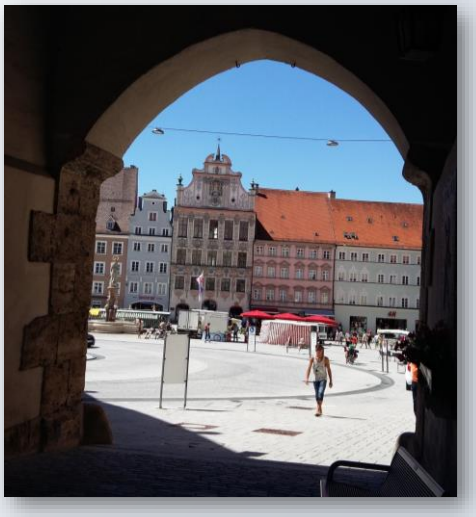
OLAP Specifications
Much more than Running Numbers

Birgitta Hauser
Diplom-Betriebswirt (BA)
Database and Software Architect
Hauser@SSS-Software.de

i-UG

Landsberg am Lech



Agenda

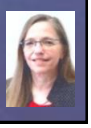
OLAP Specifications

- Functions
- Window-Order-Clause
- Window-Partition-Clause

OLAP Categories

- Numbering Specifications
- Ordered Specifications
- Aggregate Specifications

OLAP – Specifications



OLAP Specifications

OLAP = OnLine Analytical Processing

- Ability to return **ranking, row numbering**, and **aggregate function** information as a scalar value in a query result
- Can be specified within the **SELECT List** or the **ORDER BY** clause

OLAP Categories

- Numbering Specifications → `Row_Number()`
- Ordered OLAP Specifications → `Rank(), Dense_Rank()`
  → `Lag(…), Lead(…), Ntile()`
  `Cume_Dist(), Percent_Rank()`
- Aggregate Specifications → all kinds of aggregate functions
  `Sum(), Avg(), Count(), Max() …`
  `Variance(), Median() …`

New RELEASE 6.1

New RELEASE 7.3

New RELEASE 7.3

OLAP Specifications - Syntax

Over = Definition of the window Over the Result Set

Window-Order-Clause for all OLAP Functions

- **Sort sequence** of the rows
  → the final Order By may sort the result in a different sequence
- Sometimes requested

Window-Partition-Clause for all OLAP Functions

- **Boundaries between partitions** within the window → **Level Break**
- Optional

OLAP Specifications

- Sequence of the OLAP function may or may not match the ORDER BY

```
Select Row_Number() Over() RowNbr,
       CustNo, Amount
  From SalesCusty
  Where SalesYear = 2009
```

ROWNBR	CUSTNO	AMOUNT
1	10003	4589,86
2	10004	2673,95
3	10005	3741,95
4	10001	2634,20
5	10002	1636,25

- Without ORDER BY

```
Select Row_Number() Over() RowNbr,
       CustNo, Amount
  From SalesCusty
  Where SalesYear = 2009
  Order By Amount Desc
```

ROWNBR	CUSTNO	AMOUNT
1	10003	4589,86
3	10005	3741,95
2	10004	2673,95
4	10001	2634,20
5	10002	1636,25

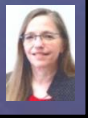
- Sequence: Amount ascending

```
Select Row_Number() Over() RowNbr,
       CustNo, Amount
  From SalesCusty
  Where SalesYear = 2009
  Order By CustNo
```

ROWNBR	CUSTNO	AMOUNT
4	10001	2634,20
5	10002	1636,25
1	10003	4589,86
2	10004	2673,95
3	10005	3741,95

- Sequence: CustNo descending

OLAP – Specifications: Windows Order Clause

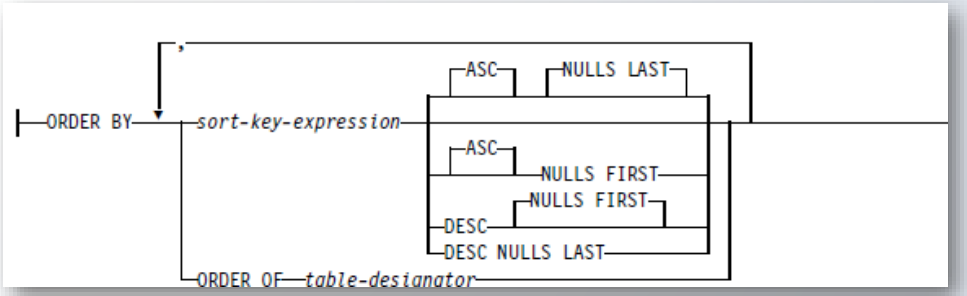


OLAP - Ranking Functions - Window-Order-Clause

Window-Order-Clause - OVER(ORDER BY Clause)

