Servicii Educationale Online

(“Aplicatie pentru evidenta participarii la cursuri online pentru o companie ce ofera servicii educationale”)

De Stoica Vlad Ionut

Cand vine vorba de Servicii educationale, online sau nu, sunt trei entitati cruciale care vin in minte: Studentii, Profesorii si Cursurile insasi.

Pe langa acestea, pentru a tine evidenta tuturor Studentilor care au participat la Cursuri, o a patra entitate este adaugata: Certificatul

Importanta celei de a cincea entitate nu este foarte mare pentru functionarea unor Servicii Educationale, dar pentru o companie la fel de mare ca oClivE, impartirea Studentilor in Grupuri este o necesitatea.

Avand cele cinci Entitati definite, pasul urmator este conturarea Relatiilor dintre ele:

Relatii	Studenti	Profesori	Cursuri	Certificate	Grupe
Studenti	1:n	-	1:n	1:n	1:1
Profesori	-	-	1:n	-	-
Cursuri	1:n	1:n	-	1:1	1:n
Certificate	1:1	-	1:1	-	-
Grupe	1:n	-	1:n	-	-

(stiu ca in laboratorul1 scrie ca relatiile se “aduna” ca o figura de stil, dar vorbind tehnic, aceasta operatie ar duce la chestii de genul 2:n+1, o operatie care ar avea mai mult sens in acest context ar fi **inmultirea**)

Relatiile ce reies sunt:

Studenti-Studenti: 1:n – Sef de Grupa, rol de reprezentant

Studenti-Cursuri: n:n – fiecare Student allege la ce Cursuri se duce, relatia reprezentand un fel de Catalog

Studenti-Certificate: 1:n – un Student poate avea mai multe Certificate

Studenti-Grupe: n:1 – Fiecare Student face parte dintr-o Grupa si doar una!

Profesori-Cursuri: n:n – Un Profesor poate preda la mai multe Cursuri si mai multi Profesri pot preda acelasi Curs, relatia dintre cele doua nu e una speciala, dar tot e nevoie de un tabel de relatie

Cursuri-Certificate: 1:1 – Fiecare Curs va oferi Studentilor un Certificat, si doar unul!, pentru a evidentia participarea lor la acesta

Cursuri-Grupe: n:n – Toate Grupele vor avea o relatie cu marea majoritate a Cursurilor (desi Studentii ce alcatuiesc Grupa, evident, nu Vor fi inscrisi la toate Cursurile), relatia reprezentand un fel De Orar

O relatie neobisnuita reiese din faptul ca mai multe Grupe participa la acelasi Curs, si mai multi Profesori predau acelasi Curs, in care in orice instanta din tabelul de relatie Orar, trebuie specificat si Profesorul ce va preda acel Curs specific

Astfel, Tabelelece reies vor fi:

Studenti
StudentID
GrupaID
SefGrupaID

Cursuri
CursID

Profesori
ProfesorID

Certificate
CertificatID
StudentID
CursID

Grupe
GrupaID

Catalog
CursID
StudentID

ProfesorCurs
ProfesorID
CursID

Orar
GrupaID
CursID
ProfesorID

Ultimul Pas: adaugarea altor campuri de informatie importante tabelelor respective:

(Luand in considerare numarul de capuri care au ALLOW NULL vs cele care nu au, e mai usor sa le enumar pe cele dintai decat pe cele NOT NULL)

Studenti:

Pe langa Cheia Primara Autoincrementatoare (**StudentiID**) si cele Externe (**GrupaID**, **SefGrupaID**), se vor afla campuri de identificare: **Nume**, **Prenume CNP**, **Sex**, **DataNasterii**; de contactare: **e_mail**, **NumarTelefon**; adresa: **Strada**, **Numar**, **Oras**, **Judet**; si alte date relevante companiei: **DataIncepereCursuri**

Noi la oClivE ne mandrim de diversitatea in oameni pe care am cultivat-o, astfel am permis ca campurile '**Prenume'**, '**CNP'**, '**NumarTelefon'**, '**e_mail**' si '**Sex'** sa fie ALLOW NULL.

CNP este, evident, UNIQUE, precum si **NumarTelefon** si **e_mail**, fiecare Student (si Proesor) trebuie sa aiba date de contact unice.

Campul **Sex** are un CONSTRAINT de 'M' ori 'F' si DEFAULT de 'M'

Profesori:

Cu exceptia Cheilor Primare si Externe si campul '**DataIncepereCursuri'**, tabelul Profesori are toate campurile tabelului Studenti, pe langa acestea sunt adaugate si Cheia Primara AutoIncrementatoare **ProfesorID** si campurile **DataAngajarii**, **Salariu** si **DomeniuPedagogic**

Cursuri:

Acest tabel are campurile **CursID**, Cheie Primara Autoincrementatoare, **MaterieCurs**, camp de indentificare, **DuarataOre**, durata unei sesiuni, **DurataSaptamani**, durata intregii programe si **LinkCurs** din moment ce este un Curs Online.

