ELEMENTE DE SECURITATE A DATELOR

I.OBIECTIVE

1. Securitatea serverului de baze de date MySQL
2. Securizarea accesului utilizatorilor

II. FUNDAMENTARE TEORETICĂ

2.1.ELEMENTE DE SECURITATE IN MYSQL

Funcția pricipală a sistemului de securitate al Sistemului de gestiune a bazelor de date MySQL este aceea de a *autentifica și autoriza* utilizatorii conectați pentru a accesa datele stocate.
Autorizarea se referă la permisiunea de a rula interogări/modificări ale datelor precum SELECT, INSERT, UPDATE sau DELETE. O clasă aparte de privilegii se referă la drepturile de administrare a bazei de date și de interacțiune cu sistemul de operare. Termenul **privilegiu** denotă în general un drept al unui utilizator de a acționa într-un anumit fel asupra unui obiect al bazei de date (tabelă, câmp, index etc.).

Utilizatori și parole MySQL. Există diferențe semnificative între sistemul de utilizatori și parole MySQL și sistemele de operare Unix sau Windows ,cele mai importante de menționat sunt: numele de utilizatori MySQL pot avea până la 16 caractere semnificative, numele utilizatorilor și parolele sunt păstrate separat de MYSQL și sunt distincte de cele din sistemul de operare, MySQL criptează parolele folosind un algoritm diferit de cel utilizat de Windows.

La instalarea sistemului de gestiune MySQL este creat implicit un utilizator numit "root" care are toate drepturile activate. Acest utilizator trebuie folosit, din motive de securitate, doar pentru administrarea sistemului de gestiune a bazelor de date MySQL. Pentru fiecare utilizator care va interacționa cu serverul, trebuie creat un utilizator nou MySQL. Deși nu este obligatorie stabilirea unor parole pentru utilizatori la crearea conturilor acestora, această operație este de cea mai mare importanță, în caz contrar securitatea serverului este grav compromisă.

Crearea unui utilizator se poate realiza cu ajutorul comenzii :

CREATE user_nume **IDENTIFIED** by password

Iar forma sa completă (cf manual MySQL)

```
CREATE USER [IF NOT EXISTS] user [auth_option]
[, user [auth_option]] ...
    DEFAULT ROLE role [, role ] ...
    [REQUIRE {NONE | tls_option [[AND] tls_option] ...}]
 [WITH resource_option [resource_option] ...]     [password_option |
lock_option] ...
user
```

```
auth_option: {
| IDENTIFIED BY auth_string
| IDENTIFIED BY RANDOM PASSWORD

| IDENTIFIED WITH auth_plugin
  | IDENTIFIED WITH auth_plugin BY 'auth_string'
  | IDENTIFIED WITH auth_plugin BY RANDOM PASSWORD
  | IDENTIFIED WITH auth_plugin AS 'auth_string'
}
tls_option: {
   SSL
 | X509
 | CIPHER 'cipher'
 | ISSUER 'issuer'
 | SUBJECT 'subject'
}
resource_option: {
    MAX_QUERIES_PER_HOUR count
  | MAX_UPDATES_PER_HOUR count
  | MAX_CONNECTIONS_PER_HOUR count
  | MAX_USER_CONNECTIONS count
}
password_option: {
    PASSWORD EXPIRE [DEFAULT | NEVER | INTERVAL N DAY]
  | PASSWORD HISTORY {DEFAULT | N}
  | PASSWORD REUSE INTERVAL {DEFAULT | N DAY}
  | PASSWORD REQUIRE CURRENT [DEFAULT | OPTIONAL]
} lock_option: {
ACCOUNT LOCK
  | ACCOUNT UNLOCK
       }
```

Detalii suplimentare pot fi consultate la adresa http://www.mysqltutorial.org/mysql-create-user.aspx). Se pot utiliza comenzile DROP pentru șteregere, respectiv SHOW USER pentru vizualizarea utilizatorilor, detalii și exemple https://www.mysqltutorial.org/mysql-show-users/.

Principiul "*minimului de privilegii necesare*" poate fi folosit pentru a spori securitatea oricărui calculator și este aplicabil și serverului MySQL. El este foarte simplu, dar este în același timp foarte puternic și de importanță în securizarea sistemului cu baze de date, în acest principiu se afirmă faptul că :

Un utilizator (sau proces) trebuie să aibă cel mai scăzut nivel de privilegii, însă suficient pentru a putea executa sarcinile care i-au fost alocate.

