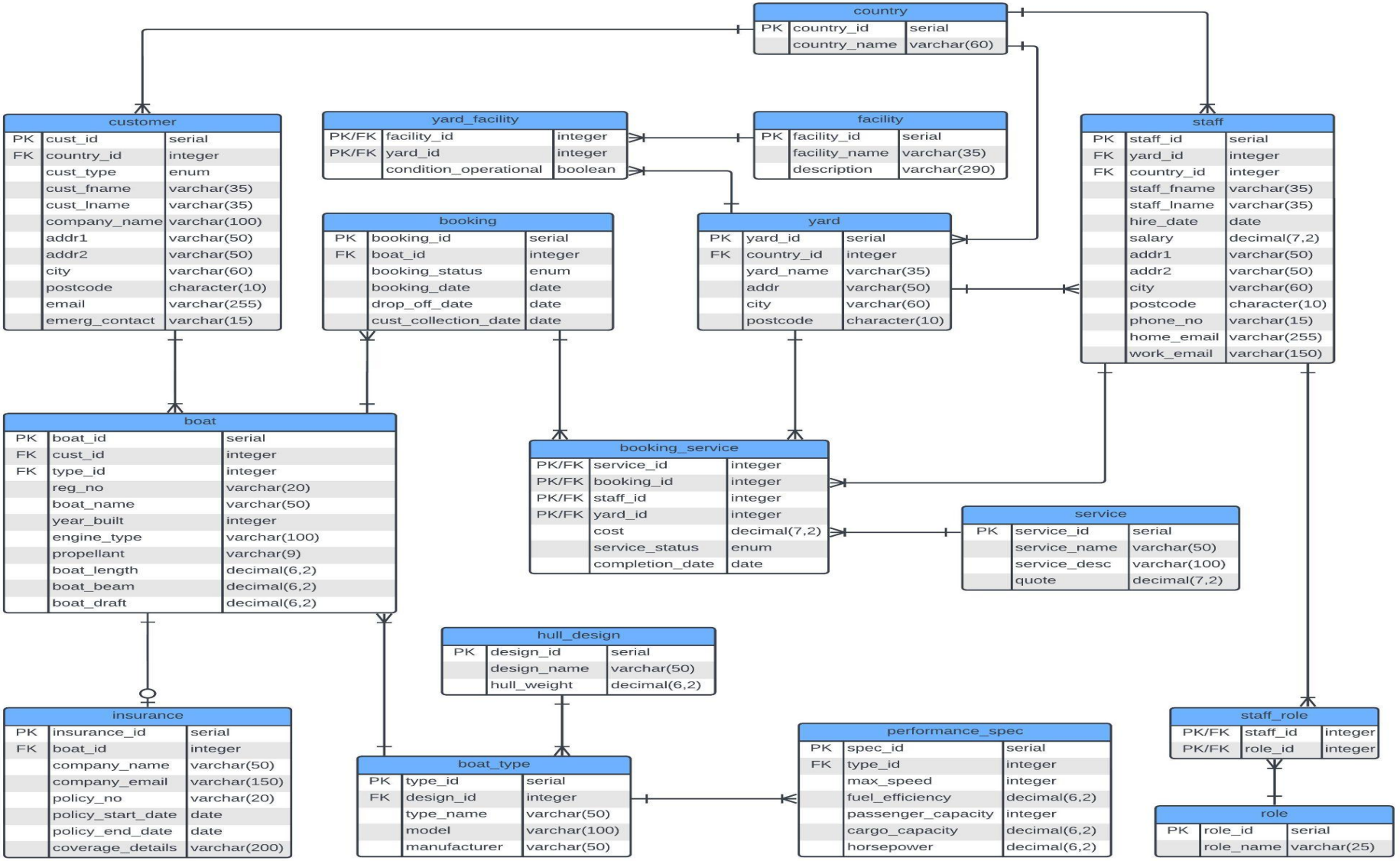
Group 1

ENTITY RELATIONSHIP DIAGRAM



ASSUMPTIONS

Each staff member, customer, boat, service, boatyard, etc., can be uniquely identified using appropriate identifiers (e.g., staff ID, customer ID).
Units of measurement for attributes like dimensions, capacity, and performance specs are consistent across entries.
Customer type can be either company or individual but not both.
If the customer type is individual then the company name is null.
If the customer first and last name would be null if the customer type is company.
A service can have a status of completed or ongoing.
A boat can have zero or one insurance
Not every boat has an engine.
A boat can have only one owner but a customer may have one or many boats.
A booking can contain multiple services
A booking can only belong to one boat.
Service cost is the final amount incurred in repair/maintenance and is unique to extent of maintenance a boat needs
Service quote is the first estimate for a particular kind of service
A boat can be linked to one or more bookings, in the future or past.
A booking can have a status of booked or prebooked (service has not been confirmed)
Payment is made after service(s) have been completed.
One Yard can have different services.
Each yard can host multiple facilities, and a facility may be available at different yards.
Many members of the staff can perform on many different services
Staff works at only one yard

DATA DICTIONARY

COUNTRY						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
country_id	PK	**Y**	SERIAL			
country_name			CHARACTER VARYING(60)	NOT NULL		The name of the country.

ROLE						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
role_id	PK	**Y**	SERIAL			
role_name			CHARACTER VARYING(25)	NOT NULL		The name of the role, e.g., “Manager”, “Cleaner”, etc.

CUSTOMER						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
cust_id	PK	**Y**	SERIAL			
country_id	FK	**Y**	INTEGER	NOT NULL	country.country_id	
cust_type		**Y**	ENUMERATION(customer_type)	CHECK (cust_type IN ('individual', 'company'))		The type of customer, i.e, individual or company.
cust_fname			CHARACTER VARYING(35)			The customer’s first name
cust_lname			CHARACTER VARYING(35)			The customer’s last name
company_name			CHARACTER VARYING(100)			The company name
addr1			CHARACTER VARYING(50)	NOT NULL		The customer’s first line of address
addr2			CHARACTER VARYING(50)			The customer’s second line of address(optional).
city			CHARACTER VARYING(60)	NOT NULL		
postcode			CHARACTER(10)			
email			CHARACTER VARYING(255)	NOT NULL		Customer ‘s email address.
emerg_contact			CHARACTER VARYING(15)	NOT NULL		The customer’s emergency contact number.

HULL_DESIGN						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
design_id	PK	**Y**	SERIAL			
design_name			CHARACTER VARYING(50)	NOT NULL		The name for the hull design.
hull_weight			DECIMAL(6,2)			The weight of the hull design and it will be measured in kg.

