地震前兆 Oracle LOB 数据压缩与交换及其访问效率研究*

王建军[1], 赵银刚[2♣], 刘高川[3]

(1. 甘肃省地震局, 甘肃 兰州 730000; 2. 安丘地震台, 山东 潍坊 262100; 3. 中国地震台网中心, 北京 100045)

摘要: 针对目前地震前兆 Oracle 数据库存在的存储空间大、交换速度慢、读写速度慢等问题, 分别用 Bzip2, Gzip, GzipIO 这 3 种压缩算法对 Clob 和 Blob 的压缩和未压缩数据进行读写和交换速度测试, 使用直接读取、分段读取、分段 + 线程池读取 3 种方法进行了读库速度测试。结果表明: ①无论在存储、交换还是读写速度方面 Blob 均优于 Clob; ②Blob + Gzip 为地震前兆分秒数据的“最佳”存储结构, 读写和交换速度有大幅度提升, 数据库整体容量降至目前的 7% (或更少), 秒数据的交换速率至少是目前的 7.89 倍; ③最简单且被软件开发者广泛使用的直接读取方法读库效率较差, 分段 + 线程池技术无论在 Clob 还是 Blob、压缩还是未压缩时都表现出较高的读库效率, 给 LOB 数据读取速度带来飞跃式的提升。

关键词: Oracle; LOB; 数据压缩; 交换效率; 访问效率

中图分类号: P315.73 **文献标识码**: A **文章编号**: 1000 - 0666(2019)03 - 0447 - 07

0 引言

随着信息化社会的发展, 存储、传输和处理急剧增长的海量信息的压力越来越大。为了节省信息的存储空间和提高信息的传输效率, 数据压缩作为解决海量信息存储和传输的支持技术受到极大重视 (郑翠芳, 2011)。数据压缩技术一般分为有损压缩和无损压缩。无损压缩是指重构压缩数据 (还原、解压缩) 与原数据必须完全相同, 适用于要求重构信号与原始信号完全一致的场合 (李雷定等, 2009; 郑翠芳, 2011)。压缩算法必须能够提供较高的数据压缩率以支持实时数据库海量存储的特点, 压缩和解压缩两个过程都必须具有较好的速度性能 (刘红霞, 牛富丽, 2010)。

Bzip2 是基于 Burrows - Wheeler 变换 (BWT) 的无损压缩算法, 是一种不依赖于数据内部重复性的变换方法, 它能将数据中相同的字符有效地聚集到一起, 为进一步压缩创造条件 (李冰等, 2015; Seward, 2002, 2007)。它能够把普通的数据压缩至 10% ~15%, 压缩和解压效率都非常高, 按照开源软件协议在其网站上可下载完整的 C, Java, C#源代码及相应的 dll 接口文件。它因为较高的压缩比和速度性能在国外被广泛应用 (Gilchrist, 2008; Pankratius *et al*, 2009; Mccool *et al*, 2012; Salazar, Sánchez, 2017), 但在国内的应用仍然偏少 (李冰等, 2015)。数据压缩算法有 Bzip2 和 Gzip 两种, Net 的 System.IO.Compression 提供另一种 Gzip 压缩算法, 本文简称为 GzipIO。

2007 年底地震前兆台网“十五”系统正式建成并投入运行, 该软件系统是一个分层的 4 级互联互通分布式系统, 由台站、省局、国家中心和学科中心 4 级构成。为便于各级数据交换, 全国采用统一的数据库管理系统 (Oracle10g) 和统一的数据库表结构 (周克昌等, 2009, 2010; 刘高川, 2008)。软件系统主要有 2 个: 管理系统 (B/S 架构, 服务器运行) 和处理系统 (C/S 架构, 客户端 PC 机运行), 前者每天定时将台站数据逐级交换至省局、国家中心和学科中心 (刘高川, 2008), 后者负责每天的数据预处理和产品数据计算。

* **收稿日期**: 2018 - 10 - 24.

基金项目: 中国地震局地震科技“星火计划”专项 (XH17038) 和甘肃省地震局基本科研专项 (2013IESLZ02) 联合资助.

♣**通讯作者**: 赵银刚 (1979 -), 工程师, 研究方向为地震监测与软件研发. E - mail: 40857347@qq.com.