Campurile MaterieCurs si LinkCurs sunt UNIQUE.

Certificate:

Cheie Primara Autoincrementatoare, **CertificatID**, Chei Externe, **CursID** si **StudentID** si campuri cu informatii aditionale, **NotaFinala** si **DataDobandita**

Grupe:

Cheie Primara Autoincrementatoare, **GrupaID**, camp de identificare, **NumeGrupa** si camp de informatie aditionala, **AnDesfasurare**, care se refera la anul calendaristic in care aceasta grupa isi desfasoara activitatile

Catalog:

Chei Primare si Externe, **CursID** si **StudentID** si camp cu informatie aditionala, **Punctaj**, care reprezinta punctajul current al Studentului la Cursul respectiv.

ProfesorCurs:

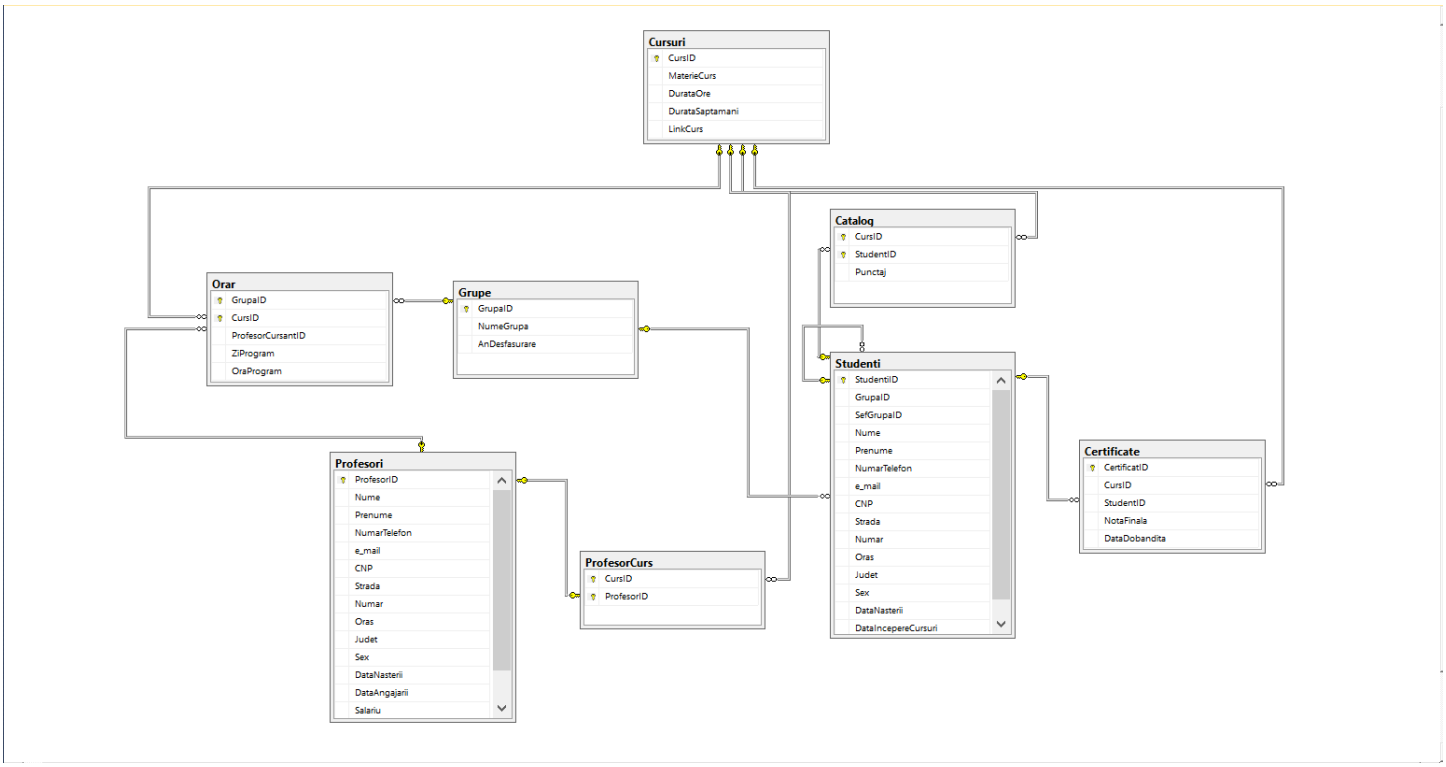
Acest tabel de conexiune contine decat Cheile Primare/Externe **CursID** si **ProfesorID**.

