SQL Expressive Powers

1. Relational Algebra or Calculus
2. Aggregation / Grouping (Extension)
3. Deductive Logics / **Analytic Functions (Windowing)**
4. Data Mining Features

Data Cube

(Multiple Aggregation: Multi-level & Multi-dimension)

1. Read Section 25.3.1 Rollup and Cube in SQL:1999)
2. Optional Reading: Oracle Data Warehousing Guide

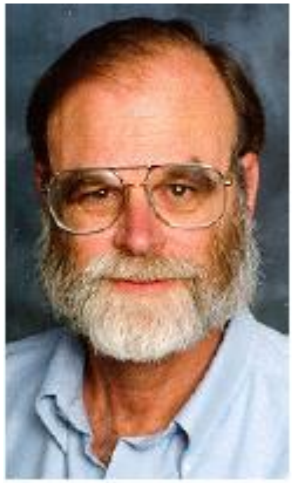
Ch 20. SQL for Aggregation in Data Warehouses

https://docs.oracle.com/en/database/oracle/oracle-database/19/dwhsg/sql-aggregation-data-warehouses.html#GUID-E051A04E-0C53-491D-9B16-B71BA00B80C2

Contents

- The limitations of Group By: Sub-totals, Cross-tab
- New Aggregation Features since Oracle8i
  - CUBE, ROLLUP, etc….

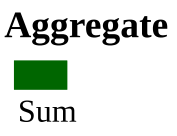
Multiple Aggregations!

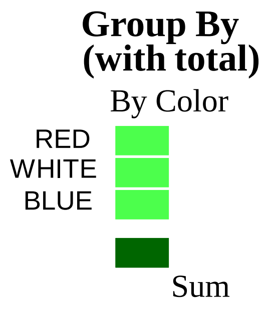


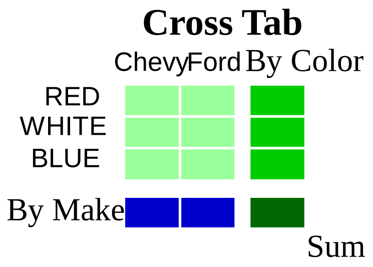
Aggregation, Group-By, Cross-Tab, and Cube

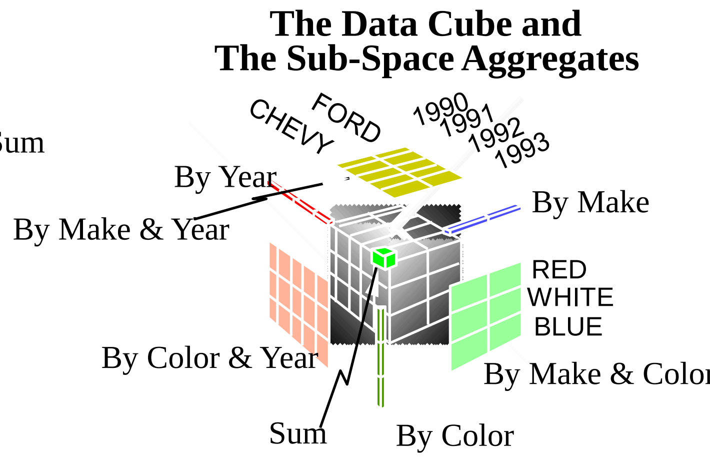
SALES

Model	Year	Color	Amount
Chevy	1990	red	5
Chevy	1990	white	87
Chevy	1990	blue	62
Chevy	1991	red	54
Chevy	1991	white	95
Chevy	1991	blue	49
Chevy	1992	red	31
Chevy	1992	white	54
Chevy	1992	blue	71
Ford	1990	red	64
Ford	1990	white	62
Ford	1990	blue	63
Ford	1991	red	52
Ford	1991	white	9
Ford	1991	blue	55
Ford	1992	red	27
Ford	1992	white	62
Ford	1992	blue	39





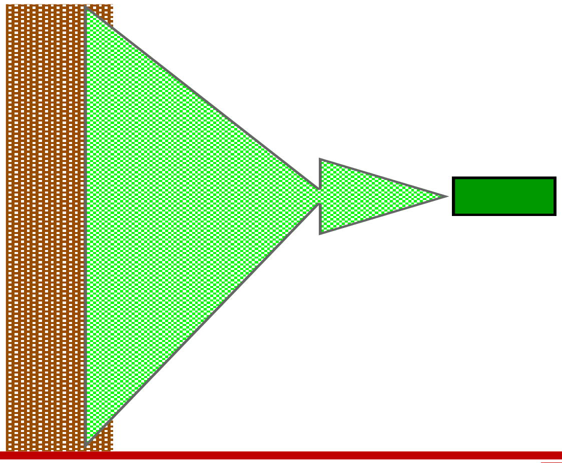




Relational Aggregate Operators

- SQL has several aggregate operators:
  - sum(), min(), max(), count(), avg()
- Other systems extend this with many others:
  - stat functions, financial functions, ...
- The basic idea is:
  - Combine all values in a column into a single scalar value.
- Syntax

