实习一：数据库应用案例设计——知乎APP

小组成员：梁昱桐、吴墨笛、袁梦

本次实习的目标是设计知乎APP的相关数据库，包括列举业务需求、设计ER图、将ER图转换为关系表、用SQL语句实现业务功能。

一、业务需求

条目式逐一列出需求

- **问答**：用户可以发布问题帖或回答问题，并且用户可以对问题关注、评论、邀请某用户回答，对回答点赞、收藏、评论、转载和举报。
- **文章**：用户可以发布文章和对文章点赞、收藏、评论、转载和举报。
- **推送**：关注的用户发布的或与关注的领域相符的问答或文章会被推送到主页。
- **私信**：用户之间可以互相收发信息。

二、ER图设计

实体

- **用户**（用户ID，用户名，IP地址，性别，生日，居住地，关注话题tag，关注数，被关注数）
- **问题帖**（问题ID，问题话题tag，问题内容，回答数，关注数，浏览量）
- **私信**（私信ID，私信内容，发送时间）
- **文章**（文章ID，文章话题tag，文章内容，收藏数，评论数，点赞数，发布时间）

弱实体

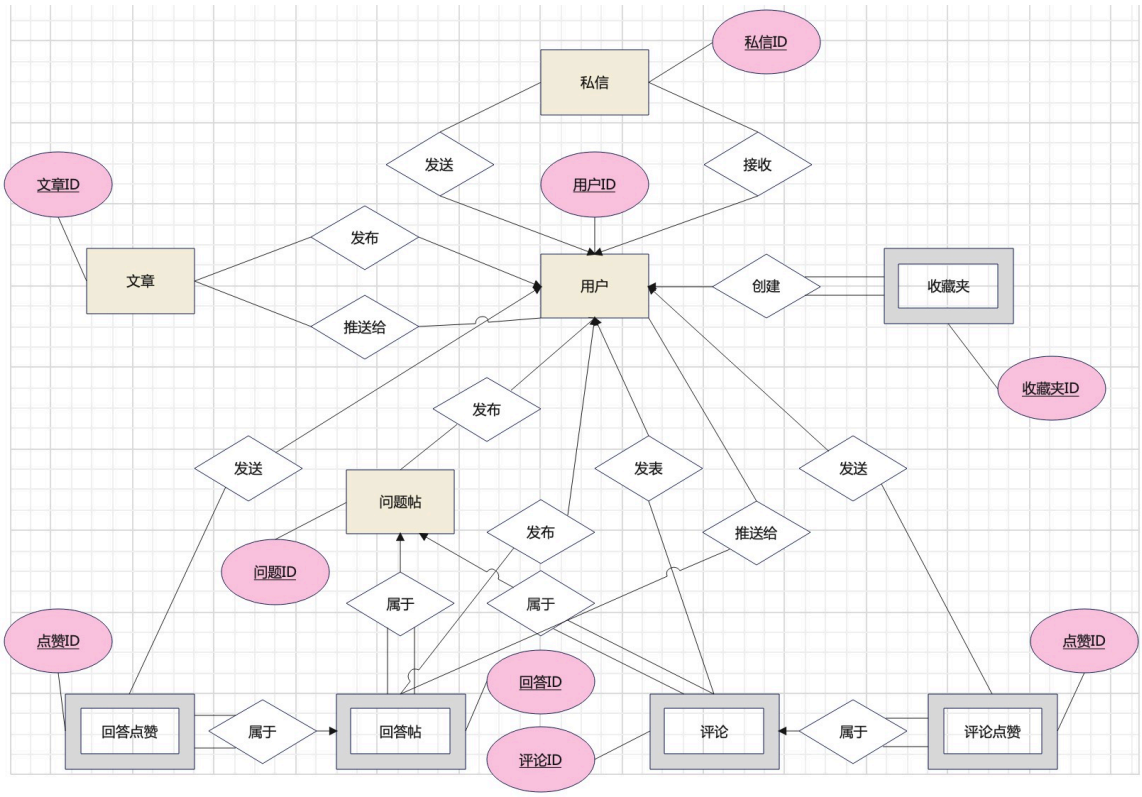
- **回答帖**（回答ID，问题ID，回答内容，收藏数，评论数，点赞数，发布时间）
- **评论** （评论ID，回答ID，评论内容，评论时间，点赞数）
- **回答点赞**（点赞ID，回答ID，点赞时间）
- **评论点赞**（点赞ID，评论ID，点赞时间）
- **收藏夹**（收藏夹ID，用户ID，回答ID，文章ID）（回答ID和文章ID为多值属性）

