# pears.AutoMatchProcessShift

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

## Original SQL

```
CREATE PROCEDURE "pears"."AutoMatchProcessShift"(
  /* Application Maintained Function / Procedure - DO NOT EDIT*/
  IN @TempShiftPlanID CHAR(20) )
BEGIN
  DECLARE @Explic SMALLINT;
  DECLARE @TempDeskID CHAR(20);
  DECLARE @VacancyID CHAR(20);
  DECLARE @CompanyID CHAR(20);
  DECLARE @ShiftDate DATE;
  DECLARE @OldWorkers INTEGER;
  DECLARE @PercentNew INTEGER;
  DECLARE @NewWorkers INTEGER;
  DECLARE @LeadTime INTEGER;
  DECLARE @DivID CHAR(20);
  DECLARE @Urgency SMALLINT;
  DECLARE @Email SMALLINT;
  DECLARE @SMS SMALLINT;
  DECLARE @Push SMALLINT;
  SET @Explic = 1;
  SELECT
"v"."TempDeskID","v"."VacancyID","e"."CompanyID","c"."DivisionID","p"."Shift
Date","datediff"("hour",CURRENT
TIMESTAMP,"p"."ShiftDate"+"p"."TimeFrom"),"c"."AutoMatchUrgency"
    INTO
@TempDeskID,@VacancyID,@CompanyID,@DivID,@ShiftDate,@LeadTime,@Urgency
    FROM "TempShiftPlan" AS "p" KEY JOIN "Vacancy" AS "v" KEY JOIN
"Employment" AS "e" KEY JOIN "Company" AS "c" WHERE "TempshiftPlanID" =
@TempShiftPlanID;
  SELECT FIRST "QtyWorkers","PercentNew","Email","SMS","Push" INTO
@OldWorkers,@PercentNew,@Email,@SMS,@Push FROM "AutoMatchDivisionConfig"
WHERE("divisionid" = @DivID OR "divisionid" IS NULL)
    AND("urgency" = @Urgency OR "urgency" IS NULL) AND(SELECT
"min"("Leadtime") FROM "AutoMatchDivisionConfig" AS "a" WHERE("urgency" =
@Urgency OR "urgency" IS NULL)
      AND("divisionid" = @DivID OR "divisionid" IS NULL) AND "leadtime" >
```

```
@LeadTime) = "leadtime" ORDER BY "divisionid" DESC;
  SET @NewWorkers = "round"(@OldWorkers*@PercentNew/100,0);
  SET @OldWorkers = (@OldWorkers-@NewWorkers);
  -- put shift plan in temp table
  INSERT INTO "tempmatchplan"(
"tempmatchplanid","vacancyid","tempshiftplanid","shiftdate" ) (
    SELECT
"uniquekey"("p"."tempshiftplanid"),@VacancyID,"p"."tempshiftplanid","p"."shi
ftdate" AS "thedate"
      FROM "tempshiftplan" AS "p" WHERE "p"."TempShiftPlanID" =
@TempShiftPlanID);
  -- put possible temps in temp table
  INSERT INTO "tempmatchcandidate"(
"tempmatchcandidateid","personid","matchscore","vacancyid","highlight" )
    SELECT "uniquekey"("t"."personid"),
      "t"."personid",0,@VacancyID,
      ((SELECT FIRST 1 FROM "employment" WHERE "companyid" = @CompanyID AND
"personid" = "person"."personid")
      +"isnull"((SELECT FIRST 1 FROM "placement" KEY JOIN "employment" WHERE
"employment"."companyid" = @CompanyID
        AND "employment"."personid" = "person"."personid" AND
"placement"."vacancyid" = @VacancyID),0)) -- use custom function (look for
existing ??)
      FROM "temppoolmember" AS "t" KEY JOIN "person"
      WHERE "t"."tempdeskid" = @TempDeskID
      AND
"employeeacceptable"("person"."personid",@CompanyID,@Explic,@VacancyID) IS
NULL
      AND "locate"((SELECT "personcurrentstates" FROM
"params"),"person"."status") > 0
      AND("isnull"("person"."OnlyMatchIfKnownAvailable",0) = 0
      OR "person"."personid" = any(SELECT "personid" FROM "tempshift" WHERE
"state" = 'A' AND "shiftdate" = @ShiftDate));
  CALL "MatchShifts"(@VacancyID,@Explic);
  -- add relevant candidates to the notify list (no attempt to eliminate
duplicates of records already in the queue)
  -- add contactstart and end to these and use in job when populating final
queue
  -- add contactstart and end to these and use in job when populating final
queue
  IF @OldWorkers > 0 THEN
    INSERT INTO "AutoMatchNotificationQueue"(
"PersonID","TempShiftPlanID","NotNewRating","LeadTime","NewWorkers","QtyWork
ers","Email","SMS","Push",
"MaxEmails","MaxSMS","MaxPush","MinDelayBetweenEmails","MinDelayBetweenSMS",
"MinDelayBetweenPush" )
      SELECT
"t"."PersonID",@TempShiftPlanID,"AutoMatchShiftMatchOrder"("t"."PersonID",@T
```

```
empShiftPlanID,"MatchScore",NULL,'Old') AS "Rating",
        @LeadTime,@NewWorkers,@OldWorkers,@Email,@SMS,@Push,
        "isnull"("MaxEmails","GlobalMaxEmails"),
        "isnull"("MaxSMS","GlobalMaxSMS"),
        "isnull"("MaxPush","GlobalMaxPush"),
        "isnull"("MinDelayBetweenEmails","GlobalMinEmailGap"),
        "isnull"("MinDelayBetweenSMS","GlobalMinSMSGap"),
        "isnull"("MinDelayBetweenPush","GlobalMinPushGap")
        FROM "TempMatchCandidate" AS "t" LEFT OUTER JOIN
"AutoMatchPersonConfig" AS "c" ON "t"."personid" = "c"."personid" WHERE
"VacancyID" = @VacancyID AND "Rating" > 0
        AND
"left"("tempshiftallowed"("t"."PersonID",@TempShiftPlanID,"VacancyID"),1)
IN( '^','' )
  END IF;
  IF @NewWorkers > 0 THEN
    INSERT INTO "AutoMatchNotificationQueue"(
"PersonID","TempShiftPlanID","NewRating","LeadTime","NewWorkers","QtyWorkers
","Email","SMS","Push",
"MaxEmails","MaxSMS","MaxPush","MinDelayBetweenEmails","MinDelayBetweenSMS",
"MinDelayBetweenPush" )
      SELECT
"t"."PersonID",@TempShiftPlanID,"AutoMatchShiftMatchOrder"("t"."PersonID",@T
empShiftPlanID,"MatchScore",NULL,'New') AS "Rating",
        @LeadTime,@NewWorkers,@OldWorkers,@Email,@SMS,@Push,
        "isnull"("MaxEmails","GlobalMaxEmails"),
        "isnull"("MaxSMS","GlobalMaxSMS"),
        "isnull"("MaxPush","GlobalMaxPush"),
        "isnull"("MinDelayBetweenEmails","GlobalMinEmailGap"),
        "isnull"("MinDelayBetweenSMS","GlobalMinSMSGap"),
        "isnull"("MinDelayBetweenPush","GlobalMinPushGap")
        FROM "TempMatchCandidate" AS "t" LEFT OUTER JOIN
"AutoMatchPersonConfig" AS "c" ON "t"."personid" = "c"."personid" WHERE
"VacancyID" = @VacancyID AND "Rating" > 0
        AND
"left"("tempshiftallowed"("t"."PersonID",@TempShiftPlanID,"VacancyID"),1)
IN( '^','' )
  END IF;
  -- clear out temp tables
  TRUNCATE TABLE "pears"."tempmatchcandidate";
  TRUNCATE TABLE "pears"."tempmatchplan"
END
GO

COMMENT TO PRESERVE FORMAT ON PROCEDURE "pears"."AutoMatchProcessShift" IS
{CREATE PROCEDURE AutoMatchProcessShift

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

