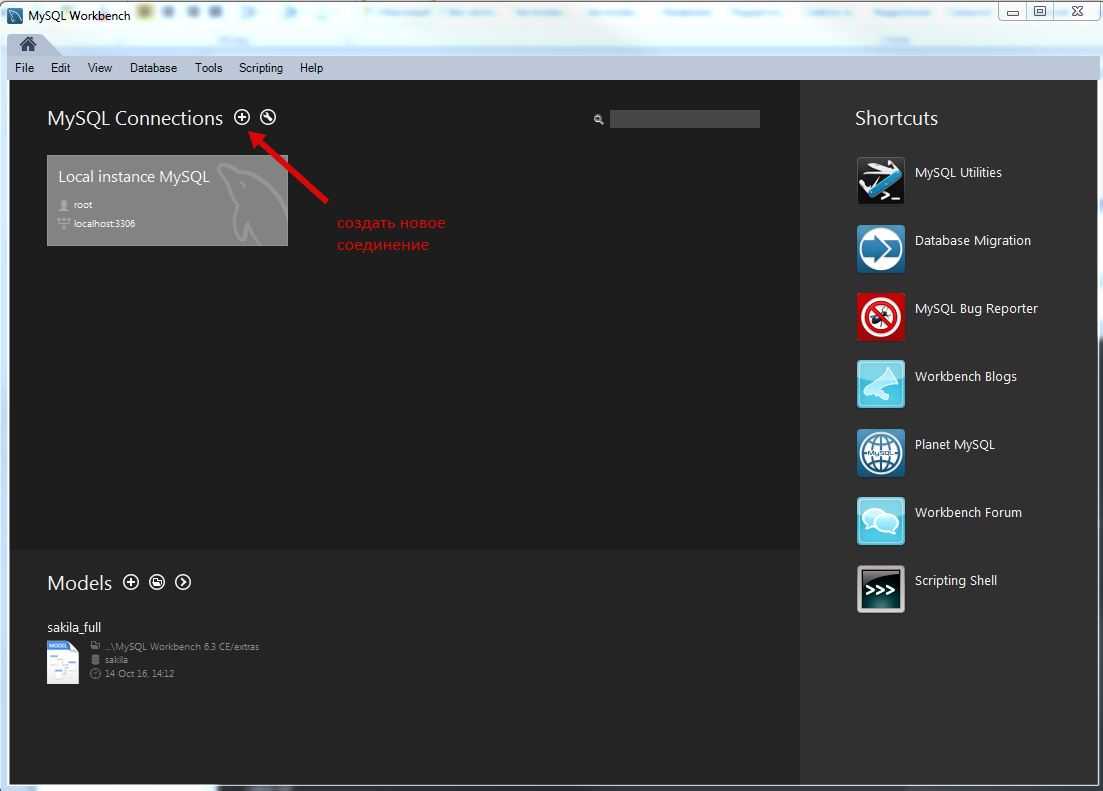
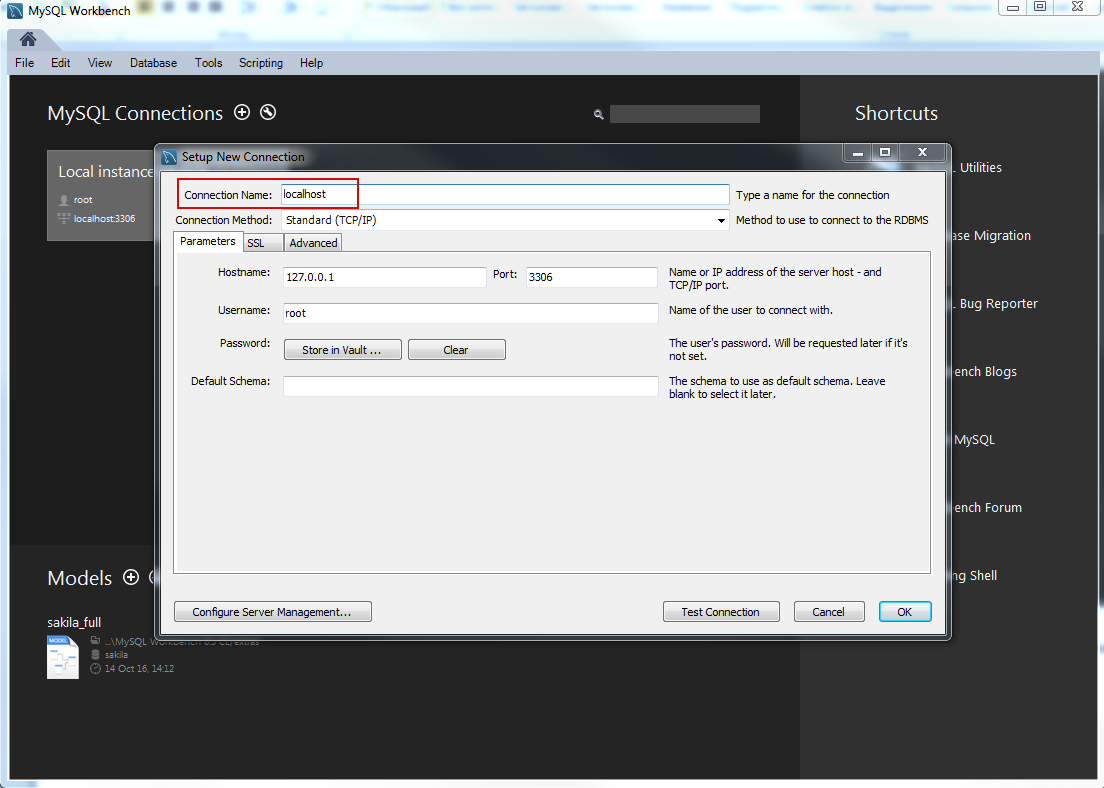
**SQL. Базовый уровень.**

