Программное обеспечение для работы с базой данных космических лучей*

А. А. Турпанов, С. А. Стародубцев
Институт космофизических исследований и аэрономии им. Ю.Г. Шафера СО РАН, Якутск, Россия
e-mail:turpanov@ikfia.ysn.ru

Разработана база данных, содержащая измерения, непрерывно проводимые в течение многих лет на станциях космических лучей "Якутск" и "Бухта Тикси". Предложены методы сетевого доступа к данным из программ, написанных на языке высокого уровня.

Ключевые слова: космические лучи, база данных, программное обеспечение.

Введение

Из межпланетного пространства на Землю постоянно падает поток заряженных частиц высоких энергий, называемых космическими лучами (КЛ). В основном ($> 95\,\%$) это протоны, приходящие из Галактики, но изредка регистрируются и КЛ солнечного происхождения.

Измерения показывают, что интенсивность КЛ изменяется в широких пределах как по величине, так и во времени. Многочисленные исследования приводят к выводу, что подобные вариации интенсивности КЛ чаще всего связаны с изменениями физических условий на Солнце, которые определяют космическую погоду в окрестности Земли. Поскольку межпланетные возмущения, причиной которых являются солнечные вспышки, выбросы корональной массы и т. д., до орбиты Земли доходят достаточно медленно (около двух-трех суток), а КЛ имеют скорость, сравнимую со световой, то в принципе возможен прогноз космической погоды по регистрации интенсивности КЛ в режиме реального времени. С этой целью в ИКФИА СО РАН была создана база данных (БД) "Межпланетная среда", которая доступна по адресу *http://www.ysn.ru/ipm* [1 – 4].

Использование СУБД в архитектуре клиент/сервер для хранения и обработки экспериментальных данных имеет значительные преимущества при создании автоматизированных систем сбора и обработки данных по сравнению с хранением данных в ASCII-файлах. Такой подход обеспечивает хорошую структурированность данных, централизованное обслуживание СУБД и гарантию ее поддержания в актуальном состоянии, так как пользователи не работают с локальными копиями данных, часто порождающими множественность версий. Системы управления реляционными базами данных предоставляют мощные высокопроизводительные средства для доступа к данным, кроме того архитектура клиент/сервер может использоваться для решения больших по объему вычислений задач в модели распределенных вычислений.

*Работа выполнена при финансовой поддержке программы "Информационно-телекоммуникационные ресурсы СО РАН", 2007 г.

Для интерактивной работы база данных "Межпланетная среда" имеет WWW-интерфейс. Для обеспечения доступа к данным из программ, написанных на языке Фортран, необходимым инструментом является библиотека подпрограмм, обеспечивающая выполнение основных классов запросов к БД и передачу результатов прямо в память вызывающей программы без применения дисковых операций. Разработанная авторами библиотека `libipmg77` является первым приближением к решению такой задачи. Эта библиотека включает три основных и две вспомогательных функции, написанных на языке C с использованием API PostgreSQL — `libpq`. Функции библиотеки могут вызываться из Фортран-программ с использованием компилятора `g77` на платформе Linux.

Ниже подробно описывается разработанное программное обеспечение.

1. Реализованные расширения SQL

Целью создания дополнительных функций является расширение набора агрегатных функций и разработка вспомогательных функций для округления переменных типа `timestamp`. Последнее обусловлено тем, что в различного рода прикладных задачах требуются данные с разным шагом дискретизации по времени. Встроенная в СУБД PostgreSQL функция `date_trunc (единица, дата—время)` имеет аргумент `дата—время` типа `timestamp` и параметр — одну из единиц `year, month, day` и т. д., до целого значения которой производится усечение аргумента.

При работе с данными часто возникает необходимость в аналогичной функции, дополняющей аргумент дата—время до некоторого округленного значения. Кроме того, в рассматриваемой области используется специфическая единица времени — оборот Бартелса — синодический период вращения Солнца (27 сут).

Поскольку запросы на вычисление средних по различным единицам времени выполняются часто, возникает необходимость индексирования поля дата–время по "круглым" значениям таких единиц. При создании индекса по значениям некоторой функции необходимо, чтобы эти значения полностью определялись значениями полей, являющихся аргументами функции, т. е. функция не должна содержать параметров. Встроенная функция `date_trunc (единица, дата—время)` этому условию не удовлетворяет, поэтому на ее основе был создан набор отдельных функций для различных единиц времени и видов округления.

