



pears.DuplicateVacancyAndPlacement



Generated schema reference. Regenerate this page from the SQL unload; keep hand-maintained business notes in the narrative namespace.

Original SQL

```
COMMENT TO PRESERVE FORMAT ON PROCEDURE
"pears"."DuplicateVacancyAndPlacement" IS
{CREATE FUNCTION DuplicateVacancyAndPlacement

/* Application Maintained Function / Procedure - DO NOT EDIT*/

(IN @OldVacancyID CHAR(20),IN @NewVacancyID CHAR(20),IN @NewEmploymentID
CHAR(20),IN @OldPlacementID CHAR(20) DEFAULT NULL,IN @NewPlacementID
CHAR(20) DEFAULT NULL,IN @CloseOffOld CHAR(1) DEFAULT 'N')
RETURNS SMALLINT
BEGIN
    DECLARE @VacancyAllowed CHAR(250);
    DECLARE @OldCandEmploymentID CHAR(20);
    DECLARE @NewCandEmploymentID CHAR(20);
    DECLARE @NewCandEmploymentCompanyID CHAR(20);
    DECLARE @ExplainStringFrom CHAR(250);
    DECLARE @ExplainStringTo CHAR(250);
    DECLARE @MovedPerson CHAR(50);
    --Check new vacancy allowed
    SET @VacancyAllowed=(SELECT vacancyallowed(@NewEmploymentID));
    IF @VacancyAllowed <> '' THEN RETURN(1)
    END IF;
    BEGIN atomic
        INSERT INTO vacancy(
VacancyID,DepartmentID,EmploymentID,STATUS,temp,POSITION,ContractRef,
ClientDepartment,NoOfPosts,StartDate,FinishDate,TempDeskID,StaffID,SiteName,
SiteContact,
addr1,addr2,addr3,town,county,country,postcode,SitePhoneNumbers,Notes,OtherN
otes,
ErNi,HolidayAllowance,Discount,Salary,Expiry,TheirRef,Currency,PayrollIdenti
fier,
TempJobTypeID,InvAddr1,InvAddr2,InvAddr3,InvTown,InvCounty,InvCountry,InvPos
tCode,
ClassCode,AnalysisCode,EntryDate,ROLE,CascadeDateTime,WorkMonday,WorkTuesday
,WorkWednesday,WorkThursday,WorkFriday,WorkSaturday,WorkSunday,WorkStartTime
```



```
,WorkNormalHours)
    SELECT
@NewVacancyID,DepartmentID,@NewEmploymentID,STATUS,temp,POSITION,ContractRef
,
ClientDepartment,NoOfPosts,StartDate,FinishDate,TempDeskID,StaffID,SiteName,
SiteContact,
addr1,addr2,addr3,town,county,country,postcode,SitePhoneNumbers,Notes,OtherN
otes,
ErNi,HolidayAllowance,Discount,Salary,Expiry,TheirRef,Currency,PayrollIdenti
fier,
TempJobTypeID,InvAddr1,InvAddr2,InvAddr3,InvTown,InvCounty,InvCountry,InvPos
tCode,
    ClassCode,AnalysisCode,CURRENT
DATE,ROLE,CascadeDateTime,WorkMonday,WorkTuesday,WorkWednesday,WorkThursday,
WorkFriday,WorkSaturday,WorkSunday,WorkStartTime,WorkNormalHours FROM
    vacancy WHERE vacancyid = @OldVacancyID; //NB Vacancy refcode
inserted BY TRIGGER IN Vacancy TABLE
-- Copy questionnaire
INSERT INTO tagvalue( taglocation,tagid,tagchoiceid,id,VALUE,textvalue)
    SELECT taglocation,tagid,tagchoiceid,@NewVacancyID,VALUE,textvalue
FROM tagvalue WHERE
    id = @OldVacancyID AND taglocation LIKE 'V%';
-- Copy requirements
INSERT INTO criterion( searchlocation,
    id,critid,sourcelocation,dictionaryid,
tagid,tagchoiceid,notflag,matchtype,textvalue,VALUE,uppervalue,taganytotal,
    extravalue,extrauppervalue,extramatchtype)
    SELECT
