D2.2 Database structure

Deliverable 2.2

Domino

Grant:	**783206**
Call:	**H2020-SESAR-2016-2**
Topic:	**SESAR-ER3-06-2016 ATM Operations, Architecture, Performance and Validation**
Consortium coordinator:	**University of Westminster**
Edition date:	**27 September 2019**
Edition:	**01.00.00**

Founding Members
EUROPEAN UNION
EUROCONTROL
SESAR JOINT UNDERTAKING

Authoring & Approval

Authors of the document

Name/Beneficiary	Position/Title	Date
Luis Delgado / University of Westminster	Project member	25 September 2019
Gérald Gurtner / University of Westminster	Project member	25 September 2019
Tanja Bolic / Università degli studi di Trieste	Project member	25 July 2019
Paola Bassi / Università degli studi di Trieste	Project member	30 July 2019

Reviewers internal to the project

Name/Beneficiary	Position/Title	Date
Andrew Cook / University of Westminster	Project coordinator	26 September 2019

Approved for submission to the SJU By — Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date
Andrew Cook	Project coordinator	27 September 2019

Rejected By - Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date

Document History

Edition	Date	Status	Author	Justification
01.00.00	27 Sept. 2019	Release	Domino Consortium	New document for review by the SJU

Domino

NOVEL TOOLS TO EVALUATE ATM SYSTEMS COUPLING UNDER FUTURE DEPLOYMENT SCENARIOS

This deliverable is part of a project that has received funding from the SESAR Joint Undertaking under grant agreement No 783206 under European Union's Horizon 2020 research and innovation programme.

Abstract

This is a technical deliverable describing the database used in Domino. The structure of the database along with information on the data sources used are included. This database has been used to store the input and outputs of the executions of the investigative case studies reported in D5.2 – Investigative case studies results.

The deliverable includes a diagram of the relational database and a description of the different tables used with information on the different fields that define these tables. Information on the pre-computation of data to create the required input for the model is also included.

Current shortcomings of the database are identified and potential solutions highlighted.

Table of Contents

List of figures

List of tables

Executive summary

Domino data are stored in a MySQL server. Access to the database is secure. This is achieved by locating the database server on a virtual machine inside the University of Westminster's cluster, with password-protected access, encrypted with an SSL (secure sockets layer) certificate.

Different data are structured in a relational database. Dedicated output tables are created to store the results of the executions in the model.

Data sources needed to execute Domino are grouped into four categories: traffic and delay; airspace environment; passengers; and, other. Some pre-computations are required: in particular, calculating the probabilities of ATFM delays and pre-computing flight plan alternatives.

The database structure, which is deployed in the MySQL server, is reported in this deliverable with detailed information on the 53 tables that are directly used as input and output by the Domino model. The information provided is sufficient to re-generate the database structure in a new server instance.

Some performance issues when managing the large raw output of the model have been identified. The consortium will consider the storage of pre-computed results and other technological solutions, such as NoSQL databases, to expedite the execution and analysis of scenarios in the model.

1 Introduction

1.1 Database structure

Domino needs to provide a documented and stable platform to manage the data that need to be secured and accessible. Data are crucial to generate the input into the model but also to store the raw results that are used in the analysis of the scenarios. Traceability is paramount and versioning is also a requirement. All this is achieved by using a relational database implemented in MySQL.

The database used in Domino has been constructed leveraging on previous experience of the consortium developed in the Vista project. However, the dedicated characteristics of Domino have been considered to tailor the technical solution presented in this deliverable.

This deliverable is a technical document which aims at presenting the different tables that are defined in the '*domino_environment'* schema. This schema contains all the input data used by the model and all the output tables generated by the model. Note that Domino's database also contains other schemas which are used as static input but that are the mere translation of data from other sources (e.g., BADA, DDR) and hence not reported in this deliverable.

1.2 Structure and contents of this deliverable

Section 2 describes the database infrastructure used in Domino. Section 3 summarises the data sources used in Domino. Section 4 presents highlights of some of the pre-computation of data considered as part of the input of Domino.

The core of the deliverable with the description of the database structure of '*domino_environment'* are presented in Section 5.

Finally, next steps are gathered in Section 6 with particular focus on shortcomings identified so far in the database, and its planned evolution.

2 Database infrastructure

All data used in the Domino project are centralised in a single, secure database hosted at the University of Westminster. The information on the database infrastructure was reported in D2.1 [1]. Here a summary is presented.

2.1 Database access

Access to the database is secure. This is achieved by locating the database server on a virtual machine inside the University of Westminster's cluster, with password-protected access, encrypted with an SSL certificate.

Partners have permissions to use the database resources for testing and production. These permissions are managed by the University of Westminster and limited to the partners considering their data requirements and subject to having adequate licencing agreements. The control of data access ensures that possible data corruption is minimised. For instance, UNITS has full writing and reading access to the data, since they are managing the content of the database in Domino, while other partners involved in the modelling have read-only access, or can create new tables but not erase any.

2.2 Database structure

The database server used in Domino is MySQL. MySQL is an open source server for relational database, which is widely used. It supports standard SQL, it is well documented, reliable, and well suited for mid-range databases.

Domino uses the database for three purposes, to:

- have standard input data with easy access;
- store pre-computed information to be used by the model;
- store the results of the model(s) in an efficient way.

The structure of the database considers the following model requirements:

- **reproducibility**: getting the same output from the same input with the same code.
- **reliability**: making sure that the input data has not changed between two runs of the model.
- **consistency**: making sure that the input is self-consistent.
- **traceability**: making sure that the output data can be linked unambiguously to a given input dataset.

Building on data management experience from past projects, Domino uses three different types of schemas, for:

- primary data, which should never be modified. This includes, for instance, DDR2 data and other sourced data (see previous section);
- secondary data, which are prepared 'off-line' by pre-processing parts of the models. These data change with the maturity of the models, and should be versioned. This may adapt the primary data into the structure required to be used as input for the model;
- output data, which are the raw results of the models. Once again, these data change during the project, and should be versioned.

By versioning the secondary data and output data, the project ensures the traceability of the results. While the primary data are in their own schemas in the database, all the direct input and output of the model are centralised in the same schema: '*domino_environment'.*

During the execution of the investigative case studies, some limitations of the database performance were identified:

First, the writing time can be a limiting factor on the computation, as concurrent executions of the model save the results on the same tables. In some instances, the bottleneck on the execution time of the simulations is this dumping of the results phase. This is particularly critical for the passenger results as Domino allows us to model individual passenger itineraries, the amount of output data can be very high (e.g. the *output_pax* table, which stores the passenger output, is over 100GB).

Secondly, as the output generated by the model grows by running several iterations on the same scenario, the use of SQL capabilities such as filtering (WHERE) and joining tables (JOIN) became very time consuming. This is particularly relevant for the individual passenger itineraries. As previously mentioned, the output table of the passengers' results is over 100GB, which make it not suitable for joining tables unless previously heavily filtered, rendering some of these performance issues. These limitations impact the possibility of tracing inputs and outputs of some results (e.g., linking scheduled passenger itineraries with their actual execution became too time consuming).

These two issues have been partially addressed by creating a dedicated database server to store the results of Domino, by carefully selecting which fields to store as output of the model including some traceability parameters to minimise the need of joining tables, and by tuning some of the server parameters. For the final version of the model, we will consider the use of a NoSQL database to store the output of the model. These databases are specifically designed to allow large concurrent writing of results, so it could solve some of the limitations identified. On the other hand, more fields will need to be stored in order to retrieve all the required information for the analysis of the results from one single registry minimising the linking of initial (scheduled) data with the actual executed one.

3 Data sources

As reported in D2.1 – Data management and resources [1], there are different data categories that have been identified as required for the model:

1. traffic and delay;
2. airspace environment;
3. passengers;
4. other.

Table 1 summarises the data sources used in Domino

Table 1. Summary of data sources

Category	Dataset	Used for
Traffic and delay	DDR2 – AIRAC 1313-1413, 1702, 1709	- ATFM regulations analysis (probability and intensity) - Routes alternatives estimation
	FlightGlobal schedules 2014	- Demand
	CODA summary delay data	- Calibration - Delay due to reasons not explicitly modelled
	CODA – Taxi times - IATA Summer Season 2010	- Taxi times estimation
	BADA performances	- Performance computation - Flight plan estimation
Airspace environment	DDR2 – AIRAC 1313-1413, 1702, 1709	- Airport and airspace capacities
Passengers	Previous itineraries 2010 and 2014	- Passengers itineraries linked to flights
Other	Cost of delay	- Estimation of cost of delay
	CRCO unit rates	- Estimation of flight plan parameters
	Airline alliances	- Required for passenger itineraries generation and re-routing
	Airport curfews	- For cancellation of flights and airline decision making processes

4 Data pre-processing

Some data used in Domino has been computed from the analysis of historical data (DDR2) or pre-computed. In some cases, these computations are based on previous projects. This section briefly highlights some of these data preparation activities.

4.1 ATFM delay

The probability of a flight experiencing ATFM delay has been computed analysing DDR2 data from the period AIRAC1313-1413, 1702 and 1709. The probability of being affected by a regulation has been estimated for:

- all flights in the historical DDR2 dataset,
- all flights excluding the ones affected by regulations at airports,
- the flights affected by regulations that are issued due to weather,
- the flights affected by regulations that are not issued due to weather (with and without the exclusion of the ones at airports).

This allows us to differentiate the reason of the ATFM delay experienced by a flight: either from an explicit ATFM regulation at the arrival airport, due to weather en-route or non-weather. These differences are needed when considering the liability for passenger compensation (Regulation 261 [2]).

Cumulative distributions of probabilities of having a given amount of delay assigned have been computed from the historical data.

Finally, for regulations that are explicit at an airport, for each day of the period AIRAC1313-1413, the number of regulations issued at airports has been computed. These days are ranked, and the Domino model selects one of those days randomly as the reference to model explicit ATFM regulations. The selection is based on a minimum and maximum percentile in order to identify nominal days in the baseline scenarios, and days with more regulations in the stressed scenarios.

4.2 Flight Plans

Flight plans have been computed based on the different routes available between origins and destinations. These routes have been clustered as in Vista [3] considering the entry and exit point of different ANSPs. This allows us to estimate the en-route charges while simplifying the sampling of alternatives.

We allow the model to run using only this pool of routes between the different origins and destinations. In that case, the AOC agent computes the trajectories (4D profiles) and flight plans (including costs (CRCO, fuel, time)) during the dispatching process. However, this is computationally very expensive. For this reason, the model allows us to use pre-computed the trajectories and/or flight plans. One of the drawbacks is that some of the decisions that could be performed by the AOC dynamically will be already pre-calculated (e.g., which nominal speed to select for a flight plan option).

