Computer Science and Application 计算机科学与应用, 2016, 6(3), 143-159
Published Online March 2016 in Hans. http://www.hanspub.org/journal/csa
http://dx.doi.org/10.12677/csa.2016.63019

Hans汉斯

Build and Research of the Meteorological and Air Pollution Data Warehouse System Based on WebGIS

Xin Zhang, Qun Yue, Meng Luo, Fei Xu

Key Laboratory of Geographic Information Science, Ministry of Education, College of Geographical Sciences, East China Normal University, Shanghai
Email: 463155245@qq.com

Received: Feb. 26th, 2016; accepted: Mar. 21st, 2016; published: Mar. 24th, 2016

Abstract

In the 21st century, weather and atmospheric pollution data warehouse system is a new generation of meteorological information storage management system, can greatly improve the existing huge amounts of meteorological and air pollution data's management and support for business research. Meteorological and air pollution data warehouse system is based on B/S architecture system, using OpenLayer as front-end WebGIS framework. The backend is supported by powerful RDBMS ORACLE database. ETL process is implemented by using PL/SQL and Java language. Business logic is implemented by Java language. Based on this, the meteorological and air pollution data warehouse system based on WebGIS is built. The system can share data in the form of visible, can display and manage data. At the same time, using the system of on-line analytical processing technology can help staff to make the decision to get useful information from data warehouses and to provide scientific and effective decision in a timely manner.

Keywords

ORACLE, Huge Amounts of Meteorological and Air Pollution Data, Data Warehouse, ETL, OLAP, WebGIS

基于WebGIS的气象及大气污染数据仓库系统的构建研究

张　新，乐　群，罗　蒙，徐　非

文章引用: 张新, 乐群, 罗蒙, 徐非. 基于 WebGIS 的气象及大气污染数据仓库系统的构建研究[J]. 计算机科学与应用, 2016, 6(3): 143-159. http://dx.doi.org/10.12677/csa.2016.63019

华东师范大学地理科学学院，地理信息科学教育部重点实验室，上海
Email: 463155245@qq.com

收稿日期：2016年2月26日；录用日期：2016年3月21日；发布日期：2016年3月24日

摘　要

气象及大气污染数据仓库系统作为21世纪新一代的气象信息存储发布管理系统，可以大大提升现有海量气象以及大气污染数据的管理水平和对业务科研的支持。气象及大气污染数据仓库系统是基于B/S架构的系统，采用OpenLayer作为前端WebGIS框架，后端则以强大的RDBMS数据库ORACLE为支撑，ETL过程使用PL/SQL和Java语言实现，业务逻辑则利用Java语言实现，以此构建基于WebGIS的海量气象及大气污染数据仓库系统。该系统以用户可见的方式进行共享数据发布、展示和数据管理，同时使用系统中的联机分析处理技术可以使决策人员从数据仓库中获得有用的信息，以提供科学和及时有效的辅助决策支持。

关键词

ORACLE，海量气象及大气污染数据，数据仓库，ETL，OLAP，WebGIS

1. 引言

大数据时代的来临揭示了一个全新数据时代的来临，大数据自然也包括气象数据以及大气污染数据，气象信息作为地球科学数据的重要组成部分，在人们日常生活、国家科学研究和全球气候研究等方面有不可替代的作用。加强气象信息高效率转换、存储、处理、共享的一体化工作，可以提高各个信息部门对气象资料的利用效率，是改善气象部门的服务水平和提升业务处理能力的重要一环[1]。21 世纪 Internet 的迅速发展使得传统的气象信息管理方式、存储方式、共享方式和发布形式已经无法满足这个大数据时代。然而在以前，气象资料都是存储为实体资料或者文本文件，布局分散而且重复，缺乏统一规范，没有一套集数据集成、数据存储、数据管理、数据检索、智能分析、地图发布等功能于一体的数据仓库系统，网络上发布的大多只是一些静态的表格或图片之类的气象资料，没有结合 WebGIS 进行空间展示、空间分析，更没有可以进行决策支持、数据挖掘的气象及污染数据仓库系统，人们对于更加灵活、生动的气象信息共享方式的需求愈加强烈[2]。

气象及大气污染数据数据仓库系统基于气象资料编码规范和命名规则，应用气象资料 ETL 分析技术、元数据技术以及数据存储管理优化技术等，支持海量数据的收集存储管理以及数据库检索，可以从不同的维度提取数据，整理合并为新的多维数据视图供用户调用，支持高效的大分析型批量查询，并进行快速响应[3]。基于 WebGIS 的气象大气污染数据数据仓库系统正是气象以及大气污染数据的承载方式，大大提高了气象资料存储管理的规范化程度、处理速度以及自动化程度，完成海量气象以及大气污染数据的高效存取、筛选、加工、后期计算、推演和预报等。

2. 气象及大气污染数据仓库系统体系结构

2.1. 系统整体体系结构

图 1 显示了系统的 4 层架构。数据层包括数据库系统和文件系统，是整个系统的最底层。应用服务器层也称之为中间层，它是数据库系统与应用系统之间的中间件，提供访问数据库的接口，负责转发用

户请求、中间信息传输管理之类的所有应用程序操作。业务逻辑层采用的是 B/S 架构中的 MVC 模式，它将不同的应用模块分开，实现应用之间解耦合。用户层通常也称之为表达层，通过浏览器为前端用户呈现出一个丰富的可交互性界面，在 WebGIS 地图上动态展示用户需要的信息，同时还提供地图交互、地理分析等交互功能[4]。

对于业务逻辑层，本系统采用的是 B/S 架构中的 MVC 模式。MVC 是 Model-View-Control 的简称，即模型－视图－控制器。它是一个存在于服务器表达层的模型，它将应用分开，改变应用之间的高度耦合。MVC 是在 20 世纪 80 年代发明的一种软件设计模式，至今已被广泛使用，最近几年被推荐为 Sun 公司 J2EE 平台的设计模式(见图 2)。

模型由 Hibernate 框架负责，完成数据持久化工作；视图由 Struts2 框架负责，实现逻辑应用层和用户层的分离；而控制器由 Spring 框架负责，负责整合管理 Hibernate 和 Struts2 框架，并对业务逻辑进行管理。框架的信息都写在配置文件中，由 Java 语言读取配置文件，将三大框架完全整合在一起，是的系

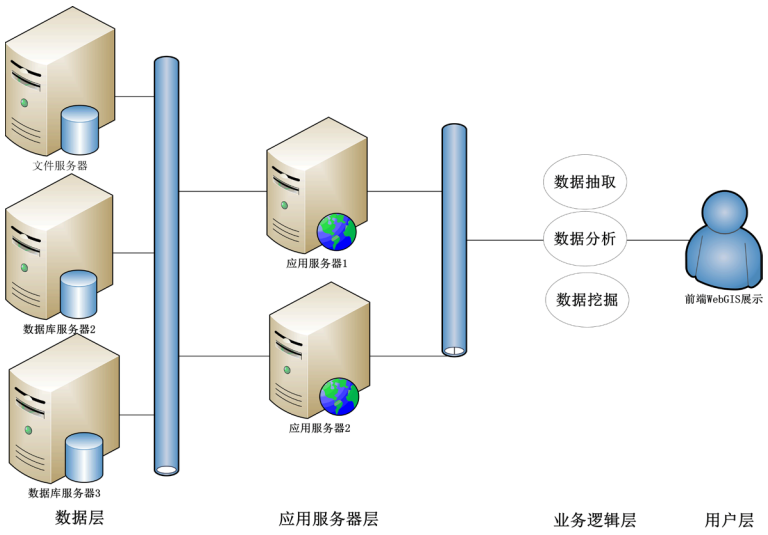

Figure 1. Meteorological and air pollution data warehouse system structure
图 1. 气象及大气污染数据仓库体系结构图