```
(IN @TempShiftPlanID CHAR(20))
BEGIN
  DECLARE @Explic SMALLINT;
  DECLARE @TempDeskID CHAR(20);
  DECLARE @VacancyID CHAR(20);
  DECLARE @CompanyID CHAR(20);
  DECLARE @ShiftDate DATE;
  DECLARE @OldWorkers INTEGER;
  DECLARE @PercentNew INTEGER;
  DECLARE @NewWorkers INTEGER;
  DECLARE @LeadTime INTEGER;
  DECLARE @DivID CHAR(20);
  DECLARE @Urgency SMALLINT;
  DECLARE @Email SMALLINT;
  DECLARE @SMS SMALLINT;
  DECLARE @Push SMALLINT;
  SET @Explic = 1;
  SELECT v.TempDeskID, v.VacancyID, e.CompanyID, c.DivisionID, p.ShiftDate,
datediff(HOUR, CURRENT TIMESTAMP, p.ShiftDate+p.TimeFrom),
c.AutoMatchUrgency
 INTO @TempDeskID, @VacancyID, @CompanyID, @DivID, @ShiftDate, @LeadTime,
@Urgency
    FROM TempShiftPlan p KEY JOIN Vacancy v KEY JOIN Employment e KEY JOIN
Company c WHERE TempshiftPlanID = @TempShiftPlanID;
  SELECT FIRST QtyWorkers, PercentNew, Email, SMS, Push INTO @OldWorkers,
@PercentNew, @Email, @SMS, @Push FROM AutoMatchDivisionConfig WHERE
(divisionid = @DivID OR divisionid IS NULL)
 AND ("urgency" = @Urgency OR "urgency" IS NULL) AND (SELECT MIN(Leadtime)
FROM AutoMatchDivisionConfig a WHERE ("urgency" = @Urgency OR "urgency" IS
NULL)
 AND (divisionid = @DivID OR divisionid IS NULL) AND leadtime > @LeadTime) =
leadtime  ORDER BY divisionid DESC;
  SET @NewWorkers = round(@OldWorkers*@PercentNew/100,0);
  SET @OldWorkers = (@OldWorkers - @NewWorkers);
-- put shift plan in temp table
  INSERT INTO tempmatchplan
(tempmatchplanid,vacancyid,tempshiftplanid,shiftdate)
    (SELECT
uniquekey(p.tempshiftplanid),@VacancyID,p.tempshiftplanid,p.shiftdate AS
thedate
    FROM tempshiftplan p WHERE p.TempShiftPlanID = @TempShiftPlanID);
-- put possible temps in temp table
  INSERT INTO tempmatchcandidate
(tempmatchcandidateid,personid,matchscore,vacancyid,highlight)
    SELECT uniquekey(t.personid),
         t.personid,0,@VacancyID,
         ((SELECT FIRST 1 FROM employment WHERE companyid= @CompanyID AND
```

