Information system of a transport company

Piotr Pawelczyk (Paw0025)

1 Assignment Specification

WHY? We need an information system for the management of the transport and logistics services.

WHO? The system can be used by two types of users. The first category represents the clients that can create new orders, new places and routes. A user account for the client can be created by themselves. The second category represents the administrator who manage the drivers, trucks and payments and set additional details of the transport. An account for an administrator can be only made by another administrator.

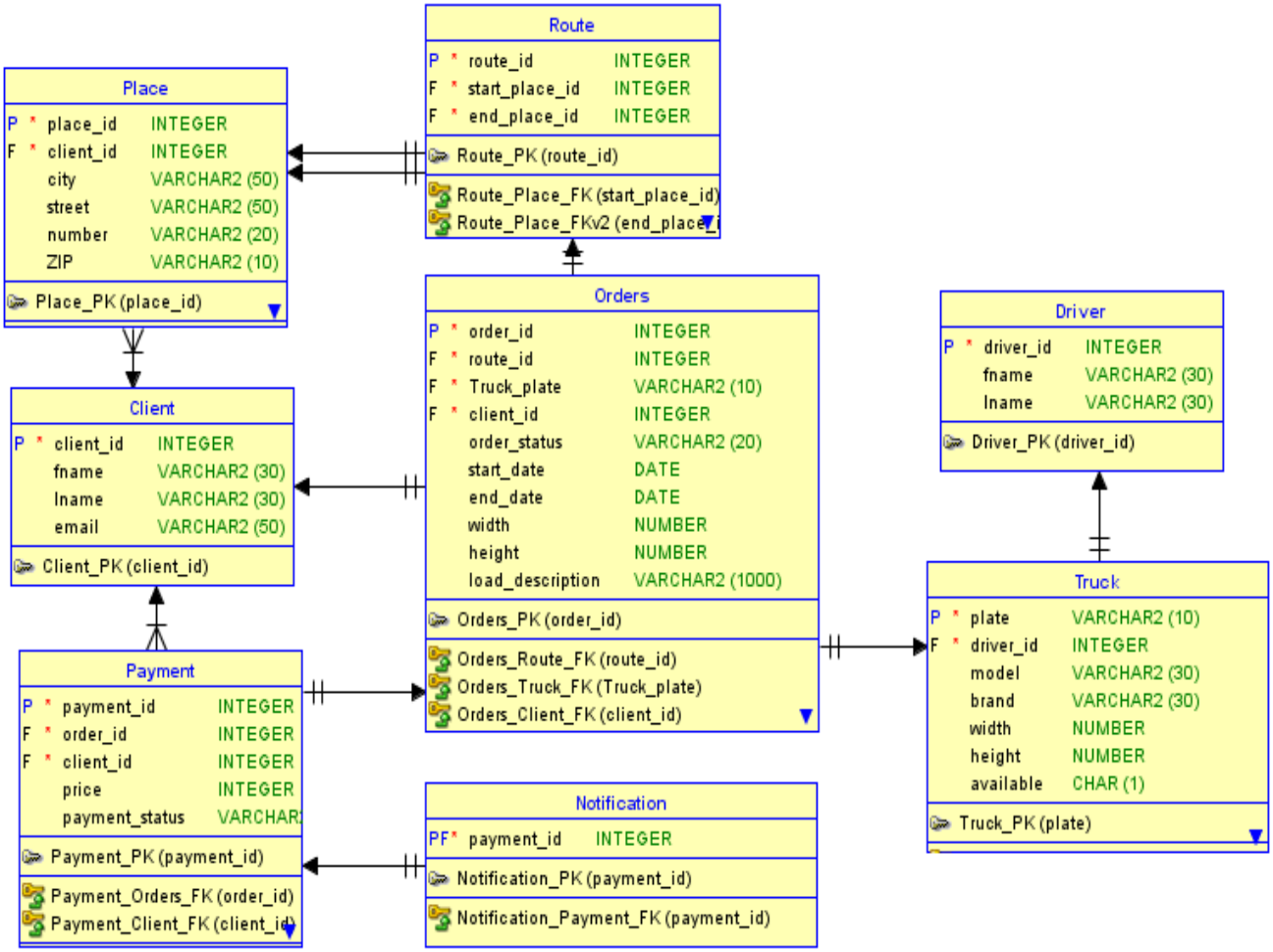
INPUTS: Detailed information about the transport.

OUTPUTS: List of client's orders, list of payments the client needs to pay, list of the places that the client has inserted, the details of an order, all clients routes, list of routes the truck has completed this month, list of drivers, list of trucks, list of orders, list of payments

FUNCTIONS: The client is able to create his own account, can insert new orders, new places and new routes. The administrator may update the information about the orders that the client created as well as introduce new drivers, new trucks and new payments to the system. Where it is necessary the system must check additional constraints.

