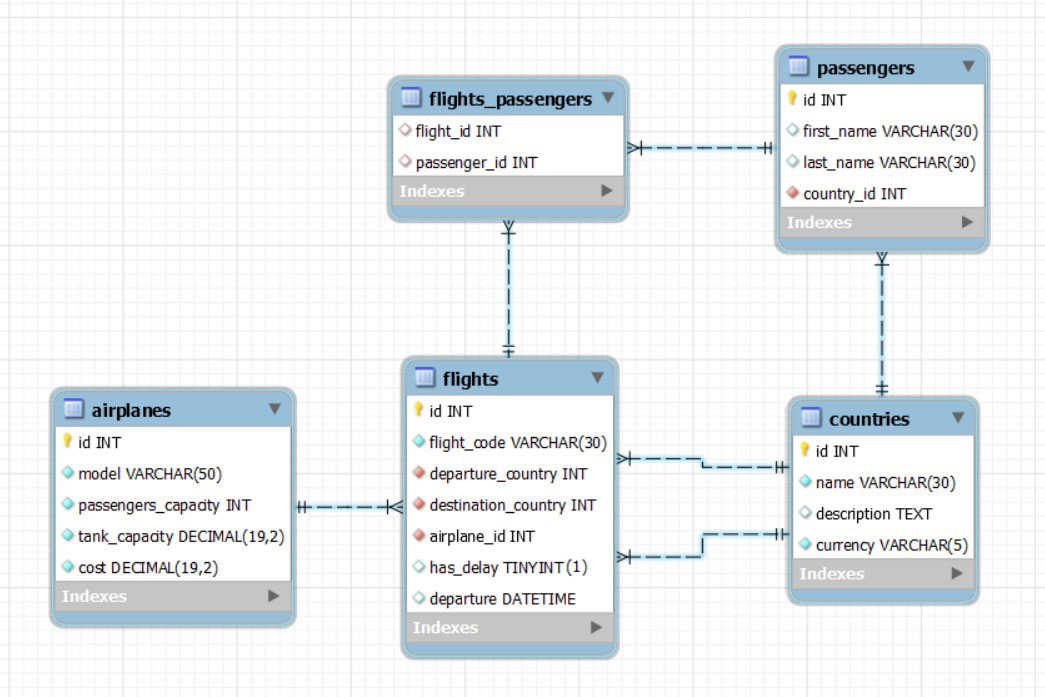
# MySQL Retake Exam Airlines DB

An airline database is an invaluable asset for many airline departments, especially flight operations. The aim is to provide the most accurate, complete and relevant data to assist in your pre-flight planning and dispatch operations. Now you are the leader of the team and you need to manage the system so they can keep track of all flights and provide safety for the passengers.

## Section 0: Database Overview

You have been given an Entity / Relationship Diagram of the Database:



The **airlines\_db’s** **Database** needs to hold information about **airplanes**, **flights**, **passengers** and **countries**.

Your task is to create a database called **airlines\_db**. Then you will have to create several **tables**.

* **countries** – contains information about the **countries**.
* **flights** – contains information about the **flights**.
  + Each flight has a **departure country, destination country** and **airplane**.
* **passengers** – contains information about the **passengers**.
  + Each passenger has a **country**.
* **airplanes** – contains information about the **airplanes**.
* flights\_passengers – a **many** to **many** **mapping** table between the **flights** and the **passengers**.

## Section 1: Data Definition Language (DDL) – 40 pts

Make sure you implement the whole database correctly on your local machine so that you can work with it.

The instructions you'll be given will be the minimal needed for you to implement the database.

