Janbol Jangabyl Data Management

Mars Colony Infrastructure Database

Project Report

December, 13th 2024

1. Requirements Analysis and Conceptual Design

For a sustainable and livable colony on Mars, it is essential to automate operations and minimize human interaction with the harsh environment. Ensuring the colony's safety and sustainability requires reliable data reporting and management systems to track vital life support information. Promptly addressing maintenance issues is critical to maintaining the colony's structural integrity.

Field engineers and scientists will need continuous updates on the status of various sectors and habitats, relying on sensor data, along with information on equipment and vehicles. Monitoring Mars climate to identify extreme weather events that could threaten the habitats is equally vital.

The database system will support colony operations by focusing on:

• Environmental Monitoring
Mars presents a harsh environment with extreme temperatures, dust storms, and radiation. The database will collect data from environmental sensors placed around the colony to track dust levels, wind speed, temperature, radiation, and other environmental factors. This data will help identify conditions that could impact the colony's health and safety.

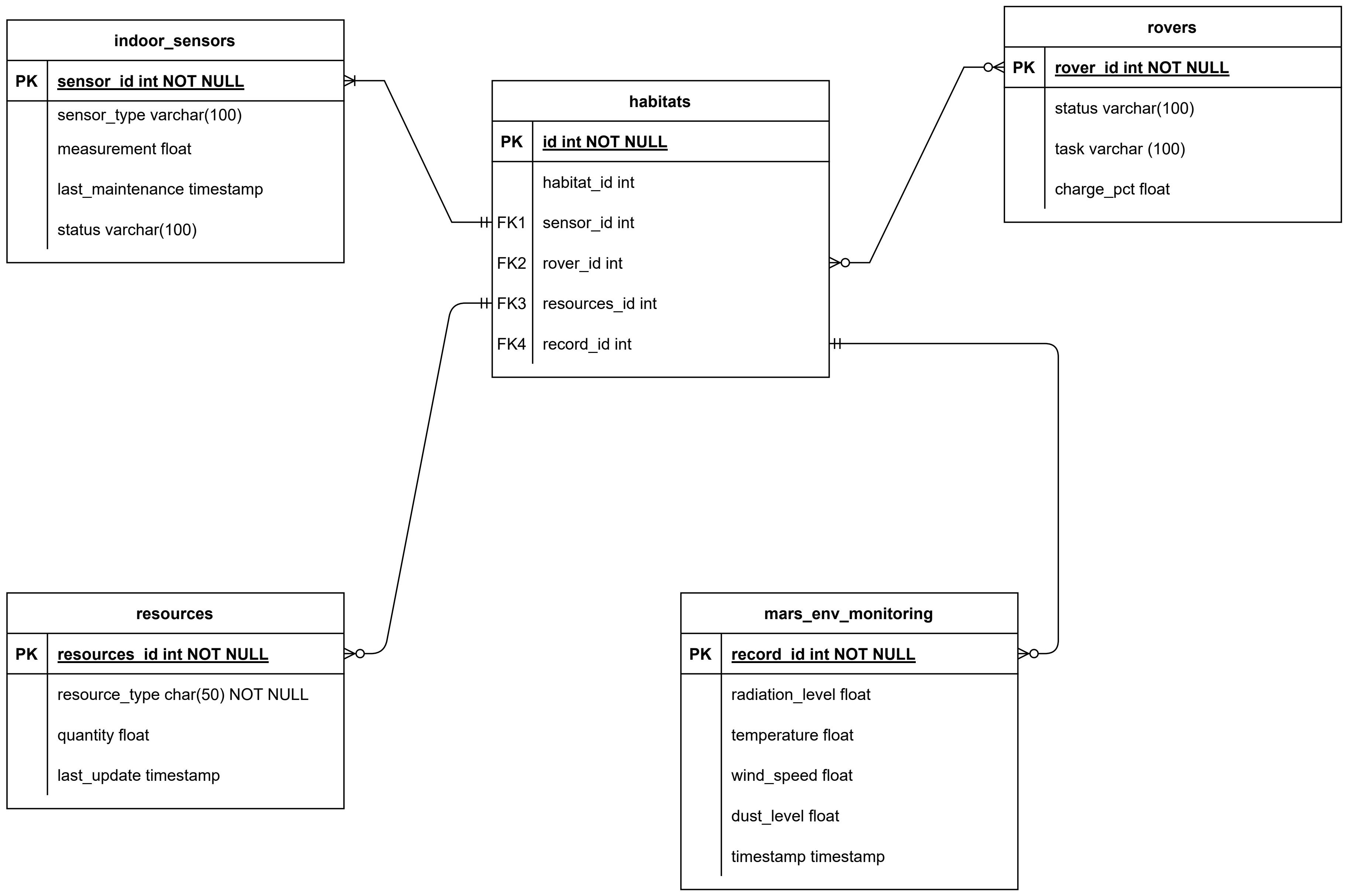
• Habitat Maintenance
Maintaining the habitats and life support systems in optimal condition is vital for the colony's survival. The database will store sensor data on habitat conditions, including structural, radiation levels, and internal pressure. This information will allow for early detection of issues and enable timely maintenance, inspections, repairs, or replacements, ensuring a safe living environment for the colonists.