中国地震台网中心前兆台网部是全国地震前兆数据的汇集中心，同时也是容量最大的前兆数据库。截至2018年8月，该数据库中存储了3 328套观测仪器（秒采样仪器364套，分采样仪器2 126套，时、日采样仪器838套）的产出数据，数据库总量约8 000 GB，目前仍在以每年约800 GB的速度递增，其中时间分辨率为分和秒的数据（以下简称为分秒数据）总量占到数据库总是的95%以上。因为所有分秒数据全部采用Clob + ASCII未压缩的存储格式，数据库出现了存储空间大、交换速度慢、读写速度慢、运维困难等问题。如“处理系统”远程读取FHDZ－M15地磁仪器1天6个要素的秒采样数据需要约4 min；中国地震台网中心前兆台网部（8 000 GB）冷备份到另一台服务器需要连续拷贝10天左右，这10天必须关闭数据库并停止所有服务，这种冷备份方式显然不实用；而热备份系统（由地震系统自主研发）因为软件原因只能对应一台服务器，如果主库和备库同时出现问题导致数据丢失，结果将是灾难性的。

Oracle数据库中，Clob和Blob（简称LOB）是2个典型的大对象数据存储结构，在各级数据库中都有广泛使用。Clob只能存储单字节字符数据，多用于存储长文本数据；Blob用于存储无结构的二进制数据，主要存储图像、视频、音频及Word文档等带格式数据（张静，王永敏，2011）。为提高地震前兆数据库的存取效率，国内地震行业学者分别进行了如下研究：姚运生等（2006）提出把一串观测数据以二进制存储为一个记录的新的数据库表结构；李井冈等（2008）对比了Clob和Blob的存取速度，发现Blob具有更高的效率，主要原因如下：一是二进制数据比字符串型数据存取快；二是Blob类型占用更少的空间，网络传输快；刘坚等（2009）把LZMA压缩算法封装成API动态链接库，调用接口函数压缩大数据后保存至Blob字段，调用数据时将数据解压缩还原，不仅节省了数据库的磁盘空间，同时也提高了数据存取效率。

本文基于NET开发平台，分别使用Bzip2，Gzip，GzipIO这3种压缩算法对Clob和Blob的压缩和未压缩数据进行了读写速度、交换速度测试和比较，使用直接读取、分段读取、分段＋线程池读取3种技术进行了读库速度测试，对每种压缩算法和读库方法的优缺点进行了归纳和总结，以此检验适用于地震前兆数据库的“最佳”压缩算法和读库方法。

1 测试数据和研究方法

1.1 测试数据选取

本文选用的测试数据全部为2009年1月1日至兰州FHDZ－M15地磁仪器记录的共31天的6个要素分秒数据。该仪器为秒采样，每个要素每天包括86 400个秒采样数据，分数据由秒采样数据通过高斯滤波计算得到，每个要素每天包括1 440个分数据。

本文测试选用的本地服务器位于甘肃省地震局信息机房，远程服务器位于中国地震局地球物理研究所信息机房，本地和远程的测试数据和表结构完全相同。测试软件为Net开发平台编写的客户端软件（运行在办公室的PC机上）。

地震前兆数据库中分秒数据的表结构见表1，观测数据用Clob＋ASCII未压缩的存储格式，每套仪器每个要素每天一条记录。

表1 地震前兆数据库分秒数据表结构

Tab. 1 The structure of minute and second data table of seismic precursor database

字段描述	英文名	主键	字段类型	取值举例	取值说明
起始时间	startDate	√	date	2006－02－23	每天一条记录
台站代码	stationID	√	Char（5）	44 010	全国统一5位台站代码
测点编码	pointID	√	Char（1）	4	定位至每套仪器
要素代码	itemID	√	Char（4）	3 125	*D*，*H*，*Z*，*F*，*I*，*X*，*Y*
采样率代码	sampleRate		Char（2）	01	01：分数据；02：秒数据
观测值序列	obsValue		Clob	86.73 86.23 … 89.12	数据之间用空格符分隔，分数据每天1 440个，秒数据每天86 400个

