For this assignment, you will want to test your queries by running them on a real database. So please begin by signing up for an Oracle account (http://www.seas.upenn.edu/ora), accessible from eniac. (For those who do not have eniac accounts, please email the instructor.) Then read over the Oracle setup instructions from the course web page (http://www.seas.upenn.edu/~zives/cis550/oracle-faq.html) and modify your eniac .profile file as directed. Also read over the Oracle guide referenced from the course web pages (http://www.cs.wisc.edu/~dbbook/openAccess/thirdEdition/Oracle/user_guide/oracle_guide.html). Finally, download hw2.sql to your eniac account, launch Oracle (using the command sql), and then start hw2 to create some sample tables for Problems 1 and 3. These will only be sparsely populated to test your solutions, you may need to INSERT more VALUES into the tables.

**Note 1**: The SQL string datatype is VARCHAR(length), and you’ll need to choose an appropriate length.

**Note 2**: The key fields are underlined in schema. Foreign keys are indicated by naming. (In other words, if x is the key of relation X, then each appearance of x outside of X is a foreign key referencing X).

**Problem 1 [30 points]**: Consider the following schema based on the TPC-H benchmark (which you’ll hear more about later on in the course):

Parts(partID: int, name: string, mfgr: string, brand: string, type: string, retailprice: float)
Suppliers(suppID: int, name: string, address: string, nationID: int, phone: string, acctbal: float)
PartSupp(partID: int, suppID: int, availqty: int, supplycost: float)
Nation(nationID: int, name: string, regionID: int)
Region(regionID: int, name: string)

Write the following queries in SQL:

1. Find the IDs of parts available from a single supplier with quantity > 500.

```sql
select partID
from PartSupp
```
2. Find the IDs of parts available with quantity > 500 considering all suppliers.

```sql
select partID
from PartSupp
group by partID
having sum(availqty) > 500
```

3. Find total number of suppliers in each region.

```sql
select r.regionID, r.name, count(s.suppID) as num_suppliers
from Suppliers s, Nation n, Region r
where s.nationID = n.nationID
  and n.regionID = r.regionID
group by r.regionID, r.name
```

4. Find the IDs of parts which are supplied by suppliers from different nations.

```sql
select a.partID
from PartSupp a, Suppliers s
where a.suppID = s.suppID
  group by a.partID
having count(distinct s.nationID) >= 2
```

5. \{\langle s, a, t, h, b, p, e, m, r, y, i, v, c\rangle \in supplier \land \langle p, s, v, c\rangle \in partsupp \land \langle p, e, m, r, y, i\rangle \in parts \land y = "generic" \land v > 500\}\}

```sql
select distinct s.name
from Suppliers s, PartSupp a, Parts p
where s.suppID = a.suppID
  and a.partID = p.partID
  and p.type = 'generic'
  and a.availqty > 500
```

