GWAC 海量星表数据处理的数据库系统选型研究*

万萌[1,2,3]，吴潮[1]，Ying Zhang[3]，徐洋[1]，魏建彦[1]

（1. 中国科学院国家天文台，北京，100012

2. 中国科学院大学，北京，100049

3. 荷兰国家数学与计算机科学研究中心，阿姆斯特丹，1098 XG）

摘要 为应对我国的宽视场地基广角相机阵（GWAC）在大数据的管理和实时处理上所带来的挑战，本文提出一种基于列存储数据库 MonetDB 的时序数据处理与管理系统设计方案。本方案充分利用 MonetDB 兼具数据处理和管理于一身的数据库平台特点，通过将交叉证认等核心数据处理算法内嵌于数据库中，从而实现将“计算带到数据中”的设计理念。同时，对本方案开展了多项关键技术的研究与测试：TPC-H 基准性能测试、大数据加载能力测试及优化研究；基于 MonetDB 的 Zone 算法实现与测试；可定制函数开发功能的测试。初步的预研结果表明列存储是切实可行的。同时，本文将对本设计方案作详细的介绍。本文提出的基于列存储 MonetDB 数据库设计的海量星表数据处理应用方案，是高效的数据处理与管理为一体的天文数据库解决方案。

关键词：天文数据库；架构设计；MonetDB；实时分析；交叉证认

中图分类号：TP311 **文献标识码**：A **文章编号**：

1. 引言

现代天文观测技术和数据处理技术的发展，使得时域天文的观测朝着更大视场和更高的时间采样率两方面的发展成为可能，也给现代时域天文的观测注入了新的活力，如超新星尤其是爆发早期超新星的发现、伽玛暴光学余辉的快速响应观测、微引力透镜事件的发现与后随观测等都得益于现代时域天文观测和数据处理技术的发展。

我国兴建中的地面广角相机阵（Ground Wide Angle Camera：GWAC）项目，由 36 台口径为 18 厘米的广角望远镜组成，每台望远镜配备有 4k×4k 的 CCD 探测器。整个相机阵的天区覆盖 5000 平方度、时间采样率为 15 秒。每个观测夜对固定天区目标的持继观测长达 4-5 个小时。从观测视场的大小和观测时间的采样频度上，GWAC 在时域天文观测中都具有特殊的优势。巨大的数据量和高时间采样率，对数据的管理和处理提出极大的挑战。

GWAC 的星表数据指标是：星表数据每幅图像大约有1.5×10^5条的记录，整个相机阵在 15 秒内共产生5.4×10^6条记录，每晚则约为 2400×36=86400 幅图，大约 2TB。对数据库管理系统的要求：1）快速的大数据入库能力，所有相机阵 15 秒内产生的观测星表入库时间控制在 15 秒以内，每个观测夜的 2TB 星表最晚完成入库时间保证在下一个观测夜开始观测前；2）在数据高速采集下能够完成实时分析，面对持续不断高密度海量星表的快速关联计算能力，即每个 CCD 每 15 内秒产生的星表数据与参考星表相关联形成光变曲线。

对于地基广角相机阵，最直接的数据库管理和处理系统的设计方案是：数据库（仅为数据存储）+外围的程序（完成快速的运算）。徐洋等人通过对关键技术交叉证认的研究开发了基于空间等经纬网格的天区分区算法[1]，使得交叉证认计算实现了极大的提速；利用 GPU 的平行计算优势：赵妍等人实现了 GPU 加速图像相减处理方法[2]，赵保学等人开发了利用 GPU 加速了天文中常用的点源提取程序 SEXtractor[3]，王森红等人开发了基于 GPU 的加速星表的交叉证认算法[4]。该方案的好处是思路直接，许多技术是已有技术。缺点是数据库不断与处围程序交换，带来不必须的 IO 时间损耗；多方程序组合缺少整体的优化。

著名数据库专家 Jim Gray 成功开发了 SDSS 巡天数据库理系统 Skyserver，他提出了 Zone

* 基金项目：本论文得到国家留学基金委员会中荷互换奖学金项目，国家重点基础研究发展计划（973 计划）（2014CB845800），国家自然科学基金（U1331202, 11533003；U1431108）资助。

