pears.Placement

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

Description

Header record for placements.

Columns

Column	Type	Null	Default	Comment
placementid	char(20)	NOT NULL		
placedate	date	NOT NULL		
employmentid	char(20)	NOT NULL		
vacancyid	char(20)	NULL		
commchgd	numeric(12,2)	NULL		
billed	smallint	NULL		
departmentid	char(2)	NOT NULL		
staffid	char(20)	NOT NULL		
temp	smallint	NOT NULL	0	
str1	char(20)	NULL		
str2	char(20)	NULL		
str3	char(20)	NULL		
str4	char(20)	NULL		
str5	char(20)	NULL		
salary	numeric(12,2)	NULL		
clientrate	numeric(12,2)	NULL		
temprate	numeric(12,2)	NULL		
note	char(1)	NULL		
State	char(1)	NULL		NULL:new A:awaits invoice I:invoiced C:cancelled
RefCode	char(20)	NULL		
TheirRef	char(50)	NULL		
ContractRef	char(20)	NULL		
DaysPerWeek	smallint	NULL		
TransferBatch	integer	NULL		

Column	Type	Null	Default	Comment
Currency	char(3)	NULL		
TempJobTypeID	char(20)	NULL		
ExtendedNotes	long varchar	NULL		
ExpenseBenefitSchemeID	char(20)	NULL		
PartCredited	smallint	NULL	0	Can only be set once invoiced. Incremented for each part credit
WorkMonday	tinyint	NULL		
WorkTuesday	tinyint	NULL		
WorkWednesday	tinyint	NULL		
WorkThursday	tinyint	NULL		
WorkFriday	tinyint	NULL		
WorkSaturday	tinyint	NULL		
WorkSunday	tinyint	NULL		
WorkNormalHours	double	NULL		
WorkStartTime	time	NULL		
lastcontactevent	timestamp	NULL		Added V2.2.2.13 Will be null for existing placements
documenttemplateid	char(12)	NULL		
WithdrawReason	char(100)	NULL		
PlacementRoleID	char(20)	NULL		
SpecialShiftContainer	tinyint	NULL		To allow shifts to be linked pre-timesheet, for special rates etc.
NotTaxable	smallint	NULL	0	Flag to override client public sector
WhenEntered	timestamp	NULL	current timestamp	
DirectEngagement	smallint	NULL	0	
AlternativeInvoiceAddressEmailID	char(20)	NULL		
SalesBrandID	char(20)	NULL		
InvoiceAddress	long varchar	NULL		
InvoiceEmail	char(250)	NULL		
InvoicePrefix	char(250)	NULL		
DirectEngagementPAYE	smallint	NULL	0	

Primary Key

- placementid

Foreign Keys

Constraint	Columns	References	Delete/update action
employment	employmentid	pears.employment (employmentid)	NOT NULL;
department	departmentid	pears.Department (departmentid)	NOT NULL;
staff	staffid	pears.staff (staffid)	NOT NULL;
TempJobType	TempJobTypeID	pears.TempJobType (TempJobTypeID)	ON DELETE SET NULL
ExpenseBenefitScheme	ExpenseBenefitSchemeID	pears.ExpenseBenefitScheme (ExpenseBenefitSchemeID)	ON DELETE SET NULL
vacancy	vacancyid	pears.vacancy (vacancyid)	
PlacementRole	PlacementRoleID	pears.PlacementRole (PlacementRoleID)	
iqacdocumenttemplate	documenttemplateid	pears.IQacDocumentTemplate (DocumentTemplateID)	ON DELETE SET NULL
AlternativeInvoiceAddressEmail	AlternativeInvoiceAddressEmailID	pears.AlternativeInvoiceAddressEmail (AlternativeInvoiceAddressEmailID)	ON DELETE SET NULL
SalesBrand	SalesBrandID	pears.SalesBrand (SalesBrandID)	ON DELETE SET NULL