SERVICE						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
service_id	PK	**Y**	SERIAL			
service_name			CHARACTER VARYING(50)	NOT NULL		The name of the service given.
service_desc			CHARACTER VARYING(100)	NOT NULL		Describing the service rendered to you.
quote			DECIMAL(7,2)	NOT NULL		The quote/cost of the service and the currency will be in pounds.

YARD						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
yard_id	PK	**Y**	SERIAL			
country_id	FK	**Y**	INTEGER		country.country_id	
yard_name			CHARACTER VARYING(35)	NOT NULL		The name of the yard.
addr			CHARACTER VARYING(50)	NOT NULL		The address of the yard.
city			CHARACTER VARYING(60)	NOT NULL		The city of where the yard is located.
postcode			CHARACTER(10)			

FACILITY						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
facility_id	PK	**Y**	SERIAL			
facility_name			CHARACTER VARYING(35)	NOT NULL		The name of the facility.
description			CHARACTER VARYING(290)	NOT NULL		Services that take place in the facility.

BOAT_TYPE						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
type_id	PK	**Y**	SERIAL			
design_id	FK	**Y**	INTEGER	NOT NULL	hull_design.design_id	
type_name		**Y**	CHARACTER VARYING(50)	NOT NULL		The name of the boat type.
model			CHARACTER VARYING(100)	NOT NULL		The model of boat.
manufacturer			CHARACTER VARYING(50)	NOT NULL		The manufacturer or company that produced the boat.

YARD_FACILITY						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
facility_id	PK/FK	**Y**	INTEGER	NOT NULL	facility.facility_id	
yard_id	PK/FK	**Y**	INTEGER	NOT NULL	yard.yard_id	
condition_operational			BOOLEAN	NOT NULL		A boolean indicating whether the facility is operational.

STAFF						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
staff_id	PK	**Y**	SERIAL			
yard_id	FK	**Y**	INTEGER	NOT NULL	yard.yard_id	
country_id	FK	**Y**	INTEGER	NOT NULL	country.country_id	
staff_fname			CHARACTER VARYING(35)	NOT NULL		The first name of the staff.
staff_lname			CHARACTER VARYING(35)	NOT NULL		The last name of the staff.
hire_date			DATE	NOT NULL		The date when the employee got hired.
salary			DECIMAL(7,2)	NOT NULL		The staff salary.
addr1			CHARACTER VARYING(50)	NOT NULL		The first line of address of the staff.
addr2			CHARACTER VARYING(50)			The second line of address of the staff(optional).
city			CHARACTER VARYING(60)	NOT NULL		The city in which the employee works.
postcode			CHARACTER(10)			
phone_no			CHARACTER VARYING(15)	NOT NULL		The staff’s phone number.
home_email			CHARACTER VARYING(255)	NOT NULL		The staff’s home email address.
work_email		**Y**	CHARACTER	NOT NULL; UNIQUE		The staff work email address.

			VARYING(150)			

STAFF_ROLE						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
staff_id	PK/FK	**Y**	INTEGER	NOT NULL	staff.staff_id	
role_id	PK/FK	**Y**	INTEGER	NOT NULL	role.role_id	

PERFORMANCE_SPEC						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
spec_id	PK	**Y**	SERIAL			
type_id	FK	**Y**	INTEGER	NOT NULL	boat_type.type_id	
max_speed			INTEGER			Maximum speed of the boat and it will be measured knots (kn).
fuel_efficiency			DECIMAL(6,2)			The fuel efficiency of the boat and it will be measured in gallons per hour (gph).
passenger_capacity			INTEGER	NOT NULL		The maximum number of passengers the boat can accommodate.
cargo_capacity			DECIMAL(6,2)			The maximum cargo capacity of the boat and it will be measured in cubic feet(ft^3).

horsepower			DECIMAL(6,2)			The Horsepower of the boat's engine and it will be measured in horsepower(hp).

BOAT						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
boat_id	PK	**Y**	SERIAL			
cust_id	FK	**Y**	INTEGER	NOT NULL	customer.cust_id	
type_id	FK	**Y**	INTEGER	NOT NULL	boat_type.type_id	
reg_no		**Y**	CHARACTER VARYING(20)	NOT NULL;UNIQUE		The registration number of the boat.
boat_name			CHARACTER VARYING(50)	NOT NULL		The name of the boat.
year_built			INTEGER	NOT NULL		The year the boat was built.
engine_type			CHARACTER VARYING(100)			The type of engine as regards to installation position in the boat.
propellant			CHARACTER VARYING(9)	NOT NULL		The substance or energy source utilised to generate propulsion for the boat, e.g; Fuel, wind, electric power.
boat_length			DECIMAL(6,2)	NOT NULL		The length of the boat in metres(m).
boat_beam			DECIMAL(6,2)	NOT NULL		The width/breadth of the boat at its widest point in metres(m)
boat_draft			DECIMAL(6,2)	NOT NULL		The vertical distance between the waterline and the lowest point of the hull in metres(m)

INSURANCE						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
insurance_id	PK	**Y**	SERIAL			
boat_id	FK	**Y**	INTEGER	NOT NULL	customer_boat.boat_id	
company_name			CHARACTER VARYING(50)	NOT NULL		The name of the insurance company
company_email			CHARACTER VARYING(150)	NOT NULL		The insurance company email.
policy_no			CHARACTER VARYING(20)	NOT NULL		The boat policy number given by the company
policy_start_date			DATE	NOT NULL		The date at which the insurance policy begins.
policy_end_date			DATE	CHECK (policy_end_date > policy_start_date),NOT NULL		The date at which the insurance policy ends.
coverage_details			CHARACTER VARYING(200)	NOT NULL		The details about the coverage provided from the insurance policy.

BOOKING						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
booking_id	PK	**Y**	SERIAL			
boat_id	FK	**Y**	INTEGER	NOT NULL	customer_boat.boat_id	
booking_status			ENUMERATION(bk_status)	CHECK(booking_status IN('booked', 'prebooked')), NOT NULL		The status of the booking, e.g., Booked and Pre Booked.
booking_date			DATE	NOT NULL		The date the booking was made.
drop_off_date			DATE	CHECK (drop_off_date >= booking_date), NOT NULL		The date at which the boat is scheduled to be dropped off.
cust_collection_date			DATE	CHECK (cust_collection_date > drop_off_date),NOT NULL		The date at which the customer collected his/her boat.

