FIȘĂ DE LUCRU · CLASA A XII-A · FIȘA 1 · ACOPERĂ LECȚIILE XII.1–XII.3

Baze de date: concepte, modelare, normalizare

Durată: 90 min Nivel: mate-info

CE VEI ȘTI DUPĂ LECȚIE

- Citești o bază de date: tabele, chei primare și străine, constrângeri, NULL.
- Transformi un enunț într-un model: entități, cardinalități, tabele asociative.
- Recunoști anomaliile și aduci un tabel în FN3, explicând fiecare pas.

Ce trebuie să știi (o pagină)

Baza de exercițiu este **ȘCOALA** (`scoala.sql` , aceeași ca în lecții): `CLASE(id_clasa PK, nume_clasa UNIQUE, profil, diriginte)` · `ELEVI(id_elev PK, nume, prenume, data_nasterii, id_clasa FK)` · `DISCIPLINE(id_disc PK, denumire UNIQUE)` · `NOTE(id_nota PK, id_elev FK, id_disc FK, nota CHECK 1–10, data_nota)` .

Noțiune	Regula	În baza ȘCOALA
tabel / înregistrare / câmp	tabel = relație, rând = înregistrare, coloană = câmp	ELEVI are 10 înregistrări și 5 câmpuri
`NULL`	lipsa valorii; nu e 0, nu e `''` ; `NULL = NULL` nu e adevărat	data nașterii lui Barbu Tudor
cheie primară (PK)	unică, niciodată NULL	`id_elev`
cheie străină (FK)	arată spre PK-ul altui tabel; poate fi NULL	`ELEVI.id_clasa → CLASE.id_clasa`
cheie candidat	ar putea fi PK (unică, minimală)	`nume_clasa` , un CNP
constrângeri	`NOT NULL` , `UNIQUE` , `CHECK` , `FOREIGN KEY`	`CHECK (nota BETWEEN 1 AND 10)`
1:N	FK pe partea **N**	o clasă → mulți elevi: FK în ELEVI
1:1	FK în oricare tabel, cu `UNIQUE`	clasă – diriginte
M:N	**tabel asociativ** cu două FK	elevi ↔ discipline prin NOTE
dependență A → B	știind A, B e unic	`id_elev → nume` , `id_clasa → diriginte`
anomalii	de **actualizare**, **inserare**, **ștergere**	dirigintele scris la fiecare elev
FN1	valori atomice, fără grupuri repetitive	nu `"Mate: 9, 10"` într-un câmp
FN2	FN1 + nimic nu depinde de **o parte** a cheii compuse	

Noțiune	Regula	În baza ȘCOALA
FN3	FN2 + nimic non-cheie nu depinde de alt non-cheie	„cheia, toată cheia, nimic altceva decât cheia”

ATENȚIE — GREȘELI FRECVENTE

În SQLite, cheile străine sunt verificate **doar** după comanda `PRAGMA foreign_keys = ON;` (DB Browser for SQLite o pornește implicit, din Preferințe). În MySQL (InnoDB) și Oracle sunt verificate mereu.

Nivelul 1 · Prezice

★ citește și prezice

P1. Fără să rulezi nimic, doar din `scoala.sql`: câte tabele are baza? Câte înregistrări are ELEVI și câte câmpuri are NOTE? Ce fel de cheie este `NOTE.id_disc`? Ce valoare are `data_nasterii` la Barbu Tudor: `0`, `''` sau altceva?

P2. Urmărește cheile. Pornind de la nota cu `id_nota = 7` și apoi de la cea cu `id_nota = 19`, află: numele elevului, disciplina, nota, clasa și dirigintele. Câte tabele ai „vizitat” de fiecare dată?