作者简介：万萌，女，工程师，硕士，主要从事天文数据库系统。Email：wanmeng@nao.cas.cn。

导师简介：吴潮，男，博士，副研究员，主要从事海量实时分析数据库架构。Email: cwu@nao.cas.cn

算法[5, 6]，即利用数据库的 SQL 直接实现多维空间索引取代了经典的分层三角网格算法：Hierarchical Triangular Mesh （HTM）索引算法 ，从而减少了数据的交互，速度也得到提升。这就是所谓的大规模科学计算和数据库架构设计的原则：将计算带到数据中来，而不是把数据放到计算中去的设计理念[7]。受这一思想的启发，我们提出了将地基广角相机阵的数据处理和数据管理合成到一个数据库平台的设计思路。

合适的数据库平台是实现我们这一设计思想的关键，因为传统的数据库平台不具备大数据快速处理的能力。LSST 项目针对天文大数据的需求提出开发全新的数据库 SciDB，因为处于开发活跃期，SciDB 的稳定性和实用性还有待检验。MonetDB 是一款开源的内存列存储数据库平台[8]，具有内部存储模型按列分块、占用存储空间更小、运行时查询优化等优势。MonetDB 底层物理存储模型与传统数据库有非常显著的不同，关系表经过垂直切分，每一列存储在一个单独的（ID,value）键值对表中，称作 BAT。BAT 左边一列，叫做头部，是对象 ID（OID）；右侧一列叫尾部，包含属性的值。头部和尾部分别用一个数组实现。使用列存储数据库的方式使得查询语句的执行只需要访问需要用到的列。当同一类型的属性存在于连续内存，可以获得较高的压缩比和缓存命中率。同时，MonetDB 的内核是建立在类“array”结构上的可编程的关系代数机，这种 CPU 友好的结构，能最大限度的利用硬件的性能实时响应用户的需求。而且，MonetDB 执行引擎中的缓存敏感的（cache-conscious）数据结构算法可以在运行时优化多级内存系统。

可以看出，表的数据量越大，尤其是当查询语句比重越大，就越适合使用列存储数据库。因为如果数据量很大，而查询访问的列比很小，即所有查询语句访问的列数和总列数的比例越小，使用列查询越能降低 SQL 语句查询时间，越适合使用列存储。

综上所述，MonetDB 适合要求快速取回结果的海量数据分析型 OLAP 应用和存储，这与 GWAC 需要提供的查询服务属于同一类型，非常适合应用于天文数据库。

大型射电望远镜巡天项目 LOFAR 利用 MonetDB 完成了数据处理系统 TKP pipeline，其星表数据一年约为 40TB[9]；SDSS 的 SQL Server 管理数据库曾被成功移植到 MonetDB 上[10]，并通过了对 SLOAN 数字巡天项目中的 SDSS BESTDR7 近 4TB 数据的管理测试。因此我们选取列存储数据库 MonetDB 作为我们系统的开发平台。

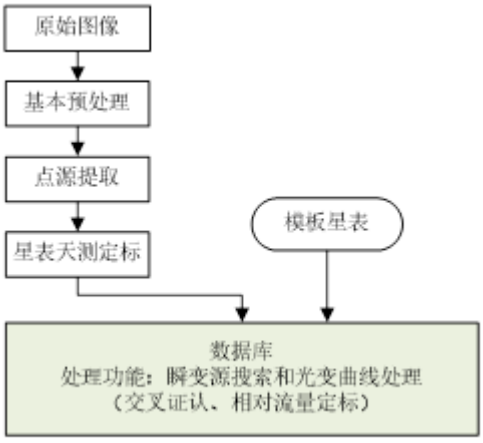


图 1 地基广角相机阵数据处理流程图

Fig. 1 The flow chart of GWAC data processing

为研究和验证以上设计思路的可行性，开展了：TPC-H 基准性能测试以验证其载入和分析能力、大数据加载速度的测试与研究；通过 MonetDB 的自定义函数接口功能测试实际自定义函数的开发功能；核心算法交叉证认 Zone 算法在 MonetDB 上的实现与测试。最后，详述了基于 MonetDB 的海量星表数据管理和处理的初步方案。

2. TPC-H 基准测试与比较

为测试 MonetDB 的实际表现，进行了 TPC-H 基准测试并比较了与传统数据库平台

PostgreSQL 及 MySQL 的差异。TPC-H 是 TPC 组织制定用来模拟决策支持类应用的一个测试集。天文数据库日常提供的服务也属于数据分析类应用，适合选择 TPC-H 作为测试集。

