Introduction to Database Searching

WHY DO WE DO DATABASE SEARCHES?

ORIGINAL PRESENTATION BY DR. DAVID PAGE, 2012

UPDATED BY ADRIAN STARZYNSKI, 2023

Target Audience

1. Those of you that want to learn to write your own searches.

2. Those of you that simply want to learn to use searches that exist and make simple modifications.

Basic Background Knowledge

- Most of the information in an EMR is stored in a database in what are called “Tables”
- This is to allow easy retrieval and use of the information
- OSCAR uses MariaDB (MySQL) as its opensource database
- **Essentially, an EMR is just an interface to the database tables**

Agenda for tonight

1) The Basics (a bit of theory)
2) Do it Yourself (some hands on)
3) Report by Templates (the cadillac of searches)

First the theory……

What is a table?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

What is a table?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

Columns

What is a table?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

Rows

What is a table?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

Fields

MMSE

- I want you to remember these three words, and I will later ask you to repeat them to me…..

  SELECT

  FROM

  WHERE

  and a few others……

SQL SYNTAX

SELECT {column name}
FROM {table name}
WHERE {= <> > < >= <=}
AND {both are true}
OR {one or other or both}
LIMIT 20

Other syntax:

BETWEEN , NOT BETWEEN, LIKE, NOT LIKE, IN, NOT IN, ORDER BY, GROUP BY, DISTINCT

Want to know who the doctors are?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

```
SELECT   doctor_name    FROM   doctors
```

Want to know who the doctors are?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

```
SELECT   doctor_name     FROM    doctors
```

Want to see a row?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

```
SELECT   doctor#,  doctor_name ,  phone_no,  hair_color
FROM    doctors
WHERE doctor# = 103

OR

SELECT *  FROM    doctors  WHERE    doctor# = 103
```

Want to see a row?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

```
SELECT   doctor#, doctor_name , phone_no, hair_color
 FROM    doctors
WHERE doctor# = 103
OR

SELECT *  FROM    doctors  WHERE    doctor# = 103
```

Want to know who the doctors are with brown hair?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

```
SELECT   doctor_name
FROM    doctors
WHERE  hair_color  =  brown
```

Want to know who the doctors are with brown hair?

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

```
SELECT   doctor_name
FROM    doctors
WHERE  hair_color  =  brown
```

So far so good?

- Often information is stored in more than one table with a “key” that connects the two tables
- This is to save duplication of information in the different tables

Example of two tables

residents

resident#	resident_name	hair_color	doctor#	address	postal_code	phone_no
345	Mike	brown	103	Courbould Ave	v2r 2r3	6048245634
456	Cathy	red	244	Mary St	v2r 4t1	6048247933
553	Jake	blond	103	Edwards St	v2r 5w7	6048248332
521	Mary	brown	167	Courbould Ave	v2r 2r3	6048245634

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

Another way of looking at it

doctors

doctor#
doctor_name
phone_no
hair_color

residents

resident#
resident_name
hair_color
doctor#
address
postal_code
phone_no

Want to know which residents are working with brown haired doctors?

doctors

doctor#
doctor_name
phone_no
hair_color

residents

resident#
resident_name
hair_color
doctor#
address
postal_code
phone_no

```
SELECT resident_name
FROM  residents,   doctors
WHERE  hair_color = brown
```

