DBMS MINI PROJECT

Space Research Station Management System (SRSMS)

Team

NAME	SRN
SURABHI M	PES1UG23AM325
SMRITHI A S	PES1UG23AM306
TANUU SHREE M	PES1UG23AM336
GAHNAVI B	PES1UG23AM900

Abstract

The Space Research Station Management System (SRSMS) is designed to efficiently manage astronauts, missions, experiments, and supply allocations in a simulated space station database. The system offers a GUI built using Python's ttkbootstrap library (Cyborg theme) integrated with a MySQL backend. Users can perform CRUD operations, execute stored procedures, and trigger audit logging to monitor database activities in real time.

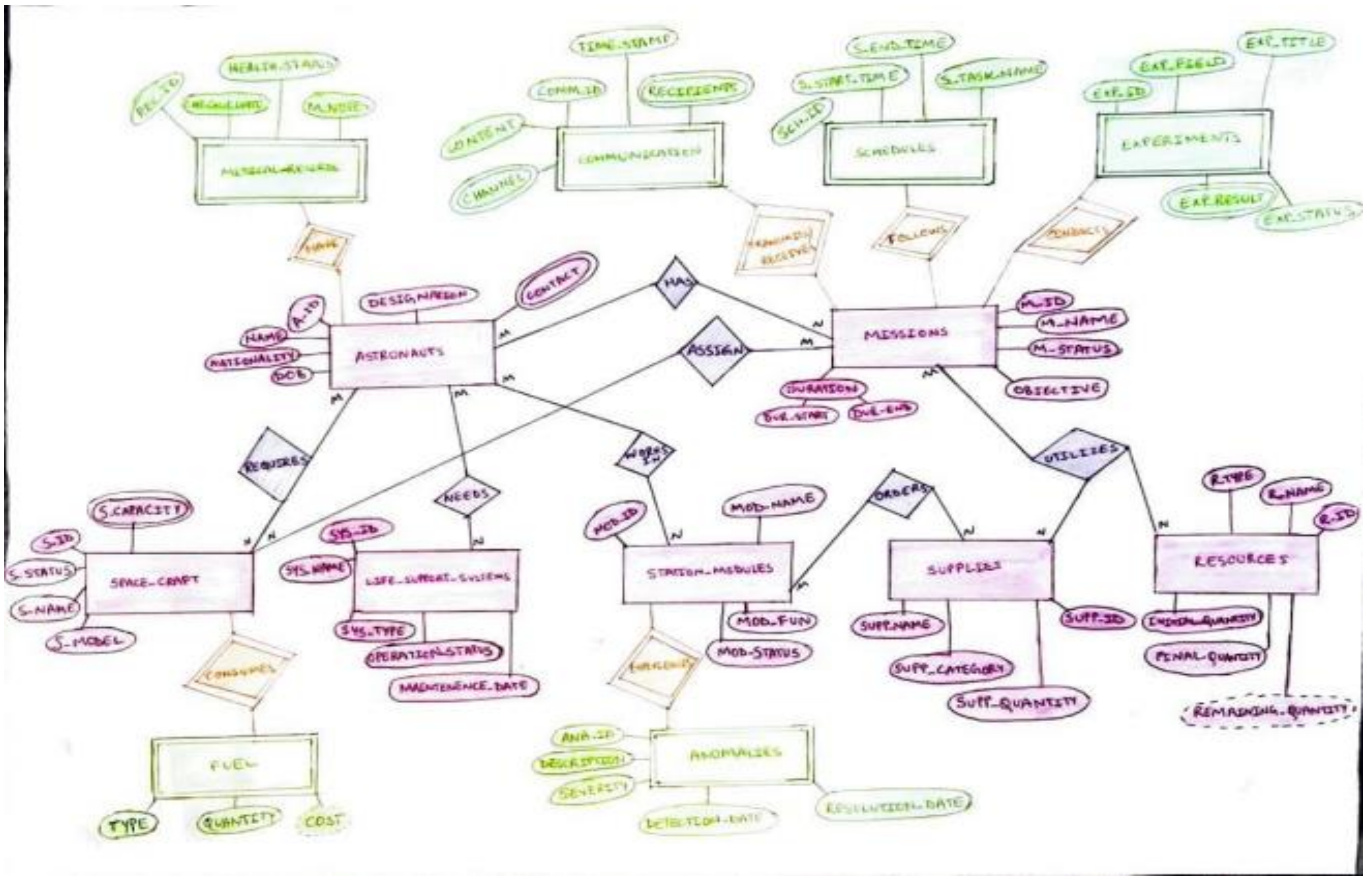
User Requirement Specification

1. The system must support three user roles: *Admin, Operator, and Viewer.*
2. Admins can perform full CRUD operations and manage audit logs.
3. Operators can insert and update data but cannot delete.
4. Viewers can only view database records.
5. Each table modification must be recorded in the Audit_Log table using triggers.
6. The GUI must allow viewing, inserting, updating, and exporting table data.

Tools / Programming Languages Used

1. Python 3.11
2. ttkbootstrap (Cyborg theme)
3. MySQL Workbench 8.0
4. mysql-connector-python
5. pandas (for CSV export)
6. tkinter for GUI layout

ER Diagram



Relational Schema

Step I :- Mapping Strong Entities

Astronauts

A_ID	name	nationality	designation	DOB	contact

Missions

M_ID	M_name	M_status	objectives	dur_start	dur_end

Spacecraft

S_ID	S_status	S_name	S_model	S_capacity

Station_modules

Mod_ID	Mod_name	Mod_fun	Mod_status

Life_Support_Systems

Sys_ID	sys_name	Sys_type	operat?_status	maintenance_date

Resources

R_ID	R_name	R_type	initial_quantity	final_quantity	remaining_quantity

Supplies

Supp_ID	Supp_name	Supp_category	Supp_quantity

Step II: Mapping Weak Entities

Experiments

Exp_ID	Exp_yield	Exp_field	Exp_title	Exp_result	Exp_status	M_ID

Schedules

Sch_ID	S_start_time	S_end_time	S_task_name	M_ID

Medical Records

Rec_ID	checkup_date	health_status	M_notes	A_ID

Anomalies

Ano_ID	description	severity	detection_date	resolution_date	Mod_ID

Communication

Comm_ID	channel	Receipients	content	time_stamp	M_ID

Fuel

Type	Quantity	cost	S_ID

Step III : Mapping binary 1:1 relationship types

None [fuel in weak entity]

