Auditing ADABAS Protection Logs

with

NatQuery/NatCDC

Contents

Overview

The ability to audit ADABAS transactions is achieved through the use of the NatWorks products NatQuery and NatCDC.

NatQuery is a workstation product designed to be both an End-User Reporting Tool and a Data Warehouse Extraction Tool. As such, once it is configured it generates the processing required to achieve either goal, as well as having the ability to integrate that data into common targets. NatQuery then, is the "front end" which creates processing which is then typically executed on a remote server.

NatCDC is a tool that utilizes the generational features of NatQuery to provide for the creation of processing which achieves Change Data Capture (CDC) processing against ADABAS, or processing that handles the translation of the PLOG files of ADABAS into usable data.

With NatCDC's ability to convert ADABAS transaction log files into usable data, and NatQuery's ability to generate extraction processing, NatWorks enhanced the Query abilities of NatQuery to extend that ability to query transactional records. With this enhancement, ADABAS transactional records can be fully queried for specific types of changes, such as who changed records, what changes were made, when were those changes made, and where the change was initiated from (batch or online, and if online - the terminal).

Audit Output

Whenever an Audit process is run, the resulting Audit Report will contain a set of Summary Reports, and depending upon the Selection Logic that can be optionally applied, the Audit Report may also contain a Detail section. The "key" to whether or not an Audit Detail report is produced is completely dependent on the User applying specific Selection Logic to the Audit Report, as the Detail section will only be displayed for records which match the User-entered Selection Logic.

Examples of a Detail Report and the Summary Reports can be found in Appendix A.

Summary Reports

The Summary Reports are made up of four sections:

- Overall Audit Summary
- Count of Field Changes by Type of Transaction
- Count of Transactions by Hour and Type
- Summary of User Entered Selection Logic by Type of Transaction

Overall Audit Summary

This portion of the Summary Reports provides overall totals from the Audit Processing.

Specifically, this report details the:

- Earliest PLOG Timestamp Found
- Last PLOG Timestamp Found
- Total number of Records Read
- Total number of Records Accepted
- Total number of unique Internal Sequence Numbers (ISNs - unique record identifiers)
- Total Number of Transactions (Store, Update and Delete)
- Total Number of After Images
- Total Number of Before Images
- Total Number of Stores
- Total Number of Updates
- Total Number of Deletes
- The "Audit Options" Used for the run, such as:
    - If an extract of selected records was requested
    - If Store Transactions were ignored
    - If Update Transactions were ignored
    - If Delete Transactions were ignored
- An Indication of whether the data was created by NATCDC or NATCDCSP

Count of Field Changes by Type of Transaction

This portion of the Summary Reports displays, in a tabular form, all fields on the source file, and for each field a count of the number of times that field was affected by a Store, Update or Delete is shown.

For example, if a field called PERSON-ID existed on the file being audited, this section of the report would show the name of the field, and would then additionally display the number of times PERSON-ID was effected by a Store. This means that if there was a value provided for PERSON-ID when a record was Stored, this counter would be incremented. If no value for PERSON-ID was provided when the record was Stored - the corresponding Store counter would not be incremented.

Similarly, if PERSON-ID was effected by an Update, meaning that the value of PERSON-ID was changed as a result of an Update (either from one value to another value, a null value was changed to a non-null value, or a null value was changed to have a value), then the Update Counter for PERSON-ID would be updated; if the value did not change then this counter would not be incremented.

Finally, if a record had a value for PERSON-ID and that record was then Deleted, the Delete Counter for PERSON-ID would be incremented, whereas if the record was Deleted and it had no value for PERSON-ID then the Delete Counter for PERSON-ID would not be incremented.

Count of Transactions By Hour and Type

This portion of the Summary Reports displays a table showing 24 rows, one for each hour in a day, and for each of those hours the number of Stores, Updates and Deletes that occurred against the file being audited.

This report can be especially useful when transactions against a specific file should normally only occur during "office hours", but transactions occur either before or after those hours.

In considering the output of the Count of Transactions By The Hour, the User is encouraged to recall how ADABAS handles transaction date/timestamps (please refer to the section entitled ADABAS PLOG Caveats).

Summary of User Entered Selection Logic

This portion of the report will detail the Selection Logic the User employed to produce the Audit Report, along with a single row table that shows the number of Store, Update and Delete transactions that were selected for reporting using that Selection Logic.

Detail Report

The Detail section will report out the User-selected details of all transactions that met the User-entered Selection Logic, in addition to providing a transaction "header".

The Header portion will detail:

- Whether the transaction was a Store, Update or Delete,
- The Internal Sequence Number (ISN - or a unique record identifier) of the effected ADABAS record,
- The Date/Timestamp related to the transaction,
- The User who initiated the transaction,
- The Terminal the transaction occurred at,
- The relative sequence number of the transaction as it appeared in the PLOG, and
- A flag indicating if the transaction was "backed out" (This flag will normally only display a value if the User selected to "Retain Back out Transactions" in the Audit Output, which is an optional feature).

Immediately following the transaction header the Audit Detail Report will display the value of the fields that the User specifically requested to be audited. In the example shown below, the first two lines show the common header fields described above.

```
STORE    ISN:          65 TIME: 2014/08/14 10:47:16:2
  USER: ARAGORN  TERMINAL: 000000000000009C SEQ#: 6345          BT-FLAG
     PERSON-ID
         V: 0000008002
     FIRST-NAME
         V: Fred
     LAST-NAME
         V: Stone
```

Following those two header lines, the display shows that the User requested to audit the fields PERSON-ID, FIRST-NAME and LAST-NAME. In the Example above, for each of these audited fields the line following the field's name shows a tag of "V", which is an abbreviation of the word "Value", and is meant to show the value of the field as a result of the STORE. So, in interpreting what the values of the audited fields were after the STORE, we see that the record was Stored with a Value of "0000008002" for PERSON-ID, a value of "Fred" for FIRST-NAME, and a value of "Stone" for LAST-NAME.

In the case of a an Update transaction, the detail portion might look like this:

```
UPDATE  ISN:          80 TIME: 2014/08/14 10:47:16:2
  USER: ARAGORN  TERMINAL: 000000000000009C SEQ#: 10453      BT-FLAG
     PERSON-ID
         V: 0000008018
     FIRST-NAME
         V: Dottie
     LAST-NAME
         B: Kramer
         A: Smith
```

In the above example, PERSON-ID and FIRST-NAME had the values of "0000008018" and "Dottie" respectively, with the "V" indicating that these were the values of the fields Before and After the UPDATE - meaning that they did not change as a result of the UPDATE. However: The field LAST-NAME *did* change, and we see two values. The value "Kramer" is tagged with a "B" (to indicate that this was the value of LAST-NAME Before the Update) with the new value of "Smith" resulting after the Update was applied (indicated by an "A" tag). In this way the User can immediately see that the field LAST-NAME was changed from a value of "Kramer" to a new value of "Smith".

In the case of a Delete, and similar to a STORE transaction, there is no "Before" or "After" values flagged.

An example of a Delete is shown below:

```
DELETE   ISN:          79 TIME: 2014/08/14 10:47:16:2
  USER: ARAGORN  TERMINAL: 000000000000009C SEQ#: 1238       BT-FLAG
     PERSON-ID
         V: 0000008017
     FIRST-NAME
         V: Elizabeth
     LAST-NAME
         V: Bickly
```

In the above example we see that the record with ISN 79 was deleted, and at the time of that Delete the fields PERSON-ID, FIRST-NAME and LAST-NAME had the values of "0000008017", "Elizabeth" and "Bickly" respectively (I.E., they all show with a tag of "V" for "Value").

In situations where a field had no value, or a field was updated to have a value when it previously had no value, or a field was updated to have no value when it previously had a value, then the appropriate "V", "B" or "A" tags will show the corresponding value of the specific field.

The ADABAS PLOG and Transaction Records

As ADABAS operates, it has the optional ability to create a log of the STORE, UPDATE and DELETE transactions that occurred against records in files that are contained in ADABAS - with these changes usually occurring through application programs. The technical name of this log file is the "Protection Log", more commonly referred to as the PLOG.

In order to understand how an audit process against ADABAS works, it is important to have a basic understanding of what the PLOG contains. Essentially, the PLOG is the mechanism that allows an ADABAS database to be restored to any specific point in time - for example: The point in time that an unexpected crash occurred. In order to be able to do that, the PLOG records every change made to every record in every file in a given ADABAS database, with each PLOG transactional record having two parts, a "header" portion and a "data" portion.

The first part of a PLOG transactional record is the "header", with the header containing information specific to the transaction itself, such as:

- The **User-ID** of the User that made the change,
- The **Terminal ID** of the terminal / workstation that the change was made at / through,
- The **Date / Timestamp** of when the change was made,
- The **Database ID** of the ADABAS database that the change occurred in,
- The **File ID** of the ADABAS file the change occurred in,
- The **Internal Sequence Number** (**ISN** - or unique identifier) of the record that was changed,
- The "**Image-Type**" of the transactional record, and
- Other addition information related to the transaction.

In the list of fields shown immediately above, special attention is drawn to the **Image-Type** field that appears in the header portion of a PLOG record, as this field will indicate whether the transactional record is a "Before Image" (BI) or whether it is an "After Image" (AI), as all transactional records in the PLOG will either be flagged as having an **Image-Type** a BI or AI.

When an existing record is updated in an ADABAS file, ADABAS will write two record images to the PLOG to properly record that transaction. All header fields of both records will be the same with the exception of **Image-Type**, with one record (the first record found in the PLOG for this transaction) being flagged with an **Image-Type** of BI (the image of the record BEFORE the Update), with the second record being flagged with an **Image-Type** of AI (or the image of the record AFTER the Update).

While an UPDATE is always reflected in the PLOG by two records, a **STORE** or **DELETE** is always only reflected by one PLOG record. In the case of a **STORE**, the PLOG will contain a single record marked as an AI (to show what the record looked like AFTER the Store occurred). In the case of a **DELETE**, the PLOG will reflect a single record flagged as a BI (to show what the record contained BEFORE the DELETE occurred).

Following the header portion of each PLOG transactional record will be a data portion, with this data portion containing a full record image of the record affected by a transaction. Depending on whether or not a given file contains any recurring field structures (in ADABAS these are referred to as Multi-Valued Fields (MUs) or Periodic-Groups (PEs)), the structure of the data portion may be different even for records from the same file, meaning that not only does the Header portion need to be interpreted as to whether any given record is part of an UPDATE or represents a STORE or a DELETE, the data portion also needs to be interpreted in order to provide a common output structure.

NatQuery-generated Audit processing handles both interpretations, leaving the User to only concern themselves with what the types of changes it is that they would like to examine.

ADABAS PLOG Caveats

In handling ADABAS PLOGS, there are several considerations which need to be reviewed which have a direct impact on how Audit Data is to be interpreted.

These considerations can be distilled down to the following:

- **How ADABAS handles Transaction Date/Timestamps**
  One piece of information that is associated with every transaction is the Date/Timestamp of when a record was Stored, Updated or Deleted. Interestingly however: ADABAS does not capture a Date/Timestamp at the individual transaction record level, ADABAS instead stores transactional records in "blocks", and it is only at the bock level that a Date/Timestamp is written. This means that the Date/Timestamp of a transactional record is taken from the block that contains that record, which in all probability will not exactly match the wall time of when the transaction physically occurred.

  In an ADABAS database that experiences a high-volume of transactions, this is not typically a problem, as blocks would be filling up and being written in very close proximity to the actual Date/Timestamp of when the transaction physically occurred.

  In an ADABAS database that experiences low transaction volumes however, it could take minutes, hours or perhaps even days for a block to be filled so that it can then be written to disk.

