Implementation of Databases Sheet 1

Ilya Kulikov 351063, Alina Shigabutdinova, Oleg Chernikov

November 2, 2015

Exercise 1.1

1. There are layers in the database system:
    - Logical data structures: this layer contains auxiliary structures: external schema description and integrity rules. The goal of this layer is to translate and optimize queries. Interface with transaction programs is SQL, which operates with relations, views and tuples.
    - Logical access structures: this layer contains auxiliary structures: access path data, internal schema description. The task of this layer is to manage cursor, sort components and dictionary. Interface with logical data structures is record oriented interface, which operates with records, sets, keys, access paths.
    - Storage structure: this layer contains auxiliary structures: DBTT, FPA, page indexes etc. The task of this layer is to manage record and index in the database system. Interface with Logical access structures is Internal record interface, which operates with records, B* trees etc.
    - Page assignment: this layer contains auxiliary structures: page and block tables. The task of this layer is to manage buffer and segments. The interface with Storage structure is System buffer interface, which operates with pages and segments.
    - Memory assignment structures: this layer contains auxiliary structures: VTOC, extent tables, system catalogue. The task of this layer is to manage files and external memory. The interface with Page assignment is File interface, which operates with blocks and files. The interface with Physical volume is Device interface, which operates with tracks, cylinders, channels etc.
2. The following sequence matches the top-down architecture:
   (E) $\Rightarrow$ (B) $\Rightarrow$ (D) $\Rightarrow$ (A) $\Rightarrow$ (C)
3. a. Data independence means that application or some external representation of the data is independent from internal storage.

It is not definitely given in the slides, but Data Independence contains two parts: Logical Data Independence and Physical Data Independence. Logical Data is a metadata and data about database itself, i.e. it describes how the actual data stored and managed in the database. So Logical Data Indeendence is a kind of mechanism, which acts independently from actual data stored on the disk. For example, if we do some change on table format, it should not change the data on the disk.
Physical data independence is the mechanism which allows to change physical data without impact the schema or some other logical data.

b. Data independence is an important feature because it is not feasible to store the same data for each application several times and to provide shared data to several applications and to be sure, that data will be accessible and consistent. More over, with data independence it is possible to use logical data structures and descriptive queries.

c. Data structures and smart modern interfaces hiding all low-level layers and working with filesystems and block devices.

Exercise 1.2

Note: we use construct *field1* **g** *aggregation_function(field2)* to denote grouping operation by *field1* and usage of an *aggregation_function* on *field2*. This notation is taken from A. Silberschatz, H.Korth, S. Sudarshan, Database System Concepts, 5th Edition.

1.
   - SELECT e1."EmployeeId", e1."LastName", e1."FirstName", e1."Phone", e1."Email" FROM "Employee" e1, "Employee" e2 WHERE e2."EmployeeId" = e1."ReportsTo" AND e1."HireDate" > e2."HireDate"
   - $\pi_{\text{EmployeeId,LastName,FirstName,Phone,Email}}(\sigma_{\text{HireDate}>\text{hd}}(\text{Employee} \bowtie_{\text{ReportsTo=EmployeeId}} \rho_{\text{eid}\leftarrow\text{EmployeeId},\ \text{hd}\leftarrow\text{HireDate}}(\text{Employee})))$
2.
   - SELECT sum("Total") FROM (SELECT iid, count(alid) c FROM (SELECT DISTINCT i."InvoiceId" iid, i."CustomerId" cid, al."ArtistId" alid FROM "InvoiceLine" il, "Invoice" i, "Track" t, "Album" al WHERE i."InvoiceId" = il."InvoiceId" AND t."TrackId" = il."TrackId" AND al."AlbumId" = t."AlbumId" ORDER BY i."InvoiceId") temp1 GROUP BY iid) temp2, "Invoice" i WHERE i."InvoiceId" = iid AND c > 1 GROUP BY "CustomerId"
   - $\pi_{\text{CustomerId,sum(Total)}}(\text{CustomerId } \mathbf{g} \text{ sum(Total)}$ $(\sigma_{\text{count(ArtistId)} > 1}((\text{InvoiceId } \mathbf{g} \text{ count(ArtistId)}(\pi_{\text{InvoiceId,ArtistId}}((((\text{InvoiceLine} \bowtie \text{Invoice}) \bowtie \text{Track}) \bowtie \text{Album}))))) \bowtie \text{Invoice}))$
3.
   - SELECT sum(il."UnitPrice"), sum(t."Bytes") FROM "Playlist" p, "PlaylistTrack" pt, "InvoiceLine" il, "Track" t WHERE p."Name"

= 'Grunge' AND pt."PlaylistId" = p."PlaylistId" AND il."TrackId" = pt."TrackId" AND t."TrackId" = pt."TrackId"

- $(\mathbf{g}\quad sum(\text{Bytes}), sum(\text{UnitPrice})(\pi_{\text{Bytes, UnitPrice}}(((\sigma_{\text{Name='Grunge'}}(\text{Playlist}) \bowtie \text{PlaylistTrack}) \bowtie \text{Track}) \bowtie \text{InvoiceLine})))$

4.
- SELECT p."Name" FROM "Playlist" p WHERE p."PlaylistId" NOT IN (SELECT DISTINCT pt."PlaylistId" ptid FROM "Track" t, "Album" al, "Artist" ar, "PlaylistTrack" pt WHERE (ar."Name" = 'Black Sabbath' OR ar."Name" = 'Chico Buarque') AND al."ArtistId" = ar."ArtistId" AND t."AlbumId" = al."AlbumId" AND pt."TrackId" = t."TrackId")
- $\pi_{\text{PlaylistId}}(\text{Playlist}) - \pi_{\text{PlaylistId}(((\sigma_{\text{Name='Black Sabbath'} \vee \text{Name='Chico Buarque'}}(Artist) \bowtie \text{Album}) \bowtie \text{Track}}) \bowtie \text{PlaylistTrack})$