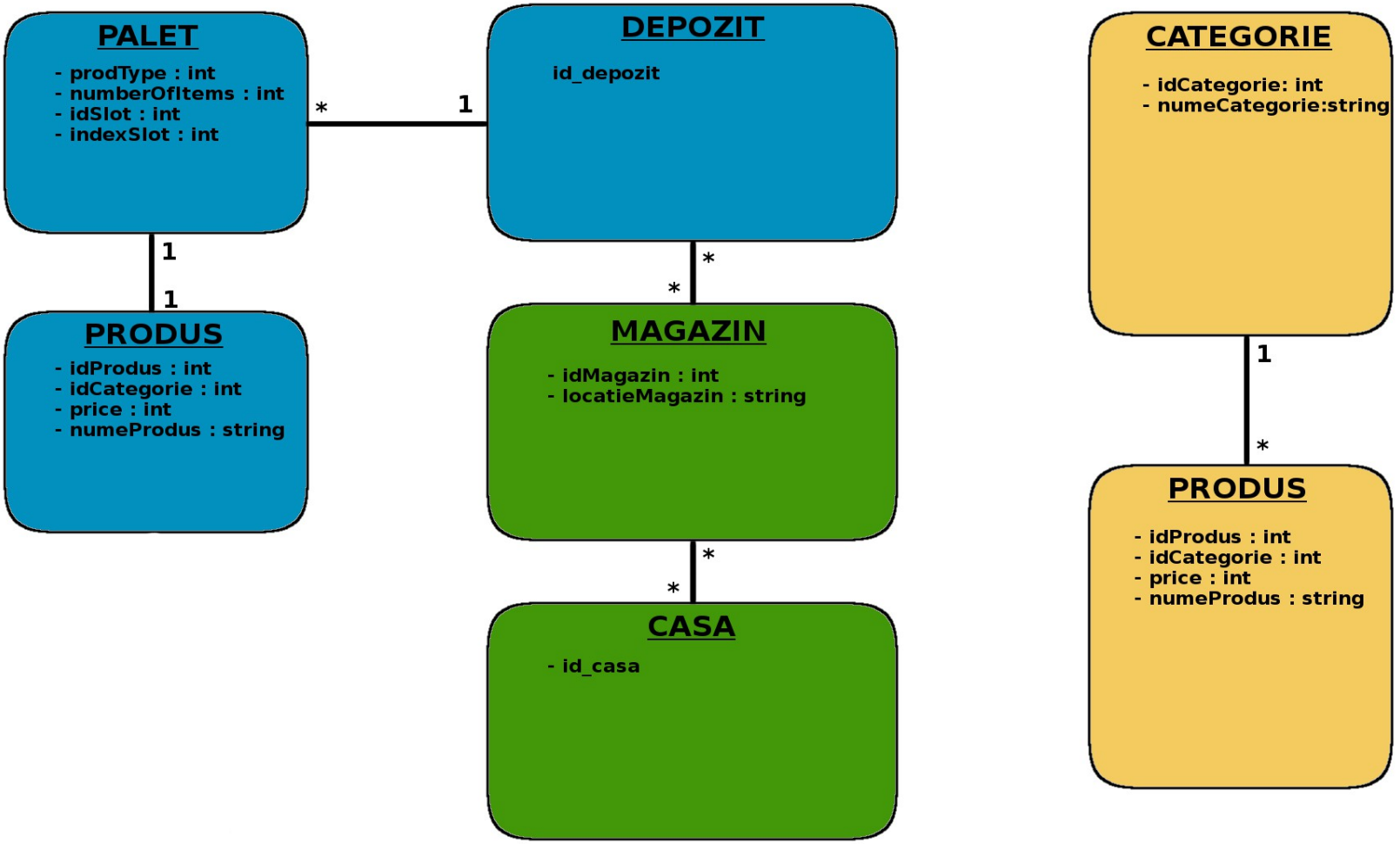
Retail - Sales - Reporting

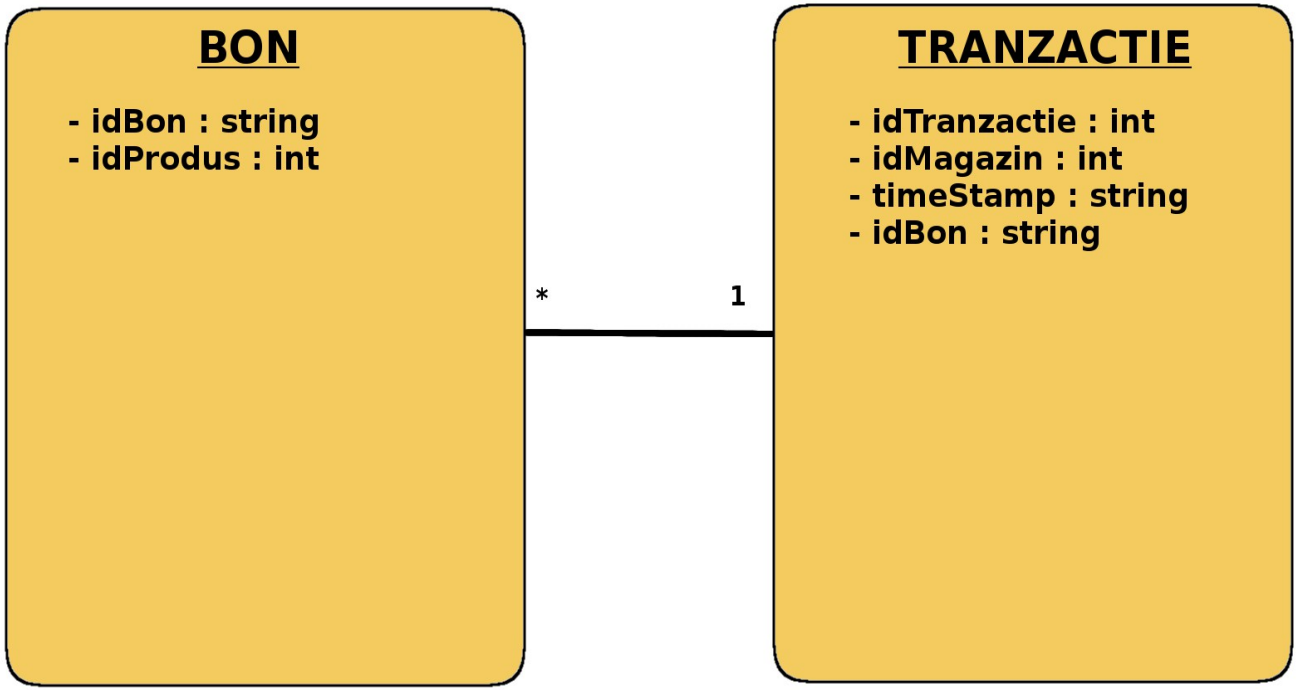
A 311 CA(b) project

Echipe initiale:

Arhitectură	Task 1	Task 2	Task 3
Bîrlea Costin	Rizescu Eusebiu	Nicolaescu Vlad	Mărgărit Paula
Voinescu Petre	Nicolescu Roxana	Șteaburdea Robert	Zaharia Ana
	Aldescu Marian	Zăblău Sever	Jipa Mihai
	Bobleagă Ioana	Ștefan Ștefan	Căciulă Andrei

Arhitectura:




BON
- idBon : string
- idProdus : int
*
1
TRANZACTIE
- idTranzactie : int
- idMagazin : int
- timeStamp : string
- idBon : string

Echipe secundare:

Dupa o analiza in profunzime am ajuns la concluzia ca task-urile propriu zise nu puteau fi incepute fara arhitectura implementata in totalitate (clasele prezentate anterior).

Astfel am impartit arhitectura egal la cele 4 echipe , urmand sa reformam echipe noi pentru cele 3 task-uri.

