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

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

המחלקה להנדסת תוכנה

11/2/07
08:30-11:30

בסיסי נתונים
מועד ב
ד"ר יוסף שפונגין
תשס"ז – סמסטר א'

חומר עזר – אין (פרט לנוסחאות המרצה המצורפות למבחן)

הוראות מיוחדות – אין.

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

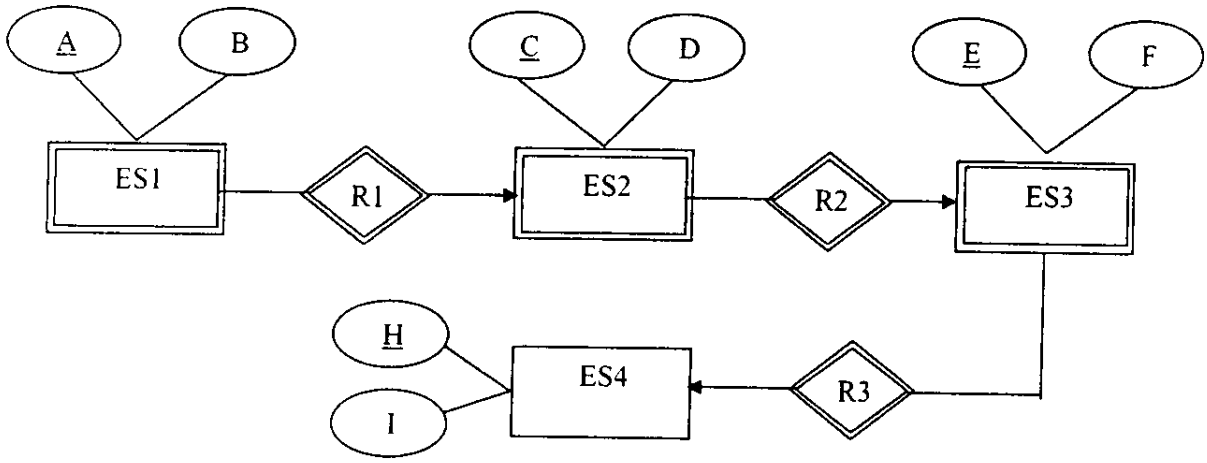
==========

קמפוס באר-שבע
טל: 08-6475621/622/618
www@sce.ac.il

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

בסיסי נתונים

.1 (9) (א) נתונה ה- E/R schema הבאה:



(1) כמה RM (Relational Models) שונות אפשר לבנות? הסבר.
(2) תן דוגמא לשתי RM שונות. ציין את ה-keys.

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

(8) (ג) (1) נתון כי $A \rightarrow\rightarrow BC$ (MD) האם נכון כי $AB \rightarrow\rightarrow C$? אם כן, הוכח זאת לפי ההגדרה של MD. אם לא, תן דוגמה נגדית.

(2) נתון כי $AB \rightarrow\rightarrow C$. האם נכון כי $A \rightarrow\rightarrow BC$? אם כן, הוכח זאת לפי ההגדרה של MD. אם לא, תן דוגמה נגדית.

.2

נתונה Relational Database Schema הבאה:

StLibrary(bID, nBooks)
StudentCourses(stID, co, lect, grade)
BooksCourses(bID, conum)
StudentBooks(studID, bID, date)
CoursesLecturers(course, lect, bID)

נסביר את משמעות ה-attributes:

bID- ID של book; בטבלה CourseLecturers, bID זה ספר מומלץ
על ידי מרצה לקורס (יכולים להיות כמה ספרים מומלצים).
nBooks- מספר ספרים (לפי bID) בספרייה;
stID-ID, studID של student;
conum, co, course - מספר קורס;
grade- ציון;
date- תאריך שבו סטודנט לקח את הספר המתאים.
lect - מרצה.

(13) (א) מצא ב-RA את כל המרצים כך, שעבור כל קורס שהמרצה מלמד לא קיים סטודנט (בין הנרשמים לקורס זה) שלקח פחות משני ספרים מתוך ספרים מומלצים לקורס זה.

(12) (ב) מצא ב- DATALOG את כל הסטודנטים שלקחו את כל הקורסים אצל אותם מרצים שבאף קורס שהם מעבירים אין נכשלים.

(10) (ג) כתוב ב-RA את ה-Constraint שמקיים את התנאים הבאים:
(1) עבור כל קורס בספריה יש לפחות 2 ספרים בכמות המקסימאלית (nBooks):
(2) כל מרצה, בכל קורס שהוא מלמד, חייב להמליץ לפחות על 2 ספרים.

.3

נניח כי בסיס נתונים מוגדר על-ידי 5 טבלאות משאלה 2.

(10) (א) נגדיר קורס כ"מצטיין" אם הממוצע של כל הציונים בקורס זה הוא לפחות 85 ואין נכשלים. מצא בשאילתה אחת, את כל המרצים שמעבירים מספר מקסימאלי של קורסים מצטיינים בין כל המרצים.

(10) (ב) נניח כי עשו ראורגניזציה של הספרייה לפי הכלל הבא: עבור כל ספר שמשתמשים בו במספר מינימאלי של קורסים, משאירים אותו בכמות השווה ל-10% מכלל הסטודנטים שלקחו ספר זה לפחות פעמיים. כתוב את השאילתא המתאימה.

.4

(5) (א) נתון את ה- DTD הבא:

```
<!DOCTYPE Company [
<!ELEMENT Company (Department*|Person+)>
<!ELEMENT Department (Person+)>
<!ELEMENT Person EMPTY>
<!ATTLIST Department Name CDATA #REQUIRED>
<!ATTLIST Person Last_Name CDATA #REQUIRED>
<!ATTLIST Person First_Name CDATA #REQUIRED>
<!ATTLIST Person City CDATA #REQUIRED>
<!ATTLIST Person Tel_Number CDATA #IMPLIED>
]>
```

תן דוגמא ל- XML-file המתאים ל-DTD זה.

(5) (ב) 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>
```

מהו הפלט של ה-XPathes הבאים?

/doc//subsub1/preceding::* (1)

//subsubsub1/following::* (2)

.5

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

10	100	—

(1) תן דוגמא ל-B-tree עם 3 רמות המתאים ל-root זה.
(2) הסבר את החיפוש של כל הרשומות עם key השווה ל-10.
(3) הסבר את החיפוש של כל הרשומות עם key השווה ל-50.
(4) הסבר את החיפוש של כל הרשומות עם key השווה ל-120.

(5) (ב) נניח עכשיו כי ב-root במקום "-" נמצא מספר 200.

(1) תן דוגמא ל-B-tree without Duplicates עם 3 רמות המתאים ל- root זה.
(2) מהו גודל המקסימאלי של file שבו יכול לטפל B-tree עם 3 רמות ועם ה-root הנתון בסעיף 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 $X \rightarrow\rightarrow Y \cap Z$;
(e) 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.