This assignment introduces an example concerning World War II capital ships. It involves the following relations:

Classes(class, type, country, numGuns, bore, displacement)
Ships(name, class, launched) --launched is for year launched
Battles(name, date_fought)
Outcomes(ship, battle, result)

Ships are built in "classes" from the same design, and the class is usually named for the first ship of that class. The relation Classes records the name of the class, the type (bb for battleship or bc for battlecruiser), the country that built the ship, the number of main guns, the bore (diameter of the gun barrel, in inches) of the main guns, and the displacement (weight, in tons). Relation Ships records the name of the ship, the name of its class, and the year in which the ship was launched. Relation Battles gives the name and date of battles involving these ships, and relation Outcomes gives the result (sunk, damaged, or ok) for each ship in each battle.

Exercise 1. (2 points)

1. Create simple SQL statements to create the above relations (no constraints for these initial creations).
2. Insert the following data.

```
CREATE TABLE Classes (
  class VARCHAR(15),
  type CHAR(2),
  country VARCHAR(13),
  numGuns INT,
  bore INT,
  displacement INT
);

CREATE TABLE Ships (
  name VARCHAR(15),
  class VARCHAR(15),
  launched INT          --year launched
);

CREATE TABLE Battles (
  name VARCHAR(15),
  date_fought DATE
);

CREATE TABLE Outcomes (
  ship VARCHAR(15),
   battle VARCHAR(15),
  result VARCHAR(10)
);

INSERT INTO Classes(class, type, country, numGuns, bore, displacement) VALUES
('Bismarck','bb','Germany',8,15,42000);
INSERT INTO Classes(class, type, country, numGuns, bore, displacement) VALUES
('Kongo','bc','Japan',8,14,32000);
INSERT INTO Classes(class, type, country, numGuns, bore, displacement) VALUES
('North Carolina','bb','USA',9,16,37000);
INSERT INTO Classes(class, type, country, numGuns, bore, displacement) VALUES
('Renown','bc','Gt. Britain',6,15,32000);
INSERT INTO Classes(class, type, country, numGuns, bore, displacement) VALUES
('Revenge','bb','Gt. Britain',8,15,29000);
INSERT INTO Classes(class, type, country, numGuns, bore, displacement) VALUES
('Tennessee','bb','USA',12,14,32000);
INSERT INTO Classes(class, type, country, numGuns, bore, displacement) VALUES
('Yamato','bb','Japan',9,18,65000);

INSERT INTO Ships(name, class, launched) VALUES
('California','Tennessee',1921);
INSERT INTO Ships(name, class, launched) VALUES
('Haruna','Kongo',1915);
INSERT INTO Ships(name, class, launched) VALUES
('Hiei','Kongo',1914);
INSERT INTO Ships(name, class, launched) VALUES
('Iowa','Iowa',1943);
INSERT INTO Ships(name, class, launched) VALUES
('Kirishima','Kongo',1914);
INSERT INTO Ships(name, class, launched) VALUES
('Kongo','Kongo',1913);
INSERT INTO Ships(name, class, launched) VALUES
('Missouri','Iowa',1944);
INSERT INTO Ships(name, class, launched) VALUES
('Musashi','Yamato',1942);
INSERT INTO Ships(name, class, launched) VALUES
('New Jersey','Iowa',1943);
INSERT INTO Ships(name, class, launched) VALUES
('North Carolina','North Carolina',1941);
INSERT INTO Ships(name, class, launched) VALUES
('Ramilles','Revenge',1917);
INSERT INTO Ships(name, class, launched) VALUES
('Renown','Renown',1916);
INSERT INTO Ships(name, class, launched) VALUES
('Repulse','Renown',1916);
INSERT INTO Ships(name, class, launched) VALUES
```

```
('Resolution','Revenge',1916);
INSERT INTO Ships(name, class, launched) VALUES
('Revenge','Revenge',1916);
INSERT INTO Ships(name, class, launched) VALUES
('Royal Oak','Revenge',1916);
INSERT INTO Ships(name, class, launched) VALUES
('Royal Sovereign','Revenge',1916);
INSERT INTO Ships(name, class, launched) VALUES
('Tennessee','Tennessee',1920);
INSERT INTO Ships(name, class, launched) VALUES
('Washington','North Carolina',1941);
INSERT INTO Ships(name, class, launched) VALUES
('Wisconsin','Iowa',1944);
INSERT INTO Ships(name, class, launched) VALUES
('Yamato','Yamato',1941);

INSERT INTO Battles(name, date_fought) VALUES
('North Atlantic',TO_DATE('27-May-1941','dd-mon-yyyy'));
INSERT INTO Battles(name, date_fought) VALUES
('Guadalcanal',TO_DATE('15-Nov-1942','dd-mon-yyyy'));
INSERT INTO Battles(name, date_fought) VALUES
('North Cape',TO_DATE('26-Dec-1943','dd-mon-yyyy'));
INSERT INTO Battles(name, date_fought) VALUES
('Surigao Strait',TO_DATE('25-Oct-1944','dd-mon-yyyy'));

INSERT INTO Outcomes(ship, battle, result) VALUES
('Bismarck','North Atlantic', 'sunk');
INSERT INTO Outcomes(ship, battle, result) VALUES
('California','Surigao Strait', 'ok');
INSERT INTO Outcomes(ship, battle, result) VALUES
('Duke of York','North Cape', 'ok');
INSERT INTO Outcomes(ship, battle, result) VALUES
('Fuso','Surigao Strait', 'sunk');
INSERT INTO Outcomes(ship, battle, result) VALUES
('Hood','North Atlantic', 'sunk');
INSERT INTO Outcomes(ship, battle, result) VALUES
('King George V','North Atlantic', 'ok');
INSERT INTO Outcomes(ship, battle, result) VALUES
('Kirishima','Guadalcanal', 'sunk');
INSERT INTO Outcomes(ship, battle, result) VALUES
('Prince of Wales','North Atlantic', 'damaged');
INSERT INTO Outcomes(ship, battle, result) VALUES
('Rodney','North Atlantic', 'ok');
INSERT INTO Outcomes(ship, battle, result) VALUES
('Scharnhorst','North Cape', 'sunk');
INSERT INTO Outcomes(ship, battle, result) VALUES
```

```
('South Dakota','Guadalcanal', 'ok');
INSERT INTO Outcomes(ship, battle, result) VALUES
('West Virginia','Surigao Strait', 'ok');
INSERT INTO Outcomes(ship, battle, result) VALUES
('Yamashiro','Surigao Strait', 'sunk');
```

