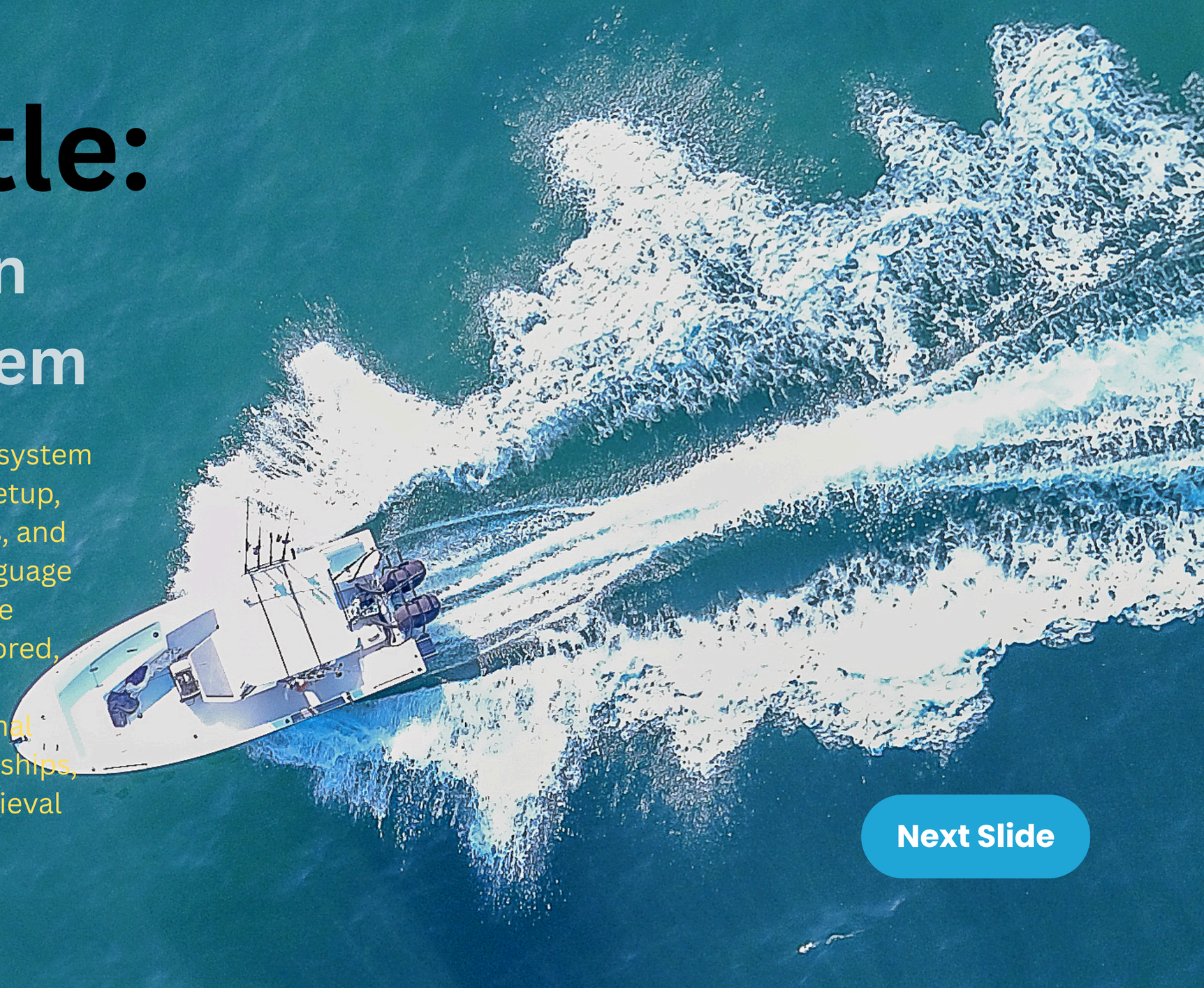
Project Title:

Boat Reservation Management System

This project presents a relational database system designed to manage a boat reservation setup, involving information about boats, sailors, and reservations. Using Structured Query Language (SQL), we explore how data from these interrelated entities can be efficiently stored, manipulated, and queried.

This system demonstrates core relational database concepts, such as entity relationships, foreign keys, normalization, and data retrieval through SQL operations.