The model allows wind to be drawn from distributions that have been pre-computed from the analysis of historical origin-destination pairs. However, in the current executions of the model, the average wind is used for replicability and simplicity.

See [3] for more details on the flight plan generation processes as these are detailed as part of the Vita project activities.

4.2.1 Further clustering of routes possibilities

Besides reusing the clustering of routes developed in Vista [3], Domino considers a new clustering based on AIRAC 1702 and 1709. The two AIRACs are selected to account for possible seasonal differences in the trajectory choices. This new clustering could be used for the final version of Domino.

The new clustering has been performed on 29 460 OD pairs, resulting in 1 284 560 flights being eligible for route clustering. These flights were further filtered to exclude:

- military flights;
- flights with origin or destination airports being “ZZZZ” or “AFIL”.

For the remaining flights, the m1 trajectories (submitted flight plans) were transformed into a geometric format, to speed up the clustering algorithm. For each OD pair, the Hausdorff distance[1] was calculated between all the trajectories belonging to the pair. Clustering was performed using the DBSCAN algorithm. DBSCAN clusters elements that are closely packed together, i.e., elements in a ε-neighbourhood and surrounded by a minimum number of neighbours. It requires two parameters: the maximum radius of the neighbourhood ε and the minimum number of elements m required for a cluster. It is important to note that DBSCAN does not require to be initialised with the number of clusters to create, but it autonomously finds the number of clusters suitable for the problem. This property fits our scenario since we cannot estimate the correct number of typical trajectories *a priori*. We set the maximum radius of the neighbourhood ε = 0.3 (which corresponds to 30km) and the minimum number of elements m = 1 as parameters of the DBSCAN algorithm. Clustering was performed on a 64-bit Intel(R) Xeon(R) E5520 @ 2.27GHz quad core CPU computer with 16GB of RAM memory and Debian 8.0 operating system. The computation time was appx. 8 hours.

The clustering decreases the number of viable routes between the OD pairs - identifying for each OD pair the set of trajectories, the total flight distance and the entry and exit points in different airspace elements. From the results it can be seen that for a good portion of OD pairs, chosen trajectories are

[1] The Hausdorff distance between a set of trajectories for a given one, is the maximum distance of this trajectory the nearest one.

usually the same: 29% have one cluster, and 18% have two clusters. Most of the OD pairs with only one cluster are the ones with short flights. Moreover, the number of OD pairs decreases with the increase in the number of clusters.

These routes are available in the database and could be used as Domino input or as the basis to compute trajectories and flight plans. They are more detailed that the ones identified in Vista, increasing the alternatives available for the AOC for each flight.

5 Database structure

Table 2. Domino's *domino_environment* tables summary

Tables used in	Used in execution Domino	Used in preparing pre-computed data	As reference

Input	• **Scenario**
	  ○ scenario
	  ○ eaman_definition
	• **Demand**
	  ○ flight_schedule
	  ○ flight_subset
	  ○ pax_itineraries
	• **Aircraft performance**
	  ○ ac_eq_badacomputed_static
	  ○ ac_mtow_static
	• **Airport related**
	  ○ airport_curfew
	  ○ airport_info_static
	  ○ mtt_static
	  ○ taxi_in_static
	  ○ taxi_out_static
	  ○ taxi_out_wake_static
	• **Costs**
	  ○ non_pax_delay_fit_static
	  ○ non_pax_delay_static
	  ○ passenger_compensation_static
	  ○ soft_cost_delay_static
	  ○ duty_of_care_static
	• **Delay/uncertainty**
	  ○ delay_parameters
	  ○ extra_cruise_dci_static
		• **En-route charges**
		  ○ CRCO_charges_static
		  ○ CRCO_fix_static
		  ○ CRCO _overfly_static
		  ○ CRCO _vat_static
		  ○ CRCO _weight_**static**
		• **Routes/Trajectories***
		  ○ airspace_static
		  ○ route_pool
		  ○ route_pool_has_airspace_static
		  ○ route_pool_o_d_generated
		  ○ route_pool_o_d_generated_has_airspace_static
		  ○ route_pool_static
		  ○ route_pool_static_has_airspace_static
		  ○ trajectory_pool
		  ○ trajectory_segment
		  ○ fp_pool
		  ○ fp_pool_point
			• flihgt_schedule_excluded

	○ flight_uncertainties_static ○ iedf_atfm_static ○ iedf_wind_static ○ prob_atfm ○ regulation_at_airport_days_static ○ regulation_at_airport_owevestatic • **Flight plans/Routes** ○ fp_pool_m ○ fp_pool_point_m • **Other** ○ airline_static
Output	• output_dci • output_eaman • output_flights • output_pax • output_RNG • output_sim_general • output_swaps

[*] The Domino model is able to work with flight plans with speeds and winds precomputed (fp_pool_m), with flight plans without the speeds, CRCO charges and winds (fp_pool) or directly using the routes and generating the trajectory as part of the simulation (route_pool). However, the computational cost increases as more activities are performed within the simulator and no precomputed.

Table 2 presents the 53 tables that are used in *domino_environment*:

- 29 as direct input for the model;
- 16 as part of the pre-computation of data;
- 1 kept as reference; and,
- 7 further tables to store the output of the model.

The diagram of the database is presented in Section 3.1 and a description of each of the tables with its different fields in Section 3.2.

5.1 Database structure diagrams

Figure 1 presents the full database structure. Due to the small size of the image, Figure 2, Figure 3, Figure 4 and Figure 5 show the page-size sections of the diagram, respectively top-left, top-right, bottom-left and bottom-right parts of the full diagram. Note that only the first fields of each table are shown in the diagram, see Section 5.2 for a full description of the tables.

