Chapter 5

How to code summary queries

Objectives

Applied

- Code summary queries that use aggregate functions, including the use of the ROLLUP and CUBE operators.

Knowledge

- Describe summary queries.
- Describe the differences between the HAVING clause and the WHERE clause.
- Describe the use of the ROLLUP and CUBE operators.

The syntax of the aggregate functions

```
AVG([ALL|DISTINCT] expression)
SUM([ALL|DISTINCT] expression)
MIN([ALL|DISTINCT] expression)
MAX([ALL|DISTINCT] expression)
COUNT([ALL|DISTINCT] expression)
COUNT(*)
```

A summary query

```
SELECT COUNT(*) AS number_of_invoices,
  SUM(invoice_total – payment_total – credit_total)
      AS total_due
FROM invoices
WHERE invoice_total – payment_total – credit_total > 0
```

The result set

	NUMBER_OF_INVOICES	TOTAL_DUE
1	40	66796.24

A summary query with COUNT(*), AVG, and SUM

```
SELECT 'After 1/1/2008' AS selection_date,
    COUNT(*) AS number_of_invoices,
    ROUND(AVG(invoice_total), 2) AS avg_invoice_amt,
    SUM(invoice_total) AS total_invoice_amt
FROM invoices
WHERE invoice_date > '01-JAN-2008'
```

The result set

	SELECTION_DATE	NUMBER_OF_INVOICES	AVG_INVOICE_AMT	TOTAL_INVOICE_AMT
1	After 1/1/2008	114	1879.74	214290.51

A summary query with MIN and MAX functions

```
SELECT 'After 1/1/2008' AS selection_date,
    COUNT(*) AS number_of_invoices,
    MAX(invoice_total) AS highest_invoice_total,
    MIN(invoice_total) AS lowest_invoice_total
FROM invoices
WHERE invoice_date > '01-JAN-2008'
```

The result set

	SELECTION_DATE	NUMBER_OF_INVOICES	HIGHEST_INVOICE_TOTAL	LOWEST_INVOICE_TOTAL
1	After 1/1/2008	114	37966.19	6

A summary query for non-numeric columns

```
SELECT MIN(vendor_name) AS first_vendor,
    MAX(vendor_name) AS last_vendor,
    COUNT(vendor_name) AS number_of_vendors
FROM vendors
```

The result set

	FIRST_VENDOR	LAST_VENDOR	NUMBER_OF_VENDORS
1	ASC Signs	Zylka Design	122

A summary query with the DISTINCT keyword

```
SELECT COUNT(DISTINCT vendor_id) AS number_of_vendors,
    COUNT(vendor_id) AS number_of_invoices,
    ROUND(AVG(invoice_total),2) AS avg_invoice_amt,
    SUM(invoice_total) AS total_invoice_amt
FROM invoices
WHERE invoice_date > '01-JAN-2008'
```

The result set

	NUMBER_OF_VENDORS	NUMBER_OF_INVOICES	AVG_INVOICE_AMT	TOTAL_INVOICE_AMT
1	34	114	1879.74	214290.51

The syntax with GROUP BY and HAVING clauses

```
SELECT select_list
FROM table_source
[WHERE search_condition]
[GROUP BY group_by_list]
[HAVING search_condition]
[ORDER BY order_by_list]
```

A summary query that calculates average invoice amount by vendor

```
SELECT vendor_id,
  ROUND(AVG(invoice_total), 2) AS average_invoice_amount
FROM invoices
GROUP BY vendor_id
HAVING AVG(invoice_total) > 2000
ORDER BY average_invoice_amount DESC
```

The result set

	VENDOR_ID	AVERAGE_INVOICE_AMOUNT
1	110	23978.48
2	72	10963.66
3	104	7125.34
4	99	6940.25
5	119	4901.26
6	122	2575.33
7	86	2433
8	100	2184.5

(8 rows selected)

A summary query that counts the number of invoices by vendor

```
SELECT vendor_id, COUNT(*) AS invoice_qty
FROM invoices
GROUP BY vendor_id
ORDER BY vendor_id
```

