**RDBMS CONCEPTS** (**CONSTRAINTS, KEYS**)

**RDBMS RELATIONAL DATABASE MANAGEMENT SYSTEM**.

A SPECIAL TYPE OF DBMS TO MANAGE DATABASES THAT INVOLVE RELATIONS ACROSS THE TABLES.

EXAMPLE: STUDENTS TABLE RELATED TO COURSES TABLE

EXAMPLE: MEDICATIONS TABLE RELATED TO PATIENTS TABLE

**CONSTRAINTS** CONDITIONS DEFINED ON TABLES FOR VALID DATA STORAGE.

EXAMPLE: COURSE DURATION SHOULD BE ABOVE 60 DAYS.

**KEYS** CONDITIONS DEFINED ON TABLES TO ENSURE UNIQUE VALUES

EXAMPLE: COURSE NAME SHOULD BE UNIQUE.

**TYPES OF CONSTRAINTS**:

1. **NOT NULL** : This column does not allow NULLS. Means, mandatory column
2. **NULL** : This column allow NULLS. Means, column value is optional
3. **UNIQUE** : This column does not allow duplicates. Allows up to 1 null value
4. **PRIMARY KEY** : This column does not allow duplicates. Does not allow null value
5. **FOREIGN KEY** : This is used to **reference** or **link** one table to another table.

Database that contain related tables are called Relational Database. Such DBMS is called **RDBMS**.

**Assume an Amazon or FlipKart Database:**

**Products** Table + **Customers** Table.

**Sales Table** should depend on **Products** & **Customers** Tables

1. **CHECK CONSTRAINT** : Used to specify conditions on column values.
2. **DEFAULT CONSTRAINT**: Used to insert a default value to the column.
3. **IDENTITY PROPERTY** : Used to insert automated **sequence** of values. Serial Numbers.

**SYNTAX**:

CREATE TABLE <<TABLENAME>>

(

COLUMNAME DATATYPE NOT NULL CONSTRAINT,

COLUMNAME DATATYPE NULL CONSTRAINT,

COLUMNAME DATATYPE CONSTRAINT,

COLUMNAME DATATYPE CONSTRAINT1 CONSTRAINT2 CONSTRAINT3

)

**University Database : Scenario**

**You are Database Operator in a University Admin Department.**

**You are requested to design / verify a plan for Data storage that should involve:**

1. **Students Data**
2. **Staff Data**
3. **Courses Data**

**Note: Students Data should be unique. Staff Data should be unique.**

**Note: Students & Staff should be enrolled / recruited for an available course.**

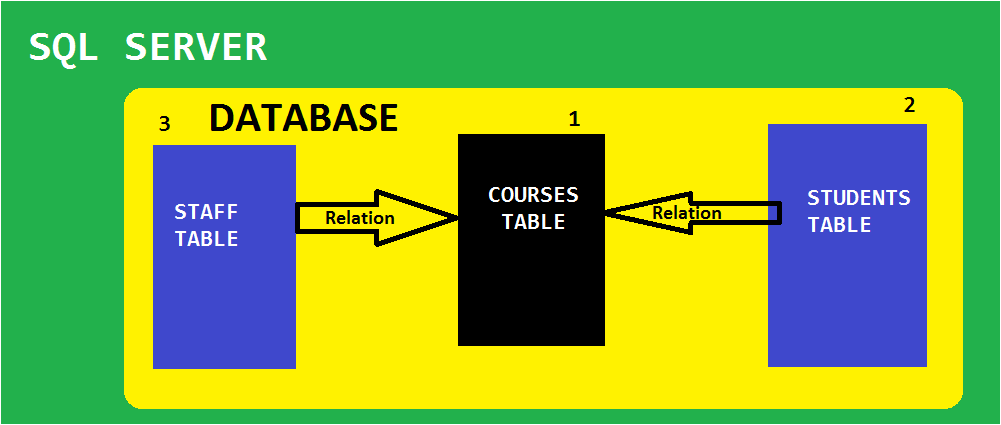
**Solution :**

**We can implement above Database Design requirement using “Constraints” & “Keys”.**

**We need to design the database with total of THREE tables:**

**Step 1: Create Courses Table first.**

**Step 2: Create Students and Staff. Relate these tables to above courses table.**

****

**-- QUERY 1:**

CREATE DATABASE UNIVERSITY\_DATABASE

**-- QUERY 2:**

USE UNIVERSITY\_DATABASE

**-- QUERY 3:**

CREATE TABLE TBLCOURSES

(

COURSE\_ID INT PRIMARY KEY,

COURSE\_NAME VARCHAR(30) NOT NULL,

COURSE\_DUR INT CHECK (COURSE\_DUR = 120 OR COURSE\_DUR = 180)