Работать будем с MySQL. Загрузите и установите его дистрибутив с сайта <https://www.mysql.com/>. Откройте **MySQL Workbench**.

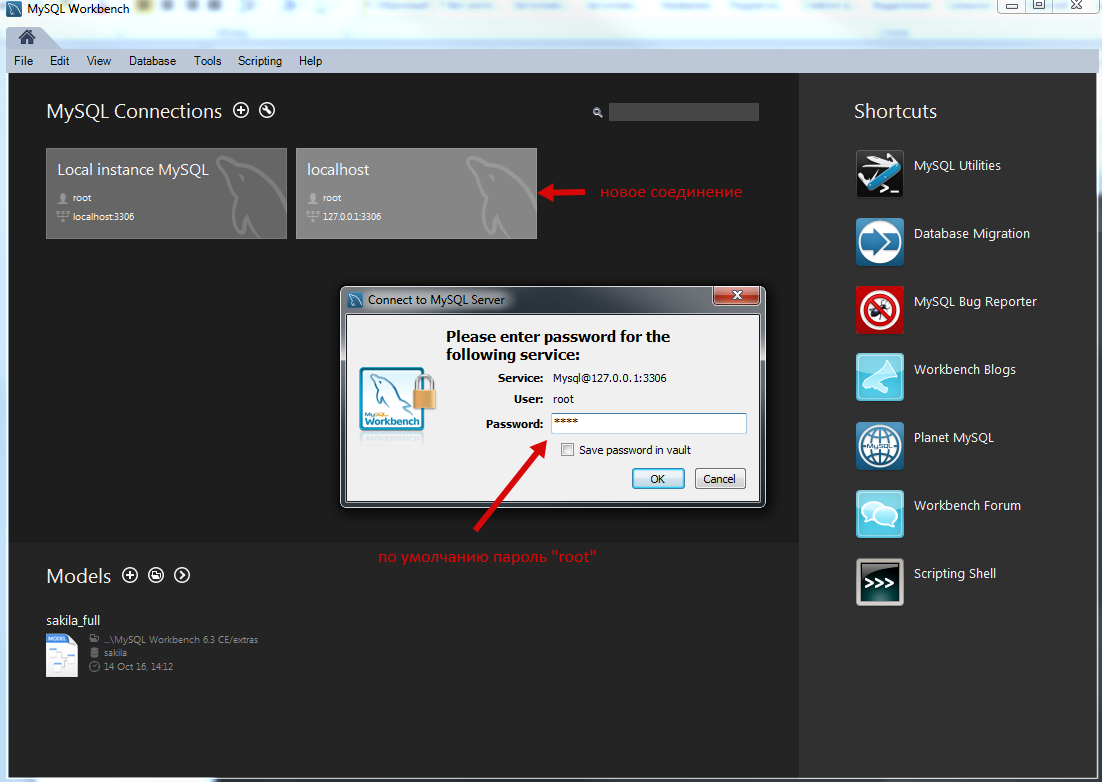


Перед вами так называемое «Домашнее окно». Чтобы создать новое соединение с MySQL Server, нажмите «+».