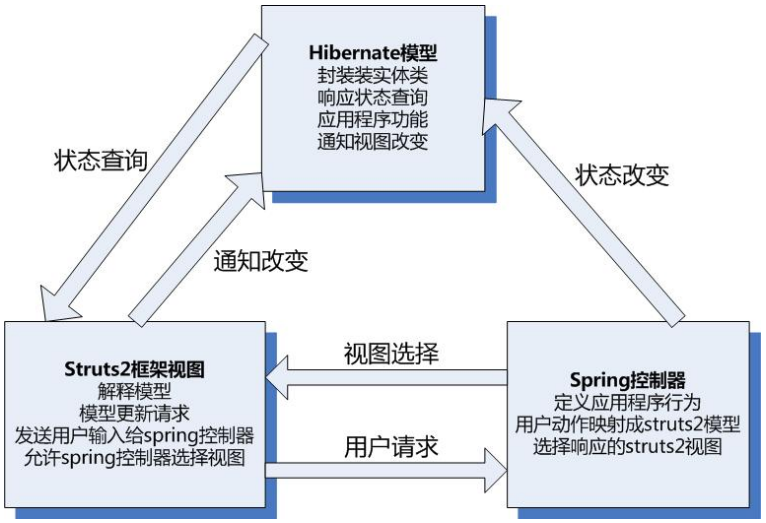

Figure 2. MVC pattern sketch
图 2. MVC 模式示意图

统层次清晰，扩展性良好，容易移植并且平台独立性比较强。

最后一层为客户层，实现方式为 Ajax、Java applet、OpenLayer 等，客户端浏览器通过 POST/GET 方式请求服务器端，然后实现页面展示。

2.2. 数据存储设计

所有 ORACLE 数据库都被设计成 OLAP 型数据库，实时性、事务性要求不高，但是侧重对决策人员的决策支持。同时为了应对根据分析人员快速的大数据量的复杂查询处理，一系列空气污染数据表和气象数据表均采用分区表策略，Oracle 的表分区通过改善可管理性、性能和可用性，为各式应用程序带来了极大的好处。通常，分区可以使某些查询以及维护操作的性能大大提高。此外，分区还可以极大简化常见的管理任务，分区是构建千兆字节数据系统或超高可用性系统的关键工具。

综合考虑数据结构、数据存储、文件管理系统和商用 RDBMS 数据库的优势和劣势，本系统的存储设计为文件系统和 RDBMS 数据库结合的方式。具体设计方式为：

1) 元数据统一采用 RDBMS 数据库存储。

2) 结构化的数据，比如用户数据、经纬度数据等，主要采用 RDBMS 数据库。

3) 卫星、格点、数值预报模式、雷达、遥感等非结构化数据主要采用文件系统与 RDBMS 系统混合的管理方式。这些非结构化数据的元数据利用 RDBMS 数据库管理，比如那些存储的文件系统的中的数据，它们的文件路径和文件名会被作为元数据管理起来，然后利用 RDBMS 数据库对元数据进行管理，实现两种存储方式的互补。

2.3. 数据实体设计

在数据库设计时，第一步就是根据系统需求和系统功能设计数据库实体，数据库实体在数据库中的表达就是数据库表。基于气象资料的编码规则和系统需求，各种不同类别的气象资料对应各自的实体结构。

气象资料一般可以分为两种，第一种是结构化的气象资料，结构化的气象资料是指各种观测站所测得的数据，从表现形式上又可分为两类，一类是从气象报文中提取的报告，另一类是从气象报文中提取的要素值，两种均为字符串格式，可以通过结构化的方式进行组织管理；第二种是二进制格式的气象资料，我们统称为非结构化数据。下面给出了本系统数据库中有代表意义的几个实体设计方案。

1) 常规天气公报实体设计方案

气象站观测到的信息可以通过各种传输手段传送，传送的一般是报文格式的数据，是各种气象要素的编码，所以需要在 ETL 过程中把报文的报头信息抽取出来，形成一个字段用来检索定位，与此同时，将报文的内容存储在数据仓库中。

如图 3，给出了常规天气公报实体设计方案图，其中 E_MARK(公报标识标志 ID)为主键，E_MARK(公报标识标志 ID)、E_ENTER_TIME(公报入库时间)、E_RECEIVE_TIME(公报收到时间)是用于管理的字段，其本身并不是气象数据，E_BULLETION(公报报文)是公报报文字段，用 CLOB 类型存储。该实体与其他实体没有任何关联关系，是独自的实体。

2) 气象要素实体设计方案

气象要素是对气象报文进行解码后得到的具有直观意义的信息。观测类型分为地上观测和空中观测，地上观测的气象要素一般分布在二维平面(比如地面观测站测得的信息)，而空中观测的气象要素一般分布在三维空间，包括了垂直空间(比如空中气球观测的信息)。二维平面空间中的气象站观测的气象要素是平面的，数目比较确定，所以各个气象站观测的要素数目大体一致；而三维空间中的气象站观测的是不同

OBS_WEA_BUL_ENT(常规天气公报实体)	
PK	E_MARK(公报识别标志ID)
	E_ENTER_TIME(公报入库时间) E_RECEIVE_TIME(公报收到时间) E_BUL_CENTER(编报中心) E_BUL_YEAR(编报年) E_BUL_MONTH(编报月) E_BUL_DAY(编报日) E_BUL_HM(编报时间-时分) E_BUL_CATEGORY(公报报类) E_CEOCODE(地理编码) E_BUL_NUMBER(公报编号) E_BUL_COR_MARK(公报订正标志) E_BUL_LEN(公报长度) E_BULLETIN(公报报文)

Figure 3. Conventional observation weather bulletin physical design scheme
图 3. 常规天气公报实体设计方案图

高度(气压)的气象要素，气象要素在不同高度(气压)的值是不一样的，往往不同高度的观测站测得的值相去甚远。

根据以上特点，在气象要素表设计时，要根据不同的气象要素特点设计各自的表结构。比如针对于二维地面观测资料，各个观测站都是站点编号，并且都是唯一的，像这样的实体设计成无关联的单表机构即可满足需求；但是对于三维高空观测资料，因为各个高空观测站的高度不一样，观测高度可能还会变动，每个高度可能有不同种类的要素，这样数据库表字段的数目就不好确定，所以多表关联来解决这个问题，即建立要素表和键值表。键值表存放观测站用于索引的唯一标识，要素表存放不同高度的气象要素信息。

