**Create table "SYSTEM"."SAMPLE"("TestID" integer, "testName" varchar(1000));**

**INSERT INTO "SYSTEM"."SAMPLE" VALUES(**

**234/\*TestID <INTEGER>\*/,**

**'jkjkjk hjkljasd ljadse2@'/\*testName <VARCHAR(1000)>\*/**

**);**

CREATE COLUMN TABLE "WORKMERK"."MESSAGE"(

                "MESSAGEID" INTEGER CS\_INT NOT NULL GENERATED BY DEFAULT AS IDENTITY,

                "SENDERID" INTEGER CS\_INT,

                "RECEIVERID" INTEGER CS\_INT,

                "MESSAGECONTENT" VARCHAR(1000),

                "QUESTIONTYPEID" INTEGER CS\_INT,

                "ANSWERCONTENT" VARCHAR(1000),

                "ISREADQUESTIONBYRECEIVER" VARCHAR(100),

                "ISREADANSWERBYSENDER" VARCHAR(100),

                "MESSAGESENDDATE" LONGDATE CS\_LONGDATE,

                "ANSWERSENDDATE" LONGDATE CS\_LONGDATE,

                "LONGITUDE" DECIMAL(11, 8) CS\_FIXED,

                "LATITUDE" DECIMAL(10, 8) CS\_FIXED,

                "RECEIVERNAME" VARCHAR(255),

                "SENDERNAME" VARCHAR(255),

                "QUESTIONTYPE" VARCHAR(255),

                "SENDERADDRESS" VARCHAR(255),

                PRIMARY KEY (

                                "MESSAGEID"

                )

) UNLOAD PRIORITY 5 AUTO MERGE;

**CREATE COLUMN TABLE "SYSTEM"."Log"(**

**"TestID" INTEGER CS\_INT NOT NULL GENERATED BY DEFAULT AS IDENTITY,**

**"testName" VARCHAR(1000),**

**PRIMARY KEY (**

**"TestID"**

**)**

**) UNLOAD PRIORITY 5 AUTO MERGE;**

**.xsaccess**

{

"exposed": true,

"authentication": [{

"method": "Form"

}],

"authorization": [],

"mime\_mapping": [{

"extension": "jpg",

"mimetype": "image/jpeg"

}],

"force\_ssl": false,

"enable\_etags": true,

"prevent\_xsrf": false,

"anonymous\_connection": null,

"cors": [{

"enabled": false

}],

"cache\_control": "no-cache, no-store",

"default\_file": "index.html"

}

**.xsapp**

{}

**.xsprivileges**

{

"privileges": [

{"name": "Execute","description": "Execute"}

]

}

**services.xsodata**

service {

"WORKMERK"."USER" as "USER" navigates ("UserChecklistId" as "CHECKLISTDATA") create using "workmerk.poc1.procedures::createAutoIdUser";

association "UserChecklistId"

principal "USER"("USERID") multiplicity "1"

dependent "CHECKLISTDATA"("USERID") multiplicity "\*";

"WORKMERK"."USERROLE" as "USERROLE" navigates ("UserRoleId" as "USER");

association "UserRoleId"

principal "USERROLE"("ROLEID") multiplicity "1"

dependent "USER"("ROLEID") multiplicity "\*";

"WORKMERK"."CITY" as "CITY" navigates ("UserCityId" as "USER");

association "UserCityId"

principal "CITY"("CITYID") multiplicity "1"

dependent "USER"("CITYID") multiplicity "\*";

"WORKMERK"."STATE" as "STATE" navigates ("UserStateId" as "USER");

association "UserStateId"

principal "STATE"("STATEID") multiplicity "1"

dependent "USER"("STATEID") multiplicity "\*" ;

"WORKMERK"."CHECKLISTDATA" as "CHECKLISTDATA";

"WORKMERK"."QUESTIONTYPE" as "QUESTIONTYPE" navigates ("MessageQuestionTypeId" as "QUESTION");

association "MessageQuestionTypeId"

principal "QUESTIONTYPE"("QUESTIONTYPEID") multiplicity "1"

dependent "MESSAGE"("QUESTIONTYPEID") multiplicity "\*";

