WEEK 10
SUBQUERIES

IM101
ADVANCED DATABASE SYSTEMS

LEARNING OUTCOMES:

At the end of the session, the students should be able to:

1. Discuss how to run Subqueries in SQL Developer;
2. Understand where and when to use subqueries;
3. Demonstrate subqueries in SQL Developer.

SUBQUERIES

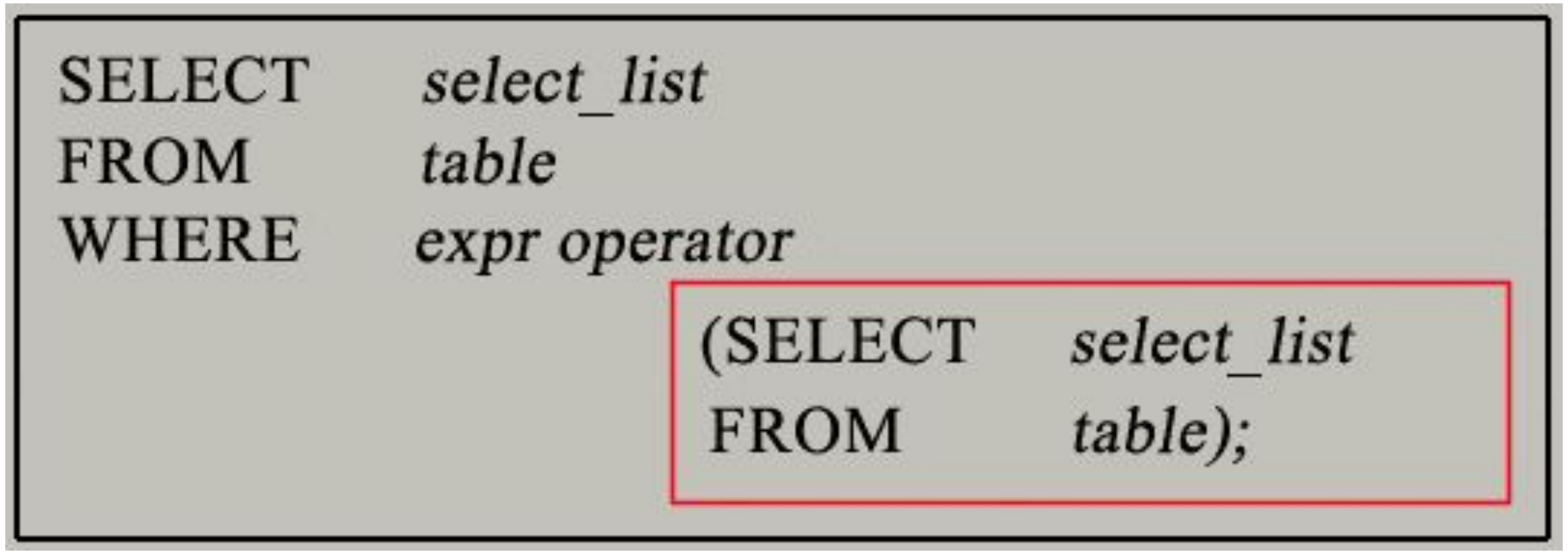
- Integrating data from two tables into one outcome
- May return single number, or multiple rows, depending on how they are used
- An SQL query that nests inside a larger query
  - A subquery may occur in :
    - A SELECT clauses
    - A FROM clauses
    - A WHERE clause
  - In SELECT, INSERT, UPDATE or DELETE
  - Attached to another SQL SELECT w/in WHERE clause

SUBQUERIES

- Comparison operators can be used
  - >
  - <
  - =
- Internal query or internal selection for subquery and external query or external selection for containing subquery.

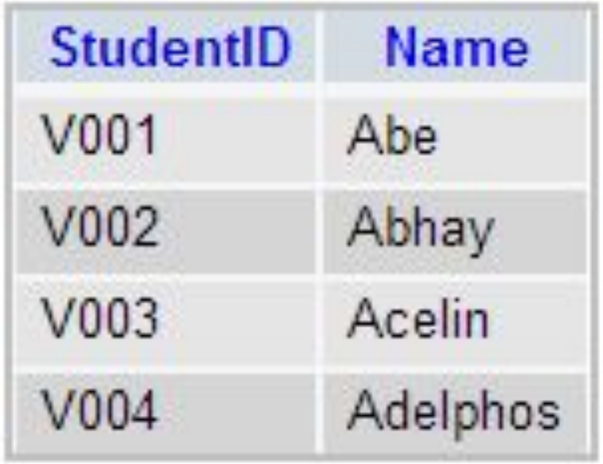
SUBQUERIES

To perform the following tasks:

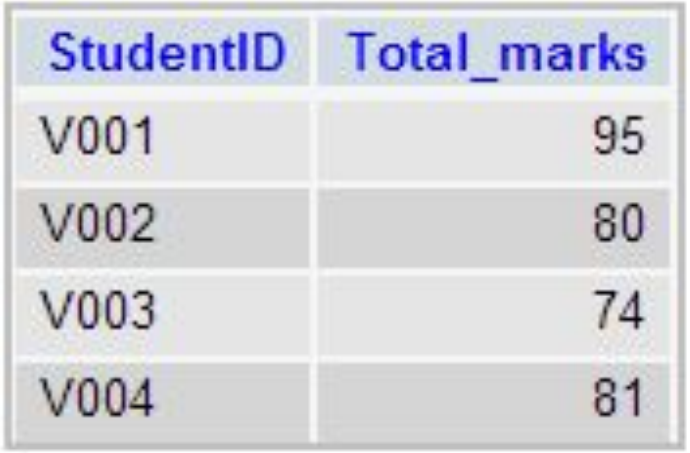


```
SELECT    select_list
FROM      table
WHERE     expr operator
                      (SELECT    select_list
                       FROM      table);
```

HOW TO USE SUBQUERY

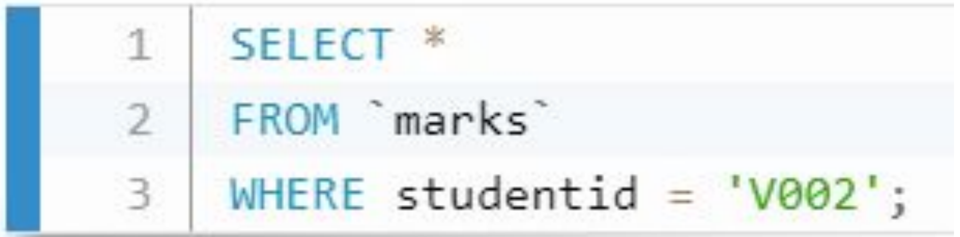


StudentID	Name
V001	Abe
V002	Abhay
V003	Acelin
V004	Adelphos



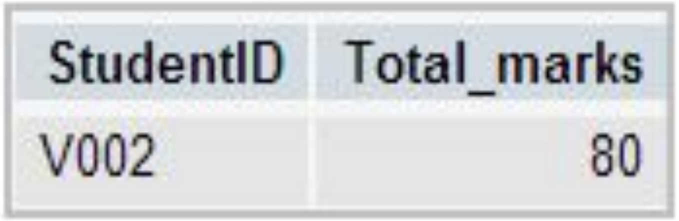
StudentID	Total_marks
V001	95
V002	80
V003	74
V004	81

First query:



```
SELECT *
FROM `marks`
WHERE studentid = 'V002';
```

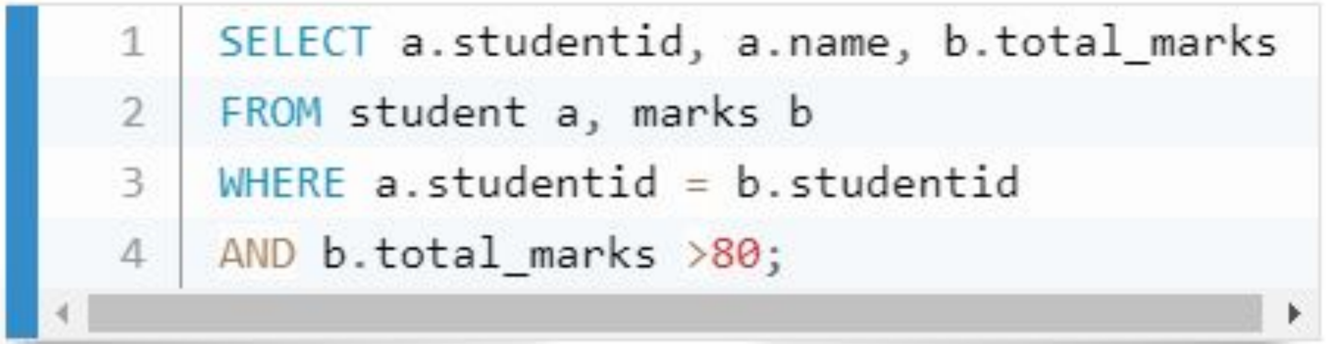
Query result:



