SPACE MISSION DATABASE

Tabitha McIntyre Brown
Elena Mariezcurrena
Apsana Rai
Molly Grenfell
Amy Clark

CODE FIRST GIRLS

WELCOME TO OUR SPACE MISSION DATABASE

OUR SPACE EXPLORATION DATABASE IS DESIGNED TO DOCUMENT AND ORGANIZE INFORMATION RELATED TO VARIOUS SPACE MISSIONS, SPACECRAFS, ASTRONAUTS, PLANETS, AND SCIENTIFIC DISCOVERIES. THIS DATABASE PROVIDES A STRUCTURED WAY TO STORE, RETRIEVE, AND ANALYZE DATA RELATING TO SPACE EXPLORATION, ENABLING USERS TO EXPLORE CONNECTIONS BETWEEN DIFFERENT TABLES. AS MUCH OF THE DATA AS POSSIBLE HAS BEEN BASED ON REAL SPACE MISSIONS!

THE TABLES IN OUR DATABASE ARE NAMED:

MISSIONS
SPACECRAFTS
ASTRONAUTS
PLANETS
& DISCOVERIES

MISSIONS:

The following information is included in the 'missions' table:

Mission ID, Mission Name, Date of Launch, Type of Mission, Mission Status

Input:

```
CREATE TABLE Missions (
    MissionID INT PRIMARY KEY,
    MissionName VARCHAR(255) NOT NULL,
    LaunchDate DATE,
    MissionType VARCHAR(50),
    Status VARCHAR(50)
);
```

Output:

MissionID	MissionName	LaunchDate	MissionType	Status
1	Apollo 11	1969-07-16	Manned Lunar Landing	Completed
2	Voyager 1	1977-09-05	Flyby	Ongoing
3	Curiosity Rover	2011-11-26	Rover	Ongoing
4	Hubble Space Telescope	1990-04-24	Orbiter	Ongoing
5	Mars Pathfinder	1996-12-04	Lander	Completed

SPACECRAFTS:

The following information is included in the 'spacecrafts' table:

Spacecraft ID, Spacecraft Name, Manufacturer,
Launch Vehicle, Mission ID

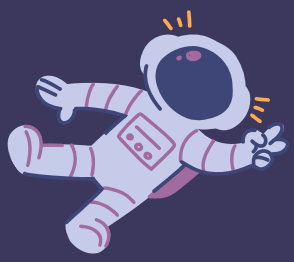
Input:

```
CREATE TABLE Spacecraft (
    SpacecraftID INT PRIMARY KEY,
    SpacecraftName VARCHAR(255) NOT NULL,
    Manufacturer VARCHAR(255),
    LaunchVehicle VARCHAR(255),
    MissionID INT,
    FOREIGN KEY (MissionID) REFERENCES Missions(MissionID)
);
```

Output:

SpacecraftID	SpacecraftName	Manufacturer	LaunchVehicle	MissionID
1	Apollo Lunar Module	Grumman	Saturn V	1
2	Voyager 1 Spacecraft	Jet Propulsion Laboratory	Titan IIIE	2
3	Curiosity Rover	Jet Propulsion Laboratory	Atlas V	3
4	Hubble Space Telescope	Lockheed Martin	Space Shuttle Discovery	4
5	Mars Pathfinder	Jet Propulsion Laboratory	Delta II	5

ASTONAUTS:



The following information is included in the 'astronauts' table:

Astronaut ID, Name, Nationality, Birth date,
Spacecraft ID

Input:

```
CREATE TABLE Astronauts (
    AstronautID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Nationality VARCHAR(50),
    BirthDate DATE,
    SpacecraftID INT,
    FOREIGN KEY (SpacecraftID) REFERENCES Spacecraft(SpacecraftID)
);
```

Output:

AstronautID	Name	Nationality	BirthDate	SpacecraftID
1	Neil Armstrong	American	1930-08-05	1
2	Buzz Aldrin	American	1930-01-20	1
3	Sally Ride	American	1951-05-26	NULL
4	Chris Hadfield	Canadian	1959-08-29	NULL