Step IV :- Mapping 1:N relationships

Life_support_systems

Sys_ID	sys_name	Sys_type	ops_status	maint_date	Mod_ID

Step V :- Mapping M:N relationships

Has

A_ID	M_ID

Assign

M_ID	S_ID

Requires

A_ID	S_ID

Works_In

A_ID	Mod_ID

Utilizes

M_ID	R_ID	Supp_ID

needs

A_ID	Sys_ID

Orders

Mod_ID	Supp_ID

Step VI :- Multivalued attributes

Astro_no

A_ID	A_no

Spacecraft_capacity

S_ID	S_capacity

Exp_results

M_ID	Exp_result

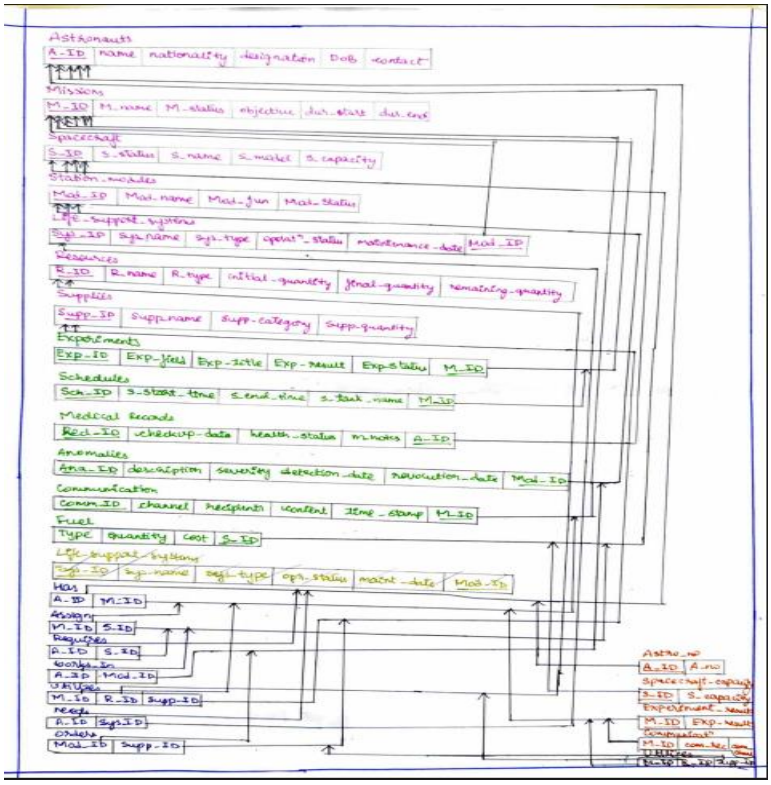
Communication

M_ID	comm_receptients	comm_channel

Step VII : N-array relationships

Utilizes

M_ID	R_ID	Supp_ID



DDL Commands

-- ==========

-- 1) Strong entities

-- ==========

```
CREATE TABLE Astronauts (
  AstronautID    INT AUTO_INCREMENT PRIMARY KEY,
  FirstName      VARCHAR(100) NOT NULL,
  LastName       VARCHAR(100) NOT NULL,
  DOB            DATE,
  Nationality    VARCHAR(80),
  JobTitle       VARCHAR(80),
  MedicalStatus  VARCHAR(60)
) ENGINE=InnoDB;
```

```
CREATE TABLE AstronautSkills (
  AstronautID INT NOT NULL,
  Skill       VARCHAR(100) NOT NULL,
  PRIMARY KEY (AstronautID, Skill),
  CONSTRAINT fk_as_sk_as FOREIGN KEY (AstronautID) REFERENCES
Astronauts(AstronautID) ON DELETE CASCADE
) ENGINE=InnoDB;

CREATE TABLE Missions (
  MissionID      INT AUTO_INCREMENT PRIMARY KEY,
  MissionName    VARCHAR(200) NOT NULL UNIQUE,
  LaunchDate     DATE,
  ReturnDate     DATE,
  MissionType    VARCHAR(80),
  CurrentStatus  VARCHAR(20) DEFAULT 'Planned'
) ENGINE=InnoDB;

CREATE TABLE StationModules (
  ModuleID      INT AUTO_INCREMENT PRIMARY KEY,
  ModuleName    VARCHAR(150) NOT NULL UNIQUE,
  ModuleType    VARCHAR(80),
  Capacity      INT UNSIGNED,
  CurrentStatus VARCHAR(40),
  Location      VARCHAR(120)
) ENGINE=InnoDB;
```

```
CREATE TABLE Resources (
  ResourceID   INT AUTO_INCREMENT PRIMARY KEY,
  ResourceName VARCHAR(150) NOT NULL,
  Unit         VARCHAR(50),
  Description  TEXT
) ENGINE=InnoDB;

CREATE TABLE Supplies (
  SupplyID       INT AUTO_INCREMENT PRIMARY KEY,
  ResourceID     INT NOT NULL,
  Quantity       DECIMAL(18,3) NOT NULL DEFAULT 0,
  Unit           VARCHAR(50),
  ExpiryDate     DATE,
  SupplierName   VARCHAR(150),
  StorageModuleID INT,
  CONSTRAINT fk_supplies_resource FOREIGN KEY (ResourceID) REFERENCES
Resources(ResourceID) ON DELETE CASCADE,
  CONSTRAINT fk_supplies_module FOREIGN KEY (StorageModuleID) REFERENCES
StationModules(ModuleID) ON DELETE SET NULL
) ENGINE=InnoDB;

CREATE TABLE LifeSupportSystems (
  SystemID    INT AUTO_INCREMENT PRIMARY KEY,
  ModuleID    INT NOT NULL UNIQUE,
  SystemType  VARCHAR(80),
  OxygenLevel DECIMAL(10,3),
  Pressure    DECIMAL(10,3),
```

```
  Temperature DECIMAL(6,2),
  CO2Level    DECIMAL(10,3),
  CurrentStatus VARCHAR(40),
  CONSTRAINT fk_lss_module FOREIGN KEY (ModuleID) REFERENCES
StationModules(ModuleID) ON DELETE CASCADE
) ENGINE=InnoDB;

CREATE TABLE Spacecrafts (
  SpacecraftID INT AUTO_INCREMENT PRIMARY KEY,
  Name        VARCHAR(150) NOT NULL UNIQUE,
  SystemType   VARCHAR(80),
  CrewCapacity INT UNSIGNED,
  CargoCapacity DECIMAL(12,2),
  LaunchDate   DATE,
  CurrentStatus VARCHAR(40)
) ENGINE=InnoDB;

-- =========================
-- 2) Weak / dependent entities
-- =========================
CREATE TABLE Experiments (
  ExperimentID INT AUTO_INCREMENT PRIMARY KEY,
  MissionID    INT NOT NULL,
  Title       VARCHAR(250) NOT NULL,
  Objective    TEXT,
  Category     VARCHAR(80),
```

```
  CurrentStatus VARCHAR(40),

  ModuleID     INT,

  LeadAstronautID INT,

  CONSTRAINT fk_experiments_mission FOREIGN KEY (MissionID) REFERENCES
Missions(MissionID) ON DELETE CASCADE,

  CONSTRAINT fk_experiments_module FOREIGN KEY (ModuleID) REFERENCES
StationModules(ModuleID) ON DELETE SET NULL,

  CONSTRAINT fk_experiments_lead FOREIGN KEY (LeadAstronautID) REFERENCES
Astronauts(AstronautID) ON DELETE SET NULL

) ENGINE=InnoDB;


CREATE TABLE Schedules (

  ScheduleID   INT AUTO_INCREMENT PRIMARY KEY,

  MissionID    INT NOT NULL,

  TaskDescription TEXT,

  TaskType     VARCHAR(80),

  StartTime    DATETIME,

  EndTime      DATETIME,

  AstronautID  INT,

  CurrentStatus VARCHAR(40),

  CONSTRAINT fk_schedules_mission FOREIGN KEY (MissionID) REFERENCES
Missions(MissionID) ON DELETE CASCADE,

  CONSTRAINT fk_schedules_astronaut FOREIGN KEY (AstronautID) REFERENCES
Astronauts(AstronautID) ON DELETE SET NULL

) ENGINE=InnoDB;


CREATE TABLE MedicalRecords (

  RecordID     INT AUTO_INCREMENT PRIMARY KEY,
```

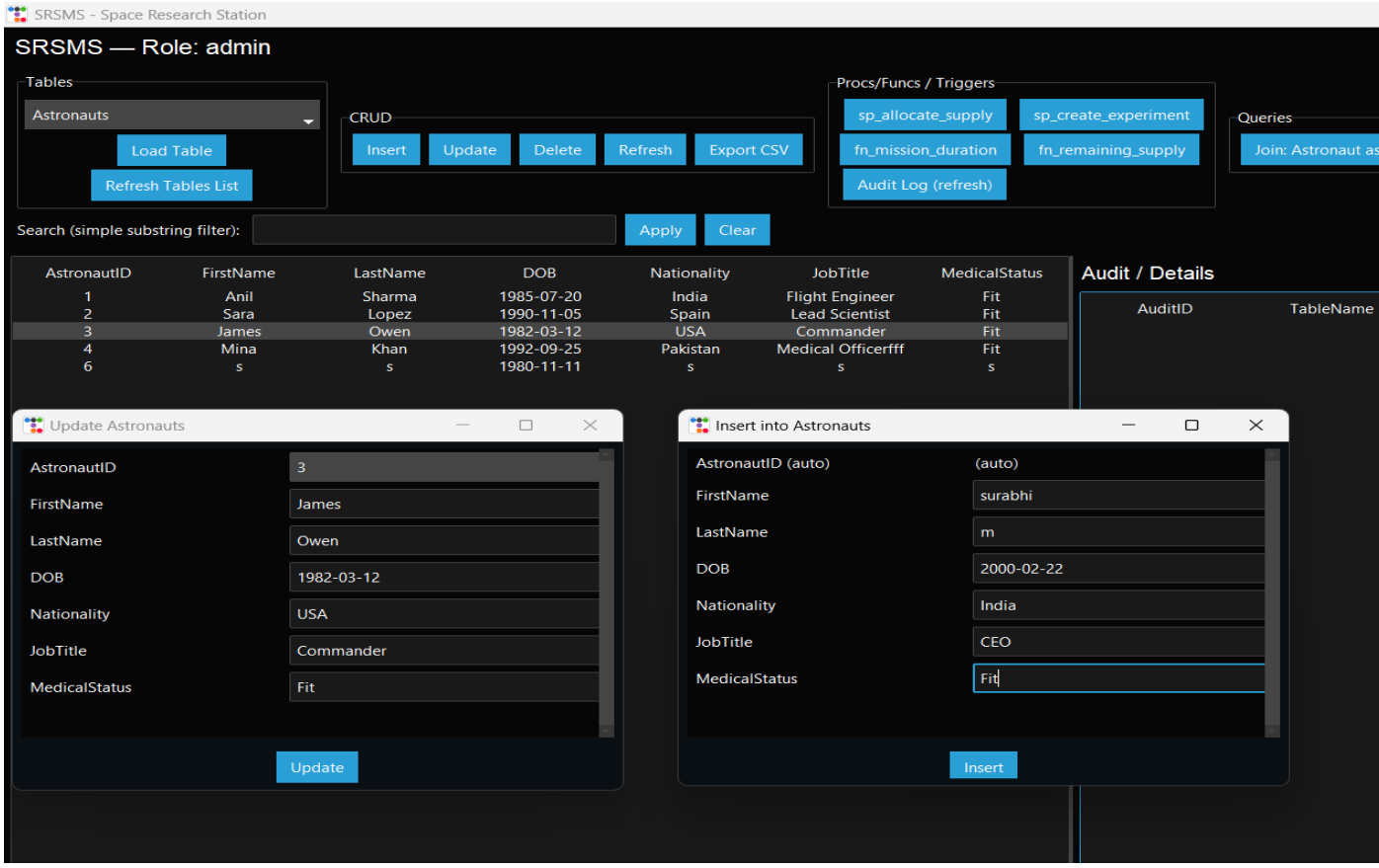
```
  AstronautID  INT NOT NULL,

  CheckupDate  DATE,

  HealthCondition VARCHAR(200),

  Treatment    TEXT,

  DoctorID     INT,

  CONSTRAINT fk_med_astronaut FOREIGN KEY (AstronautID) REFERENCES
Astronauts(AstronautID) ON DELETE CASCADE,

  CONSTRAINT fk_med_doctor FOREIGN KEY (DoctorID) REFERENCES
Astronauts(AstronautID) ON DELETE SET NULL

) ENGINE=InnoDB;


CREATE TABLE ResourceAllocations (

  AllocationID      INT AUTO_INCREMENT PRIMARY KEY,

  MissionID         INT NOT NULL,

  SupplyID          INT NOT NULL,

  QuantityAllocated DECIMAL(18,3) NOT NULL,

  AllocationDate    DATETIME DEFAULT CURRENT_TIMESTAMP,

  CONSTRAINT fk_alloc_mission FOREIGN KEY (MissionID) REFERENCES
Missions(MissionID) ON DELETE CASCADE,

  CONSTRAINT fk_alloc_supply FOREIGN KEY (SupplyID) REFERENCES Supplies(SupplyID)
ON DELETE RESTRICT

) ENGINE=InnoDB;


CREATE TABLE Communications (

  CommID       INT AUTO_INCREMENT PRIMARY KEY,

  MissionID     INT NOT NULL,

  AstronautID   INT,

  MessageType   VARCHAR(80),
```

