

HOMework 4

1. **(10 points)** Write a SELECT statement that returns the same result set as this SELECT statement, but don't use a join. Instead, use a subquery in a WHERE clause that uses the IN keyword.

```
SELECT DISTINCT
  vendor_name
FROM
  vendors
  INNER JOIN
    invoices ON vendors.vendor_id = invoices.vendor_id
ORDER BY vendor_name;
```

(Answer on next page)

```

9      -- Question 1
10     • SELECT DISTINCT vendor_name
11     FROM VENDORS
12     WHERE vendor_id IN (SELECT vendor_id
13                        FROM INVOICES)
14     ORDER BY vendor_name;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

vendor_name
Abbey Office Furnishings
Bertelsmann Industry Svcs. Inc
Blue Cross
Cahners Publishing Company
Cardinal Business Media, Inc.
Coffee Break Service
Compuserve
Computerworld
Data Reproductions Corp
Dean Witter Reynolds
Digital Dreamworks
Dristas Groom & McCormick
Edward Data Services
Evans Executone Inc
Federal Express Corporation
Ford Motor Credit Company
Franchise Tax Board
Fresno County Tax Collector
Gostanian General Building
IBM
Ingram
Malloy Lithographing Inc
Pacific Bell
Pollstar
Postmaster
Reiter's Scientific & Pro Books
Roadway Package System, Inc
Suburban Propane
United Parcel Service
Wakefield Co
Wang Laboratories, Inc.
Wells Fargo Bank
Yesmed, Inc
Zylka Design

VENDORS 114 x

Output

Action Output

#	Time	Action	Message
1	13:48:15	SELECT DISTINCT vendor name FROM VENDORS ...	34 row(s) returned

2. **(10 points)** Write a SELECT statement that returns two columns from the General_Ledger_Accounts table: account_number and account_description.

Return one row for each account number that has never been assigned to any line item in the Invoice_Line_Items table. To do that, use a subquery introduced with the NOT EXISTS operator.

This should return 54 rows. Sort the results by the account_number column.

```
8  -- Question 2
9  •  SELECT account_number, account_description
10  FROM GENERAL_LEDGER_ACCOUNTS
11  WHERE account_number NOT IN (SELECT account_number
12                                FROM INVOICE_LINE_ITEMS)
13  ORDER BY account_number;
```

Result Grid

account_number	account_description
100	Cash
110	Accounts Receivable
120	Book Inventory
162	Capitalized Lease
167	Software
181	Book Development
200	Accounts Payable
205	Royalties Payable
221	401K Employee Contributions
230	Sales Taxes Payable
234	Medicare Taxes Payable
235	Income Taxes Payable

ACCOUNTS 1 x

Output

Action Output

#	Time	Action	Message
1	14:38:21	SELECT account_number, account_description FROM GENERAL_LEDGER_ACCOUNTS W...	54 row(s) returned

3. **(10 points)** Write a SELECT statement that returns four columns: vendor_name, invoice_id, invoice_sequence, and line_item_amount.

Return a row for each line item of each invoice that has more than one line item in the Invoice_Line_Items table. *Hint: Use a subquery that tests for invoice_sequence > 1.* This should return 6 rows.

Hint: Before embarking on writing the query, it would help if you first navigated the database tables in question and first try to answer the question through navigation.

```
15  -- Question 3
16  •  SELECT Vendor_Name, ili.Invoice_ID, invoice_sequence, line_item_amount
17     FROM Invoice_Line_Items ili
18     JOIN Invoices
19     ON ili.Invoice_ID = Invoices.Invoice_ID
20     JOIN Vendors ON Invoices.Vendor_ID = Vendors.Vendor_ID
21     WHERE ili.Invoice_ID IN (SELECT invoice_id
22                             FROM INVOICE_LINE_ITEMS
23                             WHERE invoice_sequence > 1);
24
```

Result Grid | | Filter Rows: | Export: | Wrap Cell Content:

	Vendor_Name	Invoice_ID	invoice_sequence	line_item_amount
▶	Wells Fargo Bank	12	1	50.00
	Wells Fargo Bank	12	2	75.60
	Wells Fargo Bank	12	3	58.40
	Wells Fargo Bank	12	4	478.00
	Zylka Design	78	1	1197.00
	Zylka Design	78	2	765.13

Result 3 x

Output

Action Output

#	Time	Action	Message
✓ 1	14:40:06	SELECT Vendor_Name, ili.Invoice_ID, invoice_sequence, line_item_amount FROM Invoice_Li...	6 row(s) returned

4. **(10 points)** Write a SELECT statement that returns the name, city, and state of each vendor that's located in a unique city and state. In other words, don't include vendors that have a city and state in common with another vendor. This should return 38 rows.

Sort the results by the vendor_state and vendor_city columns.

Hint: For this question, first create a subquery containing a list of vendor state and vendor city that are repeated (COUNT > 1). Then, write the main query whose results are NOT IN the results from the subquery.

