**Databases (**[**INF.01014UF**](javascript:subTopicClickM(%22c0508141242_34.htm%22,%22Examination%20(INF.01014UF%20Databases)%22,%22%22))**) – Report**

Hojun, Cho ( 11739592 )

Dayeong, Lee ( 11739321 )

Table of Content

1. Database schema **1**
2. Functional dependencies 2
3. Current State of the database3
4. Database queries in terms of the Relational Algebra**4**
5. Database queries in terms of the Relational Calculus 5
6. Database queries in terms of the SQL6
7. Database queries in terms of the SQL without nested SQL Blocks…………………..…….
8. Practical implementation of the database with SQL **4**
9. Servlets (Database Modification) 5
10. Servlets (Database Queries)6
11. Database schema

**Domain :**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| CID | Int | The client ID |
| LID | Int | The airline ID |
| TDTime | Datetime | The departure time |
| TATime | Datetime | The arrival time |
| PDID | Int | The departure airport ID number |
| PAID | Int | The arrival airport ID nubmer |
| TMileage | Int | The Mileage of the airticket |
| TSeat | Int | The seat |
| PID | Int | The airport ID number |
| Pname | String | The airport name |
| PCity | String | The airport location city |
| LID | Int | The airline ID number |
| LName | String | The airline name |
| LUnion | String | The air union of airline |
| LNation | String | The nationality of airline |
| CName | String | The name of client |
| CAge | Int | The age of client |
| CGender | Boolean | The gender of client |
| CNation | String | The nationality of client |

\*\*\* The domain of PID, PDID and PAID are same. \*\*\*

* **Relation : airticket** ( CID, LID, TDTime, TATime, TMileage, TSeat, PDID, PAID )
* **Relation : airline** ( LID, LName, LUnion, LNation )
* **Relation : airport** ( PID, PName, PCity )
* **Relation : client** ( CID, CName, CAge, CGender, CNation )

1. Functional Dependencies

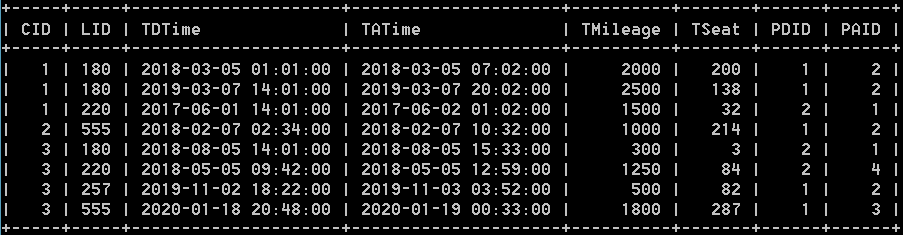
* **Relation : airticket** ( CID, LID, TDTime, TATime, TMileage, TSeat, PDID, PAID )  
  (CID, LID, TDTIme) → TATime  
  (CID, LID, TDTIme) → TMileage  
  (CID, LID, TDTIme) → TSeat  
  (CID, LID, TDTIme) → TDID  
  (CID, LID, TDTIme) → TAID
* **Relation : airline** ( LID, LName, LUnion, LNation )  
  LID → LName  
  LID → LUnion  
  LID → LNation
* **Relation : airport** ( PID, PName, PCity )  
  PID → PName  
  PID → PCity
* **Relation : client** ( CID, CName, CAge, CGender, CNation )  
  CID → CName  
  CID → CAge  
  CID → CGender  
  CID → CNation

**AirTicketDB is in the 3rd normal form:**

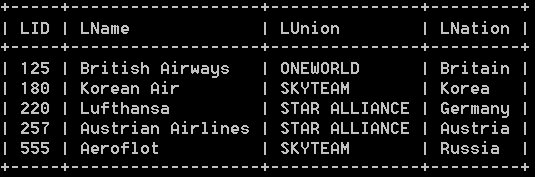
* To satisfy the 3rd normal form, non-primary keys should be dependent only on the primary key set, not on the subset of the primary key. That is, there should be no dependency between non-primary keys. Also, it should be in the the 2nd normal form.
* In the case of client, non-primary keys are CName, CAge, CGender, CNation. They are dependent only on CName. Also, there is no dependency between them.
* In the case of airticket, airline and airport, all of them satisfies the condition of 3rd normal form like the case of client.
* Therefore, Our DB, AirTicketDB, satisfies the condition of the 3rd normal form.

