The University of Texas at Austin
McCombs School of Business

MIS 381N INTRO. TO DATABASE MANAGEMENT

Select

Tayfun Keskin
Visiting Clinical Professor, The University of Texas at Austin, McCombs School of Business
Associate Teaching Professor, University of Washington Seattle, Foster School of Business

QUESTIONS

Any questions

before we begin …

AGENDA

Lecture	Select
Hands-On	Exercises
Looking Forward	Homework 3 Quiz 3

Select

SIX STEPS (WHICH TRANSLATE INTO THE SIX CLAUSES OF THE SELECT STATEMENT)

- What do you want to display?
- Where are you getting the data?
- Are there any conditions on attributes or join conditions?
- Do you need to group by any attribute?
- Are there any conditions on the aggregate functions?
- Do you want the results to appear in a certain order?

SIX STEPS (WHICH TRANSLATE INTO THE SIX CLAUSES OF THE SELECT STATEMENT)

- What do you want to display? SELECT
- Where are you getting the data? FROM
- Are there any conditions on attributes or join conditions? WHERE
- Do you need to group by any attribute? GROUP BY
- Are there any conditions on the aggregate functions? HAVING
- Do you want the results to appear in a certain order? ORDER BY

SELECT	Columns - Column names - Arithmetic expressions - Literals (text or numeric) - Scalar functions
FROM	Table or view names
WHERE	Conditions (qualifies rows)
ORDER BY	Sorts result rows

SELECT	Columns - Column names - Arithmetic expressions - Literals (text or numeric) - Scalar functions **- “Aggregate” functions**
FROM	Table or view names
WHERE	Conditions (qualifies rows)
ORDER BY	Sorts result rows
GROUP BY	**Creates sub totals in conjunction with column functions**
HAVING	**Conditions the sub totals**

LET’S TEST OUR UNDERSTANDING

How do you select all columns from a table?

How do you select specific columns from a table?

How do you filter records?

How do you sort records

LOOKING FORWARD

Read Chapters 3 and 7

Quiz 3

Homework 3

THANK YOU

BACKUP SLIDES

PART 1

Chapter 3

Select

What are 4 main parts of a SELECT statement?

- SELECT
- FROM
- WHERE
- ORDER BY

```
SELECT column_list
FROM table_source
[WHERE filter_condition]
[ORDER BY order_by_list]
```

Knowledge check

- How do you select all columns from a table?
- How do you select specific columns from a table?
- How do you filter records?
- How do you sort records?

SELECT syntax

```
SELECT column_list
 FROM table_source
 [WHERE filter_condition]
 [ORDER BY order_by_list]
```

e.g. SELECT all columns

```
SELECT *
FROM invoices
```

e.g. SELECT particular columns

```
SELECT invoice_number, invoice_date
FROM invoices
```

e.g. Sort with ORDER BY clause

```
SELECT invoice_number, invoice_date
FROM invoices
ORDER by invoice_date [DESC]
```

e.g. Select with WHERE clause

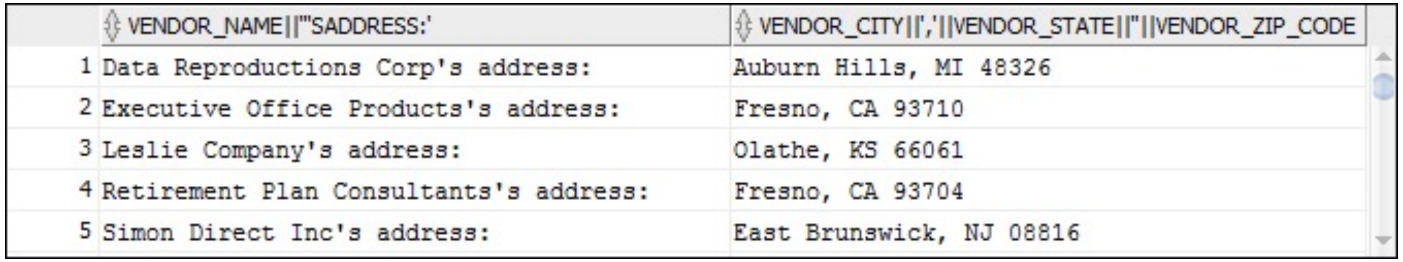
```
SELECT invoice_number, invoice_date
FROM invoices
WHERE invoice_date = ’02-MAY-2014’
```