```
25  -- Question 4
26  •  Select vendor_name, vendor_city, vendor_state
27      FROM VENDORS
28      WHERE (vendor_city, vendor_state)
29          NOT IN (SELECT vendor_city, vendor_state
30                  FROM vendors
31                  GROUP BY vendor_city, vendor_state
32                  HAVING COUNT(*) > 1 )
33      ORDER BY vendor_state, vendor_city;
```

Result Grid | | Filter Rows: | Export: | Wrap Cell

	vendor_name	vendor_city	vendor_state
▶	Diversified Printing & Pub	Brea	CA
	Vision Envelope & Printing	Gardena	CA
	Texaco	Inglewood	CA
	Publishers Marketing Assoc	Manhattan Beach	CA
	Blanchard & Johnson Associates	Mission Viejo	CA
	Blue Cross	Oxnard	CA
	Golden Eagle Insurance Co	San Diego	CA
	Towne Advertiser's Mailing Svcs	Santa Ana	CA
	The Windows Deck	Santa Barbara	CA
	Cal State Termite	Selma	CA

VENDORS 144 x

Output

Action Output

#	Time	Action	Message
✓ 1	19:44:44	Select vendor_name, vendor_cit...	38 row(s) returned

5. **(10 points)** Write a query to return one row per vendor, representing the vendor's oldest invoice (the one with the earliest date). Each row should include these four columns: vendor_name, invoice_number, invoice_date, and invoice_total. This should return 34 rows.

Sort the results by the vendor_name column.

```

42 • SELECT vendor_name, invoice_number, invoice_total, min(invoice_date) AS oldest_invoice
43 FROM VENDORS v
44 JOIN INVOICES i
45 ON v.vendor_id = i.vendor_id
46 GROUP BY vendor_name
47 ORDER BY vendor_name;

```

vendor_name	invoice_number	invoice_total	oldest_invoice
Abbey Office Furnishings	203339-13	17.50	2014-07-05
Bertelsmann Industry Svcs. Inc	509786	6940.25	2014-06-18
Blue Cross	547481328	224.00	2014-06-03
Cahners Publishing Company	587056	2184.50	2014-06-30
Cardinal Business Media, Inc.	133560	175.00	2014-06-22
Coffee Break Service	109596	41.80	2014-06-24
Compuserve	21-4748363	9.95	2014-05-03
Computerworld	367447	2433.00	2014-06-11
Data Reproductions Corp	40318	21842.00	2014-06-01
Dean Witter Reynolds	75C-90227	1367.50	2014-06-11
Digital Dreamworks	P02-3772	7125.34	2014-05-21
Dristas Groom & McCormick	94007005	220.00	2014-05-23
Edward Data Services	972110	207.78	2014-05-15
Evans Executone Inc	125520-1	95.00	2014-04-24
Federal Express Corporation	263253241	40.20	2014-04-10
Ford Motor Credit Company	9982771	503.20	2014-07-24
Franchise Tax Board	RTR-72-3662-X	1600.00	2014-05-25
Fresno County Tax Collector	P02-88D77S7	856.92	2014-05-03
Gostanian General Building	121897	450.00	2014-05-19
IBM	QP58872	116.54	2014-05-07
Ingram	31359783	1575.00	2014-06-03
Malloy Lithographing Inc	0-2058	37966.19	2014-05-28
Pacific Bell	111-92R-10096	16.33	2014-04-30
Pollstar	77290	1750.00	2014-05-13
Postmaster	CBM9920-M-T...	290.00	2014-06-23
Reiter's Scientific & Pro Books	C73-24	600.00	2014-07-19
Roadway Package System, Inc	25022117	6.00	2014-05-01
Suburban Propane	111897	16.62	2014-07-15
United Parcel Service	989319-457	3813.33	2014-04-08
Wakefield Co	97-1024A	356.48	2014-07-20
Wang Laboratories, Inc.	MABO1489	936.93	2014-06-21
Wells Fargo Bank	177271-001	662.00	2014-04-26
Yesmed, Inc	10843	4901.26	2014-05-11
Zylka Design	97/488	601.95	2014-04-24

Result 95 ×

Output

Action Output

#	Time	Action	Message
1	13:18:21	SELECT vendor name, invoice number, invoice total, min(invoice date) AS oldest invoice FROM VENDORS v JOIN INVOICES i ON v.vendor id = i....	34 row(s) returned

6. **(10 points)** Write a SELECT statement that returns these columns from the Invoices table:
- The invoice_total column.
 - A column that uses the FORMAT function to return the invoice_total column with 1 digit to the right of the decimal point.
 - A column that uses the CONVERT function to return the invoice_total column as an integer. A column that uses the CAST function to return the invoice_total column as an integer.