### Table Design

You have been tasked to create the tables in the database by the following models:

### countries

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| name | A **string** containing a maximum of **30 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**. UNIQUE** values. |
| description | A very **long** string field |  |
| currency | A **string** containing a maximum of **5 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |

### airplanes

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| model | A **string** containing a maximum of **50 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**. UNIQUE** values. |
| passengers\_capacity | **Integer,** from **1** to **2,147,483,647.** | **NULL** is **NOT** permitted**.** |
| tank\_capacity | **DECIMAL**, up to **19 digits**, **2** of which are after the **decimal point**. | **NULL** is **NOT** permitted**.** |
| cost | **DECIMAL**, up to **19 digits**, **2** of which are after the **decimal point**. | **NULL** is **NOT** permitted**.** |

### passengers

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| first\_name | A **string** containing a maximum of **30 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| last\_name | A **string** containing a maximum of **30 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| country\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table countries.  **NULL** is **NOT** permitted**.** |

### flights

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| flight\_code | A **string** containing a maximum of **30 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**. UNIQUE** values. |
| departure\_country | **Integer,** from **1** to **2,147,483,647.** | Relationship with table countries.  **NULL** is **NOT** permitted**.** |
| destination\_country | **Integer,** from **1** to **2,147,483,647.** | Relationship with table countries.  **NULL** is **NOT** permitted**.** |
| airplane\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table airplanes.  **NULL** is **NOT** permitted**.** |
| has\_delay | Can be **true** or **false**. |  |
| departure | The **date** and **time** of the departure. |  |

### flights\_passengers

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| flight\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table flights. |
| passenger\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table passengers. |

Submit your solutions in Judge on the first task. Submit **all** SQL table creation statements.

You will also be given a data.sql file. It will contain a **dataset** with random data which you will need to **store** in your **local database**. This data will be given to you so you will not have to think of data and lose essential time in the process. The data is in the form of **INSERT** statement queries.

## Section 2: Data Manipulation Language (DML) – 30 pts

Here we need to do several manipulations in the database, like changing data, adding data, etc.

Select and join only tables and columns that are needed in the exercises. Any additional or less information will be considered wrong.

### Insert

You will have to **insert** records of data into the **airplanes** table.

The new data will be based on **passengers** with **id** equal or less than **5**. **Insert data** in the **airplanes** table with the **following values**:

• **model** – set it to the first name of the passenger but **reversed** followed by **'797'**.

• **passengers\_capacity** – set it to the number of characters of the **passenger's last name** multiplied by **17**.

• **tank\_capacity** – set it to the **id** of the **passengers** multiplied by **790.**

• **cost** – set it to the number of characters of passenger's **first name** multiplied by **50.6** .

### Update

There was a mistake at the Armenian airport with the airplanes.You must change the airplanes for all **flights** that **departure** from **Armenia.** Raise the **airplane\_id** by 1 for all flights flying from **Armenia.**

### Delete

**Delete** all **flights** that don't have passengers.

## Section 3: Querying – 50 pts

And now we need to do some data extraction. **Note** that the **example results** from **this section** use a **fresh database**. It is **highly recommended** that you **clear** the **database** that has been **manipulated** by the **previous problems** from the **DML** **section** and **insert again** the **dataset** you’ve been given, to ensure **maximum consistency** with the **examples** given in this section.

### Airplanes

Extract from the **airlines\_db**, info about the **airplanes**.

**Order** the results by **cost** in **descending** order and then by **id** in **descending**.

#### Required Columns

* id
* model
* passengers\_capacity
* tank\_capacity
* cost

#### Example

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **model** | **passengers\_capacity** | **tank\_capacity** | **cost** |
| 8 | Boeing 767 | 550 | 24458.00 | 139.36 |
| 13 | Irkut MC-21 | 301 | 37530.00 | 138.68 |
| 12 | Embraer E-Jet E2 family | 465 | 24973.00 | 138.34 |
| ... | ... |  |  |  |

### Flights from 2022

Write a query that returns: **flight\_code**, **departure\_country**, **airplane\_id** and **departure** from table **flights**. **Filter** flights only from the 2022 **year**.

**Order** the results **ascending** by **airplane\_id** then by **flight\_code** and show only the first **20** results.

#### Required Columns

* flight\_code
* departure\_country
* airiplane\_id
* departure

#### Example

|  |  |  |  |
| --- | --- | --- | --- |
| **flight\_code** | **departure\_country** | **airplane\_id** | **departure** |
| BA-910 | 18 | 2 | 2022-03-26 20:05:43 |
| DP-070 | 19 | 2 | 2022-05-17 00:28:14 |
| HM-237 | 4 | 2 | 2022-08-18 01:38:08 |
| … | … | … | … |

### Private flights

Some passengers don’t have a booked flight but want to travel in private.

