GROUP PROJECT
MILITARY MANAGEMENT SYSTEM

DB Project

Authors
Kirdyashev E., Kornelyuk E., Ryazanov S., Urusova E.
NRU HSE, Moscow, Russia
November, 2023

Contents

1 Abstract

Military Management System will allow you to reach an unprecedented level of organization in your military structure. Using such cutting edge technology as PostgreSQL, revolutionary personnel and equipment management system, your military will confidently resolve all security issues.

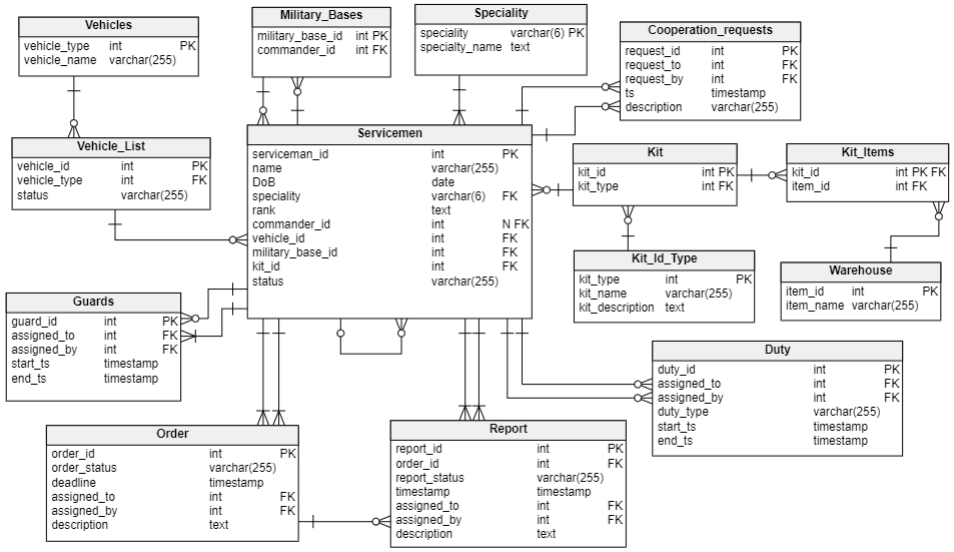
2 Description and Purpose

The Military Management System represents a groundbreaking leap forward in achieving unparalleled organizational efficiency within military structures. Leveraging state-of-the-art technology, particularly the robust capabilities of PostgreSQL, this system introduces a revolutionary approach to personnel and equipment management. With a focus on enhancing the overall effectiveness of military operations, this innovative platform empowers military leaders to address security challenges with confidence.

At its core, the Military Management System is designed to streamline and optimize the intricate web of tasks associated with overseeing personnel and equipment. Through the seamless integration of PostgreSQL, a powerful and reliable relational database management system, the platform ensures a secure and scalable foundation for data storage and retrieval. The personnel and equipment management modules offer comprehensive tools for tracking, assigning, and monitoring resources, facilitating real-time decision-making.

The overarching purpose of the Military Management System is to provide military organizations with a dynamic and responsive framework that adapts to the complexities of modern security challenges. By centralizing and automating critical processes, the system not only enhances efficiency but also contributes to the overall agility and preparedness of military forces. In essence, it serves as a strategic tool for military leaders, fostering a heightened level of organization and resource utilization, ultimately strengthening the capabilities of the armed forces in the face of diverse security scenarios.