Referenced By

Table	Constraint	Columns	Referenced columns
pears.AWRJobMaster	Placement	PlacementID	placementid
pears.contactevent	placement	placementid	placementid
pears.diary	placement	placementid	placementid
pears.ExpenseBenefitPlacementItem	Placement	PlacementID	placementid
pears.PlacementAnalysis	Placement	PlacementID	placementid
pears.placementattribution	placement	placementid	placementid
pears.PlacementDayVariation	placement	PlacementID	placementid
pears.PlacementElement	Placement	PlacementID	placementid
pears.PlacementExtension	Placement	PlacementID	placementid
pears.PlacementLink	placement	PlacementID1	placementid
pears.PlacementLink	PlacementID2	PlacementID2	placementid
pears.placementremuneration	placement	placementid	placementid
pears.PlacementTransfer	Placement	PlacementID	placementid
pears.progress	placement	placementid	placementid
pears.StaffHistory	Placement	PlacementID	placementid
pears.StagedPayments	placement	PlacementID	placementid
pears.TempJobRate	placement	PlacementID	placementid
pears.TempProvTimeSheet	Placement	PlacementID	placementid
pears.TempRateMarginAudit	placement	PlacementID	placementid
pears.TempShift	Placement	PlacementID	placementid
pears.TempTimeSheet	Placement	PlacementID	placementid

Indexes

Name	Type	Columns	Detail
placement_placedate	Index	placedate	
placement_state	Index	State	
person_lastcontactevent	Index	lastcontactevent	
PlacementWhenEntered	Index	WhenEntered	
placement_extendednotestext	Text index	ExtendedNotes	CONFIGURATION “SYS”.“default_char” IMMEDIATE REFRESH

Triggers

Name	Timing	Event
placementsalaryupd	after	update of “salary” order 1
placementinsert	before	insert order 1
placementAudit	before	update of “staffid”, “salary”,“placedate”,“extendednotes”,“refcode”,“nottaxable”,“theirref”,“invoiceaddress”,“invoiceemail”,“invoiceprefix”, “documenttemplateid” order 1
Placement_UpdateTrim	before	update order 2