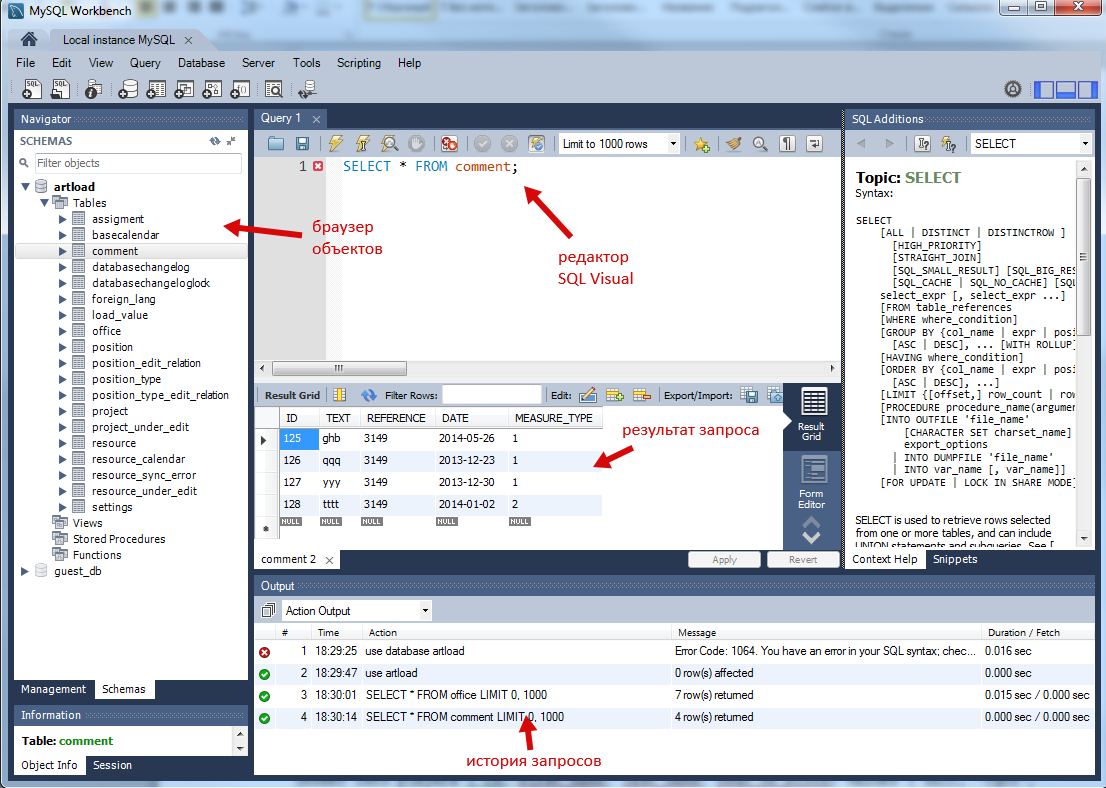


Будем устанавливать соединение с локальным сервером, поэтому никакие параметры менять не придётся. Просто придумайте и введите имя для соединения, после чего нажмите Ок.

Дважды кликните по новому соединению, введите пароль.