联系

- 用户**发布**问题帖（一对多）
- 用户**发布**回答帖（一对多）
- 用户**发布**文章（一对多）
- 用户**发表**评论（一对多）
- 评论**属于**回答帖（多对一）

- 用户**发送**回答点赞（一对多）
- 用户**发送**评论点赞（一对多）
- 回答点赞**属于**回答帖（多对一）
- 评论点赞**属于**评论（多对一）
- 用户**创建**收藏夹（一对多）
- 用户**发送**私信（一对多）
- 用户**接收**私信（一对多）
- 用户**关注**用户（多对多）
- 文章**推送给**用户（多对多）
- 回答帖**推送给**用户（多对多）

ER图



三、数据库设计

把ER图转换关系表，并使用SQL创建。为便于执行后面的数据库操作，也在本阶段自行生成了一些样例数据。

先安装 dependency：

```
pip install ipython-sql
```

加载 SQL 并连接服务器

In [ ]:
```
'''
加载SQL
'''
```

```
import pymysql
pymysql.install_as_MySQLdb()
%load_ext sql
```

In [ ]:
```
%%sql

mysql://stu2100013116:stu2100013116@162.105.146.37:43306
show databases;
use stu2100013116;
show tables;
```

```
2 rows affected.
0 rows affected.
12 rows affected.
```

Out[ ]:

Tables_in_stu2100013116
answer_like
answer_post
answer_push
article
article_push
bookmark
comment
comment_like
private_message
question_post
user_follow
user_tb

In [ ]:
```
import pymysql

conn = pymysql.connect(host='162.105.146.37', port=43306, user='stu210001
cursor = conn.cursor()

try:
    # 禁用外键约束
    cursor.execute("SET FOREIGN_KEY_CHECKS=0;")

    # 获取所有表的名称
    cursor.execute("SHOW TABLES")
    tables = cursor.fetchall()

    # 遍历所有表，对每一个表执行清空操作
    for table_name in tables:
        sql = f"TRUNCATE TABLE {table_name[0]}"
        cursor.execute(sql)
```

```
    # 重新启用外键约束
    cursor.execute("SET FOREIGN_KEY_CHECKS=1;")

    conn.commit()
    print("所有表的数据已清空。")
except Exception as e:
    print(f"错误：{e}")
    conn.rollback()
finally:
    cursor.close()
    conn.close()
```

所有表的数据已清空。

In [ ]:
```
%%sql

mysql://stu2100013116:stu2100013116@162.105.146.37:43306
show databases;
use stu2100013116;
show tables;
```

```
2 rows affected.
0 rows affected.
12 rows affected.
```

Out[ ]:

Tables_in_stu2100013116
answer_like
answer_post
answer_push
article
article_push
bookmark
comment
comment_like
private_message
question_post
user_follow
user_tb

