Sample Solutions to
COMP9311 03s1 Final Exam

Part 1: Multiple Choice

1.
   a. i
   b. iii
   c. iii
   d. ii
   e. i
   f. iii
   g. ii
   h. i
   i. iv
   j. iii

Part 2: Long Answer

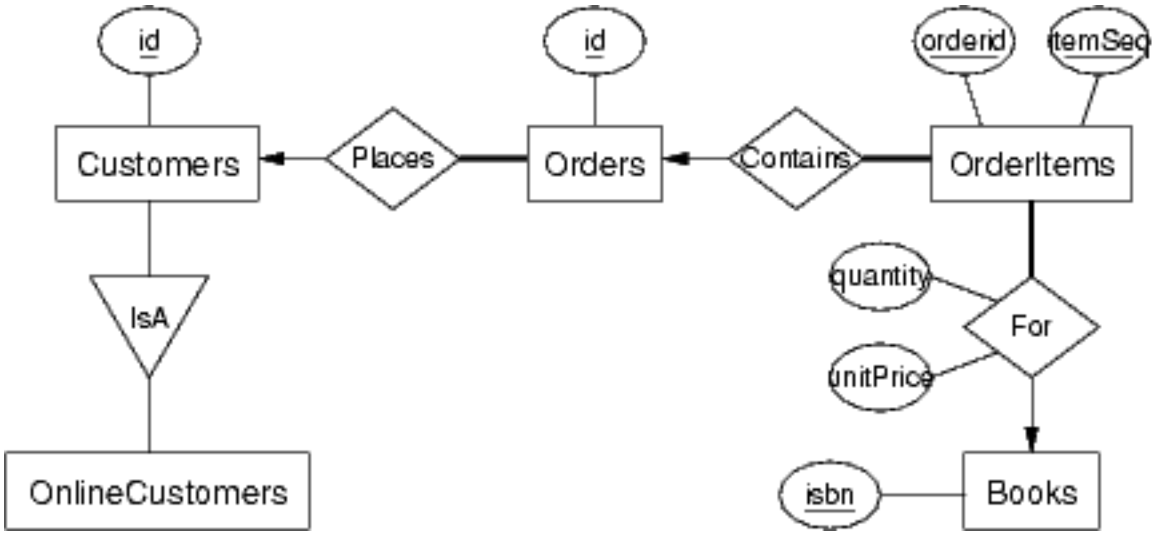
Marking Scheme: Design/programming questions are marked using an A-E grading scheme

A	correct solution, with no problems at all
B	almost-correct solution, but has non-trivial syntax problems and/or some semantic problems as well
C	partly correct solution, but with major flaws
D	has some elements of a correct solution, but is basically a mess
E	hopeless ... has some correct syntax, not much else
.	did not attempt the question

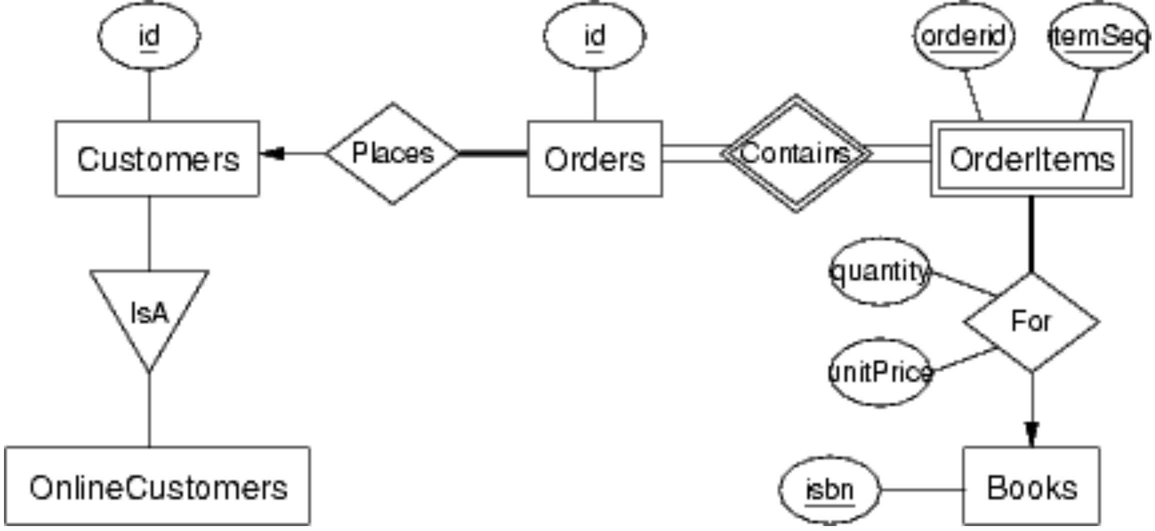
2. Since this involves an ER diagram, there are a number of plausible solutions. Three of these are below. There may well be other valid solutions that I haven't thought of.

