

SOS Query Examples

Synergistic Office Solutions
Revision: July 13, 2007

Table of Contents

Total Credits without Sort Codes, by Primary Provider	2
Charges by Sort Code	4
Payments by Place of Service and Year	5
Charges by Place of Service and Year	6
Credit Breakdown by Payor Type for Service Date Range	7
Last Payments	9
Collections by Service for Period	10
Scheduler Reminders	11
Payment Latency for Specified Payor Number	12
Intake Count for Period	13
Patient List with Dates of Referral, Intake, and Discharge	14
Patients with Zero Balance	16
Patient List by Rendering Provider	17
Count Service Dates for Patients, Grouped by Rendering Provider	18
Active Patients with No Service in Last 90 Days	19
Mailing Labels for Patients with Specific Insurance Payors	20
List Patients Based on True Date of First Service	21
Patients by Last Four Digits of Social Security Number	22
Number of Services for Patients by Service Code and Rendering Provider	23
Referring Physicians for Payors with ID's and Patient Count	24
Referral Source, Patient Count and Charges By Month	25
Length of Stay by Provider	27
Admissions by Zipcode	28
Active Patient Count For Date Range	29
Patient Count for Period by SortCode	30
Export Appointment Data for Automated Reminder System	31
List Patients with No Primary Diagnosis by Primary Provider	32
Expiring Authorizations by Primary Provider	33

SOS Query Examples

With even a very basic knowledge of SQL, the industry standard database query language, you can multiply by many times the usefulness of the data you have stored in SOS. Along with your primary applications such as Office Manager and Case Manager, SOS also provides tools with which you can run queries against your database. To get started, please review the document on this subject in the document library of the SOS web site. This link will take you there:

<http://www.sosoft.com/fod/doc404-usingdbisql.pdf>

Documents you may find helpful include the rather dated, but still relevant, annotated data dictionary:

<http://www.sosoft.com/files/downloads/sosddct.pdf>

and a set of entity diagrams showing links among tables for the most commonly used data:

<http://www.sosoft.com/files/downloads/sosdpics.pdf>

It is most important that you review the final chapter (Accessing SOS Data from Other Programs) in *sostech.pdf*, located in your SOS folder. This chapter explains the basics of the database organization. You will be lost without that foundation for all but the simplest queries.

The current document contains, in no particular order, samples of queries that we have created in response to requests from SOS users. Many include commentary that explains certain aspects of the query that may be especially interesting or worthy of note.

Use the search feature in Acrobat (the binocular icon on the toolbar) to search for keywords that are of interest to you.

As time allows I will add new examples, so if you already have a copy be sure to check the revision date against the current document to see if you are up to date.

Seth Krieger
President
Synergistic Office Solutions, Inc.

Total Credits without Sort Codes, by Primary Provider

Title	Total Credits without Sort Codes, by Primary Provider
Date	5/6/2001
Author	Seth Krieger
Product	OM
Keywords	Credit. Sort Code

```
SELECT
p.provcode,
r.provlname,
SUM(amount)
FROM
journal j,
patients p,
providers r
WHERE
j.ptnum=p.ptnum AND
p.providernum=r.providernum AND
j.trandate BETWEEN '2000-03-01' and '2000-03-15' AND
j.sortcode is null
GROUP BY
patients.provcode, providers.provlname
```

Charges by Sort Code

Title	Charges by Sort Code
Date	5/9/2001
Author	Seth Krieger
Product	OM
Keywords	Sort Code, Charges

```
SELECT
b.lucode as Code,
b.remark as Description,
SUM(a.amount) as Charges
FROM
rv_charges a JOIN lookups b ON a.sortcode=b.lunum
WHERE
trandate between '1999-01-01' and TODAY()
GROUP BY b.lucode,b.remark
```

Payments by Place of Service and Year

Title	Payments by Place of Service and Year
Date	10/3/2001
Author	Seth Krieger
Product	OM
Keywords	Payment, POS, Place of Service

```
SELECT year(c.trandate) as YEAR,  
defcode as POS,  
sum(crsplamt) as "TOTAL PAID"  
FROM jcrsplits a  
join jcredits b  
join journal c  
join rv_charges d on a.chgsplnum=d.chgsplnum  
join poscodes e on d.poscodenum=e.poscodenum  
WHERE b.credtype <> 'Adjustment'  
GROUP BY year,pos  
ORDER BY year,pos
```

Charges by Place of Service and Year

Title	Charges by Place of Service and Year
Date	10/3/2001
Author	Seth Krieger
Product	OM
Keywords	Charge, Place of Service, POS

```
SELECT sum(chgsplamt) as "totalcharge",  
defcode as "POS",  
year(trandate) as "year"  
FROM rv_charges a join poscodes b on a.poscodenum=b.poscodenum  
GROUP BY POS,year
```

Credit Breakdown by Payor Type for Service Date Range

Title	Credit breakdown by payor type for service date range
Date	3/26/2003
Author	Seth Krieger
Product	OM
Keywords	Credits, Payments, Payor Type

// Breaks down credits applied to services in a date range
// by type and groups by payor type

```
Select
payortype,
Credtype,
count(distinct cre_jnum) as "Number",
sum(crsplamt) as "Paid"
From
rv_creditsplits
Where
srv_date between '2001-01-01' and '2001-01-31'
and systranflag not in ('TT','TF')
group by
credtype,payortype
```

Credit Breakdown by Payor for Service Date Range

Title	Credit Breakdown by Payor for Service Date Range
Date	3/26/2003
Author	Seth Krieger
Product	OM
Keywords	Credits, Payments, Payor