The result set

	VENDOR_ID	INVOICE_QTY
1	34	2
2	37	3
3	48	1
4	72	2

```
(34 rows selected)
```

A summary query with a join

```
SELECT vendor_state, vendor_city,
    COUNT(*) AS invoice_qty,
    ROUND(AVG(invoice_total),2) AS invoice_avg
FROM invoices JOIN vendors
    ON invoices.vendor_id = vendors.vendor_id
GROUP BY vendor_state, vendor_city
ORDER BY vendor_state, vendor_city
```

The result set

	VENDOR_STATE	VENDOR_CITY	INVOICE_QTY	INVOICE_AVG
1	AZ	Phoenix	1	662
2	CA	Fresno	19	1208.75
3	CA	Los Angeles	1	503.2
4	CA	Oxnard	3	188

(20 rows selected)

A summary query that limits the groups to those with two or more invoices

```
SELECT vendor_state, vendor_city,
    COUNT(*) AS invoice_qty,
    ROUND(AVG(invoice_total),2) AS invoice_avg
FROM invoices JOIN vendors
    ON invoices.vendor_id = vendors.vendor_id
GROUP BY vendor_state, vendor_city
HAVING COUNT(*) >= 2
ORDER BY vendor_state, vendor_city
```

The result set

	VENDOR_STATE	VENDOR_CITY	INVOICE_QTY	INVOICE_AVG
1	CA	Fresno	19	1208.75
2	CA	Oxnard	3	188
3	CA	Pasadena	5	196.12
4	CA	Sacramento	7	253

```
(12 rows selected)
```

A summary query with a search condition in the HAVING clause

```
SELECT vendor_name, COUNT(*) AS invoice_qty,
    ROUND(AVG(invoice_total),2) AS invoice_avg
FROM vendors JOIN invoices
    ON vendors.vendor_id = invoices.vendor_id
GROUP BY vendor_name
HAVING AVG(invoice_total) > 500
ORDER BY invoice_qty DESC
```

The result set

	VENDOR_NAME	INVOICE_QTY	INVOICE_AVG
1	United Parcel Service	9	2575.33
2	Zylka Design	8	867.53
3	Malloy Lithographing Inc	5	23978.48
4	IBM	2	600.06

(19 rows selected)

A summary query with a search condition in the WHERE clause

```
SELECT vendor_name, COUNT(*) AS invoice_qty,
    ROUND(AVG(invoice_total),2) AS invoice_avg
FROM vendors JOIN invoices
    ON vendors.vendor_id = invoices.vendor_id
WHERE invoice_total > 500
GROUP BY vendor_name
ORDER BY invoice_qty DESC
```

The result set

	VENDOR_NAME	INVOICE_QTY	INVOICE_AVG
1	United Parcel Service	9	2575.33
2	Zylka Design	7	946.67
3	Malloy Lithographing Inc	5	23978.48
4	Ingram	2	1077.21

(20 rows selected)

A summary query with a compound condition in the HAVING clause

```
SELECT
   invoice_date,
   COUNT(*) AS invoice_qty,
   SUM(invoice_total) AS invoice_sum
FROM invoices
GROUP BY invoice_date
HAVING invoice_date
        BETWEEN '01-MAY-2008' AND '31-MAY-2008'
   AND COUNT(*) > 1
   AND SUM(invoice_total) > 100
ORDER BY invoice_date DESC
```

The result set

	INVOICE_DATE	INVOICE_QTY	INVOICE_SUM
1	31-MAY-08	3	11557.75
2	23-MAY-08	6	2761.17
3	22-MAY-08	2	442.5
4	20-MAY-08	3	308.64

```
(15 rows selected)
```

The same query with a WHERE clause

```
SELECT
   invoice_date,
   COUNT(*) AS invoice_qty,
   SUM(invoice_total) AS invoice_sum
FROM invoices
WHERE invoice_date
    BETWEEN '01-MAY-2008' AND '31-MAY-2008'
GROUP BY invoice_date
HAVING COUNT(*) > 1
   AND SUM(invoice_total) > 100
ORDER BY invoice_date DESC
```