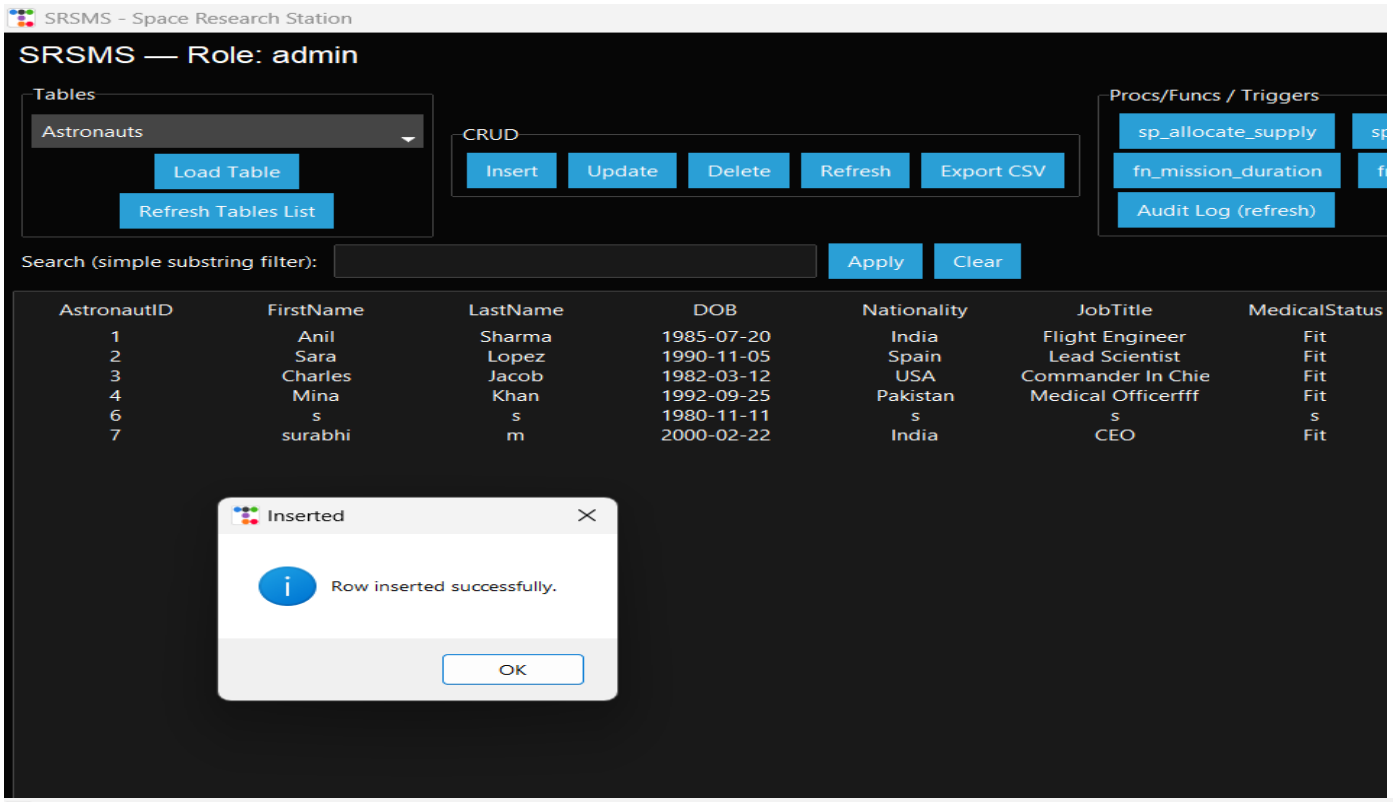
```
  TimeStamp     DATETIME DEFAULT CURRENT_TIMESTAMP,

  MessageContent TEXT,

  Recipient     VARCHAR(200),

  CONSTRAINT fk_comm_mission FOREIGN KEY (MissionID) REFERENCES
Missions(MissionID) ON DELETE CASCADE,

  CONSTRAINT fk_comm_astronaut FOREIGN KEY (AstronautID) REFERENCES
Astronauts(AstronautID) ON DELETE SET NULL

) ENGINE=InnoDB;


CREATE TABLE Anomalies (

  AnomalyID    INT AUTO_INCREMENT PRIMARY KEY,

  ModuleID     INT NOT NULL,

  DateDetected DATE DEFAULT (CURRENT_DATE),

  Severity     ENUM('Low','Medium','High','Critical') NOT NULL,

  Description  TEXT,

  ResolvedByAstronautID INT,

  ResolutionDate DATE,

  CONSTRAINT fk_anom_module FOREIGN KEY (ModuleID) REFERENCES
StationModules(ModuleID) ON DELETE CASCADE,

  CONSTRAINT fk_anom_resolver FOREIGN KEY (ResolvedByAstronautID) REFERENCES
Astronauts(AstronautID) ON DELETE SET NULL

) ENGINE=InnoDB;


-- =========================

-- 3) mapping tables (composite PKs) - preserved

-- =========================

CREATE TABLE Astronaut_Missions (

  AstronautID INT NOT NULL,
```

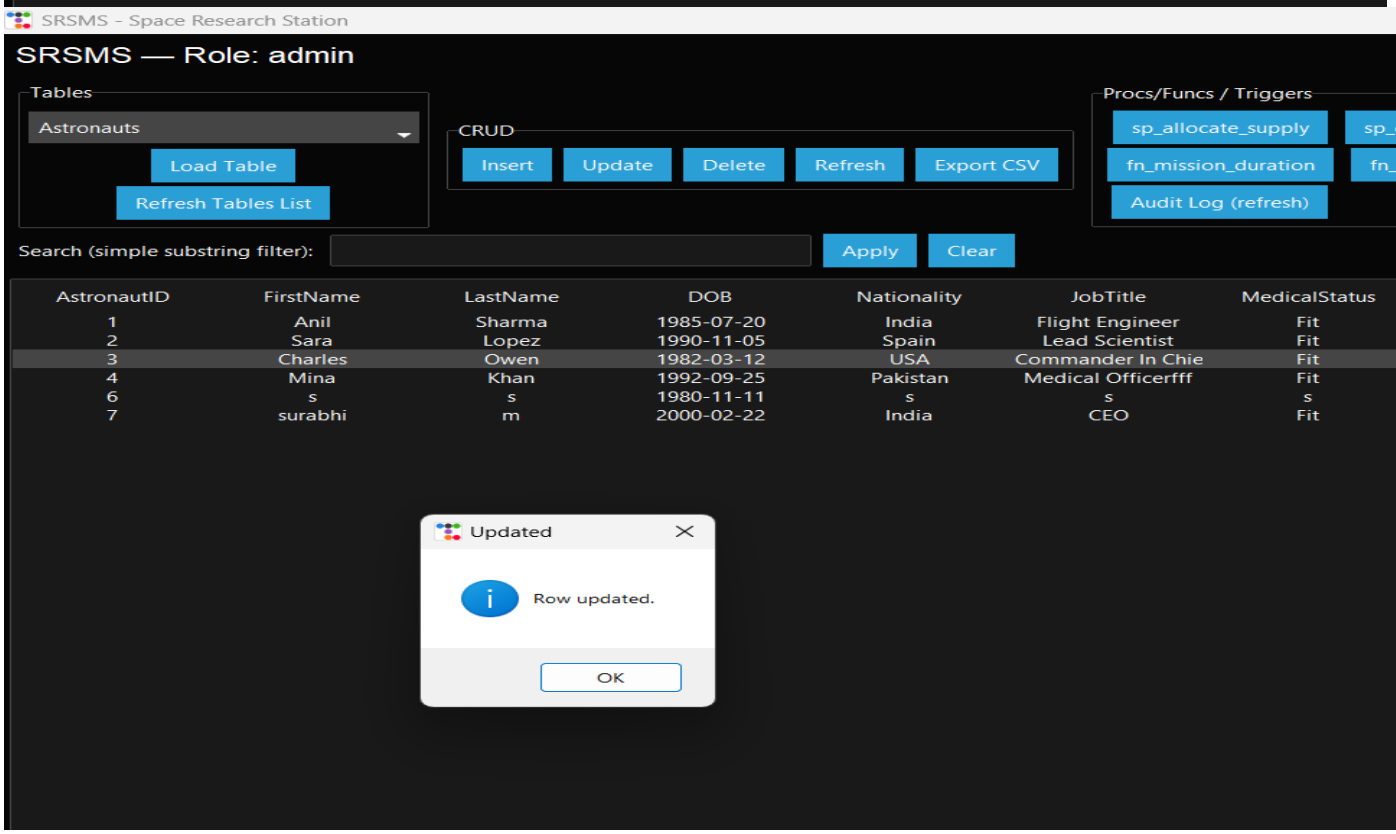
```
  MissionID   INT NOT NULL,

  Role       VARCHAR(100),

  HoursWorked DECIMAL(10,2) DEFAULT 0,

  PRIMARY KEY (AstronautID, MissionID),

  CONSTRAINT fk_am_as FOREIGN KEY (AstronautID) REFERENCES
Astronauts(AstronautID) ON DELETE CASCADE,

  CONSTRAINT fk_am_mi FOREIGN KEY (MissionID) REFERENCES Missions(MissionID) ON
DELETE CASCADE

) ENGINE=InnoDB;


CREATE TABLE Mission_Spacecraft (

  MissionID INT NOT NULL,

  SpacecraftID INT NOT NULL,

  AssignmentDate DATE DEFAULT (CURRENT_DATE),

  PRIMARY KEY (MissionID, SpacecraftID),

  CONSTRAINT fk_ms_mi FOREIGN KEY (MissionID) REFERENCES Missions(MissionID) ON
DELETE CASCADE,

  CONSTRAINT fk_ms_sp FOREIGN KEY (SpacecraftID) REFERENCES
Spacecrafts(SpacecraftID) ON DELETE RESTRICT

) ENGINE=InnoDB;


CREATE TABLE Mission_Modules (

  MissionID INT NOT NULL,

  ModuleID  INT NOT NULL,

  AssignmentDate DATE DEFAULT (CURRENT_DATE),

  PRIMARY KEY (MissionID, ModuleID),

  CONSTRAINT fk_mm_mi FOREIGN KEY (MissionID) REFERENCES Missions(MissionID) ON
DELETE CASCADE,
```