1. Current state of the database

**<Airticket>**



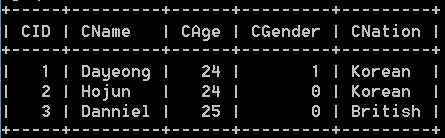
**<Airline>**



**<Airport>**



**<Client>**



1. Database queries in terms of the Relational Algebra

**Get the the locations of airport, which the clients, whose name are ‘Hojun’, visit.**

Select client Where CName = ‘Hojun’ Giving A;

Join A and airticket over CID Giving B;

Join B and airport over airport.PID = B.PDID Giving D;

Join B and airport over airport.PID = B.PAID Giving E;

D Union E Giving F;

Project F over PCity Giving Result;

**Get the names of client, who have ever used both of airline ‘korean air’ and ‘Japan air’.**

Select airline Where LName = ‘korean air’ Giving A;

Join A and airticket over LID Giving B;

Select airline Where LName = ‘Japan air’ Giving C;

Join C and airticket over LID Giving D;

B Intersect D Giving E;

Join E and client over CID Giving F;

Project F over CName Giving Result;

**Get the names of client, who have ever arrived at the airport ‘Vienna’ and have not ever arrived at the airport ‘Graz’**

Select airport Where PName = ‘Vienna’ Giving A;

Join A and airticket Over A.PID = airticket.PAID Giving B;

Select airport Where PName = ‘Graz’ Givng C;

Join C and airticket Over C.PID = airticket.PAID Giving D;

B Minus D Giving E;

Join E and client over CID Giving F;

Project F Over CName Giving Result;

1. Database queries in terms of the Relational Calculus

**Get the seat number of the ticket, of which the client is ‘hojun’, departure time is ‘2018-07-07’ and airline is ‘korean air’.**

T -> ticket

C -> client

L -> airline

(T.TSeat) : ( T.CID = C.CID & T.TDTime = ‘2018-07-07’ & T.LID = L.LID & C.CName = ‘hojun’ & L.LName = ‘korean air’)

**Get the lowest Mileage of the ticket, of which the owner is ‘hojun’ and the airline is ‘korean air’**

T1 -> airticket

T2 -> airticket

C -> client

L -> airline

(T1.TMileage) : ∀T2 ∃C ∃L ( C.CName = ‘hojun’ & L.LName = ‘korean air’ & T1.CID = C.CID & T1.LID = L.LID & T2.CID = C.CID & T2.LID = L.LID & T1.TMileage <= T2.TMileage)

**Get the highest seat number of the ticket, of which the departure airport is ‘Vienna’, the arrival airport is ‘Graz’ and the airline is ‘LOT’.**

T1 -> airticket

T2 -> airticket

P1 -> airport

P2 -> airport

L -> airline

(T1.TSeat) : ∀T2 ∃P1 ∃P2 ∃L ( T1.PDID = P1.PID & T2.PDID = P1.PID & T1.PAID = P2.PID & T2.PAID = P2.PID & T1.LID = L.LID & T2.LID = L.LID & P1.PName = ‘Vienna’ & P2.PName = ‘Graz’ & L.LName = ‘LOT’ & T1.TSeat >= T2.TSeat)

1. Database queries in terms of the SQL

**Get the client names, who has a ticket of which Korean air mileage is 4000.**

Select CName from client where CID IN

( Select CID from airticket where LID IN

( Select LID from airline where LName = 'Korean Air')

AND TMileage =4000);

**Get the client names, whose ‘SKYTEAM’ Union mileage sum is over 5000;**

Select CName from client where CID IN

( Select A.CID from

( Select \* from airticket where LID IN

( Select LID from airline where LUnion = 'SKYTEAM')

) A GROUP BY CID having sum(TMileage) >=5000

);

**Get the number of male and female, who arrive at the airport 'Incheon International Airport’.**

Select CGender, count(CID) from client where CID IN

( Select CID from airticket where PDID IN

( Select PID from airport

where PName = 'Incheon International Airport')

)GROUP BY CGender;

1. Database queries in terms of the SQL without nested SQL Blocks

**Get the client names, whose Korean air mileage is 4000.**

Select CName from airline, airticket, client where

client.CID = airticket.CID AND airticket.LID = airline.LID

AND airline.LName = 'Korean Air' AND airticket.TMileage = 4000;

