Assignment 01

Team 06

Qinyang Wu, st174540@stud.Uni-Stuttgart.de, 3519174
Huicheng Qian, st169665@stud.uni-stuttgart.de, 3443114
Kuang-Yu Li, st169971@stud.uni-stuttgart.de, 3440829

Task-1

Result table

DEPARTMENT_DESC	TYPE_DESC	99_REG_SLS_DLR	00_REG_SLS_DLR	01_REG_SLS_DLR	99-00-INC-PERCENT	00-01-INC-PERCENT
Mens Clothing	Retail Only	475785.17	473453.49	473453.49	0	0
Sporting Goods	Retail Only	14653.99	16897.10	16897.10	15	0
Womens Clothing	Retail Only	66494.18	79666.53	79666.53	19	0
Mens Clothing	Strip Malls	299935.20	299815.93	299815.93	0	0
Sporting Goods	Strip Malls	5511.10	6392.74	6392.74	15	0
Womens Clothing	Strip Malls	88123.96	98943.96	98943.96	12	0
Mens Clothing	Super Malls	428566.10	426949.48	426949.48	0	0
Sporting Goods	Super Malls	19160.50	24891.11	24891.11	29	0
Womens Clothing	Super Malls	162642.88	204878.12	204878.12	25	0
Mens Clothing	Wholesale Only	287606.89	287401.09	287401.09	0	0
Sporting Goods	Wholesale Only	10496.85	10164.36	10164.36	-3	0
Womens Clothing	Wholesale Only	29939.68	39706.47	40206.81	32	1

SQL statement

```
select t1.department_desc, t1.type_desc, t1.turnover as "99_reg_sls_dlr", t2.turnover as
"00_reg_sls_dlr", t3.turnover as "01_reg_sls_dlr", 100*(t2.turnover - t1.turnover) /
t1.turnover as "99-00-inc-percent", 100*(t3.turnover - t2.turnover) / t2.turnover as
"00-01-inc-percent"
from (
    select I.department_desc, O.type_desc, sum(reg_sls_dlr) as turnover
    from sales S, time T, item I, store O
    where S.CUR_TRN_DATE = T.CUR_TRN_DATE AND T.YEAR_KEY = 1999 AND
S.ITEM_KEY = I.ITEM_KEY AND S.STORE_KEY = O.STORE_KEY
    group by I.department_desc, O.type_desc
    ) as t1,
    (select I.department_desc, O.type_desc, sum(reg_sls_dlr) as turnover
    from sales S, time T, item I, store O
```

```
    where S.CUR_TRN_DATE = T.CUR_TRN_DATE AND T.YEAR_KEY = 2000 AND
S.ITEM_KEY = I.ITEM_KEY AND S.STORE_KEY = O.STORE_KEY
    group by I.department_desc, O.type_desc) as t2,
    (select I.department_desc, O.type_desc, sum(reg_sls_dlr) as turnover
    from sales S, time T, item I, store O
    where S.CUR_TRN_DATE = T.CUR_TRN_DATE AND T.YEAR_KEY = 2001 AND
S.ITEM_KEY = I.ITEM_KEY AND S.STORE_KEY = O.STORE_KEY
    group by I.department_desc, O.type_desc) as t3
where t1.department_desc = t2.department_desc and t1.type_desc = t2.type_desc and
t1.department_desc = t3.department_desc and t1.type_desc = t3.type_desc;
```

Explanation

Group reg_sls_dlr by department_desc and type_desc to calculate sum of each combination to create three tables for year 1999, 2000, 2001 respectively. Then , use department_desc and type_desc as unique key to calculate differences and increase.

Task-2

Result table

MANAGER	CLASS	TOT_SLS_DLR	RANK
Jim Manager	1	51079.00	4
Jim Manager	2	56297.00	2
Jim Manager	3	84588.00	1
Jim Manager	4	50230.00	5
Jim Manager	5	55007.00	3
Jim Manager	6	48930.00	6
Jim Manager	7	35421.00	7
Joe Manager	1	58450.00	4
Joe Manager	2	63125.00	3
Joe Manager	3	88374.00	1
Joe Manager	4	58189.00	5
Joe Manager	5	69033.00	2
Joe Manager	6	57188.00	6
Joe Manager	7	39926.00	7

SQL statement

```
select manager, class, sum(tot_sls_dlr) as tot_sls_dlr, row_number() over
                                        (partition by manager
                                         order by sum(tot_sls_dlr) desc) as rank
from sales T, item I, store O
where t.item_key = i.item_key and t.store_key = o.store_key
group by class, manager
order by manager asc, class asc;
```

Explanation

Basic concept is that we need to generate all the combinations for the manager and class. Based on this table, we then ranked the tot_sls_dlr in a descending order.
First, we rollup tot_sls_dlr on the store and item key. Use this key, we then match for manager and class with the item and store table. Then we used the row_number() function based on the partition by the manager.