• Resource Management
The database will track essential resources like oxygen, water, and food, ensuring accurate monitoring of their availability and consumption. Real-time updates on resource levels and usage rates will help with efficient management and prevent shortages, which are crucial for the colony's sustainability.

• Transportation Management
Efficient transportation between the colony's sectors is critical for logistics and safety, particularly for outdoor research and exploration. The database will manage information about transportation vehicles, including their operational status, battery levels, and assigned tasks. By tracking each vehicle's location (i.e., which habitat it is stationed at) and condition, the system will optimize energy consumption, ensure safety, and streamline transportation logistics across the colony.

Entity-Relationship (ER) Diagram:

The following ER diagram illustrates the database structure for this project. Since the colony will likely be centered around the living habitats, the database is designed accordingly. Habitats are the core of the colony, as they will house the human inhabitants. Consequently, many of the data are organized and linked to the habitats to ensure comprehensive monitoring of the environment and life support systems.


indoor_sensors
PK
sensor_id int NOT NULL
sensor_type varchar(100)
measurement float
last_maintenance timestamp
status varchar(100)
habitats
PK
id int NOT NULL
habitat_id int
FK1
sensor_id int
FK2
rover_id int
FK3
resources_id int
FK4
record_id int
rovers
PK
rover_id int NOT NULL
status varchar(100)
task varchar (100)
charge_pct float
resources
PK
resources_id int NOT NULL
resource_type char(50) NOT NULL
quantity float
last_update timestamp
mars_env_monitoring
PK
record_id int NOT NULL
radiation_level float
temperature float
wind_speed float
dust_level float
timestamp timestamp

2. Logical Design and Normalization

Following shows the relational schema, including its constraints, data type, and relationships.

habitats:
- id (Primary Key) – NOT NULL
- habitat_id (INTEGER) –
- sensor_id (Foreign key referencing indoor_sensors.sensor_id) NOT NULL
- rover_id (Foreign key referencing rovers.rover_id) NOT NULL
- resources_id (Foreign key referencing resources.resource_id) NOT NULL
- record_id (Foreign key referencing mars_env_monitoring.record_id) NOT NULL

indoor_sensors:
- sensor_id (Primary Key) NOT NULL
- sensor_type (VARCHAR)
- measurement (FLOAT)
- last_maintenance (TIMESTAMP)
- status (VARCHAR)

rovers:
- rover_id (Primary Key) NOT NULL
- status (VARCHAR)
- task (VARCHAR)
- charge_pct (FLOAT)

resources:
- resource_id (Primary Key) NOT NULL
- resource_type (VARCHAR)
- quantity (FLOAT)
- last_update (TIMESTAMP)

mars_env_monitoring:
- record_id (Primary Key) NOT NULL
- radiation_level (FLOAT)
- temperature (FLOAT)
- wind_speed (FLOAT)
- dust_level (FLOAT)
- timestamp (TIMESTAMP)