Privilegiile puse la dispoziție de MySQL. Informațiile despre privilegii sunt stocate în tabelele *user, db, host, tables_prv și columns_priv* din baza de date de sistem Mysql.

Detalii suplimentare :http://www.mysqltutorial.org/getting-started-with-mysql-access-control-system.aspx).

Serverul va citi conținutul acestor tabele ori de câte ori sistemul de privilegii trebuie să acționeze.

Tabelul următor prezintă *sistemul de privilegii* împreună cu numele coloanelor din tabelele de privilegii:

Privilegiu	**Denumire coloana**	**Context**	**Semnificatie**
select	Select_priv	tabele	permite selectarea (vizualizarea) datelor
insert	Insert_priv	tabele	permite adăugarea unor noi înregistrări
update	Update_priv	tabele	permite modificarea datelor
delete	Delete_priv	tabele	permite stergerea inregistrarilor
index	Index_priv	tabele	permite crearea/stergerea indecsilor
alter	Alter_priv	tabele	permite redenumire /modificarea structurii tabelei
create	Create_priv	baza de date, tabele, indecsi	permite crearea unei baze de date/tabele
drop	Drop_priv	baza de date, tabele	permite stergerea unei baze de date/tabele
grant	Grant_priv	baza de date, tabele	permite delegarea privilegiilor catre alt utilizator
shutdown	Shutdown_priv	administrare server	permite oprirea serverului din programul client
process	Process_priv	administrare server	permite vizualizarea/oprirea proceselor in executie
file	File_priv	acces la fisiere externe	permite schimbul de date intre tabele si fisiere

Câteva dintre aceste privilegii necesită o atenție aparte, astfel:

- Privilegiul *grant* permite utilizatorilor **să transmită** privilegiile lor și altor utilizatori, în acest fel, doi utilizatori cu privilegii diferite și le pot combina.
- Privilegiul *alter* poate fi folosit pentru a redenumi tabele, modificând astfel baza de date și făcând astfel inutilizabile programele altor utilizatori.
- Privilegiul *file* poate fi folosit pentru a citi informații sensibile din fișiere de pe server.
- Privilegiul *shutdown* dă posibilitatea opririi serverului de la distanță.
- Privilegiul *process* poate fi utilizat pentru a vedea conținutul interogărilor ce se execută în acel moment, inclusiv cele de setare a parolei. De asemenea el dă posibilitatea opririi forțate a conexiunilor altor utilizatori.

Există și aspecte care nu sunt acoperite de sistemul de privilegii al MySQL, un astfel de exemplu ar fi faptul că nu se poate specifica explicit faptul că unui utilizator i se refuza dreptul de a se conecta. În modelul relațional este realizată asigurarea integrității si consistenței datelor. Aceasta presupune:

- Integritatea entității: aceasta se asigură prin impunerea existenței unei chei primare; de asemenea, o cheie primară nu poate avea valoarea NULL; prin acest mecanism se asigură unicitatea unei înregistrari;

- Integritatea referențială: aceasta presupune menținerea relațiilor între înregistrările din tabele diferite; se asigură prin impunerea mecanismului de cheie străină (din una din tabele) care face referință la o cheie primară (din altă tabelă).
- Integritatea domeniului: orice câmp are o anumită dimensiune sau plajă de valori care corespunde tipului de date folosite pentru acel câmp; aplicația va impune respectarea acestui domeniu de definiție și dimensiunea pentru fiecare câmp al fiecărei tabele;
- Integritatea conform definiției utilizatorului: pentru unele câmpuri se pot impune conform logicii de business a aplicației, restricții care trebuie respectate.