建表

9个实体各自对应一张表，要注意的是，5个弱实体的表中应有强实体的主码，代表对强实体的依附。

3个多对多联系需单独创建表，表的主码是联系双方的主码。

剩余的都是一对多联系，不需要创建单独的表，而应该将单方参与实体的码作为多方参与实体的属性。

共有12张表，下面我们进行创建。

In [ ]:

```
%%sql

set @@foreign_key_checks=0;

-- 用户表
drop table if exists user_tb;

CREATE TABLE user_tb
(
    user_id INT PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    ip_address VARCHAR(15),
    gender ENUM('M', 'F', 'Other'),
    birthday DATE,
    residence VARCHAR(100),
    follow_tags VARCHAR(255),
    follow_count INT DEFAULT 0,
    followed_count INT DEFAULT 0
);

-- 问题帖表
drop table if exists question_post;

CREATE TABLE question_post
(
    question_id INT PRIMARY KEY,
    question_tag VARCHAR(50),
    question_content TEXT NOT NULL,
    answer_count INT DEFAULT 0,
    follow_count INT DEFAULT 0,
    view_count INT DEFAULT 0,
    user_id INT,
    constraint fk_question_user foreign key(user_id) references user_tb(u
);

-- 私信表
drop table if exists private_message;

CREATE TABLE private_message
(
    message_id INT PRIMARY KEY,
    message_content TEXT NOT NULL,
    send_time DATETIME NOT NULL,
    sender_id INT,
    receiver_id INT,
    constraint fk_pm_sender foreign key(sender_id) references user_tb(use
    constraint fk_pm_receiver foreign key(receiver_id) references user_tb
);

-- 文章表
drop table if exists article;

CREATE TABLE article
(
```

```
    article_id INT PRIMARY KEY,
    article_tag VARCHAR(50),
    article_content TEXT NOT NULL,
    bookmark_count INT DEFAULT 0,
    comment_count INT DEFAULT 0,
    like_count INT DEFAULT 0,
    publish_time DATETIME NOT NULL,
    user_id INT,
    constraint fk_article_user foreign key(user_id) references user_tb(us
);

-- 回答帖表（弱实体，依赖于问题帖）
drop table if exists answer_post;

CREATE TABLE answer_post
(
    answer_id INT,
    question_id INT,
    answer_content TEXT NOT NULL,
    bookmark_count INT DEFAULT 0,
    comment_count INT DEFAULT 0,
    like_count INT DEFAULT 0,
    publish_time DATETIME NOT NULL,
    user_id INT,
    PRIMARY KEY (answer_id, question_id),
    constraint fk_answer_question foreign key(question_id) references que
    constraint fk_answer_user foreign key(user_id) references user_tb(use
);

-- 评论表（弱实体，依赖于回答帖）
drop table if exists comment;

CREATE TABLE comment
(
    comment_id INT,
    answer_id INT,
    comment_content TEXT NOT NULL,
    comment_time DATETIME NOT NULL,
    like_count INT DEFAULT 0,
    user_id INT,
    PRIMARY KEY (comment_id, answer_id),
    constraint fk_comment_answer foreign key(answer_id) references answer
    constraint fk_comment_user foreign key(user_id) references user_tb(us
);

-- 回答点赞表（弱实体，依赖于回答帖）
drop table if exists answer_like;

CREATE TABLE answer_like
(
    like_id INT,
    answer_id INT,
    like_time DATETIME NOT NULL,
    user_id INT,
    PRIMARY KEY (like_id, answer_id),
    constraint fk_like_answer foreign key(answer_id) references answer_po
    constraint fk_answer_like_user foreign key(user_id) references user_t
```

```
);

-- 评论点赞表（弱实体，依赖于评论）
drop table if exists comment_like;

CREATE TABLE comment_like
(
    like_id INT,
    comment_id INT,
    like_time DATETIME NOT NULL,
    user_id INT,
    PRIMARY KEY (like_id, comment_id),
    constraint fk_like_comment foreign key(comment_id) references comment
    constraint fk_comment_like_user foreign key(user_id) references user_
);

-- 收藏夹表（弱实体，依赖于用户，回答帖和文章）
drop table if exists bookmark;

CREATE TABLE bookmark
(
    bookmark_id INT,
    user_id INT,
    answer_id INT,
    article_id INT,
    PRIMARY KEY (bookmark_id, user_id),
    constraint fk_bookmark_user foreign key(user_id) references user_tb(u
    constraint fk_bookmark_answer foreign key(answer_id) references answe
    constraint fk_bookmark_article foreign key(article_id) references art
);

-- 用户关注关系表（多对多）
drop table if exists user_follow;

CREATE TABLE user_follow
(
    follower_id INT,
    followed_id INT,
    PRIMARY KEY (follower_id, followed_id),
    constraint fk_follower foreign key(follower_id) references user_tb(us
    constraint fk_followed foreign key(followed_id) references user_tb(us
);

-- 文章推送关系表（多对多）
drop table if exists article_push;

CREATE TABLE article_push
(
    article_id INT,
    user_id INT,
    PRIMARY KEY (article_id, user_id),
    constraint fk_article_push_article foreign key(article_id) references
    constraint fk_article_push_user foreign key(user_id) references user_
);

-- 回答帖推送关系表（多对多）
drop table if exists answer_push;
```

```
CREATE TABLE answer_push
(
    answer_id INT,
    user_id INT,
    PRIMARY KEY (answer_id, user_id),
    constraint fk_answer_push_answer foreign key(answer_id) references an
    constraint fk_answer_push_user foreign key(user_id) references user_t
);

set @@foreign_key_checks=1;

show tables;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
12 rows affected.
```

