CISC 7510X Final Exam

For the below questions, use the following schema definition.

```
customer(custid,username,password,fname,lname,ssn);
account(acctid,custid,type,name)
journal(acctid,tim,transactid,amnt)
```

It is a schema for an online bank, where customers use a username & password to login, see a list of accounts, and each account can have multiple journal entries. Each transaction from account to account is recorded by *two* journal entries, one taking money from an account (negative number) and another adding it to another account (positive number); each transaction total therefore sums to zero. There's a special account with `acctid=0`, which we'll call "cash" account. Every time money comes in via cash, it is taken out of that 'cash' account, every time money goes out as cash, it is put back into 'cash'.

Pick the best answer that fits the question. Not all of the answers may be correct. If none of the answers fit, write your own answer.

1. (5 points) Transfer $500 from account 1 to account 2.

   (a)
   ```
   insert into journal values (2,now(),12345,500);
   insert into journal values (0,now(),12345,-500);
   ```

   (b)
   ```
   insert into journal values (1,now(),12345,-500);
   insert into journal values (2,now(),1,500);
   ```

   (c)
   ```
   insert into journal values (1,now(),12345,-500);
   insert into journal values (2,now(),12345,500);
   ```

   (d)
   ```
   insert into journal values (1,now(),12345,500);
   insert into journal values (2,now(),12345,-500);
   ```

   (e) Other:

2. (5 points) Find balance for account 1.

   (a) `select sum(amnt) from journal where acctid=1;`

   (b) `select transactid,sum(amnt) from journal group by transactid`

   (c) `select sum(amnt) from journal where amnt > 0`

   (d) `select max(amnt) from journal where acctid=1`

   (e) Other:

3. (5 points) Find the total amount of money in all accounts.

   (a) `select sum(amnt) from journal`

   (b) `select sum(amnt) from journal where amnt > 0`

   (c) `select sum(amnt) from journal where amnt > 0 and acctid = 0`

   (d) `select sum(amnt) from journal where acctid > 0`

   (e) Other:

4. (5 points) Find incomplete/corrupted transactions.

   (a) `select transactid from journal where sum(amnt) <> 0`

(b) `select sum(amnt) from journal group by transactid having sum(amnt) <> 0`

(c) `select sum(amnt) from journal where amnt <> 0`

(d) `select transactid from journal group by transactid having sum(amnt) <> 0`

(e) Other:

5. (5 points) Find all transactions of more than $10000.

(a) `select transactid from journal where sum(amnt) > 10000`

(b) `select transactid from journal where amnt>0 group by transactid having sum(amnt) > 10000`

(c) `select max(transactid) from journal where amnt > 10000`

(d) `select sum(amnt) from journal group by transactid having sum(amnt) > 10000`

(e) Other:

6. (5 points) Assuming account type 2 is savings, find all savings accounts with less than $500.

(a) `select acctid from account inner join journal using(acctid) group by acctid having sum(amnt) < 500 and type=2`

(b) `select acctid from account inner join journal using(acctid) where type=2 group by acctid having sum(amnt) < 500`

(c) `select acctid from account where type=2 and amnt < 500`

(d) `select acctid from journal group by acctid having sum(amnt) < 500 and type=2`

(e) Other:

7. (5 points) Find potentially fradulent accounts (different customer names using same ssn).

(a) `select * from customer a inner join customer b using (ssn) where a.custid != b.custid`

(b) `select * from customer a inner join customer b using (ssn) where a.custid > b.custid`

(c) `select * from customer a inner join customer b using (ssn) where a.custid > b.custid and a.username > b.username`

(d) `select * from customer a inner join customer b using (ssn) where a.custid > b.custid and concat(a.fname,a.lname) != concat(b.fname,b.lname)`

(e) Other:

8. (5 points) What is the most appropriate index for `customer.custid` field?

(a) Btree Index

(b) Bitmap Index

(c) Clustered Index

(d) Bitmap Clustered Index

(e) Other:

9. (5 points) What is the most appropriate index for `customer.username` field?

   (a) Btree Index

   (b) Bitmap Index

   (c) Clustered Index

   (d) Bitmap Clustered Index

   (e) Other:

10. (5 points) What is the most appropriate index for `account.type` field?

    (a) Btree Index

    (b) Bitmap Index

    (c) Clustered Index

    (d) Bitmap Clustered Index

    (e) Other:

11. (5 points) The below code:

```
create table dates as with recursive mydates (tdate) as (
    select cast('19000101' as date) tdate
    union all
    select cast(tdate + interval '1 day' as date) tdate
    from mydates where tdate <= '21000101'
) select * from mydates;
```

    (a) Is invalid.

    (b) Will return all dates between 19000101 and 21000101

    (c) Will create a table with all dates between 19000101 and 21000101

    (d) Will never return.

    (e) Other:

12. (5 points) Find daily closing balances for all accounts.

    (a)
    ```
    select tdate,acctid,sum(amnt) from journal inner join dates
    group by tdate,acctid
    ```

    (b)
    ```
    select tdate,acctid,sum(amnt) over (partition by acctid order by tdate)
    from journal inner join dates on tdate=cast(tim as date)
    ```

    (c)
    ```
    select tdate,acctid,sum(amnt) over (partition by acctid order by tdate) amnt
    from dates natural left outer join (
    select cast(tim as date) tdate,acctid,sum(amnt) amnt
    from journal group by cast(tim as date),acctid)
    ```

(d)
```
select b.tdate,a.acctid,
sum(case when a.tdate=b.tdate then amnt else 0 end amnt)
over (partition by acctid order by b.tdate) amnt
from (select cast(tim as date) tdate,acctid,sum(amnt) amnt
from journal group by cast(tim as date),acctid) a, dates b
```

(e) Other:

13. (5 points) Find all savings accounts (type 2), with average daily closing balance of less than $500 within 1-month calendar month.