- Defines the **sequence of the running number or ranks** which may differ from the sequence specified in the ORDER BY Clause
- Only allowed **in composition** with the **OLAP Ranking functions**
  - **ROW_NUMBER(), RANK(), DENSE_RANK() …**
  - Must be specified for each **OLAP function**, immediately after the function
- **Syntax analog ORDER-BY clause**
  - **Several key fields** can be listed
  - **Ascending** and **Descending** Sequence allowed (**ASC / DESC**)

OLAP Specifiactions - Window Order Clause



Ordering for calculating the Rank or Row_Number

- **Multiple** Columns/Expressions for ordering in **ascending or descending** sequence separated by a comma allowed
- **NULL values** are positioned per **default at the end** of the list
  → **NULLS FIRST:** NULL Values are positioned at the **begin** of the list

OLAP-Ranking Functions - Examples

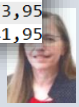
```
Select Row_Number() Over(Order By Amount Desc) RowNbr,
       CustNo, Amount
  From SalesCusty
  Where SalesYear = 2009
  Order By Amount Desc;
```

ROWNBR	CUSTNO	AMOUNT
1	10003	4589,86
2	10005	3741,95
3	10004	2673,95
4	10001	2634,20
5	10002	1636,25

- Generating running numbers based on the OVER(ORDER BY) Clause

```
Select Row_Number() Over(Order By Amount Desc) RowNbr,
       CustNo, Amount
  From hscommon10.SalesCusty
  Where SalesYear = 2009
  Order By CustNo;
```

ROWNBR	CUSTNO	AMOUNT
4	10001	2634,20
5	10002	1636,25
1	10003	4589,86
3	10004	2673,95
2	10005	3741,95



OLAP-Ranking Functions - Rank() versus Dense_Rank()

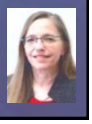
```
Select Employee, Amount,
       Rank()       Over(Order By Amount Desc) Rank,
       Dense_Rank() Over(Order By Amount Desc) DenseRank
   from SalesEmp
   Where SalesYear = 2008
   Order By Amount Desc, Employee
```

EMPLOYEE	AMOUNT	RANK	DENSERANK
Müller	120000,00	1	1
Bauer	110000,00	2	2
Jäger	110000,00	2	2
Huber	100000,00	4	3
Schmidt	100000,00	4	3
Fischer	80000,00	6	4
Meier	80000,00	6	4

Two second ranks:

- **Rank()** 2 2nd Ranks: No 3rd Rank, Next Rank = 4
- **Dense_Rank()** 2 2nd Ranks: Next Rank = 3

OLAP – Specifications:
Windows Partition Clause



OLAP-Ranking-Functions - Window-Partition-Clause

Window-Partition-Clause – Over(Partition By) Clause

- Must be specified within the **`OVER()` Clause**
  - **`Partition by`** must be specified **immediately after** **`OVER(`**
    → **`Order By`** clause **follows** the **`Partition By`** clause
  - **Multiple** column names separated by a comma can be specified
- Numbering is **reset** at **Level Break**
  - Example: Rank the sales persons for several years

OLAP - Ranking Functions - Window-Partition-Clause - Example

```
Select SalesYear, Dense_Rank() Over(Partition By SalesYear
                                    Order By Amount Desc) Rank,
       CustNo, Amount
  From SalesCustY
  Order By SalesYear, Amount Desc, CustNo;
```

SALESYEAR	RANK	CUSTNO	AMOUNT
2008	1	10002	1350,00
2008	2	10003	535,00
2008	3	10004	470,00
2008	4	10005	310,00
2008	5	10001	115,00
2009	1	10003	4589,86
2009	2	10005	3741,95
2009	3	10004	2673,95
2009	4	10001	2634,20
2009	5	10002	1636,25
2010	1	10006	9712,85
2010	2	10003	1555,75
2010	3	10001	140,97

- Reset numbering depending on the year

OLAP - Ranking Functions - Restrictions

OLAP Functions are not allowed in WHERE Conditions!