Orar:

Chei Primare/Externe **GrupaID** si **CursID**, Cheie Extrena **ProfesorCursantID** si campurile de informatie aditionala **ZiProgram** si **OraProgram**.

Campul **ZiProgram** i s-a adaugat un CONSTRAINT in care se poate adauga decat numele unei zile saptamanale sau ‘NULL’, care este si valoarea DEFAULT

Diagrama Finala:



aaa

Aplicatia a fost scrisa in HTML si Javascript (si un pic de CSS). Aceasta contine 8 fisiere .html si 7 fisiere .js si un fisier .json.

Interogarile simple implementate sunt:

1: Sa se afiseze grupele si numarul de studenti care fac part din acestea – form_insc.js:

```
SELECT G.GrupaID, COUNT(S.StudentID) AS NrStudenti
FROM Grupe G LEFT JOIN Studenti S
ON S.GrupaID = G.GrupaID
GROUP BY G.GrupaID, G.NumeGrupa;
```

2: Sa se afiseze ID-ul, numele, prenumele, grupa din care fac parte, datele de contact (e-mail si numar de telefon) si materiile la care s-au inscris al tuturor elevilor – student_list.js:

SELECT S.StudentID, S.Nume, S.Prenume, G.NumeGrupa, S.NumarTelefon, S.e_mail, Cu.MaterieCurs
FROM Studenti S LEFT JOIN Grupe G
ON S.GrupaID = G.GrupaID
JOIN Catalog Ca
ON Ca.StudentID = S.StudentID
JOIN Cursuri Cu
ON Cu.CursID = Ca.CursID
ORDER BY G.NumeGrupa, S.Nume, Cu.CursID;

3: Sa se afiseze toate cursurile programate in orar – grupa programata, ziua si ora in care se desfasoara, materia predate si numele si prenumele profesorului cursant – orar.js:

SELECT G.NumeGrupa, O.ZiProgram, O.OraProgram, Cu.MaterieCurs, Cu.DurataOre, P.Nume, P.Prenume
FROM Grupe G LEFT JOIN Orar O
ON G.GrupaID = O.GrupaID
JOIN Cursuri Cu
ON Cu.CursID = O.CursID
JOIN Profesori P
ON O.ProfesorCursantID = P.ProfesorID
ORDER BY G.NumeGrupa, Cu.CursID;

4: Sa se afiseze media generala a puncatjelor luate de studenti din fiecare grupa – statistici.js:

SELECT G1.NumeGrupa, AVG(Ca1.Punctaj) AS Medie
FROM Grupe G1 LEFT JOIN Studenti S1
ON S1.GrupaID = G1.GrupaID
JOIN Catalog Ca1
ON Ca1.StudentID = S1.StudentID
GROUP BY G1.NumeGrupa

5: Sa se afiseze media generala a punctajelor luate de studenti la fiecare materie – statistici.js:

```
SELECT C.MaterieCurs, AVG(Ca.Punctaj) AS Medie
FROM Cursuri C LEFT JOIN Catalog Ca
ON C.CursID = Ca.CursID
WHERE C.MaterieCurs != 'Chimie' AND C.MaterieCurs != 'Informatica'
GROUP BY C.MaterieCurs
```

6: Sa se afiseze media si ID-ul Studentului cautat – search_avg.js:

```
SELECT S.StudentID, AVG(Ca.Punctaj) AS Medie
FROM Studenti S LEFT JOIN Catalog Ca
ON S.StudentID = Ca.StudentID
WHERE S.Nume = '${form.Nume.value}' AND S.Prenume =
'${form.Prenume.value}'
GROUP BY S.StudentID ORDER BY AVG(Ca.Punctaj
```

Iar cele complexe sunt:

1: Sa se afiseze grupele a caror studenti au cea mai mare medie generala a punctajelor la fiecare materie – statistici.js:

```
SELECT M.NumeGrupa, M.MaterieCurs, M.Medie
FROM (SELECT C1.MaterieCurs, G1.NumeGrupa, AVG(Ca1.Punctaj) AS Medie,
ROW_NUMBER() OVER(PARTITION BY C1.MaterieCurs ORDER BY AVG(Ca1.Punctaj)
DESC) rn
FROM Grupe G1 LEFT JOIN Studenti S1
ON S1.GrupaID = G1.GrupaID
JOIN Catalog Ca1
ON Ca1.StudentID = S1.StudentID
JOIN Cursuri C1
```

```
    ON C1.CursID = Ca1.CursID
    WHERE C1.MaterieCurs != 'Chimie'
    GROUP BY G1.NumeGrupa, C1.MaterieCurs) AS M
    WHERE rn = 1
    ORDER BY M.NumeGrupa;
```

