Implementation of Databases

Assignment 1

Participants:
(sorted in last name order)
Ulfet CETIN
Shreya KAR
Samuel ROY

Exercise 1.1 (Database Architecture)

Part1

Five layers of Database Systems:

1. Logical Data Structures: This layer of DB translates and optimizes queries (transaction programs) written in SQL, which is also the set-oriented interface. This layer uses record-oriented interface to interface with Logical Access structures.
2. Logical Access structures: It maps physical objects to its logical representation, manages cursor, sort components and dictionary. This layer uses Internal record interface to interface with Storage structures. Auxiliary data structures used: Records, sets, keys, access paths
3. Storage Structures: Storage Structure layer plays a key part in DBMS performance by implementing mapping functions and provides clustering facilities. Maintains all physical object representations, access path structures (B-Tree and internal catalogue information). This layer uses System buffer interface to interface with Page assignment. Auxiliary data structures used: B*Trees and Records.
4. Page Assignment: Page Assignment layer performs mapping of pages into physical blocks. It reduces the physical I/O by providing large DB buffer which act as page-oriented interface. This layer uses file interface to interface with File Management. Auxiliary Data structure used: page, block tables, DB buffer
5. Memory Assignment Structures: This layer operates on bit pattern stored on the physical layer. It works together with the operating system's file management, which copes with different physical characteristics of each type of physical storage device. Device interface is used to interface with physical layer. Auxiliary Data structure used: VTOC, extent tables, system catalogue

Physical Layer:
The physical layer consists of huge volumes of bits stored on non-volatile storage devices. The user makes use of DBMS to interpret the stored bits into information.

Part2

Tasks sorted in order to match the top-down architecture:

1. (b) logical relation and cursor management | logical data
2. (d) access path management | logical access
3. (e) view formulation and management | storage structure
4. (a) buffering | page assignment
5. (c) media access | memory assignment

Part3

(a) (1) Data Independence is an important feature of DBMS which makes data independent of applications.That is, applications are insulated from the way data is structured and stored.

(2) The goal of Data Independence is also to ensure that data is available to all applications.

(3) Data Independence is achieved through 3 layers of data abstraction i.e. External Schema, Conceptual/Logical Schema and Physical Schema.

(b) (1) Data independence is an important feature of the DBMS because it ensures that changes made at one layer of data abstraction do not affect the other layers.It ultimately saves times and brings down the cost by reducing the number of modifications needed to be made to the data.

    (2) Users(applications) are shielded from the changes made in the logical view of data, e.g. modifying a relation or schema . This is called logical independence. Similarly changes pertaining to how data physically stored only need to be done in the physical view. This is known as physical data independence.

(c) Data independence in the 5 layer DBS architecture is achieved as follows:

    (1) Layer 1 (Logical structures) : Position indicator and explicit relations of the schema are hidden to ensure data independence.
    (2) Layer 2 (Logical access paths): Data Independence is achieved by hiding the internal representation of records in a database and hiding the access paths/number of paths to access data.
    (3) Layer 3 (Storage Structures) : How the buffers are managed or data is logged is hidden from the above layers, which ensures data independence.
    (4) Layer 4 (Page assignment structures): Data independence is maintained by hiding details of file mapping, indirect page assignment in this layer.
    (5) Layer 5 (Memory assignment Structures): Technical details and features of the external storage media are hidden to ensure data independence.

Exercise 1.2 (Query Languages)

Relational Algebra

1

$\rho(allEmployeeReportToPairs, \Pi_{EmployeeId,ReportsTo}(Employee))$
:returns <EmployeeId, ReportsTo> pairs

$\rho(peopleReceiveReports, \Pi_{ReportsTo}(Employee))$
:returns <ReportsTo>

$\rho(peopleReporting, \Pi_{EmployeeId}(allEmployeeReportsToPairs \bowtie peopleReceiveReports))$
:returns <EmployeeId>

$\rho(personNotReporting, \Pi_{EmployeeId}(Employee) - peopleReporting)$
:returns <EmployeeId>

$\rho(personNotReportingCity, \Pi_{City}(Employee \bowtie personNotReporting))$
:returns <City>

