Rollup:

تحدثنا في الدرس الماضي عن مثال الـ Sale History الموجود في Oracle Scheme فهو Star Scheme حيث أن الـ Fact Table هو Sales ويحوي فيه مجموعة foreign key لـ Dimensions (Time , Customer , Product , Channel) ، و Tow Measure هما Quantity و Amount.

مثال :

```
SELECT channels.channel_desc, times.calendar_month_desc,
countries.country_iso_code,
TO_CHAR(SUM(amount_sold), '9,999,999,999') SALES$
FROM sales, customers, times, channels, countries
WHERE sales.time_id  = times.time_id
AND sales.cust_id    = customers.cust_id
AND sales.channel_id = channels.channel_id
AND channels.channel_desc IN ('Direct Sales', 'Internet')
AND times.calendar_month_desc IN('2000-09', '2000-10')
AND customers.country_id = countries.country_id
AND countries.country_iso_code IN ('US','FR')
GROUP BY (channels.channel_desc, calendar_month_desc,
countries.country_iso_code);
```

بتنفيذ الاستعلام السابق نحصل على مجموع مبيعات Internet و Direct Sales والتي حدثت في US , FR ضمن شهر 9-2000 وشهر 10- 2000 وستكون مجموعة حسب Channel , Month , Country

	CHANNEL_DESC	CALENDAR_MONTH_DESC	COUNTRY_ISO_CODE	SALES$
1	Direct Sales	2000-09	US	638,201
2	Direct Sales	2000-09	FR	61,202
3	Direct Sales	2000-10	US	682,297
4	Direct Sales	2000-10	FR	44,157
5	Internet	2000-10	US	137,054
6	Internet	2000-10	FR	5,391
7	Internet	2000-09	US	124,224
8	Internet	2000-09	FR	9,597

نفوم بتعديل الاستعلام السابق وذلك بإضافة Rollup لـ Group by فتصبح كالتالي :

```
GROUP BY ROLLUP (channels.channel_desc, calendar_month_desc,
countries.country_iso_code);
```

عندما نضيف Rollup لـ Group By فلن يحصل Aggregation على الـ Channel ثم Month ثم Country فقط بل سيتم Aggregation للثلاثة ، ثم Aggregation على مستوى الأول والثاني باسقاط الـ Country ، ثم يسقط الثاني والثالث ويقوم بالـ Aggregation على مستوى الأول فقط (Channel) ثم يسقط الجميع .

This query returns the following sets of aggregation levels:

-level (expr1, expr2, expr3)

-level (expr1, expr2)

-level (expr1)

-level (grand total)

	CHANNEL_DESC	CALENDAR_MONTH_DESC	COUNTRY_ISO_CODE	SALES$
1	Internet	2000-09	FR	9,597
2	Internet	2000-09	US	124,224
3	Internet	2000-09	(null)	133,821
4	Internet	2000-10	FR	5,391
5	Internet	2000-10	US	137,054
6	Internet	2000-10	(null)	142,445
7	Internet	(null)	(null)	276,266
8	Direct Sales	2000-09	FR	61,202
9	Direct Sales	2000-09	US	638,201
10	Direct Sales	2000-09	(null)	699,403
11	Direct Sales	2000-10	FR	44,157
12	Direct Sales	2000-10	US	682,297
13	Direct Sales	2000-10	(null)	726,454
14	Direct Sales	(null)	(null)	1,425,857
15	(null)	(null)	(null)	1,702,123

نلاحظ أن النتيجة أصبحت 15 سطراً وفي الـ Query السابقة كانت 8 أسطر فقط.

باستخدام الـ Rollup باستعلام الواحد اختصرنا أكثر من استعلام حيث أننا سنحتاج لـ 4 استعلامات لنحصل على مثل هذه النتيجة حيث نحتاج إلى Aggregation على (Channel , Month , Country) وآخر على (Channel , Month) وآخر على Channel وأخيراً Sum.