1.2　LOB 数据压缩和解压缩方法

Net 中引用 Bzip2 提供的接口文件 ICSharpCode. SharpZipLib. dll 后，通过命名空间 ICSharpCode. SharpZipLib，调用 BZip2OutputStream 和 BZip2InputStream 分别完成 Bzip2 压缩和解压过程，调用 GzipOutputStream 和 GzipInputStream 分别完成 Gzip 压缩和解压过程，通过调用 System. IO. Compression. GzipStream 分别完成 GzipIO 压缩和解压过程。

1.3　数据库连接和 LOB 读写方法

Net 使用 ADO. NET 完成数据库访问，OracleConnection 进行数据库连接，OracleDataAdapter 和 DataTable 完成 LOB 数据读取和暂存，OracleCommand 完成 LOB 数据写入。

1.4　3 种 LOB 读库方法

（1）直接读取：对软件开发者而言，LOB 数据最简单最常用的读库方法就是直接使用 Select LobName 读取。

（2）分段读取：无论 Clob 还是 Blob 都可以使用 Oracle 自带的 DBMS_ LOB 包中的 substr 函数来分段读取数据，即 DBMS_ LOB. substr（lobName，n，pos），lobName 为 Lob 字段名，n 为读取的字节数，pos 为读取的起始位置。分段读取时 Clob 每次可读取的最大长度为 4 000 字节，Blob 为 2 000 字节，因此必须循环多次执行，每次循环都要重新设置读取的起始位置（pos），循环结束后需要把同一条记录的 LOB 数据按先后顺序拼接起来。

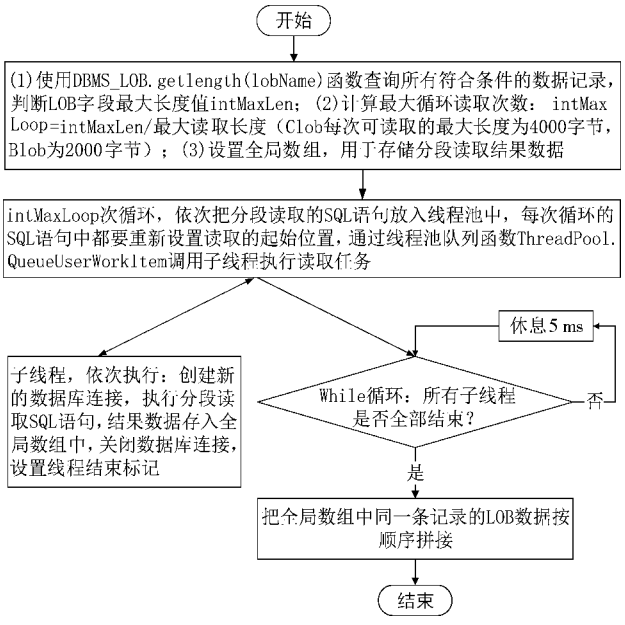

图 1　分段 + 线程池方法流程图

Fig. 1　Flow chart of the substr + threadPool method

（3）分段 + 线程池读取：NET 的 ThreadPool 类提供线程池管理，将分段读取的 SQL 语句依次放入线程池中，可以实现分段读取并行执行（多个线程同时读取）。图 1 为该方法读库流程图，使用时还需要编写一个单独的子线程类，该类中需要创建新的数据库连接，执行分段读取的 SQL 语句；任务全部添加进线程池后，还需要一个 while 循环，判断所有线程全部执行结束后才能退出循环执行后续操作。

2　LOB 数据压缩和交换速度测试

2.1　LOB 数据压缩测试

分别使用 Bzip2，Gzip，GzipIO 这 3 种压缩算法，对兰州 FHDZ - M15 地磁仪器产出的 2009 年 1 月分秒数据进行压缩测试。表 2 显示平均每条记录的压缩比，无论分或秒数据，Bzip2 的压缩比最高，Gzip 次之，GzipIO 最低；Bzip2 压缩后每条记录的容量最小，意味着占用更小的存储空间；但其压缩和解压时间最长，远高于其他 2 种算法，意味着读库（解压）和写库（压缩）消耗的时间更长。Gzip 的压缩时间大概是 GzipIO 的 2.5 倍，二者的解压时间相差很小，但分秒数据的二进制压缩比分别提高了 5% 和 3%。

