CREATE TABLE BANK(

bank\_code INTEGER,

bank\_name VARCHAR(40),

bank\_headquater VARCHAR(40),

CONSTRAINT bank\_code\_pk PRIMARY KEY(bank\_code)

);

-- KEEPS DIFFERENT BRANCHES OF A BANK

CREATE TABLE BANKBRANCH(

bank\_branch\_code INTEGER,

bank\_code INTEGER,

bank\_branch\_address VARCHAR(40),

bank\_branch\_phone VARCHAR(40),

bank\_branch\_email VARCHAR(40),

CONSTRAINT bank\_pk PRIMARY KEY(bank\_branch\_code),

CONSTRAINT bankbranch\_bank\_fk FOREIGN KEY(bank\_code) REFERENCES BANK(bank\_code)

);

-- DIFFERENT QUALIFICATIONS THAT MIGHT BE POSSESSED BY A CONTRACTOR

CREATE TABLE QUALIFICATION(

qualification\_code INTEGER IDENTITY(1,1),

qualification\_name VARCHAR(40),

CONSTRAINT qualification\_code\_pk PRIMARY KEY(qualification\_code)

);

CREATE TABLE REGION (

region\_code INTEGER,

region\_name VARCHAR(40),

CONSTRAINT region\_pk PRIMARY KEY (region\_code)

);

-- DIFFERENT TYPES OF DOCUMENTS THAT MIGHT BE PRESENTED FOR IDENTIFICATION BY CONTRACTORS

CREATE TABLE DOCUMENT(

document\_code INTEGER IDENTITY(100,1),

document\_name VARCHAR(40),

CONSTRAINT document\_pk PRIMARY KEY(document\_code)

);

-- KEEPS ALL THE CONTRACT TYPES THAT ARE AVAILABE IN WAEC

CREATE TABLE CONTRACT (

contract\_code INTEGER IDENTITY(1,1),

contract\_type VARCHAR(40),

CONSTRAINT contract\_pk PRIMARY KEY(contract\_code)

);

CREATE TABLE SCHOOL(

scl\_code VARCHAR(10),

scl\_name VARCHAR(40),

sch\_level VARCHAR(40),

scl\_region INTEGER,

CONSTRAINT junsch\_pk PRIMARY KEY(scl\_code),

CONSTRAINT sch\_region\_fk FOREIGN KEY(scl\_region) REFERENCES REGION(region\_code)

);

CREATE TABLE CONTRACTOR(

contractor\_code INTEGER,

contract\_code INTEGER,

first\_name VARCHAR(40),

middle\_name VARCHAR(40),

last\_name VARCHAR(40),

dob DATE,

photo VARBINARY(MAX),

registration\_date DATE ,

nationality VARCHAR(40),

document INTEGER,

phone VARCHAR(20), -- newly added

address VARCHAR(20), -- newly added

status VARCHAR(40), --ACTIVE OR INACTIVE

field\_of\_study VARCHAR(40),

contractor\_region INTEGER,

CONSTRAINT contractor\_pk PRIMARY KEY(contractor\_code),

CONSTRAINT contract\_detail\_fk FOREIGN KEY(contract\_code) REFERENCES CONTRACT(contract\_code),

CONSTRAINT contractor\_document\_fk FOREIGN KEY(document) REFERENCES DOCUMENT(document\_code),

CONSTRAINT contractor\_region\_fk FOREIGN KEY(contractor\_region) REFERENCES REGION(region\_code)

);

CREATE TABLE CONTRACTORBANK(

contractor\_bank INTEGER,

contractor INTEGER,

account\_Number VARCHAR(40),

account\_name VARCHAR(40),

bban VARCHAR(40),

CONSTRAINT contrabank\_pk PRIMARY KEY(contractor\_bank, contractor, account\_Number),

CONSTRAINT contractor\_bank\_fk FOREIGN KEY(contractor\_bank) REFERENCES BANK(bank\_code),

CONSTRAINT contrabank\_contractor\_INTEGER\_fk FOREIGN KEY(contractor) REFERENCES CONTRACTOR(contractor\_code)

);

