УДК 621.311

О.С.ЯНДУЛЬСЬКИЙ, **Д.В.Н**АСТЕНКО, **О.О.Д**МИТРЕНКО, **О.В.Т**ИМОХІН, **А.О.С**ТЕЛЮК

СТАТИСТИЧНИЙ АНАЛІЗ ДИСКРЕТНОЇ ІНФОРМАЦІЇ СИСТЕМ КЕРУВАННЯ ЕНЕРГООБ'ЄКТАМИ ЗА УМОВ СПОРАДИЧНОСТІ ВХІДНИХ ДАНИХ

O.YANDULSKII, **D.N**ASTENKO, **O.D**MYTRENKO, **O.T**YMOKHIN, **A.S**TELIUK

STATISTICAL ANALYSIS OF DISCRETE INFORMATION OF THE POWER EQUIPMENT CONTROL SYSTEMS IN THE EVENT OF SPORADICAL INPUT DATA

Анотація. Розглянуто різноманітні підходи до збереження інформації в автоматизованих системах в енергетиці та застосування сучасних систем управління базами даних. Доведено, що при використанні стандартних шляхів з застосуванням SQL-запитів для періодичної інформації, при наявності спорадичних даних, результати розрахунків статистичних показників незадовільні. Наведено алгоритм, що дозволяє перейти від статистичного аналізу періодичної інформації до статистичного аналізу спорадичних даних, які зберігаються в системі керування базами даних Oracle. Алгоритм дозволяє провести аналіз без втрати інформації.

Ключові слова: статистичний аналіз, спорадичні виміри, автоматизовані системи керування в електроенергетиці, системи керування базами даних.

Аннотация. Рассмотрены различные подходы сохранения информации в автоматизированных системах в энергетике и применение современных систем управления базами данных. Показано, что при использовании стандартных подходов с SQL-запросами для периодической информации, при наличии спорадических данных, результаты расчетов статистических показателей являются неудовлетворительными. Приведен алгоритм, позволяющий перейти от статистического анализа периодической информации к статистическому анализу спорадических данных, которые хранятся в системе управления базами данных Oracle. Алгоритм позволяет провести анализ без потери информации.

Ключевые слова: статистический анализ, спорадические измерения, автоматизированные системы управления в электроэнергетике, системы управления базами данных.

Annotation. The different approaches of saving information of automatic control systems in power systems and the usage of modern database management systems have been considered. It is shown that the use of standard approaches to the SQL-queries to periodic information, in the presence of sporadic data, the calculations of statistical indicators are unsatisfactory. An algorithm of transition from the statistical analysis of the periodic information to the statistical analysis of sporadic data which are stored in a database management system Oracle is proposed. The algorithm allows analyzing data without losing information.

Key words: statistical analysis, sporadical measurements, automatic control systems in electrical systems, data managers.

Вступ

Інтелектуалізація електроенергетичних систем та мереж неможлива без зберігання, обробки та аналізу значних масивів інформації, отриманих від пристроїв різного призначення – вимірювальних, релейного захисту та автоматики, лічильників електроенергії тощо. Для виконання даної задачі використовуються автоматичні та автоматизовані системи моніторингу та керування, зокрема, оперативно-інформаційні комплекси, системи диспетчерського керування та збору даних, різноманітні системи збору. Інформація, яка отримується та обробляється, може бути як періодичною (з однаковими або різними проміжками дискретизації), так і спорадичною (внаслідок виникнення події або при зміні значення контрольованої величини). Отримані дані повинні оброблятися і використовуватись для рішення різноманітних задач з контролю, прогнозування та керування. Наприклад, для ефективного керування енергосистемою в цілому і її окремими частинами постає питання аналізу даних про функціонування енергосистеми та її

окремих частин, який виконується на основі інформації, що збирається автоматичною системою керування технологічними процесами (АСК ТП).

Інформація, яка передається в АСК ТП, у більшості випадків має значний обсяг, причому при аналізі часто використовуються дані, які накопичені за певний період часу. Тому для зручного та швидкого доступу до такої інформації повинно бути організовано середовище для зберігання даних, що задовольняє наступним вимогам:

- надійне збереження інформації;
- інформаційна безпека;
- надання доступного для багатьох користувачів інтерфейсу з метою обробки та відображення цієї інформації;
- оптимізація доступу до інформації (пошук, сортування тощо);
- швидкодія.

