For the following SQL schema:

create table Building (

id int primary key,

address varchar(5000) not null

) engine=InnoDB;

create table Apartment (

id int primary key,

number varchar(31) not null,

building int not null references Building(id)

on update cascade on delete cascade

) engine=InnoDB;

create table RoomType (

id int primary key,

description varchar(5000)

) engine=InnoDB;

create table RoomTypeName (

type int references RoomType(id)

on update cascade on delete cascade,

name varchar(255),

primary key(type, name)

) engine=InnoDB;

create table Room (

id int primary key,

area double not null,

type int not null references RoomType(id)

on update cascade,

apartment int not null references Apartment(id)

on update cascade on delete cascade

) engine=InnoDB;

create table ApartmentFloor (

apartment int references Apartment(id)

on update cascade on delete cascade,

floor int,

primary key(apartment, floor)

) engine=InnoDB;

create table Person (

id int primary key,

name varchar(5000) not null

) engine=InnoDB;

create table Owner (

person int references Person(id)

on update cascade on delete cascade,

apartment int references Apartment(id)

on update cascade on delete cascade,

primary key(person, apartment)

) engine=InnoDB;

Part A.

Assume that: the following are the sizes, counts and times:

1. A record requires an average of 6 bytes administrative overhead, including the space required for specifying the rid.
2. An int uses 4 bytes.
3. A double uses 8 bytes.
4. A Date uses 8 bytes.
5. A building address uses an average of 50 bytes.
6. An apartment number uses an average of 10 bytes.
7. A person name uses an average of 25 bytes.
8. A room description uses an average of 100 bytes.
9. A room type name uses an average of 10 bytes.
10. There are 1K buildings.
11. There are 20K apartments.
12. There are 100K rooms.
13. There are 20 room types and an average of 5 names per room type.
14. The disk block size is 8K = 8192 bytes.
15. A random disk access requires 6ms.
16. A page can be transferred (read or write) in 0.05 ms = 50 microseconds.

Compute the approximate time required for every query listed below for each of the following 4 table structures:

1. Clustered Heap. An unordered collection of records packed into sequential blocks.
2. Clustered Sort. An ordered collection of records packed into sequential blocks. In each case, order it by the field being queried (e.g., for the first query assume ordered by composer id).
3. Clustered B-tree. The ordering and clustering is by the field being queried.
4. Unclustered heap with index. Assume the index is a B-tree index in which only one level of the tree is on disk, and the other levels are in main memory. Unclustered means the following:
   1. The records are randomly located with respect to the query field.
   2. The blocks are randomly located on the disk.
   3. The blocks are only 2/3 full.

Here are the queries:

1. Find all living rooms (room type is named "Living Room" for which one may assume there is only one room type).
2. Find all apartments at "100 Main Street".

Part B.

Show a query evaluation plan for each of the following queries. Note that a "query evaluation plan" includes strategies.

1. Find the addresses of all buildings that have a Living Room with area greater than 1000 s.f. owned by Fred in an apartment on the fifth floor or higher.
2. select distinct b.address
3. from Apartment a, Building b, ApartmentFloor f, RoomTypeName n, Owner o, Person p, Room r, RoomType t
4. where a.building = b.id
5. and f.apartment = a.id
6. and f.floor >= 5
7. and n.type = t.id
8. and n.name = 'Living Room'
9. and o.apartment = a.id
10. and o.person = p.id
11. and p.name = 'Fred'
12. and r.type = t.id
13. and r.area > 1000
14. and r.apartment = a.id
15. Find all apartments, showing the address and apartment number that have 3 or more rooms with the same type.
16. select b.address, a.number
17. from Building b, Apartment a
18. where b.id = a.building
19. and exists (
20. select \*
21. from RoomType t
22. where 3 <= (select count(\*)
23. from Room r
24. where r.type = t.id
25. and r.apartment = a.id)
26. )
27. List the buildings in order by address together with the number of apartments in each building.
28. select b.address, count(\*)
29. from Building b, Apartment a
30. where b.id = a.building
31. group by b.id

order by b.address