--KEEPS CONTRACTOR QUALIFICATIONS

CREATE TABLE CONTRACTORQUALIFICATION(

contra\_qual\_code INTEGER,

contractor\_id INTEGER,

qualification\_date DATE,

description VARCHAR(300),

CONSTRAINT contractor\_qual\_pk PRIMARY KEY(contra\_qual\_code, contractor\_id),

CONSTRAINT contra\_qual\_fk FOREIGN KEY(contra\_qual\_code) REFERENCES QUALIFICATION(qualification\_code),

CONSTRAINT contractor\_id\_fk FOREIGN KEY(contractor\_id) REFERENCES CONTRACTOR(contractor\_code)

);

CREATE TABLE CONTRACTORSCHOOL(

contractor\_code INTEGER,

contractor\_sch VARCHAR(10),

contracotr\_subject VARCHAR(10), -- newly added

join\_date DATE,

status VARCHAR(20), -- this indentified whether the school is current or previous

CONSTRAINT contractor\_school\_pk PRIMARY KEY(contractor\_code, contractor\_sch),

CONSTRAINT contractor\_school\_fk FOREIGN KEY(contractor\_sch) REFERENCES SCHOOL(scl\_code),

CONSTRAINT contractor\_code\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code)

);

CREATE TABLE CONTRACTORTUTORIALCLASS(

contractorID INTEGER,

schoolID VARCHAR(10),

start\_date DATE,

status VARCHAR(10), -- whether current or previous

CONSTRAINT tutorial\_class\_pk PRIMARY KEY(contractorID, schoolID),

CONSTRAINT tutorial\_class\_contractor FOREIGN KEY(contractorID) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT tutorial\_class\_sch FOREIGN KEY(schoolID) REFERENCES SCHOOL(scl\_code)

);

CREATE TABLE EXAMDIET(

diet\_code VARCHAR(10),

dietName VARCHAR(20),

CONSTRAINT examDiet\_dietCode\_pk PRIMARY KEY(diet\_code),

);

CREATE TABLE EXAMCATEGORY(

exam\_category\_code VARCHAR(10),

exam\_category\_name VARCHAR(100), -- WASSCE, GABECE OR NAT

diet\_code VARCHAR(10), -- INDICATES THE TIME OF THE EXAM

CONSTRAINT examCategory\_pk PRIMARY KEY(exam\_category\_code),

CONSTRAINT examCategoryDiet\_fk FOREIGN KEY(diet\_code) REFERENCES EXAMDIET(diet\_code),

);

CREATE TABLE BLACKLIST(

contractor\_code INTEGER,

dates DATE,

description VARCHAR(500),

CONSTRAINT blacklist\_examiner\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code)

);

--NOW I AM COMING TO SPECIFY TABLES FOR SUBJECTS AND EXAMS

CREATE TABLE SUBJECT(

subject\_code VARCHAR(10),

subject\_name VARCHAR(60),

exam\_category\_code VARCHAR(10),

iscore\_subject VARCHAR(5),

CONSTRAINT cabecesubject\_pk PRIMARY KEY(subject\_code, exam\_category\_code),

CONSTRAINT SUBJECT\_exam\_category\_code\_fk FOREIGN KEY(exam\_category\_code) REFERENCES EXAMCATEGORY(exam\_category\_code)

);

-- THIS TABLE RECORDS ALL THE SUBJECT PAPERS AND THEIR RATES

CREATE TABLE SUBJECTPAPER(

subject\_paper\_code VARCHAR(10), -- THEORY, ESSAY ETC

subject\_code VARCHAR(10), -- THE EXAM SUBJECT

exam\_category\_code VARCHAR(10), -- SUBJECT LEVEL (GABECE, WASSCE OR NAT)

subject\_paper\_rate DECIMAL(8,2),

CONSTRAINT gsub\_paper\_pk PRIMARY KEY(subject\_paper\_code),

CONSTRAINT gsub\_paper\_fk FOREIGN KEY(subject\_code, exam\_category\_code) REFERENCES SUBJECT(subject\_code, exam\_category\_code),

);