Original SQL

```
-- IQX database structure split by table
-- Source: IQXDatabaseStructure - with comments.sql
-- Table: "pears"."Placement"
-- Table comment: Header record for placements.
-- Statement count: 30

CREATE TABLE "pears"."Placement" (
    "placementid"                   CHAR(20) NOT NULL
   ,"placedate"                     DATE NOT NULL
   ,"employmentid"                  CHAR(20) NOT NULL
   ,"vacancyid"                     CHAR(20) NULL
   ,"commchgd"                      NUMERIC(12,2) NULL
   ,"billed"                        SMALLINT NULL
   ,"departmentid"                  CHAR(2) NOT NULL
   ,"staffid"                       CHAR(20) NOT NULL
   ,"temp"                          SMALLINT NOT NULL DEFAULT 0
   ,"str1"                          CHAR(20) NULL
   ,"str2"                          CHAR(20) NULL
   ,"str3"                          CHAR(20) NULL
   ,"str4"                          CHAR(20) NULL
   ,"str5"                          CHAR(20) NULL
   ,"salary"                        NUMERIC(12,2) NULL
   ,"clientrate"                    NUMERIC(12,2) NULL
   ,"temprate"                      NUMERIC(12,2) NULL
   ,"note"                          CHAR(1) NULL
```

```
    ,"State"                            CHAR(1) NULL
    ,"RefCode"                          CHAR(20) NULL
    ,"TheirRef"                         CHAR(50) NULL
    ,"ContractRef"                      CHAR(20) NULL
    ,"DaysPerWeek"                      SMALLINT NULL
    ,"TransferBatch"                    INTEGER NULL
    ,"Currency"                         CHAR(3) NULL
    ,"TempJobTypeID"                    CHAR(20) NULL
    ,"ExtendedNotes"                    long VARCHAR NULL
    ,"ExpenseBenefitSchemeID"           CHAR(20) NULL
    ,"PartCredited"                     SMALLINT NULL DEFAULT 0
    ,"WorkMonday"                       tinyint NULL
    ,"WorkTuesday"                      tinyint NULL
    ,"WorkWednesday"                    tinyint NULL
    ,"WorkThursday"                     tinyint NULL
    ,"WorkFriday"                       tinyint NULL
    ,"WorkSaturday"                     tinyint NULL
    ,"WorkSunday"                       tinyint NULL
    ,"WorkNormalHours"                  DOUBLE NULL
    ,"WorkStartTime"                    TIME NULL
    ,"lastcontactevent"                 TIMESTAMP NULL
    ,"documenttemplateid"               CHAR(12) NULL
    ,"WithdrawReason"                   CHAR(100) NULL
    ,"PlacementRoleID"                  CHAR(20) NULL
    ,"SpecialShiftContainer"            tinyint NULL
    ,"NotTaxable"                       SMALLINT NULL DEFAULT 0
    ,"WhenEntered"                      TIMESTAMP NULL DEFAULT CURRENT
TIMESTAMP
    ,"DirectEngagement"                 SMALLINT NULL DEFAULT 0
    ,"AlternativeInvoiceAddressEmailID" CHAR(20) NULL
    ,"SalesBrandID"                     CHAR(20) NULL
    ,"InvoiceAddress"                   long VARCHAR NULL
    ,"InvoiceEmail"                     CHAR(250) NULL
    ,"InvoicePrefix"                    CHAR(250) NULL
    ,"DirectEngagementPAYE"             SMALLINT NULL DEFAULT 0
    ,PRIMARY KEY ("placementid" ASC)
)
GO


COMMENT ON COLUMN "pears"."Placement"."State" IS
    'NULL:new A:awaits invoice I:invoiced C:cancelled'
GO


COMMENT ON COLUMN "pears"."Placement"."PartCredited" IS
    'Can only be set once invoiced. Incremented for each part credit'
GO
```

```
COMMENT ON COLUMN "pears"."Placement"."lastcontactevent" IS
    'Added V2.2.2.13 Will be null for existing placements'
