

DPW Billing Information:

DPW Billing Database Structure

DPW_BILL_MSTR

RESOURCE_CODE	VARCHAR2(10)	Resource_code under which the services will be billed
ORDER_NO	VARCHAR2(6)	Labor charge code to which labor is charged for work request
FUND_ACCT_NO	VARCHAR2(8)	Customer funding account to bill services to
ORDERING_WI_CODE	VARCHAR2(6)	Customer work item code
LAST_POSTED_DATE	DATE	Last date that charges were pulled from CEFMS for labor on referenced charge code
REQ_EMP_ID_NO	VARCHAR2(9)	CEFMS employee id no of the person requesting services
STATUS_CODE	VARCHAR2(1)	Status code for the work order. A- Active C- Complete E- Exceeded I- Invalid S- Suspended X- Expired
SHOPS	VARCHAR2(30)	Text field for denoting which shop is to perform work
PRAC_NO	VARCHAR2(16)	Purchase Request to be billed for services
PRAC_LINE_NO	NUMBER(4)	Line item of purchase request to be billed for services

DPW_ORDER_HISTORY

ORDER_NO	VARCHAR2(6)	Labor charge code for work order
DATE_ENTERED	DATE	Date the amendment information was entered
SEQ_NO	NUMBER(4)	Amendment number; 0 amendment is when the work order is initially entered into the system.
ORDER_ESTIMATE	NUMBER(14,2)	Change in the estimate; To get the total estimate all amendments must be summed
STATUS_CODE	VARCHAR2(1)	Status code for the work order at the time of the amendment
ORDERING_WI_CODE	VARCHAR2(6)	Current work item at the time of the amendment
REQ_EMP_ID_NO	VARCHAR2(9)	Current requesting employee at the time of the amendment
FUND_ACCT_NO	VARCHAR2(8)	Current funding account at the time of the amendment. Once charges have accumulated in the system for a work order, this value cannot be changed.
SOURCE	VARCHAR2(8)	User ID or process which caused the amendment
SHOPS	VARCHAR2(30)	Current shop information
PRAC_NO	VARCHAR2(16)	Current purchase request for billing
PRAC_LINE_NO	NUMBER(4)	Current purchase request for billing

DPW_BILL_TRANS_REG

ORDER_NO	VARCHAR2(6)	Labor charge code
RESOURCE_CODE	VARCHAR2(10)	Resource code for billing services performed
SOURCE	VARCHAR2(8)	Process which created the transaction
EFFECT_DATE	DATE	Date the transaction was entered into the system
HRS	NUMBER(14,2)	Number of hours worked on charge code
CHARGE_AMT	NUMBER(14,2)	Amount charge to customer based on number of hours

)	worked multiplied by the rate for the resource code under which the charges are to be billed.
POSTED_FLAG	VARCHAR2(1)	Current status of the transaction. Y – has been sent to CEFMS for payment. N – has not been sent to CEFMS for payment. E – exceeds the estimate, so cannot be sent to CEFMS for payment. L - transaction has been marked as a loss

DPW_BILLING_HISTORY

ORDER_NO	VARCHAR2(6)	Labor charge code
PRAC_NO	VARCHAR2(16)	Purchase request to which the transaction was billed
PRAC_LINE_NO	NUMBER(4)	Line item of the purchase request billed
CHARGE_AMT	NUMBER(14,2)	Amount of transaction
HRS	NUMBER(14,2)	Number of hours for the transaction
POSTED_DATE	DATE	Date the transaction was entered into the system
FUND_ACCT_NO	VARCHAR2(8)	Funding account billed for transaction
RESOURCE_CODE	VARCHAR2(10)	Resource code under which the transaction was billed
SOURCE	VARCHAR2(8)	Process which created the transaction
BILL_DATE	DATE	Date the transaction was sent to CEFMS for payment

DPW_BILLING

CHARGE_AMT	NUMBER(14,2)	Amount sent to CEFMS. This may include more than one billing transaction.
FILE_EXTRACT_FLAG	VARCHAR2(1)	Flag to denote whether or not the transaction has been sent to CEFMS
FUND_ACCT_NO	VARCHAR2(8)	Funding account billed
PRAC_NO	VARCHAR2(16)	Purchase request billed
PRAC_LINE_NO	NUMBER(4)	Line item of purchase request
RESOURCE_CODE	VARCHAR2(10)	Resource code under which the services were billed
STATUS_FLAG	VARCHAR2(1)	Flag to denote whether or not the transaction has been retained in the billing system.
TRANS_DATE	DATE	Date the transaction was sent to CEFMS for payment