Існують різні підходи до зберігання інформації. Перший – зберігання в оперативній пам'яті (тобто після перезавантаження серверу інформація назавжди втрачається). Такий підхід, хоч і відзначається досить високою швидкодією, та не забезпечує достатню надійність. Другий метод полягає у зберіганні тільки частини інформації у файлах, наприклад, текстових, власного формату і т.ін. Перегляд цієї інформації, пошук, сортування необхідно здійснювати вручну. Даний спосіб використовується у багатьох системах, але відзначається низькою швидкодією, ускладненими методами організації пошуку та сортування. Третій метод – це зберігання інформації в сучасних високошвидкісних системах керування базами даних (СКБД), які забезпечують виконання вищенаведених вимог.

Як приклад роботи з середовищем зберігання інформації розглянемо базу даних (БД), що зберігає значення частоти струму ОЕС України в одній з найбільш поширених СКБД – Oracle. Значення частоти записуються в базу даних з відміткою часу. Вказана інформація отримана від датчиків частоти і в загальному вигляді є сукупністю спорадичних даних, які зберігаються у таблиці вимірів (наприклад Measurements) з атрибутами (або полями чи стовпчиками):

- тип виміру (наприклад id_measurement_type);
- час виміру (наприклад time);
- значення виміру (наприклад value).

Спрощена структура такої таблиці у вигляді SQL-запиту може мати наступний вигляд:

```
CREATE TABLE Measurements
(
  id_measurement_type  NUMBER,
  time                 DATE,
  value                NUMBER,
  CONSTRAINT Measurements_PK PRIMARY KEY (id_measurement_type, time),
  CONSTRAINT Meas_Measurement_Type_FK
    FOREIGN KEY (id_measurement_type)
      REFERENCES Measurement_Type (id_measurement_type)
 );
```

Причому слід зазначити:

- первісним ключем таблиці виступатиме пара полів **id_measurement_type, time**;
- поле **id_measurement_type** являє собою зовнішній ключ відносно до таблиці **Measurement_Type,** в якій зберігається інформація про типи вимірів;
- тип поля **time – DATE** було обрано для зберігання інформації про відмітки часу з точністю до секунди. Для збереження часу у більш точному форматі може використовуватися тип даних **TIMESTAMP** (точність до 10e-9 секунди). Для того ж, щоб додатково включати інформацію про часовий пояс, слід використовувати тип даних **TIMESTAMP WITH TIME ZONE**.

Побудова SQL- запитів для статистичного аналізу збережених даних у випадку періодичних даних, які зберігаються через рівні проміжки часу, не викликає труднощів. Для цього в більшості реляційних СКБД використовуються агрегатні функції – мінімум, максимум, середньоквадратичне

відхилення і т.ін. Наприклад, для знаходження середнього значення частоти (типу виміру з кодом 1024), що зчитується з періодом в 1 секунду, на інтервалах в 5 секунд (див. табл. 1), можна використати SQL- запит, наведений нижче.

Таблиця 1

Вхідні дані у випадку періодичного запису

id_measurement_type	time	value
1024	11.03.2011 14:00:00	49,978
1024	11.03.2011 14:00:01	49,985
1024	11.03.2011 14:00:02	49,985
1024	11.03.2011 14:00:03	50,000
1024	11.03.2011 14:00:04	50,012
1024	11.03.2011 14:00:05	50,012
1024	11.03.2011 14:00:06	50,012
1024	11.03.2011 14:00:07	50,012
1024	11.03.2011 14:00:08	50,012
1024	11.03.2011 14:00:09	50,012
1024	11.03.2011 14:00:10	50,007
1024	11.03.2011 14:00:11	49,999
1024	11.03.2011 14:00:12	49,991
1024	11.03.2011 14:00:13	49,991
1024	11.03.2011 14:00:14	49,991

Запит 1:

```
SELECT
    TRUNC(time, 'MI') + TRUNC (TO_CHAR (time, 'SS') / 5) / 17280 AS
"Час",
    COUNT(*) AS "Кількість",
    AVG(value) AS "Середнє"
FROM Measurements
WHERE id_measurement_type = 1024
GROUP BY TRUNC(time, 'MI') + TRUNC (TO_CHAR (time, 'SS') / 5) / 17280;
```

Результат виконання запиту наведено в табл. 2.

