Sname	SID	Rating	Age
Marx	23	8	52
Martin	25	9	51
Adams	27	8	36
Carrey	33	10	22

Bname	BID	Fee	Location
Wayfarer	109	120	Hout Bay
SeaPride	108	500	Fish Hoek
Yuppie	101	400	Hout Bay
Joy	104	200	Hout Bay

SID	BID	Day	Deposit
23	109	01/08/2014	120
23	108	08/08/2014	120
25	101	08/08/2014	0
27	101	09/08/2014	100
27	109	15/08/2014	120
33	109	04/09/2014	0
33	104	11/09/2014	0

Part of the Sailing Club Database:

Some SAILORS tuples

Some BOATS tuples

Some RESERVES tuples

This example database is for a sailing club where sailor can hire a boat for the day. SID is the key of the Sailors table, BID is the key of the Boats table, and so Reserves use SID and BID to say who has reserved which boat for which day. Fee is the amount one pays per hour that one takes the boat out for, and Deposit is the amount one pays to reserve a particular boat ahead of time.
There is also a CAPTAINS table with the same schema as the SAILORS table that is not shown here.

Consider a Sailing Club Database with Schema:

Boats-schema = (BID, Bname, fee, location)
Sailors-schema = (SID, Sname, rating, age)
Reserves-schema = (SID, BID, day, deposit)
Captains-schema = (SID, Sname, rating, age)

1. Get everything in the Sailors table.
2. Get sailor ID, rank & age of all sailors, ordered from highest to lowest rank. Rank is 10 times rating.
3. Get alphabetical list of sailors with rating less than 10.
4. Find how much deposit money there is in total and how many tuples are in the reserves table.
5. Get all info on boats in Fishhoek but I'm not sure how you spell Fishoek.
6. In what locations are boats kept?
7. Get the names of all Boats that have a fee value recorded in the database.
8. Get ID of all boats that have not been reserved.
9. Get all reservation info, including all details on the boats being reserved.
10. For all reservations, get the name of the sailor, along with the day and name of the boat booked.
11. Get the average deposit paid for each boat.
12. Get the average deposit paid for each boat that has been booked by more than one person.
13. Get the average firm deposit paid for each boat that has been booked by more than one person, in increasing order of amount. A firm deposit is one which exceeds R10
14. Get name & rating of sailors with rating exceeding 7 who made any reservation with 0 deposit.
15. Get names of boats located in a place other than Hout Bay or Fish Hoek.
16. Get names of boats having a fee larger than any boat located in Hout Bay.
17. Get names that are in both the sailors and the captains relations.
18. Get names of boats that have exactly 1 reservation.
19. Get sailor ID and total deposit paid for sailors who have booked more than 1 boat.
20. Get all reservation info including details of the boat booked.
21. Get all information on every boat. If a boat has reservations, including all its reservations info.
22. Create a new tuple for the boat named "Nino" which has fee R150, BID 110, and is in Fish Hoek.
23. Remove all bookings from Reserves where there is no deposit.
24. Increase the fee of every boat by 12%.
25. Make a view called Bookings which hides the Deposit value i.e. only has the other 3 attributes.
26. Create a table called Reserves with 3 integer attributes BID, SID & deposit, and a date attribute Day. Allow only deposit to be omitted, and ensure SID and BID values exist in the database. When someone books a boat it is for the whole day.
27. Add a new attribute NEEDSREPAIR to the Boats table. It is usually "N".
28. We should not be ageist. Remove the Age attribute.
29. Remove the Captains relation altogether so that nobody can try insert or use Captains in future.