3 ER Diagram


Vehicles
vehicle_type int PK
vehicle_name varchar(255)
Military_Bases
military_base_id int PK
commander_id int FK
Speciality
speciality varchar(6) PK
specialty_name text
Cooperation_requests
request_id int PK
request_to int FK
request_by int FK
ts timestamp
description varchar(255)
Vehicle_List
vehicle_id int PK
vehicle_type int FK
status varchar(255)
Servicemen
serviceman_id int PK
name varchar(255)
DoB date
speciality varchar(6) FK
rank text
commander_id int N FK
vehicle_id int FK
military_base_id int FK
kit_id int FK
status varchar(255)
Kit
kit_id int PK
kit_type int FK
Kit_Items
kit_id int PK FK
item_id int FK
Kit_Id_Type
kit_type int PK
kit_name varchar(255)
kit_description text
Warehouse
item_id int PK
item_name varchar(255)
Guards
guard_id int PK
assigned_to int FK
assigned_by int FK
start_ts timestamp
end_ts timestamp
Duty
duty_id int PK
assigned_to int FK
assigned_by int FK
duty_type varchar(255)
start_ts timestamp
end_ts timestamp
Order
order_id int PK
order_status varchar(255)
deadline timestamp
assigned_to int FK
assigned_by int FK
description text
Report
report_id int PK
order_id int FK
report_status varchar(255)
timestamp timestamp
assigned_to int FK
assigned_by int FK
description text

4 Classes of Users

4.1 Descriptions

1. Brigade Commander
   (a) Corresponding Rank: Colonel, Major General
   (b) Options: View personnel allocation, View equipment allocation, View chains of command, View history of orders given, View history of reports received, Reallocate personnel, Issue orders
2. Battalion Commander
   (a) Corresponding Rank: Major, Lieutenant Colonel
   (b) Options: View personnel allocation, View equipment allocation, View chains of command, View history of orders given, View history of orders received, View history of reports received, View history of reports given, Reallocate personnel, Issue an order, Submit a report, Request cooperation with another brigade, Reassign divisions
3. Company Commander
   (a) Corresponding Rank: Captain
   (b) Options: View personnel allocation, View equipment allocation, View chains of command, View history of orders given, View history of orders received, View history of reports received, View history of reports given, View history of duties given, Create a duty, Reallocate personnel, Issue an order, Submit a report
4. Platoon Commander
   (a) Corresponding Rank: Second Lieutenant, First Lieutenant
   (b) Options: View personnel allocation, View equipment allocation, View chains of command, View history of orders given, View history of orders received, View history of reports received, View history of reports given, View history of guards given, Assign guards, Reallocate personnel, Issue an order, Submit a report
5. Warehouse Manager
   (a) Corresponding Rank: Warrant Officer
   (b) Options: View equipment allocation, View warehouse, View vehicles' statuses, View history of orders received, View history of reports given, Edit warehouse table, Submit a report

4.2 Implementation

1. Brigade Commander

   (a) View Personnel Allocation [Direct Subordinates]

```
SELECT s.serviceman_id, s.name, s.rank
FROM Servicemen s
WHERE s.commander_id IN (SELECT serviceman_id
                         FROM Servicemen
                         WHERE rank IN ('Colonel', 'Major General'));
```

   (b) View Personnel Allocation [All Subordinate Levels]

```
WITH RECURSIVE Subordinates AS (
    SELECT serviceman_id, name, rank, commander_id
    FROM Servicemen
    WHERE commander_id IN (SELECT serviceman_id
                           FROM Servicemen
                           WHERE rank IN ('Colonel', 'Major General'))
    UNION
    SELECT s.serviceman_id, s.name, s.rank, s.commander_id
    FROM Servicemen s
    INNER JOIN Subordinates sub ON s.commander_id = sub.serviceman_id
)
SELECT * FROM Subordinates;
```

   (c) View Equipment Allocation [Direct Subordinates]

```
SELECT k.kit_id, kit.kit_name, kit.kit_description, s.serviceman_id,
    s.name
FROM Kit k
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type
JOIN Servicemen s ON k.kit_id = s.kit_id
WHERE s.commander_id IN (SELECT serviceman_id FROM Servicemen WHERE
    rank IN ('Colonel', 'Major General'));
```