$\Pi_{FirstName,LastName}(Customer \bowtie personNotReportingCity)$

2

$\rho(youngerThanThirtyFiveWhenEmployed, \Pi_{EmployeeId}(\sigma_{DATEDIFF(HireDate,BirthDate)<35}Employee))$
:returns <EmployeeId>

$\rho(youngerThanThirtyFiveWhenEmployedRenamed,$
$\quad \rho_{(SupportRepId \leftarrow EmployeeId)}(youngerThanThirtyFiveWhenEmployed))$
:returns <SupportRepId>

$\Pi_{FirstName,LastName,Country}(Customer \bowtie youngerThanThirtyFiveWhenEmployedRenamed)$

3

$\rho(employeesHelpingBrazilians, \Pi_{SupportRepId}(\sigma_{Country="Brazil"}Customer))$
:returns <SupportRepId>

$\rho(employeesHelpingBraziliansRenamed, \rho_{EmployeeId \leftarrow SupportRepId}(employeesHelpingBrazilians))$
:returns <EmployeeId>

$\rho(managersOfEmployees, \Pi_{ReportsTo}(Employee \bowtie employeesHelpingBrazilianCustomersRenamed))$
:returns <ReportsTo>

$\rho(managersOfEmployeesRenamed, \rho_{EmployeeId \leftarrow ReportsTo}(managersOfEmployees))$
:returns <EmployeeId>

$\Pi_{FirstName,LastName}(Employees \bowtie managersOfEmployeesRenamed)$

4

$\rho(allArtistID, \Pi_{ArtistId}(Artist))$
:returns <ArtistId>

$\rho(albumReleasedArtistIDs, \Pi_{ArtistId}(Album))$
:returns <ArtistId>

allArtistID - albumReleasedArtistIDs

SQL

1

Find the names of customers who live in the same city as the top employee (The one not managed by anyone).

```
SELECT c."FirstName", c."LastName"
FROM public."Customer" c
WHERE c."City" =
(
  SELECT e."City"
  FROM public."Employee" e
  WHERE e."ReportsTo" is NULL OR e."ReportsTo" NOT IN
  (
    SELECT otherEmployee."EmployeeId"
    FROM public."Employee" otherEmployee
    WHERE otherEmployee."EmployeeId" != e."EmployeeId"
  )
);
```

2

List the names and the countries of those customers who are supported by an employee who was younger than 35 when hired. (HINT: For SQL use year as the first parameter in TIMESTAMPDIFF(). For Relational ALgebra use the function DATEDIFF)

```
SELECT c."FirstName", c."LastName", c."Country"
FROM public."Customer" c
WHERE c."SupportRepId" IN
(
  SELECT e."EmployeeId"
  FROM public."Employee" e
  WHERE e."HireDate" < e."BirthDate" + INTERVAL '35 years'
);
```

3

Find the managers of employees supporting Brazilian customers.

```
SELECT DISTINCT m."FirstName", m."LastName"
FROM public."Employee" m, public."Employee" e, public."Customer" c
WHERE c."Country" = 'Brazil' AND
  c."SupportRepId" = e."EmployeeId" AND
  e."ReportsTo" = m."EmployeeId"
```

4

Which artists did not make any albums at all? Include their names in your answer.

```
SELECT a."Name"
FROM public."Artist" a
WHERE a."ArtistId" NOT IN
(
  SELECT al."ArtistId"
  FROM public."Album" al
)
```