2.2　LOB 数据交换速度测试

目前地震前兆管理系统软件在数据交换时采用的是“dbLink + Insert”技术，数据交换的核心命令为 insert into XX from XX @ dbLinkName，dbLinkName 为远程数据库的 dbLink。根据目前的地震前兆数据交换机制，交换过程中无需解析 LOB 数据，因此压缩和解压效率对交换速度没有影响，能影响到交换速度的只是每条记录的容量大小。登录远程（北京）数据库后直接运行该交换命令，其执行时间作为实际交换时间，即平均每条记录由本地（兰州）传输到远程（北京）的时间。兰州 FHDZ - M15 地磁仪器 2009 年 1 月所有分秒数据平均每条记录的数据交换速度测试结果见表 3，预估交换倍率 =“Clob 未压缩”容量/其他容量，实际交换倍率 =“Clob 未压缩”交换时间/其他交换时间。

由表 3 可见，无论 Clob 还是 Blob，3 种压缩结构的实际交倍率都没有预估交换倍率高，秒数据 Blob 压缩的实际交换倍率有 7 ~ 9 倍的提升，而分

表 2 平均每条记录的压缩比

Tab. 2 Compression rate of every record

数据	压缩算法	压缩前容量/KB	压缩后 Clob/KB	压缩后 Blob/KB	压缩时间/s	解压时间/s	Clob 压缩比	Blob 压缩比
秒数据	Bzip2	588.51	**37.462**	**28.096**	0.189 09	0.020 16	**6.24%**	**4.68%**
	Gzip	588.51	63.376	47.531	0.033 70	0.006 92	10.45%	7.84%
	GzipIO	588.51	86.906	65.179	**0.013 28**	**0.006 33**	14.36%	10.77%
分数据	Bzip2	8.842	**0.961**	**0.720**	0.004 58	0.001 13	**10.89%**	**8.15%**
	Gzip	8.842	1.399	1.048	0.001 00	0.000 34	15.76%	11.81%
	GzipIO	8.842	2.008	1.505	**0.000 35**	**0.000 27**	22.62%	16.95%

注：黑体数值为压缩后容量最小、压缩和解压时间最快、压缩比最高者.

表 3 平均每条记录的数据交换速度测试

Tab. 3 Data exchange speed test of every record

字段类型	压缩算法	秒数据				分数据			
		每条记录容量/KB	预估交换倍率	实际交换时间/s	实际交换倍率	每条记录容量/KB	预估交换倍率	实际交换时间/s	实际交换倍率
Blob（二进制）	Bzip2	28.09	20.9	0.051	8.97	0.72	12.3	0.043	1.13
	Gzip	47.53	12.4	0.058	7.89	1.05	8.4	0.044	1.11
	GzipIO	65.18	9.0	0.064	7.14	1.50	5.9	0.044	1.09
	未压缩	588.51	1.0	0.248	1.84	8.84	1.0	0.045	1.06
Clob（ASCII）	Bzip2	37.46	15.7	0.068	6.75	0.96	9.2	0.045	1.06
	Gzip	63.38	9.3	0.083	5.49	1.40	6.3	0.045	1.08
	GzipIO	86.91	6.8	0.106	4.30	2.01	4.4	0.046	1.05
	未压缩	588.51	—	0.456	—	8.84	—	0.048	—

数据的实际交换倍率提升幅度很小；秒数据 Blob 和 Clob 这 2 种未压缩结构相比，在存储容量完全相同的情况下实际交换倍率却有 1.84 倍的提升。

3 LOB 数据读写速度测试

3.1 3 种压缩算法的读写速度测试

使用直接读取方法对 Blob 和 Clob 的 4 种存储结构（3 种压缩 + 未压缩）进行读写速度测试，由表 4 可见：①就写库速度而言，无论 Clob 还是 Blob 速度最快是 GzipIO，Gzip 次之，两者相差极小，Bzip2 因为压缩时间长导致本地写库速度远慢于其它；②就读取速度而言，Clob 中 Bzip2 最快，Blob 中 Gzip 最快，即便偶有慢于其他方法的现象，读取速度仍与最快速度相差最小；③对同一种压缩或未压缩结构，2 种 LOB 类型的写库速度基本相当，但 Blob 的读库速度远优于 Clob。