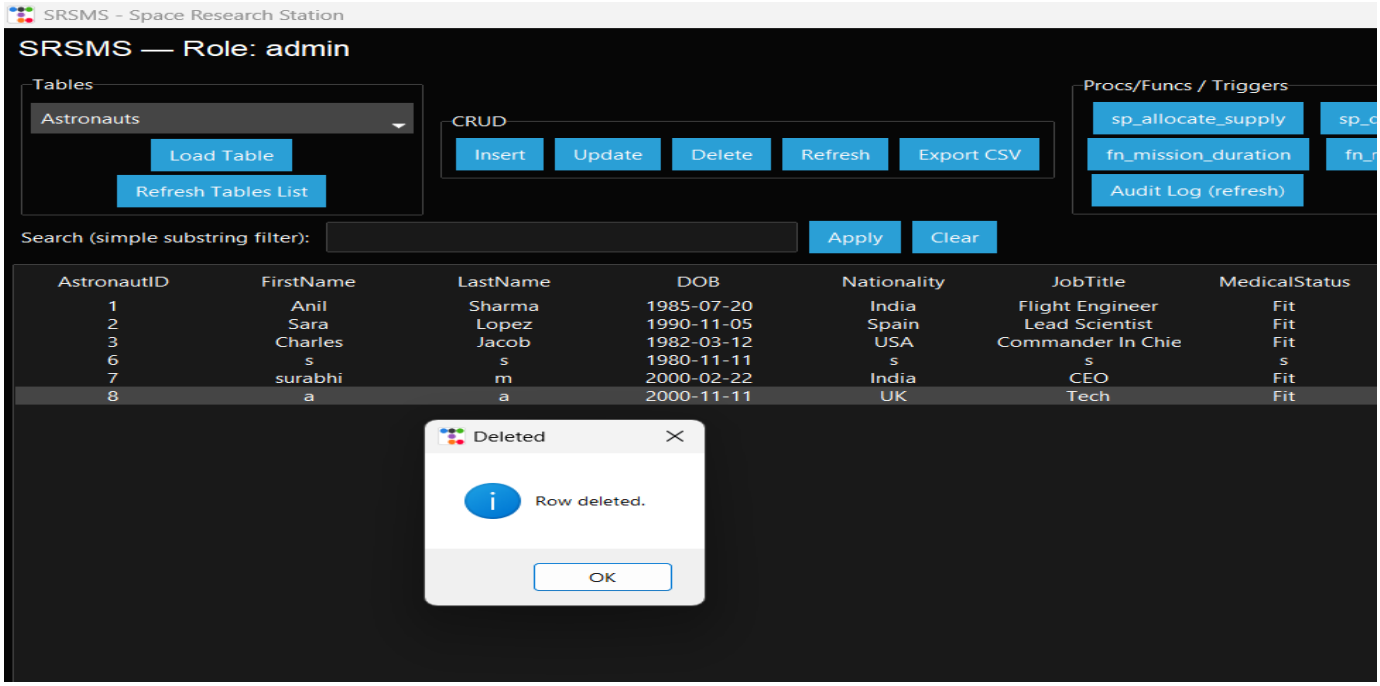
```
  CONSTRAINT fk_mm_mod FOREIGN KEY (ModuleID) REFERENCES
StationModules(ModuleID) ON DELETE CASCADE

) ENGINE=InnoDB;


CREATE TABLE Experiment_Astronauts (

  ExperimentID INT NOT NULL,

  AstronautID  INT NOT NULL,

  Role        VARCHAR(80),

  PRIMARY KEY (ExperimentID, AstronautID),

  CONSTRAINT fk_ea_exp FOREIGN KEY (ExperimentID) REFERENCES
Experiments(ExperimentID) ON DELETE CASCADE,

  CONSTRAINT fk_ea_as FOREIGN KEY (AstronautID) REFERENCES
Astronauts(AstronautID) ON DELETE CASCADE

) ENGINE=InnoDB;
```

CRUD Operation Screenshots

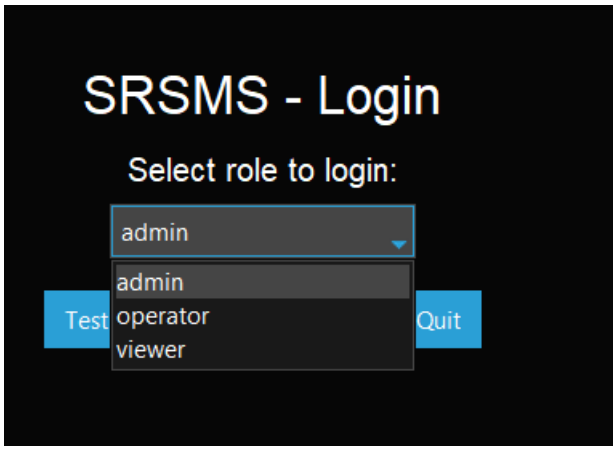
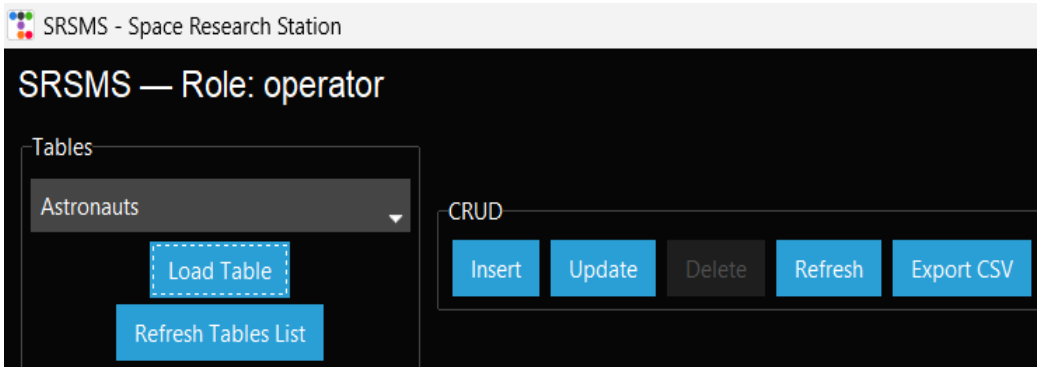

SRSMS - Space Research Station
SRSMS — Role: admin
Tables
Astronauts
Load Table
Refresh Tables List
CRUD
Insert
Update
Delete
Refresh
Export CSV
Procs/Funcs / Triggers
sp_allocate_supply
sp_create_experiment
fn_mission_duration
fn_remaining_supply
Audit Log (refresh)
Queries
Join: Astronaut as
Search (simple substring filter):
Apply
Clear
AstronautID FirstName LastName DOB Nationality JobTitle MedicalStatus
1 Anil Sharma 1985-07-20 India Flight Engineer Fit
2 Sara Lopez 1990-11-05 Spain Lead Scientist Fit
3 James Owen 1982-03-12 USA Commander Fit
4 Mina Khan 1992-09-25 Pakistan Medical Officerfff Fit
6 s s 1980-11-11 s s s
Audit / Details
AuditID TableName
Update Astronauts
AstronautID 3
FirstName James
LastName Owen
DOB 1982-03-12
Nationality USA
JobTitle Commander
MedicalStatus Fit
Update
Insert into Astronauts
AstronautID (auto) (auto)
FirstName surabhi
LastName m
DOB 2000-02-22
Nationality India
JobTitle CEO
MedicalStatus Fit
Insert