Here is their names:

```
Milton Nascimento & Bebeto
Azymuthã
Joo Gilberto
Bebel Gilberto
Jorge Vercilo
Baby Consuelo
Ney Matogrosso
Luiz Melodia
Nando Reis
Pedro íLus & A Parede
Banda Black Rio
Fernanda Porto
Os Cariocas
A Cor Do Som
Kid Abelha
Sandra De áS
Hermeto Pascoalã
Baro Vermelho
Edson, DJ Marky & DJ Patife Featuring Fernanda Porto
Santana Feat. Dave Matthews
Santana Feat. Everlast
Santana Feat. Rob Thomas
Santana Feat. Lauryn Hill & Cee-Lo
Santana Feat. The Project G&B
Santana Feat. áMan
Santana Feat. Eagle-Eye Cherry
Santana Feat. Eric Claptoní
Vincius De Moraes & Baden Powellí
Vincius E Qurteto Em Cyí
Vincius E Odette Lara
Vinicius, Toquinho & Quarteto Em Cyö
Motrhead & Girlschool
Peter Tosh
R.E.M. Feat. KRS-One
Simply Red
Whitesnake
Christina Aguilera featuring BigElf
Aerosmith & Sierra Leone's Refugee Allstars
Los Lonely Boys
Corinne Bailey Rae
Dhani Harrison & Jakob Dylan
Jackson Browne
Avril Lavigne
Big & Rich
Youssou N'Dour
Black Eyed Peas
Jack Johnson
Ben Harper
Snow Patrol
Matisyahu
The Postal Service
Jaguares
The Flaming Lips
Jack's Mannequin & Mick Fleetwood
Regina Spektor
Xis
Nega Gizza
Gustavo & Andres Veiga & Salazar
```

```
Rodox
Charlie Brown Jr.
Pedro íLus E A Parede
Los Hermanos
Mundo Livre S/A
Otto
Institutoçã
Nao Zumbi
DJ Dolores & Orchestra Santa Massa
Seu Jorge
Sabotage E Instituto
Stereo Maracana
Academy of St. Martin in the Fields, Sir Neville Marriner & William Bennett
```

5

List the top 5 most purchased tracks over all.

Note that, in this database, there are more than 5 tracks that has been sold the most, based on TrackId.
We asked this question in L2P.

There are three possible scenarios we found:

1. Same song by the same artist may exist in different albums, that we noticed. For this purpose, we assumed if the tuple (name of the artist, the name of the track) is the same for the very same tracks with different TrackId values, then tracks are regarded as the same, regardless of their TrackId values.

2. Looking for the tracks that have been sold the most based on the multiplication of the fields "Quantity" and "UnitPrice" from the table "Invoice"

3. Looking for the tracks that have been sold the most based on the field "Quantity"

We speculated the following foundings of our ideas on those three approaches:

1. The most feasible one; because this approach allows one to find which (the song name, the artist name) tuple "objects" have been sold the most, regardless of TrackId of the track.

2. Feasible, but it observes "the tracks that have the same name by the same artist, but with different TrackId" as different.

3. Also feasible, but not to the same extend as the first one; this approach allows one to find which tracks has been sold the most(in quantities), but counts the tracks that have the same name and the same artist as different.5

Our actual answer is the "first approach", yet, we wanted to convey the idea that we analyzed the tables and determined that information given in assignment text is not enough:

- e.g., there can be two song from same artist with the same track name, yet have different melodies/lyrics/arrangement, which also happens in real life, that cannot be count as "the same track" in a musical perspective.
- Yet, between those three approaches, only the first approach yields five pseudo-unique tracks that has been sold the most.
- The rest of the approaches yield answers that are not unique → based on the ordering, top 5 changes.

Our answer(s) based on the assumptions given above:

1. **First approach:**
   (this is our designated answer, in case submitting only one answer is mandatory)

```
SELECT TrackName
FROM (
  SELECT t1."Name" AS TrackName, SUM(inv."Quantity") as Total
  FROM "Track" t1, "Artist" art, "Album" a1, "InvoiceLine" inv
  WHERE
    art."ArtistId" = a1."ArtistId" AND
    a1."AlbumId" = t1."AlbumId" AND
    t1."TrackId" = inv."TrackId"
  GROUP BY (art."Name", t1."Name")
  ORDER BY Total DESC
  LIMIT 5
) as derivedTable;
```

2. Second approach:

```
SELECT *
FROM public."Track"
WHERE "TrackId" IN
(
  SELECT "TrackId"
  FROM public."InvoiceLine"
  GROUP BY "TrackId"
  ORDER BY SUM("Quantity"*"UnitPrice")
  DESC
  LIMIT 5
);
```

3. Third Approach

```
SELECT *
FROM public."Track"
WHERE "TrackId" IN
(
  SELECT "TrackId"
  FROM public."InvoiceLine"
  GROUP BY "TrackId"
  ORDER BY SUM("Quantity")
  DESC
  LIMIT 5
);
```