1、 假设 a,b 表中有相同的字段，如何从一个表中减去另一个表中的记录？

select a.passenger_id from(select passenger_id from 表 1) a
left join (select passenger_id from 表 2) b
on a.passenger_id = b.passenger_id where b.passenger_id is null

2、 写出你所了解的 MySQL 支持的数据类型或者是 Hive 中支持的数据类型，选其中一个即可。

MySQL：

一.数值类型

Mysql 支持所有标准 SQL 中的数值类型，其中包括严格数据类型(INTEGER,SMALLINT,DECIMAL,NUMBERIC（精确存储数值），以及近似数值数据类型(FLOAT,REAL,DOUBLE PRESISION),并在此基础上进行扩展。

扩展后增加了 TINYINT,MEDIUMINT,BIGINT 这 3 种长度不同的整形，并增加了 BIT 类型，用来存放位数据。

类型	大小	范围（有符号）	范围（无符号）	用途
TINYINT	1 字节	(-128，127)	(0，255)	小整数值
SMALLINT	2 字节	(-32 768，32 767)	(0，65 535)	大整数值
MEDIUMINT	3 字节	(-8 388 608，8 388 607)	(0，16 777 215)	大整数值
INT 或 INTEGER	4 字节	(-2 147 483 648，2 147 483 647)	(0，4 294 967 295)	大整数值
BIGINT	8 字节			极大整数值

INT 类型:

在 MySQL 中支持的 5 个主要整数类型是 TINYINT，SMALLINT，MEDIUMINT，INT 和 BIGINT。这些类型在很大程度上是相同的，只有它们存储的值的大小是不相同的。

MySQL 以一个可选的显示宽度指示器的形式对 SQL 标准进行扩展，这样当从数据库检索一个值时，可以把这个值加长到指定的长度。例如，指定一个字段的类型为 INT(6)，就可以保证所包含数字少于 6 个的值从数据库中检索出来时能够自动地用空格填充。需要注意的是，使用一个宽度指示器不会影响字段的大小和它可以存储的值的范围。

万一我们需要对一个字段存储一个超出许可范围的数字，MySQL 会根据允许范围最接近它的一端截短后再进行存储。还有一个比较特别的地方是，MySQL 会在不合规定的值插入表前自动修改为 0。

UNSIGNED 修饰符规定字段只保存正值。因为不需要保存数字的正、负符号，可以在储时节约一个“位”的空间。从而增大这个字段可以存储的值的范围。

ZEROFILL 修饰符规定 0（不是空格）可以用来真补输出的值。使用这个修饰符可以阻止 MySQL 数据库存储负值。

FLOAT、DOUBLE 和 DECIMAL 类型

MySQL 支持的三个浮点类型是 FLOAT、DOUBLE 和 DECIMAL 类型。FLOAT 数值类型用于表示单精度浮点数值，而 DOUBLE 数值类型用于表示双精度浮点数值。

与整数一样，这些类型也带有附加参数：一个显示宽度指示器和一个小数点指示器。比如语句 FLOAT(7,3) 规定显示的值不会超过 7 位数字，小数点后面带有 3 位数字。

对于小数点后面的位数超过允许范围的值，MySQL 会自动将它四舍五入为最接近它的值，再插入它。

DECIMAL 数据类型用于精度要求非常高的计算中，这种类型允许指定数值的精度和计

数方法作为选择参数。精度在这里指为这个值保存的有效数字的总个数，而计数方法表示小数点后数字的位数。比如语句 DECIMAL(7,3) 规定了存储的值不会超过 7 位数字，并且小数点后不超过 3 位。

忽略 DECIMAL 数据类型的精度和计数方法修饰符将会使 MySQL 数据库把所有标识为这个数据类型的字段精度设置为 10，计算方法设置为 0。

UNSIGNED 和 ZEROFILL 修饰符也可以被 FLOAT、DOUBLE 和 DECIMAL 数据类型使用。并且效果与 INT 数据类型相同。

二.字符串类型

MySQL 提供了 8 个基本的字符串类型,分别:CHAR、VARCHAR、BINARY、VARBINARY、BLOB、TEXT、ENUM 各 SET 等多种字符串类型。