Echipe secundare:

Task 1	Task 2	Task 3
Rizescu Eusebiu	Nicolaescu Vlad	Mărgărit Paula
Nicolescu Roxana	Șteaburdea Robert	Zaharia Ana
Aldescu Marian	Zăblău Sever	Jipa Mihai
Bobleagă Ioana	Ștefan Ștefan	Căciulă Andrei
	Voinescu Petre	Bîrlea Costin

Cerinta:

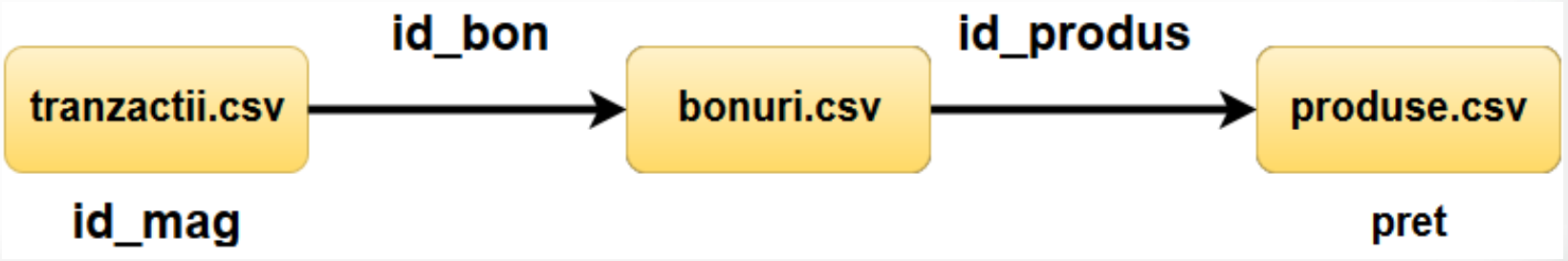
Task 1 :

- O lista cu toate magazinele si vanzarile totale facute de acestea
- O lista cu toate produsele si vanzarile totale facute pentru acestea
- Care este valoarea cosului mediu
- Care sunt categoriile cele mai bine vandute pentru fiecare magazin
- Care sunt perechile de produse care se vand cel mai bine impreuna

Rezolvare:

Task 1 – 1 :

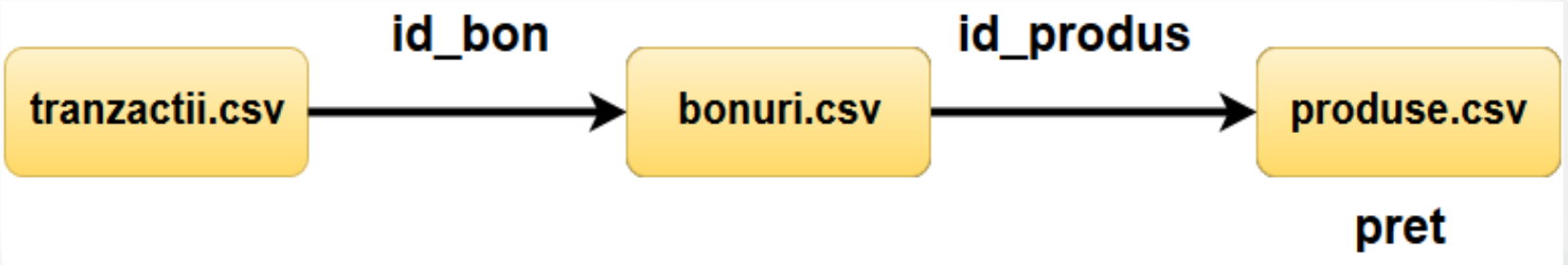
- Parcurgem ' tranzactii.csv ' ;
- Retinem: id_mag , id_bon ;
- Cu id_bon identificam bonul corespunzator si identificam id_produs ;
- Cu id_produs obtinem pretul din produse.csv care va fi adunat la o suma ' suma[id_mag] ' ;



Rezolvare:

Task 1 – 2 :

- Parcurgem ' tranzactii.csv ' ;
- Retinem id_bon ;
- Cu id_bon identificam bonul corespunzator si identificam id_produs ;
- Cu id_produs obtinem pretul din produse.csv care va fi adunat la o suma ;



Rezolvare:

Task 1 – 3 :

- Se aduna sumele ' suma [id_mag] ' de la Task 1 – 1 ;
- Se imparte suma la numarul de linii -1 din fisierul ' tranzactii.csv ' ;