3.2 3 种 LOB 读库方法的读取速度测试

分别使用 3 种 LOB 读库方法对压缩和未压缩结构进行读库速度测试，测试结果见表 5：①直接读取方法在 Clob 未压缩中读取效率最差，远慢于其他 2 种方法；在 Blob 中除了秒数据 1 天本地读取速度较快外，其他读取效率最差，并且随着读取天数的增加，与分段 + 线程池方法的读取速度差距增大。②分段读取方法在 Clob 未压缩中相对直接读取速度有大幅度提升，但在 Blob 秒数据读取时表现不稳定，频繁出现读取时间远高于其它方法现象。③分段 + 线程池方法在 Clob 未压缩中读取速度最快，远优于其他 2 种方法，在 Blob 中虽偶有略慢于其它方法，仍与读取最快速度相差极小。④无论哪种读库方法，Gzip 的读取速度都优于 GzipIO。

相对而言，与其他 2 种方法相比，无论在 Clob 还是 Blob，压缩还是未压缩，分段 + 线程池方法都能表现出最高的读库效率，尤其在 Clob 未压缩的读取速度有飞跃式的提升。Blob + Gzip 的存储结构结合分段 + 线程池读库方法，可使地震前兆数据库的读库性能达到“最佳”。

表 4　4 种存储结构的读写速度测试结果

Tab. 4　Read – write speed test results of four storage structures

字段类型	压缩算法	秒数据								分数据			
		1 天（2009 – 01 – 01）				31 天（2009 – 01 – 01—31）				31 天（2009 – 01 – 01—31）			
		本地		远程		本地		远程		本地		远程	
		读/s	写/s	读/s	写/s	读/s	写/s	读/s	写/s	读/s	写/s	读/s	写/s
Blob（二进制）	Bzip2	0. 20	1. 21	1. 73	2. 92	5. 36	44. 43	39. 98	96. 32	1. 74	2. 32	26. 76	51. 93
	Gzip	**0. 11**	0. 23	**1. 64**	2. 06	**2. 91**	6. 88	**39. 52**	63. 91	1. 48	1. 37	26. 71	49. 96
	GzipIO	0. 12	**0. 14**	1. 87	**2. 04**	3. 10	**4. 11**	45. 82	**63. 02**	1. 56	**1. 23**	26. 75	**49. 95**
	未压缩	0. 36	0. 37	4. 63	6. 43	11. 02	12. 28	194. 44	200. 60	**0. 97**	1. 40	**25. 83**	50. 39
Clob（ASCII）	Bzip2	**1. 03**	1. 23	**17. 15**	3. 01	**28. 54**	44. 72	**556**	105. 63	3. 07	2. 14	**43. 42**	51. 53
	Gzip	1. 47	0. 24	28. 24	**2. 13**	42. 52	7. 48	944	76. 76	2. 83	1. 41	46. 67	50. 38
	GzipIO	1. 95	**0. 16**	38. 21	2. 22	57. 29	**4. 97**	1 278	**76. 71**	**2. 69**	**1. 33**	55. 21	**50. 14**
	未压缩	11. 94	0. 39	243. 70	6. 58	372. 52	12. 58	7 568	205. 56	6. 25	1. 58	139. 15	50. 88

注：读取时间 = 读库时间 + 解压时间，写入时间 = 压缩时间 + 写库时间，黑体数值为 4 种存储结构中读写速度最快者 .

