Sami Shamoon
College of Engineering (R.A)
Software Engineering Department

המכללה האקדמית להנדסה
סמי שמעון (ע"ר)
המחלקה להנדסת תוכנה

המחלקה להנדסת תוכנה

28/1/07
16.00-19.00

בסיסי נתונים

מועד א

ד"ר יוסף שפונגין
תשס"ז – סמסטר א'

חומר עזר – אסור .

בהצלחה ! **השאלון מכיל 8 דפים (כולל דף זה).**

==========

קמפוס באר-שבע
ביאליק פינת בזל באר-שבע 84100
www@sce.ac.il
טל: 08-6475621/622/618
פקס: 08-6475623

המכללה האקדמית להנדסה סמי שמעון (ע"ר)
המחלקה להנדסת תוכנה

בסיסי נתונים

.1

(9) (א) נתונים ה-Interfaces הבאים.

```
1) interface Int_A (key (x)){
2) attribute string x;
3) attribute Set<Struct S {string y, string z}>st;
4) relationship Set<Int_B> R_1_A_B
        inverse Int_B :: R_1_B_A;
5) relationship Int_B R_2_A_B
        inverse Int_B :: R_2_B_A;
6) relationship Set<Int_B> R_3_A_B
        inverse Int_B :: R_3_B_A;
   };
```

```
1) interface Int_B (key (u)) {
2) attribute string u;
3) attribute Bag<Struct S {string v, string w}>st;
4) relationship <Int_A> R_1_B_A
        inverse Int_A :: R_1_A_B;
5) relationship Int_A R_2_B_A
        inverse Int_A :: R_2_A_B;
6) relationship Int_A R_3_B_A
        inverse Int_B :: R_3_A_B;
   };
```

(1) כתוב את כל ה-RMs (Relational Models) המתאימים
ל-Interfaces הנ"ל. ציין את ה-keys של הטבלאות.
(2) בנה את ה- E/R דיאגראמה שלפי המשמעות (objects, relationships) תהיה
מתאימה ל- interfaces הנ"ל.
(3) כתוב את ה-RM המתאים ל-E/R שבנית ב-(2).

(8) (ב) ב- R(A, B, C, D, E, F, G, H) relation מתקיימים ה- FD's הבאים:

$$A \to B, B \to C, C \to A, D \to E, E \to F, F \to E, G \to H$$

(1) האם R נמצא ב- BCNF? אם לא, פרק אותו לפי BCNF.
(2) האם R נמצא ב- 3NF? אם לא, פרק אותו לפי 3NF.
(3) האם על סמך התוצאות ב- א' ו-ב' ניתן לעשות מסקנה לגבי
4NF של הטבלה? הסבר.

(8) (ג) ב-relation R(A, B, C, D, E) נתונות 3 עמודות- A, B, C:

A	B	C	D	E
1	2	3	*	*
1	2	3	*	*
1	3	2	*	*
1	2	2	*	*

(1) האם ניתן להשלים את הטבלה כך שיתקיים $AB \rightarrow\rightarrow D$ אך לא יתקיים $AB \rightarrow D$? הסבר.

(2) האם ניתן להשלים את הטבלה כך שהיא תהיה ב- 4NF? אם כן, השלם. הסבר.

(3) האם ניתן להשלים את הטבלה כך שהיא לא תהיה ב- 4NF? אם כן, השלם ופרק אותה לפי 4NF. הסבר.

.2

נתונה הסכמה הבאה (Relational Database Schema):

StLibrary(<u>bID</u>, nBooks)
StudentCourses(<u>stID</u>, <u>co</u>, grade)
BooksCourses(<u>bID</u>, <u>conum</u>)
StudentBooks(<u>studID</u>, <u>bID</u>, <u>date</u>)

נסביר את משמעות ה-attributes:

bID- ID של book;
nBooks- מספר ספרים (לפי bID) בספרייה;
stID, studID- ID של student;
conum, co - מספר קורס;
grade- ציון;
date- תאריך שבו סטודנט לקח את הספר המתאים.