TPC-H 测试包括 22 条 SELECT 查询语句。测试流程为先用 DBGEN 工具产生测试数据，加载程序装载数据到三种数据库后，数据库处于初始状态，未进行其他任何操作。此时将 22 条查询顺序执行一遍，分别记录三种数据库 22 条查询的总响应时间。测试环境同上，MySQL 版本为 5.6.16。测试环境具体情况如下表 1 所示。

表 1 测试环境具体情况表

Table 1 Description of the test parameters

Software and hardware of testing platform	Database	Version
Dawn A650 Sercer， CPU: Dual-Core AMD Opteron(tm) Processor 2214 HE, 4GB RAM，OS：Fedora release 18	MonetDB	Feb2013-SP2
	PostgreSQL	9.2.0
	MySQL	5.6.16

图 2 为 MonetDB 和 PostgreSQL、MySQL 的 TPC-H SF=1 (1GB) 数据量基准性能测试结果对比，具体代码见[†]。可以看出 22 条查询总时间 MonetDB 消耗只有 PostgreSQL 的约 1/15，MySQL 的 1/137，单条查询提速至少有 3 倍。造成查询响应时间存在较大差异的主要原因是：TPC-H 业务类型是在线数据分析 OLAP 类型，数据量大，查询语句比重大，复杂的查询多，符合适合列存储数据库的典型指标。表的数据量越大，就越适合使用列存储数据库，尤其是当查询语句比重也很大时，则使用列查询能显著降低 SQL 语句查询时间。如果查询访问的列比，即所有查询语句访问的列数和总列数的比例越小，越适合列存储。这解释了对于大表 Lineitem 上的连接、且查询访问列比低的查询的 Q18，Q19，Q21，MonetDB 响应速度领先数十倍的原因。可见 MonetDB 列存储数据库在分析式查询时具有较强优势。

结论：MonetDB 在 TPC-H 测试中的表现相对于 MySQL 和 PostgreSQL 具有明显的优势，MySQL 表现最差，而 PostgresSQL 居中。

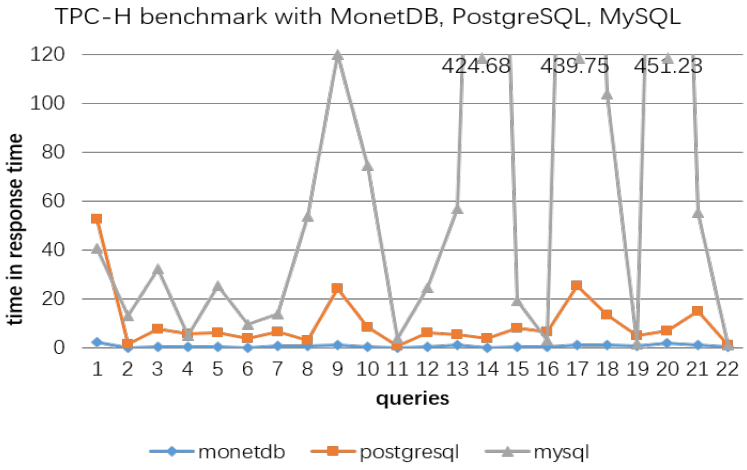


图 2 MonetDB 和 PostgreSQL，MySQL TPC-H 1G 基准测试

Fig. 2 MonetDB PostgreSQL MySQL TPC-H 1G Benchmark result

3. 关键算法与功能分析测试

根据地基广角相机阵数据处理需求：从处理速度上，要求在 15 秒的采样时间间隔内完成数据的所有处理；从功能上要求匹配出瞬变源并能生成光变曲线。我们进行了如下关键算法和功能的测试：大数据的加载能力；交叉证认算法的实现；扩展处理功能的开发实现。

3.1 大数据加载测试

大数据加载测试主要分为两种情况测试。测试 1：数据库随着数据所加载数据量的增加对加载速度的影响；测试 2：影响加载速度因素的测试分析。

[†] 三种数据库 TPC-H 脚本和结果见 https://github.com/hlfwm/TPC-H

测试 1：

测试方法为：通过仿真 1000 个20 *列*×20 *万行* ‡类似于实际观测的星表数据，连续加载到被测试的数据库中，分析加载速度的变化。测试平台的软硬件环境如 Error! Reference source not found.所示。测试的具体实现过程：对仿真的星表，使用大文件导入库命令 COPY BINARY INTO：