表 5　3 种 LOB 读库方法的读库速度测试结果

Tab. 5　Read speed test results of three LOB read methods

字段类型	压缩算法	读库方法	秒数据						分数据					
			1 天		10 天		31 天		1 天		10 天		31 天	
			本地	远程	本地	远程	本地	远程	本地	远程	本地	远程	本地	远程
Blob（二进制）	Bzip2	直接读取	**0. 19**	1. 80	1. 75	14. 72	4. 97	36. 56	0. 057	1. 374	0. 526	10. 51	1. 59	26. 84
		分段读取	0. 52	6. 85	1. 83	11. 33	4. 83	19. 39	0. 040	0. 954	0. 124	1. 32	**0. 25**	**1. 37**
		分段 + 线程池	0. 21	**1. 33**	**1. 45**	**5. 49**	**4. 18**	**10. 14**	**0. 039**	**0. 937**	**0. 122**	**1. 13**	0. 28	1. 44
	Gzip	直接读取	**0. 11**	1. 74	1. 03	16. 26	2. 88	41. 49	0. 058	1. 425	0. 476	10. 16	1. 52	26. 83
		分段读取	0. 66	11. 25	1. 41	16. 63	3. 18	27. 22	**0. 035**	0. 945	0. 062	1. 15	0. 11	1. 31
		分段 + 线程池	0. 14	**1. 64**	**0. 81**	**5. 37**	**2. 11**	**8. 57**	0. 041	**0. 940**	**0. 061**	**1. 11**	**0. 09**	**1. 23**
	GzipIO	直接读取	**0. 12**	**1. 91**	1. 09	17. 49	3. 08	45. 69	0. 065	1. 460	0. 534	10. 24	1. 67	27. 23
		分段读取	0. 88	14. 79	1. 71	22. 03	3. 79	36. 63	0. 049	1. 110	0. 074	1. 36	0. 11	1. 60
		分段 + 线程池	0. 17	2. 60	**0. 92**	**5. 79**	**2. 38**	**9. 09**	**0. 043**	**0. 956**	**0. 068**	**1. 18**	**0. 10**	**1. 48**
	未压缩	直接读取	**0. 38**	**6. 34**	**3. 66**	70. 01	10. 89	198. 20	**0. 045**	1. 414	0. 332	9. 77	1. 06	26. 15
		分段读取	6. 69	127. 46	12. 58	185. 94	25. 41	300. 55	0. 131	2. 860	0. 199	3. 49	0. 32	5. 14
		分段 + 线程池	0. 79	6. 84	4. 80	**20. 48**	**10. 76**	**40. 09**	0. 049	**0. 992**	**0. 099**	**2. 39**	**0. 20**	**3. 49**
Clob（ASCII）	未压缩	直接读取	12. 62	245. 94	121. 31	2 654. 53	384. 05	7 579. 19	0. 228	5. 681	2. 177	54. 24	6. 69	140. 84
		分段读取	4. 19	65. 97	13. 69	186. 07	23. 33	353. 76	0. 094	1. 944	0. 227	3. 94	0. 44	6. 65
		分段 + 线程池	**0. 56**	**5. 85**	**4. 37**	**18. 41**	**10. 87**	**39. 34**	**0. 052**	**0. 958**	**0. 116**	**2. 09**	**0. 26**	**3. 89**

注：读取时间 = 读库时间 + 解压时间，黑体数值为 3 种读库方法中速度最快者，黑体加框数值为 Clob 或 Blob 中速度最快者 .

4　讨论

实验表明 Blob 在存储性能上优于 Clob，但 Clob 字段在提高长文本数据的检索速度方面存在优势（张静，王永敏，2011）。上述测试结果再次验证了 Blob 无论在存储、交换还是读写速度方面均优于 Clob，但 Clob + ASCII 未压缩格式可以使用 DBMS_LOB. substr 函数读取部分数据（由分隔符反推每个数据的起始位置），此时的读取速度远优于整体读取，而 Blob 因为采取二进制存储而无法做到。对地震前兆数据库而言，读取部分数据的情况极少，大量的实际应用是整体读取（数据处理、绘图、下载等）。

最佳的压缩算法为压缩比最高、压缩和解压速度最快的算法，但实际中很难同时满足这样的要求。Bzip2 的压缩比最高，但压缩和解压时间偏长，Gzip 和 GzipIO 的压缩和解压时间短但压缩比略低。压缩和解压 2 个过程都必须具有较好的速度性能，这 2 个问题的解是相互矛盾的，我们就是要找到两者的平衡点，使其达到最优性能（刘红霞，牛富丽，2010）。

从表 2，4 和 5 的测试结果来看，3 种压缩算法的压缩比差异导致平均每条记录的容量不一样，进而导致读写速度与压缩算法相关，表 4 和 5 的读写时间包括了压缩和解压时间，以此来反映压缩和解压速度。如果单从分秒数据的读写速度来考虑，Gzip 和 GzipIO 优于 Bzip2；Gzip 和 GzipIO 相比，前者读库速度占优，后者写库速度占优，但 Gzip 的分秒数据压缩比高出 GzipIO 的 5% 和 3%，将为磁盘节省更多的存储空间，数据交换速度将更快。