"WORKMERK"."CHECKLIST" as "CHECKLIST" navigates ("CheckListItemId" as "CHECKLISTDATA");

association "CheckListItemId"

principal "CHECKLIST"("ITEMID") multiplicity "1"

dependent "CHECKLISTDATA"("ITEMID") multiplicity "\*";

"WORKMERK"."MESSAGE" as "MESSAGE" create using "workmerk.poc1.procedures::createAutoIdMessage";

"WORKMERK"."Test" as "Test";

"WORKMERK"."Sample\_Auto" as "Sample\_Auto" create using "workmerk.poc1.procedures::createAutoIdSample";

}

**PROCEDURE "WORKMERK"."workmerk.poc1.procedures::createAutoIdAttachment"(**

**IN row TT\_ATTACHMENT,**

**OUT error TT\_ERROR**

**)**

**LANGUAGE SQLSCRIPT**

**SQL SECURITY INVOKER**

**DEFAULT SCHEMA WORKMERK**

**AS**

**BEGIN**

**DECLARE lv\_AttachmentID INTEGER;**

**DECLARE lv\_MessageId INTEGER;**

**DECLARE lv\_Attachment BLOB;**

**DECLARE lv\_SenderId INTEGER;**

**DECLARE lv\_ReceiverId INTEGER;**

**DECLARE lv\_Time LONGDATE;**

**DECLARE lv\_Type STRING;**

**DECLARE lv\_senderName STRING;**

**DECLARE lv\_receiverName STRING;**

**SELECT "ATTACHMENTID",**

**"MESSAGEID",**

**"ATTACHMENT",**

**"SENDERID",**

**"RECEIVERID",**

**"TIME",**

**"TYPE",**

**"SENDERNAME",**

**"RECEIVERNAME"**

**INTO lv\_AttachmentID,**

**lv\_MessageId,**

**lv\_Attachment,**

**lv\_SenderId,**

**lv\_ReceiverId,**

**lv\_Time,**

**lv\_Type,**

**lv\_senderName,**

**lv\_receiverName**

**FROM :row;**

**IF (:lv\_Type = '') THEN**

**error = SELECT 400 AS http\_status\_code,**

**'Invalid Decription' AS error\_message,**

**'Description Field cannot be empty' AS detail**

**FROM dummy;**

**ELSE**

**INSERT INTO "WORKMERK"."ATTACHMENT"("MESSAGEID", "ATTACHMENT", "SENDERID", "RECEIVERID", "TIME", "TYPE", "SENDERNAME", "RECEIVERNAME")**

**VALUES(lv\_MessageId, lv\_Attachment, lv\_SenderId, lv\_ReceiverId, lv\_Time, lv\_Type, lv\_senderName, lv\_receiverName);**

**END IF;**

**END;**

**PROCEDURE "WORKMERK"."workmerk.poc1.procedures::createAutoIdLog"(**

**IN row TT\_LOG,**

**OUT error TT\_ERROR**

**)**

**LANGUAGE SQLSCRIPT**

**SQL SECURITY INVOKER**

**DEFAULT SCHEMA WORKMERK**

**AS**

**BEGIN**

**DECLARE lv\_LogId INTEGER;**

**DECLARE lv\_LogContent STRING;**

**DECLARE lv\_Time LONGDATE;**

**DECLARE lv\_LogAttachment BLOB;**

**DECLARE lv\_LogAttachmentType STRING;**

**DECLARE lv\_UserID INTEGER;**

**DECLARE lv\_UserName STRING;**

**DECLARE lv\_Longitude STRING;**

**DECLARE lv\_Latitude STRING;**

**DECLARE lv\_Address STRING;**

**SELECT "LOGID",**

**"LOGCONTENT",**

**"TIME",**

**"LOGATTACHMENT",**

**"LOGATTACHMENTTYPE",**

**"USERID",**

**"USERNAME",**

**"LONGITUDE",**

**"LATITUDE",**

**"ADDRESS"**

**INTO lv\_LogId,**

**lv\_LogContent,**

**lv\_Time,**

**lv\_LogAttachment,**

**lv\_LogAttachmentType,**

**lv\_UserID,**

**lv\_UserName,**

**lv\_Longitude,**

**lv\_Latitude,**

**lv\_Address**

**FROM :row;**

**IF (:lv\_Time = '') THEN**

**error = SELECT 400 AS http\_status\_code,**

**'Invalid Decription' AS error\_message,**

**'Description Field cannot be empty' AS detail**

**FROM dummy;**

**ELSE**

**INSERT INTO "WORKMERK"."LOG"("LOGCONTENT", "TIME", "LOGATTACHMENT", "LOGATTACHMENTTYPE", "USERID", "USERNAME", "LONGITUDE", "LATITUDE", "ADDRESS")**