```
49  -- Question 6
50  •  SELECT invoice_total AS original_total,
51         FORMAT(invoice_total,1) AS format_version,
52         CONVERT(invoice_total,signed) AS convert_version,
53         CAST(invoice_total AS signed) AS cast_version
54  FROM INVOICES;
55
```

	original_total	format_version	convert_version	cast_version
▶	3813.33	3,813.3	3813	3813
	40.20	40.2	40	40
	138.75	138.8	139	139
	144.70	144.7	145	145
	15.50	15.5	16	16
	42.75	42.8	43	43
	172.50	172.5	173	173
	95.00	95.0	95	95
	601.95	602.0	602	602
	42.67	42.7	43	43
	42.50	42.5	43	43
	662.00	662.0	662	662
	16.33	16.3	16	16
	6.00	6.0	6	6
	856.92	856.9	857	857
	9.95	10.0	10	10
	10.00	10.0	10	10
	104.00	104.0	104	104
	116.54	116.5	117	117
	6.00	6.0	6	6
	4901.26	4,901.3	4901	4901
	108.25	108.3	108	108
	9.95	10.0	10	10
	1750.00	1,750.0	1750	1750
	129.00	129.0	129	129
	10.00	10.0	10	10
	207.78	207.8	208	208
	109.50	109.5	110	110
	450.00	450.0	450	450
	63.40	63.4	63	63

Result 96 ×

Output

Action Output

#	Time	Action	Message
✓ 1	13:20:16	SELECT invoice_total AS original_total, FORMAT(invoice_total,1) AS format_ver...	114 row(s) returned

7. **(10 points)** Write a SELECT statement that returns these columns from the Invoices table:

- The invoice_date column.
- A column that uses the CAST function to return the invoice_date column with its full date and time.
- A column that uses the CAST function to return the invoice_date column with just the year and the month.

