1. mysql/oracle等都是DBMS数据库管理系统，SQLDevelop/navicat等是数据库软件集成开发环境。
2. 与数据文件打交道的只有服务器软件。请求来自客户端软件。
3. 在连接到mysql时，没有任何数据库打开，需要使用use来选择数据库。
4. SELECT DISTINCT ID,NAME FROM …;ID,NAME组合不同就行。
5. SELECT ID FROM USER LIMIT 5;返回5行。
6. SELECT ID FROM USER LIMIT 4,5;从第4行开始，返回5行。
7. SELECT ID FROM USER LIMIT 4 OFFSET 5;从第5行开始，返回4行。
8. 6和7中的第4行是，从0行开始数的。
9. 重命名规则与ORACLE一样。
10. ORDER BY和ORACLE一样。
11. MYSQL不区分大小写。Select \* from user where name = ‘wa’|’WA’;一样。
12. BETWEEN包括开始值和结束值。
13. IN操作与ORACLE一样。优点：
    1. 在使用长的合法选项清单时，IN操作符的语法更清楚且更直观；
    2. 在使用IN时，计算的次序更容易管理（因为使用的操作符更少）；
    3. IN操作符一般比OR操作符清单执行更快；
    4. 最大优点是可以包含其他SELECT语句。
14. 通配符和ORACLE一样。
15. 正则表达式：
16. SELECT \* FROM USER WHERE ID REGEXP ‘100|2’;匹配字符里100或2
17. SELECT \* … ID REGEXP ‘[ABC] TOM;可以匹配 A TOM,B TOM,C TOM;
18. [^ABC]匹配不在ABC范围的字符；
19. 匹配范围：[0-9] [A-Z]
20. 匹配特殊字符：\\. \\\f
21. Concat()连接字符串

Select Concat(id, ’,’,name,’.’) str from a;

1. 在使用日期和字符串匹配时注意
2. where Date(order\_date) = ‘2018-09-01;//不管order\_date是什么时间类型，都要用date()来，匹配，后面形式的字符串，从而不会出错。’
3. where Time(order\_date)=’2018-0-01 23:34:34’;//同理
4. where Year(order\_date)=2018 and Month(order\_date)=9
5. count(\*) 可取null，count(column) 不可取null,与oracle一样。
6. group by与 having与oracle一样。
7. 联结
   1. Where a.id = b.id;//完全相等才返回
   2. From a Inner join b on a.id = b.id;//与a)完全一样
   3. 用自连接代替子查询

Where a.id = (select \*)🡪where a.id = b.id and b.name = ‘’;

* 1. From a left outer join b on a.id = b.id;//以a.id为基准，b没有的显示为null
  2. From a right outer join b on a.id = b.id;//以b.id为基准，从右边选择表为基表

1. Union和oracle一样
2. 全文索引
3. 通配符和正则表达式一般不走索引
4. 更强大的、更灵活的搜索
5. 更智能化的结果

要求：必须索引被搜索的列，而且要随着数据的改动不断地重新索引。

创建：在创建表的时候用FULLTEXT（字段1,…），来启用该字段的全文索引。

若要导入数据，应该在导入数据后才创建索引。

搜索：SELECT NOT\_TEXT FROM USER WHERE MATCH(NOT\_TEXT) AGAINST(‘RA’);

1. 传递给MATCH的值一定时与FULLTEXT中的完全一样，多个就多个，顺序一样
2. 不区分大小写，除非使用BINARY方式。
3. 全文搜索以文本匹配的良好程度排序。
4. 对于自增的ID，可以通过NULL来插入。数据库自动忽略。但其他列必须允许NULL才行。
5. 优化INSERT操作，提高整体性能
   1. INSERT LOW\_PRIORITY INTO，如果数据索引最重要，用这个降低INSERT优先权。
   2. 多条插入：INSERT INTO A VALUES(…),(…),(…),…;
   3. INSERT INTO SELECT…;列明不需要相同，数据类型匹配就行。
6. 更新操作
7. 如果更新多行，要使一行发生错误时，更新继续，要用IGNORE关键字：

UPDATE IGNORE USER …;

1. 为了删除某个列的值，可以设置为NULL
2. TRUNCATE TABLE，实际上是删除表，在创建表。DELETE是逐行删除数据。
3. AUTO\_INCREMENT列，每个表只允许一个，且必须被索引，如成为主键。
4. 可以直接用INSERT的值，来指定AUTO\_INCREMENT列，只要是唯一的，但下一次的增量将使用插入的值，所以当插入值不是最大时，会出错？
5. MYSQL默认值只支持常量。不支持函数。
6. 引擎类型
   1. INNODB：可靠的事务处理引擎。
   2. MEMORY：功能上等同于MYISAM，但数据存储在内存中，速度很快。
   3. MYISAM：不支持事务。
   4. 外键不能跨引擎。
7. 更新表
8. 一般表不应该被更新
9. 一般用于定义外键

ALTER TABLE USER ADD CONSTRAINT KEY\_NAME FOREIGN KEY(ID) REFERENCES COMPANY(USER\_ID);

1. 视图和ORACLE一样
2. 存储过程

优点：简单、安全、高性能

缺点：编写复制，访问权限

1. MYSQL存储过程
   1. 无参：CREATE PROCEDURE PRO()
   2. 有参：CREATE PROCEDURE PRO( IN P1 CHAR(2),OUT P2 CHAR(2))
   3. 调用
      1. 无参：CALL PRO();
      2. 有参：CALL PRO(‘AB’,@NAME);
      3. 用SELECT @NAME;获取存储过程输出值。
2. 游标
   1. 只能用于存储过程。
   2. 主要用于交互式应用。
   3. 和ORACLE一样。
3. 触发器和ORACLE一样。
4. 事务
   1. MYSQL自动提交，但事务里，需要手动。
   2. 可以通过SET AUTOCOMMIT=0;来设置本连接的更改不自动提交。
   3. 简单的ROLLBACK和COMMIT可以写入或撤销整个事务。
   4. 通过SAVEPOINT P1; ROLLBACK TO P1;更灵活的控制回退位置。
   5. 通过START TRANSACTION开始事务，COMMIT，ROLLBACK提交/关闭事务。
5. 如果从一个表中删除大量数据，应该使用OPTIMIZE TABLE来回收所用的空间。
6. 日志文件
   1. 错误日志
   2. 查询日志
   3. 二进制日志
   4. 缓慢查询日志