在多表关联结构中，由于一个高空观测站可能在不同的高度进行观测，每个高度对其相关气象要素(如位势高度、温度、露点温度、风向、风速等)，因此键值表的每一行对应要素表的多行，每个观测站的要素表的行数取决于观测的位势高度。这种表结构设计方法有助于减少数据冗余，提高数据检索效率。当然，不管是地上观测的气象要素实体还是空中观测的气象要素实体，气象要素实体表结构都包含三个主要部分：主键字段(索引字段)、数据质量控制字段和气象要素字段。其中，主键字段是唯一不重复的，用于排除重复数据，避免不必要的冗余，可以唯一检索出一条气象要素信息，用于查看更新；入库时间字段一般用于审计之用；数据质量控制字段是气象上的一种置信度，是根据特定的质量控制算法算出的百分比，可以将其整合到 ETL 流程中，自动进行质量控制；气象要素字段来源于国内通信系统(除台北)实时上传的高空报文解报数据和地面自动站逐小时数据(Z 文件)。如图 4，展示了单表实体结构方案设计图的一部分。

3) 其他气象及污染数据实体设计

用户信息实体包括用户名、密码、邮箱、性别等用户数据字段，空气污染数据库包括经纬度信息实体、颗粒空气污染物数据实体、污染性气体数据实体，还有城市历史天气统计信息实体、城市历史天气综述实体、农业气象资料实体等。

2.4 文件存储路径设计

为了将非结构化数据合理的管理起来，就需要将其文件路径存储在关系型数据库中，而文件目录路径设计的就显得尤为重要。

SURF_WEA_ELE_ENT(常规地面气象要素实体)	
PK	E_RSURFID(地面观测站点ID)
	E_ENTER_TIME(资料入库时间) E_LY(资料时次年) E_LM(资料时次月) E_LD(资料时次日) SHIP_CALL(船舶呼号) V_ROOM(国家/区域代码) STA_IND_NUM(区站号) E_OBSER_LY(资料观测年) E_OBSER_LM(资料观测月) E_OBSER_LD(资料观测日) E_OBSER_LHM(资料观测时分) V_LON(经度) V_LAT(纬度) V_HEIGHT(测站高度) V_STA_TYPE(台站类型) V_WIND_SPPED(风速) V_WIND_DIRE(风向) V_DRY_HNMIDITY(干球温度) V_DEW_HUMIDITY(露点湿度) V_RELA_HUMIDITY(相对湿度) V_HOR_VIS(水平能见度) V_CUR_WEA(现在天气) V_TOL_CLOUD(总云量) ..... F_WIND_DIR_REL(风向可信度) F_WIND_SPEED_REL(风速可信度) F_ARE_TEMP_REL(干球气温可信度) F_DEW_HUMIDITY_DEL(露点湿度可信度) F_REL_DEW_HUM_DEL(相对湿度可信度) .....

Figure 4. Conventional ground meteorological elements entity figure
图 4. 常规地面气象要素实体方案图

首先根据二进制文件的类型建立相应根节点目录，数值模式产品根目录名称为 Data Schema，遥感资料根目录的名称为 remote sensing，雷达资料根目录的名称为 radar，卫星资料的根目录名称为 satellite。

其次，由于非结构化数据的数据量庞大，需要为其设计二级缓存，将近期可能会用到的数据暂放在缓存中，既提高了数据存取效率，又方便数据快速迁移。

最后，根据数据本身的特点，建立子目录，以数值模式产品为例，子目录分为 T639 全球中期天气数值模式产品、GRAPES_MESO 中国及周边区域数值模式产品和 T213L31 中期分析预报系统产品等。

2.5. 元数据设计

2.5.1. 气象及污染数据仓库系统元数据

数据仓库系统中，元数据的应用相当广泛，基本贯穿于系统的设计始终，包括数据实体创建、资料入库维护、资料检索等方面，对于提高系统灵活性、方便管理、提高效率等方面有很大的作用。

本系统中的元数据主要包括以下几种：

1) 气象及污染资料元数据

气象及污染资料元数据用来存储数据种类信息，有结构化元数据和非结构化元数据，这些种类信息用于检索系统和导航信息。

2) 站点号元数据

站点号元数据包含了站点信息，包括区域站点信息表、公报站点信息表、要素站点信息表，站点分布的区域有全球、亚洲、中国以及自定义区域。

3) 控制信息元数据

控制信息元数据包含了日志文件的保留时间、过期数据的过期期限等。

4) 业务数据备份元数据

业务数据备份元数据包含了业务数据的备份时间、备份频率、备份位置等。

5) 格点元数据

格点元数据包含了国内外各种网格场的定义信息。

2.5.2. 元数据在气象及污染数据仓库系统中的作用

元数据可以帮助促进数据集的高效利用，有效的维护和管理数据，提供有关数据的各种检索信息。

1) 数据库实体创建中元数据的作用

由于数据仓库系统中有非常多的表、索引、簇、视图、存储过程、函数、物化视图等数据库对象需要创建和维护，以人工的方式进行创建和管理无疑是一个非常低效的过程，而且有可能因为人为失误造成不可逆的损失，因为本系统中设计了基于 WEB 的气象数据对象管理模块，用于实现气象数据对象元数据的创建、管理和维护。

首先，用记事本编辑一个数据库实体描述 XML 文件，将 XML 文件中的数据库实体信息录入对应的元数据表中，具体元数据表关联图如图 5。

从图 6 可以看出 XML 文件中的数据库实体信息以及关联关系，当数据仓库管理员需要创建又或者维护一张数据表的时候：

1) 将 XML 作为配置文件，其中的信息传输到前端 WEB 页面，数据库管理员通过前端的 WEB 页面选择数据名称，通过 MDT_DT_TYPE 选择种类别名。

2) 根据种类别名，通过 MDT_DT_INFO 获得数据名称对应的要素表名、键表名、要素表是否分区和键表是否分区等信息。

3) 如果需要创建表分区，根据要素表名链接到 MDT_DT_Part_Info，从 MDT_DT_Part_Info 中可以得到分区表名、分区级别、分区字段、分区字段数据类型、分区类型、分区值、分区名和所在表空间等信息。