Relationships:

Table	Related Table	Relationship Type
habitats	indoor_sensors	One Mandatory–to–Many Mandatory
habitats	rovers	Many Optional–to–Many Optional
habitats	resources	One Mandatory–to–Many Optional
habitats	mars_env_monitoring	One Mandatory–to–Many Optional

Following shows 3NF tables. Note that full tables are not included:

habitats

id	habitat_id	sensor_id	rover_id	resources_id	record_id
1	1	1	1	3	5
2	2	2	2	4	6
3	3	3	4	12	1
4	4	10	5	11	3
5	4	4	6	14	17
6	5	18	11	13	5

indoor_sensors

sensor_id	sensor_type	measurement	last_maintenance	status
1	Temperature	22	2023-12-15 21:11:04.293263	Needs Maintenance
2	Atmospheric	101.8	2024-11-04 21:11:04.293271	Active
3	Temperature	24	2024-11-16 21:11:04.293272	Needs Maintenance
4	Pressure	96.2	2024-01-06 21:11:04.293273	Active
5	Radiation	100.0	2024-08-15 21:11:04.293274	Active
6	Radiation	95.3	2024-07-14 21:11:04.293275	Active
7	Pressure	104.1	2024-10-30 21:11:04.293275	Active
8	Radiation	97.6	2024-08-12 21:11:04.293276	Active

rovers

rover_id	status	task	charge_pct
1	Operational	Material Collection	81.2
2	Operational	Exploration	66.5
3	Operational	Transport	53.2
4	Operational	Material Collection	65.5

mars_env_monitoring

record_id	radiation_level	temperature	wind_speed	dust_level	timestamp
1	1.42	-56.1	85.8	202.3	2024-12-08 00:11:04.293990
2	0.94	-44.8	32.6	443.9	2024-12-09 10:11:04.293992
3	1.07	-45.2	22.0	425.5	2024-12-09 01:11:04.293994
4	1.33	-18.2	71.1	467.8	2024-12-10 06:11:04.293995
5	0.97	-17.3	81.0	392.7	2024-12-09 07:11:04.293996