)

**-- QUERY 4:**

INSERT INTO TBLCOURSES VALUES (101, **'COMPUTERS'**, 120)

INSERT INTO TBLCOURSES VALUES (102, **'ROBOTICS'**, 180)

**-- QUERY 5:**

SELECT \* FROM TBLCOURSES

INSERT INTO TBLCOURSES VALUES (102,'CIVIL', 180) -- ERROR.ID IS DUPLICATED

INSERT INTO TBLCOURSES VALUES (103, NULL, 180) -- ERROR.COURSE NAME CANNOT BE NULL

INSERT INTO TBLCOURSES VALUES (103,'CIVIL',-180) -- ERROR.INVALID DURATION

**-- QUERY 6:**

CREATE TABLE TBLSTUDENTS

(

STD\_ID INT UNIQUE,

STD\_NAME VARCHAR(30),

STD\_AGE INT CHECK (STD\_AGE >= 18),

STD\_COURSE\_ID INT **REFERENCES** TBLCOURSES (COURSE\_ID)

)

**-- QUERY 7:**

INSERT INTO TBLSTUDENTS VALUES (1001, 'SAI', 34, 101)

INSERT INTO TBLSTUDENTS VALUES (1002, 'JON', 34, 101)

INSERT INTO TBLSTUDENTS VALUES (1003, 'JOHNE', 34, 102)

INSERT INTO TBLSTUDENTS VALUES (1004, 'JOHNY', 34, 102)

**-- QUERY 8:**

SELECT \* FROM TBLSTUDENTS

**-- QUERY 9: HOW TO TEST UNIQUE KEY?**

INSERT INTO TBLSTUDENTS VALUES (1004, 'JENY', 34, 102)

**-- QUERY 10: HOW TO TEST FOREIGN KEY?**

INSERT INTO TBLSTUDENTS VALUES (1005, 'JENY', 34, 103)

SELECT \* FROM TBLSTUDENTS

**-- QUERY 11:**

CREATE TABLE TBLSTAFF

(

STF\_ID INT IDENTITY(100001, 1),

STF\_NAME VARCHAR(30) NOT NULL,

STF\_GENDER CHAR CHECK (STF\_GENDER IN ('M','F')) DEFAULT 'F',

STF\_COURSE\_ID INT **REFERENCES** tblcourses(course\_id)

)

**-- QUERY 10:**

INSERT INTO TBLSTAFF VALUES ('JOHN', 'M', 101)

INSERT INTO TBLSTAFF VALUES ('JOHNY', 'M', 101)

**-- QUERY 11:**

SELECT \* FROM TBLSTAFF

**-- QUERY 12: HOW TO INSERT DEFAULT VALUES INTO STF\_GENDER COLUMN?**

RULE : WE NEED TO SUPPLY REMAINING COLUMNS AND THEIR VALUES.

MISSING COLUMNS WILL TAKE DEFAULT VALUES.

INSERT INTO TBLSTAFF(STF\_NAME, STFCOURSE\_ID) VALUES ('AMIN', 102)

**-- QUERY 13:**

SELECT \* FROM TBLSTAFF

**DATABASE DIAGRAMS** :

USED TO VERIFY THE RELATIONS BETWEEN TABLES IN THE DATABASE.

ALSO CALLED AS **ENTITY RELATIONSHIP** DIAGRAM (**ER DIAGRAM**).

**IN SSMS:**

CONNECT TO YOUR LOCAL SQL SERVER >EXPAND DATABASES FOLDER > GO TO ABOVE CREATED UNIVERSITY DATABASE > EXPAND. YOU SEE DATABASE DIAGRAMS FOLDER.

RIGHT CLICK "**DATABASE DIAGRAM**" FOLDER > NEW DATABASE DIAGRAM>

SELECT REQUIRED TABLES > ADD. THEN WE SEE THE RELATIONAL DIAGRAM (ER DIGRAM).

**PARENT** TABLES ARE IDENTIFIED BY A SMALL **YELLOW** KEY SYMBOL.

CHILD TABLES ARE IDENTIFIED BY A SMALL WHILTE LINK (CHAIN) SYMBOL.

**IN YOUR LATER CLASSES:** SELF REFERENCING KEYS

COMPOSITE KEYS

CASCADING KEYS…