INFO0009-2 Databases 2024-2024

Project Part 1

IMPORTANT. THE ENGLISH VERSION OF THE PROJECT HAS BEEN TRANSLATED WITH GOOGLE TRANSLATE. PLEASE ADDRESS YOUR QUESTIONS TO THE PROFESSOR AND/OR ASSISTANT IF CERTAIN THINGS SEEM TO HAVE BEEN LOST IN TRANSLATION.

Rail Timetable Management System

This project is based on GTFS. Some things have been simplified. We also added some prerequisites. Feel free to check out the GTFS documentation for any examples, ideas, etc. additional information that might be useful for this project. However, it is essential to respect the project prerequisites.

The objective of this project is to design a database for a centralized, organized and efficient system for managing and accessing information on rail network operations. The ambition of this project is to capture timetable information around the world.

An agency refers to the entity that operates a rail public transport service. Agency information is critical to the model because it provides context for the rest of the data, such as routes, trips, and stops. They help users understand who is responsible for the transportation service and where to find more information about it. The information we want to store for an agency is: a unique identifier for the agency, the name of the agency, the URL of the agency's website, the time zone in which the agency operates, the languages used by the agency, the agency's phone number, the URL of the agency's pricing information page, and the agency's email address. As addresses differ between countries, we store the address of an agency's main office in an attribute. There is at least one main language. Languages are represented using ISO 639 two-letter language. All information except email address is required.

> *Let's take the example of the SNCB agency. Its unique identifier could be "SNCB". The name of the agency is "SNCB/NMBS", and its website is accessible via the URL "https://www.belgiantrain.be/". The agency operates in the "Europe/Brussels" time zone. It supports French ("fr"), Dutch ("ne") and German ("de"), which are the three official languages in Belgium. His phone number could be "003225282828". Information regarding fares is available at the URL "https://www.belgiantrain.be/fr/tickets-and-railcards/overview-products". SNCB does not have a general email address. Finally, their head office address is "Rue de France 80, 1070 Brussels, Belgium".*

A route defines the sequence of stops for a particular service. For a route, we store a unique identifier which is numeric. We also store a route name (e.g., "Eupen -- Ostend") as well as the type (e.g., "S" or "IC"). Routes are identified by their identifier, and by the combination of name and type. It is important to note that a route does not take direction into account. Indeed, "Eupen -- Ostend" represents the journeys of this route in both directions. We do not represent management at the route level; the direction will be stored at the route level (described later).

Stops represent the physical locations where trains pick up or drop off passengers. Stops are identified by a unique identifier. We also want to store their coordinates, consisting of a longitude and a latitude. We have two types of stops. A stop is either a station or a platform. A station has one or more platforms. For docks, we want to store their code (e.g. "1", "2b", "C", ...). For stations, we store their name.

Routes serve at least two stops. These *stops served* stop routes to their stops. We also store the sequence number for each *stop served*. We can calculate the sequence number for the reverse direction. For example, for "Eupen -- Ostend", the first, second and last (11th) stop are: Eupen, Welkenraedt and Ostend. The sequence numbers are 1, 2 and 11 respectively. In the reverse direction, the sequence numbers are 11, 10 and 1.

A trip can be thought of as an instance of a route for a particular service. A unique, alphanumeric identifier is assigned to them. Paths have a direction stored as an attribute. A value of 0 means it goes in one direction and a

value of 1 means it goes in the opposite direction. For trips, we need to store information for each of the stops served on that route. We need to record the theoretical arrival and departure times. There is no arrival time for the stop where the journey begins. There is no departure time for the stop where the ride ends. We must ensure that these arrival and departure times respect the sequence of stops served. In our application we also want to provide the sequence number of the stops served at the trip level. We also need to ensure that trips provide information for all served stops on their route.

A service defines route offerings for certain days of the week, usually for repeating schedules. A service is identified by a service identifier which can be descriptive using alphanumeric characters (for example, “WORK WEEK”, “WEEKEND”, “HOLIDAY”, ...). A service has a start and end date. All services *can* (!) be linked to one or more names of days of the week. Services are also linked to exceptions to regular service models. It is used for specific dates that deviate from the normal calendar, such as holidays or special events. Exceptions have an exception code: a value of 1 means that a service was added for the specified date, and a value of 2 means that the service was removed for the specified date. Exceptions are identified by their service and date.

> *For example, the WORK WEEK service describes Monday to Friday service and is valid from 01/01/2024 to 12/31/2024. The HOLIDAY service, also valid from 01/01/2024 to 12/31/2024, is not tied to any particular day of the week and is used to represent services on these special days. In 2024, Christmas fell on a Wednesday, 12/25/2024 was an exception for WEEKDAY where the service was removed (value 2). 12/25/2024 was also an exception for HOLIDAY because the service was added (value 1).*

Mission

You are asked to create an entity-relationship model in the form of a diagram describing the universe of discourse for which we want to design and develop a database. Don't forget to provide the keys and cardinality constraints of the entity sets and model relationships. If necessary, specify additional integrity constraints, as well as weak relationships and entities. If any aspects are unclear or ambiguous, ask the question on eCampus or on the course, and/or note the assumptions in your deliverable. Convert the entity-relationship model to a relational model. Make sure the relationships in this model are in BCNF (you will need to justify this), and propose a decomposition if necessary.

Submission

The first part of the project is carried out in teams of three students. The work must be submitted via eCampus before 2025-03-14 at 11:59 p.m. in a PDF file. The PDF file contains

- A title page (title, names, courses, etc.)
- The entity-relationship model (please use a vector image and make sure your diagram and its cardinalities are readable on screen) and, if necessary, the list of justifications, assumptions, explanations, etc. ;
- The domains of each attribute;
- The keys to entities and relationships;
- Conversion to the relational model;
- Additional integrity constraints to the diagram (if applicable);
- Analysis of normal forms.

If you have any questions, ask them in the Discussions section of eCampus or email me at c.debruyne@uliege.be if your question contains part of your solution. Dia is a diagramming tool available on Windows, Mac and Linux. Another option is draw.io, a website that can store your work on Google Drive and Dropbox.