Write a query that returns: **flight\_code**, **full\_name** and **country** for all passengers that don’t have a booked flight.

To find their private **flight\_code** you must take the first 2 letters of their last name in upper case followed by their **country id**.

**Order** by **country\_id** in **ascending** order.

#### Required Columns

* flight\_code (first 2 characters from last name + country\_id)
* full\_name (first\_name + " " + last\_name)
* country\_id

#### Example

|  |  |  |
| --- | --- | --- |
| **flight\_code** | **full\_name** | **country\_id** |
| SL6 | Ginnie Slesser | 6 |
| JA18 | Ericka Jales | 18 |
| BE24 | Myrwyn Beet | 24 |
| SA32 | Angel Saines | 32 |

### Leading destinations

Top leading magazine "Air Above" wants to write an article about the top destinations with over 20 booked tickets.

Extract from the database, the **countries** that have **20** or **more** booked tickets for flights as a **destination country** by **passengers**.

**Order** the results **descending** by **booked\_tickets.**

#### Required Columns

* name (country)
* currency
* booked\_tickets (number of booked tickets from passengers to the country as a destination)

#### Examples

|  |  |  |
| --- | --- | --- |
| **name** | **currency** | **booked\_tickets** |
| Italy | EUR | 29 |
| United States | USD | 26 |
| Argentina | ARS | 24 |
| China | CNY | 24 |
| … | … | … |
| Nigeria | NGN | 20 |

### Parts of the day

From the database extract the **flight\_code** and **departure**. **Morning** is from 5AM to 11:59AM, **Afternoon** is from 12PM to 4:59PM, **Evening** is from 5PM to 8:59PM and **Night** is from 9PM to 4:59AM.

**Order** the results **descending** by **flight\_code**.

#### Required Columns

* flight\_code
* departure
* day\_part (Morning is from 5AM to 11:59AM, Afternoon is from 12PM to 4:59PM, Evening is from 5PM to 8:59PM and Night is from 9PM to 4:59AM)

#### Example

|  |  |  |
| --- | --- | --- |
| **flight\_code** | **departure** | **day\_part** |
| ZT-264 | 2023-08-10 08:18:37 | Morning |
| ZP-782 | 2022-08-11 16:32:39 | Afternoon |
| ZN-100 | 2023-02-02 05:58:04 | Morning |
| . . . | . . . | . . . |
| AB-776 | 2022-10-22 20:48:40 | Evening |

## Section 4: Programmability – 30 pts

The time has come for you to prove that you can be a little more dynamic on the database. So, you will have to write several procedures.

### Number of flights

Create a **user defined function** with the name **udf\_count\_flights\_from\_country(country VARCHAR(50))** that receives a **country name** and returns the total number of **flights** departing from that country.

#### Required Columns

* flights\_count (udf\_count\_flights\_from\_country)

#### Example

|  |
| --- |
| **Query** |
| SELECT udf\_count\_flights\_from\_country('Brazil') AS 'flights\_count'; |
| flights\_count |
| **5** |

|  |
| --- |
| **Query** |
| SELECT udf\_count\_flights\_from\_country('Philippines') AS 'flights\_count'; |
| flights\_count |
| **7** |

### Delay flight

A flight has been delayed. Your task is to find the **flight** and change the **delay** status to true and the **departure** time by 30 minutes.

Create a stored procedure udp\_delay\_flight which accepts the following parameters:

* code VARCHAR(50)

#### Result

|  |
| --- |
| **Query** |
| CALL udp\_delay\_flight('ZP-782'); |
| This execution will update flight with code 'ZP-782' |

#### Result

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **flight\_code** | **has\_delay**  before | **departure**  before | **->** | **has\_delay**  after | **departure**  after |
| ZP-782 | 0 | 2022-08-11 16:32:39 | -> | 1 | 2022-08-11 17:02:39 |
| XY-389 | 1 | 2022-04-27 00:51:17 | -> | 1 | 2022-04-27 01:21:17 |
| EA-907 | 0 | 2023-07-10 21:15:57 | -> | 1 | 2023-07-10 21:45:57 |