StudentID	Total_marks
V002	80

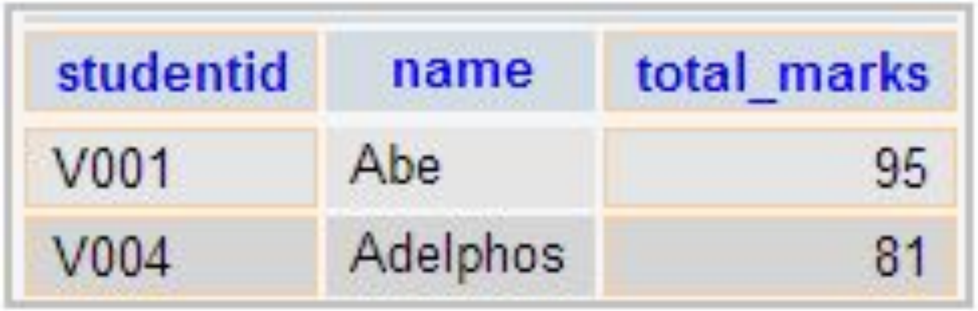
HOW TO USE SUBQUERY

Second query:



```
SELECT a.studentid, a.name, b.total_marks
FROM student a, marks b
WHERE a.studentid = b.studentid
AND b.total_marks >80;
```

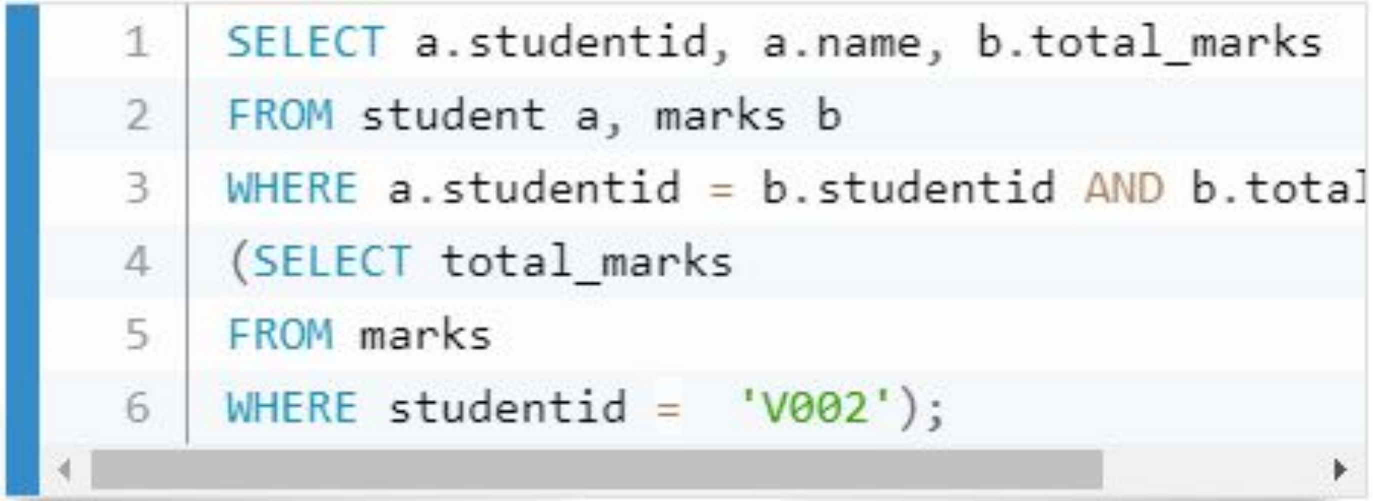
Query result:



studentid	name	total_marks
V001	Abe	95
V004	Adelphos	81

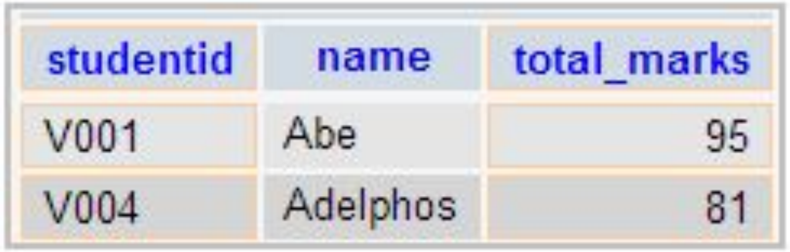
HOW TO USE SUBQUERY

SQL Code:



```
SELECT a.studentid, a.name, b.total_marks
FROM student a, marks b
WHERE a.studentid = b.studentid AND b.total
(SELECT total_marks
FROM marks
WHERE studentid =  'V002');
```

Query result:

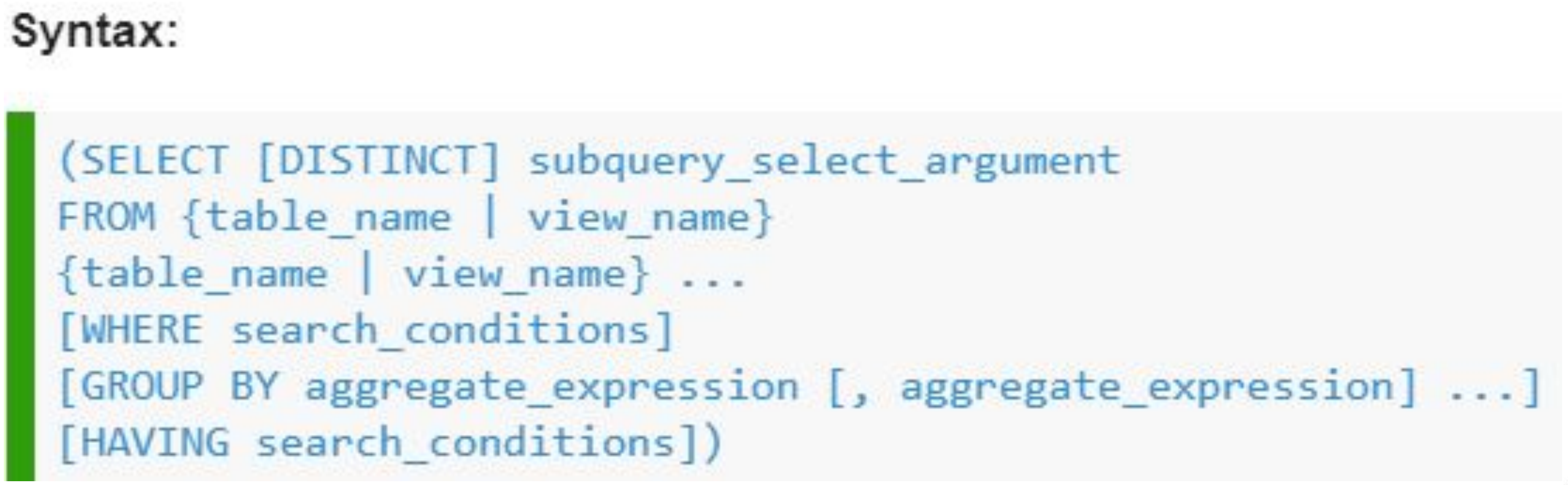


studentid	name	total_marks
V001	Abe	95
V004	Adelphos	81

SUBQUERY RULE

- A SELECT subquery statement is nearly identical to the SELECT statement,

Here is a Subquery syntax



Syntax:

```
(SELECT [DISTINCT] subquery_select_argument
FROM {table_name | view_name}
{table_name | view_name} ...
[WHERE search_conditions]
[GROUP BY aggregate_expression [, aggregate_expression] ...]
[HAVING search_conditions])
```

SUBQUERIES GUIDELINES

When using subqueries there are some guidelines to consider

- Parenthesis may have a subquery
- A subquery must put on the comparison operators right hand side
- Subqueries cannot modify their outcomes internally,
- Using single-row operation with subqueries in single-row
- If a subquery (inner query) returns a null value to the outer query, no rows will be returned by the outer query by using any WHERE clause comparison operators.

SUBQUERIES TYPES

Types of Subqueries

- Single subquery line: returns zero or a row.
- Many subqueries in columns: returns one column or more.
- Correlated subqueries: The outer SQL statement applies to one or more columns.
- Nested subqueries: Put subqueries within another subquery.