searchlocation,@NewVacancyID,critid,sourcelocation,dictionaryid,
tagid,tagchoiceid,notflag,matchtype,textvalue,VALUE,uppervalue,taganytotal,
    extravalue,extrauppervalue,extramatchtype FROM criterion WHERE
    id = @OldVacancyID AND searchlocation LIKE 'V%';
-- Copy rates
INSERT INTO TempJobRate(
TempJobRateID,VacancyID,TempPayBandID,PayRate,ChargeRate,StartDate,EndDate,G
rade)
    SELECT(SELECT
uniquekey(TempJobRateID)),@NewVacancyID,TempPayBandID,PayRate,ChargeRate,Sta
rtDate,EndDate,Grade FROM
    TempJobRate WHERE vacancyid = @OldVacancyID;
-- Copy T & A rules
INSERT INTO cardreaderoverridesettings(
CardReaderSettingsID,Companyid,defaultshiftlength,MachineID,maxbreaklength,M
axEarliness,MaxExtraBreakTime,MaxLateness,maxshiftlength,MinBeforeExtending,
MinBeforeExtendStart,MinBreakLength,VacancyID)
    SELECT
uniquekey(VacancyID),Companyid,defaultshiftlength,MachineID,maxbreaklength,M
axEarliness,MaxExtraBreakTime,MaxLateness,maxshiftlength,MinBeforeExtending,
```



```
MinBeforeExtendStart,MinBreakLength,@NewVacancyID FROM
cardreaderoverridesettings WHERE
    vacancyid = @OldVacancyID;
-- Cascade Rules
INSERT INTO CascadeRule(
CascadeRuleID,CompanyID,VacancyID,AgencyStoredSelectionID,CascadeHours,Hours
From,CascadeLevel,CustomFunction,VacancyRule)
SELECT
uniquekey(CascadeRuleID),CompanyID,@NewVacancyID,AgencyStoredSelectionID,Cas
cadeHours,HoursFrom,CascadeLevel,CustomFunction,VacancyRule FROM
    CascadeRule WHERE vacancyid = @OldVacancyID;
-- PLACEMENT
IF @OldPlacementID IS NOT NULL AND @NewPlacementID IS NOT NULL THEN
-- Candidates Employment
SET @OldCandEmploymentID=(SELECT EmploymentID FROM Placement WHERE
PlacementID = @OldPlacementID);
SET @NewCandEmploymentID=uniquekey(@OldCandEmploymentID);
SET @NewCandEmploymentCompanyID=(SELECT CompanyID FROM Employment
WHERE EmploymentID = @NewEmploymentID);
INSERT INTO Employment(
EmploymentID,temp,personid,companyid,startdate,leavedate,noreemploy,note,sal
ary,POSITION,department,Concurrent,ExtendedNotes)
SELECT
@NewCandEmploymentID,temp,personid,@NewCandEmploymentCompanyID,startdate,lea
vedate,noreemploy,note,salary,POSITION,department,Concurrent,ExtendedNotes
FROM
    Employment WHERE EmploymentID = @OldCandEmploymentID;
-- Placement
INSERT INTO placement(
placementid,placedate,employmentid,vacancyid,departmentid,staffid,temp,clien
trate,temprate,note,State,RefCode,TheirRef,ContractRef,DaysPerWeek,
TransferBatch,Currency,TempJobTypeID,ExtendedNotes,ExpenseBenefitSchemeID,Wo
rkMonday,WorkTuesday,WorkWednesday,WorkThursday,WorkFriday,WorkSaturday,Work
Sunday,WorkNormalHours,WorkStartTime,documenttemplateid)
SELECT
@NewPlacementID,placedate,@NewCandEmploymentID,@NewVacancyID,departmentid,st
affid,temp,clientrate,temprate,note,State,RefCode,TheirRef,ContractRef,DaysP
erWeek,
TransferBatch,Currency,TempJobTypeID,ExtendedNotes,ExpenseBenefitSchemeID,Wo
rkMonday,WorkTuesday,WorkWednesday,WorkThursday,WorkFriday,WorkSaturday,Work