**VALUES(lv\_LogContent, lv\_Time, lv\_LogAttachment, lv\_LogAttachmentType, lv\_UserID, lv\_UserName, lv\_Longitude, lv\_Latitude, lv\_Address);**

**END IF;**

**END;**

**PROCEDURE "WORKMERK"."workmerk.poc1.procedures::createAutoIdMessage"(**

**IN row TEMP\_MESSAGE,**

**OUT error TT\_ERROR**

**)**

**LANGUAGE SQLSCRIPT**

**SQL SECURITY INVOKER**

**DEFAULT SCHEMA WORKMERK**

**AS**

**BEGIN**

**DECLARE lv\_ID INTEGER;**

**DECLARE lv\_SenderId INTEGER;**

**DECLARE lv\_ReceiverId INTEGER;**

**DECLARE lv\_MsgContent STRING;**

**DECLARE lv\_QtnTypId INTEGER;**

**DECLARE lv\_AnsContent STRING;**

**DECLARE lv\_IsReadReceiver STRING;**

**DECLARE lv\_IsSendSender STRING;**

**DECLARE lv\_MsgSendDate LONGDATE;**

**DECLARE lv\_MsgAnswerDate LONGDATE;**

**DECLARE lv\_Longitude String;**

**DECLARE lv\_Latitude Decimal;**

**DECLARE lv\_SenderName STRING;**

**DECLARE lv\_ReceiverName STRING;**

**DECLARE lv\_questionType STRING;**

**DECLARE lv\_senderAddress STRING;**

**SELECT "MESSAGEID",**

**"SENDERID",**

**"RECEIVERID",**

**"MESSAGECONTENT",**

**"QUESTIONTYPEID",**

**"ANSWERCONTENT",**

**"ISREADQUESTIONBYRECEIVER",**

**"ISREADANSWERBYSENDER",**

**"MESSAGESENDDATE",**

**"ANSWERSENDDATE",**

**"LONGITUDE",**

**"LATITUDE",**

**"SENDERNAME",**

**"RECEIVERNAME",**

**"QUESTIONTYPE",**

**"SENDERADDRESS"**

**INTO lv\_ID,**

**lv\_SenderId,**

**lv\_ReceiverId,**

**lv\_MsgContent,**

**lv\_QtnTypId,**

**lv\_AnsContent,**

**lv\_IsReadReceiver,**

**lv\_IsSendSender,**

**lv\_MsgSendDate,**

**lv\_MsgAnswerDate,**

**lv\_Longitude,**

**lv\_Latitude,**

**lv\_SenderName,**

**lv\_ReceiverName,**

**lv\_questionType,**

**lv\_senderAddress**

**FROM :row;**

**IF (:lv\_MsgContent = '') THEN**

**error = SELECT 400 AS http\_status\_code,**

**'Invalid Decription' AS error\_message,**

**'Description Field cannot be empty' AS detail**

**FROM dummy;**

**ELSE**

**INSERT INTO "WORKMERK"."MESSAGE"("SENDERID", "RECEIVERID", "MESSAGECONTENT", "QUESTIONTYPEID", "ANSWERCONTENT", "ISREADQUESTIONBYRECEIVER", "ISREADANSWERBYSENDER", "MESSAGESENDDATE", "ANSWERSENDDATE", "LONGITUDE", "LATITUDE", "SENDERNAME", "RECEIVERNAME", "QUESTIONTYPE", "SENDERADDRESS") VALUES(lv\_SenderId, lv\_ReceiverId, lv\_MsgContent, lv\_QtnTypId, lv\_AnsContent, lv\_IsReadReceiver, lv\_IsSendSender, lv\_MsgSendDate, lv\_MsgAnswerDate,lv\_Longitude, lv\_Latitude, lv\_SenderName, lv\_ReceiverName, lv\_questionType, lv\_senderAddress);**