  This means that while a CDC process such as NatCDC and NatQuery-generated Auditing will show the Date/Timestamp of every transaction - it may not actually reflect the exact Date/Timestamp of when the transaction physically occurred - NatWorks will deliver that information which is available from within these "block levels".

- **User-IDs**
  In a similar manner to how ADABAS handles Date/Timestamps, ADABAS does *not* store the User-ID of the User who initiated a Store, Update or Delete transaction that gets recorded in the PLOG.

  Stepping back for a moment; an ADABAS PLOG records all sorts of information about the operation of ADABAS as it runs, with the primary items recorded being transactional records (Stores, Updates, and Deletes - or "DATA" records). In addition to "DATA" records, other types of record stores also exist in the PLOG, such as records which show when a User logs in ("OPEN" records) and when a User application issues commands that either makes any given transaction permanent (an "End Transaction" or "ET") or otherwise removes the transaction (a "Back out Transaction" or "BT").

In order to provide the initiating User-ID of any given transaction along with the transactional data, NatCDC/NatCDCSP processes both "OPEN" and "ET"/"BT" records found in the PLOG in addition to processing the transactional record that relates to the file to be audited.

When processing a PLOG file however, it is important to remember that any given PLOG covers a specific period of time. While SAG customer can be assured that "DATA" records will not be split across separate PLOG datasets, this is not necessarily true for "OPEN" records.

The issue with "OPEN" records can be seen in the following example. Let's assume that a User logs on at 6:00AM, and so the PLOG records that logon and also records the User's User-ID in addition to the terminal the User logged in at. As each PLOG file is uniquely numbered, we'll call that PLOG #1. At 6:10AM, the Computer Center takes the steps necessary to "switch" the current PLOG, meaning that the PLOG that was capturing ADABAS transactions (PLOG #1) is closed and a new PLOG is started, and we will call this PLOG #2.

When PLOG #2 is processed, it will not contain the "OPEN" record of the User as that information was recorded on PLOG #1. PLOG #2 can still be processed, but any and all transactions which the User handled will be given a User-ID value of "unknown".

In most shops, but depending entirely on the volume of transactions, PLOGs are switched once a day at a time when no Users are logged on, so this does not usually present a problem.

Should this situation arise, and using the example cited above, the two PLOG datasets would need to be concatenated together and then processed by NATCDC/NATCDCSP in order for the final output to properly reflected the initiating User-ID.

- **Split Transactions / Re-Designated Transactions**
  If PLOG processing is turned on for a given ADABAS database, then transactional information against all files that are contained in that database are recorded in the corresponding PLOG. The transactional recording begins when ADABAS is started and it is automatically closed when ADABAS is shutdown. Additionally, PLOG datasets can be closed via operator commands to ADABAS (in which case the current PLOG is closed and a new one is automatically started), or they can be closed (with a new one automatically started) in certain conditions; for example when a designated PLOG file has reached a certain size.

  When a PLOG is closed, a concern is raised about whether or not an UPDATE transaction can ever be "split" across PLOGs, due to an UPDATE being reflected in a PLOG by two (2) transactional records (a "Before" Image followed by an "After" image). This "split" then would be a situation where the "Before" image is written to a PLOG that is being closed, while the corresponding "After" image from that UPDATE then gets written to the next PLOG.

  Per Software AG, the above situation can never occur, an UPDATE's Before and After

transactions will always be kept together in the same PLOG.

However, there are some unique and special situations where the PLOG will reflect an UPDATE as being a DELETE followed by a STORE. In these unique situations, the record existed in one Data Storage Block, but as a result of the update the record was expanded to the point where it could no longer physically fit into the original Data Storage Block where it resided previously. As a result, the "old" version of the record is removed from the original Data Storage Block through the use of an ADABAS-issued DELETE, with ADABAS then immediately issuing a STORE for the "new" version of the record now stored in a new Data Storage Block that has enough room to hold the expanded record. So: In these cases NatWorks Auditing will reflect exactly what ADABAS did - which was the original record being removed (I.E. a DELETE) from the old block with the new 'updated' record being inserted (I.E. a STORE) into a new block.

Rules for Auditing

With the introduction of audit handling in NatQuery/NatCDC, new Selection Logic operators have been added, specifically the operators "Changed", "Increased" and "Decreased", in addition to the existing operators (Equal, Not Equal, Greater Than, Less Than, Contains and Does Not Contain) . The rules behind these three new operators are as follows:

- **Changed**
  - **STORE Handling**
    When the selection logic operator of "**Changed**" is used against a transaction field, then the details of a record that represents a **STORE** will only be displayed if the field in question is stored with a value. If a record is stored with no value for the field in question then the field did not technically change, and the transaction will not be reported on in the Detail section of the Audit report.

  - **UPDATE Handling**
    When the selection logic of "**Changed**" is used against a transaction field, then this transaction will only be reported on in the Detail section of the Audit report if the value of the field actually changed. It will not be reported on in the Details section if the field did not change at all (I.E. a null or zero value stayed as a null or zero value, or the field had a value but that value remained unchanged).

  - **DELETE Handling**
    When the selection logic operator of "**Changed**" is used against a transaction field, then the details of a record that represents a **DELETE** will only be displayed if the field in question had a value when it was deleted. If a record is deleted with no value for the field being audited, then the field did not technically change, and the transaction will not be reported on in the Detail section of the Audit report.

- **Increased**
  - **STORE Handling**
    When the selection logic operator of "**Increased**" is used against a transaction field, then the details of a **STORE** transaction will only show in the detail display if the field in question was stored with a non-blank or non-zero value (depending on the format of the transaction field) when the record that contained that field was stored. If the record was stored with no value for the audited field, then the field did not technically change and will not be reported on in the Detail section.

  - **UPDATE Handling**
    When the selection logic operator of "**Increased**" is used against a transaction field, then the details of an **UPDATE** transaction will only show in the Detail section if the field being audited physically increased from its prior value (the "Before" value of the field as

compared to the "After" value). The transaction will not be reported on in the Details section if the field did not change at all (I.E. a null or zero value stayed as a null or zero value) or the field had a value but that value remained unchanged.

- **DELETE Handling**
  When the selection logic operator of "**Increased**" is used against a transaction field, then the use of this operator will preclude the display of the details of all **DELETE** transactions. This is because it does not appear to be logical to perceive a field as "increasing" when the record that contains that field is deleted.

- **Decreased**
  - **STORE Handling**
    When the selection logic operator of "**Decreased**" is used, then the use of this operator will preclude the display of the details of all **STORE** transactions. This is because it does not appear to be logical to perceive a field as "decreasing" when the record that contains that field is stored.

  - **UPDATE Handling**
    When the selection logic operator of "**Decreased**" is used against a transaction field, then the details of an **UPDATE** transaction will only show in the Detail section if the field being audited physically decreased from its prior value (the "Before" value of the field as compared to the "After" value). The transaction will not be reported on in the Details section if the field did not change at all (I.E. a null or zero value stayed as a null or zero value) or the field had a value but that value remained unchanged.

  - **DELETE Handling**
    In a similar fashion, when "**Decreased**" is used, the details of a **DELETE** transaction will only show in the detail display if the field in question had a non-blank or non-zero value when the record that contained that field was deleted.

Configuring NatQuery to Produce Audit Reports

In order to run PLOG Audit reports, NatQuery must be installed and configured, and NatCDC must be installed and configured.

To install and configure NatQuery, please refer to the NatQuery Installation and Configuration Manual.

To install and configure NatCDC, please refer to the NatCDC Installation and Operations Manual.

With NatQuery installed and properly configured, with NATCDC also being installed and configured, the only task that needs to be handled in order to create Audit reports is to create a JCL / Script Template that will be specific to running Audit queries. This template is called the **Production CDC Audit Process**, and an example of this template is provided for most ADABAS platforms. This is a task that only needs to be completed once, although it may take several iterations to make sure the JCL / Script template consistently functions as desired.

To configure the Production CDC Audit Process JCL / Script template, perform the following:

1. **Open the Administer JCL / Script Window**
   On an empty NatQuery desktop, and using a version of NatQuery that has been given an Administrative License Key, open the **Administer JCL / Script** window.

   This is done by clicking on **Administer** -> **Environment Configuration** -> **Server Connection Configuration** -> **JCL / Script Information**.

   Taking the above action will open the **Administer JCL / Script** windows.

2. **Select the Production CDC Audit Process**
   With the Administer JCL / Script windows displayed, the User will now select the **Production CDC Audit Process** JCL/Script template using the drop-down selection box labeled as **Server JCL/Script Component** and located in the top right-hand corner of the **Administer JCL/Script** window.

   This action should open what is currently available for the Production CDC Audit Process JCL / Script template.

3. **Edit the Production CDC Audit Process JCL / Script Template**
   If, after selecting the **Production CDC Audit Process JCL / Script** template the main text window opens to a message that states "**This template does not contain any JCL/Script information**", then the template has not yet been created. In this case, it is suggested that the User click the button entitled **Copy From Example Template**; performing that action will copy the installation-provided Example template into the main text window.

In providing an example template, NatWorks is attempting to help the User by providing a base template which can then be customized to meet the needs of the specific shop. In providing example templates however, NatWorks makes no claim that they will immediately work in the User's shop - in fact it is highly likely that it may take several iterations of changes before the template works at the User's site.

When the template has been modified to conform to the JCL / Script requirements for the batch environment at the User's site, the User will click the **Save** button to save the current version of the template into the Environment configuration path.

In approaching the task of customizing the **Production CDC Audit Process**, it is suggested that the User first do the steps above to create a base **Production CDC Audit Process** template, correcting the Script / JCL template to conform visually as close as possible to what is required. Subsequently, the User should then perform the steps necessary to create a simple Audit Process as described in the **Creating Audit Report** section below.

Creating an Audit Report

There are three steps that must be completed in order to run an Audit Process:

1. **Creating the Transaction File - Running NatCDC**
   As noted above, ADABAS creates a log of the activity that occurred against it by writing transaction information to the PLOG. The raw PLOG format however is not conducive to Auditing, so the first step is to convert the PLOG information into usable data, and for that we run a NATCDC (a process which will only process the transactions for one file) or a NATCDCSP process (a process which can process the transactions of multiple files in a single pass). A discussion of what that entails appears below in the section called Creating the Transaction File - Running NatCDC.

2. **Importing the DDM for the Transaction File**
   When the NatCDC Process is generated, one of the options that will be utilized is to have that process additionally generate a Data Definition Module (DDM) that describes the layout of fields that will exist on the Transaction file that is created by NATCDC. In order for NatQuery to understand this layout, this DDM must be imported into NatQuery, and a discussion of what this step entails is described below in the section called Importing the Generated DDM.

3. **Generating and Executing an Audit Process**
   Once the transaction file is created and NatQuery understands the layout of that transaction file, NatQuery can then be used to generate (and execute if permitted) the program that is required to produce the desired Audit results. The effort to accomplish this is described in the section called Generating and Executing an Audit Process.

4. **Retrieving the Audit Process Report**
   Once the Audit Process has been executed on the ADABAS / NATURAL Server machine the results can be automatically downloaded to the NatQuery workstation where it can be automatically opened in an editor of the User's choice. The effort to monitor, download and then invoke an external editor is described in the section called Retrieving the Audit Process Report.

Creating the Transaction File - Running NatCDC

In order to run an Audit Process, a User must first generate and execute a NATCDC or a NATCDCSP process that will convert PLOG information into usable data - or otherwise this task will be performed by a Database Administrator (DBA), or by an automated process set up by a DBA.

Prior to moving forward with Auditing, the User and/or DBA should review the **NatCDC/NatCDCSP Installation and Operations Manual** which gives the information needed to setup the NatQuery environment so that the NATCDC PLOG process can be executed.

