Cerner Millennium Explorer 2-3 Exercises

Most of the exercises are divided into core and challenge steps. Some of the exercises only contain core steps or challenge steps. Generally the core steps will reinforce the key concepts discussed in the class. The challenge steps will reinforce related concepts or provide additional reinforcement of the key concepts. When time is allotted to work on two or more exercises, it is recommended that you complete all the core steps in all the assigned exercises, before working on any of the challenge steps in any of the assigned exercises.

The exercises can be completed using Visual Explorer, Command Line Discern Explorer, Discern Visual Developer, Explorer Menu or a combination of these tools.

The exercises suggest using the following naming convention when saving files: class_*initials*_ex1.*ext*, where *initials* are the first letters of your first, middle, and last names, and *ext* is the file extension. Visual Explorer will automatically add a .VCL extension to the file name when the file is saved. When working in Command Line Discern Explorer or Discern Visual Developer, please use .PRG as the extension when saving files.

Exercise 1 – A core/challenge exercise

Objective: Review joins, prompts, and qualifications

For this exercise you will create an executable program that will select a person's alias and orders for each of their encounters. Then if time permits, add prompts and qualifications to only select people with names that match the values entered by the user at the prompt.

Core Steps

1) Open a new source code file named class_*initials*_ex1.*ext* **Source File:** ____________
2) Select the following information
   a) From the Person table get the full formatted name
   b) From the Encounter table get the encounter id
   c) From the Orders table get the order mnemonic
   d) From the Person Alias table get the alias for a person
3) Create the following Joins
   a) Join the Encounter table to Person table
   b) Join the Orders table to the Encounter table
   c) Join the Person Alias table to the Person table

Challenge Steps

4) Add prompts and additional qualifications
   a) Prompt the user to enter a last name
   b) Prompt the user to enter a first name
   c) Qualify against the Person.Name_Last_Key and Person.Name_First_Key fields.
   d) Add functions to convert what the user enters to upper case and allow for pattern matching on a wild card.

Exercise 2 – A core/challenge exercise

Objective: Practice Writing an Outerjoin Query

For this exercise you will create a query that selects names from the person table, and the person's aliases from the person_alias table. If the person does not have any aliases, we still want to select their name.

Core Steps

1) Open a new source code file named class_*initials*_ex2.*ext* **Source File:** ———————
2) Select the following information
   a) From the person table get the full formatted name
   b) From the person alias table get
      i) The alias
      ii) The display value, and cdf meaning of the person alias type code
3) Add an outerjoin to select people even if they do not have aliases stored on the person alias table

Challenge Steps

4) Add a qualification to only select aliases that are social security numbers.
5) Assume you made a friend at the Health Care Conference and you want to send them your source code. Make sure the qualification you added in step 4 will work on their system.

Exercise 3 – A core/challenge exercise

Objective: Practice using Outerjoin

For this exercise you will create a query that shows the names of people who have orders, the orders, and order comments.

Core Steps

1) Open a new source code file named class_*initials*_ex3.*ext* **Source File:** ———————
2) Select the following information
   a) From the person table get the full formatted name
   b) From the orders table get the order mnemonic
   c) From the order comment table get the long text id number
3) Join the orders table to the person table and the order comment table to the orders table
4) Add qualifications to only select people with person_ids greater than zero and orders with order_ids greater than zero

5) Run the query and verify that you are only selecting the person information when the person has an order and the order has comments.

6) Add an outerjoin to the order comment table to show people with orders even if the orders do not have order comments

Challenge Steps

7) From the long text table get the first 100 characters of the long text field by using the SUBSTRING function. The SUBSTRING function can be found in the Discern Explorer Help file/ Command Reference/ Functions

8) Join the long text table to the order comment table

9) Add an outerjoin to the long text table to show people with orders even if the orders do not have long text on the long text table

Exercise 4 – A core exercise

Objective: Use a nested select with the not exists operator to create an exception query.

In this exercise you will create a query that shows people with encounters where there are no orders for the encounter.

1) Open a new source code file named class_*initials*_ex4.*ext* **Source File:** ____________

2) Select the first 20 characters of the full formatted name, the meaning value and code value of the encounter type class code from the encounter and person tables.