Sunday,WorkNormalHours,WorkStartTime,documenttemplateid FROM
    Placement WHERE PlacementID = @OldPlacementID;
-- Placement Rates
INSERT INTO TempJobRate(
TempJobRateID,PlacementID,TempPayBandID,PayRate,ChargeRate,StartDate,EndDate
,Grade)
SELECT(SELECT
uniquekey(TempJobRateID)),@NewPlacementID,TempPayBandID,PayRate,ChargeRate,S
```



```
tartDate,EndDate,Grade FROM
    TempJobRate WHERE PlacementID = @OldPlacementID;
-- Placement Elements
INSERT INTO PlacementElement(
PlacementElementTypeID,PlacementID,VALUE,Volatile)
    SELECT UniqueKey(PlacementElementTypeID),PlacementID,VALUE,Volatile
FROM
    PlacementElement WHERE PlacementID = @OldPlacementID;
-- Descriptive Text
SET @ExplainStringFrom=(SELECT string(Company.Name,' - ',Person.Name,'
- ',Vacancy.RefCode) FROM Vacancy KEY JOIN Employment KEY
JOIN(Person,Company) WHERE Vacancy.VacancyID = @OldVacancyID);
SET @ExplainStringTo=(SELECT string(Company.Name,' - ',Person.Name,' -
',Vacancy.RefCode) FROM Vacancy KEY JOIN Employment KEY JOIN(Person,Company)
WHERE Vacancy.VacancyID = @NewVacancyID);
SET @MovedPerson=(SELECT Person.Name FROM Person KEY JOIN Employment
KEY JOIN Placement WHERE PlacementID = @OldPlacementID);
-- Close off old Placement
IF @CloseOffOld = 'Y' THEN
    UPDATE Employment SET LeaveDate = StartDate,ExtendedNotes =
TRIM(string('Employment cancelled, transferred to
',@ExplainStringTo,"char"(13),"char"(13),ExtendedNotes)) WHERE
    EmploymentID = @OldCandEmploymentID;
    UPDATE Placement SET ExtendedNotes = TRIM(string('Placement
cancelled, transferred to
',@ExplainStringTo,"char"(13),"char"(13),ExtendedNotes)) WHERE
    PlacementID = @OldPlacementID
END IF;
-- Add explanatory text to new Employment & Placement
UPDATE Employment SET ExtendedNotes = TRIM(string('Employment
transferred from ',@ExplainStringFrom,"char"(13),"char"(13),ExtendedNotes))
WHERE
    EmploymentID = @NewCandEmploymentID;
    UPDATE Placement SET ExtendedNotes = TRIM(string('Placement transferred
from ',@ExplainStringFrom,"char"(13),"char"(13),ExtendedNotes)) WHERE
    PlacementID = @NewPlacementID;
-- Update Audit Trail
CALL AuditLog('VACANCY',@OldVacancyID,'Placement Cancelled and
moved',string(@ExplainStringFrom,' - ',@MovedPerson),@ExplainStringTo);
CALL AuditLog('VACANCY',@NewVacancyID,'Copied
Vacancy',@ExplainStringFrom,@ExplainStringTo);
CALL AuditLog('PLACEMENT',@OldPlacementID,'Placement Cancelled and
moved',@ExplainStringFrom,@ExplainStringTo);
CALL AuditLog('PLACEMENT',@NewPlacementID,'Moved
Placement',@ExplainStringFrom,@ExplainStringTo)
END IF
// END OF placement SECTION
END; // END OF atomic statment
```



```
RETURN(0)
exception
  WHEN others THEN RETURN(2)
END
}
```

From:

<https://iqxusers.co.uk/iqxhelp/> - iqx

Permanent link:

https://iqxusers.co.uk/iqxhelp/doku.php?id=database:procedures:pears_duplicatevacancyandplacement

Last update: 2026/08/07 19:24

