Submarine Fleet Readiness

Relational Database Design

Owen Thomas

Table of Contents

Executive Summary

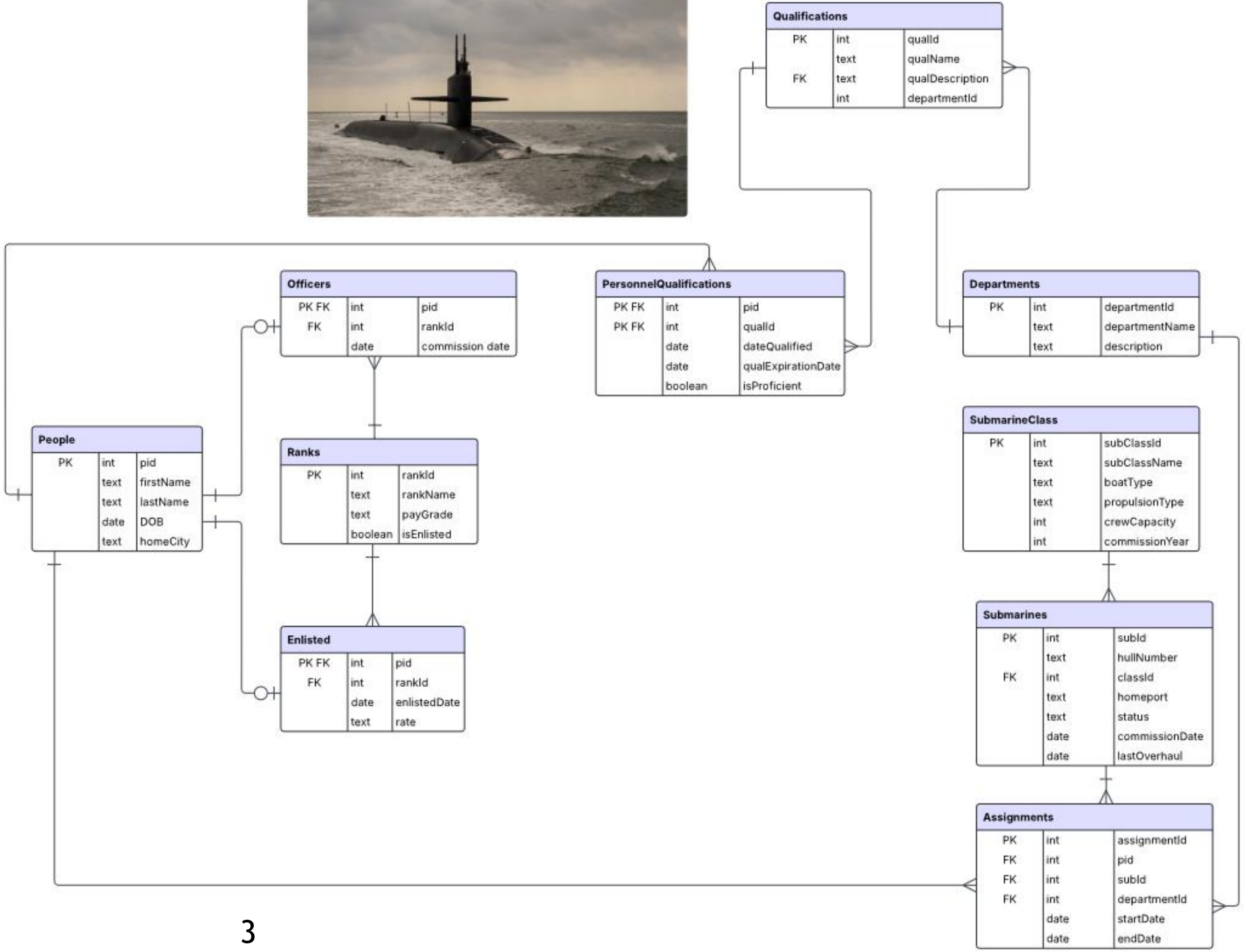
- The Submarine Fleet Readiness Relational Database is designed to assist Navy leadership in maintaining an accurate, organized representation of submarine personnel, qualifications, and operational status throughout the fleet. Ensuring that the submarine crews are properly manned is the key to maintaining a positive defense posture and achieving nuclear deterrence ultimately providing the United States national security.
- The purpose of this database is to streamline the process of achieving this information enabling naval leadership to make meaningful, accurate and effective decisions when dealing with the rigorous tasks of keeping Submarines properly manned.
- This presentation will begin with an ER Diagram that illustrates the relational database design forming the backbone of fleet management. Following this, is a walkthrough of each table in the ER Diagram along with the create statement and sample data that demonstrates how the database operates. In addition, we will go over the views, reports, stored procedures, and triggers that are included to automate common tasks, mitigate errors and provide useful insight. Lastly, the security and access control measures are detailed to show how different roles interact with the database.

Key Terms

- Hull Number – a unique identifier assigned to each Submarine like a license plate number
- Class – a category grouping Submarines with the same design, specifications and features
- Officers – Commissioned member with leadership roles (minimum Bachelor Degree)
- Enlisted – non-commissioned member who performs technical operations
- Qualification – certification showing a person is trained for a specific duty
- Overhaul – major maintenance period to upgrade or repair the Submarine

Submarine Fleet Readiness

ER Diagram



Qualifications

PK	int	qualId
	text	qualName
	text	qualDescription
FK	int	departmentId

Officers

PK FK	int	pid
FK	int	rankId
	date	commission date

PersonnelQualifications

PK FK	int	pid
PK FK	int	qualId
	date	dateQualified
	date	qualExpirationDate
	boolean	isProficient

Departments

PK	int	departmentId
	text	departmentName
	text	description

People

PK	int	pid
	text	firstName
	text	lastName
	date	DOB
	text	homeCity

Ranks

PK	int	rankId
	text	rankName
	text	payGrade
	boolean	isEnlisted

SubmarineClass

PK	int	subClassId
	text	subClassName
	text	boatType
	text	propulsionType
	int	crewCapacity
	int	commissionYear

Enlisted

PK FK	int	pid
FK	int	rankId
	date	enlistedDate
	text	rate

Submarines

PK	int	subId
	text	hullNumber
FK	int	classId
	text	homeport
	text	status
	date	commissionDate
	date	lastOverhaul

Assignments

PK	int	assignmentId
FK	int	pid
FK	int	subId
FK	int	departmentId
	date	startDate
	date	endDate