نلاحظ في النتيجة التي حصلنا عليها أن السطر الثالث لم يظهر في الاستعلام الأول حيث أنه ينتج عن Aggregation لـ Month و Channel دون الـ Country ، وتكون قيمة Sales فيه هي مجموع القيمتين التي قبلها.

ثم في السطر السابع أسقطنا الـ Month و Country وفي السطر الأخير Sum للجميع.

وإذا نفذنا نفس الاستعلام السابق ولكن بإخراج الـ Channel من الـ Rollup :

```
GROUP BY channels.channel_desc , ROLLUP (calendar_month_desc,
countries.country_iso_code);
```

نلاحظ أن نتيجته 14 سطراً والفرق بينه وبين الاستلام السابق أن الـ Channel ستبقى ثابتة دائماً بالتالي لن ينتج لدينا السطر الأخير والذي كان عبارة عن Sum لجميع القيم.

Cube :

وهو Option آخر بدل الـ Rollup وهو أشمل من الـ Rollup :

```
GROUP BY CUBE (channels.channel_desc, calendar_month_desc,
countries.country_iso_code);
```

تقوم Group by Cube بجلب جميع احتمالات التقاطعات أي تقوم بعمل Rollup ولكن بشكل أوسع

This query returns the following sets of aggregation levels:

-level (expr1, expr2, expr3)

-level (expr1, expr2)

-level (expr1, expr3)

-level (expr2, expr3)

-level (expr1)

-level (expr2)

-level (expr3)

- level (grand total)

CHANNEL_DESC	CALENDAR_MONTH_DESC	COUNTRY_ISO_CODE	SALES$
			1,702,123
		FR	120,347
		US	1,581,775
	2000-09		833,224
	2000-09	FR	70,799
	2000-09	US	762,425
	2000-10		868,899
	2000-10	FR	49,548
	2000-10	US	819,351
Internet			276,266
Internet		FR	14,988
Internet		US	261,278
Internet	2000-09		133,821
Internet	2000-09	FR	9,597
Internet	2000-09	US	124,224
Internet	2000-10		142,445
Internet	2000-10	FR	5,391
Internet	2000-10	US	137,054
Direct Sales			1,425,857
Direct Sales		FR	105,360
Direct Sales		US	1,320,497
Direct Sales	2000-09		699,403
Direct Sales	2000-09	FR	61,202
Direct Sales	2000-09	US	638,201
Direct Sales	2000-10		726,454
Direct Sales	2000-10	FR	44,157
Direct Sales	2000-10	US	682,297

عند التنفيذ سيظهر 27 سطراً ، قام بـ Group By على أساس Month, Channel, Country

أولاً Channel Month Country

ثم Channel Month

ثم Channel Country

ثم Month Country

ثم Month

ثم Channel

ثم Country

ثم Sum

أي كل احتمالات تقاطعات هذه الـ Expression

يمكن أيضاً إخراج أحد هذه الـ Expression خارج الـ Cube مثلما فعلنا بالـ Rollup وهنا سنخرج الـ Channel

This query returns the following sets of aggregation levels:

-level (expr1, expr2, expr3)

-level (expr1, expr2)

-level (expr1, expr3)

-level (expr1)

```
GROUP BY channel_desc, CUBE(calendar_month_desc,
countries.country_iso_code);
```

CHANNEL_DESC	CALENDAR_MONTH_DESC	COUNTRY_ISO_CODE	SALES$
Internet			276,266
Internet		FR	14,988
Internet		US	261,278
Internet	2000-09		133,821
Internet	2000-09	FR	9,597
Internet	2000-09	US	124,224
Internet	2000-10		142,445
Internet	2000-10	FR	5,391
Internet	2000-10	US	137,054
Direct Sales			1,425,857
Direct Sales		FR	105,360
Direct Sales		US	1,320,497
Direct Sales	2000-09		699,403
Direct Sales	2000-09	FR	61,202
Direct Sales	2000-09	US	638,201
Direct Sales	2000-10		726,454
Direct Sales	2000-10	FR	44,157
Direct Sales	2000-10	US	682,297

