

1.

1. 在新数据库中新建一张 user 表,插入几条数据,属性包含:唯一标识(id),姓名(name)性别(sex),年龄(age),联系方式(phone), 数据如下:

('John Doe', 'Male', 25, '123-456-7890')

('Jane Smith', 'Female', 31, '987-654-3210')

('Bob Johnson', 'Male', 22, '555-123-4567')

Run | New Tab | Active Connection

INSERT INTO user.user_data (name, sex, age, phone) VALUES ('John Doe', 'Male', 25, '123-456-7890'), ('Jane Smith', 'Female', 31, '987-654-3210'), ('Bob Johnson', 'Male', 22, '555-123-4567');

SQL

id	name	sex	age	phone
1	John Doe	Male	25	123-456-7890
2	Jane Smith	Female	31	987-654-3210
3	Bob Johnson	Male	22	555-123-4567

2.

2. 写出 SQL语句,查询 user 表中所有年龄在 20-30 范围内的用户

Run | New Tab | JSON | Active Connection

SELECT * FROM user.user_data WHERE age BETWEEN 20 AND 30;

SQL

	id	name	sex	age	phone
▶	1	John Doe	Male	25	123-456-7890
	3	Bob Johnson	Male	22	555-123-4567

3.

3. 写出SQL语句, 向user表中添加自己的个人信息, 并添加几条和你姓名同姓的虚拟信息。

Run | New Tab | Active Connection

INSERT INTO user.user_data (name, sex, age, phone) VALUES ('wkn', 'Male', 20, '13997709912'), ('wkn', 'Female', 18, '13456748661'), ('wkn', 'Male', 19, '13886557897');

SQL

	id	name	sex	age	phone
▶	1	John Doe	Male	25	123-456-7890
	2	Jane Smith	Female	31	987-654-3210
	3	Bob Johnson	Male	22	555-123-4567
	23	wkn	Male	20	13997709912
	24	wkn	Female	18	13456748661
	25	wkn	Male	19	13886557897
	32	wkn	Male	27	13997324912
	33	wkn	Female	23	124356748661
	34	wkn	Male	28	132346557897
✱	NULL	NULL	NULL	NULL	NULL

4.

4. 写出 SQL 语句,查询 user 表中年龄在 20-30 范围内,名字包含“你的姓氏”的用户,并按照年龄从大到小排序输出

```
Run | New Tab | JSON | Active Connection
SELECT * FROM user.user_data
WHERE age BETWEEN 20 AND 30
AND name LIKE 'wkn'
ORDER BY age DESC;
```

	id	name	sex	age	phone
▶	34	wkn	Male	28	132346557897
	32	wkn	Male	27	13997324912
	33	wkn	Female	23	124356748661
	23	wkn	Male	20	13997709912
*	NULL	NULL	NULL	NULL	NULL

5.

5. 写出 SQL 语句,计算 user 表中所有用户的平均年龄

```
Run | New Tab | JSON | Active Connection
SELECT AVG(age) AS average_age FROM user.user_data;
```

	average_age
▶	23.6667

6.

6. 新建两张表team 表(id,teamName)和score 表(id,teamid,userid,score)。其中score 表中的 teamid 为指向 team表id 的外键,userid 为指向 user表id的外键

```
Run | New Tab | Active Connection
CREATE TABLE user_team (
  id INT AUTO_INCREMENT PRIMARY KEY,
  teamName VARCHAR(100)
);
Run | New Tab | Copy
CREATE TABLE user_score (
  id INT AUTO_INCREMENT PRIMARY KEY,
  teamid INT,
  userid INT,
  score INT,
  FOREIGN KEY (teamid) REFERENCES user_team(id),
  FOREIGN KEY (userid) REFERENCES user_user(id)
);
```

1 • `SELECT * FROM user.user_team;`

Result Grid














Filter Rows:

Edit:   

Export/Import: 

	id	teamName
*	NULL	NULL




user_data user_team user_score x



          Limit to 1000 rows   

1 • `SELECT * FROM user.user_score;`

Result Grid

Filter Rows:

Edit:   

Export/Import:  

	id	teamid	userid	score
*	NULL	NULL	NULL	NULL

7. 在team表中插入合适的记录, 写出 SQL语句,查询 teamName 为"ECNU"的队伍中, 年龄小于 20 的用户们, 结果不得为空。

```
> Run | New Tab | Active Connection
INSERT INTO team (teamName) VALUES ('ECNU'), ('Team B'), ('Team C');
> Run | New Tab | Copy
INSERT INTO score (teamid, userid, score) VALUES
(1, 5, 90), -- wkn, Female, 18
(1, 6, 85), -- wkn, Male, 19
(2, 1, 80), -- John Doe, 25
(2, 2, 88); -- Jane Smith, 31
> Run | New Tab | JSON
SELECT u.name, u.age FROM user u
JOIN score s ON u.id = s.userid
JOIN team t ON s.teamid = t.id
WHERE t.teamName = 'ECNU' AND u.age < 20;
```

user_data user_score x user_data

Limit to 1000 rows

1 • SELECT * FROM user.user_score;

Result Grid Filter Rows: Edit: Export/Import:

	id	teamid	userid	score
	1	2	1	80
▶	2	3	2	88
	5	1	5	90
	6	1	6	85
*	NULL	NULL	NULL	NULL

user_data user_score user_data user_score user_team x

Limit to 1000 rows

```
1 • SELECT * FROM user.user_team;
```

Result Grid | Filter Rows: | Edit: | Export/Import

	id	teamName
▶	1	ECNU
	2	Team B
	3	Team C
*	NULL	NULL

user_data user_score user_data user_score x

Limit to 1000 rows

```
1 • SELECT * FROM user.user_score;
2 • SELECT u.name, u.age FROM user_data u
3 JOIN user_score s ON u.id = s.userid
4 JOIN user_team t ON s.teamid = t.id
5 WHERE t.teamName = 'ECNU' AND u.age < 20;
6
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	name	age
▶	wkn	18
	wkn	19

8. 写出 SQL 语句,计算 teamName为“ECNU”的总分(假设 score 存在 null值,null值默认为 0 加入计算)。

```
> Run | New Tab | JSON | Active Connection
SELECT COALESCE(SUM(s.score), 0) AS total_score
FROM user_score s
JOIN user_team t ON s.teamid = t.id
WHERE t.teamName = 'ECNU';
```

user_data user_score x user_data

Limit to 1000 rows

```
1 • SELECT * FROM user.user_score;
2 • SELECT COALESCE(SUM(s.score), 0) AS total_score
3   FROM user_score s
4   JOIN user_team t ON s.teamid = t.id
5   WHERE t.teamName = 'ECNU';
6
7
```

Result Grid

	total_score
▶	175

9.

9. 写出SQL语句,删除user表中个人信息的记录。

```
> Run | New Tab | Active Connection
SET SQL_SAFE_UPDATES = 0; -- 关闭安全更新模式
> Run | New Tab
DELETE FROM user_score WHERE userid IN (SELECT id FROM user_data WHERE name = 'wkn');
> Run | New Tab
DELETE FROM user.user_data WHERE name = 'wkn';
```

Result Grid

	id	name	sex	age	phone
	1	John Doe	Male	25	123-456-7890
	2	Jane Smith	Female	31	987-654-3210
▶	3	Bob Johnson	Male	22	555-123-4567
*	NULL	NULL	NULL	NULL	NULL