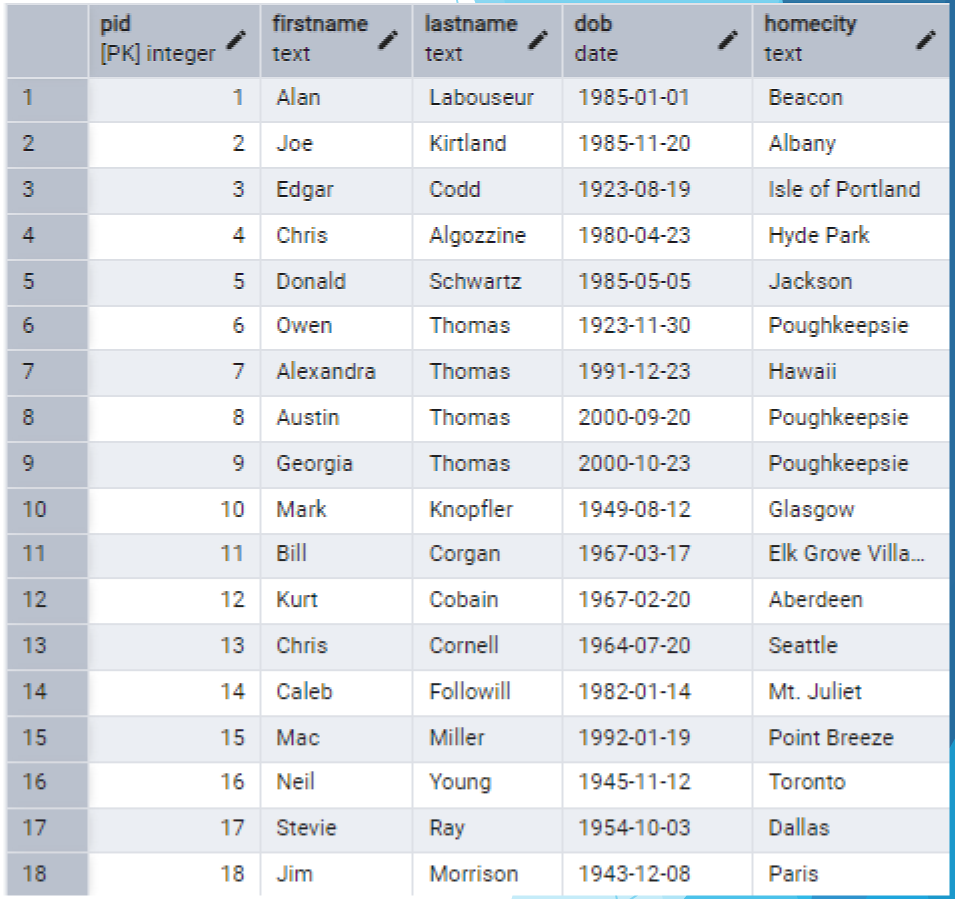
Tables

People Table

Functional Dependencies: pid → firstName, lastName, DOB, homeCity

- The People table contains a unique ID for each Sailor along with their first name, last name, date of birth (DOB), and home city. This table stores common information for the two subtypes Officers and Enlisted. This table identifies individuals across the database and links to other tables such as Officers, Enlisted and Assignments.

```
CREATE TABLE People (
    pid                     int not null,
    firstName               text not null,
    lastName                text not null,
    DOB                     date not null check (DOB <= '2007-04-01'),
    homeCity                text,
    primary key (pid)
    );
```



	pid [PK] integer	firstname text	lastname text	dob date	homecity text
1	1	Alan	Labouseur	1985-01-01	Beacon
2	2	Joe	Kirtland	1985-11-20	Albany
3	3	Edgar	Codd	1923-08-19	Isle of Portland
4	4	Chris	Algozzine	1980-04-23	Hyde Park
5	5	Donald	Schwartz	1985-05-05	Jackson
6	6	Owen	Thomas	1923-11-30	Poughkeepsie
7	7	Alexandra	Thomas	1991-12-23	Hawaii
8	8	Austin	Thomas	2000-09-20	Poughkeepsie
9	9	Georgia	Thomas	2000-10-23	Poughkeepsie
10	10	Mark	Knopfler	1949-08-12	Glasgow
11	11	Bill	Corgan	1967-03-17	Elk Grove Villa...
12	12	Kurt	Cobain	1967-02-20	Aberdeen
13	13	Chris	Cornell	1964-07-20	Seattle
14	14	Caleb	Followill	1982-01-14	Mt. Juliet
15	15	Mac	Miller	1992-01-19	Point Breeze
16	16	Neil	Young	1945-11-12	Toronto
17	17	Stevie	Ray	1954-10-03	Dallas
18	18	Jim	Morrison	1943-12-08	Paris

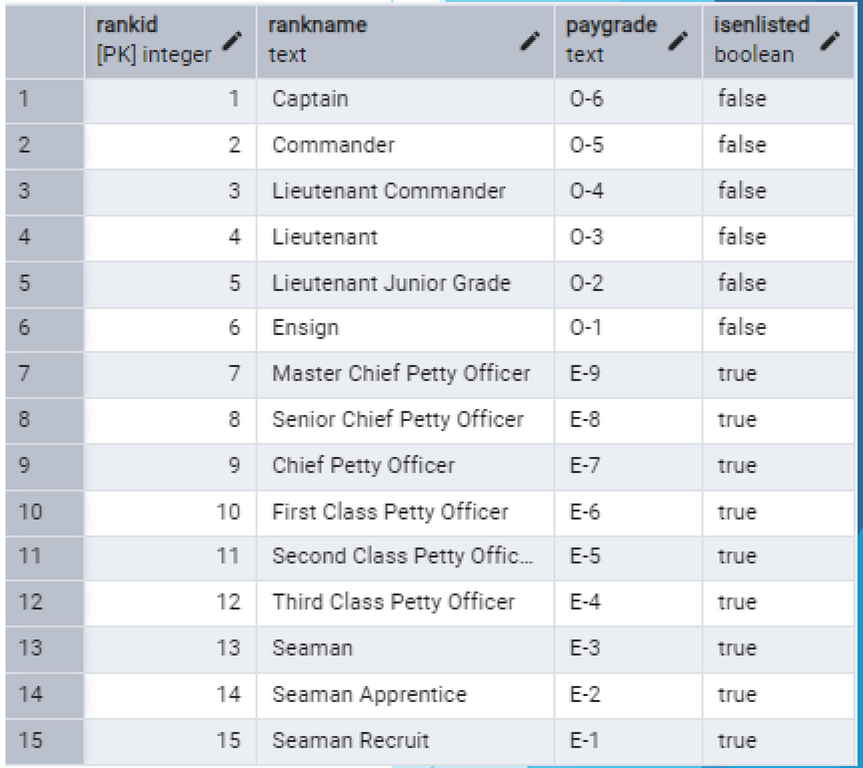
Check constraint to check age if Sailor is of age to serve

Rank Table

Functional Dependencies: rankId → rankName, paygrade, isEnlisted

- The Ranks table defines the ranks that are standard Navy wide. This includes both Officer and Enlisted ranks, paygrade and a Boolean flag that indicates if the rank is part of the enlisted community or not. This table supports the Officers and Enlisted table giving the ability to enforce a rank structure.

```
CREATE TABLE Ranks (
    rankId              int not null,
    rankName            text not null,
    payGrade            text not null,
    isEnlisted          boolean,
    primary key (rankId)
    );
```



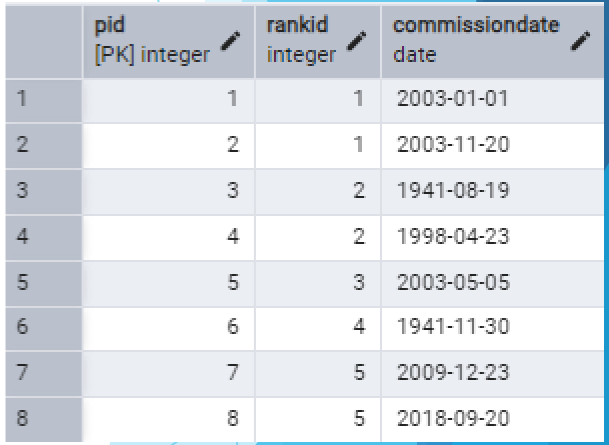
	rankid [PK] integer	rankname text	paygrade text	isenlisted boolean
1	1	Captain	O-6	false
2	2	Commander	O-5	false
3	3	Lieutenant Commander	O-4	false
4	4	Lieutenant	O-3	false
5	5	Lieutenant Junior Grade	O-2	false
6	6	Ensign	O-1	false
7	7	Master Chief Petty Officer	E-9	true
8	8	Senior Chief Petty Officer	E-8	true
9	9	Chief Petty Officer	E-7	true
10	10	First Class Petty Officer	E-6	true
11	11	Second Class Petty Offic...	E-5	true
12	12	Third Class Petty Officer	E-4	true
13	13	Seaman	E-3	true
14	14	Seaman Apprentice	E-2	true
15	15	Seaman Recruit	E-1	true