GO


COMMENT ON COLUMN "pears"."Placement"."SpecialShiftContainer" IS
    'To allow shifts to be linked pre-timesheet, for special rates etc.'
GO


COMMENT ON COLUMN "pears"."Placement"."NotTaxable" IS
    'Flag to override client public sector'
GO


COMMENT ON TABLE "pears"."Placement" IS
    'Header record for placements.'
GO


ALTER TABLE "pears"."Placement"
    ADD NOT NULL FOREIGN KEY "employment" ("employmentid" ASC)
    REFERENCES "pears"."employment" ("employmentid")
GO


ALTER TABLE "pears"."Placement"
    ADD NOT NULL FOREIGN KEY "department" ("departmentid" ASC)
    REFERENCES "pears"."Department" ("departmentid")
GO


ALTER TABLE "pears"."Placement"
    ADD NOT NULL FOREIGN KEY "staff" ("staffid" ASC)
    REFERENCES "pears"."staff" ("staffid")
GO


ALTER TABLE "pears"."Placement"
    ADD FOREIGN KEY "TempJobType" ("TempJobTypeID" ASC)
    REFERENCES "pears"."TempJobType" ("TempJobTypeID")
    ON DELETE SET NULL
GO


ALTER TABLE "pears"."Placement"
```

```
    ADD FOREIGN KEY "ExpenseBenefitScheme" ("ExpenseBenefitSchemeID" ASC)
    REFERENCES "pears"."ExpenseBenefitScheme" ("ExpenseBenefitSchemeID")
    ON DELETE SET NULL
GO


ALTER TABLE "pears"."Placement"
    ADD FOREIGN KEY "vacancy" ("vacancyid" ASC)
    REFERENCES "pears"."vacancy" ("vacancyid")
GO


ALTER TABLE "pears"."Placement"
    ADD FOREIGN KEY "PlacementRole" ("PlacementRoleID" ASC)
    REFERENCES "pears"."PlacementRole" ("PlacementRoleID")
GO


ALTER TABLE "pears"."Placement"
    ADD FOREIGN KEY "iqacdocumenttemplate" ("documenttemplateid" ASC)
    REFERENCES "pears"."IQacDocumentTemplate" ("DocumentTemplateID")
    ON DELETE SET NULL
GO


ALTER TABLE "pears"."Placement"
    ADD FOREIGN KEY "AlternativeInvoiceAddressEmail"
("AlternativeInvoiceAddressEmailID" ASC)
    REFERENCES "pears"."AlternativeInvoiceAddressEmail"
("AlternativeInvoiceAddressEmailID")
    ON DELETE SET NULL
GO


ALTER TABLE "pears"."Placement"
    ADD FOREIGN KEY "SalesBrand" ("SalesBrandID" ASC)
    REFERENCES "pears"."SalesBrand" ("SalesBrandID")
    ON DELETE SET NULL
GO


CREATE INDEX "placement_placedate" ON "pears"."Placement"
    ( "placedate" DESC )
GO


CREATE INDEX "placement_state" ON "pears"."Placement"
    ( "State" )
```

```
GO


CREATE INDEX "person_lastcontactevent" ON "pears"."Placement"
    ( "lastcontactevent" DESC )
GO


CREATE INDEX "PlacementWhenEntered" ON "pears"."Placement"
    ( "WhenEntered" )
GO


CREATE TEXT INDEX "placement_extendednotestext" ON "pears"."Placement"
    ( "ExtendedNotes" ) CONFIGURATION "SYS"."default_char" IMMEDIATE REFRESH
GO


CREATE TRIGGER "placementsalaryupd" after UPDATE OF "salary"
ORDER 1 ON "pears"."Placement"
REFERENCING NEW AS "new_plac"
FOR each ROW
BEGIN
  UPDATE "employment" SET "salary" = "new_plac"."salary" WHERE
"employmentid" = "new_plac"."employmentid"
END
GO


COMMENT TO PRESERVE FORMAT ON TRIGGER
"pears"."Placement"."placementsalaryupd" IS
{CREATE TRIGGER placementsalaryupd
 after UPDATE OF salary
ORDER 1 ON pears.Placement
REFERENCING NEW AS new_plac
FOR each ROW
BEGIN
  UPDATE employment SET salary = new_plac.salary WHERE employmentid =
new_plac.employmentid
END
}
GO


CREATE TRIGGER "placementinsert" BEFORE INSERT ORDER 1 ON
"pears"."placement"
REFERENCING NEW AS "new_placement"
FOR each ROW
```

```
BEGIN
  DECLARE "autonumplacs" SMALLINT;
  DECLARE "lastplacnum" INTEGER;
  SET "new_placement"."refcode" = "trim"("new_placement"."refcode");
  SET "new_placement"."theirref" = "trim"("new_placement"."theirref");