// Breaks down credits applied to services in a date range
// groups by insurance payor

```
Select
payorname,
payorname,
credtype,
count(distinct cre_jnum) as "Number",
sum(crsplamt) as "Paid"
From
rv_creditsplits
Where
srv_date between '2001-01-01' and '2001-01-31'
and payortype = 'I'
and systranflag not in ('TT','TF')
group by
credtype,payorname,payorname
```

Last Payments

Title	Last Payments
Date	1/12/2005
Author	Seth Krieger
Product	OM
Keywords	Payment

You asked for a report that includes last payment information. This information is readily available in the database, but we have never included it in any of the standard reports.

```
SELECT
(a.Lastname + ', ' + a.Firstname) AS "PtName",
a.id,
TRIM(c.firstname + ' '+c.payorname) AS "Payor",
b.lastpaydate,
b.lastpayamt,
b.postedbal
FROM
patients a
JOIN ptpayers b ON a.ptnum = b.ptnum
JOIN payors c ON b.payorname = c.payorname
WHERE
postedbal > 0
```

Collections by Service for Period

Title	Collections by Service for Period
Date	2/17/2005
Author	Seth Krieger
Product	OM
Keywords	Payment, Collection, Service

"I have a need for a report that would breakdown monies collected for each service code over a specified time period."

```
// this one selects based on the service date
SELECT srvcode, sum(crsplamt) AS Payments
FROM rv_creditsplits
WHERE CredType <> 'Adjustment'
AND Srv_Date BETWEEN '2001-01-01' and '2001-12-31'
GROUP BY srvcode
```

```
// this one selects based on the date payment was applied
SELECT srvcode, sum(crsplamt) AS Payments
FROM rv_creditsplits
WHERE CredType <> 'Adjustment'
AND DateApplied BETWEEN '2001-01-01' and '2001-12-31'
GROUP BY srvcode
```

```
// this one selects based on the date payment was received
SELECT srvcode, sum(crsplamt) AS Payments
FROM rv_creditsplits
WHERE CredType <> 'Adjustment'
AND Cre_Date BETWEEN '2001-01-01' and '2001-12-31'
GROUP BY srvcode
```

Scheduler Reminders

Title	Scheduler Reminders
Date	2/1/2005
Author	Seth Krieger
Product	Appointment Scheduler
Keywords	Scheduler, Reminder

"Is there an easy query that would simply print all the swcheduler reminders off the patient info form p.2?"

```
SELECT LastName, FirstName, ID, NoSchedReason
FROM Patients
WHERE Flag = 0
ORDER BY LastName, FirstName
```

The Flag condition limits to active patients. If you want only those that have a reminder entered, add another condition. Note that the " at the end of line three is two apostrophes, not a single quotation mark.

```
SELECT LastName, FirstName, ID, NoSchedReason
FROM Patients
WHERE Flag = 0 AND NoSchedReason <> ''
ORDER BY LastName, FirstName
```

Finally, the one below combines the name and id into a single field to make the output a bit cleaner:

```
SELECT LastName+', '+ FirstName+' / '+ ID AS "Patient", NoSchedReason
FROM Patients
WHERE Flag = 0 AND NoSchedReason <> ''
ORDER BY LastName, FirstName
```

Payment Latency for Specified Payor Number

Title	Payment Latency for Specified Payor Number
Date	1/12/2005
Author	Seth Krieger
Product	OM
Keywords	Payment, Payor

In order to determine how many days it takes a payor to pay your claims, you can examine firstbilled and paiddate values in JChgSplits. The following query will tell you these dates and compute the number of days between them. Just change the payor number (get the payor number from the second column in Lookups > Ins Carriers/Plans) and the desired service date range in the WHERE clause.

```
SELECT a.jnum,c.amount AS "Fee",
a.chgsplamt AS "SplitToPayor",
a.firstbilled,
a.paiddate,
datediff(day,a.firstbilled,a.paiddate) AS "PaymentDays"
FROM jchgsplits a
JOIN ptpayors b ON a.ptpayornum = b.ptpayornum
JOIN journal c ON a.jnum = c.jnum
WHERE b.payornum = 101
AND c.trandate BETWEEN '2004-1-1' AND '2004-12-31'
```

Intake Count for Period

Title	Intake Count for Period
Date	8/27/2004
Author	Seth Krieger
Product	OM, CM
Keywords	Intake Admission

"Is there a way that I might be able to run a report that will tell me the exact number of intakes in a given period of time?"

```
SELECT
Patients.LicNum, Providers.ProvCode, Count(*) as "Count"
FROM
Patients
LEFT OUTER JOIN Providers ON Patients.ProviderNum = Providers.ProviderNum
WHERE
Patients.IntakeDate BETWEEN '2003-01-01' AND '2003-12-31'
GROUP BY
Patients.LicNum, Providers.ProvCode
ORDER BY
Patients.LicNum, Providers.ProvCode
```

The "Licnum" represents the data set number. If you have only one data set, you can leave that out.