Selection of running numbers or ranks requires additional

- Nested **Sub-Selects**
- **Common-Table-Expressions** (CTE)
  - Example: Determine the customers with the 2nd to 4th highest Sales

```
With x as (Select a.*, Dense_Rank() Over(Order By Amount) as RankEmp
             from SalesEmp a
             Where SalesYear = 2009)
Select * From x
  Where RankEmp between 2 and 4;
```

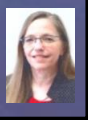
- OLAP Function in a **CTE**

ID	SALESYEAR	EMPLOYEE	AMOUNT	RANKEMP
9	2009	Müller ...	90000,00	2
11	2009	Schmidt ...	100000,00	3
8	2009	Huber ...	110000,00	4
12	2009	Gärtner ...	110000,00	4
15	2009	Schuster...	110000,00	4

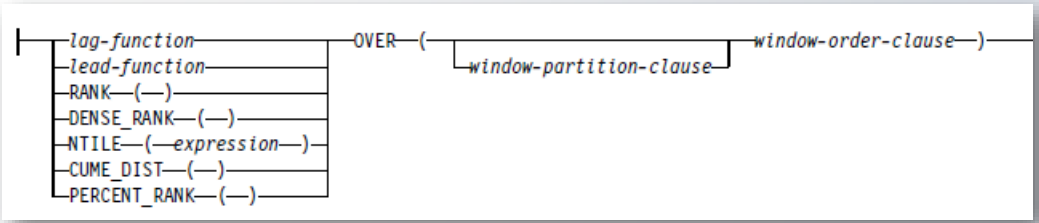
```
Select *
  From (Select a.*, Dense_Rank() Over(Order By Amount) as RankEmp
          From SalesEmp a
          Where SalesYear = 2009) x
  Where RankEmp between 2 and 4;
```

- OLAP Function in a (nested) **Sub-Select**

OLAP - Ordered Specifications



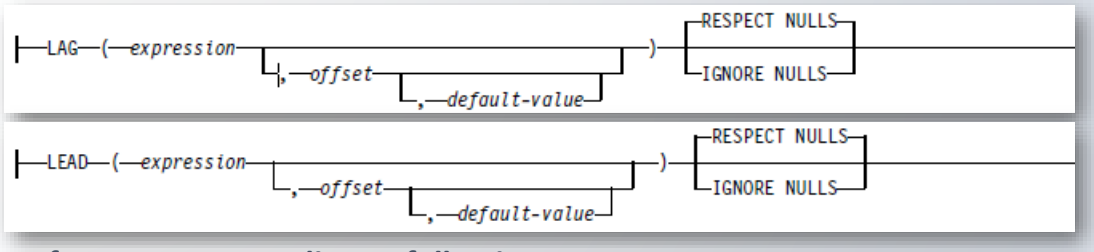
Ordered OLAP Specifications



Ordered OLAP Specifications

- `Rank()/Dense_Rank()` **Ordinal rank** of a row within the window
- `Lag`-Function Reference to a **preceding row** in the window
- `Lead`-Function Reference to a **following row** in the window
- `Ntile()` **Quantile rank** of a row within the window
- `Cume_Dist()` **Percentile ranking** of a row within the window including the current result set
- `Percent_Rank()` **relative percentile rank** of a row within the window

Ordered OLAP Specifications - LAG and LEAD Functions



References to preceding or following Rows

Additional Options allow greater Flexibility

- ***Offset*** **relative position** of the **previous/following row** based on the current row
  Must be a positive integer value (Not specified = 1)
- ***Default-Value*** **Replacement** for a NULL value
- ***IGNORE NULLS*** Rows where the expression returns a **NULL value** are **not considered**