BOOKING_SERVICE						
Attribute_Name	***KEY***	***INDEX***	***Data Type & Size***	***Domains & Constraints***	***FK Reference***	***Description***
service_id	PK/FK	**Y**	INTEGER	NOT NULL	service.service_id	
booking_id	PK/FK	**Y**	INTEGER	NOT NULL	booking.booking_id	
staff_id	PK/FK	**Y**	INTEGER	NOT NULL	staff.staff_id	
yard_id	PK/FK	**Y**	INTEGER	NOT NULL	yard.yard_id	
cost		**Y**	DECIMAL(7,2)	NOT NULL		The cost of the service and its currency will be in pounds.
service_status			ENUMERATION?(s_status)	CHECK (service_status IN(‘completed’, ‘ongoing’’), NOT NULL		Status of the service, indicating whether it is completed or ongoing.
completion_date			DATE	NOT NULL		The date when the service is completed.

QUERY SCREENSHOTS

```sql
--QUERY 1
--Description: Retrieves a list of the top 30 customers who have spent the most on services, along with the corresponding bookings.
SELECT
    CASE
        WHEN C.CUST_TYPE = 'Company' THEN c.COMPANY_NAME
        WHEN C.CUST_TYPE = 'Individual' THEN CONCAT(c.CUST_FNAME, ' ', c.CUST_LNAME)
    END AS "Customer",
    c.EMAIL AS "Customer Email",
    COUNT(DISTINCT bk.BOOKING_ID) AS "Total Bookings",
    SUM(bs.COST) AS "Total Spending (£)"
FROM
    CUSTOMER c
    JOIN BOAT b ON c.CUST_ID = b.CUST_ID
    JOIN BOOKING bk ON b.BOAT_ID = bk.BOAT_ID
    JOIN BOOKING_SERVICE bs ON bk.BOOKING_ID = bs.BOOKING_ID
GROUP BY
    c.CUST_ID, c.CUST_FNAME, c.CUST_LNAME, c.email
HAVING
    COUNT(DISTINCT bk.BOOKING_ID) > 0
ORDER BY
    "Total Spending (£)" DESC
LIMIT 30;
```

```
dbprin_cw-# LIMIT 30;
      Customer       |        Customer Email         | Total Bookings | Total Spending (£)
---------------------+-------------------------------+----------------+--------------------
 Isabella Whitlock   | isabella.whitlock@gmail.com   |              2 |            1947.06
 Tamas Whacket       | tamas.whacket@gmail.com       |              2 |            1922.79
 Evvy Vondra         | evvy.vondra@gmail.com         |              2 |            1847.91
 Gretchen Braban     | gretchen.braban@gmail.com     |              2 |            1787.89
 Feodor Leyzell      | feodor.leyzell@gmail.com      |              2 |            1770.09
 Jere Bossom         | jere.bossom@gmail.com         |              2 |            1740.31
 Jacki Sokell        | jacki.sokell@gmail.com        |              2 |            1674.78
 Olag Darrow         | olag.darrow@gmail.com         |              2 |            1656.22
 Olympie Hussy       | olympie.hussy@gmail.com       |              2 |            1653.61
 Marleah Greatbanks  | marleah.greatbanks@gmail.com  |              2 |            1650.09
 Bettina Kibblewhite | bettina.kibblewhite@gmail.com |              2 |            1617.03
 Sharline Canny      | sharline.canny@gmail.com      |              2 |            1593.99
 Mala Dymick         | mala.dymick@gmail.com         |              2 |            1593.87
 Idette Fleay        | idette.fleay@gmail.com        |              2 |            1593.72
 Cleon Pyffe         | cleon.pyffe@gmail.com         |              2 |            1590.83
 Leah Shevlane       | leah.shevlane@gmail.com       |              2 |            1586.04
 Dannie Jzhakov      | dannie.jzhakov@gmail.com      |              2 |            1581.44
 Kiah Piggins        | kiah.piggins@gmail.com        |              2 |            1525.45
 Grove Murfin        | grove.murfin@gmail.com        |              2 |            1524.35
 Alasteir Burford    | alasteir.burford@gmail.com    |              2 |            1496.70
 Rubin Pierson       | rubin.pierson@gmail.com       |              2 |            1476.64
 Collie Bethell      | collie.bethell@gmail.com      |              2 |            1471.18
 Modestia Carah      | modestia.carah@gmail.com      |              2 |            1464.32
 Obediah Mower       | obediah.mower@gmail.com       |              2 |            1459.09
 Erminia Kopelman    | erminia.kopelman@gmail.com    |              2 |            1442.21
 Cobbie Lobbe        | cobbie.lobbe@gmail.com        |              2 |            1438.91
 Jennee Luesley      | jennee.luesley@gmail.com      |              2 |            1423.06
 Sophia Mathivet     | sophia.mathivet@gmail.com     |              2 |            1359.38
 Aluin Cornner       | aluin.cornner@gmail.com       |              2 |            1346.98
 Wilie Cowerd        | wilie.cowerd@gmail.com        |              2 |            1339.33
(30 rows)
```

Query 1 Output

```sql
--QUERY 2
--Description: Retrieves the 10 most common boat types, their total count and their cost statistics
SELECT
    bt.TYPE_NAME AS "10 Most Common Boat Types",
    COUNT(*) AS "Number of boat type occurences across all yards",
    CONCAT('£', SUM(bs.COST)) AS "Total Service cost",
    CONCAT('£', ROUND(AVG(bs.COST), 2)) AS "Average Service cost",
    CONCAT('£', MAX(bs.COST)) AS "Max Service Cost",
    CONCAT('£', MIN(bs.COST)) AS "Min Service Cost"
FROM
    BOAT b
JOIN
    BOAT_TYPE bt ON b.TYPE_ID = bt.TYPE_ID
JOIN
    BOOKING bk ON b.BOAT_ID = bk.BOAT_ID
JOIN
    BOOKING_SERVICE bs ON bk.BOOKING_ID = bs.BOOKING_ID
GROUP BY
    bt.TYPE_NAME
ORDER BY
    COUNT(*) DESC
LIMIT 10;
```

10 Most Common Boat Types	Number of boat type occurences across all yards	Total Service cost	Average Service cost	Max Service Cost	Min Service Cost
Pontoon	28	£13873.78	£495.49	£994.03	£116.24
Offshore Support	21	£12711.74	£605.32	£983.08	£220.50
Hovercraft	20	£12487.46	£624.37	£986.17	£169.20
Ro-Ro Ship	20	£12858.93	£642.95	£979.93	£352.30
Powerboat	19	£10947.64	£576.19	£996.64	£158.22
Yacht	19	£11223.30	£590.70	£984.73	£254.50
Cruise Ship	18	£9555.04	£530.84	£958.73	£165.19
Fishing Boat	18	£8683.40	£482.41	£921.46	£157.64
Jet Ski	17	£12247.05	£720.41	£992.83	£171.14
Feeder Vessel	16	£9880.68	£617.54	£985.71	£142.64

