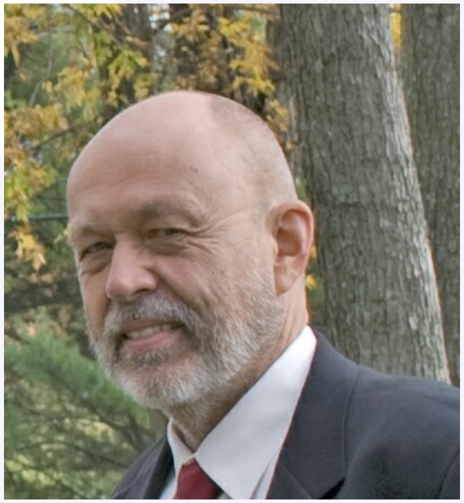
Advanced DataTools Webcast

Informix OLAP and Advanced SQL Functions by Lester Knutsen

March 7, 2019 at 2:00pm EDT

Advanced DataTools

Lester Knutsen

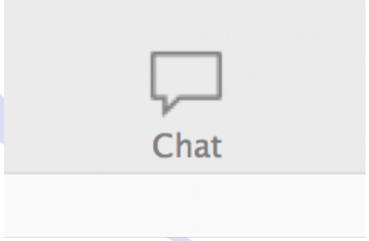


Lester Knutsen is President of Advanced DataTools Corporation, and has been building large data warehouse and business systems using Informix Database software since 1983. Lester focuses on large database performance tuning, training, and consulting. Lester is a member of the IBM Gold Consultant program and was presented with one of the Inaugural IBM Information Champion awards by IBM. Lester was one of the founders of the International Informix Users Group and the Washington Area Informix User Group.

lester@advancedatatools.com
www.advancedatatools.com
703-256-0267 x102

Webcast Guidelines

- The Webcast is being recorded. The Webcast replay and slides may be available in a few days.
- Please Mute your Line. Background sounds will distract everyone.
- Use the Chat Button in the upper right to ask questions.



Agenda

- Advanced SQL Functions
  - Aggregate Functions
  - Date Functions
- SQL OLAP Functions
  - OLAP Window Frame
  - OLAP Aggregate Functions

SQL Functions

- Aggregate Functions
  - AVG, COUNT, MIN, MAX, RANGE, STDEV, SUM, VARIANCE
- Time Functions
  - DAY, CURRENT, MONTH, WEEKDAY, YEAR, DATETIME

SQL Aggregate Functions

- Count(*) - returns the number of rows that satisfy the WHERE clause
- Count(column) – returns the total number of non-NULL values in the specified column
- Count(distinct column) - Returns the number of unique non-NULL values in the specified column
- Count(unique column)

SQL Aggregate Functions

- Max (column) - returns the highest value
- Min (column) - returns the lowest value
- Sum (column) - returns the sum of values
- Avg (column) – average of all values
- Stdev (column) – standard deviation
- Variance (column) – estimate of the population variance
- Range (column) - range of values

  range(expr) = max(expr) - min(expr);

SQL Aggregate Functions

```
Select      count(*) as record_count,
        count( city ) as city_count,
        count( unique city ) as unique_city_count,
        count( distinct city ) as distinct_city_count,
        count( unique state) as unique_state,
        count( unique zip ) as unique_zip,
        max(start_date) as max_start_date,
        min(start_date) as min_start_date,
        count( unique year(start_date)) as count_unique_year_start_date,
        count( unique quarter(start_date)) as count_unique_quarter_start_date,
        count( unique month(start_date)) as count_unique_month_start_date,
        sum( total_bill) as sum_total_bill,
        avg( total_bill) as avg_total_bill,
        stdev( total_bill) as stdev_total_bill,
        variance( total_bill) as variance_total_bill,
        max( total_bill) as max_total_bill,
        min( total_bill) as min_total_bill
from bills
```

Output

```
record_count        605280
city_count          605280
unique_city_count   18751
distinct_city_cou+  18751
unique_state        52
unique_zip          41861
max_start_date      06/29/2009
min_start_date      01/01/2007
count_unique_year+  3
count_unique_quar+  4
count_unique_mont+  12
sum_total_bill      37263400.92
avg_total_bill      61.5639058287074
stdev_total_bill    33.2759339232192
variance_total_bi+  1107.28777846245
max_total_bill      104.64
min_total_bill      3.03
```