Ways to specify columns in a query's SELECT clause

- ❑ **All columns** in base table
- ❑ **Column name** in base table
- ❑ **String Expression**
  - Use the string operator || to concatenate column or literal strings together
  - Literal strings are defined by you like the following:
  - **Select city || ',' || state** or **Select 'Mr. or Mrs.' || last_name**
- ❑ **Arithmetic Expression**
  - Use +, -, /, * to do arithmetic calcs on numeric columns
  - e.g. `Select invoice_total – payment_total`
- ❑ **Alias** – give custom column names for calculated columns
  - e.g. `Select (invoice_total – payment_total) as "Amount Owed"`
  - NOTE: Include Quotes (not ticks) around column alias if your column name has spaces are includes spaces. Good rule of thumb is to always use **AS** w/ **no spaces**

How to include apostrophes in literal values

```
SELECT vendor_name || '''s address: ',
       vendor_city || ', '
                   || vendor_state
                   || ' '
                   || vendor_zip_code
FROM vendors
```



	VENDOR_NAME||'''SADDRESS:'	VENDOR_CITY||','||VENDOR_STATE||''||VENDOR_ZIP_CODE
1	Data Reproductions Corp's address:	Auburn Hills, MI 48326
2	Executive Office Products's address:	Fresno, CA 93710
3	Leslie Company's address:	Olathe, KS 66061
4	Retirement Plan Consultants's address:	Fresno, CA 93704
5	Simon Direct Inc's address:	East Brunswick, NJ 08816

Terms to know

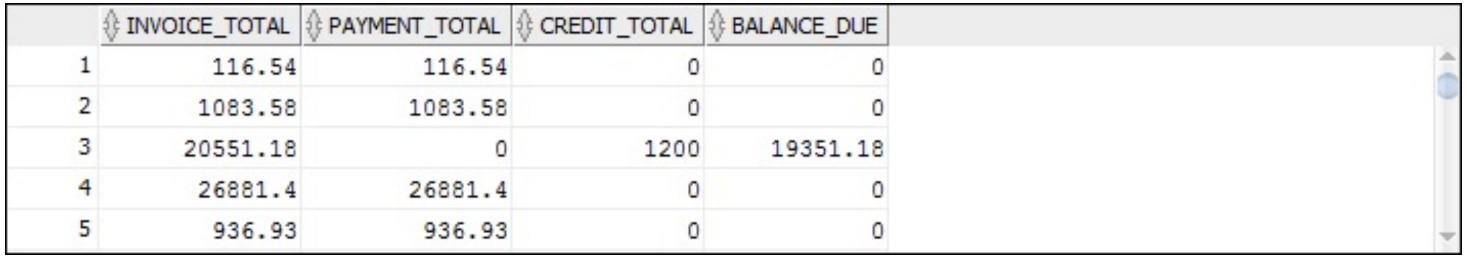
- String expression
- Literal value
- String literal (string constant)
- Concatenation operator

The arithmetic operators in order of precedence

*	Multiplication
/	Division
+	Addition
-	Subtraction

A SELECT statement that calculates balance due

```
SELECT invoice_total, payment_total, credit_total,
       invoice_total – payment_total – credit_total
           AS balance_due
FROM invoices
```



	INVOICE_TOTAL	PAYMENT_TOTAL	CREDIT_TOTAL	BALANCE_DUE
1	116.54	116.54	0	0
2	1083.58	1083.58	0	0
3	20551.18	0	1200	19351.18
4	26881.4	26881.4	0	0
5	936.93	936.93	0	0