COPY BINARY INTO tablename FROM('path_to_file_0', 'path_to_file_1..,'path_to_file_19');

分别对 MonetDB 和 PostgreSQL 进行加载能力的对比。图 3 是对 MonetDB 进行数据加载测试的结果，图 4 是 PostgresSQL 的测试结果。对比两图得出如下结论：（1）MonetDB 随着数据库单表数据的增加，对注入的速度无明显影响，平均每个星表文件数据的载入时间为 0.94 秒。而 PostgresSQL 则在载入 500 个星表文件后，速底明显降低，平均载入速度为 6.0 秒。MonetDB 没有出现速率陡降现象。这与 MonetDB 是内存数据库相关，表现在磁盘存取、内外存的数据传递、缓冲区管理、排队等待及锁的延迟方面均比磁盘数据库快很多。（2）MonetDB 每隔几十个文件会有一个突跳点，加载时间突然提高至 10 秒左右。主要原因是由于 MonetDB 需要时间将数据写到磁盘，而导致的 IO 开销。这是在今后实际使用中需要解决的一个问题，希望能通过提高数据写入硬盘的频度来解决。

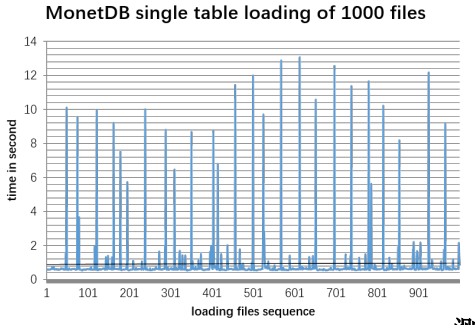


图 4 MonetDB 单表数据加载测试

Fig. 3 MonetDB 单表数据加载测试

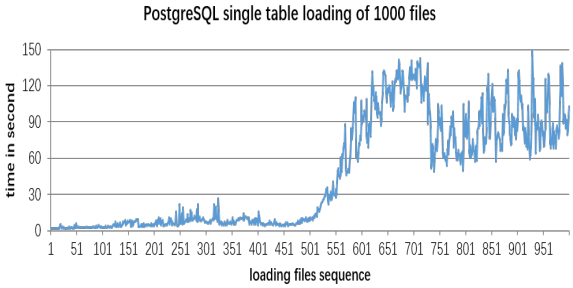


图 3 PostgreSQL 单表数据加载测试

Fig. 4 PostgreSQL single table load test

测试 2：

测试目标：对 MonetDB 进行更大量星表数据的压力加载测试，以测试在大数据下的数据加载性能的表现。

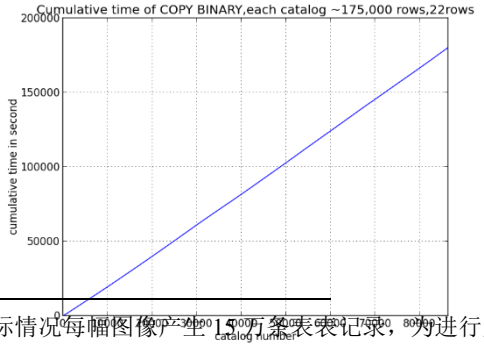
测试方法：仿真星表 86400 个星表，每个星表约为 175,000 行，包含 22 列属性数据。每个列文件大小为 1.4M，数据量总共 2.47 TB。按顺序不断往数据库注入星表数据。

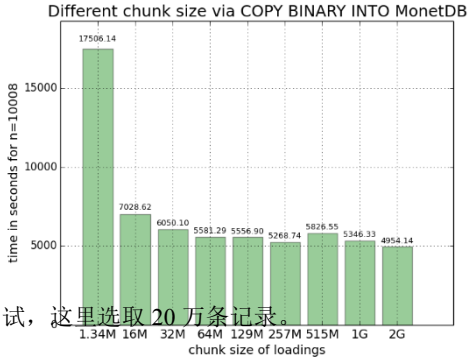
测试平台：CPU: 2 sockets Intel Xeon CPU E5-2690 v2 @ 3.00GHz. 40 processor. 内存: 8*16GB。操作系统: Scientific Linux release 6.5，Linux kernel:2.6.32-358.el6.x86_64。

测试结果：如**图 5** 所示，载入速度基本是缓慢的线性变化。平均每个星表的载入时间为 1.75 秒，总载入时间为 42 小时。86400 个星表相当于 36 个 CCD 向一个数据表里注入，所以实际使用应比压力测试的结果还要好。