(10 rows)

Query 2 Output

```sql
--QUERY 3
--Description: Retrieves the all top revenue making countries with at least £13,000 per year
SELECT
    countries."Country" AS "Countries with the Highest Revenue",
    countries."Total_Revenue" AS "Total Revenue"
FROM
    (
        -- Subquery: Calculate total revenue for each country
        SELECT
            c.COUNTRY_NAME AS "Country",
            CONCAT('£', SUM(bs.COST)) AS "Total_Revenue"
        FROM
            YARD y
            JOIN STAFF s ON y.YARD_ID = s.YARD_ID
            JOIN COUNTRY c ON y.COUNTRY_ID = c.COUNTRY_ID
            JOIN BOOKING_SERVICE bs ON s.STAFF_ID = bs.STAFF_ID
        GROUP BY
            c.COUNTRY_NAME
        HAVING
            SUM(bs.COST) > 13000
    ) AS countries
ORDER BY
    countries."Total_Revenue" DESC;
```

Countries with the Highest Revenue	Total Revenue
Barbados	£17017.26
Algeria	£16946.33
Angola	£16478.36
Bahrain	£15980.73
Andorra	£15902.98
Antigua and Barbuda	£15902.53
Australia	£15419.87
Armenia	£15401.32
Argentina	£15167.95
Afghanistan	£14903.02
Austria	£14663.15
Azerbaijan	£13890.61
Bangladesh	£13884.56
Albania	£13702.97
Bahamas	£13190.36

(15 rows)

Query 3 Output

```sql
--QUERY 4
--Description: Retrieves the average age of boats that have Anode Replacement service and their classifications
SELECT
    bt.TYPE_NAME AS "Boat Type",
    bt.MODEL AS "Boat Model",
    bt.MANUFACTURER AS "Manufacturer",
    ROUND(AVG(b.YEAR_BUILT)) AS "Average Boat age",
    COUNT(DISTINCT b.BOAT_ID) AS "Service Count"
FROM
    BOAT b
JOIN
    BOAT_TYPE bt ON b.TYPE_ID = bt.TYPE_ID
JOIN
    BOOKING bk ON b.BOAT_ID = bk.BOAT_ID
JOIN
    BOOKING_SERVICE bs ON bs.BOOKING_ID = bk.BOOKING_ID
JOIN
    SERVICE S ON bs.SERVICE_ID = s.SERVICE_ID
WHERE
    s.SERVICE_NAME = 'Anode Replacement'
GROUP BY
    bt.TYPE_NAME, bt.MODEL, bt.MANUFACTURER;
```

Boat Type	Boat Model	Manufacturer	Average Boat age	Service Count
Dredger	Q	Bavaria	2007	1
FPSO	Eldorado	Sweetwater	1993	1
Feeder Vessel	A3	Bayliner	1987	1
Heavy-Lift Ship	Durango	Azimut	2012	1
Jet Ski	LS	Sweetwater	2011	1
Powerboat	RSX	Bennington	1997	2
Ro-Ro Ship	Camaro	Sunseeker International	2000	1
Salvage Tug	Miata MX-5	Zodiac	2000	1
Skiff	Landaulet	Bavaria	2001	1
Yacht	6000	Zodiac	2003	2

(10 rows)

Query 4 Output

```sql
--QUERY 5
--Description: Retrieves number of times an electronic system upgrade was done in each country and their average cost
SELECT
    co.COUNTRY_NAME AS "Country",
    COUNT(bs.SERVICE_ID) AS "Boat Electronic System Upgrades",
    CONCAT('£', ROUND(AVG(bs.COST), 2)) AS "Average Cost in each Country"
FROM
    BOOKING_SERVICE bs
JOIN
    SERVICE S ON bs.SERVICE_ID = s.SERVICE_ID
JOIN
    BOOKING bk ON bs.BOOKING_ID = bk.BOOKING_ID
JOIN
    BOAT b ON b.BOAT_ID = bk.BOAT_ID
JOIN
    CUSTOMER c ON b.CUST_ID = c.CUST_ID
JOIN
    COUNTRY co ON c.COUNTRY_ID = co.COUNTRY_ID
WHERE
    s.SERVICE_NAME = 'Electronic System Upgrade'
GROUP BY
    co.COUNTRY_NAME
ORDER BY
    "Boat Electronic System Upgrades" DESC;
```

Country	Boat Electronic System Upgrades	Average Cost in each Country
Andorra	2	£484.44
Bahamas	2	£339.89
Barbados	2	£839.48
Australia	1	£686.95
Afghanistan	1	£302.93
Bahrain	1	£516.76
Bangladesh	1	£655.39
Austria	1	£698.13
Antigua and Barbuda	1	£635.07
Armenia	1	£699.97

(10 rows)

Query 5 Output

Database Security

In developing a secure database for an international company like Solent Marine Solutions, we have made careful considerations to address security issues that revolve around privileges, roles, and the controlled access to the database objects. Below are the key security considerations implemented:

1. **Role-Based Access Control:**
   Roles have been created to match with the role table in the database. This simplifies the management of access permissions.

```
dbprin_cw=# \du
                                   List of roles
    Role name     |                         Attributes                         | Member of
------------------+------------------------------------------------------------+-----------
 Cleaner          |                                                            | {}
 Engineer         |                                                            | {}
 Manager          |                                                            | {}
 Repair Specialist |                                                           | {}
 Safety Inspector |                                                            | {}
 Technician       |                                                            | {}
```

The roles have also been associated with a login and password for secure authentication.

```
dbprin_cw=# SELECT rolname, rolsuper, rolinherit, rolcreaterole, rolcreatedb, rolcanlogin FROM pg_roles;
         rolname          | rolsuper | rolinherit | rolcreaterole | rolcreatedb | rolcanlogin
--------------------------+----------+------------+---------------+-------------+-------------
 Engineer                 | f        | t          | f             | f           | t
 Manager                  | f        | t          | f             | f           | t
 Repair Specialist        | f        | t          | f             | f           | t
 Safety Inspector         | f        | t          | f             | f           | t
 Technician               | f        | t          | f             | f           | t
 Cleaner                  | f        | t          | f             | f           | t
```

2. **Limited Privileges:**
   Users are granted only the essential permissions required to perform their designated tasks, minimising the risk of unauthorised actions. This was implemented as follows;

MANAGER PRIVILEGES

The manager role was granted access to all tables and privileges.