  SET "new_placement"."contractref" = "trim"("new_placement"."contractref");
  IF "new_placement"."refcode" IS NULL THEN
    SELECT "autoplacnumber","nextplacnumber" INTO
"autonumplacs","lastplacnum" FROM "params";
    IF "isnull"("autonumplacs",0) <> 0 THEN
      SET "new_placement"."refcode" = "isnull"("lastplacnum",0)+1;
      UPDATE "params" SET "nextplacnumber" = "isnull"("nextplacnumber",0)+1
    END IF
  END IF
END
GO


COMMENT TO PRESERVE FORMAT ON TRIGGER "pears"."Placement"."placementinsert"
IS
{CREATE TRIGGER placementinsert
 BEFORE INSERT ORDER 1
ON pears.placement
REFERENCING NEW AS new_placement
FOR each ROW
BEGIN
  DECLARE autonumplacs SMALLINT;
  DECLARE lastplacnum INTEGER;
  SET new_placement.refcode = TRIM(new_placement.refcode);
  SET new_placement.theirref = TRIM(new_placement.theirref);
  SET new_placement.contractref = TRIM(new_placement.contractref);
  IF new_placement.refcode IS NULL THEN
    SELECT autoplacnumber,nextplacnumber INTO autonumplacs,lastplacnum FROM
params;
    IF isnull(autonumplacs,0)<>0 THEN
      SET new_placement.refcode=isnull(lastplacnum,0)+1;
      UPDATE params SET nextplacnumber=isnull(nextplacnumber,0)+1
    END IF
  END IF
END
}
GO


CREATE TRIGGER "placementAudit" BEFORE UPDATE OF "staffid",
"salary","placedate","extendednotes","refcode","nottaxable","theirref","invo
iceaddress","invoiceemail","invoiceprefix",
"documenttemplateid" ORDER 1 ON "pears"."placement"
```

```
REFERENCING OLD AS "old_place" NEW AS "new_place"
FOR each ROW
WHEN(EXISTS(SELECT * FROM "AuditItems" WHERE "AreaName" = 'Placement' AND
"AuditFlag" = 1 AND "auditemid" <> 'XXMAN0000000000015'))
BEGIN
  DECLARE "AuditList" long VARCHAR;
  DECLARE "OldDescrip" CHAR(250);
  DECLARE "NewDescrip" CHAR(250);
  SELECT "string"(',',"list"("ItemName"),',') INTO "AuditList" FROM
"AuditItems" WHERE "AreaName" = 'Placement' AND "AuditFlag" = 1;
  -- Consultant
  IF "locate"("AuditList",',Consultant,') > 0 AND UPDATE("staffid") AND
"old_place"."staffid" <> "new_place"."staffid" THEN
    SELECT "userid" INTO "OldDescrip" FROM "staff" WHERE "staffid" =
"old_place"."staffid";
    SELECT "userid" INTO "NewDescrip" FROM "staff" WHERE "staffid" =
"new_place"."staffid";
    CALL
"AuditLog"('PLACEMENT',"new_place"."placementid","string"('Consultant
Updated Our Ref. - ',"new_place"."refcode"),"OldDescrip","NewDescrip");
    INSERT INTO "staffhistory"(
"staffhistoryid","placementid","oldstaff","newstaff","whoentered","whenenter
ed" ) VALUES(
"uniquekey"("new_place"."staffid"),"new_place"."placementid","old_place"."st
affid","new_place"."staffid","userstaffid",CURRENT TIMESTAMP )
  END IF;
  -- Salary
  IF "locate"("AuditList",',Salary,') > 0 AND UPDATE("salary") THEN
    CALL "AuditLog"('PLACEMENT',"new_place"."placementid","string"('Salary
Updated Our Ref. -
',"new_place"."refcode"),"old_place"."salary","new_place"."salary")
  END IF;
  -- Notes
  IF "locate"("AuditList",',Notes,') > 0 AND UPDATE("extendednotes") THEN
    CALL "AuditLog"('PLACEMENT',"new_place"."placementid","string"('Notes
Updated Our Ref. -
',"new_place"."refcode"),"old_place"."extendednotes","new_place"."extendedno
tes")
  END IF;
  -- TheirRef
  IF "locate"("AuditList",',TheirRef,') > 0 AND UPDATE("TheirRef") THEN
    CALL "AuditLog"('PLACEMENT',"new_place"."placementid","string"('TheirRef
Updated Our Ref. -
',"new_place"."refcode"),"old_place"."TheirRef","new_place"."TheirRef")
  END IF;
  -- Our Ref
  IF "locate"("AuditList",',Our Ref,') > 0 AND UPDATE("refcode") THEN