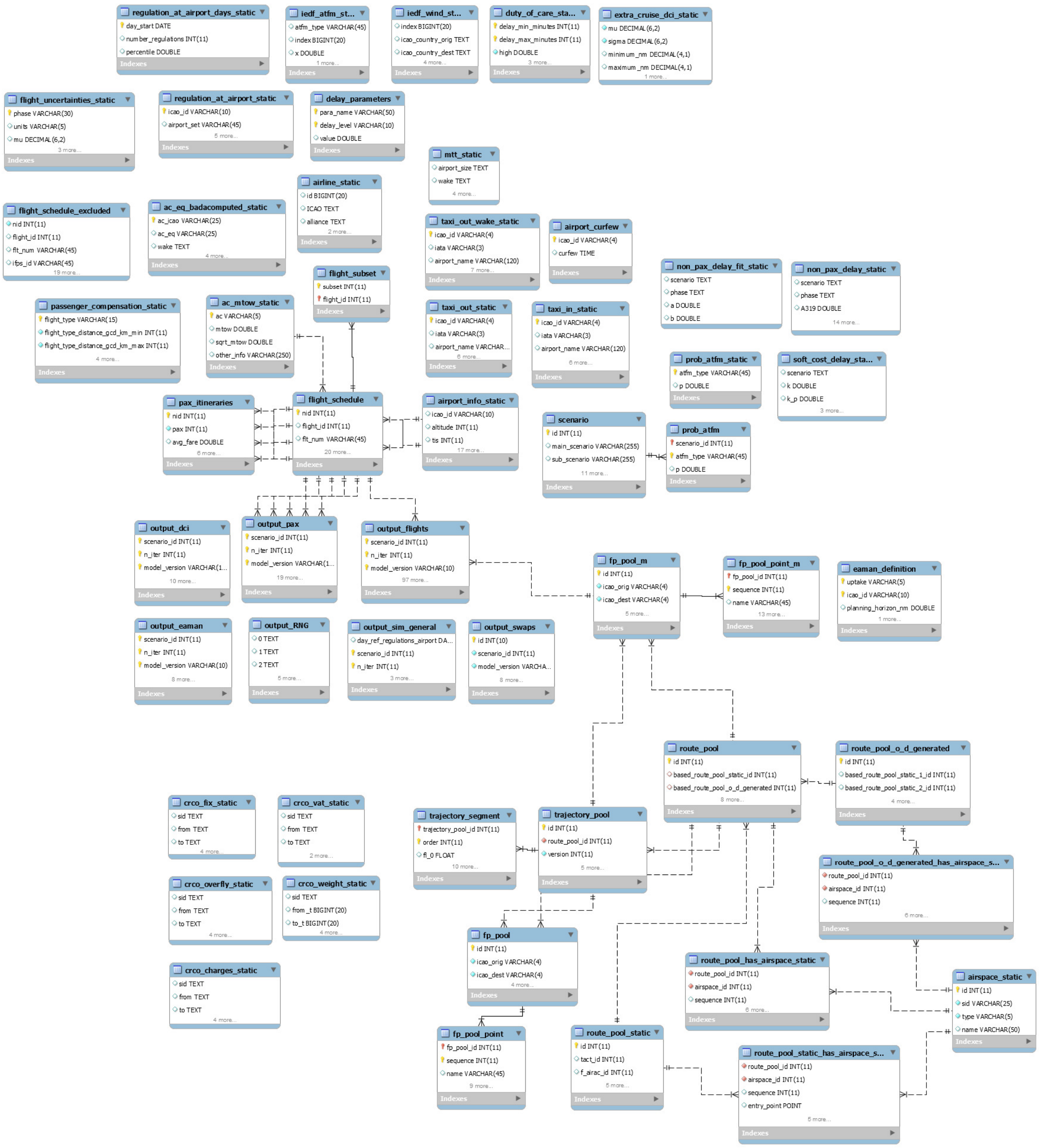

Figure 1. Full database diagram

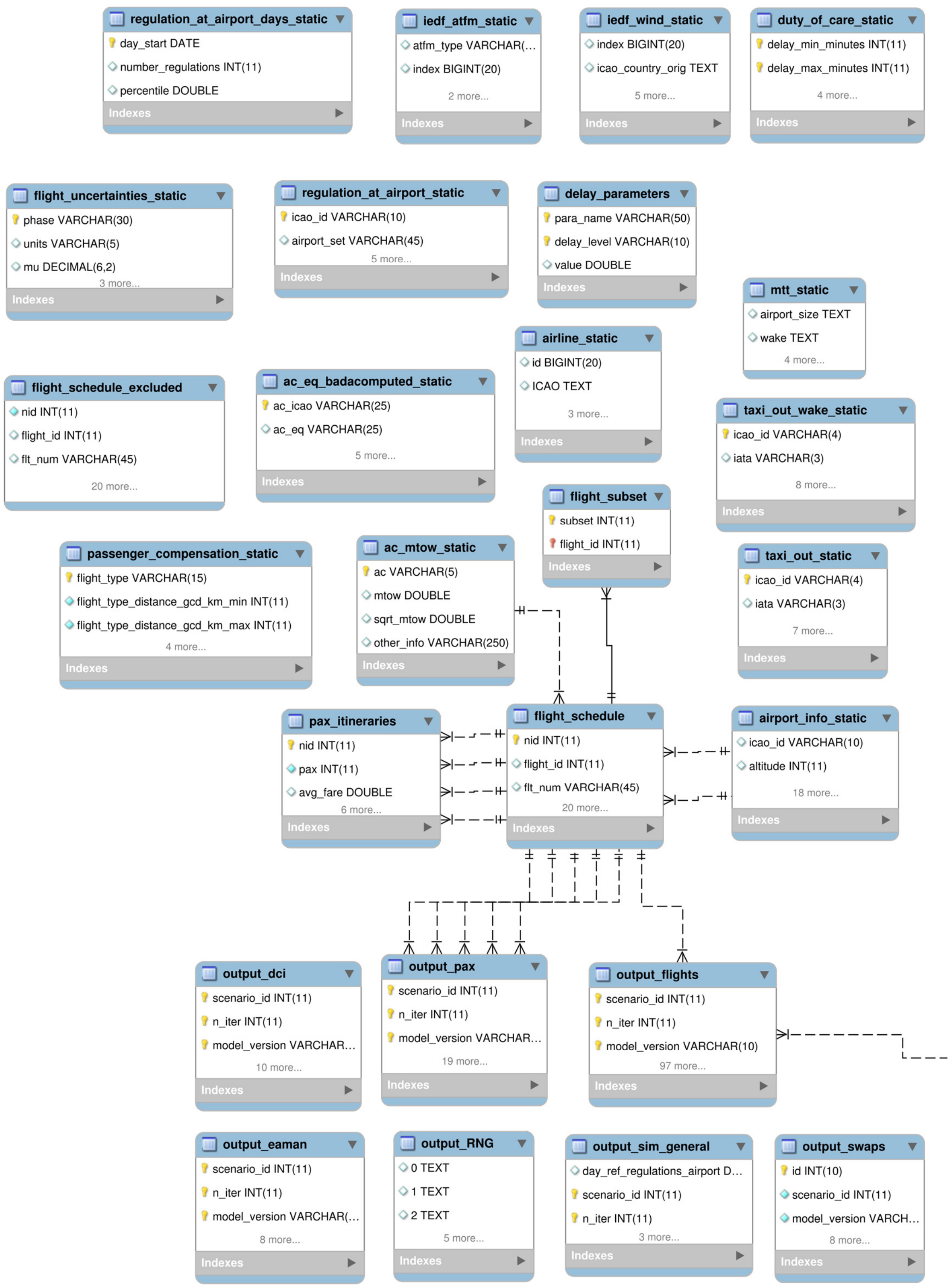

Figure 2. Part 1/4 database diagram (top-left part)

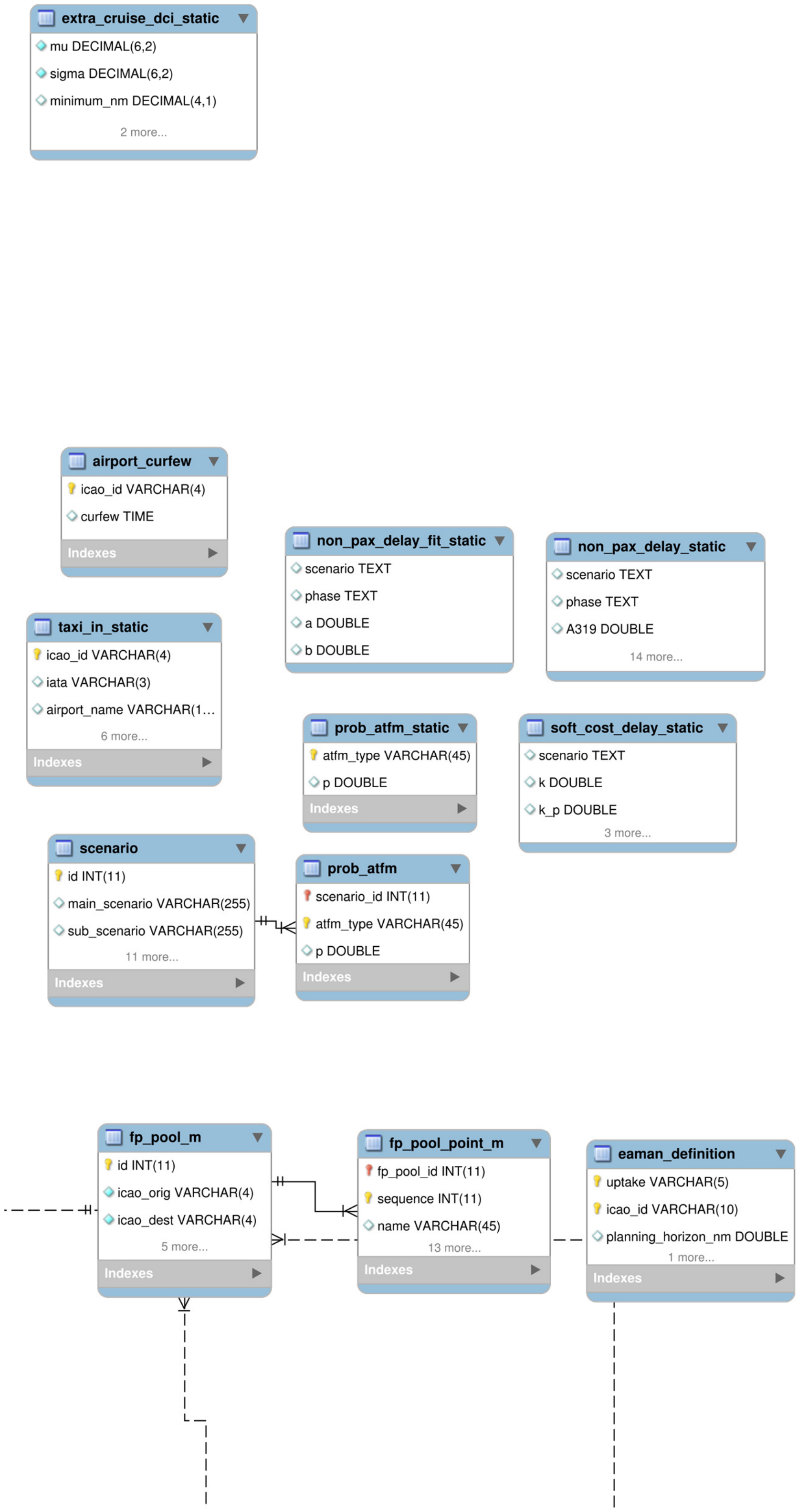

Figure 3. Part 2/4 database diagram (top-right part)

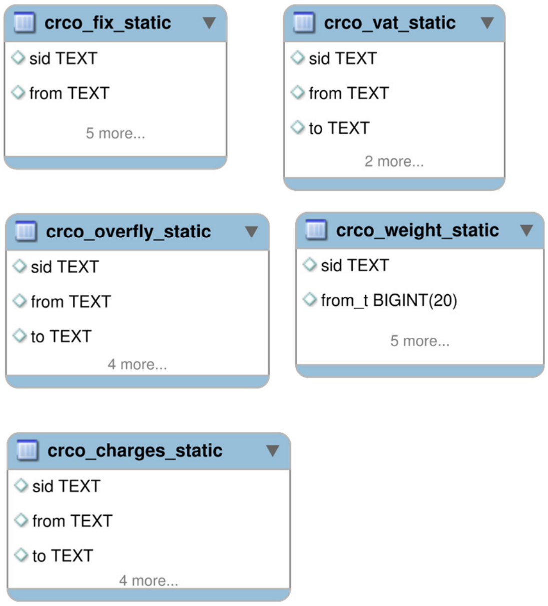

Figure 4. Part 3/4 database diagram (bottom-left part)

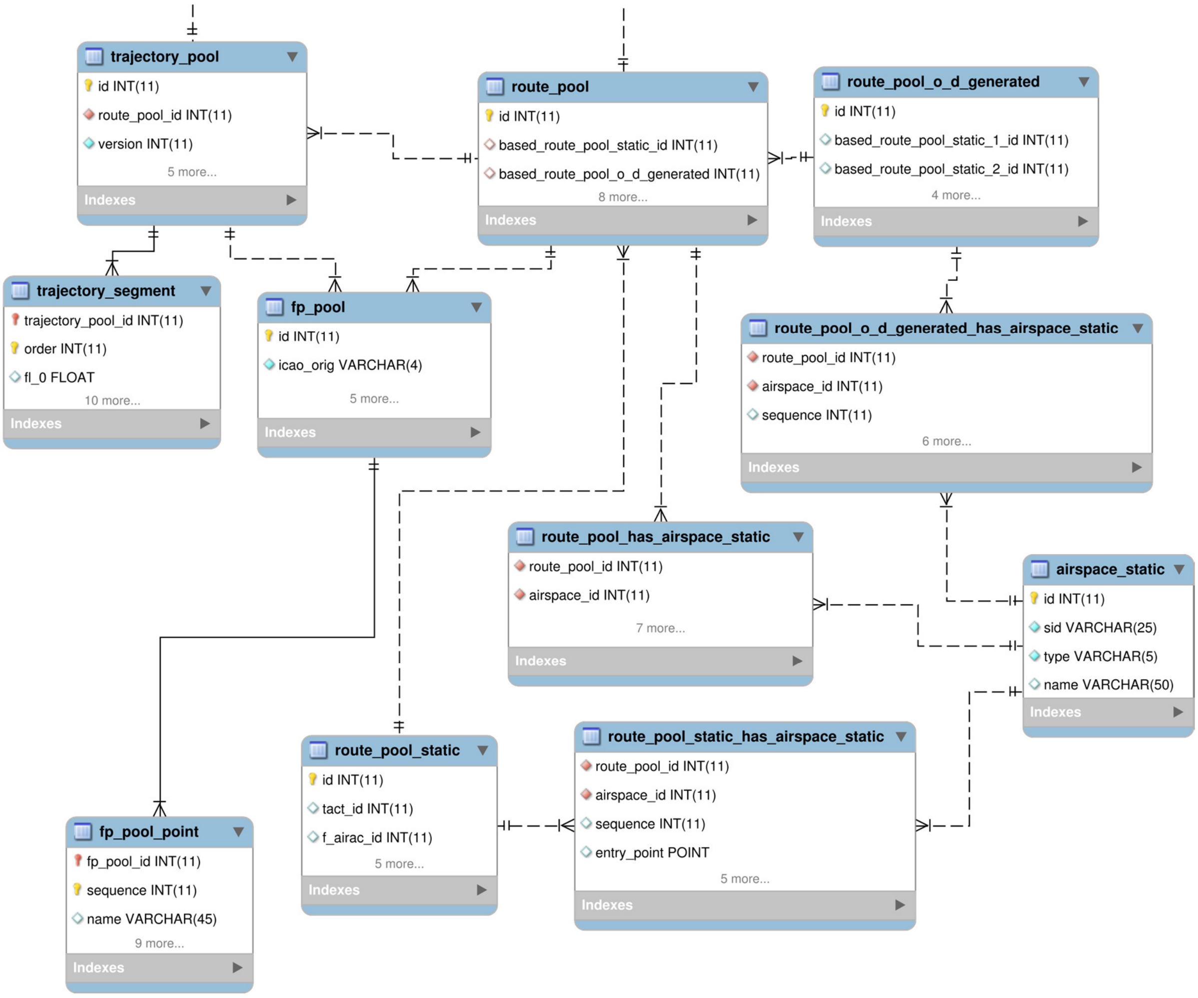

Figure 5. Part 4/4 database diagram (bottom-right part)

5.2 Database tables

5.2.1 Input data for the model

Table 3. Ac_eq_badacomputed_static

Rationale: Table to relate aircraft type with BADA model to be used to compute performances. Each ICAO aircraft type has an equivalent aircraft performance model to be used (it could be the same) and an associated BADA model to be used.