Patient List with Dates of Referral, Intake, and Discharge

Title	Patient List with Dates of Referral, Intake, and Discharge
Date	7/9/2004
Author	Seth Krieger
Product	OM, CM
Keywords	Referral, Intake, Discharge

"I need a listing of:

Active clients for a specific Patient category (one of our sites, #103), that will list:

1. *Date of referral (and I want to be able to specify all referrals on this date or later (in this case 5/1/04)*
2. *Date of intake (and I want to be able to specify all intakes on this date or later, in this case 5/1/04)*
3. *Date of discharge (and leaving blank or whatever if client has NOT been discharged)*
4. *The entry in User-Sort 3 (being able to specify which one), in this case "MHP" or "GZ" or "PA" or "MED" or "MED/MHP" or "MED/INS"*
5. *The entry (if there is one) in User-defined (client) field #3"*

As you have already realized, SQL is very "plain language" ish. It basically makes sense when you read it, even if you aren't familiar with the language -- at least up to a point. The one thing that might throw you a little are the correlation names. Basically, we give table names a short alias to avoid retyping the name over and over. Here is the query:

```
SELECT a.LastName, a.FirstName, a.ID, a.ReferralDate, a.IntakeDate, a.DischargeDate, a.UserSort3,
c.Fld3
FROM Patients a
JOIN PtCategs b ON a.ptcategnum = b.ptcategnum
LEFT OUTER JOIN UDDDataPt c ON a.UDDDataNum = c.UDDDataNum
WHERE b.CategCode = 'ABC'
AND a.flag = 0
AND a.LicNum = 103
AND a.ReferralDate >= '2004-05-01'
AND a.IntakeDate >= '2004-05-01'
AND a.UserSort3 IN ('MHP','GZ','PA','MED','MED/MHP','MED/INS')
ORDER BY a.LastName, a.FirstName, a.ID
```

Notice that each table named in the FROM clause is followed by a single letter. That creates the alias, which can be used even in the SELECT list that precedes the FROM clause. By saying...

```
FROM Patients a
```

We can now use "a" wherever complete syntax would dictate that we would have to use the entire table name. We can therefore do...

```
ORDER BY a.LastName, a.FirstName, a.ID
```

Instead of...

```
ORDER BY Patients.LastName, Patients.FirstName, Patients.ID
```

Additional notes:

The other thing that is not intuitive, and that requires a good understanding of the structure of the database with which you are working, is when to relate tables with a JOIN rather than a LEFT OUTER JOIN or a RIGHT OUTER JOIN. In your query we are JOINing Patients to PtCategs because you have specified that you will be selecting based on membership in a particular category. If a patient has not been assigned to a category, you don't want them in the result set. The JOIN essentially makes a match between Patients and PtCategs required. That is, there must be matching data in the specified fields or the row will not be included in the results. (For more on how we join tables together, the last chapter of OMTECH.PDF or SOSTECH.PDF in your SOS folder is an absolute must.) I don't know what your categories are, so I just made up a category code "ABC" and use that as one of the selection criteria. A LEFT OUTER JOIN between these tables with everything else the same, would result in both ABC category patients but also those patients who have not been assigned to any category -- VERY different results! In the case of the User Defined Data, which is held in the related table UDDataPt, we want to include those patients who do have a UDData record, but also those who may not, so we use the OUTER type of JOIN. Some people "get" this stuff right away; others don't. Any intro to SQL or relational databases should include a more thorough explanation.

Let me go through the WHERE conditions in order. Think of these as your filter or selection criteria.

WHERE

b.CategCode = 'ABC'

...the patient must be assigned to patient category "ABC" on the Additional tab of the Patient form

AND a.flag = 0

...the patient must be in the Active Patients list

AND a.LicNum = 103

...the patient must be in data set (also called "sublicense") number 103

AND a.ReferralDate >= '2004-05-01'

...the patient must have a referral date and it must be 5/1/04 or later

AND a.IntakeDate >= '2004-05-01'

...the patient must have an intake date and it must be 5/1/04 or later

AND a.UserSort3 IN ('MHP','GZ','PA','MED','MED/MHP','MED/INS')

...the patient must have a value in the UserSort 3 field and it must be one of the values specified in the list between the parentheses.

Patients with Zero Balance

Title	Patients with Zero Balance
Date	7/9/2004
Author	Seth Krieger
Product	OM
Keywords	Zero Balance

"Is there a way to print out a list of the active clients who have zero balances, without printing out a list of ALL of the active clients? Having this list would make it easy to find the clients who could be moved to the "inactive" list."

```
// list of active accounts...
SELECT LastName, FirstName, ID
FROM patients
WHERE flag = 0 AND dischargedate IS NULL
ORDER BY LastName, FirstName, ID
```

```
// list of active accounts with zero balance...
SELECT LastName, FirstName, ID
FROM Patients a JOIN PatientBalance b on a.ptnum = b.ptnum
WHERE b.ptbalance = 0 and a.flag = 0 AND a.dischargedate IS NULL
ORDER BY LastName, FirstName, ID
```

```
//list of discharged patients that have no balance...
SELECT LastName, FirstName, ID
FROM Patients a
JOIN PatientBalance b on a.ptnum = b.ptnum
WHERE b.ptbalance = 0
AND a.flag = 0
AND a.dischargedate IS NOT NULL
ORDER BY LastName, FirstName, ID
```

If you wanted to list by a particular Patient Category, you would have to use a second table, PtCategs, and you would have to add the additional condition to the WHERE clause of the query. Here's an example, assuming the first patient category you want is "ONE":

```
SELECT a.LastName, a.FirstName, a.ID
FROM patients a
JOIN ptcategs b ON a.ptcategnum = b.ptcategnum
WHERE a.flag = 0 AND a.dischargedate IS NULL AND b.categcode = 'ONE'
ORDER BY LastName, FirstName, ID
```