    CALL "AuditLog"('PLACEMENT',"new_place"."placementid","string"('Our Ref
```

```
Updated Our Ref. -
',"new_place"."refcode"),"old_place"."refcode","new_place"."refcode")
  END IF;
  -- PlaceDate
  IF "locate"("AuditList",',Placement Date,') > 0 AND UPDATE("placedate")
THEN
    CALL
"AuditLog"('PLACEMENT',"new_place"."placementid","string"('Placement Date
Updated Our Ref. -
',"new_place"."refcode"),"string"("old_place"."placedate"),"string"("new_pla
ce"."placedate"))
  END IF;
  IF "locate"("AuditList",',Invoice Email,') > 0 AND UPDATE("InvoiceEmail")
THEN
    CALL
"AuditLog"('PLACEMENT',"new_place"."placementid","string"('Placement Date
Invoice Email -
',"new_place"."refcode"),"old_place"."invoiceemail","new_place"."invoiceemai
l")
  END IF;
  IF "locate"("AuditList",',Invoice Address,') > 0 AND
UPDATE("InvoiceAddress") THEN
    CALL
"AuditLog"('PLACEMENT',"new_place"."placementid","string"('Placement Invoice
Address -
',"new_place"."refcode"),"old_place"."invoiceaddress","new_place"."invoicead
dress")
  END IF;
  IF "locate"("AuditList",',Invoice Prefix,') > 0 AND
UPDATE("InvoicePrefix") THEN
    CALL
"AuditLog"('PLACEMENT',"new_place"."placementid","string"('Placement Date
Invoice Prefix -
',"new_place"."refcode"),"old_place"."invoiceprefix","new_place"."invoicepre
fix")
  END IF;
  IF "locate"("AuditList",',Override IR35 Public Sector,') > 0 AND
UPDATE("nottaxable") THEN
    CALL "AuditLog"('PLACEMENT',"new_place"."placementid","string"('Override
IR35 Public Sector -
',"new_place"."refcode"),"old_place"."nottaxable","new_place"."nottaxable")
  END IF;
  --  Override Invoice Layout
  IF "locate"("AuditList",',Override Invoice Layout,') > 0 AND
UPDATE("documenttemplateid") THEN
    SET "OldDescrip" = (SELECT "name" FROM "iqacdocumenttemplate" WHERE
"documenttemplateid" = "old_place"."documenttemplateid");
    SET "NewDescrip" = (SELECT "name" FROM "iqacdocumenttemplate" WHERE
```

```
"documenttemplateid" = "new_place"."documenttemplateid");
    CALL "AuditLog"('PLACEMENT',"old_place"."PlacementId","string"('Override
Invoice Layout Updated - ',"new_place"."refcode"),"OldDescrip","NewDescrip")
  END IF
END
GO


COMMENT TO PRESERVE FORMAT ON TRIGGER "pears"."Placement"."placementAudit"
IS
{CREATE TRIGGER placementAudit
 BEFORE UPDATE OF staffid, salary, placedate, extendednotes,
refcode,nottaxable, theirref,invoiceaddress, invoiceemail, invoiceprefix,
documenttemplateid
 ORDER 1 ON pears.placement
REFERENCING OLD AS old_place NEW AS new_place
FOR each ROW
WHEN(EXISTS(SELECT* FROM AuditItems WHERE AreaName = 'Placement' AND
AuditFlag = 1 AND auditemid <> 'XXMAN0000000000015'))
BEGIN
  DECLARE AuditList long VARCHAR;
  DECLARE OldDescrip CHAR(250);
  DECLARE NewDescrip CHAR(250);
  SELECT string(',',list(ItemName),',') INTO AuditList FROM AuditItems WHERE
AreaName = 'Placement' AND AuditFlag = 1;
  -- Consultant
  IF locate(AuditList,',Consultant,') > 0 AND UPDATE(staffid) AND
old_place.staffid <> new_place.staffid THEN
    SELECT userid INTO OldDescrip FROM staff WHERE staffid =
old_place.staffid ;
    SELECT userid INTO NewDescrip FROM staff WHERE staffid =
new_place.staffid ;
    CALL AuditLog('PLACEMENT',new_place.placementid,string('Consultant
Updated Our Ref. - ',new_place.refcode),OldDescrip,NewDescrip);
    INSERT INTO staffhistory (staffhistoryid, placementid, oldstaff,
newstaff, whoentered, whenentered) VALUES
(uniquekey(new_place.staffid),new_place.placementid, old_place.staffid,
new_place.staffid, userstaffid, CURRENT TIMESTAMP)
  END IF;