Таблиця 2

Результат SQL-запиту 1 для періодичних даних

Час	Кількість	Середнє
11.03.2011 14:00:00	5	49,9920
11.03.2011 14:00:05	5	50,0120
11.03.2011 14:00:10	5	49,9958

Більш складним виявляється випадок, коли для економії дискового простору та мережевого трафіку дані записуються в базу даних тільки у разі зміни значення (по спорадиці). В цьому випадку кількість вимірів на однакових проміжках часу може різнитися, а на деяких інтервалах може бути відсутньою. Використання попереднього запиту для цієї моделі призведе до негативного результату.

Вихідні дані для випадку спорадичного запису наведені у табл. 3 і за змістом не відрізняються від наведених у табл. 1.

Таблиця 3

Вхідні дані у випадку спорадичного запису

id_measurement_type	time	value
1024	11.03.2011 14:00:00	49,978
1024	11.03.2011 14:00:01	49,985
1024	11.03.2011 14:00:03	50,000
1024	11.03.2011 14:00:04	50,012
1024	11.03.2011 14:00:10	50,007
1024	11.03.2011 14:00:11	49,999
1024	11.03.2011 14:00:12	49,991

Результат, отриманий після використання Запиту 1, наведено в табл. 4.

Таблиця 4

Результат SQL-запиту 1 для спорадичних даних

Час	Кількість	Середне
11.03.2011 14:00:00	4	49,99375
11.03.2011 14:00:10	3	49,99900

Отриманий результат свідчить про те, що для випадку спорадичних даних використання стандартних агрегатних функцій напряму не є вірним, тобто втрачено значення з міткою часу «11.03.2011 14:00:05», а для двох інших міток обчислені значення відрізняються від попередніх. Тому для використання стандартних агрегатних функцій потрібно розв'язати дві задачі:

- задача 1. Мати значення виміру на початку кожного інтервалу вимірювання (в нашому випадку відновити відсутній запис для «11.03.2011 14:00:05»);
- задача 2. Враховувати нерівномірність записів вимірів, а саме: якщо відсутній періодичний запис – актуальним залишається попереднє значення спорадичного виміру.

Для вирішення даних завдань авторами запропоновано наступний спосіб, який розглянуто нижче покроково.

Крок 1. Генерація відсутніх міток часу. Для цього сформовано SQL-запит, що створює повний набір часових відміток з заданим інтервалом (5 секунд) у заданому діапазоні.

Запит 2:

```
SELECT
      (ROWNUM-1) * INTERVAL '5' SECOND +
            TO_DATE('11.03.2011 14:00:00','DD.MM.RRRR HH24:MI:SS') time
FROM Dual
CONNECT BY LEVEL <4;
```

Результат, отриманий після використання Запиту 2, наведено в табл. 5.

Таблиця 5

Результат SQL-запиту 2

time
11.03.2011 14:00:00
11.03.2011 14:00:05
11.03.2011 14:00:10

Крок 2. Об'єднання отриманої на попередньому кроці таблиці з даними вимірів з табл. 3 за допомогою лівого з'єднання (left inner join), що дозволяє отримати записи, які містять час та значення вимірів для всіх часових відміток, які кратні заданому інтервалу.

Запит 3:

```
SELECT T.time, F.value
FROM
    (
    SELECT
        (ROWNUM-1) * INTERVAL '5' SECOND +
            TO_DATE('11.03.2011 14:00:00','DD.MM.RRRR HH24:MI:SS')
time
    FROM Dual
    CONNECT BY LEVEL <4
    ) T,
    (
    SELECT time, value
    FROM Measurements m
    WHERE id_measurement_type = 1024
    ) F
WHERE T.time = F.time(+);
```

Результат, отриманий після використання Запиту 3, наведено в табл. 6.