Officers Table

Functional Dependencies: pid → rankId, commissionDate

- The Officers table contains data for commissioned personnel. Officers are subtype to People. Each Officer must be listed in the People table prior to being inserted into the Officers table, enforcing referential integrity. The pid column serves as the primary key in this table, uniquely identifying each Officer. Each Officer is linked to a rank from the Ranks table. This table also records the date they were commissioned.

```
CREATE TABLE Officers (
    pid                     int not null references People(pid),
    rankId                  int not null references Ranks(rankId),
    commissionDate          date not null check (commissionDate <= '2025-04-01'),
    primary key (pid)
);
```



	pid [PK] integer	rankid integer	commissiondate date
1	1	1	2003-01-01
2	2	1	2003-11-20
3	3	2	1941-08-19
4	4	2	1998-04-23
5	5	3	2003-05-05
6	6	4	1941-11-30
7	7	5	2009-12-23
8	8	5	2018-09-20

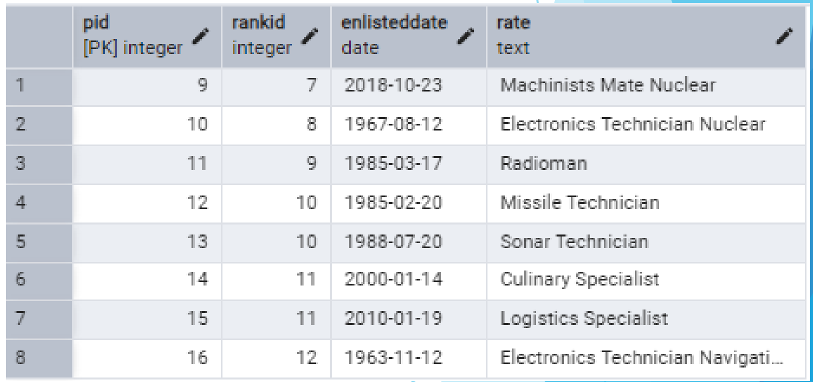
Check constrain to ensure commission date is in the past

Enlisted Table

Functional Dependencies: pid → rankId, enlistedDate, rate

- The Enlisted table contains information on the Enlisted personnel. Similar to Officers, it is a subtype of People and each enlisted member must be first listed in the People table. The pid column serves as the primary key in this table, uniquely identifying each Enlisted member. Each enlisted member is linked to a rank from the Ranks table. A key difference in this table from Officers table is the rate column. Enlisted members are assigned a rate when they enlist in the United States Navy. You can think of rate as the sailor's job title.

```
CREATE TABLE Enlisted (
    pid                     int not null references People(pid),
    rankId                  int not null references Ranks(rankId),
    enlistedDate            date not null check (enlistedDate <= '2025-04-01'),
    rate                    text not null,
    primary key (pid)
);
```



	pid [PK] integer	rankid integer	enlisteddate date	rate text
1	9	7	2018-10-23	Machinists Mate Nuclear
2	10	8	1967-08-12	Electronics Technician Nuclear
3	11	9	1985-03-17	Radioman
4	12	10	1985-02-20	Missile Technician
5	13	10	1988-07-20	Sonar Technician
6	14	11	2000-01-14	Culinary Specialist
7	15	11	2010-01-19	Logistics Specialist
8	16	12	1963-11-12	Electronics Technician Navigati...

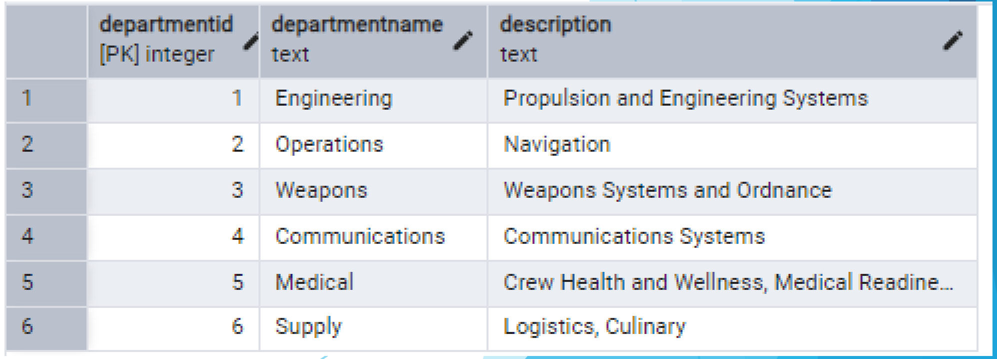
Check constraint to ensure enlisted date is prior to "current date"

Departments Table

Functional Dependencies: departmentId → departmentName, description

- The Departments table defines the major divisions with the Submarine similar to departments in a business setting (Finance, HR). The departments on a Submarine include Engineering, Weapons, Operations, Communications, Medical and Supply. Each department is uniquely identified by departmentId, which serves as the primary key. This table also includes the department name, and a description of the department's responsibilities.

```
CREATE TABLE Departments (
    departmentId            int not null,
    departmentName          text not null,
    description             text,
    primary key (departmentId)
    );
```



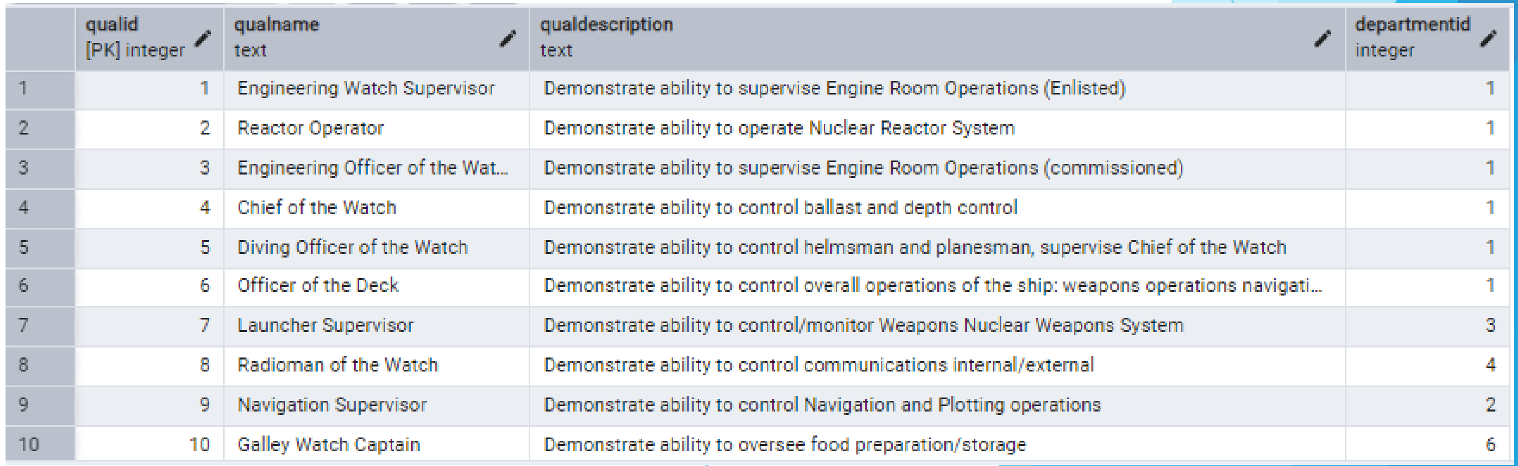
	departmentid [PK] integer	departmentname text	description text
1	1	Engineering	Propulsion and Engineering Systems
2	2	Operations	Navigation
3	3	Weapons	Weapons Systems and Ordnance
4	4	Communications	Communications Systems
5	5	Medical	Crew Health and Wellness, Medical Readine...
6	6	Supply	Logistics, Culinary

