1. Consider the following relation for published books:

BOOK (Book\_title, Authorname, Book\_type, Listprice, Author\_affil, Publisher)

Author\_affil referes to the affiliation of the author. Suppose the following

dependencies exist:

Book\_title -> Publisher, Book\_type

Book\_type -> Listprice

Author\_name -> Author-affil

a) What normal form is the relation in? Explain your answer.

The key for this relation is Book\_title,Authorname. This relation is in 1NF and not in

2NF as no attributes are FFD on the key. It is also not in 3NF.

b) Apply normalization until you cannot decompose the relations further. State the

reasons behind each decomposition.

2NF decomposition:

Book0(Book\_title, Authorname)

Book1(Book\_title, Publisher, Book\_type, Listprice)

Book2(Authorname, Author\_affil)

This decomposition eliminates the partial dependencies.

3NF decomposition:

Book0(Book\_title, Authorname)

Book1-1(Book\_title, Publisher, Book\_type)

Book1-2(Book\_type, Listprice)

Book2(Authorname, Author\_affil)

2. Consider the following relation:

R (Doctor#, Patient#, Date, Diagnosis, Treat\_code, Charge)

In this relation, a tuple describes a visit of a patient to a doctor along with a treatment

code and daily charge. Assume that diagnosis is determined (uniquely) for each

patient by a doctor. Assume that each treatment code has a fixed charge (regardless

of patient). Is this relation in 2NF? Justify your answer and decompose if necessary.

Then argue whether further normalization to 3NF is necessary, and if so, perform it.

From the question’s text, we can infer the following functional dependencies:

{Doctor#, Patient#, Date}{Diagnosis, Treat\_code, Charge}

{Treat\_code}{Charge}

Because there are no partial dependencies, the given relation is in 2NF already. This

however is not 3NF because the Charge is a nonkey attribute that is determined by

another nonkey attribute, Treat\_code. We must decompose further:

R (Doctor#, Patient#, Date, Diagnosis, Treat\_code)

R1 (Treat\_code, Charge)

We could further infer that the treatment for a given diagnosis is functionally

dependant, but we should be sure to allow the doctor to have some flexibility when

prescribing cures.

3. List and briefly explain the ACID properties provided by a Database Management

System.

Atomicity. A transaction is an atomic unit of processing; it should either be

performed in its entirety or not performed at all.

Consistency preservation. A transaction should be consistency preserving, meaning

that if it is completely executed from beginning to end without interference from other

transactions, it should take the database from one consistent state to another.

Isolation. A transaction should appear as though it is being executed in isolation from

other transactions, even though many transactions are executing concurrently. That is,

the execution of a transaction should not be interfered with by any other transactions

executing concurrently.

Durability or permanency. The changes applied to the database by a committed

transaction must persist in the database. These changes must not be lost because of

any failure.

4. Briefly what is a query tree and what are its components?

An SQL query is first translated into an equivalent extended relational algebra

expression represented as a query tree data structure.

5. Briefly describe the basic Two-Phase Locking (2PL) protocol.

Two-phase locking may limit the amount of concurrency that can occur in a schedule

because a transaction T may not be able to release an item X after it is through using

it if T must lock an additional item Y later; or conversely, T must lock the additional

item Y before it needs it so that it can release X. Hence, X must remain locked by T

until all items that the transaction needs to read or write have been locked; only then

can X be released by T.Meanwhile, another transaction seeking to access X may be

forced to wait, even though T is done with X; conversely, if Y is locked earlier than it is

needed, another transaction seeking to access Y is forced to wait even though T is not

using Y yet. This is the price for guaranteeing serializability of all schedules without

having to check the schedules themselves.

6. Consider a disk with the following characteristics (these are not parameters of any

particular disk unit): block size B=512 bytes, interblock gap size G=128 bytes, number

of blocks per track B/T=20, number of tracks per surface=400. A disk pack consists of

15 double-sided disks.

a) What is the total capacity of a track and what is its useful capacity (excluding

interblock gaps)?

Total track size = B/T \* (B+G) = 20 \* (512 B + 128 B) = 12,800 B = 12.8 KB

Useful capacity of a track = B/T \* B = 20 \* 512 B = 10,240 B = 10.24 KB

b) How many cylinders are there?

Number of cylinders = number of tracks = T/S = 400

c) What is the total capacity and the useful capacity of a cylinder?

Total cylinder capacity = D \* Sides \* B/T \* (B + G) = 15 \* 2 \* 20 \* (512 B + 128

B) = 384,000 B = 384 KB

Useful cylinder capacity = \* Sides \* B/T \* B = 15 \* 2 \* 20 \* 512 B = 307,200 B =

307.2 KB

d) What is the total capacity and the useful capacity of a disk pack?

