# Database Basics (MSSQL) Exam

# Colonial Journey

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 a strategy called the SoftUnia Initiative. 20000 citizens are send aboard 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.

## Database Overview

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



The **ColonialJourney** 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 **ColonialJourney**. 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**.
* **TravelCards –** 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 are given will be the minimal needed for you to implement the database.

# Section 1. DDL

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

### Planets

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| Id | **Integer,** from **1** to **2,147,483,647.** | Unique table **identificator**, **Identity** |
| Name | **String** up to 30 symbols. Non Unicode | **NULL** is **not** allowed |

### Spaceports

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| Id | **Integer,** from **1** to **2,147,483,647.** | Unique table **identificator**, **Identity** |
| Name | **String** up to 50 symbols. Non Unicode | **NULL** is **not** allowed |
| PlanetId | **Integer,** from **1** to **2,147,483,647.** | **NULL** is **not** allowed, Relationship with table Planets |

### Spaceships

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| Id | **Integer** from **0** to **2,147,483,647** | Unique table **identificator**, **Identity** |
| Name | **String** up to 50 symbols. Non Unicode | **NULL** is **not** allowed |
| Manufacturer | **String** up to 30 symbols. Non Unicode | **NULL** is **not** allowed |
| LightSpeedRate | **Integer** from **0** to **2,147,483,647** | Has a **default value** of 0. |

### Colonists

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| Id | **Integer** from **0** to **2,147,483,647** | Unique table **identificator**, **Identity** |
| FirstName | **String** up to 20 symbols. Non Unicode | **NULL** is **not** allowed |
| LastName | **String** up to 20 symbols. Non Unicode | **NULL** is **not** allowed |
| Ucn | **String** up to 10 symbols. Non Unicode | **NULL** is **not** allowed **UNIQUE** values. |
| BirthDate | **Date** | **NULL** is **not** allowed |

### Journeys

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| Id | **Integer** from **0** to **2,147,483,647** | Unique table **identificator**, **Identity** |
| JourneyStart | **DateTime** | **NULL** is **not** allowed |
| JourneyEnd | **DateTime** | **NULL** is **not** allowed |
| Purpose | **String** up to 11 symbols. Non Unicode | Should **only** contain one of the following purposes: “**Medical**”, “**Technical**”, “**Educational**”, “**Military**”.  **NULL** is **allowed** |
| DestinationSpaceportId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Spaceports. |
| SpaceshipId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Spaceships |

### TravelCards

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| Id | **Integer** from **0** to **2,147,483,647** | Unique table **identificator**, **Identity** |
| CardNumber | A **string** containing exactly **10 characters**.Non Unicode | **NULL** is **not** allowed **UNIQUE** values. |
| JobDuringJourney | **String** up to 8 symbols. Non Unicode | Should **only** contain one of the following jobs: “**Pilot**”, “**Engineer**”, “**Trooper**”, “**Cleaner**”, “**Cook**”.  **NULL** is **allowed** |
| ColonistId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Colonists |
| JourneyId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Journeys |

## Database Design

Submit all of yours **create** **statements** to the **Judge** system.

CREATE TABLE Planets

(

Id INT PRIMARY KEY IDENTITY,

[Name] VARCHAR(30) NOT NULL

)

CREATE TABLE Spaceports

(

Id INT PRIMARY KEY IDENTITY,

[Name] VARCHAR(50) NOT NULL,

PlanetId INT NOT NULL FOREIGN KEY REFERENCES Planets(Id)

)

CREATE TABLE Spaceships

(

Id INT PRIMARY KEY IDENTITY,

[Name] VARCHAR(50) NOT NULL,

Manufacturer VARCHAR(30) NOT NULL,

LightSpeedRate INT DEFAULT 0

)

CREATE TABLE Colonists

(

Id INT PRIMARY KEY IDENTITY,

FirstName VARCHAR(20) NOT NULL,

LastName VARCHAR(20) NOT NULL,

Ucn VARCHAR(10) NOT NULL UNIQUE,

BirthDate DATE NOT NULL

)

CREATE TABLE Journeys

(

Id INT PRIMARY KEY IDENTITY,

JourneyStart DATETIME NOT NULL,

JourneyEnd DATETIME NOT NULL,

Purpose VARCHAR(11) CHECK (Purpose='Medical' OR Purpose='Technical' OR Purpose='Educational'OR Purpose='Military'),

