Database Management System – cs422 DE

Assignment 2 – Week 2

---

(1) A relational database consists of a collection of

A. Tables
B. Fields
C. Records
D. Keys

ANS:

A

(2) A _____ in a table represents a relationship among a set of values.

A. Column
B. Key
C. Row
D. Entry

ANS:

C

(3) For each attribute of a relation, there is a set of permitted values, called the _____ of that attribute.

A. Domain
B. Relation
C. Set
D. Schema

ANS:

A

(4) Course(course_id, sec_id, semester)
Here the course_id, sec_id and semester are _____ and course is a _____ .

A. Relations, Attribute
B. Attributes, Relation
C. Tuple, Relation
D. Tuple, Attributes

ANS:

B

(5) Department (dept_name, building, budget) and
Employee (emp_id , name, dept_name, salary)
Here the dept_name attribute appears in both the relations.
Using the common attributes in relation schema is one way of relating _____ relations.

A. Attributes of common
B. Tuple of common
C. Tuple of distinct
D. Attributes of distinct

ANS:

D

(6) Student (ID, name, dept_name, tot_pts)
In this query which attribute form the primary key?

A. name
B. dept_name
C. tot_pts
D. ID

ANS:
D

(7) The_____ operation allows the combining of two relations by merging pairs of tuples, one from each relation, into a single tuple.

A. Select
B. Join
C. Union
D. Intersection

ANS:
B

(8) Discuss the differences between the five Join operations: Theta join, Equijoin, Natural join, Outer join (left), and Semijoin. Example of each is appreciated.

ANS:
Please consider we have two relations for examples.

1. Person (PersonId, Name, Age, Phone, AddressId)
2. Address (AddressId, Street, City, State, Zip)

- Theta join – defining a relation to work for different comparison operators over the Cartesion products of two relations
  e.g., getting Name and City information of people older than 50
  $\Pi_{Person.Name\ ,\ Address.City}$ (Person $\bowtie$ ($_{Person.AddressId = Address.AddressId\ and\ Person.Age > 50}$) Address)

- Equijoin – defining a relation to work only for equal comparison operator over the Cartesion products of two relations
  e.g., getting Name and City information of 20-year-old people
  $\Pi_{Person.Name\ ,\ Address.City}$ (Person $\bowtie$ ($_{Person.AddressId = Address.AddressId\ and\ Person.Age = 20}$) Address)

- Natural join – same with Equijoin, but eliminate the common attributes
  e.g., getting PersonId, Name, AddressId and City – 4 attributes of all people
  ($\Pi_{Name,\ AddressId}$ (Person)) $\bowtie$ $\Pi_{AddressId,\ City}$ (Address))

- Outer join (left) – defining a relation to take every matching tuple and even unmatched tuples from the left relation by filling NULL for the remaining attributes of right relation
  e.g., getting all people' Name with city (with NULL if that person has no address yet)
  $\Pi_{Person.Name,\ Address.City}$ (Person $\rtimes$ ($_{Person.AddressId = Address.AddressId}$) Address)

- Semijoin – defining a relation to get every possible tuples that can meet with the condition from the left relation
  e.g., getting all people who possibly live Fairfield city

Person ▷ Person.AddressId = Address.AddressId and Address.City = 'Fairfield') Address

(9) **A relational database contains details about journeys from Chicago to a variety of destinations and contains the following relations:**

Operator (**opCode**, opName)
Journey (**opCode, destCode**, price)
Destination (**destCode**, destName, distance)

Each operator is assigned a unique code (opCode) and the relation *Operator* records the association between this code and the Operator's name (opName).

Each destination has a unique code (destCode) and the relation *Destination* records the association between this code and the destination name (destName), and the distance of the destination from Chicago.

The relation *Journey* records the price of an adult fare from Chicago to the given destination by a specified operator; several operators may operate over the same route.

Formulate the following queries using relational algebra.

1) List the details of journeys less than $100.
2) List the names of all destinations.
3) Find the names of all destinations within 20 miles.
4) List the names of all operators with at least one journey priced at under $5.
5) List the names of all operators and prices of journeys to 'Boston'.

ANS:

1) $\sigma_{price > 100}$ Journey
2) $\Pi_{destName}$Destionation
3) $\Pi_{destName}(\sigma_{distance < 20}$ (Destionation))
4) $\Pi_{Operator.opName}$ (Operator $\bowtie_{Operator.opCode = Journey.opCode}$ ($\sigma_{price <5}$ Journey))
5) $\Pi_{Operator.opName,\ Journey.Price}$ ((Operator $\bowtie_{Operator.opCode = Journey.opCode}$ Journey) $\bowtie_{Journey.destCode = Destionation.destCode}$ ($\sigma_{destName <'Boston'}$ (Destination)))

(10) Solve Q 5.8 (a-d) on page no. 130 from the course text book (5th edition).

a) $\Pi_{hotelNo}$ ($\sigma_{price > 50}$ (Room))
ANS:
Listing the hotelNo of Room which price is greater than 50

b) $\sigma_{Hotel.hotelNo = Room.hotelNo}$(Hotel × Room)
ANS:
Listing all hotel and room details where Hotel's hotelNo and Room's hotelNo are same

c) $\Pi_{hotelName}$ (Hotel $\bowtie_{Hotel.hotelNo = Room.hotelNo}$ ($\sigma_{price > 50}$ (Room)))

ANS:

Listing hotelName of Hotel where a Room's price is greater than 50

d) Guest $\bowtie$ ($\sigma_{dateTo \geq '1\text{-}Jan\text{-}2007'}$ (Booking))

ANS:

Listing all Guest with Booking which are booked to the date 1-Jan-2007. If there is not booking of some guest, it will put NULL for Booking attributes for that Guest.