对地震前兆数据库而言，如果采用 Blob + Gzip 存储结构，数据库整体容量降至目前的 7%（或更少），分秒数据读写速度有大幅度提升，秒数据交换速度至少是目前的 7.89 倍，台站数据可在最短时间内交换到 5 个前兆学科中心，从而让地震前兆数据发挥更为重要的时效性。

直接读取方法是最简单且被软件开发者使用最广泛的一种读库方法，但其读库效率较差。分段读取方法在 Clob 秒数据读取时平均循环 150 次（588.51 Kb/4 000 b）才能读取一条记录，但读取速度却远优于直接读取，这应该是它所使用的 Substr 函数将大对象转换为 Varchar2 类型所致；Clob 每次可读取的最大长度为 4 000 字节，Blob 为 2 000 字节，在记录容量相同的情况下，Blob 的循环次数是 Clob 的 2 倍，部分程度会导致 Blob 读取效率下降，这应该是它在 Blob 秒数据读取时表现不稳定，频繁出现读取时间远高于其他 2 种方法的根本原因。分段 + 线程池方法采用的是多线程并行读取技术，恰好弥补了这个不足，无论在 Clob 还是 Blob、压缩还是未压缩时都表现出了较高的读库效率。

分段 + 线程池方法的缺点是读取过程中要消耗大量的数据库连接，数据库中必须设置有足够的连接数量（Open_ Cursors）。NET 中线程池管理具有每个可用处理器最多 25 个线程的默认限制，笔者在大量的读库测试中监测到的最大并发线程数只有 19 个，虽然在 LOB 读取时同时开启的线程总数可能高达 200 ~ 300 个，但实际上并发读取的最多只能有 25 个，其他线程全部处于等待状态。全国地震前兆数据库目前的 Open_ Cursors 总数全部设置为 30 000，按此推算可同时支持1 200人使用该方法并发读库，而且数据库访问必须在地震行业专网下，该配置足以支持“分段 + 线程池”方法在地震系统内部使用。

2015 年 10 月，笔者利用 Bzip2 算法完成地磁学科中心数据库 Ceabak 表空间压缩，并提供给全国分析预报人员访问，所有分秒数据全部改用 Blob + Bzip2 压缩结构，压缩前地磁数据库总量约为 3 500 GB，压缩后为 126 GB，整体压缩比约为 3.6%，该压缩数据库运行至今未出现任何问题。2015 年 10 月，笔者在全国地磁台网数据处理软件 V2015.6 版本中将 Clob 读取改用分段读取方法（之前版本采用直接读取方法），2017 年 11 月在 V2017.2 版本中将 Clob 和 Blob 全部改用分段 + 线程池读库技术，这 2 次升级后该软件读库速度均有大幅度提升，目前每天的用户总数约 200 人次，台站人员利用该软件协助地磁学科进行分秒数据远程监控也得已实现。

5 结论

本文选取 2009 年 1 月 1 日至 31 日兰州 FHDZ - M15 地磁仪器的 6 个要素分秒数据，分别用 Bzip2，Gzip，GzipIO 压缩算法进行压缩测试，用 Clob 和 Blob 进行了读写和交换速度测试，并使用 3 种读库方法进行了读库速度测试，主要结论如下：Blob 无论在存储、交换还是读写速度方面均优于 Clob，但 Clob 在长文本数据检索方面存在优势；Blob + Gzip 为地震前兆数据库分秒数据的“最佳”存储结构，读写速度均有大幅度提升，数据库整体容量至少可降至目前的 7%（或更少），秒数据交换速率至少是目前的 7.89 倍，台站数据可在最短时间内交换到 5 个前兆学科中心，从而让地震前兆数据发挥更为重要的时效性；最简单且被软件开发者使用最广泛的直接读取方法读库效率较差，分段读取方法在 Clob 中远优于直接读取，但在 Blob 中表现不稳定，分段 + 线程池技术无论在 Clob 还是 Blob、压缩还是未压缩时都表现出较

高的读库效率，给 LOB 数据的读取速度带来飞跃式的提升。

本文所有测试数据全部来自国家地磁台网中心，在此表示诚挚的感谢！

参考文献：

李冰，龙冰洁，刘勇 . 2015. 一种基于后缀排序快速实现 Burrows – Wheeler 变换的方法[J]. 电子与信息学报，37(2)：504 – 508.