3) Add a nested select to show people with encounters when there are no orders for the encounter

Exercise 5 – A challenge exercise

Objective: Practice using a nested select with the not exists operator to create and exception query.

For this exercise you will create a query that shows the names of people and their street address when they do not have a MRN.

1) Open a new source code file named class_*initials*_ex5.*ext* **Source File:** ____________

2) Select the following information

   a) From the person table get the last name

   b) From the address table get the first 20 characters of the street_addr field

3) Join the address table to the person table

   Note: the Address table contains all addresses, including addresses for people and organizations. To only select addresses that belong to a person, join the Address.Parent_Entity_ID to Person.Person_ID and add an additional qualifier to only select records where the Address.Parent_Entity_Name is equal to "PERSON".

4) Add a qualification to the person table to only select people that do not have an MRN alias on the person alias table.

5) Verify that the people you are selecting do not have an MRN on the person alias table.

Exercise 6 – A core exercise

Objective: Practice using aggregate functions and report writer commands.

For this exercise you will create a report that displays the person's name, sex, and when the record was updated. You will group the records by sex and practice using some of the aggregate functions and report writer commands.

1) Open a new source code file named class_*initials*_ex6.*ext* **Source File:** ____________

2) Select the following information
   a) Select the full formatted name
   b) Select the CDF meaning for the sex code
   c) Select the date and time the record was last updated
   d) Create an expression using the datetimecmp() function to calculate the number of days between the current date and the date the record was updated. The following example shows how to use the DATETIMECMP:

```
D1 = DATETIMECMP(CNVTDATETIME(CURDATE -1,0), O.ORIG_DT_TM )
```

For more information, look in the Discern Explorer Help file under Command Reference / Functions.

   e) Sort the output by the CDF meaning of the sex code
   f) Qualify where the sex code is greater than 0.0

3) Print the following information in the detail section of the report
   a) The first 20 characters of the full formatted name
   b) The number of days between the current date and when the record was updated. You calculated this number in step 2d above
   c) The date the record was updated
   d) If the record was updated over 365 days ago, print "That was over a year ago!"

4) Place a report title, and the date and time the report was created at the top of the report

5) Create a head section for the CDF meaning of the sex code. Print the sex code and the CDF meaning of the sex code in this section

6) Create a foot section for the CDF meaning of the sex code. Place the following items in this section

a) A count of the number of people with the current sex code

b) An average number of days since the records were updated do not use records that have not been updated in calculating the average.

Note: the updt_cnt field is set to zero when the record is created and incremented by one each time the record is updated. Only use records where the updt_cnt is greater than zero to calculate the average.

c) The maximum number of days since a record was updated

d) The minimum number of days since a record was updated

e) Create text string labels to identify the above calculations, i.e.

```
Total number of people with this sex code:  402
Average number of days since the record was updated:  99
The oldest update was made  376 days ago.
The newest update was made    1 days ago.
```

7) Place the following items at the end of the report by copying the items you created above into the foot report section

a) A count of all the records selected for the report

b) The average number of days since the records were updated

c) The maximum number of days since a record was updated

d) The minimum number of days since a record was updated

e) Create text string labels to identify the above calculations

Exercise 7 – A core exercise

Objective: Practice using Explorer Menu to execute programs on demand.

In this exercise you will use Explorer Menu to execute some of the programs created in previous exercises.

Core Steps

1) If you have been working in command line Discern Explorer and have not created at least three programs in any of the previous exercises
   a) choose three of your source code files
   b) edit the files and add the commands needed to create a program
      i) drop program *program_name* go,
      ii) create program *program_name*, ... end go
   c) include the source code file at the Discern Explorer command line
2) Review the source code from three of your previous exercises and
   a) Verify that they have at least one prompt
   b) Are using the SELECT INTO clause
   c) Identify the program name
3) Open the Explorer Menu application.
4) Add a personal folder under the Main Menu folder
5) Add the three programs identified in step 2 to your personal folder
6) Execute the three programs from Explorer Menu and verify that you are getting the correct output.

The following exercises are for HNAM Explorer 3

Exercise 8 – A core exercise

Objective: Practice working with a record structure

In exercise you will use a record structure to temporarily store information from the person table.

