3.4

scalability(可扩展性)

由于数据生成器的初始设计被要求用于生成从1GB到100TB的数据集，这个数据集扩展了五个数量级，所以代码的可扩展性是一个关键的设计要求。可扩展性在两个方面得到解决：算法效率和并行性。

算法效率在任何实用程序中都很重要。在任何一个执行过程中，一个将被要求产生多达一万亿个实体的模块将变得至关重要。MUDD采用了许多标准技术来确保其代码路径在其偶尔冗长的执行过程中的效率。(例如，大量使用向量例程和其他可重用的缓冲区，以避免不必要地调用malloc()和free()等等)。

然而，对于这个范围的应用程序，可伸缩性的真正关键是实现并行性。使用MUDD，没有必要以线性，单内核的方式生成数据。它可以使用最适合计算和IO资源的并行程度来生成数据。这个功能的实现很简单。

首先，RNG必须在所有目标平台上通用。MUDD通过提供可移植的RNG来完成这一任务，而不是依赖特定操作系统提供的例程。

接下来，无论比例因子如何，对于特定表内的每一行，对RNG的调用次数必须是确定的。

对于大多数列类型和数据域，这是一个简单的任务，因为大多数常见的值（即数字类型）可以通过对RNG的单个调用来生成。在调用次数可以变化的情况下，如在基于简化语法的英文文本产生中，必须确定每行调用次数的上限。如果对于RNG的调用是有条件的(“对于有周末时间的40%商店，从这些值中选择一个 ，否则S\_WKND\_ HOURS就是NULL的。)，不管这个条件的输出是怎么样的，必须做出对RNG的调用。虽然这会导致对RNG的额外调用，但它将恢复每行调用量度的确定性性质。给定表格的行生成例程的最后一个操作是为了确保在这一行内已经进行了所需的调用次数，如有必要，调用RNG附加次数。

有了这两个因素，对于构建特定表的一系列协作生成器的任何成员，决定要生成的行的数量是很容易的，并且，在任何数据生成器的工作之前，将RNG初始化成前一个生成器对前一行生成的值。MUDD使用64位线性同余随机数生成器。因此，有可能编写有效地“跳转”到RNG的生成值序列中的任何特定点的例程。如果这是不可能的话，那么并行的大部分好处将会丢失，因为在开始分区之前，每个并行生成任务将需要依次生成所有先前的随机数。

可以使任意一组计算和IO资源饱和（请参见下面的性能部分）。此外，由于并行执行中的每个生成过程是完全独立的，因此不需要集中协调或其他IPC开销。数据生成可以使用一个单一的机器或者多个机器的集合。它可以一次完成，或者数据集可以随着时间和计算资源的允许而分阶段建立。每个配置的结果数据集都是一样的。

4、Data population/Distribution

4.1Domain Scaling versus Scaling

扩展数据集可以从两个方面来做。在一种情况下，数据集里元组的数量被扩展，但基本的价值集（领域）保持不变。这可以类比成一个商业系统，其中商店的数量保持不变，但每年的交易量增加。在另一种情况下，元组的数量保持不变，但用于生成它们的域被扩展。例如，零售数据集中可能引入了一种新型商店，或者数据集可能涵盖更长的历史时期。显然，在数据集中这两种类型的扩展都可以被认为是正确的，通常一个测试将采用这两种方法来增长数据集。MUDD选择主要使用data set scaling而不是domain scaling。通过保持domains不变，增加被管理的数据的量，可以认为实现对数据库管理系统更有意义的全系统的测试是可行的。当然，由于在测试一个数据库管理系统的关键部分是其智能地管理和联结joins的能力。而且由于主键和外键的关系必须按定义使用data scaling，所以在这种情况下结果也算是两种方法的混合。

4.2合成数据中的可比性区域

因为MUDD是在数据库管理系统决策支持方法的性能评估中使用的，它需要生成可用于适合benchmark查询集的数据。一个数据集为什么非常适合这种性能评估的工作的精确本质，超出了本论文的范围。但是，它的要求的一个方面将会被提及。

非合成数据通常包含不连续性，但它们很少位于数据样本的一致或可重复部分。虽然可以生成数据来模拟这些行为（请参阅下面的“在合成生成器中使用实数据”），对于查询性能评估来说，如果忽略到此行为，用户会多一点。查询工作负载需要从一个查询接着下一个查询提供足够的可变性，从而确保对于查询的响应是从DBMS中派生来的—即不是简单地从查询的最后一个相同的执行中检索。与此同时，它们需要确保经过两次基准测试执行的查询集可以提供对被测DBMS的比较—即他们代表了相同数量的工作。本质上，SQL查询基准测试引入了另一个随机性级别，除了基础数据之外，还需要对其进行控制以保证正在测试的查询执行之间的可比性。

举一个例子，考虑一整年的零售可能性。研究表明，在年末假期销售的比例很高，通常最后两个月的销售额可以占到年销售额的30%左右(大部分是在12月下旬的时间)。显然，这种数据倾斜是数据集中的一个重要特征，需要由数据生成器来捕获。然而，数据的使用也需要被考虑。如果精确地捕捉到年底的突破，那么很可能不会有两天产生完全相同的销量。在我们查询的例子中，那么就不可能确保基于这个关键范围内的日期（或者全年其他地方）的查询在被测系统上产生可比较的负载量或活动量。留给数据集/基准测试设计人员的选项是根据基准中基于该范围的日期来移除查询，或者找到一些方法来确保他们和彼此具有可比性。