While a separate manual covers how to generate NatCDC/NATCDCSP processing, there are two generation options that must be used in order to create the basis for Auditing:

- The User must utilize the option to **Generate a DDM** that describes the layout of the final output, and,
- The User must utilize the **All** option for the **Logical Record Handling** option (as opposed to the delivery of a **Delta** or **Logical First and Last**).

To move forward immediately with Auditing, it is strongly suggested that a User use a **NatCDC** process to create the input file for Auditing versus using the output of a "Multi-File" **NATCDCSP** process (see the Important Note further below) as a NatCDC process takes less steps and is a more straight-forward.

Simplistically, and after the NatQuery Environment Configuration is configured to support the PLOG processing of the desired source file, the User will initiate the **Generate NatCDC Objects for Single File** by clicking on **Administer** -> **Generate NatCDC Objects for Single File** while on an empty NatQuery desktop. This action will invoke the **Generate NatCDC Objects for Single File**, which presents 6 tabs.

On the first tab, named as **File Information**, the User will select the file they wish to Audit, and they will additionally supply the **DBID** (Database ID) that the file is contained in, along with the **File Number** of the desired file. While **Expanded File information** can be supplied on this first tab, this is only needed in very special circumstances and can likely be ignored. The User will then click the second tab, named as **Field Information**.

On the second tab, the User has the ability to "drop" fields they are not interested in from the final output. While this is a feature that can drastically cut down the overall size of the output file, it is initially suggested to not Omit any fields, while noting that C* fields are automatically Omitted and should probably be left that way unless the User is specifically interested in these values. So, in most cases the second tab can be examined but with no changes made, such that the User will now click on the third tab, named as **Output Options**.

The **All** option for **Logical Record Handling** is found on this tab, and in order to generate any Audit processing, the **All** option must be checked as shown in **Figure 1**.

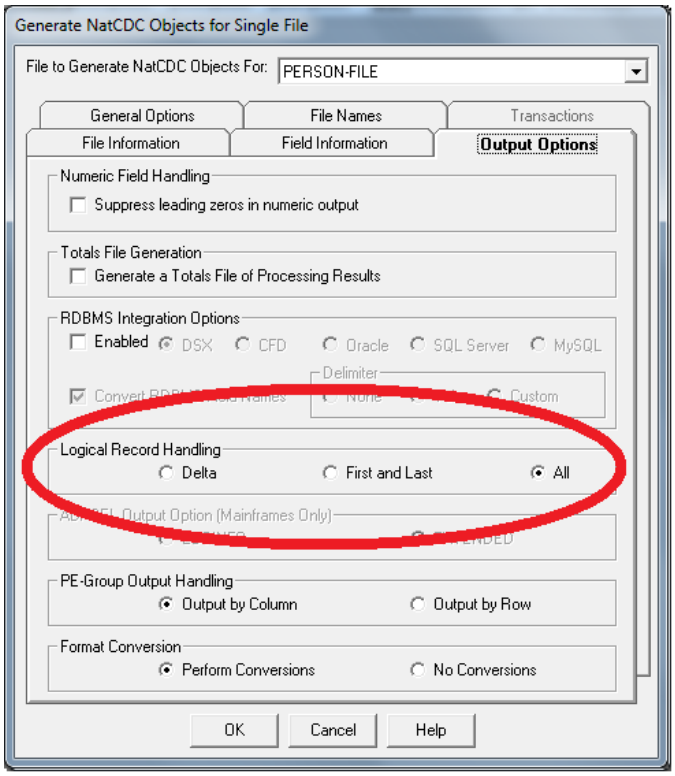

Figure 1

In most cases, and other than setting the Logical Record Handling option to "All" as seen in **Figure 1**, the User will then proceed to the fourth tab by clicking the tab named as **General Options**.

The **DDM Generation** option is found on the **General Options** tab of the **Generate NatCDC Objects for Single File** window. In order to generate any Audit process, the option to **Generate a DDM of Final NatCDC Output** must be checked as shown in **Figure 2**.

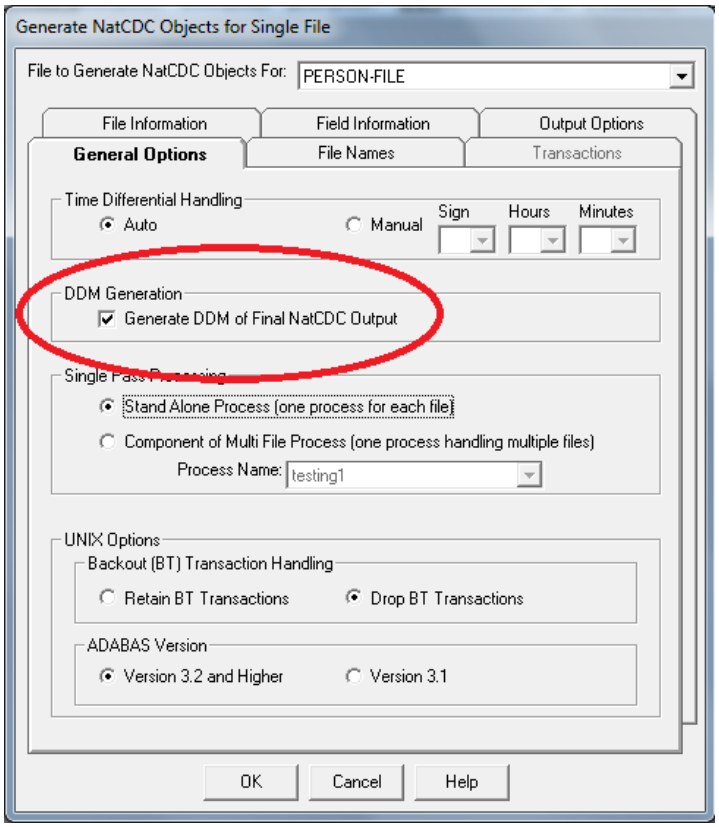

Figure 2

While still on the General Options tab, the User should check to insure that the **Single Pass Processing** option is set to use the **Stand Alone Process** option, which is the easiest and fastest way to produce the input file for an Audit Process. In order to run an Audit process it is strongly recommended that the User selects the "**Stand Alone Process**" option by selecting the radio button beside this option as seen in **Figure 3**. While the output of a Multi File Process can be used as input for auditing there is an additional step required (see the **Important Note** further below).

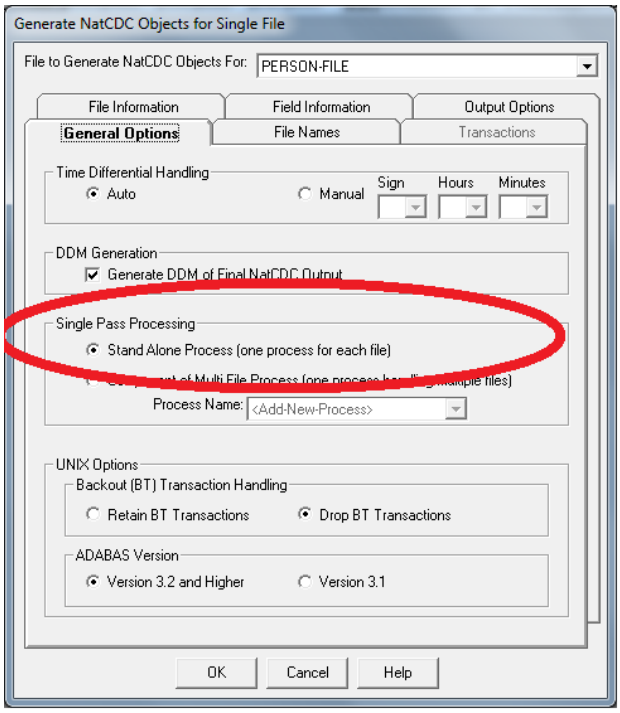

Figure 3

With the **DDM Generation** and **Single Pass Processing** options set, the User can review the additional options available on the General Options tab, but in most cases they can just accept the defaults and then proceed to the fifth tab by clicking on the tab named as **File Names**.

On the File Names tab, the User will review the file names and other data that will be used to execute the PLOG processing request.

Of specific interest on the File Names tab is the PLOG Data (Input) text field, which must be pointed at the output of the execution of the ADAPLP utility (for Open Systems) or ADASEL (for mainframe systems) against a raw PLOG file. It is the output of ADAPLP that becomes the input to a NATCDC or NATCDCSP execution on UNIX or Windows; in the mainframe environment the execution of a NATCDC or NATCDCSP run would follow the execution of the ADASEL PLOG utility.

As described in the NatCDC Installation and Operations Manual, the sixth tab named as **Transactions** will be disabled when NatCDC is run against ADABAS on UNIX or Windows platforms.

With all appropriate tabs reviewed and input as described above, the User will hit the OK button, which will move the PLOG process to the server, and, depending upon the configuration of NatQuery, can additionally remotely execute the PLOG process that will produce the input file that can then be audited.

Important Note:
The basic difference between a **Stand Alone NatCDC** process and a **Multi-File NATCDCSP** process is that a NATCDC process will only process the PLOG transactions for a single file in one execution whereas a NATCDCSP process can process the PLOG transactions for multiple files in one execution. This means that the final output of a NATCDC process will only contain the transactional records for the single file it processed, whereas the output of a NATCDCSP process can contain transactional records for multiple files (unless the NatCDCSP process was only run against a single file - which is allowed).

If the output of a NatCDCSP process is to be used as input to an audit process, the output file of the NATCDCSP process must only contain transactions for the specific file of interest (which can occur if NATCDCSP is run for only one file), or the transactions from the file of interest must first be "split-out" so that only those transactions from the file of interest are in a single file. On UNIX, LINUX and other Open Systems, this splitting can be accomplished using the "splitter' program provided by NatWorks. In mainframe environments, this split can usually be accomplished with most sort packages, or alternatively NatWorks provides a NATURAL program that will accomplish this splitting.

Importing the Generated DDM

As one of the outputs of the **Generate NatCDC Objects for Single Pass**, a Data Definition Module (DDM) will have been generated that exactly matched the expected layout of the final output of the NatCDC Process.

In order to utilize the audit capabilities of NatQuery, this DDM needs to be imported into NatQuery so that the file layout can be programmatically understood.

Importing the DDM is accomplished by using the Import DDM function, which is accessed from an empty NatQuery desktop by clicking on **Administer** -> **Environment Configuration** -> **DDMs/FDTs** -> **Import DDM / FDT**. Performing this action will invoke the **Import DDM(s) / FDT(s)** window as seen in **Figure 4**.

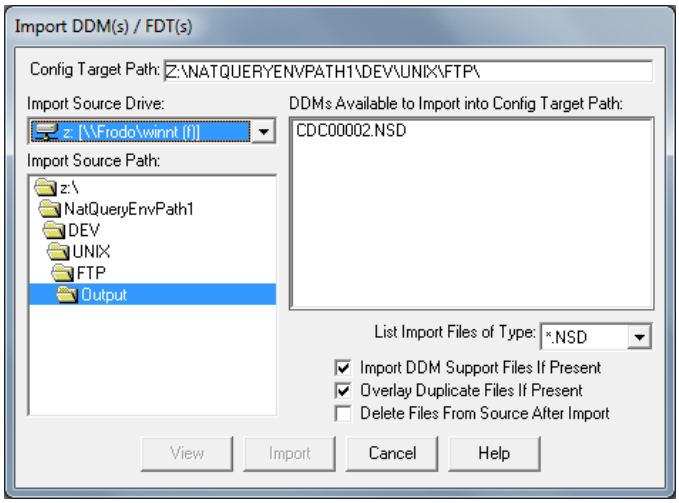

Figure 4

To import the DDM, the User will click on the appropriate DDM as shown in the **DDMs Available to Import into Config Target Path** list box. If there are multiple DDMs shown and the User is unsure of which one to select, then the User can click on each DDM and then click the View button, which will not only show the DDM, but will also show how it was created.