在测试过程中，还发现一个现象：MonetDB 的数据载入速度与星表文件大小有着一个反常的关系，16MB 的星表文件的载入整体速度要比 1.34MB 的星表数据快 2.5 倍。具体的关系见 **图 6** 所示。主要原因是较大的文件读取单元可以更好的利用磁盘 IO，从提高了文件的载入速度。对于 GWAC，每个星表数据列文件约为 1-2MB，因此，存在着时一步优化的空间，在实际使用中需要通过调整 MonetDB 内置参数对比进行优化。

图 5





‡ 实际情况每幅图像产生 15 万星表表记录，为进行压力测试，这里选取 20 万条记录。

MonetDB 86400 个二进制星表，单表数据加载累加时间曲线　　图 6 MonetDB 不同数据块，10008 个二进制星表，单表数据加载速度测试

Fig. 5 MonetDB 86400 binary catalogs, single table cumulative loading curve.　Fig. 6 MonetDB variant chunk size, 10008 binary catalogs, single table loading benchmark

3.2 交叉证认算法的实现

交叉证认是 GWAC 巡天中搜索瞬变源和光变曲线生成的关键算法。交叉证认问题必须要依靠有效的分区策略，通过将天区按赤纬进行 Zone 分区的策略[5, 6]，将天区划分成一个个水平条带 zone，每个源都有一个属于自己的 zone id 属性，交叉证认前首先比较 zone id 可以大大降低比较次数。Zone 可集成到数据库内部，减少了数据 IO 时间损耗。

下表是一个 mini-GWAC[§]中要用到的实际星表案例，MonetDB 在 16GB 内存的服务器上所作的 20 角秒半径误差内交叉证认的测试结果，zone 高度 20 角秒。测试环境为：Intel Core i7-2600K CPU Quad-Core @ 3.40GHz, 8M Cache, 4GB 内存，Scientific Linux 6.4，linux kernel 2.6.32，MonetDB Jan2014 版本和 PostgreSQL 9.3.0 版本。

使用的 sql 语句为：

```
select x0.id as id1, t0.id as id2 from extractedcatalog x0 ,template t0
where x0.zone between floor(t0.decl-0.0056) and floor(t0.decl+0.0056)    --zone filter
and x0.decl between t0.decl-0.0056 and t0.decl+0.0056                    --dec filter
 and 3600*DEGREES(2*ASIN(SQRT((x0.x-t0.x)*(x0.x-t0.x)+(x0.y-t0.y)*(x0.y-t0.y)
+(x0.z-t0.z)*(x0.z-t0.z) )/2)) < 20;                                     --accurate computation
```

表 2 MonetDB、PostgreSQL 有无 zone 1.5 万*1.2 万条交叉证认

Table 2 15k*12k rows of cross-match MonetDB Postgres with/without zone

Catalogue A 15218 rows Catalogue B 12540 rows	Matched rows	time
MonetDB Feb2013 版 无 zone	12027 条	302s
MonetDB Feb2013 版+zone	12027 条	4.2s
MonetDB Jan2014 版 无 zone	12027 条	238s
MonetDB Jan2014 版+zone	12027 条	1.8s
PostgreSQL 9.3.0 版+zone	12027 条	30.8s

表 2 测试结果表明，同样使用 zone 算法，MonetDB Jan2014 比 PostgreSQL 加速比达 17 倍。同时，测试两个具有 17 万条记录的星表进行交叉证认，所需时间大约为 4.3 秒。

3.3 用户自定函数据功能测试

MonetDB 提供了用户定义函数 User Defined Functions（UDF）对外接口以利于扩展内核功能，用户可以灵活地根据实际需要添加自定义函数。MonetDB 的多层架构结构如图 7 所示。UDF 层首先被 MAL 解释器识别（MonetDB Assembly Language，SQL 语句解析成 MAL，提供给内核执行），然后由 SQL 编译器识别。为了大批量数据导入，开发了 UDF Columncopy，直接将 ASCII 文件导入底层列存储单元 BAT(Binary Association Table)。