   (d) View Equipment Allocation [All Subordinate Levels]

```
WITH RECURSIVE Subordinates AS (
    SELECT serviceman_id
    FROM Servicemen
    WHERE commander_id IN (SELECT serviceman_id FROM Servicemen WHERE
        rank IN ('Colonel', 'Major General'))
    UNION
    SELECT s.serviceman_id
    FROM Servicemen s
    INNER JOIN Subordinates sub ON s.commander_id = sub.serviceman_id
)
SELECT k.kit_id, kit.kit_name, kit.kit_description, s.serviceman_id,
    s.name
FROM Kit k
```

```
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type
JOIN Servicemen s ON k.kit_id = s.kit_id
WHERE s.serviceman_id IN (SELECT serviceman_id FROM Subordinates);
```

(e) View chains of command

```
SELECT *
FROM Servicemen
WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank IN ('Colonel', 'Major General'));
```

(f) View history of orders given

```
SELECT *
FROM "Order"
WHERE assigned_by IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank IN ('Colonel', 'Major General'));
```

(g) View history of reports received

```
SELECT *
FROM Report
WHERE assigned_to IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank IN ('Colonel', 'Major General'));
```

(h) Reallocate Personnel

```
UPDATE Servicemen
SET commander_id = [new_commander_id]
WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank IN ('Colonel', 'Major General'));
```

(i) Issue Orders

```
INSERT INTO "Order" (order_id, order_status, deadline, assigned_to,
    assigned_by, description)
VALUES ([values]);
```

2. Battalion Commander

(a) View personnel allocation [Direct Subordinates]

```
SELECT *
FROM Servicemen
WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank IN ('Major', 'Lieutenant Colonel'));
```

(b) View personnel allocation [All Subordinate Levels]

```
WITH RECURSIVE Subordinates AS (
    SELECT serviceman_id
    FROM Servicemen
    WHERE commander_id IN (SELECT serviceman_id
                    FROM Servicemen
                    WHERE rank IN ('Major', 'Lieutenant Colonel'))
    UNION
    SELECT s.serviceman_id
    FROM Servicemen s
    INNER JOIN Subordinates sub ON s.commander_id = sub.serviceman_id
)
SELECT * FROM Subordinates;
```

(c) View equipment allocation [Direct Subordinates]

```
SELECT k.*, kit.*
FROM Kit k
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type
WHERE k.kit_id IN (SELECT kit_id FROM Servicemen
WHERE commander_id IN (SELECT serviceman_id
                FROM Servicemen
                WHERE rank IN ('Major', 'Lieutenant Colonel'))
             );
```

(d) View equipment allocation [All Subordinate Levels]

```
WITH RECURSIVE Subordinates AS (
    SELECT serviceman_id
    FROM Servicemen
    WHERE commander_id IN (SELECT serviceman_id
                    FROM Servicemen
                    WHERE rank IN ('Major', 'Lieutenant Colonel'))
    UNION
    SELECT s.serviceman_id
    FROM Servicemen s
    INNER JOIN Subordinates sub ON s.commander_id = sub.serviceman_id
)
SELECT k.*, kit.*
FROM Kit k
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type
WHERE k.kit_id IN (SELECT kit_id
              FROM Servicemen
              WHERE serviceman_id IN (SELECT serviceman_id
                                      FROM Subordinates));
```

(e) View chains of command

```
WITH RECURSIVE Command_Chain AS (
    SELECT serviceman_id, name, rank, commander_id
    FROM Servicemen
    WHERE serviceman_id IN (SELECT serviceman_id
```

```
                        FROM Servicemen
                        WHERE rank IN ('Major', 'Lieutenant Colonel'))
    UNION ALL
    SELECT s.serviceman_id, s.name, s.rank, s.commander_id
    FROM Servicemen s
    INNER JOIN Command_Chain cc ON s.commander_id = cc.serviceman_id
)
SELECT * FROM Command_Chain;
```

(f) View history of orders given

```
SELECT *
FROM "Order"
WHERE assigned_by IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank IN ('Major', 'Lieutenant Colonel'));
```

(g) View history of orders received

```
SELECT *
FROM "Order"
WHERE assigned_to IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank IN ('Major', 'Lieutenant Colonel'));
```

(h) View history of reports received

```
SELECT *
FROM "Order"
WHERE assigned_to IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank IN ('Major', 'Lieutenant Colonel'));
```