CREATE TABLE COURSEWORK(

subject\_id VARCHAR(10),

Cwrok\_exam\_category VARCHAR(10), --THIS INDICATE IF PRACTICAL IS FOR WASSCE OR GABECE

Cwork\_examDiet VARCHAR(10),

Cwork\_name VARCHAR(20),

courseword\_rate DECIMAL(8,2),

remarks VARCHAR(500),

CONSTRAINT coursework\_pk PRIMARY KEY(subject\_id),

CONSTRAINT coursework\_subject\_fk FOREIGN KEY(subject\_id, Cwrok\_exam\_category) REFERENCES SUBJECT(subject\_code, exam\_category\_code),

CONSTRAINT courseWork\_SubjectLevel\_fk FOREIGN KEY(Cwrok\_exam\_category) REFERENCES EXAMCATEGORY(exam\_category\_code),

CONSTRAINT courseWork\_ExamDiet\_fk FOREIGN KEY(Cwork\_examDiet) REFERENCES EXAMDIET(diet\_code),

);

CREATE TABLE PRACTICAL(

subject\_id VARCHAR(10),

exam\_category\_code VARCHAR(10), --THIS INDICATE IF PRACTICAL IS FOR WASSCE OR GABECE

diet\_code VARCHAR(10), --IDENTIFIES (MAY/JUNE OR NOVEMBER/DECEMBER)

practical\_name VARCHAR(20),

practical\_rate DECIMAL(8,2),

remarks VARCHAR(500),

CONSTRAINT practical\_work\_pk PRIMARY KEY(subject\_id),

CONSTRAINT practical\_dietCode\_fk FOREIGN KEY(diet\_code) REFERENCES EXAMDIET(diet\_code),

CONSTRAINT practical\_work\_subject\_fk FOREIGN KEY(subject\_id, exam\_category\_code) REFERENCES SUBJECT(subject\_code, exam\_category\_code)

);

CREATE TABLE ORAL(

oral\_subject VARCHAR(10),

oral\_exam\_category VARCHAR(10),

oral\_exam\_diet VARCHAR(10),

oral\_description VARCHAR(50),

CONSTRAINT oral\_oralSubject\_pk PRIMARY KEY(oral\_subject),

CONSTRAINT oralSubject\_subject FOREIGN KEY(oral\_subject, oral\_exam\_category) REFERENCES SUBJECT(subject\_code, exam\_category\_code),

CONSTRAINT oral\_category\_fk FOREIGN KEY(oral\_exam\_category) REFERENCES EXAMCATEGORY(exam\_category\_code),

CONSTRAINT oral\_OralExamDiet\_fk FOREIGN KEY(oral\_exam\_diet) REFERENCES EXAMDIET(diet\_code),

);

--ALLOCATE COURSEWORK TO A CONTRACTOR

CREATE TABLE COURSEWORKALLOCATION(

SN INTEGER,

contractor\_code INTEGER,

Csubject\_code VARCHAR(10),

number\_of\_works INTEGER,

dates DATE,

CONSTRAINT courseWorkAllocation\_SN\_pk PRIMARY KEY(SN),

CONSTRAINT courseWorkAllocation\_Contractor\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT courseWork\_courseName\_fk FOREIGN KEY(Csubject\_code) REFERENCES COURSEWORK(subject\_id),

);

--ALLOCATE PRACTICAL TO A CONTRACTOR

CREATE TABLE PRACTICALALLOCATION(

SN INTEGER,

contractor\_code INTEGER,

practical\_subject VARCHAR(10),

number\_of\_works INTEGER,

dates DATE,

CONSTRAINT practicalAllocation\_SN\_pk PRIMARY KEY(SN),

CONSTRAINT practicalAllocation\_Contractor\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT practical\_courseName\_fk FOREIGN KEY(practical\_subject) REFERENCES PRACTICAL(subject\_id),

);

--ALLOCATE ORAL TO A CONTRACTOR

CREATE TABLE ORALALLOCATION(

SN INTEGER,

contractor\_code INTEGER,

oral\_subject VARCHAR(10),

number\_of\_works INTEGER,

dates DATE,

CONSTRAINT oralAllocation\_SN\_pk PRIMARY KEY(SN),

CONSTRAINT oralAllocation\_Contractor\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT oral\_courseName\_fk FOREIGN KEY(oral\_subject) REFERENCES ORAL(oral\_subject),