DPW_BILL_TRANSFER

TO_CHG_CODE	VARCHAR2(6)	The work order the charges were moved to
FROM_CHG_CODE	VARCHAR2(6)	The work order the charges were moved from
EFFECT_DATE	DATE	The minimum effect_date of the charges moved
TRANS_DATE	DATE	The date the transfer was performed
HRS	NUMBER(14,2)	The number of hours moved by transfer

DPW_LABOR_DETAIL

ORDER_NO	VARCHAR2(6)	Labor charge code
RESOURCE_CODE	VARCHAR2(10)	Resource code under which the services are billed
EMP_ID_NO	VARCHAR2(9)	CEFMS employee id for individual charging to the labor charge code for services performed
FROM_TO_IND	VARCHAR2(1)	Indicated when a charge is part of a labor cost transfer F- From charge code T- To charge code null – Not a part of a labor cost transfer
LABOR_HOURS_SEQ_ID	NUMBER(9)	CEFMS sequence id used for identifying labor transactions

SOURCE	VARCHAR2(8)	Process which created the transaction
WRK_DATE	DATE	Date the service was performed
POSTED_DATE	DATE	Date the transaction was entered into the billing system
BILL_DATE	DATE	Date the transaction was sent to CEFMS for payment
HRS	NUMBER(14,2)	Number of hours worked by employee in performing service
CHARGE_AMT	NUMBER(14,2)	Amount to the charged for service performed
POSTED_FLAG	VARCHAR2(1)	Status of the transaction. Same as in dpw_bill_trans_reg
EFFECT_DATE	DATE	Date the labor transaction was processed in CEFMS as a part of labor distribution process
BILLED_ORDER_NO	VARCHAR2(6)	Work order under which the charge was paid. This is changed when a billing transfer is performed.

DPW_RATES

HRS_RATE	NUMBER(14,2)	Hourly rate for services performed
RESOURCE_CODE	VARCHAR2(10)	Resource code to which the hourly rate is tied
EFFECT_DATE	DATE	Date which the rate took effect

DPW_BILL_PRC

FUND_ACCT_NO	VARCHAR2(8)	Funding account for purchase request
PRAC_NO	VARCHAR2(16)	purchase request
PRAC_LINE_NO	NUMBER(4)	line item for purchase request
RESOURCE_CODE	VARCHAR2(10)	resource code for the purchase request

STATUS

STATUS_CODE	VARCHAR2(1)	Work order status code A- ACTIVE C- COMPLETE S- SUSPENDED E- EXCEEDED X- EXPIRED I- INVALID
STATUS_NAME	VARCHAR2(20)	Work order status description
HIDDEN	VARCHAR2(1)	This flag is used to determine whether a user can change a status N – The status field is not hidden, so the user can change the status Y- The status field is hidden, so the user can not change the status

DPW Billing Procedure

The DPW billing procedure is a stored procedure called dpw_billing_prod, which is stored in a file called dpw_bill_prod.sql. This procedure is execute as a part of the run_dpw_bill.sql script which is executed under the dpw_bill sid, U4DPW9P1. Listed below is an outline of the steps performed by the billing procedure.

- 1) Check for work order against funds that have expired. Change the status of these work orders to 'X' for expired.
- 2) Check for work orders where the funds are no longer expired. Change the status of these work orders back to the status before the funds expired.
- 3) Check for work orders that are no longer exceeded. Change the status of these work orders to 'A' for active. Also, change the posted_flag of the exceeded transactions to 'N' for not paid.