Once the User has selected the correct DDM, the User can then click the **Import** button, which will result in the DDM being imported into the designated NatQuery Environment Path (**Config Target Path**). After selecting the DDM and prior to clicking **Import**, the User should consider clicking the **Delete Files From Source After Import**; performing this action prior to clicking **Import** will allow the DDM to be imported into the NatQuery Environment Path and will then delete the DDM from the designated Output / Download directory of NatQuery.

After the DDM is imported, the User will close the **Import DDM(s) / FDT(s)** window by clicking the **Cancel** button. As the window closes, the User will be presented with a messagebox similar to that shown in **Figure 5**; the User should click the **Yes** button to allow a verification process to occur. Once

the verification completes, the DDM will have been imported into the NatQuery Environment Path, and will now be ready for use by an Audit query.

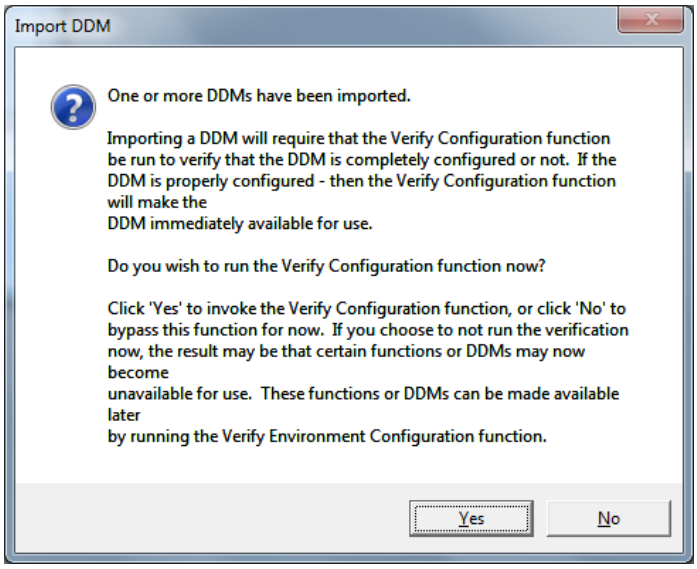

Figure 5

After clicking the **Yes** button to the messagebox shown above, the User will be presented with a **Verify Configuration** report window. as shown in **Figure 6**. In almost all cases, there will be no problem with the new DDM. This is because the DDM will represent a sequential file, and as such it will not need any configuration for such things as **Occurrence Information** or **Descriptor Statistics**. In most cases then the User can simply click the **OK** button, which will then return the User to an empty NatQuery desktop.

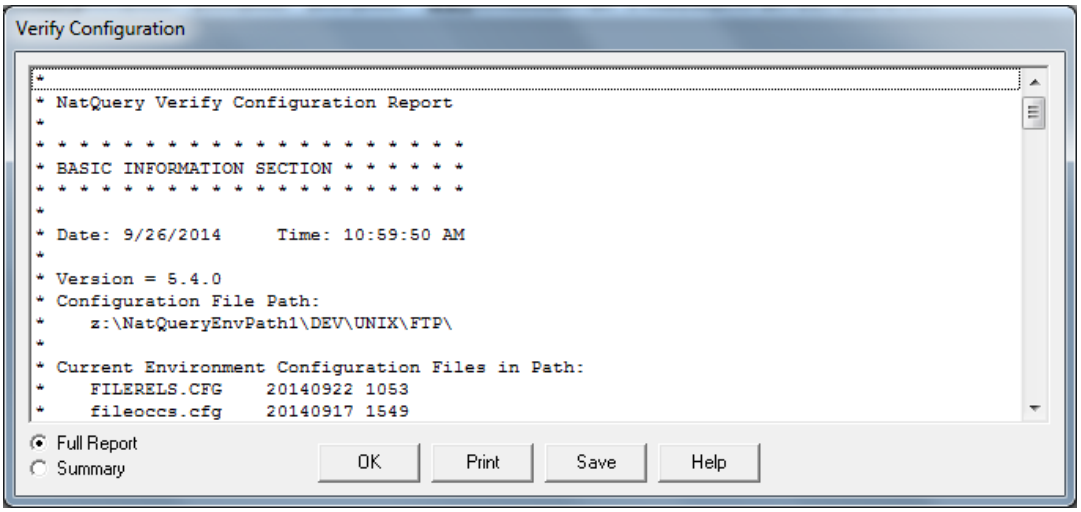

Figure 6

With the DDM imported, the User needs only to verify that the NATCDC process gets executed without error on the Server Platform. If no errors are encountered, then the input file for an Audit process has been created and the User can move to the next step.

Generating and Executing an Audit Process

Having run the necessary NatCDC process that will create the input file for an Audit run, and with the corresponding DDM imported into NatQuery, the process of creating an Audit report can now begin, with this process following the general steps to create a NatQuery query.

As an overview, the basic steps to create an Audit report are as follows:

1. Select the DDM
2. Select the Fields from the DDM to be Audited
3. Enter the Selection Logic for the Records to be Audited
4. Sending the Audit Query to the Server
5. Retrieving the Audit Report

Each of these steps is outlined in the following sections.

Select the DDM

To select the DDM which represents the NatCDC output, the User will click on the **New Query** icon from the NatQuery toolbar. Performing this action will invoke the **Select File** window as seen in **Figure 7**.

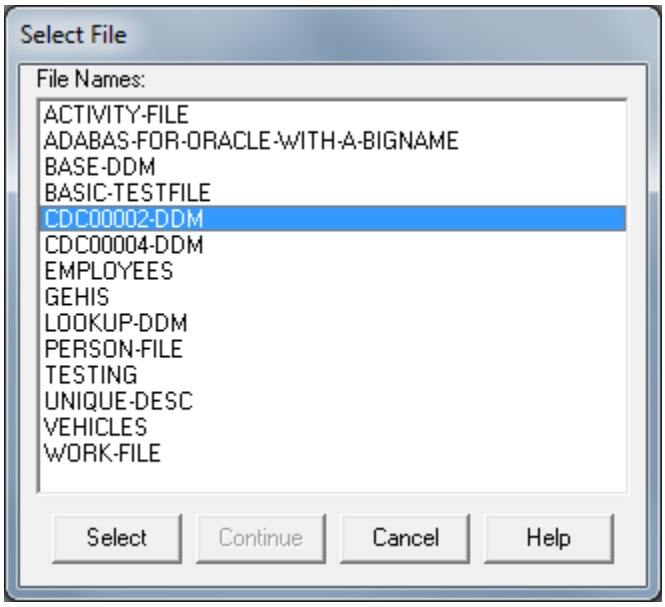

Figure 7

The User will click on the DDM name that represents the output file from NatCDC, and will then click the **Select** button.

Subsequent to clicking the **Select** button, the User will be presented with a messagebox titled as **PLOG Audit Process** similar to what is seen in **Figure 8**. The purpose of this messagebox is to request from the User whether they wish to create an Audit query or a "traditional" query. The difference between the two is that if the User requests an **Audit Query**, then the resulting generation will contain the logic necessary to interpret and match transactional records as necessary in order to deduce Store, Update, or Delete transactions, which is the basis for proper Auditing. If the User requests a "traditional" query, then no such interpretation will occur, with these records then being queried in their "raw" form (I.E. as individual Before and After records but no indication of whether these individual transactions represent Stores, Updates or Deletes).

In response to the **PLOG Audit Process** messagebox, and assuming that the User wishes to create an Audit process, the User should click the **Yes** button. Otherwise the User should click the **No** button to create a "traditional" query or they should click the **Cancel** button to cancel the generation process altogether.

Subsequent to clicking the **Yes** button, the User will be automatically presented with a **Select Fields from *ddmname* to Audit** (with ***ddmname*** being the name of the DDM the User selected in this step), and they can then continue with the next step.

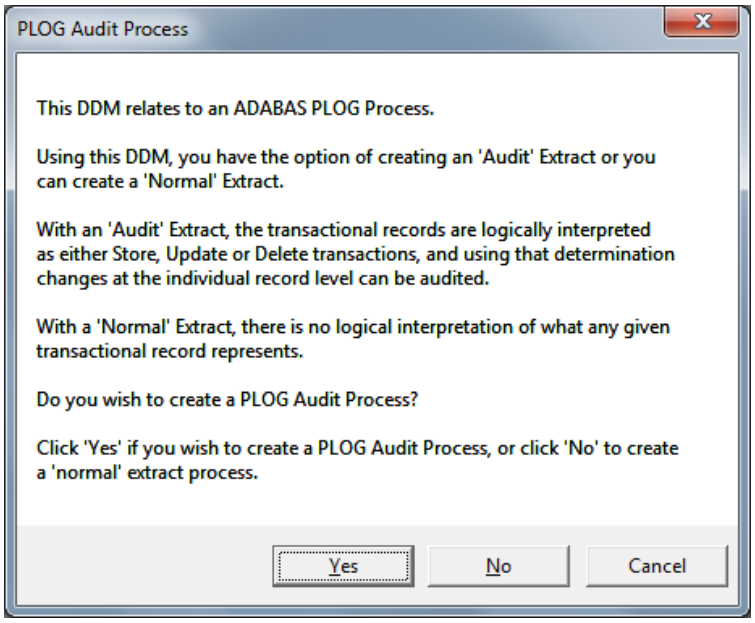

Figure 8

Select the Fields from the DDM to be Audited

On the **Select Fields from *ddmname* to be Audited**, an example of which is shown in **Figure 9** below, the User will be presented with just the data fields (not the "header" fields) from the DDM which are available to be audited.

The fields that are selected in this window will be the fields which the User specifically wishes to Audit, and will be the fields that will be specifically shown in the Detail section of the resulting Audit Report. While all fields can be selected to be audited- ***this is highly discouraged*** **-** as the resulting report will have the potential to be extremely voluminous.

As a hard rule, the User should proceed by selecting ***ONLY*** those fields which the User wishes to examine and audit.

Another important recommendation in determining the fields to be audited is that, in addition to selecting the fields to be audited, the User should select any "key" fields that will serve to tie the transactions being reported on back to the current records in the source ADABAS database. In considering this, it should be noted that the Internal Sequence Number (ISN) will be shown in the Detail Audit Report, with an ISN being unique to each record in a given ADABAS source file.

In the DDM being used for this document, the field **PERSON-ID** is a key (unique) field to the actual ADABAS source file. To make the Detail Audit Report be of the most use in linking the transaction back to the record in the source ADABAS file, key fields should be selected so that they will show on the Audit Report.

For the purpose of an example, the following steps will assume that the User is interested in examining all transactions which either affected the field **FIRST-NAME** and **LAST-NAME**, and because **PERSON-ID** is a key field, it will also be selected.

To select a field on the **Select Fields from *ddmname* to Audit** window, the User can single left-click on the field and then click the Add button, they can just hit the Enter key while the field is highlighted in the left pane, or they can simply double-click the field. Any of these actions will copy the name of the selected field from the left pane (labeled as **Fields Available for Auditing**) to the right pane (labeled as **Fields Selected to be Audited**).

As can be seen in **Figure 9**, for the purpose of example, the fields **PERSON-ID**, **FIRST-NAME,** and **LAST-NAME** have been selected so as to Audit for changes to the field **LAST-NAME**, while also having the Audit report show the contents of the related field **FIRST-NAME**, in addition to showing the contents of the key field of **PERSON-ID** .