Qualification Table

Functional Dependencies: qualId → qualName, qualDescription, departmentId

- The Qualifications Table lists the various certifications that Sailors can achieve. Each qualification is uniquely identified by qualId serving as the primary key. The table also contains the qualification name, a description, and the qualification the department belongs to. This ensures referential integrity with the Departments table by ensuring every qualification is linked to a valid department. Each qualification is linked to a Sailor via the PersonnelQualifications table which tracks each Sailor's individual qualifications.

```
CREATE TABLE Qualifications (
    qualId                  int not null,
    qualName                text not null,
    qualDescription         text,
    departmentId            int not null references
Departments(departmentId),
    primary key (qualId)
);
```



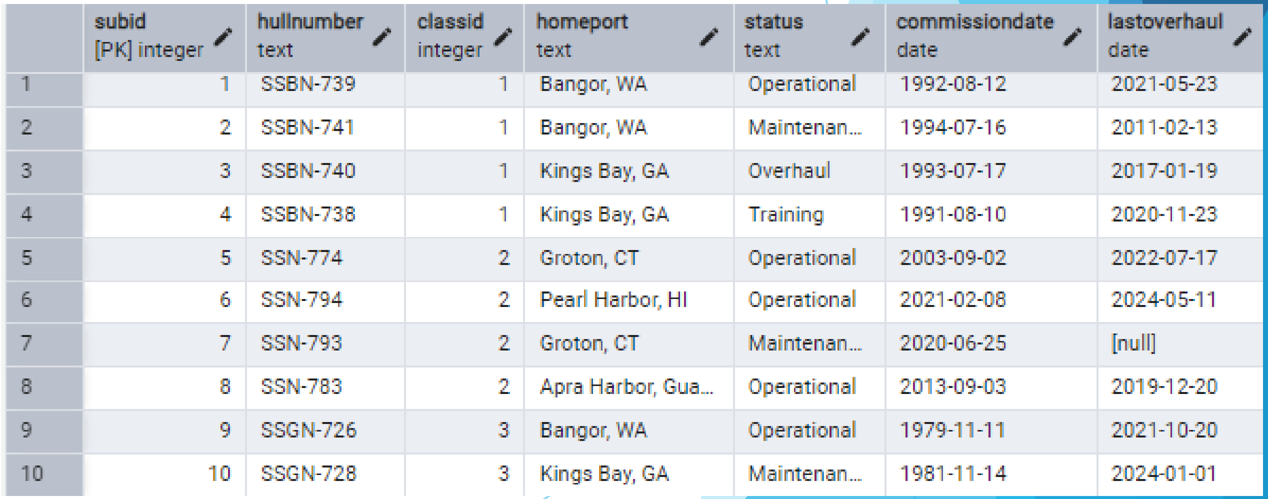
	qualid [PK] integer	qualname text	qualdescription text	departmentid integer
1	1	Engineering Watch Supervisor	Demonstrate ability to supervise Engine Room Operations (Enlisted)	1
2	2	Reactor Operator	Demonstrate ability to operate Nuclear Reactor System	1
3	3	Engineering Officer of the Wat...	Demonstrate ability to supervise Engine Room Operations (commissioned)	1
4	4	Chief of the Watch	Demonstrate ability to control ballast and depth control	1
5	5	Diving Officer of the Watch	Demonstrate ability to control helmsman and planesman, supervise Chief of the Watch	1
6	6	Officer of the Deck	Demonstrate ability to control overall operations of the ship: weapons operations navigati...	1
7	7	Launcher Supervisor	Demonstrate ability to control/monitor Weapons Nuclear Weapons System	3
8	8	Radioman of the Watch	Demonstrate ability to control communications internal/external	4
9	9	Navigation Supervisor	Demonstrate ability to control Navigation and Plotting operations	2
10	10	Galley Watch Captain	Demonstrate ability to oversee food preparation/storage	6

SubmarineClass Table

<u>Functional Dependencies: </u>subClassId → subClassName, boatType, propulsionType, crewCapacity, commissionYear

- The SubmarineClass table defines the different classes of Submarines in the fleet. Each record captures the class name, boat type, propulsion type, crew capacity and the year the class was commissioned. Each class is uniquely identified by subClassId serving as the primary key. This table links back through the classId foreign key in Submarines Table.

```
CREATE TABLE SubmarineClass (
    subClassId              int not null,
    subClassName            text not null,
    boatType                text not null,
    propulsionType          text not null,
    crewCapacity            int not null check (crewCapacity > 0),
    commissionYear          int check (commissionYear <= 2025),
    primary key (subClassId)
    );
```



	subid [PK] integer	hullnumber text	classid integer	homeport text	status text	commissiondate date	lastoverhaul date
1	1	SSBN-739	1	Bangor, WA	Operational	1992-08-12	2021-05-23
2	2	SSBN-741	1	Bangor, WA	Maintenan...	1994-07-16	2011-02-13
3	3	SSBN-740	1	Kings Bay, GA	Overhaul	1993-07-17	2017-01-19
4	4	SSBN-738	1	Kings Bay, GA	Training	1991-08-10	2020-11-23
5	5	SSN-774	2	Groton, CT	Operational	2003-09-02	2022-07-17
6	6	SSN-794	2	Pearl Harbor, HI	Operational	2021-02-08	2024-05-11
7	7	SSN-793	2	Groton, CT	Maintenan...	2020-06-25	[null]
8	8	SSN-783	2	Apra Harbor, Gua...	Operational	2013-09-03	2019-12-20
9	9	SSGN-726	3	Bangor, WA	Operational	1979-11-11	2021-10-20
10	10	SSGN-728	3	Kings Bay, GA	Maintenan...	1981-11-14	2024-01-01