Ordered OLAP Specifications
LAG and LEAD Functions – Example 1

```
Select SalesYear, CustNo, ItemNo, Item, Total,
       Lag(Total)  Over(Order By Total Desc)  as "Previous",
       Lead(Total) Over(Order By Total Desc)  as "Next"
   from SalesVWYear a
   Where SalesYear = 2009
   Order By SalesYear, Total Desc
```

SALESYEAR	CUSTNO	ITEMNO	ITEM	TOTAL	Previous	Next
2009	10003	5400	King,Stephen - Shining	3736,76	-	2837,50
2009	10005	5400	King,Stephen - Shining	2837,50	3736,76	1374,35
2009	10001	5100	King,Stephen - Es	1374,35	2837,50	1191,75
2009	10004	5300	Grisham,John - Die Akte	1191,75	1374,35	1146,35
2009	10002	5400	King,Stephen - Shining	1146,35	1191,75	896,65
2009	10001	5300	Grisham,John - Die Akte	896,65	1146,35	853,10
2009	10003	5200	King,Stephen - Drei	853,10	896,65	771,80

- **LAG(Total):** **Returns the TOTAL of the previous row within the window**
- **LEAD(Total):** **Returns the TOTAL of the next row within the window**

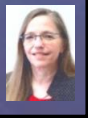
Ordered OLAP Specifications
LAG and LEAD Functions – Example 2

```
With x as (Select Year(SalesDate) SalesYear, CustNo, Sum(Amount) Total
             From Sales
             Group By Year(SalesDate), CustNo),
     y as (Select SalesYear, CustNo,
                  Lag(Total, 2, 0) Over(Partition By CustNo Order By SalesYear)  as PrvPrvYear,
                  Lag(Total, 1, 0) Over(Partition By CustNo Order By SalesYear)  as PrvYear,
                  Total
             from x)
    Select SalesYear, CustNo,
           PrvPrvYear "2008", PrvYear "2009", PrvYear - PrvPrvYear "2009 - 2008",
                              Total   "2010", Total - PrvPrvYear   "2010 - 2008",
                                              Total - PrvYear      "2010 - 2009"
      From y
      Where SalesYear = 2010
      Order By CustNo, Salesyear
```

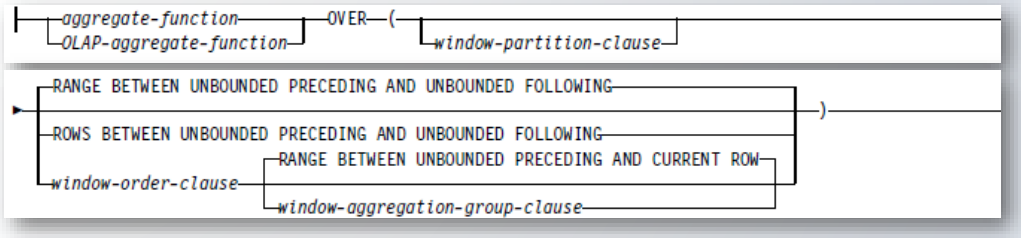
SALESYEAR	CUSTNO	2008	2009	2009 - 2008	2010	2010 - 2008	2010 - 2009
2010	10001	115,00	2634,20	2519,20	281,94	166,94	-2352,26
2010	10003	535,00	4589,86	4054,86	1555,75	1020,75	-3034,11
2010	10006	0,00	0,00	0,00	19425,70	19425,70	19425,70

- Return the values of the Total Column within the 2 preceding rows
- NULL Values within the preceding rows are replaced with 0
- Calculate the differences between the sales from 3 different years

OLAP – Aggregation Specifications



Aggregation OLAP Specifications



Supports the following Aggregate Functions

Functions before Release 7.3		Functions New Release 7.3	
		Statistic Functions	Regression Functions
AVG()	StdDev()	Correlation()	Regr_Count()
Count()	StdDev_Samp()	Covariance()	Regr_Intercept()
Count_Big()	Variance()	Covariance_Samp()	Regr_R2()
Max()	Variance_Samp()	Median()	Regr_Slope()
Min()		Percentile_Cont()	Regr_AVGX()
Sum()		Percentile_Disc()	Regr_AVGY()
			Regr_SXX()
			Regr_SXY()
			Regr_SYY()