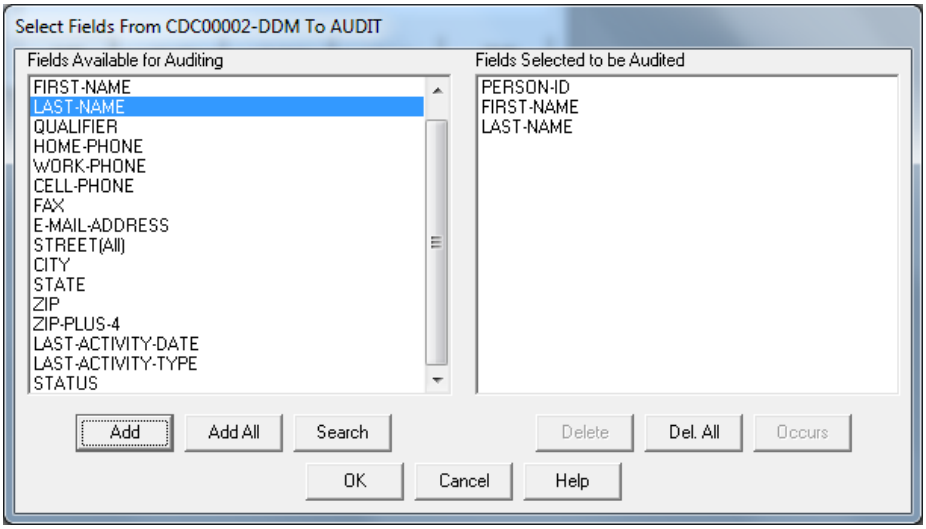

Figure 9

Once the fields to be Audited are selected, the User will click the **OK** button. This will cause the **Select Fields from *ddmname* to Audit** to close, with the User being automatically presented with the **Selection Logic** window, and can continue to the next section.

Enter the Selection Logic for the Records to be Audited

The Select Logic window should now be displayed and should look similar to **Figure 10**.

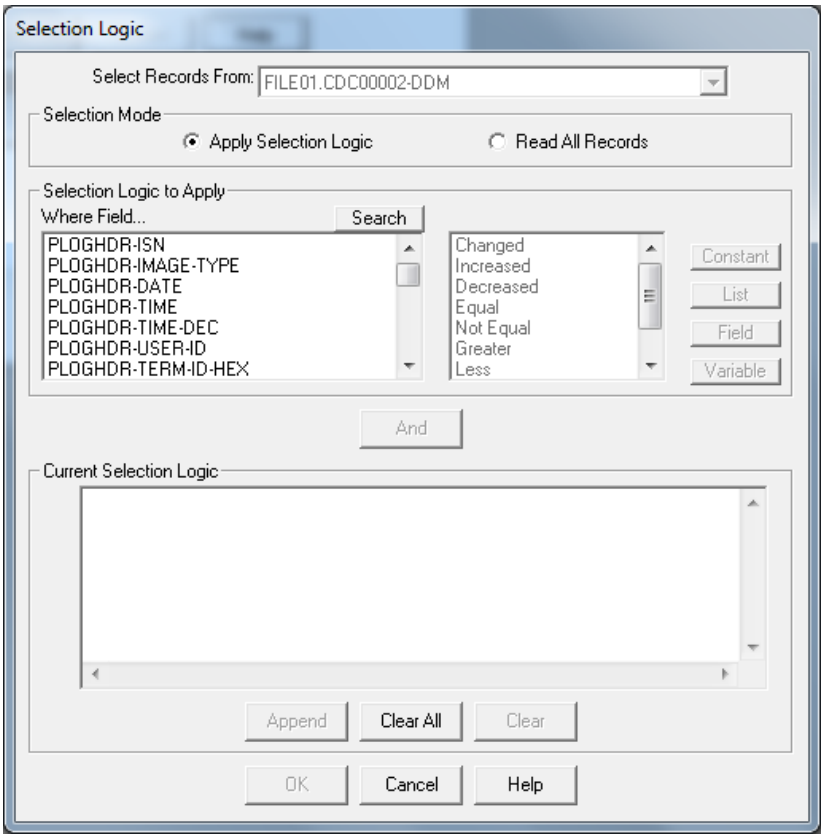

Figure 10

As opposed to how the **Select Fields From *ddmname* to Audit** presented fields, the **Selection Logic** window will present ALL fields from the selected DDM, which will now include the "header" fields previously discussed.

Referring back to the example Audit Report being performed, the User is interested in Auditing all records where the field **LAST-NAME** was changed, so the User would find and then select the field LAST-NAME from the "**Where Field...**" list box. The field can be found by the User either by simply scrolling the "**Where Field...**" list box, and once found can be selected by a single left-click. Alternatively, the **Search** button may be clicked, which will allow for more options to locate the needed field.

Subsequent to selecting the field **LAST-NAME**, the middle list box labeled as "**is...**" will become enabled. Per our example, the User is interested in Auditing records where **LAST-NAME** was changed, so the User will select the "**CHANGED**" operator form the "is..." list box.

Performing the above would result in the **Selection Logic** window that will look similar to what is seen in **FIGURE 11**. As the User has now entered the Selection Logic needed to produce the desired results, the

User would now click the **OK** button; this will return the User to the NatQuery desktop, which should look similar to **FIGURE 12**. The User can now continue to the next step.

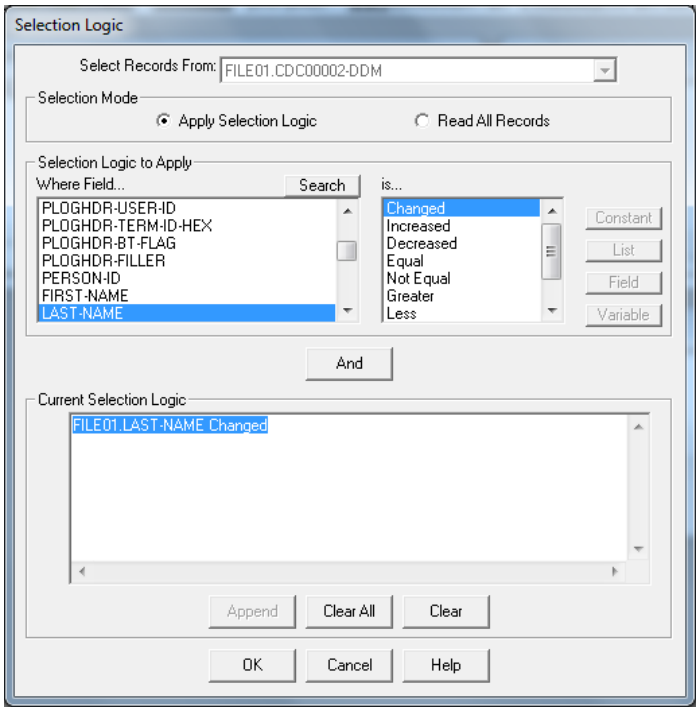

FIGURE 11

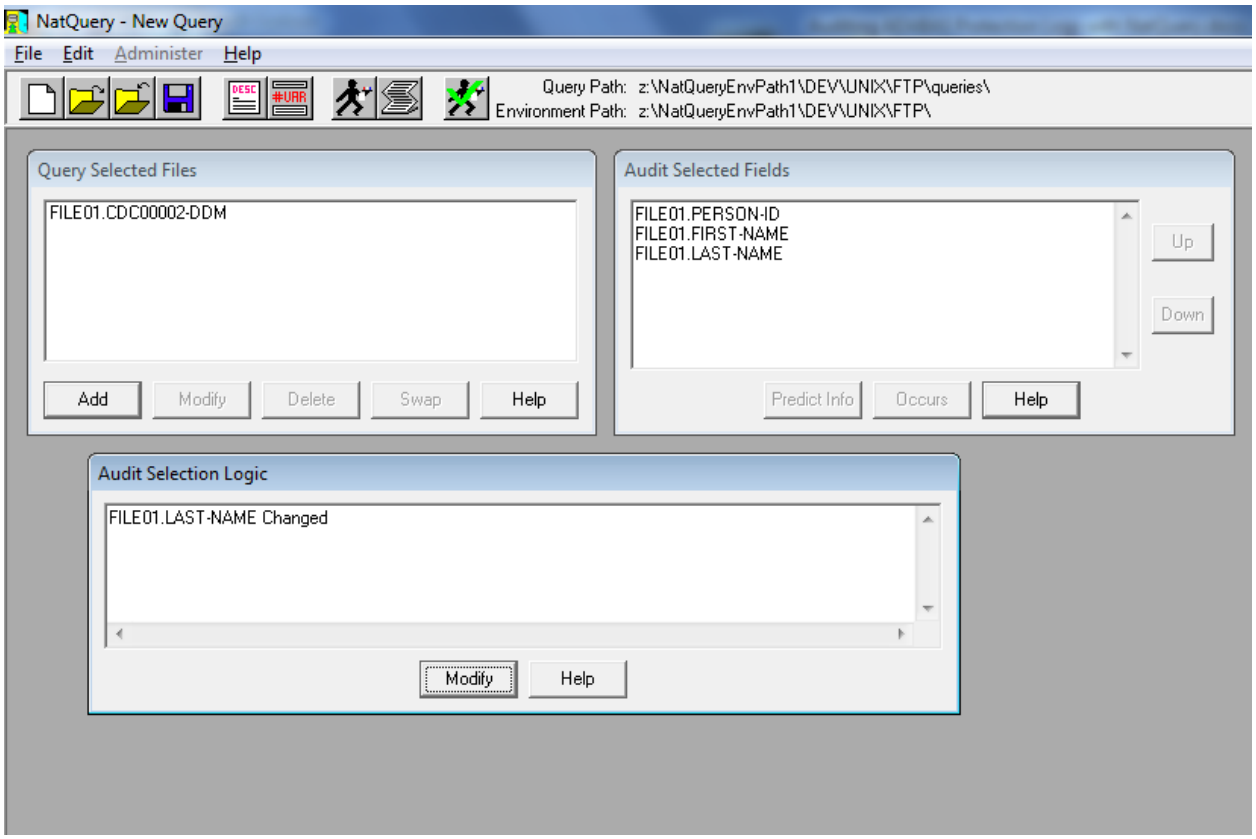

FIGURE 12

Sending the Audit Query to the Server

With the Audit Query displayed on the NatQuery desktop, the User can now send this query to the Server for execution. To accomplish this, the User will now click the **Send to Server** icon (the icon which looks like a person running who is carrying a tray), or can alternatively proceed by clicking the **File** Icon and then clicking **Send To Server**.

Since the Audit Query being handled is a new query, no description of this query exists as of yet, so in response to the User clicking **Send To Server**, the next window to be displayed will be the **Query Description** window as seen in **FIGURE 13**.

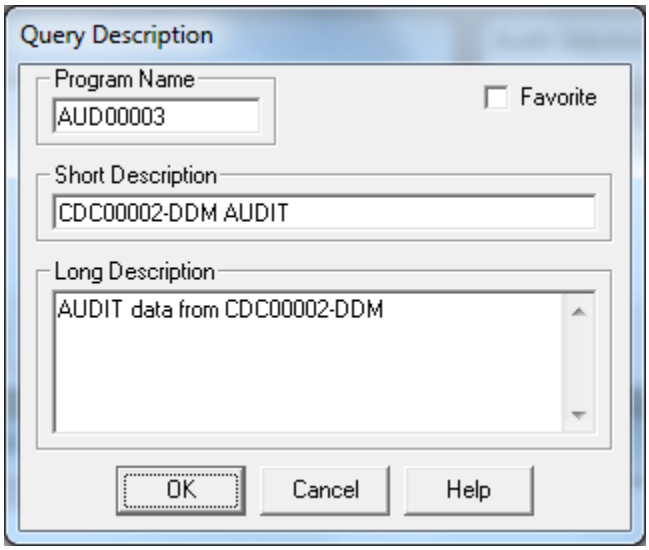

FIGURE 13

As can be seen above, NatQuery automatically provides a default but unique program name and it also provides a default Short and Long Description of the Audit query. The information that is provided in both the short description and the long description can be of significant value in finding this query for possible re-use later, so the User is encouraged to provide descriptions which will allow the query to be found quickly later.

