mysql

1、为什么一定要设一个主键？

因为你不设主键的情况下，innodb也会帮你生成一个隐藏列，作为自增主键。所以啦，反正都要生成一个主键，那你还不如自己指定一个主键，在有些情况下，就能显式的用上主键索引，提高查询效率！

2、你们主键是用自增还是UUID?

肯定答自增啊。innodb 中的主键是聚簇索引。如果主键是自增的，那么每次插入新的记录，记录就会顺序添加到当前索引节点的后续位置，当一页写满，就会自动开辟一个新的页。如果不是自增主键，那么可能会在中间插入，就会引发页的分裂，产生很多表碎片！。

上面那句话看不懂没事，大白话一句就是:用自增插入性能好！

另外，附一个测试表给你们，表名带uuid的就是用uuid作为主键。大家看一下就知道性能差距了:

表 5-1：向InnoDB表插入数据的测试结果

表名	行数	时间（秒）	索引大小（MB）
userinfo	1 000 000	137	342
userinfo_uuid	1 000 000	180	544
userinfo	3 000 000	1233	1036
userinfo_uuid	3 000 000	4525	1707

如上图所示，当主键是UUID的时候，插入时间更长，而且占用空间更大！

额，大家千万不要忘了，当你回答自增主键后，想一下《自增主键用完该怎么办？》

ps：这个问题，你要是能把UUID讲出合理的理由也行。

3、主键为什么不推荐有业务含义?

有如下两个原因

(1)因为任何有业务含义的列都有改变的可能性,主键一旦带上了业务含义，那么主键就有可能发生变更。主键一旦发生变更，该数据在磁盘上的存储位置就会发生变更，有可能会引发页分裂，产生空间碎片。

(2)带有业务含义的主键，不一定是顺序自增的。那么就会导致数据的插入顺序，并不能保证后面插入数据的主键一定比前面的数据大。如果出现了，后面插入数据的主键比前面的小，就有可能引发页分裂，产生空间碎片

4、表示枚举的字段为什么不用enum类型？

在工作中表示枚举的字段，一般用tinyint类型。

那为什么不用enum类型呢？下面两个原因

(1)ENUM类型的ORDER BY操作效率低，需要额外操作

(2)如果枚举值是数值，有陷阱

举个例子，表结构如下

```
CREATE TABLE test (foobar ENUM('0', '1', '2'));
```

此时，你执行语句

```
mysql> INSERT INTO test VALUES (1);
```

查询出的结果为

就产生了一个坑爹的结果。

插入语句应该像下面这么写，插入的才是1

```
mysql> INSERT INTO test VALUES (`1`);
```

5、货币字段用什么类型?

回答:如果货币单位是分，可以用Int类型。如果坚持用元，用Decimal。

千万不要答float和double，因为float和double是以二进制存储的，所以有一定的误差。

打个比方，你建一个列如下

```
CREATE TABLE `t` (
 `price` float(10,2) DEFAULT NULL,
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

然后insert给price列一个数据为1234567.23，你会发现显示出来的数据变为1234567.25，精度失准！

6、时间字段用什么类型?

此题无固定答案，应结合自己项目背景来答！把理由讲清楚就行！

(1)varchar，如果用varchar类型来存时间，优点在于显示直观。但是坑的地方也是挺多的。比如，插入的数据没有校验，你可能某天就发现一条数据为2013111的数据，请问这是代表2013年1月11日，还是2013年11月1日？

其次，做时间比较运算，你需要用STR_TO_DATE等函数将其转化为时间类型，你会发现这么写是无法命中索引的。数据量一大，是个坑！

(2)timestamp，该类型是四个字节的整数，它能表示的时间范围为1970-01-01 08:00:01到2038-01-19 11:14:07。2038年以后的时间，是无法用timestamp类型存储的。

但是它有一个优势，timestamp类型是带有时区信息的。一旦你系统中的时区发生改变，例如你修改了时区

```
SET TIME_ZONE = "america/new_york";
```

你会发现，项目中的该字段的值自己会发生变更。这个特性用来做一些国际化大项目，跨时区的应用时，特别注意！

(3)datetime，datetime储存占用8个字节，它存储的时间范围为1000-01-01 00:00:00 ~ 9999-12-31 23:59:59。显然，存储时间范围更大。但是它坑的地方在于，他存储的是时间绝对值，不带有时区信息。如果你改变数据库的时区，该项的值不会自己发生变更！

(4)bigint，也是8个字节，自己维护一个时间戳，表示范围比timestamp大多了，就是要自己维护，不大方便

7、为什么不直接存储图片、音频、视频等大容量内容?

我们在实际应用中，都是用HDFS来存储文件。然后mysql中，只存文件的存放路径。mysql中有两个字段类型被用来设计存放大容量文件，也就是text和blob类型。但是，我们在生产中，基本不用这两个类型！

主要原因有如下两点

(1)Mysql内存临时表不支持TEXT、BLOB这样的大数据类型，如果查询中包含这样的数据，在排序等操作时，就不能使用内存临时表，必须使用磁盘临时表进行。导致查询效率缓慢

(2)binlog内容太多。因为你数据内容比较大，就会造成binlog内容比较多。大家也知道，主从同步是靠binlog进行同步，binlog太大了，就会导致主从同步效率问题！

因此，不推荐使用text和blob类型！

8、字段为什么要定义为NOT NULL?

(1)索引性能不好

Mysql难以优化引用可空列查询，它会使索引、索引统计和值更加复杂。可空列需要更多的存储空间，还需要mysql内部进行特殊处理。可空列被索引后，每条记录都需要一个额外的字节，还能导致MYisam 中固定大小的索引变成可变大小的索引。

—— 出自《高性能mysql第二版》

(2)查询会出现一些不可预料的结果

这里举一个例子，大家就懂了。假设，表结构如下

```
create table table_2 (
 `id` INT (11) NOT NULL,
 name varchar(20) NOT NULL
)
```

你执行语句

```
select count(name) from table_2;
```

你会发现结果为2，但是实际上是有四条数据的！类似的查询问题，其实有很多，不一一列举。

记住，因为null列的存在，会出现很多出人意料的结果，从而浪费开发时间去排查Bug.

```
select User,Host from user;
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'root' WITH GRANT OPTION;
```

```
mysql>FLUSH PRIVILEGES
FLUSH PRIVILEGES;
```