可以存储的范围从简单的一个字符到巨大的文本块或二进制字符串数据。

字符串类型	字节大小	描述及存储需求
CHAR	0-255 字节	定长字符串
VARCHAR	0-255 字节	变长字符串
TINYBLOB	0-255 字节	不超过 255 个字符的二进制字符串
TINYTEXT	0-255 字节	短文本字符串
BLOB	0-65535 字节	二进制形式的长文本数据
TEXT	0-65535 字节	长文本数据
MEDIUMBLOB	0-16 777 215 字节	二进制形式的中等长度文本数据
MEDIUMTEXT	0-16 777 215 字节	中等长度文本数据
LOGNGBLOB	0-4 294 967 295 字节	二进制形式的极大文本数据
LONGTEXT	0-4 294 967 295 字节	极大文本数据
VARBINARY(M)		M 允许长度 0-M 个字节的定长字节符串，值的长度+1 个字节
BINARY(M)		M 允许长度 0-M 个字节的定长字节符串

1. CHAR 和 VARCHAR 类型

CHAR 类型用于定长字符串，并且必须在圆括号内用一个大小修饰符来定义。这个大小修饰符的范围从 0-255。比指定长度大的值将被截短，而比指定长度小的值将会用空格作填补。

CHAR 类型可以使用 BINARY 修饰符。当用于比较运算时，这个修饰符使 CHAR 以二进制方式参于运算，而不是以传统的区分大小写的方式。

CHAR 类型的一个变体是 VARCHAR 类型。它是一种可变长度的字符串类型。CHAR 和 VARCHGAR 不同之处在于 MYSQL 数据库处理这个指示器的方式：CHAR 把这个大小视为值的大小，不长度不足的情况下就用空格补足。而 VARCHAR 类型把它视为最大值并且只使用存储字符串实际需要的长度(增加一个额外字节来存储字符串本身的长度)来存储值。所以短于指示器长度的 VARCHAR 类型不会被空格填补，但长于指示器的值仍然会被截短。因为 VARCHAR 类型可以根据实际内容动态改变存储值的长度，所以在不能确定字段需要多少字符时使用 VARCHAR 类型可以大大地节约磁盘空间、提高存储效率。VARCHAR 类型在使用 BINARY 修饰符时与 CHAR 类型完全相同。

2. TEXT 和 BLOB 类型

对于字段长度要求超过 255 个的情况下，MySQL 提供了 TEXT 和 BLOB 两种类型。根据存储数据的大小，它们都有不同的子类型。这些大型的数据用于存储文本块或图像、声音文件等二进制数据类型。

TEXT 和 BLOB 类型在分类和比较上存在区别。BLOB 类型区分大小写，而 TEXT 不区分大小写。大小修饰符不用于各种 BLOB 和 TEXT 子类型。比指定类型支持的最大范围大的值将被自动截短。

三.日期和时间类型

在处理日期和时间类型的值时，MySQL 带有 5 个不同的数据类型可供选择。它们可以被分成简单的日期、时间类型，和混合日期、时间类型。

根据要求的精度，子类型在每个分类型中都可以使用，并且 MySQL 带有内置功能可以把多样化的输入格式变为一个标准格式。

类型	大 小 (字节)	格式	用途
DATE	4	YYYY-MM-DD	日期值
TIME	3	HH:MM:SS	时间值或持续时间
YEAR	1	YYYY	年份值
DATETIME	8	YYYY-MM-DD HH:MM:SS	混合日期和时间值
TIMESTAMP	4	YYYYMMDD HHMMSS	混合日期和时间值，时间戳

四.复合类型

MySQL 还支持两种复合数据类型 ENUM 和 SET，它们扩展了 SQL 规范。虽然这些类型在技术上是字符串类型，但是可以被视为不同的数据类型。

一个 ENUM 类型只允许从一个集合中取得一个值；而 SET 类型允许从一个集合中取得任意多个值。

ENUM 类型

ENUM 类型因为只允许在集合中取得一个值，有点类似于单选项。在处理相互排拆的数据时容易让人理解，比如人类的性别。ENUM 类型字段可以从集合中取得一个值或使用 null 值，