PLANETS:

The following information is included in the 'planets' table:

Planet ID, Planet Name, Distance From Earth,
Orbital Period

Input:

```
CREATE TABLE Planets (
    PlanetID INT PRIMARY KEY,
    PlanetName VARCHAR(255) NOT NULL,
    DistanceFromEarth FLOAT,
    OrbitalPeriod FLOAT
);
```

Output:

	PlanetID	PlanetName	DistanceFromEarth	OrbitalPeriod
▶	1	Mars	225000000	687
	2	Venus	41000000	225
	3	Jupiter	778500000	4332
	4	Saturn	1433500000	10759
	5	Neptune	4495100000	60190

DISCOVERIES:

The following information is included in the 'discoveries' table:

Discovery ID, Mission ID, Description, Date

Input:

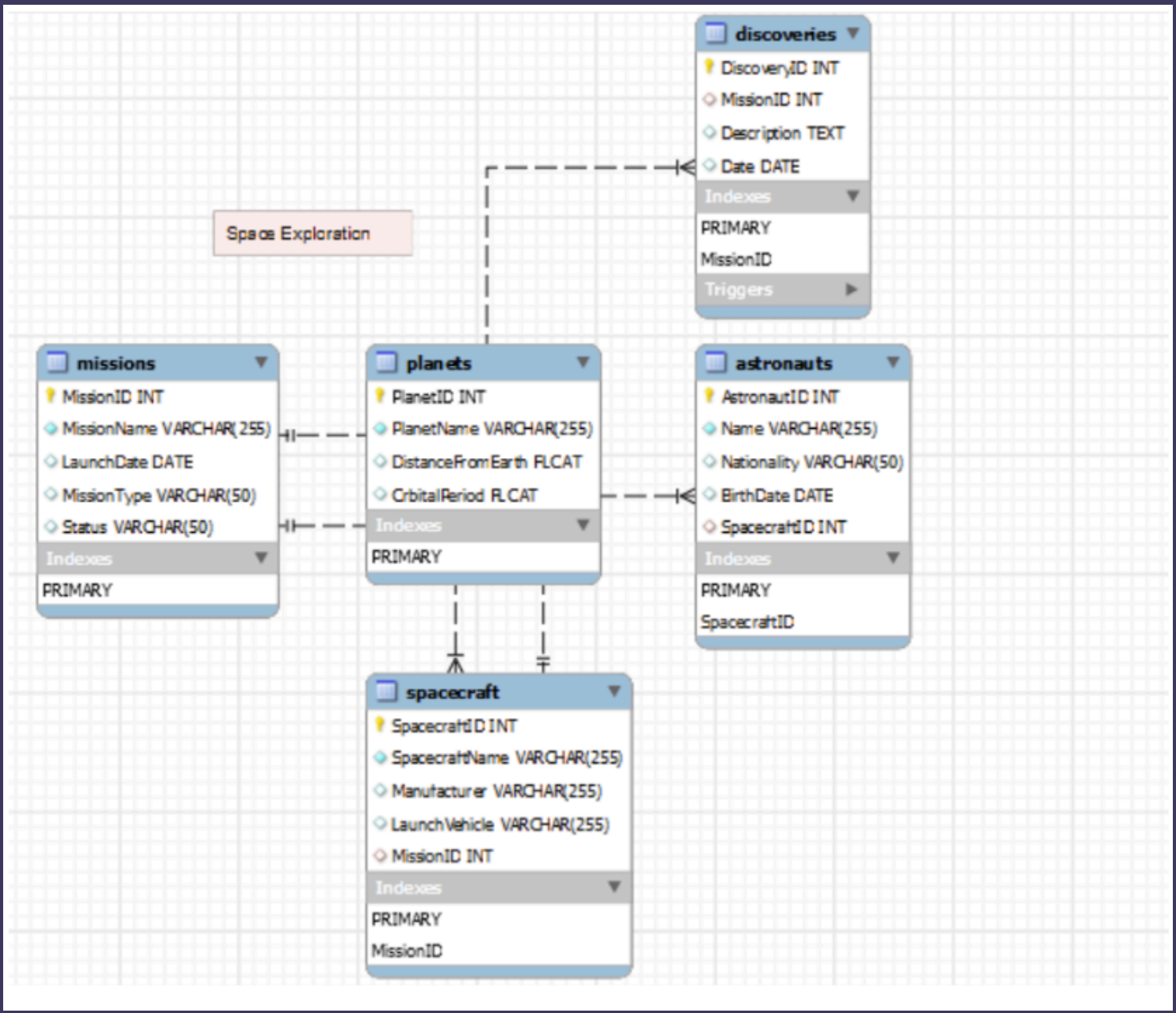
```
CREATE TABLE Discoveries (
    DiscoveryID INT PRIMARY KEY,
    MissionID INT,
    Description TEXT,
    Date DATE,
    FOREIGN KEY (MissionID) REFERENCES Missions(MissionID)
);
```

