



## Lab 2: Select, Table Joins, Unions Solution

1. Write a SELECT statement that returns three columns from the Vendors table: VendorName, VendorCity, and VendorState.

```
USE AP;  
SELECT VendorName, VendorCity, VendorState  
FROM Vendors;
```

A screenshot of the Microsoft SQL Server Management Studio (SSMS) interface. The 'Object Explorer' on the left shows the database structure for 'DESKTOP-S32HEE2\SQLSERVER (SQL Server 12.0.4100.1 - DE...)' with the 'AP' database selected. The 'Query Editor' in the center shows the SQL query: 

```
USE AP;  
SELECT VendorName, VendorCity, VendorState  
FROM Vendors;
```

 The query is highlighted with a red box. The 'Results' pane at the bottom shows the output of the query, which is a table with three columns: VendorName, VendorCity, and VendorState. The table contains 8 rows of data. The status bar at the bottom indicates 'Query executed successfully.' and '122 rows'.

	VendorName	VendorCity	VendorState
1	US Postal Service	Madison	WI
2	National Information Data Ctr	Washington	DC
3	Register of Copyrights	Washington	DC
4	Jobtrak	Los Angeles	CA
5	Newbrige Book Clubs	Washington	NJ
6	California Chamber Of Commerce	Sacramento	CA
7	Towne Advertiser's Mailing Svcs	Santa Ana	CA
8	BFI Industries	Fresno	CA

2. Write a SELECT statement that returns three columns from the Invoices table, named Number, Total, and Credits:

Number            Column alias for the InvoiceNumber column  
Total             Column alias for the InvoiceTotal column  
Credits           Column alias for the sum of the PaymentTotal and CreditTotal columns

And filter for invoices with an InvoiceTotal that's less or equal to \$200.

```
USE AP;  
SELECT InvoiceNumber AS Number,  
       InvoiceTotal AS Total,  
       PaymentTotal + CreditTotal AS Credits  
FROM Invoices  
WHERE InvoiceTotal <= 200;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL code:

```
USE AP;  
SELECT InvoiceNumber AS Number,  
       InvoiceTotal AS Total,  
       PaymentTotal + CreditTotal AS Credits  
FROM Invoices  
WHERE InvoiceTotal <= 200;
```

The query has been executed successfully, and the results are displayed in the Results pane. The results show 65 rows of data with the following columns: Number, Total, and Credits.

	Number	Total	Credits
1	263253241	40.20	40.20
2	963253234	138.75	138.75
3	2-000-2993	144.70	144.70
4	963253251	15.50	15.50
5	963253261	42.75	42.75
6	963253237	172.50	172.50
7	125520-1	95.00	95.00
8	263253250	42.67	42.67

The status bar at the bottom indicates that the query was executed successfully and returned 65 rows.

3. Write a SELECT statement that returns one column from the Vendors table named "Full Name". Create this column from the VendorContactFName and VendorContactLName columns. Format it as follows: last name, comma, first name. Sort the result set by last name from "A-Z".

USE AP;

SELECT VendorContactLName + ', ' + VendorContactFName AS [Full Name]

FROM Vendors

ORDER BY VendorContactLName ASC;

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor on the right contains the following SQL code:

```
USE AP;
SELECT VendorContactLName + ', ' + VendorContactFName AS [Full Name]
FROM Vendors
ORDER BY VendorContactLName ASC;
```

The query has been executed successfully, and the results are displayed in the Results pane. The results show a single column named "Full Name" with 122 rows of data. The first few rows are:

Full Name
Aaronsen, Thom
Aileen, Joan
Alberto, Francesco
Alexis, Alexandro
Alondra, Zev
Angelica, Nashalle
Antavius, Troy
Anthoni, Kaitlyn

The status bar at the bottom indicates that the query was executed successfully and that there are 122 rows of data.

4. Write a SELECT statement that determines whether the PaymentDate column of the Invoices table has any valid values. To be valid, PaymentDate must be a null value if there is a balance due and a non-null value if there is no balance due. Code a compound condition in the WHERE clause that tests for these conditions. (Balance: InvoiceTotal minus the sum of PaymentTotal and CreditTotal)