Aggregation OLAP Specifications with empty Window-Aggregation-Group-Clause

```
Select  SalesYear, CustNo, CustName1, Amount,
        Sum(Amount) Over() "Sum Part",
        Avg(Amount) Over() "Avg Part",
        Max(Amount) Over() "Maximum Part",
        Min(Amount) Over() "Minmum Part",
        Sum(Amount) * 100,00 / Sum(Amount) Over() "% Year"
  From SalesCustY
  Where SalesYear = 2009
  Group By Grouping Sets((SalesYear, CustNo, CustName1, Amount), ())
  Order By SalesYear, CustNo;
```

SALESYEAR	CUSTNO	CUSTNAME1	AMOUNT	Sum Part	Avg Part	Maximum Part	Minmum Part
2009	10001 ...	Fruits, Vegetables & Co...	2634,20	15276,21	3055,24	4589,86	1636,25
2009	10002 ...	Herrmann & Bauer GmbH ...	1636,25	15276,21	3055,24	4589,86	1636,25
2009	10003 ...	Goldbach GmbH ...	4589,86	15276,21	3055,24	4589,86	1636,25
2009	10004 ...	CSP GmbH ...	2673,95	15276,21	3055,24	4589,86	1636,25
2009	10005 ...	Alzenauer Dönertreff ...	3741,95	15276,21	3055,24	4589,86	1636,25

- Empty Over() Clause: Aggregation over all rows

Aggregation OLAP Specifications without Window-Aggregation-Group-Clause

```
Select  SalesYear, CustNo, CustName1, Amount,
        Sum(Amount) Over(Partition By SalesYear) "Sum Part",
        Avg(Amount) Over(Partition By SalesYear) "Avg Part",
        Max(Amount) Over(Partition By SalesYear) "Maximum Part",
        Min(Amount) Over(Partition By SalesYear) "Minmum Part"
  From SalesCustY
  Order By SalesYear, CustNo
```

SALESYEAR	CUSTNO	CUSTNAME1	AMOUNT	Sum Part	Avg Part	Maximum Part	Minmum Part
2008	10001	Fruits, Vegetables & Co	115,00	2780,00	556,00	1350,00	115,00
2008	10002	Herrmann & Bauer GmbH	1350,00	2780,00	556,00	1350,00	115,00
2008	10003	Goldbach GmbH	535,00	2780,00	556,00	1350,00	115,00
2008	10004	The Company	470,00	2780,00	556,00	1350,00	115,00
2008	10005	Alzenauer Dönertreff	310,00	2780,00	556,00	1350,00	115,00
2009	10001	Fruits, Vegetables & Co	2634,20	15276,21	3055,24	4589,86	1636,25
2009	10002	Herrmann & Bauer GmbH	1636,25	15276,21	3055,24	4589,86	1636,25
2009	10003	Goldbach GmbH	4589,86	15276,21	3055,24	4589,86	1636,25
2009	10004	The Company	2673,95	15276,21	3055,24	4589,86	1636,25
2009	10005	Alzenauer Dönertreff	3741,95	15276,21	3055,24	4589,86	1636,25
2010	10001	Fruits, Vegetables & Co	281,94	1837,69	918,84	1555,75	281,94
2010	10003	Goldbach GmbH	1555,75	1837,69	918,84	1555,75	281,94

- **Neither** the Window-Aggregation-Group Clause **nor** the Window-Order-Clause is specified
  → **All rows** within the partition are accumulated

Aggregation OLAP Specifications - Rolling Results

```
Select  SalesYear, CustNo, Amount,
        Sum(Amount) Over(Order By Amount desc) "Rolling Sum",
        Avg(Amount) Over(Order By Amount Desc) "Rolling Avg",
        Max(Amount) Over(Order By Amount Desc) "Rolling Maximum",
        Min(Amount) Over(Order By Amount Desc) "Rolling Minmum"
From SalesCustY
Where SalesYear = 2009
Order BY SalesYear, Amount Desc
```

SALESYEAR	CUSTNO	AMOUNT	Rolling Sum	Rolling Avg	Rolling Maximum	Rolling Minmum
2009	10003	4589,86	4589,86	4589,86	4589,86	4589,86
2009	10005	3741,95	8331,81	4165,90	4589,86	3741,95
2009	10004	2673,95	11005,76	3668,58	4589,86	2673,95
2009	10001	2634,20	13639,96	3409,99	4589,86	2634,20
2009	10002	1636,25	15276,21	3055,24	4589,86	1636,25