Total capacity of a disk pack = D \* Sides \* T/S \* B/T \* (B+G) = 15 \* 2 \* 400 \* 20

\* (512 B + 128 B) = 153,600,000 B = 153.6 MB

Useful capacity of a disk pack = D \* Sides \* T/S \* B/T \* B = 15 \* 2 \* 400 \* 20 \*

512 B = 122,880,000 B = 122.88 MB

e) Suppose the disk drive rotates the disk pack at a speed of 2400 rpm (revolutions

per minute); what is the transfer rate in bytes/msec and the block transfer time btt

in msec? What is the average rotational delay rd in msec? What is the bulk transfer

rate (see Appendix B)?

Transfer Rate (TR) = Total Track Size/Time for 1 Rev.(ms) =

12,800/(60\*1000/2400) = 512 Bytes/ms

Block Transfer Time (BTT) = B/TR = 512 Bytes/512 Bytes/ms = 1 ms

Avg. Rotational Delay (RD) = Time for 1 Rev./Sides = (60\*1000/24)/2 = 25/2 =

12.5 ms

Bulk transfer rate = TR \* (B/(B+G)) = 512\*(512/640) = 409.6 Bytes/ms

f) Suppose the average seek time is 30 msec. How much time does it take (on the

average) in msec to locate and transfer a single block given its block address?

average time to locate and transfer a block = (D\*Sides) + RD + BTT = 2\*15 +

12.5 ms + 1 ms = 43.5 ms

g) Calculate the average time it would take to transfer 20 random blocks and compare

it with the time it would take to transfer 20 consecutive blocks using double

buffering to save seek time and rotational delay.

time to transfer 20 random blocks = = B/T \* (D\*Sides) + RD + BTT = 20 \* 43.5

ms = 870 ms

time to transfer 20 consecutive blocks using double buffering = (D\*Sides) + RD +

(B/T \* BTT) = (15 \* 2) + 12.5 ms + (20 \* 1 ms) = 62.5 ms

870 ms / 62.5 ms = 13.92  13.92 times faster for consecutive blocks with double

buffering.

7. Consider a disk with block size B=512 bytes. A block pointer is P=6 bytes long, and a

record pointer is P R =7 bytes long. A file has r=30,000 EMPLOYEE records of fixedlength.

Each record has the following fields: NAME (30 bytes), SSN (9 bytes),

DEPARTMENTCODE (9 bytes), ADDRESS (40 bytes), PHONE (9 bytes), BIRTHDATE (8

bytes), SEX (1 byte), JOBCODE (4 bytes), SALARY (4 bytes, real number). An

additional byte is used as a deletion marker.

a) Calculate the record size R in bytes.

Record length R = (30 + 9 + 9 + 40 + 9 + 8 + 1 + 4 + 4) + 1 = 115 B

b) Calculate the blocking factor bfr and the number of file blocks b assuming an

unspanned organization.

Blocking factor bfr = floor(B/R) = floor(512 B/115 B) = 4 records per block

Number of blocks needed for file (b) = ceiling(r/BFR) = ceiling(30,000/4) = 7,500

c) Suppose the file is ordered by the key field SSN and we want to construct a primary

index on SSN. Calculate

i. the index blocking factor bfri (which is also the index fan-out fo);

Index record size Ri = (VSSN + P) = (9 + 6) = 15 B

Index blocking factor bfri = fo = floor(B/R i ) = floor(512 B/15 B) = 34

ii. the number of 1st level index entries and the number of first-level index blocks;

Number of 1st level index entries r1 = number of file blocks b = 7,500 entries

Number of 1st level index blocks b1 = ceiling(r1 /bfri ) = ceiling(7,500/34) = 221

blocks

iii. the number of levels needed if we make it into a multi-level index

Number of 2nd level index entries r2 = number of 1st level blocks b1 = 221

entries

Number of 2nd level index blocks b2 = ceiling(r2 /bfri ) = ceiling(221/34) = 7

blocks

Number of 3rd level index entries r3 = number of 2nd level index blocks b2 = 7

entries

Number of 3rd level index blocks b3 = ceiling(r3 /bfri ) = ceiling(7/34) = 1

Since the 3rd level has only one block, it is the top index level.

Hence, the index has x = 3 levels

iv. the total number of blocks required by the multi-level index

Total number of blocks for the index bi = b1 + b2 + b3 = 221 + 7 + 1 = 229

blocks

v. the number of block accesses needed to search for and retrieve a record from

the file--given its SSN value--using the primary index.