The other approach is to add the patient category to the output and sort the data on that value as well. You would then divide up the report in your word processor before outputting it:

```
SELECT b.CategCode, a.LastName, a.FirstName, a.ID
FROM patients a
JOIN ptcategs b ON a.ptcategnum = b.ptcategnum
WHERE a.flag = 0 AND a.dischargedate IS NULL AND b.categcode = 'ONE'
ORDER BY b.CategCode, a.LastName, a.FirstName, a.ID
```

Patient List by Rendering Provider

Title	Patient List by Rendering Provider
Date	9/5/2003
Author	Seth Krieger
Product	OM, CM
Keywords	Patient List, Rendering Provider

Here is a query that will show the patient on the list for every provider who ever provided services for that patient (so that patient could show up on multiple lists). The output is sorted by rendering provider, but there are no page breaks or other fancy stuff.

```
SELECT DISTINCT
d.provcode,a.lastname, a.firstname, a.id
FROM
patients a JOIN journal b JOIN jcharges c JOIN providers d
WHERE
a.flag = 0
GROUP BY
provcode,lastname, firstname, id
ORDER BY
provcode,lastname, firstname, id
```

"a.flag = 0" restricts the output to active patients
If you wanted to output a single provider you would add this on the line below "a.flag"

```
AND d.provcode = 'ABC'
```

Count Service Dates for Patients, Grouped by Rendering Provider

Title	Count Service Dates for Patients, Grouped by Rendering Provider
Date	5/6/2003
Author	Seth Krieger
Product	OM
Keywords	Service, Charge

“Any suggestions on how to create a query that would provide a count of the Dates Of Service for each patient within a specified date range? I would also need either rendering provider or primary

```
SELECT
id, provcode, count(distinct trandate) as SrvDateCount
FROM
rv_charges
WHERE
trandate between '2001-01-01' and '2001-03-31'
GROUP BY
id, provcode
```

Obviously set the date range correctly. Note that the DISTINCT in this case must be within the parens with the column you want the distinct count of.

Active Patients with No Service in Last 90 Days

Title	Active patients with no service in the last 90 days
Date	7/13/05
Author	Seth Krieger
Product	OMWin
Keywords	Active patient last service

```
SELECT "name/id", lfeedate as "Last Service", priprvcode as "Primary Provider"  
FROM rv_patients  
WHERE flag = 0 and lfeedate < (current date - 90)
```

You can, of course, change the date range by altering the number in the final expression. The condition "flag = 0" restricts the output to active patients. The quotes around the column name "name/id" are required so that the query processor does not try to divide name by id. When you enclose a string within quotes it is treated as a column name, so embedded special characters are ignored. In this case the view RV_PATIENTS contains an expression that creates this convenient compound value from the firstname, lastname, and id fields. Here is what the output looks like:

name/id	Last Service	Primary Provider
CHAPPIE, Regina / 1002	2003-08-01	KP
CHESTER, Kenny / 1006	2001-04-19	KP
EATON, Lyle / 1008	2001-04-18	KP
GRAHAM, Jehu / 1011	2001-04-24	KP
GIACOMO, Kent / 1012	2001-04-30	KP
MCDADE, Helen / 1014	2001-04-26	KP
PRITCHETT, Jodi / 1017	2001-01-18	KP
BNOINSBAL, Arnold / 1021	2003-04-12	KP
CARTWRIGHT, Modesto / 1022	2003-08-01	KP

Mailing Labels for Patients with Specific Insurance Payors

Title	Mailing labels for patients with specific insurance payors
Date	5/20/2005
Author	Seth Krieger
Product	OMWin
Keywords	Insurance payor provider mailing label

We need to send letters to all of our psychiatrists' patients categorized by insurance payor. How can I get mailing labels for these letters? The scenario is that our docs have removed themselves from the Magellan and United Behavioral Health panels and will only accept fee-for-services from those patients. Obviously we need to inform the patients - several times and in several ways. The first step is to identify them and prepare labels for the letters to be sent.

OK, so we can identify the patients using the primary provider code. Good. Let's start with this query. I'll include the payorname in the output so you can check the export. You can ignore that column when you read the data into Word to create your labels.

```
SELECT
(a.firstname + ' '+a.lastname) as "name",
a.addr1 as "addr1",
a.addr2 as "addr2",
(a.city+', '+a.state+', '+a.zip) as "addr3",
b.payorname as "payor"
FROM
rv_patients a join rv_policies b on a.ptnum = b.ptnum
WHERE
a.flag = 0 and
a.priprvcode='prv' and
(b.payorname like '%magellan%' or b.payorname like '%ubh%') and
(b.active is null or b.active <= current date) and
(b.inactive is null or b.inactive >= current date)
; output to c:\sos\labels.csv format ascii
```

The resulting csv file can be loaded into Excel for additional manipulation, if you like, or can be used as is. If you check the help in Word regarding mailing labels, that should take you the rest of the way. Oh, note the fourth line up from the bottom. This one is where I am specifying the payor names. If you have a UBH payor that you have spelled out as "United Behavioral Health" you should add another "or..." to that condition, or change the '%ubh%' condition. This expression means that it will include payors with the characters "ubh" anywhere in the name. That could result in some payors you do not want, such as a payor named "Flubhead" or something like that. You should be able to fine tune that with a little experimentation. On the fifth line up from the bottom you must replace 'prv' with your psychiatrist's provider code. The other conditions restrict the output to active patients with active matching policies as of today's date.