SUBQUERIES COMPARE TO JOINS

- Both joins and subqueries are used to merge data from various tables into one single result.
- Subqueries return either single value or row set, while join used to return rows
- Possible to restate most subqueries as joins and most joins can be restated as subqueries

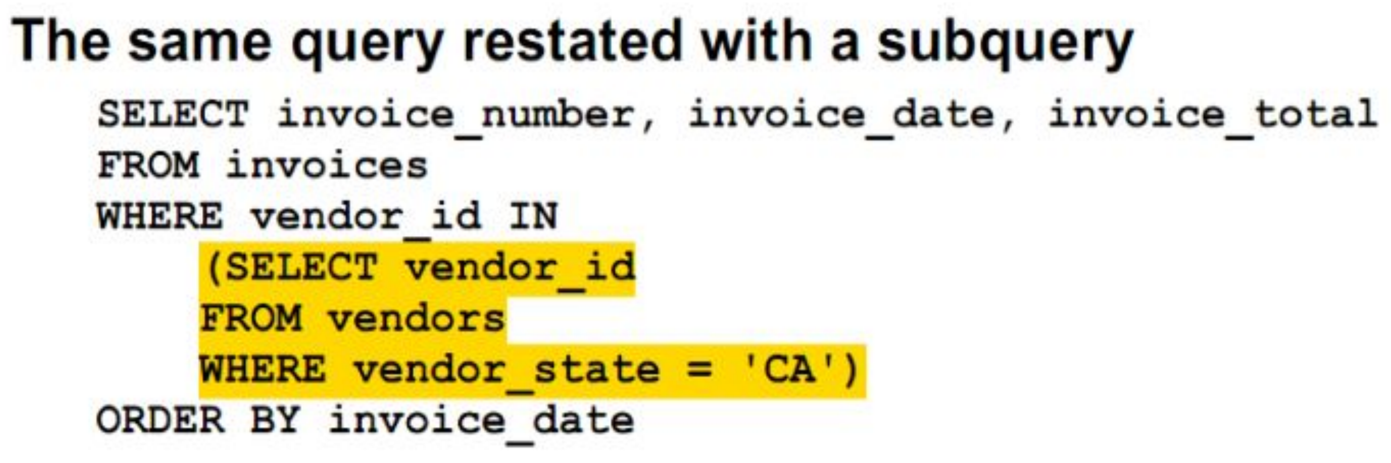
SUBQUERIES COMPARE TO JOINS

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
```

This statement uses a join to combine the Vendors and Invoices tables so the vendor_state column can be tested for each invoice.

SUBQUERIES COMPARE TO JOINS



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
```

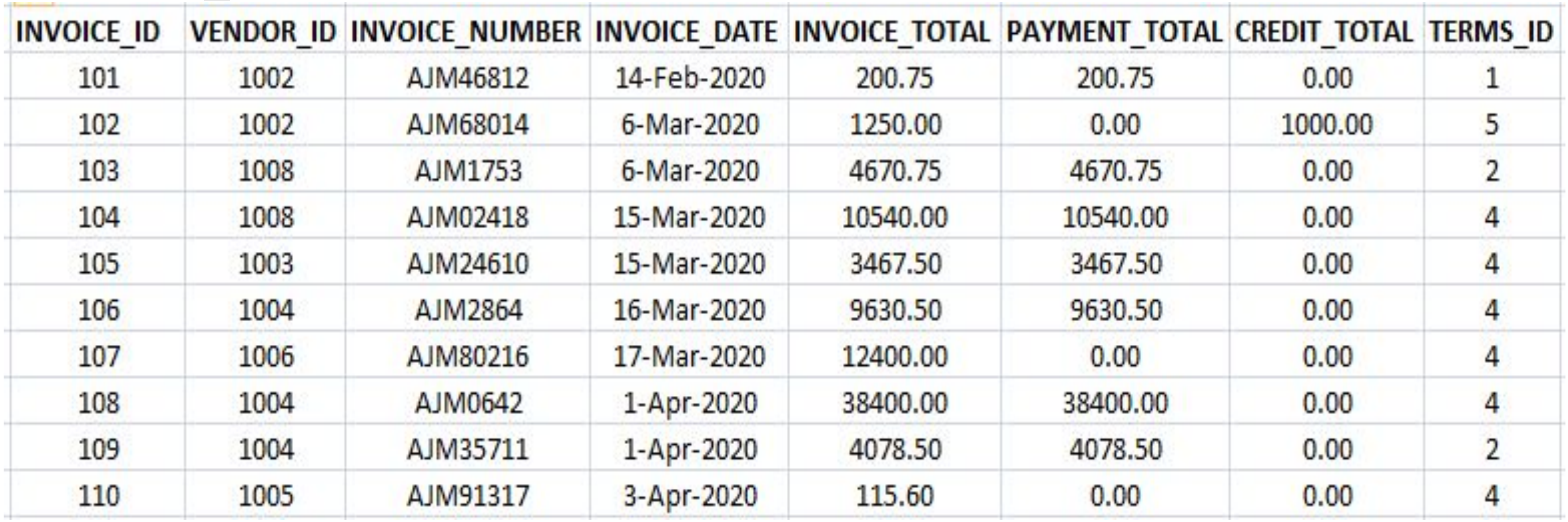
This statement uses a subquery to return a result set that consists of the vendor_id column for each vendor in California. Then, that result set is used with the IN operator in search condition so that only invoices with a vendor_id in the result set are included in the final result.

SUBQUERIES COMPARE TO JOINS

- Both SELECT statements in this figure return a result set that consists of selected rows and columns from the INVOICES TABLE.
- In this case, only the INVOICES from VENDORS in CALIFORNIA are returned.
- As your queries get more complex, you may find that they are easier to code by using subqueries, regardless of the relationships that are involved.

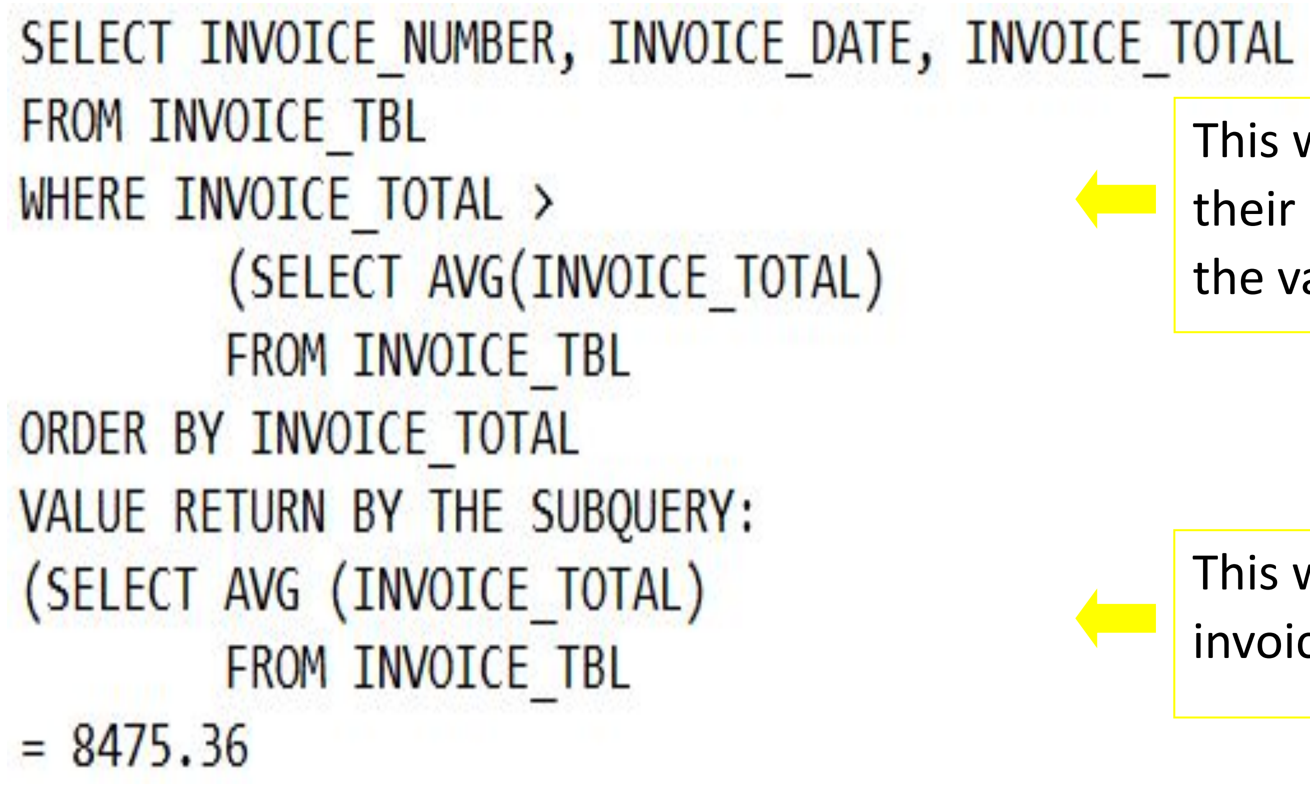
SUBQUERIES IN WHERE CLAUSE

INVOICE_TBL



INVOICE_ID	VENDOR_ID	INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL	PAYMENT_TOTAL	CREDIT_TOTAL	TERMS_ID
101	1002	AJM46812	14-Feb-2020	200.75	200.75	0.00	1
102	1002	AJM68014	6-Mar-2020	1250.00	0.00	1000.00	5
103	1008	AJM1753	6-Mar-2020	4670.75	4670.75	0.00	2
104	1008	AJM02418	15-Mar-2020	10540.00	10540.00	0.00	4
105	1003	AJM24610	15-Mar-2020	3467.50	3467.50	0.00	4
106	1004	AJM2864	16-Mar-2020	9630.50	9630.50	0.00	4
107	1006	AJM80216	17-Mar-2020	12400.00	0.00	0.00	4
108	1004	AJM0642	1-Apr-2020	38400.00	38400.00	0.00	4
109	1004	AJM35711	1-Apr-2020	4078.50	4078.50	0.00	2
110	1005	AJM91317	3-Apr-2020	115.60	0.00	0.00	4