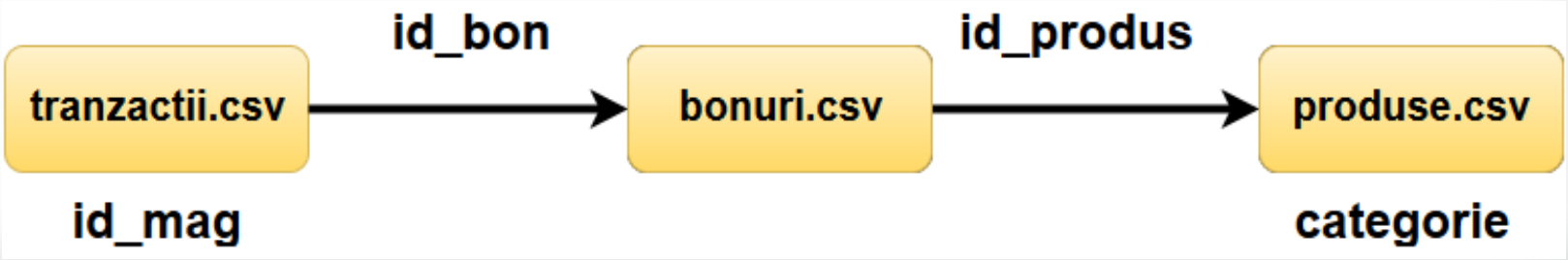
```
/* Taskul 1_3: */
void task1_3( unsigned long long suma, int length, LinkedList<Tranzactie> &listaTranzactii ) {
    double valoare_cos_mediu;
    cout<<suma<<" "<<length<<endl;
    /* cosul mediu este raportul dintre suma totala aflata de la task 1_2 si numarul de tranzactii */
    valoare_cos_mediu = double( suma )/double( length );

    /* Afisare valoarea cosului mediu */
    cout<<valoare_cos_mediu<<endl;
}
```

Rezolvare:

Task 1 – 4 :

- O matrice cu magazine pe linii si categorii pe coloane ;
- Se initializeaza cu 0 ;
- Avem id_mag si id_bon din ' tranzactii.csv ' , cu id_bon obtinem din ' bonuri.csv ' id_produs , cu care extragem din ' produse.csv ' categoria .
- Adunam in matrice la pozitia cu linia magazinului respectiv si coloana categoriei corespunzatoare ;



Rezolvare:

Task 1 – 5 :

- Parcurgem lista de tranzactii ;
- Pentru fiecare id_bon parcurgem lista de bonuri si cautam id-urile produselor de pe bonul respectiv .
- Id-urile gasite se introduc intr-un vector , verificam sa nu avem 2 produse identice pe acelasi bon .
- Pentru produsele de pe bonul respectiv generam combinari de numarProduse luate cate 2 pentru a gasi toate perechile iar intr-o matrice de numarProduse * numarProduse vom itera pozitiile corespunzatoare id_prod din fiecare pereche
- Stergem de fiecare data produsele din vector .
- Pentru matricea obtinuta calculam maximul .
- Vom afisa lista de produse care au id-urile corespunzatoare indicilor maximelor din matrice .

Probleme intampinate:

Pentru Task 1 :

- Output-uri neinteligibile
- Matrici si vectori declarate eronat ceea ce a rezultat in output-uri cu valori foarte mari
- Segmentation fault a dominat terminalul .

Cerinta:

Task 2 :

- Zilele cu cele mai multe produse vandute
- Zilele cu cei mai multi cumparatori
- Sa se poate vedea rapid continutul unui bon introducand id-ul acestuia
- Cati clienti ar beneficia de introducerea unei a doua case in magazin

Rezolvare:

Task 2- 1 :

- Parcurgem lista de tranzactii ;
- Pentru fiecare magazin , tinem un vector de 366 de pozitii (pe cate zile avem evidenta tranzactiilor) ;
- Pentru fiecare tranzactie verific de cate ori apare id-ul bonului in lista de bonuri , ceea ce reprezinta numarul de produse vandute la acea tranzactie ;
- Sortam vectorii corespunzatori fiecarui magazin ;

Rezolvare:

Task 2– 2 :

- Pentru fiecare tranzactie , verificam timestamp-ul ;
- Incrementam pozitia corespunzatoare zilei respective pentru magazinul respectiv ;

Rezolvare:

Task 2- 3 :