Detalii suplimentare privind controlul accesului, conexiuni criptate, pluginuri pot fi consultate]n manualul de referință la adresa https://dev.mysql.com/doc/refman/5.7/en/security.html , respectiv în tutorialul de la adresa https://www.mysqltutorial.org/mysql-administration/

2.2.CREAREA UTILIZATORILOR ȘI STABILIREA PRIVILEGIILOR

Comanda GRANT. Comanda **GRANT** permite crearea utilizatorilor și/sau stabilirea, respectiv modificarea privilegiilor acestora pe patru nivele de privilegii:

Global - se aplică tuturor bazelor de date existente pe un server
Database - se aplică tuturor tabelelor dintr-o bază de date
Table - se aplică tuturor coloanelor dintr-o tabelă
Column - se aplică doar coloanelor specificate explicit

Forma simplificată a comenzii **GRANT** este:

GRANT privilegii [coloane] **ON** componenta **TO** nume_utilizator
[IDENTIFIED BY 'parola' **][WITH GRANT OPTINS]**

Semnificația clauzelor (clauzele din paranteze drepte sunt opționale) este următoarea:

- **privilegii** - este o lista de privilegii despărțite prin virgulă. Ele pot fi alese dintre cele prezentate anterior (SELECT, INSERT, ALTER ...) plus clauza specială ALL PRIVILEGES (sau simplu ALL) care specifică toate privilegiile posibile. Pentru a crea doar utilizatorul, fără să i se dea nici un drept, se va folosi clauza USAGE**.**
- **coloane** - o listă de una sau mai multe coloane. Permite stabilirea privilegiilor la nivel de coloane.
- **componenta** - este numele unei baze de date sau tabele asupra cărora vor fi stabilite privilegiile. Toate bazele de date se pot specifica prin ***.***. Acest nivel se numeste nivel Global de privilegii. Semnul ***.*** se poate inlocui si cu ***,** dar in acest caz nu trebuie sa existe nici o bază de date selectată, în caz contrar privilegiile se aplică doar asupra bazei de date respective. Nivelul Database se poate specifica prin *nume_bd.**, iar nivelul Table prin *nume_bd.nume_tabela***.** Dacă se specifică doar *nume_tabela* se va interpreta ca fiind o tabelă a bazei de date selectate. Nivelul *Column* se va obține prin folosirea unei liste de coloane nevide.

- **nume_utilizator** - este numele utilizatorului (contul) căruia i se atribuie privilegiile. El poate conține și un nume de mașină fizică de pe care are dreptul să se conecteze. Astfel, utilizatorul "dan" va fi interpretat ca "dan@localhost" si va fi diferit dan@ceva.com**".** Acest lucru este

util pentru că pot exista in realitate utilizatori diferiti cu nume identice dar care lucrează pe stații diferite. Tot de aici se poate restrictiona accesul doar de la anumite stații. Un grup de stații se pot specifica printr-un domeniu generic. Spre exemplu "dan@'%.ro'" inseamna ca utilizatorul "dan" se poate conecta de la orice statie din domeniul ".ro". Numele simplu "dan" este echivalent cu "dan@'%'" și permite conectarea de la orice statie.

- **parola** - specifică parola cu care utilizatorul se va conecta. Parola trebuie în general să nu fie ușor de ghicit (să nu fie chiar numele contului sau un cuvânt din dicționar). Dacă optiunea IDENTIFIED BY lipsește, utilizatorul se va putea conecta fără parolă, ceea ce reprezintă o gravă problemă de securitate.

- **WITH GRANT OPTION** - dă dreptul utilizatorului să ofere privilegii echivalente cu ale sale ,altor utilizatori. Acestă opțiune poate fi utilizată pentru a delega dreptul de administrare, inclusiv crearea de utilizatori, unor administratori ale unor baze de date particulare de pe server. Acești administratori nu vor putea însă să interacționeze cu alte baze de date.

Sintaxa completă de oferire a privilegiilor este :

```
    GRANT     priv_type
[(column_list)]
[, priv_type [(column_list)]] ...
    ON [object_type] priv_level
    TO user_or_role [, user_or_role] ...
    [WITH GRANT OPTION]
    [AS user
        [WITH ROLE
            DEFAULT
          | NONE
          | ALL
          | ALL EXCEPT role [, role ] ...
          | role [, role ] ...
        ]
    ]}

GRANT PROXY ON user_or_role
    TO user_or_role [, user_or_role] ...
    [WITH GRANT OPTION]

GRANT role [, role] ...
    TO user_or_role [, user_or_role] ...
    [WITH ADMIN OPTION]
 object_type: {
TABLE
  | FUNCTION
  | PROCEDURE}

priv_level: {
    *
  | *.*
  | db_name.*
  | db_name.tbl_name
  | tbl_name
  | db_name.routine_name}
```

