Murach Chapter 6   Part 1

How to Code Subqueries

Week 5

Knowledge Points in this lecture

- Basic Subquery
- JOIN vs Subquery
- Subquery in FROM clause, Top-N query
- Subquery in WHERE clause

Subquery Basics

- Subquery
  - Also called inner query
  - Is a SELECT statement inside another SELECT statement
  - Similar to basic SELECT statement, but normally don’t contain GROUP BY or HAVING clauses
- Main query
  - Also called outer query
  - Is a SELECT statement that contains another SELECT statement

Four ways to introduce a subquery in a SELECT statement

- In a WHERE clause as a search condition
- In a HAVING clause as a search condition
- In the FROM clause as a table specification
- In the SELECT clause as a column specification

- Subqueries in WHERE, HAVING clauses are most common.
- Usually subqueries don’t appear in GROUP BY or ORDER BY.

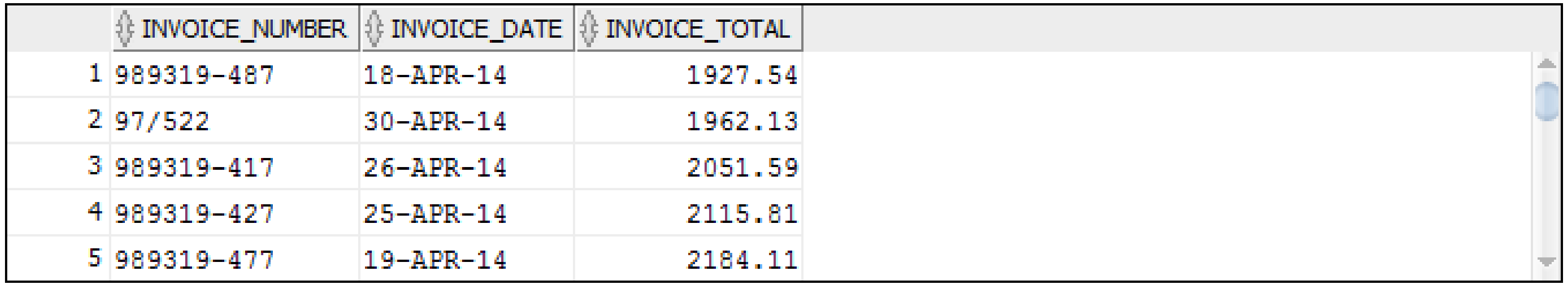
A subquery in a WHERE clause

```
SELECT invoice_number, invoice_date, invoice_total
FROM invoices
WHERE invoice_total >
    (SELECT AVG(invoice_total)
     FROM invoices)
ORDER BY invoice_total
```

The value returned by the subquery

```
1879.7413
```

The result set



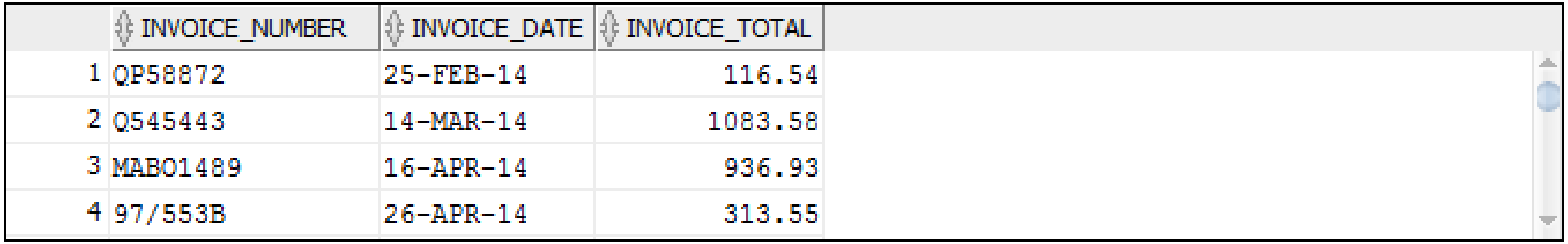
	INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL
1	989319-487	18-APR-14	1927.54
2	97/522	30-APR-14	1962.13
3	989319-417	26-APR-14	2051.59
4	989319-427	25-APR-14	2115.81
5	989319-477	19-APR-14	2184.11

```
(21 rows)
```