List Patients Based on True Date of First Service

Title	List patients based on true date of first service
Date	05/26/2005
Author	Seth Krieger
Product	OMWin
Keywords	Intake HAVING alias correlation name aggregate MIN

A user complained that he could not use the intake date recorded in the system because it did not reflect the true date of first service. (The intake date will default to the current system date when initializing a new patient account unless you enter a date manually.)

The following query selects accounts based on the date of the earliest transaction date found in the patient ledger. It will output the accounts that had their first service entry during the date range specified:

```
SELECT
Lastname + ', ' + Firstname + ' / ' + id AS "Account",
MIN(trandate) AS "Intake"
FROM
Journal a JOIN Patients b ON a.ptnum = b.ptnum
WHERE
a.trantype = 'S'
GROUP BY
account
HAVING
Intake BETWEEN '2005-04-01' AND '2005-04-30'
```

Note the use of the aggregate function MIN, along with the GROUP BY to determine the earliest service date for each patient. The condition “a.trantype = ‘S’” restricts the transactions to just charge entries. The query also uses correlation names to alias the name and id expression (aliased as “account”), the MIN(trandate) expression as “intake”, and the tables (aliased as “a” and “b”).

Also note that we have conditions in both a WHERE clause and a HAVING clause. Conditions that examine the results of aggregate expressions must be placed in a HAVING clause following the GROUP BY. The rule of thumb here is that you can put your condition in the WHERE if it is looking at values that exist in the original rows of the table, but if you want to restrict output based on values that don't exist until the data is grouped, then you must put the condition in the HAVING clause.

In this case, the trantype value is present in the journal table, so that condition goes in the WHERE. On the other hand, the minimum date can't be determined until we have grouped the rows in journal by patient account. We therefore stick our “intake” (MIN(trandate)) condition in the HAVING, after the GROUPED BY.

Patients by Last Four Digits of Social Security Number

Title	Find Patients Based on Last Four Digits of Social Security Number
Date	07/25/2005
Author	Seth Krieger
Product	All
Keywords	RIGHT substring search social security alias expression

Recently we had an audit done by an outside agency that used the last 4 digits of the SS# to identify the patient charges they chose to audit. We have a need to take a look at several of those charts, but the only identifying data we now have is the last 4 SS# digits. How can I retrieve the patients with just this information?

The following query should provide you with what you need. Just replace 1234 in the last line with the digits you want to search. Don't neglect the single quotes around the number:

```
SELECT
lastname, firstname, id, socsec
FROM
patients
WHERE
RIGHT(socsec,4) = '1234'
```

If you have a list of numbers you would like to do in a single shot, you can modify it as follows:

```
SELECT
lastname, firstname, id, socsec, RIGHT(socsec,4) as "last4"
FROM
patients
WHERE
last4 IN = ('1234','5678','2468','4321')
```

or, for a complete list of patients, sorted by the last four digits:

```
SELECT
RIGHT(socsec,4) as "last4", lastname, firstname, id, socsec
FROM
patients
ORDER BY last4
```

Number of Services for Patients by Service Code and Rendering Provider

Title	Number of Services for Patients by Service Code and Rendering Provider
Date	08/01/2005
Author	Seth Krieger
Product	OMWin
Keywords	Service cpt provider intake GROUP ORDER UPPER alias

Here is the information I need:

Clients name: (using the Intake as of Sept. 2003)

Providers name:

Number of times in Individual sessions (CPT 90806 Code: IP)

Number of times in Group (CPT 90853 Code: Grp)

Is that possible to get the clients name and the number of sessions they have been in for individual and group sessions?

I took a little liberty with your request to show the number of sessions of all rendered services, but this can be trimmed down, if you like. You did not indicate whether the provider was the rendering or primary. I used rendering, so you will see the number of sessions for each provider for each patient (if there are multiple providers rendering services to a single patient). The output of the query will go to a Lotus-format file called sessioncount.wks in the C:\SOS folder. Excel will open this file without complaint. If you want to change the file name or location, modify the last line of the query. (Note that you can un-comment line 15 if you want to restrict the results to just the IP and GRP service codes. I would suggest running the more inclusive query first in case the providers are using different codes than you expect.)

Note that we use JOIN rather than LEFT OUTER JOIN to link the tables in this case. No data will be omitted because all these values and rows must be present for every charge entry. You can't have a charge without a patient (ptnum), provider (providernum), or service (servicenum).

```
SELECT
(UPPER(a.LastName)+' '+a.FirstName+' / '+a.id) as "Client",
c.ProvCode as "Rendering_Provider",
d.CPTCode,
d.srvcode,
COUNT(*) as "NumSessions"
FROM
patients a
JOIN jcharges b on a.ptnum = b.ptnum
JOIN providers c on b.providernum=c.providernum
JOIN services d on b.servicenum=d.servicenum
WHERE
a.intakedate > '2003-08-31'
AND a.flag = 0
// AND d.srvcode IN ('IP','GRP')
// remove the beginning slashes on line above to return only those service codes
GROUP BY client,rendering_provider,cptcode,srvcode
ORDER BY client,rendering_provider,cptcode,srvcode
;OUTPUT TO c:\sos\sessioncount.wks FORMAT LOTUS
```

Referring Physicians for Payors with ID's and Patient Count

Title	Referring Physicians for Payors with ID's and Patient Count
Date	08/03/2005
Author	Seth Krieger
Product	OMWin
Keywords	Referral Payor ID COUNT GROUP DISTINCT

I'm writing a query and want to print out medicare and bcbs numbers in addition to default id for referral sources. I can get the defaultid from refsrcs but the other numbers come up as null. I upgraded to the latest version of OM and it now looks like there is a different system for entering specific insurance numbers for referral sources so I was wondering if this is a factor? Also is there a way to get a count for the number of patients a particular referring physician has sent our way?

