

# Advanced SQL

UKOUG Conference  
Masterclass

Daniel Fink  
OptimalDBA

# Learning Features

- Don't try to create the final query the first time
  - Use the Lego approach
  - Start with raw data and then add in the additional processing
- Always double check results
  - Eyeball usually works
  - May need a calculator
- Begin to rethink how you construct a query

# Logistics

- Presentation format
  - Slides for concept
  - SQL\*Plus script/output
    - Use personal DEMO schema
    - Script and output
- Ask Questions during topic

# Agenda

- Analytical Functions
- Regular Expressions

*I will not cover 11g*

# Analytical Functions

- Documented in Oracle Data Warehousing Guide and SQL Guide
- Core Concepts
  - Processing
  - Partition
  - Window
- Analytical Function Types
  - Ranking
  - Other Value
  - Window Aggregation
  - Statistical \*won't cover these

# Uses of Analytical SQL

- Top/Bottom N queries
  - Top 5 sales people
  - Bottom 15 customers
- Comparisons
  - Month to month sales increases/decreases
- Ranking/Ratios
  - Top 10 selling items
  - % of overall sales by item

# Why Analytical SQL

- Improved code
  - Readability
  - Supportability
- Improved Performance
  - Fewer passes on a table
  - Less resource consumption
  - Faster run time
- Part of the core product

# SQL\*Plus Demo

- Report on all employees sorted by salary and show their numeric ranking

Analytical Function

```
SELECT      e.name ,  
            e.salary ,  
            DENSE_RANK( ) OVER  
              (ORDER BY salary DESC)  
              AS sal_rank  
FROM        employee e  
ORDER BY    e.salary DESC
```

ORDER BY clause

# Partitioning

*Not to be confused with table/index partitioning*

- A set of rows grouped by a defined data element
  - Default partition is entire result set
  - Fixed by data values
- Functions are applied within the partition
  - Values are "reset" at partition boundaries
- 1 partition per function
  - Multiple partitions per statement

# SQL\*Plus Demo

- Report on all employees sorted by department number and salary and show their numeric ranking within their department

# Processing

```
SELECT      e.name ,
            d.name dept_name ,
            e.salary ,
            DENSE_RANK() OVER
              (PARTITION BY d.name
               ORDER BY e.salary DESC)
              AS sal_rank
FROM        employee e ,
            department d
WHERE       d.dept_id = e.dept_id
ORDER BY   d.dept_id, e.salary DESC
```

SELECT  
WHERE/joins  
GROUP BY/HAVING

```
SELECT      e.name,  
            d.name dept_name,  
            e.salary  
FROM        employee e,  
            department d  
WHERE       d.dept_id = e.dept_id
```

Partitioning  
Ordering  
Windowing  
Apply  
Functions

```
DENSE_RANK() OVER  
  (PARTITION BY d.name  
   ORDER BY e.salary DESC)  
AS sal_rank
```

ORDER BY

```
ORDER BY  d.dept_id,  
          e.salary DESC
```

Employee	Department	Salary
-----	-----	-----
Allison Ballinger	Administration	\$90,000
Julie Johnson	Sales and Marketing	\$75,000
Stan Marsh	Accounting	\$45,000
Larry Wilton	Sales and Marketing	\$30,000
Thomas Walton	Logistics and Supply	\$25,000
Tina Walton	Logistics and Supply	\$80,000
Shelly Walton	Sales and Marketing	\$30,000
Eric Kraus	Logistics and Supply	\$27,000
John Dennis	Accounting	\$46,000
Bobby Harris	Sales and Marketing	\$30,000
Doug Harris	Logistics and Supply	\$50,000
Erika Deeter	Accounting	\$60,000
William Dietrich	Administration	\$30,000
Allison Dietrich	Sales and Marketing	\$30,000
Andy Schmidt	Sales and Marketing	\$70,000
Rachel Middleton	Administration	\$850,000
Henry Parry	Sales and Marketing	\$30,000
Bev George	Sales and Marketing	\$30,000
Oscar Perry	Sales and Marketing	\$45,000
Billy Yolto	Logistics and Supply	\$30,000
Vincent Johns	Logistics and Supply	\$50,000