SQL Date Functions

- Year – year of date
- Quarter - quarter of date
- Month – month number of date
- Weekday – weekday number of date
- Last_day – last day of month
- Months_between – number between two dates
- Day - number of a day of the month
- Date – converts to a date
- TO_CHAR – converts date to character
- TO_DATE – converts character to date
- Today – todays date

SQL Date Functions

```
select
        year(start_date) as f_year,
        quarter(start_date) as f_quarter,
        month(start_date) as f_month,
        weekday(start_date) as f_weekday,
        months_between ( max (start_date) , min (start_date)) as f_months_between,
        count(*) as record_count,
        sum(total_bill) as sum_total_bill
from bills
group by 1, 2, 3, 4
order by 1, 2, 3, 4
```

Output

```
f_year           2007
f_quarter          1
f_month            1
f_weekday          0
f_months_between  0.67741935483871
record_count      2688
sum_total_bill    162580.84

f_year           2007
f_quarter          1
f_month            1
f_weekday          1
f_months_between  0.90322580645161
record_count      3456
sum_total_bill    209143.72
```

SQL Date Functions

```
select
        bill_number,
        start_date,
        year(start_date) as f_year,
        quarter(start_date) as f_quarter,
        month(start_date) as f_month,
        weekday(start_date) as f_weekday,
        last_day(start_date) as f_last_day,
        to_char ( start_date ) as f_to_char,
        today as f_today,
        months_between ( today , start_date ) as f_months_between,
        total_bill
from bills
where bill_number = 1
```

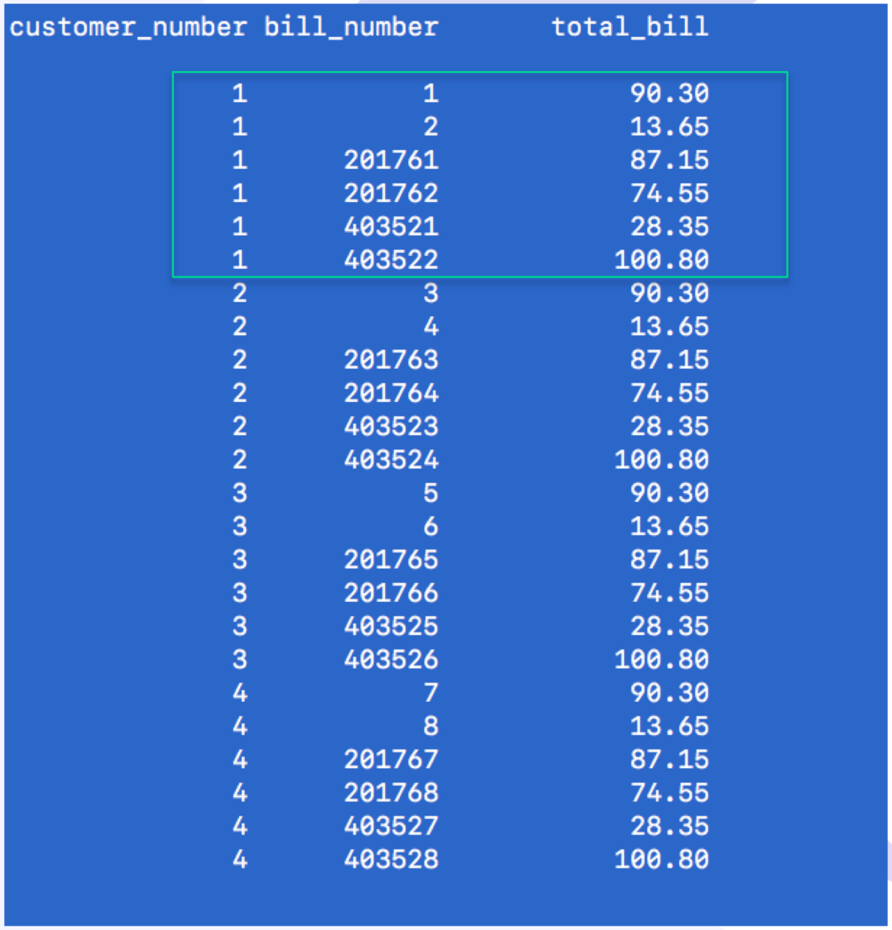
Output

```
bill_number       1
start_date        05/03/2009
f_year            2009
f_quarter         2
f_month           5
f_weekday         0
f_last_day        05/31/2009
f_to_char         05/03/2009
f_today           11/28/2018
f_months_between  114.806451612903
total_bill        90.30
```