Next Slide

Objective:

The main goal of this project is to implement a structured and normalized SQL database that
Stores details about sailors (like name, age, and rating)

- Maintains a list of boats (with color and name)
- Tracks which sailor reserves which boat on which date

We use SQL queries to answer practical questions such as:

- Which boats have been reserved?
- Which sailors have the highest ratings?
- What are the most popular boat colors?
- On what dates were boats reserved?

Entities and Relationships:

Sailor: Contains details of each sailor (ID, name, age, rating)

1. Boat: Contains information on boats (ID, name, color)
2. Reserve: Links sailors and boats with reservation dates

These entities are related through foreign keys and represent a many-to-many relationship between sailors and boats through the Reserve table

Tools & Technologies Used:

- Database: MySQL / PostgreSQL / SQLite
- Language: SQL
- Design Tools: ERD diagram tools like dbdiagram.io (optional)

Scope of the Project:

The project is ideal for simulating a small-scale boat reservation system. It allows users to:

- Create and manage reservation records
- Retrieve useful insights (e.g., most active sailors, frequently reserved boats)
- Enforce data integrity through SQL constraints

Expected Outcomes:

- A normalized database with tables for Sailor, Boat, and Reserve
- Correct usage of primary and foreign keys
- A series of optimized SQL queries to retrieve actionable information
- Hands-on understanding of joins, subqueries, group by, and aggregate functions

Consider the following schema:

Sailors (sid, sname, rating, age)

Boats (bid, bname, color)

Reserves (sid, bid, day(date))

	sid	bid	day
▶	1	101	2025-05-09
	2	102	2025-05-09
	3	103	2025-05-10
	4	104	2025-05-11
	5	105	2025-05-12
	1	102	2025-05-13
	2	103	2025-05-14
	3	104	2025-05-15
	4	105	2025-05-16
	5	101	2025-05-17
	5	102	2025-05-17

	bid	bname	colour
▶	101	Voyager	Red
	102	Explorer	Green
	103	Seafarer	Blue
	104	Mariner	Red
	105	Wanderer	Green
*	NULL	NULL	NULL

	sid	sname	rating	age
▶	1	Bob	5	25
	2	Alice	3	30
	3	Charlie	4	28
	4	David	2	21
	5	Eve	5	24
*	NULL	NULL	NULL	NULL

- I've included all the information about the sailors who have reserved boat number 101.
- 2. Find the name of the boat reserved by Bob
- 3. Find the names of sailors who have reserved a red boat, and list in the order of age.

SQL File 16* SQL File 8* SQL File 5* sailors boats reserve

Limit to 1000 rows

```
1  select* from sailors s
2  join reserve r on s.sid = r.sid
3  where r.bid = 101
4  #zaid
5
6
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sid	sname	rating	age	sid	bid	day
1	Bob	5	25	1	101	2025-05-09
3	Charlie	4	28	3	101	2025-05-10

SQL File 16* SQL File 8* SQL File 5* sailors boats reserve

Limit to 1000 rows

```
1  select * from boats b
2  join reserve r on b.bid = r.bid
3  join sailors s on r.sid = s.sid
4  where sname = 'bob'
5  #zaid
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

bid	bname	colour	sid	bid	day	sid	sname	rating	age
101	Voyager	Red	1	101	2025-05-09	1	Bob	5	25
105	Wanderer	Green	1	105	2025-05-12	1	Bob	5	25

SQL File 16* | SQL File 8* | SQL File 5* | sailors | boats | reserve | SQL File 8* | SQL File 9*

Limit to 1000 rows

```
1  select * from sailors s
2  join reserve r on s.sid = r.sid
3  join boats b on r.bid = b.bid
4  where colour = "red"
5  order by age asc
6  #zaid
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sid	sname	rating	age	sid	bid	day	bid	bname	colour
5	Eve	5	24	5	104	2025-05-11	104	Mariner	Red
1	Bob	5	25	1	101	2025-05-09	101	Voyager	Red
3	Charlie	4	28	3	101	2025-05-10	101	Voyager	Red

- Find the names of sailors who have reserved at least one boat
- Find the IDs and names of sailors who have reserved two different boats on the same day.
- Find the IDs of sailors who have reserved a red boat or a

Database Server Tools Scripting Help

SQL File 16* SQL File 8* SQL File 5* sailors boats reserve SQL F

Create a new table in the active schema in connected server

Limit to 1000 rows