4) 根据要素表名链接到 MDT_Data_Cons_Info 和 MDT_DT_Index_Info，从中获得建表需要的索引信息和约束信息。

经过以上步骤，数据库管理员通过 WEB 页面就可以获得创建或者维护一张气象数据表所需的大部分信息。

3. 分区存储

3.1. 数据库分区技术

数据库分区的基本原理就是通过访问一个表或者索引的较小片断，而不是访问整个表和索引，以提

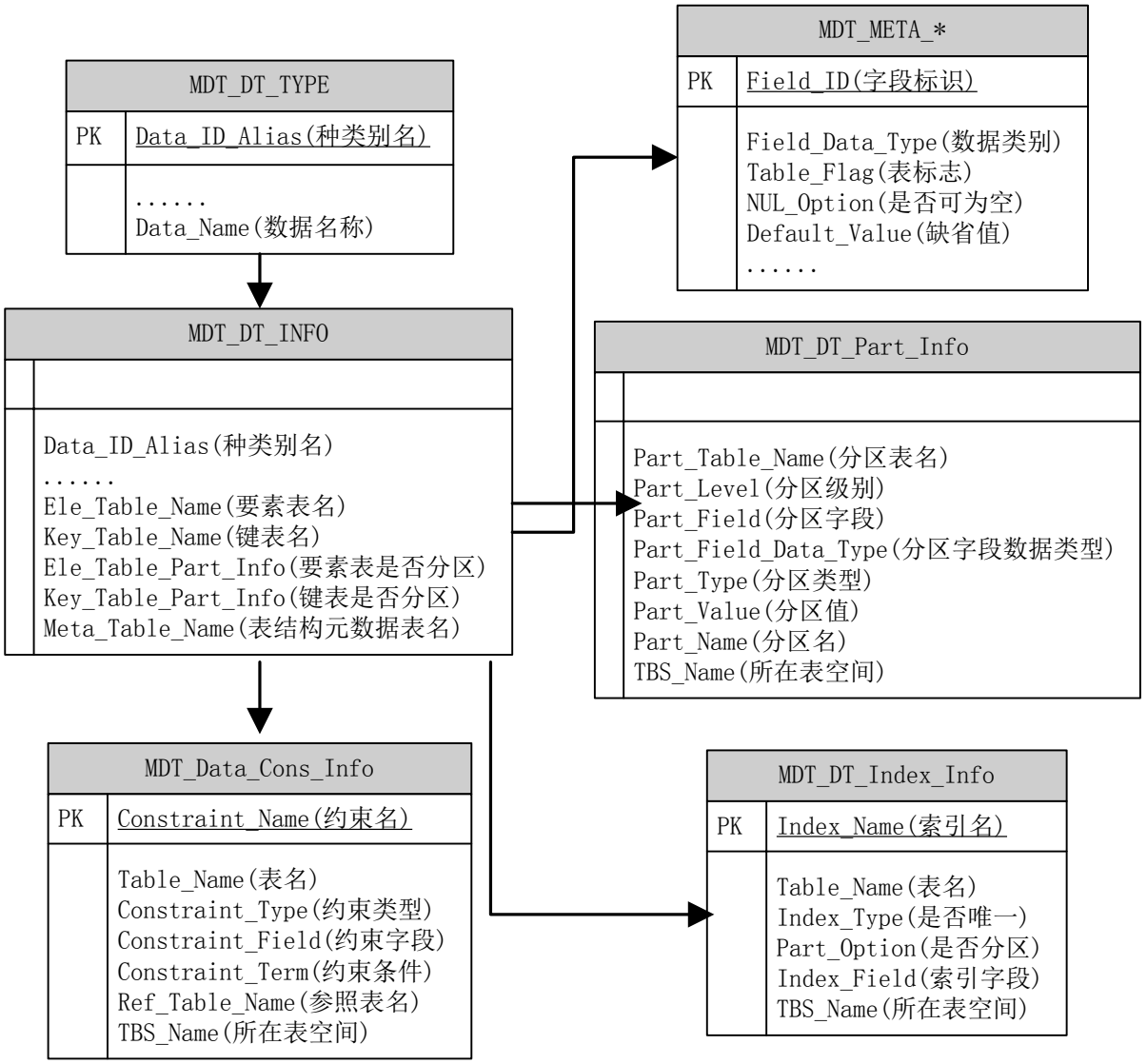

Figure 5. Metadata tables associated diagrams
图 5. 元数据表关联图

高数据库的性能。如果将一个表的不同分区放置在不同的磁盘上，磁盘整体的吞吐量就会成倍上升，并且分区表和普通的数据表的操作是一样的，对于数据存储和检索具有透明性。

数据库分区技术在数据库管理和性能上有以下优点：

1) 提高数据的可用性、安全性。
2) 减轻管理维护的负担。
3) 改善某些查询的性能。
4) 对应用程序的透明性。

3.2. 分区方案的测试

现有数据库的规模肯定不会恒定不变，随着资料的不断入库，各资料的保存为备份期限一般都在六个月以上，在这种情况下，同种资料的所有数据如果都放在一张表中的话，那么这张表就有可能成为超

过 1000 万行的大表。上面也讨论过，这种架构的数据库在存储方面的性能会随着数据量的增加而不断下降。鉴于 Oracle 数据库的分区方案在提高数据的可用性、安全性、减轻管理的负担、改善某些查询的性能和对应用程序的透明性等方面良好的设计，考虑采用 Oracle 数据库的分区方案，但是分区多种多样，性能差别也很大。

以常规地面要素资料为例，分别设计了五种分区方案进行模拟测验，根据实验结果决定使用哪种分区方案。

对常规地面要素资料设计以下五种存储结构：

1) 单表方式。

2) 多表多个分区方式(一天一张表，每张表对 E_OBSER_LHM 即资料观测时分字段进行列表分区，共 4 个分区)。

3) 单表多个分区方式 1(对 E_OBSER_LD 即资料观测日字段进行列表分区，总共 31 个分区)。

4) 单表多个分区方式 2(对 V_LON 即经度进行范围分区，总共 25 个分区)。

5) 单表多个分区方式 3(对 E_OBSER_LD 即资料观测日字段进行范围分区，总共 31 个，再对 V_LON 即经度进行散列子分区，每个范围分区共 16 个散列子分区)。

对 31 天常规地面要素资料插入、删除和查询的性能进行测试结果如下：

1) 将 31 天的常规地面要素资料插入数据库，单表方式比分区方式插入速度快，其次是单表多个分区方式 1、多表多个分区方式和单表多个分区方式 2，它们三个插入速度相当，最后单表多个分区方式 3 略慢。

2) 对于删除一天的常规地面要素资料，单表多个分区方式 2 和单表多个分区方式 3 速度最快，并且用时接近，比单表多个分区方式 1 略快，而单表方式和多表多个分区方式速度最慢且相当。

3) 对于常规地面要素数据查询，单用户模拟查询，取多次查询时间的平均值，单表多个分区方式 2 和单表多个分区方式 3 速度相当，并且比其他方式要快很多，另外三种速度相当；多用户模拟并发查询，单表多个分区方式 3 最快，其次单表多个分区方式 2，另外三种速度相当。

3.3. 分区方案的选择

综合考虑数据库在管理上的便捷性和灵活性以及查询的速度性能，结合测试结果，针对气象数据仓库系统的数据库分区方案制定了合理的分区方案。

由于实时气象资料主要针对于实时的业务需求，对于时间的准确性、时效性以及查询效率有更高的要求，所以，对这类数据(有结构化数据也有非结构化数据)以 12 个月为单位构建 12 张表，即一年 12 个月，每个月份对应一张数据表。对于每一张数据表，以不同月份的天数来创建范围主分区，即一个月份一张数据表，这张数据表的主分区数就是当月的天数。对于查询性能要求更高的数据，可以在主分区上根据经度进行散列子分区，每个主分区共 16 个散列子分区。当然，资料量不大的数据可以不用考虑分区，单表即可满足要求。也有一些常年不变、长期保存的历史数据，按照年份进行范围分区，而以区域进行划分的数据可以按照区域代码进行散列分区。