**END IF;**

**END;**

**PROCEDURE "WORKMERK"."workmerk.poc1.procedures::createAutoIdUser"(**

**IN row TT\_USER,**

**OUT error TT\_ERROR**

**)**

**LANGUAGE SQLSCRIPT**

**SQL SECURITY INVOKER**

**DEFAULT SCHEMA WORKMERK**

**AS**

**BEGIN**

**DECLARE lv\_ID INTEGER;**

**DECLARE lv\_FName STRING;**

**DECLARE lv\_LName STRING;**

**DECLARE lv\_Address1 STRING;**

**DECLARE lv\_Address2 STRING;**

**DECLARE lv\_CityId INTEGER;**

**DECLARE lv\_StateId INTEGER;**

**DECLARE lv\_Zipcode STRING;**

**DECLARE lv\_Phone STRING;**

**DECLARE lv\_Email STRING;**

**DECLARE lv\_Pwd STRING;**

**DECLARE lv\_DevideID STRING;**

**DECLARE lv\_DeviceTyp STRING;**

**DECLARE lv\_IsMobVer STRING;**

**DECLARE lv\_IsUsrAct STRING;**

**DECLARE lv\_RoleId INTEGER;**

**DECLARE lv\_DeviceToken String;**

**DECLARE lv\_Longitude String;**

**DECLARE lv\_Latitude String;**

**SELECT "USERID",**

**"FIRSTNAME",**

**"LASTNAME",**

**"ADDRESS1",**

**"ADDRESS2",**

**"CITYID",**

**"STATEID",**

**"ZIPCODE",**

**"PHONE",**

**"EMAIL",**

**"PASSWORD",**

**"DEVICEID",**

**"DEVICETYPE",**

**"ISMOBILEVERIFIED",**

**"ISUSERACTIVE",**

**"ROLEID",**

**"DEVICETOKEN",**

**"LONGITUDE",**

**"LATITUDE"**

**INTO lv\_ID,**

**lv\_FName,**

**lv\_LName,**

**lv\_Address1,**

**lv\_Address2,**

**lv\_CityId,**

**lv\_StateId,**

**lv\_Zipcode,**

**lv\_Phone,**

**lv\_Email,**

**lv\_Pwd,**

**lv\_DevideID,**

**lv\_DeviceTyp,**

**lv\_IsMobVer,**

**lv\_IsUsrAct,**

**lv\_RoleId,**

**lv\_DeviceToken,**

**lv\_Longitude,**

**lv\_Latitude**

**FROM :row;**

**IF (:lv\_FName = '') THEN**

**error = SELECT 400 AS http\_status\_code,**

**'Invalid Decription' AS error\_message,**

**'Description Field cannot be empty' AS detail**

**FROM dummy;**

**ELSE**

**INSERT INTO "WORKMERK"."USER"("FIRSTNAME", "LASTNAME", "ADDRESS1", "ADDRESS2", "CITYID", "STATEID", "ZIPCODE", "PHONE", "EMAIL", "PASSWORD", "DEVICEID", "DEVICETYPE", "ISMOBILEVERIFIED", "ISUSERACTIVE", "ROLEID", "DEVICETOKEN", "LONGITUDE", "LATITUDE")**

**VALUES(lv\_FName, lv\_LName, lv\_Address1, lv\_Address2, lv\_CityId, lv\_StateId, lv\_Zipcode, lv\_Phone, lv\_Email, lv\_Pwd, lv\_DevideID, lv\_DeviceTyp, lv\_IsMobVer, lv\_IsUsrAct, lv\_RoleId, lv\_DeviceToken, lv\_Longitude, lv\_Latitude);**