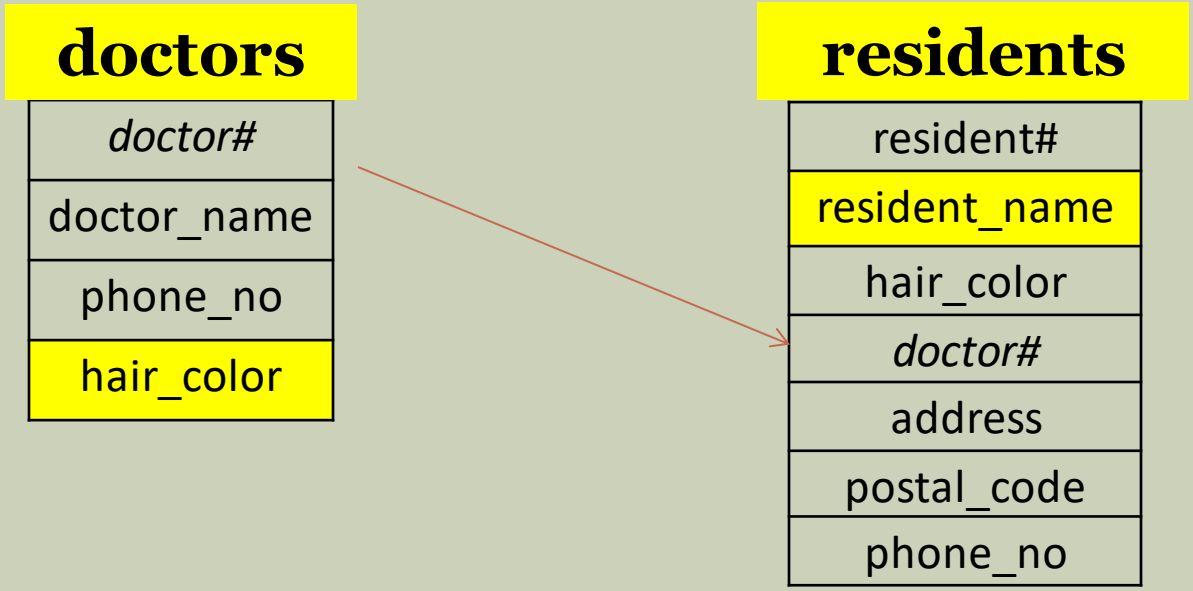
Why won't this work?

- You need to LINK the tables
- And you need to give each column a

UNIQUE name

Otherwise, the computer will produce and infinite number of permutations and combinations……

Want to know which residents are working with brown haired doctors?



SELECT residents.resident_name
FROM residents, doctors
WHERE doctors.hair_color = brown
AND resident.doctor# = doctors.doctor#

residents

resident#	resident_name	hair_color	doctor#	address	postal_code	phone_no
345	Mike	brown	103	Courbould	v2r 2r3	6048245634
456	Cathy	red	244	Courbould	v2r 2r3	6048245634
553	Jake	blond	103	Courbould	v2r 2r3	6048245634
521	Mary	brown	167	Courbould	v2r 2r3	6048245634

doctors

doctor#	doctor_name	phone_no	hair_color
103	Dr Smith	6048585756	brown
244	Dr Ross	6048586778	grey
167	Dr Voth	6048587523	brown
177	Dr Viljoen	6048583458	blond

```
SELECT residents.resident_name    FROM
residents,      doctors
WHERE  doctors.hair_color  =  brown
AND resident.doctor#  =  doctors.doctor#
```

Aliases- a convenient abbreviation

```
SELECT r.resident_name
FROM  residents r,     doctors d
WHERE  d.hair_color  =  brown
AND  r.doctor#  =   d.doctor#
```

```
SELECT residents.resident_name
FROM  residents,     doctors
WHERE  doctors.hair_color  =  brown
AND resident.doctor#  =  doctors.doctor#
```