2: Sa se afiseze numele si prenumele studentilor care fac parte din grupele afisate anterior (1)) care au punctajul pe materie mai mare decat media afisata – statistici.js:

```
    SELECT S.Nume, S.Prenume, G.NumeGrupa, C.MaterieCurs, Ca.Punctaj
    FROM (SELECT G1.NumeGrupa, C1.MaterieCurs, AVG(Ca1.Punctaj) AS Medie, ROW_NUMBER() OVER(PARTITION BY C1.MaterieCurs ORDER BY AVG(Ca1.Punctaj) DESC) rn
    FROM Grupe G1 LEFT JOIN Studenti S1
    ON S1.GrupaID = G1.GrupaID
    JOIN Catalog Ca1
    ON Ca1.StudentID = S1.StudentID
    JOIN Cursuri C1
    ON C1.CursID = Ca1.CursID
    WHERE C1.MaterieCurs != 'Chimie'
    GROUP BY G1.NumeGrupa, C1.MaterieCurs) AS M, Studenti S LEFT JOIN Catalog Ca
    ON S.StudentID = Ca.StudentID
    JOIN Cursuri C ON C.CursID = Ca.CursID
    JOIN Grupe G
    ON S.GrupaID = G.GrupaID
    WHERE G.NumeGrupa = M.NumeGrupa AND C.MaterieCurs = M.MaterieCurs AND M.rn = 1
    ORDER BY G.NumeGrupa;
```

3: Sa se afiseze numele, prenumele si datele de contact a tuturor elevilor a caror medie este mai mare decat media grupei din care fac parte si sunt inrolati la un numar variabil de materii sau mai multe – statistici.js:

```
SELECT S.StudentID, S.Nume, S.Prenume, G.NumeGrupa, S.NumarTelefon, S.e_mail, COUNT(Cu.MaterieCurs) AS NrMateriiInrolate, AVG(Ca.Punctaj) AS MedieGenerala
FROM (SELECT G1.NumeGrupa, AVG(Ca1.Punctaj) AS Medie
FROM Grupe G1 LEFT JOIN Studenti S1
ON S1.GrupaID = G1.GrupaID
JOIN Catalog Ca1
ON Ca1.StudentID = S1.StudentID
GROUP BY G1.NumeGrupa) AS GM, Studenti S LEFT JOIN Grupe G
ON S.GrupaID = G.GrupaID
JOIN Catalog Ca
ON Ca.StudentID = S.StudentID
JOIN Cursuri Cu
ON Cu.CursID = Ca.CursID
GROUP BY S.StudentID, S.Nume, S.Prenume, G.NumeGrupa, S.NumarTelefon, S.e_mail, GM.NumeGrupa, GM.Medie
HAVING GM.NumeGrupa = G.NumeGrupa AND AVG(Ca.Punctaj) > GM.Medie AND COUNT(Cu.MaterieCurs) >= '${value}';
```

4: Sa se afiseze numele si prenumele profesorilor care preda la grupe a caror medie la materia predata de profesorul respectiv este mai mare decat media generala a materiei – statistici.js:

```
SELECT G.NumeGrupa, C.MaterieCurs, P.Nume, P.Prenume, AVG(Ca.Punctaj) AS Medie
FROM (SELECT C.MaterieCurs, AVG(Ca.Punctaj) AS Medie
FROM Cursuri C LEFT JOIN Catalog Ca
ON C.CursID = Ca.CursID
WHERE C.MaterieCurs != 'Chimie' AND C.MaterieCurs != 'Informatica'
GROUP BY C.MaterieCurs) AS MC, Orar O LEFT JOIN Grupe G
```

```
    ON O.GrupaID = G.GrupaID
    JOIN Cursuri C
    ON C.CursID = O.CursID
    JOIN Profesori P
    ON P.ProfesorID = O.ProfesorCursantID
    JOIN Studenti S
    ON S.GrupaID = G.GrupaID
    JOIN Catalog Ca
    ON S.StudentID = Ca.StudentID AND Ca.CursID = C.CursID
    WHERE C.MaterieCurs != 'Chimie'
    GROUP BY G.NumeGrupa, C.MaterieCurs, P.Nume, P.Prenume, MC.MaterieCurs,
MC.Medie
    HAVING MC.MaterieCurs = C.MaterieCurs AND AVG(Ca.Punctaj) > MC.Medie
    ORDER BY C.MaterieCurs DESC, G.NumeGrupa;
```