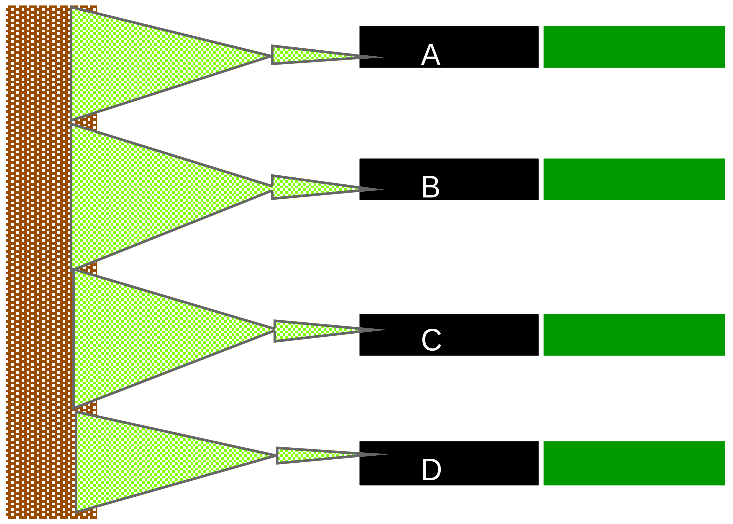
```
select  sum(amount)
from    sales;
```



Relational Group By Operator

- Group By allows aggregates over table sub-groups
- Result is a new table
- Syntax:

```
select      model, sum(amount)
from        sales
group by    model;
```



Aggregates

- Add up amounts by date, product
- In SQL:

```
SELECT date, prodid, SUM(amount) FROM sales
GROUP BY date, prodid
```

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

$\Rightarrow$

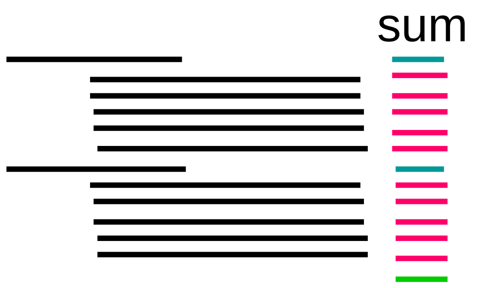
sale	prodId	date	amt
	p1	1	62
	p2	1	19
	p1	2	48