(i) View history of reports given

```
SELECT *
FROM Report
WHERE assigned_by IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank IN ('Major', 'Lieutenant Colonel'));
```

(j) Reallocate personnel

```
UPDATE Servicemen
SET commander_id = [new_commander_id]
WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank IN ('Major', 'Lieutenant Colonel'));
```

(k) Issue an order

```
INSERT INTO "Order" (order_id, order_status, deadline, assigned_to,
    assigned_by, description)
VALUES ([new_values]);
```

(l) Submit a report

```
INSERT INTO Report (report_id, order_id, report_status, timestamp,
    assigned_to, assigned_by, description)
VALUES ([new_values]);
```

(m) Request cooperation with another brigade

```
INSERT INTO Cooperation_requsts (request_id, request_to, request_by,
    timestamp, description)
VALUES ([new_values]);
```

(n) Reassign Divisions

```
UPDATE Servicemen
SET commander_id = [new_commander_id]
WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank IN ('Major', 'Lieutenant Colonel'));
```

3. Company Commander

(a) View personnel allocation [Direct Subordinates]

```
SELECT *
FROM Servicemen
WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank = 'Captain');
```

(b) View personnel allocation [All Subordinate Levels]

```
WITH RECURSIVE Subordinates AS (
    SELECT serviceman_id
    FROM Servicemen
    WHERE commander_id IN (SELECT serviceman_id
                           FROM Servicemen
                           WHERE rank = 'Captain')
    UNION
    SELECT s.serviceman_id
    FROM Servicemen s
    INNER JOIN Subordinates sub ON s.commander_id = sub.serviceman_id
)
SELECT * FROM Subordinates;
```

(c) View equipment allocation [Direct Subordinates]

```
SELECT k.*, kit.*
FROM Kit k
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type
WHERE k.kit_id IN (SELECT kit_id
                   FROM Servicemen
                   WHERE commander_id IN (SELECT serviceman_id
```

```
                                        FROM Servicemen
                                        WHERE rank = 'Captain'));
```

(d) View equipment allocation [All Subordinate Levels]

```
WITH RECURSIVE Subordinates AS (
    SELECT serviceman_id
    FROM Servicemen
    WHERE commander_id IN (SELECT serviceman_id
                           FROM Servicemen
                           WHERE rank = 'Captain')
    UNION
    SELECT s.serviceman_id
    FROM Servicemen s
    INNER JOIN Subordinates sub ON s.commander_id = sub.serviceman_id
)
SELECT k.*, kit.*
FROM Kit k
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type
WHERE k.kit_id IN (SELECT kit_id
                   FROM Servicemen
                   WHERE serviceman_id IN (SELECT serviceman_id FROM
                       Subordinates));
```

(e) View chains of command

```
WITH RECURSIVE Command_Chain AS (
    SELECT serviceman_id, name, rank, commander_id
    FROM Servicemen
    WHERE serviceman_id IN (SELECT serviceman_id
                            FROM Servicemen
                            WHERE rank = 'Captain')
    UNION ALL
    SELECT s.serviceman_id, s.name, s.rank, s.commander_id
    FROM Servicemen s
    INNER JOIN Command_Chain cc ON s.commander_id = cc.serviceman_id
)
SELECT * FROM Command_Chain;
```

(f) View history of orders given

```
SELECT *
FROM "Order"
WHERE assigned_by IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank = 'Captain');
```

(g) View history of orders received

```
SELECT *
FROM "Order"
WHERE assigned_to IN (SELECT serviceman_id
```

```
                    FROM Servicemen
                    WHERE rank = 'Captain');
```

(h) View history of reports received

```
SELECT *
FROM Report
WHERE assigned_to IN (SELECT serviceman_id
                    FROM Servicemen
                    WHERE rank = 'Captain');
```

(i) View history of reports given

```
SELECT *
FROM Report
WHERE assigned_by IN (SELECT serviceman_id
                    FROM Servicemen
                    WHERE rank = 'Captain');
```