Cube with Grouping Function:

وهو Function مساعد يستخدم لتعبئة الفراغ الذي يظهر عند التنفيذ ، يستخدم لتحسين التقرير أي اسمه Grouping و يأخذ اسم حقل فيعيد قيمة 1 في حال يتم التجميع على الحقل في هذا السطر و 0 في عدم التجميع عليه ولدينا تابع الـ Decode يقوم بفك الترميز حسب أول قيمة ممرة له مع ثاني قيمة .

في المثال سنضع القيمة Multi-Channel Sum اذا كانت القيمة 1 والا اسم الـ Channel .

```
SELECT
DECODE(GROUPING(channels.channel_desc), 1, 'Multi-channel sum', channels.channel_desc) AS
Channel,
DECODE (GROUPING (times.calendar_month_desc), 1, 'Multi-times sum', times.calendar_month_desc)
AS Times,
DECODE (GROUPING (countries.country_iso_code), 1, 'Multi-country sum', countries.country_iso_code)
AS Country,
TO_CHAR(SUM(amount_sold), '9,999,999,999') SALES$
FROM sales, customers, times, channels, countries
WHERE sales.time_id  = times.time_id
AND sales.cust_id   = customers.cust_id
AND sales.channel_id = channels.channel_id
AND channels.channel_desc IN ('Direct Sales', 'Internet')
AND times.calendar_month_desc IN('2000-09', '2000-10')
AND customers.country_id = countries.country_id
AND countries.country_iso_code IN ('US','FR')
GROUP BY CUBE (channels.channel_desc, calendar_month_desc, countries.country_iso_code);
```

CHANNEL	TIMES	COUNTRY	SALES$
Multi-channel sum	Multi-times sum	Multi-country sum	1,702,123
Multi-channel sum	Multi-times sum	FR	120,347
Multi-channel sum	Multi-times sum	US	1,581,775
Multi-channel sum	2000-09	Multi-country sum	833,224
Multi-channel sum	2000-09	FR	70,799
Multi-channel sum	2000-09	US	762,425
Multi-channel sum	2000-10	Multi-country sum	868,899
Multi-channel sum	2000-10	FR	49,548
Multi-channel sum	2000-10	US	819,351
Internet	Multi-times sum	Multi-country sum	276,266
Internet	Multi-times sum	FR	14,988
Internet	Multi-times sum	US	261,278
Internet	2000-09	Multi-country sum	133,821
Internet	2000-09	FR	9,597
Internet	2000-09	US	124,224
Internet	2000-10	Multi-country sum	142,445
Internet	2000-10	FR	5,391
Internet	2000-10	US	137,054
Direct Sales	Multi-times sum	Multi-country sum	1,425,857
Direct Sales	Multi-times sum	FR	105,360
Direct Sales	Multi-times sum	US	1,320,497
Direct Sales	2000-09	Multi-country sum	699,403
Direct Sales	2000-09	FR	61,202
Direct Sales	2000-09	US	638,201
Direct Sales	2000-10	Multi-country sum	726,454
Direct Sales	2000-10	FR	44,157
Direct Sales	2000-10	US	682,297

وهو عبارة عن تحسين للإظهار.

✓ هنالك Technique مستخدمة في الـ Warehouse وهو الـ Analytic Function أو Sliding Window .

Sliding window :

هناك حساب آخر في oracle schema وهو hr سنقوم ببعض الـ queries عليه لنتعرف على الـ sliding window.

ملاحظة: يفضل أن نكون connected as sys لكي نستطيع فتح أي schema فقط بإعطائه اسمها لانه administrator وذلك لتجنب disconnect و connect عند الانتقال بين الـ sh & hr.