Employee	Department	Salary
-----	-----	-----
Allison Ballinger	Administration	\$90,000
William Dietrich	Administration	\$30,000
Rachel Middleton	Administration	\$850,000
Julie Johnson	Sales and Marketing	\$75,000
Larry Wilton	Sales and Marketing	\$30,000
Shelly Walton	Sales and Marketing	\$30,000
Bobby Harris	Sales and Marketing	\$30,000
Allison Dietrich	Sales and Marketing	\$30,000
Andy Schmidt	Sales and Marketing	\$70,000
Henry Parry	Sales and Marketing	\$30,000
Bev George	Sales and Marketing	\$30,000
Oscar Perry	Sales and Marketing	\$45,000
Stan Marsh	Accounting	\$45,000
John Dennis	Accounting	\$46,000
Erika Deeter	Accounting	\$60,000
Thomas Walton	Logistics and Supply	\$25,000
Tina Walton	Logistics and Supply	\$80,000
Eric Kraus	Logistics and Supply	\$27,000
Doug Harris	Logistics and Supply	\$50,000
Billy Yolto	Logistics and Supply	\$30,000
Vincent Johns	Logistics and Supply	\$50,000

Employee	Department	Salary
-----	-----	-----
Rachel Middleton	Administration	\$850,000
Allison Ballinger	Administration	\$90,000
William Dietrich	Administration	\$30,000
Julie Johnson	Sales and Marketing	\$75,000
Andy Schmidt	Sales and Marketing	\$70,000
Oscar Perry	Sales and Marketing	\$45,000
Larry Wilton	Sales and Marketing	\$30,000
Shelly Walton	Sales and Marketing	\$30,000
Bobby Harris	Sales and Marketing	\$30,000
Allison Dietrich	Sales and Marketing	\$30,000
Henry Parry	Sales and Marketing	\$30,000
Bev George	Sales and Marketing	\$30,000
Erika Deeter	Accounting	\$60,000
John Dennis	Accounting	\$46,000
Stan Marsh	Accounting	\$45,000
Tina Walton	Logistics and Supply	\$80,000
Doug Harris	Logistics and Supply	\$50,000
Vincent Johns	Logistics and Supply	\$50,000
Billy Yolto	Logistics and Supply	\$30,000
Eric Kraus	Logistics and Supply	\$27,000
Thomas Walton	Logistics and Supply	\$25,000

Employee	Department	Salary	Rank
-----	-----	-----	-----
Rachel Middleton	Administration	\$850,000	1
Allison Ballinger	Administration	\$90,000	2
William Dietrich	Administration	\$30,000	3
Julie Johnson	Sales and Marketing	\$75,000	1
Andy Schmidt	Sales and Marketing	\$70,000	2
Oscar Perry	Sales and Marketing	\$45,000	3
Larry Wilton	Sales and Marketing	\$30,000	4
Shelly Walton	Sales and Marketing	\$30,000	4
Bobby Harris	Sales and Marketing	\$30,000	4
Allison Dietrich	Sales and Marketing	\$30,000	4
Henry Parry	Sales and Marketing	\$30,000	4
Bev George	Sales and Marketing	\$30,000	4
Erika Deeter	Accounting	\$60,000	1
John Dennis	Accounting	\$46,000	2
Stan Marsh	Accounting	\$45,000	3
Tina Walton	Logistics and Supply	\$80,000	1
Doug Harris	Logistics and Supply	\$50,000	2
Vincent Johns	Logistics and Supply	\$50,000	2
Billy Yolto	Logistics and Supply	\$30,000	3
Eric Kraus	Logistics and Supply	\$27,000	4
Thomas Walton	Logistics and Supply	\$25,000	5