Output:

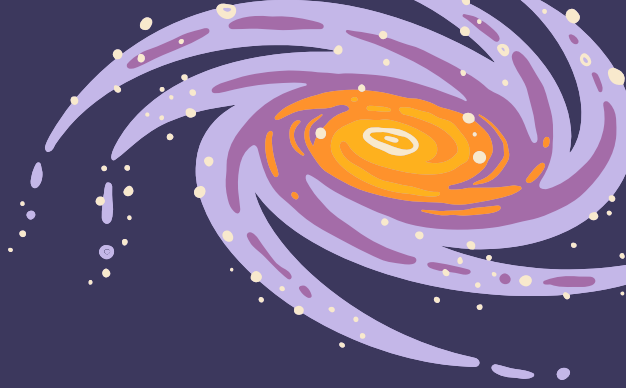
DiscoveryID	MissionID	Description	Date
1	1	First human landing on the Moon	1969-07-20
2	2	Discovered the edge of the solar system (Heliop...	2012-08-25
3	3	Discovered evidence of ancient water on Mars	2012-08-06
4	4	Captured high-resolution images of distant gala...	1990-05-20
5	5	Analyzed the Martian atmosphere	1997-07-04

E-R DIAGRAM:

An ER diagram provides a visual representation of the relationships between all of the tables in the database and gives a quick overview of all the elements in it.



CREATE VIEW:



Using this 'CREATE VIEW' statement will combine data from multiple tables to present an overview of each mission, including the mission name, launch date, mission type, spacecraft name, astronaut names (if any), and any discoveries made during the mission.

```sql
CREATE VIEW MissionDetailsView AS
SELECT
    m.MissionName,
    m.LaunchDate,
    m.MissionType,
    s.SpacecraftName,
    a.Name AS AstronautName,
    d.Description AS DiscoveryDescription,
    d.Date AS DiscoveryDate
FROM
    Missions m
LEFT JOIN
    Spacecraft s ON m.MissionID = s.MissionID
LEFT JOIN
    Astronauts a ON s.SpacecraftID = a.SpacecraftID
LEFT JOIN
    Discoveries d ON m.MissionID = d.MissionID;
```

After creating the view, we can query it like any other table:

```sql
SELECT * FROM MissionDetailsView;
```