After the User has reviewed and possibly changed the **Program Name**, the **Short Description** and / or the **Long Description**, the User would click the **OK** button to proceed. This action will cause the **Send To Server** window to be displayed similar to what is seen in **Figure 14**.

When NatQuery presents the **Send To Server** window, it will automatically select an **Extract Type** of **NatCDC Audit Report**, and because the User previously indicated that they wanted an Audit query - NatQuery will not allow **Extract Type** to be changed. If another **Extract Type** was wanted then the User should click the **Cancel** button and start over to create the query they prefer.

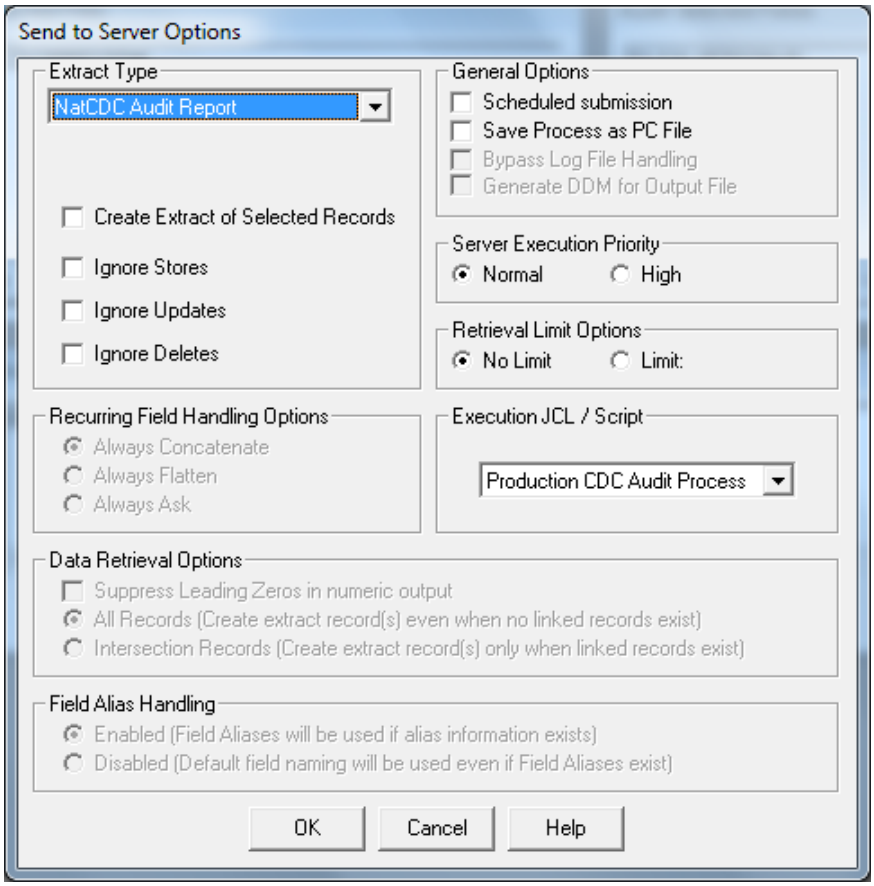

FIGURE 14

On the **Send to Server Options** window, there are currently four options that are specific to Audit query creation. These options are all located in the **Extract Type** frame when an **Extract Type** of **NatCDC Audit Report** is selected, with these being:

1. **Create Extract of Selected Records**

   In referring back to the example of finding all transactions where "**Last-Name Changed**", it can be inferred that it is likely that not all records in the original PLOG transaction file will meet this criteria. Therefore, the records which match this Selection Logic will be a subset of the original file.

   In certain instances, it may be necessary to create a subset of records as its own extract, such that another Audit Query could then be generated that uses this subset file as its input; thereby providing the ability to drill down further into this subset of transactions.

   If the User wishes to create an extract of the original PLOG that only contains transactional records that match the User's entered Selection Logic, then the User should insure that the checkbox associated with **Create Extract of Selected Records** is ticked.

Later in the processing, the User will be given the opportunity to give the extract file a name other then the default provided and can then subsequently use that named file for input into other Audit or even "regular" NatQuery extracts.

2. **Ignore Stores**
   In certain instances, the User may not be interested in seeing **STORE** transactions in the Detail section of the Audit report.

   Should this be the case, then placing a tick in the checkbox labeled **Ignore Stores** will cause the generated Audit Report to not include any **STORE** transactions in the Detail section of the final output.

   Placing a tick in this checkbox however will not affect the counting of **STORE** transactions in the Summary reports.

3. **Ignore Updates**
   In certain instances, the User may not be interested in seeing **UPDATE** transactions in the Detail section of the Audit report.

   Should this be the case, then placing a tick in the checkbox labeled **Ignore Updates** will cause the generated Audit Report to not include any **UPDATE** transactions in the Detail section of the final output.

   Placing a tick in this checkbox however will not affect the counting of **UPDATE** transactions in the Summary reports.

4. **Ignore Deletes**
   In certain instances, the User may not be interested in seeing **DELETE** transactions in the Detail section of the report.

   Should this be the case, then placing a tick in the checkbox labeled Ignore Deletes will cause the generated Audit Report to not include any **DELETE** transactions in the Detail section of the final output.

   Placing a tick in this checkbox however will not affect the counting of **DELETE** transactions in the Summary reports.

While not being a feature that only relates to Audit Reporting, it is suggested that the User take note of the **Retrieval Limit Options**, with specific attention to the ability to either **Limit** the number of transactions reported on to a User-provided number of transactions, or have **No Limit**. If Limit is used, then the generated Audit Process will stop processing when it reaches the User-enter number of transactions.

After reviewing and possibly selecting the options desired that are shown on the Send to Server window, the User will click the **OK** button to finalize the processing needed to send the query to the server.

Subsequent to clicking the OK button, the User will be presented with a User prompt similar to what is shown in **Figure 15**.

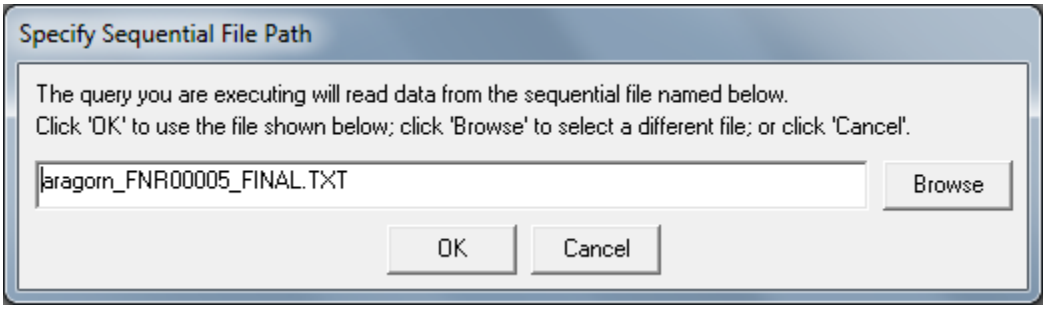

FIGURE 15

The **Specify Sequential File Path** prompt allows the User to specify the name of the sequential file that will be used as input to the Audit Report processing.

Some discussion is needed to understand where the file name that is being presented comes from.

When the initial PLOG process was run that processed the raw PLOG file, one of the outcomes was the generation of a DDM that would match the layout of the output from the PLOG process, and another outcome is a sequential file of usable PLOG data after that PLOG process is executed. As a result of the DDM being generated at the same time the process was generated to create the sequential input file for an Audit Process, the name of the sequential file that was to be created gets physically stored in the DDM file. So: When an Audit Process is created that points at this generated DDM, the Audit Query will "pick up" the name of the sequential file from the DDM.

After the query is created and saved however, the name of the Sequential File that was used as input then becomes stored as part of the internal Audit Query specification itself, so that the next time the Audit Query is run it will point at the same file it used as input the last time the Audit Process was run (assuming the Audit Query is in fact saved after it is submitted).

If this is the first time an Audit is being run against the transactional input file provided by a NATCDC process, or the User is otherwise re-executing the Audit process again and wishes to use the original input file, then the User only needs to click the OK button to continue the process of sending the Audit Process to the Server.

If however the User has previously run an Audit Process that created an Extract file (I.E. a subset of the an original transaction log input file), then the User will supply that file name.

Either way, if the User changes the suggested file name to any other value and then saves that query, the newly provided file name will be saved with the internal query specification, and that new name will be presented for use the next time the Audit Query is Sent to the Server (and can be over-ridden at that time).

Subsequent to responding to the **Specify Sequential File Path** prompt, the User will, if NatQuery has been configured to move generated objects to the Server, be presented with a prompt similar to what is seen in **Figure 16**. This prompts simply asks for confirmation to begin the process of moving the Audit Report process to the ADABAS/NATURAL server - and unless the User no longer wants to submit the generated process, the User should click the **Yes** button. Clicking the **No** button will have the effect of cancelling the query.

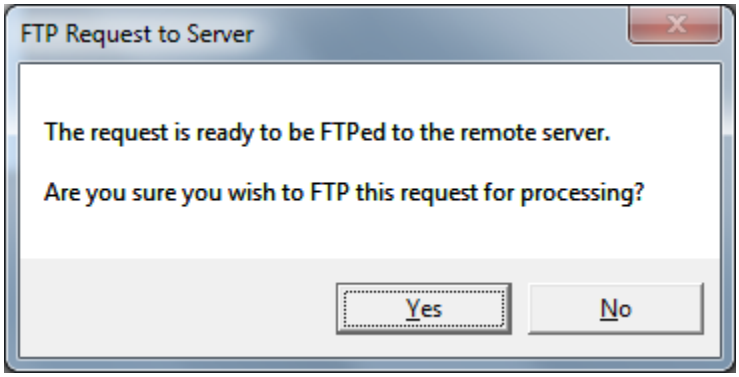

FIGURE 16

In most cases, NatQuery will have been configured by an Administrator to use File Transfer Protocol (FTP) to move the generated process to the server. In that case, NatQuery will respond to the User clicking **Yes** on the **FTP Request To Server** prompt by presenting a prompt similar to what is seen in **Figure 17**.

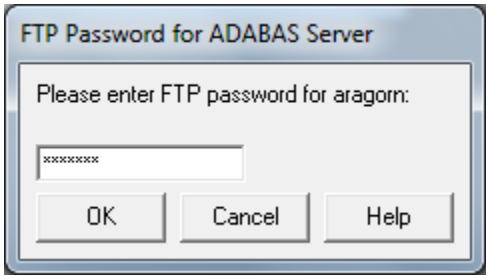

FIGURE 17

In response to the **FTP Password for ADABAS Server** prompt, the User will enter their FTP password and will then either hit the Enter key or will click the **OK** button.

When NatQuery is used against Windows and Open Systems platforms, then NatQuery can be configured to "remotely execute" the Audit Process that was just moved to the NATURAL/ADABAS

Server platform. If this is the case, then the User will be presented with a messagebox similar to what is seem in **Figure 20**. If this prompt is presented, the User will click the OK button to have NatQuery send the appropriate command(s) to the remote platform. When NatQuery is used against mainframes this prompt will not be shown, as the FTP process itself will initiate the execution (if properly configured).