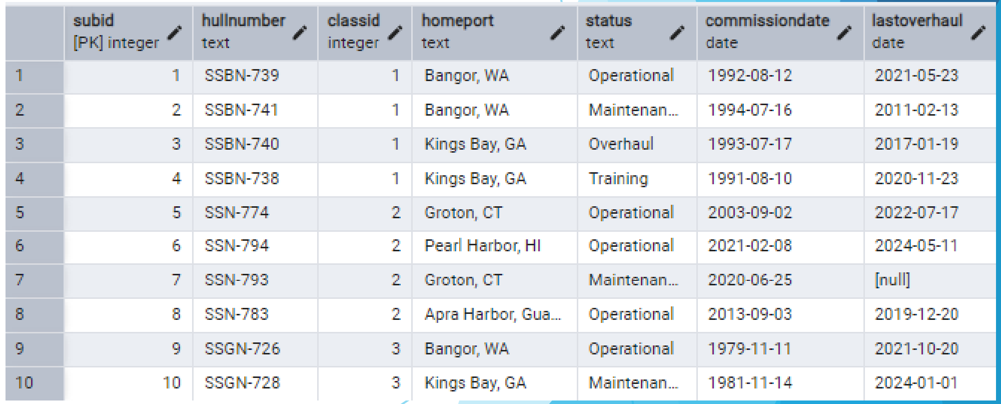
Check constraint to ensure crew capacity is greater than zero

Submarines Table

Functional Dependencies: subId → hullNumber, classId, homeport, status, commissionDate, lastOverhaul

- The Submarine table stores detailed information about each Submarine in the fleet. Each Submarine is uniquely identified by the subId column. The table includes the hull number, homeport, operational status (Operational, Maintenance, Overhaul, Training), commission date and last overhaul date. An overhaul is a rigorous maintenance period where critical maintenance is performed to prolong life of Submarine and add upgrades that cannot be done in a standard maintenance period. Each Submarine is liked to its class through the classId foreign key, referencing the SubmarineClass table.

```
CREATE TABLE Submarines (
    subId                 int not null,
    hullNumber            text not null unique,
    classId               int not null references SubmarineClass(subClassId),
    homeport              text not null,
    status                text not null check (status IN ('Operational', 'Maintenance', 'Training', 'Overhaul', 'Deployed')),
    commissionDate        date not null check (commissionDate <= '2025-04-01'),
    lastOverhaul          date check (lastOverhaul is null OR lastOverhaul <= '2025-04-01'),
    primary key (subId)
);
```



	subid [PK] integer	hullnumber text	classid integer	homeport text	status text	commissiondate date	lastoverhaul date
1	1	SSBN-739	1	Bangor, WA	Operational	1992-08-12	2021-05-23
2	2	SSBN-741	1	Bangor, WA	Maintenan...	1994-07-16	2011-02-13
3	3	SSBN-740	1	Kings Bay, GA	Overhaul	1993-07-17	2017-01-19
4	4	SSBN-738	1	Kings Bay, GA	Training	1991-08-10	2020-11-23
5	5	SSN-774	2	Groton, CT	Operational	2003-09-02	2022-07-17
6	6	SSN-794	2	Pearl Harbor, HI	Operational	2021-02-08	2024-05-11
7	7	SSN-793	2	Groton, CT	Maintenan...	2020-06-25	[null]
8	8	SSN-783	2	Apra Harbor, Gua...	Operational	2013-09-03	2019-12-20
9	9	SSGN-726	3	Bangor, WA	Operational	1979-11-11	2021-10-20
10	10	SSGN-728	3	Kings Bay, GA	Maintenan...	1981-11-14	2024-01-01

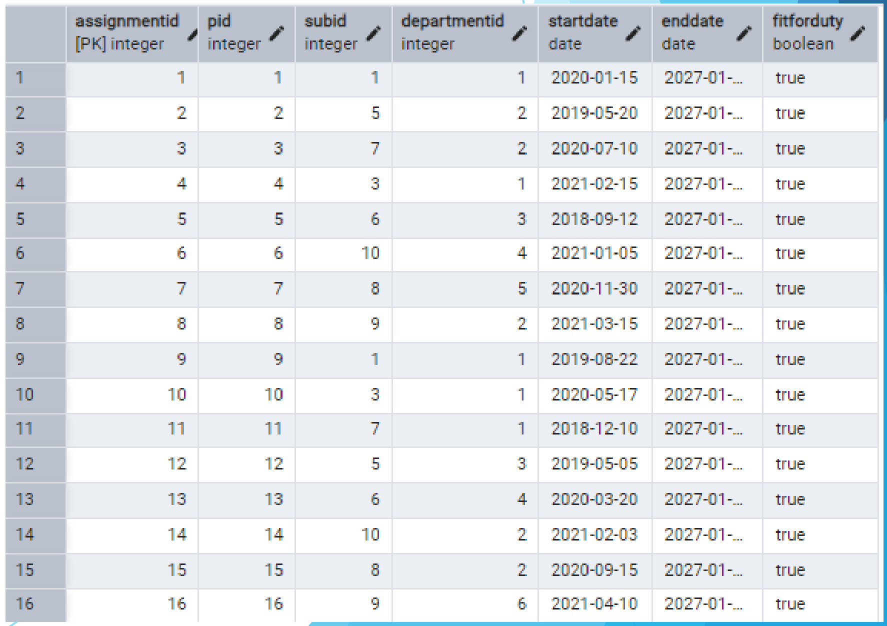
Check constraints of controlling status of Submarine, commission date is prior to "current date", and that the last overhaul is prior to the "current date"

Assignments Table

Functional Dependencies: assignmentId → pid, subId, departmentId, startDate, endDate, fitForDuty

- The Assignments table tracks which personnel are assigned to which Submarine. Each assignment is uniquely identified by assignmentId as the primary key. The table links personnel, submarines and departments through foreign keys, enforcing referential integrity. In addition, it records the start and end date of each individual's assignment and whether the Sailor is fit for duty or not.

```
CREATE TABLE Assignments (
    assignmentId            int not null,
    pid                     int not null references People(pid),
    subId                   int not null references Submarines(subId),
    departmentId            int not null references Departments(departmentId),
    startDate               date not null,
    endDate                 date,
    fitForDuty              boolean,
    primary key (assignmentId)
    );
```



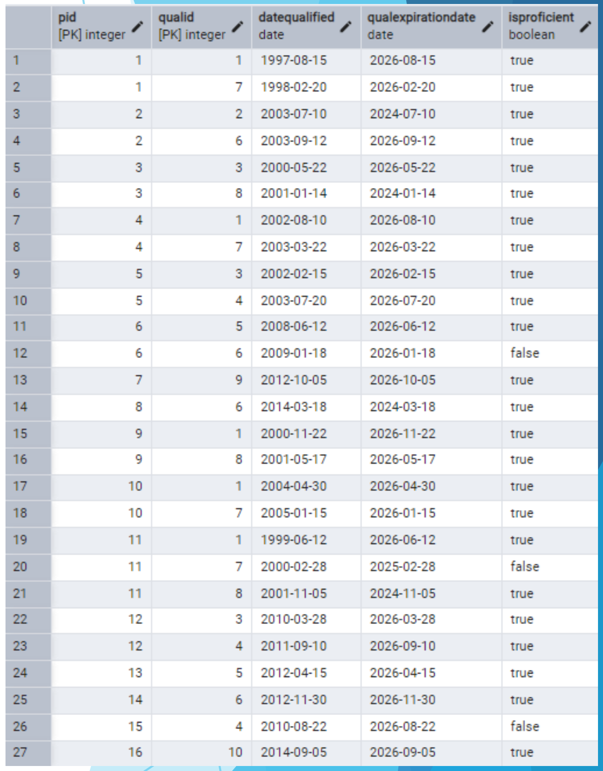
	assignmentid [PK] integer	pid integer	subid integer	departmentid integer	startdate date	enddate date	fitforduty boolean
1	1	1	1	1	2020-01-15	2027-01-...	true
2	2	2	5	2	2019-05-20	2027-01-...	true
3	3	3	7	2	2020-07-10	2027-01-...	true
4	4	4	3	1	2021-02-15	2027-01-...	true
5	5	5	6	3	2018-09-12	2027-01-...	true
6	6	6	10	4	2021-01-05	2027-01-...	true
7	7	7	8	5	2020-11-30	2027-01-...	true
8	8	8	9	2	2021-03-15	2027-01-...	true
9	9	9	1	1	2019-08-22	2027-01-...	true
10	10	10	3	1	2020-05-17	2027-01-...	true
11	11	11	7	1	2018-12-10	2027-01-...	true
12	12	12	5	3	2019-05-05	2027-01-...	true
13	13	13	6	4	2020-03-20	2027-01-...	true
14	14	14	10	2	2021-02-03	2027-01-...	true
15	15	15	8	2	2020-09-15	2027-01-...	true
16	16	16	9	6	2021-04-10	2027-01-...	true