以常规地面要素资料为范例资料，以单表多个分区方式 3 为范例方式，将 2015 年 1 月作为资料入库时间，给出创建分区表的部分 SQL 语句：

```
createtableSURF_WEA_ELE_ENT
(
E_RSURFID varchar2(32) primary key,
E_ENTER_TIME date not null,
```

```
E_LYvarchar2(10),
E_LM varchar2(10),
E_LD varchar2(10),
SHIP_CALL varchar2(30),
V_ROOM varchar2(30),
STA_TND_NUM varchar2(30),
E_OBSER_LY varchar2(10),
E_OBSER_LM varchar2(10),
E_OBSER_LD varchar2(10),
E_OBSER_LHM varchar2(10),
V_LON number,
V_LAT number,
……
F_WIND_DIR_REL number,
F_WIND_SPEED_REL number,
F_ARE_TEMP_REL number,
F_DEW_HUMIDITY_DEL number,
F_REL_DEW_HUM_DEL number,
……
)
partition by range(E_ENTER_TIME)subpartition by hash(V_LON) subpartitions16 store in
(LON_1, LON_2, LON_3,……, LON_14, LON_15, LON_16)
(
partition part_01 values less than(to_date('2015-01-01','yyyy-mm-dd')),
partition part_02 values less than(to_date('2015-01-02','yyyy-mm-dd')),
partition part_03 values less than(to_date('2015-01-03','yyyy-mm-dd')),
……
partition part_29 values less than(to_date('2015-01-29','yyyy-mm-dd')),
partition part_30 values less than(to_date('2015-01-30','yyyy-mm-dd')),
partition part_31 values less than(to_date('2015-01-31','yyyy-mm-dd'))
);
```

3.4. 分区在数据库管理、性能提高上的应用

1) 数据清理

随着时间的推移，系统中的实时数据有一部分会过期，对于这些过期的数据可以以分区为单位进行截断(TRUNCATE)操作，对其他分区没有影响，而且截断操作不计入回滚表空间，速度远快于 DELETE 操作。

2) 滚动视窗

由于实时观测数据是每天都会更新，所以每天都会向表中加载新数据，并且分区功能支持数据仓库中的‘滚动视窗’加载进程，因此可以对该表进行范围分区，使每个分区包含一天的数据。这样加载进

程只是简单地添加新的分区。添加一个分区的操作比修改整个表效率高很多，因为 DBA 不需要修改任何其他分区。

3) 分区智能联接

分区功能可以通过称为分区智能联接的技术提高多表联接的性能，因此在系统中，如果当两个表要联接在一起，而且每个表都用联接键来分区时，我们就可以使用分区智能联接。分区智能联接将大型联接分解成较小的发生在各个分区间的联接，从而用较少的时间完成全部联接，这给系统中的串行和并行的执行都带来显著的性能改善。

4) 数据备份与恢复

对于以日或者月为单位进行分区的分区表，可以以日或者月为单位进行逻辑导出来备份以及逻辑导入来进行数据恢复，没有进行备份或者恢复的其他分区不会受到影响。对于非结构化的数据，备份时不仅要备份数据库中的元数据，还要备份文件系统中的数据。

4. 系统优化

4.1. 并行和 OLAP 系统

4.1.1. 并行处理机制

并行技术是OLAP系统的一个非常重要的技术，这种数据并行处理方式在0LAP系统非常有用，OLAP系统的表通常来说都非常大，如果系统的 CPU 比较多的话，让所有的 CPU 一起共同来处理这些数据，效果就会比串行执行要高的多；然而对于 OLTP 系统，通常来讲，并行并不合适，原因是 OLTP 系统上几乎在所有的 SQL 操作中，数据访问路径基本上以索引访问为主，并且返回结果集非常小，这样的 SQL 操作的处理速度一般非常快，此时不需要启用并行。

4.1.2. 并行执行的适用范围以及 SQL 并行优化实例

Oracle 的并行技术在下面的场景中都可以使用：Parallel Query(并行查询)、Parallel DDL(并行 DDL 操作，比如建表，建索引等)、Parallel DML(并行 DML 操作，比如 INSERT, UPDATE, DELETE 等)。

对于气象及污染数据仓库系统中的查询操作，可以使用并行技术优化查询语句速度。通过E_BUL_DAY(编报日)进行分组，以 E_RECEIVE_TIME(公报入库时间)进行排序，查询常规天气公报数据，这种分组排序 SQL 是典型的 OLAP 系统的 SQL，其 SQL 语句如下所示：

select E_MARK, E_ENTER_TIME,E_RECEIVE_TIME,E_BUL_CENTER,E_BUL_YEAR,E_BUL_MONTH,E_BUL_DAY,E_BUL_HM,E_BUL_CATEGORY,E_CEOCODE,E_BUL_NUMBER,E_BUL_COR_MARK,E_BUL_LEN,E_BULLETIN from OBS_WEA_BUL_ENT group by E_BUL_DAY order by E_RECEIVE_TIME desc;

然而一个查询能够并行执行，需要满足以下条件：

1) SQL 语句中有 Hint 提示，比如 PARALLEL 或者 PARALLEL_INDEX。

2) SQL 语句中引用的对象被设置了并行属性。

3) 多表关联中，至少有一个表执行全表扫描(FULLTABLESCAN)或者跨越分区的 INDEXRANGE SACN。

本系统中应用的方法是 Hint 的方式，修改后的 SQL 如下所示：

select/*+parallel(OBS_WEA_BUL_ENT4)*/E_MARK,E_ENTER_TIME,E_RECEIVE_TIME,E_BUL_CENTER,E_BUL_YEAR,E_BUL_MONTH,E_BUL_DAY,E_BUL_HM,E_BUL_CATEGORY,E_CEOCODE,E_BUL_NUMBER,E_BUL_COR_MARK,E_BUL_LEN,E_BULLETIN from OBS_WEA_BUL_ENT group by E_BUL_DAY order by E_RECEIVE_TIME desc;

上面的 SQL 语句通过加上 Hint 提示，即加上/*+parallel(OBS_WEA_BUL_ENT4)*/便可以利用多个 CPU 进行并行查询，提高查询速度，本例中并行度为 4，即利用 4 个并行服务进程进行数据扫描(子进程)，另外 4 个并行服务进程负责数据的排序(父进程)，他们同时工作，子进程实时将扫描到的数据块传递给父进程，父进程同时将数据排序；父进程按照字段 E_RECEIVE_TIME 的列值将结果集分成了四个部分，这样每个子进程将相应的数据传到对应的父进程上，所以每个子进程都可能会和所有的父进程有数据传递，最后父进程将所有的排序数据传递给并行服务协调进程，由它将最终的结果返给用户。

需要注意的是并行查询不可以使用在一个远程引用的对象上(比如使用 DBLINK 访问的对象)。

4.2. 直接加载

Oracle 提供两种类型的插入语句：常规插入(conventional insert)和直接路径插入(direct-path insert)，直接路径插入的目的是为了高效地加载大量的数据，它以牺牲部分功能为代价，因此受到很多的限制。直接路径加载把数据直接插入到要修改的段的高水位(HWM)以上，从而生成了最少量的 undo(只生成数据字典的 undo，不生成块中数据的 undo)，且不通过高速缓存，因此它的性能比常规插入要好。如果性能是首要目标，还可以考虑配合使用最小日志模式(nologging)。

直接路径加载的限制：

1) 一张表同时只能有一个直接路径插入，因此不适合小数据量的插入，只适合大批量的数据加载；

2) 在 HWM 下的空闲空间不会被利用；

3) 一张表在做直接路径插入的同时，同一会话不能对其做任何操作(select 都不可以)；

4) 只有 insert into …… select ……语句、merge 语句和使用 OCI 直接路径接口的应用程序才可以使用。

根据直接加载的优缺点，决定将直接加载应用在数据迁移中，主要涉及中间临时表到业务表的转换。还是以从 SURF_WEA_ELE_TMP(常规地面要素资料临时表)中抽调数据到 SURF_WEA_ELE_ENT(常规地面要素资料表)为例，在并行的基础上增加直接加载的 Hint，SQL 又变为如下形式：

insert/* + append parallel(SURF_WEA_ELE_ENT 4)*/into SURF_WEA_ELE_ENT select/* + parallel(SURF_WEA_ELE_TMP) */* from SURF_WEA_ELE_TMP;