Таблиця 6

Результат після виконання Запиту 3

Time	value
11.03.2011 14:00:00	49,978
11.03.2011 14:00:05	
11.03.2011 14:00:10	50,007

Крок 3. Об'єднання даних табл. 6 з початковими, наведеними в табл. 3, виконується за допомогою оператора UNION:

Запит 4:

```
SELECT *
FROM
    (
    SELECT T.time, F.value
    FROM
        (
        SELECT
            (ROWNUM-1) * INTERVAL '5' SECOND +
                TO_DATE('11.03.2011 14:00:00','DD.MM.RRRR HH24:MI:SS')
time
        FROM Dual
        CONNECT BY LEVEL <4
        ) T,
        (
        SELECT time, value
        FROM Measurements
        WHERE id_measurement_type = 1024
        ) F
    WHERE
        T.time = F.time(+)
    )
UNION
    (
    SELECT time, value
    FROM Measurements
    WHERE id_measurement_type = 1024
    );
```

Результат, отриманий після використання Запиту 4, наведено в табл. 7.

Таблиця 7

Результат після виконання Запиту 4

time	value
11.03.2011 14:00:00	49,978
11.03.2011 14:00:01	49,985
11.03.2011 14:00:03	50,000
11.03.2011 14:00:04	50,012
11.03.2011 14:00:05	
11.03.2011 14:00:10	50,007
11.03.2011 14:00:11	49,999
11.03.2011 14:00:12	49,991

Як видно з наведеного результату, потрібний запис з'явився в сформованій таблиці без заповненого поля **value**.

Крок 4. Заповнення отриманих на попередньому етапі пустих полів значенням попереднього спорадичного виміру. Для цього за допомогою аналітичної функції SUM(x) та скалярної функції NVL2(x,y,z) в одну групу об'єднуються записи, що мають значення, та всі наступні, які значень не мають.

Запит 5:

```
SELECT
    D.time,
    D.value,
    SUM(NVL2(D.value,1,0)) OVER (ORDER BY D.time) GR
FROM (
    SELECT T.time, F.value
    FROM
        (
        SELECT
            (ROWNUM-1) * INTERVAL '5' SECOND +
                TO_DATE('11.03.2011 14:00:00','DD.MM.RRRR HH24:MI:SS')
time
        FROM Dual
        CONNECT BY LEVEL <4
        ) T,
        (
        SELECT time, value
        FROM Measurements
        WHERE id_measurement_type = 1024
        ) F
    WHERE
        T.time = F.time(+)
    UNION
    SELECT time, value
    FROM Measurements
    WHERE id_measurement_type = 1024
    ) D;
```

Результат, отриманий після використання Запиту 5, наведено в табл. 8.

Таблиця 8

Результат після виконання Запиту 5

time	value	GR
11.03.2011 14:00:00	49,978	1
11.03.2011 14:00:01	49,985	2
11.03.2011 14:00:03	50,000	3
11.03.2011 14:00:04	50,012	4
11.03.2011 14:00:05		4
11.03.2011 14:00:10	50,007	5
11.03.2011 14:00:11	49,999	6
11.03.2011 14:00:12	49,991	7

В результаті використання запиту 5 досягається об'єднання в одну групу 4 попередніх спорадичних та згенерованого періодичного записів.

Крок 5. За допомогою іншої аналітичної функції MAX(x) можна поширити існуюче значення на всю групу, а за допомогою аналітичної функції LEAD(x,y,z) обчислити тривалість інтервалів, на яких значення вимірів залишаються незмінними (поле «period»). Зазначимо, що значенням третього аргументу функції LEAD(x,y,z) потрібно взяти кінцевий час аналізу тих вимірювань, які розглядаються в даному прикладі.

Запит 6:

```
SELECT
    time,
    MAX(value) OVER (PARTITION BY GR) value,
    ROUND(
      (LEAD(time, 1, TO_DATE('11.03.2011 14:00:15','DD.MM.RRRR
HH24:MI:SS'))
        OVER (ORDER BY time) - time)*60*24*60) period
FROM
    (SELECT
        D.time, D.value,
        SUM(NVL2(D.value,1,0)) OVER (ORDER BY D.time) GR
    FROM (
        SELECT T.time, F.value
        FROM
            (
            SELECT
                (ROWNUM-1) * INTERVAL '5' SECOND +
                 TO_DATE('11.03.2011 14:00:00','DD.MM.RRRR
HH24:MI:SS') time
            FROM Dual
            CONNECT BY LEVEL < 4
            ) T,
            (
            SELECT time, value
            FROM Measurements
            WHERE id_measurement_type = 1024
            ) F
        WHERE
            T.time = F.time(+)
        UNION
        SELECT time, value
        FROM Measurements m
        WHERE id_measurement_type = 1024
        ) D
```

```
)A;
```

Результат, отриманий після використання Запиту 6, наведено в табл. 9.

