

Clinical Track Generator

Change List:

04/15/2024	PAB	New
04/18/2024	PAB	Explain Queries

Overall Intent

This module will run inside of /THEO and /TABLET applications. It reads the data from the currently focused template that was filled out by the patient, and generates a list of suggested “Clinical Tracks.” No data is stored – only viewed or printed.

Setup

Before running this code you need to get the **Patient_ID** and the **Encounter_ID** from EncounterHistory. Since you will run this module from the template module, you will have access to those two variables within the PHP calling code. Encounter_ID is a primary key and comes from EncounterHistory and Patient_ID is a foreign key from EncounterHistory.

Code

This is the SQL code that will generate the output for the Clinical Tracks Report.

```
/*      Clinical Track Generator
      Code to calculate Clinical Tracks
      UPDATE:      04/02/2024      PAB      New
                  04/05/2024      PAB      Add Intersect Function
                  04/06/2024      PAB      Add SortOrder, Fix small bug

*/
```

The code in PHP that calls this SQL code needs to initialize two variables: Encounter_ID and Patient_ID. You will have those in the PHP code somewhere and I’m not sure where they are.

```
/* Calling code needs to initialize these two variables before calling this SQL Code */
SELECT @Encounter_ID      = <get current Encounter_ID>
SELECT @Patient_ID        = <get current Patient_ID>
*/
```

This next section sets some SQL environment variables and creates a temporary table in SQL called tempTBots. It also adds an index to the table.

tempTBots will hold all of the TBot values for this particular template's data – unique to this patient's encounter. TBot values are key identifiers of a type of problem. As an example: The patient smokes cigarettes would be a TBot value of 128 (as an example). The patient has kidney disease would be a TBot of 433 (as an example).

```
USE Wellness_eCastEMR_Data
GO
DROP Table Wellness_eCastEMR_Data.dbo.tempTBots
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE Wellness_eCastEMR_Data.[dbo].[tempTBots](
    [tempTBots_ID] [int] IDENTITY(1,1) NOT NULL,
    [TBot] varchar(10) )
GO
CREATE INDEX tempTBots_idx
    ON Wellness_eCastEMR_Data.dbo.tempTBots(TBot)
GO
```

In the SQL code we are declaring the variables that will be used.

We also set a flag in CTracksHistory (Hidden = 1) that effectively hides any previous rows that had that same Patient_ID and Encounter_ID. Reason: We don't want TWO sets of Clinical Tracks for any patient or encounter combination.

```
SET ANSI_PADDING OFF
GO
DECLARE @Patient_ID INT, @Encounter_ID INT, @n INT, @max INT,
    @CTrackName Varchar(50), @CTrackQty Varchar(50), @CTrackFreq Varchar(50),
    @CTrackMaster_ID INT, @TDate DATE, @SortOrder SMALLINT

UPDATE Wellness_DataArchive.dbo.CTracksHistory
SET Hidden          = 1 WHERE
Patient_ID          = @Patient_ID AND
Encounter_ID        = @Encounter_ID
```

We initialize the TDate variable with today's date.

```
SELECT @TDate      = GetDate() -- You'll use this in CTracksHistory
```

This query puts data in the tempTBots table. It uses a SQL function called "CONCAT" to do this. It has to format the TBots in a very specific way for the next query.

```
INSERT INTO Wellness_eCastEMR_Data.[dbo].[tempTBots]
SELECT CONCAT(TM.TML3_TBotMaster_ID,'-',TM.TML3_TBotData) FROM
Wellness_eCastEMR_Data.dbo.ETL3 ET
JOIN Wellness_eCastEMR_Template.dbo.TML3 TM ON ET.TML3_ID = TM.TML3_ID
JOIN eCastMaster.dbo.TBotMaster TB ON TM.TML3_TBotMaster_ID = TB.TBotMaster_ID
WHERE ET.Encounter_ID      = @Encounter_ID
```

Next it rolls through the Master list of Clinical Tracks (CTracksMaster – there are about 32 of them in the table) and uses the INTERSECT function of SQL to find any matches to the TBots associated with a medical problem in the tempTBots table compared to the set of TBots associated with the Clinical Trigger.

The table called CTracksMasterTBots is a list of specific TBots that match a Clinical Track. Each Clinical Track may have one or more TBots in CTracksMasterTBots. This query finds out if there are ANY matches, and if so INSERTS a row into CTracksHistory.

CTracksHlstory is the table you will display in the report.

--- Starting the loop through CTracksMaster...

```
SELECT @n = 1
SELECT @max = COUNT(*) FROM Wellness_DataArchive.dbo.CTracksMaster
WHILE @n <= @max
BEGIN
```

--- Load up your variables for the INSERT into CTracksHistory

```
SELECT
  @CTrackMaster_ID = CTrackMaster_ID,
  @CTrackName      = CTrackName,
  @CTrackQty       = CTrackQty,
  @CTrackFreq      = CTrackFreq,
  @SortOrder       = SortOrder
  FROM Wellness_DataArchive.dbo.CTracksMaster WHERE CTrackMaster_ID = @n
SELECT TBot FROM Wellness_DataArchive.dbo.CTracksMasterTBots TB WHERE
  TB.CTracksMaster_ID = @n
INTERSECT
SELECT TBot FROM Wellness_eCastEMR_Data.dbo.TempTBots TTB
--- Check to see if the INTERSECT resulted in some rows found
```

```

IF @@ROWCOUNT <> 0
BEGIN
    INSERT INTO Wellness_DataArchive.dbo.CTracksHistory
    (CTrackMaster_ID,Patient_ID,Encounter_ID,CTrackDate,CTrackQty,
    CTrackFreq,SortOrder,Hidden)
    VALUES
    (@CTrackMaster_ID,@Patient_ID,@Encounter_ID,@TDate,@CTrackQty,
    @CTrackFreq,@SortOrder,0)
END
SELECT @n = @n + 1 -- Bump the counter that pushes through CTracksMaster's rows
END

```

The next query is the one you will use to pull the Clinical Tracks out of the table so you can display them. See next page for an example of what Clinical Tracks look like.

---This Query gives you the Clinical Tracks specifically for the encounter/patient you're on

```

SELECT CTM.CTrackMaster_ID,CTM.CTrackName,CTH.CTrackFreq
FROM Wellness_DataArchive.dbo.CTracksHistory CTH
JOIN Wellness_DataArchive.dbo.CTracksMaster CTM
ON CTH.CTrackMaster_ID = CTM.CTrackMaster_ID
WHERE (CTH.Hidden IS NULL or CTH.Hidden = 0)
ORDER BY CTH.SortOrder

```

Output

Make the header of the report EXACTLY like the header of the Patient report with the same formatting, fonts and colors, etc. Add the results of the Query above to the bottom of that report. Repeat headers on multiple pages.

Example:



New River Family
1300 St. Mary's Street
Suite 502
Raleigh, NC 27605P: 919-833-8998
F: 919-772-2727

Account Number:
Policy Number: **12345**

Patient Name: **Raul Abdul**
DOB: **04/23/1966**
Age: **57**
Provider: **Karen A Williams, DO**

DOS: **April 16, 2024**

Suggested Clinical Tracks (For Physician Review)

Clinical Track Component	Frequency
Repeat Wellness Screening	Yearly
Chronic Care Management	While managing the patient's chronic conditions
Remote Physiological Monitoring	Continuously
Remote Therapeutic Monitoring	Continuously
Advance Care Planning Education and Contracts	Monthly
Fall Risk Evaluation/Needs Analysis	Quarterly
Smoking Cessation Counseling	Quarterly
Weight Loss Counseling	Quarterly
Diabetes Counseling	Quarterly
Depression Screening	Monthly
Anxiety Screening	Monthly
PTSD Screening	Monthly
Extended ADL/iADL Determination	PRN
Extended SDoH Determination	PRN
Medication Management	Quarterly
Hypertension Counseling	PRN
Dietary Assistance/Counseling	PRN
Substance Use/Abuse Counseling	Quarterly

Clinical Track Component

Hearing Screening

Vision Screening

Breast Cancer Screening

Prostate Cancer Screening

Pain Screening

Sleep Apnea Counseling

Exercise Counseling

Dental Visit Assistance

Podiatry Visit Assistance

Social Support Assistance

Transportation Assistance

Frequency

Twice per Year

Twice per Year

PRN

PRN

PRN

PRN

PRN

PRN

PRN

Quarterly

PRN

Calling Module

Allow the user to VIEW the report from the following button selections:

The screenshot shows a web interface for generating reports. At the top, there are three input fields with dropdown arrows. Below these fields is a dropdown menu that is open, showing two options: 'Provider Report' and 'Clinical Track'. The 'Clinical Track' option is highlighted in blue. Below the dropdown menu are four buttons: 'Generate Report', 'View Report', 'Save Report', and 'Print Report'. The 'View Report' button has a small downward arrow next to it, indicating it is a dropdown menu.

Allow the user to SAVE the report from the following button selections:

The screenshot shows the same web interface as above. The dropdown menu is open, showing three options: 'Save Provider Report', 'Save Patient Report', and 'Save Clinical Track'. The 'Save Clinical Track' option is highlighted in blue. Below the dropdown menu are four buttons: 'Generate Report', 'View Report', 'Save Report', and 'Print Report'. The 'Save Report' button has a small downward arrow next to it, indicating it is a dropdown menu.

Allow the user to PRINT the report from the following button selections:

The screenshot shows a web interface for report generation. At the top, there are two empty dropdown menus, each with a downward arrow icon. Below these is a horizontal line. To the right of this line, a dropdown menu is open, displaying three options: "Print Provider Report", "Print Patient Report", and "Print Clinical Track". The "Print Clinical Track" option is highlighted with a blue background. Below the horizontal line, there is a row of four buttons: "Generate Report", "View Report" with a downward arrow, "Save Report" with a downward arrow, and "Print Report" with a downward arrow.

WellTrackONE.us/tablet

Incorporate these changes AND the changes you made to the “View Report”, “Save Report” and “Print Report” into the /tablet version of our code (which we call “PreventONE”).