(j) View history of duties given

```
SELECT *
FROM Duty
WHERE assigned_by IN (SELECT serviceman_id
                FROM Servicemen
                WHERE rank IN ('Captain'));
```

(k) Create a duty:

```
INSERT INTO Duty (duty_id, assigned_to, assigned_by, duty_type,
    start_ts, end_ts)
VALUES ([new_values]);
```

(l) Reallocate personnel

```
UPDATE Servicemen
SET commander_id = [new_commander_id]
WHERE commander_id IN (SELECT serviceman_id
                    FROM Servicemen
                    WHERE rank = 'Captain');
```

(m) Issue an order

```
INSERT INTO "Order" (order_id, order_status, deadline, assigned_to,
    assigned_by, description)
VALUES ([new_values]);
```

(n) Submit a report

```
 INSERT INTO Report (report_id, order_id, report_status, timestamp,
     assigned_to, assigned_by, description)
 VALUES ([new_values]);
```

4. Platoon Commander

(a) View personnel allocation [Direct Subordinates]

```
SELECT *
FROM Servicemen
WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank IN ('Second Lieutenant', 'First
                           Lieutenant'));
```

(b) View personnel allocation [All Subordinate Levels]

```
WITH RECURSIVE Subordinates AS (
    SELECT serviceman_id
    FROM Servicemen
    WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank IN ('Second Lieutenant', 'First
                           Lieutenant'))
    UNION
    SELECT s.serviceman_id
    FROM Servicemen s
    INNER JOIN Subordinates sub ON s.commander_id = sub.serviceman_id)
    SELECT * FROM Subordinates;
```

(c) View equipment allocation [Direct Subordinates]

```
SELECT k.*, kit.*
FROM Kit k
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type
WHERE k.kit_id IN (SELECT kit_id
              FROM Servicemen
              WHERE commander_id IN (SELECT serviceman_id
                                     FROM Servicemen
                                     WHERE rank IN ('Second
                                         Lieutenant','First
                                         Lieutenant'));
```

(d) View equipment allocation [All Subordinate Levels]

```

WITH RECURSIVE Subordinates AS (
    SELECT serviceman_id
    FROM Servicemen
    WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank IN ('Second Lieutenant', 'First
                           Lieutenant'))
    UNION
    SELECT s.serviceman_id
    FROM Servicemen s
```

```
    INNER JOIN Subordinates sub ON s.commander_id = sub.serviceman_id
)
SELECT k.*, kit.*
FROM Kit k
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type
WHERE k.kit_id IN (SELECT kit_id
                   FROM Servicemen
                   WHERE serviceman_id IN (SELECT serviceman_id
                                           FROM Subordinates));
```

(e) View chains of command

```
WITH RECURSIVE Command_Chain AS (
    SELECT serviceman_id, name, rank, commander_id
    FROM Servicemen
    WHERE serviceman_id IN (SELECT serviceman_id
                            FROM Servicemen
                            WHERE rank IN ('SecondLieutenant', 'First
                                Lieutenant'))
UNION ALL
SELECT s.serviceman_id, s.name, s.rank, s.commander_id
FROM Servicemen s
INNER JOIN Command_Chain cc ON s.commander_id = cc.serviceman_id)
SELECT * FROM Command_Chain;
```

(f) View history of orders given

```
SELECT *
FROM "Order"
WHERE assigned_by IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank IN ('Second Lieutenant', 'First
                          Lieutenant'));
```

(g) View history of orders received

```
SELECT *
FROM "Order"
WHERE assigned_to IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank IN ('Second Lieutenant', 'First
                          Lieutenant'));
```

(h) View history of reports received

```
SELECT *
FROM Report
WHERE assigned_to IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank IN ('Second Lieutenant', 'First
                          Lieutenant'));
```

(i) View history of guards assigned

```
SELECT *
FROM Report
WHERE assigned_by IN (SELECT serviceman_id
                      FROM Servicemen
                      WHERE rank = 'Captain');
```