- Privilegiile globale sunt specificate folosind sintaxa ON *.*și sunt stocate în tabela mysql.user.
- Privilegiile la nivel de bază de date sunt specificate folosind sintaxa ON database_nume.* și sunt stocate în tabela mysql.db. Dacă se folosește sintaxa ON * și există o bază de date implicită, privilegiile specificate se vor referi la aceasta.
- Privilegiile la nivel de tabel sunt specificate folosind sintaxa ON database_name.table_name și sunt stocate în tabela mysql.tables_priv. Există posibilitatea de a nu prefixa denumirea tabelei prin numele bazei de date, dacă există o bază de date implicită și tabela specificată există în contextul acesteia.
- Privilegiile la nivel de coloană se fac prin specificarea atributelor respective, între paranteze, după indicarea drepturilor de acces respective. Acestea sunt stocate în tabela mysql.columns_priv.
- Privilegiul ALL se referă la toate drepturile de acces disponibile la nivelul la care se acordă (global, bază de date, tabel, coloană).
- Pot fi specificate drepturi de acces pentru obiecte care nu există încă, privilegiile fiind aplicate din momentul în care acestea există în baza de date.

Pentru alte exemple de utilizare în alocarea diferențiată a privilegiilor, se va consulta și http://www.mysqltutorial.org/mysql-grant.aspx

După crearea unui utilizator, parola sa se poate schimba prin comanda:

SET PASSWORD FOR utilizator **= PASSWORD**("noua_parola"**);**

Alte exemple pot fi consultate la adresa http://www.mysqltutorial.org/mysql-changing-password.aspx

Vizualizarea privilegiilor. Vizualizarea privilegiilor unui utilizator se poate face prin comanda:

SHOW GRANTS FOR utilizator**;**

Conectarea la server se poate face local sau de la distanță folosind diverși clienti MySQL. Toți acești clienți vor permite trimiterea spre server a informațiilor legate de numele utilizatorului și parola. Clientul standard (din distribuția MySQL) se va folosi astfel: *mysql -h nume_server -u nume_utilizatorp.* După lansarea comenzii clientul va cere parola de conectare: Enter password: *********

Retragerea privilegiilor. Inversul comenzii GRANT este comanda REVOKE. Ea este utilizată pentru a retrage drepturile de la un utilizator. Sintaxa sa este asemănătoare cu cea a comenzii GRANT:

REVOKE privilegii [coloane]
ON componenta
FROM nume_utilizator

Clauzele din această comandă sunt identice cu cele de la comanda GRANT. Pentru a retrage privilegiul acordat prin clauza WITH GRANT OPTION, se va utiliza sintaxa:

REVOKE GRANT OPTIONS
ON componenta
FROM nume_utilizator

Mai multe exemple ce prezintă revocarea diferențiată a privilegiilor alocate pot fi consultate la adresa http://www.mysqltutorial.org/mysql-revoke.aspx

O abordare structurată a întregului sistem de acordare /revocare de privilegii pentru situația în care există un număr mare de utilizatori ai bazei de date și care sunt caracterizați de dinamică în utilizarea acesteia (necesită creare /ștergere repetată) este de a utiliza soluția bazat pe crearea de **roluri în sistem** și alocarea drepturilor acestor **utilizatori abstracți,** respectiv ulterior alocarea acestor roluri utilizatorilor fizici existenti.

Exemple de utilizare și sintaxa pot fi consultate la adresa http://www.mysqltutorial.org/mysql-roles/

Deasemenea, utilizatorii care nu mai posedă responsabilități asupra datelor vor fi eliminați din sistem folosind comanda **DROP.**
Sintaxa și exemple de utilizare pentru comanda DROP, utilizată pentru ștergerea unui utilizator pot fi consultate la adresa http://www.mysqltutorial.org/mysql-drop-user/.

Managementul utilizatorilor și privilegiilor ,în manualul de sistem MySQL
http://dev.mysql.com/doc/workbench/en/wb-mysql-connections-navigator-management-users-andprivileges.html

În concluzie, câteva reguli simple, dar importante pentru asigurarea securității unei baze de date sunt date:

1. protejați user-ul root cu parola;
2. acordati drept de acces la tabela user din mysql doar utilizatorului root;
3. nu stocati parole în text clar în baza de date, folosiți algoritmi de criptare și stocati rezultatul
4. nu acordați niciodată drepturi depline tuturor utilizatorilor;
5. alegeți inteligent parolele;
6. limitați accesul la mașina host ce deține baza de date; întotdeauna acordați fiecărui utilizator **minimum de drepturi** necesare pentru utilizarea aplicației, conform logicii de business ;
7. nu ezitați să revocați orice drepturi acordate inutil sau incorect, pentru a vă asigura că nivelul de securitate necesar este menținut.