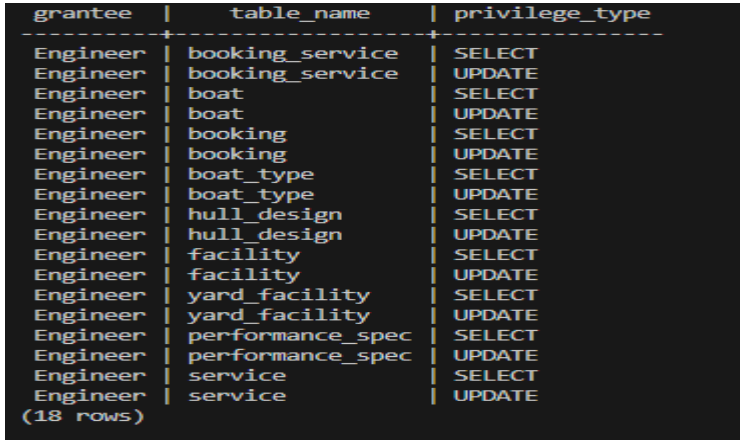
grantee	table_name	privilege_type	gr
Manager	country	INSERT	
Manager	country	SELECT	
Manager	country	UPDATE	
Manager	country	DELETE	
Manager	country	TRUNCATE	
Manager	country	REFERENCES	
Manager	country	TRIGGER	
Manager	insurance	INSERT	
Manager	insurance	SELECT	
Manager	insurance	UPDATE	
Manager	insurance	DELETE	
Manager	insurance	TRUNCATE	
Manager	insurance	REFERENCES	
Manager	insurance	TRIGGER	
Manager	booking_service	INSERT	
Manager	booking_service	SELECT	
Manager	booking_service	UPDATE	
Manager	booking_service	DELETE	
Manager	booking_service	TRUNCATE	
Manager	booking_service	REFERENCES	
Manager	booking_service	TRIGGER	
Manager	boat	INSERT	
Manager	boat	SELECT	
Manager	boat	UPDATE	
Manager	boat	DELETE	
Manager	boat	TRUNCATE	
Manager	boat	REFERENCES	
Manager	boat	TRIGGER	
Manager	booking	INSERT	
Manager	booking	SELECT	
Manager	booking	UPDATE	
Manager	booking	DELETE	
Manager	booking	TRUNCATE	
Manager	booking	REFERENCES	
Manager	booking	TRIGGER	
Manager	customer	INSERT	
Manager	customer	SELECT	
Manager	customer	UPDATE	
Manager	customer	DELETE	
Manager	customer	TRUNCATE	
Manager	customer	REFERENCES	
Manager	customer	TRIGGER	
Manager	yard	INSERT	
Manager	yard	SELECT	
Manager	yard	UPDATE	
Manager	yard	DELETE	
Manager	yard	TRUNCATE	
Manager	yard	REFERENCES	
Manager	yard	TRIGGER	
Manager	boat_type	INSERT	
Manager	boat_type	SELECT	
Manager	boat_type	UPDATE	
Manager	boat_type	DELETE	
Manager	boat_type	TRUNCATE	
Manager	boat_type	REFERENCES	
Manager	boat_type	TRIGGER	
Manager	top_customers_spending	SELECT	
Manager	common_boat_types	SELECT	
Manager	hull_design	INSERT	
Manager	hull_design	SELECT	
Manager	hull_design	UPDATE	
Manager	hull_design	DELETE	
Manager	hull_design	TRUNCATE	
Manager	hull_design	REFERENCES	
Manager	hull_design	TRIGGER	
Manager	staff	INSERT	
Manager	staff	SELECT	
Manager	staff	UPDATE	
Manager	staff	DELETE	
Manager	staff	TRUNCATE	
Manager	staff	REFERENCES	
Manager	staff	TRIGGER	
Manager	staff_role	INSERT	
Manager	staff_role	SELECT	

The Repair Specialist role was granted access to the boat, hull design, boat, boat type, performance specifications, service, booking_service, booking, facility, and yard_facility tables. They were also granted the SELECT, INSERT, UPDATE, and DELETE privileges for those tables.

```
      grantee       |    table_name     | privilege_type
--------------------+-------------------+----------------
 Repair Specialist | booking_service   | INSERT
 Repair Specialist | booking_service   | SELECT
 Repair Specialist | booking_service   | UPDATE
 Repair Specialist | booking_service   | DELETE
 Repair Specialist | boat              | INSERT
 Repair Specialist | boat              | SELECT
 Repair Specialist | boat              | UPDATE
 Repair Specialist | boat              | DELETE
 Repair Specialist | booking           | INSERT
 Repair Specialist | booking           | SELECT
 Repair Specialist | booking           | UPDATE
 Repair Specialist | booking           | DELETE
 Repair Specialist | boat_type         | INSERT
 Repair Specialist | boat_type         | SELECT
 Repair Specialist | boat_type         | UPDATE
 Repair Specialist | boat_type         | DELETE
 Repair Specialist | hull_design       | INSERT
 Repair Specialist | hull_design       | SELECT
 Repair Specialist | hull_design       | UPDATE
 Repair Specialist | hull_design       | DELETE
 Repair Specialist | facility          | INSERT
 Repair Specialist | facility          | SELECT
 Repair Specialist | facility          | UPDATE
 Repair Specialist | facility          | DELETE
 Repair Specialist | yard_facility     | INSERT
 Repair Specialist | yard_facility     | SELECT
 Repair Specialist | yard_facility     | UPDATE
 Repair Specialist | yard_facility     | DELETE
 Repair Specialist | performance_spec  | INSERT
 Repair Specialist | performance_spec  | SELECT
 Repair Specialist | performance_spec  | UPDATE
 Repair Specialist | performance_spec  | DELETE
 Repair Specialist | service           | INSERT
 Repair Specialist | service           | SELECT
 Repair Specialist | service           | UPDATE
 Repair Specialist | service           | DELETE
(36 rows)
```

REPAIR SPECIALIST PRIVILEGES

The Engineer role was granted access to the boat, hull design, boat, boat type, performance specifications, service, booking_service, booking, facility, and yard_facility tables. They were also granted the SELECT and UPDATE privileges for those tables.