OK, looks like you just want referral sources associated with active patients based on intake date. Couple of things you need to consider...

Firstly, the referring physicians used on claims may not match the referral source stored in Patients. The latter could be "yellow pages", former patients, or whatever, while the former should be only physicians or other healthcare providers. Given that you are interested in the ID's, it seems that you are probably more interested in the referring physician that prints on the claim forms. That is in the claim setup table (PtCSU) not in Patients. To get to those, you would have to add a JOIN to PtCSU. In addition you will have to (left outer) join refpayorids and payors:

```
SELECT
trim(refname + ' ' + b.firstname) as "RefName",
b.address1,
b.address2,
b.city,
b.zip,
b.phone,
b.Defaultid,
e.payorname,
d.providerid,
d.secondaryid,
COUNT(distinct a.ptnum) AS "PtCount"
FROM patients A
JOIN ptcsu c ON c.ptnum = a.ptnum
JOIN refsrcs b ON b.refsrcnum = c.refsrcnum
LEFT OUTER JOIN refpayorids d ON b.refsrcnum = d.refsrcnum
LEFT OUTER JOIN payors e ON e.payorname = d.payorname
WHERE
a.flag=0
AND a.intakedate > '1980-10-01'
AND a.licnum='101'
AND payorname IN (<list payor numbers here>)
GROUP BY refname,b.address1,
b.address2,b.city,b.zip,b.phone,b.defaultid,e.payorname,d.providerid,d.secondaryid
ORDER BY refname
;OUTPUT TO c:\sos\refinfo.wks FORMAT LOTUS
```

Referral Source, Patient Count and Charges By Month

Title	Referral Source, Patient Count and Charges By Month
Date	09/22/2005
Author	Seth Krieger
Product	OM
Keywords	Charges Referral Count

The unique thing about this query is that it creates a grid with the months across the top. For each Referral Source/Year there is a single row, like so:

Results											
	Ref'd By	Year	Jan Count	Jan Chgs	Feb Count	Feb Chgs	Mar Count	Mar Chgs	Apr Count	Apr Chgs	May C
1	ADRIAN	2001	3	701.81	2	255.87	2	170.87	1	0.87	
2	ADRIAN	2002	2	230.87	1	220.00	0	0.00	0	0.00	
3	ALLEN	2001	1	255.00	1	100.00	1	285.00	1	200.00	
4	ALONSO	2001	1	257.93	1	172.87	1	288.75	1	4.24	
5	ANGLIN	2001	1	0.38	1	50.38	1	9.13	1	1.25	
6	ANGLIN	2002	1	1.25	0	0.00	0	0.00	0	0.00	
7	ARAUJO	2001	0	0.00	0	0.00	0	0.00	0	0.00	
8	ARAUJO	2002	2	210.33	0	0.00	0	0.00	0	0.00	
9	ASADI	2001	0	0.00	0	0.00	0	0.00	0	0.00	
10	BAUDINO	2001	6	1035.00	5	852.68	4	643.36	3	503.16	
11	BAUDINO	2002	8	1737.14	7	1330.00	0	0.00	0	0.00	

```

SELECT b.refname as "Ref'd By", YEAR(c.trandate) as "Year",
COUNT (DISTINCT (IF MONTH(c.trandate) = 1 THEN c.ptnum ENDIF)) as "Jan Count",
SUM(IF MONTH(c.trandate) = 1 THEN c.amount ELSE 0 ENDIF) as "Jan Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 2 THEN c.ptnum ENDIF)) as "Feb Count",
SUM(IF MONTH(c.trandate) = 2 THEN c.amount ELSE 0 ENDIF) as "Feb Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 3 THEN c.ptnum ENDIF)) as "Mar Count",
SUM(IF MONTH(c.trandate) = 3 THEN c.amount ELSE 0 ENDIF) as "Mar Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 4 THEN c.ptnum ENDIF)) as "Apr Count",
SUM(IF MONTH(c.trandate) = 4 THEN c.amount ELSE 0 ENDIF) as "Apr Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 5 THEN c.ptnum ENDIF)) as "May Count",
SUM(IF MONTH(c.trandate) = 5 THEN c.amount ELSE 0 ENDIF) as "May Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 6 THEN c.ptnum ENDIF)) as "Jun Count",
SUM(IF MONTH(c.trandate) = 6 THEN c.amount ELSE 0 ENDIF) as "Jun Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 7 THEN c.ptnum ENDIF)) as "Jul Count",
SUM(IF MONTH(c.trandate) = 7 THEN c.amount ELSE 0 ENDIF) as "Jul Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 8 THEN c.ptnum ENDIF)) as "Aug Count",
SUM(IF MONTH(c.trandate) = 8 THEN c.amount ELSE 0 ENDIF) as "Aug Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 9 THEN c.ptnum ENDIF)) as "Sep Count",
SUM(IF MONTH(c.trandate) = 9 THEN c.amount ELSE 0 ENDIF) as "Sep Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 10 THEN c.ptnum ENDIF)) as "Oct Count",
SUM(IF MONTH(c.trandate) = 10 THEN c.amount ELSE 0 ENDIF) as "Oct Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 11 THEN c.ptnum ENDIF)) as "Nov Count",
SUM(IF MONTH(c.trandate) = 11 THEN c.amount ELSE 0 ENDIF) as "Nov Chgs",
COUNT (DISTINCT (IF MONTH(c.trandate) = 12 THEN c.ptnum ENDIF)) as "Dec Count",
SUM(IF MONTH(c.trandate) = 12 THEN c.amount ELSE 0 ENDIF) as "Dec Chgs",
COUNT(DISTINCT a.ptnum) as "# Count", SUM(c.amount) as "TotChgs"
FROM patients a
JOIN refs b on a.refsrcnum = b.refsrcnum
JOIN journal c on a.ptnum = c.ptnum