$\xrightarrow{\text{rollup}}$

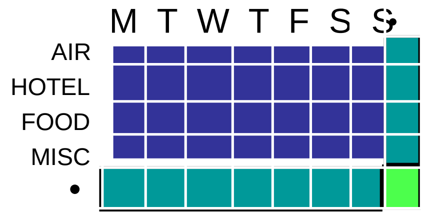
$\xleftarrow{\text{drill-down}}$

Problems With Group-By

- Single-level aggregation



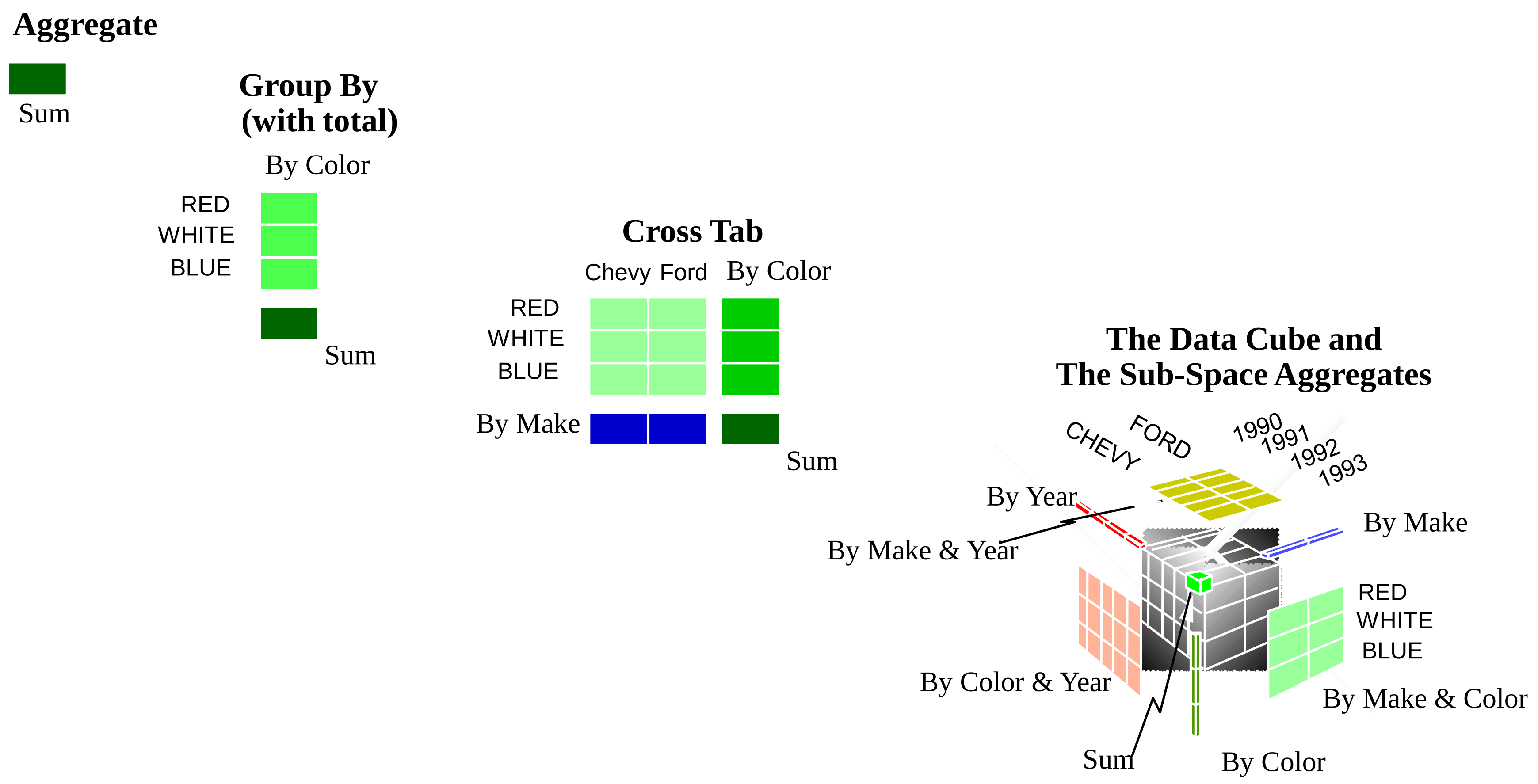
- But, users want
  - Sub-totals and totals: e.g. drill-down & roll-up reports
  - CrossTabs



- Conventional wisdom
  - These are not relational operators
  - They are in many report writers and query engines

Solution: Data CUBE Relational Operator

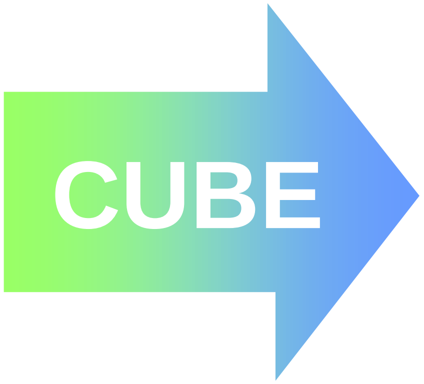
- Data Cube generalizes Group By and Aggregates



Cube

SALES

Model	Year	Color	Amount
Chevy	1990	red	5
Chevy	1990	white	87
Chevy	1990	blue	62
Chevy	1991	red	54
Chevy	1991	white	95
Chevy	1991	blue	49
Chevy	1992	red	31
Chevy	1992	white	54
Chevy	1992	blue	71
Ford	1990	red	64
Ford	1990	white	62
Ford	1990	blue	63
Ford	1991	red	52
Ford	1991	white	9
Ford	1991	blue	55
Ford	1992	red	27
Ford	1992	white	62
Ford	1992	blue	39



DATA CUBE

Model	Year	Color	Amount
ALL	ALL	ALL	942
chevy	ALL	ALL	510
ford	ALL	ALL	432
ALL	1990	ALL	343
ALL	1991	ALL	314
ALL	1992	ALL	285
ALL	ALL	red	165
ALL	ALL	white	273
ALL	ALL	blue	339
chevy	1990	ALL	154
chevy	1991	ALL	199
chevy	1992	ALL	157
ford	1990	ALL	189
ford	1991	ALL	116
ford	1992	ALL	128
chevy	ALL	red	91
chevy	ALL	white	236
chevy	ALL	blue	183
ford	ALL	red	144
ford	ALL	white	133
ford	ALL	blue	156
ALL	1990	red	69
ALL	1990	white	149
ALL	1990	blue	125
ALL	1991	red	107
ALL	1991	white	104
ALL	1991	blue	104
ALL	1992	red	59
ALL	1992	white	116
ALL	1992	blue	110