III.Probleme propuse

3.1. Pentru baza de date MovieRentals se va analiza soluția de securitate implementată.

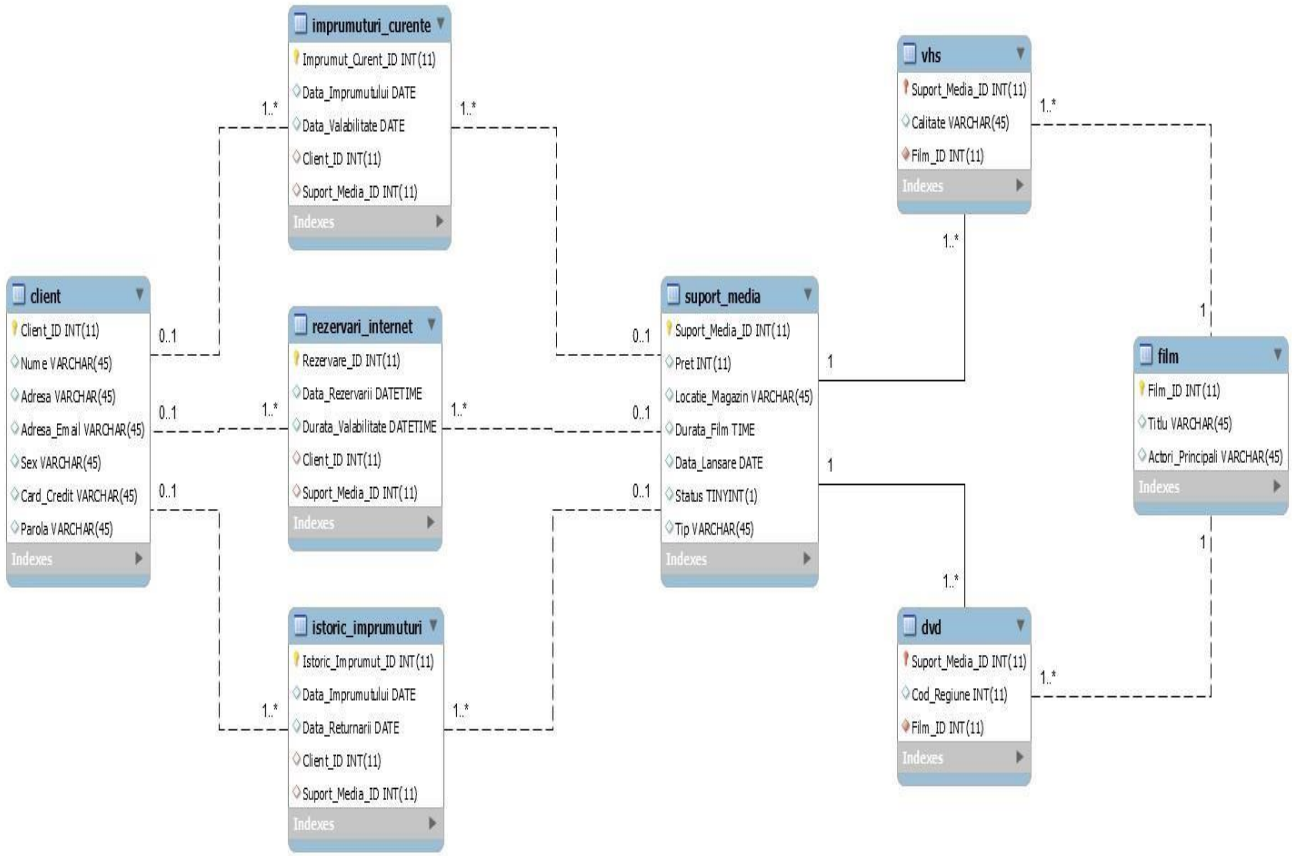
Baza de date trebuie să modeleze în mod corect toate entitățile și relațiile dintre entități pentru a putea susține activitatea unei firme de împrumuturi de filme. Modelul bazei de date trebuie creat în așa fel încât sa ia în considerare orice situație ce poate apărea în acest domeniu de activitate.