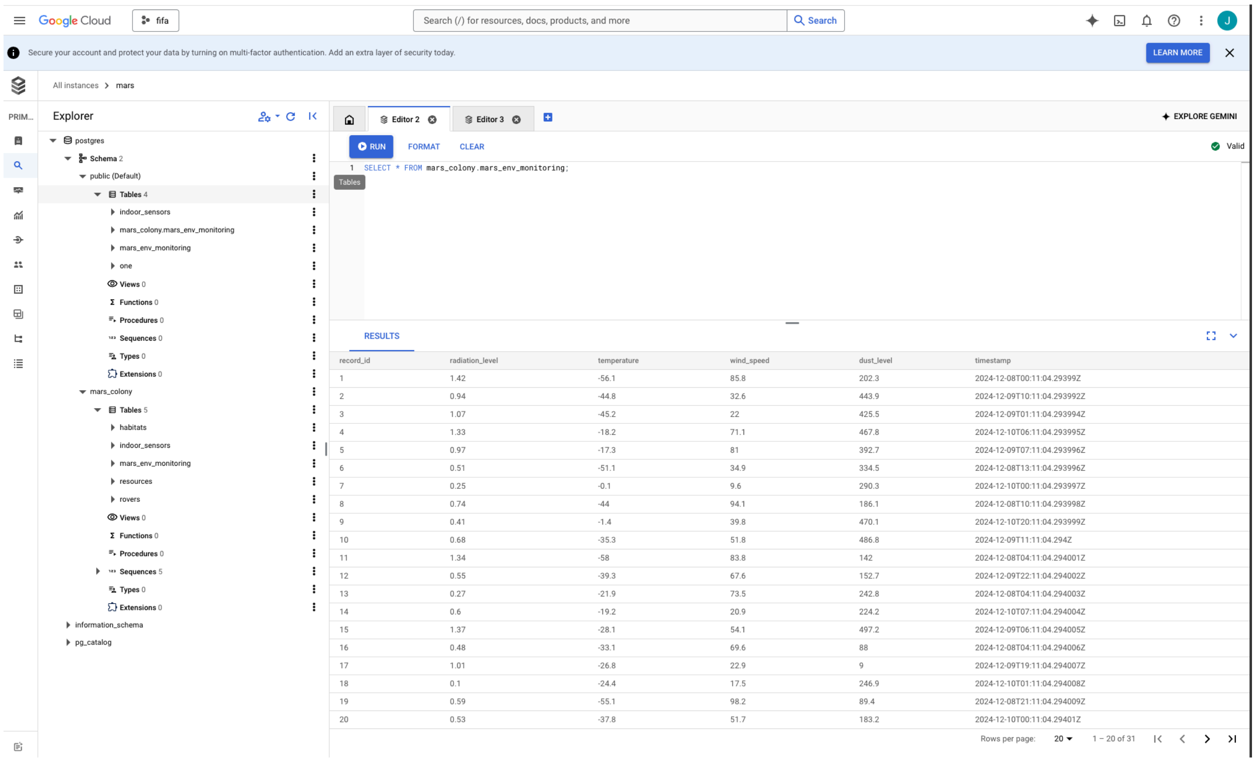
resources

resource_id	resource_type	quantity	last_update
1	Food	312.39	2024-12-10 03:11:04.293792
2	Energy	330.46	2024-12-10 16:11:04.293795
3	Energy	136.39	2024-12-10 07:11:04.293796
4	Water	739.6	2024-12-09 08:11:04.293797

3. Implementation and Data Insertion

For implementation and data insertion please see the mars_database.ipynb file.

Screenshot below shows schema mars_colony in Postgres DB running on CloudSQL which is accessible from any IP.



```sql
SELECT * FROM mars_colony.mars_env_monitoring;
```

record_id	radiation_level	temperature	wind_speed	dust_level	timestamp
1	1.42	-56.1	85.8	202.3	2024-12-08T00:11:04.29399Z
2	0.94	-44.8	32.6	443.9	2024-12-09T10:11:04.293992Z
3	1.07	-45.2	22	425.5	2024-12-09T01:11:04.293994Z
4	1.33	-18.2	71.1	467.8	2024-12-10T06:11:04.293995Z
5	0.97	-17.3	81	392.7	2024-12-09T07:11:04.293996Z
6	0.51	-51.1	34.9	334.5	2024-12-08T13:11:04.293996Z
7	0.25	-0.1	9.6	290.3	2024-12-10T00:11:04.293997Z
8	0.74	-44	94.1	186.1	2024-12-08T10:11:04.293998Z
9	0.41	-1.4	39.8	470.1	2024-12-10T20:11:04.293999Z
10	0.68	-35.3	51.8	486.8	2024-12-09T11:11:04.294Z
11	1.34	-58	83.8	142	2024-12-08T04:11:04.294001Z
12	0.55	-39.3	67.6	152.7	2024-12-09T22:11:04.294002Z
13	0.27	-21.9	73.5	242.8	2024-12-08T04:11:04.294003Z
14	0.6	-19.2	20.9	224.2	2024-12-10T07:11:04.294004Z
15	1.37	-28.1	54.1	497.2	2024-12-09T06:11:04.294005Z
16	0.48	-33.1	69.6	88	2024-12-08T04:11:04.294006Z
17	1.01	-26.8	22.9	9	2024-12-09T19:11:04.294007Z
18	0.1	-24.4	17.5	246.9	2024-12-10T01:11:04.294008Z
19	0.59	-55.1	98.2	89.4	2024-12-08T21:11:04.294009Z
20	0.53	-37.8	51.7	183.2	2024-12-10T00:11:04.29401Z

Rows per page: 20 1 – 20 of 31

4. Query Development and Reporting

For query development and reporting, please see the sql_queries_set.ipynb file.