1.1. Функции для усечения и округления даты—времени

Интерфейс. Функции для округления и усечения даты—времени могут непосредственно использоваться в SQL-запросах. Их имена и назначение приведены в таблице.

Аргументы. Аргумент этих функций должен иметь тип `timestamp`.

Единица времени	Усечение	Дополнение
5 мин	`min5_trunc(timestamp)`	`min5_compl(timestamp)`
1 ч	`hour_trunc(timestamp)`	`hour_compl(timestamp)`
1 сут.	`day_trunc(timestamp)`	`day_compl(timestamp)`
Один оборот Бартелса	`brn_trunc(timestamp)`	`brn_compl(timestamp)`
1 мес.	`month_trunc(timestamp)`	`month_compl(timestamp)`
1 год	`year_trunc(timestamp)`	`year_compl(timestamp)`

Описание. Функции выполнены путем переработки исходного кода встроенной в СУБД функции `date_trunc()`. В дополнение к стандартным единицам измерения времени в функциях `brn_trunc()` и `brn_compl()` реализована обработка специфической для данной предметной области единицы — солнечных суток по календарю Бартелса.

1.2. Агрегатные функции

В стандарте SQL-92 определены пять агрегатных функций — `AVG`, `MAX`, `MIN`, `SUM` и `COUNT`. В то же время в статистическом анализе данных широко используются два момента распределения случайной величины — дисперсия и стандартное отклонение. С точки зрения языка SQL эти величины являются агрегатными функциями, а значит, в принципе можно найти способ обращаться к ним непосредственно с помощью языка запросов.

В СУБД PostgreSQL предусмотрена возможность расширения набора агрегатных функций с помощью оператора `CREATE AGREGATE`. Этот оператор не входит в стандарт SQL-92, а является расширением, реализованным в PostgreSQL. Функции, объявленные с помощью оператора `CREATE AGREGATE`, в дальнейшем распознаются планировщиком и оптимизатором запросов как агрегатные и к ним применяются все соответствующие синтаксические правила языка SQL. В базе данных “Межпланетная среда” (`ipm`) для вычисления дисперсии и стандартного отклонения созданы две дополнительные агрегатные функции.

Интерфейс. `VAR(Arg)`, `SDEV(Arg)` — агрегатные функции для вычисления несмещенной оценки дисперсии и стандартного отклонения. Могут применяться в операторе `SELECT` так же, как стандартные агрегатные функции языка `SQL`.

Аргументы. Функции `VAR()` и `SDEV()` являются перегруженными и в качестве аргументов могут иметь поля таблиц или функции от них следующих типов: `int2`, `int4`, `float4`.

Возвращаемые значения. Тип возвращаемого результата для аргументов любого типа всегда один — `float4`. Если множество записей, на котором производится вычисление агрегата, пусто или состоит из одного элемента, то возвращается нулевое значение. Согласно стандарту SQL-92 агрегатные функции в этом случае должны возвращать пустые значения, но такое поведение не было реализовано из-за технических сложностей. Для реальных данных нулевое значение дисперсии практически не встречается, однако в целях предосторожности, чтобы распознавать не имеющие смысла значения, можно рекомендовать одновременно с дисперсией или стандартным отклонением запрашивать и счетчик количества использованных записей.

Описание. Техника программирования агрегатных функций подчинена определенным правилам, препятствующим созданию массивов данных, по которым вычисляется агрегат, или повторному обращению к элементу данных в ходе вычислений. Это ограничение не позволяет прямо использовать определения для вычисления дисперсии и стандартного отклонения, так как в этих функциях фигурируют средние значения случайной величины, сами являющиеся агрегатными функциями и неизвестные до окончания перебора всего множества записей. Для преодоления этой ситуации формулы преобразованы в подходящий для однопроходных вычислений вид. Следет отметить, что вычисления дисперсии по таким формулам приводят к быстрому накоплению ошибок округления, поэтому на промежуточных этапах использована удвоенная точность вычислений.

2. Библиотека пользователя

2.1. Простой запрос — функция ipmbas()

Основной потребностью пользователей, работающих с временными рядами экспериментальных данных, является получение ряда данных за определенный период времени. Для обеспечения такого доступа к данным из программ, написанных на Фортране, служит функция `ipmbas()`.