Field	Type	Primary key	Foreign Key	Rationale	Other info
ac_icao	varchar(25)	*		Aircraft ICAO code (e.g., A124)	
ac_eq	varchar(25)			Equivalent aircraft type used in performance (e.g., H_JET, E170)	
wake	text			Wake turbulence (i.e, H, M, J, L)	
engine_type	text			Engine type of aircraft (i.e., JET, PISTON, TURBOPROP)	
bada_code_ac_model	varchar(25)			Code from BADA3 or BADA4 used for the performance (e.g., A340-642)	
bada_version	double			Bada version used (i.e, 3, 4)	
type	text			Source of link between aircraft and bada_code_ac_model (i.e., historic, manually_added)	

Table 4. Flight_subset

Rationale: Table to link subset of flights with their schedules. Different flights are used in different scenarios, mainly for testing purposes (the final scenarios have all the flights). This table allows to select a bunch of flights easily with a single subset id.

Field	Type	Primary key	Foreign Key	Rationale	Other info
subset	int	*		subset of flight schedules id	
flight_id	int	*	flight_schedule.nid	id of the flight schedule that is part of the subset	

Table 5. Airline static

Rationale: Information on airline type, hubs and alliances. The alliance information is used to build alliance objects within the model, important for the rebooking strategy of the airline. The information on the hubs is not used. The airline type field is used for the post-analysis only.

Field	Type	Primary key	Foreign Key	Rationale	Other info
id	bigint(20)			Incremental index	

ICAO	text			Airline ICAO code (e.g.,RYR)	
alliance	text			Alliance code	
hubs	text			Hub airport used by the airline as hub	
AO_type	text			CHT (charter), FSC (legacy carriers), LCC (low cost), REG (regional)	Low = all LCC flights High = FSC flights into a hub; REG flights into a hub Base = all other flights

Table 6. Airport_curfew

Rationale: For airports with curfew the time when the curfew applies. This is only used in the strategy decisions of the airline to compute their cost function. The curfew is not yet enforced on the flights themselves in the current model version.

Field	Type	Primary key	Foreign Key	Rationale	Other info
icao_id	varchar(4)	*		Airport ICAO code (e.g., EDDM)	
curfew				Hours of night curfew at the airport in local time	

Table 7. Airport_info_static

Rationale: Static information about the airports. This information collects different 'static' information about the airport. In particular, some statistics on the taxi-in, taxi-out, minimum connecting time etc. have been pre-computed and added to the database to be used in the model.

Field	Type	Primary key	Foreign Key	Rationale	Other info
icao_id	varchar(10)			Airport ICAO code (e.g., EDDM)	
altitude	int(11)			Airport altitude in ft	
tis	int(11)				from DDR2
trs	int(11)				from DDR2
taxi_time	int(11)				
coords	point			Airport coordinates	WKT format
time_zone	int(11)				
mean_taxi_out	double			Average taxi-out time at the airport in minutes	
std_taxi_out	double			Standard deviation of taxi-out time at the airport in minutes	
mean_taxi_in	double			Average taxi-in time at the	

				airport in minutes	
std_taxi_in	double			Standard deviation of taxi-in time at the airport in minutes	
MCT_standard	double			Standard minimum connecting time (min)	
MCT_domestic	double			Domestic minimum connecting time (min)	
MCT_international	double			International minimum connecting time (min)	
ECAC	tinyint(1)			Boolean to indicate if airport is part of ECAC	
atfm_area	tinyint(1)			Boolean to indicate if flights departing from airport could be affected by ATFM	
nas	varchar(2)			ANSP code from where airport is located	
declared_capacity	double			Airport declared capacity	
size	varchar(45)			Size of airport (small, medium, large or blank)	
better_mean_taxi_out	float			Better estimation for the taxi out times (min)	Not used in simulation

Table 8. Airsapce_static

Rationale: ANSP airspaces linking their id to their name. This table is used to retrieve the name of the ANSP, whereas the model uses an id to identify it.

Field	Type	Primary key	Foreign Key	Rationale	Other info
id	int(11)	*		Incremental index	
sid	varchar(25)			String id of the airspace	e.g., LP
type	varchar(5)			Type 'NAS' for all airspaces	
name	varchar(50)			Airspace name	e.g., PORTUGAL

Table 9. CRCO_charges_static

Rationale: CRCO charges information used to generate routes options, not in the Domino execution *per se*, but for the pre-computation of flight plan alternatives.

Field	Type	Primary key	Foreign Key	Rationale	Other info
sid	text			NAS ICAO code (e.g.,EG)	
from	text			Start date of validity of the unit rate	
to	text			End date of validity of the	

				unit rate	
unit_rate	bigint(20)			Unit rate associated with NAS (Euro)	
ex_rate	double			Exchange rate	
ex_unit	text			Currency associated with NAS	
name	text			NAS name	

Table 10. CRCO_fix_static

Rationale: CRCO charges information used to generate routes options, not in Domino execution. For NAS which charge a fix cost to the use of their airspace (i.e., Shanwick oceanic, Crossing Iceland).

Field	Type	Primary key	Foreign Key	Rationale	Other info
sid	text			NAS ICAO code (e.g.,EG)	
from	text			Start date of validity of the unit rate	
to	text			End date of validity of the unit rate	
unit_rate	bigint(20)			Unit rate associated with NAS	
ex_rate	double			Exchange rate	
ex_unit	text			Currency associated with NAS	
name	text			NAS name	

Table 11. CRCO_overfly_static

Rationale: CRCO charges information used to generate routes options, not in Domino execution. For NAS which charge for overfly cost to the use of their airspace (i.e., Algeria, Iceland).

Field	Type	Primary key	Foreign Key	Rationale	Other info
sid	text			NAS ICAO code (e.g.,EG)	
from	text			Start date of validity of the unit rate	
to	text			End date of validity of the unit rate	
unit_rate	bigint(20)			Unit rate associated with NAS	
ex_rate	double				
ex_unit	text			Currency associated with NAS	
name	text			NAS name	

Table 12. CRCO_VAT_static

Rationale: CRCO charges information used to generate routes options, not in Domino execution. VAT per NAS on CRCO.

Field	Type	Primary	Foreign	Rationale	Other info

		key	Key		
sid	text			NAS ICAO code (e.g.,EG)	
from	text			Start date of validity of the unit rate	
to	text			End date of validity of the unit rate	
vat	double			VAT - value added tax	
name	text			NAS name	

Table 13. CRCO_weight_static

Rationale: CRCO charges information used to generate routes options, not in Domino execution. For NAS which charge based on weight (i.e., Russia, Tunisia).

Field	Type	Primary key	Foreign Key	Rationale	Other info
sid	text			NAS ICAO code (e.g.,EG)	
from_t	bigint(20)				
to_t	bigint(20)				
unit_rate	bigint(20)			Unit rate associated with NAS	
ex_rate	double				
ex_unit	text			Currency associated with NAS	
name	text			NAS name	

Table 14. Delay_parameters

Rationale: Table to store parameters linked with delay management in the simulation for different levels of delay. This table is used to set some parameter values depending on the scenario, and also to calibrate the model by adjusting some of the parameters.

Field	Type	Primary key	Foreign Key	Rationale	Other info
para_name	varchar(50)	*		Name of the paramete r	capacity_modifier to reduce capcity at airports extra_climb_tweak to adjust climb phase lambda_tat to adjust turaround times non_ATFM_delay_lambda to adjust delay due to non-ATFM reasons perc_day_max percentile maximum used to sample days to select day for ATFM regulations at airport perc_day_min percentile minimum used to sample days to select day for ATFM regulations at airport taxi_time_modifer to adjust taxi times

delay_level	varchar(10)	*		Delay level in simulatio n (D, H)	
value	double				

Table 15. Duty_of_care_static

Rationale: Parameters for the duty of care of passengers. This table is used to build the cost function of the airline by setting the different time thresholds and level of care for passengers.

Field	Type	Primary key	Foreign Key	Rationale	Other info
delay_min_minutes	int(11)	*		from which minute of delay the care start	
delay_max_minutes	int(11)	*		up to which minute of delay the care goes	
high	double			Value for high case of care per pax	
base	double			Value for baseline case of care per pax	
low	double			Value for low case of care per pax	
Uptake	double			Percentage of passengers claiming the care	

Table 16. Eaman _definition

Rationale: Definition of E-AMAN scope for different cases. This table is used to fix the planning and execution horizon of the AMAN. The extent of these horizons depend on the scenario simulated.

Field	Type	Primary key	Foreign Key	Rationale	Other info
uptake	varchar(5)	*		Level of uptake (D: default)	
icao_id	varchar(10)	*		Airport ICAO code	
planning_horizon_nm	double			Planning horizon in NM aroudn the airport	
execution_horizon_nm	double			Execution horizon in NM aroudn the airport	

Table 17. Extra_cruise_dci_static

Rationale: Information on the distribution used to extend the cruise, if DCI is applied.