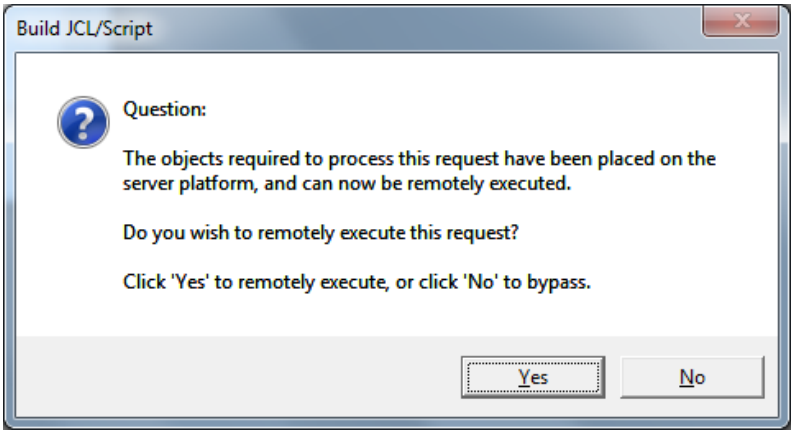

FIGURE 20

If the prompt shown in **Figure 20** is presented and the User has clicked **Yes** to that prompt, then after NatQuery has sent the appropriate command to the remote server, the User will be given an acknowledgement prompt similar to what is shown in **Figure 21**. This prompt asks for the **Network Server Password**, which is usually required to successfully perform a remote execution request.

If the User has not yet provided their password for the current session of NatQuery, then the displayed **Network Server ID** will be the value of the **Server User ID** from the **NatQuery Configuration**, **User Identification** tab, with the **Network Server Password** being blank. The User can simply enter the appropriate password for the provided **Network Server User ID** and then click the OK button, or the User can alternatively overtype the provided **Network Server ID** with the value of a User Id that is authorized to perform remote execution requests, then provide the appropriate password for that User ID, and then click the **OK** button. Once entered, NatQuery will provide both entered values for the duration of the NatQuery session (NatQuery never stores passwords beyond a given session).

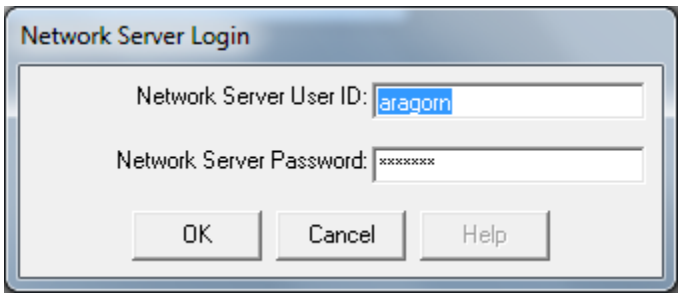

FIGURE 21

If NatQuery has been properly configured to interact with the remote server, then the User should now receive a prompt similar to what is shown in **Figure 22**.

The User can now proceed to the next step, Retrieving the Audit Report.

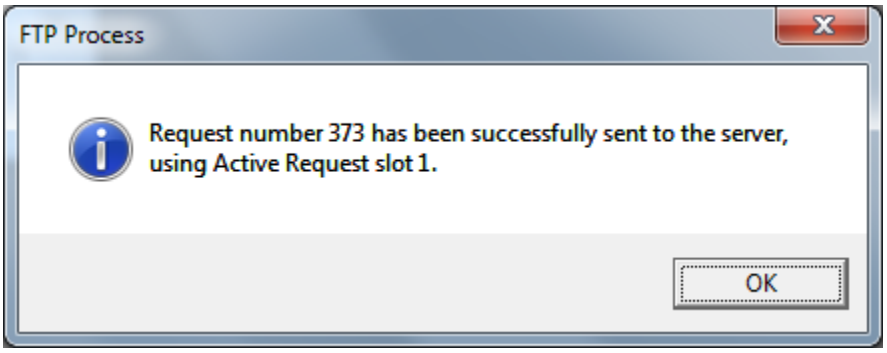

FIGURE 22

Retrieving the Audit Report

When NatQuery submits a request, such as an Audit Report request, the User can track the execution progress by utilizing the **Check Server** function of NatQuery, with this function also allowing the User to immediately download and open the Audit Report.

To access the **Check Server** function, the User can click the **Check Server** icon (an icon which appears to be a man running with a tray with a large green checkmark for the top), or can alternatively click the **File** drop down menu and then click the **Check Server** option. Either method will invoke the **Check Server** window which will look similar to what is shown in **Figure 23**.

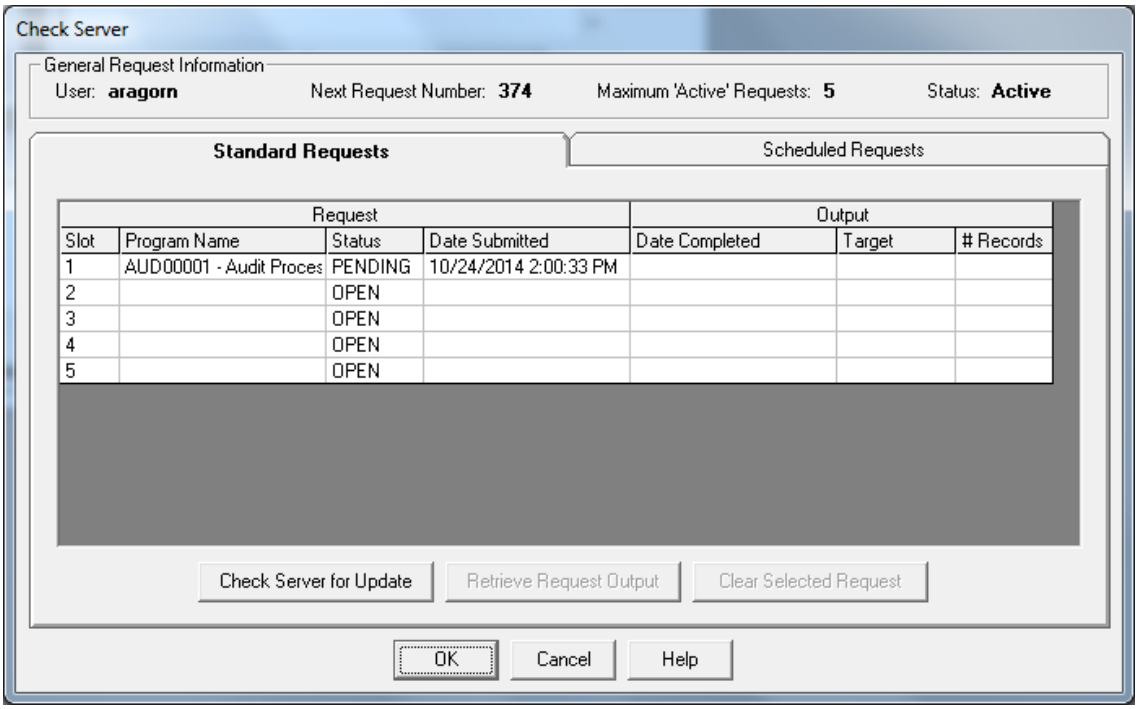

Figure 23

On the **Check Server** window, the User can monitor what Query / Audit processes they have sent for execution on the server platform, and if those request are completed successfully - the User can initiate an automated download of the results.

On the **Check Server** window shown in **Figure 23** above, we can see the User "aragorn" has submitted a request, and when this window is initially displayed all requests that may be shown will display a **Status** of "**Pending**".

To check to see if any Pending requests have been completed, the User will click the **Check Server for Update** button. Depending upon how NatQuery has been configured and the processing load of the server, in most cases any Pending queries will be executed in a short amount of time, and subsequent to clicking the **Check Server for Update** button, the display will be changed for any executed queries, which will update the **Status** from **PENDING** to either **FAILED** or **Done**.

If NatQuery reports that there are no changes to report, then the Audit Process has not yet executed or is in the process of being executed, and the User should use the Check Server and the Check Server for Update function at a later time. If the **PENDING** status persists for a significant length of time, then the User should check with the Administrator to see if there is an unknown issue with the server or possibly a problem with the Audit Report itself.

If NatQuery reports that a process has **FAILED**, then moving the mouse over the query should display the error number of the error encountered, and after making note of this error the User should then contact the NatQuery Administrator for assistance in diagnosing what the problem may be.

In most cases however, when the Check Server for Update button is pressed, the update of the display will result in the **Status** of **PENDING** queries being updated to have a **Status** of **DONE**, and with the **Check Server** window being updated to look similar to what is shown in **Figure 24**.

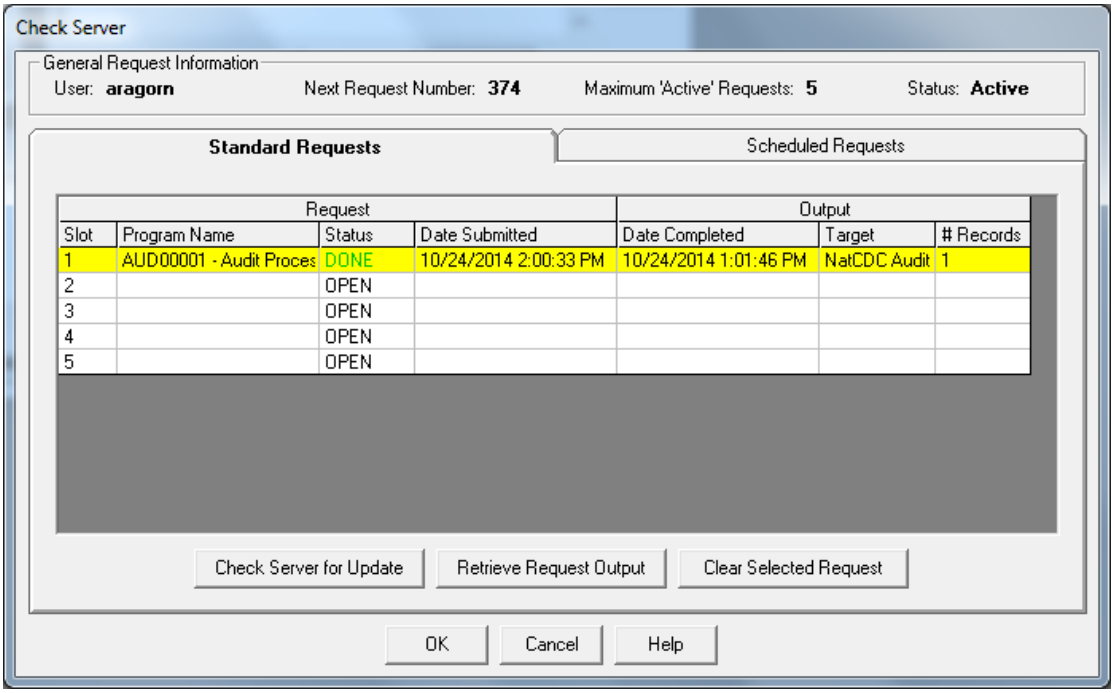

Figure 24

When a request shows as **DONE**, then the User can click on the request row to highlight the request that is **DONE**. Subsequent to clicking on a request that has a Status of **DONE**, the **Retrieve Request Output** button will become enabled.

Clicking the **Retrieve Request Output** button will initiate the process of moving the Audit Report from the remote server down to the NatQuery workstation, with that report then being opened in the designated external editor.

Once successfully downloaded, the **Check Server** display will be updated to remove the reference to the output that was just downloaded - thus freeing up that "slot" for another request from the User.

Setting an External Editor

The result of a NatCDC Audit Process is a report. While that report can be printed directly from the server machine, by default NatQuery can support the automated downloading of that report, with the report then being opened on the workstation for review and / or local printing.

When NatQuery is instructed to download an Audit report, NatQuery will attempt to automatically open that report for viewing on the workstation, and by default the simple Windows editor called "NotePad" will be used, as Notepad comes with all versions of Windows.

The User however has the option of using any editor they have access to on their workstation environment, such as Microsoft Word.

If the User wishes to change NatQuery's default behavior of using NotePad as an external Editor, this can be changed using the **NatQuery Configuration** function.

To change the editor that NatQuery will use when opening a Audit report, perform the following:

1. **Open NatQuery Configuration Window**
   On an empty NatQuery Desktop, click **Administer** -> **NatQuery Configuration**.