SQL OLAP Functions

- The Over Clause
- **func(arg) over(partition by clause**
  order by clause
  window frame clause) •
- Defines the “domain” of OLAP function calculation
- partition by: divide into partitions
- order by: ordering within each partition
- window frame: sliding window within each partition
- all clauses optional

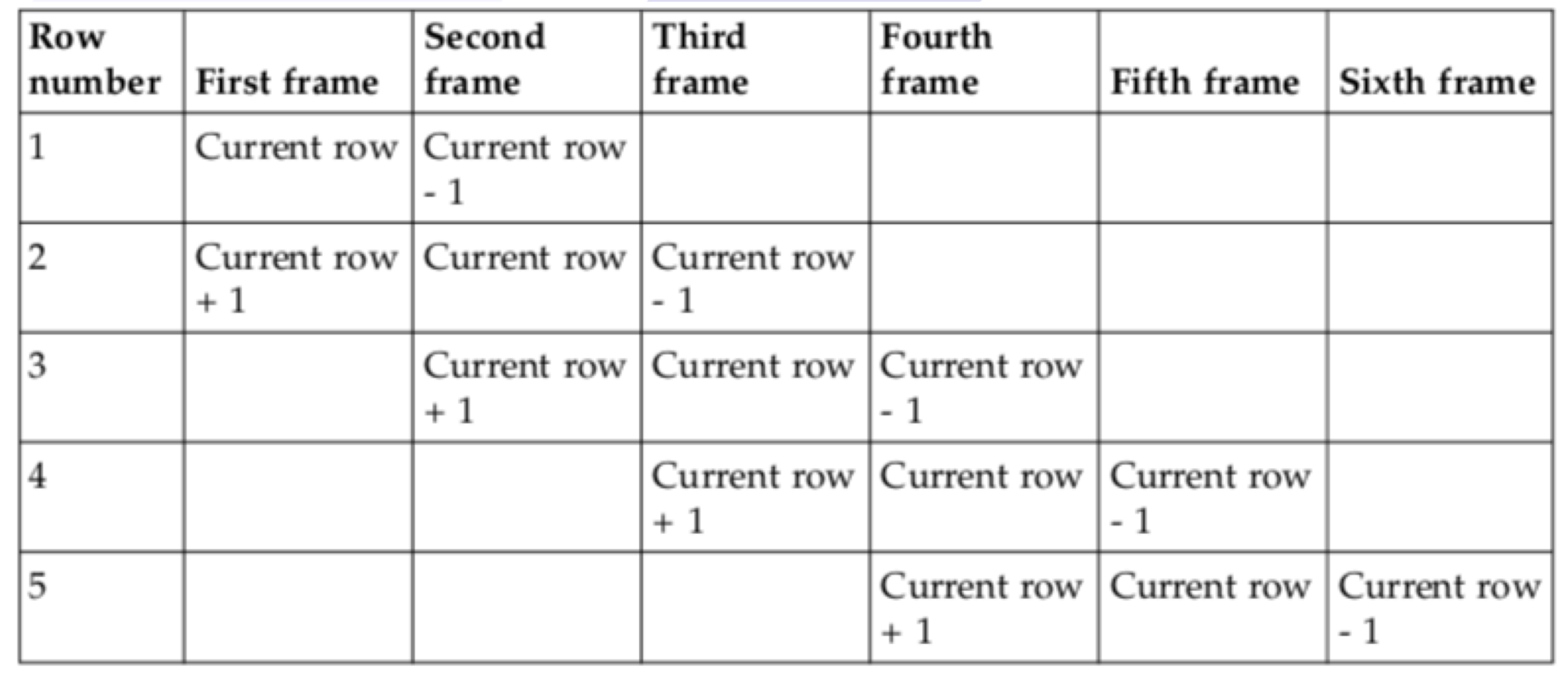
Window Example by Customer_number



```
customer_number bill_number        total_bill

              1           1             90.30
              1           2             13.65
              1      201761             87.15
              1      201762             74.55
              1      403521             28.35
              1      403522            100.80
              2           3             90.30
              2           4             13.65
              2      201763             87.15
              2      201764             74.55
              2      403523             28.35
              2      403524            100.80
              3           5             90.30
              3           6             13.65
              3      201765             87.15
              3      201766             74.55
              3      403525             28.35
              3      403526            100.80
              4           7             90.30
              4           8             13.65
              4      201767             87.15
              4      201768             74.55
              4      403527             28.35
              4      403528            100.80
```

