CISC 7510X Final Exam

For the below questions, use the following schema definition.

```
account(accountid,username,firstname,lastname)
digitalasset(assetid,description,dollarprice)
transaction(tranid,accountid,timstamp)
transactiondetail(tranid,assetid,qty,price)
```

It is a schema for a digital-asset store, with accounts, digitalassets, and transactions that link accounts to digitalassets. Each transaction can have multiple items, which are in transactiondetail table. 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) Find account id of John Doe.
   (a) select lastname,firstname from account where firstname='John' and lastname='Doe'
   (b) select accountid from transaction where firstname='John' and lastname='Doe'
   (c) select accountid from account where firstname='John' and lastname='Doe'
   (d) select accountid from account inner join transactiondetail using(accountid) where firstname='John' and lastname='Doe'
   (e) Other:
2. (5 points) Find the average price of a digitalasset.
   (a) select avg(price) from transactiondetail
   (b) select avg(dollarprice) from digitalasset
   (c) select avg(qty*dollarprice) from digitalasset
   (d) select avg(price) from digitalasset
   (e) Other:
3. (5 points) Find number of transactions by account.
   (a) select accountid,count(*) from transaction natural inner join transactiondetail group by accountid
   (b) select assetid,count(*) from transactiondetail group by assetid
   (c) select tranid,count(*) from transactiondetail group by tranid
   (d) select accountid,count(*) from transaction group by accountid
   (e) Other:
4. (5 points) Find names (descriptions) of all digital assets ever purchased by 'John Doe'.
   (a) select count(*) from account a natural inner join transaction b natural inner join transactiondetail c where a.lastname='Doe' and a.firstname='John'
   (b) select description from account a natural inner join transaction b natural inner join transactiondetail c natural inner join digitalasset d where a.lastname='Doe' and a.firstname='John' group by description

(c) select description from account a natural inner join digitalasset d where a.lastname='Doe' and a.firstname='John' group by description

(d) select distinct description from account a natural inner join digitalasset d where a.lastname='Doe' and a.firstname='John'

(e) Other:

5. (5 points) Find all transactions that total more than $1000.

(a) select tranid from transaction a natural inner join transactiondetail b group by tranid having sum(qty*price) > 1000

(b) select tranid from transaction a natural inner join transactiondetail b where qty*price > 1000 group by tranid

(c) select tranid from account a inner join transaction a natural inner join transactiondetail b where qty*price > 1000 group by tranid

(d) select tranid from transactiondetail b where qty*price > 1000

(e) Other:

6. (5 points) Find accounts who have never purchased anything.

(a) select a.* from account a natural inner join transaction b where b.tranid is null

(b) select a.* from account a left join transactiondetail b on a.accountid=b.accountid where b.tranid=0

(c) select a.* from account a inner join transaction b on a.accountid=b.accountid where b.tranid > 0

(d) select a.* from account a natural left outer join transaction b where b.tranid is null

(e) Other:

7. (5 points) Find top 10 accounts who spent the most in 2023.

(a) select top 10 accountid from transaction a natural inner join transactiondetail b where timstamp >='20230101' and timstamp < '20240101'

(b) select accountid from transaction a natural inner join transactiondetail b where timstamp >='20230101' and timstamp < '20240101' order by sum(qty*price) desc

(c) select accountid,row_number() over (order by sum(qty*price) desc) rn from transaction a natural inner join transactiondetail b where timstamp >='20230101' and timstamp < '20240101' and rn <= 10

(d) select accountid,sum(qty*price) v from transaction a natural inner join transactiondetail b where timstamp >='20230101' and timstamp < '20240101' group by accountid order by 2 desc limit 10

(e) Other:

8. (5 points) What is the most appropriate index for account.username 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 digitalasset.description 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 digitalasset.assetid field?

    (a) Btree Index

    (b) Bitmap Index

    (c) Clustered Index

    (d) Bitmap Clustered Index

    (e) Other:

11. (5 points) The below code (tip: write out the first few output numbers):

```
with recursive n(n) as (
    select 2 n union all
    select n+1 from n where n<1000
)
select a.n
from n a left join n b on b.n < sqrt(a.n)
group by a.n
having a.n=2 or min(a.n % b.n) > 0
```

    (a) Is invalid

    (b) Will generate a list of numbers 1 to 1000

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

    (d) Will output list of all prime numbers between 1 and 1000

    (e) Other:

12. (5 points) Find average number of items per transaction.

    (a) select avg(transaction) from account a natural inner join transaction b

    (b) select avg(*) from account a natural inner join transaction b where accountid > 0

    (c) select avg(cnt) from (select tranid,count(*) cnt from transaction a natural inner join transactiondetail b group by tranid) a

    (d) select avg(sum(1.0)) over () from account a

    (e) Other:

13. (5 points) Find items that were bought on sale (dollar price is higher than transaction price).

(a) select * from digitalasset a natural inner join transactiondetail b where dollarprice > price

(b) select * from digitalasset a natural inner join transactiondetail b group by tranid having dollarprice > price

(c) select count(*) from digitalasset a natural inner join transactiondetail b group by tranid having dollarprice > price

(d) select * from transactiondetail b where dollarprice > price

(e) Other:

14. (5 points) Find the last sale price for each asset.

(a) select assetid,max(price) ls from transactiondetail order by timstamp

(b) select assetid,max(timstamp) over (partition by assetid order by price) ls from transactiondetail

(c) select assetid,last_value(price) over (partition by assetid order by timstamp) ls from transactiondetail

(d) select assetid,last_value(price) over (partition by assetid order by timstamp) ls from transaction p natural inner join transactiondetail pi

(e) Other:

15. (5 points) Find percentage of transactions with above average amount.

(a) select row_number() over () / count(*) from transaction a inner join transactiondetail b where qty*price > avg(qty*price)

(b) select tranid,sum(qty*price) px, avg(sum(qty*price)) over () avgpx transaction a inner join transactiondetail b where px > avgpx

(c) select percentage(qty*price) from transactiondetail where qty*price > avg(qty*price)

(d) select sum(case when qty*price>avg() then 1.0 else NULL end) / sum(1.0) from transaction inner join transactiondetail

(e) Other:

16. (5 points) Find all accounts who purchased 'SillyCoin' during the first month of 2023.

(a) select * from account where transaction = 'SillyCoin'

(b) select * from account inner join transaction inner join transactiondetail where item='SillyCoin'

(c) select * from transaction inner join transactiondetail where description='SillyCoin'

(d) select distinct from transactiondetail inner join account using(accountid) having description='SillyCoin'

(e) Other:

17. (5 points) Find accounts who purchased 'SillyCoin' and also 'FunnyCoin'.

(a) select * from account where purchased in ('SillyCoin', 'FunnyCoin')

(b) select * from account inner join transactiondetail on accountid and description in ('SillyCoin', 'FunnyCoin')

(c) select accountid from transaction where description in ('SillyCoin', 'FunnyCoin0')

(d) select accountid from transaction a inner join transactiondetail inner join transaction b inner join transactiondetail where a.description='SillyCoin' and b.description='FunnyCoin'

(e) Other:

18. (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:

19. (5 points) For "account inner join transaction", and no indexes, most modern databases will perform:

    (a) merge join.

    (b) hash join.

    (c) indexed lookup join.

    (d) inner loop join.

    (e) Other:

20. (5 points) Partitions:

    (a) Are similar to views.

    (b) Are similar to temporary tables.

    (c) Allow for physical clustering of logically similar data.

    (d) All of the above.

    (e) Other: