**Example Query**

This query was built to look at the number of Chinook captured in the Lemhi watershed during mark-recapture sampling and for each survey, break the numbers up into Marked/Captured/Recaptured for a population estimate.

The steps below are a guide to build the example query within the ISEMP\_CDB database.

1. Selecting the "Create" tab on the Microsoft Access ribbon at the top of the screen (Figure 1).Graphical user interface, application

   Description automatically generated

Figure 1. Screen grab of the ribbon section of Microsoft Access with steps 1 and 2 indicated.

1. Select “Query Design” to build a custom query (Figure 1). If at any point you lose the ribbon associated with query design, you can reselect it at the top of the screen. See the image below for terminology references. Graphical user interface, application

   Description automatically generated

Figure 2. Screen grab of a Microsoft Access query being built in Query Design with the expression builder button and totals button depicted as well as the query builder workspace and table indicated. Also, the Query tools tab is highlighted at the top of the window.

\*Note- If you accidently click out of the ribbon, you can return to the appropriate tab by clicking the “Query Tools” tab at the top of the ribbon, indicated in Figure 2.

1. The add tables window will open. Select (double click) the DataCollectionEvent, FishCapturePass, and FishObservation tables and add them to the workspace. After you close the table selection window, you may resize the tables you have added to the workspace for easier viewing. You can always click the “Add Tables” button on the ribbon if you forgot a table or accidently removed one. \*Note- You can add query tables as well but not necessary for this query build.
2. The database already has the relationships built between the main tables. Next you will select the data you wish to have in the query table, listed below. This can be done by 1) double clicking the item from the table lists within the workspace (Figure 2) or 2) by manually entering it into the field box of the query builder table (Figure 2). \*Note- when adding fields via double click, be sure to check the query builder table to ensure the selection was properly added.   
     
   For this query, select from the DataCollectionEvent table list: dceType, Watershed, StreamName, SiteName, SurveyDateTime, Season, and GRTSSiteID. From the FishCapturePass table CaptureMethod was selected, and from the FishObservations table Species was selected.  
     
   Explanation of selections-  
   For this query we want the result to only look at mark-recapture events so we will select dceType so that we can filter the type of data collection event. The Watershed data column allows us to filter down to just the Lemhi. We include StreamName so we know where the mark/recapture site occurs. SiteName provides us with a specific location, so we need to include it. We include SurveyDateTime so we have a reference for when exactly sampling occurred at each site. Season is helpful to compare sampling at sites for different times of year-Spring, Summer, Fall, Winter. Lastly from the DCE table we include the GRTSSiteID in case the mark-recapture occurred on a site with a designated ID used from the Generalized Random Tessellation Stratified (GRTS) design master sample list.  
     
   We included CaptureMethod from the FishCapturePass table because each method has its own bias and assumptions, so we want to separate out fish if multiple methods were used in the same sample. Also, in this instance we are going to limit our methods to exclude angling. Lastly, from the FishObservations table we included the field Species so we can filter to only Chinook.  
     
   \*Note-Everything in the steps below that is highlighted in red is exactly what needs to appear in the indicated location. For example, “Like “\*Example\*””. In this instance only what is in red is entered and the exterior quotes are not entered. It is possible to copy and paste into the query builder table but be careful because after you paste, Access may add in extra quotes once you exit the cell. Also, anything contained within asterisks changes the parameters to contains instead of an exact match where only quotes are used. Example “\*Contains\*” vs “Exact”. The asterisks allow for more flexibility in case some data is slightly different, example “Electro Fishing” vs “Electro herding”. To avoid missing data the best option in this example for criteria is “\*Electro\*”.
3. A standard “select” query type will return a row of data for each record in the tables provided. The goal of the query we are building is to create some sub totals for quick population estimates. Now, we will select the “Totals” icon in the top ribbon located next to “Parameters” (Figure 2). Selecting the “Total” button changes the query type and allows us to group similar records and perform basic functions such as: sum, min, max and more.
4. With the query now a totals-query, we will get one row for each species, each different SiteName, StreamName, SurveyDateTime, GRTSSiteID, and Capture Method. The more selections you have as “Group By” the more variables you introduce and potentially create more rows in the results table. What we want to do is create an advanced query to sum Pass 1 Mark, Pass 2 Capture, and Pass 2 Recaptures for use in population estimates and we add those next.  
     
   \*Alternative- If we simply select “FishCount” from the obsRovingFishObservation table, then in the Total row of the query builder table, change it from “Group By” to “Sum” the query would total up the records for each grouping. Resulting in a simple number of fish captured.
5. Now to build our fields to evaluate Marked, Captured, and Recaptured Chinook. This part can be confusing if you’re not familiar with formulas like those used in Microsoft Excel. First the Pass 1 Mark, we want to look for records with a “New Tag” in the “Tag2RecaptureType” and were caught during the “Mark” event in “dceType”. In our query builder table, we click in the top of a new column in the Field row. First, we name the column, to do so type in the label in this case “Pass 1 Marked” and end with a semicolon so we have “Pass 1 Marked:”. Everything before the semicolon is now set as the column header.
6. Now we want to build our expression. It is possible to type the expression directly into the query table, but it is easier if we go to the ribbon at the top of the screen and to the right of the “Add Tables” button there is a smaller “Builder” button (Figure 2). Select this and it will open a window to help build an expression in the field your cursor was in.   
     
   What we want our expression to do is to sum records if those records meet our criteria. For a “Pass 1 Mark” column we want to total fish up if they occur on a Mark event and were given a New Tag. The functions in that statement to build in our expression are: Sum, If, And. We want to SUM, IF, mark AND new tag occur. Access is pretty nice in that it will try to assist you by making suggestions as you type, if you want to use a suggestion you can double click or hit the tab key to input the selection from the helper. Now, to build our expression.   
     
   \*Tip- Sometimes in builder, the access “helper” doesn’t offer any options. First try entering a space to separate from the characters occurring prior to your desired entry point. Or if you type a period “.” and then backspace to remove it, then start typing again it should prompt you with autofill options as you type.
7. The expression for Pass 1 Marked that we want to use is:   
   Pass 1 Marked: Sum(IIf([obs\_RovingFish\_FishObservation].[Tag2RecaptureType]="New Tag" And [HO\_DataCollectionEvent].[dceType]="Mark",[obs\_RovingFish\_FishObservation].[FishCount]))  
     
   Once our expression is done, we can hit “OK” on the right to close the builder window.  
     
   Explanation of expression-  
   What we are doing is totaling up the fish count data value from our fish records if those records have a “New Tag” in the Tag2RecaptureType and occurred in a DCE with the dceType as “Mark”. For the expression to work we need to tell Access where to look for those data and the way Access handles this is Table.Field. First the table name then “.” and then the field name to refer to the data column field within the table. In our expression we use the function Sum(). Everything with the parentheses is summed. We only want to sum values when certain criteria are met, so inside the parentheses of the sum function we use an IIF() function. Inside the parentheses of the IIF function we’re saying if the data in the Tag2RecaptureType column of the FishObservation table = “New Tag” and the data in the dceType of the DataCollectionEvent table= “Mark” then when TRUE the value is equal to the FishCount in the FishObservation table. This IIF function statement runs through all the fish records reporting the FishCount for TRUE statements and reporting no value for FALSE statements. The SUM() function will total all the values returned by our IIF function statement.
8. Now we want to finish up by adding columns for the Pass 2 Captured and Pass 2 Recaptures and those expressions are below.  
     
   Pass 2 Captured: Sum(IIf([HO\_DataCollectionEvent].[dceType]="Recapture",[obs\_RovingFish\_FishObservation].[FishCount]))  
     
   Pass 2 Recaptures: Sum(IIf([obs\_RovingFish\_FishObservation].[Tag2RecaptureType]="Non-Efficiency Recapture" And [HO\_DataCollectionEvent].[dceType]="Recapture",[obs\_RovingFish\_FishObservation].[FishCount]))
9. Once the expressions are built for Pass 1 Marked, Pass 2 Captured, Pass 2 Recaptured, we need to make sure they are all changed from “Group By” to “Expression” in the Total row of our query builder table. The total row in the query builder table needs to be changed to “Expression” anytime a you create your own expression to calculate a field.
10. Lastly before we run our query, we want to do some clean up and “filtering”. For this query we’re only interested in looking at the Lemhi watershed so in the query table builder, under the field-Watershed, we can click into the criteria row and enter “Like “\*Lemhi\*””. Tip-by using the \* access looks for a partial match and without them it looks for an exact match.   
      