除此之外的输入将会使 MySQL 在这个字段中插入一个空字符串。另外如果插入值的大小

写与集合中值的大小写不匹配，MySQL 会自动使用插入值的大小写转换成与集合中大小写一致的值。
ENUM 类型在系统内部可以存储为数字，并且从 1 开始用数字做索引。一个 ENUM 类型最多可以包含 65536 个元素，其中一个元素被 MySQL 保留，用来存储错误信息，
这个错误值用索引 0 或者一个空字符串表示。
MySQL 认为 ENUM 类型集合中出现的值是合法输入，除此之外其它任何输入都将失败。这说明通过搜索包含空字符串或对应数字索引为 0 的行就可以很容易地找到错误记录的位置。
SET 类型
SET 类型与 ENUM 类型相似但不相同。SET 类型可以从预定义的集合中取得任意数量的值。并且与 ENUM 类型相同的是任何试图在 SET 类型字段中插入非预定义的值都会使 MySQL 插入一个空字符串。如果插入一个即有合法的元素又有非法的元素的记录，MySQL 将会保留合法的元素，除去非法的元素。
一个 SET 类型最多可以包含 64 项元素。在 SET 元素中值被存储为一个分离的 “位” 序列，这些 “位” 表示与它相对应的元素。 “位” 是创建有序元素集合的一种简单而有效的方式。
并且它还去除了重复的元素，所以 SET 类型中不可能包含两个相同的元素。希望从 SET 类型字段中找出非法的记录只需查找包含空字符串或二进制值为 0 的行。
通过对每种数据类型的用途，物理存储，表示范围等有一个概要的了解。这样在面对具体应用时，就可以根据相应的特来来选择合适的数据类型，使得我们能够争取在满足应用的基础上，
用较小的存储代价换来较高的数据库性能。

Hive 的内置数据类型可以分为两大类：
(1)、基础数据类型；
(2)、复杂数据类型。
其中，基础数据类型包括：TINYINT, SMALLINT, INT, BIGINT, BOOLEAN, FLOAT, DOUBLE, STRING, BINARY,TIMESTAMP,DECIMAL,CHAR,VARCHAR,DATE。下面的表格列出这些基础类型所占的字节以及从什么版本开始支持这些类型。

数据类型	所占字节	开始支持版本
TINYINT	1byte，-128 ~ 127	
SMALLINT	2byte，-32,768 ~ 32,767	
INT	4byte,-2,147,483,648 ~ 2,147,483,647	
BIGINT	8byte,-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807	
BOOLEAN		
FLOAT	4byte 单精度	
DOUBLE	8byte 双精度	

STRING		
BINARY		从 Hive0.8.0 开始支持
TIMESTAMP		从 Hive0.8.0 开始支持
DECIMAL		从 Hive0.11.0 开始支持
CHAR		从 Hive0.13.0 开始支持
VARCHAR		从 Hive0.12.0 开始支持
DATE		从 Hive0.12.0 开始支持

复杂类型包括 ARRAY,MAP,STRUCT,UNION，这些复杂类型是由基础类型组成的。

ARRAY：ARRAY 类型是由一系列相同数据类型的元素组成，这些元素可以通过下标来访问。比如有一个 ARRAY 类型的变量 fruits，它是由['apple','orange','mango']组成，那么我们可以通过 fruits[1]来访问元素 orange，因为 ARRAY 类型的下标是从 0 开始的；

MAP：MAP 包含 key->value 键值对，可以通过 key 来访问元素。比如" userlist" 是一个 map 类型，其中 username 是 key，password 是 value ;那么我们可以通过 userlist['username'] 来得到这个用户对应的 password；

STRUCT :STRUCT 可以包含不同数据类型的元素。这些元素可以通过" 点语法" 的方式来得到所需要的元素，比如 user 是一个 STRUCT 类型，那么可以通过 user.address 得到这个用户的地址。