(13) (א) מצא ב-RA את כל הסטודנטים "החרוצים ביותר", שכל אחד מהם לקח לפחות פעמיים, את כל הספרים המתאימים, עבור כל קורס שהוא למד.

(12) (ב) מצא ב- DATALOG את כל הקורסים שעבורם בספריה יש בדיוק ספר אחד בכמות (nBooks) המקסימאלית מבין כל הספרים בספריה.

(10) (ג) כתוב ב-RA את ה-Constraint שמקיים את התנאים הבאים:
(1) עבור כל קורס בספריה יש לפחות שלושה ספרים שונים;
(2) עבור כל קורס קיימים לפחות שני ספרים בכמות לא פחות מ-20 (כל אחד מהספרים)

.3

נניח כי בסיס נתונים מוגדר על-ידי 4 טבלאות משאלה 2.

(10) (א) נגדיר סטודנט מצטיין כסטודנט שהממוצע שלו לפחות 85. מצא בשאילתה אחת, את כל הסטודנטים המצטיינים שכל אחד מהם מקיים את התנאים הבאים:
(1) לסטודנט אין אף ציון של פחות מ- 80;
(2) הסטודנט פנה לספריה מספר רב ביותר של פעמים מבין כל הסטודנטים.

(10) (ב) נניח כי עשו ראורגניזציה של הספרייה לפי הכלל הבא: עבור כל ספר שנמצא בכמות מינימאלית מבין כל הספרים, הוסיפו מספר ספרים השווה ל-10% מסך כל הסטודנטים הלומדים קורסים לפי ספר זה. כתוב את השאילתא המתאימה.

4.

(10) נתון XML-file הבא:

```
<doc> This is Doc
    <sub par="HELLO">
        <subsub1> What is this?
            <subsubsub1> And this? </subsubsub1>
        </subsub1>
        <subsub1> How are You? </subsub1>
        <subsub2> Where?</subsub2>
    </sub>
    <sub>
        <subsub1> What is this? </subsub1>
    </sub>
</doc>
```

(א) כתוב את ה-DTD המתאים ל- XML-file הנ"ל.

(ב) מהו הפלט של ה-XPathes הבאים?

(1) /doc/sub/following::*

(2) //subsub1/ancestor::*

5.

(5) (א) נניח כי בנוי Multiple Index בצורה של
"Sparse Index on Sparse Index on Sparse Index"

נתון:

ב-File של נתונים יש 1000000000 רשומות;

אורך של כל רשומה – 400 bytes;

גודל של Block – 4000 bytes;

אורך של רשומה ב- Index – 40 bytes.

כמה פעולות I/O נצטרך כדי למצוא רשומה מסוימת? הסבר.

(5) (ב) נניח כי ב-B-Tree with Duplicates ה-root מיוצג באופן הבא:

—	10	—

תן דוגמא ל-B-tree עם 3 רמות המתאים ל-root זה.

בהצלחה!

1. ODL

(a) Example

```
Interface Star (key name) {
attribute string name;
attribute Struct Addr {string street, string city} address;
relationship Set<Movie> starredIn
        inverse Movie::stars; };
```

(b) Types in ODL Atomic: integer, float, character, string, boolean, enumeration.
Complex (T denotes the type) : Set<T>, Bag<T>, List<T>,
Array<T,i>, Struct N {T_1, F_1, ...,T_n, F_n}.

2. FD Rules about FD's.

(a) The Splitting/Combining Rule;
(b) An FD $A_1A_2...A_n \rightarrow B_1B_2...B_m$ is Trivial if the B's are a subset of the A's;
(c) The above FD is Nontrivial if at least one of the B's is not among the A's;
(d) The above FD is Completely nontrivial if none of the B's is also one of the A's;
(e) The above FD is equivalent to $A_1A_2...A_n \rightarrow C_1C_2...C_k$, where the C's are all those B's that are not also A's;
(f) The Transitive Rule
(g) If $\{B_1B_2,...,B_m\} \subseteq \{A_1A_2,...,A_n\}$, then $A_1A_2...A_n \rightarrow B_1B_2...B_m$;
(h) If $A_1A_2...A_n \rightarrow B_1B_2...B_m$, then $A_1A_2...A_nC_1...C_k \rightarrow B_1B_2...B_mC_1...C_k$ for any set of C's