- 4) Insert records into dpw_labor_detail for the new labor. New labor is determined by the charge codes in dpw_bill_mstr and their associated last_posted_date. Any labor transactions with an effect_date greater than or equal to the last_posted_date and less than the sysdate will be inserted into the dpw_labor_detail table. The labor is inserted using 3 queries: one for direct charged labor transactions (LABDIST), one for labor cost transfers made to a work order (LBRTRAN), and one for labor cost transfers made from a work order (LBRTRAN).
- 5) Update the hours for the new records inserted into dpw_labor_detail. This update is done based on the charge_code and labor_hours_seq_id.
- 6) Generate the records for dpw_bill_trans_reg which summarize the transactions in dpw_labor_detail by charge_code and for the current date.
- 7) Update the new records in dpw_bill_trans_reg with the charge_amt.
- 8) Update the last_posted_date in dpw_bill_mstr for those work orders which had records inserted into dpw_bill_trans_reg.
- 9) Create dpw_bill_prc table for current billing process. The records are deleted from the table for the previous billing process and then inserted based on billing information in dpw_bill_mstr for those work orders which have unbilled transactions in dpw_bill_trans_reg (posted_flag = 'N'). Only those purchase requests which have a balance and are associated with a work order which has unbilled cost will be inserted into the table.
- 10) Check for work orders where the transactions added for the current billing process have caused the estimate to be exceeded. A transaction in the current billing process may need to be split into the portion that can be billed and the portion that exceeds the estimate. The transactions in the detail table are also split in case the exceeded cost is transferred to another work order. Finally, the work order is marked exceeded.
- 11) Create billing records in dpw_billing for those work orders that have unbilled cost and a purchase request record in dpw_bill_prc, which shows that we have funds available to bill against.
- 12) Update the posted_flag to 'R' for the dpw_bill_trans_reg records that are summarized in the dpw_billing records for the current date. This is done so that the records are easily identified for inserting information into dpw_billing_history and updating records in dpw_labor_detail.
- 13) Insert records into dpw_billing_history where the posted_flag equals 'R' in dpw_bill_trans_reg. The funding account and purchase request information for the new records will come from dpw_bill_mstr.
- 14) Update records in dpw_labor_detail where the order_no, source, and posted_date match a record in dpw_bill_trans_reg where the posted_flag equals 'R'.
- 15) Update dpw_bill_trans_reg to set the posted_flag equal to 'Y' where the posted_flag equals 'R'.

DPW Billing Transfers

The work order clerks may request billing transfers by e-mail. The billing transfer process is performed by a stored procedure, dpw_billing_transfer, which is stored in the file dpw_transfer.sql. The dpw_billing_transfer procedure requires two work orders as input. The following steps are used to perform a billing transfer.

- 1) Connect to the dpw_billing database, U4DPW9P1.
- 2) Check for Deborah Williams running billing process. This must be done to insure that there are no problems with the transfer caused by billing process updates incorrectly updating the cost of the transfer. To check for users connected to the database, use the following query: select username from v\$session; If the userid u4rmsddw is shown, don't run the transfer process without having contacted Deborah to make sure billing is complete.
- 3) Since the stored procedure uses DBMS_OUTPUT.PUTLINE to print messages, you will need to *set serveroutput on*.

- 4) *execute dpw_billing_transfer('from work order', 'to work order');* The work orders will be listed in the message from the work order clerk.
- 5) *select * from dpw_bill_trans_Reg where effect_date = 'current date';* This will show the transactions performed on the current date. The billing transfer transactions will be displayed last. Check the transactions to make sure that the hours and cost transferred agree.
- 6) Using the information from the query in the previous set, respond to the mail message and let the work order clerk know that the transfer has been completed. It is a good idea to let the work order clerk know the number of hours and the cost transferred in case there is any problem later. Also, let the work order clerk know if the transfer has caused the work order where the cost was moved to exceed its estimate.

DPW Billing Loss

When the work order clerks request that cost against a work order be marked a loss, the `dpw_billing_loss` stored procedure is used. This procedure marks all unpaid charges in the billing system a loss. Once the transactions are marked a loss, the status of the work order is changed to 'C', complete. This is done to prevent any further activity on the work order. To run the stored procedure, use the following command:

```
execute dpw_billing_loss('work order no');
```

This procedure should not be run while the billing procedure is running, so use the information in step 2 of the previous section before attempting to execute the procedure.

DPW Billing User Interface

The user interface for `dpw_billing` consists of six stored procedures: `dpw_billing_screen`, `dpw_billing_save`, `dpw_billing_print`, `view_wicode_info`, `view_emp_info`, and `view_prac_info`. These procedures work together to make the user interface which allows for entry and update of customer information on work orders. The `dpw_billing_screen` is the main procedure for the user interface. This procedure displays that input screen which has view buttons for look up information. The view buttons call the various view procedures to look up information based on the data in the `dpw_billing_screen`. The save button is tied to the `dpw_billing_save` procedure which validates the information entered before saving it to the database. Once the save is complete, the `dpw_billing_print` procedure is called by `dpw_billing_save` and a work order form is displayed in the browser for printing. The sections below will discuss some of the more complicated pieces in detail. The URL for this set of procedures is http://erd30.erd.usace.army.mil:2048/dpw/plsql/dpw_billing_screen.