Таблиця 9

Результат після виконання Запиту 6

time	value	period
11.03.2011 14:00:00	49,978	1
11.03.2011 14:00:01	49,985	2
11.03.2011 14:00:03	50,000	1
11.03.2011 14:00:04	50,012	1
11.03.2011 14:00:05	50,012	5
11.03.2011 14:00:10	50,007	1
11.03.2011 14:00:11	49,999	1
11.03.2011 14:00:12	49,991	3

Крок 6. Використовуючи значення поля period як ваговий коефіцієнт, можна виконувати над цими результатами довільні статистичні розрахунки за допомогою вбудованих агрегатних функцій. Наприклад, для отримання результатів, ідентичних наведеним на початку, можна виконати запит 7:

```
SELECT
    TRUNC(time, 'MI') + TRUNC (TO_CHAR (time, 'SS') / 5) / 17280 AS
"Час",
    SUM(period) AS "Кількість",
    SUM(period*value)/SUM(period) AS "Середнє"
FROM
(SELECT
    time,
    MAX(value) OVER (PARTITION BY GR) value,
    ROUND(
      (LEAD(time, 1, TO_DATE('11.03.2011 14:00:15','DD.MM.RRRR
HH24:MI:SS'))
      OVER (ORDER BY time) - time)*60*24*60) period
FROM
    (SELECT
        D.time, D.value,
        SUM(NVL2(D.value,1,0)) OVER (ORDER BY D.time) GR
    FROM (
        SELECT T.time, F.value
        FROM
            (
            SELECT
              (ROWNUM-1) * INTERVAL '5' SECOND +
                TO_DATE('11.03.2011 14:00:00','DD.MM.RRRR HH24:MI:SS')
time
            FROM Dual
            CONNECT BY LEVEL <4
            ) T,
            (
            SELECT time, value
            FROM Measurements
            WHERE id_measurement_type = 1024
            ) F
        WHERE
            T.time = F.time(+)
```

```
        UNION
        SELECT time, value
        FROM Measurements
        WHERE id_measurement_type = 1024
        ) D
    )A
)O
GROUP BY TRUNC(time, 'MI') + TRUNC (TO_CHAR (time, 'SS') / 5) / 17280;
```

Результат, отриманий після використання Запиту 7, наведено в табл. 10.

Таблиця 10

Результат після виконання Запиту 7

Час	Кількість	Середнє
11.03.2011 14:00:00	5	49,9920
11.03.2011 14:00:05	5	50,0120
11.03.2011 14:00:10	5	49,9958

Легко помітити збіг отриманого результату (табл. 10) із початковим (табл. 2), що підтверджує правильність обраного алгоритму.

Висновки

Розглянутий підхід дозволяє:

- розширити область застосування агрегатних функцій від обчислень вимірів з дискретними періодичними часовими відмітками до обчислень спорадичних вимірів з довільними неперіодичними часовими відмітками, що дозволяє виконати аналіз інформації без втрати значущих даних;
- виконувати статистичну обробку довільної спорадичної інформації з прийнятною швидкістю.

Література

1. Кайт Том. Oracle для профессионалов. [Текст] Пер. с англ. – СПб.: ООО ДиаСофтЮП, 2003. — 672 с. ISBN 5–93772–072–5.
2. Oracle Database SQL Language Reference, 11g Release 2 (11.2) E17118–03 Copyright © 1996, 2010, Oracle and/or its affiliates. Primary Authors: Diana Lorentz, Mary Beth Roeser. [електронний документ у форматі PDF].
3. ГОСТ 24.104–85. Автоматизированные системы управления. Общие требования.