Practice String/Numeric Expressions & Column Alias:

1. Select all vendors from NY but concatenate Address into this format: **Address; City, State Zip** Don’t include alias this time

2. Add a column alias to this newly created column called **“Vendor Address”**

3. Pull ***vendor_id***, ***invoice total***, ***payment total***, and ***difference of invoice_total & payment_total***. Rename calculate calculated to “Amount Owed”

String expression to derive full name

```
SELECT first_name || ' ' || last_name
```

String expression to derive full name w/ alias

```
SELECT first_name || ' ' || last_name as “Full Name”
```

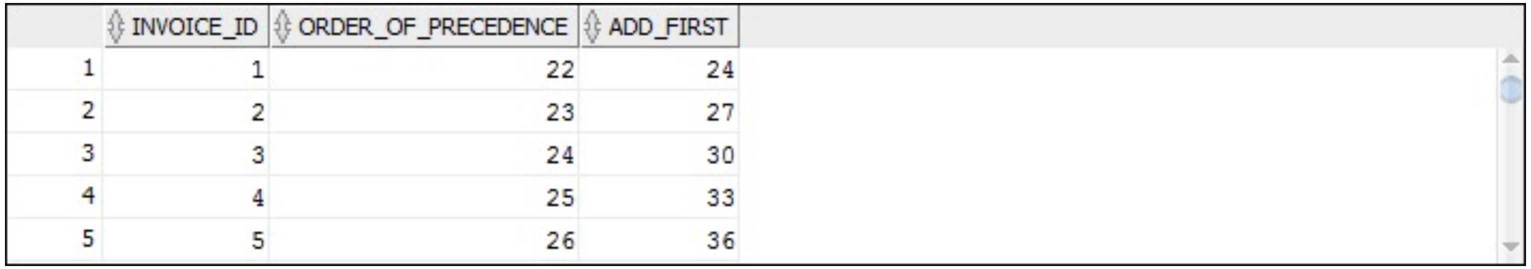
Arithmetic expression to calculate a value

```
SELECT invoice_number,
       invoice_total – payment_total - credit_total
FROM invoices
```

Order of operations

A SELECT statement that uses parentheses

```
SELECT invoice_id,
    invoice_id + 7 * 3 AS order_of_precedence,
    (invoice_id + 7) * 3 AS add_first
FROM invoices
ORDER BY invoice_id
```



	INVOICE_ID	ORDER_OF_PRECEDENCE	ADD_FIRST
1	1	22	24
2	2	23	27
3	3	24	30
4	4	25	33
5	5	26	36

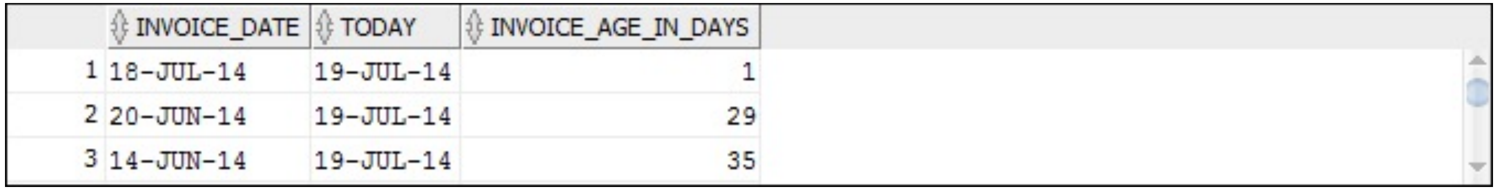
Scalar functions

- ❑ **Scalar** = Operates on a single value and returns a single value

- ❑ **SYSDATE** – returns today’s date/time. Like NOW() function in Excel
- ❑ **ROUND** – round decimals to whole numbers
- ❑ **SUBSTR** – Returns certain part of a string. Like MID() function in Excel
- ❑ **TO_CHAR** – convert number/date to string
- ❑ **TO_DATE** – convert string to a date
- ❑ **MOD** – returns remainder of division of two numbers

