_name` - имя пользователя;

`ip_address` - адрес или имя сервера.

Опция `-u` предназначена для указания имени пользователя.

Опция `-p` предписывает клиенту `mysqli` запросить пароль. Если пароль для пользователя еще не установлен (пароль пуст), нужно нажать [Enter] при появлении соответствующего запроса.

Опция `-h` предназначена для указания имени хоста или IP-адреса сервера.

Если вход выполняется с компьютера, на котором установлена база данных, то вводить `-h` и `ip_address` не требуется.

После успешного ввода пароль при появлении соответствующего запроса будет выполнен вход в СУБД «КАТРАПС» через клиент. Чтобы выйти, необходимо ввести `quit` или `exit` и нажать [Enter].

Для подключения к конкретной базе данных команду выше можно расширить наименованием базы данных:

```
mysqli -u //user_name// -p -h //ip_address// //db_name//
```

Где:

`db_name` - имя базы данных, к которой необходимо получить доступ.

Далее необходимо ввести пароль указанного пользователя, если пароль введен верно, на экране появится следующее:

```
MariaDB [test]>
```

В скобках указано имя базы данных, к которой выполнено подключение.

Создание схемы базы данных

Ролевая модель СУБД «КАТРАПС» предусматривает наделение основными полномочиями по управлению ИС только «Администраторам ИС», которые создаются и отслеживаются «Администратором СУБД». Администратор ИС может создавать свои схемы баз данных, таблицы, индексы, временные таблицы, а также использовать модуль маскирования остаточной информации, работоспособность которого описана в отдельном руководстве.

Базы данных хранят информацию в таблицах, обеспечивая эффективное управление данными. Внесение изменений в базу данных или получение данных выполняются с помощью операторов SQL, называемых запросами.

Операторы SQL заканчиваются точкой с запятой (или \G). Можно разделить оператор SQL на несколько строк, однако он не будет передан на сервер клиентом до тех пор, пока не будет завершен точкой с запятой и не будет нажат [Enter]. Чтобы отменить оператор SQL после того, как начали его вводить, нужно ввести \c и нажать [Enter].

По основному соглашению зарезервированные слова печатаются заглавными буквами. Однако в этом нет необходимости. СУБД «КАТРАПС» не учитывает регистр в зарезервированных словах. Однако имена баз данных и таблиц в операционной системе чувствительны к регистру. Это связано с тем, что они ссылаются на связанные каталоги и файлы файловой системы. Имена столбцов не чувствительны к регистру, поскольку файловая система сама по себе не влияет на них.

Чтобы иметь возможность добавлять данные и манипулировать ими, сначала необходимо создать структуру базы данных из клиента ``mariadb``:

```
CREATE DATABASE kat_test;

USE kat_test;
```

Оператор SQL `CREATE DATABASE` создает подкаталог с именем ``kat_test`` в файловой системе ОС в каталоге, в котором хранятся файлы данных СУБД «КАТРАПС». Он не создаст никаких данных - просто создаст место для добавления таблиц, которые, в свою очередь, будут содержать данные. Оператор SQL `USE` установит новую базу данных в качестве базы данных по умолчанию. Она останется активной схемой БД до её изменения и до выхода из СУБД «КАТРАПС».

Создание таблицы в БД предполагает команду вида:

```
CREATE TABLE book (

isbn CHAR ( 20 ) PRIMARY KEY ,

title VARCHAR ( 50 ),

author_id INT ,

Publisher_id INT ,

year_pub CHAR ( 4 ),

description TEXT );
```

Этот оператор SQL создает таблицу с шестью столбцами. Первый столбец (`isbn`) представляет собой идентификационный номер каждой строки. Он имеет тип символов фиксированной ширины — 20 символов. Это будет столбец первичного ключа, в котором будут индексироваться данные. Тип данных столбца «`title`» — это символьный столбец переменной ширины, содержащий не более пятидесяти символов. Третий и четвертый столбцы (`author_id` и `Publisher_id`) будут содержать целочисленные типы данных. В пятом столбце указан год публикации каждой книги. Последний столбец (`description`) предназначен для типа данных `TEXT`, что означает, что это столбец переменной ширины, и он может содержать до 65535 байт данных для каждой строки. Дополнительная информация о типах данных приведена в Приложении 1.