Field	Type	Primary key	Foreign Key	Rationale	Other info
mu	decimal(6,2)			μ of Normal distribution used to model extra cruise	
sigma	decimal(6,2)			σ of Normal distribution used to model extra cruise	
minimum_nm	decimal(4,1)			Minimum number of NM that the cruise is extended	
minimum_nm	decimal(4,1)			Maximum number of NM that the cruise is extended	
source	varchar(45)			Source for this distribution for traceability	

Table 18. Flight_schedule

Rationale: Schedules used in Domino (not excluded flights). Includes the origin, destination, schedule off-block time and on-block time. Used to build the Flight object of the model and initialise it.

Field	Type	Primary key	Foreign Key	Rationale	Other info
nid	int(11)	*			
flight_id	int(11)			id of the flight schedule	
fit_num	varchar(45)			unique code per flight joining airline_origindestination	
ifps_id	varchar(45)			Ifps flight id	
callsign	varchar(15)			Callsign of the flight	
airline	varchar(10)			Airline of the flight	
airline_type	varchar(3)			AO type of the airline	
origin	varchar(10)		airport_info_static.icao_id	Origin of the flight	
destination	varchar(10)		airport_info_static.icao_id	Destination of the flight	
gcdistance	int(11)			Great-circle distance from origin to destination	
long_short_dist	char(1)				
sobt	datetime			Scheduled off-block time	
sibt	datetime			Scheduled in-block time	
aircraft_type	varchar(5)		ac_mtow_static.ac	Aircraft ICAO code (e.g., A124)	
mtow	int(11)			Maximum Take-Off Weight (MTOW) of the aircraft (in metric tones)	

wk_tbl_cat	char(1)			Aircraft wake category	
registration	varchar(10)			Aircraft registration (tail number)	
max_seats	int(11)			Maximum number of seats for this type of aircraft	
pax_assigned_cc	int(11)			Number of passengers assigned in ComplexityCosts project as reference	
ecac_200	varchar(4)			If the airport of arrival (arr), departure (dep) or both (both) are part of the top 200 airports of ECAC	
exclude	int(1)			If the flight should be excluded. E.g., cargo flight. In this table all flights are not excluded.	
prev_flight_nid	int(11)			Id of the previous rotation of the flight	
source	varchar(100)			Source of schedules for traceability	

Table 19. Flight_schedule_excluded

Rationale: Schedules used in Domino (excluded flights). Used only to compute original demand at airports to adjust capacity and as reference.

Field	Type	Primary key	Foreign Key	Rationale	Other info
nid	int(11)	*			
flight_id	int(11)			id of the flight schedule	
fit_num	varchar(45)			unique code per flight joining airine_origindestination	
ifps_id	varchar(45)			Ifps flight id	
callsign	varchar(15)			Callsign of the flight	
airline	varchar(10)			Airline of the flight	
airline_type	varchar(3)			AO type of the airline	
origin	varchar(10)		airport_info_static.icao_id	Origin of the flight	
destination	varchar(10)		airport_info_static.icao_id	Destination of the flight	
gcdistance	int(11)			Great-circle distance from origin to destination	
long_short_dist	char(1)				

sobt	datetime			Scheduled off-block time	
sibt	datetime			Scheduled in-block time	
aircraft_type	varchar(5)		ac_mtow_static.ac	Aircraft ICAO code (e.g., A124)	
mtow	int(11)			Maximum Take-Off Weight (MTOW) of the aircraft (in metric tonnes)	
wk_tbl_cat	char(1)			Aircraft wake category	
registration	varchar(10)			Aircraft registration (tail number)	
max_seats	int(11)			Maximum number of seats for this type of aircraft	
pax_assigned_cc	int(11)			Number of passengers assigned in ComplexityCosts project as reference	
ecac_200	varchar(4)			If the airport of arrival (arr), departure (dep) or both (both) are part of the top 200 airports of ECAC	
exclude	int(1)			If the flight should be excluded. E.g., cargo flight. In this table all flights are excluded.	
prev_flight_nid	int(11)			Id of the previous rotation of the flight	
source	varchar(100)			Source of schedules for traceability	

Table 20. Flight_uncertainties_static

Rationale: Statistics used in the model for the uncertainty on the different flight phases modelled as Normal distributions.

Field	Type	Primary key	Foreign Key	Rationale	Other info
phase	varchar(30)	*		Phase to where the uncertainty applies (i.e., climb, cruise)	
units	varchar(5)			Units of the Normal distribution (min for climb, NM for cruise)	
mu	decimal(6,2)			µ of Normal distribution used to model the uncertainty	
sigma	decimal(6,2)			σ of Normal distribution used to model the uncertainty	
computed_as_crossing_fl	int(11)			Threshold to differentiate between climb/cruise used	
source	varchar(45)			Source from these uncertainties for traceability.	

Table 21. FP_pool

Rationale: Pool of flight plans (do not have speed computed for each one). For computational reasons, this is used in the execution of the model when an airline chooses the flight plan of a flight. It includes a reference to another table to get the 2D route that the flight follows.

Field	Type	Primary key	Foreign Key	Rationale	Other info
id	int(11)	*		Incremental index	
icao_orig	varchar(4)			ICAO code of the origin airport of the flight	
icao_dest	varchar(4)			ICAO code of the destination airport of the flight	
bada_code_ac_model	varchar(50)			Code from BADA3 or BADA4 used for the performance (e.g., A340-642)	
fp_distance_nm	decimal(30,3)			Flight length in NM	
trajectory_pool_id	int(11)		trajectory_pool.id	Id of trajctory_pool table	Which trajectory (4D profile is used to generate this flight plan)
route_pool_id	int(11)		route_pool.id	Id of route_pool table	Which route (2D profile is used to generate this flight plan)

Table 22. FP_pool_m

Rationale: Pool of flight plans (with pre-computed speed (i.e., Mach) and CRCO charges). This can be used in the model allowing changes of speed, winds and CRCO charges (e.g. unit rates), but since those computations are done in the execution of the simulation, it requires more computational time.

Field	Type	Primary key	Foreign Key	Rationale	Other info
id	int(11)	*		Incremental index	
icao_orig	varchar(4)			ICAO code of the origin airport of the flight	
icao_dest	varchar(4)			ICAO code of the destination airport of the flight	
bada_code_ac_model	varchar(50)			Code from BADA3 or BADA4 used for the performance	

				(e.g., A340-642)	
fp_distance_nm	decimal(30,3)			Flight length in NM	
trajectory_pool_id	int(11)		trajectory_pool.id	Id of trajectory_pool table	Which trajectory (4D profile is used to generate this flight plan)
route_pool_id	int(11)		route_pool.id	Id of route_pool table	Which route (2D profile is used to generate this flight plan)
crco_cost_EUR	double			CRCO cost of flight plan (EUR)	

Table 23. FP_pool_point

Rationale: All the points (longitude, latitude, altitude) which are part of a trajectory in fp_pool. Some pre-computed information (distance form origin/to destination, ANSP airspace in which the point lies, weight of the aircraft at this point) is used for quick computations within the model.

Field	Type	Primary key	Foreign Key	Rationale	Other info
fp_pool_id	int(11)	*	fp_pool.id	Id of fp_pool table	
sequence	int(11)	*		Sequence number of the point in the flight plan	
name	varchar(45)			Name associated with the point indicating the type or coordinates (takeoff, TOC, TOD, landing)	
coords	point			Point coordinates	WKT format
alt_ft	decimal(15,3)				
time_min	double			Minute in which the point was reached counting from departure	
dist_from_orig_nm	double			Distance between the point and the origin in NM	
dist_to_dest_nm	double			Distance between the point and the destination in NM	
wind	double			Wind considered. In this case is always 0.	0 for all entries
ansp	varchar(15)			NAS ICAO code where the point is located (e.g., EG)	
weight	decimal(15,3)			Aircraft weight at the point (kg)	

fuel	decimal(15,3)			Planned fuel consumed to reach the point from departure	

Table 24. Fp_pool_point_m

Rationale: Same as FP_pool_point but with information on speed: wind, planned speed, minimum and maximum speed (considering the aircraft type, flight level and weight), MRC speed.

Field	Type	Primary key	Foreign Key	Rationale	Other info
fp_pool_id	int(11)	*	fp_pool.id	Id of fp_pool table	
sequence	int(11)	*			
name	varchar(45)			Name associated with the point indicating the type or coordinates	
coords	point			Point coordinates	
alt_ft	decimal(15,3)				
time_min	double			Minute in which the point will be reached from departure	
dist_from_orig_nm	double			Distance between the point and the origin in NM	
dist_to_orig_nm	double			Distance between the point and the destination in NM	
wind	double				
ansp	varchar(15)			NAS ICAO code (e.g., EG) where the point is located.	
weight	decimal(15,3)			Aircraft weight at the point (kg)	
fuel	decimal(15,3)			Fuel consumed to reach the point from departure	
planned_avg_speed_kt	decimal(10,3)			Average planned flight speed at that point in Kt	
max_speed_kt	decimal(10,3)			Maximum speed that can be used at that point by the flight	
min_speed_kt	decimal(10,3)			Minimum speed that can be used at that point by the flight	
mrc_speed_kt	decimal(10,3)			Maximum Range Cruise speed at that point	

Table 25. Full_primary_delay_tactical_cost_static

Rationale: Tactical delay cost table. Used to build a cost function for quick cost computation, without breaking the cost down per type (curfew, compensation etc).

Field	Type	Primary key	Foreign Key	Rationale	Other info
ac	varchar(5)	*		Aircraft ICAO code (e.g., A124)	

sqrt_mtow	double			Square root of MTOW	
scenario	varchar(4)	*		Low , base or high	
phase	varchar(10)	*			'at-gate' for all entries
5	int(11)			Cost at 5 minutes of delay	
15	int(11)			Cost at 15 minutes of delay	
30	int(11)			Cost at 30 minutes of delay	
60	int(11)			Cost at 60 minutes of delay	
90	int(11)			Cost at 90 minutes of delay	
120	int(11)			Cost at 120 minutes of delay	
180	int(11)			Cost at 180 minutes of delay	
240	int(11)			Cost at 240 minutes of delay	
300	int(11)			Cost at 300 minutes of delay or higher	

Table 26. Iedf_atfm_static

Rationale: Inverse empirical cumulative function for ATFM delay. Used to build the empirical distribution from which ATFM delays are sampled.

Field	Type	Primary key	Foreign Key	Rationale	Other info
atfm_type	varchar(45)			Type of ATFM regulation providing the delay: all_vista, weather_all, non_weather_all, all, weather_excluding_airports, non_weather_excluding_airports, all_excluding_airports	
index	bigint(20)			Incremental index. A different one for each atfm_type	
x	double				
y	double				

Table 27. Iedf_wind_static

Rationale: Inverse empirical cumulative function for average cruise wind between origin-destination. Used to build the empirical distribution from which average cruise wind is sampled.