The same result set

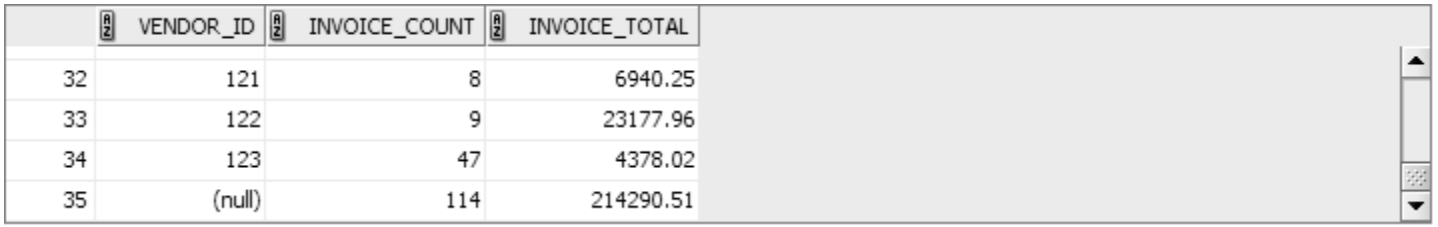
	INVOICE_DATE	INVOICE_QTY	INVOICE_SUM
1	31-MAY-08	3	11557.75
2	23-MAY-08	6	2761.17
3	22-MAY-08	2	442.5
4	20-MAY-08	3	308.64

(15 rows selected)

A summary query with a final summary row

```
SELECT vendor_id, COUNT(*) AS invoice_count,
    SUM(invoice_total) AS invoice_total
FROM invoices
GROUP BY ROLLUP(vendor_id)
```

The result set



	VENDOR_ID	INVOICE_COUNT	INVOICE_TOTAL
32	121	8	6940.25
33	122	9	23177.96
34	123	47	4378.02
35	(null)	114	214290.51

```
(35 rows selected)
```

A summary query with a summary row for each grouping level

```
SELECT vendor_state, vendor_city, COUNT(*) AS qty_vendors
FROM vendors
WHERE vendor_state IN ('IA', 'NJ')
GROUP BY ROLLUP (vendor_state, vendor_city)
ORDER BY vendor_state, vendor_city
```

The result set

	VENDOR_STATE	VENDOR_CITY	QTY_VENDORS
1	IA	Fairfield	1
2	IA	Washington	1
3	IA	(null)	2
4	NJ	East Brunswick	2
5	NJ	Fairfield	1
6	NJ	Washington	1
7	NJ	(null)	4
8	(null)	(null)	6

A summary query with a summary row at the start of the result set

```
SELECT vendor_id, COUNT(*) AS invoice_count,
    SUM(invoice_total) AS invoice_total
FROM invoices
GROUP BY CUBE(vendor_id)
```

The result set

	VENDOR_ID	INVOICE_COUNT	INVOICE_TOTAL
1	(null)	114	214290.51
2	34	2	1200.12
3	37	3	564
4	48	1	856.92

(35 rows selected)

A summary query with a summary row for each set of groups

```
SELECT vendor_state, vendor_city, COUNT(*) AS qty_vendors
FROM vendors
WHERE vendor_state IN ('IA', 'NJ')
GROUP BY CUBE(vendor_state, vendor_city)
ORDER BY vendor_state, vendor_city
```

The result set

	VENDOR_STATE	VENDOR_CITY	QTY_VENDORS
1	IA	Fairfield	1
2	IA	Washington	1
3	IA	(null)	2
4	NJ	East Brunswick	2
5	NJ	Fairfield	1
6	NJ	Washington	1
7	NJ	(null)	4
8	(null)	East Brunswick	2
9	(null)	Fairfield	2
10	(null)	Washington	2
11	(null)	(null)	6