直接加载和并行执行一起使用时，Oralce 会按照并行度启动相应数量的并行服务进程，像串行执行的直接加载的方式一样，每个并行服务进程都单独分配额外的空间用于加载数据(实际上 Oracle 为每个并行服务程序分配了一个临时段)，每个并行服务进程将数据首先加载到各自的临时段上，当所有的并行进程执行完毕后，将各自的数据块合并到一起，放到表的高水位之后，如果事务提交，则将高水位线移到新加载的数据之后，这样数据插入的性能将会大大提升。

4.3. 分区索引

针对于分区表的索引有两种，一种是全局索引(global index)，另一种是本地索引(local index)。全局索引即可以分区，也可以不分区。即可以建 range 分区，也可以建 hash 分区，即可建于分区表，又可创建于非分区表上，就是说，全局索引是完全独立的，因此它也需要我们更多的维护操作。GLOBAL 索引是全局类型的索引，根据实际情况可以调整分区的类别，而并非按照分区结构定义来组织，维护代价相对较高一些，在 OLTP 环境用得相对较多。

本地索引其分区形式与表的分区完全相同，依赖列相同，存储属性也相同。对于本地索引，其索引分区的维护自动进行，就是说你 add、drop、split、truncate 表的分区时，本地索引会自动维护其索引分区。一般使用 LOCAL 索引较为方便，而且维护代价较低，并且 LOCAL 索引是在分区的基础上去创建索

引，类似于在一个子表内部去创建索引，这样开销主要是在分区上，可以很规范的管理起来，在 OLAP 系统中应用很广泛。

鉴于本地索引对于数据仓库系统的契合度高，并且使用和维护较为方便，所以本系统采用的分区索引是本地索引。依然以常规地面要素资料为范例资料，给出创建 SURF_WEA_ELE_ENT(常规地面要素资料实体)表的本地分区索引的部分 SQL 语句。

```
create index IDX_SWEE_E_ENTER_TIME on SURF_WEA_ELE_ENT(E_ENTER_TIME) local
(
partitioni_range_p1 tablespace tbspart01,
partition i_range_p2 tablespace tbspart01,
partition i_range_p3 tablespace tbspart01,
……
partition i_range_p29 tablespace tbspart02,
partition i_range_p30 tablespace tbspart02,
partition i_range_p31 tablespace tbspart02
);
```

由以上 SQL 可以看出，本地索引的分区完全继承分区表的分区的属性，包括分区类型，分区的范围值即不需指定也不能更改，也就是说本地索引的分区维护完全依赖于其索引所在表，这样维护起来方便快捷。

5. 系统功能

5.1. ETL 功能

ETL 分别是“Extract”、“Transform”、“Load”三个单词的首字母缩写也就是“抽取”、“转换”、“装载”，但我们日常往往简称其为数据抽取。ETL 是 BI/DW(商务智能/数据仓库)的至关重要的一步，按照不同的业务规则提炼并提升数据的价值，完成数据源向目标数据仓库的转化，是搭建数据仓库的关键。

图 6 说明了气象及大气污染数据库 ETL 流程。

由于气象及大气污染数据来源广泛、种类繁多、数据量庞大、数据格式不一、无效值空值较多，所以本系统并没有采用 Informatics 等 ETL 工具，利用了更为灵活多变的方法，比如本系统对于气象及大气污染数据文件和气象及大气污染信息网站的 ETL 过程主要由 Java 语言实现，而气象及大气污染数据库的 ETL 过程主要由 PL/SQL 和 Java 实现。具体来说，对于 txt 文本类型的气象及大气污染数据文件主要利用 Java 的 BufferedReader 类来完成，excel 文件类型的气象及大气污染数据库文件主要利用 Java 的 POI 3.13 [5]组件来处理，而气象及大气污染信息网站上的信息主要通过 Java 的 Jsoup 1.8.3 [6]组件来进行抓取；对于其他气象及大气污染数据库中的气象及大气污染数据，如果同样是 ORACLE 数据库并且能建立 dblink 无疑是最快的方式，如果不能建立 dblink 的话，expdp、impdp 和 SQL 脚本的导入导出都是可用的方法，如果是不同的数据库 sql 脚本的导入导出是最常用的方法，以上过程就是 Extract 过程。气象及大气污染数据文件和气象及大气污染信息网站得到的数据在项目中将会以数组的方式呈现，而对于其他气象及大气污染数据库中得到的气象及大气污染数据在项目中将会以 expdp 文件和 SQL 脚本的方式呈现，这一转换过程称之为 Transform。以数组方式呈现的数据由 Java 调用 ORACLE 数据库中的存储过程并传递数组到 PL/SQL 运行环境中，传入到存储过程中的数组在 PL/SQL 运行环境中利用 forall 语法执行整体大批量插入，不用在 PL/SQL 运行环境和 SQL 运行环境之间进行来回切换，而且自动使用绑定变量，比 for 语

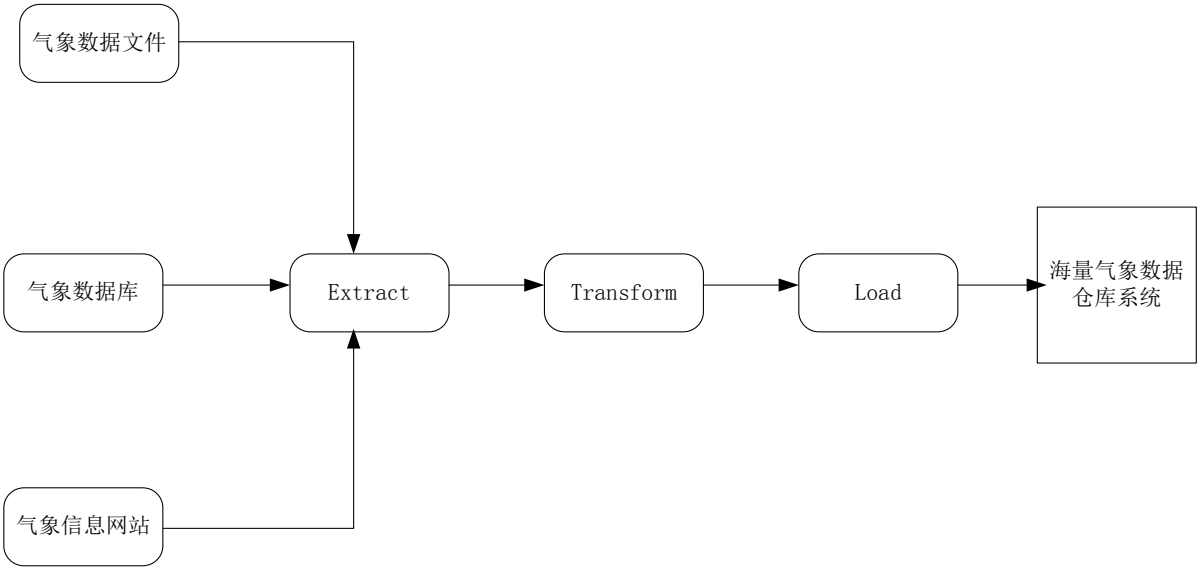

Figure 6. Meteorological and air pollution data flow chart of the ETL
图 6. 气象及大气污染数据 ETL 流程图