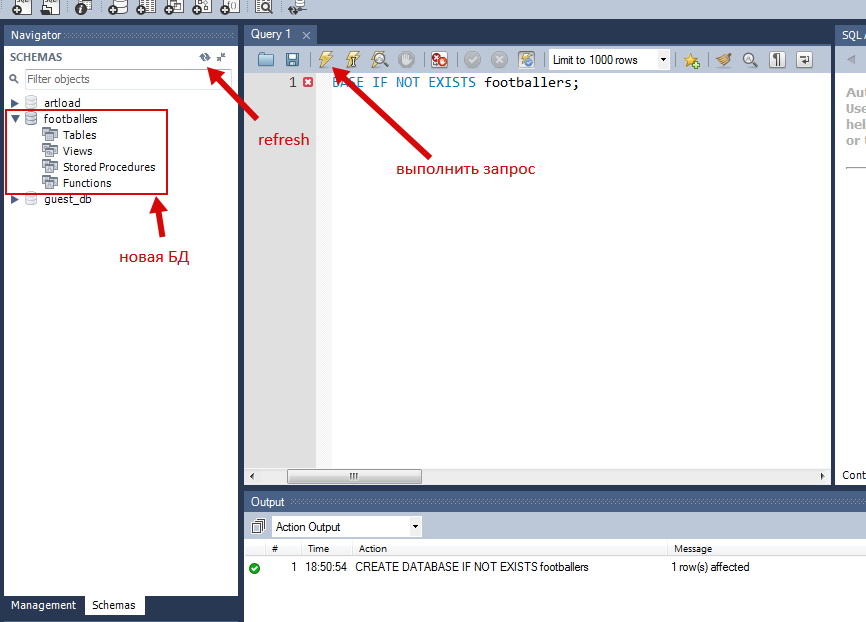
Авторизовавшись, вы можете начать работу. MySQLWorkbench предоставляет обширный функционал для визуализации базы данных, но на данный момент мы ограничимся следующими инструментами:



В MySQL Workbench есть возможность создавать и редактировать схему базы данных при помощи графического интерфейса, но мы будем делать это SQL-запросами.

Создайте новую базу данных:

CREATE DATABASE IF NOT EXISTS footballers;



После выполнения запроса нажмите на кнопку «обновить» в навигаторе, вы увидите, что в списке схем баз данных появилась одна новая. В ней пока нет таблиц, добавим одну, в которой будем хранить список игроков. Для начала при помощи команды USE начнём работу с базой данных:

USE footballers;

Добавим новую таблицу, в которой для каждого футболиста будем хранить следующую информацию:

id – уникальный идентификатор, который будет служить первичным ключом таблицы ( числовое значение, обязательное, автозаполняемое)

first\_name – имя (строковое значение, до 30 символов, обязательное)

last\_name - фамилия (строковое значение, до 30 символов, обязательное)

year\_of\_birth – год рождения

CREATE TABLE IF NOT EXISTS players (

id INT UNSIGNED NOT NULL AUTO\_INCREMENT primary key,

first\_name VARCHAR(30) NOT NULL,

last\_name VARCHAR(30) NOT NULL,

year\_of\_birth YEAR

);

…и добавим в неё несколько игроков:

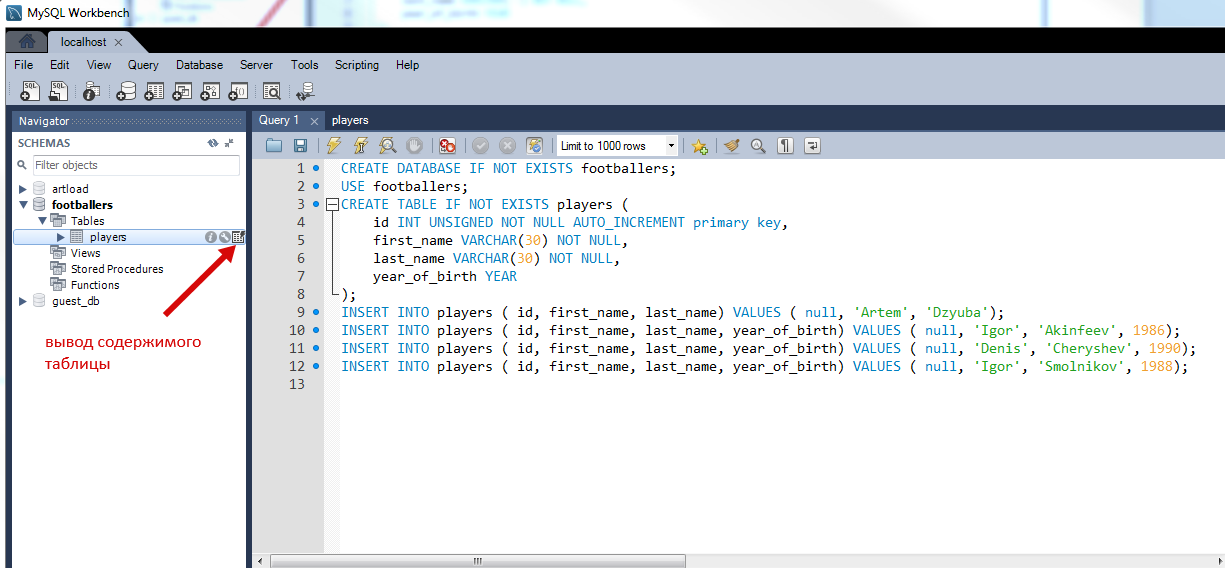
INSERT INTO players ( id, first\_name, last\_name) VALUES ( null, 'Artem', 'Dzyuba');