(j) Assign a guard

```
INSERT INTO Guards (guard_id, assigned_to, assigned_by, start_ts,
    end_ts)
VALUES ([new_values]);
```

(k) Reallocate personnel

```
UPDATE Servicemen
SET commander_id = [new_com_id]
WHERE commander_id IN (SELECT serviceman_id
                       FROM Servicemen
                       WHERE rank IN ('Second Lieutenant', 'First
                           Lieutenant'));
```

(l) Issue an order

```
INSERT INTO "Order" (order_id, order_status, deadline, assigned_to,
    assigned_by, description)
VALUES ([values]);
```

(m) Submit a report

```
INSERT INTO Report (report_id, order_id, report_status, timestamp,
    assigned_to, assigned_by, description)
VALUES ([values]);
```

5. Warehouse Manager

(a) View Equipment Allocation

```
SELECT k.*, kit.*
FROM Kit k
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type;
```

(b) View Warehouse Contents

```
SELECT * FROM Warehouse;
```

(c) View vehicles' statuses

```
SELECT vl.*, v.*
FROM Vehicle_List vl
JOIN Vehicles v ON vl.vehicle_type = v.vehicle_type;
```

(d) View history of orders received

```
SELECT *
FROM "Order"
WHERE assigned_to IN (SELECT serviceman_id FROM Servicemen
WHERE rank = 'Warrant Officer');
```

(e) View history of reports given

```
SELECT *
FROM Report
WHERE assigned_by IN (SELECT serviceman_id FROM Servicemen
WHERE rank = 'Warrant Officer');
```

(f) Edit warehouse table

```
UPDATE Warehouse SET [column_name] = [new_value] WHERE [condition];
```

(g) Submit a report

```
INSERT INTO Report (report_id, order_id, report_status, timestamp,
    assigned_to, assigned_by, description) VALUES ([values]);
```

(h) Tracking Equipment Status Changes:

```
SELECT k.kit_id, kit.kit_name, kit.kit_description,
    ks.status_change_date, ks.new_status
FROM Kit k
JOIN Kit_Id_Type kit ON k.kit_type = kit.kit_type
JOIN Kit_Status ks ON k.kit_id = ks.kit_id;
```

(i) Stock Management:

```
SELECT item_id, COUNT(*) as quantity
FROM Warehouse
GROUP BY item_id;
```

(j) Staff Training Needs Analysis:

```
SELECT s.speciality, COUNT(*) as personnel_count
FROM Servicemen s
GROUP BY s.speciality;
```

5 Constraints

1. Servicemen can be assigned with only one order only at the same time.
2. Servicemen may only give orders to the personnel in their direct command.
3. Status of the order must be one of the following: {Completed, Failed, In Progress, Cancelled}
4. Order issue date should be earlier than order's completion date
5. If the order is completed - completion date must be saved
6. If serviceman's status is "200" or "MIA" they cannot be assigned any orders and must be replaced in two weeks after the status assignment.
7. Vehicle's status must be one of the following: {Combat-ready, Destroyed, Repairs, Maintenance}
8. Vehicle with status "Destroyed" must be removed from the table when the replacement had arrived.

6 Functional Dependencies

1. Servicemen
   (a) serviceman_id - name, DoB, speciality, rank, commander_id, vehicle_id, military_base_id, status_id, kit_id
   (b) commander_id - military_base_id
   (c) speciality - kit_id
2. Speciality
   (a) speciality - speciality_name
3. Military_bases
   (a) military_base_id - commander_id
4. Vehicle_List
   (a) vehicle_id - vehicle_type, status
5. Vehicles
   (a) vehicle_type - vehicle_name
6. Status_Codes
   (a) status_id - status_name
7. Order
   (a) order_id - order_status, deadline, assigned_to, assigned_by, description
8. Kit
   (a) kit_id - kit_type
9. Kit_Items
   (a) no dependencies
10. Kit_Id_Type
    (a) kit_type - kit_name, kit_description
11. Warehouse
    (a) item_id - item_name
12. Duty
    (a) duty_id - serviceman_id, duty_type, start_ts, end_ts
13. Guard
    (a) guard_id - serviceman_id, start_ts, end_ts