2 Conceptual Model

ER Diagram



Entity Types

Legend**: Table**, Primary key , *foreign key*, attribute

Place (place_id, *client_id,* city, street, number, ZIP)
Route (route_id, *start_place_id, end_place_id*)
Driver (driver_id, fname, lname)
Client (client_id, fname, lname)
Payment (payment_id, *order_id, client_id,* price, payment_status)
Notification (*payment_id*)
Truck (plate, *driver_id,* model, brand, width, height, available)
Order (order_id, *route_id, truck_plate, client_id*, order_status, start_date, end_date, width, height, load_description)

3 Data Models

Description of tables is depicted in the following tables.

Table **Place**

No	Column Name	PK	FK	Data Type	ND	Index	IR	Description
1	place_id	P		Integer	N	A		
2	client_id		F(client)	Integer	N			Client who created such address record
3	city			VARCHAR (50)	N			Address – city name
4	street			VARCHAR (50)	N			Address – street name
5	number			VARCHAR (20)	N			Address – numer of the building
6	ZIP			VARCHAR (10)	N			Address – ZIP code

Table **Route**

No	Column Name	PK	FK	Data Type	Null	Index	IR	Description
1	route_id	P		Integer	N	A		
2	start_place_id		F(place)	Integer	N			Embarking place id
3	end_place_id		F(place)	Integer	N			Delivery place id

Table **Driver**

No	Column Name	PK	FK	Data Type	Null	Index	IR	Description
1	driver_id	P		Integer	N	A		
2	fname			VARCHAR (30)	N			Fist name of the driver
3	lname			VARCHAR (30)	N			Last name of the driver

Table **Client**

No	Column Name	PK	FK	Data Type	Null	Index	IR	Description
1	client_id	P		Integer	N	A		
2	fname			VARCHAR (30)	N			First name of the client
3	lname			VARCHAR (30)	N			Last name of the client

Table **Payment**

No	Column Name	PK	FK	Data Type	NULL	Index	IR	Description
1	payment_id	P		Integer	N	A		
2	order_id		F(order)	Integer	A			Order that is assigned to the payment
3	client_id		F(client)	Integer	N			The client who ordered a transport
4	price			Integer	A			Price of the transport
5	payment_status			VARCHAR (20)	N		3	

Table **Notification**

No	Column Name	PK	FK	M	Data Type	NULL	Index	IR	Description
1	payment_id	P	F(payment)	Y	Integer	N	A		Unpaid payment

Table **Truck**

No	Column Name	PK	FK	Data Type	Null	Index	IR	Description
1	plate	P		VARCHAR (10)	N	A		Truck's plates(id)
2	driver_id		F(driver)	Integer	A			Driver assign ned to the truck
3	model			VARCHAR (30)	N			Name of the truck's model
4	brand			VARCHAR (30)	N			Name of the trucks brand
5	width			DOUBLE	N			Maximum width of the load
6	height			DOUBLE	N			Maximu height of the load
7	available			Boolean (1)	N		4	Availability of the truck

Table **Order**

No	Column Name	PK	FK	Data Type	Null	Index	IR	Description
1	order_id	P		Integer	N	A		
2	route_id		F(route)	Integer	N			Route of the order
3	Truck_plate		F(truck)	VARCHAR (10)	A			Truck that takes the order
4	client_id		F(client)	Integer	N			Client who put an order
5	order_status			VARCHAR (20)	N		1	
6	start_date			Date	N		2	Date of embarking
7	end_date			Date	A		2	Date of delivering
8	width			DOUBLE	N			Width of the load
9	height			DOUBLE	N			Height of the load
10	load_description			VARCHAR (1000)	N			

Integrity Restrictions:

1. order_status : not_started, executed, finished
2. start_date <= end_date
3. payment_status : unpaid, paid, paid_but_deleted, failed_to_pay
4. available : <0,1>

4 Functional Analisys

4.1 List of functions

1. Client management

Table: Client

1.1 Client Insert

Responsibility: Client

1.2 Client Update

Responsibility: Client

1.3 List of clients (with a definition of a filter to search users)
Responsibility: Admin
1.4 Client detail
Responsibility : Admin; Client see only the own record

2. Place management
Table: Place
2.1 New place
Responsibility: Client
2.2 Place details
Responsibility: Admin; Client see only places created by himself
2.3 List of Places
Responsibility: Admin; Client see only places created by himself