A SELECT statement that uses the SYSDATE and ROUND functions

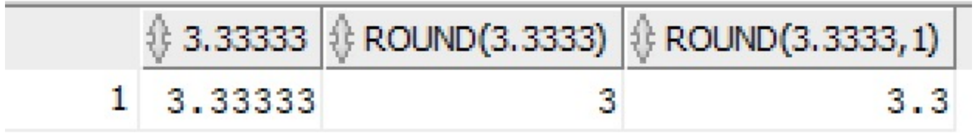
```
SELECT invoice_date,
    SYSDATE AS today,
    ROUND(SYSDATE - invoice_date)  AS invoice_age_in_days
FROM invoices
```



	INVOICE_DATE	TODAY	INVOICE_AGE_IN_DAYS
1	18-JUL-14	19-JUL-14	1
2	20-JUN-14	19-JUL-14	29
3	14-JUN-14	19-JUL-14	35

ROUND(number to found [,number of decimals])

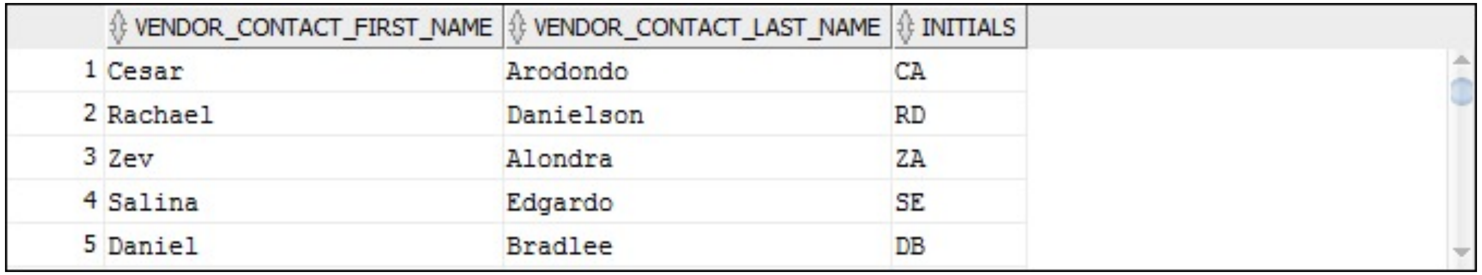
```
SELECT   3.33333,
         ROUND(3.3333),
         ROUND(3.3333,1)
FROM dual
```



	3.33333	ROUND(3.3333)	ROUND(3.3333,1)
1	3.33333	3	3.3

A SELECT statement that uses SUBSTR

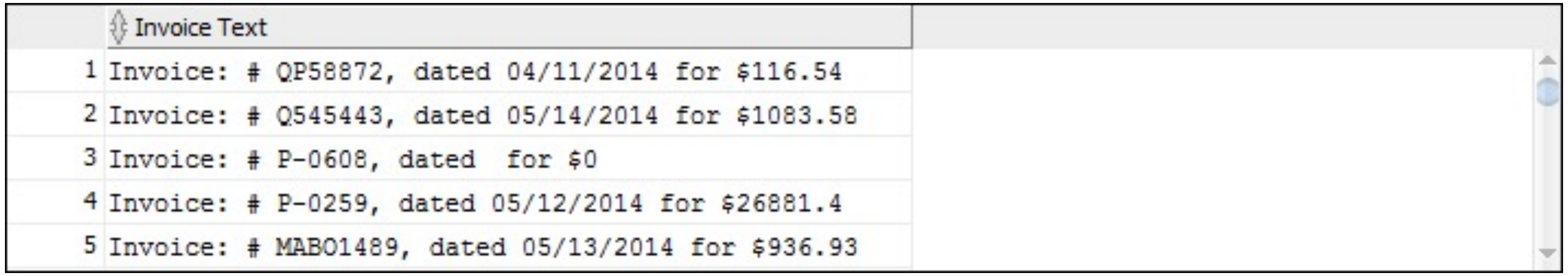
```
SELECT vendor_contact_first_name,
vendor_contact_last_name,
    SUBSTR(vendor_contact_first_name, 1, 1) ||
    SUBSTR(vendor_contact_last_name, 1, 1) AS initials
FROM vendors
```