Number of block accesses to search for a record = x + 1 = 3 + 1 = 4

8. Suppose we have the following requirements for a university database that is used to

keep track of students’ transcripts: (Functional Dependency Diagram)

FD1: {SSSN}{SNAME, SNUM, SCADDR, SCPHONE, SPADDR, SPPHONE, BDATE, SEX,

CLASS, MAJOR, MINOR, PROG}

FD2: {SNUM}{SNAME, SSSN, SCADDR, SCPHONE, SPADDR, SPPHONE, BDATE, SEX,

CLASS, MAJOR, MINOR, PROG}

FD3: {DEPTNAME}{DEPTCODE, DEPTOFFICE, DEPTPHONE, DEPTCOLLEGE}

FD4: {DEPTCODE}{DEPTNAME, DEPTOFFICE, DEPTPHONE, DEPTCOLLEGE}

FD5: {CNUM}{CNAME, CDESC, CREDIT, LEVEL, CDEPT}

FD6: {SECCOURSE, SEMESTER, YEAR, SECNUM}{INSTRUCTORNAME}

FD7: {SECCOURSE, SEMESTER, YEAR, SECNUM, SSSN}{GRADE}

FD1 and FD2 refer to student attributes; we can define a relation STUDENT and

choose either SSSN or SNUM as its primary key. Similarly, FD3 and FD4 refer to

department attributes, with either DEPTNAME or DEPTCODE as primary key. FD5

defines COURSE attributes, and FD6 SECTION attributes. Finally, FD7 defines GRADES

attributes. We can create one relation for each of STUDENT, DEPARTMENT, COURSE,

SECTION, and GRADES as shown below, where the primary keys are underlined. The

COURSE, SECTION, and GRADES relations are in 3NF and BCNF if no other

dependencies exist. The STUDENT and DEPARTMENT relations are in 3NF and BCNF

according to the general definition given in Sections 18.4 and 18.5, but not according

to the definitions of Section 18.3 since both relations have secondary keys.

The foreign keys will be as follows:

STUDENT.MAJOR  DEPARTMENT.DEPTCODE

STUDENT.MINOR  DEPARTMENT.DEPTCODE

COURSE.CDEPT  DEPARTMENT.DEPTCODE

SECTION.SECCOURSE  COURSE.CNUM

GRADES.(SECCOURSE, SEMESTER, YEAR, SECNUM)  SECTION.(SECCOURSE,

SEMESTER, YEAR, SECNUM)

GRADES.SNUM -> STUDENT.SNUM

Note: We arbitrarily chose SNUM over SSSN for primary key of STUDENT, and

DEPTCODE over DEPTNAME for primary key of DEPARTMENT.

9. Change transaction T2 in Figure 21.2b to read:

read\_item(X);

X:= X+M;

if X > 90 then exit

else write\_item(X);

Discuss the final result of the different schedules in Figure 21.3 (a) and (b), where M

= 2 and N = 2, with respect to the following questions. Does adding the above

condition change the final outcome? Does the outcome obey the implied consistency

rule (that the capacity of X is 90)?

The above condition does not change the final outcome unless the initial value of X >

88. The outcome, however, does obey the implied consistency rule that X < 90, since

the value of X is not updated if it becomes greater than 90.

10. Suppose that the system crashes before the [read\_item,T3,A] entry is written to the

log in Figure 23.1 (b); will that make any difference in the recovery process?

There will be no difference in the recovery process, because read\_item operations are

needed only for determining if cascading rollback of additional transactions is

necessary.

11. Suppose that we use the deferred update protocol for the example in Figure 23.6.

Show how the log would be different in the case of deferred update by removing the

unnecessary log entries; then describe the recovery process, using your modified log.

Assume that only redo operations are applied, and specify which operations in the log

are redone and which are ignored.

In the case of deferred update, the write operations of uncommitted transactions are

not recorded in the database until the transactions commit. Hence, the write

operations of T2 and T3 would not have been applied to the database and so T4

would have read the previous (committed) values of items A and B, thus leading to a

recoverable schedule. By using the procedure RDU\_M (deferred update with

concurrent execution in a multiuser environment), the following result is obtained:

The list of committed transactions T since the last checkpoint contains only transaction

T4. The list of active transactions T' contains transactions T2 and T3. Only the WRITE

operations of the committed transactions are to be redone. Hence, REDO is applied to:

[write\_item,T4,B,15] [write\_item,T4,A,20]

The transactions that are active and did not commit i.e., transactions T2 and T3 are

canceled and must be resubmitted. Their operations do not have to be undone since

they were never applied to the database.