李井冈，姚运生，李胜乐，等 . 2008. 基于 Oracle 的地震前兆数据库表结构对比[J]. 计算机工程与设计，29(1)：243 – 245.

李雷定，马铁华，尤文斌 . 2009. 常用数据无损压缩算法分析[J]. 电子设计工程，17(1)：49 – 50.

刘高川 . 2008. 地震前兆数据交换系统设计[D]. 北京：中国地震局地球物理研究所，1 – 85.

刘红霞，牛富丽 . 2010. 实时数据库数据压缩算法探讨与改进[J]. 化工自动化及仪表，37(6)：72 – 75.

刘坚，李胜乐，王子影 . 2009. 基于 LZMA 的数据库压缩存储应用研究[J]. 大地测量与地球动力学，29(6)：144 – 147.

姚运生，李井冈，李胜乐 . 2006. 提高地震前兆数据库存取效率的新表结构[J]. 大地测量与地球动力学，26(3)：126 – 130.

张静，王永敏 . 2011. 数据库应用系统中 LOB 应用技术研究[J]. 计算机技术与发展，21(2)：166 – 169.

郑翠芳 . 2011. 几种常用无损数据压缩算法研究[J]. 计算机技术与发展，21(9)：73 – 76.

周克昌，蒋春花，纪寿文，等 . 2010. 地震前兆数据库系统设计[J]. 地震，30(2)：143 – 151.

周克昌，张崇立，纪寿文，等 . 2009. 中国地震前兆台网主要问题探讨[J]. 地震地磁观测与研究，30(1)：76 – 80.

Gilchrist J. 2008. Parallel Data Compression with bzip2[C]. Parallel & Distributed Computing & Systems.

Mccool M，Robison A D，Reinders J. 2012. Chapter 12 – Bzip2 Data Compression，in：Structured Parallel Programming[M]. Amsterdam：Elsevier Inc，291 – 297.

Pankratius V，Jannesari A，Tichy W F. 2009. Parallelizing Bzip2：A Case Study in Multicore Software Engineering[J]. IEEE Software，26(6)：70 – 77.

Salazar J S，Sánchez E A. 2017. Enhanced Parallel bzip2 Compression with Lock – Free Queue[J]. Uniciencia，31(2)：37 – 49.

Seward J. 2002. The bzip2 and libbzip2 official homepage (http：//sources. redhat. com/bzip2/).

Seward J. 2007. bzip2 and libbzip2，version 1. 0. 3，a program and library for data compression.

Research on Compression，Exchange and Access Efficiency of Seismic Precursor Oracle LOB data

WANG Jianjun[1]，ZHAO Yingang[2]，LIU Gaochuan[3]

(1. *Gansu Earthquake Agnecy*，*Lanzhou* 73000，*Gansu*，*China*)

(2. *Anqiu Earthquake Station*，*Weifang* 262100，*Shandong*，*China*)

(3. *China Earthquake Networks Center*，*Beijing* 100045，*China*)

Abstract

Aiming at the problems of huge storage space，low exchange speed and low read – write speed of the current seismic precursor Oracle database，the read – write speed and exchange speed tests are performed on the compressed and uncompressed Clob and Blob data by three compression algorithms，including Bzip2，Gzip and GzipIO. The reading speed test is performed by the direct reading，substr reading，and substr + threadPool reading techniques. The results show that：①Blob is superior to Clob in terms of storage，exchange，or read – write speed；②For the seismic precursor database，Blob + Gzip is the optimal storage structure of the minute and second data. The read – write speed is greatly improved，and the overall capacity of the database is reduced to 7% (or less). The exchange rate of the second data is at least 7. 89 times of the present rate；③The simplest and most widely used direct reading method by software developers has poor database read efficiency，while the substr + threadPool technique shows higher database reading efficiency no matter for Clob or Blob，for compressed or uncompressed，which brings a leap – forward improvement in the read speed of LOB data.

Keywords：Oracle；LOB；data compression；exchange efficiency；access efficiency