Интерфейс.

```
FUNCTION IPMBAS(LOGICAL Debug, INTEGER*4 nRows, CHARACTER FieldsList,
                CHARACTER Table, INTEGER*4 Abst1, INTEGER*4 Abst2,
                CHARACTER Cond, Array1, ...)
```

Функция `ipmbas()` производит выборку из одной таблицы `Table` данных или функций от них, перечисленных в списке `FieldsList`, удовлетворяющих условию `Cond` и принадлежащих отрезку абсолютного времени `[Abst1, Abst2]`. Под абсолютным временем понимается число секунд с 1970-01-01 00:00:00 UTC. Результаты возвращаются в бинарном виде в одномерные массивы `Array1, ...` размерности `nRows`, зарезервированные пользователем в своей программе.

Функция не имеет возможности проверить соответствие между количеством и типами запрошенных данных и количеством и типами массивов, указанных для размещения результатов. Пользователь с помощью справочной информации по базе данных должен обеспечить это соответствие, а также одинаковый порядок перечисления запрашиваемых данных в списке `FieldsList` и массивов `Array1, ...`, отведенных для возвращаемых значений. Поддерживаемые типы данных — `int2, int4, float4, float8`.

В режиме отладки, задаваемом с помощью аргумента `Debug`, функция выводит в стандартный поток ошибок текст сгенерированного SQL-запроса и другую отладочную нформацию.

Аргументы. `Debug` — логическая переменная или константа, управляющая режимом отладки.

`nRows` — целочисленная переменная или константа, равная фактической размерности массивов `Array1, ...`, отведенных для возвращаемых значений. Все массивы, используемые при обращении к функции, должны иметь одинаковую размерность.

`FieldsList` — символьная переменная или константа произвольной длины, содержащая разделенный запятыми список полей или простых (не агрегатных) функций от них, запрашиваемых пользователем. Содержательная часть строки должна быть ограничена `NULL`-символом (`0x00`). Информацию о названиях полей и встроенных фукциях можно найти в описании базы данных.

`Table` — символьная переменная или константа произвольной длины, содержащая имя таблицы базы данных, в которой выполняется поиск. Содержательная часть строки должна быть ограничена `NULL`-символом (`0x00`). Информацию о именах таблиц можно найти в описании базы данных.

`Abst1, Abst2` — целочисленные переменные, содержащие начальное и конечное значения абсолютного времени, ограничивающие (включительно) период времени, в котором выполняется поиск.

`Cond` — символьная переменная или константа произвольной длины, содержащая логическое выражение, которое используется в качестве дополнительного критерия для

отбора записей из заданного временного периода. Если дополнительного условия не требуется, в `Cond` следует поместить текст "`true`". Содержательная часть строки должна быть ограничена `NULL`-символом (`0x00`).

Возвращаемые значения.

`nRowsRet = IPMBAS()` — при нормальном завершении в качестве значения, связанного с именем, функция возвращает значение `INTEGER*4 nRowsRet`, равное числу записей, попавших в выборку, или 0, если таких записей нет. Это число может превышать объявленную пользователем размерность массивов `nRows`, что не приводит к возникновению ошибки. Пользователь может использовать функцию `min(nRows, nRowsRet)` для определения количества фактически полученных строк. В случае возникновения ошибки возвращаемое значение отрицательно и содержит код ошибки.

`Array1, ...` — в массивы `Array1, ...` помещается не более `nRows` фактически полученных строк результата упорядоченных по значению `abstime(dt)`. Данные разных полей помещаются в массивы в том же порядке, в котором поля перечислены в списке `FieldsList`. Проверка соответствия количества указанных массивов количеству запрошенных полей или объявленной и фактической размерностей массивов не производится (в силу невозможности реализовать такую проверку).

2.2. Выборка агрегатных значений — функция ipmagr()

В процессе работы с данными часто возникает необходимость получения агрегатных значений, например, средних, при совместном анализе данных из таблиц с разным шагом дискретизации, дисперсии и стандартного отклонения, при оценке ошибок измерений и т. д. Синтаксис запросов для получения агрегатных значений отличается от синтаксиса запросов для получения самих данных, поэтому для выполнения таких запросов создана специальная функция `ipmagr()` (agregate query).