Exercise 2. (16 points)

Write SQL queries for the following requirements. (The questions of this exercise are of two points each.)

1. The treaty of Washington in 1921 prohibited capital ships heavier than 35,000 tons. List the ships that violated the treaty of Washington.

```
SELECT name
FROM Classes NATURAL JOIN Ships
WHERE displacement>35000 AND launched>=1921;
```

2. List the name, displacement, and number of guns of the ships engaged in the battle of Guadalcanal.

```
SELECT Outcomes.ship, Classes.displacement, Classes.numGuns
FROM Outcomes  LEFT OUTER JOIN Ships ON Outcomes.ship=Ships.name
               LEFT OUTER JOIN Classes ON Ships.class=Classes.class
WHERE Outcomes.battle='Guadalcanal';
```

3. List all the capital ships mentioned in the database. (Remember that not all ships appear in the Ships relation.)

```
(SELECT name AS ship FROM Ships) UNION
(SELECT ship FROM Outcomes);
```

4. Find those countries that had both battleships and battlecruisers.

```
SELECT C1.country
FROM Classes C1, Classes C2
WHERE C1.country=C2.country AND C1.type='bb' AND C2.type='bc';
```

5. Find those ships that "lived to fight another day"; they were damaged in one battle, but later fought in another.

```
CREATE VIEW OutcomesWithDate AS
SELECT Outcomes.ship, Outcomes.battle, Outcomes.result, Battles.date_fought
FROM Outcomes JOIN Battles ON Outcomes.battle=Battles.name;
```

```
SELECT X.ship
FROM OutcomesWithDate X JOIN OutcomesWithDate Y
     ON X.ship=Y.ship AND
        X.result='damaged' AND
        X.date_fought<Y.date_fought;

DROP VIEW OutcomesWithDate;
```

Note. Empty result returned as there aren't enough tuples.

