1. Transferring data from a database to an application's data structures requires us to:

1.Parse a set of results, which are obtained following a query to the database

2.Convert these results to a type (int, String...) that can be put in one of the application's data structures

Those 2 actions have a cost for a computer, which we call overhead cost. What are the two aspects of the overhead cost that were mentioned in class?

\*A result set (rs) in which to put the results of the query to the database

 Parsing that set of results increases the overhead cost

\*An ArrayList (listeInfos) in which to put the result set's results

 Before being put in the ArrayList, the result of a query has to be converted to a type compatible with the app (which increases the overhead cost)

a) create a table named RupeeStore including the columns: storeID, storeName, and storeType.

Sol: create table RupeeStore (storeID int,storeName varchar(255),storeType varchar(255));

b) create a table named House including the columns: houseID, address, and another attribute which cannot ever have a null value in any row.

Sol : create table House (houseID int,address varchar(255),size int NOT NULL);

c) change the RupeeStore table so that the column storeID becomes a primary key.

Sol : alter table RupeeStore add primary key (storeID);

d) add to the House table a column named closestStore, which is a foreign key linked to the primary key storeID of the table RupeeStore.

Sol : alter table House add closestStore varchar(255) ,add foreign key (closestStore) references RupeeStore(storeID);

e) create a table named Customer including the columns: customerID, firstName, LastName, and age. Note: all customers have to be 16 years old or older.

Sol : create table Customer (customerID int,firstName varchar(255),LastName varchar(255),age int,check (age>=16));

f) make up a default name for the rupee store.

Sol : alter table RupeeStore alter storeID set dafault “god”;

g) insert two stores into the RupeeStore table (with values in all the columns).

Sol : insert into RupeeStore values (1,”Dollarama”,”grocery”);

insert into RupeeStore values (2,”Winners”,”clothing”);

h) insert two houses into the House table (with values in all the columns).

Sol : insert into House values (1,”DDO”,42,2);

insert into House values (2,”Downtown”,55,3);

i) insert two customers into the Customer table (with values in all the columns).

Sol :insert into House values (1,”Guru”,"God”,100000);

insert into customer values (2,”Davinder singh”,”kharoud,18);

j) ask MySQL for all the customers in the Customer table.

Sol : select \* from customer;

k) ask MySQL for the stores' names in the RupeeStore table.

Sol : select storeName from customer;

3. Look at the database represented in this E-R diagram: Then:

• Create a database called 'Commerce'.

Sol: create database Commerce;

• In that database, create the MySQL tables that correspond to the entities and attributes represented in the diagram.

◦ In each table, you will have to put column names corresponding to the attributes' names.

Sol: create table order (date varchar(255),number int not null,primary key(number));

create table product (ProductName varchar(255),SalePrice int,ProductId int not null,quantity int,primary key(ProductId));

◦ Do not forget to add constraints for the primary keys, either at table creation or by altering the table.

• Using the INSERT INTO command, insert two orders into the Order table and insert two products into the Product table.

Insert into order values (22,1,1);

Insert into order values (23,2,2);

Insert into products values (“Biscuits”,22,1,32);

Insert into products values (“candy”,23,2,33);

• Using the SELECT command, make at least two different queries to the database.

Select \* from order;

Select SalePrice from product;