[Question 1 [Computing queries – 50 points]. Consider the following relations

\[ R(A,B,C) = \{<1,2,3>,<1,0,3>\} \]
\[ S(B,C,D) = \{<2,3,7>,<1,1,4>,<2,2,4>,<2,3,5>\} \]

(1.a) Cartesian product with selection
\{<1,2,3,2,3,7>,<1,2,3,2,2,4>,<1,2,3,2,3,5>\}
Group by S.C … two groups
\{\{<1,2,3,2,3,7>,<1,2,3,2,3,5>\} \{<1,2,3,2,2,4>\}\}
Select groups having COUNT > 1
\{<1,2,3,2,3,7>,<1,2,3,2,3,5>\}
Answer
\{<3,6>\}

(1.b) Answer
\{<2>\}

(1.c) Cartesian product with selection
\{<1,2,3,2,3,7>,<1,2,3,2,2,4>,<1,2,3,2,3,5>\}
Group by S.C … two groups
\{\{<1,2,3,2,3,7>,<1,2,3,2,3,5>\} \{<1,2,3,2,2,4>\}\}
Select groups having COUNT > 1
\{<1,2,3,2,3,7>,<1,2,3,2,3,5>\}
Answer
\{<3,6>\}
Question 2

Answer:

```sql
CREATE TABLE Course
(
    cid CHAR(6) PRIMARY KEY,
    ctitle CHAR(16),
    dept CHAR(3)
)

CREATE TABLE Student
(
    sid NUMERIC(8) PRIMARY KEY,
    sname VARCHAR(16),
    dg VARCHAR(4),
    city VARCHAR(10),
    CHECK ((dg = 'BSC') OR (dg = 'BENG') OR (dg = 'BA'))
)

CREATE TABLE S_Take_C
(
    sid NUMERIC(8),
    cid CHAR(6),
    yr NUMERIC(4),
    trm CHAR(1),
    mark INTEGER,
    CHECK ((trm = 'W') OR (trm = 'S') OR (trm = 'F')),
    CHECK ((mark ≥ 0) AND (mark ≤ 100)),
    UNIQUE(sid,cid),
    FOREIGN KEY(sid)
        REFERENCES Student(sid)
        ON DELETE CASCADE,
    FOREIGN KEY(cid)
        REFERENCES Course(cid)
        ON DELETE CASCADE
        ON UPDATE CASCADE
)
Question 3 [Relational Algebra – 50 points].

(3.a)

\[ \pi_{\text{sid}, \text{sname}}(\sigma_{\text{cid}=\text{csc343}}((\text{Student} \bowtie \text{S\_Take\_C}))) \]

or

\[ \pi_{\text{sid}, \text{sname}}((\sigma_{\text{cid}=\text{csc343}}(\text{Student}) \bowtie \text{S\_Take\_C})) \]

(3.b)

\[ R1 = \pi_{\text{yr, trm, mark}}(\sigma_{\text{cid}=\text{csc343}}(\text{S\_Take\_C})) \]

\[ R2 = \pi_{\text{yr, trm, mark}}(\sigma_{\text{mark1}>\text{mark}}(R1 \bowtie \rho_{\text{mark}=>\text{mark1}}(\text{R1}))) \]

Answer = \( R1 - R2 \)

(3.c) [20 points] “List student ids for students who have taken every computer science course”

Answer:

\[ R1 = \pi_{\text{cid}}(\sigma_{\text{dept}=\text{CSC}}(\text{Course})) \]

\[ \text{Answer} = (\pi_{\text{sid, cid}}(\text{S\_Take\_C}) / R1) \]
Question 4 [SQL – 50 points].

(4.a)

```
SELECT cid, cname
FROM Course
WHERE cid not in (SELECT cid FROM S_Take_C)
```

(4.b)

Answer:

```
SELECT COUNT (DISTINCT T.sid)
FROM Course C, S_Take_C T
WHERE C.cid = T.cid AND C.dept = 'CSC'
```

(4.c)

Answer:

```
CREATE VIEW SA (sid, savg) AS
SELECT T.sid, AVG (T.mark)
FROM S_Take_C T
GROUP BY T.sid

SELECT S.sid, S.sname
FROM Student S, SA
WHERE S.sid = SA.sid AND savg ≥ ALL (SELECT savg FROM SA)
```
Question 5 [True/False questions – 70 points].

(5.a) [F] The equality \((S \bowtie R) \bowtie Q = (S \bowtie (R \bowtie Q))\) holds if and only if \(S\), \(R\), \(S\) and \(Q\), \(R\) and \(Q\) share respectively at least one attribute;

(5.b) [F] Pointers are better than value-based references in a database because the latter are hardware-dependent;

(5.c) [T] Every relation in the Relational Model has at least one superkey consisting of all its attributes;

(5.d) [F] A trigger is an event that triggers an SQL statement to execute whenever the event occurs;

(5.e) [F] In embedded SQL, cursors can only be declared for database tables, and not for results of queries;

(5.f) [T] A DBMS supports mechanisms that allow multiple transactions to execute concurrently against a single database without interfering with each other;

(5.g) [T] If query \(Q\) evaluates to the empty table, then the condition \(0 > \text{ANY} (Q)\) is false;

(5.h) [F] If \(S\) is empty, then \(S \bowtie R = R\);

(5.i) [F] In embedded SQL, SQLSTATUS is a special programming language variable whose value describes the execution status of the application;

(5.j) [F] In SQL, all views can be updated just like database relations;

(5.k) [T] If relation \(R\) consists of a single attribute, then \(R \bowtie R = R\);

(5.l) [F] In SQL-DDL, one can define new aggregate functions that can then be used in SQL queries;

(5.m) [F] JDBC is a statement-level interface for executing SQL within a Java program;

(5.n) [F] Object-oriented databases make it possible for Java programs to access relational databases.