Out[ ]:

Tables_in_stu2100013116
answer_like
answer_post
answer_push
article
article_push
bookmark
comment
comment_like
private_message
question_post
user_follow
user_tb

生成一些数据

In [ ]:

```
%%sql

-- 插入用户数据
INSERT INTO user_tb (user_id, username, ip_address, gender, birthday, res
(1, 'Alice', '192.168.1.1', 'F', '1990-05-15', '北京', '科技,教育', 10, 5),
(2, 'Bob', '192.168.1.2', 'M', '1988-12-22', '上海', '游戏,美食', 5, 8),
(3, 'Carol', '192.168.1.3', 'F', '1995-07-08', '广州', '旅游,时尚', 7, 6),
(4, '李克强', '192.168.1.4', 'Other', '2333-01-04', '火星', '科技', 1, 1000
(5, '习近平', '192.168.1.5', 'M', '1953-06-15', '陕西', '政治', 0, 13000000

-- 插入问题帖数据
INSERT INTO question_post (question_id, question_tag, question_content, a
(1, '科技', '未来的主要能源是什么？', 2, 3, 150, 1),
(2, '科技', '如何理解开普勒第二定律？', 10, 32, 1500, 4),
(3, '教育', '如何提高教育质量？', 1, 2, 50, 1),
(4, '游戏', '2023年值得期待的游戏有哪些？', 3, 4, 200, 2);

-- 插入私信数据
INSERT INTO private_message (message_id, message_content, send_time, send
(1, '你好，Alice！', '2023-01-10 10:00:00', 2, 1),
(2, '最近怎么样？', '2023-01-11 15:30:00', 1, 2),
(3, '明天一起吃饭吗？', '2023-01-12 20:45:00', 3, 1);

-- 插入文章数据
INSERT INTO article (article_id, article_tag, article_content, bookmark_c
(1, '科技', 'AI的未来发展趋势', 5, 2, 15, '2024-04-18 09:00:00', 1),
(2, '科技', 'mamba为什么被拒稿了', 2, 1, 12, '2024-04-18 12:00:00', 4),
(3, '美食', '探索上海的美食天堂', 8, 3, 20, '2024-04-10 12:00:00', 2),
(4, '时尚', '2023年春季时尚趋势', 3, 1, 10, '2024-04-17 11:00:00', 3),
(5, '时尚', '2024年治国理政趋势', 2952, 2952, 2952, '2024-04-18 11:00:00', 5
```

```
-- 插入回答帖数据
INSERT INTO answer_post (answer_id, question_id, answer_content, bookmark
(1, 1, '太阳能和风能将是未来的主要能源', 3, 2, 10, '2023-01-10 14:00:00', 2),
(2, 2, '增加教育资源和改进教学方法', 2, 1, 5, '2023-01-12 16:00:00', 3);

-- 插入评论数据
INSERT INTO comment (comment_id, answer_id, comment_content, comment_time
(1, 1, '非常同意你的观点！', '2023-01-10 14:30:00', 5, 1),
(2, 1, '有没有考虑到水能？', '2023-01-10 15:00:00', 3, 3);

-- 插入回答点赞数据
INSERT INTO answer_like (like_id, answer_id, like_time, user_id) VALUES
(1, 1, '2023-01-10 14:35:00', 1),
(2, 2, '2023-01-12 16:05:00', 2);

-- 插入评论点赞数据
INSERT INTO comment_like (like_id, comment_id, like_time, user_id) VALUES
(1, 1, '2023-01-10 14:40:00', 2),
(2, 2, '2023-01-10 15:05:00', 1);

-- 插入收藏夹数据
INSERT INTO bookmark (bookmark_id, user_id, answer_id, article_id) VALUES
(1, 1, 1, NULL),
(2, 2, NULL, 2),
(3, 3, 2, 3);

-- 插入用户关注关系数据
INSERT INTO user_follow (follower_id, followed_id) VALUES
(1, 2),
(2, 1),
(3, 1);

-- 插入文章推送关系数据
INSERT INTO article_push (article_id, user_id) VALUES
(1, 2),
(2, 1),
(3, 2);

-- 插入回答帖推送关系数据
INSERT INTO answer_push (answer_id, user_id) VALUES
(1, 3),
(2, 1);
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
5 rows affected.
4 rows affected.
3 rows affected.
5 rows affected.
2 rows affected.
2 rows affected.
2 rows affected.
2 rows affected.
3 rows affected.
3 rows affected.
3 rows affected.
2 rows affected.
```