In addition to the sql_queries_set.ipynb all the SQL retrieval queries and reports have been implemented on MAC Terminal with connection to the CloudSQL Postgres database and the outputs are attached in the next page.

```
Last login: Fri Dec 13 19:34:32 on ttys016
janbol@Janbols-MacBook-Pro-2 ~ % psql -h 34.85.134.242 -p 5432 -U postgres -d postgres

Password for user postgres:
psql (14.15 (Homebrew), server 16.4)
WARNING: psql major version 14, server major version 16.
         Some psql features might not work.
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off
)
Type "help" for help.

postgres=> SELECT habitat_id, habitats.rover_id, status, task, charge_pct
FROM mars_colony.habitats
LEFT JOIN mars_colony.rovers
ON habitats.rover_id = rovers.rover_id
WHERE rovers.status = 'Needs Maintenance';
 habitat_id | rover_id |      status       |        task         | charge_pct
------------+----------+-------------------+---------------------+------------
          4 |        6 | Needs Maintenance | Material Collection |         83
          2 |       10 | Needs Maintenance | Transport           |       66.3
(2 rows)

postgres=> SELECT habitat_id, indoor_sensors.sensor_id, sensor_type, status
FROM mars_colony.habitats
LEFT JOIN mars_colony.indoor_sensors
ON habitats.sensor_id = indoor_sensors.sensor_id
WHERE indoor_sensors.status = 'Needs Maintenance' and indoor_sensors.sensor_type='Temperature'
;
 habitat_id | sensor_id | sensor_type |      status
------------+-----------+-------------+-------------------
          1 |         1 | Temperature | Needs Maintenance
          3 |         3 | Temperature | Needs Maintenance
          3 |         3 | Temperature | Needs Maintenance
(3 rows)

postgres=> SELECT habitat_id, dust_level, timestamp
FROM mars_colony.habitats
LEFT JOIN mars_colony.mars_env_monitoring
ON habitats.record_id = mars_env_monitoring.record_id
WHERE dust_level > 300;
 habitat_id | dust_level |         timestamp
------------+------------+----------------------------
          1 |      392.7 | 2024-12-09 07:11:04.293996
          2 |      334.5 | 2024-12-08 13:11:04.293996
          4 |      425.5 | 2024-12-09 01:11:04.293994
          5 |      392.7 | 2024-12-09 07:11:04.293996
          3 |      470.1 | 2024-12-10 20:11:04.293999
          1 |      497.2 | 2024-12-09 06:11:04.294005
(6 rows)

postgres=> SELECT habitat_id, resource_type, quantity
FROM mars_colony.habitats
LEFT JOIN mars_colony.resources
ON habitats.resource_id = resources.resource_id
WHERE resource_type = 'Water';
 habitat_id | resource_type | quantity
------------+---------------+----------
          2 | Water         |    739.6
          4 | Water         |   725.96
          3 | Water         |   282.76
```

```
(3 rows)

postgres=> SELECT habitat_id, rovers.rover_id, task, charge_pct
FROM mars_colony.habitats
LEFT JOIN mars_colony.rovers
ON habitats.rover_id = rovers.rover_id
WHERE task = 'Exploration';
 habitat_id | rover_id |    task     | charge_pct
------------+----------+-------------+------------
          2 |        2 | Exploration |       66.5
          1 |        7 | Exploration |       66.5
(2 rows)

postgres=> SELECT habitat_id, rovers.rover_id, status, task, charge_pct
FROM mars_colony.habitats
LEFT JOIN mars_colony.rovers
ON habitats.rover_id = rovers.rover_id
WHERE charge_pct > 70;
 habitat_id | rover_id |      status       |        task         | charge_pct
------------+----------+-------------------+---------------------+------------
          1 |        1 | Operational       | Material Collection |       81.2
          4 |        6 | Needs Maintenance | Material Collection |         83
          2 |        9 | Operational       | Material Collection |         99
(3 rows)

postgres=> SELECT habitats.habitat_id, COUNT(rovers.rover_id)
FROM mars_colony.habitats
LEFT JOIN mars_colony.rovers