```
56  -- Question 7
57  •  SELECT invoice_date,
58         CAST(invoice_date AS DATETIME) AS date_time,
59         DATE_FORMAT(CAST(invoice_date as DATE), '%Y/%M') AS simple_date
60  FROM INVOICES;
61
```

Result Grid | | Filter Rows: | Export: | Wrap Cell Content:

	invoice_date	date_time	simple_date
▶	2014-08-02	2014-08-02 00:00:00	2014/August
	2014-08-01	2014-08-01 00:00:00	2014/August
	2014-07-31	2014-07-31 00:00:00	2014/July
	2014-07-30	2014-07-30 00:00:00	2014/July
	2014-07-28	2014-07-28 00:00:00	2014/July
	2014-07-25	2014-07-25 00:00:00	2014/July
	2014-07-24	2014-07-24 00:00:00	2014/July
	2014-07-24	2014-07-24 00:00:00	2014/July
	2014-07-24	2014-07-24 00:00:00	2014/July
	2014-07-24	2014-07-24 00:00:00	2014/July
	2014-07-23	2014-07-23 00:00:00	2014/July
	2014-07-23	2014-07-23 00:00:00	2014/July
	2014-07-23	2014-07-23 00:00:00	2014/July
	2014-07-22	2014-07-22 00:00:00	2014/July
	2014-07-22	2014-07-22 00:00:00	2014/July
	2014-07-21	2014-07-21 00:00:00	2014/July
	2014-07-21	2014-07-21 00:00:00	2014/July
	2014-07-20	2014-07-20 00:00:00	2014/July
	2014-07-19	2014-07-19 00:00:00	2014/July
	2014-07-19	2014-07-19 00:00:00	2014/July
	2014-07-18	2014-07-18 00:00:00	2014/July
	2014-07-16	2014-07-16 00:00:00	2014/July
	2014-07-15	2014-07-15 00:00:00	2014/July

Result 98 ×

Output

Action Output

#	Time	Action	Message
✓ 1	13:23:45	SELECT invoice_date, CAST(invoice_date AS DATETIME) AS date_time, DATE...	114 row(s) returned

8. **(10 points)** Write a SELECT statement that returns these columns from the Invoices table:

- The invoice_number column
- The invoice_date column
- The invoice_date column plus 30 days
- The payment_date column
- A column named days_to_pay that shows the number of days between the invoice date and the payment date
- The number of the invoice date's month
- The four-digit year of the invoice date

When you have this working, add a WHERE clause that retrieves just the invoices for the month of May based on the invoice date, not the number of the invoice month. The query should return 29 rows.

```
62  -- Question 8
63  • SELECT invoice_number,
64          invoice_date,
65          invoice_date+30,
66          payment_date,
67          payment_date-invoice_date AS days_to_pay,
68          MONTH(invoice_date) AS invoice_date_month,
69          LEFT(invoice_date, 4) AS invoice_date_year
70  FROM INVOICES
71  WHERE MONTH(invoice_date)=5;
```

Result Grid

	invoice_number	invoice_date	invoice_date+30	payment_date	days_to_pay	invoice_date_month	invoice_date_year
▶	25022117	2014-05-01	20140531	2014-06-10	109	5	2014
	P02-88077S7	2014-05-03	20140533	2014-05-30	27	5	2014
	21-4748363	2014-05-03	20140533	2014-05-22	19	5	2014
	4-321-2596	2014-05-05	20140535	2014-06-05	100	5	2014
	963253242	2014-05-06	20140536	2014-06-05	99	5	2014
	QP58872	2014-05-07	20140537	2014-05-19	12	5	2014
	24863706	2014-05-10	20140540	2014-06-15	105	5	2014
	10843	2014-05-11	20140541	2014-05-29	18	5	2014
	963253235	2014-05-11	20140541	2014-06-09	98	5	2014
	21-4923721	2014-05-13	20140543	2014-05-28	15	5	2014
	77290	2014-05-13	20140543	2014-07-05	192	5	2014
	963253246	2014-05-13	20140543	2014-06-09	96	5	2014
	4-342-8069	2014-05-14	20140544	2014-06-13	99	5	2014
	972110	2014-05-15	20140545	2014-05-27	12	5	2014
	963253263	2014-05-16	20140546	2014-06-10	94	5	2014
	121897	2014-05-19	20140549	2014-07-03	184	5	2014

Result 100 ×

Output

Action Output

#	Time	Action	Message
✓ 1	13:26:00	SELECT invoice_number, invoice_date, invoice_date+30, payment_date, ...	29 row(s) returned

9. **(10 points)** Write a SELECT statement that returns these columns from the Vendors table:

- The vendor_name column.
- The vendor_name column in all capital letters.
- The vendor_phone column.
- A column that displays the last four digits of each phone number.

When you get that working right, add the columns that follow to the result set.

- The vendor_phone column with the parts of the number separated by dots, as in 555.555.5555. *Hint: For this, you can use the REPLACE function in MySQL.*
- A column that displays the first word in each vendor name if there is a space in the vendor_name and display the full vendor_name if there isn't a space in the vendor_name.

