主键不能为空也不能重复，一个表里可以没有主键

外键表上的数据来源于 另一张表

**数据库设计过程**

需求分析->概要设计->详细设计->代码编写->绘制ER(实体关系)图->确定表字段

**mysql数据类型**

1. tinyint，smallint，int，bigint，
2. decimal(p,s)[定点型]，[float(p,s)，double(p,s)][浮点型]，
3. date，time，datetime，
4. char[定长]，varchar[变长]，text

汉字占两个字节

truncate删的比较彻底，自动增长列回归至初始，不能加条件

#查询玩家表中全部的行和列

#SELECT \* FROM users;

#从玩家表中查询玩家的qq和昵称

#SELECT user\_qq,user\_name FROM users;

#查询玩家QQ和昵称，并显示为“玩家QQ”和“玩家昵称”起别名

#SELECT `user\_qq` AS 玩家QQ,`user\_name` AS '玩家昵称' FROM users;#as可以省略,可以在字段名上加`,并不是所有的数据库都支持``

#显示参与了游戏的玩家QQ

#SELECT DISTINCT user\_qq,gno FROM scores;

#显示玩家表中第3至第5条数据

#SELECT \* FROM users LIMIT 2,3;#从0开始

#查询QQ号为12301的玩家信息、查询分数大于2500分的数据

#SELECT \* FROM users WHERE USER\_QQ = '12301';

#SELECT \* FROM scores WHERE SCORE > 2500;#<>不等于 = 等于

#查询游戏编号为2且分数大于4000分的分数信息

#SELECT \* FROM scores WHERE GNO = 2 AND SCORE > 4000;#AND OR NOT与或非

#查询游戏编号为1和2的分数信息

#SELECT \* FROM scores WHERE GNO = 1 OR GNO = 2;

#查询游戏编号为1和2的分数信息

#SELECT \* FROM scores WHERE GNO IN (1,2);

#查询游戏编号不为1和2的分数信息

#SELECT \* FROM scores WHERE GNO NOT IN (1,2);

#查询分数在2500(含)到3000(含)的分数信息

#SELECT \* FROM scores WHERE SCORE BETWEEN 2500 AND 3000;#包含边界,小数在左边大数在右边

#查询分数不在2500(含)到3000(含)的分数信息

#SELECT \* FROM scores WHERE SCORE NOT BETWEEN 2500 AND 3000;

#查询1993年1月1日到1995年7月31日出生的玩家

#SELECT \* FROM users WHERE USER\_BIRTHDAY BETWEEN '1993-01-01' AND '1995-07-31';

#通配符

#\_表示一个字符

#%表示任意多个字符

#[]表示一个字符,可以指定范围,例如[0,1,3],正则表达式

#[^]否定形式

#查询所有姓孙的玩家信息

#SELECT \* FROM users WHERE USER\_NAME LIKE '孙%';

#查询所有非姓孙的玩家信息

#SELECT \* FROM users WHERE USER\_NAME NOT LIKE '孙%';

#查询生日为NULL的玩家信息

#SELECT \* FROM users WHERE USER\_BIRTHDAY IS NULL;

#查询生日不为NULL的玩家信息

#SELECT \* FROM users WHERE USER\_BIRTHDAY IS NOT NULL;

#查询分数表中编号为1的所有分数信息，并按照分数升序排序

#SELECT \* FROM scores WHERE GNO = 1 ORDER BY SCORE ASC;

#SELECT \* FROM scores ORDER BY SCORE DESC;

#查询分数表所有信息，并按照游戏编号的升序和分数的降序进行排序

#SELECT \* FROM scores ORDER BY GNO ASC ,SCORE DESC;

#查询玩家表中一共有多少名玩家信息

#SELECT COUNT(USER\_QQ) AS '玩家数量' FROM users;

#SELECT COUNT(\*) AS '玩家数量' FROM users;

#查询QQ号是12301的玩家游戏的总分数

#SELECT SUM(SCORE) FROM scores WHERE USER\_QQ = 12301;