法遍历执行 insert 语句的速度快 10 倍以上，大大提高 Load 效率；而对于 dblink 的方式用 create table t1_name as select * from t2_name@dblink_name 即可，方便快捷，那么 expdp 文件和 SQL 脚本在 SQLPLUS 下面运行 impdp 命令和执行 SQL 脚本即可，这一过程称之为 Load。

针对于遥感数据、卫星云图、数据预报产品等二进制大文件类型的数据，从硬件层面进行数据迁移无疑是效率最高的方式。

5.2. 相似分析

ETL 过程中对空值的处理尤为重要，本系统对空值的处理方法就是利用相似分析。

以空气污染数据为例，进行相似分析，并将相似分析整合到系统中去。在空气环境自动站检测过程中，由于仪器故障、系统故障、操作失误、停电等各种各样的原因，数据的缺失是一个长时间以来一直存在的问题。

假设样本的 x 因子和 y 因子分别可以用平面直角坐标系的 x 轴和 y 轴完全表示，那么一个样本集合在该坐标系中就可以以一条样本曲线的方式呈现，很明显，如果将两条曲线进行比较，影响这两条曲线相似度的不仅仅是两条曲线的相近程度(值相似)，还有两条曲线的形似程度(形相似) [7]。

空气污染检测系统通常都是一天 24 个时次，每小时产生一个值，这样的话，如果数据完整，我们就可以把一天 24 个值放在直角坐标系中，描绘成一条曲线，即污染数据日曲线。查找最近几年完整的日污染数据，提炼出样本，组成样本集合。若某一天的空气污染数据有 k 个缺测值，那么由余下的 24-k 个值也可以绘制成一条曲线，根据相似分析原理，可以在样本集合中找出一个完整的样本，其日变化曲线与缺测曲线相应部分形状类似并且贴近，那么该完整样本的值可以代替缺测的值。

本系统利用已有完整数据样本，利用 Java 语言实现了相似分析，利用相似离度[8]对缺测值进行合理的插值。图 7 显示的是南京在 2014 年 11 月 02 号的 Line Chart，红色是 AQI，蓝色是 PM2.5，绿色是 PM10，横坐标是一天 24 小时，纵坐标是污染物浓度的大小范围。其中，05 时和 14 时的 AQI、PM2.5 和 PM10 等污染物的值因为某种原因缺测，所以值为零。在 ETL 过程中，利用相似分析完成对缺测值的插值，如图 8，可以看到缺失数据拟合状况良好，能够反应出当天的空气污染变化趋势。

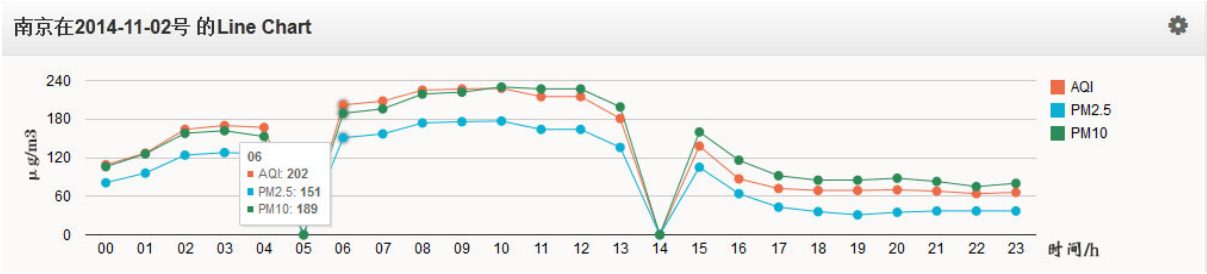

Figure 7. Air pollution data linear graph before interpolation
图 7. 插补之前空气污染数据线状图

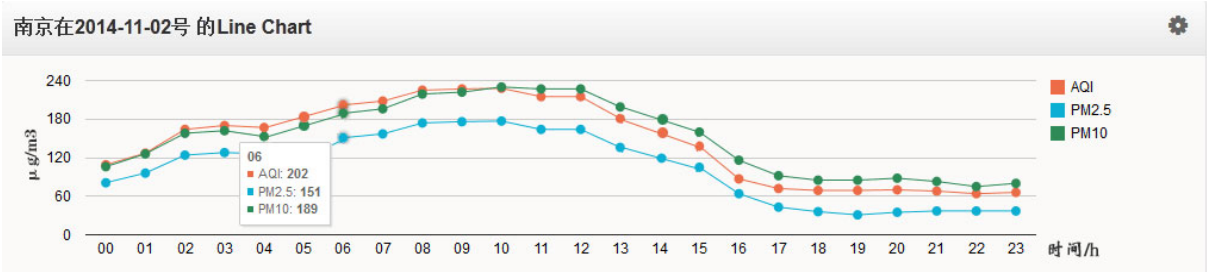

Figure 8. Air pollution data linear graph after interpolation
图 8. 插补之后空气污染数据线状图

5.3. WebGIS 展示

本系统将 OpenLayer 的 OSM 矢量图层地图作为地图底图，包括行政区划、河流、水库等要素，将气象站点作为专题图层来进行显示。输入网址到浏览器中，登录后就可以直接访问该系统。实现的功能主要有全国空气污染站点展示、空气污染详细信息展示、平均值和极值的求解、空气污染信息图表展示、气象信息图表展示、气象信息分析结果展示和等值线的实现等。系统实现的难点是如何使地图的空间数据和某种气象业务数据关联起来。也就是说，在地图上选择站点和须查询的气象数据类别，通过空间数据库得到站点号，然后通过站点号和类别在业务数据库中选择相应的数据[9]。

本系统的前台显示主要利用 OpenLayer 框架，采用 OpenLayers 作为客户端不存在浏览器依赖性。由于 OpenLayers 采用 JavaScript 语言实现，而应用于 Web 浏览器中的 DOM(文档对象模型)由 JavaScript 实现，同时，Web 浏览器(比如 IE，FF 等)都支持 DOM。OpenLayersAPIs 采用动态类型脚本语言 JavaScript 编写，实现了类似与 Ajax 功能的无刷新更新页面，能够带给用户丰富的桌面体验(它本身就有一个 Ajax 类，用于实现 Ajax 功能)。目前，OpenLayers 所能够支持的 Format 有：XML、GML、GeoJSON、GeoRSS、JSON、KML、WFS、WKT(Well-KnownText)。在 OpenLayers.Format 名称空间下的各个类里，实现了具体读写这些 Format 的解析器。

地图的操作包括放大、缩小、漫游和复位等，放大可以通过鼠标双击、滚轮、点击放大控件来实现，缩小可以通过滚轮和点击缩小控件来实现，漫游可以通过鼠标拖动来实现，复位可以通过点击复位按钮实现。

以带有经纬度坐标的大气污染数据为例，空气污染信息显示主要包括在地图上显示站点的位置和污染数据信息，站点图标大小根据 AQI 的大小分为五个不同等级(0~100、100~200、200~300、300~400、400~无穷大)，可参考图例；站点经纬度坐标根据服务器端通过查询数据库中的经纬度表获得，图 9 显示的是 2014 年 8 月 1 号 10 点的全国气象站点的图标位置，点击其中一图标后，比如北京市的图标，显示出北京市的污染数据信息(北京污染程度：五级 AQI：204 PM2.5：153 PM10：161 CO：1.508 SO_2：4 NO_2：50 O_3 臭氧一