P3. Paznicul bazei. Cu `PRAGMA foreign_keys = ON`, care comenzi sunt **acceptate** și care **refuzate**? Pentru cele refuzate, numește constrângerea.

```
a) INSERT INTO clase VALUES (6, 'IX A', 'uman', 'Pop');
b) INSERT INTO elevi VALUES (11, 'Pop', NULL, '2010-01-01', 1);
c) INSERT INTO note VALUES (21, 3, 2, 11, '2026-10-20');
d) INSERT INTO note VALUES (22, 15, 1, 8, '2026-10-20');
e) INSERT INTO elevi VALUES (3, 'Nou', 'Elev', NULL, 2);
f) INSERT INTO elevi VALUES (11, 'Nou', 'Elev', NULL, NULL);
g) INSERT INTO note VALUES (23, 10, 4, 10, NULL);
```

P4. Cardinalitatea. Scrie 1:1, 1:N sau M:N și justifică într-o propoziție: țară – capitală; client – comandă; film – actor; echipă – jucător (într-un sezon); mașină – număr de înmatriculare; student – cameră de cămin; elev – cerc/club.

P5. Citește ERD-ul unui cinematograf: `FILME(PK id_film, titlu, durata)`, `SALI(PK id_sala, nume, locuri)`, `PROIECTII(PK id_proiectie, FK id_film, FK id_sala, data_ora)`, `BILETE(PK id_bilet, FK id_proiectie, rand, loc, pret)`. a) Poate un film să ruleze în două săli? b) Poate un bilet să fie valabil la două proiecții? c) Ce relație există între FILME și SALI și prin ce tabel e realizată? d) Câte chei străine are modelul?

P6. Datele pot doar să infirme. Tabelul de mai jos e de la un festival de robotică. Care dintre dependențele `cnp → nume`, `nume → cnp`, `atelier → mentor`, `mentor → atelier`, `(cnp, atelier) → nota`, `nota → cnp` sunt **contrazise** de date?

cnp	nume	atelier	mentor	nota
501	Ana Pop	Arduino	Ilie	9
502	Dan Ene	Arduino	Ilie	7
501	Ana Pop	Drone	Vlad	10

cnp	nume	atelier	mentor	nota
503	Ana Pop	Drone	Vlad	8
502	Dan Ene	Web	Ilie	9

P7. În ce formă normală? Spune cea mai mare formă normală (sau „nici FN1”) pentru fiecare tabel (câmpurile dinaintea lui `;` formează cheia): a) `ELEV(id; nume, telefoane)` cu `telefoane = "0722…, 0744…"`; b) `NOTA(id_elev, id_disc, data; nume_elev, nota)` — cheia e tripleta; c) `ELEV(id_elev; nume, id_clasa, diriginte)`; d) `CLASE` din baza ȘCOALA.

P8. Ce anomalie e? Un service auto ține totul într-un tabel `REPARATII(id_rep, nr_auto, proprietar, tel_proprietar, mecanic, data, cost)`. Numește anomalia (actualizare / inserare / ștergere) din fiecare situație: a) proprietarul mașinii TM-07-ABC își schimbă telefonul; b) service-ul angajează un mecanic care încă n-a reparat nimic; c) se șterge singura reparație a mașinii B-99-XYZ.

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Un elev a ales `nume` drept cheie primară în ELEVI: „oricum îl caută toată lumea după nume”. Dă **două** situații reale în care modelul lui se strică și repară-l.

P10. Ca să memoreze cluburile, un elev a adăugat coloana `id_club` în ELEVI. Ana Popescu se înscrie la robotică **și** la șah. Ce se întâmplă? Propune structura corectă (cu cheia primară).

P11. Ce e greșit în modelul de mai jos, în care „o clasă are mai mulți elevi”? Repară-l.

```
CLASE(PK id_clasa, nume_clasa, FK id_elev → ELEVI)
ELEVI(PK id_elev, nume, prenume)
```

P12. Tabelul `CONCURS(id_participant, proba; nume, liceu, oras_liceu, punctaj)` a fost „normalizat” de un elev astfel: `PARTICIPANTI(PK id_participant, nume, liceu, oras_liceu)` și `REZULTATE(PK (id_participant, proba), punctaj)`. A scăpat de dependența parțială, dar o problemă a rămas. Care? Termină normalizarea.

P13. Ce greșeală a făcut elevul? „Tabelul `ELEVI(id, nume, telefon1, telefon2, telefon3)` e în FN1, pentru că fiecare câmp are o singură valoare.” Ce se întâmplă cu un elev care are 4 numere? Dar cu căutarea „al cui e numărul 0744…”? Repară.