7 SQL Code

7.1 Tables

```
CREATE TABLE Kit (
    kit_id int NOT NULL PRIMARY KEY,
    kit_type int NOT NULL
);

CREATE TABLE Kit_Id_Type (
    kit_type int NOT NULL PRIMARY KEY,
    kit_name varchar(255) NOT NULL,
    kit_description text NOT NULL
);

CREATE TABLE Kit_Items (
    kit_id int NOT NULL PRIMARY KEY,
    item_id int NOT NULL
);

CREATE TABLE Military_bases (
    military_base_id int NOT NULL PRIMARY KEY,
    commander_id int NOT NULL
);

CREATE TABLE "Order" (
    order_id int NOT NULL PRIMARY KEY,
    order_status varchar(255) NOT NULL,
    deadline timestamp NOT NULL,
    assigned_to int NOT NULL,
    assigned_by int NOT NULL,
    description text NOT NULL
);

CREATE TABLE Report (
    report_id int NOT NULL PRIMARY KEY,
    order_id int NOT NULL,
    report_status varchar(255) NOT NULL,
    timestamp timestamp NOT NULL,
    assigned_to int NOT NULL,
    assigned_by int NOT NULL,
    description text NOT NULL
);

CREATE TABLE Servicemen (
    serviceman_id int NOT NULL PRIMARY KEY,
    name varchar(255) NOT NULL,
    DoB date NOT NULL,
```

```
    speciality varchar(6) NOT NULL,
    rank text NOT NULL,
    commander_id int NULL,
    vehicle_id int NOT NULL,
    military_base_id int NOT NULL,
    kit_id int NOT NULL,
    status varchar(255) NOT NULL
);

CREATE TABLE Speciality (
    speciality varchar(6) NOT NULL PRIMARY KEY,
    specialty_name text NOT NULL
);

CREATE TABLE Vehicle_List (
    vehicle_id int NOT NULL PRIMARY KEY,
    vehicle_type int NOT NULL,
    status varchar(255) NOT NULL
);

CREATE TABLE Vehicles (
    vehicle_type int NOT NULL PRIMARY KEY,
    vehicle_name varchar(255) NOT NULL
);

CREATE TABLE Warehouse (
    item_id int NOT NULL PRIMARY KEY,
    item_name varchar(255) NOT NULL,
    disposal_date date NOT NULL
);

CREATE TABLE Duty (
    duty_id int NOT NULL,
    assigned_to int NOT NULL,
    assigned_by int NOT NULL,
    duty_type varchar(255) NOT NULL,
    start_ts timestamp NOT NULL,
    end_ts timestamp NOT NULL,
    CONSTRAINT Duty_pk PRIMARY KEY (duty_id)
);

CREATE TABLE Guards (
    guard_id int NOT NULL,
    assigned_to int NOT NULL,
    assigned_by int NOT NULL,
    start_ts timestamp NOT NULL,
    end_ts timestamp NOT NULL,
    CONSTRAINT Guards_pk PRIMARY KEY (guard_id)
);
```

```
CREATE TABLE Cooperation_requests (
    request_id int NOT NULL,
    request_to int NOT NULL,
    request_by int NOT NULL,
    ts timestamp NOT NULL,
    description varchar(255) NOT NULL,
    CONSTRAINT Cooperation_requests_pk PRIMARY KEY (request_id)
);
```