PersonnelQualifications Table

Functional Dependencies: pid, qualId → dateQualified, qualExpirationDate, isProficient

- The PersonnelQualifications table tracks which qualifications each Sailor holds. It uses a composite primary key of pid and qualId to ensure each qualification is unique per person. The table links back to both People and Qualifications through foreign keys. In addition it records when the qualification was completed, its expiration date and whether the Sailor is currently proficient. Qualifications are taken very seriously in the Navy. These qualifications allow you to stand watches where you are operating equipment with thousands of pounds of hydraulic pressure, a nuclear reactor or nuclear missiles. These qualifications have proficiency dates that ensure the watch stander is staying proficient with the given watch.

```
CREATE TABLE PersonnelQualifications (
    pid                          int not null references People(pid),
    qualId                       int not null references Qualifications(qualId),
    dateQualified                date not null check (dateQualified <= '2025-04-01'),
    qualExpirationDate           date not null check (qualExpirationDate > dateQualified),
    isProficient                 boolean,
    primary key (pid, qualId)
);
```



	pid [PK] integer	qualid [PK] integer	datequalified date	qualexpirationdate date	isproficient boolean
1	1	1	1997-08-15	2026-08-15	true
2	1	7	1998-02-20	2026-02-20	true
3	2	2	2003-07-10	2024-07-10	true
4	2	6	2003-09-12	2026-09-12	true
5	3	3	2000-05-22	2026-05-22	true
6	3	8	2001-01-14	2024-01-14	true
7	4	1	2002-08-10	2026-08-10	true
8	4	7	2003-03-22	2026-03-22	true
9	5	3	2002-02-15	2026-02-15	true
10	5	4	2003-07-20	2026-07-20	true
11	6	5	2008-06-12	2026-06-12	true
12	6	6	2009-01-18	2026-01-18	false
13	7	9	2012-10-05	2026-10-05	true
14	8	6	2014-03-18	2024-03-18	true
15	9	1	2000-11-22	2026-11-22	true
16	9	8	2001-05-17	2026-05-17	true
17	10	1	2004-04-30	2026-04-30	true
18	10	7	2005-01-15	2026-01-15	true
19	11	1	1999-06-12	2026-06-12	true
20	11	7	2000-02-28	2025-02-28	false
21	11	8	2001-11-05	2024-11-05	true
22	12	3	2010-03-28	2026-03-28	true
23	12	4	2011-09-10	2026-09-10	true
24	13	5	2012-04-15	2026-04-15	true
25	14	6	2012-11-30	2026-11-30	true
26	15	4	2010-08-22	2026-08-22	false
27	16	10	2014-09-05	2026-09-05	true

Check constraint of date qualified prior to "current date" and that the qualification expiration date is greater than the date the qualification was achieved

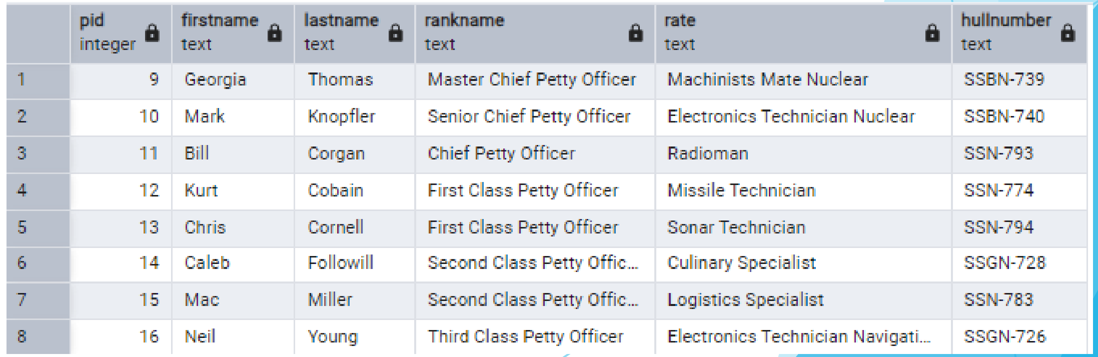
Views

View - enlistedRoster

This view displays the roster of all enlisted personnel, combining their basic personnel details with their rank, rate and hull number they are assigned to. It joins the People, Enlisted, Ranks, Assignments and Submarines table. I found this view to be useful when getting an understanding of the fleet's status.

```
CREATE VIEW enlistedRoster as

    Select p.pid, p.firstName, p.lastName, r.rankName, e.rate, s.hullNumber
from People p inner join Enlisted e on p.pid = e.pid
              inner join Ranks r on e.rankId = r.rankId
              left outer join Assignments a on p.pid = a.pid
              left outer join Submarines s on a.subId = s.subId;
```

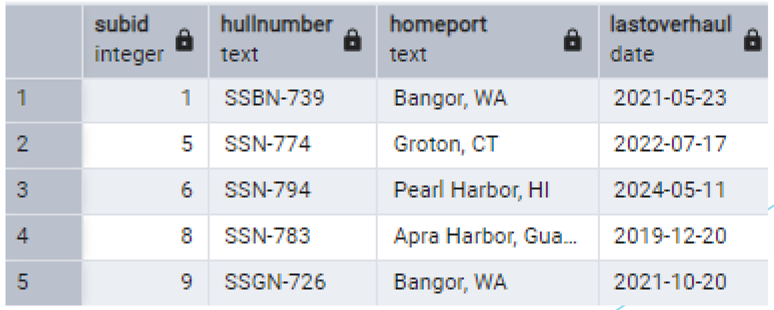
```
    Select *
    from enlistedRoster;
```



	pid	firstname	lastname	rankname	rate	hullnumber
	integer	text	text	text	text	text
1	9	Georgia	Thomas	Master Chief Petty Officer	Machinists Mate Nuclear	SSBN-739
2	10	Mark	Knopfler	Senior Chief Petty Officer	Electronics Technician Nuclear	SSBN-740
3	11	Bill	Corgan	Chief Petty Officer	Radioman	SSN-793
4	12	Kurt	Cobain	First Class Petty Officer	Missile Technician	SSN-774
5	13	Chris	Cornell	First Class Petty Officer	Sonar Technician	SSN-794
6	14	Caleb	Followill	Second Class Petty Offic...	Culinary Specialist	SSGN-728
7	15	Mac	Miller	Second Class Petty Offic...	Logistics Specialist	SSN-783
8	16	Neil	Young	Third Class Petty Officer	Electronics Technician Navigati...	SSGN-726