- Fiecare suport de stocare media este identificat unic printr-un cod , o categorie de preț și locația sa în magazin, clasele media sunt VHS si DVD și pentru casete video este importantă calitatea, iar pentru DVD codul de regiune.
- Pentru a oferi informatii clientilor, fiecare media are asociata informație minimă despre film.Pe un astfel de suport de stocare este inregistrat un singur film , însă acesta poate exista atât pe caseta cat și pe DVD.
- Un film este definit unic de un identificator, are un titlu si se vor stoca numele actorilor principali.Un astfel de suport media poate fi imprumutat de catre un client , pentru fiecare imprumut se va stoca data de inceput si sfarsit.
- Clientul este identificat de un ID, numele și adresa sa (oraș, strada, cod postal), iar pentru a putea realiza rezervări prin Internet se va utiliza cardul de credit

- Un client poate realiza mai multe împrumuturi, fără a fi necesară înregistrarea unui istoric al împrumuturilor .

Baza de date trebuie să satisfacă următoarele funcționalități:

- Să stocheze toți clientii firmei
- Să stocheze toate împrumuturile curente, precum și rezervările pe internet
- Să stocheze un istoric al împrumuturilor
- Să stocheze toate suporturile media a firmei, precum și datele specifice acestora
- Să conțină proceduri pentru a modifica datele existente: adăugarea unui nou client, unui nou împrumut, unei noi rezervări internet, unui nou suport media, ștergerea unui împrumut sau a unei rezervări internet



Se vor implementa următoarele proceduri, vederi și triggere și se va securiza accesul celor doi utilizatori client si furnizor.

1. **ADAUGARE_CLIENT_NOU**(Nume, Adresa ,Adresa_Email, Sex, Card_Credit, Parola)
   Insereaza un client nou.
   Nume- numele clientului

Adresa – adresa clientului
Adresa_Email – adresa email a clientului
Sex – sexul clientului
Card_Credit – cardul de credit al cleintului
Parola – parola de logare in aplicatia java a clinetului

2. **ADAUGARE_IMPRUMUT_NOU**(Client_ID, Data_Valabilitate, Suport_Media_ID)
   Adauga un imprumut nou.
   Client_ID – id clientului care realizeaza imprumutul
   Data_Valabilitate – data de valabilitate a imprumutului
   Suport_Nedia_ID – id-ul suportului media pe care clientul il imprumuta

3. **ADAUGARE_VHS_NOU**(Suport_Media_ID, Pret,Locatie_Magazin, Durata_Film, Data_Lansare, Calitate, Film_ID)
   Adauga un nou VHS.
   Suport_Media_ID – id-ul suportului media ce va fi introdus in baza de date
   Locatie_Magazin – locatia in magazin a vhs-ului
   Durata_Film – durata filmului
   Data_Lansare – data de lansare a filmului
   Calitate – calitatea vhs-ului
   Film_ID – id-ul filmului ce va fi continut pe acest vhs

4. **ADAUGARE_DVD_NOU**(Suport_Media_ID, Pret, Locatie_Magazin, Durata_Film, Data_Lansare, Cod_Regiune, Film_ID)
   Adauga un nou DVD.
   Suport_Media_ID – id-ul suportului media ce va fi introdus in baza de date
   Locatie_Magazin – locatia in magazin a vhs-ului
   Durata_Film – durata filmului
   Data_Lansare – data de lansare a filmului
   Cod regiune – codul de regiune al dvd-ului
   Film_ID – id-ul filmului ce va fi continut pe acest vhs

5. **ADAUGARE_REZERVARE_INTERNET**(Client_ID, Suport_Media_ID, Durata_Valabilitate)
   Adauga o noua rezervare pe internet.
   Client_ID – id-ul clientului care realizeaza rezervarea
   Suport_Media-ID – id-ul suportului media care va fi rezervat
   Durata_Valabilitate – durata de valabilitate a rezervarii

6. **ADAUGARE_FILM**(Film_ID, Titlu, Actori_Principali)
   Adauga un nou film.
   Film_ID – id-ul filmului care va fi adaugat
   Titlu – titlul filmului ce va fi adaugat
   Actori_Principali – actorii principali ai filmului