#查询QQ号是12302的玩家的平均分数

#SELECT AVG(SCORE) FROM scores WHERE USER\_QQ = 12302;

#查询游戏编号是1的游戏的最高分数

#SELECT MAX(SCORE) FROM scores WHERE GNO = 1;

#SELECT MIN(SCORE) FROM scores WHERE GNO = 1;

#查询QQ号是12302的玩家的总分、平均分和最高分

#SELECT SUM(SCORE),AVG(SCORE),MAX(SCORE) FROM scores WHERE USER\_QQ = 12302;

#查询每个玩家的总分数、平均分数、最高分数

#SELECT MAX(SCORE),AVG(SCORE),MAX(SCORE) from scores GROUP BY USER\_QQ;

#查询每个玩家的平均分数，并显示玩家QQ号和平均分数

#SELECT USER\_QQ,AVG(SCORE)from scores GROUP BY USER\_QQ;

#查询平均分数大于4000的玩家QQ号、总分数、平均分数

#SELECT USER\_QQ,AVG(SCORE),MAX(SCORE) from scores GROUP BY USER\_QQ HAVING AVG(SCORE) > 4000;#数据分组之后需用having，不能用where

#查询所有用户的平均分数，和总分数,并按平均分数倒序排列

#SELECT AVG(SCORE),SUM(SCORE) FROM scores GROUP BY USER\_QQ ORDER BY AVG(SCORE) DESC;

#SELECT AVG(SCORE),SUM(SCORE) FROM scores WHERE GNO = 2 GROUP BY USER\_QQ HAVING AVG(SCORE) > 1000 ORDER BY AVG(SCORE) DESC;

#查询分数信息，显示玩家昵称、游戏名称和分数

#隐式内连接

#SELECT USER\_NAME,Gname,SCORE FROM users,scores,games WHERE users.USER\_QQ = scores.USER\_QQ AND games.GNO = scores.GNO;

#显式内连接(运行效率高一些)

#SELECT USER\_NAME,GName,SCORE from users U INNER JOIN scores S ON U.USER\_QQ = S.USER\_QQ INNER JOIN games G ON S.GNO = G.GNO;#没有对应的数据就放弃连接

#查询每个玩家的昵称、总分和平均分

#SELECT USER\_NAME,SUM(SCORE),AVG(SCORE) from users U INNER JOIN scores S ON U.USER\_QQ = S.USER\_QQ GROUP BY S.USER\_QQ;

#查询平均分数大于3500的分数信息，显示玩家昵称，总分数、平均分数，并按照平均分数降序排列

#SELECT USER\_NAME,SUM(SCORE),AVG(SCORE) from users U INNER JOIN scores S ON U.USER\_QQ = S.USER\_QQ GROUP BY S.USER\_QQ HAVING AVG(SCORE) > 3500 ORDER BY SCORE DESC;

#查询所有玩家关于5号游戏的分数信息

#左外连接

#SELECT USER\_NAME,'5' AS 'GNO',SCORE FROM users LEFT OUTER JOIN scores ON users.USER\_QQ = scores.USER\_QQ AND GNO = 5;

#查询游戏类型是“棋牌类”的游戏的分数信息

#SELECT s.\* FROM games g,scores s WHERE g.GNO = s.GNO AND g.GType = '棋牌';

#SELECT \* FROM scores WHERE GNO IN(SELECT GNO FROM games WHERE GType = '棋牌');

#查询没有参与5号游戏的玩家QQ

#SELECT USER\_QQ from scores WHERE GNO <> 5;

#查询QQ号是”12301”的玩家所有分数并计算出总分数和平均分数，并显示到同一结果集中

#SELECT \* FROM scores WHERE USER\_QQ = '12301' UNION SELECT '总分','' ,SUM(SCORE) FROM scores UNION SELECT '平均分','' ,AVG(SCORE) FROM scores;

#SELECT USER\_NAME FROM users UNION ALL SELECT GName FROM games;#不加all都显式出来，加all后若遇见重复的则保留一个