نستعلم عن معلومات الـ employee hr :

```
SELECT first_name, last_name, salary, department_id
FROM EMPLOYEES;
```

تم عرض معلومات الموظفين (107موظف) ، واذا نفذنا الاستعلام التالي :

```
SELECT AVG(salary)
FROM employees;
```

يقوم بعرض قيمة واحدة تعبر عن المتوسط الحسابي للرواتب .

بتنفيذ الاستعلام التالي:

```
SELECT department_id, AVG(salary)
FROM employees
GROUP BY department_id
ORDER BY department_id;
```

يعرض متوسط رواتب كل قسم من الأقسام

DEPARTMENT_	AVG(SALARY)
10	4400
20	9500
30	4150
40	6500
50	3475.555556
60	5760
70	10000
80	8955.882353
90	19333.33333
100	8601.333333
110	10154
	7000

نريد الآن في استعلام واحد عرض اسم الموظف وراتبه و رقم القسم و لكل موظف متوسط رواتب القسم الذي يعمل به وأعلى وأقل ومجموع راتب بالقسم وعدد موظفين القسم .

لتنفيذ هذا الأمر نحن بحاجة الى sub query بعدد القيم التي نريدها أي 5 sub query إضافة إلى الـ query الأساسية.

لكن نستخدم technique جديد ضمن الـ aggregation function اسمه analytic function sliding window حيث أننا لن نكتفي بـ sum (salary) وإنما sum (salary) over (partition by department_id) أي سنفتح window للمعالجة مباشرة دون الحاجة لوضع group by على مستوى الاستعلام ككل ،ففي حال أردنا استخدام Aggregation لوحده فسنضطر لاستخدام group by أما باستخدام over فنقوم بتحديد المعالجة بعدها مباشرة :

```
SELECT employee_id, first_name, salary, department_id,
SUM(salary) OVER (PARTITION BY department_id) AS sum_dept_sal,
AVG(salary) OVER (PARTITION BY department_id) AS avg_dept_sal,
MAX(salary) OVER (PARTITION BY department_id) AS max_dept_sal,
MIN(salary) OVER (PARTITION BY department_id) AS min_dept_sal,
COUNT(*)  OVER (PARTITION BY department_id) AS num_dept_emp
FROM employees;
```

ننفذ فيعطي لكل موظف راتبه والقسم الذي ينتمي له والـ sum لرواتب الموظفين ضمن هذا القسم ونفس الأمر Avg , Max ,Min , Count

EMPLOYEE_ID	FIRST_NAME	SALARY	DEPARTMENT_ID	SUM_DEPT_SAL	AVG_DEPT_SAL	MAX_DEPT_SAL	MIN_DEPT_SAL	NUM_DEPT_EMP
200	Jennifer	4400	10	4400	4400	4400	4400	1
201	Michael	13000	20	19000	9500	13000	6000	2
202	Pat	6000	20	19000	9500	13000	6000	2
114	Den	11000	30	24900	4150	11000	2500	6
115	Alexander	3100	30	24900	4150	11000	2500	6
116	Shelli	2900	30	24900	4150	11000	2500	6
117	Sigal	2800	30	24900	4150	11000	2500	6
118	Guy	2600	30	24900	4150	11000	2500	6
119	Karen	2500	30	24900	4150	11000	2500	6
203	Susan	6500	40	6500	6500	6500	6500	1
120	Matthew	8000	50	156400	3475.555556	8200	2100	45
121	Adam	8200	50	156400	3475.555556	8200	2100	45
122	Payam	7900	50	156400	3475.555556	8200	2100	45
123	Shanta	6500	50	156400	3475.555556	8200	2100	45
124	Kevin	5800	50	156400	3475.555556	8200	2100	45
125	Julia	3200	50	156400	3475.555556	8200	2100	45
126	Irene	2700	50	156400	3475.555556	8200	2100	45
127	James	2400	50	156400	3475.555556	8200	2100	45
128	Steven	2200	50	156400	3475.555556	8200	2100	45
129	Laura	3300	50	156400	3475.555556	8200	2100	45
130	Mozhe	2800	50	156400	3475.555556	8200	2100	45
131	James	2500	50	156400	3475.555556	8200	2100	45
132	TJ	2100	50	156400	3475.555556	8200	2100	45
133	Jason	3300	50	156400	3475.555556	8200	2100	45
134	Michael	2900	50	156400	3475.555556	8200	2100	45
135	Ki	2400	50	156400	3475.555556	8200	2100	45
136	Hazel	2200	50	156400	3475.555556	8200	2100	45