Windows Frame



Row number	First frame	Second frame	Third frame	Fourth frame	Fifth frame	Sixth frame
1	Current row	Current row - 1				
2	Current row + 1	Current row	Current row - 1			
3		Current row + 1	Current row	Current row - 1		
4			Current row + 1	Current row	Current row - 1	
5				Current row + 1	Current row	Current row - 1

SQL OLAP Functions

- OLAP Aggregation Functions
- First_Value – first value in a OLAP Window
- Last_Value – last value in a OLAP Window
- Ratio_to_Report - calculates the fractional ratio of each row to the rest of the rows in the window

SQL OLAP Functions

- OLAP Ranking Functions
- Lag and Lead
- Rank
- Dense_Rank
- Percent_Rank
- Cum_dist -
- Ntile -

SQL Window Functions

- Max (column) - returns the highest value
- Min (column) - returns the lowest value
- Sum (column) - returns the sum of values
- Avg (column) – average of all values
- Stdev (column) – standard deviation
- Variance (column) – estimate of the population variance
- Range (column) - range of values

  range(expr) = max(expr) - min(expr);

SQL OLAP Examples

- Raw Data
  - 4 Customers
  - 6 Bills each
  - 24 Bills total

```
-- The raw data
select
        customer_number,
        bill_number,
        total_bill
from    bills
where   customer_number <=4;
```

SQL OLAP Functions

```
-- OLAP partition – Group the data by customer number – Count number of bills per customer
select
        customer_number,
        bill_number,
        total_bill,
        count(*) over  ( partition by customer_number ) as f_count_by_customer
from    bills
where   customer_number <=4;
```

SQL OLAP Functions

-- OLAP order by -- Using the Window Customer Number, how many bills by each customer

```
select
        customer_number,
        bill_number,
        total_bill,
        count(*) over  ( order by customer_number ) as f_count_by_customer
from    bills
where   customer_number <=4;
```

SQL OLAP Functions

```
-- OLAP Window – Counting the preceding and next rows
select
        customer_number,
        bill_number,
        total_bill,
        count(*) over  ( partition  by customer_number
                        order by customer_number
                        rows between 1 preceding and 1 following ) as
f_count_by_customer
from    bills
where   customer_number <=4;
```

SQL OLAP Functions

- OLAP Numbering Functions – Row_Number

```
select
        customer_number,
        bill_number,
        total_bill,
        row_number() over  ( partition by customer_number ) as f_ow_number
from    bills
where   customer_number <=4;
```

SQL OLAP

```
-- OLAP partition  - what is in the preceding and next row
select
     bill_number,
     total_bill,
     lag(total_bill) over  ( partition  by customer_number
               order by customer_number  ),
               -- rows between 1 preceding and 1 following ) as f_count_by_customer,
     lead(total_bill)  over  ( partition  by customer_number
               order by customer_number )
               -- rows between 1 preceding and 1 following ) as f_count_by_customer
from    bills
where   customer_number <=4;
```

SQL OLAP

```
-- OLAP partition  - Ranking Customers vs Dense Rank vs Percent Rank
select
        -- customer_number,
        bill_number,
        total_bill,
        rank() over  ( order by total_bill  ) as rank,
        dense_rank() over  ( order by total_bill  ) as dense_rank
from    bills
where   customer_number <=4;

select
        -- customer_number,
        bill_number,
        total_bill,
        rank() over  ( order by total_bill  ) as rank,
        percent_rank () over  ( order by total_bill  ) as percent_rank
from    bills
where   customer_number <=4;
```