3. Driver management
Table: Driver, *Responsibility:* Admin
3.1 New Driver
3.2 Update Driver
3.3 List of Drivers
3.4 List of routes a Driver has completed in last 30 days

4. Payment management
Table: Payment
4.1 New Payment
Responsibility: Admin
4.2 Update Payment
Responsibility: Admin
4.3 Delete Payment
Responsibility: Client – occurs alongside deleting an order, only if it payment's status is 'unpaid';
Admin – admin can delete a payment if the payment status is 'paid_but_deleted'
4.4 List of client's payments
Responsibility: Admin; Client
4.5 Sum of client's unpaid payments
Responsibility: Admin; Client

5. Route management
Table: Route, *Responsibility*: Client
5.1 New Route
5.2 Route details
5.3 List of client's routes

6. Truck Management

Table: Truck, *Responsibility:* Admin

6.1 New Truck

6.2 Update Truck

a) Update driver_id

b) Update available

6.3 List of Trucks

7. Order Management

Table: All

7.1 New Order

Responsibility: Client

7.2 Update Order

Responsibility: Admin

7.3 Delete Order

Responsibility: Client

7.4 List of Orders (with a definition of a filter)

Responsibility: Admin; Client can see only his orders

7.5 Specific order details with additional information about payment

Responsibility: Admin; Client can see only his orders

8. Other Functions

Table: Notification

8.1 Notify users who have unpaid payments – this function is executed once a day. It finds payments that have an order to be finished soon and that are unpaid. Such payment is inserted into table notification (in a real one an e-mail would be sent)

4.2 Detail Description of Functions

4.2.1 Driver management

3.4 List of routes a Driver has completed in last 30 days

Input:

The specified driver's id *$driver_id*

Date of performing a function *$current_date*

1. Create a cursor of all the orders that a specific driver has completed in last 30 days

Cursor *$specific_order* =
SELECT * FROM orders
WHERE TRUCK_PLATE=
(SELECT plate FROM truck WHERE driver_id = *$driver_id*
AND (*$current_date* -orders.START_DATE)<30
AND orders.order_status='executed')

2. For all of these orders extract the details of the places that the delivery had been to

   SELECT p1.city, p1.street, p1."number" ,p2.city, p2.STREET, p2."number"
   FROM route r1
   JOIN place p1 ON r1.start_place_id=p1.place_id
   JOIN place p2 ON r1.end_place_id=p2.place_id
   WHERE r1.route_id= *$specific_order.route_id*

4.2.2 Function 4 Payment Management

4.5 Sum of client's unpaid payments
Input:
the client's ID *$client_id*

SELECT sum (price) FROM payment
WHERE client_id=*$client_id* AND payment_status='unpaid'

4.2.3 Function 6 Truck Management

6.2b Update Truck – Update available

Input:
The specified truck's plate *$truck_plate*
Date of performing a function *$current_date*

This function is a transaction.

1. Create a cursor that cursor selects all the orders that the truck is assigned to and have not yet started.

Cursor *$specified_order* =
SELECT * FROM orders WHERE truck_plate=*$truck_plate*
AND start_date>*$current_date*

2. For all of these orders assigned to that truck set their attribute truck_plate to null.

   UPDATE orders SET truck_plate = null
   WHERE order_id = *$specified_order.order_id*

3. The Truck's attribute available is set to 0

   UPDATE truck SET available=0 WHERE plate=*$truck_plate*

4.2.4. Function 7. Order Management

7.1 New Order
Input:
clients ID *$client_id*
date specified by the client *$date*,
route specified by the client *$route*
width of clients order *$width ,*
height of clients order *$height,*
description of clients order *$description,*
date of performing a function *$current_date*

This function is a transaction.

1. Create a cursor of all the available trucks:

   Cursor *$available_truck=*
   SELECT * FROM truck WHERE available=1

2.For every available truck the system must check if it does not have any other orders at that time (up to 3 days before and after the truck must not start any other order).

SELECT * FROM orders
WHERE truck_plate=*$available_truck.plate*
AND start_date-*$current_date* < 3
AND start_date-*$current_date* >(-3)

3. If there is no such truck available the operation is finished unsuccessfully and the Client is informed that such date is unavailable.

4. If such trucks are available, for each of them the system must check if the width and height of the load is not too big for the trucks

```
SELECT * FROM truck
WHERE plate=$available_truck.plate
AND width>$width
AND height>$height
```

5. If there are no such trucks, the operation is finished unsuccessfully and the Client is informed that such width and height of a load is impossible to transport

6. If such trucks are available new record is inserted

```
INSERT INTO Orders
(route_id,client_id,width,height,load_description,start_date,order_status)
VALUES
($route, $client_id, $width, $height, $description, $date, 'not_started')
```

7.3 Delete Order

Input: the client's ID *$client_id*
date of performing a function *$current_date*
order specified by the client *$order_id*

This function is a transaction.