**END IF;**

**END;**

**PROCEDURE "WORKMERK"."workmerk.poc1.procedures::createAutoIdVehicleInspection"(**

**IN row TT\_VehicleInspection,**

**OUT error TT\_ERROR**

**)**

**LANGUAGE SQLSCRIPT**

**SQL SECURITY INVOKER**

**DEFAULT SCHEMA WORKMERK**

**AS**

**BEGIN**

**DECLARE lv\_VehicleInspectionId INTEGER;**

**DECLARE lv\_VehicleNo STRING;**

**DECLARE lv\_Date DAYDATE;**

**DECLARE lv\_Time SECONDTIME;**

**DECLARE lv\_Operator STRING;**

**DECLARE lv\_Exterior STRING;**

**DECLARE lv\_Interior STRING;**

**DECLARE lv\_MileageStart INTEGER;**

**DECLARE lv\_MileageEnd INTEGER;**

**DECLARE lv\_UserName STRING;**

**DECLARE lv\_UserID INTEGER;**

**DECLARE lv\_Longitude STRING;**

**DECLARE lv\_Latitude STRING;**

**DECLARE lv\_Address STRING;**

**SELECT "VEHICLEINSPECTIONID",**

**"VEHICLENO",**

**"DATE",**

**"TIME",**

**"OPERATOR",**

**"EXTERIOR",**

**"INTERIOR",**

**"MILEAGESTART",**

**"MILEAGEEND",**

**"USERNAME",**

**"USERID",**

**"LONGITUDE",**

**"LATITUDE",**

**"ADDRESS"**

**INTO lv\_VehicleInspectionId,**

**lv\_VehicleNo,**

**lv\_Date,**

**lv\_Time,**

**lv\_Operator,**

**lv\_Exterior,**

**lv\_Interior,**

**lv\_MileageStart,**

**lv\_MileageEnd,**

**lv\_UserName,**

**lv\_UserID,**

**lv\_Longitude,**

**lv\_Latitude,**

**lv\_Address**

**FROM :row;**

**IF (:lv\_VehicleNo = '') THEN**

**error = SELECT 400 AS http\_status\_code,**

**'Invalid Decription' AS error\_message,**

**'Description Field cannot be empty' AS detail**

**FROM dummy;**

**ELSE**

**INSERT INTO "WORKMERK"."VEHICLEINSPECTION"("VEHICLENO", "DATE", "TIME", "OPERATOR", "EXTERIOR", "INTERIOR", "MILEAGESTART", "MILEAGEEND", "USERNAME", "USERID", "LONGITUDE", "LATITUDE", "ADDRESS")**

**VALUES(lv\_VehicleNo, lv\_Date, lv\_Time, lv\_Operator, lv\_Exterior, lv\_Interior, lv\_MileageStart, lv\_MileageEnd, lv\_UserName, lv\_UserID, lv\_Longitude, lv\_Latitude, lv\_Address);**

**END IF;**

**END;**

**.xsaccess**

**{**

**"exposed": true,**

**"authentication": [{**

**"method": "Form"**

**}],**

**"authorization": [],**

**"mime\_mapping": [{**

**"extension": "jpg",**

**"mimetype": "image/jpeg"**

**}],**

**"force\_ssl": false,**

**"enable\_etags": true,**

**"prevent\_xsrf": false,**

**"anonymous\_connection": null,**

**"cors": [{**

**"enabled": false**

**}],**

**"cache\_control": "no-cache, no-store",**

**"default\_file": "index.html"**

**}**

**.xsapp**

**{}**

**.xsprivileges**

**{**

**"privileges": [**

**{"name": "Execute","description": "Execute"}**

**]**

**}**

**Services.xsodata**

**service {**

**"WORKMERK"."USER" as "USER" create using "workmerk.poc1.procedures::createAutoIdUser";**

**"WORKMERK"."USERROLE" as "USERROLE";**

**"WORKMERK"."QUESTIONTYPE" as "QUESTIONTYPE";**

**"WORKMERK"."MESSAGE" as "MESSAGE" create using "workmerk.poc1.procedures::createAutoIdMessage";**

**"WORKMERK"."ATTACHMENT" as "ATTACHMENT" create using "workmerk.poc1.procedures::createAutoIdAttachment";**

**"WORKMERK"."LOG" as "LOG" create using "workmerk.poc1.procedures::createAutoIdLog";**

**"WORKMERK"."VEHICLEINSPECTION" as "VEHICLEINSPECTION" create using "workmerk.poc1.procedures::createAutoIdVehicleInspection";**

**}**