pears.BoilerPlate

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

Description

Contains standard text which can be dropped into various types of note field.

Columns

Column	Type	Null	Default	Comment
BoilerPlateID	char(20)	NOT NULL		
Type	char(2)	NOT NULL		
Description	char(30)	NOT NULL		
SortOrder	integer	NULL		
BoilerPlateText	long varchar	NULL		

Primary Key

- BoilerPlateID

Foreign Keys

- No outgoing foreign keys found.

Referenced By

Table	Constraint	Columns	Referenced columns
pears.BoilerPlatedivision	BoilerPlate	BoilerPlateid	BoilerPlateID
pears.NotificationTemplate	BoilerPlate	BoilerPlateID	BoilerPlateID

Indexes

- No indexes found.

Triggers

Name	Timing	Event
BoilerPlateDeleteAudit	after	delete order 1
BoilerPlateUpdateAudit	after	update of “BoilerPlateID”, “Description”,“Type”, “BoilerPlateText” order 1

Original SQL

```
-- IQX database structure split by table
-- Source: IQXDatabaseStructure - with comments.sql
-- Table: "pears"."BoilerPlate"
-- Table comment: Contains standard text which can be dropped into various
types of note field.
-- Statement count: 6

CREATE TABLE "pears"."BoilerPlate" (
    "BoilerPlateID"                 CHAR(20) NOT NULL
   ,"Type"                          CHAR(2) NOT NULL
   ,"Description"                   CHAR(30) NOT NULL
   ,"SortOrder"                     INTEGER NULL
   ,"BoilerPlateText"               long VARCHAR NULL
   ,PRIMARY KEY ("BoilerPlateID" ASC)
)
GO


COMMENT ON TABLE "pears"."BoilerPlate" IS
    'Contains standard text which can be dropped into various types of note
field.'
GO


CREATE TRIGGER "BoilerPlateDeleteAudit" after DELETE ORDER 1 ON
"pears"."BoilerPlate"
REFERENCING OLD AS "old_Plate"
FOR each ROW
BEGIN
  CALL "AuditLog"('BOILERPLATE',"old_Plate"."BoilerPlateID",
  'Deleted Boiler Plate',
  "string"(' Type = ',"old_Plate"."Type",
  ', Description = ',"old_Plate"."Description",
  ', Order = ',"old_Plate"."SortOrder",
  ', BoilerPlateText = ',"old_Plate"."BoilerPlateText"),
  '')
END
```

```
GO


COMMENT TO PRESERVE FORMAT ON TRIGGER
"pears"."BoilerPlate"."BoilerPlateDeleteAudit" IS
{CREATE TRIGGER BoilerPlateDeleteAudit
 after DELETE ORDER 1 ON
BoilerPlate
REFERENCING OLD AS old_Plate
FOR each ROW
BEGIN
  CALL AuditLog('BOILERPLATE', old_Plate.BoilerPlateID,
               'Deleted Boiler Plate',
               string(' Type = ', old_Plate.Type,
                      ', Description = ', old_Plate.Description,
                      ', Order = ', old_Plate.SortOrder,
                      ', BoilerPlateText = ', old_Plate.BoilerPlateText),
               '');
END
}
GO


CREATE TRIGGER "BoilerPlateUpdateAudit" after UPDATE OF "BoilerPlateID",
"Description","Type",
"BoilerPlateText" ORDER 1 ON "pears"."BoilerPlate"
REFERENCING OLD AS "old_Plate" NEW AS "new_Plate"
FOR each ROW
BEGIN
  IF UPDATE("BoilerPlateID") THEN
    CALL
"AuditLog"('BOILERPLATE',"old_Plate"."BoilerPlateID","string"('BOILERPLATE
Updated - ID :
',"old_Plate"."BoilerPlateID"),"old_Plate"."BoilerPLateID","new_Plate"."Boil
erPLateID")
  END IF;
  IF UPDATE("Description") THEN
    CALL
"AuditLog"('BOILERPLATE',"old_Plate"."BoilerPlateID","string"('BOILERPLATE
Updated - Description :
',"old_Plate"."Description"),"old_Plate"."Description","new_Plate"."Descript
ion")
  END IF;
  IF UPDATE("Type") THEN
    CALL
"AuditLog"('BOILERPLATE',"old_Plate"."BoilerPlateID","string"('BOILERPLATE
Updated - Type :
',"old_Plate"."Type"),"old_Plate"."Type","new_Plate"."Type")
```

```
  END IF;
  IF UPDATE("BoilerPlateText") THEN
    CALL
"AuditLog"('BOILERPLATE',"old_Plate"."BoilerPlateID","string"('BOILERPLATE
Updated - BoilerPlateText :
',"old_Plate"."BoilerPlateText"),"old_Plate"."BoilerPlateText","new_Plate"."
BoilerPlateText")
  END IF
END
GO


COMMENT TO PRESERVE FORMAT ON TRIGGER
"pears"."BoilerPlate"."BoilerPlateUpdateAudit" IS
{CREATE TRIGGER BoilerPlateUpdateAudit
after UPDATE OF
BoilerPlateID, Description, TYPE, BoilerPlateText
 ORDER 1 ON pears.BoilerPlate
REFERENCING OLD AS old_Plate NEW AS new_Plate
FOR each ROW
BEGIN
  IF UPDATE(BoilerPlateID) THEN
    CALL AuditLog('BOILERPLATE',old_Plate.BoilerPlateID,string('BOILERPLATE
Updated - ID :
',old_Plate.BoilerPlateID),old_Plate.BoilerPLateID,new_Plate.BoilerPLateID)
  END IF ;
  IF UPDATE(Description) THEN
    CALL AuditLog('BOILERPLATE',old_Plate.BoilerPlateID,string('BOILERPLATE
Updated - Description :
',old_Plate.Description),old_Plate.Description,new_Plate.Description)
  END IF ;
  IF UPDATE(TYPE) THEN
    CALL AuditLog('BOILERPLATE',old_Plate.BoilerPlateID,string('BOILERPLATE
Updated - Type : ',old_Plate.Type),old_Plate.Type,new_Plate.Type)
  END IF ;
  IF UPDATE(BoilerPlateText) THEN
    CALL AuditLog('BOILERPLATE',old_Plate.BoilerPlateID,string('BOILERPLATE
Updated - BoilerPlateText :
',old_Plate.BoilerPlateText),old_Plate.BoilerPlateText,new_Plate.BoilerPlate
Text)
  END IF ;
END
}
GO
```