SELECT model, year, color, sum(amount)
FROM sales
GROUP BY CUBE(mode, year, color);

Rollup

```
SELECT job, deptno, sum(sal)
FROM emp
GROUP BY ROLLUP(deptno,job);
```

```
    DEPTNO JOB         SUM(SAL)
---------- --------- ----------
        10 CLERK           1300
        10 MANAGER         2450
        10 PRESIDENT       5000
        10 ________        8750
        20 CLERK           1900
        20 ANALYST         6000
        20 MANAGER         2975
        20 ________       10875
        30 CLERK            950
        30 MANAGER         2850
        30 SALESMAN        5600
        30 ________        9400
     _____ ________       29025
```

Very
Large
Data
Bases

Rollup(2)

```
SELECT deptno, job sum(sal)
FROM emp
GROUP BY deptno, job

UNION ALL

SELECT deptno, NULL, sum(sal)
FROM emp
GROUP BY deptno

UNION ALL

SELECT NULL, NULL, sum(sal)
FROM emp
```

Cube

```
SELECT deptno, job, sum(sal)
FROM emp
GROUP BY CUBE(deptno,job);
```

```
    DEPTNO JOB         SUM(SAL)
---------- --------- ----------
                          29025
 _____     _________
           CLERK           4150
 _____
           ANALYST         6000
 _____
           MANAGER         8275
 _____
           SALESMAN        5600
 _____
           PRESIDENT       5000
 _____
        10                 8750
           _________
        10 CLERK           1300
        10 MANAGER         2450
        10 PRESIDENT       5000
        20                10875
           _________
        20 CLERK           1900
        20 ANALYST         6000
        20 MANAGER         2975
        30                 9400
           _________
        30 CLERK            950
        30 MANAGER         2850
        30 SALESMAN        5600
```

Grouping Function

- How to distinguish null and all?

```
SELECT deptno, job, sum(sal), grouping(job) as T1
FROM emp
GROUP BY cube(deptno, job);
```

```
    DEPTNO JOB         SUM(SAL)         T1
---------- --------- ---------- ----------
                          29025          1
           CLERK           4150          0
           ANALYST         6000          0
           MANAGER         8275          0
           SALESMAN        5600          0
           PRESIDENT       5000          0
        10                 8750          1
        10 CLERK           1300          0
        10 MANAGER         2450          0
        10 PRESIDENT       5000          0
                                  .........
```

Grouping Function

- How to distinguish null and all? decode + grouping

```
SELECT decode(grouping(deptno),0,to_char(deptno), 'ALL') as deptno,
                    decode(grouping(job),0,job, 'ALL') as job,
                    sum(sal)
FROM emp
GROUP BY cube(deptno, job)
```

```
DEPTNO    JOB         SUM(SAL)
-------- --------- ----------
ALL       ALL            29025
ALL       CLERK           4150
ALL       ANALYST         6000
ALL       MANAGER         8275
ALL       SALESMAN        5600
ALL       PRESIDENT       5000
10        ALL             8750
10        CLERK           1300
10        MANAGER         2450
10        PRESIDENT       5000
                          ……. . .
                           ………
```

Very Large Data Bases

Grouping_ID

```
Aggregation Level      Bit Vector      Grouping_ID
--------------------------------------------------
a,b                    0  0            0
a,ALL                  0  1            1
ALL,b                  1  0            2
ALL, ALL               1  1            3
```

```
SELECT deptno,job, sum(sal),  grouping_id(deptno, job) as GRP_ID
FROM emp
GROUP BY CUBE(deptno, job);
```

```
    DEPTNO JOB         SUM(SAL)     GRP_ID
---------- --------- ---------- ----------
                          29025          3
           CLERK           4150          2
           ANALYST         6000          2
           MANAGER         8275          2
           SALESMAN        5600          2
           PRESIDENT       5000          2
        10                 8750          1
        10 CLERK           1300          0
                                 ……….
```

References

1. Oracle 11G Release 2 Data Warehousing Guide Chapter 21, “SQL for Aggregation in Data Warehouses”

2. Jim Gray et. al., Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Total, International Conference on Data Engineering (ICDE), 1996