7.2 References

```
-- Reference: Kit_Id_Type_Kit (table: Kit)
ALTER TABLE Kit ADD CONSTRAINT Kit_Id_Type_Kit
    FOREIGN KEY (kit_type)
    REFERENCES Kit_Id_Type (kit_type)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Kit_Items_Kit (table: Kit_Items)
ALTER TABLE Kit_Items ADD CONSTRAINT Kit_Items_Kit
    FOREIGN KEY (kit_id)
    REFERENCES Kit (kit_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Kit_Servicemen (table: Servicemen)
ALTER TABLE Servicemen ADD CONSTRAINT Kit_Servicemen
    FOREIGN KEY (kit_id)
    REFERENCES Kit (kit_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Order_Servicemen (table: Order)
ALTER TABLE "Order" ADD CONSTRAINT Order_Servicemen
    FOREIGN KEY (assigned_by)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Report_Order (table: Report)
ALTER TABLE Report ADD CONSTRAINT Report_Order
    FOREIGN KEY (order_id)
    REFERENCES "Order" (order_id)
```

```
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Report_Servicemen (table: Report)
ALTER TABLE Report ADD CONSTRAINT Report_Servicemen
    FOREIGN KEY (assigned_to)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Servicemen_Military_bases (table: Military_bases)
ALTER TABLE Military_bases ADD CONSTRAINT Servicemen_Military_bases
    FOREIGN KEY (commander_id)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Servicemen_Order (table: Order)
ALTER TABLE "Order" ADD CONSTRAINT Servicemen_Order
    FOREIGN KEY (assigned_to)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Servicemen_Report (table: Report)
ALTER TABLE Report ADD CONSTRAINT Servicemen_Report
    FOREIGN KEY (assigned_by)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Servicemen_Servicemen (table: Servicemen)
ALTER TABLE Servicemen ADD CONSTRAINT Servicemen_Servicemen
    FOREIGN KEY (commander_id)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Speciality_Servicemen (table: Servicemen)
ALTER TABLE Servicemen ADD CONSTRAINT Speciality_Servicemen
    FOREIGN KEY (speciality)
    REFERENCES Speciality (speciality)
    NOT DEFERRABLE
```

```
    INITIALLY IMMEDIATE
;

-- Reference: VehSerLink_Servicemen (table: Servicemen)
ALTER TABLE Servicemen ADD CONSTRAINT VehSerLink_Servicemen
    FOREIGN KEY (vehicle_id)
    REFERENCES Vehicle_List (vehicle_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Vehicles_VehSerLink (table: Vehicle_List)
ALTER TABLE Vehicle_List ADD CONSTRAINT Vehicles_VehSerLink
    FOREIGN KEY (vehicle_type)
    REFERENCES Vehicles (vehicle_type)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Warehouse_Kit_Items_Assigned (table: Kit_Items)
ALTER TABLE Kit_Items ADD CONSTRAINT Warehouse_Kit_Items_Assigned
    FOREIGN KEY (item_id)
    REFERENCES Warehouse (item_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: military_bases_Servicemen (table: Servicemen)
ALTER TABLE Servicemen ADD CONSTRAINT military_bases_Servicemen
    FOREIGN KEY (military_base_id)
    REFERENCES Military_bases (military_base_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Servicemen_Guards (table: Guards)
ALTER TABLE Guards ADD CONSTRAINT Servicemen_Guards
    FOREIGN KEY (assigned_to)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Guards_Servicemen (table: Guards)
ALTER TABLE Guards ADD CONSTRAINT Guards_Servicemen
    FOREIGN KEY (assigned_by)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
```

```
;

-- Reference: Duty_Servicemen (table: Duty)
ALTER TABLE Duty ADD CONSTRAINT Duty_Servicemen
    FOREIGN KEY (assigned_by)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Servicemen_Duty (table: Duty)
ALTER TABLE Duty ADD CONSTRAINT Servicemen_Duty
    FOREIGN KEY (assigned_to)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Servicemen_Cooperation_requests (table: Cooperation_requests)
ALTER TABLE Cooperation_requests ADD CONSTRAINT Servicemen_Cooperation_requests
    FOREIGN KEY (request_by)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;

-- Reference: Cooperation_requests_Servicemen (table: Cooperation_requests)
ALTER TABLE Cooperation_requests ADD CONSTRAINT Cooperation_requests_Servicemen
    FOREIGN KEY (request_to)
    REFERENCES Servicemen (serviceman_id)
    NOT DEFERRABLE
    INITIALLY IMMEDIATE
;
```