Employee	Department	Salary	Rank
Rachel Middleton	Administration	\$850,000	1
Allison Ballinger	Administration	\$90,000	2
William Dietrich	Administration	\$30,000	3
Julie Johnson	Sales and Marketing	\$75,000	1
Andy Schmidt	Sales and Marketing	\$70,000	2
Oscar Perry	Sales and Marketing	\$45,000	3
Bev George	Sales and Marketing	\$30,000	4
Larry Wilton	Sales and Marketing	\$30,000	4
Henry Parry	Sales and Marketing	\$30,000	4
Allison Dietrich	Sales and Marketing	\$30,000	4
Shelly Walton	Sales and Marketing	\$30,000	4
Bobby Harris	Sales and Marketing	\$30,000	4
Erika Deeter	Accounting	\$60,000	1
John Dennis	Accounting	\$46,000	2
Stan Marsh	Accounting	\$45,000	3
Tina Walton	Logistics and Supply	\$80,000	1
Doug Harris	Logistics and Supply	\$50,000	2
Vincent Johns	Logistics and Supply	\$50,000	2
Billy Yolto	Logistics and Supply	\$30,000	3
Eric Kraus	Logistics and Supply	\$27,000	4
Thomas Walton	Logistics and Supply	\$25,000	5

# Composite Partition

- Multiple data fields can be defined as a partition
  - RANK reset at each partition boundary
- Somewhat useful...
  - Multiple RANKS
  - Separate reports with Aggregate RANK probably better option

# SQL\*Plus Demo

- Report on sales by salesperson in Nov/Dec 2008 for Boots, and Tents and rank according to type and monthly sales by type

# ORDER BY

- Analytical Functions are allowed in ORDER BY clause
  - PARTITION
  - ORDER BY
- May not be the clearest of code
  - Query column name
  - Sort order from clause

# SQL\*Plus Demo

- Report on all employees salaries by department and sort by their ranking within the department

# Aggregate Values

- You can RANK on an aggregate value
- Avoids using subquery to generate aggregate value and then applying function

**RANK ( ) OVER**

**ORDER BY ( SUM ( SALARY ) DESC )**

# SQL\*Plus Demo

- Report on the total salary in each department and rank by total salary

# Multiple Functions

- Each function is applied according to it's own clause
- Multiple partitions can be used in a single statement

# SQL\*Plus Demo

- Report on employee salary and rank by department and job title

# Other Value

- LAG - previous value
- LEAD - subsequent value
- FIRST - first value in order
- LAST - last value in order

# Lag/Lead

- Find a value in a row before/after the current row
  - `LAG/LEAD(expression)`
  - Expression can be data or function
- Optional Parameters
  - Offset - relative position (default is 1)
  - Default - value if offset not in partition (default is NULL)

# SQL\*Plus Demo

- What is the monthly difference in sales for Allison Dietrich in 2007?
- What is the monthly difference in sales for Henry Parry and Allison Dietrich in 2007?
- What is the difference in monthly sales from Allison Dietrich compared to the previous year?

# Windowing

- Range of rows relative to the current row
  - Default is all rows in result set
  - Dynamic based on row or data
- 1 per function
  - Multiple windows per statement
  - Will not span a partition boundary

# SQL\*Plus Demo

- What were the 2007 monthly sales for Allison Dietrich and what was running total for the year?
- What were the 2007 monthly sales for Allison Dietrich and what was the 3 month average (the current month is the mid point)?

Employee	Month	Sales	Average Sales
-----	-----	-----	-----
Allison Dietrich	01-JAN-07	\$68,758,209	\$63,779,320
Allison Dietrich	01-FEB-07	\$58,800,431	\$59,911,777
Allison Dietrich	01-MAR-07	\$52,176,691	\$54,751,431
Allison Dietrich	01-APR-07	\$53,277,171	\$54,504,683
Allison Dietrich	01-MAY-07	\$58,060,187	\$57,630,093
Allison Dietrich	01-JUN-07	\$61,552,921	\$58,068,567
Allison Dietrich	01-JUL-07	\$54,592,592	\$61,304,622
Allison Dietrich	01-AUG-07	\$67,768,354	\$57,979,014
Allison Dietrich	01-SEP-07	\$51,576,097	\$57,100,083
Allison Dietrich	01-OCT-07	\$51,955,799	\$52,414,127
Allison Dietrich	01-NOV-07	\$53,710,486	\$56,141,177
Allison Dietrich	01-DEC-07	\$62,757,245	\$53,555,005