```
USE AP;
SELECT *
FROM Invoices
WHERE ((InvoiceTotal - PaymentTotal - CreditTotal > 0) AND PaymentDate IS NULL)
OR
((InvoiceTotal - PaymentTotal - CreditTotal <= 0) AND PaymentDate IS NOT NULL);
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL code:

```
USE AP;
SELECT *
FROM Invoices
WHERE ((InvoiceTotal - PaymentTotal - CreditTotal > 0) AND PaymentDate IS NULL)
OR
((InvoiceTotal - PaymentTotal - CreditTotal <= 0) AND PaymentDate IS NOT NULL);
```

The Object Explorer on the left shows the database structure for 'DESKTOP-S32HEE2\SQLSERVER (SQL Server 12.0.4100.1 - DE)'. The 'Invoices' table is highlighted under the 'dbo' schema.

The Results pane shows the following data:

	InvoiceID	VendorID	InvoiceNumber	InvoiceDate	InvoiceTotal	PaymentTotal	Credit Total	TermsID	InvoiceDueDate
1	1	122	989319-457	2015-12-08 00:00:00	3813.33	3813.33	0.00	3	2016-01-08 00:00:00
2	2	123	263253241	2015-12-10 00:00:00	40.20	40.20	0.00	3	2016-01-10 00:00:00
3	3	123	963253234	2015-12-13 00:00:00	138.75	138.75	0.00	3	2016-01-13 00:00:00
4	4	123	2-000-2993	2015-12-16 00:00:00	144.70	144.70	0.00	3	2016-01-16 00:00:00
5	5	123	963253251	2015-12-16 00:00:00	15.50	15.50	0.00	3	2016-01-16 00:00:00
6	6	123	963253261	2015-12-16 00:00:00	42.75	42.75	0.00	3	2016-01-16 00:00:00
7	7	123	963253237	2015-12-21 00:00:00	172.50	172.50	0.00	3	2016-01-21 00:00:00

The status bar at the bottom indicates 'Query executed successfully.' and '114 rows'.

5. Write a SELECT statement that returns four columns: VendorName, VendorZipCode, DefaultAccountNo and AccountDescription from the Vendors table and GLAccounts table. The result set should have one row for each vendor, with the zipcode, account number and account description for that vendor's default account number. And filter for Vendors whose AccountDescription is Computer Equipment and sort the result set by VendorName from A to Z.

USE AP;

SELECT VendorName, VendorZipCode, DefaultAccountNo, AccountDescription

FROM Vendors

JOIN GLAccounts

ON Vendors.DefaultAccountNo = GLAccounts.AccountNo

AND GLAccounts.AccountDescription = 'Computer Equipment'

ORDER BY VendorName ASC;

The screenshot shows the Microsoft SQL Server Management Studio interface. The SQL query editor contains the following query:

```
USE AP;  
SELECT VendorName, VendorZipCode, DefaultAccountNo, AccountDescription  
FROM Vendors  
JOIN GLAccounts  
ON Vendors.DefaultAccountNo = GLAccounts.AccountNo  
AND GLAccounts.AccountDescription = 'Computer Equipment'  
ORDER BY VendorName ASC;
```

The query results are displayed in the Results pane, showing a table with 5 rows and 4 columns: VendorName, VendorZipCode, DefaultAccountNo, and AccountDescription.

	VendorName	VendorZipCode	DefaultAccountNo	AccountDescription
1	American Express	90096	160	Computer Equipment
2	IBM	94161	160	Computer Equipment
3	Micro Center	43221	160	Computer Equipment
4	Wang Laboratories, Inc.	91185	160	Computer Equipment
5	Wells Fargo Bank	85038	160	Computer Equipment

The status bar at the bottom indicates "Query executed successfully." and "5 rows".

