Consider a database with 3 tables, STUDENT-INFO, COURSE-INFO, and ENROLMENT. Assume
• the STUDENT-INFO table has 30,000 \((3 \times 10^4)\) rows,
• the COURSE-INFO table has 1,000 \((10^3)\) rows, [BiYing checked CORS & said 1365 for Spr 2009. Thx]
• the ENROLMENT table has 100,000 \((10^5)\) rows.

<table>
<thead>
<tr>
<th>STUDENT-INFO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student-ID</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COURSE-INFO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course-ID</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ENROLMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student-ID</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

T6-D2: (Efficient Query Processing) [Note: First read notes and also do T6-D1.]
(a) Give a "concise English description" of the output of the following primitive DB operations:

\[
R1 \leftarrow \text{e-select from ENROLMENT where Course-ID='UIT2201'}
\]
\[
R2 \leftarrow \text{e-join R1 and STUDENT-INFO}
\]
\[
\text{LIST} \leftarrow \text{e-project Course-ID, Name, Instructor from R2}
\]

(b) Give an "SQL query" statement to obtain each of the following:

- List the Student-ID of the students enrolled in the course with Course-ID 'UIT2201';
- List the Name, Tel-No of the students enrolled in the course with Course-ID 'UIT2201';
- List the Name, Tel-No of students taught by the professor named 'Leong Hon Wai'.

(c) Now, give a sequence of basic primitive DB operations (e-project, e-select, and e-join) to produce the results in (b) above. Make it as efficiently as possible.

T6-Q2: (15 points) (Continued from T6-D2 above)
After discussion on T6-D2, you are given the following new queries:

i. List the Student-ID, Name of 'Physics' majors in 'UAA2204' (Course-ID);
ii. List the Course-ID, Course Name, Instructors, for the student named 'Fish Leong';

For each problem, do the following:

a. Give an "SQL query" statement that will solve the problem, and
b. Give a sequence of basic DB operations (using only e-project, e-select, and e-join) to efficiently produce the same result.
(IMPORTANT NOTE: When using the basic primitive DB operations (*e-project, e-select, and e-join*) for the database query problems T6-D2 & T6-Q2, pay attention to the following:

1. the work done for each DB-operation -- (estimated) number of "row operations";
2. the estimated sizes of the intermediate tables produced;

They will help you estimate the total number of row operations needed for the entire query.)

Practice Problems: (not graded)
These are practice problems for you to try out. *(If you have difficulties with these practice problems, please quickly see your classmates or the instructor for help.)*

T6-PP1: (SQL Query) Read Chapter 13.3 (pp 598-606) of [SG3].
T6-PP2: (SQL Query) Problems 1, 2, 3 on page 606 (Chapter 13) of [SG3].
T6-PP3: (SQL Query) Problems 4, 5 on page 617 (Chapter 13) of [SG3].

Discussion Problems: -- Prepare (individually) for tutorial discussion.

T6-D1: (SQL Query)
Problems 6 on page 617 (Ch. 13) of [SG].
*(Note: First read Ch.13.3 of [SG3] to learn about SQL.)*

T6-D2: (This problem is given in the previous page)

Problems to be Handed in for Grading by the Deadline:
*(Note: Please submit hard copy to me. Not just soft copy via email.)*

T6-Q1: (5 points) (SQL Query) [Modified from Problems 7, p625, Ch 13 of [SG].]
(a) Using the Employees table of Figure 13.6 and the InsurancePolicies table of Figure 13.7, write an SQL query (the declarative type) that retrieves FirstName, LastName, PlanType, DataIssued for all employees who have insurance policy of PlanType 'B2'.
(b) Give a sequence of basic DB operations *(using only e-project, e-select, and e-join)* to implement the above query. If you can, make it as efficiently as possible.

T6-Q2: (Please see previous page for this problem)

T6-Q3: (10 points) (Question from a former Quiz) --
Question Q4 from [Spring 2010 Quiz].

A-Problems: OPTIONAL Challenging Problems for Further Exploration

A6-2013: (Really LARGE numbers -- how to do it!)
The running times for some entries in the table in T5-Q5 would cause overflow in your calculators -- and so, it was given as "too big to compute". Use your ingenuity (and knowledge of mathematics) to find a way (actually, also an algorithm) to compute these very big numbers with the help of calculators.
*[For example, for the function T(n)=2^n, when n=1000, the running time is 3.40 x 10^{291} yrs.]*
(Hint: John Napier, 1614)