SQL OLAP

```
-- OLAP partition  - Ratio to Report
select sum( total_bill ) from bills;

select
        state,
        sum(total_bill ) as sales,
        ratio_to_report ( sum( total_bill) ) over  ( ) * 100 as p_ratio_to_report
from    bills
group by state
order by sales;
```

References

- Fred Ho's Blog – SQL OLAP Functions in Informix
  - https://www.ibm.com/developerworks/community/blogs/fredho66/entry/sql_olap_functions_in_informix1?lang=en
- IBM Informix Knowledge Center
  - https://www.ibm.com/support/knowledgecenter/en/SSGU8G_12.1.0/com.ibm.sqls.doc/ids_sqs_2583.htm
- The importance of the OLAP functions by Fernando Nunes
  - http://informix-technology.blogspot.com/2014/03/the-importance-of-olap-functions.html

Questions?

Send follow-up questions to
Lester@advancedatatools.com

Informix Webcasts from the IBM Champions at Advanced DataTools

- **Informix Client Server Encryption by Thomas Beebe** - Thursday, April 11, 2019 at 2:00pm EST

- **Informix Databases Migrations, Upgrades and Exports by Mike Walker** - Thursday, May 2, 2019 at 2:00pm EST

Registration and more information:
https://advancedatatools.com/Informix/NextWebcast.html

Informix Training in 2019

Attend classes online on the web, or in person at our training center in Virginia. All you need is a web browser to connect to our WebEx training system and an SSH client (like Putty) to connect to our training lab for hands-on exercises. Each student uses an 8-core Linux server, with 16GB RAM, SSD drives with Informix 12, and several large databases for benchmark exercises.

- **March 11-14, 2019 - Advanced Informix Performance Tuning**
  - This course is for database administrators and application developers who will be responsible for managing, optimizing, and tuning an Informix database server. The focus is on skills, procedures, and scripts to improve the performance of your database server. The course will provide a toolkit of scripts and utilities to start monitoring and optimizing your Informix database server. Our advanced course is taught together by Lester Knutsen and Art Kagel, two of the most experienced Informix DBA consultants in the world.

- **April 22-25, 2019 - Informix for Database Administrators**
  - This course is for new database administrators, programmers, and technical support personnel who will be setting up, managing, and tuning IBM Informix databases.