[§] Mini-GWAC 是地基广角相机阵的前导项目，由 12 台 8cm 的大视场相机组成。现已建成安放于国家天文台兴隆观测站。

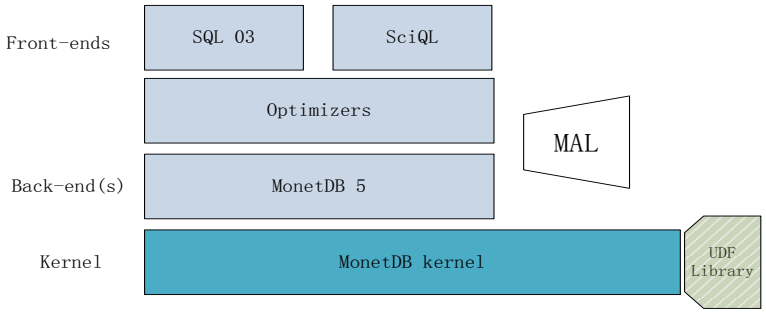


图 7 MonetDB 架构及与 UDF 关系

Fig. 7 The architecture and relationship of MonetDB with UDF

自定义函数据 Columncopy 的实现过程如下（开发语言为 C 语言）：

1. 81_svom.mal 文件用于 monetdb 服务器启动时自动加载这个 UDF 函数。
2. 81_svom.sql 文件向 SQL 函数列表加入 columncopy 函数签名，也用于自动加载。
3. Makefile.ag 向编译器引入新模块依赖的库文件信息。
4. svom.c, svom.h 为 UDF 实现文件。

函数	*解释*
_append_bat(sql, t, cname, b)	将结果 BAT *b* 追加到表 *t* 某列后。
BATnew(TYPE_void, TYPE_int, nr_rows);	创建一个 BAT。
columncopy(tablename, columnname, columnrows, filename)	SQL 函数，将文件 *filename* 里的 *columnrows* 行导入到 *tablename* 表的 *columnname* 列中。

5. svom.mal 定义 MAL 函数地址和对应的 SQL 签名的映射。
6. columncopy UDF 函数，将 BAT 结构追加到数据库中该表存储结构的后面完成导入。
7. UDF 与源代码一同编译，注册成为数据库服务器的内置函数，调用方式：

   sql>call columncopy(tablename，columnname，columnrows，filename);

   此次 UDF 为今后复用设计其他 UDF，扩展 MonetDB 服务器的能力打下了良好基础。

4. 初步设计方案

考虑每个相机的并列关系，每个相机对应一个数据库单元，36 个数据库单元最终由一台控制服务器连接，控制服务器负责数据库单元的添加和连接。这种设计的好处是数据库单元只需负责自身 CCD 的存储和计算，在物理上完成了数据表的分割。分布式数据库架构见图 8。

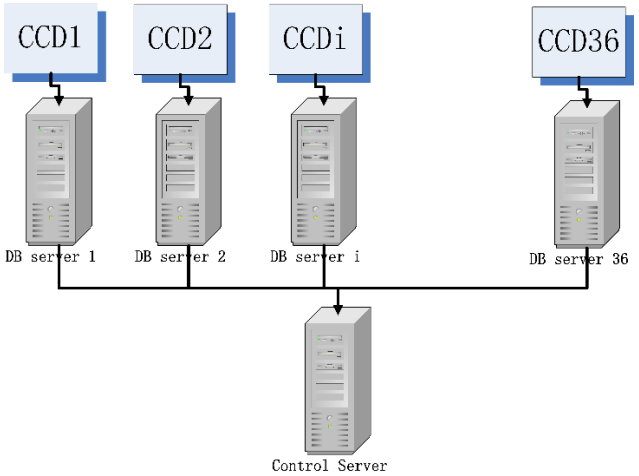


图 8 分布式数据库阵列架构

Fig. 8 Distributed architecture of Database array

分布式的数据库各数据表的名称和功能如下。

表单名称	功　　能
image	每幅原始图像都在这个表里有一个唯一标识，每行代表一幅图像。

targets	每一幅原始 FITS 图像，经过 SEXtractor 程序提取的点源记录到这个表。一个源可能会在观测中多次出现，其每次出现时的观测指标全部都会源源不断地插入 targets 表中。
uniquecatalog	每一行代表一个唯一化的源，uniquecatalog 里的每一行可能与 targets 里的多行相关联，它的位置、星等等信息是 targets 表里多个源的均值。这个表通过 associatedsource 表把所有这个源在 targets 表的出现都绑定在一起，所以 uniquecatalog 可以看做是寻找一个特定源的光变曲线的入口。
associatedsource	targets 表和 uniquecatalog 表匹配上的关联都存在这个表内。
tempuniquecatalog	临时表，targets 表与 uniquecatalog 表交叉的临时结果。
transient	保存检测到的暂现源。