Out[ ]: []

四、数据库操作

列出某项业务功能，给出对应的SQL语句。

结合业务需求，涉及到不同操作以及不同的查询类型即可，有个语句。

使用PyMySQL完成其中的操作

In [ ]:
```
%%sql

SElECT * from user_tb;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
5 rows affected.
```

Out[ ]:

user_id	username	ip_address	gender	birthday	residence	follow_tags	follow_c
1	Alice	192.168.1.1	F	1990-05-15	北京	科技,教育	
2	Bob	192.168.1.2	M	1988-12-22	上海	游戏,美食	
3	Carol	192.168.1.3	F	1995-07-08	广州	旅游,时尚	
4	李克强	192.168.1.4	Other	2333-01-04	火星	科技	
5	习近平	192.168.1.5	M	1953-06-15	陕西	政治	

1. 查询用户关注的所有话题标签

In [ ]:
```
%%sql

SELECT username, follow_tags
FROM user_tb
WHERE user_id = 1;  -- 假设查询用户ID为1的用户关注的话题
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
1 rows affected.
```

Out[ ]:

username	follow_tags
Alice	科技,教育

2. 更新用户居住地

In [ ]:
```
%%sql

SELECT * FROM user_tb;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
5 rows affected.
```

Out[ ]:

user_id	username	ip_address	gender	birthday	residence	follow_tags	follow_c
1	Alice	192.168.1.1	F	1990-05-15	北京	科技,教育	
2	Bob	192.168.1.2	M	1988-12-22	上海	游戏,美食	
3	Carol	192.168.1.3	F	1995-07-08	广州	旅游,时尚	
4	李克强	192.168.1.4	Other	2333-01-04	火星	科技	
5	习近平	192.168.1.5	M	1953-06-15	陕西	政治	

In [ ]:
```sql
%%sql

UPDATE user_tb
SET residence = '深圳'
WHERE user_id = 2;  -- 假设更新用户ID为2的用户居住地为深圳

SELECT * FROM user_tb;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
1 rows affected.
5 rows affected.
```

Out[ ]:

user_id	username	ip_address	gender	birthday	residence	follow_tags	follow_c
1	Alice	192.168.1.1	F	1990-05-15	北京	科技,教育	
2	Bob	192.168.1.2	M	1988-12-22	深圳	游戏,美食	
3	Carol	192.168.1.3	F	1995-07-08	广州	旅游,时尚	
4	李克强	192.168.1.4	Other	2333-01-04	火星	科技	
5	习近平	192.168.1.5	M	1953-06-15	陕西	政治	

3. 查找所有关注某个话题的问题帖，并按浏览量降序排序

In [ ]:
```sql
%%sql

SELECT question_id, question_content, view_count
FROM question_post
WHERE question_tag LIKE '%科技%'  -- 假设查找所有关于“科技”话题的问题
ORDER BY view_count DESC;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
2 rows affected.
```

Out[ ]:

question_id	question_content	view_count
2	如何理解开普勒第二定律?	1500
1	未来的主要能源是什么?	150

4. 插入新的私信内容

In [ ]:
```
%%sql

SElECT * from private_message;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
3 rows affected.
```

Out[ ]:

message_id	message_content	send_time	sender_id	receiver_id
1	你好，Alice！	2023-01-10 10:00:00	2	1
2	最近怎么样?	2023-01-11 15:30:00	1	2
3	明天一起吃饭吗?	2023-01-12 20:45:00	3	1