Enough theory, lets do some hands on

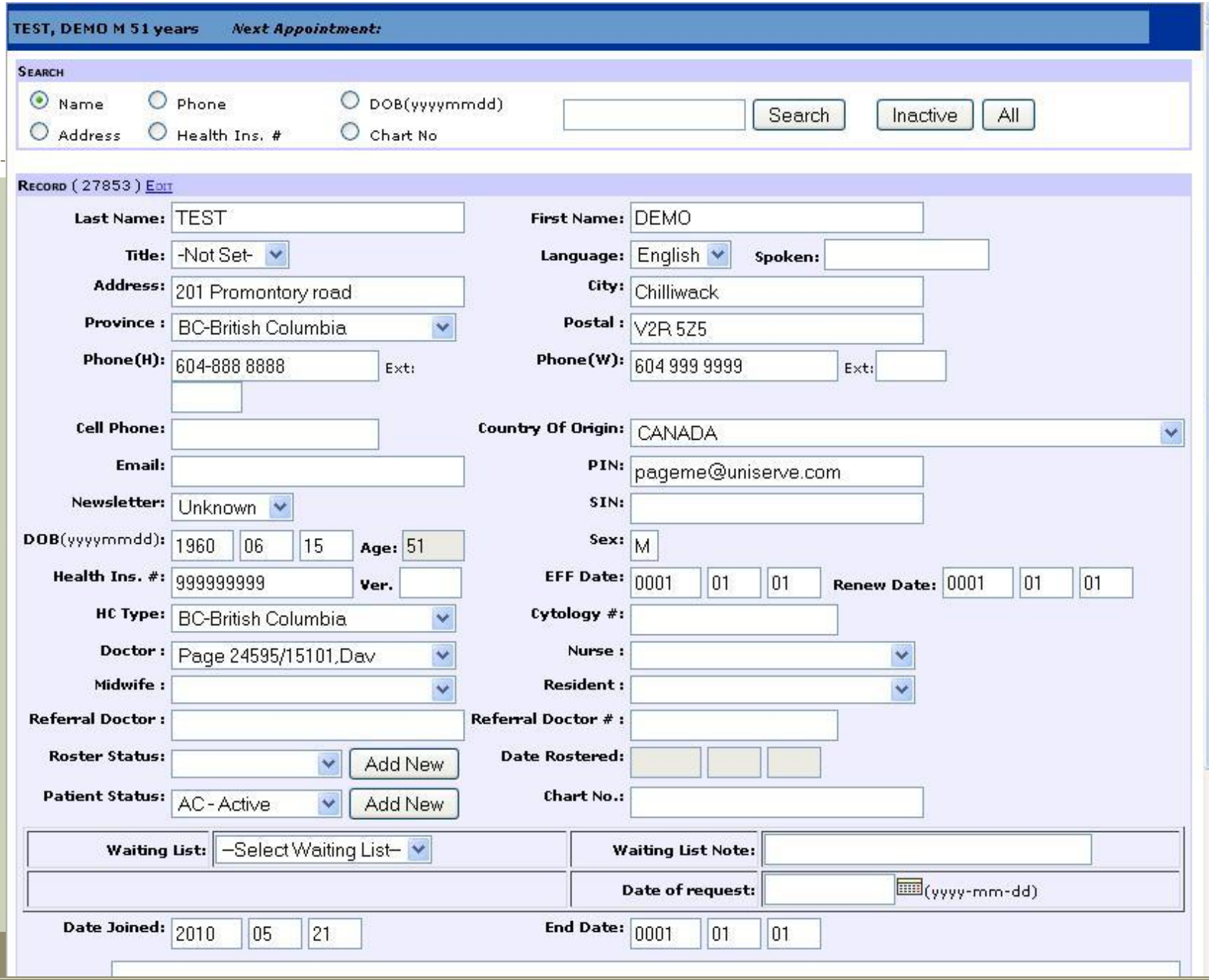
- From appointment screen in OSCAR
- Administration
- Report
- Query by Example

show tables;

This will display all the tables in OSCAR

Commonly used tables in OSCAR

- demographic
- eChart
- dxresearch
- drugs
- measurements
- appointment
- billing
- billingmaster
- provider
- preventions


TEST, DEMO M 51 years
Next Appointment:
SEARCH
Name
Phone
DOB(yyyymmdd)
Address
Health Ins. #
Chart No
Search
Inactive
All
RECORD (27853) EDIT
Last Name: TEST
First Name: DEMO
Title: -Not Set-
Language: English
Spoken:
Address: 201 Promontory road
City: Chilliwack
Province : BC-British Columbia
Postal : V2R 5Z5
Phone(H): 604-888 8888
Ext:
Phone(W): 604 999 9999
Ext:
Cell Phone:
Country Of Origin: CANADA
Email:
PIN: pageme@uniserve.com
Newsletter: Unknown
SIN:
DOB(yyyymmdd): 1960 06 15
Age: 51
Sex: M
Health Ins. #: 999999999
Ver.
EFF Date: 0001 01 01
Renew Date: 0001 01 01
HC Type: BC-British Columbia
Cytology #:
Doctor : Page 24595/15101,Dav
Nurse :
Midwife :
Resident :
Referral Doctor :
Referral Doctor # :
Roster Status:
Add New
Date Rostered:
Patient Status: AC - Active
Add New
Chart No.:
Waiting List: –Select Waiting List–
Waiting List Note:
Date of request:
(yyyy-mm-dd)
Date Joined: 2010 05 21
End Date: 0001 01 01

describe {table};

This will list the columns in that particular table
(e.g. demographic, look at a patients demographics page first)

describe demographic;

Lets ask some questions

- First let us see what is in the demographic table

(refer to the demographic table property handout)

```
select *
from demographic
limit 20
```

(* = select all)

```
select first_name, last_name
from demographic
limit 20
```

Let us find the patients older than 100!

```
select first_name, last_name
from demographic
where year_of_birth < 1911
limit 200;
```

Let us filter out the 0000-00-00

```
select first_name, last_name
from demographic
where year_of_birth < 1911
and year_of_birth <> 0000
limit 200;
```