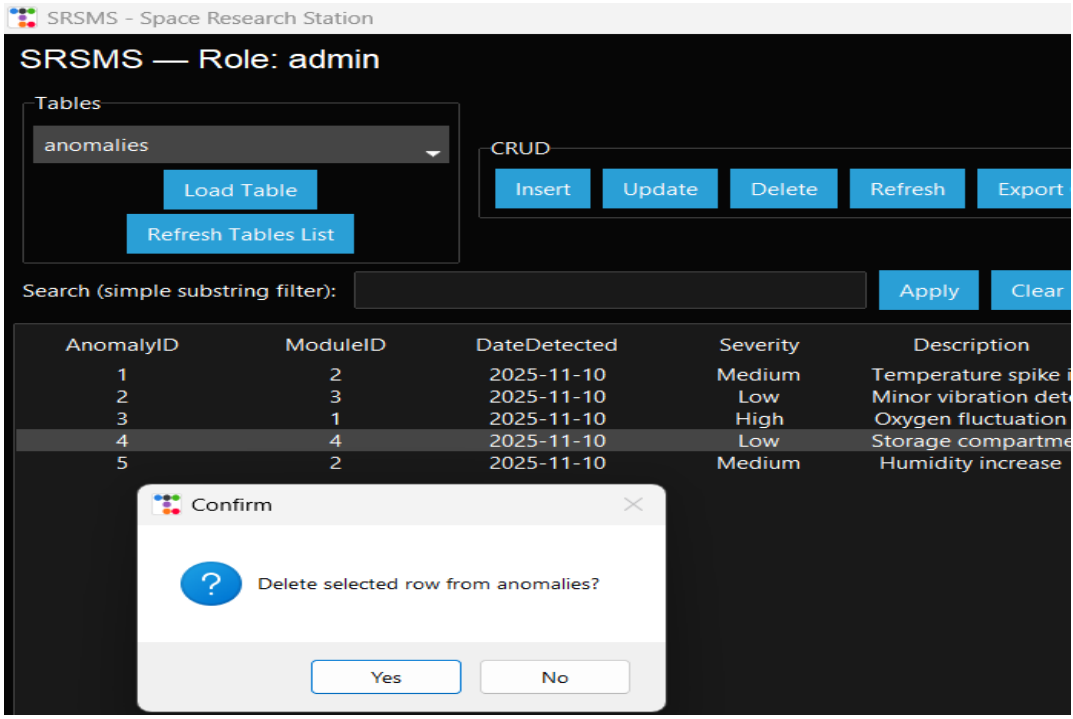
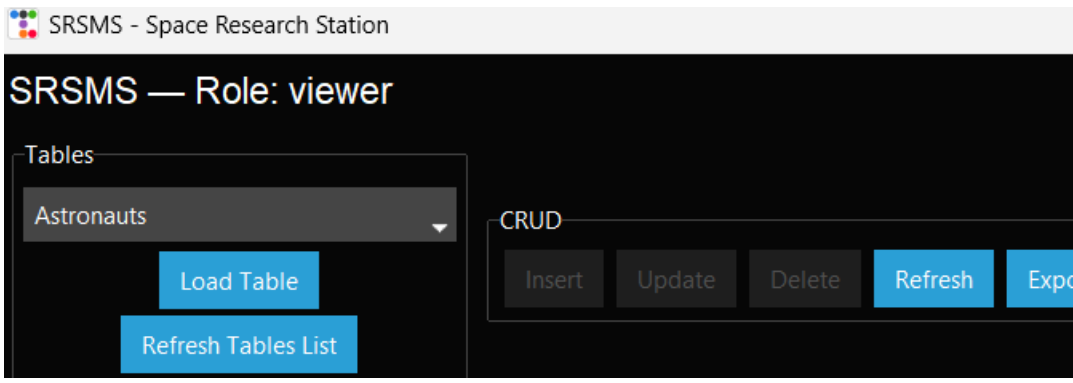

SRSMS - Space Research Station
SRSMS — Role: admin
Tables
Astronauts
Load Table
Refresh Tables List
CRUD
Insert
Update
Delete
Refresh
Export CSV
Procs/Funcs / Triggers
sp_allocate_supply
fn_mission_duration
Audit Log (refresh)
Search (simple substring filter):
Apply
Clear
AstronautID FirstName LastName DOB Nationality JobTitle MedicalStatus
1 Anil Sharma 1985-07-20 India Flight Engineer Fit
2 Sara Lopez 1990-11-05 Spain Lead Scientist Fit
3 Charles Jacob 1982-03-12 USA Commander In Chie Fit
4 Mina Khan 1992-09-25 Pakistan Medical Officerfff Fit
6 s s 1980-11-11 s s s
7 surabhi m 2000-02-22 India CEO Fit
Inserted
Row inserted successfully.
OK

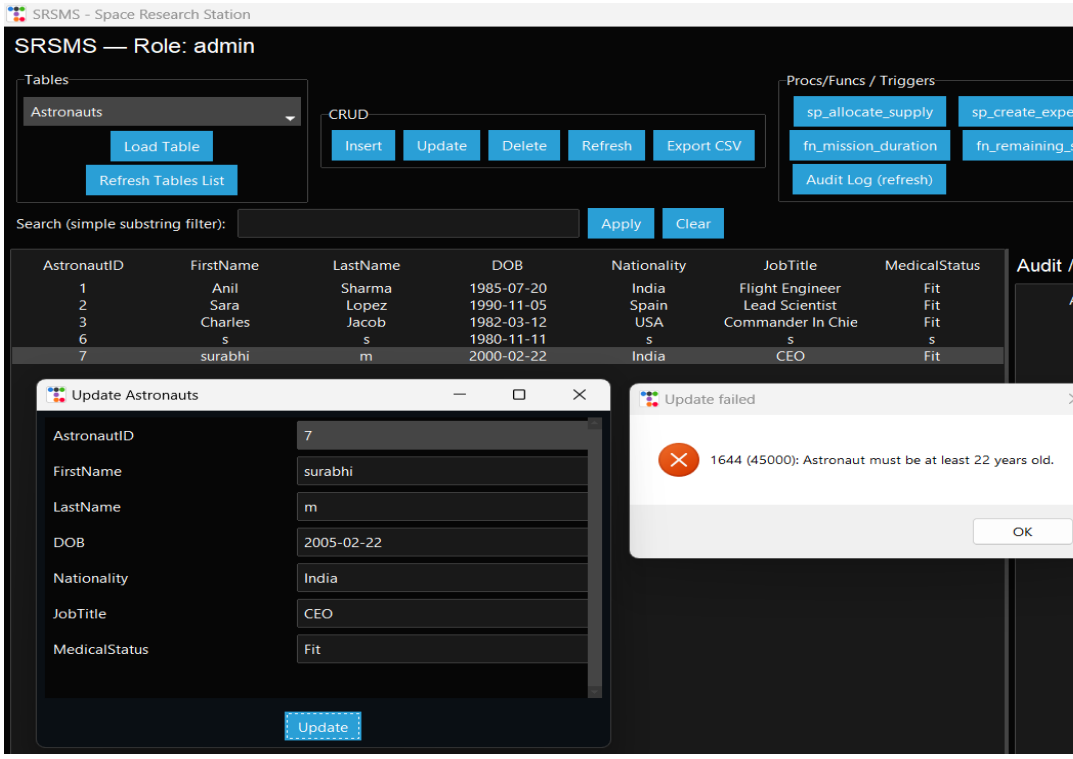
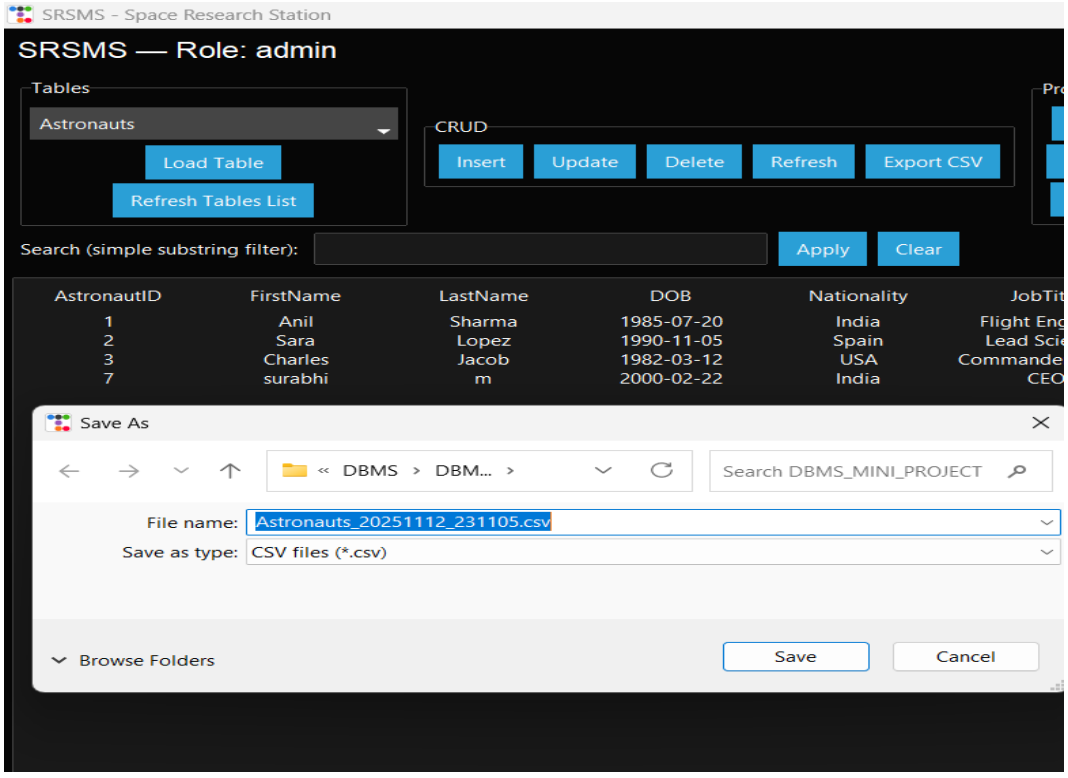