A query that uses an inner join

```
SELECT invoice_number, invoice_date, invoice_total
FROM invoices JOIN vendors
    ON invoices.vendor_id = vendors.vendor_id
WHERE vendor_state = 'CA'
ORDER BY invoice_date
```

The result set



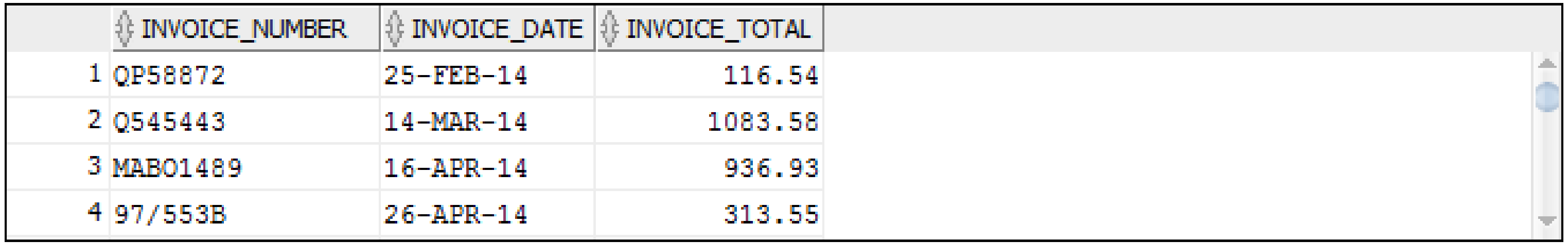
	INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL
1	QP58872	25-FEB-14	116.54
2	Q545443	14-MAR-14	1083.58
3	MAB01489	16-APR-14	936.93
4	97/553B	26-APR-14	313.55

(40 rows)

The same query restated with a subquery

```
SELECT invoice_number, invoice_date, invoice_total
FROM invoices
WHERE vendor_id IN
    (SELECT vendor_id
    FROM vendors
    WHERE vendor_state = 'CA')
ORDER BY invoice_date
```

The same result set



	INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL
1	QP58872	25-FEB-14	116.54
2	Q545443	14-MAR-14	1083.58
3	MAB01489	16-APR-14	936.93
4	97/553B	26-APR-14	313.55

```
(40 rows)
```

Advantages of joins

- A join can include columns from both tables.
- A join is more intuitive when it uses an existing relationship.

Advantages of subqueries

- A subquery can pass an aggregate value to the outer query.
- A subquery is more intuitive when it uses an ad hoc relationship.
- Long, complex queries can be easier to code with subqueries.

The syntax of a WHERE clause that uses an IN phrase with a subquery

```
WHERE test_expression [NOT] IN (subquery)
```

A query that returns vendors without invoices

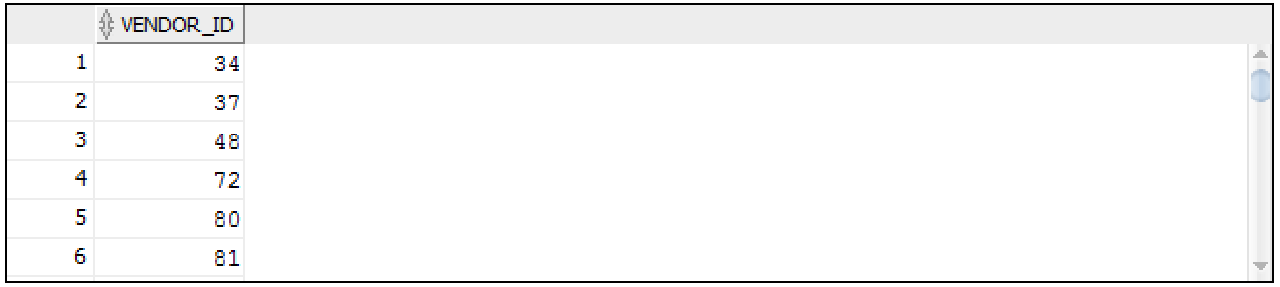
```
SELECT vendor_id, vendor_name, vendor_state
FROM vendors
WHERE vendor_id NOT IN
    (SELECT DISTINCT vendor_id
    FROM invoices)
ORDER BY vendor_id
```

