p5q8
GMB

Databases 2003

~~Answer to Question 1~~

1. (a) *Define the operators in the core relational algebra.* *[5 marks]*

   As defined in the lecture notes, the core relational algebra consists of the following five (classes of) operators:

   i. The set-theoretic operators: union, intersection and difference.
   ii. Selection, $\sigma_{P(\vec{A})}$
   iii. Projection, $\pi_{\vec{A}}$
   iv. Products and joins: cartesian product $\times$ and natural joins $\bowtie$
   v. Renaming $\rho_{B/A}$

   A perfect answer would in addition to describing these operators, also mention the concept of *union compatibility* where appropriate.

   (b) *Define the domain relational calculus.* *[4 marks]*

   The relational calculus is an alternative to the relational algebra, and was also introduced by Codd. In contrast to the algebra, which is pretty procedural, the relational calculus is declarative. The relational calculus can be presented in one of two ways: the tuple relational calculus, and the domain relational calculus. The difference lies in what variables range over (in TRC they range over entire rows, in DRC they range over field values).

   A DRC query has the form $\{\langle x_1, \ldots, x_n\rangle | P(\langle x_1, \ldots, x_n\rangle)\}$, where the x_i are either domain variables, or constants, and $P(\langle x_1, \ldots, x_n\rangle)$ is a valid DRC formula. The result of the query is the set of all tuples $\langle x_1, \ldots, x_n\rangle$ for which the formula evaluates to true.

   Atomic DRC formulae are of the form $\langle x_1, \ldots, x_n\rangle \in R$ or $x_i \ \mathsf{op} \ x_j$ where op is taken from a given set of operators. DRC formulae are then either atomic, or make use of the familiar first-order logic operators (conjunction, disjunction, negation, implication, existential and universal quantification).

   (c) *Show how the relational algebra can be encoded in the domain relational calculus.* *[3 marks]*

   This is covered in the notes. We assume for simplicity that the attribute names coincide with our use of domain variables. Here are some cases, the rest are similar.

$$\begin{aligned}
R^* &= \{\langle x_1, \ldots, x_n\rangle | \langle x_1, \ldots, x_n\rangle \in R\} \\
(R \cap S)^* &= \{\langle x_1, \ldots, x_n\rangle | \langle x_1, \ldots, x_n\rangle \in R^* \wedge \langle x_1, \ldots, x_n\rangle \in S^*\} \\
(\pi_{x_i}(R))^* &= \{\langle x_i\rangle | \langle x_1, \ldots, x_n\rangle \in R^*\} \\
(\sigma_{P(\vec{x})}(R))^* &= \{\langle x_1, \ldots, x_n\rangle | \langle x_1, \ldots, x_n\rangle \in R^* \wedge P(\vec{x})\}
\end{aligned}$$

2. *A* constraint *can be expressed using relational algebra. For example, $R = \emptyset$ specifies the constraint that relation R must be empty, and $(R \cup S) \subseteq T$ specifies that every tuple in the union of R and S must be in T.*

   Consider the following schema.

   `RockStar(`$\overline{\texttt{name}}$`, address, gender, birthday)`
   `RockManager(`$\overline{\texttt{managername}}$`, starname)`

(a) *Give a constraint to express that rock stars must be either male or female.* *[1 mark]*

$$\sigma_{\texttt{gender}\neq\text{“}\texttt{Female}\text{”}\wedge\texttt{gender}\neq\text{“}\texttt{Male}\text{”}}(\texttt{MovieStar}) = \emptyset$$

(b) *Give a constraint to express the referential integrity constraint between the* ***RockStar*** *and* ***RockManager*** *relations. (Note:* ***starname*** *is intended to be a foreign key.) [3 marks]*

$$\pi_{\texttt{starname}}(\texttt{RockManager}) \subseteq \pi_{\texttt{name}}(\texttt{RockStar})$$

(c) *Give a constraint to express the functional dependency* ***name→address*** *for the* ***RockStar*** *relation.* *[4 marks]*

The idea is to form all pairs of RockStar tuples (r_1, r_2) and check that we do **not** find a pair that agrees in the name fields and disagrees in the address field. Thus we use a cartesian product.

$$\sigma_{RS1.\texttt{name}=RS2.\texttt{name}\wedge RS1.\texttt{address}\neq RS2.\texttt{address}}(RS1 \times RS2) = \emptyset$$

where RS1(2) is just a shorthand for the appropriate renaming of the **RockStar** relation.