7. **RETURNEAZA_SUPORT_MEDIA**(Client_ID, Imprumut_Curent_ID)
   Returneaza un suport media.
   Client_ID – id-ul clientului care returneaza suportul media
   Imprumut_Curent_ID – id-ul imprumutului care va fi returnat

8. **ANULEAZA_REZERVARE_INTERNET**(Client_ID, Rezervare_ID)
   Anuleaza o rezervare de pe internet.
   Client_ID – id-ul clientului care anuleaza rezervarea.
   Rezervare_ID – id-ul rezervarii ce va fi anulata.

Vederi

1. **Informatii_VHS**: Contine Titlu, Actori_Principali, Tip, Durata_Film, Data_Lansare, Pret
2. **Informatii_DVD**: Contine Titlu, Actori_Principali, Tip, Durata_Film, Data_Lansare, Pret
3. **Informatii_Suport_Media**: Contine Suport_Media_ID, Titlu, Actori_Principali, Tip, Durata_Film, Data_Lansare, Pret, Status
4. **DVD_VHS**: Repezinta un join intre tabele DVD si VHS
5. **Informatii_Suport_Media_Pt_Rezervari_Internet**: Contine Suport_Media_ID, Titlu, Tip, Pret. Este un view support pentru view-ul **Informatii_Rezervari_Internet.**
6. **Informatii_Rezervari_Internet**: Contine Rezervare_ID, Nume, Titlu, Tip, Data_Rezervarii, Durata_Valabilitate

Triggere:

1. **verifica_rezervari_internet:** La fiecare inserar in tabelul Rezervari_Internet, verifica validitatea noii inregistrari(data rezervarii trebuie sa fie mai mare decat durata de valabilitate).
2. **verifica_imprumuturi_curente:** La fiecare inserare in tabelul Imprumuturi_Curente, verifica validitatea noii inregistrari(data imprumutului trebuie sa fie mai mare decat data de valabilitate).

Securizarea userilor client și furnizor pe procedurile lor specific se poate realiza cu ajutorul scriptului următor :

```
DROP USER 'client'@localhost;
CREATE USER 'client'@'localhost';
GRANT SELECT ON movierental.* TO 'client'@'localhost';
GRANT INSERT ON movierental.* TO 'client'@'localhost';
GRANT UPDATE ON movierental.* TO 'client'@'localhost';
GRANT DELETE ON movierental.* TO 'client'@'localhost';
GRANT SHOW VIEW ON movierental.* TO 'client'@'localhost';
GRANT EXECUTE ON PROCEDURE movierental.ADAUGARE_IMPRUMUT_NOU TO
'client'@'localhost';
GRANT EXECUTE ON PROCEDURE movierental.ADAUGARE_CLIENT_NOU TO
'client'@'localhost';
GRANT EXECUTE ON PROCEDURE movierental.ADAUGARE_REZERVARE_INTERNET TO
'client'@'localhost';
GRANT EXECUTE ON PROCEDURE movierental.RETURNEAZA_SUPORT_MEDIA TO
'client'@'localhost';
GRANT EXECUTE ON PROCEDURE movierental.ANULEAZA_REZERVARE_INTERNET TO
'client'@'localhost';

DROP USER 'furnizor'@localhost;
CREATE USER 'furnizor'@'localhost';
GRANT SELECT ON movierental.* TO 'furnizor'@'localhost';
GRANT INSERT ON movierental.* TO 'furnizor'@'localhost';
GRANT UPDATE ON movierental.* TO 'furnizor'@'localhost';
GRANT DELETE ON movierental.* TO 'furnizor'@'localhost';
```

```
GRANT SHOW VIEW ON movierental.* TO 'supplier'@'localhost';
GRANT EXECUTE ON PROCEDURE movierental.ADAUGARE_CLIENT_NOU TO
'furnizor'@'localhost';
GRANT EXECUTE ON PROCEDURE movierental.ADAUGARE_VHS_NOU TO
'furnizor'@'localhost';
GRANT EXECUTE ON PROCEDURE movierental.ADAUGARE_DVD_NOU TO
'furnizor'@'localhost';
GRANT EXECUTE ON PROCEDURE movierental.ADAUGARE_FILM TO
'furnizor'@'localhost';
```