Relational algebra and queries

Grégoire Détrez

Jule 13, 2007

Domain Description

```
Airports(code,city)
FlightCodes(code, airlineName)
Flights(origin, destination, departure, arrival, code)
  origin -> Airports.code
  destination -> Airports.code
  code -> FlightCodes.code
```

The listed flight code is the prime flight (i.e. the one used by the operating airline). For simplicity, we assume that *departure* and *arrival* are integers denoting full hours, all in the same time zone, and that $0 \leq departure < arrival < 24$.

Question 1

Using this schema, write an SQL query that finds all airports that have departures or arrivals (or both) of flights operated by Lufthansa or SAS (or both).

Question 2

Using the schema, write an SQL query that shows the names of all cities together with the number of flights that depart from them, and sorts them by the number of flights in descending order (i.e. the city with the largest number of departures first).

Question 3

Using the above schema, write a view that lists all connections from any city X to any other city Y involving 1 or 2 legs (i.e. separate flights between two cities: if you fly from Gothenburg to Paris with a change in Frankfurt, the connection has two legs). The query must return a table with the following information (and nothing else):

- the departure city X and destination city Y
- the departure time from X and the arrival time in Y
- the number of legs
- the total time from departure in X to arrival in Y
- the total time spent in the air

A change is possible if and only if

1. it happens at the same airport,
2. the changing time is at least 1 hour, and
3. the connecting flight is on the same day.

Question 4

Express the query of question 1 by a relational algebra expression.

Question 5

Translate the following relational algebra expression to an SQL query:

$$\pi_{First.departure, Second.arrival}(\rho_{First}(Flights) \bowtie_{First.destination=Second.origin} \rho_{Second}(Flights)) \quad (1)$$

Domain Description

```
Tracks(trackId,title, length)
    length > 0
Artists(artistId, name)
Albums(albumId,title, yearReleased)
TracksOnAlbum(album,trackNr,track)
    album ->  Albums.albumId
    track ->  Tracks.trackId
    (album,track) unique
    trackNr > 0
Participates(track, artist)
    track ->  Tracks.trackId
    artist ->  Artists.artistId
```

```
Users(username, email, name)
    email unique
Playlists(user, playlistName)
    user ->  Users.username
InList(user, playlist, number,track)
    (user, playlist) ->  Playlists.(u
    track ->  Tracks.trackId
PlayLog(user,time,track)
    user ->  Users.username
    track ->  Tracks.trackId
    (user,time) unique
```

Domain Description (cont.)

An artist can be either a solo artist or a group, the design makes no difference between the two kinds. Tracks are recorded by one or more artists, and each track can appear on one or more albums (but no more than once on each album) to account for e.g. “Greatest hits” or collection albums.
Users of the site can register, in order to create playlists, which are simply ordered collections of tracks.
Finally, the system stores a log over all songs played by registered users, to calculate statistics and to give suggestions and feedback.
(Note: The actual music files to be streamed is considered to be stored separately, outside the scope of this schema.)

Question 1

Write an SQL query that lists all artists appearing on any album released this year (2016).

Question 2

Question 3

Question 4

Write an SQL query that finds the title, length and album title of the longest track in the database. If the track appears on more than one album, list the album where it appeared first. If more than one track of the same length qualifies, list the one that was released first, as given by the album it appears on. If there is still a tie, list all such tracks.

Question 5

What does the following relational algebra expression compute (answer in plain text):

$$\tau_x(\gamma_{playlistName, COUNT(*) \to x}(\sigma_{playlistName = playlist}(Playlists \times InList)))$$

Question 6

Translate the following relational algebra expression(s) to corresponding SQL:

$$let\ R1 = \gamma_{user,track,COUNT(*)\rightarrow nrTimes}(PlayLog)$$

$$\sigma_{avgNrTimes>=10}(\gamma_{track,AVG(nrTimes)\rightarrow avgNrTimes}(R1))$$

Question 7

Translate the following SQL query to relational algebra:

```
SELECT album, MAX(trackNr) AS nrOfTracks, SUM(length) AS totalLength
FROM Albums, TracksOnAlbum, Tracks
WHERE albumId = album
AND trackId = track
GROUP BY albumId
ORDER BY totalLength DESC;
```

Question 8

Write a relational algebra expression that lists the artist(s) appearing in the highest number of distinct playlists. In case of a tie for highest number of different playlists, list all such artists.