SUBQUERIES IN WHERE CLAUSE



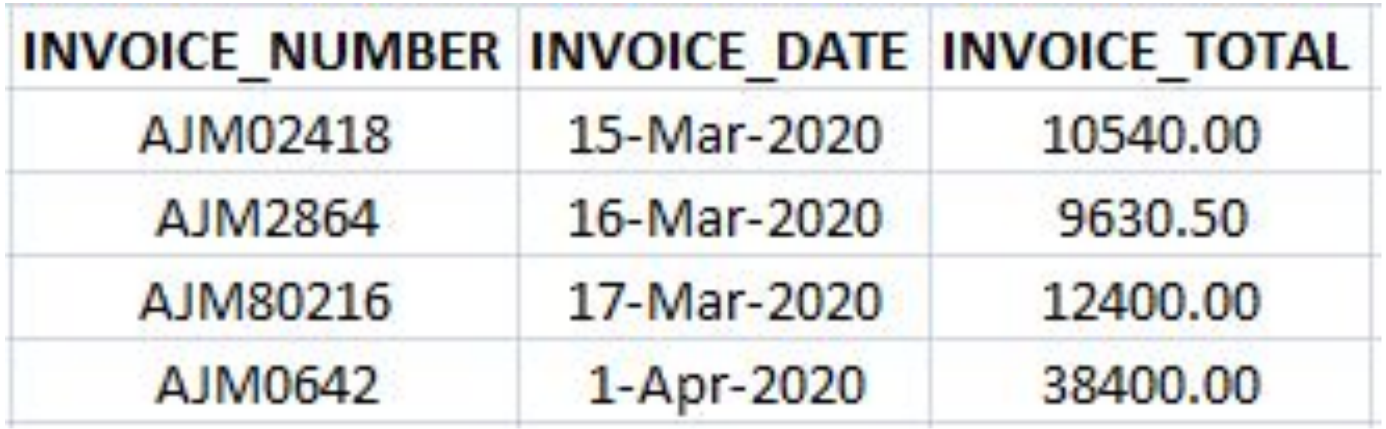
```
SELECT INVOICE_NUMBER, INVOICE_DATE, INVOICE_TOTAL
FROM INVOICE_TBL
WHERE INVOICE_TOTAL >
        (SELECT AVG(INVOICE_TOTAL)
        FROM INVOICE_TBL
ORDER BY INVOICE_TOTAL
VALUE RETURN BY THE SUBQUERY:
(SELECT AVG (INVOICE_TOTAL)
        FROM INVOICE_TBL
= 8475.36
```

This will list all invoices where their invoice total is greater than the value return by the subquery.

This will get the average of the invoice total.

SUBQUERIES IN WHERE CLAUSE

RESULT SET:

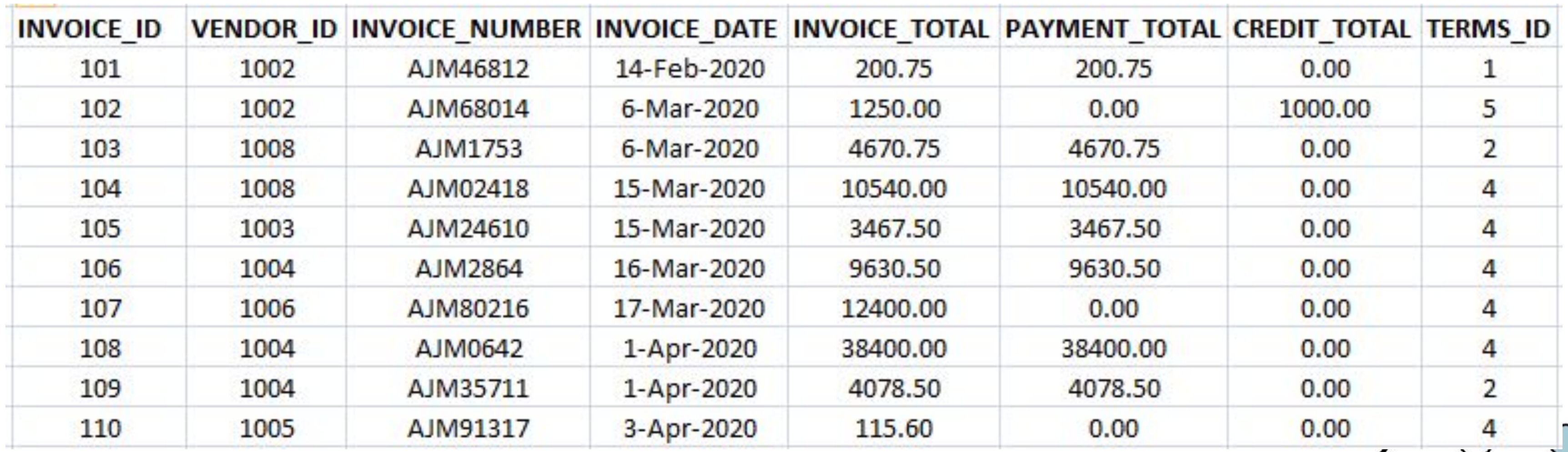


INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL
AJM02418	15-Mar-2020	10540.00
AJM2864	16-Mar-2020	9630.50
AJM80216	17-Mar-2020	12400.00
AJM0642	1-Apr-2020	38400.00

Subqueries in WHERE clause – Using IN Operator

- provide a list of values that are tested against the test expression.
- the subquery must return a single column of values.

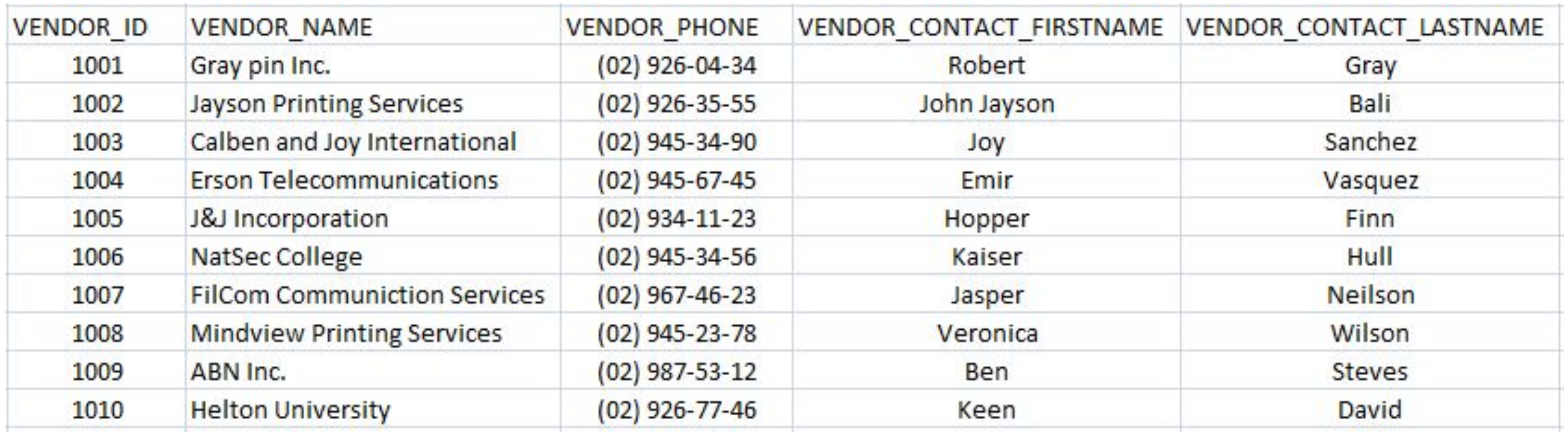
INVOICE_TBL



INVOICE_ID	VENDOR_ID	INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL	PAYMENT_TOTAL	CREDIT_TOTAL	TERMS_ID
101	1002	AJM46812	14-Feb-2020	200.75	200.75	0.00	1
102	1002	AJM68014	6-Mar-2020	1250.00	0.00	1000.00	5
103	1008	AJM1753	6-Mar-2020	4670.75	4670.75	0.00	2
104	1008	AJM02418	15-Mar-2020	10540.00	10540.00	0.00	4
105	1003	AJM24610	15-Mar-2020	3467.50	3467.50	0.00	4
106	1004	AJM2864	16-Mar-2020	9630.50	9630.50	0.00	4
107	1006	AJM80216	17-Mar-2020	12400.00	0.00	0.00	4
108	1004	AJM0642	1-Apr-2020	38400.00	38400.00	0.00	4
109	1004	AJM35711	1-Apr-2020	4078.50	4078.50	0.00	2
110	1005	AJM91317	3-Apr-2020	115.60	0.00	0.00	4