   This action will open the **NatQuery Configuration** Window, with the **User Identification** displayed by default.

2. **Select the General Defaults Tab**
   With the **NatQuery Configuration** window open, click the **General Defaults** tab.

   This action will display the **General Defaults** tab of the **NatQuery Configuration** window.

3. **Select or Enter the Editor of Choice**
   At the bottom of the **General Defaults** tab in a text box labeled as **External Editor**, and by default this will display the value of **NOTEPAD.EXE**.

   To change the Editor that is used by NatQuery to open an Audit report, then the User may manually enter the name of the executable, or may browse to it using the **Browse** button. If manually entered, then depending on the Environment Variables set up for Windows, specifically the Path variable, then the full path to the executable of the Editor will most likely be needed.

   If Microsoft Word is installed on the workstation, then the executable for Microsoft Word is typically WINWORD.EXE. However, in almost all cases the full path of WINWORD.EXE must be entered (unless the path to WINWORD.EXE is included in the environment variables for the

workstation).

4. **Close the NatQuery Configuration Window**
   With the path and executable selected, the User can now click the **OK** button to close the NatQuery Configuration window.

By performing the steps outlined above, when the User is presented with a prompt as to whether or not they wish to open the resulting Audit report in an Editor, the Editor specified above will be used.

Editing a Sequential DDM to Change the Default File Name

When the initial process that will convert raw PLOG data into usable information is first generated and the User requests that a DDM be generated that matches the final output, then NatQuery will generate the DDM as requested. In the "heading" (comment section) of the generated DDM, NatQuery will store the name of the output file that should be created when the PLOG process is run.

In this way, the DDM can be associated with the actual file name of output file created, thus allowing for that name to be made available to a User without the User needing to look the actual file name up or otherwise consult with the NatQuery Administrator.

In certain instances, the Administrator or User may wish to change the name of the sequential file that is associated with an existing Audit DDM.

In order to accomplish this, the User will first insure that there are no queries or other windows open on the NatQuery desktop. The User will then click the **Administer** drop down menu, and will then click on **Environment Configuration** -> **DDMs / FDTs** -> **Build/Edit DDM**. Performing those actions will invoke the Build / Edit DDM window which will look similar to **Figure 25**.

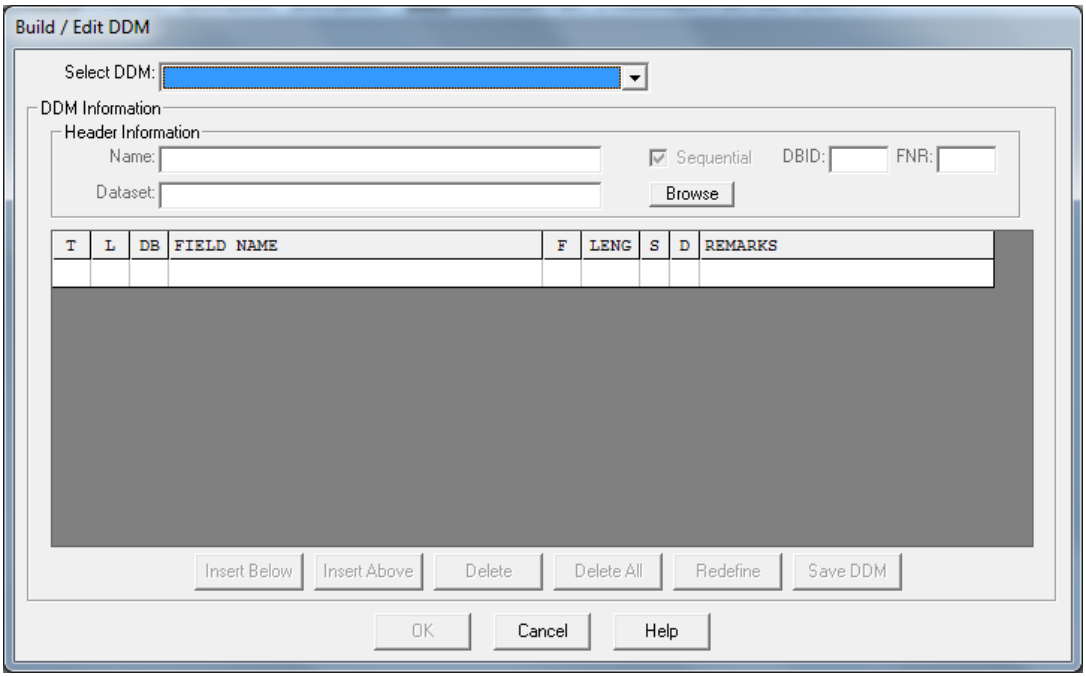

FIGURE 25

To change the sequential file name associate with a given sequential DDM, the User will first select the DDM of interest by using the **Select DDM** drop down list box.

With a sequential DDM selected, the **Build / Edit DDM** will be updated to look similar to what is seen in **Figure 26**.

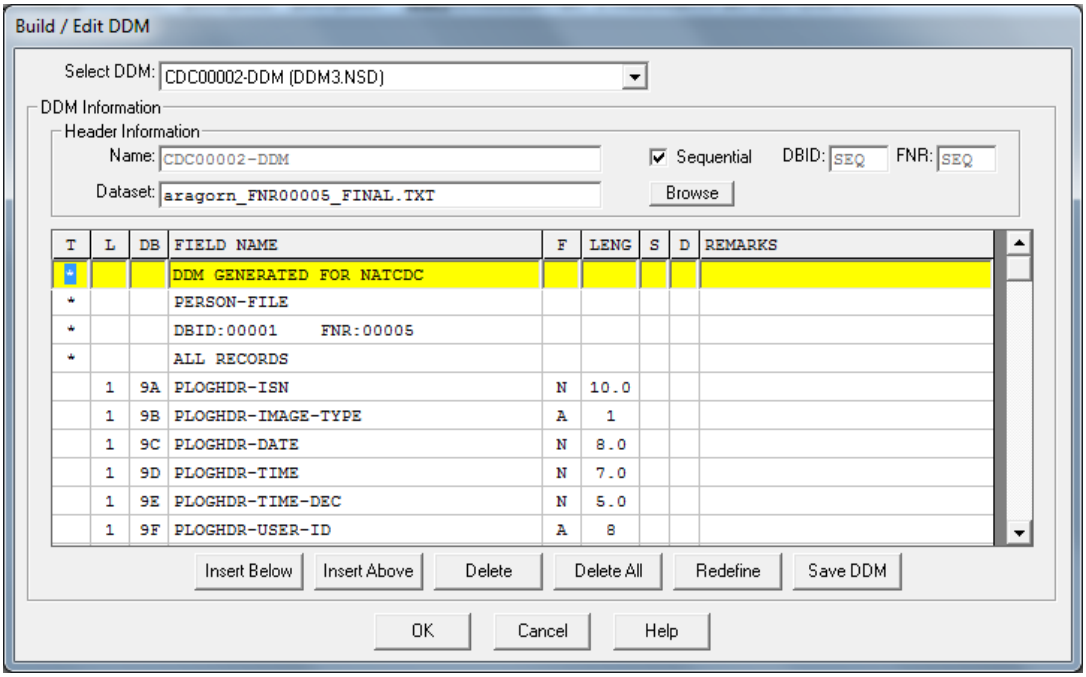

FIGURE 26

To change the name of the file associate by default with this DDM, the User will change the value of the field **Dataset** to be the new value, and will then click the **Save DDM** button to save the new file name.

Performing that action will then cause any new Audit Reports that are generated to automatically to default to using the new file name, however all existing Audit Report queries will continue to use the name of the file that was last accessed by the Audit Report process.

In some cases, the User can utilize the **Browse** button to locate the appropriate input file - but this is only in the case where the server's File System can be accessed from the NatQuery workstation - in all other cases the file name will have to be entered manually (or otherwise entered through a copy & paste operation).

APPENDIX A

The following is an example of the OVERALL AUDIT SUMMARY section of an Audit Summary Report.

```
                    NATCDC PROTECTION LOG AUDIT REPORT
                 FILE:PERSON-FILE   DBID:00001   FNR:00005
          DATE: 09/10/14                              PAGE:     12



     OVERALL AUDIT SUMMARY


             EARLIEST PLOG TIMESTAMP = 2014/08/14 01:05:16:4
                 LAST PLOG TIMESTAMP = 2014/08/14 23:47:16:2

                     TOTAL RECORDS READ =             75
                  TOTAL RECORDS ACCEPTED =             59

                   TOTAL UNIQUE ISNS READ =             56

                       TOTAL TRANSACTIONS =             65

                TOTAL AFTER IMAGE RECORDS =             61
               TOTAL BEFORE IMAGE RECORDS =             14

                         TOTAL STORES =                 51
                        TOTAL UPDATES =                 10
                        TOTAL DELETES =                  4


              AUDIT OPTIONS USED:
              ****************************************
                CREATE EXTRACT OF SELECTED RECORDS = FALSE
               IGNORE REPORTING STORE TRANSACTIONS = FALSE
              IGNORE REPORTING UPDATE TRANSACTIONS = FALSE
              IGNORE REPORTING DELETE TRANSACTIONS = FALSE


THE INPUT FILE FOR THIS RUN WAS CREATED BY NATCDC
```

The following is an example of the COUNT OF FIELD CHANGES BY TYPE OF TRANSACTION section of an Audit Summary Report.

```
                    NATCDC PROTECTION LOG AUDIT REPORT
                 FILE:PERSON-FILE   DBID:00001   FNR:00005
          DATE: 09/10/14                              PAGE:     13


  COUNT OF FIELD CHANGES BY TYPE OF TRANSACTION

  **********FIELD-NAME********** ***STORE*** ***UPDATE** ***DELETE**
  PERSON-ID                                51           2           4
  FIRST-NAME                               51           1           4
  LAST-NAME                                51           2           4
  QUALIFIER                                51                       4
  HOME-PHONE                               50                       4
  WORK-PHONE                               29                       1
  CELL-PHONE                               22           1           2
  FAX
  E-MAIL-ADDRESS                           49           1           3
  STREET                                   51                       4
  CITY                                     49           2           4
  STATE                                    49                       4
  ZIP                                      49                       4
  ZIP-PLUS-4                               19           1           3
  LAST-ACTIVITY-DATE                       51           1           4
  LAST-ACTIVITY-TYPE                       47           2           4
  STATUS                                   43           3           4
```

The following is an example of the COUNT OF FIELD CHANGES BY THE HOUR section of an Audit Summary Report.

```
                NATCDC PROTECTION LOG AUDIT REPORT
             FILE:PERSON-FILE   DBID:00001   FNR:00005
 DATE: 09/10/14                                PAGE:     14



COUNT OF TRANSACTIONS BY THE HOUR

****HOUR****  ***STORE***  ***UPDATE**  ***DELETE**

00:00-00:59
01:00-01:59
02:00-02:59
03:00-03:59
04:00-04:59
05:00-05:59
06:00-06:59
07:00-07:59
08:00-08:59
09:00-09:59
10:00-10:59            51           10            4
11:00-11:59
12:00-12:59
13:00-13:59
14:00-14:59
15:00-15:59
16:00-16:59
17:00-17:59
18:00-18:59
19:00-19:59
20:00-20:59
21:00-21:59
22:00-22:59
23:00-23:59
```

The following is an example of the SUMMARY OF SELECTION LOGIC section of an Audit Summary Report.

```
                NATCDC PROTECTION LOG AUDIT REPORT
             FILE:PERSON-FILE   DBID:00001   FNR:00005
 DATE: 09/10/14                                PAGE:     15



USER ENTERED SELECTION LOGIC FOR THIS RUN:

     FILE01.LAST-NAME Changed



TRANSACTIONS SELECTED WITH THIS LOGIC

     ***STORE***  ***UPDATE**  ***DELETE**
               51            2            4
```