Для создания базы данных и таблиц с использованием расширенного синтаксиса команд подходит следующий пример:

```
CREATE DATABASE IF NOT EXISTS kat_test;

USE kat_test;

CREATE TABLE IF NOT EXISTS books (

BookID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
```

```
Title VARCHAR(100) NOT NULL,
SeriesID INT, AuthorID INT);

CREATE TABLE IF NOT EXISTS authors
(id INT NOT NULL PRIMARY KEY AUTO_INCREMENT);

CREATE TABLE IF NOT EXISTS series
(id INT NOT NULL PRIMARY KEY AUTO_INCREMENT);

INSERT INTO books (Title, SeriesID, AuthorID)
VALUES('The Fellowship of the Ring', 1, 1),
('The Two Towers', 1, 1), ('The Return of the King', 1, 1),
('The Sum of All Men', 2, 2), ('Brotherhood of the Wolf', 2, 2),
('Wizardborn', 2, 2), ('The Hobbit', 0, 1);
```

Для просмотра результата создания таблиц в БД (получения списка таблиц) используется следующая команда:

```
SHOW TABLES;
```

```
+-----+
| Tables_in_test |
+-----+
| authors      |
| books        |
| series       |
+-----+
3 rows in set (0.00 sec)
```

Для получения информации о таблице используется следующая команда:

```
DESCRIBE books;
```

```
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| BookID | int(11) | NO   | PRI | NULL    | auto_increment |
```

```
| Title | varchar(100) | NO | | NULL | |
| SeriesID | int(11) | YES | | NULL | |
| AuthorID | int(11) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
```

Столбец *Field* содержит имена, необходимые для извлечения данных из таблицы. Другие столбцы представляют полезную информацию о структуре и типе данных в базе данных.

Извлечение данных

Основной метод получения данных из таблиц — использование оператора SELECT, с которым может использоваться множество опций. Простой вариант использования:

```
SELECT title
FROM books;
```

Результат отобразит все строки таблицы. Если в таблице тысячи строк, СУБД «КАТРАПС» отобразит тысячи. Чтобы ограничить количество извлекаемых строк, можно добавить LIMIT в оператор SELECT следующим образом:

```
SELECT title
FROM books
LIMIT 5;
```

Это ограничит количество отображаемых строк до пяти.

Для извлечения данных из двух таблиц, связанных значением одного из столбцов, можно использовать команду JOIN следующим образом:

```
SELECT title, name_last
FROM books
JOIN authors USING (AuthorID);
```

Основная таблица, из которой извлекаются данные, указана в предложении FROM. Таблица, к которой идет присоединение, указывается в предложении JOIN вместе с названием столбца (т. е., AuthorID), который используется для соединения.

Чтобы получить записи из основной таблицы в связке с данными из присоединенной таблицы, но с использованием конкретного значения в связанной таблице, можно использовать предложение WHERE с оператором SELECT:

```
SELECT title AS 'Kafka Books'
FROM books
JOIN authors USING (AuthorID)
```

```
WHERE name_last = 'Kafka';
```

```
+-----+
```

```
| Kafka Books |
```

```
+-----+
```

```
| The Castle |
```

```
| The Trial |
```

```
| The Metamorphosis |
```

```
| America |
```

```
+-----+
```

Параметр AS рядом с заголовком имени столбца указан, чтобы изменить заголовок столбца в наборе результатов. Это так называемый *псевдоним*.

Для запроса всех данных из таблицы оператор SELECT используется следующим образом:

```
SELECT * FROM books;
```

```
+-----+-----+-----+-----+
```

```
| BookID | Title | SeriesID | AuthorID |
```

```
+-----+-----+-----+-----+
```

```
| 1 | The Fellowship of the Ring | 1 | 1 |
```

```
| 2 | The Two Towers | 1 | 1 |
```

```
| 3 | The Return of the King | 1 | 1 |
```

```
| 4 | The Sum of All Men | 2 | 2 |
```

```
| 5 | Brotherhood of the Wolf | 2 | 2 |
```

```
| 6 | Wizardborn | 2 | 2 |
```

```
| 7 | The Hobbit | 0 | 1 |
```

```
+-----+-----+-----+-----+
```

```
7 rows in set (0.00 sec)
```

Символ звездочки (*) указывает на выбор всех столбцов.