Field	Type	Primary key	Foreign Key	Rationale	Other info
index	bigint(20)			Incremental index. A different one for each triad (icao_country_orig, icao_country_dest, type_wind)	
icao_country_orig	text			NAS ICAO code (e.g.,EG)	
icao_country_dest	text			NAS ICAO code (e.g.,EG)	

type_wind	text			How the wind has been computed, from country-to-country using the first letter of the NAS; and if considering whole flight or only the cruise segments.	country_avg_segment, 1letter_avg_segment, country_avg_flight, 1letter_avg_flight, generic
x	double				
y	double				
type	text			Type of computation based on historic data or generic values if not available	

Table 28. MTT_static

Rationale: Minimum turnaround time for different aircraft types (considering wake turbulence), for different airports (small, medium and large) and for different types of airline. Used as base statistic (quantile) for the turnaround time sampling.

Field	Type	Primary key	Foreign Key	Rationale	Other info
airport_size	text			Size of airport (small, medium, large)	
wake	text			Wake turbulence category (i.e, H-heavy, M - medium, J - super, L - light)	
REG	bigint(20)			Regional airline MTT	
CHT	bigint(20)			Charter airline MTT	
LCC	bigint(20)			Low cost airline MTT	
FSC	bigint(20)			Legacy carrier or scheduled carrier MTT	

Table 29. Non_pax_delay_fit_static

Rationale: Table summarising a regression of non-passenger related costs as a function of time. This is used to build a function for quick cost of delay computations.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario	text			Low, base, high	
phase	text			airborne, at_gate, taxi	
a	double			1st coefficient fitting	
b	double			2nd coefficient fitting	

Table 30. Non_pax_delay_static

Rationale: Non-passenger delay cost for different phases, scenarios and aircraft types. This is used to build the detailed cost function for the airlines in the model.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario	text			low, base, high	
phase	text			airborne, at gate, taxi	
A319	double				
A320	double				
A321	double				
A332	double				
AT43	double				
AT72	double				
B733	double				
B734	double				
B735	double				
B738	double				
B744	double				
B752	double				
B763	double				
DH8D	double				
E190	double				

Table 31. Passenger_compensation_static

Rationale: Information on Regulation 261 rules [2] and uptake. This information is used in the detailed cost of delay function for the airlines. This table represents the actual rule; the uptake ratio is estimated.

Field	Type	Primary key	Foreign Key	Rationale	Other info
flight_type	varchar(15)	*		Long, medium or short	
flight_type_distance_gcd_km_min	int(11)			Minimum distance in GCD for which this compensation applies	
flight_type_distance_gcd_km_max	int(11)			Maximum distance in CGD up to which this compensation applies	
delay_min_minutes	int(11)	*		Minimum delay experienced by passenger at arrival from which this compensation applies	
delay_max_minutes	int(11)	*		Maximum delay experienced by passenger at arrival from which this compensation applies	
compensation	double			Amount of compensation according to Reg 261 (EU)	
uptake	double			Percentage of passengers claiming	

compensation

Table 32. Pax_itineraries

Rationale: Input passenger itineraries. This table is used to build the passenger objects and initialise them.

Field	Type	Primary key	Foreign Key	Rationale	Other info
nid	int(11)	*		Incremental index	
pax	int(11)			number of passengers in group	
avg_fare	double			Average fare of the passengers in the group	
ticket_type	varchar(10)			Type of fare: flex or economy	
leg1	int(11)		flight_schedule.nid	Index of flight_schedule table of the 1st flight leg	
leg2	int(11)		flight_schedule.nid	Index of flight_schedule table of the 2nd flight leg or NULL	
leg3	int(11)		flight_schedule.nid	Index of flight_schedule table of the 3rd flight leg or NULL	
leg4	int(11)		flight_schedule.nid	Index of flight_schedule table of the 4th flight leg or NULL	
source	varchar(100)			source of data for traceability	

Table 33. Prob_atfm

Rationale: Probability of being delayed due to ATFM for different reasons. These probabilities were computed from historical data.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario_id	int(11)	*	scenario.id	Index of scenario table	
atfm_type	varchar(45)	*		all_vista, weather_all, non_weather_all, all, weather_excluding_airports, non_weather_excluding_airports, all_excluding_airports	
p	double				

Table 34. Regulation_at_airport_days_static

Rationale: Number of regulations at airports for historical days with their percentiles. This table allows the model to select random days for regulations, but controls their impact based on the number of regulations. High delay scenarios are built in particular by selecting higher quantiles.

Field	Type	Primary key	Foreign Key	Rationale	Other info
day_start	date	*		Day to which regulations are associated	
number_regulations	int(11)			Number of regulations at airport which were issued on that day	
percentile	double			Percentile according to number of regulations	

Table 35. Regulation_at_airport_static

Rationale: Definition of regulations for each historical day at airports, used as references to model ATFM regulations. This includes the starting time of the regulation, its ending time, and the new capacity of the airport.

Field	Type	Primary key	Foreign Key	Rationale	Other info
icao_id	varchar(10)	*		Airport ICAO code (e.g., EDDM)	
airport_set	varchar(45)			If regulation is applied to an airport set (e.g. EBBR/MB)	
reg_sid	varchar(45)	*		Id of regulation	
reg_reason	varchar(5)			V, E, N, U, S, C, G, T, W, O	
reg_period_start	datetime	*		Date and time of start of the regulation	
reg_period_end	datetime	*		Date and time of end of the regulation	
capacity	int(11)			Capacity associated with the airport	

Table 36. Route_pool

Rationale: Routes possible between origin and destination pairs. These are based either on a route_pool_static (i.e., computed from a given historical flight plan) or a route_pool_o_d_generated when the origin-destination did not exist in the historical data and an estimation has been made.

Field	Type	Primary key	Foreign Key	Rationale	Other info
id	int(11)	*		Incremental index	
based_route_po	int(11)		route_p	Index of route_pool_static_id	

ol_static_id			ool_static_id.id	table	
based_route_pool_o_d_generated	int(11)		route_pool_o_d_generated.id	Index of route_pool_o_d_generated table	
tact_id	int(11)			Id of the historical source	
f_airac_id	int(11)			Id of the historical source	
icao_orig	varchar(4)			Airport ICAO code (e.g., EDDM)	
icao_dest	varchar(4)			Airport ICAO code (e.g., EDDM)	
fp_distance_km	int(11)			Flight length in km	
fp_distance_km_orig	int(11)			Distance in km from origin	
f_database	varchar(15)			Source database for traceability	ddr_1409 for all entries
type	tinytext			Type of data source: historic, based historic with intermediate, based_historic	

Table 37. Route_pool_has_airspace_static

Rationale: Table to link the route_pool with the airspace information. This is used to compute CRCO charges for individual flight plans.

Field	Type	Primary key	Foreign Key	Rationale	Other info
route_pool_id	int(11)		route_pool.id	Index of route_pool table	
airspace_id	int(11)		airspace_static.id	Index of airspace_static table	
sequence	int(11)			Order in the sequence of airspaces crossed by the route	
entry_point	point			Coordinates of entry point in the airspace	
exit_point	point			Coordinates of exit point in the airspace	
distance_entry	int(11)			Distance between the entry point and the origin	
distance_exit	int(11)			Distance between the exit point and the origin	
gcd_km	decimal(15,10)			Great circle distance traveled	

				to go from the entry point to the exit point	
airspace_orig_sid	varchar(50)			Airport ICAO code (e.g., EDDM)	

Table 38. Route_pool_o_d_generated

Rationale: If the route between a given origin-destination did not exist in the empirical data, then a route was generated. This generated route is generated by using historical routes and linking them realistically.

Field	Type	Primary key	Foreign Key	Rationale	Other info
id	int(11)	*		Incremental index	
based_route_pool_static_1_id	int(11)			Index of route_pool_static table first historical route used to generate the new route.	
based_route_pool_static_2_id	int(11)			Index of route_pool_static table second historical route used to generate the new route.	
icao_orig	varchar(4)			Airport ICAO code (e.g., EDDM)	
icao_dest	varchar(4)			Airport ICAO code (e.g., EDDM)	
fp_distance_km	int(11)			Flight length in km	
type	tinytext			Type of data source: based historic with intermediate, based_historic	

Table 39. Route_pool_o_d_generated_has_airspace_static

Rationale: Table to link the route_pool_o_d_generated with the airspace information.

Field	Type	Primary key	Foreign Key	Rationale	Other info
route_pool_id	int(11)		route_pool_o_d_generated.id	Index of route_pool_o_d_generated table	
airspace_id	int(11)		airspace_static.id	Index of airspace_static table	
sequence	int(11)			Order in the sequence of airspaces crossed by the route	
entry_point	point			Coordinates of entry point in the airspace	
exit_point	point			Coordinates of exit point in the airspace	
distance_entry	int(11)			Distance between the entry point and the origin	
distance_exit	int(11)			Distance between the exit point and the origin	
gcd_km	decimal(15,10)			Great circle distance traveled to go from the entry point to the	

				exit point	
airspace_orig_sid	varchar(50)			Airspace ICAO code	

Table 40. Route_pool_static

Rationale: Historic route between origin and destination.

Field	Type	Primary key	Foreign Key	Rationale	Other info
id	int(11)	*		Incremental index	
tact_id	int(11)			Id of the historical source	
f_airac_id	int(11)			Id of the historical source	
icao_orig	varchar(4)			Airport ICAO code (e.g., EDDM)	
icao_dest	varchar(4)			Airport ICAO code (e.g., EDDM)	
fp_distance_km	int(11)			Flight length in km	
f_database	varchar(15)			Source database for traceability	ddr_1409 for all entries
type	tinytext			Type of data source: historic	historic for all entries

Table 41. Route_pool_static_has_airspace_static

Rationale: Table to link historic route_pool with airspace.

Field	Type	Primary key	Foreign Key	Rationale	Other info
route_pool_id	int(11)		route_pool_o_d_generated.id	Index of route_pool_o_d_generated table	
airspace_id	int(11)		airspace_static.id	Index of airspace_static table	
sequence	int(11)			Order in the sequence of airspaces crossed by the route	
entry_point	point			Coordinates of entry point in the airspace	
exit_point	point			Coordinates of exit point in the airspace	
distance_entry	int(11)			Distance between the entry point	

				and the origin	
distance_exit	int(11)			Distance between the exit point and the origin	
gcd_km	decimal(15,10)			Great circle distance traveled to go from the entry point to the exit point	
airspace_orig_sid	varchar(50)			Airspace ICAO code	

Table 42. Scenario

Rationale: Information on the scenario to be modelled in Domino.