**Get the client names, whose ‘SKYTEAM’ Union mileage sum is over 5000;**

Select CName from airline, airticket, client where

airline.LUnion = 'SKYTEAM' AND airticket.LID = airline.LID

AND airticket.CID = client.CID

GROUP BY client.CID having sum(TMileage) >= 5000;

**Get the number of male and female, who arrive at the airport 'Incheon International Airport’.**

Select CGender, count(DISTINCT client.CID)

from airticket, airport, client where

airticket.PDID = airport.PID AND airticket.CID = client.CID

AND airport.PName = 'Incheon International Airport'

GROUP BY CGender;

1. Practical implementation of the database with SQL

CREATE DATABASE 11739592\_airticketDB;

USE 11739592\_airticketDB;

# create all relations

CREATE TABLE airticket(

CID INT NOT NULL,

LID INT NOT NULL,

TDTime DATETIME NOT NULL,

TATime DATETIME NOT NULL,

TMileage INT NOT NULL,

TSeat INT NOT NULL,

PDID INT NOT NULL,

PAID INT NOT NULL,

primary key (CID,LID,TDTime),

foreign key (CID) references client(CID) ON DELETE CASCADE,

foreign key (LID) references airline(LID) ON DELETE RESTRICT,

foreign key (PDID) references airport(PID) ON DELETE RESTRICT,

foreign key (PAID) references airport(PID) ON DELETE RESTRICT

);

CREATE TABLE client(

CID INT NOT NULL primary key,

CName VARCHAR(30) NOT NULL,

CAge INT NOT NULL,

CGender BOOLEAN NOT NULL,

CNation VARCHAR(30) NOT NULL

);

CREATE TABLE airport(

PID INT NOT NULL primary key,

PName VARCHAR(50) NOT NULL,

PCity VARCHAR(20) NOT NULL

);

CREATE TABLE airline(

LID INT NOT NULL primary key,

LName VARCHAR(30) NOT NULL,

LUnion VARCHAR(30) NOT NULL,

LNation VARCHAR(30) NOT NULL

);

# insert the content into the relations

INSERT INTO airticket(CID, LID, TDTime, TATime, TMileage, TSeat, PDID, PAID) VALUES

(1, 180, "2018-03-05 01:01:00", "2018-03-05 07:02:00", 2000, 200, 1, 2),

(1, 180, "2019-03-07 14:01:00", "2019-03-07 20:02:00", 2500, 138, 1, 2),

(1, 220, "2017-06-01 14:01:00", "2017-06-02 01:02:00", 1500, 32, 2, 1),

(2, 555, "2018-02-07 02:34:00", "2018-02-07 10:32:00", 1000, 214, 1, 2),

(3, 257, "2019-11-02 18:22:00", "2019-11-03 03:52:00", 500, 82, 1, 2),

(3, 180, "2018-08-05 14:01:00", "2018-08-05 15:33:00", 300, 3, 2, 1),

(3, 555, "2020-01-18 20:48:00", "2020-01-19 00:33:00", 1800, 287, 1, 3),

(3, 220, "2018-05-05 09:42:00", "2018-05-05 12:59:00", 1250, 84, 2, 4);

INSERT INTO client(CID, CName, CAge, CGender, CNation)

VALUES (1, "Dayeong", 24, TRUE, "Korean"),

(2, "Hojun", 24, FALSE, "Korean"),

(3, "Danniel", 25, FALSE, "British");

INSERT INTO airport(PID, PName, PCity)

VALUES (1, "Incheon International Airport", "Incheon"),

(2, "Vienna international Airport", "Vienna"),

(3, "London Heathrow Airport", "London"),

(4, "Domodedovo International Airport", "Moscow");

INSERT INTO airline(LID, LName, LUnion, LNation)

VALUES (180, "Korean Air", "SKYTEAM", "Korea"),

(220, "Lufthansa", "STAR ALLIANCE", "Germany"),

(555, "Aeroflot", "SKYTEAM", "Russia"),

(257, "Austrian Airlines", "STAR ALLIANCE", "Austria"),

(125, "British Airways", "ONEWORLD", "Britain");

# print the content of tables (just for testing if everything went right)

SELECT \* FROM airticket;

SELECT \* FROM airline;

SELECT \* FROM airport;

SELECT \* FROM client;

# now start with the queries (see queries from chapter 6 and 7)