DestinationSpaceportId INT NOT NULL FOREIGN KEY REFERENCES Spaceports(Id),

SpaceshipId INT NOT NULL FOREIGN KEY REFERENCES Spaceships(Id)

)

CREATE TABLE TravelCards

(

Id INT PRIMARY KEY IDENTITY,

CardNumber CHAR(10) NOT NULL UNIQUE,

JobDuringJourney VARCHAR(8) CHECK (JobDuringJourney='Pilot' OR JobDuringJourney='Engineer' OR JobDuringJourney='Trooper' OR JobDuringJourney='Cleaner' OR JobDuringJourney='Cook'),

ColonistId INT NOT NULL FOREIGN KEY REFERENCES Colonists(Id),

JourneyId INT NOT NULL FOREIGN KEY REFERENCES Journeys(Id)

)

# Section 2. DML

**Before you start, you must import “**DataSet-ColonialJourney.sql**”. If you have created the structure correctly, the data should be successfully inserted without any errors.**

In this section, you have to do some data manipulations:

## Insert

**Insert** sample data into the database. Write a query to add the following records into the corresponding tables. **All Ids should be auto-generated**.

**Planets**

|  |
| --- |
| **Name** |
| Mars |
| Earth |
| Jupiter |
| Saturn |

**Spaceships**

|  |  |  |
| --- | --- | --- |
| **Name** | **Manufacturer** | **LightSpeedRate** |
| Golf | VW | 3 |
| WakaWaka | Wakanda | 4 |
| Falcon9 | SpaceX | 1 |
| Bed | Vidolov | 6 |

INSERT INTO Planets ([Name])

VALUES

('Mars'),

('Earth'),

('Jupiter'),

('Saturn')

INSERT INTO Spaceships ([Name], Manufacturer, LightSpeedRate)

VALUES

('Golf', 'VW', 3),

('WakaWaka', 'Wakanda', 4),

('Falcon9', 'SpaceX', 1),

('Bed', 'Vidolov', 6)

## Update

Update all spaceships light speed rate with 1where the **Id** is between **8** and **12.**

UPDATE Spaceships

SET LightSpeedRate=LightSpeedRate+1

WHERE Id BETWEEN 8 AND 12

## Delete

Delete first three inserted **Journeys** (be careful with the relationships).

DELETE TravelCards

WHERE JourneyId IN (1,2,3)

DELETE Journeys

WHERE Id IN (1,2,3)

# Section 3. Querying

**You need to start with a fresh dataset, so recreate your DB and import the sample data again (**DataSet-ColonialJourney.sql**).**

## Select all military journeys

Extract from the database, all **Military** journeys in the format "dd-MM-yyyy". Sort the results **ascending** by **journey start.**

### Required Columns

* **Id**
* **JourneyStart**
* **JourneyEnd**

### Example

|  |  |  |
| --- | --- | --- |
| **Id** | **JourneyStart** | **JourneyEnd** |
| 7 | 04/01/2019 | 09/12/2049 |
| 3 | 21/02/2019 | 03/01/2049 |
| ... | ... | ... |

SELECT Id, FORMAT(JourneyStart,'dd/MM/yyyy') AS JourneyStart,

FORMAT(JourneyEnd,'dd/MM/yyyy') AS JourneyEnd

FROM Journeys

WHERE Purpose='Military'

ORDER BY JourneyStart

## Select all pilots

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

### `Required Columns

* **Id**
* **FullName**

### Example

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

SELECT c.Id, CONCAT(c.FirstName, ' ',c.LastName) AS full\_name

FROM Colonists AS c

JOIN TravelCards AS tc ON c.Id=tc.ColonistId

WHERE tc.JobDuringJourney = 'Pilot'

ORDER BY c.Id

## Count colonists

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

### Required Columns

* **Count**

### Example

|  |
| --- |
| **Count** |
| 16 |

SELECT COUNT(\*) AS [Count]

FROM Colonists AS c

JOIN TravelCards AS tc ON c.Id=tc.ColonistId

JOIN Journeys AS j ON tc.JourneyId=j.Id

WHERE j.Purpose='Technical'

## Select 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 |
| ... | ... |

SELECT s.[Name], s.Manufacturer

FROM Spaceships AS s

JOIN Journeys AS j ON s.Id=j.SpaceshipId

JOIN TravelCards AS tc ON j.Id=tc.JourneyId

JOIN Colonists AS c ON tc.ColonistId=c.Id

WHERE DATEDIFF(year, c.BirthDate,'2019-01-01') <30 AND tc.JobDuringJourney = 'Pilot'