Интерфейс.

```
FUNCTION IPMAGR(LOGICAL Debug, INTEGER*4 nRows, CHARACTER FieldsList,
                CHARACTER Table, INTEGER*4 Abst1, INTEGER*4 Abst2,
                CHARACTER Cond, INTEGER*4 Units, INTEGER*4 RoundType,
                Array1, ...)
```

Функция `ipmagr()` производит выборку из одной таблицы `Table` агрегатных функций от ее полей с агрегированием по временному интервалу, определяемому единицами времени `Units`. Агрегатные функции для всех полей кроме поля `dt` должны быть явно перечисленны в списке `FieldsList`. Вычисление агрегатных значений производится для записей, принадлежащих отрезку абсолютного времени `[Abst1, Abst2]` и удовлетворяющих (необязательному) условию `Cond`. Границы отрезка в зависимости от значения аргумента `RoundType` усекаются или округляются до ближайшего целого значения единиц `Units`. Под абсолютным временем понимается число секунд с 1970-01-01 00:00:00 UTC. Результаты возвращаются в бинарном виде в одномерные массивы `Array1, ...` размерности `nRows`, зарезервированные пользователем в своей программе.

Функция не имеет возможности проверить соответствие между количеством и типами запрошенных данных и количеством и типами массивов, указанных для размещения результатов. Пользователь с помощью справочной информации по базе данных должен

обеспечить это соответствие, а также одинаковый порядок перечисления запрашиваемых данных в списке FieldsList и массивов Array1, ..., отведенных для возвращаемых значений. Поддерживаемые типы данных — int2, int4, float4, float8.

В режиме отладки, задаваемом с помощью аргумента Debug, функция выводит в стандартный поток ошибок текст сгенерированного SQL-запроса и другую отладочную нформацию.

Аргументы.

Debug, nRows, Array1, ..., FieldsList, Table, Abst1, Abst2, Cond — имеют тот же смысл, что и одноименные аргументы функции ipmbas().

Units — целочисленная переменная или константа, определяющая единицы времени, по "круглым" значениям которых производится группировка записей при агрегировании:

Units = 300 — 5 мин;

Units = 60 — 1 ч;

Units = 24 — 1 сут.;

Units = 27 — один оборот Солнца по календарю Бартелса;

Units = 30 — один календарный мес.;

Units = 365 — один календарный год.

Значения констант, определяющих временные единицы, выбраны равными соотношению "соседних" единиц только из мнемонических соображений и никакого другого смысла не имеют.

RoundType — целочисленная переменная или константа с допустимыми значениями 0 или 1, определяющая, будут ли агрегатные значения отнесены к началу или к концу временного интервала, по которому производилось агрегирование. RoundType = 0 соответствует привязке к началу интервала, RoundType = 1 — к концу.

Возвращаемые значения.

nRowsRet = IPMAGR() — при нормальном завершении в качестве значения, связанного с именем, функция возвращает значение INTEGER*4 nRowsRet, равное числу агрегатных значений, попавших в выборку, или 0, если выборка пуста. Это число может превышать объявленную пользователем размерность массивов nRows, что не приводит к возникновению ошибки. Пользователь может использовать функцию min(nRows, nRowsRet) для определения количества фактически полученных строк. В случае возникновения ошибки возвращаемое значение отрицательно и содержит код ошибки.

Функция IPMAGR() автоматически не получает информации о числе записей, использованных при вычислении агрегатных значений. Если такая информация необходима, то в список FieldsList следует включить функцию COUNT(*). Это особенно важно при использовании агрегатной функции FAVG(), производящей "плавающее" усреднение целочисленных полей, так как она возвращает 0.0 при отсутствии записей в интервале агрегирования.

Array1, ... – в массивы Array1, ... помещается не более nRows фактически полученных строк результата, упорядоченных по значению abstime(dt). Полученные данные помещаются в массивы в том же порядке, в котором агрегатные функции перечислены в списке FieldsList. Проверка соответствия количества указанных массивов количеству запрошенных функций или соответствия объявленной и фактической размерностей массивов не производится (в силу невозможности реализовать такую проверку).

2.3. Поиск непрерывного по заданному условию сегмента данных — функция ipmseg()

При анализе временных рядов часто возникает необходимость выделения непрерывных отрезков данных, удовлетворяющих некоторому заданному условию. Однако в реляционных СУБД получение такой выборки из набора данных, которым представлен временной ряд, не может быть выполнено с помощью единственного запроса. Построению такого запроса препятствуют две причины.

1. В реляционной модели данных не существует понятия непрерывности набора данных, так как понятие "соседний элемент данных" предполагает возможность введения понятия расстояния, а вместе с ним и порядка, что противоречит требованию неупорядоченности сверху вниз кортежей в отношении. На языке таблиц это означает, что упорядоченность записей не может быть свойством таблицы. Упорядоченность, а вместе с ней и понятие соседнего элемента данных выборки реализуются в результате выполнения операции `SELECT` с использованием раздела `ORDER BY`, применяемой непосредственно или при создании курсора.