6. Write a SELECT statement that returns two columns: VendorName and Name (A concatenation of VendorContactFName and VendorContactLName, with a comma in between). The result set should have one row for each vendor whose contact has the same last name as another vendor's contact. Sort the final result set by VendorName from A to Z.

USE AP;

```
SELECT DISTINCT v1.VendorName,  
                v1.VendorContactFName + ', ' + v1.VendorContactLName AS Name  
FROM Vendors AS v1 JOIN Vendors AS v2  
ON (v1.VendorID <> v2.VendorID) AND  
   (v1.VendorContactLName = v2.VendorContactLName)  
ORDER BY VendorName ASC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL code:

```
USE AP;  
SELECT DISTINCT v1.VendorName,  
                v1.VendorContactFName + ', ' + v1.VendorContactLName AS Name  
FROM Vendors AS v1 JOIN Vendors AS v2  
ON (v1.VendorID <> v2.VendorID) AND  
   (v1.VendorContactLName = v2.VendorContactLName)  
ORDER BY VendorName ASC;
```

The query results are displayed in the Results pane, showing two rows:

	VendorName	Name
1	Blue Shield of California	Kylie, Smith
2	Roadway Package System, Inc	Sam, Smith

The status bar at the bottom indicates "Query executed successfully." and "2 rows".

7. Use the UNION operator to generate a result set consisting of two columns from the Vendors table: VendorName and VendorState. If the vendor is in New York, the VendorState value should be "NY"; otherwise, the vendorState value should be "Outside NY". Sort the final result set by VendorName from Z to A.

```
USE AP;
SELECT VendorName, VendorState
FROM Vendors
WHERE VendorState = 'NY'
UNION
SELECT VendorName, 'Outside NY'
FROM Vendors
WHERE VendorState <> 'NY'
ORDER BY VendorName DESC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL query:

```
USE AP;
SELECT VendorName, VendorState
FROM Vendors
WHERE VendorState = 'NY'
UNION
SELECT VendorName, 'Outside NY'
FROM Vendors
WHERE VendorState <> 'NY'
ORDER BY VendorName DESC;
```

The query results are displayed in the Results pane, showing a table with two columns: VendorName and VendorState. The results are sorted by VendorName in descending order.

VendorName	VendorState
Zylka Design	Outside NY
Zip Print & Copy Center	Outside NY
Zee Medical Service Co	Outside NY
Yesmed, Inc	Outside NY
Yale Industrial Trucks-Fresno	Outside NY
Wells Fargo Bank	Outside NY
Wang Laboratories, Inc.	Outside NY
Wakefield Co	Outside NY

The status bar at the bottom indicates that the query was executed successfully and returned 122 rows.

8. Write a SELECT statement that returns two columns from the GLAccounts table: AccountNo and AccountDescription. The result set should have one row for each account number that has never been used (i.e. AccountNo in InvoiceLineItems table has null value). Sort the final result set by AccountNo. (HINT: join GLAccounts table and InvoiceLineItems table.)

USE AP;

SELECT GLAccounts.AccountNo, AccountDescription

FROM GLAccounts LEFT JOIN InvoiceLineItems

ON GLAccounts.AccountNo = InvoiceLineItems.AccountNo

WHERE InvoiceLineItems.AccountNo IS NULL

ORDER BY GLAccounts.AccountNo;

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor window displays the following SQL query:

```
USE AP;
SELECT GLAccounts.AccountNo, AccountDescription
FROM GLAccounts LEFT JOIN InvoiceLineItems
ON GLAccounts.AccountNo = InvoiceLineItems.AccountNo
WHERE InvoiceLineItems.AccountNo IS NULL
ORDER BY GLAccounts.AccountNo;
```

The query is executed successfully, and the results are displayed in the Results pane. The results show a list of account numbers and descriptions that have never been used in the InvoiceLineItems table.

AccountNo	AccountDescription
100	Cash
110	Accounts Receivable
120	Book Inventory
162	Capitalized Lease
167	Software
181	Book Development
200	Accounts Payable
205	Royalties Payable

The status bar at the bottom indicates that the query executed successfully and returned 54 rows.