grantee	table_name	privilege_type
Engineer	booking_service	SELECT
Engineer	booking_service	UPDATE
Engineer	boat	SELECT
Engineer	boat	UPDATE
Engineer	booking	SELECT
Engineer	booking	UPDATE
Engineer	boat_type	SELECT
Engineer	boat_type	UPDATE
Engineer	hull_design	SELECT
Engineer	hull_design	UPDATE
Engineer	facility	SELECT
Engineer	facility	UPDATE
Engineer	yard_facility	SELECT
Engineer	yard_facility	UPDATE
Engineer	performance_spec	SELECT
Engineer	performance_spec	UPDATE
Engineer	service	SELECT
Engineer	service	UPDATE

(18 rows)

ENGINEER PRIVILEGES

The Technician role was granted access to the boat, hull design, boat, boat type, performance specifications, service, booking_service, booking, facility, and yard_facility tables. They were also granted the SELECT and UPDATE privileges for those tables.

```
dbprin_cw-# WHERE grantee = 'Technician';
  grantee   |    table_name     | privilege_type
------------+-------------------+----------------
 Technician | booking_service   | SELECT
 Technician | booking_service   | UPDATE
 Technician | boat              | SELECT
 Technician | boat              | UPDATE
 Technician | booking           | SELECT
 Technician | booking           | UPDATE
 Technician | boat_type         | SELECT
 Technician | boat_type         | UPDATE
 Technician | hull_design       | SELECT
 Technician | hull_design       | UPDATE
 Technician | facility          | SELECT
 Technician | facility          | UPDATE
 Technician | yard_facility     | SELECT
 Technician | yard_facility     | UPDATE
 Technician | performance_spec  | SELECT
 Technician | performance_spec  | UPDATE
 Technician | service           | SELECT
 Technician | service           | UPDATE
(18 rows)
```

TECHNICIAN PRIVILEGES

The Safety Inspector role was granted access to the boat, hull design, boat, boat type, performance specifications, service, booking_service, booking, facility, and yard_facility tables. They were granted the SELECT privileges for those tables.

```
dbprin_cw-# WHERE grantee = 'Safety Inspector';
     grantee      |    table_name     | privilege_type
------------------+-------------------+----------------
 Safety Inspector | booking_service   | SELECT
 Safety Inspector | boat              | SELECT
 Safety Inspector | booking           | SELECT
 Safety Inspector | boat_type         | SELECT
 Safety Inspector | hull_design       | SELECT
 Safety Inspector | facility          | SELECT
 Safety Inspector | yard_facility     | SELECT
 Safety Inspector | performance_spec  | SELECT
 Safety Inspector | service           | SELECT
(9 rows)
```

SAFETY INSPECTOR PRIVILEGES

For the cleaner role, SELECT and UPDATE privileges on the BOAT table to retrieve and update cleaning-related information.
SELECT privilege on the FACILITY table to view information about facilities.
SELECT, INSERT, UPDATE, DELETE privileges on the YARD_FACILITY table to manage cleaning activities related to yard facilities.

```
dbprin_cw=# SELECT grantee, table_name, privilege_type
dbprin_cw-# FROM information_schema.table_privileges
dbprin_cw-# WHERE grantee = 'Cleaner';
 grantee |  table_name   | privilege_type
---------+---------------+----------------
 Cleaner | boat          | SELECT
 Cleaner | boat          | UPDATE
 Cleaner | facility      | SELECT
 Cleaner | yard_facility | INSERT
 Cleaner | yard_facility | SELECT
 Cleaner | yard_facility | UPDATE
 Cleaner | yard_facility | DELETE
(7 rows)
```

CLEANER PRIVILEGES

3. Trigger Privileges:
   Triggers that enforce business rules and data integrity have been created.
   The check_customer_values function is crucial for ensuring the proper insertion and updating of customer data. Staff members have the required privileges to execute this function.

4. Security by Design:
   Security measures, like using the appropriate data types, have been integrated into the database design process. This also ensures that the database is less vulnerable to certain types of attacks.

5. Securing Views:
   Views have been created from the performance-sensitive queries of the database.
   The manager role has been granted SELECT privileges on the views.

```
dbprin_cw=# CREATE VIEW top_customers_spending AS
dbprin_cw-# SELECT
dbprin_cw-#     CASE
dbprin_cw-#         WHEN C.CUST_TYPE = 'Company' THEN c.COMPANY_NAME
dbprin_cw-#         WHEN C.CUST_TYPE = 'Individual' THEN CONCAT(c.CUST_FNAME, ' ', c.CUST_LNAME)
dbprin_cw-#     END AS "Customer",
dbprin_cw-#     c.EMAIL AS "Customer Email",
dbprin_cw-#     COUNT(DISTINCT bk.BOOKING_ID) AS "Total Bookings",
dbprin_cw-#     SUM(bs.COST) AS "Total Spending (£)"
dbprin_cw-# FROM
dbprin_cw-#     CUSTOMER c
dbprin_cw-#     JOIN BOAT b ON c.CUST_ID = b.CUST_ID
dbprin_cw-#     JOIN BOOKING bk ON b.BOAT_ID = bk.BOAT_ID
dbprin_cw-#     JOIN BOOKING_SERVICE bs ON bk.BOOKING_ID = bs.BOOKING_ID
dbprin_cw-# GROUP BY
dbprin_cw-#     c.CUST_ID, c.CUST_FNAME, c.CUST_LNAME, c.email
dbprin_cw-# HAVING
dbprin_cw-#     COUNT(DISTINCT bk.BOOKING_ID) > 0
dbprin_cw-# ORDER BY
dbprin_cw-#     "Total Spending (£)" DESC
dbprin_cw-# LIMIT 30;
CREATE VIEW
dbprin_cw=# GRANT SELECT ON top_customers_spending TO "Manager";
GRANT
dbprin_cw-# CREATE VIEW common_boat_types AS
```

```
dbprin_cw=# CREATE VIEW common_boat_types AS
dbprin_cw-# SELECT
dbprin_cw-#     bt.TYPE_NAME AS "10 Most Common Boat Types",
dbprin_cw-#     COUNT(*) AS "Number of boat type occurrences across all yards",
dbprin_cw-#     CONCAT('£', SUM(bs.COST)) AS "Total Service cost",
dbprin_cw-#     CONCAT('£', ROUND(AVG(bs.COST), 2)) AS "Average Service cost",
dbprin_cw-#     CONCAT('£', MAX(bs.COST)) AS "Max Service Cost",
dbprin_cw-#     CONCAT('£', MIN(bs.COST)) AS "Min Service Cost"
dbprin_cw-# FROM
dbprin_cw-#     BOAT b
dbprin_cw-# JOIN
dbprin_cw-#     BOAT_TYPE bt ON b.TYPE_ID = bt.TYPE_ID
dbprin_cw-# JOIN
dbprin_cw-#     BOOKING bk ON b.BOAT_ID = bk.BOAT_ID
dbprin_cw-# JOIN
dbprin_cw-#     BOOKING_SERVICE bs ON bk.BOOKING_ID = bs.BOOKING_ID
dbprin_cw-# GROUP BY
dbprin_cw-#     bt.TYPE_NAME
dbprin_cw-# ORDER BY
dbprin_cw-#     COUNT(*) DESC
dbprin_cw-# LIMIT 10;
CREATE VIEW
dbprin_cw=# GRANT SELECT ON common_boat_types TO "Manager";
GRANT
```

CONCLUSION

In conclusion, the implemented security measures collectively contribute to a robust and well-protected database environment for Solent Marine Solutions. By adopting a holistic approach that spans access control, limited privileges, triggers, security by design, and secure views, we ensure the confidentiality, integrity, and availability of critical information, aligning with best practices in database security.

