select \* from table limit (start-1)\*limit,limit; 其中start是页码，limit是每页显示的条数 (全表扫描,速度会很慢 且 有的数据库结果集返回不稳定(如某次返回1,2,3,另外的一次返回2,1,3).

SELECT \* FROM product WHERE ID > =(select id from product limit 866613, 1) limit 20 很快

**走索引快速定位**

SELECT \* FROM your\_table WHERE pk>=1000 ORDER BY pk ASC LIMIT 0,20

MySQL的DESC只是语法级别支持，优化器和存储引擎没有支持，内部处理时只是悄悄把DESC转为ASC，然后把处理结果倒序输出，所以DESC操作本身更耗时

Db优化策略:

表字段避免null值出现，null值很难查询优化且占用额外的索引空间，推荐默认数字0代替null。

尽量使用INT而非BIGINT，如果非负则加上UNSIGNED（这样数值容量会扩大一倍），当然能使用TINYINT、SMALLINT、MEDIUM\_INT更好。

使用枚举或整数代替字符串类型

尽量使用TIMESTAMP而非DATETIME

单表不要有太多字段，建议在20以内

用整型来存IP

Mysql模糊查询

%：表示任意0个或多个字符。可匹配任意类型和长度的字符，有些情况下若是中文，请使用两个百 分号（%%）表示。

Where in ()在这个list里

XOR 异或，只能一个成立

Mysql round()

ROUND(7253.86, 1) -> 7253.9

ROUND(7253.86,-3) -> 7000

<> 不相等

LEFT(s,n) allows you to extract n characters from the start of the string s.

搜索条件有’ 需要用’’代替

Distinct 唯一，每个值只能出现一次

[mysql](http://lib.csdn.net/base/mysql)对于类型为varchar数据默认不区分大小写，但如果该字段以“\*\_bin”编码的话会使mysql对其区分大小写。

vim /etc/my.cnf lower\_case\_table\_names=0（0：大小写敏感；1：大小写不敏感）

Group by 作用晚于 where,group by指的是按组计算

Having 可以对聚合后的值进行筛选 having sum(key)>1000

**外连接时**，on条件是在生成临时表时使用的条件，它不管on中的条件是否为真，**都会返回左边表**中的记录。而where条件是在临时表生成好后，再对临时表进行过滤的条件。（内连接则需要匹配）

on是生成临时表时使用的条件，上面我们采用的是左外连接，**左外连接是以左表为基础的，左表的记录将会全部表示出来**,**而右表只会显示符合搜索条件的记录**。也就是说emp是左表，dept是右表，条件是emp的deptno与dept中的deptno相等且为40时才连接，**但emp表中不存在deptno为40的记录，也就是右表没有符合条件的记录，而记录不足的地方均用NULL来补充。**  
**而where是在临时表生成好后，再对临时表进行过滤。也就是说emp表与dept的连接条件只是emp的deptno与dept中的deptno相等,然后在对生成的临时表进行筛选，由于emp表中不存在deptno为40的记录，所以未找到符合条件的记录。**

表连接：

, 也是 cross join, 全组合

,也是Inner join，不过比cross join出现时，需要更多的限制条件，on a.id = b.id

联合查询理解成把多个查询语句的查询结果集中在一起显示，语法示例如下。

select column\_name(s) from table\_name1 UNION select column\_name(s) from table\_name2 #默认以table1的字段名称为主

union all能把重复的字段也显示