	VENDOR_CONTACT_FIRST_NAME	VENDOR_CONTACT_LAST_NAME	INITIALS
1	Cesar	Arodondo	CA
2	Rachael	Danielson	RD
3	Zev	Alondra	ZA
4	Salina	Edgardo	SE
5	Daniel	Bradlee	DB

A SELECT statement that uses TO_CHAR

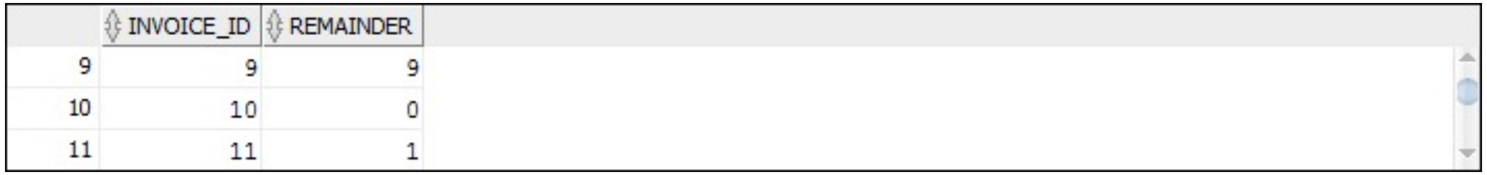
```
SELECT 'Invoice: # '
       || invoice_number
       || ', dated '
       || TO_CHAR(payment_date, 'MM/DD/YYYY')
       || ' for $'
       || TO_CHAR(payment_total)
      AS "Invoice Text"
FROM invoices
```



```
    Invoice Text
  1 Invoice: # QP58872, dated 04/11/2014 for $116.54
  2 Invoice: # Q545443, dated 05/14/2014 for $1083.58
  3 Invoice: # P-0608, dated  for $0
  4 Invoice: # P-0259, dated 05/12/2014 for $26881.4
  5 Invoice: # MAB01489, dated 05/13/2014 for $936.93
```

A SELECT statement that uses the MOD function

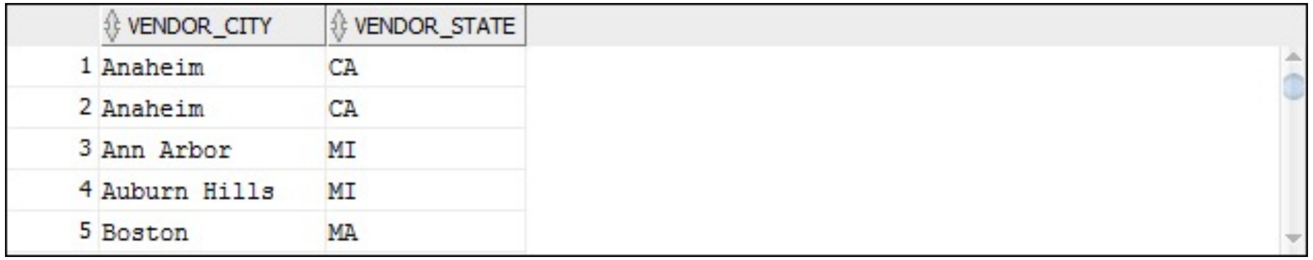
```
SELECT invoice_id,
    MOD(invoice_id, 10) AS Remainder
FROM invoices
ORDER BY invoice_id
```



	INVOICE_ID	REMAINDER
9	9	9
10	10	0
11	11	1

A SELECT statement that returns all rows

```
SELECT vendor_city, vendor_state
FROM vendors
ORDER BY vendor_city
```