12. Describe the meaning of a Table Scan. The condition where the execution of a

query results in the runtime processor scanning every record in the table.

13. Why are the table scans considered to be a problem? Due to performance issue

when the table is large.

14. How are the table scans avoided? Create index.

15. Describe the relationship

a) Schema and Relation - A schema has multiple relations; relations are unique in the

schema

b) Relation and Attribute - A relation has multiple attributes; an attribute is unique in

the releation

c) Relation and Tuples - A relation has multiple tuples; relations are tables; and tuples

are rows

16. Translate the following set of relational operations into a single SQL query:

SELECT Pno

FROM WORKS\_ON w,

EMPLOYEE e,

DEPARTMENT d,

PROJECT p

WHERE w.Essn = e.Ssn

AND e.Lname = 'Smith'

AND e.Dnum = d.Dnumber

AND d.Mgr\_ssn = e.Ssn

AND d.Dnumber = p.Dnum;

17. Which of following languages is used to define the application’s queries?

a) DDL b) DQL c) DML d) Metadata

18. What does DDL define? Data definition language statements in the language of the

chosen DBMS that specify the conceptual and external level schemas of the database

system

19. What mechanism is used to authorize user? User account’s role

20. How do we eliminate duplicates from a select’s result set? Distinct Clause

21. Which of the following describe the act of making a transaction’s changes to the

database state permanent? COMMIT

22. What best describes the referential integrity? Foreign to Primary Key

23. Which of the clauses is only useful when used in SELECT statements containing

aggregation function? GROUP BY

24. Which of the following is not true of views?

a. Serves to decouple clients from the internal schema

b. Can be used to cache the results from complex aggregated queries

c. Implements CRUD operations across joined tables

d. Can be used to provide clients access to otherwise protected tables

25. Describe a records blocking factor. Represents the average number of records per

block for the file ceil B/R

26. What are the two strategies employed by the query processor when performing

searches on two attributes from a single table.

a. Consider the case where one attribute is indexed and the other is not

Search for the records with index attribute value, and linearly search the identified

records for non-indexed value

b. Consider the case when both attributes are separately indexed

Use the example of searching for a student with last-name and phone number

Perform two searches and return the intersection

27. Name and describe the four levels of Transaction Isolation provided by SQL clients

a) Read Uncommitted - allows all violation types.

b) Read Committed - allows both nonrepeatable read and phantom violations.

c) Repeatable Reads - allows phantom violations.

d) Serializable - eliminates the possibility of all three types of violations.

28. How does 2 Phase Locking guarantee the schedules when all transactions are

following 2 phase locking? Guarantees only serializable schedules

29. What activities identify the layout of the table files and index files on the target file

system? Physical Design Activities

30. What describes the general solution to normalizing relations that are not in second

and third normal form? Normalization through decomposition

31. Is this true that functional dependencies are symmetric? False

32. A relation in First Normal Form must be in None form.

33. Fundamentally what do indexing and other optimization techniques minimize? Disk

I/O

34. What is the disadvantage of Unspanned Records? Unused space

35. What is one of the characteristics of Unordered Data Files? Binary search is not

possible

36. What is the main purpose of using an index? Reduce Disk I/O

37. Under what circumstances a secondary index is required? Indexing non-ordered

attributes

38. Is the following schedule serializable? R1(Y); R1(X); W1(X);R2(X);W1(Y);C1;W2(X);

Yes

39. 39. Which of the locks allow shared access to an object?

a) Write Locks

b) Read Locks

c) Read-Write Locks

d) Binary Locks

40. EER Diagram

41. ER Diagram

42. Which of the following EER

diagram(s) is/are incorrect and why?

Only (c) is incorrect.

43. Figure 9.8 shows an ER schema for

a database that may be used to keep

track of transport ships and their

locations for maritime authorities. Map

this schema into a relational schema, and specify all primary keys and foreign keys.

Other Notes

1NF: Remove composite attributes and multivalued attributes.

2NF: 1NF and every non-prime attribute A in R is fully functionally dependent on the

primary key.

3NF: 2NF and no non-prime attribute A in R is transitively dependent on the primary key.

Recoverable schedules - Recovery is possible

Cascading rollback may occur in some recoverable schedules - Uncom. trans. may need

to be rolled back

Cascadeless schedule - Avoids cascading rollback

Strict schedule - Transactions can neither read nor write an item X until the last

transaction that wrote X has committed or aborted - Simpler recovery process (Restore

the before image)

Serializable schedules - Always considered to be correct when concurrent transactions are

executing - Places simultaneous transactions in series (Transaction T1 before T2, or vice

versa)