Field	Type	Primary key	Foreign Key	Rationale	Other info
id	int(11)	*		Incremental id	
main_scenario	varchar(255)			Test, baseline, tactical_adjustments, pretactical_tactical_synergy, unitary_4DTA, unitary_FP, unitary_FAC or full_TBO	
sub_scenario	varchar(255)			Sub-scenario within the main scenario.	
priority	int(11)			Scenario priority: -1 (test), 0, 1, 2, 3, 4, 5	
FAC	int(11)			Level of FAC mechanism (0,1,2)	
FP	int(11)			Level of FP mechanism (0,1,2)	
4DTA	int(11)			Level of 4DTA mechanism (0,1,2)	
coordinated	tinytext(1)			If mechanisms are implemented coordinated or not	
buffers	varchar(5)			Buffers in scenario (D, L)	
delays	varchar(5)			Delay level in scenario (D,H)	
uptake	varchar(5)			Uptake level of mechanism (D, L)	
flight_set	int(11)			Flight set id to identify schedules in scenario from table flight_subset	
regulations_airport_day	date			Date use as reference to generate explicit ATFM regulations at aiports from table regulation_at_airport_static	
description	text			Textual description of scenario	

Table 43. Soft_cost_delay_static

Rationale: Passenger soft cost parameters for different scenarios. This table is used to estimate the impact of delay on the airline. This is based on the disutility of the passengers, using a logit rule, assuming that a decrease in passenger utility translates partly into a loss of market share for the airline.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario	text			Low scenario, Base scenario, High scenario	
k	double				
k_p	double				
a	double				
b	double				
c	double				

Table 44. Taxi_in_static

Rationale: Taxi-in information from CODA (from IATA Summer Season 2010). Distributions are built from the statistics included in this table, from which the taxi times are sampled.

Field	Type	Primary key	Foreign Key	Rationale	Other info
icao_id	varchar(4)	*		Airport ICAO code (e.g., EDDM)	
iata	varchar(3)			Airport IATA code (e.g., MUC)	
airport_name	varchar(120)			Name of the airport	
mean_txi	int(11)			Mean taxi-in time	
std_deviation	int(11)			Std deviation	
10th_perc	int(11)			10th percentile	
median	int(11)			Median	
90th_perc	int(11)			90th percentile	
source	varchar(350)			source of the data for traceability	IATA Summer Season 2010

Table 45. Taxi_out_static

Rationale: Taxi-out information from CODA (from IATA Summer Season 2010).

Field	Type	Primary key	Foreign Key	Rationale	Other info
icao_id	varchar(4)	*		Airport ICAO code (e.g., EDDM)	
iata	varchar(3)			Airport IATA code (e.g., MUC)	

airport_name	varchar(120)			Name of the airport	
mean_txo	int(11)			Mean taxi-out itme	
std_deviation	int(11)			Std deviation	
10th_perc	int(11)			10th percentile	
median	int(11)			Median	
90th_perc	int(11)			90th percentile	
source	varchar(350)			source of the data for traceability	IATA Summer Season 2010

Table 46. Taxi_out_wake_static

Rationale: Taxi-out information from CODA (from IATA Summer Season 2010).

Field	Type	Primary key	Foreign Key	Rationale	Other info
icao_id	varchar(4)	*		Airport ICAO code (e.g., EDDM)	
iata	varchar(3)			Airport IATA code (e.g., MUC)	
airport_name	varchar(120)			Name of the airport	
wake_turbulence	varchar(1)	*		Wake turbulence (i.e, H, M, J, L)	
mean_txo	int(11)			Mean taxi-out time	
std_deviation	int(11)			Std deviation	
10th_perc	int(11)			10th percentile	
median	int(11)			Median	
90th_perc	int(11)			90th percentile	
source	varchar(350)			source of the data for traceability	IATA Summer Season 2010

Table 47. Trajectory_pool

Rationale: A 4D trajectory for a given route (2D). These trajectories have been generated using the BADA model and are used to initialise the planned trajectory in the model.

Field	Type	Primary key	Foreign Key	Rationale	Other info
id	int(11)	*			
route_pool_id	int(11)		route_pool .id	Index of route_pool table to which this trajectory applies	
version	int(11)			version used to generate the trajectory	
distance_orig_fp_km	float			Flight length in km	
bada_code_ac_model	text			Code from BADA3 or BADA4 used for the performance (e.g., A340-642)	
bada_version	double			Bada version used (i.e, 3, 4)	
version_description	varchar(45)			Test - no wind, Test - speed	

		in info, Nominal - avg wind cruise - 0.7 payload or blank	
status	int(11)	Status of the trajectory computation: 0 – all computation ok	0 for all entries

Table 48. Trajectory_segment

Rationale: Segments of the trajectory. This table is used during the iterative cycle when the flight follows its trajectory in the simulation. It allows quick access to the distance, time needed, and weight of the aircraft at different points.

Field	Type	Primary key	Foreign Key	Rationale	Other info
trajectory_pool_id	int(11)	*	trajectory_pool_id.id	Index of trajectory_pool_id table	
order	int(11)	*		Order in which this segment is used in the trajectory	
fl_0	float			flight level of first point	
fl_1	float			Flight level of second point	
distance_nm	float			Distance between the points	
time_min	float			Time to go from first point to second point	
fuel_kg	float			Fuel consumed in segment (kg)	
weight_0	float			weight of the plane in the first point	
weight_1	float			weight of the plane in the second point	
avg_m	float			Average Mach speed	
avg_wind	float			Average wind (kt) in segment	
segment_type	varchar(15)			Climb, cruise or descent	
status	int(11)			Status of segmetn computation	

5.2.2 Output data for the model

This section includes all tables used to store the data produced by the model.

Table 49. Output_dci

Rationale: Specific output of the 4DTA mechanism due to DCI. This is used to monitor the decision making process of the airline regarding DCI and track any potential issues.

Field	Type	Primary key	Foreign Key	Rationale	Other info
n_iter	int(11)	*		Execution iteration	
model_version	varchar(10)	*		model version of Domino used to generate these results	
flight_uid	int(11)	*		flight identifier	
scenario_id	int(11)	*		Scenario id	
dci_check_timestamp	varchar(50)	*		Timestamp on when the DCI has been computed (pushback_ready, top_of_climb, top_of_climb_slow_down)	top_of_climb_slow_down indicates that at TOC the aircraft decides to slow down instead of increasing speed.
origin	varchar(4)			ICAO code of the origin airport of the flight	
destination	varchar(4)			ICAO code of the destination airport of the flight	
estimated_delay	double			Delay estimated at dci_check_timestamp	
perc_selected	double			percentage of speed selected: 0 → MCR, 1→ VMO	
recovering_delay	double			Delay that is expected to be recovered	
dfuel	double			Extra fuel planned due to change on speed	
extra_fuel_available	double			Fuel available at dci_check_timestamp to do DCI	
recoverable_delay	double			Maximum delay could be recovered flying at VMO	

Table 50. Output_eaman

Rationale: Specific output of the FAC mechanism. This table is used to explore the efficiency and consistency of the FAC mechanism. In particular, on which data the E-AMAN based its decisions and what these resulted in.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario_id	int(11)	*		Scenario id	
n_iter	int(11)	*		Execution iteration	
model_version	varchar(10)	*		Model version of Domino used to generate these results	
uid	int(11)	*		identifier of flight	
eaman_planned_clt	datetime			Planned controlled landing time (arrival slot time)	
eaman_planned_assigned_delay	double			Delay assigned to flight at planning horizon min	
eaman_planned_absorbed_air	double			Delay planned to be absobed in the air by slowign down min	NULL if no delay is assigned
eaman_planned_perc_selected	double			Speed selected to absorb delay	NULL if no delay is assigned
eaman_planned_fuel	double			Planned fuel variation due to speed slection	NULL if no delay is assigned
eaman_tactical_clt	datetime			Tactical (final) controlled landing time (arrival slot time)	
eaman_tactical_assigned_delay	double			Holding delay needed min	

Table 51. Output_swaps

Rationale: Output associated with FP mechanism. This is used primarily to estimate the impact of the flight swapping on the cost of the airlines.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario_id	int(11)	*		Scenario id	
model_version	varchar(30)	*		Model version of Domino	
n_iter	int(11)	*		Execution number	

id_swap	int(11)	*		Id of swap done
flight1_uid	int(11)	*		Index of flight uid
flight2_uid	int(11)			Index of flight uid
flight1_id	int(11)			Index of flight
flight2_id	int(11)			Index of flight
cost_swap_flight	float			Cost of swapping
i_order	int(11)			

Table 52. Output_flights

Rationale: Flight centric output metrics per flight. If flight is cancelled then AOBT and all fields related to actual (including all m3 (actual trajectories)) are NULL. This table is primarily used to compute flight-centric metrics, but is also crossed with passenger data to produce related indicators. The table is highly detailed, comprising most information on the processes happening before and during the flight. However, the number of flights is relatively small (27k), so this table is small enough to be handled and analysed.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario_id	int(11)	*		Scenario id	
n_iter	int(11)	*		Execution iteration	
model_version	varchar(10)	*		model version of Domino used to generate these results	
uid	int(11)	*		Flight uid	
id	int(11)		flight_schedule.nid	Index of flight_schedule table	Link to flight id from schedule
aoc_uid	int(11)			Airline uic from simulation	
origin_uid	int(11)			Origin uic from simulation	
destination_uid	int(11)			Destination uic from simulation	
origin	varchar(4)			ICAO code of the origin airport of the flight	
destination	varchar(4)			ICAO code of the destination airport of the flight	
fp_pool_id	int(11)		fp_pool_m.id	Index of fp_pool_m table	Which flight plan from the fp_pool_m has been used by flight
ao_iata	varchar(5)			ICAO code of the origin	

		airport of the flight	
ac_icao	varchar(10)	Aircraft ICAO code (e.g., A124)	
ac_model	varchar(30)	Aircraft type from BADA	
ac_registration	varchar(10)	Aircraft registration (tail number)	
sobt	datetime	Scheduled off-block time	
sibt	datetime	Scheduled in-block time	
cobt	datetime	Calculated off-block time if ATFM delay assigned	
eobt	datetime	Estimated off-block time (last estimated)	
eibt	datetime	Estimated initial off-block time (first estimtated)	
pbrt	datetime	Pushback ready time	
aobt	datetime	Actual off-block time	NULL if cancelled
aibt	datetime	Actual in-block time	
atfm_delay	double	ATFM delay	If ATFM delay = 0 but atfm_reason = Null then flight not regulated If ATFM delay = 0 and atfm_reason has reason then flight regulated but regulation assigned a delay of 0
atfm_reason	text	Regulation reason of the ATFM delay	C: Capacity W: Weather Null: Flight no regulated If reason has "_AP" ending, e.g. "C_AP", means that