SRSMS - Space Research Station
SRSMS — Role: admin
Tables
Astronauts
Load Table
Refresh Tables List
CRUD
Insert
Update
Delete
Refresh
Export CSV
Procs/Funcs / Triggers
sp_allocate_supply
fn_mission_duration
Audit Log (refresh)
Search (simple substring filter):
Apply
Clear
AstronautID FirstName LastName DOB Nationality JobTitle MedicalStatus
1 Anil Sharma 1985-07-20 India Flight Engineer Fit
2 Sara Lopez 1990-11-05 Spain Lead Scientist Fit
3 Charles Owen 1982-03-12 USA Commander In Chie Fit
4 Mina Khan 1992-09-25 Pakistan Medical Officerfff Fit
6 s s 1980-11-11 s s
7 surabhi m 2000-02-22 India CEO Fit
Updated
Row updated.
OK



Functionalities / Features

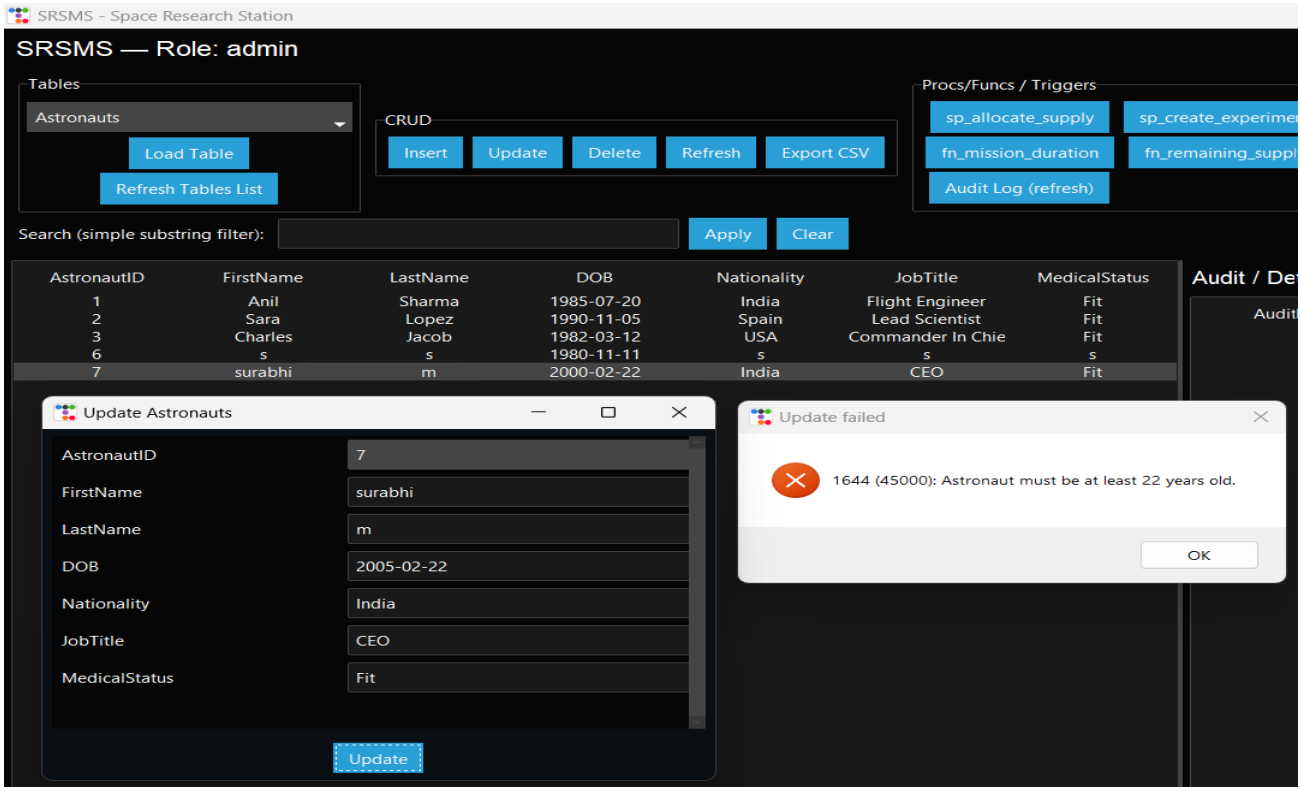
Feature	Description	Screenshot
Login	Role-based authentication	
Insert / Update Astronaut	Operator access	

Feature	Description	Screenshot
Delete Record	Admin only	
View only	Viewer	

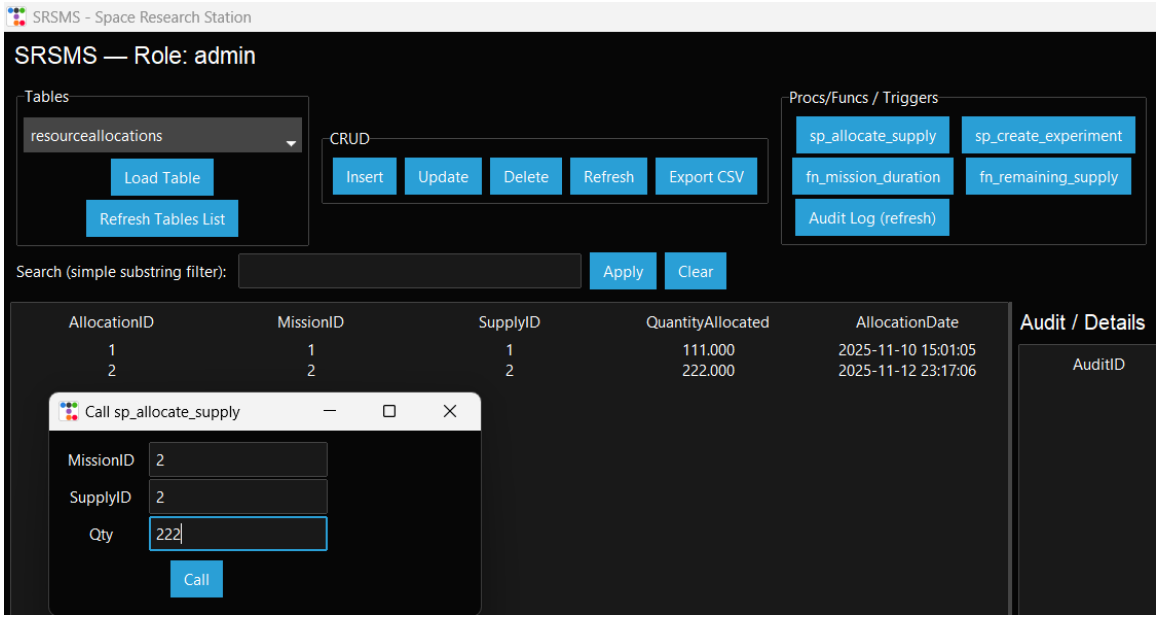
Feature	Description	Screenshot
Triggers	Astronaut must be above 22 years of age	
Export CSV	Export the table record as a CSV file	