**Problem 2[10 points]**: Consider the following Inverted Index schema:

Word(\text{wordid}: \text{int}, \text{wordname}: \text{string})
Write the SQL DDL statements to create these relations, including all primary and foreign key integrity constraints.

```sql
create table Word (
    wordid integer,
    wordname varchar(64),
    primary key (wordid));

create table DocumentURL(
    docid integer,
    url varchar(1024),
    primary key (docid));

create table Occurs (
    wordid integer,
    docid integer,
    foreign key (wordid) references Word,
    foreign key (docid) references DocumentURL);

Problem 3[60 points]: Use the schema from Homework 1’s PBAY system:

Sellers(sellerID: int, rating: char, email: string)
Items(itemID: int, type: string)
Buyers(buyerID: int, email: string, city: string, state: string)
Stock(itemID: int, sellerID: int, startBid: float, quantity: int, endingTime: int)
Purchases(itemID: int, buyerID: int, sellerID: int, price: float, purchaseQuantity: int, bidTime: int)

Write the following queries in SQL (from Problem 1 and Problem 2 of Homework 1):

1. Find the IDs of items purchased for price < $50.

   ```sql
   select distinct itemID
   from Purchases
   where price < 50
   ```

2. Find the emails of buyers from PA who buy items with purchaseQuantity > 3.

   ```sql
   ```
select distinct b.email
from Buyers b, Purchases p
where b.buyerID = p.buyerID
    and b.state = 'PA'
    and p.purchaseQuantity > 3

3. Find the IDs of buyers who purchased items of purchaseQuantity less than 10% of the quantity provided by the same seller the buyer purchase from in the stock.

    select distinct p.buyerID
    from Stock s, Purchases p
    where p.itemID = s.itemID
        and p.sellerID = s.sellerID
        and p.purchaseQuantity < 0.1 * s.quantity

4. Find the IDs of buyers who purchased items with type “furniture” for over 10% of the startBid price of the items they bought.

    select distinct p.buyerID
    from Items i, Stock s, Purchases p
    where i.itemID = s.itemID
        and s.itemID = p.itemID
        and s.sellerID = p.sellerID
        and p.price > 1.1 * s.startBid
        and i.type = 'furniture'

5. Find the IDs of buyers who either always make purchases with purchaseQuantity < 5 or haven’t made any purchases.

    select buyerID
    from Buyers
    minus
    select buyerID
    from Purchases
    where purchaseQuantity >= 5

6. Find the types of items stocked by ≥ 2 sellers or bought by ≥ 2 buyers.
select i.type  
   from Items i, Stock s 
   where i.itemID = s.itemID 
   group by i.type 
   having count(distinct s.sellerID) >= 2
union 
select i.type  
   from Items i, Purchases p 
   where i.itemID = p.itemID 
   group by i.type 
   having count(distinct p.buyerID) >= 2

7. \{ Q \mid \exists P \in Purchase, \exists S \in Sellers (S.rating = "A" \land P.sellerID = S.sellerID \land Q.buyerID = P.buyerID \land P.purchaseQuantity = 2) \}\}

select p.buyerID  
   from Purchases p, Sellers s 
   where p.sellerID = s.sellerID 
   and p.purchaseQuantity = 2 
   and s.rating = 'A'

8. \{ < e > \mid \exists i, s(\exists r(\exists s, r, e \in Sellers) \land \exists d, q, n(\exists i, s, d, q, n \in Stock \land (d < 20) \land (q = 5)) \land \exists b, p, u, m(\exists i, b, s, p, u, m \in Purchase \land (p > 50))) \}\}

select distinct s.email 
   from Sellers s, Stock t, Purchases p 
   where s.sellerID = t.sellerID 
   and t.sellerID = p.sellerID 
   and t.itemID = p.itemID 
   and t.startBid < 20 
   and t.quantity = 5 
   and p.price > 50

9. \pi_{email}(\sigma_{city="philadelphia"}(Buyers) \bowtie \pi_{buyid}(\sigma_{price<2*startBid}(\sigma_{type="book"\land purchaseQuantity=2}(Items \bowtie Purchase) \bowtie Stock)))

select distinct b.email 
   from Buyers b 
   where b.city = 'philadelphia'
and b.buyerID in (select p.buyerID
from Items i, Purchases p, Stock s
where p.price < 2 * s.startBid
and i.type = 'book'
and p.purchaseQuantity = 2
and i.itemID = p.itemID
and p.itemID = s.itemID
and p.sellerID = s.sellerID)

10. \( \pi_{\text{rating}}(\pi_{s1}(\sigma_{i1 \neq i2 \land s1 = s2} (\rho_{\text{itemID} \rightarrow i1, \text{sellerID} \rightarrow s1} (\text{Stock}) \bowtie \rho_{\text{itemID} \rightarrow i2, \text{sellerID} \rightarrow s2} (\sigma_{\text{quantity} \geq 3} (\text{Stock})))))) \bowtie_{s1 = \text{sellerID}} \text{Sellers} \)

select distinct s.rating
from Sellers s, Stock t
where s.sellerID = t.sellerID
and t.quantity >= 3
and exists (select t2.sellerID
from Stock t2
where t2.sellerID = t.sellerID
and t2.itemID <> t.itemID)