For IN/ NOT IN:

- Subquery must return a single column of values.

Result of Previous Query & its Subquery

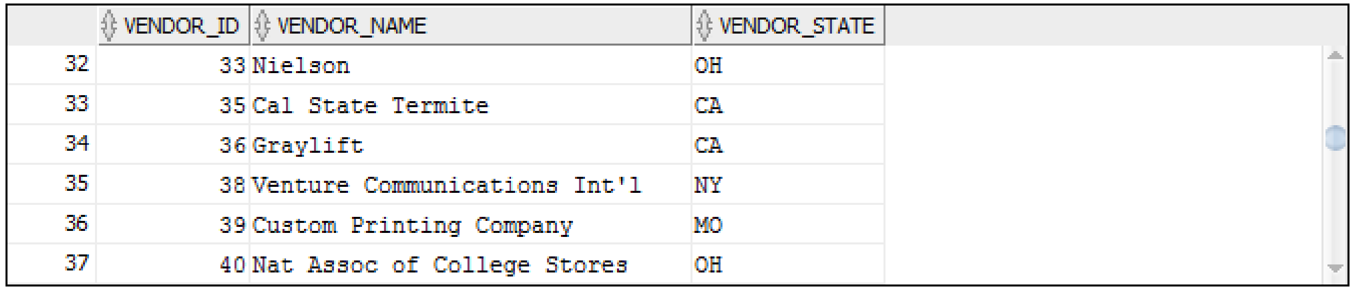
The result of the subquery in previous query



	VENDOR_ID
1	34
2	37
3	48
4	72
5	80
6	81

```
(34 rows)
```

The result set of previous query



	VENDOR_ID	VENDOR_NAME	VENDOR_STATE
32	33	Nielson	OH
33	35	Cal State Termite	CA
34	36	Graylift	CA
35	38	Venture Communications Int'l	NY
36	39	Custom Printing Company	MO
37	40	Nat Assoc of College Stores	OH

```
(88 rows)
```

The query restated without a subquery

```
SELECT v.vendor_id, vendor_name, vendor_state
FROM vendors v LEFT JOIN invoices i
    ON v.vendor_id = i.vendor_id
WHERE i.vendor_id IS NULL
ORDER BY v.vendor_id
```

Subquery in FROM Clause & Top N Query

Example:

Print the information of top 3 largest invoices.

```
SELECT *
FROM   (SELECT invoice_number, invoice_total
        FROM   invoices
        ORDER BY invoice_total DESC)
WHERE ROWNUM <= 3;
```

The syntax of a WHERE clause with an expression that uses the value returned by a subquery

```
WHERE expression comparison_operator
    [SOME|ANY|ALL] (subquery)
```

[A]: A is optional;   A|B: A or B

A query with a subquery in a WHERE condition

```
SELECT invoice_number, invoice_date,
    invoice_total - payment_total - credit_total
        AS balance_due
FROM invoices
WHERE invoice_total - payment_total - credit_total  > 0
    AND invoice_total - payment_total - credit_total <
    (
    SELECT AVG(invoice_total - payment_total - credit_total)
    FROM invoices
    WHERE invoice_total - payment_total - credit_total > 0
    )
ORDER BY invoice_total DESC
```

If no keywords: SOME, ANY, ALL, subquery must return a single value.