3. MD Rules about MD's.

(a) The Trivial Rule: If $A_1A_2...A_n \rightarrow\rightarrow B_1B_2...B_m$ then $A_1A_2...A_n \rightarrow C_1C_2...C_k$, where the C's are the B's plus one or more of the A's;
(b) Combining Rule;
(c) If $A_1A_2...A_n \rightarrow\rightarrow B_1B_2...B_m$ is a MD for relation R, then also $A_1A_2...A_n \rightarrow C_1C_2...C_k$, where the C's are all attributes of R not among the A's and B's;
(d) If X, Y and Z are sets of attributes, $X \rightarrow\rightarrow Y$ and $X \rightarrow\rightarrow Z$, then $X \rightarrow\rightarrow Y \cap Z$;
(c) If X, Y and Z are sets of attributes, $X \rightarrow\rightarrow Y$ and $X \rightarrow\rightarrow Z$, then $X \rightarrow\rightarrow (Y - Z)$;
(f) If $Y \subseteq X$, then $X \rightarrow\rightarrow Y$;
(g) If $X \cup Y = R$, then $X \rightarrow\rightarrow Y$;
(h) If $X \rightarrow\rightarrow Y$, then $X \rightarrow\rightarrow Y - X$;
(i) The Transitive Rule.

4. RA (a) Set Operations: $R \cup S, R \cap S, R - S$.
(b) Projection: $\pi_{A_1,A_2,...,A_n}(R)$
(c) Selection: $\sigma_C(R)$, where C - condition
(d) Cartesian Product: $R \times S$
(e) Natural Join: $R \bowtie S$
(f) Theta-Join: $R \bowtie_C S$, where C – condition
(g) Renaming: $\rho_{S(A_1,...,A_n)}(R)$

5. Datalog

Example: LongMovie(t,y) ← Movie(t,y,l,c,s,p) AND l ≥ 100

6. SQL

(a) SELECT Example:

```
SELECT Star1.name, Star2.name
FROM MovieStar AS Star1, MovieStar AS Star2
WHERE Star1.address = Star2.address
      AND Star1.name < Star2.name;
```

(b) UNION, EXCEPT, INTERSECT

Example:

```
(SELECT name, address FROM MovieStar)
   INTERSECT
(SELECT name, address FROM MovieExec);
```

(c) Subqueries

Example:

```
SELECT name FROM MovieExec
WHERE  cert =
        (SELECT producer FROM Movie
         WHERE title = 'Star Wars' );
```

(d) Conditions Involving Relations

(1) EXISTS R is a condition that is true iff R is not empty;
(2) s IN R is true iff s is equal to one of the values in R;
(3) s > ALL R is true iff s is greater than every value in R;
(4) s > ANY R is true iff s is greater than at least one value in R;

(5) Example:

```
SELECT name FROM MovieExec
WHERE cert IN
   (SELECT producer FROM Movie
    WHERE (title, year) IN
       (SELECT movieTitle, movieYear FROM StarsIn
        WHERE starName = 'Harrison Ford')
    );
```

(e) Correlated Subqueries

Example:

```
SELECT title FROM Movie AS Old
WHERE year < ANY
    (SELECT year FROM Movie
     WHERE title = Old.title);
```

(f) Aggregation functions: SUM, AVG, MIN, MAX, COUNT.