);

CREATE TABLE EXAMCENTER(

region\_code INTEGER,

centre\_NUMBER VARCHAR(10),

centre\_name VARCHAR(40),

centre\_address VARCHAR(40),

contact\_person VARCHAR(40),

phone VARCHAR(40),

email VARCHAR(40),

CONSTRAINT examcenter\_pk PRIMARY KEY(centre\_NUMBER),

CONSTRAINT examcenter\_region\_fk FOREIGN KEY(region\_code) REFERENCES REGION(region\_code),

);

-- SCRIPT ALLOCATION TO EXAMINERS FOR MARKING

CREATE TABLE SCRIPTALLOCATION(

center\_code VARCHAR(10),

subjectCode VARCHAR(10),

exam\_category\_code VARCHAR(10),

examinerID INTEGER,

subjectPaperCode VARCHAR(10),

numOfScript INTEGER,

numberOfScriptMarked INTEGER,

allocatioction\_date DATE,

CONSTRAINT seniorScriptMarking\_pk PRIMARY KEY(center\_code, subjectCode, examinerID),

CONSTRAINT schoolid\_fk FOREIGN KEY(center\_code) REFERENCES EXAMCENTER(centre\_NUMBER),

CONSTRAINT subjectMarked\_fk FOREIGN KEY(subjectCode, exam\_category\_code) REFERENCES SUBJECT(subject\_code, exam\_category\_code),

CONSTRAINT Script\_examiner\_fk FOREIGN KEY(examinerID) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT ScriptAllocation\_examPaperCode\_fk FOREIGN KEY(subjectPaperCode) REFERENCES SUBJECTPAPER(subject\_paper\_code),

);

-- ALLOWANCE PAYMENT TO DIFFERENT CONTRACTORS BASED ON THE LOCATION AND STATUS

CREATE TABLE TOWNSTATUS(

status\_code INTEGER,

town\_description VARCHAR(40),

CONSTRAINT townstatus\_pk PRIMARY KEY(status\_code)

);

CREATE TABLE TOWN(

town\_code INTEGER,

town\_name VARCHAR(40),

town\_location INTEGER,

CONSTRAINT town\_town\_code\_pk PRIMARY KEY(town\_code),

CONSTRAINT townLocation\_fk FOREIGN KEY(town\_location) REFERENCES TOWNSTATUS(status\_code),

);

CREATE TABLE VEHICLE(

vehicle\_code INTEGER,

vehicle\_type VARCHAR(40),

--vehicle\_rate DECIMAL(8,2),

CONSTRAINT vehicle\_pk PRIMARY KEY(vehicle\_code)

);

CREATE TABLE VIHECLEREGISTRATION(

vehicle\_code INTEGER,

insurance VARCHAR(20),

vehicleNumber VARCHAR(20),

contractor\_code INTEGER,

CONSTRAINT VehicleRegistration\_VehicleCode\_Pk PRIMARY KEY(vehicleNumber),

CONSTRAINT VihecleReg\_contractor\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT VihecleReg\_vehicleCode\_fk FOREIGN KEY(vehicle\_code) REFERENCES VEHICLE(vehicle\_code),

);

CREATE TABLE VEHICLEREFUNDRATE(

SN INTEGER,

townCode INTEGER,

vehicle\_code INTEGER,

refundRate DECIMAL(8,2),

CONSTRAINT vehiclRefundRate\_towncode\_pk PRIMARY KEY(townCode),

CONSTRAINT VihecleRefundRate\_towncode\_fk FOREIGN KEY(townCode) REFERENCES TOWN(town\_code),

CONSTRAINT VihecleRefundRate\_vehicleCode\_fk FOREIGN KEY(vehicle\_code) REFERENCES VEHICLE(vehicle\_code),

);