ON habitats.rover_id = rovers.rover_id
GROUP BY habitats.habitat_id;
 habitat_id | count
------------+-------
          3 |     3
          5 |     1
          4 |     2
          2 |     3
          1 |     2
(5 rows)

postgres=> SELECT habitat_id, radiation_level, temperature, wind_speed, dust_level
FROM mars_colony.habitats
LEFT JOIN mars_colony.mars_env_monitoring
ON habitats.record_id = mars_env_monitoring.record_id;
 habitat_id | radiation_level | temperature | wind_speed | dust_level
------------+-----------------+-------------+------------+------------
          1 |            0.97 |       -17.3 |         81 |      392.7
          2 |            0.51 |       -51.1 |       34.9 |      334.5
          3 |            1.42 |       -56.1 |       85.8 |      202.3
          4 |            1.07 |       -45.2 |         22 |      425.5
          4 |            1.01 |       -26.8 |       22.9 |          9
          5 |            0.97 |       -17.3 |         81 |      392.7
          2 |            0.25 |        -0.1 |        9.6 |      290.3
          3 |            0.41 |        -1.4 |       39.8 |      470.1
          2 |            0.41 |         -51 |         70 |      294.9
          3 |            0.67 |       -18.3 |       85.6 |      243.4
          1 |            1.37 |       -28.1 |       54.1 |      497.2
(11 rows)

postgres=> SELECT habitats.habitat_id, SUM(resources.quantity) as total_acc_food
            FROM mars_colony.habitats
            LEFT JOIN mars_colony.resources
            ON habitats.resource_id = resources.resource_id
            WHERE resources.resource_type = 'Food'
            GROUP BY habitats.habitat_id;
 habitat_id | total_acc_food
------------+----------------
          1 |         312.39
          3 |          720.8
(2 rows)

postgres=> SELECT habitats.habitat_id, indoor_sensors.sensor_id, indoor_sensors.status,  rover
s.rover_id, rovers.status
            FROM mars_colony.habitats
            LEFT JOIN mars_colony.indoor_sensors ON habitats.sensor_id = indoor_sensors.sensor
_id and indoor_sensors.status = 'Needs Maintenance'

            LEFT JOIN mars_colony.rovers ON habitats.rover_id = rovers.rover_id and rovers.sta
tus = 'Needs Maintenance';
 habitat_id | sensor_id |      status       | rover_id |      status
------------+-----------+-------------------+----------+-------------------
          1 |         1 | Needs Maintenance |          |
          2 |           |                   |          |
          3 |         3 | Needs Maintenance |          |
          4 |        10 | Needs Maintenance |          |
          4 |           |                   |        6 | Needs Maintenance
          5 |           |                   |          |
          2 |        17 | Needs Maintenance |       10 | Needs Maintenance
          3 |         3 | Needs Maintenance |          |
          2 |           |                   |          |
          3 |           |                   |          |
          1 |           |                   |          |
(11 rows)

(END)
 habitat_id | sensor_id |      status       | rover_id |      status
------------+-----------+-------------------+----------+------------------
          1 |         1 | Needs Maintenance |          |
          2 |           |                   |          |
          3 |         3 | Needs Maintenance |          |
          4 |        10 | Needs Maintenance |          |
          4 |           |                   |        6 | Needs Maintenace
          5 |           |                   |          |
          2 |        17 | Needs Maintenance |       10 | Needs Maintenace
          3 |         3 | Needs Maintenance |          |
          2 |           |                   |          |
          3 |           |                   |          |
          1 |           |                   |          |
(11 rows)

~
~
~
~
~
~
~
~
~
~
~
```