通过观测目标星表与模板星表的交叉证认，如果能与模板星表匹配的则进入时序测光通道进行光变曲线的处理并进行管理。如果从模板星表无法找到匹配星，则认为是瞬变源（候选体）。其中流量相对定标的过程为：将交叉证认匹配上的两两星对之间(观测星与模板星匹配组)计算出流量比率，并取这一组的流量比率中值记为：Rm，然后修正所有观测星的星等计算公式为：$normag = -2.5\log(flux \cdot Rm)$。

其中$normag$为相对流量定标后的观测星表的星等，$flux$为观测星的流量。相对流量定标把测光星等都定标到模板星表一致的水平，这样光变曲线是一个有意义的相对光变量。所有瞬变源都存放于 transient，所有光变曲线存放于 associatedsource。

我们将交叉证认的匹配情况分成 4 种，即：多对多，一对多，一对一，无匹配。

多对多：剪除“多对多”关联以防止数据库爆炸，简化成一对多关系，继续处理。

一对多：分配新的 unique id 给 uniquecatalog 表新插入的源，替代旧的 uniqueid。新的一对多关联插入 associatedsource，设定为 type=2；插入新 unique id 与旧 targets id 关联，这是为给被取代的 unique id 新位置，标记 tpye=6。清空被一对多取代的原有分枝。

一对一：插入 tempuniquecatalog 中的一对一关联到 associatedsource 表中，标记为 type=3。

无匹配：插入新的源到 uniquecatalog，插入这些新的关联到 associatedsource，设置为 type=4。判断暂现源候选插入 transient 表。至此所有提取物处理完毕。

本系统通过以上设计完成地基广角相机阵的数据处理和管理功能。具体的代码开展与系统优化及最后的测试，将是我们下一步工作的主要内容。

5. 讨论与总结

我国兴建中的地基广角相机阵面临着大数据处理和管理的挑战：每 15 秒钟每台相机产生1.5×10^5记录，总相机数为 36 台，并且要求实时处理和入库管理。为应对挑战，提出一种充分利用数据库兼具大数据管理和处理于一身特点，基于 MonetDB 的大数据处理和管理的系统设计方案。本方案的计设特点是希望通过 MonetDB 自身具有大数据处理的特点，将核心数据处理功能内嵌于数据库内，减少因数据管理和处理分属于两个系统所带来的 IO 时间损耗。为验证 MonetDB 数据库宽视场海量时序数据的处理与管理的可行性，开展多方的测试和预先研究。

预研阶段主要通过了以下测试：（1）TPC-H 基准测试。本测试主要针对数据库决策能力的测试，测试结果表明：MonetDB 具有明显的表现优势，PostgreSQL 次之，MySQL 最差。（2）开展关键算法与功能测试。大数据加载能力测试表明 MonetDB 相对于 PostgreSQL 具有明显的优势。86400 个星表相当于 96 个观测小时的时表数据，平均每个入库时间为 1.75 秒。载入时间随着载入数据量的增长，缓慢线性增长。这个时间基本能满足项目要求，但仍存在优化的空间。研究发现星表文件的大小与载入速度有一个反常的关系，可以能过 MonetDB 的内置函数进行优化。同时，将数据存储表按不同观测夜进行分割，从而提高载入速度。交叉证认 Zone 索引算法的实现与测试表明，MonetDB 相对 PostgresSQL 具有更快的处理速度，每 17 万条记录证认需要的时间为 4.3 秒。通过调研和与 MonetDB 开发人员讨论表明，通过调整 MonetDB 的内置参数，能进一步优化交叉证认的时间。并预期证认每 17 万条的记录可以在小于 1 秒内完成。基于 MonetDB 用户自定义函数的接口，开发 Columncopy 批文件导入函数，表明 MonetDB 可以为以后的实际系统开发提供方便灵活的接口。

基于调研与测试结果，提出本方案的初步设计构架。本方案主要完成两大数据处理功能：瞬变源的实时搜索和海量光变曲线的生成与管理。确立分布式数据库的架构方式，整个系统

由 36 个并列的数据库单元组成，每个数据库单元对应一个独立望远镜且功能相同。一个主控数据库单元管理各个数据库单元。对单独的数据库单元设计了表单的结构及相互关系。同时描述了相对流量定标的实现过程。该设计方案从理论上分析，是切实可行的，但实际的开发与测试必须要经过大量的优化工作及调试工作，这将是我们下一工作的重点内容。

