**Report “Lab 6”**

**Anton Tserakhau**

# Analytic Functions - Basic

## Task 01: Create Ad Hoc SQL FIRST\_VALUE, LAST\_VALUE

I have created ad hoc SQL, which will analyze measurement using Analytic Functions FIRST\_VALUE, LAST\_VALUE.

Select cities with minimum and maximum sales for each country in January 2012, add the field with category (minimum or maximum sales), sort by country in alphabetical order.

SELECT country\_name AS country

, city\_name AS city

, TO\_CHAR ( amount

, '9,999,999' )

AS amount

, CASE

WHEN total\_price = max\_val THEN 'City with max value'

WHEN total\_price = min\_val THEN 'City with min value'

END

city\_category

, TO\_CHAR ( total\_price

, '$999,999,999,999' )

AS total\_price

FROM (SELECT t.\*

, FIRST\_VALUE ( total\_price ) OVER (PARTITION BY country\_name) AS max\_val

, LAST\_VALUE ( total\_price ) OVER (PARTITION BY country\_name) AS min\_val

FROM ( SELECT countries.region\_desc AS country\_name

, cities.city\_desc AS city\_name

, SUM ( oper.unit\_amount ) AS amount

, SUM ( oper.total\_price\_dol ) AS total\_price

FROM t\_operations oper

LEFT JOIN t\_restaurants rest

ON oper.restaurant\_id = rest.restaurant\_id

LEFT JOIN t\_restaurant\_types

ON rest.restaurant\_type\_id = t\_restaurant\_types.restaurant\_type\_id

LEFT JOIN t\_dishes dishes

ON dishes.dish\_id = oper.dish\_id

LEFT JOIN t\_dish\_types

ON dishes.dish\_type\_id = t\_dish\_types.dish\_type\_id

LEFT JOIN t\_dish\_cuisines

ON dishes.dish\_cuisine\_id = t\_dish\_cuisines.dish\_cuisine\_id

LEFT JOIN u\_dw\_references.lc\_cities cities

ON rest.restaurant\_geo\_id = cities.geo\_id

LEFT JOIN u\_dw\_references.t\_geo\_object\_links links

ON links.child\_geo\_id = cities.geo\_id

LEFT JOIN u\_dw\_references.cu\_countries countries

ON links.parent\_geo\_id = countries.geo\_id

WHERE TRUNC ( oper.event\_dt

, 'MONTH' ) = TO\_DATE ( '01-JAN-2012'

, 'DD-MON-YYYY' )