- Initializam un vector cu 0 cu lungimea egala cu numarul de produse disponibile in magazin ;
- Cautam id-ul dat in lista de bonuri ;
- Incrementam pozitia corespunzatoare cand gasim in lista id-ul cautat ;
- Parcurgem vectorul si cautam in lista de produse denumirea fiecarui produs cumparat ;

Rezolvare:

Task 2- 4 :

- Pentru fiecare magazin , calculam numarul mediu de clienti pe saptamana ;
- Consideram ca jumatate din clienti ar merge la a doua clasa ;
- Se observa totusi ca numarul de clienti pe saptamana este foarte mic , deci probabilitatea ca doi clienti sa ajunga la casa aproximativ in acelasi timp este foarte mica ;

Probleme intampinate:

Pentru Task 2 – 1 , 2 :

- Folosim o matrice cu nrMagazine linii si cate o coloana pentru fiecare zi
- Fiecare linie trebuie sortata : Problema apare cand trebuie sa retinem carei zile ii corespunde numarul cel mai mare ;

Solutie:

- Declaram o matrice echivalenta de permutare care are pe pozitia [i][j] numarul j ;
- Cand sortam matricea de produse vandute , interschimbam si valorile din matricea de permutare si astfel tinem minte in ce zi au fost vandute cele mai multe produse ;

Probleme intampinate:

Pentru Task 2 – 4 :

- Este calculat numarul mediu de cumparatori pe saptamana pentru fiecare magazin . Am considerat ca la a doua casa ar merge jumatate din acestia .
- Problema este ca in medie , numarul de clienti pe saptamana este cam 20 (deci foarte mic) , ceea ce inseamna ca deschiderea unei a doua case nu este justificata .

Cerinte:

Task 3 :

- Sa se afle rapid in care slot se gaseste un anumit tip de produs
- Atunci cand I se cere din partea unui magazin un palet cu un anumit tip de produs , care este succesiunea de mutari de paleti ce trebuie facute pentru a obtine paletul solicitat
- Considerand ca la inceputul anului fiecare magazin a pornit cu 1 palet din fiecare tip si solicita depozitului un nou palet cand ajung sa aiba 10% dintr-un anumit produs , care este prima comanda pe care nu o poate onora din depozit

Note:

Task 3 :

- Slot-urile si paletii au fost interpretati ca niste vectori de stive , insa implementarea acestora s-a dovedit sa fie inutila
- Abordarea consta in utilizarea unui hashtable
- Se foloseste hashtable-ul precum un vector de frecventa

Rezolvare:

Task 3- 1 :

- Se apeleaza HASH [id_prod] si se parcurge lista aferenta bucket-ului repectiv ;
- Se extrage doar primul element al perechii respectiv idSlot ;

Note :

- HASH este functia de Hashtable .
- ProdType reprezinta numele produsului (key) .
- Este probabil ca tipul respectiv de produs sa se afle in paletii distribuiti pe mai multe slot-uri , de aceea este nevoie de o lista pentru retinerea lor .

Rezolvare:

Task 3- 2 :

- Se apeleaza HASH [id_prod] si se parcurge lista aferenta bucket-ului repectiv ;
- Folosim indexSort pentru a calcula numarul de mutari necesare scoaterii unui palet ;

Formula :

2 * (capacitatea_Slotului_respectiv - indexSort) + 1 ;

Rezolvare:

Task 3- 3 :

- Sortam lista de bonrui dupa data , dupa care o parcurgem ;
- Contorizam numarul de produse aflate in magazine si numarul de paleti ramasi in depozit ;
- Verificam daca magazinul are nevoie de o reaprovizionare :
  - La fiecare pas al iteratiei ;
  - La fiecare produs aflat pe bonul curent ;
- Daca este nevoie , vom importa din depozit un palet nou cu ajutorul hashtable-ului ;

Probleme intampinate:

Pentru Task 3 :

- Probleme interne

Costin Bîrlea
Paula – Elena Mărgărit
Vlad Nicolaescu
Ana Zaharia
Roxana Nicolescu
Marian Aldescu
Eusebiu Rizescu
Ioana Bobleagă
Mihai Jipa
Ștefan Ștefan
Robert Șteaburdea
Sever Zăblău
Petre Voinescu

Retail - Sales - Reporting

A 311 CA(b) project