```

```
WHERE c.trantype = 'S' AND c.trandate > '2000-12-31'  
GROUP BY b.refname, YEAR(c.trandate)  
ORDER BY b.refname, YEAR(c.trandate)
```

Length of Stay by Provider

Title	Length of Stay by Provider
Date	10/17/2005
Author	Seth Krieger
Product	OM/CM
Keywords	Intake Discharge Length of Stay

Let's take a look at this request from a detailed and summary angle.

```
SELECT
Lastname,firstname,id,
COALESCE(provcode,'None') AS "Provider",
Intakedate,dischargedate,
DateDiff(day,intakedate,dischargedate) AS "Days"
FROM patients a
JOIN providers b ON a.providernum=b.providernum
WHERE
intakedate IS NOT NULL
and discharge date IS NOT NULL
```

....will give you a list of patients, their intake and discharge dates, and the number of days between those dates.

Extending that to give us a summary, with provider, average length of stay, and the number of patients from which the statistics are derived:

```
SELECT
COALESCE(provcode,'None') AS "Provider",
COUNT(*) AS "N",
AVG(DateDiff(day,intakedate,dischargedate)) AS "Days"
FROM patients a
JOIN providers b ON a.providernum=b.providernum
WHERE
intakedate IS NOT NULL
and dischargedate IS NOT NULL
GROUP BY "Provider"
ORDER BY "Provider"
```

Admissions by Zipcode

Title	Admissions by Zipcode
Date	10/17/2005
Author	Seth Krieger
Product	OM/CM
Keywords	Intake Discharge Length of Stay Admissions

Start with the basic count of intakes between two dates, grouped by zip code:

```
SELECT
zip,
COUNT(*) AS "N"
FROM rv_patients
WHERE
intakedate BETWEEN '2004-01-01' AND '2004-12-31'
GROUP BY zip
ORDER BY zip
```

Next, let's make it a little more interesting by breaking the results down further by year and month:

```
SELECT
zip,
YEAR(intakedate) AS "Yr",
MONTH(intakedate) AS "Mon",
COUNT(*) AS "N"
FROM rv_patients
WHERE
intakedate BETWEEN '1990-01-01' AND '2005-12-31'
GROUP BY zip, "Yr", "Mon"
ORDER BY zip, "Yr", "Mon"
```

You could, of course, change the order of the columns in the SELECT clause and in the ORDER BY line to reorganize the output.

Active Patient Count For Date Range

Title	Active Patient Count For Date Range
Date	10/17/2005
Author	Seth Krieger
Product	OM/CM
Keywords	Intake Discharge Length of Stay Admissions

This simple query tells you how many patients have charge entries in their ledgers between two dates. The keyword DISTINCT in the COUNT() function eliminates duplicates.

```
SELECT
COUNT(DISTINCT ptnum)
FROM
journal
WHERE
trantype = 'S'
AND amount > 0
AND trandate BETWEEN '2004-01-01' AND '2004-12-31'
```

Here's a variation using the rv_charges view that breaks down the patient count by rendering provider. In this case, we use rv_charges as an easy way to get to the provider code associated with the charges:

```
SELECT
provcode AS "Provider", COUNT(DISTINCT ptnum) AS "Clients Seen"
FROM
rv_charges
WHERE
amount > 0
AND trandate BETWEEN '2004-01-01' AND '2004-12-31'
GROUP BY
provcode
```

Patient Count for Period by SortCode

Title	Patient Count for Period by SortCode
Date	10/18/2005
Author	Seth Krieger
Product	OM/CM
Keywords	Intake Discharge Length of Stay Admissions SortCode Sort

This one is a variation on other, earlier queries. The difference here is the introduction of SortCode. SortCode is a value specified when entering charges and credits. In order to associate patients with particular SortCode values, we must inspect the transactions linked to the patient and to the SortCode.

If you were to inspect the values in the sortcode field of the **journal** table, you would see only numbers that do not reflect your selections in the SortCode field in the transaction entry windows. The number is a link to a row in the **lookups** table that contains the SortCode shorthand code and description. You must, therefore, include a JOIN from journal.sortcode to lookups.lunum, which is the analogous value in the lookups table.

The aggregate function structure COUNT(DISTINCT PTNUM) forces the query to count only unique patients, eliminating duplicates. If you were to omit the keyword DISTINCT, you would end up with a count of the number of transactions instead of the number of unique patients. Note that the query returns only charge entries (a.trantype = 'S') because we want to count only patients who have been seen during the period. We don't want to count payments or adjustments (which share a trantype of 'P').

This example also uses a standard function to represent the current date, TODAY(*). An alternate syntax would simply be the words CURRENT DATE. The result would be identical. You could, of course, also substitute a specific date.

One more thing: note that if there are transactions for the same patient but that have different SortCodes, the patient will appear in the count for each of the SortCodes linked to his or her charge entries.