   Note that some things that might be regarded as incorrect from the samples below, would be treated as *correct* if there is an accompanying assumption or justification.

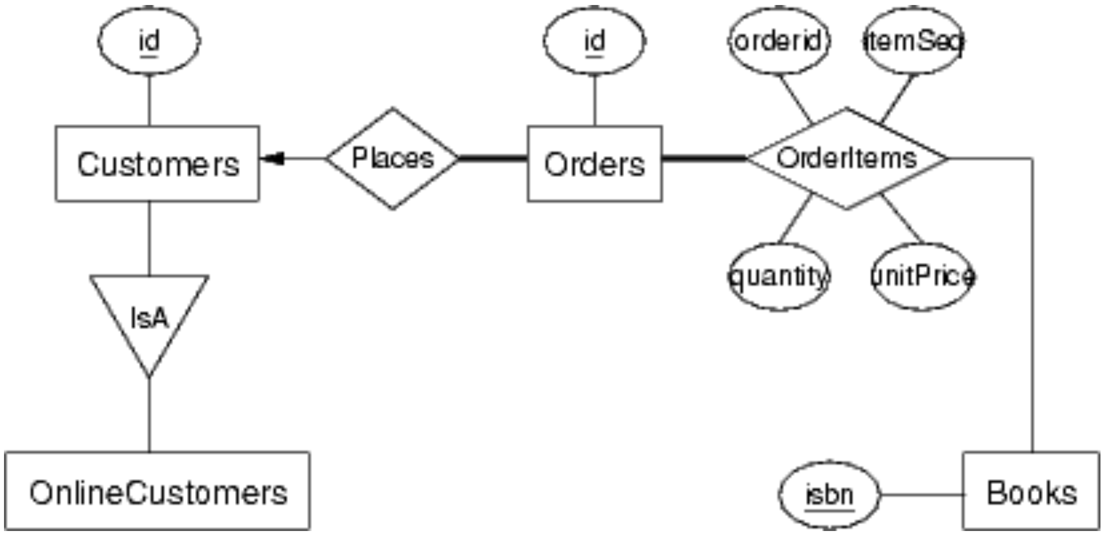
   Version 1: with OrderItems as a full entity



   Version 2: with OrderItems as a weak entity



Version 3: with OrderItems as a relationship



In the examples where OrderItems are an entity, the attributes for quantity and unitPrice could equally-well be attributes of OrderItems or of the relationship between OrderItems and Books.

Use any of the recognised notations for indicating cardinality and participation constraints is allowed, not just the notation above.

3. The simplest answer to this problem is to quote just the "obvious" FDs, based on the primary keys, and then conclude that the schema is in 3NF. This solution would look something like:

```
Books:
isbn -> title,authors,publisher,yearPublished,price,qtyInStock

Customers:
id -> name,address,phone

OnLineCustomers:
id -> username,password,email

Orders:
id -> customer,datePlaced,discount,dateShipped,datePaid

OrderItems:
orderid,itemSeq -> bookisbn,quantity,unitPrice
```

Prooof of 3NF:

If the above "obvious" FDs are the ones given, then the proof of 3NF has to say something like ...

In each table, all non-key attributes in each relation are functionally dependent on the entire key

=> each table is in 3NF => the schema is 3NF

This solution is awarded an A grade.

If some steps are omitted (e.g. don't say "for each table", or don't say "each table is 3NF, etc.), then B grade.

For lower grades, based on how much of the above solution was given.

It's ok (but slightly irrelevant) to mention that there is enough "linkage" between the tables so that the data can be put together again. (If we'd started from a single "super-table" and done a decomposition, this comment would make more sense).

More complex answers might consider additional *possible* FDs. If they do, each FD has to be supported by an assumption, e.g.

```
OnLineCustomers:
id -> username,password,email

[ email -> username ] must also have assumption only one account per user
[ username -> email ] must also have assumption only one account per user
```

If these FDs are included, then ought to conclude that the schema is *not* in 3NF, using the argument:

Some tables (e.g. OnLineCustomers) have non-key attributes that are functionally dependent on other non-key attributes => this table (and thus the whole schema) is not in 3NF (in fact, not even 2NF)

A complete argument like this, is awarded an A grade.

Lower grades are awarded if omitted assumptions to justify other FDs or skipped some of the "obvious" FDs or skipped parts of the proof.

