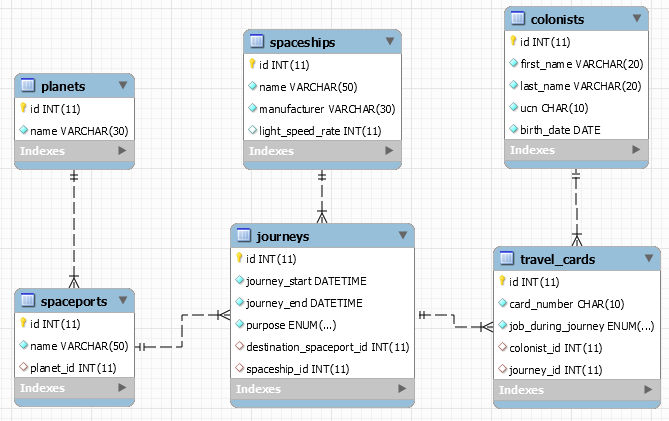
# Database Basics (MySQL) Exam

# Colonial Journey Management System

2000 years from now, the known space is colonized by the human race. However, the four Citadel Council races are planning to populate new home worlds in the SoftUnia Galaxy as part of strategy called the SoftUnia Initiative. 20000 citizens are send abroad space transportation vessels. The Council has asked you to create a Colonization Management system so they can keep track of the colonists journeys trough the stars.

## Section: Database Overview

You have given and Entity / Relationship Diagram of the CJMS Database:



The CJMS Database holds information about colonists, their travel cards, information about the journeys, types of space vessels and destination planets. Your task is to create a database called **colonial\_journey\_management\_system\_db**. Then you will have to create several **tables**.

* **planets** – contains information about **planets**.
* **spaceports** – contains information about **space ports**.
* **spaceships –** contains information about **space ships**.
* **colonists –** contains information about **colonists**.
* **journeys –** contains information about **journeys**.
* **travel\_ cards –** contains information about **travel cards**.

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

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

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

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

**planets**

|  |  |  |
| --- | --- | --- |
| **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**.** |

**spaceports**

|  |  |  |
| --- | --- | --- |
| **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 **50 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| planet\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table **planets.** |

#### spaceships

|  |  |  |
| --- | --- | --- |
| **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 **50 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| manufacturer | A **string** containing a maximum of **30 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| light\_speed\_rate | **Integer,** from **0** to **2,147,483,647.** | Has a **default value** of 0. |

#### colonists

|  |  |  |
| --- | --- | --- |
| **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 **20 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| last\_name | A **string** containing a maximum of **20 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| ucn | A **string** containing exactly **10 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.**  **UNIQUE** values. |
| birth\_date | Date **without** time. | **NULL** is **NOT** permitted**.** |

#### journeys

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| journey\_start | Date **with** time. | **NULL** is **NOT** permitted**.** |
| journey\_end | Date **with** time. | **NULL** is **NOT** permitted**.** |
| purpose | **A string** containing a maximum of **11 characters**. Unicode is **NOT** needed. | Should **only** contain one of the following purposes: “**Medical**”, “**Technical**”, “**Educational**”, “**Military**” |
| destination\_spaceport\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table **spaceports.** |
| spaceship\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table **spaceships.** |

**travel\_cards**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| card\_number | A **string** containing exactly **10 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.**  **UNIQUE** values. |
| job\_during\_journey | A **string** containing a maximum of **6** characters. Unicode is **NOT** needed. | Should **only** contain one of the following jobs: “**Pilot**”, “**Engineer**”, “**Trooper**”, “**Cleaner**”, “**Cook**” |
| colonist\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table **colonists.** |
| journey\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table **journeys.** |

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: Data Manipulation Language (DML) – 30 pts

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

### Data Insertion

You will have to **INSERT** records of data into the travel\_cards table, based on the **colonists** table.

For colonists with id between **96** and **100(inclusive)**, **insert data** in the **travel\_cards** table with the following values:

* For **colonists** born after ‘**1980-01-01**’, the **card\_number** must be combination between the **year of birth**, **day** and the **first 4** **digits** from the **ucn**. For the rest – **year of birth**, **month** and the **last 4 digits** from the **ucn**.
* For **colonists** with **id** that can be divided by **2** without remainder, job must be **‘Pilot’,** for colonists with id that can be divided by **3** without remainder – ‘**Cook**’, and everyone else – ‘**Engineer**’.
* **Journey id** is the **first digit** from the colonist’s **ucn**.

### Data Update

**UPDATE** those journeys’ purpose, which meet the following conditions:

* If the journey’s **id** is dividable by **2** without remainder – **‘Medical’**.
* If the journey’s **id** is dividable by **3** without remainder – **‘Technical’**.
* If the journey’s **id** is dividable by **5** without remainder – **‘Educational’**.
* If the journey’s **id** is dividable by **7** without remainder – **‘Military’**.

### Data Deletion

**REMOVE** from **colonists**, those which **are not** assigned to any **journey**.

## Section: Querying – 100 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.

### 04.Extract all travel cards

Extract from the database, all **travel cards.** Sort the results by **card number ascending.**

#### Required Columns

* **card\_number**
* **job\_during\_journey**

#### Example

|  |  |
| --- | --- |
| **card\_number** | **job\_during\_journey** |
| **0032031181** | Engineer |
| 0037637193 | Engineer |
| ... | ... |

### 05. Extract all colonists

Extract from the database, all **colonists.** Sort the results by **first name**, as **second criteria**, sort them by **last name**, and finally by **id** in **ascending** order.