Example:

```
SELECT COUNT(DISTINCT name)
FROM MovieExec;
```

(g) GROUP + HAVING

Example:

```
SELECT name, SUM(length)
FROM MovieExec, Movie
WHERE producer = cert
GROUP BY name
HAVING MIN(year) < 1930;
```

(h) INSERT

Example:

```
INSERT INTO Studio(name)
   SELECT DISTINCT studioName FROM Movie
   WHERE studioName NOT IN
        (SELECT name FROM Studio);
```

(i) DELETE

Example:

```
DELETE FROM StarsIn
WHERE movieTitle = 'The Maltese Falcon' AND
movieYear = 1942;
```

(j) UPDATE

Example:

```
UPDATE MovieExec
SET name = 'Pres. ' || name
WHERE cert IN (SELECT pres FROM Studio);
```

7. XML

Example:

```
<doc attr="ABC">
    <subdoc> TEXT</subdoc>
</doc>
```

8. DTD

(a) Example:

```
<!DOCTYPE NEWSPAPER [
<!ELEMENT NEWSPAPER (ARTICLE+)>
<!ELEMENT ARTICLE (HEADLINE, BODY)>
<!ELEMENT HEADLINE(#PCDATA)>
<!ELEMENT BODY (#PCDATA)>
<!ATTLIST ARTICLE AUTHOR CDATA #REQUIRED>
<!ATTLIST ARTICLE EDITOR CDATA #IMPLIED>
]>
```

(b) Number of occurrences:

```
<!ELEMENT element_name (child_name)>    - exactly one occurrence;
<!ELEMENT element_name (child_name+)>   - minimum one occurence;
<!ELEMENT element_name (child_name*)>   - zero or more occurrences;
<!ELEMENT element_name (child_name?)>   - zero or one occurrence.
```

(c) Declaring either/or content

Example: `<!ELEMENT note(header, (message | body))>`

(d) Attribute Types: PCDATA, CDATA, (en1|en2|...), ID, IDREF, IDREFS.

(e) Default attribute value can have:

value	The default value of the attribute
#REQUIRED	The attribute value must be included in the element
#IMPLIED	The attribute does not have to be included
#FIXED value	The attribute value is fixed

Example. DTD:

```
<!ELEMENT element_name EMPTY>
<!ATTLIST element_name attr CDATA "0">
```

Valid XML: `<element_name attr="100"/>`

9. Xpath

(a) Path Expressions

node_name	Selects all child nodes of the node
/	Selects from the root node
//	Selects nodes from the current node, no matter where they are
@	Selects attributes

(b) Predicates Examples:

```
/bookstore/book[1]
/bookstore/book[last()-1]
//title[@lang]
/bookstore/book[price>35.00]/title
```

(c) Unknown nodes

*	Matches any element node
@*	Matches any attribute
node()	Matches any node of any kind

Examples: /bookstore/*
//title[@*]

(d) Several paths Example: //book/title | //book/price

(e) XPath Axes

ancestor	Selects all ancestors (parent, grandparent, etc.) of the current node
attribute	Selects all attributes of the current node
child	Selects all children of the current node
descendant	Selects all descendants (children, grandchildren, etc.) of the current node
following	Selects everything in the document after the closing tag of the current node
preceding	Selects everything that is before the start tag of the current node
self	Selects the current node

Examples: //chapter[2]/preceding::*
/library/book[last()]/following::*

10. XQuery FLWOR Expression

Example:

```
for $x in doc("books.xml")/bookstore/book
where $x/price>30
order by $x/title
return $x/title
```

Example:

```
for $x in doc("books.xml")/bookstore/book
return if ($x/@category="Children")
then <child>{data($x/title)}</child>
else <adult>{data($x/title)}</adult>
```

11. B-Trees

The rules:
(a) At the root, there are at least two used pointers;
(b) At a leaf, the last pointer points to the next block. Among the other pointers, at least $\left\lfloor \frac{n+1}{2} \right\rfloor$ pointers are point to data;
(c) At an interior node, at least $\left\lceil \frac{n+1}{2} \right\rceil$ pointers are actually used.

12. Linear Hash table

The rules:
(a) The number of buckets n is always chosen so the average number of records per bucket is a fixed fraction;
(b) Overflow blocks are permitted;
(c) The number of bits used to number the entries of the bucket array is $\lceil \log_2 n \rceil$, where n is the current number of buckets.