P14. Pași amestecați. Pune în ordine pașii proiectării unei baze de date. Un pas este **greșit** și nu trebuie folosit.

```
A) Transformă fiecare relație M:N într-un tabel asociativ.
B) Subliniază în enunț substantivele (entități) și verbele (relații).
C) Alege atributele și cheia primară pentru fiecare entitate.
D) Verifică dacă tabelele pot răspunde la toate rapoartele cerute.
E) Stabilește cardinalitatea fiecărei relații.
F) Pune cheia străină pe partea „1” a fiecărei relații 1:N.
G) Pune cheile străine pe partea „N” a relațiilor 1:N.
```

Nivelul 3 · Modelează

★★ scrie cod

Scrie tabelele în forma `TABEL(PK cheie, câmpuri, FK câmp → TABEL)`. „Testele" sunt întrebări la care modelul tău **trebuie** să poată răspunde.

P15. Playlistul. Utilizatori, playlisturi, melodii. Aceeași melodie poate apărea în mai multe playlisturi și chiar **de două ori** în același playlist.

test	trebuie să meargă
Care e a 3-a melodie din playlistul „Sală"?	da
În câte playlisturi apare melodia X?	da
Melodia X pusă pe pozițiile 1 și 4 ale aceluiași playlist	acceptată

P16. Campionatul. Echipe, jucători (fiecare la o echipă), meciuri (gazde contra oaspeți), goluri (cine, în ce minut). Teste: „cine a jucat acasă pe 12 octombrie?", „golgheterul", „scorul unui meci" (fără să-l memorezi separat).

P17. Închirieri de biciclete. Adu tabelul în FN3, scriind la fiecare pas ce dependență elimini.

id_inch	data	cod_bic	model	pret_ora	cnp_client	nume_client	ore
1	03.05	B7	City	8	601	Ilie	2
2	03.05	B9	Mountain	12	602	Ene	3
3	04.05	B7	City	8	602	Ene	1

Teste: prețul modelului City se schimbă într-un **singur** loc; poți adăuga un model nou înainte de a cumpăra vreo bicicletă.

P18. Rețete. Rețete, ingrediente, cantități. Teste: „lista de cumpărături pentru clătite", „toate rețetele care folosesc ouă". Unde pui `cantitate` și de ce nu în INGREDIENTE?

P19. Șefi și prieteni. a) Angajații unei firme: fiecare are un șef, care e tot angajat (directorul nu are). b) Prieteniile dintr-o rețea socială. Ambele relații sunt ale unui tabel **cu el însuși**. Ce cardinalitate are fiecare și cum o implementezi?

P20. Extinde ȘCOALA. Adaugă profesorii și cine ce predă la ce clasă. Teste: „cine predă Informatică la X B?" (un singur răspuns); un profesor poate preda două discipline la aceeași clasă; **nu** pot exista doi profesori de Informatică la X B. Alege cu grijă cheia primară.

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Adevărat sau fals? „Un tabel în FN1 a cărui cheie primară are **un singur** câmp este automat în FN2." Explică. Este automat și în FN3? Dă un contraexemplu din baza ȘCOALA (sau dintr-o variantă a ei).

P22. Câte chei? `ORAR(zi, ora, sala, clasa, profesor, disciplina)`. Știm: într-o sală, la o oră, e o singură clasă; o clasă, la o oră, e într-o singură sală; un profesor, la o oră, e la o singură clasă. Găsește **toate** cheile candidat. Pe care ai alege-o primară?

P23. Ce se întâmplă dacă... șterg clasa X B (`id_clasa = 3`) din CLASE? Răspunde în trei cazuri: a) cheile străine verificate; b) cheile străine **neverificate** (SQLite fără PRAGMA); c) FK-ul a fost declarat `ON DELETE SET NULL`. Care e cel mai periculos?

P24. Coloana `cnp` are constrângerea `UNIQUE`. Câți elevi pot avea `cnp` = NULL? Ghicește, apoi verifică în SQLite inserând trei elevi fără CNP. Leagă răspunsul de regula `NULL = NULL`.

P25. Merită? Cineva propune să adăugăm în ELEVI coloana `medie`, „ca să nu o calculăm de fiecare dată". Ce formă normală / ce regulă încalcă? Ce anomalie poate apărea? Există situații în care e totuși o idee bună?