View – OperationalSubs

- This view provides a list of all Submarines that are currently operational. It displays the subId, hull number, homeport and the date of last overhaul. There is requirements that our military has that requires a certain amount of Submarines to be out to sea at all times. This is useful when planning an integrated schedule involving all Submarines.
-
```
CREATE VIEW operationalSubs as

   Select subId, hullNumber, homeport, lastOverhaul
from Submarines
where status = 'Operational';

   Select *
   from operationalSubs;
```

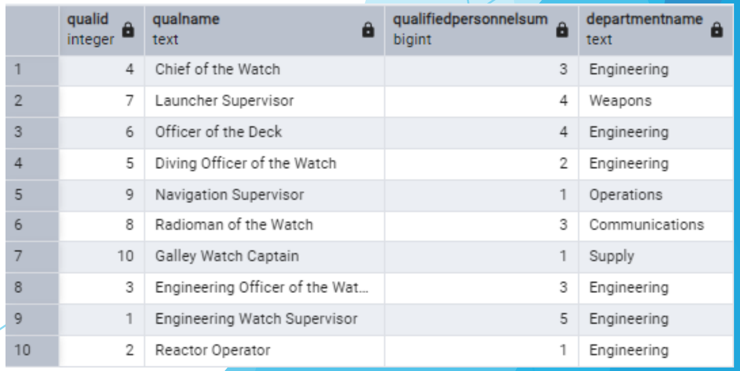


	subid integer	hullnumber text	homeport text	lastoverhaul date
1	1	SSBN-739	Bangor, WA	2021-05-23
2	5	SSN-774	Groton, CT	2022-07-17
3	6	SSN-794	Pearl Harbor, HI	2024-05-11
4	8	SSN-783	Apra Harbor, Gua...	2019-12-20
5	9	SSGN-726	Bangor, WA	2021-10-20

View - qualNumbers

- This view gives a qualification report that shows how may personnel are qualified for each qualification across the fleet. It shows the qualification ID, name, total number of personnel qualified each watch and the associated department the qual belongs to. Without qualified Sailors a Submarine cannot be manned resulting in a non-operational status. You can see how Navy leadership would want to track this.

```
CREATE VIEW qualNumbers as
Select q.qualId, q.qualName, count(pq.pid) as qualifiedPersonnelSum, d.departmentName
from Qualifications q left outer join PersonnelQualifications pq on q.qualId = pq.qualId
                      inner join Departments d on q.departmentId = d.departmentId
                      group by q.qualId, d.departmentName;
```

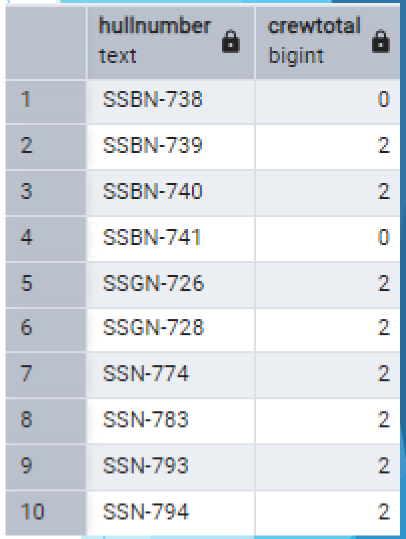


	qualid integer	qualname text	qualifiedpersonnelsum bigint	departmentname text
1	4	Chief of the Watch	3	Engineering
2	7	Launcher Supervisor	4	Weapons
3	6	Officer of the Deck	4	Engineering
4	5	Diving Officer of the Watch	2	Engineering
5	9	Navigation Supervisor	1	Operations
6	8	Radioman of the Watch	3	Communications
7	10	Galley Watch Captain	1	Supply
8	3	Engineering Officer of the Wat...	3	Engineering
9	1	Engineering Watch Supervisor	5	Engineering
10	2	Reactor Operator	1	Engineering

Reports

Report – Crew Total

- This report shows each Submarine's hull number along with the total number of personal assigned. It uses a left outer join between Submarines and Assignments. The report filters for active assignments as there is sample data of crew members with lapsed assignment dates. This can help leadership identified undermanned crews who need support or overmanned crews who can afford to share the wealth.



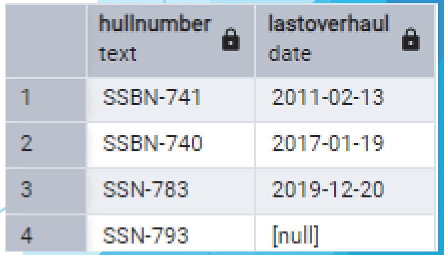
	hullnumber text	crewtotal bigint
1	SSBN-738	0
2	SSBN-739	2
3	SSBN-740	2
4	SSBN-741	0
5	SSGN-726	2
6	SSGN-728	2
7	SSN-774	2
8	SSN-783	2
9	SSN-793	2
10	SSN-794	2

```
Select s.hullNumber, count(a.pid) as crewTotal
from Submarines s left outer join Assignments a on s.subId = a.subId
and (a.endDate is null or a.endDate > '2025-04-01')
group by s.subId, s.hullNumber
order by s.hullNumber;
```

Report – Overdue Overhaul Report

- This report identifies Submarines that are overdue for an overhaul. It checks for Submarines where the last overhaul date is either missing or is older than April 1, 2020. I touch on the issue of hardcoding dates in the known problems portion of the presentation. As discussed earlier, overhauls can prolog the life of a Submarine and is an important factor in maintaining fleet readiness.

-
  ```
  Select hullNumber, lastOverhaul
   from Submarines
   where lastOverhaul is null or lastOverhaul < '2020-04-01’
   order by lastOverhaul;
  ```



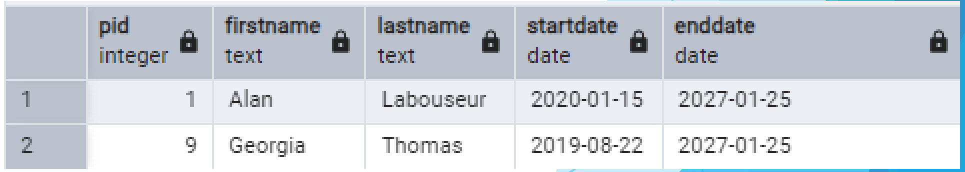
	hullnumber text	lastoverhaul date
1	SSBN-741	2011-02-13
2	SSBN-740	2017-01-19
3	SSN-783	2019-12-20
4	SSN-793	[null]

Stored Procedures

Stored Procedure – CrewAssignments

- This procedure returns a list of personnel currently assigned to a specific submarine, based on the provided hull number. It outputs details like the name of the sailor and their assignment dates. This is a quick way of viewing a full crew roster for a given Submarine.

```
CREATE OR REPLACE FUNCTION crewAssignments (TEXT, REFCURSOR) RETURNS REFCURSOR AS
$$
DECLARE
    p_hullNumber TEXT := $1;
    resultset REFCURSOR :=$2;
BEGIN
    OPEN resultset FOR
        Select p.pid, p.firstName, p.lastName, a.startDate, a.endDate
        from Assignments a inner join People p on a.pid = p.pid
                           inner join Submarines s on a.subId = s.subId
                           where s.hullNumber = p_hullNumber and (a.endDate is null or a.endDate > '2025-04-01')
                           order by p.lastName;
                        return resultset;
END;
$$
LANGUAGE PLPGSQL;

--Test #1

Select crewAssignments('SSBN-739', 'results');
Fetch all from results;

--Test #2

Select crewAssignments('SSN-794', 'results');
Fetch all from results;
```