	Employee	Month	Sales	Average Sales
-1	-----	-----	-----	-----
0	Allison Dietrich	01-JAN-07	\$68,758,209	\$63,779,320
+1	Allison Dietrich	01-FEB-07	\$58,800,431	\$59,911,777
	Allison Dietrich	01-MAR-07	\$52,176,691	\$54,751,431
	Allison Dietrich	01-APR-07	\$53,277,171	\$54,504,683
	Allison Dietrich	01-MAY-07	\$58,060,187	\$57,630,093
	Allison Dietrich	01-JUN-07	\$61,552,921	\$58,068,567
	Allison Dietrich	01-JUL-07	\$54,592,592	\$61,304,622
	Allison Dietrich	01-AUG-07	\$67,768,354	\$57,979,014
	Allison Dietrich	01-SEP-07	\$51,576,097	\$57,100,083
	Allison Dietrich	01-OCT-07	\$51,955,799	\$52,414,127
	Allison Dietrich	01-NOV-07	\$53,710,486	\$56,141,177
	Allison Dietrich	01-DEC-07	\$62,757,245	\$53,555,005

	Employee	Month	Sales	Average Sales
	-----	-----	-----	-----
-1	Allison Dietrich	01-JAN-07	\$68,758,209	\$63,779,320
0	Allison Dietrich	01-FEB-07	\$58,800,431	
+1	Allison Dietrich	01-MAR-07	\$52,176,691	
	Allison Dietrich	01-APR-07	\$53,277,171	\$54,504,683
	Allison Dietrich	01-MAY-07	\$58,060,187	\$57,630,093
	Allison Dietrich	01-JUN-07	\$61,552,921	\$58,068,567
	Allison Dietrich	01-JUL-07	\$54,592,592	\$61,304,622
	Allison Dietrich	01-AUG-07	\$67,768,354	\$57,979,014
	Allison Dietrich	01-SEP-07	\$51,576,097	\$57,100,083
	Allison Dietrich	01-OCT-07	\$51,955,799	\$52,414,127
	Allison Dietrich	01-NOV-07	\$53,710,486	\$56,141,177
	Allison Dietrich	01-DEC-07	\$62,757,245	\$53,555,005

	Employee	Month	Sales	Average Sales
	-----	-----	-----	-----
	Allison Dietrich	01-JAN-07	\$68,758,209	\$63,779,320
-1	Allison Dietrich	01-FEB-07	\$58,800,431	\$59,911,777
0	Allison Dietrich	01-MAR-07	\$52,176,691	
+1	Allison Dietrich	01-APR-07	\$53,277,171	
	Allison Dietrich	01-MAY-07	\$58,060,187	\$57,630,093
	Allison Dietrich	01-JUN-07	\$61,552,921	\$58,068,567
	Allison Dietrich	01-JUL-07	\$54,592,592	\$61,304,622
	Allison Dietrich	01-AUG-07	\$67,768,354	\$57,979,014
	Allison Dietrich	01-SEP-07	\$51,576,097	\$57,100,083
	Allison Dietrich	01-OCT-07	\$51,955,799	\$52,414,127
	Allison Dietrich	01-NOV-07	\$53,710,486	\$56,141,177
	Allison Dietrich	01-DEC-07	\$62,757,245	\$53,555,005

Employee	Month	Sales	Average Sales
-----	-----	-----	-----
Allison Dietrich	01-JAN-07	\$68,758,209	\$63,779,320
Allison Dietrich	01-FEB-07	\$58,800,431	\$59,911,777
Allison Dietrich	01-MAR-07	\$52,176,691	\$54,751,431
Allison Dietrich	01-APR-07	\$53,277,171	\$54,504,683
Allison Dietrich	01-MAY-07	\$58,060,187	\$57,630,093
Allison Dietrich	01-JUN-07	\$61,552,921	\$58,068,567
Allison Dietrich	01-JUL-07	\$54,592,592	\$61,304,622
Allison Dietrich	01-AUG-07	\$67,768,354	\$57,979,014
Allison Dietrich	01-SEP-07	\$51,576,097	\$57,100,083
Allison Dietrich	01-OCT-07	\$51,955,799	\$52,414,127
-1 Allison Dietrich	01-NOV-07	\$53,710,486	\$56,141,177
0 Allison Dietrich	01-DEC-07	\$62,757,245	
+1			\$53,555,005