```
1 •  SELECT s.sname
2    FROM Sailors s
3    JOIN Reserve r ON s.sid = r.sid
4    GROUP BY s.sid, s.sname;
5    #zaid
6
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sname
Bob
Alice
Charlie
David
Eve

QL File 16* SQL File 8* SQL File 5* sailors boats SQL File 8* SQL File 9* SQL File 10

Limit to 1000 rows

```
1 •  SELECT s.sid, s.sname
2    FROM Sailors s
3    JOIN Reserve r1 ON s.sid = r1.sid
4    JOIN Reserve r2 ON s.sid = r2.sid AND r1.bid <> r2.bid AND r1.day = r2.day
5    group by sid
6    #zaid
7
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sid	sname
5	Eve

- Find the name and the age ofthe youngest sailor.
- Count the number of different sailor names.
- Find the average age of sailors for each rating level.
- Find the average age of sailors for each rating level that has at least two sailors

```
SQL File 16*  SQL File 8*  SQL File 5*  sailors  boats

1  select s.sid from sailors s
2  join reserve r on s.sid = r.sid
3  join boats b on r.bid = b.bid
4  where colour in ( 'red' , 'green')
5  group by sid;
6  #zaid
```

Result Grid | Filter Rows: | Export:

sid
1
2
4
5
3

```
SQL File 16*  SQL File 8*  SQL File 5*  sailors  boats  SQL File 14*

Create a new stored procedure in the active schema in the connected server

1  SELECT s.sname, s.age
2  FROM Sailors s
3  ORDER BY s.age
4  LIMIT 1
5  #zaid
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows

sname	age
David	21

SQL File 16* SQL File 8* SQL File 5* sailors boats SQL File 1

Limit to 1000 rows

```
1  select count(sname)
2  from sailors
3  #zaid
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

count(sname)
5

SQL File 16* SQL File 8* SQL File 5* sailors boats SQL File 14

Limit to 1000 rows

```
1  SELECT rating, AVG(age) AS avg_age
2  FROM Sailors
3  group by rating
4  #zaid
5
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

rating	avg_age
5	24.5000
3	30.0000
4	28.0000

SQL File 16* | SQL File 8* | SQL File 5* | sailors | boats | SQL File 14* | SQL File 16*

Limit to 1000 rows

```
1   SELECT rating, AVG(age) AS avg_age
2   FROM Sailors
3   group by rating
4   having count(sid) >= 2
5   #zaid
6
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

rating	avg_age
5	24.5000

Performing some of the above queries by sub-query method

SQL File 21* SQL File 22* SQL File 23* SQL File 24*

Limit to 1000 rows

```sql
SELECT sname
FROM Sailors
WHERE sid IN (
    SELECT DISTINCT sid
    FROM Reserve
);
#zaid
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sname
Bob
Alice
Charlie
David
Eve

SQL File 21* SQL File 22* SQL File 23* SQL File 24* SQL File 25*

Limit to 1000 rows

```sql
select sname , sid from sailors
where sid in ( select r1.sid from reserve r1 , reserve r2 where r1.sid =r2.sid and r1.bid<> r2.bid and r1.day =r1.day)
#zaid
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

sname	sid
Bob	1
Alice	2
Charlie	3
David	4
Eve	5
NULL	NULL

SQL File 21* SQL File 22* SQL File 23* SQL File 24* SQL File 26* SQL File 27* SQL File 28*

Limit to 1000 rows

```sql
SELECT rating, (
    SELECT AVG(age)
    FROM Sailors s2
    WHERE s2.rating = s1.rating
) AS avg_age
FROM Sailors s1
GROUP BY rating;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

rating	avg_age
5	24.5000
3	30.0000
4	28.0000
2	21.0000

Query Database Server Tools Scripting Help

SQL File 21* SQL File 22* SQL File 23* SQL File 24* SQL File 26* SQL File 27*

Limit to 1000 rows

```sql
select  distinct sid from reserve
where bid in ( select bid from boats where colour in ('red' or 'green'))
#zaid
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sid
1
2
3
4
5