Core Steps

1) Open a new source code file named class_*initials*_ex8.*ext* **Source File:** ____________
2) Write a select statement to select person ids and names from the person table. Select 100 records only.
3) Add the following record statement to create a varying length list record structure. You will use this record structure to temporarily store the numeric person information.

```
!report prolog

;define the record structure
record person (
        1 qual [*]
                2 person_id = f8
                2 name = vc )
```

4) In the Head Report section, initialize a variable named CNT to 0.
5) In the Detail section
   a) Increment the variable named cnt
   b) If needed, add positions to the record structure qual list
   c) Store the person_id and name values in the record structure
6) In the Foot Report section
   a) Use the alterlist() function to remove any unused positions in the record structure
   b) Use a for loop to display the values stored in the record structure. Display the person id and name of each person on one line of the report

Exercise 9 – A core exercise

Objective: Practice joining from a record structure to a table

In exercise you will join a record structure to the person table

Core Steps

1) Open a new source code file named class_*initials*_ex9.*ext* **Source File:** ____________

2) Use the following program to create and populate a record structure that contains person identification numbers and street addresses.

```
!report prolog
free record person
record person(
        1 list [*]
                2 person_id = f8
                2 street    = c50 )

select into "NL:"
        a.parent_entity_id,
        a.street_addr
from    address a
where   a.parent_entity_name = "PERSON" and
        a.street_addr > "1"

head report
        cnt = 0

detail
        cnt = cnt +1
        if(mod(cnt,10) = 1)
                stat = alterlist(person->list, cnt+9)
        endif
        person->list[cnt].person_id = a.parent_entity_id
        person->list[cnt].street = substring(1,50,a.street_addr)

foot report
        stat = alterlist(person->list, cnt)
with maxrec = 200
```

3) Identify the area where the record structure is being defined by adding a comment above that section.

   ; Defining record structure

4) Identify the area where space in memory is being allocated by adding a comment above that section

   ; Allocate space in memory

5) Identify the area where the record structure is being loaded by adding a comment above that section.

   ; Loading record structure

6) Add a second select statement using the Build Query that joins the Person record structure to the Person table where the Person_id stored in the record structure is equal to the Person_id field on the Person table.

7) In the detail section of the second select statement, display the
   a) first 50 characters of the full formatted name from the Person table
   b) person id from the record structure (In Visual Explorer, use the CMD button to add field)
   c) street address from the record structure (In Visual Explorer, use the CMD button to add field)

Exercise 10 – A core exercise

Objective: Practice working with a varying list record structure

In this exercise you will select the orders for one person and load the person's name, the order ids and the order_mnemonics into a record structure. In the Foot Report section you will display the values stored in the record structure.

1) Open a new source code file named class_*initials*_ex10.*ext* **Source File:**——————
2) Select the following information
   a) From the person table select the full formatted name and person id
   b) From the orders table select the order mnemonic and the order id
3) Join the orders table to the person table
4) Sort the output by the person identification number, sub-sort by the order identification number
5) Run the query and record a person identification number for a person who has several orders
6) Modify the query to only select the person identified in step 5, and that person's orders
7) Define a record structure that will store
   a) the first 25 characters of the person's full formatted name
   b) the order identification number for each order
   c) the first 25 characters of the order mnemonic for each order

   NOTE: In order to store the order identification numbers and order mnemonic for each order, you will need to create a varying list segment for the orders
8) In the Head Report section, use the alterlist() function, to initialize10 positions in the record structure to store the information for the orders
9) Add a Head P.person_id section. In this section add code to store the first 25 characters of the person's full formatted name in the record structure
10) In the Detail section
    a) If needed, add positions to the orders segment of the record structure
    b) Store the order identification number and the first 25 characters of the order mnemonic in the orders segment of the record structure

11) In the Foot Report section
    a) Remove any unused positions in the orders segment of the record structure
    b) Display the first 25 characters of the person's named stored in the record structure
    c) Display the order identification numbers and the first 25 characters of the order mnemonics stored in the order segment of the record structure

Exercise 11 – A challenge exercise

Objective: Practice working with a multiple varying list record structure

In this exercise you will modify the source code created in exercise 10 to select the orders for a range of person_id's on the person table. Load each person's name and the number of orders for the person into one varying segment of a record structure, load the order id and the order mnemonic into another varying segment of the record structure that is inside the person segment.