#### Required Columns

* **id**
* **full\_name(**first\_name + last\_name separated by a single space**)**
* **ucn**

#### Example

|  |  |  |
| --- | --- | --- |
| **id** | **full\_name** | **ucn** |
| 35 | Aigneis McConville | 9225403496 |
| 92 | Althea Kelinge | 9998159318 |
| ... | ... | ... |

### Extract all military journeys

Extract from the database, all **Military** journeys. Sort the results **ascending** by **journey start.**

#### Required Columns

* **id**
* **journey\_start**
* **journey\_end**

#### Example

|  |  |  |
| --- | --- | --- |
| **id** | **journey\_start** | **journey\_end** |
| 7 | 2019-01-04 23:44:40 | 2049-12-09 04:00:54 |
| 3 | 2019-02-21 22:06:34 | 2049-01-03 11:00:22 |
| ... | ... | ... |

### Extract all pilots

Extract from the database all colonists, which have a **pilot job.** Sort the result by **id, ascending.**

#### Required Columns

* **id**
* **full\_name**

#### Example

|  |  |
| --- | --- |
| **id** | **full\_name** |
| 6 | Clark Cowan |
| 18 | Wald Bim |
| ... | ... |

### Count all colonists that are on technical journey

Count all colonists, that are on **technical journey.**

#### Required Columns

* **Count**

#### Example

|  |
| --- |
| **count** |
| 16 |

### 09.Extract the fastest spaceship

Extract from the database the fastest **spaceship** and its destination **spaceport name**. In other words, the ship with the **highest** light speed rate.

#### Required Columns

* **spaceship\_name**
* **spaceport\_name**

#### Example

|  |  |
| --- | --- |
| **spaceship\_name** | **spaceport\_name** |
| SSE Priestess | Yggdrasil Station |

### 10.Extract spaceships with pilots younger than 30 years

Extract from the database those **spaceships**, which have pilots, **younger** than 30 years old. In other words, 30 years from 01/01/2019. Sort the results **alphabetically** by spaceship **name**.

#### Required Columns

* **name**
* **manufacturer**

#### Example

|  |  |
| --- | --- |
| **name** | **manufacturer** |
| Anarchy | Fivebridge |
| ... | ... |

### 11.Extract all educational mission planets and spaceports

Extract from the database names of all **planets** and their **spaceports**, which have **educational** missions. Sort the results by **spaceport name** in **descending** order.

#### Required Columns

* **planet\_name**
* **spaceport\_name**

#### Example

|  |  |
| --- | --- |
| **planet\_name** | **spaceport\_name** |
| Kascarth | Yggdrasil Station |
| Lescore | Tartarus |
| ... | ... |

### 12.Extract all planets and their journey count

Extract from the database all **planets’ names** and their **journeys count**. Order the results by journeys **count**, **descending** and by **planet name ascending**.

#### Required Columns

* **planet\_name**
* **journeys\_count**

#### Example

|  |  |
| --- | --- |
| **planet\_name** | **journeys\_count** |
| Otroyphus | 4 |
| Eipra | 2 |
| ... | ... |

### 13.Extract the shortest journey

Extract from the database the **shortest** **journey,** its destination **spaceport** **name**, **planet name** and **purpose**.

#### Required Columns

* **Id**
* **planet\_name**
* **spaceport\_name**
* **journey\_purpose**

#### Example

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **planet\_name** | **spaceport\_name** | **journey\_purpose** |
| 3 | Casmadus | Minerva Station | Military |

### 14.Extract the less popular job

Extract from the database the **less popular job** in the **longest journey**. In other words, the job with less assign colonists.

#### Required Columns

* **job\_name**

#### Example

|  |
| --- |
| **job\_name** |
| Engineer |

## Section: Programmability – 30 pts

### 15. Get colonists count

Create a **user defined function** with the name **udf\_count\_colonists\_by\_destination\_planet (planet\_name VARCHAR (30))** that receives **planet name** and returns the count of all colonists sent to that planet.

#### Example

|  |  |
| --- | --- |
| **Query** | |
| SELECT p.name, udf\_count\_colonists\_by\_destination\_planet(‘Otroyphus’) AS count  FROM planets AS p  WHERE p.name = ‘Otroyphus’; | |
| name | count |
| **Otroyphus** | **35** |

### 16. Modify spaceship

Create a **user defined stored procedure** with the name **udp\_modify\_spaceship\_light\_speed\_rate(spaceship\_name VARCHAR(50), light\_speed\_rate\_increse INT(11))** that receives a **spaceship name** and **light speed increase value** and increases spaceship light speed **only if** the given spaceship **exists**. If the modifying is not successful **rollback** any changes and throw an **exception** with **error code ‘45000’** and **message**: “Spaceship you are trying to modify does not exists.”.

#### Example

|  |  |
| --- | --- |
| **Query** | |
| CALL udp\_modify\_spaceship\_light\_speed\_rate (‘Na Pesho koraba’, 1914);  SELECT name, light\_speed\_rate FROM spacheships WHERE name = ‘Na Pesho koraba’ | |
| **Response** | |
| Spaceship you are trying to modify does not exists. | |
| **Query** | |
| CALL udp\_modify\_spaceship\_light\_speed\_rate (‘USS Templar’, 5);  SELECT name, light\_speed\_rate FROM spaceships WHERE name = ‘USS Templar’ | |
| name | light\_speed\_rate |
| **USS Templar** | **11** |