GROUP BY countries.region\_desc

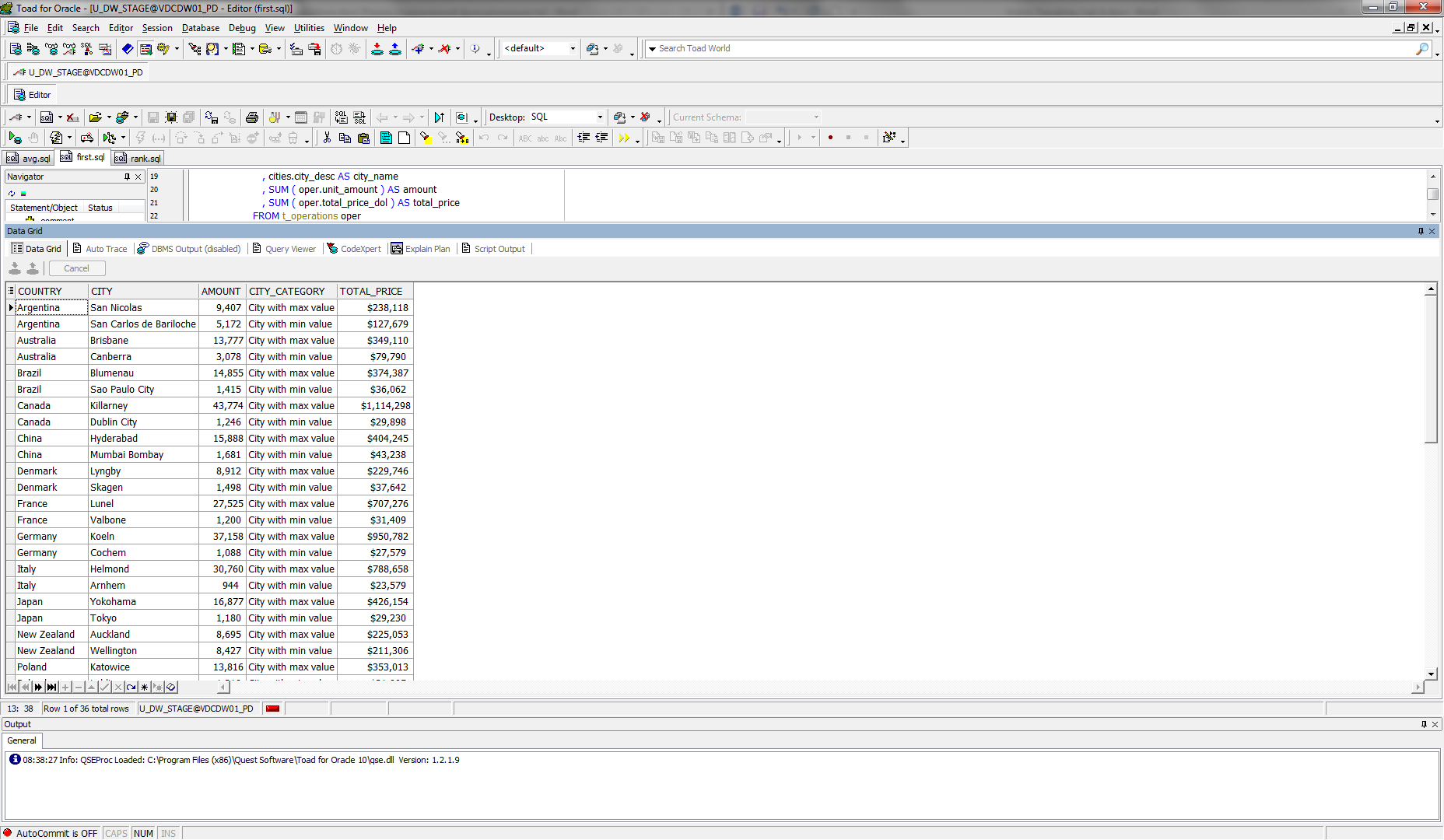
, cities.city\_desc

ORDER BY countries.region\_desc

, total\_price DESC) t)

WHERE total\_price = max\_val

OR total\_price = min\_val;



## Task 02: Create Ad Hoc SQL RANK, DENSE\_RANK, ROWNUM

I have created ad hoc SQL, which will analyze measurement using Analytic Functions RANK, DENSE\_RANK, ROWNUM.

Select TOP-3 cities with the highest sales by country in January 2012, add the fields indicating the ranking of sales by country and sales ranking over the world, sort by country in alphabetical order.

SELECT \*

FROM (SELECT country\_name AS country

, city\_name AS city

, DENSE\_RANK ( ) OVER (PARTITION BY country\_name ORDER BY total\_price DESC) AS rating\_sales\_in\_country

, RANK ( ) OVER (ORDER BY total\_price DESC) AS rating\_sales\_in\_world

, TO\_CHAR ( amount

, '9,999,999' )

AS amount

, TO\_CHAR ( total\_price

, '$999,999,999,999' )