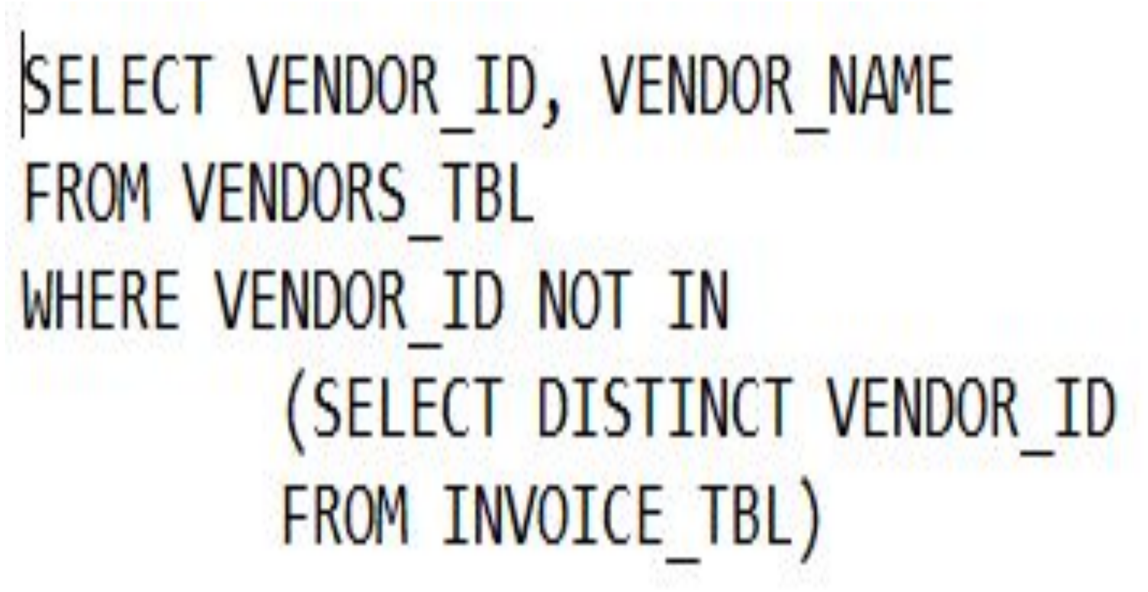
Subqueries in WHERE clause – Using IN Operator

VENDORS_TBL



VENDOR_ID	VENDOR_NAME	VENDOR_PHONE	VENDOR_CONTACT_FIRSTNAME	VENDOR_CONTACT_LASTNAME
1001	Gray pin Inc.	(02) 926-04-34	Robert	Gray
1002	Jayson Printing Services	(02) 926-35-55	John Jayson	Bali
1003	Calben and Joy International	(02) 945-34-90	Joy	Sanchez
1004	Erson Telecommunications	(02) 945-67-45	Emir	Vasquez
1005	J&J Incorporation	(02) 934-11-23	Hopper	Finn
1006	NatSec College	(02) 945-34-56	Kaiser	Hull
1007	FilCom Communiction Services	(02) 967-46-23	Jasper	Neilson
1008	Mindview Printing Services	(02) 945-23-78	Veronica	Wilson
1009	ABN Inc.	(02) 987-53-12	Ben	Steves
1010	Helton University	(02) 926-77-46	Keen	David

Subqueries in WHERE clause – Using IN Operator



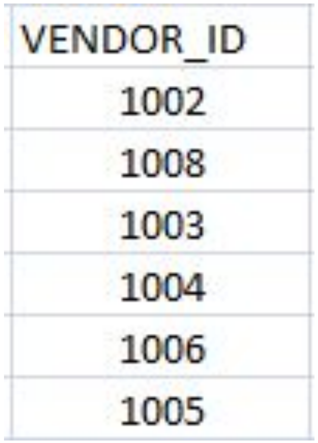
```
SELECT VENDOR_ID, VENDOR_NAME
FROM VENDORS_TBL
WHERE VENDOR_ID NOT IN
        (SELECT DISTINCT VENDOR_ID
        FROM INVOICE_TBL)
```

This will list all the vendors not in the result set of the subquery. This statement use IN operator to test.

This will list all the vendors in invoice _tbl

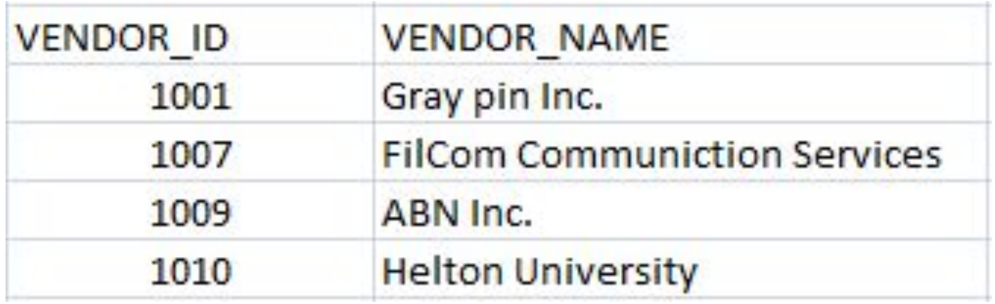
Subqueries in WHERE clause – Using IN Operator

THE RESULT OF THE SUBQUERY:



VENDOR_ID
1002
1008
1003
1004
1006
1005

THE RESULT SET:



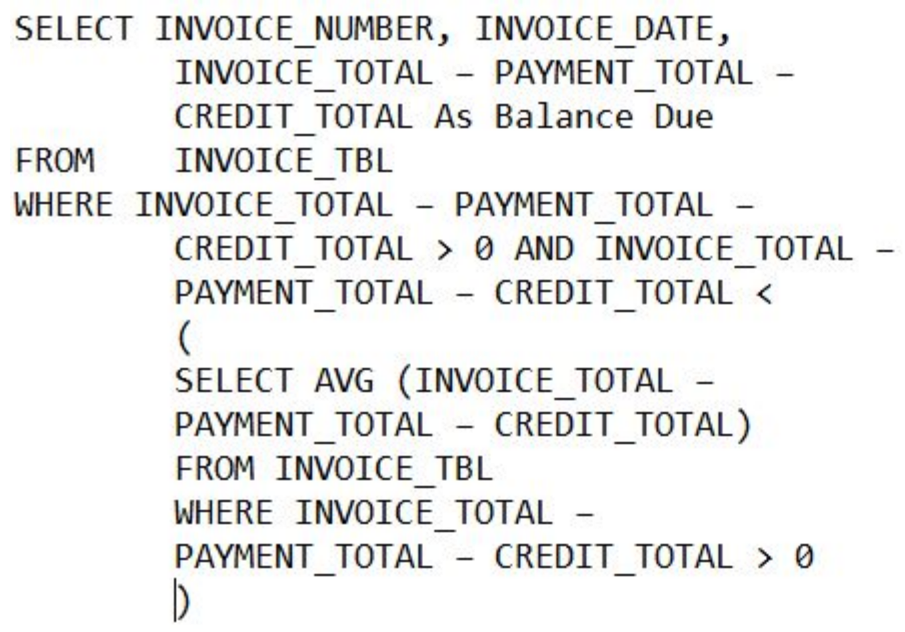
VENDOR_ID	VENDOR_NAME
1001	Gray pin Inc.
1007	FilCom Communiction Services
1009	ABN Inc.
1010	Helton University

Subqueries in WHERE clause – Using EXPRESSION

- You can use the comparison operators to compare an expression with the result of subquery.
- When you use a comparison operator, the subquery must return a single value.
- However, you can also use the comparison operators with subqueries that return two or more values
- using SOME, ANY or ALL keyword to modify the comparison operator

Subqueries in WHERE clause – Using EXPRESSION

- **Syntax:**
  - **WHERE** expression **comparison_operator** [SOME|ANY|ALL] **(Subquery)**



```
SELECT INVOICE_NUMBER, INVOICE_DATE,
        INVOICE_TOTAL – PAYMENT_TOTAL –
        CREDIT_TOTAL As Balance Due
FROM    INVOICE_TBL
WHERE  INVOICE_TOTAL – PAYMENT_TOTAL –
        CREDIT_TOTAL > 0 AND INVOICE_TOTAL –
        PAYMENT_TOTAL – CREDIT_TOTAL <
        (
        SELECT AVG (INVOICE_TOTAL –
        PAYMENT_TOTAL – CREDIT_TOTAL)
        FROM INVOICE_TBL
        WHERE INVOICE_TOTAL –
        PAYMENT_TOTAL – CREDIT_TOTAL > 0
        )
```

This will list invoices that has a balance due greater than 0 and less than the return value of the subquery.