```
73  -- Question 9
74  •  SELECT  vendor_name,
75          UPPER(vendor_name) AS uppercase,
76          vendor_phone,
77          RIGHT(vendor_phone, 4) AS last_four,
78          REPLACE((REPLACE((REPLACE(vendor_phone, '(', '')), ')', '-')), '-', '.') AS phone_with_dots
79  CASE vendor_name
80      WHEN (LOCATE(' ', vendor_name) = 0)
81      THEN (SUBSTR(vendor_name, 1, (INSTR(vendor_name, ' '))))
82      ELSE vendor_name
83  END AS modified_vendor
84  FROM VENDORS;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	vendor_name	uppercase	vendor_phone	last_four	phone_with_dots	modified_vendor
▶	US Postal Service	US POSTAL SERVICE	(800) 555-1205	1205	800.555.1205	US
	National Information Data Ctr	NATIONAL INFORMATION DATA CTR	(301) 555-8950	8950	301.555.8950	National
	Register of Copyrights	REGISTER OF COPYRIGHTS	NULL	NULL	NULL	Register
	Jobtrak	JOBTRAK	(800) 555-8725	8725	800.555.8725	Jobtrak
	Newbrige Book Clubs	NEWBRIGE BOOK CLUBS	(800) 555-9980	9980	800.555.9980	Newbrige
	California Chamber Of Commerce	CALIFORNIA CHAMBER OF COMMERCE	(916) 555-6670	6670	916.555.6670	California
	Towne Advertiser's Mailing Svcs	TOWNE ADVERTISER'S MAILING SVCS	NULL	NULL	NULL	Towne

Result 145 x

Output

Action Output

#	Time	Action	Message
✓ 1	19:47:11	SELECT vendor_name, UPPER(...	122 row(s) returned

10. **(10 points)** Display all vendors that don't have invoices. Use NOT EXISTS. The result set should contain 88 rows. The result set should contain the following columns:
- Vendor ID
 - Vendor Name
 - Vendor State

```
87  -- Question 10
88  •  SELECT vendor_id, vendor_name, vendor_state
89      FROM VENDORS v
90      WHERE NOT EXISTS (SELECT *
91                          FROM INVOICES i
92                          WHERE v.vendor_id = i.vendor_id);
93
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Co

	vendor_id	vendor_name	vendor_state
▶	1	US Postal Service	WI
	2	National Information Data Ctr	DC
	3	Register of Copyrights	DC
	4	Jobtrak	CA
	5	Newbrige Book Clubs	NJ
	6	California Chamber Of Commerce	CA
	7	Towne Advertiser's Mailing Svcs	CA
	8	BFI Industries	CA
	9	Pacific Gas & Electric	CA
	10	Robbins Mobile Lock And Key	CA
	11	Bill Marvin Electric Inc	CA
	12	City Of Fresno	CA
	13	Golden Eagle Insurance Co	CA
	14	Expedata Inc	CA
	15	ASC Signs	CA
	16	Internal Revenue Service	CA
	17	Blanchard & Johnson Associates	CA
	18	Fresno Photoengraving Company	CA
	19	Crown Printing	CA

VENDORS 101 x

Output :

Action Output

#	Time	Action	Message
✓ 1	13:28:18	SELECT vendor_id, vendor_name, vendor_state FROM VENDORS v WHERE ...	88 row(s) returned

11. **(10 points)** Get the most recent invoice date for each vendor. Use subqueries and NOT joins. The query should return 122 rows. The result set should contain the following columns:
- Vendor Name
 - Latest Invoice Date

```

94  -- Question 11
95  •  SELECT vendor_name,
96      (SELECT max(invoice_date)
97      FROM INVOICES i
98      WHERE v.vendor_id = i.vendor_id) AS 'latest_invoice_date'
99  FROM VENDORS v;

```

Result Grid

vendor_name	latest_invoice_date
Abbey Office Furnishings	2014-07-05
American Booksellers Assoc	NULL
American Express	NULL
ASC Signs	NULL
Ascom Hasler Mailing Systems	NULL
AT&T	NULL
Aztek Label	NULL
Baker & Taylor Books	NULL
Bertelsmann Industry Svcs. Inc	2014-06-18
BFI Industries	NULL
Bill Jones	NULL
Bill Marvin Electric Inc	NULL
Blanchard & Johnson Associates	NULL
Blue Cross	2014-08-01
Blue Shield of California	NULL
Boucher Communications Inc	NULL
Cahners Publishing Company	2014-06-30
Cal State Termite	NULL
California Business Machines	NULL
California Chamber Of Commerce	NULL

Result 102 x

Output

Action Output

#	Time	Action	Message
✓ 1	13:30:13	SELECT vendor_name, (SELECT max(invoice_date) FROM INVOICES i ...	122 row(s) returned

12. **(10 points)** Get the largest invoice total for the top vendor in each state. Largest invoice total means MAX(sum of invoices). Display the following two columns in your resultset:

- Vendor State
- Max Sum of Invoices

The query should return 10 rows.