MissionName	LaunchDate	MissionType	SpacecraftName	AstronautName	DiscoveryDescription	DiscoveryDate
Apollo 11	1969-07-16	Manned Lunar Landing	Apollo Lunar Module	Neil Armstrong	First human landing on the Moon	1969-07-20
Apollo 11	1969-07-16	Manned Lunar Landing	Apollo Lunar Module	Buzz Aldrin	First human landing on the Moon	1969-07-20
Curiosity Rover	2011-11-26	Rover	Curiosity Rover	NULL	Discovered evidence of ancient water on Mars	2012-08-06
Hubble Space Telescope	1990-04-24	Orbiter	Hubble Space Telescope	NULL	Captured high-resolution images of distant gala...	1990-05-20
Mars Pathfinder	1996-12-04	Lander	Mars Pathfinder	NULL	Analyzed the Martian atmosphere	1997-07-04
Voyager 1	1977-09-05	Flyby	Voyager 1 Spacecraft	NULL	Discovered the edge of the solar system (Heliop...	2012-08-25

STORED FUNCTION:

The stored function uses AVG() to find the average distance from the earth of all the planets in the planets table. This query is then stored in the database for future use.

```
DELIMITER //
CREATE FUNCTION AVERAGE_PLANET_DISTANCE_FROM_EARTH()
RETURNS INT
DETERMINISTIC
BEGIN
  RETURN (SELECT AVG(DISTANCEFROMEARTH) FROM PLANETS);
END //
DELIMITER ;
```

We can then call on the stored function using a SELECT statement like so:

```
SELECT AVERAGE_PLANET_DISTANCE_FROM_EARTH() AS
AvgDistanceFromEarth;
```

Output:

	AvgDistanceFromEarth
▶	1394619981

STORED PROCEDURE:

WHICH SPACECRAFTS ARE CURRENTLY ON ONGOING MISSIONS?

The stored procedure uses a subquery to first find the missions that are ongoing from the missions table, and then to extract which spacecrafts are being used for those missions from the spacecrafts table.

```
DELIMITER //
CREATE PROCEDURE SPACECRAFTS_CURRENTLY_ON_MISSION ()
BEGIN
SELECT SPACECRAFTNAME
FROM SPACECRAFT
WHERE MISSIONID IN (
SELECT MISSIONID
FROM MISSIONS
WHERE STATUS = 'ONGOING');
END //
DELIMITER ;
```

We can then call on the stored procedure like so:

CALL SPACECRAFTS_CURRENTLY_ON_MISSION ();

Output:

SPACECRAFTNAME
Voyager 1 Spacecraft
Curiosity Rover
Hubble Space Telescope

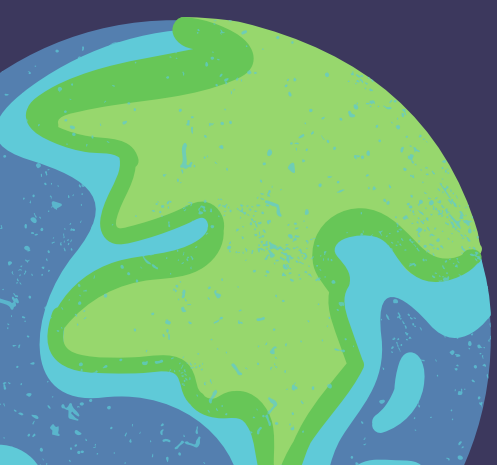
JOINS:

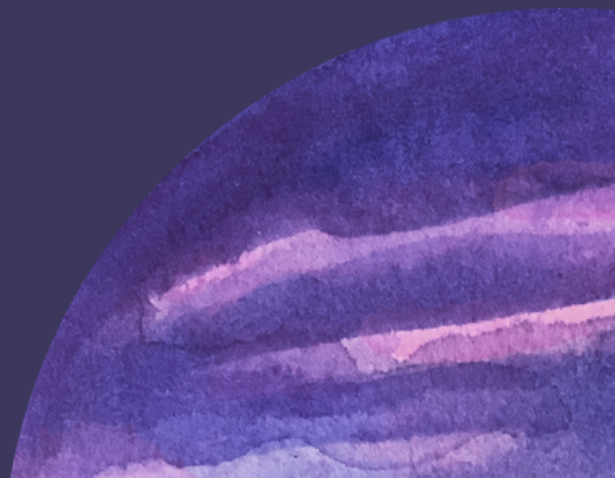
In this example we have used a join to retrieve the names of astronauts, the names of the missions they were involved in, and the status of those missions.

```
SELECT a.Name AS AstronautName, m.MissionName, m.Status
FROM Astronauts a
JOIN Spacecraft s ON a.SpacecraftID = s.SpacecraftID
JOIN Missions m ON s.MissionID = m.MissionID;
```

This works by joining the Astronauts, Spacecraft, and Missions tables together and then returns a list showing which astronauts participated in which missions and the status of those missions.