- **Without** Window-Aggregate-Group-Clause but **with** an Window-Order-Clause → **Rolling Results**
- Compute rolling sum, rolling average, rolling maximum, rolling minimum of the sales accumulated for each customer for the year 2009

Aggregation OLAP Specifications
Window-Aggregation-Group – Rows - Example

```
with SalesPlan as (Select * from (Values(2008, 2500,00 ),
                                        (2009, 15000,00),
                                        (2010, 20000,00)) x (SalesYear, Plan)),
     SalesCust as (Select SalesYear, CustNo, Sum(Total) TotalCustYear
                     From SalesVwYear
                     Group By SalesYear, CustNo),
     Totals    as (Select s.SalesYear, CustNo, TotalCustYear, Plan,
                     Sum(TotalCustYear) Over(Partition By s.SalesYear)                    "Total Year",
                     Sum(TotalCustYear) Over(Partition By s.SalesYear Order By CustNo)    "Rolling Sum",
                     Sum(TotalCustYear) Over(Partition By s.SalesYear Order By CustNo) - Plan "Rolling Diff Plan"
                   From SalesCust s join SalesPlan p on s.SalesYear = p.SalesYear)
Select SalesYear, CustNo,  "Total Year",
       TotalCustYear,      TotalCustYear       * 100,0000 / "Total Year" "% Year",
       "Rolling Sum",      "Rolling Sum"       * 100,0000 / "Total Year" "% Rolling",
       Plan,               TotalCustYear       * 100,0000 / Plan         "% Plan",
                           "Rolling Sum"       * 100,0000 / Plan         "% Rolling Plan",
       "Rolling Diff Plan","Rolling Diff Plan" * 100,0000 / Plan         "% Rolling Diff Plan"
  from Totals
  Order By SalesYear, CustNo;
```

- Target Figures
- Accumulated Sales Customer and Year
  - Total Year
  - Rolling Total/Year
  - Rolling Diff. Plan
- Calculate Percent

SALES YEAR	CUSTNO	Total Year	TOTALCUSTYEAR	% Year	Rolling Sum	% Rolling	PLAN	% Plan	% Rolling Plan	Rolling Diff Plan	% Rolling Diff Plan
2008	10001	2780,00	115,00	4,1366	115,00	4,1366	2500,00	4,6000	4,6000	-2385,00	-95,4000
2008	10002	2780,00	1350,00	48,5611	1465,00	52,6978	2500,00	54,0000	58,6000	-1035,00	-41,4000
2008	10003	2780,00	535,00	19,2446	2000,00	71,9424	2500,00	21,4000	80,0000	-500,00	-20,0000
2008	10004	2780,00	470,00	16,9064	2470,00	88,8489	2500,00	18,8000	98,8000	-30,00	-1,2000
2008	10005	2780,00	310,00	11,1510	2780,00	100,0000	2500,00	12,4000	111,2000	280,00	11,2000
2009	10001	15276,21	2634,20	17,2438	2634,20	17,2438	15000,00	17,5613	17,5613	-12365,80	-82,4386
2009	10002	15276,21	1636,25	10,7110	4270,45	27,9549	15000,00	10,9083	28,4696	-10729,55	-71,5303
2009	10003	15276,21	4589,86	30,0458	8860,31	58,0007	15000,00	30,5990	59,0687	-6139,69	-40,9312
2009	10004	15276,21	2673,95	17,5040	11534,26	75,5047	15000,00	17,8263	76,8950	-3465,74	-23,1049
2009	10005	15276,21	3741,95	24,4952	15276,21	100,0000	15000,00	24,9463	101,8414	276,21	1,8414
2010	10001	21263,39	281,94	1,3259	281,94	1,3259	20000,00	1,4097	1,4097	-19718,06	-98,5903
2010	10003	21263,39	1555,75	7,3165	1837,69	8,6425	20000,00	7,7787	9,1884	-18162,31	-90,8115
2010	10006	21263,39	19425,70	91,3574	21263,39	100,0000	20000,00	97,1285	106,3169	1263,39	6,3169