```
personid=person.personid)
         + isnull((SELECT FIRST 1 FROM placement KEY JOIN employment WHERE
employment.companyid = @CompanyID
                  AND employment.personid=person.personid AND
placement.vacancyid=@VacancyID),0)) -- use custom function (look for
existing ??)
   FROM temppoolmember t KEY JOIN person
   WHERE t.tempdeskid=@TempDeskID AND
          employeeacceptable(person.personid,@CompanyID,@Explic,@VacancyID)
IS NULL AND
          locate((SELECT personcurrentstates FROM params),person.status) >
AND
          (    isnull(person.OnlyMatchIfKnownAvailable,0)=0 OR
            person.personid IN (SELECT personid FROM tempshift WHERE
state='A' AND shiftdate = @ShiftDate));
  CALL MatchShifts(@VacancyID,@Explic);
-- add relevant candidates to the notify list (no attempt to eliminate
duplicates of records already in the queue)
-- add contactstart and end to these and use in job when populating final
queue
-- add contactstart and end to these and use in job when populating final
queue
  IF @OldWorkers > 0 THEN
    INSERT INTO AutoMatchNotificationQueue
(PersonID,TempShiftPlanID,NotNewRating,LeadTime,NewWorkers,QtyWorkers,Email,
SMS,Push,
    MaxEmails, MaxSMS, MaxPush, MinDelayBetweenEmails, MinDelayBetweenSMS,
MinDelayBetweenPush )
      SELECT  t.PersonID, @TempShiftPlanID,
AutoMatchShiftMatchOrder(t.PersonID ,@TempShiftPlanID,MatchScore,NULL,'Old')
AS Rating,
      @LeadTime, @NewWorkers, @OldWorkers, @Email, @SMS, @Push,
      isnull(MaxEmails,GlobalMaxEmails),
      isnull(MaxSMS,GlobalMaxSMS),
      isnull(MaxPush,GlobalMaxPush),
      isnull(MinDelayBetweenEmails,GlobalMinEmailGap),
      isnull(MinDelayBetweenSMS,GlobalMinSMSGap),
      isnull(MinDelayBetweenPush,GlobalMinPushGap)
       FROM TempMatchCandidate t LEFT OUTER JOIN AutoMatchPersonConfig c ON
t.personid = c.personid WHERE VacancyID = @VacancyID AND Rating > 0
      AND LEFT(tempshiftallowed(t.PersonID,@TempShiftPlanID,VacancyID),1) IN
('^','');
  END IF;
  IF @NewWorkers > 0 THEN
    INSERT INTO AutoMatchNotificationQueue
(PersonID,TempShiftPlanID,NewRating,LeadTime,NewWorkers,QtyWorkers,Email,SMS
,Push,
    MaxEmails, MaxSMS, MaxPush, MinDelayBetweenEmails, MinDelayBetweenSMS,
```

```
MinDelayBetweenPush )
      SELECT  t.PersonID, @TempShiftPlanID,
AutoMatchShiftMatchOrder(t.PersonID ,@TempShiftPlanID,MatchScore,NULL,'New')
AS Rating,
      @LeadTime, @NewWorkers, @OldWorkers, @Email, @SMS, @Push,
      isnull(MaxEmails,GlobalMaxEmails),
      isnull(MaxSMS,GlobalMaxSMS),
      isnull(MaxPush,GlobalMaxPush),
      isnull(MinDelayBetweenEmails,GlobalMinEmailGap),
      isnull(MinDelayBetweenSMS,GlobalMinSMSGap),
      isnull(MinDelayBetweenPush,GlobalMinPushGap)
       FROM TempMatchCandidate t LEFT OUTER JOIN AutoMatchPersonConfig c ON
t.personid = c.personid WHERE VacancyID = @VacancyID AND Rating > 0
      AND LEFT(tempshiftallowed(t.PersonID,@TempShiftPlanID,VacancyID),1) IN
('^','');
  END IF;
-- clear out temp tables
  TRUNCATE TABLE tempmatchcandidate;
  TRUNCATE TABLE tempmatchplan;
END
}
```