5. Optimization and Final Documentation

As part of the optimization process, several columns from the selected tables, which are expected to be used more frequently, were indexed, as shown below. The corresponding SQL queries are available in the sql_queries_set.ipynb file. By indexing specific columns, data retrieval speed is improved, as the database uses the index rather than scanning each row to match the query criteria. While indexing enhances retrieval performance, it can slow down data insertion, updates, and deletions, as the index must be updated whenever these operations occur.

```
# Indexing `status` column in `indoor_sensors`

CREATE INDEX idx_status on mars_colony.indoor_sensors(status);



Verify the Index:
('indoor_sensors_pkey', 'CREATE UNIQUE INDEX indoor_sensors_pkey ON
mars_colony.indoor_sensors USING btree (sensor_id)')
('idx_status', 'CREATE INDEX idx_status ON mars_colony.indoor_sensors USING btree
(status)’)
```

```
# Indexing `status` column in `rovers` table

CREATE INDEX rovers_status on mars_colony.rovers(status);

Verify the Index:

('rovers_pkey', 'CREATE UNIQUE INDEX rovers_pkey ON mars_colony.rovers USING btree
(rover_id)')
('rovers_status', 'CREATE INDEX rovers_status ON mars_colony.rovers USING btree
(status)’)
```

```
# indexing `dust_level` column in `mars_env_monitoring`

CREATE INDEX dust_lvl on mars_colony.mars_env_monitoring(dust_level);

('rovers_pkey', 'CREATE UNIQUE INDEX rovers_pkey ON mars_colony.rovers USING btree
(rover_id)')
('rovers_status', 'CREATE INDEX rovers_status ON mars_colony.rovers USING btree
(status)’)
```

How to use the database and general instructions

The database is hosted on CloudSQL, with a PostgreSQL setup configured to allow access from any machine with a public IP address. While this configuration is not typical in industry, where access is usually restricted based on various factors for security, it was done here for proof-of-concept purposes. Allowing any public IP to access and modify the data would be a high-security risk in a real-world scenario.

This database (`mars_colony`) consists of five distinct entities: `habitats, indoor_sensors, mars_env_monitoring, resources, and rovers`. Each table contains different sets of data, aimed at helping engineers and colonists on Mars track, monitor, and analyze vital information for mission success.

a. `habitats`: Contains references related to the condition of each habitat.
b. `indoor_sensors`: Holds data on temperature, pressure, radiation, and other environmental factors that need to be closely monitored, as humans will be living inside without specialized suits.
c. `mars_env_monitoring`: Stores data from sensors installed outside the habitats, helping technicians and field engineers monitor Mars' ever-changing weather conditions, which could influence outdoor maintenance and construction work and their safety.
d. `resources`: Tracks essential supplies such as energy, water, and food. This data might be helpful to schedule the replenishment and shipping of resources based on their depletion.
e. `rovers`: Monitors the status of rovers, including their locations, charge levels, and operational status.

To access the database, psycopg2 (a Python library) can be used. However, for this proof of concept, it's recommended to connect via the terminal using the credentials provided below.

Credentials:

- Host ID/Instance ID: 34.85.134.242
- Database name: postgres
- User: postgres
- password: mars

Connection Instructions:

To connect to the database on CloudSQL, please run the following code snippet and enter the password when it is prompted:

```
psql -h 34.85.134.242 -p 5432 -U postgres -d postgres
```

After a successful connection, SQL data retrieval and report queries can be run, as detailed in the attached documents.

NOTE: It is not recommended to run the `mars_database.ipynb` file. Instead, use `sql_queries_set.ipynb` to retrieve data using `psycopg2`.