OLAP Aggregate Functions

Functions returning a single value from a window

- `FIRST_VALUE()` — **First Row** in an OLAP window
- `LAST_VALUE()` — **Last Row** in an OLAP window
- `NTH_VALUE()`
  - `NTH_VALUE() FROM FIRST` — **Nth Row** in an OLAP window beginning with the **first row**
  - `NTH_VALUE() FROM LAST` — **Nth Row** in an OLAP window beginning with the **last row**
- `RATIO_TO_REPORT()` — **Ratio** of an argument **to the sum** of the arguments in an OLAP window

OLAP Aggregate Functions
First_Value(), Last_Value(), Nth_Value()

```
Select  SalesYear, Int(CustNo) CustNo, Trim(CustName1)CustName, Amount,
        Int(First_Value(CustNo)  Over(Partition By SalesYear Order By Amount desc)) "First Value",
        Int(Last_Value(CustNo)   Over(Partition By SalesYear))                       "Last Value" ,
        Int(Nth_Value(CustNo, 4) Over(Partition By SalesYear))                       "4th Value",
        Int(Nth_Value(CustNo, 4) From Last Over(Partition By SalesYear))             "4th From Last"
From SalesCustY
Order BY SalesYear, Amount Desc  ;
```

SALESYEAR	CUSTNO	CUSTNAME	AMOUNT	First Value	Last Value	4th Value	4th From Last
2008	10002	Herrmann & Bauer GmbH	1350,00	10002	10001	10005	10003
2008	10003	Goldbach GmbH	535,00	10002	10001	10005	10003
2008	10004	CSP GmbH	470,00	10002	10001	10005	10003
2008	10005	Alzenauer Dönertreff	310,00	10002	10001	10005	10003
2008	10001	Fruits, Vegetables & Co	115,00	10002	10001	10005	10003
2009	10003	Goldbach GmbH	4589,86	10003	10002	10001	10005
2009	10005	Alzenauer Dönertreff	3741,95	10003	10002	10001	10005
2009	10004	CSP GmbH	2673,95	10003	10002	10001	10005
2009	10001	Fruits, Vegetables & Co	2634,20	10003	10002	10001	10005
2009	10002	Herrmann & Bauer GmbH	1636,25	10003	10002	10001	10005
2010	10003	Goldbach GmbH	1555,75	10003	10001	-	-
2010	10001	Fruits, Vegetables & Co	281,94	10003	10001	-	-

OLAP Aggregate Functions - Ratio_To_Report ()

```
Select SalesYear, CustNo, CustName1, Amount,
       Ratio_To_Report(Amount) Over(Partition By SalesYear
                                    Order By Amount Desc)          "Ratio Rolling",
       Dec(Amount, 11, 2) / Sum(Amount) Over(Partition By SalesYear
                                    Order By Amount Desc)          "% Rolling",
       Ratio_To_Report(Amount) Over(Partition By SalesYear)        "Ratio Amount",
       Dec(Amount, 11, 2) / Sum(Amount) Over(Partition By SalesYear) "% Amount"
  From SalesCustY
  Order By SalesYear, Amount Desc;;
```

SALESYEAR	CUSTNO	CUSTNAME1	AMOUNT	Ratio Rolling	% Rolling	Ratio Amount	% Amount
2008	10002	Herrmann & Bauer GmbH	1350,00	1	1,00000...	0,48561...	0,48561...
2008	10003	Goldbach GmbH	535,00	0,28381...	0,28381...	0,19244...	0,19244...
2008	10004	The Company	470,00	0,19957...	0,19957...	0,16906...	0,16906...
2008	10005	Alzenauer Dönertreff	310,00	0,11632...	0,11632...	0,11151...	0,11151...
2008	10001	Fruits, Vegetables & Co	115,00	0,04136...	0,04136...	0,04136...	0,04136...