	AstronautName	MissionName	Status
▶	Neil Armstrong	Apollo 11	Completed
	Buzz Aldrin	Apollo 11	Completed





SUBQUERIES:

In this example we have used a subquery to retrieve the name and launch date of the mission with the earliest launch date.

```
SELECT MissionName, LaunchDate
FROM Missions
WHERE LaunchDate = (
    SELECT MIN(LaunchDate)
    FROM Missions
);
```

Output:

	MissionName	LaunchDate
▶	Apollo 11	1969-07-16

Here is an example with a GROUP BY statement which will retrieve the total number of missions for each type of mission:

```
SELECT MissionType, COUNT(*) AS MissionCount
FROM Missions
GROUP BY MissionType;
```

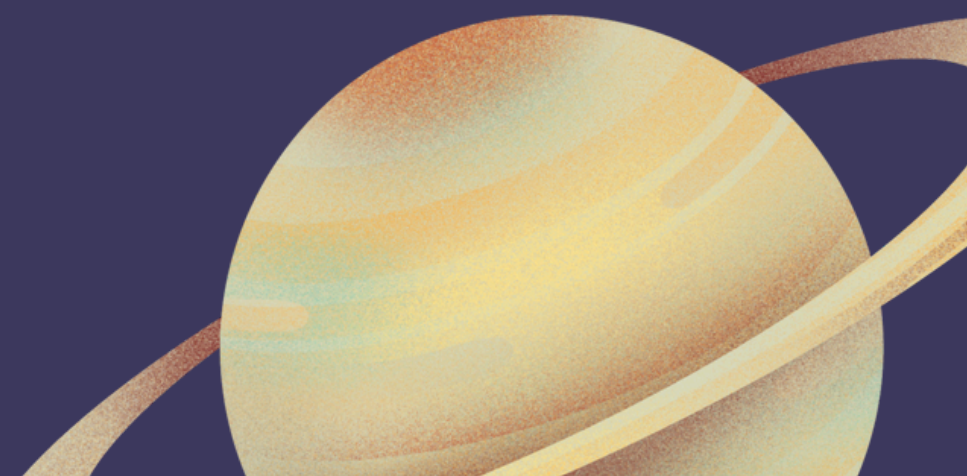
Output:

	MissionType	MissionCount
▶	Manned Lunar Landing	1
	Flyby	1
	Rover	1
	Orbiter	1
	Lander	1

TRIGGER:

Trigger is used to automate procedures in response to certain events - like when a new discovery is made and is used to log details about any discovery made going forwards, effectively saving time. This example logs details when a new discovery is recorded for a mission.

```
DELIMITER $$
CREATE TRIGGER log_discovery_trigger
AFTER INSERT ON Discoveries
FOR EACH ROW
BEGIN
    INSERT INTO Missions (MissionID, Description, Date)
    VALUES (NEW.MissionID, NEW.Description, NEW.Date);
END $$
DELIMITER ;
```



EVENT:

Events are scheduled operations automatically executed at given time or intervals. The following event is set to check and delete completed missions every year from the database.

```
DELIMITER $$
CREATE EVENT delete_completed_missions
ON SCHEDULE EVERY 1 YEAR
DO
BEGIN
    DELETE
    FROM Missions
    WHERE Status = 'completed';
END $$
DELIMITER $$
```

THANK YOU!!

We hope you enjoyed exploring our space database! Did you find the hidden astronaut?