SQL Components

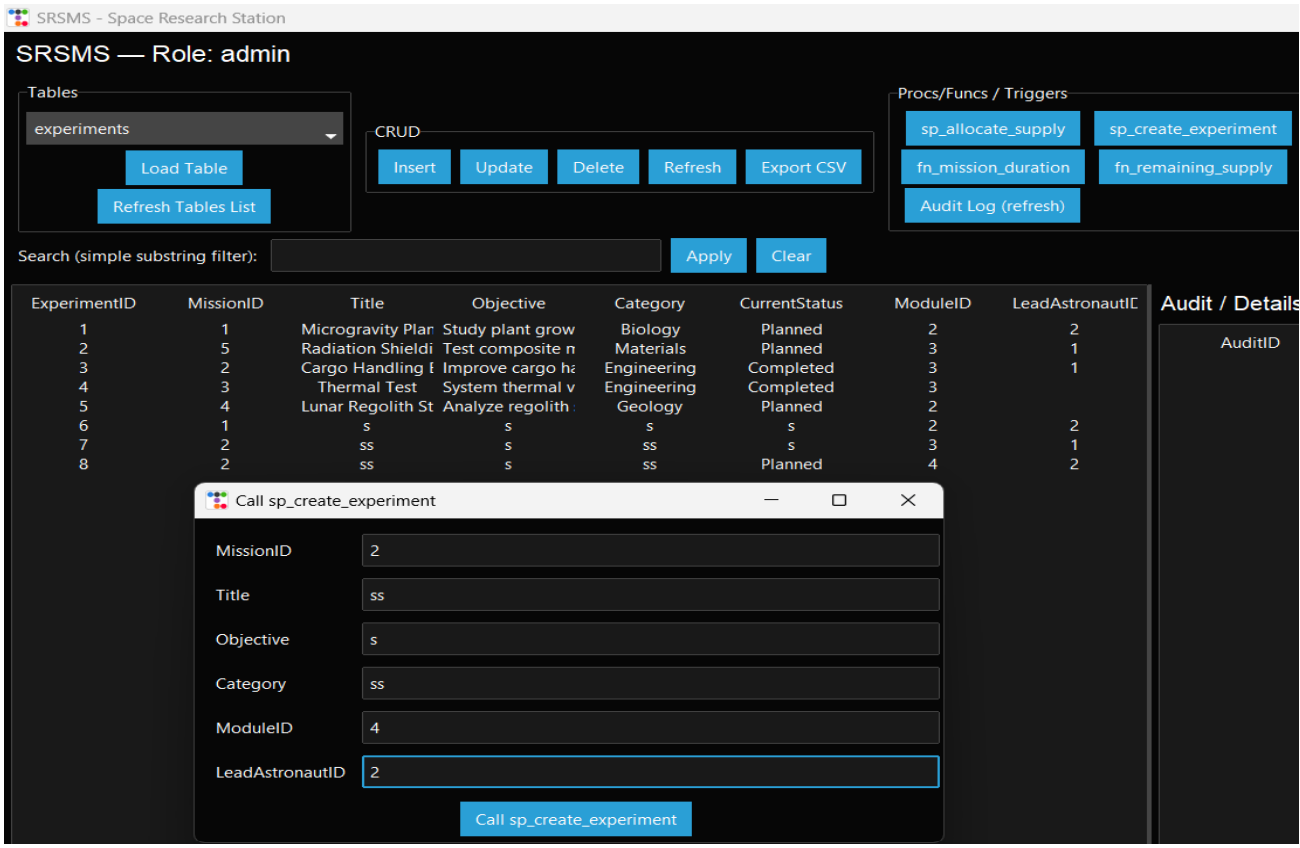
1. Triggers for insert/update astronauts under the age of 22



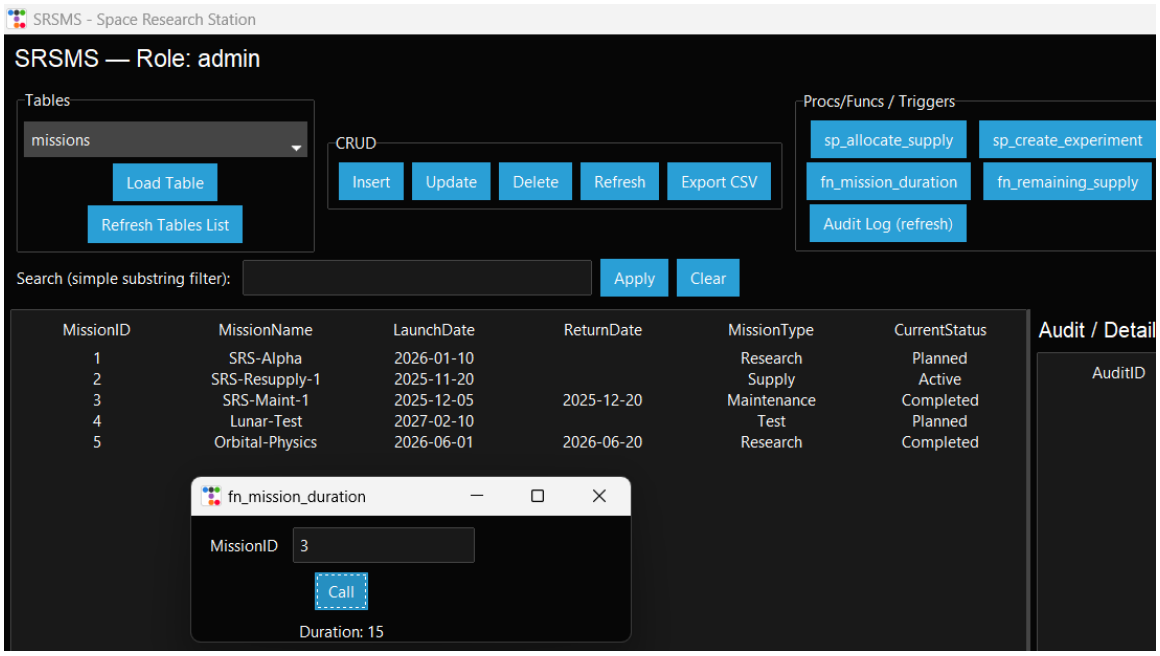
2. Stored Procedure: sp_allocate_supply



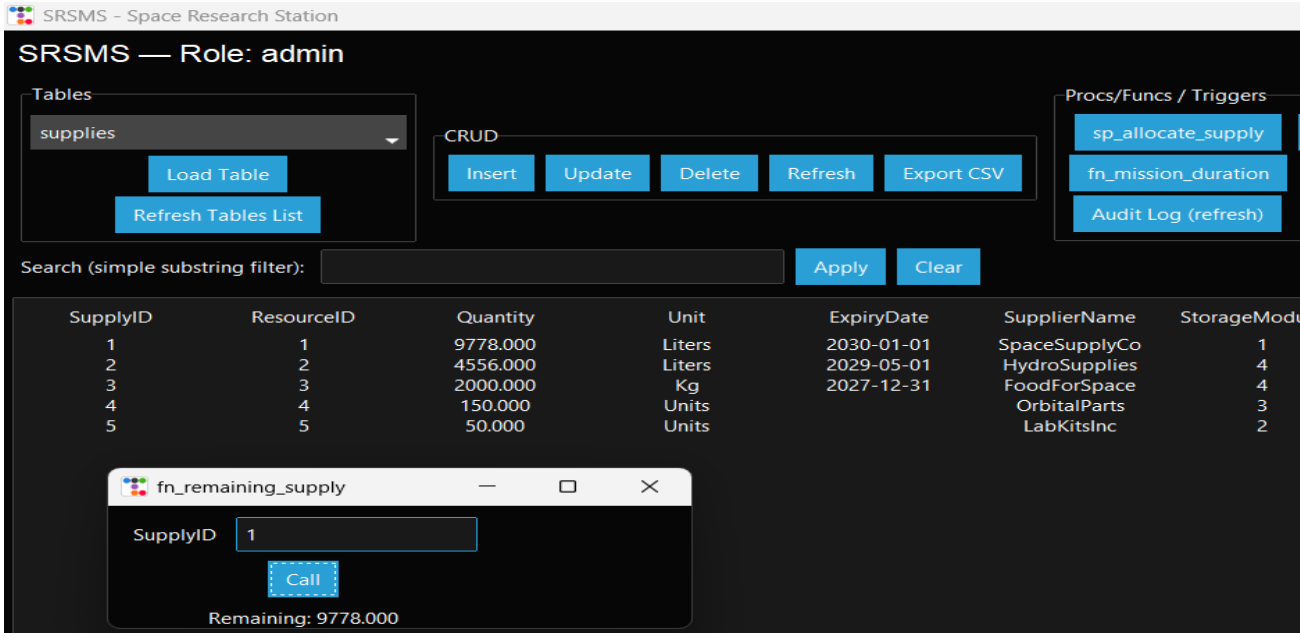
3. Stored Procedure: sp_create_experiment