In [ ]:
```
%%sql

INSERT INTO private_message (message_id, message_content, send_time, send
VALUES (4, '我们下周讨论项目细节', NOW(), 1, 3);  -- 假设用户1发送消息给用户3

SElECT * from private_message;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
1 rows affected.
4 rows affected.
```

Out[ ]:

message_id	message_content	send_time	sender_id	receiver_id
1	你好，Alice！	2023-01-10 10:00:00	2	1
2	最近怎么样?	2023-01-11 15:30:00	1	2
3	明天一起吃饭吗?	2023-01-12 20:45:00	3	1
4	我们下周讨论项目细节	2024-04-19 05:48:14	1	3

5. 统计每个用户的文章平均点赞数

In [ ]:
```
%%sql

SELECT user_id, AVG(like_count) AS avg_likes
FROM article
GROUP BY user_id;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
5 rows affected.
```

Out[ ]:

user_id	avg_likes
1	15.0000
2	20.0000
3	10.0000
4	12.0000
5	2952.0000

6. 查找某用户发送的所有私信内容及接收者用户名

In [ ]:
```
%%sql

SELECT pm.message_content, pm.send_time, u.username AS receiver_name
FROM private_message AS pm
JOIN user_tb AS u ON pm.receiver_id = u.user_id
WHERE pm.sender_id = 1;  -- 假设查询用户ID为1发送的所有私信
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
2 rows affected.
```

Out[ ]:

message_content	send_time	receiver_name
最近怎么样?	2023-01-11 15:30:00	Bob
我们下周讨论项目细节	2024-04-19 05:48:14	Carol

7. 查找某个用户收到的所有点赞数量（包括评论和回答的点赞）

In [ ]:
```
%%sql

SELECT SUM(t.total_likes) AS total_received_likes
FROM (
    SELECT c.like_count AS total_likes
    FROM comment AS c
    WHERE c.user_id = 1
    UNION ALL
    SELECT a.like_count
    FROM answer_post AS a
    WHERE a.user_id = 1
) AS t;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
1 rows affected.
```

Out[ ]:

total_received_likes
5

8. 查询最近一周内发布的文章及其作者

In [ ]:
```
%%sql
```

```
SELECT a.article_id, a.article_content, u.username
FROM article AS a
JOIN user_tb AS u ON a.user_id = u.user_id
WHERE a.publish_time >= DATE_SUB(CURDATE(), INTERVAL 7 DAY);
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
4 rows affected.
```

Out[ ]:

article_id	article_content	username
1	AI的未来发展趋势	Alice
2	mamba为什么被拒稿了	李克强
4	2023年春季时尚趋势	Carol
5	2024年治国理政趋势	习近平

9. 统计每个话题下的问题数量

In [ ]:

```
%%sql

SELECT question_tag, COUNT(*) AS question_count
FROM question_post
GROUP BY question_tag;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
3 rows affected.
```

Out[ ]:

question_tag	question_count
科技	2
教育	1
游戏	1

10. 更新问题帖的回答数量

In [ ]:

```
%%sql
SELECT * FROM question_post;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
4 rows affected.
```

Out[ ]:

question_id	question_tag	question_content	answer_count	follow_count	view_cou
1	科技	未来的主要能源是什么?	2	3	1
2	科技	如何理解开普勒第二定律?	10	32	15
3	教育	如何提高教育质量?	1	2	
4	游戏	2023年值得期待的游戏有哪些?	3	4	2

In [ ]:

```
%%sql

UPDATE question_post
SET answer_count = answer_count + 1
WHERE question_id = (SELECT question_id FROM answer_post WHERE answer_id
SELECT * FROM question_post;
```

```
 * mysql://stu2100013116:***@162.105.146.37:43306
1 rows affected.
4 rows affected.
```

Out[ ]:

question_id	question_tag	question_content	answer_count	follow_count	view_cou
1	科技	未来的主要能源是什么?	3	3	1
2	科技	如何理解开普勒第二定律?	10	32	15
3	教育	如何提高教育质量?	1	2	
4	游戏	2023年值得期待的游戏有哪些?	3	4	2