# Offset

- **DEFAULT** equivalent to  
**ROWS BETWEEN UNBOUNDED PRECEDING**  
**AND UNBOUNDED FOLLOWING**  
**RANGE BETWEEN UNBOUNDED PRECEDING**  
**AND UNBOUNDED FOLLOWING**

\*Within the **PARTITION!**

# SQL\*Plus Demo

- What were the 2007 monthly sales for Allison Dietrich and Henry Parry and what was running total for the year?

# Logical Offset

- Bases the offset on data, not row counts
- Useful if data is missing

# SQL\*Plus Demo

- What were the 2007 monthly sales for Allison Dietrich and what was the 3 month average (the current month is the mid point)?

# Filter limitation

- Analytical Functions cannot be used to filter records
  - WHERE (filter predicate) is applied BEFORE functions are processed

# SQL\*Plus Demo

- What were the top 3 Salespeople for 2007?

SELECT  
WHERE/joins  
GROUP BY/HAVING

```
SELECT      name emp_name
            ,      SUM(sales_month) year_sales
FROM        monthly_sales_person
WHERE       order_month
            BETWEEN '01-JAN-07' AND '01-DEC-07'
            AND    RANK() OVER
                   (ORDER BY SUM(sales_month) DESC) >= 3
GROUP BY   name
```

Partitioning  
Ordering  
Windowing  
Apply  
Functions

```
RANK() OVER
  (ORDER BY SUM(sales_month) DESC)
rank
```

ORDER BY

```
ORDER BY  year_sales DESC
```

# Then...how can I create a Top N Query?

- Subquery
- With WITH \*

\*no this is not a typo...

```
WITH <name> AS (subquery)
SELECT * FROM <name>
```

```
WITH <name> AS (subquery),
     <name> AS (subquery)
SELECT * FROM <name>
```

# WITH

- Creates a named subquery
- Subquery can be referenced in subsequent query(ies)
  - Can be nested
- **Opinion**
  - *Cleaner, easier to read code than multiple nested inline subqueries*
  - *Develop using modular approach*

# SQL\*Plus Demo

- What were the top 3 Salespeople for 2007?

# Analytical Function Review

- Rethink sql
- Very powerful, but some limitations
  - Cannot use in WHERE clause
  - Often need subqueries to restrict output

# Regular Expressions

- 'Documented' in SQL reference, Application Developers Guide - Fundamentals
- Pattern Matching and Processing
  - Several different ways to match the same data
  - Learning Pattern Matching is critical

- Similar to older functions/condition
  - LIKE            REGEXP\_LIKE
  - INSTR          REGEXP\_INSTR
  - SUBSTR        REGEXP\_SUBSTR
  - REPLACE       REGEXP\_REPLACE

- All about patterns
  - Recognize
  - Describe
  - Manipulate
- Know your data
  - Data drives the pattern
  - Beware of data that does not fit the pattern

# REGEXP\_LIKE

- Condition that evaluates to TRUE or FALSE
  - WHERE regexp\_like ( )
  - No need for % wildcard
  - Matches pattern anywhere in string
- Use to locate data before modifying it

# SQL\*Plus Demo

- Locate employees named Parry or Perry
- Locate employees with 'dd' in their name

# Format and Options

- REGEXP\_LIKE (string, pattern, parameters)
  - String - character string or character data
  - Pattern - regular expression
  - Parameters \*
    - 'i' case insensitive searching
    - 'c' case sensitive searching
    - 'n' allow '.' to match newline character
    - 'm' source string is multiple lines
    - 'x' ignore whitespace

# Patterns

- . Any character (one or more)
- \* 0 or more of the previous
- ? 0 or 1 of the previous
- + 1 or more of the previous
- ^ Beginning of the line
- \$ End of the line

- {m} Exact m of previous
- {m,} M or more of previous
- {m,n} Between m and n of previous
- [...] Characters to match
- [^...] Characters not to match
- () Group/sub-expression of characters
- | Or