Lets only look at the active patients

```
select first_name, last_name
from demographic
where year_of_birth < 1911
and year_of_birth <> 0000
and patient_status = 'AC'
limit 200;
```

Doing Arithmetic with selected Information

SYNTAX

* / - +

MAX MIN

SUM

AVG

COUNT

Want to find your oldest patient?

```
select min(year_of_birth)
from demographic
where year_of_birth <>0000
and patient_status = 'AC';
```

Then

```
select first_name, last_name
from demographic
where year_of_birth = 1904
and patient_status = 'AC';
```

Some more interesting searches…

1) What is the average year of birth of our patients?

```
select avg(year_of_birth) from demographic where patient_status = 'AC';
```

2) What is the sum of our patients' year of births?

```
select sum(year_of_birth) from demographic where patient_status = 'AC';
```

3) How many patients are listed as active in our server?

```
select count(demographic_no) from demographic where patient_status = 'AC';
```

Now lets try using two tables…

- Let us list all our patients that have been entered into the Disease Registry with CHF (ICD 428)

(refer to the dxresearch table properties)

```
select demographic_no
 from dxresearch
where dxresearch_code = 428;
```

This works, but we want names….

```
select demo.first_name,  demo.last_name
from dxresearch dx,  demographic demo
where  dx.dxresearch_code  =  428
and  dx.demographic_no  =  demo.demographic_no;
```

Now to the Cadillac of searches, "Report by Templates"

This is a Query by Example engine with two differences:

1. It allows easy export of the results to a spreadsheet like Excel
2. It allows "variable inputs"

Basic structure of a Report by template

<report title="**Title**" description="**Description of what the report does**" active="1">

<query>

Place query here

</query>

<param id="**name**" type="**(text)(list)(date)**" description="**Description**">
</param>

</report>

This is the type of input

Param Types Examples

EXAMPLES OF HOW TO USE THESE 3 PARAM FIELD TYPES IN YOUR RBT (AFTER </QUERY>)

TEXT

LIST

DATE

LIST

```
AND
     be.code_date >= DATE_SUB(CURDATE(),{intervalday} ) AND
     be.code_date <= CURDATE()
    ORDER BY
     be.code_date DESC;


...

<param id="intervalday" type="list" description="Number of days:">
    <choice id="interval 30 day">2</choice>
    <choice id="interval 60 day">10</choice>
    <choice id="interval 90 day">30</choice>
    <choice id="interval 120 day">100</choice>
    <choice id="interval 1000000 day">ALL</choice>
    </param>
```

TEXT

```
<param id="query"  type="text"  description="Query">
</param>
```

DATE

<param id="**dateFrom**" type=**"date"** description="**dateFrom**">
</param>

The date format is YYYY-MM-DD

So this is how the Disease Registry search would look:

<report title="**Disease Registry lookup**" description="**Search for patients in the Disease Registry by ICD 9 code**" active="1">

<query>

Paste in Query

select demo.first_name, demo.last_name
from dxresearch d,demographic demo
where d.dxresearch_code = 428
and d.demographic_no = demo.demographic_no;

</query>

<param id="**searchtext**" type="**text**" description="**ICD 9 code**"> </param>

</report>

So this is how the Disease Registry search would look:

```
<report title="Disease Registry lookup" description="Search for patients in the Disease Registry by ICD 9 code" active="1">

<query>

select demo.first_name "First Name", demo.last_name "Last Name"
from dxresearch d,demographic demo
where d.dxresearch_code = {ICD9}
and d.demographic_no = demo.demographic_no;

</query>

<param id="ICD9" type="text" description="ICD 9 code"> </param>

</report>
```

Variable text input id-can be any name you choose

So how do you load Report by Templates??

Method 1: Standard upload

Report by Template is stored as a text (.txt) file.

- Administration/Reports/Report by Template
  - Add template/Browse for file/Upload and Add

Method 2: Overwriting existing template

What I do most of the time is upload a bunch of “Blank” Report by templates, and then I will copy the text file and paste over the “Blank”.

Why do this?

1. Because otherwise you have no way of ordering your Reports (they come in sequential)
2. This way you can copy an existing Report and duplicate it so that you can make modifications to it

Wrap up

Hopefully, you now know:

- How to list the Tables
  (show tables;)
- How to see the columns in a particular Table
  (describe {table name};)
- How to retrieve data from a Table
  (Select {**column name**} From {**table name**} Where {**filter**};)
- How to insert the Query into a Report by Template
- How to upload and edit “Reports by Template”

The End