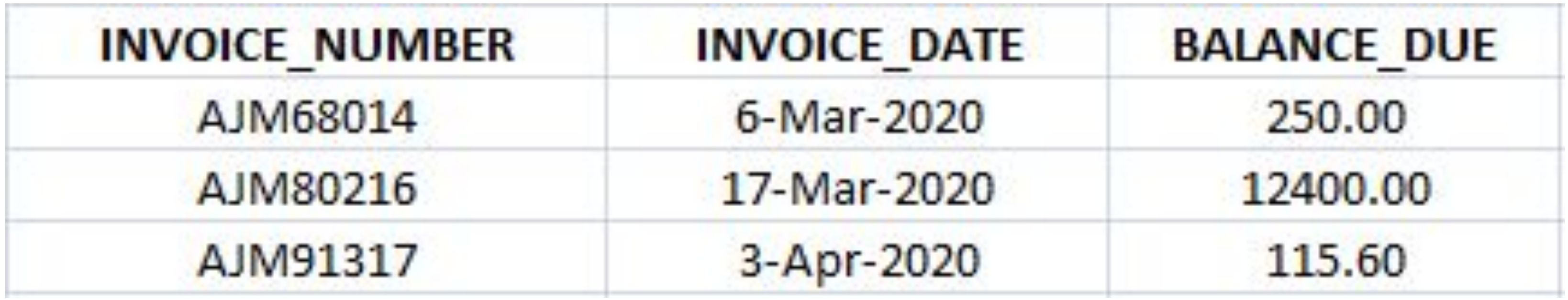
This will return the average value of the invoices where the difference of their invoice total, payment total and credit total is greater than 0.

Subqueries in WHERE clause – Using EXPRESSION

VALUE RETURN BY THE SUBQUERY:

4255.20

RESULT SET:

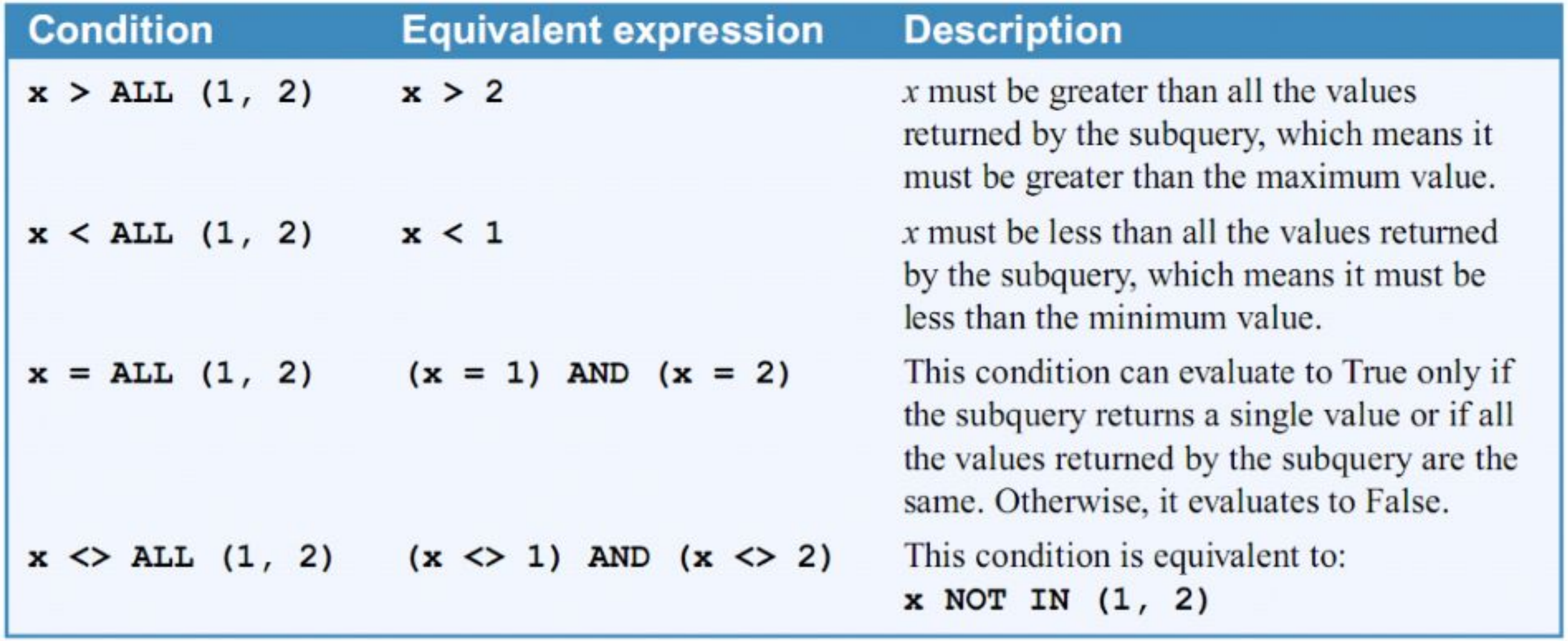


INVOICE_NUMBER	INVOICE_DATE	BALANCE_DUE
AJM68014	6-Mar-2020	250.00
AJM80216	17-Mar-2020	12400.00
AJM91317	3-Apr-2020	115.60

ALL Keywords

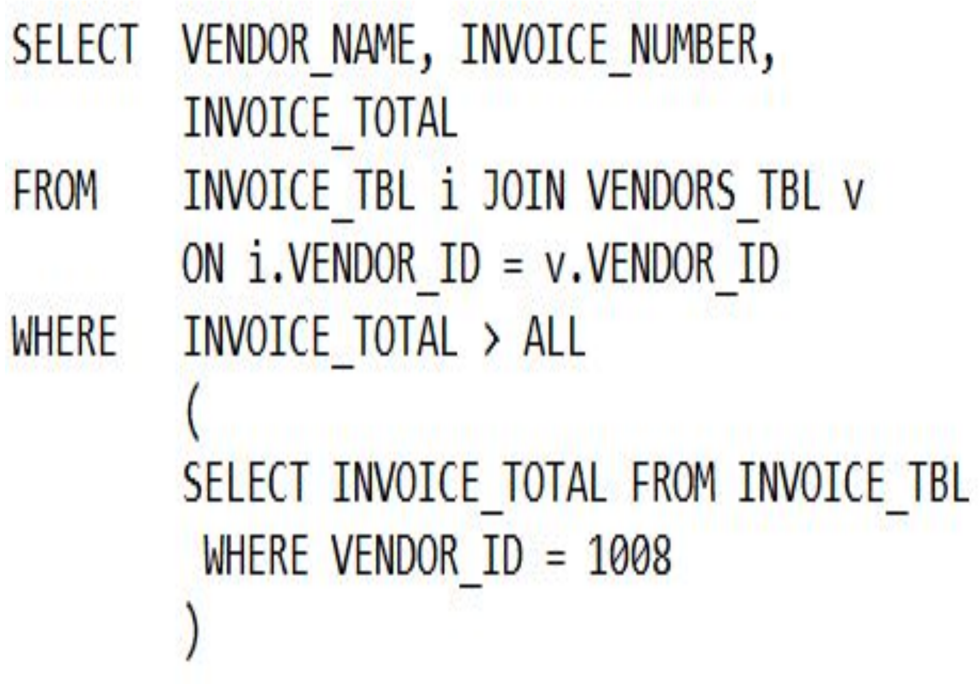
- This keyword modifies the comparison operator so the condition must be true for all the values returned by a subquery.
- This is equivalent to coding a series of conditions connected by AND operators.
- If no rows are returned by the subquery, a comparison that uses the all keyword is always true.
- If all the rows returned by the subquery contain a null value, a comparison that uses the ALL keyword is always false.