1350,00 / 1350,00	1,00000000000000000000000000
535,00 / (1350,00 + 535,00)	0,28381962864721485411140583
470,00 / (1350,00 + 535,00 + 470,00)	0,19957537154989384288747346
310,00 / (1350,00 + 535,00 + 470,00 + 310,00)	0,11632270168855534709193245
115,00 / (1350,00 + 535,00 + 470,00 + 310,00 + 115,00)	0,04136690647482014388489208

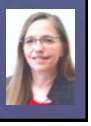
OLAP Aggregate Functions - Ratio_To_Report ()

```
Select SalesYear, CustNo, CustName1, Amount,
       Ratio_To_Report(Amount) Over(Partition By SalesYear
                                    Order By Amount Desc)            "Ratio Rolling",
       Dec(Amount, 11, 2) / Sum(Amount) Over(Partition By SalesYear
                                         Order By Amount Desc)       "% Rolling",
       Ratio_To_Report(Amount) Over(Partition By SalesYear)          "Ratio Amount",
       Dec(Amount, 11, 2) / Sum(Amount) Over(Partition By SalesYear) "% Amount"
  From SalesCustY
  Order By SalesYear, Amount Desc;;
```

SALESYEAR	CUSTNO	CUSTNAME1	AMOUNT	Ratio Rolling	% Rolling	Ratio Amount	% Amount
2008	10002	Herrmann & Bauer GmbH	1350,00	1	1,00000...	0,48561...	0,48561...
2008	10003	Goldbach GmbH	535,00	0,28381...	0,28381...	0,19244...	0,19244...
2008	10004	The Company	470,00	0,19957...	0,19957...	0,16906...	0,16906...
2008	10005	Alzenauer Dönertreff	310,00	0,11632...	0,11632...	0,11151...	0,11151...
2008	10001	Fruits, Vegetables & Co	115,00	0,04136...	0,04136...	0,04136...	0,04136...

```
1350,00 / (1350,00 + 535,00 + 470,00 + 310,00 + 115,00)  0,485611510791366906474820 10
 535,00 / (1350,00 + 535,00 + 470,00 + 310,00 + 115,00)  0,192446043165467625899280 57
 470,00 / (1350,00 + 535,00 + 470,00 + 310,00 + 115,00)  0,169064748201438848920863 30
 310,00 / (1350,00 + 535,00 + 470,00 + 310,00 + 115,00)  0,111510791366906474820143 88
 115,00 / (1350,00 + 535,00 + 470,00 + 310,00 + 115,00)  0,041366906474820143884892 08
```

Any Questions?



Special Thanks to

Holger Scherer – RZKH Rechenzentrum Kreuznach

- For providing an IBM i-System enabling the creation of the samples/code used in my presentations
- http://www.rzkh.de

Your data is save! … in the bunker

References

IBM i information center

- SQL Reference
  https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_74/db2/rbafzpdf.pdf?view=kc
- PDF Files for Database
  https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_74/rzatd/rzatdprintable.htm
- Database Information Finder
  https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_74/rzatd/rzatdfinder.htm

Redbooks

- SQL Procedures, Triggers, and Functions on IBM DB2 for i
  http://www.redbooks.ibm.com/abstracts/sg248326.html?Open

Speaker‘s Biography

Birgitta Hauser
Diplom-Betriebswirt (BA)
Database and Software Architect

Birgitta Hauser worked on the IBM i and its predecessors since 1992. She graduated with a business economics diploma, and started programming on the AS/400 in 1992. She worked and works as traditional RPG Programmer but also as Database and Software Engineer, focusing on IBM i application and database modernization.

Currently she is self-employed and works in Consulting and Application and Database Modernization on IBM i and Db2 for i. Since July, 2019 she is occasionally working for Fresche Solutions Inc. (Montréal) as a contractor.

She also works in education as a trainer for RPG and SQL developers.

Since 2002 she has frequently spoken at the COMMON User Groups and other IBM i and Power Conferences in Germany, other European Countries, USA and Canada.

In addition, she is co-author of two IBM Redbooks and also the author of several articles and papers focusing on RPG and SQL for the ITP Verlag (a German publisher), IT Jungle Guru and IBM DeveloperWorks.

In 2015 she received the John Earl Speaker Scholarship Award. In 2018 she received the Al Barsa Memorial Scholarship Award.

IBM Champion 2020

Thank you!

OLAP Specifications – Much more then running numbers!

Now i know!

Birgitta Hauser
Diplom-Betriebswirt (BA)
Database and Software Architect

Hauser@SSS-Software.de