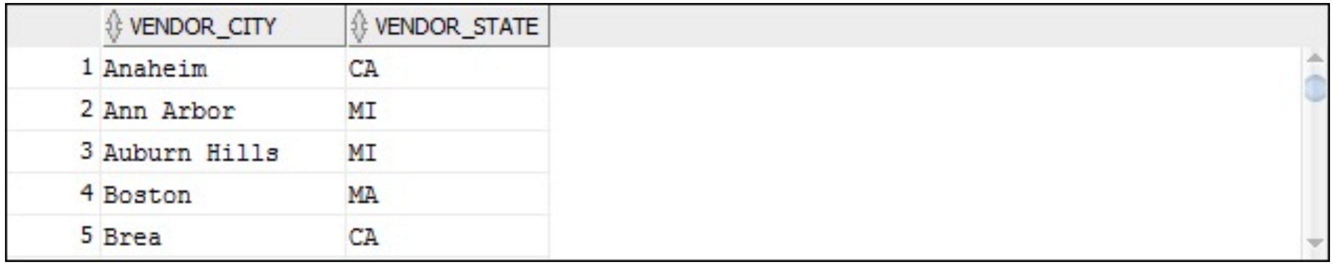


	VENDOR_CITY	VENDOR_STATE
1	Anaheim	CA
2	Anaheim	CA
3	Ann Arbor	MI
4	Auburn Hills	MI
5	Boston	MA

```
(122 rows selected)
```

A SELECT statement with no duplicate rows

```
SELECT DISTINCT vendor_city, vendor_state
FROM vendors
ORDER BY vendor_city
```

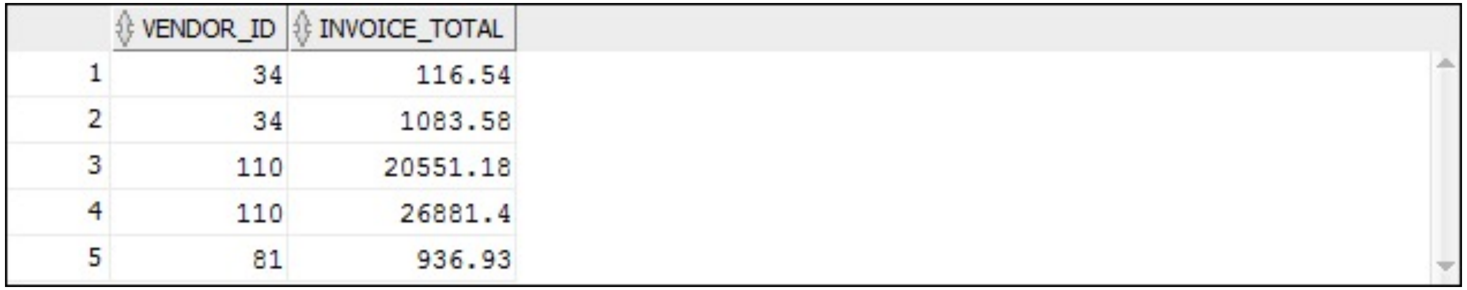


	VENDOR_CITY	VENDOR_STATE
1	Anaheim	CA
2	Ann Arbor	MI
3	Auburn Hills	MI
4	Boston	MA
5	Brea	CA

```
(53 rows selected)
```

A SELECT statement that uses the ROWNUM pseudo column to limit the number of rows