鉴于在数据集中倾斜的重要性，并且在这个特定的数据集中这种特殊的偏差是主要的，省略是个不好的选择。替代的是，这个数据集需要调整，引入可比性区域—实质上是分布中的flat spot—可以被用来提供生成数据的最终用户要求的变化性和可比性。

4.3在合成生成器中真实数据的用处

合成数据生成器面临着固有的挑战。如果数据太综合(例如，完全一致的分布)，由于没有捕获真实数据集的“有趣“的性质，它有着被拒绝的风险。相反的情况下，如果它使用从事务或装置收集的数据（“即从现实世界中”），那么对研究人员和基准来说，风险变地很小或者将没有价值，因为既不能产生可比较的工作负载结果，也不能解决有趣的假设性问题。MUDD试图找到这两个端点的适当组合。对于它的大部分数据，它使用传统的合成分布，产生均匀分布的整数，或使用高斯分布的word selections。 然而，对于一些关键的分布来说，数据生成器依赖来自现实世界的数据来生成更真实的数据集。

The first class of columns that relies upon actual rather than synthetic data contains columns of names.无论是人名还是列名，MUDD依赖1990年人口普查的数据来确定允许值的范围。结果是一个名字组合的人口，它是静态的但是很庞大，并在数据集内增加“可识别性”。

基于现实数据的第二个类处理地址。数据生成器有两个问题。一方面，地址域必须符合在地理数据中已经被定义好的层次结构，即正确地显示国家、州、城市等层次，如果不能完全正确的话，甚至到了邮政编码和电话号码需要合理的地步。同时，该领域必须随着数据集的扩展而增长。尽管数据集中约翰·史密斯的数量可以随着数据集的规模而线性增加，对于居住在俄亥俄州桑达斯基（Sandusky，Ohio）100 Elm Street的客户数量来说，对它做同样的操作是不可行的。MUDD的解决方案是结合已知的分布（美国地质调查局的城镇名单和普通城镇名称列表，以及县，州，邮政编码和电话区号之间的适当关联）。最终的结果是一个地址生成器，虽然不是真的，但是非常现实。同时，数据显示了可扩展的地理位置; 小数据集只限于一两个国家，而大数据集包含越来越多的国家，而网络销售渠道则排在第一位。

4.4数据总量中的MDD issues问题

4.4.1层次结构

TPC-DS schema结构都显示简单的单一继承关系。那就是，对于层次结构中给定的级别，确切地说总是只有一个父级。有了这个保证，和给定的层次结构中的每一个级别的基数，这个生成器变得非常直截了当。

对于分层表中任何给定的行，行号唯一确定该行在每个层级中的位置。需要为层次结构中的某部分上的表生成适当的连接字段，这个是一个简单的模板函数序列，基于已知的特定级别的基数，总高度，目标层次结构。Skew can be introduced into the joins to a hierarchical  
table simply be picking non-uniform pseudo row numbers on which to perform this decomposition.

4.4.2skew 到sparseness

也许多维数据集中最显着的特征是空间概念, 事实表中特定行引用的每个维度至少在架构上是独立的。也就是说，特定行的各个维度的值之间没有必要存在相互依赖关系。结合任何给定维度的广泛领域的可能性，维度“space”可以是巨大的。这样做的直接副作用是这个可能空间的大部分将是空的。例如，一个零售数据库可能包含产品的尺寸，颜色和购买日期的这些维度。所有尺寸和颜色组合产品都不可能在任何一天都有被选购到—就像这些属性的特定组合几乎不会在任何一个特定的日子里被购买一样。只有在真正的购买数量增加的情况下，才有可能覆盖整个维度空间。对于数据生成器，这个有两个有意义的影响。

首先，数据集需要通过确保相对于正在生成的数据集的足够大的domain来对朝稀疏数据转换的趋势做出解释。考虑到在一个普遍的多维schema中维度的数目，这几乎不能算是一个问题。

其次，数据集必须同时考虑在大多数数据集中存在的clustering和skew问题（例如，周六会比周一有更多的购物量），并且确保数据它自己是可以预测的和可重复的行为。这里的关键是提供一个强大且灵活的方法，通过它来定义可以被引入到数据中的数据倾斜。当它们被应用在给定的事实表共享的多个维度时，他们的分布将会通过傅里叶转换来组合，这个将会放大任何给定的分布倾斜，并在数据集中产生期望的短暂性和“突发性”。

4.4.3

慢慢地改变维度

除了基于事务的事实表和基于属性的维度表，一个典型的多维度系统包括一些维度，这些维度的数据在整个系统的生命周期中演变。它们被称为缓慢变化的维度（SCD），它们捕捉数据集的历史演变。作为一个例子，考虑一个跟踪数月或数年的零售销售的系统。在那一个时期，基础产品线，价格结构，销售区域地理—几乎事务上下文的每一部分都有可能改变。为了可能得到有意义的纵向分析，数据集的用户能够recreate that context是非常重要的，将旧的定价模式的销售与使用新的（或提议的）定价模式的销售进行比较。虽然使用多维的数据仓库，可以有很多方法来解决这种类型的数据，一种常见的技术是包含版本信息，在多维表中，通常以开始日期和结束日期的形式。然后查询可以限定应该使用dimension entry的哪个修订版来探查事实表。

在生成的数据集中重新创建此行为。MUDD有将一个表识别成一个SCD的能力，将那些在生成数据的日期范围内修改的表中的那些列与那些静态的列相隔离，并且将所需的日期和schema的日期唯独进行同步。

4.5捕获领域分布