CREATE TABLE VEHICLEREFUND(

towncode INTEGER, -- specifies the town

vehicle\_code INTEGER,

vehicleNumber VARCHAR(20), -- IDENTIFIES THE VEHICLE ITSELF

contractor\_code INTEGER,

startDate DATE,

endDate DATE,

amountEarned DECIMAL(8,2),

CONSTRAINT vehicle\_refund\_pk PRIMARY KEY(vehicleNumber),

CONSTRAINT VihecleRefund\_towncode\_fk FOREIGN KEY(towncode) REFERENCES VEHICLEREFUNDRATE(townCode),

CONSTRAINT vehicle\_code\_fk FOREIGN KEY(vehicleNumber) REFERENCES VIHECLEREGISTRATION(vehicleNumber),

CONSTRAINT contractor\_vehicleRefund\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT vehicleRefund\_vehicleCode\_fk FOREIGN KEY(vehicle\_code) REFERENCES VEHICLE(vehicle\_code),

);

CREATE TABLE TRANSPORTFARERATE(

towncode INTEGER,

fareRate DECIMAL(8,2),

CONSTRAINT transportfare\_pk PRIMARY KEY(towncode, fareRate),

CONSTRAINT TransportFare\_townCode\_fk FOREIGN KEY(towncode) REFERENCES TOWN(town\_code),

);

CREATE TABLE OVERNIGHTALLOWANCE(

townCode INTEGER,

contractor\_code INTEGER,

overnight\_rate DECIMAL(8,2),

startDate DATE,

endDate DATE,

amountEarned DECIMAL(8,2),

CONSTRAINT townForOvernightAllowance\_fk FOREIGN KEY(townCode) REFERENCES TOWN(town\_code),

CONSTRAINT contractor\_overnightAllowance\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

);

CREATE TABLE TRANSPORTREFUND(

townCode INTEGER,

contractor\_code INTEGER,

startDate DATE,

endDate DATE,

amount\_earned DECIMAL(8,2),

CONSTRAINT TransportRefund\_towncode\_fk FOREIGN KEY(townCode) REFERENCES TOWN(town\_code),

CONSTRAINT TrasportRefund\_contractor\_code\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

);

--TABLES TO CAPTURE DIFFRENT MEETINGS AND ENTITLEMENT FOR EACH

CREATE TABLE MEETINGS(

meeting\_code INTEGER,

meeting\_type VARCHAR(40), -- DIFFERENT TYPES OF MEETINGS (VETTING, COORDIANTION, REPORT)

exam\_category\_code VARCHAR(10), -- TALS ABOUT THE TYPE OF EXAM (GABECE, WASSCE ETC)

meeting\_rate DECIMAL(8,2),

lunch\_allowance DECIMAL(8,2), -- LUNCH ALLOWANCE

refreshment DECIMAL(8,2), -- REFRESHMENT ALLOWANCE

CONSTRAINT meeting\_meetingCode\_pk PRIMARY KEY(meeting\_code),

CONSTRAINT meeting\_exam\_category\_code FOREIGN KEY(exam\_category\_code) REFERENCES EXAMCATEGORY(exam\_category\_code),

);

CREATE TABLE MEETINGDETAILS( -- THIS WILL CAPTURE ALL TYPES OF MEETINGS -- VETTING AND COORDINATION MEETINGS AND REPORT

MSN INTEGER, -- MEETING SERIAL NUMBER

meeting\_code INTEGER,

mStart\_Date DATE,

mEnd\_date DATE,

exam\_category\_code VARCHAR(10),

subject\_code VARCHAR(10),

CONSTRAINT meetingDetails\_pk PRIMARY KEY(MSN),

CONSTRAINT meetingDetails\_meetingCode\_fk FOREIGN KEY(meeting\_code) REFERENCES MEETINGS(meeting\_code),

CONSTRAINT meeting\_examCategory\_fk FOREIGN KEY(exam\_category\_code) REFERENCES EXAMCATEGORY(exam\_category\_code),

CONSTRAINT meeting\_subjectCode\_fk FOREIGN KEY(subject\_code, exam\_category\_code) REFERENCES SUBJECT (subject\_code, exam\_category\_code),

);