AS total\_price

FROM ( SELECT countries.region\_desc AS country\_name

, cities.city\_desc AS city\_name

, SUM ( oper.unit\_amount ) AS amount

, SUM ( oper.total\_price\_dol ) AS total\_price

FROM t\_operations oper

LEFT JOIN t\_restaurants rest

ON oper.restaurant\_id = rest.restaurant\_id

LEFT JOIN t\_restaurant\_types

ON rest.restaurant\_type\_id = t\_restaurant\_types.restaurant\_type\_id

LEFT JOIN t\_dishes dishes

ON dishes.dish\_id = oper.dish\_id

LEFT JOIN t\_dish\_types

ON dishes.dish\_type\_id = t\_dish\_types.dish\_type\_id

LEFT JOIN t\_dish\_cuisines

ON dishes.dish\_cuisine\_id = t\_dish\_cuisines.dish\_cuisine\_id

LEFT JOIN u\_dw\_references.lc\_cities cities

ON rest.restaurant\_geo\_id = cities.geo\_id

LEFT JOIN u\_dw\_references.t\_geo\_object\_links links

ON links.child\_geo\_id = cities.geo\_id

LEFT JOIN u\_dw\_references.cu\_countries countries

ON links.parent\_geo\_id = countries.geo\_id

WHERE TRUNC ( oper.event\_dt

, 'MONTH' ) = TO\_DATE ( '01-JAN-2012'

, 'DD-MON-YYYY' )