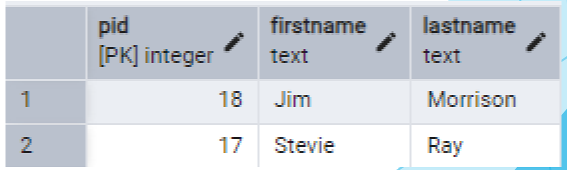


	pid integer	firstname text	lastname text	startdate date	enddate date
1	1	Alan	Labouseur	2020-01-15	2027-01-25
2	9	Georgia	Thomas	2019-08-22	2027-01-25

Stored Procedure - readySpares

- This procedure returns a list of personnel who do not have a current assignment. This is useful for identifying available Sailors who can support a Submarine that is undermanned or needs additional support.

```
CREATE OR REPLACE FUNCTION readySpares(REFCURSOR) RETURNS REFCURSOR AS
$$
DECLARE
    resultset REFCURSOR := $1;
BEGIN
    OPEN resultset FOR
        Select p.pid, p.firstName, p.lastName
        from People p
        where p.pid not in (Select pid
                            from Assignments
                            where (endDate is null or endDate > '2025-04-01')
                            and pid is not null
                            )
        order by p.lastName;
    return resultset;
END;
$$
LANGUAGE PLPGSQL;

--TEST #1

Select readySpares('results')
Fetch all from results;
```

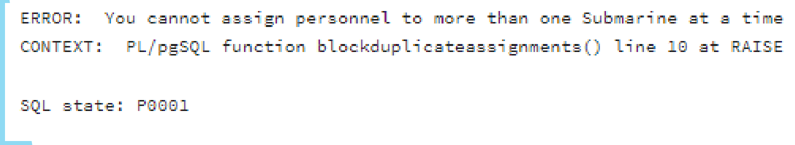


	pid [PK] integer	firstname text	lastname text
1	18	Jim	Morrison
2	17	Stevie	Ray

Triggers

Trigger - duplicateAssignments

- This trigger prevents personnel from being assigned to more than one Submarine at a time. Before an insert/update, the trigger checks if the individual already has a current assignment. If so it prevents commit and provides feedback via a raise exception

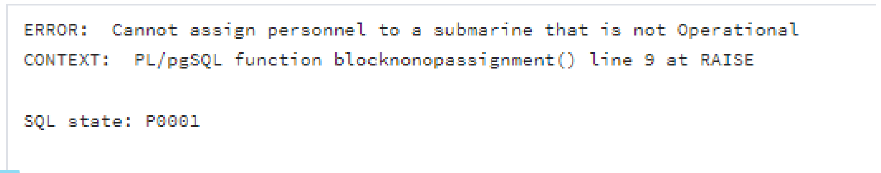


```
ERROR:  You cannot assign personnel to more than one Submarine at a time
CONTEXT:  PL/pgSQL function blockduplicateassignments() line 10 at RAISE

SQL state: P0001
```

```
CREATE OR REPLACE FUNCTION blockDuplicateAssignments()
RETURNS TRIGGER AS
$$
BEGIN
    IF EXISTS (
        Select a.assignmentId
        from Assignments a
        where a.pid = NEW.pid and (a.endDate is null or a.endDate > '2025-04-01')
                              and a.assignmentId != NEW.assignmentId
                                  )
              THEN
                RAISE EXCEPTION 'You cannot assign personnel to more than one Submarine at a time';
    END IF;
    RETURN NEW;
END;
$$
LANGUAGE PLPGSQL;

CREATE TRIGGER duplicateAssignments
BEFORE INSERT OR UPDATE ON Assignments
FOR EACH ROW
EXECUTE PROCEDURE blockDuplicateAssignments();

--Test of Trigger trying to assign pid 1 to subId 1 then 5 when they already have an assignment

INSERT INTO ASSIGNMENTS (assignmentId, pid, subId, departmentId, startDate, endDate, fitForDuty)
VALUES

(30, 1, 1, 1, '2025-07-23', null, true),
(31, 1, 5, 1, '2025-07-23', null, true);
```

Trigger - strategicAssignment

- This trigger prevents a personnel from being assigned to a Submarine that is not operational. Prior to an insert/update, the trigger checks the operational status of the Submarine. If the assigned Submarine is not operational, the trigger provides feedback via a raise exception.



```
ERROR:  Cannot assign personnel to a submarine that is not Operational
CONTEXT:  PL/pgSQL function blocknonopassignment() line 9 at RAISE

SQL state: P0001
```

```
CREATE OR REPLACE FUNCTION blockNonOpAssignment()
RETURNS TRIGGER AS
$$
BEGIN
    IF EXISTS (
        Select s.subId
        from Submarines s
        where s.subId = NEW.subId and s.status not in ('Operational', 'Deployed')
    )
    THEN
        RAISE EXCEPTION 'Cannot assign personnel to a submarine that is not Operational';
    END IF;
    RETURN NEW;
END;
$$
LANGUAGE PLPGSQL;

CREATE TRIGGER strategicAssignment
BEFORE INSERT OR UPDATE on Assignments
FOR EACH ROW
EXECUTE PROCEDURE blockNonOpAssignment();

--Test by trying to assign personnel to a non operational boat
--Test #1 - Trigger should alert
--Insert into People table first
INSERT INTO People (pid, firstName, lastName, DOB, homeCity)
VALUES
(55, 'Testy', 'McTester', '1989-01-01', 'Whoville');
INSERT INTO Officers (pid, rankId, commissionDate)
VALUES

(55, 4, '2015-01-01');

INSERT INTO Assignments (assignmentId, pid, subId, departmentId, startDate, endDate, fitForDuty)
VALUES
(55, 55, 3, 1, '2025-04-04', null, true);
```

Security & Access Control

- Control access of a Database is a extremely important aspect. This is managed through grant and revoke statements. Below is an outline of the roles and what privileges they are granted. Please see sql script for statements.
- Commodore – Full access to all tables and permission to execute all stored procedures. The Commodore oversees an entire Submarine Squadron.
- Admin Officer – Also granted full access to all tables and execution of stored procedures. The Administration Officer reports to the Commodore and handles manning in the Submarine force.
- Submarine CO, XO, COB – these members are limited to Select privileges only. This is read-only access. They are not responsible for maintain these records as that is the Admin Officers responsibility.

Known Problems

- Hardcoded Date issue – Throughout the script I have hard coded the current date to 2025-04-01 instead of using current_date.
- The trigger that enforces age requirements for the military is also covered under the check constrain in DOB. This is redundant unless the check constraint is removed. Submarines are notorious for redundant systems in the event of a failure…I enforced that idea here.

Future Enhancements

- Implement current_date so that the tracking of elements such as proficiency/qualification is accurate and adaptable to real-time operations.
- Expand roles to allow other leaders subordinate to the existing roles be able to interact with the database
- Develop a front end for the database with use feedback and error messages
- Improving constraints
- Create Indexes that can support a large data set
- Incorporate other vessels into database such as aircraft carriers, destroyers etc. to capture a full view of Naval operations.
- Add the following tables: Watchstations, WatchPrereqs, PersonnelWatchstations to improve the capability of leadership in making staffing decisions. These tables would show all existing watches, the prerequisites for each watch and who is qualified each watch.