  -- Salary
  IF locate(AuditList,',Salary,') > 0 AND UPDATE(salary) THEN
    CALL AuditLog('PLACEMENT',new_place.placementid,string('Salary Updated
Our Ref. - ',new_place.refcode),old_place.salary,new_place.salary)
  END IF;
  -- Notes
  IF locate(AuditList,',Notes,') > 0 AND UPDATE(extendednotes) THEN
    CALL AuditLog('PLACEMENT',new_place.placementid,string('Notes Updated
Our Ref. -
```

```
',new_place.refcode),old_place.extendednotes,new_place.extendednotes)
  END IF;
  -- TheirRef
  IF locate(AuditList,',TheirRef,') > 0 AND UPDATE(TheirRef) THEN
    CALL AuditLog('PLACEMENT',new_place.placementid,string('TheirRef Updated
Our Ref. - ',new_place.refcode),old_place.TheirRef,new_place.TheirRef)
  END IF;
  -- Our Ref
  IF locate(AuditList,',Our Ref,') > 0 AND UPDATE(refcode) THEN
    CALL AuditLog('PLACEMENT',new_place.placementid,string('Our Ref Updated
Our Ref. - ',new_place.refcode),old_place.refcode,new_place.refcode)
  END IF;
  -- PlaceDate
  IF locate(AuditList,',Placement Date,') > 0 AND UPDATE(placedate) THEN
    CALL AuditLog('PLACEMENT',new_place.placementid,string('Placement Date
Updated Our Ref. -
',new_place.refcode),string(old_place.placedate),string(new_place.placedate)
)
  END IF;
  IF locate(AuditList,',Invoice Email,') > 0 AND UPDATE(InvoiceEmail) THEN
    CALL AuditLog('PLACEMENT',new_place.placementid,string('Placement Date
Invoice Email -
',new_place.refcode),old_place.invoiceemail,new_place.invoiceemail)
  END IF;
  IF locate(AuditList,',Invoice Address,') > 0 AND UPDATE(InvoiceAddress)
THEN
    CALL AuditLog('PLACEMENT',new_place.placementid,string('Placement
Invoice Address -
',new_place.refcode),old_place.invoiceaddress,new_place.invoiceaddress)
  END IF;
  IF locate(AuditList,',Invoice Prefix,') > 0 AND UPDATE(InvoicePrefix) THEN
    CALL AuditLog('PLACEMENT',new_place.placementid,string('Placement Date
Invoice Prefix -
',new_place.refcode),old_place.invoiceprefix,new_place.invoiceprefix)
  END IF;
  IF locate(AuditList,',Override IR35 Public Sector,') > 0 AND
UPDATE(nottaxable) THEN
    CALL AuditLog('PLACEMENT',new_place.placementid,string('Override IR35
Public Sector -
',new_place.refcode),old_place.nottaxable,new_place.nottaxable)
  END IF;
  --  Override Invoice Layout
  IF locate(AuditList,',Override Invoice Layout,') > 0 AND
UPDATE(documenttemplateid) THEN
  SET OldDescrip=(SELECT name FROM iqacdocumenttemplate WHERE
documenttemplateid = old_place.documenttemplateid);
  SET NewDescrip=(SELECT name FROM iqacdocumenttemplate WHERE
documenttemplateid = new_place.documenttemplateid);
```

```
    CALL AuditLog('PLACEMENT',old_place.PlacementId,string('Override Invoice
Layout Updated - ',new_place.refcode),OldDescrip,NewDescrip)
  END IF;
END
}
GO


CREATE TRIGGER "Placement_UpdateTrim" BEFORE UPDATE ORDER 2 ON
"pears"."placement"
REFERENCING NEW AS "new_placement"
FOR each ROW
BEGIN
  SET "new_placement"."refcode" = "trim"("new_placement"."refcode");
  SET "new_placement"."theirref" = "trim"("new_placement"."theirref");
  SET "new_placement"."contractref" = "trim"("new_placement"."contractref")
END
GO


COMMENT TO PRESERVE FORMAT ON TRIGGER
"pears"."Placement"."Placement_UpdateTrim" IS
{CREATE TRIGGER Placement_UpdateTrim
 BEFORE UPDATE ORDER 2 ON
pears.placement
REFERENCING NEW AS new_placement
FOR each ROW
BEGIN
  SET new_placement.refcode = TRIM(new_placement.refcode);
  SET new_placement.theirref = TRIM(new_placement.theirref);
  SET new_placement.contractref = TRIM(new_placement.contractref)
END
}
GO
```