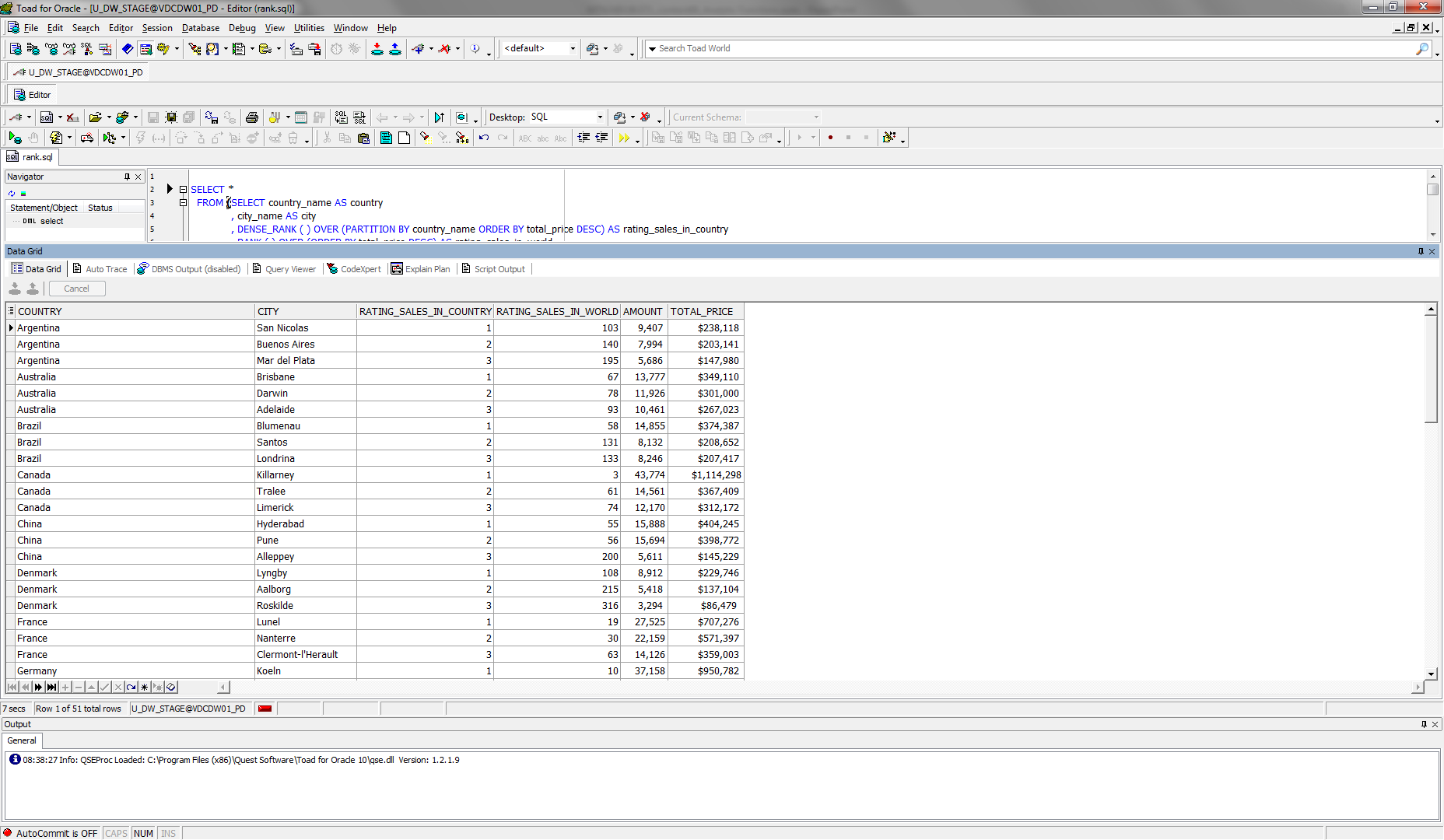
GROUP BY countries.region\_desc

, cities.city\_desc

ORDER BY countries.region\_desc

, cities.city\_desc))

WHERE rating\_sales\_in\_country <= 3;



## Task 03: Create Ad Hoc SQL AGGREAGATE FUNCS

I have created ad hoc SQL, which will analyze measurement using Analytic Functions AGGREAGATE FUNCS (MAX, MIN, AVG).

Select cities with sales for each country in January 2012, add the fields showing minimum / average / maximum sales by country, sort by country in alphabetical order.

SELECT country\_name AS country

, city\_name AS city

, TO\_CHAR ( amount

, '9,999,999' )

AS amount

, TO\_CHAR ( min\_sales

, '$999,999,999,999' )

AS min\_sales

, TO\_CHAR ( avg\_sales

, '$999,999,999,999' )

AS avg\_sales

, TO\_CHAR ( max\_sales

, '$999,999,999,999' )

AS max\_sales

, TO\_CHAR ( total\_price

, '$999,999,999,999' )

AS total\_price

FROM ( SELECT DECODE ( GROUPING ( countries.region\_desc ), 1, 'All Countries', countries.region\_desc ) AS country\_name

, DECODE ( GROUPING ( cities.city\_desc ), 1, 'All Cities', cities.city\_desc ) AS city\_name

, SUM ( oper.unit\_amount ) AS amount

, MIN ( SUM ( oper.total\_price\_dol ) ) OVER (PARTITION BY countries.region\_desc) AS min\_sales

, AVG ( SUM ( oper.total\_price\_dol ) ) OVER (PARTITION BY countries.region\_desc) AS avg\_sales

, MAX ( SUM ( oper.total\_price\_dol ) ) OVER (PARTITION BY countries.region\_desc) AS max\_sales

, SUM ( oper.total\_price\_dol ) AS total\_price

FROM t\_operations oper

LEFT JOIN t\_restaurants rest

ON oper.restaurant\_id = rest.restaurant\_id

LEFT JOIN t\_restaurant\_types

ON rest.restaurant\_type\_id = t\_restaurant\_types.restaurant\_type\_id

LEFT JOIN t\_dishes dishes

ON dishes.dish\_id = oper.dish\_id

LEFT JOIN t\_dish\_types

ON dishes.dish\_type\_id = t\_dish\_types.dish\_type\_id

LEFT JOIN t\_dish\_cuisines

ON dishes.dish\_cuisine\_id = t\_dish\_cuisines.dish\_cuisine\_id

LEFT JOIN u\_dw\_references.lc\_cities cities

ON rest.restaurant\_geo\_id = cities.geo\_id

LEFT JOIN u\_dw\_references.t\_geo\_object\_links links

ON links.child\_geo\_id = cities.geo\_id

LEFT JOIN u\_dw\_references.cu\_countries countries

ON links.parent\_geo\_id = countries.geo\_id

WHERE TRUNC ( oper.event\_dt

, 'MONTH' ) = TO\_DATE ( '01-JAN-2012'

, 'DD-MON-YYYY' )

GROUP BY ROLLUP ( countries.region\_desc, cities.city\_desc )

ORDER BY countries.region\_desc

, cities.city\_desc);