ALL Keywords



Condition	Equivalent expression	Description
`x > ALL (1, 2)`	`x > 2`	*x* must be greater than all the values returned by the subquery, which means it must be greater than the maximum value.
`x < ALL (1, 2)`	`x < 1`	*x* must be less than all the values returned by the subquery, which means it must be less than the minimum value.
`x = ALL (1, 2)`	`(x = 1) AND (x = 2)`	This condition can evaluate to True only if the subquery returns a single value or if all the values returned by the subquery are the same. Otherwise, it evaluates to False.
`x <> ALL (1, 2)`	`(x <> 1) AND (x <> 2)`	This condition is equivalent to: `x NOT IN (1, 2)`

ALL Keywords



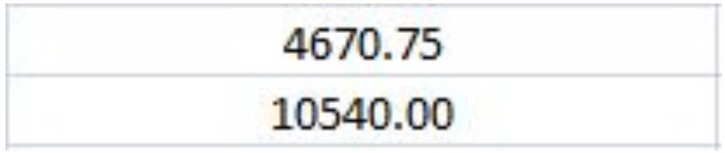
```
SELECT  VENDOR_NAME, INVOICE_NUMBER,
        INVOICE_TOTAL
FROM    INVOICE_TBL i JOIN VENDORS_TBL v
        ON i.VENDOR_ID = v.VENDOR_ID
WHERE   INVOICE_TOTAL > ALL
        (
        SELECT INVOICE_TOTAL FROM INVOICE_TBL
         WHERE VENDOR_ID = 1008
        )
```

This will list all the vendors that has invoice greater than the maximum value of the returned value in subquery.

This will list all the invoice total of vendor that has vendor id 1008

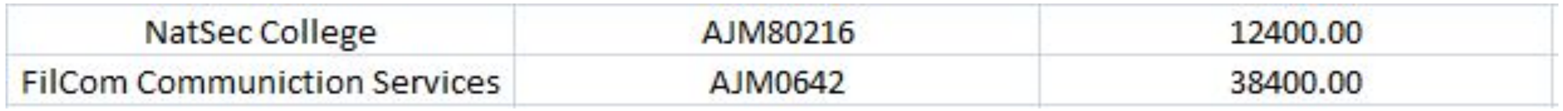
ALL Keywords

RESULT OF SUBQUERY:



4670.75
10540.00

RESULT SET:

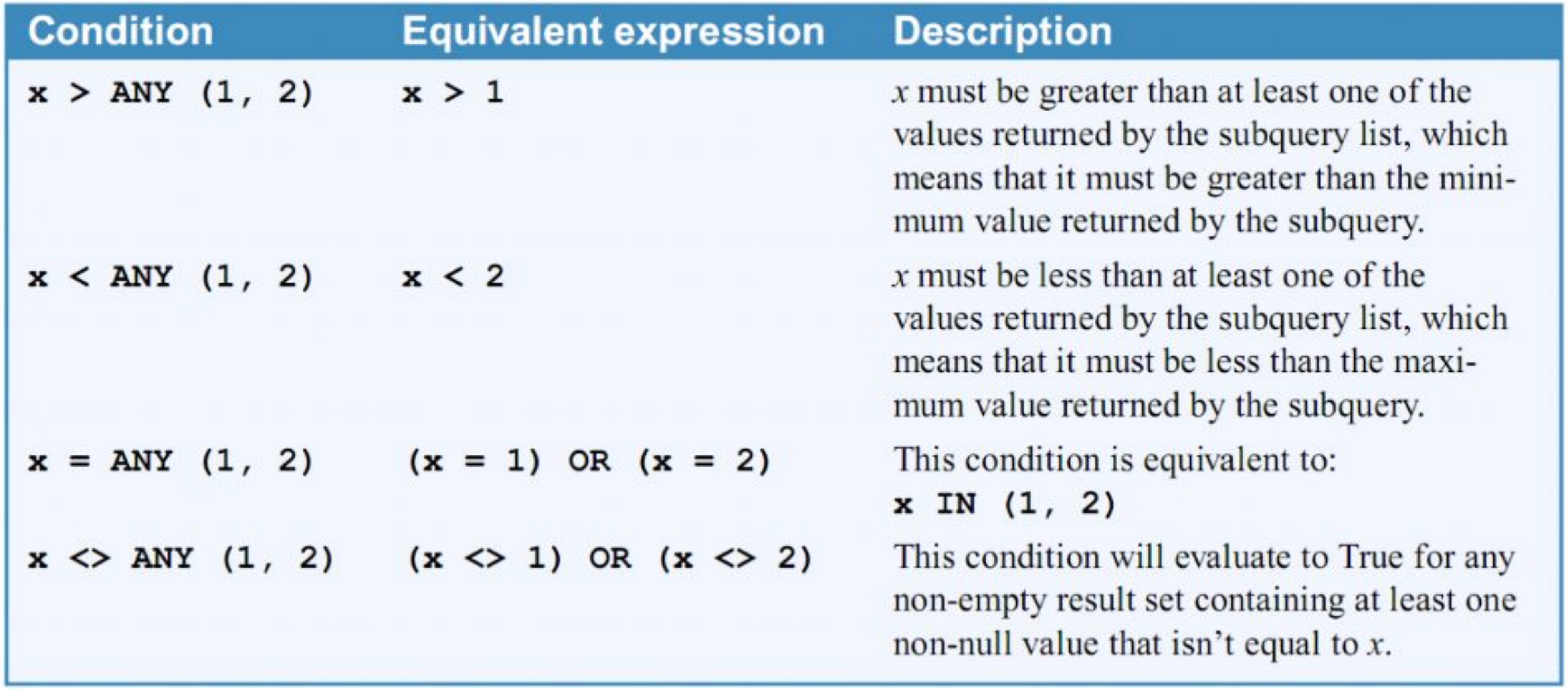


NatSec College	AJM80216	12400.00
FilCom Communiction Services	AJM0642	38400.00

ANY and SOME keywords

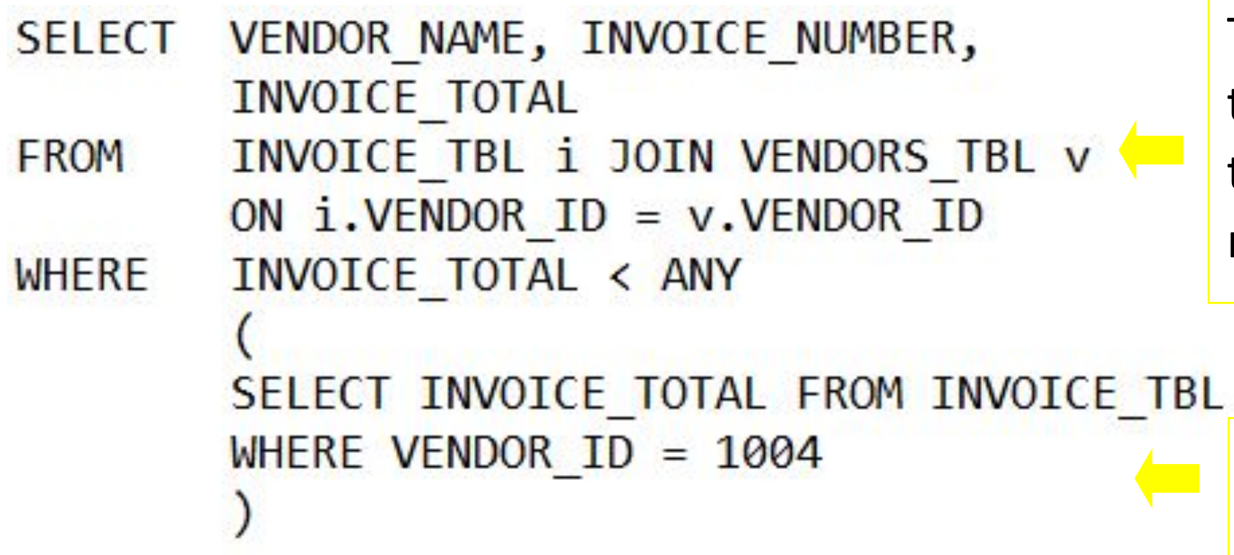
- to test if a comparison is true for any, or some, of the values returned by a subquery.
- equivalent to coding a series of conditions connected with OR operators.
- ANY and SOME are equivalent keywords.
- SOME is the ANSI-standard keyword, but ANY is more commonly used.
- a comparison that uses the ANY or SOME keyword is always false if no rows returned subquery contain no value

ANY and SOME keywords



Condition	Equivalent expression	Description
`x > ANY (1, 2)`	`x > 1`	*x* must be greater than at least one of the values returned by the subquery list, which means that it must be greater than the minimum value returned by the subquery.
`x < ANY (1, 2)`	`x < 2`	*x* must be less than at least one of the values returned by the subquery list, which means that it must be less than the maximum value returned by the subquery.
`x = ANY (1, 2)`	`(x = 1) OR (x = 2)`	This condition is equivalent to: `x IN (1, 2)`
`x <> ANY (1, 2)`	`(x <> 1) OR (x <> 2)`	This condition will evaluate to True for any non-empty result set containing at least one non-null value that isn't equal to *x*.

