SQL语句不区分大小写

1 关系模型

1.1 主键

主键：能通过该字段唯一标识记录，该字段不允许有重复
联合主键：允许通过多个字段唯一标识记录，即两个或更多的字段都设置为主键。这种主键称为联合主键。

1.2 外键

可以把数据与另一个表关联起来，这种列称为外键。
FOREIGN KEY (class id) 指定了class id作为外键
REFERENCES CLASSES (id) 指定了这个外键将关联到CLASSES 表中的id列

1.3 索引

ALTER TABLE students
ADD INDEX id_score (score);
在students表中增加score字段的索引id_score。若索引有多列，则：
ALTER TABLE students
ADD INDEX id_name_score (name, score);

- 可以对一个表建立多个索引。索引的优点是提高了查询效率，缺点是在插入、更新和删除记录时需要同时修改索引。所以索引越多，插入，更新，删除记录的速度就越慢
- 对于主键，关系数据库会自动对其创建主键索引。

唯一索引：一些列根据业务要求，具有唯一性约束，就可以给该列添加一个唯一索引
ALTER TABLE students
ADD UNIQUE INDEX uni_name (name);

对该列添加一个唯一约束：
ALTER TABLE students
ADD CONSTRAINT uni_name UNIQUE (name);
其中uni_name为约束名。

2 查询数据库

SELECT * from table_name

2.1 条件查询

使用WHERE条件来设定查询条件:
SELECT * FROM <表名> WHERE <条件表达式>
如:
SELECT * from students where score > 80;

条件表达式可以用AND/OR连接，以及NOT <条件> 表示不符合该条件的记录。
NOT <条件>实例：SELECT * from students where NOT class_id = 2;

若要组合三个或者更多条件，需要使用括号。
SELECT * from students WHERE (score > 80 AND score < 90) OR gender = 'M';
如果不加括号，条件运算按照NOT、AND、OR的优先级进行，即NOT优先级最高，其次是AND，最后是OR。加上括号可以改变优先级。

2.2 投影查询

2.3 排序

ORDER BY 列名/ORDER BY 列名 DEC：前一个为递增，后一个为递减
注意：ORDER BY需要跟在WHERE条件的后面

2.4 分页查询

LIMIT <每页显示的记录数> OFFSET <相对于第一条记录的偏移 (从零开始)>
放在ORDER BY的后面

2.5 聚合查询

查询的记录数：COUNT()函数
SELECT COUNT(*) from strudents

其他聚合函数：

- SUM()
- AVG()
- MAX()
- MIN()

分组聚合功能，使用GROUP BY <列名>
根据列名进行分组，相同的为一组，再使用聚合函数
SELECT class_id COUNT(*) from strudents GROUP BY class_id;

使用多个列进行分组
select class_id, gender, COUNT(*) from students GROUP BY class_id, gender;

3 事务

能，被称为数据把多个操作作为一个整体进行操作的功库的事务。数据库事务可以确保该事务范围内的所有操作都可以全部成功或者全部失败。

事务具有ACID这4个特性：
A: 原子性，事务所包含的所有操作要么全部成功，要么全部失败回滚。
C: 一致性，事务完成后，所有数据的状态都是一致的。
I: 隔离性，隔离性是当多个用户并发访问数据库时，比如操作同一张表时，数据库为每一个用户开启的事务，不能被其他事务的操作所干扰，多个并发事务之间要相互隔离。
D: 持久性，即事务完成后，对数据库数据的修改被持久化存储。

对于单条SQL语句，数据库系统自动将其作为一个事务执行，这种事务被称为隐式事务
显示事务，手动把多条SQL语句作为一个事务执行，BEGIN开启一个事务，COMMIT提交一个事务

```
BEGIN;
SQL语句;
COMMIT;
```

3.1 隔离级别

- Read Uncommitted: 是隔离级别最低的一种事务级别。在这种隔离级别下，一个事务会读到另一个事务更新后但未提交的数据，会有脏读。
- Read commited: 只有在事务提交后，其更新结果才会被其他事务看见，可以解决脏读问题，但是可能会遇到不可重复读问题。
  不可重复读问题是指：在一个事务内，多次读同一数据，在这个事务还没结束时，如果另一个事务恰好修改了这个数据，那么，在第一个事务中，两次读取的数据就可能不一致。
- Repeatable Read: 在一个事务中，对于同一份数据的读取结果总是相同的，无论是否有其他事务对这份数据进行操作，以及这个事务是否可以提交，可以解决脏读、不可重复读。
- Serialization: 事务串行化执行，隔离级别最高，牺牲了系统的并发性，可以解决并发事务的所有问题。
- inner join: 只返回同时存在两张表的数据
- right outer join: 返回右表存在的行，若某一行只在右表存在，那么结果就会以NULL填充剩下的字段
- left outer join: 返回左表都存在的行。