Добавление данных

Добавление данных в таблицу выполняется с использованием оператора INSERT:

```
INSERT INTO books (Title, SeriesID, AuthorID)
```

```
VALUES ("Lair of Bones", 2, 2);
```

```
Query OK, 1 row affected (0.00 sec)
```

Список столбцов указывается в скобках после имени таблицы, затем вводится ключевое слово **VALUES**, за которым следует список значений в круглых скобках, в том же порядке, в котором были перечислены столбцы. Сообщение в конце указывает на то, что выполнение оператора SQL прошло нормально и была введена одна строка в таблицу.

С помощью одной команды в таблицу можно добавить несколько записей:

```
INSERT INTO books
```

```
(Title, SeriesID, AuthorID)
```

```
VALUES('The Trial', '1', '23'),
```

```
('The Metamorphosis', '2', '15'),
```

```
('America', '3', '6');
```

В этом примере три записи добавляются одним оператором. Это позволяет предоставить список имен столбцов один раз. Ключевое слово **VALUES** также указывается только один раз, за ним следует отдельный набор значений для каждой записи, каждый из которых заключен в круглые скобки и разделен запятыми.

Изменение данных

Чтобы изменить данные в таблице используется команда **UPDATE**:

```
UPDATE books
```

```
SET Title = "The Hobbit"
```

```
WHERE BookID = 7;
```

```
Query OK, 1 row affected (0.00 sec)
```

```
Rows matched: 1 Changed: 1 Warnings: 0
```

В инструкции **SET** перечисляются столбцы и значения для их установки. **WHERE** говорит о том, что надо изменить только те строки, в которых **BookID** имеет значение 7. Из возвращенного сообщения видно, что обновлена одна строка.

Удаление данных

Чтобы изменить настройки таблицы, используется оператор **ALTER TABLE**.

Если нужно удалить строку данных в таблице, используется оператор **DELETE**:

```
DELETE FROM books
```

```
WHERE BookID = '2034';
```

Чтобы полностью удалить таблицу (включая ее данные), используется оператор **DROP TABLE**, за которым следует имя таблицы. Действие, вызываемое оператором **DROP TABLE**.

Управление пользователями СУБД «КАТРАПС»

Создание пользователей и предоставление прав доступа

Учетные записи в СУБД «КАТРАПС» представлены в виде связки <имя пользователя>@<удаленный хост, с которого можно подключаться>. Это может вызвать ошибки доступа, поэтому необходимо понимать, что учетные записи root@localhost и root@192.168.0.15 — отличаются друг от друга.

Создание учетных записей в СУБД «КАТРАПС» выполняется с помощью команды **CREATE USER**.

Данный метод является универсальным. Он позволяет создать пользователя в системе без каких либо прав. После права назначаются командой GRANT.

Пример создания учетной записи:

```
> CREATE USER 'dbuser'@'localhost' IDENTIFIED BY 'password';
```

** в данном примере будет создана учетная запись **dbuser@localhost** (доступ разрешен только с локального компьютера) и паролем **password**.*

После можно задать права командой:

```
> GRANT <тип привилегий> ON <объект> TO <пользователь> <дополнительные опции>;
```

Например:

```
> GRANT ALL PRIVILEGES ON *.* TO 'dbuser'@'localhost';
```

* где:

ALL PRIVILEGES — предоставляет полные права на использование данных.

. — права предоставляются на все базы и все таблицы.

dbuser — имя учетной записи.

localhost — доступ для учетной записи будет предоставлен только с локального компьютера.

Создание учетной записи

Как уже было сказано, имя учетной записи состоит из частей <имя пользователя> + @ + <удаленный хост, с которого можно подключаться>.