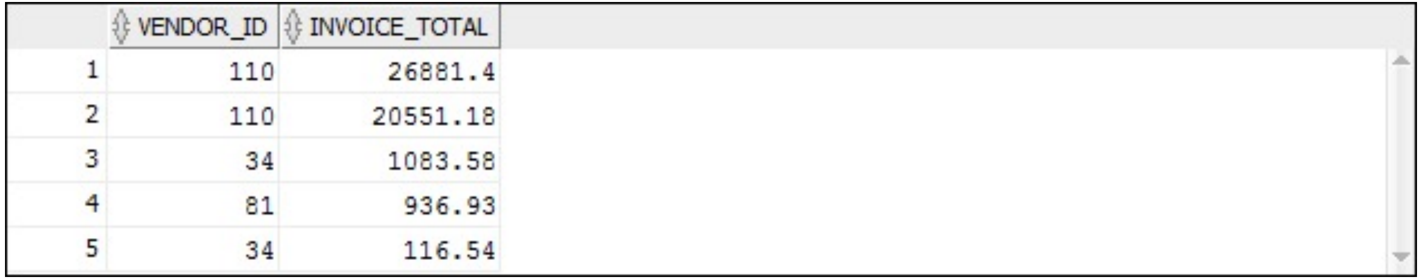
```
SELECT vendor_id, invoice_total
FROM invoices
WHERE ROWNUM <= 5
```



	VENDOR_ID	INVOICE_TOTAL
1	34	116.54
2	34	1083.58
3	110	20551.18
4	110	26881.4
5	81	936.93

A SELECT statement that sorts the result set after the WHERE clause

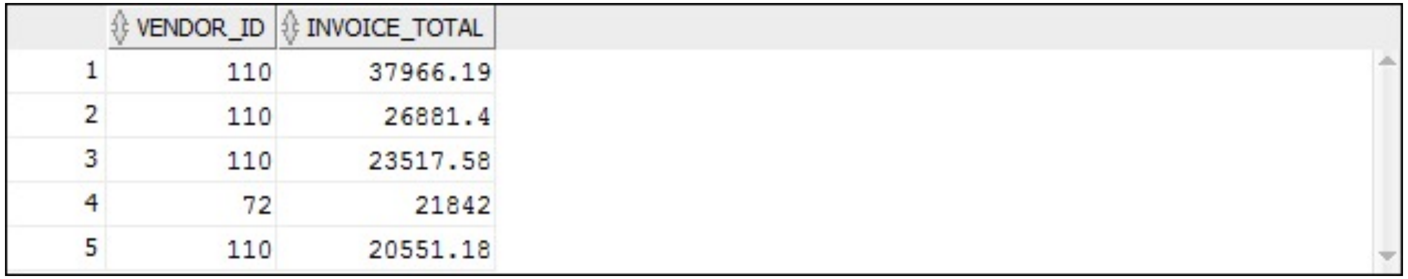
```
SELECT vendor_id, invoice_total
FROM invoices
WHERE ROWNUM <= 5
ORDER BY invoice_total DESC
```



	VENDOR_ID	INVOICE_TOTAL
1	110	26881.4
2	110	20551.18
3	34	1083.58
4	81	936.93
5	34	116.54

A SELECT statement that sorts the result set before the WHERE clause

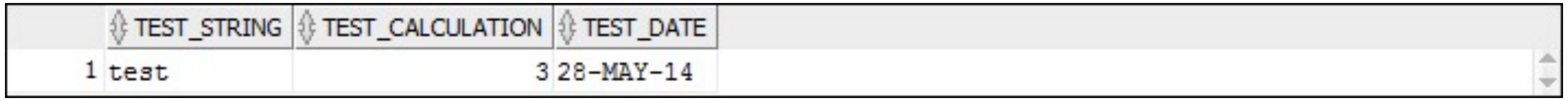
```
SELECT vendor_id, invoice_total
FROM (SELECT * FROM invoices
      ORDER BY invoice_total DESC)
WHERE ROWNUM <= 5
```



	VENDOR_ID	INVOICE_TOTAL
1	110	37966.19
2	110	26881.4
3	110	23517.58
4	72	21842
5	110	20551.18

A SELECT statement that uses the Dual table

```
SELECT 'test'  AS test_string,
       10-7    AS test_calculation,
       SYSDATE AS test_date
FROM Dual
```



	TEST_STRING	TEST_CALCULATION	TEST_DATE
1	test	3	28-MAY-14