4.

a. How many different books does the bookstore have in stock?

```
select count(distinct title||authors)
from   Books
where  qtyInStock > 0;
```

b. Which books have multiple editions in stock?

```
select title,authors
from   Books
group  by title,authors
having count(distinct yearPublished) > 1;
```

c. Email addresses of customers with unpaid but shipped orders

```
select distinct c.email
from   OnLineCustomers c, Orders o
where  c.id=o.customer and o.dateShipped is not null and o.datePaid is null;
```

d. How many orders have been placed by each on-line customer (incl zero)?

```
-- using PostgreSQL syntax
select c.id,c.username,count(o.id)
from   OnLineCustomers c left outer join Orders o on (c.id=o.customer)
group  by c.id,c.username;
```

```
-- using Oracle 8 syntax
select c.id,c.username,count(o.id)
from   OnLineCustomers c, Orders o
where  c.id = o.customer(+)
group  by c.id,c.username;
```

e. Price increase of 10% on all books whose value is less than $100

```
update Books
set    price = price * 1.10
where  price < 100.00;
```

5.

a. Can an order be met from book stocks in warehouse?

```
create or replace function
        shippable(ordid integer) return varchar
is
        n integer;
        ok varchar(10);
begin
        select id into n from Orders where id=ordid;
        ok := 'true';
        n := 0;
        for it in
                (select b.isbn,i.quantity,b.qtyInStock
                 from   OrderItems i, Books b
                 where  i.orderid=ordid and i.bookisbn=b.isbn)
        loop
                n := n + 1;
                if (it.quantity > it.qtyInStock) then
                        ok := 'false';
                end if;
        end loop;
        return ok;
exception
        when no_data_found then
                dbms_output.put_line('No such order: '||ordid);
                return 'false';
end;
/

-- OR

create or replace function
        shippable(ordid integer) return varchar
is
        n integer;
begin
        select id into n from Orders where id=ordid;
        select count(isbn) into n
        from   OrderItems i, Books b
        where  i.orderid=ordid and i.bookisbn=b.isbn
               and i.quantity > b.qtyInStock;
        return (n = 0);
exception
        when no_data_found then
                dbms_output.put_line('No such order: '||ordid);
                return 'false';
end;
/
```

b. Total value of books on an order

```
create or replace function
        orderAmount(ordid integer) return real
is
        val real;
        tot real;
        discnt real;
begin
        select id into n from Orders where id=ordid;
        select discount into discnt from Orders where id=ordid;
        tot := 0.00;
        for it in
                (select quantity,unitprice
                 from   OrderItems
                 where  orderid=ordid)
        loop
                val := it.quantity * it.unitprice;
                tot := tot + val;
        end loop;
        tot := tot * (1.0 - discnt);
        return tot;
exception
        when no_data_found then
                dbms_output.put_line('No such order: '||ordid);
                return null;
end;
/

-- OR

create or replace function
        orderAmount(ordid integer) return real
is
        tot real;
begin
        select id into n from Orders where id=ordid;
        select sum(i.quantity*i.unitprice*(1.0-discount)) into tot
        from   OrderItems i, Orders o
        where  i.orderid=ordid and o.id=i.orderid;
        return tot;
exception
        when no_data_found then
                dbms_output.put_line('No such order: '||ordid);
                return null;
end;
/
```

c. Trigger to set discount in new orders

```
create trigger orderInsertTrigger
before insert on Orders
for each row
declare
        nords  integer;
        discnt real;
begin
        select count(*) into nords
        from   Orders o
        where  o.customer = :new.customer
               and orderAmount(o.id) > 100
               and (sysdate()-o.datePlaced) < 90;
        discnt := nords * 0.05;
        if (discnt > 0.3) then discnt := 0.3; end if;
        :new.discount := discnt;
end;
/
```

6.

a.
```
select * from Orders where id = 7654321;
```
create index SomeName on Orders (id);

b.
```
select * from Orders where customer = 123456;
```
create index SomeName on Orders (customer);

c.
```
select authors from Books where title like '%Database%';
```
No index can help with "like" queries

d.
```
select title from Books where price between 50.00 and 100.00;
```
create index SomeName on Books (price);

e.
```
select name,datePlaced from Orders,Customers where Orders.customer =
Customer.id;
```
existing index on Customers is ok
create index on Orders (customer);

7.

a. Which grant fails? Why does it fail?

Failed operation ... C: grant select on Orders to D;
Why it fails ... because C didn't get the [select] privilege with grant option.

b. Which privileges apply after all grants?

	select	insert	delete	update
A	v	v	v	v
B	v	v	v	v
C	v	v	v	v
D	X	X	X	v
E	v	v	v	X

c. Which privileges apply after all revokes?

	select	insert	delete	update
A	v	v	v	v
B	v	v	v	v
C	X	v	v	X
D	X	X	X	X
E	v	v	v	X