Database optimisation

To optimise our database, we have identified the three most performance-sensitive transactions and conducted a thorough analysis using a transaction analysis matrix;

QUERY 1 - Retrieve Top Spending Customers

```sql
--QUERY 1
--Description: Retrieves a list of the top 30 customers who have spent the most on services,
--along with the corresponding bookings.
SELECT
    CASE
        WHEN C.CUST_TYPE = 'Company' THEN c.COMPANY_NAME
        WHEN C.CUST_TYPE = 'Individual' THEN CONCAT(c.CUST_FNAME, ' ', c.CUST_LNAME)
    END AS "Customer",
    c.EMAIL AS "Customer Email",
    COUNT(DISTINCT bk.BOOKING_ID) AS "Total Bookings",
    SUM(bs.COST) AS "Total Spending (£)"
FROM
    CUSTOMER c
    JOIN BOAT b ON c.CUST_ID = b.CUST_ID
    JOIN BOOKING bk ON b.BOAT_ID = bk.BOAT_ID
    JOIN BOOKING_SERVICE bs ON bk.BOOKING_ID = bs.BOOKING_ID
GROUP BY
    c.CUST_ID, c.CUST_FNAME, c.CUST_LNAME, c.email
HAVING
    COUNT(DISTINCT bk.BOOKING_ID) > 0
ORDER BY
    "Total Spending (£)" DESC
LIMIT 30;
```

QUERY 2 - Retrieve Top Revenue Countries

```sql
--QUERY 3
--Description: Retrieves the all top revenue making countries with at least £13,000 per year
SELECT
    countries."Country" AS "Countries with the Highest Revenue",
    countries."Total_Revenue" AS "Total Revenue"
FROM
    (
        -- Subquery: Calculate total revenue for each country
        SELECT
            c.COUNTRY_NAME AS "Country",
            CONCAT('£', SUM(bs.COST)) AS "Total_Revenue"
        FROM
            YARD y
            JOIN STAFF s ON y.YARD_ID = s.YARD_ID
            JOIN COUNTRY c ON y.COUNTRY_ID = c.COUNTRY_ID
            JOIN BOOKING_SERVICE bs ON s.STAFF_ID = bs.STAFF_ID
        GROUP BY
            c.COUNTRY_NAME
        HAVING
            SUM(bs.COST) > 13000
    ) AS countries
ORDER BY
    countries."Total_Revenue" DESC;
```

QUERY 3 - Retrieve Electronic System Upgrade Statistics by Country

```sql
--QUERY 5
--Description: Retrieves number of times an electronic system upgrade was done in each country
--and their average cost
SELECT
    co.COUNTRY_NAME AS "Country",
    COUNT(bs.SERVICE_ID) AS "Boat Electronic System Upgrades",
    CONCAT('£', ROUND(AVG(bs.COST), 2)) AS "Average Cost in each Country"
FROM
    BOOKING_SERVICE bs
JOIN
    SERVICE S ON bs.SERVICE_ID = s.SERVICE_ID
JOIN
    BOOKING bk ON bs.BOOKING_ID = bk.BOOKING_ID
JOIN
    BOAT b ON b.BOAT_ID = bk.BOAT_ID
JOIN
    CUSTOMER c ON b.CUST_ID = c.CUST_ID
JOIN
    COUNTRY co ON c.COUNTRY_ID = co.COUNTRY_ID
WHERE
    s.SERVICE_NAME = 'Electronic System Upgrade'
GROUP BY
    co.COUNTRY_NAME
ORDER BY
    "Boat Electronic System Upgrades" DESC;
```

Transaction Analysis Matrix

Transaction	service				country				customer				boat				booking				booking_service				yard				staff			
	C	**R**	**U**	**D**	**C**	**R**	**U**	**D**	**C**	**R**	**U**	**D**	**C**	**R**	**U**	**D**	**C**	**R**	**U**	**D**	**C**	**R**	**U**	**D**	**C**	**R**	**U**	**D**	**C**	**R**	**U**	**D**
Retrieve Top spending customers										**X**				**X**				**X**				**X**										
Retrieve Top Revenue Countries						**X**																**X**				**X**				**X**		
Retrieve Electronic System Upgrade Statistics by Country		**X**				**X**				**X**				**X**				**X**				**X**										

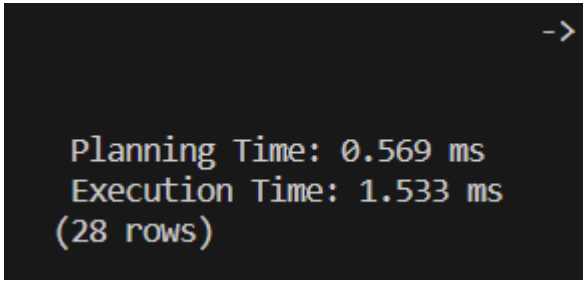
Estimating Performance Requirements

To estimate the performance requirements for these transactions, we have considered the following:

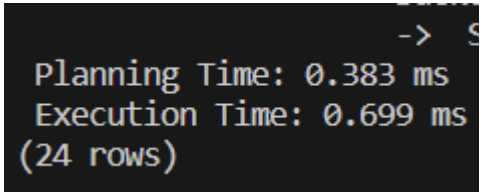
1. Data Volume: We expect a steady growth rate in the tables created.

2. Query Complexity: To estimate the query complexity, execution plan analysis was carried out on the three most performance-sensitive transactions using the ‘EXPLAIN ANALYZE’ commands.