ORDER BY s.[Name]

## Select 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

* **PlanetName**
* **JourneysCount**

### Example

|  |  |
| --- | --- |
| **PlanetName** | **JourneysCount** |
| Otroyphus | 4 |
| Eipra | 2 |
| ... | ... |

SELECT p.[Name], COUNT(j.JourneyStart) AS JourneysCount

FROM Planets AS p

JOIN Spaceports sp ON p.Id = sp.PlanetId

JOIN Journeys j on j.DestinationSpaceportId = sp.Id

GROUP BY p.[Name]

ORDER By JourneysCount DESC, p.[Name]

## Select Second Oldest Important Colonist

Find all colonists and their job during journey with rank 2. Keep in mind that all the selected colonists with rank 2 must be the oldest ones. You can use ranking over their job during their journey.

### Required Columns

* **JobDuringJourney**
* **FullName**
* **JobRank**

### Example

|  |  |  |
| --- | --- | --- |
| **JobDuringJourney** | **FullName** | **JobRank** |
| Cleaner | Hale O'Doireidh | 2 |
| Cook | Laurie Askin | 2 |
| … | … | … |

SELECT k.JobDuringJourney, c.FirstName + ' ' + c.LastName AS FullName, k.JobRank

FROM (

SELECT tc.JobDuringJourney AS JobDuringJourney, tc.ColonistId,

DENSE\_RANK() OVER (PARTITION BY tc.JobDuringJourney ORDER BY co.Birthdate ASC) AS JobRank

FROM TravelCards AS tc

JOIN Colonists AS co ON co.Id = tc.ColonistId

GROUP BY tc.JobDuringJourney, co.Birthdate, tc.ColonistId

) AS k

JOIN Colonists AS c ON c.Id = k.ColonistId

WHERE k.JobRank = 2

ORDER BY k.JobDuringJourney

# Section 4. Programmability

## Get Colonists Count

Create a **user defined function** with the name **dbo.udf\_GetColonistsCount(PlanetName VARCHAR (30))** that receives **planet name** and returns the count of all colonists sent to that planet.

### Example

|  |
| --- |
| **Query** |
| SELECT dbo.udf\_GetColonistsCount('Otroyphus') |
| Count |
| **35** |

-- 11. Get Colonists Count

CREATE FUNCTION dbo.udf\_GetColonistsCount(@PlanetName VARCHAR (30))

RETURNS INT

AS

BEGIN

RETURN (SELECT COUNT(\*)

FROM Journeys AS j

JOIN Spaceports AS s ON s.Id = j.DestinationSpaceportId

JOIN Planets AS p ON p.Id = s.PlanetId

JOIN TravelCards AS tc ON tc.JourneyId = j.Id

JOIN Colonists AS c ON c.Id = tc.ColonistId

WHERE p.[Name] = @PlanetName)

END

## Change Journey Purpose

Create a **user defined stored procedure**, named **usp\_ChangeJourneyPurpose(@JourneyId, @NewPurpose)**, that receives an **journey id** and **purpose**, and attempts to **change the purpose of that journey**. An purpose will only be changed if all of these conditions **pass**:

* If the **journey id** doesn’t exists, then it **cannot be changed.** **Raise an error** with the message “The journey does not exist!”
* If the **journey** has already that purpose, **raise an error** with the message “You cannot change the purpose!”

If all the above conditions pass, **change the purpose of that journey**.

### Example

|  |  |
| --- | --- |
| **Query** | **Output** |
| **EXEC usp\_ChangeJourneyPurpose 4, 'Technical'** |  |
| **EXEC usp\_ChangeJourneyPurpose 2, 'Educational'** | **You cannot change the purpose!** |
| **EXEC usp\_ChangeJourneyPurpose 196, 'Technical'** | **The journey does not exist!** |

CREATE PROCEDURE usp\_ChangeJourneyPurpose(@JourneyId INT, @NewPurpose VARCHAR(11))

AS

BEGIN

IF NOT EXISTS (SELECT 1 FROM Journeys WHERE Id=@JourneyId)

THROW 50010, 'The journey does not exist!', 1;

IF @NewPurpose IN (SELECT Purpose FROM Journeys WHERE Id=@JourneyId )

THROW 50015, 'You cannot change the purpose!', 1;

UPDATE Journeys

SET Purpose=@NewPurpose

WHERE Id=@JourneyId

END