1) Copy the source code file from exercise 10 class_*initials*_ex10.*ext* to class_*initials*_ex11.*ext* **Source File:** ____________
2) Comment out the qualification on a single person identification number
3) Comment out the report writer sections and execute the select statement. In Visual Explorer, Run Query
4) Identify a range of person identification numbers
5) Modify the query to only select person identification numbers in the range you identified in step 4
6) Modify the record structure by adding a varying list segment for the person. In this segment add a record item to store the first 25 characters of the person's full formatted name
7) Modify the levels of the varying list orders segment created in steps 7b and 7c of exercise 10.
    a) Place the orders varying list segment at the same level as the person's name
    b) Place the order id and order mnemonic one level below the orders varying list segment
8) In the Head Report section, use the alterlist()function to initialize 10 positions in the varying list segment for the person
9) In the Head P.Person_id section
    a) If needed add positions to the varying list segment for the person
    b) Store the first 25 characters of the person's full formatted name in the person segment of the record structure
    c) Use the alterlist() function to initialize 10 positions in the varying list segment for the orders
10) In the Detail section

a) If needed add positions to the varying list segment for the orders

b) Store the order identification number for the order in the record structure

c) Store the first 25 characters of the order mnemonic in the record structure

11) Add a Foot P.Person_id section and use the alterlist()function to remove any unused positions from the orders segment for this person

12) In the Foot Report section

a) Use the alterlist() function to remove any unused positions from the person segment of the record structure

b) Use two for loops, one inside the other, to display the values stored in the varying segments of the record structure.

Exercise 12 – A core exercise 1467 -AP- Ex12

Objective: Practice creating an indexed custom table and then join to it.

In this exercise you will create an indexed custom table and

Core Steps

1) Open a new source code file named class_*initials*_ex12.*ext* **Source File:**__________
2) Write a Select statement to only select people with last names that begin with the same letter your last name begins with.
3) Modify the select statement to create a custom table named *initials*custtable with an index on the person_id
4) Execute CCLORATABLE GO and verify that a Discern Explorer table definition has been created for your custom table
5) Use a simple select at the command line to verify that data has been written to your custom table

   select * from *initials*custtable go

Challenge Steps 1467 - AP- Ex12b

6) Write a select that
   a) Selects the order mnemonic from the Orders table
   b) Selects the person name and person id from your custom table
   c) Plan on the Orders table and join to your custom table.

Exercise 13 – A core exercise

Objective: Practice using an Orjoin

For this exercise you will create a query that shows the names of people, their street address, or their aliases.

1) Open a new source code file named class_*initials*_ex13.*ext* **Source File:**__________
2) Select the following information

a) From the person table get the first 20 characters of the last name

b) From the address table get the first 20 characters of the street_addr field

c) From the person alias table get the first 20 characters of the alias

3) Using an orjoin to join either the address table or the person alias table to the person table.

Note: the Address table contains all addresses, including addresses for people and organizations. To only select addresses that belong to a person, join the Address.Parent_Entity_ID to Person.Person_ID and add an additional qualifier to only select records where the Address.Parent_Entity_Name is equal to "PERSON".

4) Add a decode statement that will tell you which side of the orjoin the record came from.

5) If you remove the orjoin would you have more or less records in the output? Why?

Exercise 14 – A challenge exercise

Objective: Practice using Orjoin and the Decode() function

For this exercise you will create a query that shows encounter types, the orders related to each encounter, or the names of personnel associated with the encounter and what type of relationship the personnel had to the encounter.

1) Open a new source code file named class_*initials*_ex14.*ext* **Source File:**________

2) Select the following information

   a) From the encounter table get the encounter identification number and the display value of the encounter type code

   b) From the orders table get the order mnemonic

   c) From the encntr_prsnl_reltn table get the display value of the encntr_prsnl_r_cd

   d) From the prsnl table get the full formatted name

   e) Add a decode statement to determine which side of the orjoin the record came from.

3) Join the orders table to the encounter table or join the encntr_prsnl_reltn table to the encounter table, using an orjoin. Join the prsnl table to the encntr_prsnl_reltn table.

4) Sort the output by the encounter identification number. Sub-sort by the expression created in step 2e above so that all the orders for the encounter are listed before all the personnel related to the encounter.