--Task1

--Large objects (photos, videos, CAD files, etc.) are stored as a large object:

-- •blob: binary large object -- object is a large collection of uninterpreted

-- binary data (whose interpretation is left to an application outside of the database system)•

-- clob: character large object

-- object is a large collection of character data

--Task2

--You can give users access to tables. These privileges can be any combination

--of SELECT, INSERT, UPDATE, DELETE, REFERENCES, ALTER, INDEX, or others

--A role is a way to distinguish among various users as far as what

--these users can access/update in the database.

--A user is a database level security principal.

--Logins must be mapped to a database user to connect to a database.

```
--2.1

create role accountant;

create role administrator;

create role support;

grant select on accounts, transactions to accountant;

grant update on transactions to accountant;

grant delete on transactions to accountant;

grant all privileges on accounts, customers, transactions to administrator;

grant select on accounts, customers, transactions to support;



--2.2

create user student;

grant accountant to student;



create user professor;
```

```
grant administrator to professor;

--2.3

grant support to accountant;

--2.4

revoke select on transactions from student;

--Task3

alter table accounts

add constraint acc_con check (accounts.currency is not null);

--Task5

--5.1
```

```
create unique index acc_ind on accounts(account_id, currency);

--5.2

create index acc_tran on accounts(currency, balance);



--Task6

begin;



create view task6 as

   select src_account,dst_account,amount,status

   from transactions

   where transactions.status='init';



create view task62 as

   select transactions.src_account,transactions.dst_account,accounts.balance+ transactions.amount as now
```

```
    from transactions,accounts

    where transactions.status='init' and transactions.dst_account = accounts.account_id;




create view task622 as

    select transactions.src_account,transactions.dst_account,accounts.balance- transactions.amount as now

    from transactions,accounts

    where transactions.status='init' and transactions.src_account = accounts.account_id;




update transactions

set status='commited'

where status='init';



rollback;




commit;
```