INSERT INTO players ( id, first\_name, last\_name, year\_of\_birth) VALUES ( null, 'Igor', 'Akinfeev', 1986);

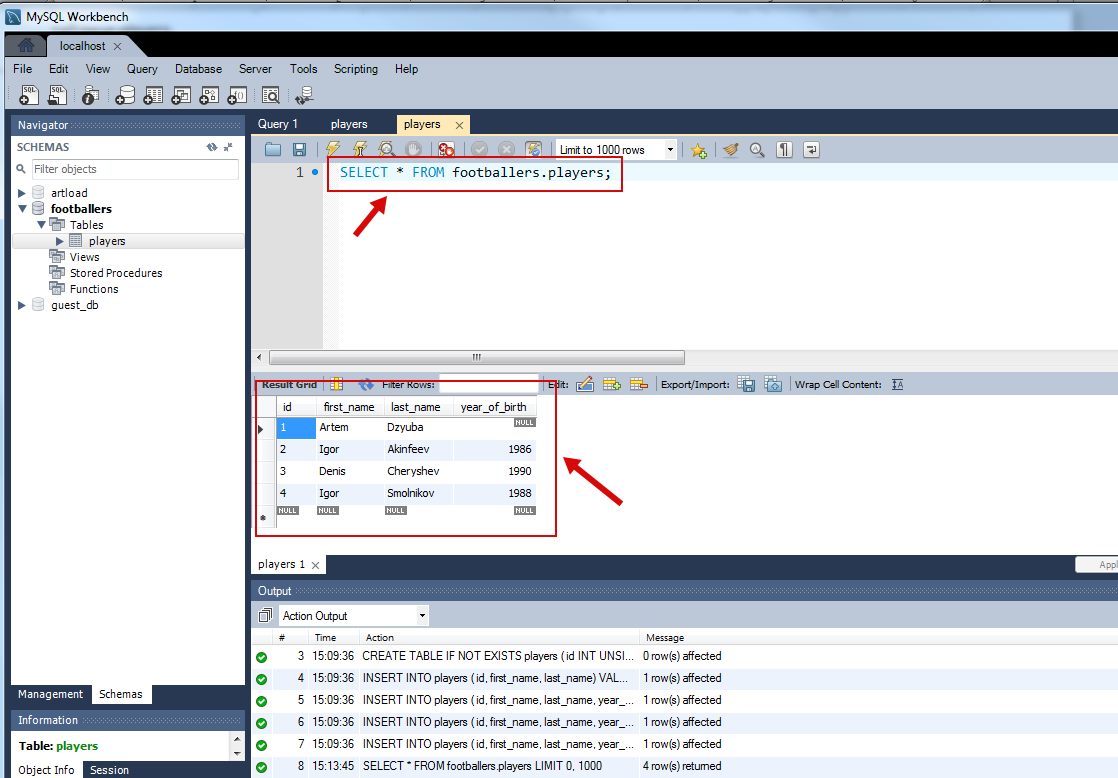
INSERT INTO players ( id, first\_name, last\_name, year\_of\_birth) VALUES ( null, 'Denis', 'Cheryshev', 1990);

INSERT INTO players ( id, first\_name, last\_name, year\_of\_birth) VALUES ( null, 'Igor', 'Smolnikov', 1988);

Обновите отображение схемы БД, вы увидите новую таблицу.