Таким образом, если нужно разрешить подключение пользователю dbuser с компьютера 192.168.0.15, нужно создать:

```
> CREATE USER 'dbuser'@'192.168.0.15' IDENTIFIED BY 'password';
```

Чтобы предоставить доступ с любого узла, указывается знак % вместо IP:

```
> CREATE USER 'dbuser'@'%' IDENTIFIED BY 'password';
```

Можно разрешить подключение для подсети, заменив октет знаком %, например:

```
> CREATE USER 'dbuser'@'192.168.0.%' IDENTIFIED BY 'password';
```

** в этом случае мы получим доступ с любого компьютера в сети **192.168.0.0/24**.*

Также необходимо пользователю предоставить привилегии:

```
> GRANT ALL PRIVILEGES ON *.* TO 'dbuser'@'192.168.0.15';
```

Просмотр существующих пользователей и их привилегий

Список пользователей выводится следующей командой:

```
> SELECT user, host FROM mysql.user;
```

Список привилегий для каждого пользователя выводится отдельно:

```
> SHOW GRANTS FOR 'root'@'localhost';
```

** где 'root'@'localhost' — учетная запись, для которой запрошен просмотр привилегии; если опустить FOR, команда выдаст результат для пользователя, под которым выполнено подключение к СУБД.*

Смена пароля

В СУБД «КАТРАПС» смена пароля выполняется командой:

```
> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('New_Password');
```

** в данном примере будет задан пароль New_Password для пользователя root.*

Смена пароля пользователю root после установки

Первый раз пароль задается из командной строки операционной системы:

```
> mysqladmin -u root password
```

Блокировка учетной записи

Блокировка учетной записи позволяет привилегированным администраторам блокировать/разблокировать учетные записи пользователей. Если учетная запись заблокирована, новые клиентские подключения не будут разрешены (существующие подключения не затрагиваются).

Учетные записи пользователей можно заблокировать при создании с помощью инструкции CREATE USER или изменить после создания с помощью инструкции ALTER USER. Например:

```
CREATE USER 'lorin'@'localhost' ACCOUNT LOCK;
```

или

```
ALTER USER 'marijn'@'localhost' ACCOUNT LOCK;
```

Сервер вернет ошибку ER_ACCOUNT_HAS_BEEN_LOCKED, когда заблокированные пользователи попытаются подключиться:

```
mysql -ulorin
```

```
ERROR 4151 (HY000): Access denied, this account is locked
```

Оператор ALTER USER также используется для разблокировки пользователя:

```
ALTER USER 'lorin'@'localhost' ACCOUNT UNLOCK;
```

Оператор SHOW CREATE USER покажет, заблокирована ли учетная запись:

```
SHOW CREATE USER 'marijn'@'localhost';

+-----+
| CREATE USER for marijn@localhost |
+-----+

| CREATE USER 'marijn'@'localhost' ACCOUNT LOCK |
```

Срок действия пароля пользователя

Срок действия пароля позволяет администраторам прекращать срок действия паролей пользователей вручную или автоматически.

Системные переменные

Есть две системные переменные, которые влияют на срок действия пароля:

default_password_lifetime, определяет промежуток времени между требованием пользователю изменить свой пароль. 0, по умолчанию означает, что автоматическое истечение срока действия пароля неактивно;

disconnect_on_expired_password, определяет, разрешено ли клиенту подключаться, если срок действия его пароля истек, или разрешено ли ему подключаться в режиме песочницы, способном выполнять ограниченный набор запросов, связанных со сбросом пароля, в частности SET PASSWORD и SET.

Ограничение срока действия пароля пользователя

Помимо автоматического истечения срока действия пароля, определенного параметром *default_password_lifetime*, время истечения срока действия пароля может быть установлено для отдельного пользователя, переопределяя глобальные значения с помощью операторов CREATE USER или ALTER USER, например:

```
CREATE USER 'monty'@'localhost' PASSWORD EXPIRE INTERVAL 120 DAY;

ALTER USER 'monty'@'localhost' PASSWORD EXPIRE INTERVAL 120 DAY;
```

Ограничения можно отключить с помощью ключевого слова NEVER, например:

```
CREATE USER 'monty'@'localhost' PASSWORD EXPIRE NEVER;

ALTER USER 'monty'@'localhost' PASSWORD EXPIRE NEVER;
```

Установленный вручную предел можно восстановить до системного значения по умолчанию, используя DEFAULT, например:

```
CREATE USER 'monty'@'localhost' PASSWORD EXPIRE DEFAULT;
```

```
ALTER USER 'monty'@'localhost' PASSWORD EXPIRE DEFAULT;
```