DPW Billing Screen

The `dpw_billing_screen` is the main procedure for the user interface which allows users to input new work orders as well as update existing work orders. The procedure holds the code for two of the browser screens. The decision of which screen is to be displayed is made by checking to see if the variable which holds the order number is null. Listed below is an outline of the logic within `dpw_bill_screen.sql`

- 1) Try to get the last posted date for the work order. If the work order is null the query will have an error, so an exception stmt is added to set the last posted date to null in this case.
- 2) Set up the javascript to be used by the browser when one of the "View" buttons is pressed.
- 3) Check the contents of the variable which contains the order no.
- 4) If the order number is null display a screen that has an input box for the order number and a button to press once the order number has been filled in. This button causes the `dpw_billing_screen` procedure to be called again with the `psOrderNo` variable containing the information typed into the input box.
- 5) If the order number is not null, proceed with checks to make sure that the order number is a valid charge code in CEFMS. If it is a valid charge code in CEFMS, then the procedure

- is able to get the purchase request information for display. The information selected is the description, certified amount of the labor purchase request, balance of the labor purchase request, purchase request number, and purchase request line number. In the case of the order number not existing in CEFMS, an error message is displayed letting the user know that no data was found for the charge code.
- 6) Try to get the status code of the work order. If the work order selected already has a record in the billing system, get its current status code from dpw_bill_mstr. If the work order does not have record in the billing system, the status is automatically set to 'A', or active, by the exception clause.
 - 7) If the status code is 'C', or complete, the work order may not be edited. In this case the user is sent to the work order form print screen using the dpw_billing_print call.
 - 8) Try to get the amount of charges that have been paid for the work order. This is done with a query on dpw_bill_trans_reg where records where the posted_flag equals 'Y'. If there are no records for the work order where the posted_flag equals 'Y', the variable is set to zero.
 - 9) Try to get the amount of charges that are unpaid for the work order. This is done with a query on dpw_bill_trans_reg where records where the posted_flag equals 'N'. If there are no records for the work order where the posted_flag equals 'N', the variable is set to zero.
 - 10) Try to get the amount of charges that exceed the estimate for the work order. This is done with a query on dpw_bill_trans_reg where records where the posted_flag equals 'E'. If there are no records for the work order where the posted_flag equals 'E', the variable is set to zero.
 - 11) Try to get the amount of charges that have been marked a loss for the work order. This is done with a query on dpw_bill_trans_reg where the posted_flag = 'L'. If there are no records for the work order where the posted_flag = 'L', the variable is set to zero.
 - 12) Display the header for the screen including the current date, the last posted date of the work order, and a link to get back to the screen where you enter the work order number.
 - 13) Set up the information for the "Save" button, which will be placed on the form later. The information lets the browser know that when the button is pressed that it should call the dpw_billing_save procedure.
 - 14) Create a table that will contain all the display information, input boxes, and view buttons. This is done so that the browser will neatly organization the information on the screen.
 - 15) Display the order number and have a hidden variable to hold the information for use by other procedures called from this one.
 - 16) Display the labor purchase request and purchase request line number.
 - 17) If the psAction variable is set to 'E' then an error occurred while trying to save, so the dpw_billing_save procedure has sent the user back to the dpw_billing_screen to correct the error. This is done using a link in the dpw_billing_save procedure. When an error has occurred, the screen should display the form with the information that the user had entered. This is done using the if psAction = 'E' clause within the code.
 - 18) If the psAction is not 'E' then the screen will attempt to look up the information on the work order in the system. The information retrieved includes the funding account number, customer's purchase requested number, customer's purchase request line number, ordering work item code, resource code, requesting employee's CEFMS employee id, work order status code, the work order estimate, and the shops description. If the work order is not in the system, and exception clause is used to set the variables for this information to null. The only piece of information that would not be null is the resource code. The resource code is determined by a decode on the purchase request's labor_receive_org_code.
 - 19) If the status code is still null after the look up is complete, the status code is set to active since all new work orders will start with a status of active.
 - 20) The status table maintains the various possible status code, their descriptions, and whether or not the user can change the status. The status name for the current status is pulled from the status table.