1. Create a cursor for all the client's orders that have not yet started

```
Cursor $notstarted_orders=
    SELECT * FROM orders
    WHERE client_id=$client_id
    AND start_date>$current_date
```

2. If there are no such orders the function is finished unsuccessfully.

3. If there are such orders Client chooses a specified order, with an order_id=*$order_id,* that exists in the cursor. If the Payment that has an attribute order_id=*$order_id* has a payment_status='unpaid' both the Payment and the Order are deleted.

DELETE FROM payment WHERE order_id = *$order_id*
DELETE FROM orders WHERE order_id=*$order_id*

4. If the payment that has an attribute order_id=*$order_id* has a payment_status =’paid’ the payment’s status is changed to ‘paid_but_deleted’ and then it’s attribute order_id is set to null and the order is deleted.

   UPDATE payment set payment_status='paid_but_deleted'
   WHERE order_id=*$order_id*

   UPDATE payment set order_id=NULL WHERE order_id=p_id
   DELETE FROM orders WHERE order_id=*$order_id*

7.5 Specific order details with additional information about payment

SELECT o.order_id, o.route_id, o.truck_plate, o.client_id, o.order_status, o.start_date, o.end_date, o.height, o.width, o.load_description p.price, p.payment_status
FROM orders o
JOIN payment p on p.order_id=o.order_id

4.2.5 Function 8.1 Notify users who have unpaid payments

1. This fuction is performed at an interval - once a day.

2. A cursor is created that scans all the payments that have an attribute payment_status set to ‘unpaid’

Cursor *$unpaid_payment*=
SELECT * FROM payment WHERE payment_status=’unpaid’

3. For each of cursors records it is necessary to check if the corresponding order (that has an attribute order_id = *$unpaid_payment*.order_id)meet the conditions: (start_date-$*current_date*)<5 and (start_date-$*current_date*)>0. If it does then a record is inserted in the table Notification. In a real system, an e-mail is sent.

INSERT INTO notification (payment_id)
VALUES (*$unpaid_payment*.payment_id)

4. If the corresponding order(that has an attribute order_id = *$unpaid_payment*.order_id) has an attribute start_date such meets this condition:
(start_date-*$current_date*<1) the status of the payment is changed to ‘failed_to_pay’.

UPDATE payment set payment_status='failed_to_pay'
WHERE payment_id=*$unpaid_payment*.payment_id

5. This function automatically deletes notifications if the payment that is in the table notification has had its status changed to 'paid','paid_but_deleted' or 'failed_to_pay' then such record is deleted from the table notification.(it is necessary due to a possibly huge number of records in the table and we do not need to handle history of notifications)

DELETE FROM notification
WHERE payment_id=*$unpaid_payment*.payment_id

4 Design of User Interface

5.1 Menu

1.New: (*Responsibility*: Client)
- a) place – action 2.1 New place
- b) route – action 5.1 New route
- c) order – action 7.1 New order

2. Profile (*Responsibility:* Client)
- a) My details - action 1.4 Client detail
- b) Edit profile – action 1.2 Client Update

3. Orders
- a) List of orders - action 7.4 List of orders (*Responsibility:* Client, Admin)
- b) Delete order – action 7.3 Delete Order (*Responsibility:* Client)
- c) Update order – action 7.2 Update Order (*Responsibility:* Admin)
- d) Specific order – action 7.5 Specific order details with additional information about payment (*Responsibility:* Client, Admin)

4. Payments (*Responsibility:* Client, Admin)
- a) all payments - action 4.4 List of client's payments
- b) sum to pay - action 4.5 Sum of client's unpaid payments

5. Places and routes (*Responsibility:* Client, Admin)
- a) list of places – action 2.3 List of Places
- b) list of routes – action 5.2 List of client's routes
- c) specific place – action 2.2 Place details
- d) specific route – action 5.2 Route details

6. Administration (*Responsibility:* Admin)
- a) Client management
  1. List of clients – action 1.3 List of clients
  2. Client detail – action 1.4 Client detail

b) Driver management

1. New driver – action 3.1 New Driver
2. Update driver – action 3.2 Update Driver
3. List of drivers – action 3.3 List of Drivers
4. List of driver routes –
   action 3.4 List of routes a Driver has completed in last 30 days

c) Payment management

1. New payment – action 4.1 New Payment
2. Update payment – action 4.2 Update Payment
3. Delete payment – action 4.3 Delete Payment
4. List of client's payments – action 4.4 List of client's payments
5. Unpaid payments - action 4.5 Sum of client's unpaid payments