1. **SELECT** * **FROM** SAILORS
2. SELECT SID, RATING * 10, AGE FROM SAILORS **ORDER BY** RATING DESC
3. SELECT SNAME FROM SAILORS **WHERE** RATING <= 9 ORDER BY SNAME
   - AND, OR, NOT <, <=, >, >=, =, < >
   - ALSO: WHERE *FEE* BETWEEN 200 AND 500 FOR DESCENDING ORDER: ORDER BY … DESC
4. SELECT **SUM** (DEPOSIT) **AS** TOTAL, **COUNT (**DEPOSIT) AS HOWMANY FROM RESERVES
   - There are 5 aggregate functions. They compute a single value from a set of values.
   - *avg* & *sum* can only be applied to numbers; *count, min* & *max* can be applied to any attribute
5. SELECT * FROM BOATS WHERE LOCATION **LIKE** "_IS%K"
6. SELECT **DISTINCT** LOCATION FROM BOATS
7. SELECT BNAME FROM BOATS WHERE FEE **IS NOT NULL**
8. SELECT BID FROM BOATS) **EXCEPT** (SELECT BID FROM RESERVES)
   - The two relations to which UNION/EXCEPT/INTERSECT are applied must be union compatible – that means they must have the same relation scheme
   - union/intersect/except all retains duplicates
   - EXCEPT is sometimes called MINUS
9. **SELECT * FROM RESERVES, BOATS WHERE RESERVES.BID = BOATS.BID**
10. SELECT SNAME, DAY, BNAME FROM SAILORS AS S, RESERVES AS R, BOATS AS B WHERE S.SID = R.SID AND R.BID = B.BID
11. SELECT BID, AVG(DEPOSIT) FROM RESERVES **GROUP BY** BID
12. SELECT BID, AVG(DEPOSIT) FROM RESERVES GROUP BY BID **HAVING** COUNT (DISTINCT SID) > 1
13. SELECT BID, AVG(DEPOSIT) AS AVERAGEDEPOSIT FROM RESERVES WHERE DEPOSIT > 10 GROUP BY BID HAVING COUNT (DISTINCT SID) > 1 ORDER BY AVERAGEDEPOSIT
    (1) "where" is checked and tuples discarded
    (2) remaining tuples ordered on "group by"
    (3) aggregate values computed for each group
    (4) "having" checked and groups discarded
    (5) results sorted as specified in "order by"
14. SELECT SNAME, RATING FROM SAILORS WHERE RATING > 7 AND SID **IN**
    (SELECT SID FROM RESERVES WHERE DEPOSIT = 0)
15. SELECT BNAME FROM BOATS WHERE LOCATION NOT IN ("HOUT BAY", "FISH HOEK")
16. SELECT DISTINCT BNAME FROM BOATS WHERE FEE **> SOME**
    (SELECT FEE FROM BOATS WHERE LOCATION = "HOUT BAY")
    - Also: = some, <=some, >= some, <> some, > all ("greater than all"), >=all, <> all, etc.

17. SELECT SNAME FROM SAILORS WHERE **EXISTS**
    (SELECT * FROM CAPTAINS WHERE CAPTAINS.SID = SAILORS.SID)
18. SELECT BNAME FROM BOATS AS B WHERE **UNIQUE**
    (SELECT BID FROM RESERVES WHERE RESERVES.BID = B.BID)
19. SELECT SID, TOTAL_DEPOSIT **FROM (SELECT** SID, COUNT(BID), SUM(DEPOSIT) FROM RESERVES WHERE DEPOSIT IS NOT NULL AND DEPOSIT > 0 GROUP BY SID) AS RESULT(SID, NUM_BOATS, TOTAL_DEPOSIT) WHERE NUM_BOATS > 1
20. SELECT * FROM BOAT **INNER JOIN** RESERVES ON BOAT.BID = RESERVES.BID
21. SELECT * FROM BOAT **LEFT OUTER JOIN** RESERVES ON BOAT.BID = RESERVES.BID
    - Join Types define how tuples which do not match are handled
        - inner join
        - left outer join
        - right outer join
        - full outer join
    - Join Conditions define which tuples in the two relations match:
        - natural (uses all attributes that are common to both)
        - on <predicate> (used when attributes have different names)
        - using (A1, A2, … , An) (similar to natural join condition, except A1 … An are the join attributes, states which (rather than all) attributes common to both to "join" on)
22. **INSERT** INTO BOATS VALUES ("Nino", 110, 150, "Fish Hoek")
23. **DELETE** FROM RESERVES WHERE DEPOSIT IS NULL OR DEPOSIT = 0
24. **UPDATE** BOATS SET FEE = FEE * 1.12
25. **CREATE VIEW** BOOKINGS AS SELECT SID, BID, DAY FROM RESERVES
26. **CREATE TABLE** RESERVES
    (BID INTEGER NOT NULL,
    SID INTEGER NOT NULL,
    DAY DATETIME NOT NULL,
    DEPOSIT INTEGER,
    PRIMARY KEY (BID, SID),
    CHECK (BID IN (SELECT BID FROM BOATS)), CHECK (SID IN (SELECT SID FROM SAILORS)))
27. **ALTER TABLE** BOATS **ADD** NEEDSREPAIR CHAR(1) DEFAULT "N"
28. ALTER TABLE SAILORS **DROP** AGE
29. **DROP TABLE** CAPTAINS