Нажмите на маленький серый значок справа от названия таблицы, откроется новая вкладка, и вы увидите все записи в таблице players.



Обратите внимание на запрос, который был сгенерирован, чтобы получить все поля всех записей в таблице. Попробуйте модифицировать его следующим образом:

SELECT first\_name, last\_name FROM footballers.players;

Выполните запрос, обратите внимание на то, какие данные будут выведены.

Запросите данные по футболистам, год рождения которых мы знаем:

SELECT \* FROM footballers.players WHERE year\_of\_birth IS NOT NULL;

Изменим информацию об Артёме, добавив год его рождения при помощи команды UPDATE:

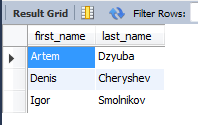
UPDATE players

SET year\_of\_birth=1988

WHERE id=1;

Допустим, нам нужно получить имена и фамилии футболистов, которые родились после 1987 года:

SELECT first\_name, last\_name FROM players WHERE year\_of\_birth > 1987;



Теперь мы хотим добавить информацию о клубах, за которые играют эти ребята. Можно было бы просто добавить в таблицу players новую колонку team, но мы так делать не будем. В соответствии с правилами нормализации баз данных нам стоит добавить новую сущность «teams», создать соответствующую таблицу и добавить связь между записями в этой таблице и футболистами.

CREATE TABLE IF NOT EXISTS teams (

id INT UNSIGNED NOT NULL AUTO\_INCREMENT primary key,

team\_name VARCHAR(20)

);

Добавим клубы:

INSERT INTO teams (id, team\_name) VALUES (null, 'Zenit');

INSERT INTO teams (id, team\_name) VALUES (null, 'CSKA');

INSERT INTO teams (id, team\_name) VALUES (null, 'Villarreal');

Добавим новое поле в таблицу players при помощи команды ALTER TABLE:

ALTER TABLE players ADD team\_id INTEGER NOT NULL;

Установим связь между футболистами и их клубами:

UPDATE players SET team\_id=1 WHERE id IN (1,4);

UPDATE players SET team\_id=2 WHERE id=2;

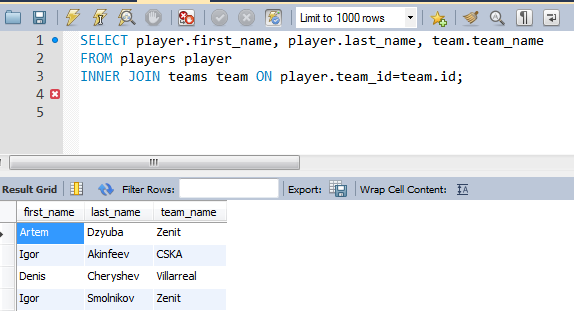
UPDATE players SET team\_id=3 WHERE id=3);

Выведем всех футболистов вместе с их клубами. Для этого воспользуемся таким инструментов, как присоединение (JOIN):

SELECT player.first\_name, player.last\_name, team.team\_name

FROM players player

INNER JOIN teams team ON player.team\_id=team.id;



**Задания для самостоятельного выполнения.**

**Задание 1.** Создайте БД для хранения информации о сотрудниках. Для каждого сотрудника будем хранить следующую информацию:

* Имя
* Фамилия
* Должность
* Зарплата

Все поля обязательные и не могут быть пустыми. У каждого сотрудника может быть только одна должность, могут быть сотрудники с одинаковыми должностями.

Добавьте 3-5 записей в каждую таблицу.

**Задание 2.** Составьте запросы на выборку данных (SELECT):

* Все сотрудники с зарплатами меньше 30 000 рублей.
* Всех сотрудники, занимающие определённую должность (например - дизайнеры), с зарплатой меньше 30 000 рублей.

**Задание 3 (повышенной сложности).** Доработайте схему БД таким образом, чтобы можно было хранить информацию о подчинённых. Без каких-либо ограничений: каждый сотрудник может иметь несколько подчинённых (либо не иметь их вовсе), и сам может иметь несколько начальников (либо вовсе начальников не иметь). Составьте запрос на выборку всех подчинённых определённого сотрудника. (Подсказка: гуглить отношение “many-to-many”).

Результатом выполнения заданий являются составленные вами SQL запросы.