总之，通过研究工作，找到一种可行的应对 GWAC 大数据挑战的数据处理与管理为一体的系统设计方法，这将为 MonetDB 在中国天文界的大数据处理和管理的应用打下基础。

参考文献：

[1] 徐洋, 吴潮, 万萌, et al., 用于光学瞬变源搜寻的交叉证认快速算法 [J]. 天文研究与技术. 2013, 3: p. 011.

Xu Yang, Wu Chao, Wan Meng, et al., A Fast Cross-Identification Algorithm for Searching[J]. Astronomical Research and Technology, 2013, 10（3）：273-282.(in Chinese)

[2] Zhao Y, Luo Q, Wang S, et al. Accelerating Astronomical Image Subtraction on Heterogeneous Processors [C]. in *eScience (eScience), 2013 IEEE 9th International Conference on*. 2013. IEEE p. 70-77.

[3] Zhao B, Luo Q, and Wu C. Parallelizing Astronomical Source Extraction on the GPU [C]. in *eScience (eScience), 2013 IEEE 9th International Conference on*. 2013. IEEE p. 88-97.

[4] Wang S, Zhao Y, Luo Q, et al. Accelerating in-memory cross match of astronomical catalogs [C]. in *eScience (eScience), 2013 IEEE 9th International Conference on*. 2013. IEEE p. 326-333.

[5] Gray J, Nieto-Santisteban M A, and Szalay A S, The zones algorithm for finding points-near-a-point or cross-matching spatial datasets [J]. arXiv preprint cs/0701171. 2007.

[6] Gray J, Szalay A S, Thakar A R, et al., There goes the neighborhood: Relational algebra for spatial data search [J]. arXiv preprint cs/0408031. 2004.

[7] Szalay A S and Blakeley J A, Gray's laws: database-centric computing in science[M]. v1.1 ed. 2009, United States of America: Microsoft Research. p. 5-11.

[8] MonetDB[EB/OL]. 2015-10-01; http://www.monetdb.org.

[9] Scheers L H A.Transient and variable radio sources in the LOFAR sky: an architecture for a detection framework [D]. Amsterdam:Univ. of Amsterdam,2011: p. 25-25.

[10] Ivanova M, Nes N, Goncalves R, et al. MonetDB/SQL Meets SkyServer: the Challenges of a Scientific Database [C]. in *Scientific and Statistical Database Management, 2007. SSBDM '07. 19th International Conference on*. 2007. p. 13-13.

A Pre-research on GWAC Massive Catalog Data Storage and Processing System

Wan Meng[1,2,3], Wu Chao[1], Ying Zhang3, Xu Yang[1], Wei Jianyan[1]

(1. National Astronomical Observatories, Chinese Academy of Sciences, Beijing 100012, China, Email:cwu@nao.cas.cn

2. University of Chinese Academy of Sciences, Beijing, 100049, China

3. Centrum Wiskunde & Informatica, Amsterdam, 1098 XG)

GWAC (Ground Wide Angle Camera) poses huge challenges in large-scale catalogue storage and real-time processing of quick search of transients among wide field-of-view time-series data. Firstly, this paper proposes the concept of using databases' capabilities of fast data processing and parallelism, which will improve system performance and availability through the integration of data storage and computing platform. To understand the feasibility of Column-store MonetDB in vast catalogue management, we carry out a variety of pilot experiments of key technologies. We conduct TPC-H benchmark, data loading benchmark and optimization, and key algorithm testing of astronomical source association, all compared with the traditional row store database. Then, we use MonetDB to realize cross-match Zone algorithm. UDF function is developed for customizable data loading. Tests results show t MonetDB database has a remarkable performance in large amounts of data management and is efficient in real-time data process, thus has the ability to deal with 2.5T catalog data. In the end we propose a wide field of view massive time serial observation

data processing solution using the in-memory column store database MonetDB. The experimental results show that the feasibility of the scheme. The design plan of MonetDB-based massive catalogue data processing solution, is an efficient astronomical database solution that combines data processing and data management.
Key words: Astronomical Database; Architecture Design; MonetDB; Real-time Analysis; Source association.