    Now if we leave the SurveyDateTime field as a “Group By” we essentially create a row for each data collection record on a site, but we want to look at the site as a whole. What we can do is change it from “Group By” to “Min”, that will give us a relative date for the site survey without breaking up the data for each DCE date-time.   
    Additionally, it might be nice to just have the year in its own column. We add a new column in the query table builder and type “Year:”. Next select the builder button on the ribbon. Our final expression will be:   
      
    Year: Year([HO\_DataCollectionEvent].[SurveyDateTime])  
      
    This expression will extract the year from the SurveyDateTime. We can leave this as a “Group By” in the Total Row of our query builder table because the expression is not doing any calculations. Year is a wide ranging variable and shouldn’t create any extra rows when being used as a “Group By” unless a survey spanned multiple years, unlikely.  
      
    The next field to filter is Species, which we will limit to Chinook by going to the Species column already in our query builder table and entering in the Criteria row “Like “\*Chinook\*””. Lastly, we want to make sure the query only looks at DCE’s that will match our Marked-Captured-Recaptured goal.   
      
    Find the dceType column in the query builder table, we need to change it from a “Group By” or it will separate records for each site by the dceType, when we only want to filter. We change it to a “Where”. This limits the scope of the query to the criteria selected but doesn’t include that data in the actual results table. Our criteria for this will be “Like “\*Mark\*” Or Like “\*Recapture\*””. Be sure that the check box in the “Show” row is unchecked or it will cause an error.
11. Now we are ready to run our query. If desired, you can go into the query table and select fields to sort. It will sort left to right across the columns as it encounters them. You can move the columns in the query builder table (by highlighting a column and click-dragging) and the results table will be displayed in the order of the query. It is possible to organize the query builder table so you can sort data in a specific manner, then in the results table you can also rearrange the columns if desired (by highlighting a column and click-dragging).
12. To finish, click run in the top left on the ribbon and the query runs and displays the results table. We now have totals of Marked/Captured/Recaptured Chinook sampled during a Mark or Recapture event in the Lemhi Basin with some site info and capture method.
13. If needed, when under the home tab on the ribbon. You can modify your query design by clicking view in the top left on the ribbon and then selecting “Design View”.

**Other Access Tips/Notes**

1) Query Type

* Totals Query for Depletion Event
  + Build a totals query with fields: dceType, dceName, StreamName, SurveyDateTime, Season, GRTSSiteID, CaptureMethod, Species, and FishCount. You may add additional fields if desired. Under dceType the criteria would be “Like \*Depletion\*””. Change the FishCount Total row from GroupBy to Sum. The resulting query will give you a total fish count for each species in every depletion DCE. Next you add expression columns to total FishCount for each pass of the depletion. Below is an expression that will return a total for fish captured during the 1st pass.
    - Pass 1 Total Count: Sum(IIf([obs\_RovingFish\_FishCapturePass].[PassNumber]=1,[obs\_RovingFish\_FishObservation].[FishCount]))
      * If you modify the above expression by changing the 1. You can add more columns to display all passes. The max number of passes in the ISEMP\_CDB is 7.
* A crosstab query can be nice when looking at data, it will display one data type (variable) across the top of the table in multiple columns-like Species or PassNumber. Then you can have multiple row types- site name, stream name, etc., and you choose a value to tabulate and it fills everything in.
  + This type of query could be used for the depletion query by using the pass number as the column variable which would alleviate the need to build any expressions.

2) Tips

* If you’re only wanting to build a query to look at site/capture info then do not add the FishObservation table to the query workspace, or you will get a row for each fish record displaying you’re queried data, even if you don’t select a data field from the FishObservation table.
* 99.9% of FishObservation records do have a fish count value but not all.
* Not all fish records have lengths/weights, but 0’s are removed.
* Sometimes the final table you want to build via query is too complicated for Access to run in the manner desired. But it might be possible by first creating a sub-query table. Then when selecting tables for a new query you can go under the “queries” tab and select your sub query to use that subset of data.

3) Notes

* A site may be a NonGRTSSite, but it is possible it contains a GRTS within it. Those fish will be appropriately labeled in the GRTS\_Subsite data field in the FishObservation table for a Mark event. But, the fish on the recapture event will not have the GRTS\_Subsite filled in. This can be deduced with fish waypoints and GRTS site waypoints if necessary.
* Most sampling followed the same protocols but there are slight variations between crews and over time. This can make querying across watersheds and basins tricky.