وهنا نلاحظ بالسطر 2 ، 3 أن sum هو عبارة عن مجموع رواتب الموظفين في نفس القسم.

لهذا الـ window نوعين : إما أن يكون physical او logical

Physical : (ثابت) أي انه قمنا بتحجيمها بتحديد عدد الأسطر التي سننفذ عليها الـ Aggregation :

```
SELECT employee_id, first_name, department_id, salary,
SUM(salary) OVER (PARTITION BY department_id ORDER BY salary
                  ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING) AS sum_3emp_sal
FROM employees
WHERE department_id = 80;
```

EMPLOYEE_ID	FIRST_NAME	DEPARTMENT_ID	SALARY	SUM_3EMP_SAL
173	Sundita	80	6100	12300
179	Charles	80	6200	18500
167	Amit	80	6200	18800
166	Sundar	80	6400	19400
165	David	80	6800	20200
161	Sarath	80	7000	20800
155	Oliver	80	7000	21200
164	Mattea	80	7200	21500
172	Elizabeth	80	7300	21900
171	William	80	7400	22200
154	Nanette	80	7500	22400
160	Louise	80	7500	23000
159	Lindsey	80	8000	23500
153	Christopher	80	8000	24400
177	Jack	80	8400	25000
176	Jonathon	80	8600	25800
175	Alyssa	80	8800	26400
158	Allan	80	9000	26800
152	Peter	80	9000	27500

نجمع الراتب على أساس الأقسام و نرتب القسم الواحد حسب الـ salaray ليتم تجميعه ولكن بدلاً من تجميع القسم الواحد بشكل كامل يتم تجميع n سطر قبل السطر الحالي و m سطر بعده حيث في مثالنا n=1 , m= 1 .

أي يرتب الـ records الناتجة ضمن sliding window حسب الراتب ويتم الجمع لكل row مع الذي قبله والذي بعده.

هذا الترتيب لا علاقة له بالترتيب الذي نكتبه بالـ Query .

في السطر الأول يجمع قيمة الثاني معه ، والأخير يجمع قيمة قبل الأخير معه فقط (الصورة السابقة لا تحوي جميع الأسطر).

إذا قمنا بنفس الاستعلام السابق ولكن بإضافة orderBy firstname على مستوى الاستعلام ككل فلن تكون نتيجتها مطابقة للاستعلام السابق وذلك لأننا وضعنا orderBy خارجية , فسيحدث الترتيب وفقها ووفق التجميعة الداخلية .

يجب الانتباه في هذه الحالة أن يكون كلا الـ orderBy متوافقين أو أن نكون واثقين من صحة الاستعلام الذي نفعله .

يمكن الاستغناء عن Partition في الـ over فسيأخذ كامل الـ Table وإذا فيه Criteria سيأخذ وفقها.

Logical :

```
SELECT employee_id, first_name, department_id, salary, hire_date,
SUM(salary) OVER (PARTITION BY department_id ORDER BY hire_date
                    RANGE BETWEEN UNBOUNDED PRECEDING AND current row) AS
sum_upto_emp_sal
FROM employees
WHERE department_id IN (60, 20, 30);
```

غيرنا الـ OrderBy حسب الـ hair date وبدلنا الـ Row بالـ Range (من جميع الأسطر السابقة إلى السطر الحالي)

هذه تسمى Logical Sliding Window وهي متغيرة وفق الـ Criteria والـ Data .