- 21) Check dpw_bill_trans_reg for any transactions. If there are any transactions in dpw_bill_trans_reg, the user may not be change the customer information, which includes the funding account, purchase request, and purchase request line item. A View Prac Info button is displayed beside the funding account info. This button executes the view_prac_info procedure to look up purchase requests based on the funding account and resource code of the work order.
- 22) Display the resource code and set up a hidden variable to pass the resource code on to the other procedures called by dpw_billing_screen.
- 23) Display a text field for the shops information.
- 24) Display the input box for the ordering work item. This is mainly a reference field in case the job requires materials purchases on the customer's funding. There is also a "View funding Info" button which calls the view_wicode_info procedure using the work item code to look for any funding accounts referencing that work item.
- 25) Display an input box for the requesting employee's CEFMS employee id. There is also a "View Emp Info" button on this line. This button calls the view_emp_info procedure to look up any employees, which have a CEFMS employee id beginning with the input for the request employee id.
- 26) Display an input box for the work order estimate.
- 27) Display the certified amount of the labor purchase request that the labor charge code is tied to.
- 28) Display the total of charges paid by the customer for services performed under the work order.
- 29) Display the charges against the work order which have been processed by the billing program, but are left unpaid in the dpw_bill_trans_Reg table because the customer's purchase request does not have an available balance.
- 30) Display the charges against the work order that exceed the work order's estimate.
- 31) Display the amount that has been marked a loss for the work order.
- 32) Display the description for the labor purchase request, which should be a description of work to be performed.
- 33) Display the status of the work order. If the status is changeable by the user, the status will be a pick list of the statuses available to the user. If the status is not changeable, the current status will be displayed and a hidden variable will be set up to hold the current status for use by other procedures called from dpw_billing_screen.
- 34) Display a "Save" button and an "Undo" button. The save button calls the dpw_billing_save procedure with the current information from dpw_billing_screen. The undo button resets the information in the input boxes back to what they were when the screen was originally displayed after the work order number was selected.

DPW Billing Save

The dpw_billing_save procedure source code is in the dpw_bill_save.sql file. This procedure validates the information entered by the user and saves if appropriate. Listed below is an outline of the source code for the procedure.

- 1) Define a link so that in the case of an error the user can be sent back to dpw_billing_screen with the information that they had entered before saving. A part of the link is the variable psAction, which lets dpw_billing_screen know that it should display the information that was passed from dpw_billing_save.
- 2) Check each of the variables passed from dpw_bill_screen to make sure that the user entered something in each. This is done by checking the length of the variable. If the length is zero then the value for the variable is null. This method can be used because all the information is passed as varchar2. The only exception to this is the line number. In that case the nvl() function is used to check for a null value.
- 3) Check to see if the purchase request and line number given by the customer exist in CEFMS. If no records are found in CEFMS, the user receives a message letting them know the purchase request does not exist and that they may return to the previous screen to correct the information.

- 4) Check to see if the purchase request is tied to the funding account that was given. If this is not the case, the user is sent back to correct the information.
- 5) Check to see if the resource code of the purchase request matches the resource code of the work order. If the resource code does not match, the charges can not be billed so the user must go back and choose a purchase request with the appropriate resource code.
- 6) Check to see if the work item is a valid work item in CEFMS.
- 7) Check to see if the requesting employee id is valid in CEFMS.
- 8) Check to make sure that the order estimate is numeric.
- 9) Check to see if the work order already has a record in dpw_bill_mstr. If it does not then a new record will be created in dpw_bill_mstr. A record is inserted into dpw_order_history to show the initial information of the work order.
- 10) If the work order already exists, select the new sequence number for this save into the variable tiNewSeq. This is done by selecting the maximum seq_no for the work order and adding one to it.
- 11) Select the current total for the estimate. This done by summing the order_estimate field for all records associated with the work order.
- 12) Determine the difference in the value entered by the user for the estimate and the old estimate. This is the amount that will be saved in the order_estimate field for the new record in dpw_order_history.
- 13) Select the sum of the paid charges for the work order. This is saved in the variable tnBillCharges.
- 14) Select the sum of the unpaid charges that fall within the old estimate. This is saved in the variable tnUpstedCharges.
- 15) Place the total amount of the new estimate in tnEstimate
- 16) Save the amount of the estimate less the paid charges in the tnRemainEst variable.
- 17) Save the amount of the estimate less all charges within the estimate (posted_flag equal N or Y) in the variable tnEstBal
- 18) If the status for work order is exceeded, check to see if the estimate has been increase or if the reimaining balance is positive or if the status for the work order is active and the remaining balance of the new estimate is negative, the records in dpw_bill_trans_reg need to be gone through to split transactions if necessary and to mark the charges that exceed the new estimate.
- 19) Once the records in dpw_bill_trans_reg have been done, do the same process for the records in dpw_labor_detail. If the current status is active, change that status to exceeded.
- 20) Update the dpw_bill_mstr record with the new information and insert a new record into dpw_order_history.
- 21) Call the dpw_billing_print procedure to display the work order form for printing.