2. Даже если бы мы, зная структуру данных, сумели определить понятие *запись является соседней по отношению к данному множеству записей*, включение очередной записи в выборку потребовало бы проверки того, что данная запись является соседней *по отношению к уже отобранным*. Такая проверка породила бы неразрешимую рекурсию — в разделе `WHERE` оператора `SELECT` пришлось бы применять ссылку на множество уже отобранных записей, которое в начальный момент пусто.

При решении рассматриваемой задачи для выполнения выборок описанного типа в библиотеку пользователя включена функция `ipmint()`, реализующая последовательность из трех связанных запросов.

Интерфейс.

```
FUNCTION IPMINT(LOGICAL Debug, INTEGER*4 Abst1, INTEGER*4 Abst2,
                CHARACTER Table, CHARACTER Cond)
```

Функция для нахождения границ *первого непрерывного отрезка данных* таблицы `Table`, все записи которого удовлетворяют заданному пользователем условию `Cond` и принадлежат указанному пользователем периоду абсолютного времени `[Abst1, Abst2]`. Под непрерывным отрезком данных понимается такое подмножество записей таблицы `Table`, которое, будучи упорядоченным по времени, образует арифметическую прогрессию с шагом, равным шагу дискретизации временного ряда, представленного в таблице. Под абсолютным временем понимается число секунд с 1970-01-01 00:00:00 UTC.

Аргументы.

`Debug` — логическая переменная или константа, управляющая режимом отладки. Если `Debug = .TRUE.`, то в стандартный поток ошибок выводятся тексты и результаты выполняемых запросов.

`Abst1, Abst2` — целочисленные переменные, содержащие начальное и конечное значения абсолютного времени, ограничивающие (включительно) период времени, в котором выполняется поиск.

`Table` — символьная переменная или константа произвольной длины, содержащая имя таблицы базы данных "Межпланетная среда", в которой выполняется поиск. Содержательная часть строки должна быть ограничена `NULL`-символом (`0x00`).

`Cond` — символьная переменная или константа произвольной длины, содержащая логическое выражение, используемое в качестве критерия для поиска отрезка из заданно-

го периода. В отрезок включаются те записи, для которых данное логическое выражение принимает значение `TRUE`. Содержательная часть строки должна быть ограничена `NULL`-символом (`0x00`).

Возвращаемые значения.

`IPMINT` — при нормальном завершении в качестве значения, связанного с именем, функция возвращает значение типа `INTEGER*4`, равное числу записей в первом найденном отрезке, или 0, если таких записей нет. В случае возникновения ошибки возвращаемое значение отрицательно и содержит код ошибки.

`Abst1, Abst2` — если искомое множество записей существует, то в аргументы `Abst1` и `Abst2` помещаются начальное и конечное значения абсолютного времени для найденного множества записей; если это множество состоит из одной записи, значения `Abst1` и `Abst2` будут совпадать; если записей не найдено, значения `Abst1` и `Abst2` не изменяются.

Таким образом, разработанное дополнительное программное обеспечение позволяет существенно ускорить процесс обработки и анализа данных регистрации космических лучей.

Список литературы

[1] Turpanov A.A., Starodubtsev S.A., Grigoryev V.G. et al. The automatized system for the collection, treatment and analysis of neutron monitor data in real-time // Proc. 27-th ICRC. Hamburg, Germany, 2001. P. 2325–2328.

[2] Стародубцев С.А., Турпанов А.А., Турпанов В.А. и др. Автоматизированная система прогноза космической погоды по данным нейтронных мониторов в режиме реального времени // Тр. конф. по физике солнечно-земных связей. Солнечно-земная физика. Иркутск: ИСЗФ СО РАН, 2002. С. 86–88.

[3] Kozlov V., Ksenofontov L., Kudela K. et al. Real-time COsmic ray database (RECORD) // Proc. 28-th ICRC. Tsukuba, Japan, 2003. P. 3473–3476.

[4] Starodubtsev S., Turpanov A., Kudela K. et al. Real-time cosmic ray distributed (RECORD) database: A status report // Proc. 29-th ICRC. Pune, India, 2005. P. 465–468.

Поступила в редакцию 21 января 2008 г.