# SQL\*Plus Demo

- Locate employees with 'h' in their name
- Locate employees with 'h' surrounded by at least one character on each side
- Locate employees whose name starts with 'h'
- Locate employees whose name ends with 'h'

- Locate employees whose name starts with TH
- Locate employees whose name starts with T or H
- Locate employees with 'll' in their name
- Locate employees with 'll' in their name, but exclude 'lly'

# Matching Lists

- A-Z
  - a-z
  - 0-9
  - a-zA-Z0-9
- UPPERCASE alpha
  - lowercase alpha
  - Numbers
  - Alphanumeric

# Character Classes

- [:alpha:]
  - Alphabetic
- [:lower:]
  - lowercase alpha
- [:upper:]
  - UPPERCASE alpha
- [:digit:]
  - Numbers
- [:alnum:]
  - Alphanumeric
- [:cntrl:]
  - Control

# SQL\*Plus Demo

- Find all the order comments that end with a lower case letter
- Find all the order comments that have a number in them
- Find all the order comments that have a newline in them

# Finding Patterns

- Locate comments with Phone Numbers

- US 10 numbers (3 3 4)

- Variable formats

- ( 783 ) 555 - 2327

- 872 . 555 . 8730

- 872 555 8730

- 3245558728

- What is the pattern?

- Let's build the pattern to match
  - Numeric format 3 3 4
    - [0-9]{3} [0-9]{3} [0-9]{4}
    - [[:digit:]]{3} [[:digit:]]{3} [[:digit:]]{4}

# What about between the numbers?

(783)555-2327

872.555.8730

872 555 8730

3245558728

**(\(| | )###(\| | | .)###(-| | | .)####**

- ? Match 0 or 1
- {0,} Match 0 or more

(783)555-2327

872.555.8730

872 555 8730

3245558728

(\(| )?###(\| |.)?###(-| |.)?####

(\(| ){0,}###(\| |.){0,}###(-| |.){0,}####

# SQL\*Plus Demo

- Find all the order comments that contain a US telephone number

# REGEXP\_SUBSTR

- Return parts of a string based on the pattern
  - Very similar to SUBSTR
- Compliment to REGEXP\_LIKE
  - You can see what you match

- (string, pattern, position, occurrence, parameters)
  - String - character string or character data
  - Pattern - regular expression
  - Position - place to start in the string
  - Occurrence - which occurrence of the pattern to replace

# SQL\*Plus Demo

- Find all the US telephone numbers in order comments

# Positions and Occurrences

- Start searching at Y position
  - Default is 1
  - -N is not supported
- Match the Nth occurrence
  - Default is 1
  - -N is not supported

# SQL\*Plus Demo

- Find the second occurrence of the pattern ".NNN"
- Find the first occurrence of the pattern ".NNN" after character 47

# REGEXP\_INSTR

- Return location of a string based on the pattern
  - Very similar to INSTR
  - Return option is different
- Compliment to REGEXP\_LIKE
  - You can see where you match

- (string, pattern, position, occurrence, return\_option, parameters)
  - String - character string or character data
  - Pattern - regular expression
  - Position - place to start in the string
  - Occurrence - which occurrence of the pattern to replace
  - Return Option - return the first (0) or last (1) character position of the pattern

# SQL\*Plus Demo

- Find the starting position of the telephone numbers
- Find the ending position of the telephone numbers

# REGEXP\_REPLACE

- Replaces parts of a string based on the pattern
- Very powerful!
  - First introduction to regular expressions

- (string, pattern, replace\_string, position, occurrence, parameters)
  - String - character string or character data
  - Pattern - regular expression
  - Replace\_string - character string to use as replacement
  - Position - place to start in the string
  - Occurrence - which occurrence of the pattern to replace

# SQL\*Plus Demo

- Replace multiple spaces in comments with single spaces

# Regular Expression Review

- Extends basic functionality
- Learn Pattern Matching
  - Know Thy Data

# Daniel Fink

[www.optimaldba.com](http://www.optimaldba.com)

[daniel.fink@optimaldba.com](mailto:daniel.fink@optimaldba.com)