QUESTION BANK FOR INSTRUCTORS

VERSION DATED: February 20, 2013

TO ACCOMPANY

# Database System Concepts

**Sixth Edition**

**Abraham Silberschatz**
Yale University

**Henry F. Korth**
Lehigh University

**S. Sudarshan**
Indian Institute of Technology, Bombay

# Contents

# Preface

This question bank provides a set of questions for different chapters of the book Database System Concepts 6th Ed. The goal of this question bank is to provide instructors with a large set of questions that could help them in creating examinations.

Each question has associated with it a number that is a rough estimate of the number of minutes that a good student should take to answer the question (for some of the tougher questions, the time taken may be significantly higher than the associated number). To take average students into account, the numbers should be inflated by a factor of around 1.5 or 2. (Obviously the time depends greatly on the specific student population, so your mileage may vary.)

Most of these questions appeared in examinations of courses taught by us. We welcome contributions of questions, to make this question bank a larger and more valuable resource.

Our current policy is to not publish answers along with the questions, to limit the damage done if any part of the material gets accidentally published on the Web by anybody. This may be revised based on feedback from instructors.

Avi Silberschatz, Hank Korth and S. Sudarshan.

C H A P T E R 1

# Introduction and Relational Model (Chapters 1 and 2)

We have not provided any questions for Chapter 1 (Introduction) and Chapter 2 (Relational Model) currently.

CHAPTER 3

# SQL (Chapters 3 and 4)

Questions on SQL covering Chapters 3 and 4 are provided here.

## Exercises

**3.1** Given the schema

```
item(itemid, name, category, price)
itemsale(transid, itemid, qty)
transaction(transid, custid, date)
customer(custid, name, street-addr, city)
```

where primary keys are underlined, write the following queries in SQL:

**a.** Find the name and price of the most expensive item (if more than one item is the most expensive, print them all). . . .5

**b.** Print the total sales (in terms of units and total price) of every item category in every customer-city. . . .5

**c.** Find items with no sales at all to customers in Mumbai . . .5

**d.** Find customers who bought the same quantity of the same item on subsequent dates. . . .5

**e.** Find all customers who did not buy any item in category "Electronics". . . .5

**3.2** Suppose you are given a relation $r(R, M)$, where $R$ indicates roll number and $M$ the marks scored.

**a.** Write an SQL query to compute the top 5 distinct marks. . . .3

**b.** Write an SQL query to find the "dense rank" of each student. That is, all students with the top mark get a rank of 1, those with the next highest mark get a rank of 2, and so on. Hint: split the task into parts, using the **with** clause. . . .10

**3.3** Consider a system to store the marks of students on various exams, for several courses.

**a.** Define a view `totalmarks(course, rollno, marks)` to get total marks for students of each course, given a table: `weights(course, exam, weightage)`. Here `weightage` is a factor by which you multiply the marks for the corresponding exam. . . .4

**b.** Define a view `grades(course, rollno, grade)` to get the grades for students, given a relation `cutoff(course, grade, cutoff-mark)`. Assume that marks below the lowest cutoff mark get an FF grade. . . .6

**3.4** I want to design a system to automate project groups signing up for project demo slots.

(Note: formal coverage of schema design is in Chapter 8, but this question can still be asked of smart students)

**a.** Design a relational schema for this task. You should record group members for groups. Enforce the following constraints: every student is in at most one group, each slot has at most one group, each group has at most one slot. . . .4

**b.** Write SQL queries to list students who are not in any group, and groups that have not yet signed up for a slot. . . .3+ 3

**c.** Write an SQL query to find all slots for which there is no group signed up. . . .3

**3.5** Given a relation $income(name, value)$ write an SQL query to find the first 10 tuples of $income$, sorted by $value$. The query should only output these 10 tuples; assume for simplicity that the income values of different people are guaranteed to be different. (Don't worry about efficiency.) . . .7

**3.6** Write an SQL query to list the name of each student and the total number of courses he/she is registered for. Assume relations $student(rollno, name)$ and $registered(rollno, course)$ are given. *Make sure that your query lists students who aren't registered for any course, with a count of 0.* Hint: use a nested subquery. . . .5

**3.7** Write SQL expressions to do the following. Assume you are given two relations, student(name, rollno) and marks(rollno, exam, mark)

**a.** Show names of all students who have got marks in at least two exams. . . .2

**b.** Find the names of the students with highest total marks (summed across all exams for each student). . . .3

**3.8** Given a table $r$ with a foreign key referencing $s$, what is the effect of adding the clause **on delete cascade** to the foreign key declaration? . . .2

**3.9** Give an example of a pair of relations with integrity constraints, and inserts into both of them such that, regardless of whichever insert is first, the integrity constraints are violated by the first insert, although they are satisfied after the second insert. (To handle such situations (among other reasons) integrity constraints are only checked at the end of transaction.) . . .5