大型ERP等数据库系统常见的几种设计 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/141/2021_2022__E5_A4_A7_E5_9E_8BERP_E7_c29_141409.htm 1. 自增长 primary key 采用自增长 primary key主要是性能。早期的数据库系统，经常采用某种编号，比如身份证号码，公司编号等等作为数据库表的 primary key。然而，很快，大家就发现其中的不利之处。 比如早期的医院管理系统，用身份证号码作为病人表的 primary key。然而，第一，不是每个人都有身份证；第二，对于国外来的病人，不同国家的病人的证件号码并不见得没有重复。因此，用身份证号码作为病人表的 primary key是一个非常糟糕的设计。考虑到没有医生或者护士会刻意去记这些号码，使用自增长 primary key是更好的设计。 公司编号采用某种特定的编码方法，这也是早期的数据库系统常见的做法。它的缺点也显而易见：很容易出现像千年虫的软件问题，因为当初设计数据库表的时候设计的位数太短，导致系统使用几年后不能满足要求，只有修改程序才能继续使用。问题在于，任何人设计系统的时候，在预计某某编号多少位可以够用的时候，都存在预计不准的风险。而采用自增长 primary key 则不存在这种问题。同样的道理，没有人可以去记这些号码。使用自增长 primary key另外一个原因是性能问题。略有编程常识的人都知道，数字大小比较比字符串大小比较要快得多。使用自增长 primary key可以大大地提高数据查找速度。 2. 避免用复合主键 (compound primary key) 这主要还是因为性能问题。数据检索是要用到大量的 primary key 值比较，只比较一个字段比比较多个字段快很多。使用单个 primary key 从编

程的角度也很有好处， sql 语句中 where 条件可以写更少的代码，这意味着出错的机会大大减少。 3. 双主键 双主键是指数据库表有两个字段，这两个字段独立成为主键，但又同时存在。 数据库系统的双主键最早用在用户管理模块。最早的来源可能是参照操作系统的用户管理模块。 操作系统的用户管理有两个独立的主键：操作系统自己自动生成的随机 ID (Linux, windows 的 SID), login id。这两个 ID 都必须是唯一的，不同的是，删除用户 test 然后增加一个用户 test, SID 不同，login id 相同。采用双主键主要目的是为了防止删除后增加同样的 login id 造成的混乱。比如销售经理 hellen 本机共享文件给总经理 peter, 一年后总经理离开公司，进来一个普通员工 peter ，两个peter 用同样的 login id, 如果只用 login id 作操作系统的用户管理主键，则存在漏洞：普通员工 peter 可以访问原来只有总经理才能看的文件。操作系统自己自动生成的随机 ID 一般情况下面用户是看不到的。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com