Here is the query:

```
SELECT
lucode AS "SortCode", COUNT(DISTINCT PTNUM) AS "ClientCount"
FROM
journal a
JOIN lookups b ON a.sortcode = b.lunum
WHERE
a.trantype = 'S'
AND a.trandate BETWEEN '2005-10-01' AND TODAY(*)
GROUP BY
"SortCode"
ORDER BY
"SortCode"
```

Export Appointment Data for Automated Reminder System

Title	Export Appointment Data for Automated Reminder System
Date	03/21/06
Author	Seth Krieger
Product	SCHED
Keywords	Appointment Reminder Export Televox

The following query was done for a user who wanted to automatically export names, phone numbers, and appointment dates and times for uploading to Televox (www.televox.com) for automated appointment reminder calls.

// In this query, UserDefined field one on the patient form is checked for a "N". This field
// should be set up as "OK for auto appt reminders". If you enter an "N" in that field
// the patient will not be included in the calls.

```
SELECT
  a.firstname, a.lastname,
  (replace(a.phone1,'-',')) AS phnumber,
  a.apptdate,
  SUBSTR(CAST(a.apptstarttime AS CHAR),1,5) AS ApptTime
FROM
  rv_appts a
  JOIN patients c ON a.ptnum = c.ptnum
  LEFT OUTER JOIN uddatapt d on c.udatanum = d.udatanum
WHERE
  // exclude cancellations
  cancelflag = 0
  // only if phone number is at least 7 digits
  AND length(trim(phone1)) > 7
  //appts for tomorrow or next day
  AND (apptdate = dateadd(day,1,current date) OR apptdate = dateadd(day,2,current date))
  // UD field is ok for phone reminders
  AND (d.fld1 <> 'N' or d.fld1 is null)
;
// set file for output
OUTPUT TO \sos\phone-reminder.txt FORMAT ASCII QUOTE "
```

List Patients with No Primary Diagnosis by Primary Provider

Title	List Patients with No Primary Diagnosis by Primary Provider
Date	03/24/06
Author	Seth Krieger
Product	OMWin
Keywords	Diagnosis Dx Patient List

This is a relatively easy query that can be used to generate a list of patients for which no diagnosis has been entered. Technically, the selection is for a missing Dx1, so if for some reason there is no Dx1, but there is a Dx entered in Dx2, 3, or 4 the name will still appear on the list.

Note that we use LEFT OUTER JOINS between the Patients table and the Providers and PtCSU tables so that if there is no primary provider entered, or the default claim setup is missing for some reason, we will still see the patient name in the result set. Also notice the OR condition in the WHERE clause. Here we are looking for a missing Dx link (null) or a link number of zero (which is what it should be if no Dx has been selected). It is ALWAYS a good idea to put your OR expressions within parentheses so that the query parser does not make it's own decision about this matter. Here it would not really matter because there is no ambiguity, but it is still a good habit and one that will save you much confusion when your conditions are more complex.

```
SELECT
  Providers.ProvCode,Providers.ProvLName, Providers.ProvFName,
  Patients.LastName, Patients.FirstName, Patients.ID
FROM
  Patients
  LEFT OUTER JOIN Providers ON Patients.ProviderNum = Providers.ProviderNum
  LEFT OUTER JOIN PtCSU ON Patients.PtNum = PtCSU.PtNum
WHERE
  Patients.LicNum = 101 AND
  Patients.Flag = 0 AND
  PtCSU.TypeFlag = 'D' AND
  (PtCSU.Dx1 IS NULL OR PtCSU.Dx1 = 0)
ORDER BY
  Providers.ProvCode,Patients.LastName, Patients.FirstName,Patients.ID
```

Expiring Authorizations by Primary Provider

Title	Expiring Authorizations by Primary Provider
Date	09/19/06
Author	Seth Krieger
Product	OMWin
Keywords	Auth Authorization Manage Care Expiring Expiration

I don't know about other people but, I could really use query to provide a simple table of PA's that are nearly exhausted. The MC auth report is just too big.

Criteria is simply PAs < given_number and Exp Date between given_dates

Output like:

Provider | Patient | PAs Remaining | Exp.Date

sorted by provider, patient

I added a couple of refinements, including a column for the insurer's name, and both the primary provider and authorized provider. You can change the ORDER BY to reflect the one you want. I also tuned up the conditions in the WHERE clause to eliminate inactive/discharged patients and inactive authorizations, and included the expiration date in the selection conditions (third to last line). The "TODAY() + 14" in this example means that auths with expiration dates within the next 14 days will be selected. Obviously, the "3" in the same line selects auths with less than 3 visits remaining.

```
SELECT
  f.provcode AS "PrimaryProvider",
  g.provcode AS "AuthorizedProvider",
  (e.lastname + ', ' + e.firstname + ' / ' + e.id) AS "Patient",
  d.payorname AS "Insurer",
  (a.maxvisits - a.usedvisits) AS "VisitsLeft",
  a.enddate AS "ExpDate"
FROM
  ptauths a
  JOIN ptpolicies b ON a.ptpolnum = b.ptpolnum
  JOIN ptpayors c ON b.ptpayornum = c.ptpayornum
  JOIN payors d ON c.payornum = d.payornum
  JOIN patients e ON a.ptnum = e.ptnum
  LEFT OUTER JOIN providers f ON e.providernum = f.providernum
  LEFT OUTER JOIN providers g ON a.providernum = g.providernum
WHERE
  a.status = 'A' //active auths only
  AND e.flag = 0 //active patients only
  AND e.dischargedate is null // no discharge date entered
  AND ("VisitsLeft" < 3 OR "ExpDate" < TODAY() + 14)
ORDER BY
  "PrimaryProvider", "Patient"
```