```
100  -- Question 12
101  •  SELECT vendor_state, MAX(sum_total) AS max_sum_of_invoices
102  FROM
103  (SELECT vendor_state, i.vendor_id, SUM(invoice_total) AS sum_total
104  FROM INVOICES i
105  INNER JOIN VENDORS v
106  ON v.vendor_id = i.vendor_id
107  GROUP BY vendor_id) AS vendor_total
108  GROUP BY vendor_state
109  ORDER BY vendor_state;
```

< Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	vendor_state	max_sum_of_invoices
▶	AZ	662.00
	CA	7125.34
	DC	600.00
	MA	1367.50
	MI	119892.41
	NV	23177.96
	OH	207.78
	PA	265.36
	TN	4378.02
	TX	2154.42

Result 149 x

Output

Action Output

#	Time	Action	Message
✓ 1	20:03:38	SELECT vendor_state, MAX(sum...	10 row(s) returned

13. (20 points) Write a SELECT statement that returns two columns: vendor_id and the largest unpaid invoice for each vendor. To get this, you can group the result set by the vendor_id column. This should return 7 rows.

Write a second SELECT statement that uses the first SELECT statement in its FROM clause. The main query should return a single value that represents the sum of the largest unpaid invoices for each vendor. If your query is correct, then you should get this single value as 22101.39.

```
107      -- Question 13
108      -- Part 1
109      •  SELECT vendor_id, MAX(invoice_total) AS unpaid_invoice
110      FROM INVOICES
111      WHERE invoice_total - credit_total - payment_total > 0
112      GROUP BY Vendor_ID;
```

<

Result Grid | | Filter Rows: | Export: | Wrap Cell Content:

	vendor_id	unpaid_invoice
▶	37	224.00
	72	85.31
	80	90.36
	83	579.42
	106	503.20
	110	20551.18
	123	67.92

Result 146 x

Output

Action Output ▼

#	Time	Action	Message
✓ 1	19:49:23	SELECT vendor_id, MAX(invoice...	7 row(s) returned

```
114 -- Part 2
115 • SELECT SUM(unpaid_invoice) AS sum_unpaid
116 FROM (SELECT vendor_id, MAX(invoice_total) AS unpaid_invoice
117        FROM INVOICES
118        WHERE invoice_total - credit_total - payment_total > 0
119        GROUP BY vendor_id) AS unpaid_invoices;
120
```

< Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	sum_unpaid
▶	22101.39

Output

Action Output

#	Time	Action	Message
✓ 1	19:51:33	SELECT SUM(unpaid_invoice) A...	1 row(s) returned



14. **(10 points)** Write a SELECT statement that answers this question: *Which invoices have a payment total that's greater than the average payment total for all invoices with a payment total greater than 0?*

Return the invoice_number and invoice_total columns for each invoice. This should return 20 rows.

Sort the results by the invoice_total column in descending order.

```
121  -- Question 14
122  •  SELECT invoice_number, invoice_total
123      FROM INVOICES
124      WHERE payment_total > (SELECT AVG(payment_total)
125                              FROM INVOICES
126                              WHERE payment_total > 0)
127      ORDER BY invoice_total DESC;
```


<

Result Grid |  Filter Rows: | Export:  | Wrap Cell

	invoice_number	invoice_total
▶	0-2058	37966.19
	P-0259	26881.40
	0-2060	23517.58
	40318	21842.00
	P02-3772	7125.34
	509786	6940.25
	10843	4901.26
	989319-457	3813.33
	989319-447	3689.99
	989319-437	2765.36
	367447	2433.00
	989319-467	2318.03

INVOICES 148 ✕

Output

 Action Output ▼

	#	Time	Action	Message
✓	1	19:55:22	SELECT invoice_number, invoic...	20 row(s) returned