Task-3

SQL statement

```
SELECT year_key, date_desc, division_desc, sum_pml_sls_dlr, AVG(sum_pml_sls_dlr)
OVER (
   PARTITION BY year_key, division_desc
   ORDER BY trn_date
   ROWS 31 PRECEDING
   ) AS avg_up_to_date
FROM(
   SELECT year_key, F.CUR_TRN_DATE as trn_date, date_desc, division_desc,
sum(pml_sls_dlr) as sum_pml_sls_dlr
   FROM sales F, time T, store S, item I
   WHERE F.cur_trn_date = T.cur_trn_date and F.store_key = S.store_key and
F.item_key = I.item_key
   GROUP BY  year_key, division_desc, F.CUR_TRN_DATE, date_desc);
```

Explanation

First, we created a table from rollup by year key, division description and date. Based on this table, we used the function WINDOW introduced in lecture OLAP. By setting window framing to 31, we could get the moving average from day-1 to day-31. Then reset the window after

another partition of the year and division combination. In order to show the column date_desc in an ascending order according to day, not Lexicology order of string, we order the window partition by key CUR_TRN_DATE instead of date_desc.
partial result is shown below:

YEAR_KEY	DATE_DESC	DIVISION_DESC	SUM_PML_SLS_DLR	AVG_UP_TO_DATE
1999	1.12.1999 00:00:00	Athletics	71005.65	71005.65
1999	2.12.1999 00:00:00	Athletics	83763.28	77384.46
1999	3.12.1999 00:00:00	Athletics	61930.80	72233.24
1999	4.12.1999 00:00:00	Athletics	78788.09	73871.95
1999	5.12.1999 00:00:00	Athletics	47727.12	68642.98
1999	6.12.1999 00:00:00	Athletics	60124.71	67223.27
1999	7.12.1999 00:00:00	Athletics	38764.94	63157.79
1999	8.12.1999 00:00:00	Athletics	83760.18	65733.09
1999	9.12.1999 00:00:00	Athletics	68933.28	66088.67
1999	10.12.1999 00:00:00	Athletics	68847.50	66364.55
1999	11.12.1999 00:00:00	Athletics	97831.58	69225.19
1999	12.12.1999 00:00:00	Athletics	84887.35	70530.37
1999	13.12.1999 00:00:00	Athletics	108659.98	73463.42
1999	14.12.1999 00:00:00	Athletics	76630.16	73689.61
1999	15.12.1999 00:00:00	Athletics	97341.34	75266.39
1999	16.12.1999 00:00:00	Athletics	61681.69	74417.35
1999	17.12.1999 00:00:00	Athletics	126404.21	77475.40
1999	18.12.1999 00:00:00	Athletics	98253.66	78629.75
1999	19.12.1999 00:00:00	Athletics	91079.76	79285.01
1999	20.12.1999 00:00:00	Athletics	125878.46	81614.68

Task-4

SQL statement

```
select month_desc, state_desc, style, sum(reg_sls_qty) as "reg_sls_qty"
from sales F, time T, store S, item I
where F.cur_trn_date = T.cur_trn_date and F.store_key = S.store_key and F.item_key =
I.item_key and (T.year_key = 2000 or T.year_key = 2001)
group by rollup(month_desc, state_desc, style);
```

To get all results, we used roll up ton dimension month_desc, state_desc, style with the constraint year on 2000, 2001

To consider all grouping combinations, we just need to use cube for listed attributes dimension

```
select month_desc, state_desc, style, sum(reg_sls_qty) as "reg_sls_qty"
from sales F, time T, store S, item I
where F.cur_trn_date = T.cur_trn_date and F.store_key = S.store_key and F.item_key =
I.item_key and (T.year_key = 2000 or T.year_key = 2001)
group by cube(month_desc, state_desc, style);
```

Result Table

four different grouping combinations have to be considered:

MONTH_DESC	STATE_DESC	STYLE	REG_SLS_QTY
NULL	NULL	NULL	66428.00
December	NULL	NULL	66428.00
December	CT	NULL	32858.00
December	MA	NULL	33570.00
December	CT	Spring/Fall	10510.00
December	CT	Summer	852.00
December	CT	Winter	21496.00
December	MA	Spring/Fall	9036.00
December	MA	Summer	748.00
December	MA	Winter	23786.00

all possible grouping combinations:

MONTH_DESC	STATE_DESC	STYLE	REG_SLS_QTY
NULL	CT	Spring/Fall	10510.00
NULL	CT	Summer	852.00
NULL	CT	Winter	21496.00
NULL	CT	NULL	32858.00
NULL	MA	Spring/Fall	9036.00
NULL	MA	Summer	748.00
NULL	MA	Winter	23786.00
NULL	MA	NULL	33570.00
NULL	NULL	Spring/Fall	19546.00
NULL	NULL	Summer	1600.00
NULL	NULL	Winter	45282.00
NULL	NULL	NULL	66428.00
December	CT	NULL	32858.00
December	MA	NULL	33570.00
December	NULL	NULL	66428.00
December	NULL	Spring/Fall	19546.00
December	NULL	Summer	1600.00
December	NULL	Winter	45282.00
December	CT	Spring/Fall	10510.00
December	MA	Spring/Fall	9036.00
December	CT	Summer	852.00
December	MA	Summer	748.00
December	CT	Winter	21496.00
December	MA	Winter	23786.00

Calculation: How many grouping combinations do exist for n dimension attributes.

Total of 2^n combinations of groupings exist.