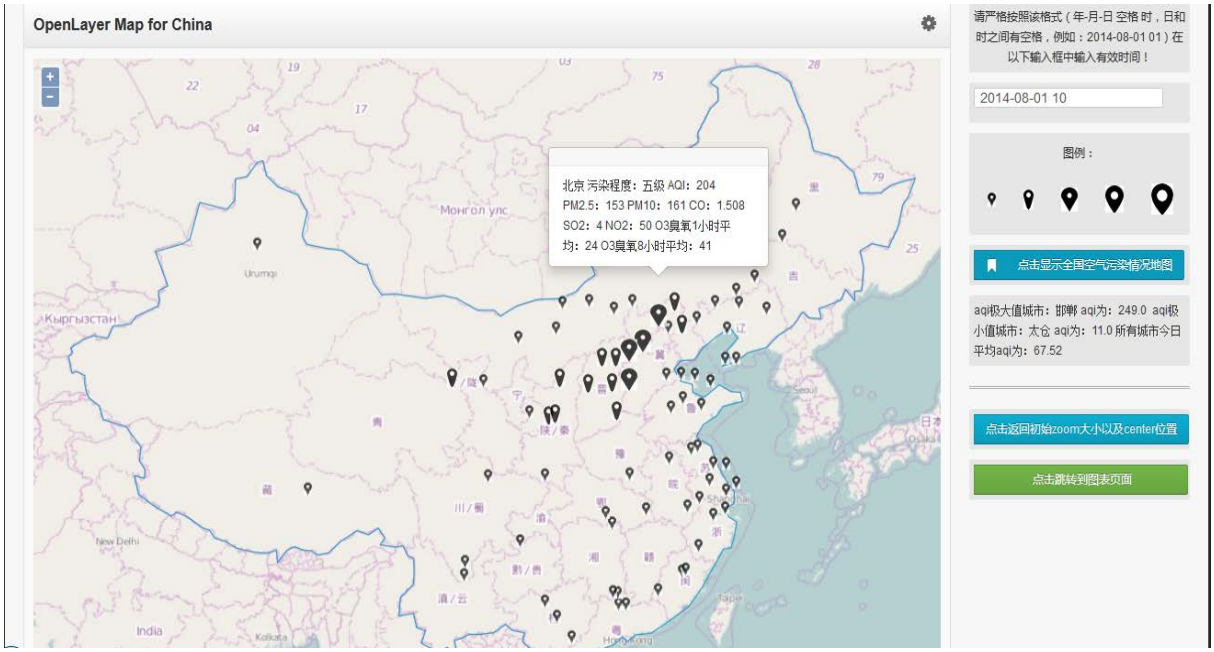

Figure 9. Air pollution site information display and the operator interface
图 9. 空气污染站点信息显示情况和操作界面

小时平均：24 O_3 臭氧 8 小时平均：41)，在显示出全国空气污染情况之后，还可以看到该时刻 AQI 极大值城市为邯郸，AQI 为 249.0，AQI 极小值城市为太仓，AQI 为 11.0，所有城市今日平均 AQI 为：67.52。用户可以输入时间，来查看不同时期的空气污染数据。用户还可以点击跳转到图表页面的按钮(在选中某一城市后才会显示出该按钮)，图表有线状图，表示的是当天 00 点到 23 点 AQI、PM2.5 和 PM10 的波动情况。

同样的，带有经纬度坐标的气象数据和等值线可以以同样的原理展示的 WebGIS 地图上，只不过是前端框架 OpenLayers 搭载的数据不同。

6. 结论

将数据仓库技术应用到气象领域，就可以从一个全新的角度将气象及大气污染资料进行分析处理，将各种数据经过清洁、抽取、变换、概括和聚集等操作，按气象及大气污染预报的需要进行数据重组和数据存储，提供全局的、统一的、语义一致的、组织良好的数据视图，在此基础上进行联机分析、数据挖掘等技术处理，发现各物理量和气象要素与某种天气现象之间的关系，减少重复的研究，提高效率[10]，前端以 WebGIS 技术支持，为用户提供气象以及大气污染数据快速查询、浏览、分析功能，用户通过与浏览器交互操作，就可以方便地寻找其所需的气象及污染数据，为气象服务保障事业提供更强有力的服务支持[11]。

本论文针对目前气象数据仓库系统存在的问题，结合气象资料本身的特点，着重从大型数据库的角度描述了气象及大气污染数据仓库的构建过程以及优化方法，使用数据仓库技术实现了对浩如烟海的气象数据资料的高效存储管理、ETL、数据挖掘和商业智能，还为气象及大气污染数据仓库的系统优化提供诸多建议，帮助系统优化加速，促进了气象资料的共享，提高对客户的服务速度，提升客户服务体验。

同时随着气象探测技术的不断发展，探测工具和探测方法都在不断创新，尤其是非常多的自动气象探测站点的投入使用，气象及大气污染数据量的增长日益剧增，实现一个能对海量气象资料实现自动化统一组织管理，并且能够将气象信息检索和气象信息共享服务整合到一起的气象数据仓库系统，对于气象业务的智能自动化有很大帮助。伴随 WebGIS 以及数据仓库技术在气象领域的广泛应用，气象及大气

污染数据仓库系统对我国的气象事业的贡献必将愈加巨大[12]。

参考文献 (References)

[1] 周展程, 孙志强. 建设地市级气象信息数据仓库的必要性和可行性[J]. 价值工程, 2011, 30(8): 163-164.

[2] 饶卫民, 章家恩, 肖红生, 等. 地理信息系统(GIS)在农业上的应用现状概述[J]. 云南地理环境研究, 2004, 16(2): 13-17.

[3] 王红霞, 王志伟, 王培红. 数据仓库在气象上的应用研究[J]. 电脑开发与应用, 2006, 19(7): 9-12.

[4] 赵文芳, 刘旭林, 聂凯. 基于 WebGIS 的气象综合显示系统改进与实现[J]. 应用气象学报, 2015, 26(3): 378-384.

[5] http://poi.apache.org/, POI 3. 13, 2015-09-29.

[6] http://jsoup.org/, jsoup 1. 8. 3, 2015.

[7] 王建华, 于鹏, 孙俊. 相似分析在空气连续监测缺测资料插补处理中的应用[J]. 中国环境监测, 2000, 16(3): 32-35.

[8] 李开乐. 相似离度及其应用技术[J]. 气象学报, 1986, 44(2): 176-183.

[9] 罗琦, 韩茜, 李文莉, 罗雪梅, 陈学君. 基于 WEBGIS 的气象科学数据查询显示系统的设计与实现[J]. 干旱气象, 2010, 28(4): 495-498.

[10] 李建东. 数据仓库技术在气象领域的应用前景分析[J]. 气象与环境科学, 2009, 32(z1): 174-176.

[11] 刘南, 刘仁义. Web GIS 原理及其应用[M]. 北京: 科学出版社, 2002.

[12] 王林, 等. WebGIS 技术在气象行业中的应用[J]. 科技广场, 2007(5): 195-196.