			the regulation is at an airport and has been explicitly modelled.
exot	double	Estimated taxi-out time	
exit	double	Estimated taxi-in time	
axot	double	Actual taxi-out time	
axit	double	Actual taxi-in time	
atot	datetime	Actual take-off time	
clt	datetime	Controlled landing time at runway (landing slot)	
alt	datetime	Actual landing time	
m1_tow	double	Planned take-off weight (kg)	
m1_lw	double	Planned landing weight (kg)	
m1_fp_dist_nm	double	Planned FP distance (NM)	
m1_climb_dist_nm	double	Planned climb distance (NM)	
m1_cruise_dist_nm	double	Planned cruise distance (NM)	TOD-TOC
m1_descent_dist_nm	double	Planned descent distance (NM)	
m1_num_cruise_climbs	int(11)	Planned number of cruise segments	
m1_toc_nm	double	Distance from origin when TOC is planned to be reached	
m1_tod_nm	double	Distance from origin when TOD is planned to be reached	
m1_toc	datetime	When TOC is planned to be reached: If flight executed atot + planned TOC time If cancelled sobt+exot+planned TOC time	
m1_tod	datetime	When TOD is planned to be reached: If flight executed atot +	

		planned TOD time If cancelled sobt+exot+planned TOD
m1_toc_fuel_kg	double	Planned amount of fuel to be used when reaching TOC (kg)
m1_tod_fuel_kg	double	Planned amount of fuel to be used when reaching TOD (kg)
m1_avg_cruise_fl	double	Planned average cruise level (FL)
m1_avg_cruise_speed_kt	double	Planned average cruise speed (kt)
m1_avg_cruise_speed_m	double	Planned average cruise speed (mach)
m1_avg_cruise_wind_kt	double	Planned average wind at the cruise altitude (kt)
m1_fp_time_min	double	Planned FP time (min)
m1_climb_time_min	double	Planned climb time (min)
m1_cruise_time_min	double	Planned time in cruise (min)
m1_descent_time_min	double	Planned descent time (min)
m1_fuel_kg	double	Planned fuel (kg)
m1_climb_fuel_kg	double	Planned fuel for climb (kg)
m1_cruise_fuel_kg	double	Planned cruise fuel (kg)
m1_descent_fuel_kg	double	Planned descent fule (kg)
m3_tow	double	Actual take-off weight (kg)
m3_lw	double	Actual landing weight (kg)
m3_fp_dist_nm	double	Actual FP distance (NM)
m3_climb_dist_nm	double	Actual climb distance (NM)
m3_cruise_dist_nm	double	Actual cruise distance (NM)
m3_descent_dist_nm	double	Actual descent distance (NM)
m3_num_cruise_climbs	double	Actual number of cruise steps
m3_toc_nm	double	Actual distance to TOC (NM)

m3_tod_nm	double	Actual distance to TOD (NM)
m3_toc	datetime	Actual time when TOC is reached
m3_tod	datetime	Actual time when TOD is reached
m3_toc_fuel_kg	double	Actual fuel used up to TOC (kg)
m3_tod_fuel_kg	double	Actual fuel used up to TOD (kg)
m3_avg_cruise_speed_kt	double	Actual average cruise speed in kt
m3_avg_cruise_speed_m	double	Actual average cruise speed in Mach
m3_avg_cruise_wind_kt	double	Actual average cruise wind in kt
m3_fp_time_min	double	Actual flight plan time (min)
m3_climb_time_min	double	Actual initial climb time (min)
m3_cruise_time_min	double	Actual time of cruise phase (between TOC and TOD) (min)
m3_descent_time_min	double	Actual time of descent (min)
m3_holding_time	double	Actual amount of holding at arrival (min)
m3_fuel_kg	double	Actual total amount of fuel used (kg)
m3_climb_fuel_kg	double	Actual total amount of fuel used climb phase (kg)
m3_cruise_fuel_kg	double	Actual amount of fuel used in the cruise phase (between TOC and TOD) (kg)
m3_descent_fuel_kg	double	Actual total amount of fuel used descent phase (kg)
m3_holding_fuel_kg	double	Actual total amount of fuel used at holding (kg)
duty_of_care	double	Money spent on duty of care for passengers due to delay (EUR)
soft_cost	double	'Soft cost' triggered by this flight to the

		company due to loss of market share (EUR)
transfer_cost	double	Cost of rebooking some passengers on another alliance's airline's flight (EUR)
compensation_cost	double	Compensation paid to passengers based on Regulation 261 (EUR)
non_pax_cost	double	Crew and maintenance cost due to delay (EUR)
non_pax_curfew_cost	double	Cost non due to passengers due to cancellation of flight for curfew (EUR)
fuel_cost_m1	double	Planned cost of fuel (EUR)
fuel_cost_m3	double	Actual cost of fuel (EUR)
crco_cost	double	Route charges (EUR)
main_reason_delay	varchar(10)	C, W, TA, ER, CANCEL, CANCEL_CF or RD
exclude	int(11)	If the flight should be excluded

Table 53. Output_pax

Rationale: Passenger centric output metrics per passenger group. The number of passenger groups being quite high (more than 1M per simulation), we keep as little information as possible in this table, because of its size. This includes the scheduled flight(s) and the actual one(s), as well as some information on the compensation received by the passengers.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario_id	int(11)	*		Scenario id	
n_iter	int(11)	*		Execution umber	
model_version	varchar(10)	*		Model version of Domino	
id	int(11)	*		Internal passengers group id in simulation	
n_pax	int(11)			number of passengers n group	
pax_type	varchar(20)			type of fare: flex or economy	
fare	double			average fare paid per passenger in group (EUR)	
origin_uid	int(11)			Airport of origin uid in simulation	
destination_uid	int(11)			Airport of destination uid in	

				simulation	
time_at_gate	datetime			Time when pax arrived to gate to board	
compensation	double			Amount paid in compensation (Reg 261)	
duty_of_care	double			Amount paid in duty of care	
initial_sobt	datetime			Initial schedule time off-block for first leg of pax	
initial_sibt	datetime			Initial schedule time in-block for last leg of pax	
leg1	int(11)		flight_schedule.nid	Index of flight_schedule table	First flight used by group
leg2	int(11)		flight_schedule.nid	Index of flight_schedule table	Second flight used by group or NULL
leg3	int(11)		flight_schedule.nid	Index of flight_schedule table	Third fligth used by group or NULL
leg4	int(11)		flight_schedule.nid	Index of flight_schedule table	Fourth fligth used by group or NULL
leg5	int(11)		flight_schedule.nid	Index of flight_schedule table	Fifth flight used by group or NULL
destination	varchar(4)			ICAO code of the destination airport of the flight	
origin	varchar(4)			ICAO code of the origin airport of the flight	
id2	varchar(100)			Internal new id for group if group was split due to missed connections.	

Table 54. Output_RNG

Rationale: Random number generator parameters used in given run to allow for reproducibility of results.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario_id	int(11)	*		Scenario id	
n_iter	int(11)	*		Number of execution	
model_version	varchar(10)	*		Model version of Domino	
0	text				
1	text				
2	text			Needed to reproduce seed and	

3	text	re-run simulation if desired	
4	text		

Table 55. Output_sim_general

Rationale: General parameters of the simulation. This includes in particular the scenario chosen for the simulation.

Field	Type	Primary key	Foreign Key	Rationale	Other info
scenario_id	int(11)	*		Scenario id	
n_iter	int(11)	*		Execution number	
model_version	varchar(10)	*		Model version of Domino	
day_ref_regulations_airport	date			Day to which regulation at airport are based on	
eaman_l0_queue	varchar(100)			How is FAC implemented in Level 0: as a queue or optimising	
eaman_solver	varchar(100)			Which solver is used by the FAC "google_or", "pyomo"	

6 Next steps and look ahead

The database continues to evolve to capture the requirements of storing the Domino results. The database presented in this deliverable has been used to generate and analyse the results of the investigative case studies reported in D5.2 - Investigative case studies results. Generating those results have helped us to identify some limitations that will be solved in the final version of the model.

These limitations refer to increasing the number of parameters stored in the output of some tables to facilitate the analysis of the results, and the required time to concurrently write results in the database. These drawbacks will be solved by adding more detailed output to the database from the model, reducing the need of joining output tables, and by considering the use of NoSQL databases for the storage of the output of the model. This will allow us to preform new computations and to simplify and expedite the computation of some results. The output tables are significantly large, which in some cases mean that linking them with other tables to build complex metrics can be very computational time consuming (e.g., counting number of actual passengers in a flight linking the output_pax and the output_flight tables). This information can be pre-computed and stored in the database.

In particular some of the fields we have identified to be added to the database are:

- output_flights
    - buffers in the schedules and actual buffers available
    - how long flights wait at gate for passengers
    - number of passengers in flights (planned and actual)
- output_pax
    - how passengers have waited for connections
    - connecting times
    - number of missed connections
    - original itineraries of the different passenger groups had planned (this traceability is kept only indirectly, currently)

The expected changes that will be made to the passenger output might trigger a reconsideration of the number of tables used to represent this information.

As presented in Section 4, a new clustering of routes could be considered for the final deliverable. This will mean that new trajectories (4D profiles) and flight plans will need to be recomputed. We consider that this will, however, not significantly change the results, and hence is not a high priority in the development of Domino.

7 References

[1] Domino Project Consortium, "D2.1 Data management and sources," 2018.

[2] European Parliament, "Regulation No 261/2004 of the European Parliament and of the Council. Establishing common rules on compensation and assistance to passengers in the event of denied boarding and of cancellation or long delay of flights, and repealing Regulation No 295/91," 2004.

[3] Vista Project Consortium, "D5.2 Final Report," 2018.

8 Acronyms

AOBT: Actual Off Block Time

ATM: Air traffic management

ATFM: Air traffic flow management

CRCO: Central Route Charges Office

DDR2: Demand Data Repository (second phase)

H2020: Horizon 2020 research programme

MCT: Minimum connecting time

MTT: Minimum turnaround time

SES: Single European Sky

SESAR: Single European Sky ATM research

SJU: SESAR Joint Undertaking

SQL: Structured Query Language

NoSQL: Non-relational database (Non-SQL)

TOC: Top of climb

TOD: Top of descent

-END OF DOCUMENT-