CREATE TABLE MEETINGATTENDANCE(

meeting\_code INTEGER,

contractor\_code INTEGER, -- THIS IDENTIFIED THE ID OR CODE ASSIGNED TO THE CONTRACTOR

time\_in datetime,

time\_out datetime,

CONSTRAINT meeting\_attendance\_fk FOREIGN KEY(meeting\_code) REFERENCES MEETINGDETAILS(MSN),

);

--INVIGILATION

CREATE TABLE INVIGILATION(

centerNo VARCHAR(10),

invigilator\_code INTEGER,

subject\_paper\_code VARCHAR(10),

invigilation\_Date DATE,

amountEarned DECIMAL(8,2),

CONSTRAINT contractor\_invigilation\_fk FOREIGN KEY(invigilator\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT invigilator\_centerNo\_fk FOREIGN KEY(centerNo) REFERENCES EXAMCENTER(centre\_NUMBER),

CONSTRAINT INVIGILATOR\_subjectPaperCode\_fk FOREIGN KEY(subject\_paper\_code) REFERENCES SUBJECTPAPER(subject\_paper\_code),

);

CREATE TABLE REPORTRATE(

report\_rate\_code VARCHAR(10),

report\_rate DECIMAL(8,2),

report\_description VARCHAR(20),

CONSTRAINT reportRateCode\_pk PRIMARY KEY(report\_rate\_code),

);

CREATE TABLE REPORT(

contractor\_code INTEGER,

diet\_code VARCHAR(10),

subject\_paper\_code VARCHAR(10),

exam\_category\_code VARCHAR(10),

report\_rate\_code VARCHAR(10),

CONSTRAINT REPORT\_contractorCode\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT REPORT\_deitCode\_fk FOREIGN KEY(diet\_code) REFERENCES EXAMDIET(diet\_code),

CONSTRAINT REPORT\_subjectPaperCode\_fk FOREIGN KEY(subject\_paper\_code) REFERENCES SUBJECTPAPER(subject\_paper\_code),

CONSTRAINT REPORT\_examCategory\_fk FOREIGN KEY(exam\_category\_code) REFERENCES EXAMCATEGORY(exam\_category\_code),

CONSTRAINT REPORT\_reportRateCode\_fk FOREIGN KEY(report\_rate\_code) REFERENCES REPORTRATE(report\_rate\_code)

);

CREATE TABLE SUPERVISION(

centerNo VARCHAR(10),

examiner\_code INTEGER,

subject\_paper\_code VARCHAR(10),

supervision\_Date DATE,

amountEarned DECIMAL(8,2),

CONSTRAINT contractor\_supervision\_fk FOREIGN KEY(examiner\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT supervision\_centerNo\_fk FOREIGN KEY(centerNo) REFERENCES EXAMCENTER(centre\_NUMBER),

CONSTRAINT supervision\_subjetPaperCode\_fk FOREIGN KEY(subject\_paper\_code) REFERENCES SUBJECTPAPER(subject\_paper\_code),

);

CREATE TABLE VETTING(

contractor\_code INTEGER,

subject\_paper\_code VARCHAR(10),

number\_of\_script INTEGER,

centerNo VARCHAR(10),

dates DATE,

CONSTRAINT VETTING\_contractorCode\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT VETTING\_subjetPaperCode\_fk FOREIGN KEY(subject\_paper\_code) REFERENCES SUBJECTPAPER(subject\_paper\_code),

CONSTRAINT VETTING\_centerNo\_fk FOREIGN KEY(centerNo) REFERENCES EXAMCENTER(centre\_NUMBER),

);

CREATE TABLE DEPARTMENT(

dept\_id VARCHAR(5),

dept\_name VARCHAR(20),

CONSTRAINT departmentID\_pk PRIMARY KEY(dept\_id),

);

CREATE TABLE WAECOFFICER(

officer\_id INTEGER,

officerFN VARCHAR(40),

officerMN VARCHAR(40),

officerLN VARCHAR(40),

phone\_number VARCHAR(40),

email VARCHAR(20),

dept\_id VARCHAR(5),

ranks VARCHAR(20),

username VARCHAR(20),

password VARCHAR(20),

CONSTRAINT subject\_officer\_pk primary KEY(officer\_id),

CONSTRAINT waecOfficer\_department\_fk FOREIGN KEY(dept\_id) REFERENCES DEPARTMENT(dept\_id),