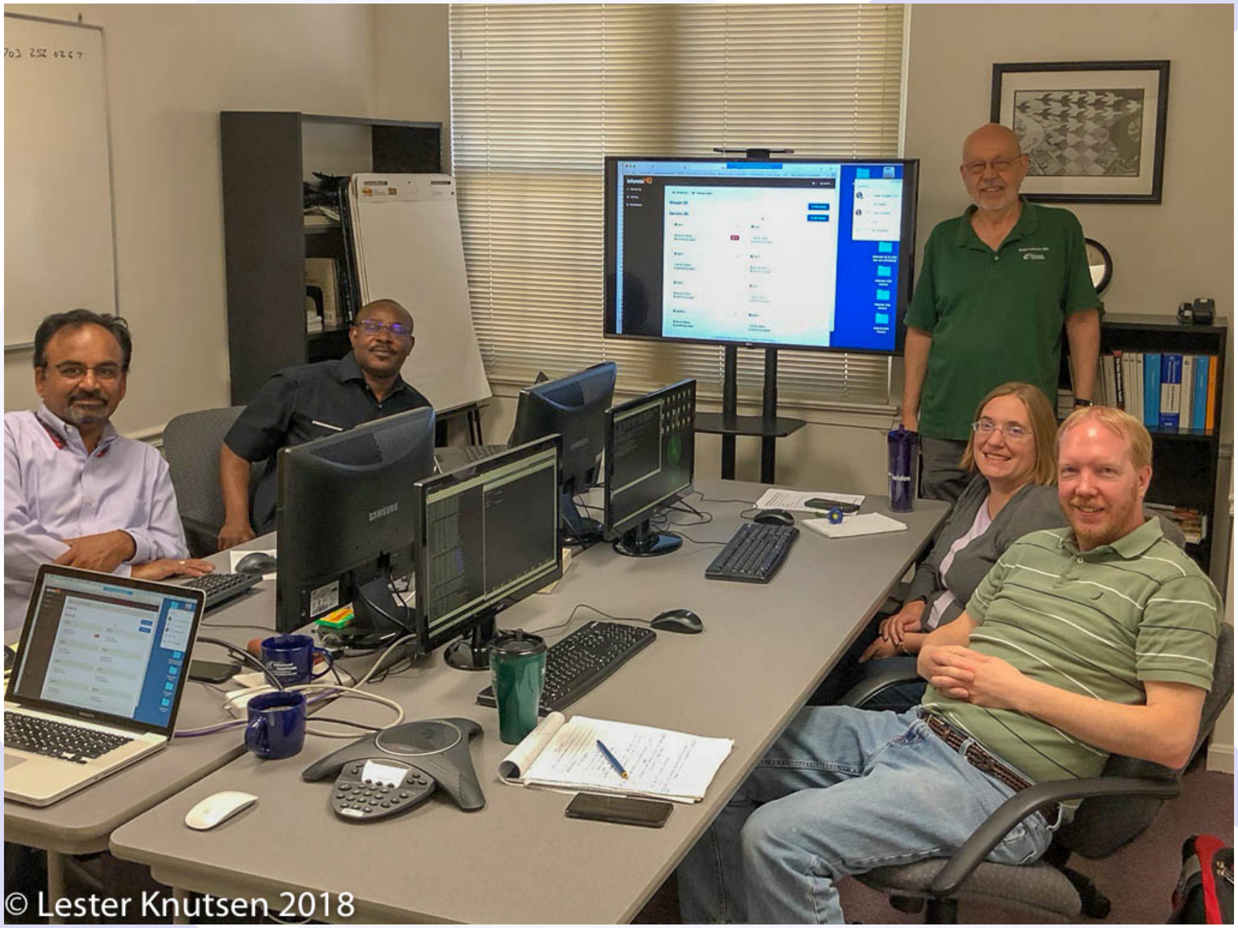
- **September 16-19, 2019 - Informix for Database Administrators**
  - This course is for new database administrators, programmers, and technical support personnel who will be setting up, managing, and tuning IBM Informix databases.

- **More Information and Registration at:**

http://www.advancedatatools.com/Training/InformixTraining.html

Advanced DataTools

Informix Training Servers


© Lester Knutsen 2018

Each Student in class will have a server running Informix 12.10 with:

- 8 CPU Cores
- 16 GB RAM
- 1 SSD Disk
- 1- 4 Disks

Advanced DataTools

Informix Support and Training from the Informix Champions!

Advanced DataTools is an Advanced Level IBM Informix Data Management Partner, and has been an authorized Informix partner since 1993. We have a long-term relationship with IBM, we have priority access to high-level support staff, technical information, and Beta programs. Our team has been working with Informix since its inception, and includes 8 Senior Informix Database Consultants, 4 IBM Champions, 2 IIUG Director's Award winners, and an IBM Gold Consultant. We have Informix specialists Lester Knutsen and Art Kagel available to support your Informix performance tuning and monitoring requirements!

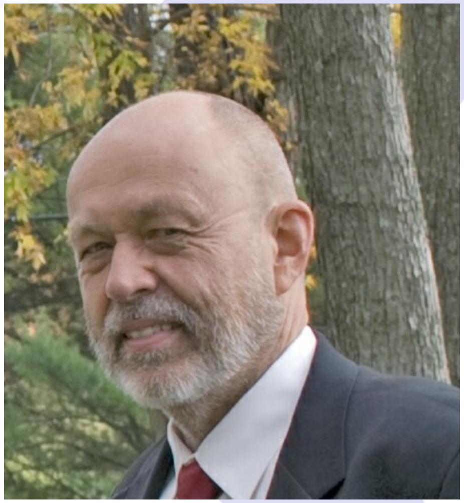
- ***Informix Remote DBA Support Monitoring***
- ***Informix Performance Tuning***
- ***Informix Training***
- ***Informix Consulting***
- ***Informix Development***

Free Informix Performance Tuning Webcast replays at:
http://advancedatatools.com/Informix/Webcasts.html
Email: info@advancedatatools.com
Web: http://www.advancedatatools.com

IBM Business Partner

Thank You

Advanced DataTools Corporation



For more information:

Lester@advancedatatools.com

http://www.advancedatatools.com