4. Function: fn_mission_duration

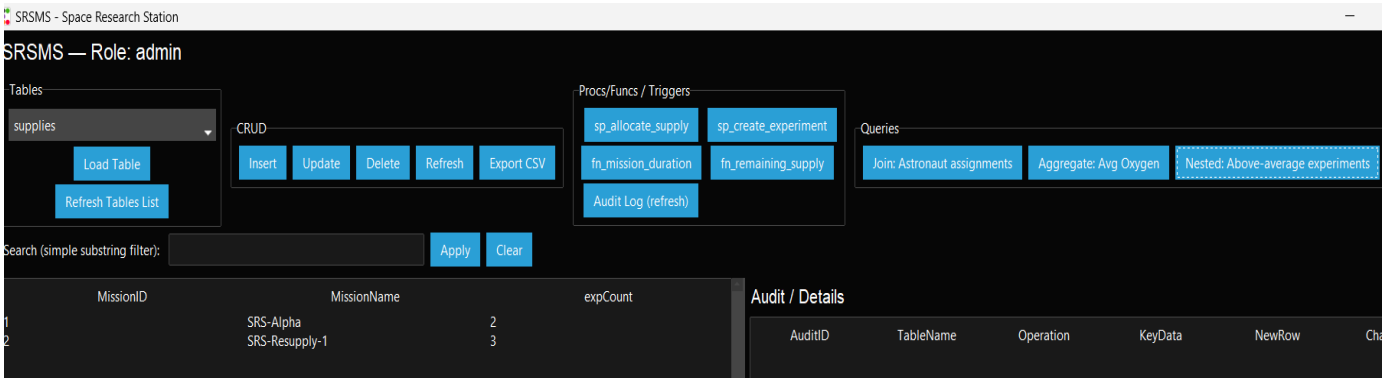


5. Function: fn_remaining_supply

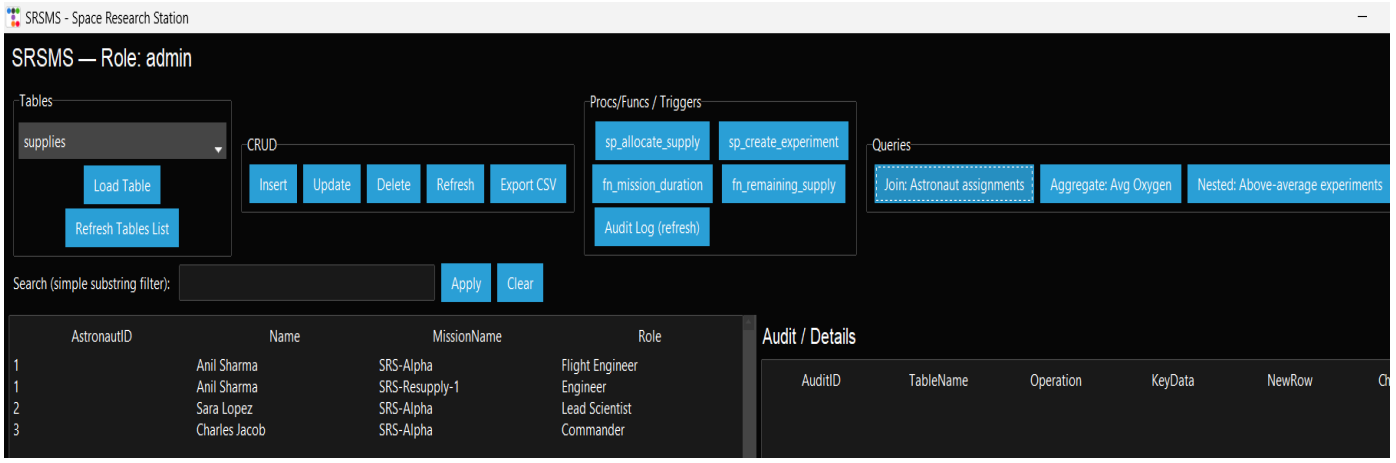


6. Queries:

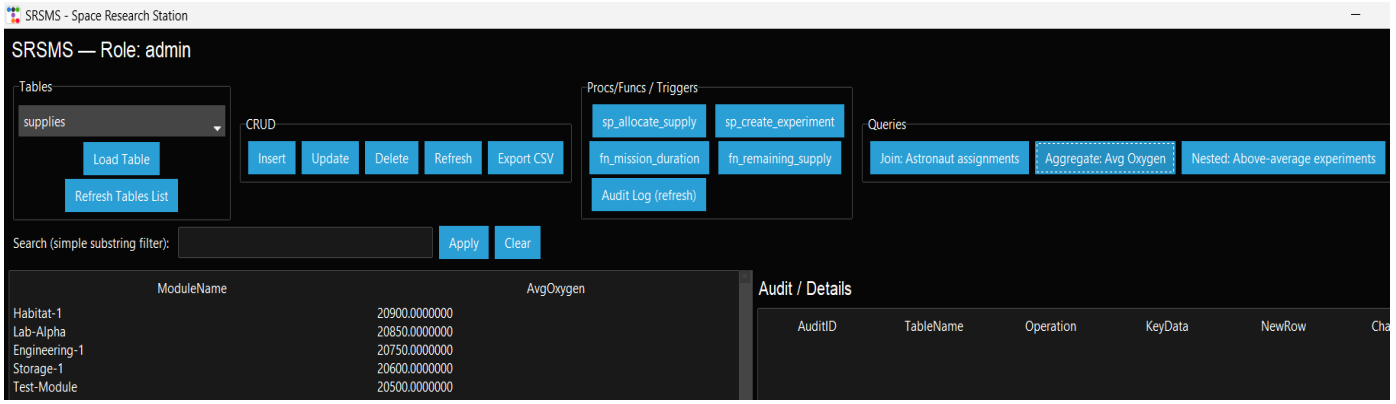
- Nested query: Above-average experiments



- Join: Astronaut assignments



- Aggregate: Average oxygen consumption



Code Snippets for Invoking Procedures/Functions

- **Stored Procedure: sp_allocate_supply**

```
DROP PROCEDURE IF EXISTS sp_allocate_supply;
DELIMITER $$
CREATE PROCEDURE sp_allocate_supply(
 IN p_missionid INT,
 IN p_supplyid INT,
 IN p_qty DECIMAL(18,3)
)
BEGIN
 DECLARE cur_qty DECIMAL(18,3);
 DECLARE v_err_msg VARCHAR(255);

 IF p_qty <= 0 THEN
  SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Quantity must be positive';
 END IF;
```

```
 IF (SELECT COUNT(*) FROM Missions WHERE MissionID = p_missionid) = 0 THEN
  SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Mission not found';
 END IF;

 START TRANSACTION;
  SELECT Quantity INTO cur_qty FROM Supplies WHERE SupplyID = p_supplyid
FOR UPDATE;
  IF cur_qty IS NULL THEN
   ROLLBACK;
   SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Supply not found';
  END IF;
  IF cur_qty < p_qty THEN
   SET v_err_msg = CONCAT('Insufficient stock. Available: ', cur_qty);
   ROLLBACK;
   SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = v_err_msg;
  END IF;

  UPDATE Supplies SET Quantity = Quantity - p_qty WHERE SupplyID = p_supplyid;
  INSERT INTO ResourceAllocations (MissionID, SupplyID, QuantityAllocated,
AllocationDate)
   VALUES (p_missionid, p_supplyid, p_qty, NOW());
 COMMIT;
END$$
DELIMITER ;
```

- **Stored Procedure: sp_create_experiments**

```
DROP PROCEDURE IF EXISTS sp_create_experiment;
DELIMITER $$
CREATE PROCEDURE sp_create_experiment(
 IN p_missionid INT,
 IN p_title VARCHAR(250),
 IN p_objective TEXT,
 IN p_category VARCHAR(80),
 IN p_moduleid INT,
 IN p_leadastronautid INT,
 OUT p_expid INT
)
BEGIN
 IF (SELECT COUNT(*) FROM Missions WHERE MissionID = p_missionid) = 0 THEN
  SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Mission not found';
 END IF;
```

```
  INSERT INTO Experiments (MissionID, Title, Objective, Category, CurrentStatus,
ModuleID, LeadAstronautID)
    VALUES (p_missionid, p_title, p_objective, p_category, 'Planned', p_moduleid,
p_leadastronautid);
  SET p_expid = LAST_INSERT_ID();
END$$
DELIMITER ;
```

- **Function: fn_mission_duration**

```
DROP FUNCTION IF EXISTS fn_mission_duration;
DELIMITER $$
CREATE FUNCTION fn_mission_duration(p_missionid INT)
RETURNS INT DETERMINISTIC
BEGIN
  DECLARE v_launch DATE;
  DECLARE v_return DATE;
  DECLARE v_days INT;
  SELECT LaunchDate, ReturnDate INTO v_launch, v_return FROM Missions WHERE
MissionID = p_missionid;
  IF v_launch IS NULL OR v_return IS NULL THEN
    RETURN NULL;
  END IF;
  SET v_days = DATEDIFF(v_return, v_launch);
  RETURN v_days;
END$$
DELIMITER ;
```

- **Function: fn_remaining_supply**

```
DROP FUNCTION IF EXISTS fn_remaining_supply;
DELIMITER $$
CREATE FUNCTION fn_remaining_supply(p_supplyid INT)
RETURNS DECIMAL(18,3) DETERMINISTIC
BEGIN
  DECLARE q DECIMAL(18,3);
  SELECT Quantity INTO q FROM Supplies WHERE SupplyID = p_supplyid;
  RETURN IFNULL(q, 0);
END$$
DELIMITER ;
```

- **Trigger: trg_check_astronaut_dob_update**

```
DROP TRIGGER IF EXISTS trg_check_astronaut_dob_update;
DELIMITER $$
```

```
CREATE TRIGGER trg_check_astronaut_dob_update
BEFORE UPDATE ON Astronauts
FOR EACH ROW
BEGIN
  IF TIMESTAMPDIFF(YEAR, NEW.DOB, CURDATE()) < 22 THEN
    SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Astronaut must be at least 22 years old.';
  END IF;
END$$

DELIMITER ;
```

- **Trigger: trg_check_astronaut_dob_insert**

```
DROP TRIGGER IF EXISTS trg_check_astronaut_dob_insert;
DELIMITER $$

CREATE TRIGGER trg_check_astronaut_dob_insert
BEFORE INSERT ON Astronauts
FOR EACH ROW
BEGIN
  IF TIMESTAMPDIFF(YEAR, NEW.DOB, CURDATE()) < 22 THEN
    SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Astronaut must be at least 22 years old.';
  END IF;
END$$

DELIMITER ;

DROP TRIGGER IF EXISTS trg_check_astronaut_dob_update;
DELIMITER $$
```

SQL File Reference

All SQL commands, triggers, and stored procedures are saved in srsdb_script.sql.

SRSDB_DBMS_MINI_PROJECT.sql

Github Repo Link

https://github.com/surabhimuralidhar/Space-Research-Station-Management-System.git