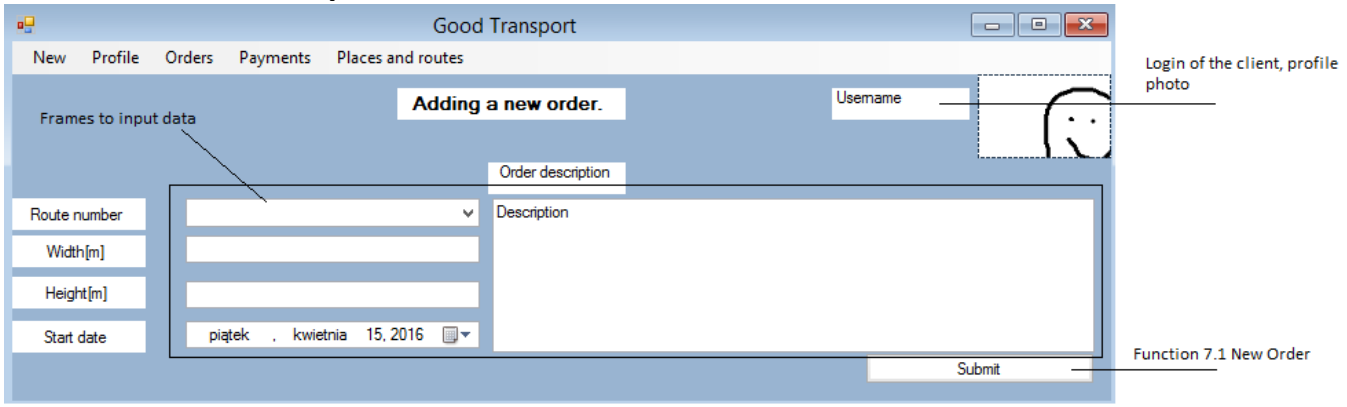
d) Truck management

1. New truck – action 6.1 New Truck
2. Update truck – action 6.2 Update Truck
3. List of trucks – action 6.3 List of Trucks

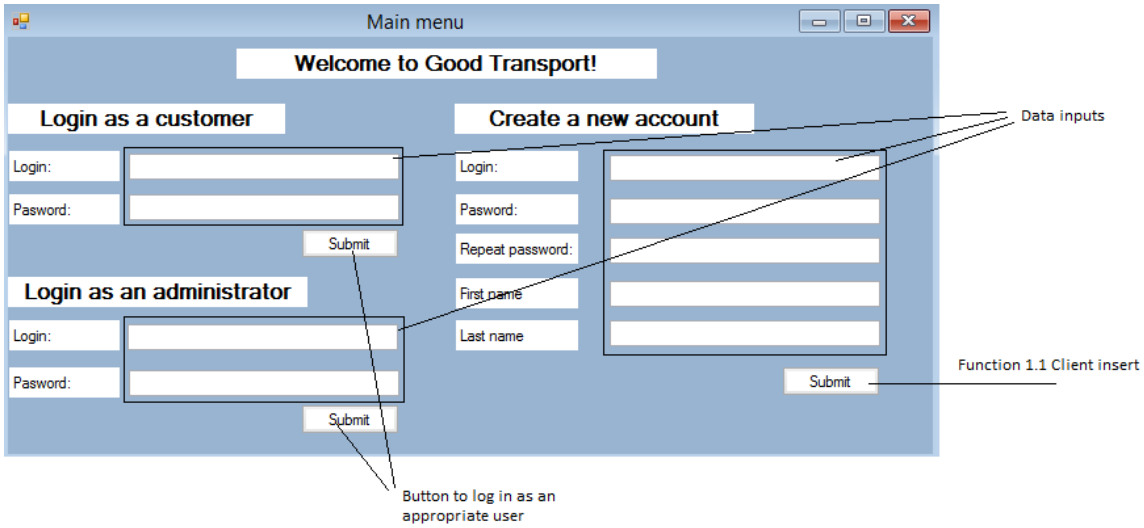
e) Order management

1. Update order – action 7.2 Update Order
2. List of orders – action 7.4 List of Orders
3. Order details – action 7.5 Specific order details with additional information about payment

5.2 New order input window



5.3 Main menu


Main menu
Welcome to Good Transport!
Login as a customer
Login:
Pasword:
Submit
Login as an administrator
Login:
Pasword:
Submit
Create a new account
Login:
Pasword:
Repeat password:
First name
Last name
Submit
Data inputs
Function 1.1 Client insert
Button to log in as an appropriate user