(a)
```
select acctid from (previous question)
inner join account using (acctid) where type=2
group by to_char(tdate,'YYYYMM'),acctid having avg(amnt) < 500
```

(b)
```
select acctid,avg(amnt) over (partition by acctid order by tdate)
avgamnt from (previous question) inner join account using (acctid)
where type=2 and avgamnt < 500 group by to_char(tdate,'YYYYMM')
```

(c)
```
select acctid,avg(amnt)
over (partition by acctid,to_char(tdate,'YYYYMM')) avgamnt
from (previous question) inner join account using (acctid)
where type=2 and avgamnt < 500
```

(d)
```
select acctid from (previous question)
inner join account using (acctid) where type=2 and avg(amnt)
over (partition by acctid,to_char(tdate,'YYYYMM')) < 500
```

(e) Other:

14. (5 points) Find above average transactions:

(a)
```
select transactid,avg(amnt) from journal
group by transactid having amnt > avg(amnt)
```

(b)
```
select * from (select transactid,amnt,avg(amnt) over () avgamnt
from journal where amnt>0) where amnt > avgamnt
```

(c)
```
select * from (select transactid,amnt,avg(amnt)
over (partition by transactid) avgamnt from journal)
where amnt > avg(amnt) over (partition by transactid)
```

(d)
```
select * from (select transactid,amnt,avg(amnt)
over (partition by transactid) avgamnt from journal)
having amnt > avg(amnt) over (partition by transactid)
```

(e) Other:

15. (5 points) Find all checking accounts (type 1) with less than 1 transaction per month in 2011.

(a)
```
select acctid from journal inner join account using (acctid)
where type=1 and to_char(tim, 'YYYY') = '2011'
group by acctid having count(*) < 12
```

(b)
```
select acctid from journal inner join account using (acctid)
where type=1 and to_char(tim,'YYYY') = '2011'
group by acctid, to_char(tim, 'YYYYMM') having count(*) < 1
```

(c)
```
select acctid from journal inner join account using (acctid)
where type=1 and to_char(tim, 'YYYY') = '2011'
group by acctid, to_char(tim, 'YYYYMM') having sum(1.0) < 1.0
```

(d)
```
select acctid from journal inner join account using (acctid)
where type=1 and to_char(tim, 'YYYY') = '2011' and count(*) < 1.0
group by acctid, to_char(tim, 'YYYYMM')
```

(e) Other:

16. (5 points) Find balances by SSN.

(a)
```
select ssn,sum(amnt) from customer c, account a, journal j
where c.acctid=a.acctid and a.acctid=j.acctid group by ssn
```

(b)
```
select ssn,sum(amnt) from customer
inner join account using (custid) inner join journal using (acctid)
group by ssn
```

(c)
```
select a.ssn,b.amnt from customer a
inner join (select acctid,sum(amnt) amnt from journal
group by acctid) b using acctid
```

(d)
```
select acctid,sum(amnt) from customer c, account a, journal j
where c.custid=a.custid and a.acctid=j.acctid group by acctid
```

(e) Other:

17. (5 points) What percentage of the money is held in checking, savings accounts?

(a)
```
select type,amnt/sum(amnt) from account
inner join journal using (acctid) where type in (1,2) group by type
```

(b)
```
select type,sum(amnt)/(select sum(amnt) total from journal)
from account inner join journal using (acctid) where type in (1,2) group by
type
```

(c)
```
select type,sum(amnt) / sum( sum(amnt) ) over ()
from account inner join journal using (acctid) where type in (1,2) group by
type
```

(d)
```
select type,sum(amnt)/tot
from account inner join journal using (acctid) inner join
(select sum(amnt) tot from journal)
where type in (1,2) group by type,tot
```

(e) Other:

18. (5 points) For "`account inner join journal`", and no indexes, most modern databases will perform:

(a) merge join.

(b) hash join.

(c) indexed lookup join.

(d) inner loop join.

(e) Other:

19. (5 points) In general, on limited memory system, no indexes, and huge tables, what join type would perform best?

    (a) merge join.

    (b) hash join.

    (c) indexed lookup join.

    (d) inner loop join.

    (e) Other:

20. (5 points) Below query is identical to: `select a.*,b.val from T1 a left outer join T2 b on a.key=b.key and a.val!=b.val`

    (a) `select a.*,b.val from T1 a inner join T2 b on a.key=b.key and a.val!=b.val`

    (b)
    ```
    with TMP as (select a.*,b.val from T1 a inner join T2 b on a.key=b.key
    where a.val!=b.val)
    select a.*,b.val from T1 a left outer join TMP b on a.key=b.key
    ```

    (c)
    ```
    with TMP as (select a.*,b.val from T1 a left outer join T2 b on a.key=b.key
    where a.val!=b.val)
    select a.* from TMP where a.val!=b.val
    ```

    (d) All of the above queries are identical.

    (d) None of the queries are identical to the question.