);

CREATE TABLE CLAIMANT(

claim\_type VARCHAR(20),

contractor\_code INTEGER,

total\_fee DECIMAL(8,2),

checked\_by INTEGER,

confirmed\_by INTEGER,

approved\_by INTEGER,

clain\_date DATE,

CONSTRAINT CLAIMANT\_contractorCode\_fk FOREIGN KEY(contractor\_code) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT CLAIMANT\_checkedBy\_fk FOREIGN KEY(checked\_by) REFERENCES WAECOFFICER(officer\_id),

CONSTRAINT CLAIMANT\_confirmedBy\_fk FOREIGN KEY(confirmed\_by) REFERENCES WAECOFFICER(officer\_id),

CONSTRAINT CLAIMANT\_approved\_byBy\_fk FOREIGN KEY(approved\_by) REFERENCES WAECOFFICER(officer\_id),

);

CREATE TABLE USERS(

users\_id INTEGER,

user\_role INTEGER,

subject\_paper VARCHAR(10),

username VARCHAR(20) UNIQUE NOT NULL,

password VARCHAR(20) NOT NULL,

CONSTRAINT users\_userID\_pk PRIMARY KEY(users\_id),

CONSTRAINT users\_userID\_fk FOREIGN KEY(users\_id) REFERENCES CONTRACTOR(contractor\_code),

CONSTRAINT users\_subjectPaper\_fk FOREIGN KEY(subject\_paper) REFERENCES SUBJECTPAPER(subject\_paper\_code)

);

CREATE TABLE SETUP( -- THIS TABLE SEEKS TO VALIDATE THE LENGTH OF CANDIDATE CODES

exam\_category\_code VARCHAR(10),

Cnadidate\_num\_length INTEGER,

exam\_paper\_length INTEGER,

CONSTRAINT setup\_examCategory\_fk FOREIGN KEY(exam\_category\_code) REFERENCES EXAMCATEGORY(exam\_category\_code),

);

--MARK ENTRY

CREATE TABLE MARKENTRY(

subject\_paper\_code VARCHAR(10),

marks DECIMAL(8,2),

keyed\_count INTEGER,

remarks VARCHAR(500),

CONSTRAINT markEntry\_subjectPaper\_fk FOREIGN KEY(subject\_paper\_code) REFERENCES SUBJECTPAPER(subject\_paper\_code),

); -- THERE SHOULD BE A CHECKING THAT VALIDATES THE EXAMINER ENTERING MARKS IS THE ONE HAS BEEN ALLOCATED THE SCRIPTS

CREATE TABLE ITEMWRITING(

subject\_code VARCHAR(10),

exam\_category\_code VARCHAR(10),

subject\_paper\_code VARCHAR(10),

examiner\_code INTEGER,

item\_count INTEGER,

item\_rate DECIMAL(8,2),

dates DATE,

CONSTRAINT itemMarking\_subjectCode\_fk FOREIGN KEY(subject\_code, exam\_category\_code) REFERENCES SUBJECT(subject\_code,exam\_category\_code),

CONSTRAINT itemMarking\_examCategory\_fk FOREIGN KEY(exam\_category\_code) REFERENCES EXAMCATEGORY(exam\_category\_code),

CONSTRAINT itemMarking\_subjectPaper\_fk FOREIGN KEY(subject\_paper\_code) REFERENCES SUBJECTPAPER(subject\_paper\_code),

CONSTRAINT itemMaking\_examiner\_fk FOREIGN KEY(examiner\_code) REFERENCES CONTRACTOR(contractor\_code)

);

CREATE TABLE LOGS(

SN INTEGER IDENTITY(1,1),

userID INTEGER,

dates DATETIME,

activity VARCHAR(MAX), -- TO CAPTURE ALL THE DETAILS OF ACTIVITY

CONSTRAINT logs\_serialNo\_pk primary KEY(SN),

CONSTRAINT logs\_userID\_fk FOREIGN KEY(userID) REFERENCES USERS(users\_id),

);