EMPLOYEE_ID	FIRST_NAME	DEPARTMENT_ID	SALARY	HIRE_DATE	SUM_UPTO_EMP_SAL
201	Michael	20	13000	17/02/2004 0	13000
202	Pat	20	6000	17/08/2005 0	19000
114	Den	30	11000	07/12/2002 0	11000
115	Alexander	30	3100	18/05/2003 0	14100
117	Sigal	30	2800	24/07/2005 0	16900
116	Shelli	30	2900	24/12/2005 0	19800
118	Guy	30	2600	15/11/2006 0	22400
119	Karen	30	2500	10/08/2007 0	24900
105	David	60	4800	25/06/2005 0	4800
103	Alexander	60	9000	03/01/2006 0	13800
106	Valli	60	4800	05/02/2006 0	18600
107	Diana	60	4200	07/02/2007 0	22800
104	Bruce	60	6000	21/05/2007 0	28800

نحن هنا نقسم حسب الـ Department فكلما أختلف القسم سيبدأ التجميع من جديد .

الـ analytic function تدعم كل الـaggregation function وتدعم أيضاً توابع الـ Rank والـ dense rank و First value

: Rank & Dense rank

مثلاً نريد ترتيب الزبائن حسب القوة الشرائية (كمية الشراء)

Rank : يرتب القيم والذي لديه القيمة الأعلى يعطيه 1 والذي يليه 2 و هكذا .

الفرق بينه وبين الـ dense rank هو عندما يكون لدي أكثر من زبون بنفس القيمة الشرائية فيقوم باخذ عددهم بعين الاعتبار بترتيب الزبون التالي لهم ، مثال :

Rank : 6 3 3 3 2 1

dense rank: 4 3 3 3 2 1

```
SELECT department_id, last_name, salary, commission_pct,
RANK() OVER (PARTITION BY department_id ORDER BY salary DESC, commission_pct) "Rank",
DENSE_RANK() OVER (PARTITION BY department_id ORDER BY salary DESC, commission_pct)
"DENSE_RANK"
FROM employees
WHERE department_id = 80;
```

التنفيذ :

DEPARTMENT_ID	LAST_NAME	SALARY	COMMISSION_PCT	Rank	DENSE_RANK
80	Russell	14000	0.4	1	1
80	Partners	13500	0.3	2	2
80	Errazuriz	12000	0.3	3	3
80	Ozer	11500	0.25	4	4
80	Cambrault	11000	0.3	5	5
80	Abel	11000	0.3	5	5
80	Zlotkey	10500	0.2	7	6
80	Vishney	10500	0.25	8	7
80	Bloom	10000	0.2	9	8
80	Tucker	10000	0.3	10	9
80	King	10000	0.35	11	10
80	Fox	9600	0.2	12	11
80	Greene	9500	0.15	13	12
80	Bernstein	9500	0.25	14	13
80	Sully	9500	0.35	15	14
80	Hall	9000	0.25	16	15
80	McEwen	9000	0.35	17	16
80	Hutton	8800	0.25	18	17
80	Taylor	8600	0.2	19	18
80	Livingston	8400	0.2	20	19
80	Olsen	8000	0.2	21	20
80	Smith	8000	0.3	22	21
80	Cambrault	7500	0.2	23	22
80	Doran	7500	0.3	24	23
80	Smith	7400	0.15	25	24
80	Bates	7300	0.15	26	25
80	Marvins	7200	0.1	27	26

80	Tuvault	7000	0.15	28	27
80	Sewall	7000	0.25	29	28
80	Lee	6800	0.1	30	29
80	Ande	6400	0.1	31	30
80	Johnson	6200	0.1	32	31
80	Banda	6200	0.1	32	31
80	Kumar	6100	0.1	34	32

نلاحظ في السطر 7-8 اختلاف القيم .

انتهت المحاضرة

Written by :

Aisha Awaty

Wordpress and preparation :

Anas Alazmeh

Reviewed by :

Mouayyad Taja