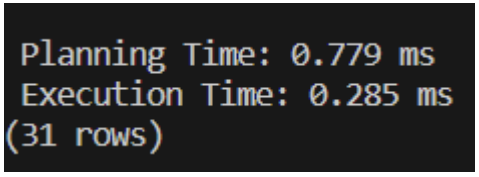
The following images show the results of each analysis;



QUERY 1 - Retrieve Top spending customers



QUERY 2 - Retrieve Top Revenue Countries



QUERY 3 - Retrieve Electronic System Upgrade Statistics by Country

As we expect a steady and consistent growth rate, the dataset would inevitably grow larger. Consequently, this would mean the planning and execution time of the queries would increase correspondingly.

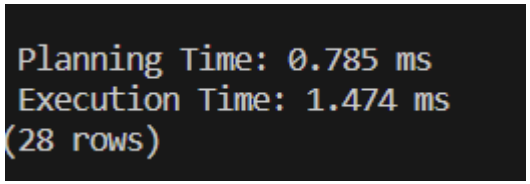
Indexing is the best optimisation tool for a database with a steady growth rate like ours (Mullins, 2022). Indexes can enhance performance by:

- Locating rows by value(s) in column(s)
- Making joins more efficient
- Correlating data across tables
- Aggregating data
- Sorting data to satisfy a query

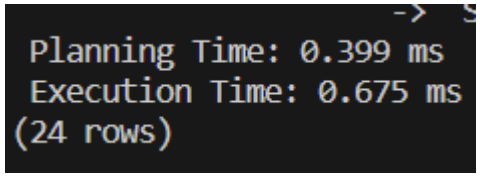
To optimise the performance-sensitive queries, indexes were added to the columns that were involved in the most 'join' operations and 'where' clauses;

```
-- Index on cust_type column in the CUSTOMER table
CREATE INDEX idx_customer_cust_type ON CUSTOMER(cust_type);

-- Index on cost column in the BOOKING_SERVICE table
CREATE INDEX idx_booking_service_cost ON BOOKING_SERVICE(cost);

-- Index on type_name column in the BOAT_TYPE table
CREATE INDEX idx_boat_type_type_name ON BOAT_TYPE(type_name);
```

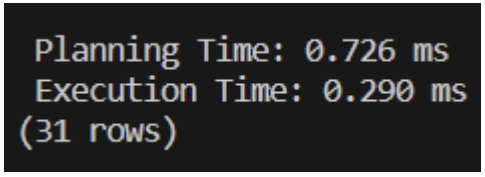
To measure the database optimisation, another execution plan analysis was carried out after the indexing, on the three queries;



QUERY 1 - Retrieve Top spending customers



QUERY 2 - Retrieve Top Revenue Countries



QUERY 3 - Retrieve Electronic System Upgrade Statistics by Country

The results showed that planning and execution time for each query reduced slightly after the indexing.

CONCLUSION

In conclusion, our database optimisation journey has been a proactive and strategic effort aimed at addressing the performance challenges posed by the three most critical transactions: retrieving top spending customers, top revenue countries, and electronic system upgrade statistics by country.

Through a transaction analysis matrix, we identified the specific operations (Create, Read, Update, Delete - CRUD) associated with each transaction across relevant database tables.

To estimate performance requirements, we considered data volume growth and analysed the complexity of the queries using EXPLAIN ANALYZE.

The results indicated a steady growth rate, necessitating a focus on planning and execution times as the dataset expands.

Our optimisation strategy centred on indexing, a powerful tool to enhance query performance. We strategically applied indexes to tables involved in frequent join operations and where clauses. The indexing process aimed to expedite data retrieval, particularly in scenarios where certain columns were frequently accessed or filtered.

Post-indexing execution plan analysis revealed promising outcomes, showcasing reductions in planning and execution times. The introduction of indexes contributed to more efficient query processing, aligning with our goal of delivering optimal performance for these critical transactions.

REFERENCES

Mullins, C. S. (2022, July 7). Techniques for optimizing data. *Database Trends and Applications*.
https://www.dbta.com/Columns/DBA-Corner/Techniques-for-Optimizing-Databases-154027.aspx

Professional, Legal and Ethical issues

Introduction

In this document, we have explored the professional, legal and ethical considerations pertaining to the development of the new Solent Marine Solution database, to facilitate their transition to being a multinational marine corporation.

Professional Issue:

A professional issue we have considered is providing accurate documentation, such as a data dictionary. A data dictionary is significant in facilitating standardisation (Chai, 2022). This means subsequent modifications by other workers to the database can be more easily achieved. Therefore, by us providing an accurate and thorough data dictionary we have upheld a high level of professionalism.

Another professional issue we have considered while developing this database, is that we are to build the system to our best capability. This is outlined by the BSC code of conduct (BSC, 2022) that individuals should perform responsibilities or tasks to their utmost capabilities. We have addressed this aspect, actively revising our code to ensure there were no negligent acts or intentional actions that would not be in the favour of the client, therefore upholding a high standard of professionalism.

Ethical Issues:

We have ensured that the customer's data is adequately protected by employing the usage of privileges. The data protection act outlines that organisations are required to implement appropriate technical and non-technical measures to protect personal data from unauthorised access (UK Government, 2018). We considered this and utilised privileges to control the level of access each role has on the database elements. For example, only a Manager would have access to all the tables, a Cleaner has access to only the yard, boat and facilities table etc. By addressing this 'security of processing requirement' of the data protection act, we have maintained a good standard of ethics.

With the company's expansion, there's a requirement for the database to store and secure a larger amount of personal data for staff, customers, their boats and service history. The Minimisation principle, outlined in the data protection act, states that we cannot collect more data than is strictly necessary (UK Government, 2018). To address this, our database design has been structured to include only data that is relevant or in line with the business rules such as customer contact for communication, address information for sending billing invoice etc. Following the minimisation principle has helped us to uphold a high standard of data protection, adhering to ethical considerations.

Legal Issues

A legal issue we have considered is refraining from collecting sensitive data. According to the data protection act, it is prohibited to process and collect personal data of special categories without a valid legal basis to do so (UK Government, 2018). We've considered this and have not designed the database to collect data of race, ethnic background and other sensitive data, ensuring that we complied with legal requirements.

References

Chai, W. (2022, December). *What is a Data Dictionary and Why Use One?* SearchAppArchitecture.

https://www.techtarget.com/searchapparchitecture/definition/data-dictionary

BCS. (2015). *BCS, The Chartered Institute for IT Code of Conduct*. BCS.org

https://www.bcs.org/media/2211/bcs-code-of-conduct.pdf

UK Government. (2018). *Data Protection Act 2018*. Legislation.gov.uk. https://www.legislation.gov.uk/ukpga/2018/12/contents/enacted