UNION: UNIONTYPE，他是从 Hive 0.7.0 开始支持的。

创建一个包含复制类型的表格可以如下：

```
1.  CREATE TABLE employees (
2.      name STRING,
3.      salary FLOAT,
4.      subordinates ARRAY<STRING>,
5.      deductions MAP<STRING, FLOAT>,
6.      address STRUCT<street:STRING, city:STRING, state:STRING, zip:INT>
7.  ) PARTITIONED BY (country STRING, state STRING);
```

3、有一个数据库表,名字是 order_info，写出查看 order_info 表字段名及其对应数据类型的语句。

mysql 查询某个数据库中某个表的所有字段名、字段类型和注释

```
select COLUMN_NAME,DATA_TYPE,COLUMN_COMMENT from information_schema.COLUMNS where table_name = '表名' and table_schema = '数据库名称';
```

这样得到的结果就是类似于：

COLUMN_NAME	DATA_TYPE	COLUMN_COMMEN
id	varchar	
position	varchar	职位
name	varchar	姓名
date	date	入职日期
phone	varchar	手机号
address	varchar	联系地址
salary	int	薪资
remark	varchar	备注
password	varchar	

http://blog.csdn.net/caoPengFlying

4、 Join 与 left join 在什么情况下会使用这两个关键词？
JOIN 在内连接时，可以不使用，其它类型连接必须使用。
如 SELECT * FROM TABLEA INNER JOIN TABLEB ON A.ID=B.ID
可以这样写：
SELECT * FROM TABLEA,TABLEB WHERE A.ID=B.ID

JOIN 有以下几种类型：

INNER(内连接）
指定返回每对匹配的行。废弃两个表中不匹配的行。如果未指定联接类型，则这是默认设置。
FULL（全连接）
指定在结果集中包含左表或右表中不满足联接条件的行，并将对应于另一个表的输出列设为 NULL。这是对通常由 INNERJOIN 返回的所有行的补充。
LEFT（左连接）

指定在结果集中包含左表中所有不满足联接条件的行，且在由内联接返回所有的行之外，将另外一个表的输出列设为 NULL。
RIGHT（右连接）
指定在结果集中包含右表中所有不满足联接条件的行，且在由内联接返回的所有行之外，将与另外一个表对应的输出列设为 NULL。
CROSS JOIN（交叉连接）
得到连接表符合 WHERE 子句的条件的记录数的乘积，即第一个表的每一个记录都与别一个表的所有记录连接出一个新的记录。
交叉连接不带 ON 子句，其它连接必须有 ON 子句
5、 有一张 score 表，包含每个人的三科成绩，数据样例如下：

name	subject	score
张三	语文	82
张三	数学	90
张三	外语	75
李四	语文	58

…	…	…

统计出每个人的最高分数，统计出有多少人三科成绩都大于 80 分

https://www.cnblogs.com/tenghoo/archive/2007/06/11/779240.html

```
select t1.stuid, t1.name, t1.subject, t1.score from stuscore t1,
( select stuid, max(score) as maxscore from stuscore group by stuid) t2
where t1.stuid = t2.stuid and t1.score = t2.maxscore

SELECT S.name
FROM Student S
GROUP BY S.name
Having MIN(S.score)>=80
```

6、 现有表 a:id1223 表 b:id1224 select * from a join b on a.id=b.id，得出的结果是什么？(注释：INNER JOIN 与 JOIN 是相同)　NULL

7、 编程面试题部分以 Java 为例，已知数组为：

String[] capital = { “A”， “B”， “E”， “A”， “T”， “Q”， “B” }

以 python 为例，已知列表为：capital=[“A”， “B”， “E”， “A”， “T”， “Q”， “B”]

请统计并打印出以上数据集合中每个字母出现的次数，Java 或者 python 任选其一。

```
capital = [ “A”， “B”， “E”， “A”， “T”， “Q”， “B” ]
a = {}
for i in capital:
    a[i] = capital.count(i)
print(a)
```