ANY and SOME keywords



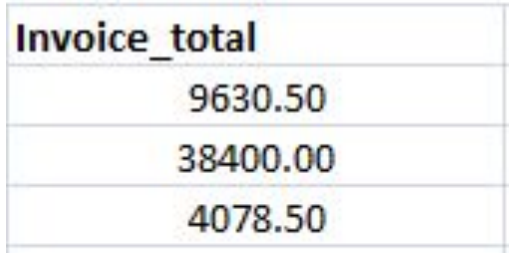
```
SELECT   VENDOR_NAME, INVOICE_NUMBER,
         INVOICE_TOTAL
FROM     INVOICE_TBL i JOIN VENDORS_TBL v
         ON i.VENDOR_ID = v.VENDOR_ID
WHERE    INVOICE_TOTAL < ANY
         (
         SELECT INVOICE_TOTAL FROM INVOICE_TBL
         WHERE VENDOR_ID = 1004
         )
```

This will list all the vendors that has invoice greater than the maximum value of the returned value in subquery.

This will list all the invoice total of vendor that has vendor id 1004

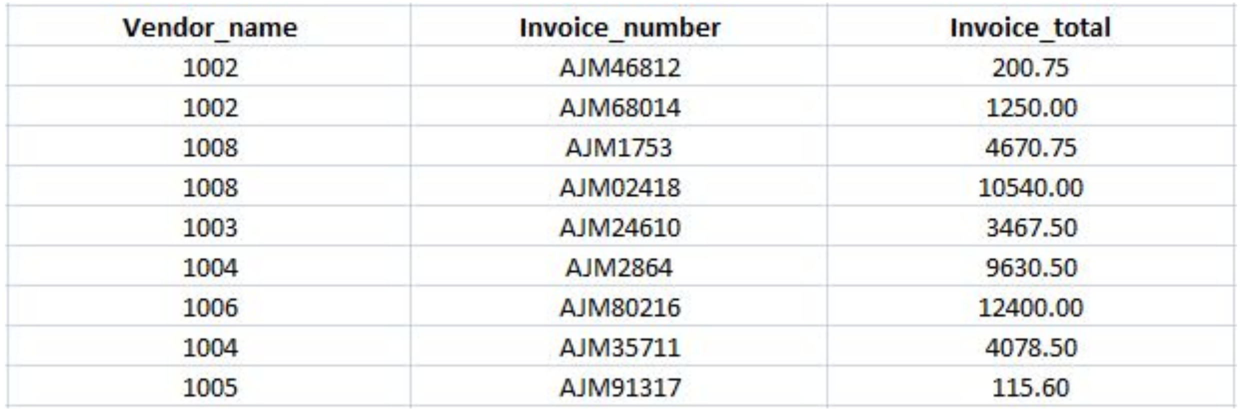
ANY and SOME keywords

RESULT OF SUBQUERY:



Invoice_total
9630.50
38400.00
4078.50

RESULT SET:

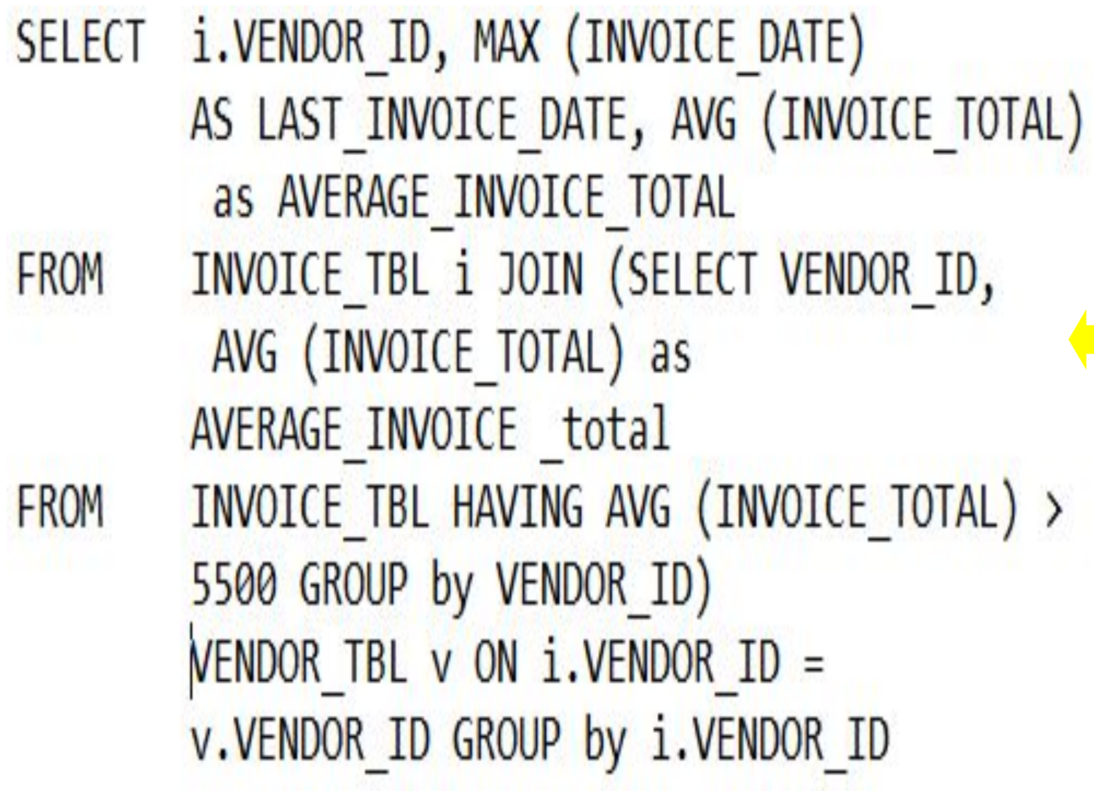


Vendor_name	Invoice_number	Invoice_total
1002	AJM46812	200.75
1002	AJM68014	1250.00
1008	AJM1753	4670.75
1008	AJM02418	10540.00
1003	AJM24610	3467.50
1004	AJM2864	9630.50
1006	AJM80216	12400.00
1004	AJM35711	4078.50
1005	AJM91317	115.60

Subquery in FROM clause

- A subquery that is coded in the FROM clause returns a result set that can be referred to as *inline view*.
- When you code a subquery in the FROM clause, you must assign names to any calculated values in the result set.
- ***Inline views* are most useful when you need to further summarize of a summary query.**
- *Inline view* is like a view in that it retrieves selected rows and columns from one or more base tables

ANY and SOME keywords



```
SELECT  i.VENDOR_ID, MAX (INVOICE_DATE)
        AS LAST_INVOICE_DATE, AVG (INVOICE_TOTAL)
         as AVERAGE_INVOICE_TOTAL
FROM    INVOICE_TBL i JOIN (SELECT VENDOR_ID,
         AVG (INVOICE_TOTAL) as
        AVERAGE_INVOICE _total
FROM    INVOICE_TBL HAVING AVG (INVOICE_TOTAL) >
        5500 GROUP by VENDOR_ID)
        VENDOR_TBL v ON i.VENDOR_ID =
        v.VENDOR_ID GROUP by i.VENDOR_ID
```

This will create an inline view that contains the vendor_id values and the average invoice totals for all the vendors with invoice total averages over 5500.

ANY and SOME keywords

RESULT OF SUBQUERY:

Vendor_id	Average_invoice_total
1008	7605.38
1004	17369.67
1006	12400.00

RESULT SET:

Vendor_id	Last_Invoice_date	Average_Invoice_total
1008	7605.38	15-Mar-2020
1004	17369.67	1-Apr-2020
1006	12400.00	17-Mar-2020

Subquery in SELECT clause

- **When you code a subquery for a column specification in the SELECT clause, the subquery must return a single value.**
- **A subquery that is coded within a SELECT clause is typically a correlated subquery.**
- A query that includes a subquery in its SELECT clause can typically be restated using a join instead of subquery.
  - Join is usually faster and more readable; subqueries are seldom coded in the SELECT clause.

Subquery in SELECT clause

```
SELECT  VENDOR_NAME, (SELECT MAX (INVOICE_DATE)
FROM    INVOICE_TBL
WHERE   INVOICE_TBL.VENDOR_ID = VENDOR_TBL.VENDOR_ID)
         AS LATEST_INVOICE_DATE
FROM    VENDORS_TBL
```

RESULT SET:

Vendor_Name	Latest_Invoice_Date
Jayson Printing Services	6-Mar-2020
Mindview Printing Services	15-Mar-2020
Calben and Joy International	15-Mar-2020
Erson Telecommunications	1-Apr-2020
NatSec College	17-Mar-2020
J&J Incorporation	3-Apr-2020

End of Lesson...