--Alternate solution with *subquery factoring*

```
WITH
OutcomesWithDate AS (
SELECT Outcomes.ship, Outcomes.battle, Outcomes.result, Battles.date_fought
FROM Outcomes JOIN Battles ON Outcomes.battle=Battles.name)

SELECT X.ship
FROM OutcomesWithDate X JOIN OutcomesWithDate Y
     ON X.ship=Y.ship AND
        X.result='damaged' AND
        X.date_fought<Y.date_fought;
```

6. Find the countries whose ships had the largest number of guns.

```
SELECT country
FROM Classes NATURAL JOIN Ships
WHERE numGuns = (SELECT max(numGuns) FROM Classes);
```

7. Find the names of the ships whose number of guns was the largest for those ships of the same bore.

```
SELECT name
FROM (SELECT name, numGuns, bore FROM Ships NATURAL JOIN Classes) X
WHERE numGuns = (
    SELECT max(numGuns)
    FROM Ships NATURAL JOIN Classes
    WHERE bore=X.bore
);
```

8. Find for each class with at least three ships the number of ships of that class sunk in battle.

```
CREATE VIEW X AS
    SELECT class
    FROM Ships
    GROUP BY class
    HAVING COUNT(name)>=3;


CREATE VIEW Y AS
    SELECT class, ship
    FROM Ships, Outcomes
    WHERE Ships.name=Outcomes.ship AND Outcomes.result='sunk';

SELECT class, COUNT(ship) AS sunk_ships
FROM X NATURAL LEFT OUTER JOIN Y
GROUP BY class;

DROP VIEW X;
DROP VIEW Y;
```

Result:

```
CLASS           SUNK_SHIPS
--------------- ----------
Iowa                     0
Revenge                  0
Kongo                    1
```

Exercise 3. (4 points)

Write the following modifications.

1. (2 points) Two of the three battleships of the Italian Vittorio Veneto class – Vittorio Veneto and Italia – were launched in 1940; the third ship of that class, Roma, was launched in 1942. Each had 15-inch guns and a displacement of 41,000 tons. Insert these facts into the database.

```
INSERT INTO Classes (class, type, country, bore, displacement)
VALUES ('Vittorio Veneto', 'bb', 'Italy', 15, 41000);

INSERT INTO Ships (name, class, launched)
VALUES ('Vittorio Veneto', 'Vittorio Veneto', 1940);

INSERT INTO Ships (name, class, launched)
VALUES ('Italia', 'Vittorio Veneto', 1940);

INSERT INTO Ships (name, class, launched)
VALUES ('Roma', 'Vittorio Veneto', 1942);
```

2. (1 point) Delete all classes with fewer than three ships.

I will consider OK a solution using in the subquery only the Ships table without the outer join with Classes, but the completely right solution is:

```
DELETE FROM Classes
WHERE class IN (
    SELECT class
    FROM Classes NATURAL LEFT OUTER JOIN Ships
    GROUP BY class
    HAVING COUNT(name)<3);
```

Deleted:

```
Yamato
North Carolina
Renown
Tennessee
Bismarck
```

3. (1 point) Modify the Classes relation so that gun bores are measured in centimeters (one inch = 2.5 cm) and displacements are measured in metric tons (one metric ton = 1.1 ton).

```
UPDATE Classes
SET bore = bore*2.5, displacement = displacement/1.1;
```

Please do this question at the very end of the assignment because some of the later questions need the bore in inch.

Exercise 4. (12 points)

Add the following constraints.
[Some constraints might not possible be added right away. In such cases, use the Exceptions table (you need to create it first) to find the violating tuples. Then delete the violating tuples. See constraints2.ppt for an example.]
[Some constraints need a view with check option in order to be enforced.]

1. (1 point) Every class mentioned in Ships must be mentioned in Classes.

```
ALTER TABLE Classes ADD CONSTRAINT classes_pk PRIMARY KEY(class);

ALTER TABLE Ships ADD CONSTRAINT ship_to_classes_fk
FOREIGN KEY(class) REFERENCES Classes(class)
    EXCEPTIONS INTO Exceptions;
```

This doesn’t go through. There are violating tuples in Ships. Let’s see them.

```
SELECT Ships.*, constraint
FROM Ships, Exceptions
```

```
WHERE Ships.rowid = Exceptions.row_id;
```

Let’s delete them.

```
DELETE FROM Ships
WHERE class IN (
    SELECT class
    FROM Ships, Exceptions
    WHERE Ships.rowid = Exceptions.row_id
);
```

Now, we can set the constraint.

```
ALTER TABLE Ships ADD CONSTRAINT ship_to_classes_fk
FOREIGN KEY(class) REFERENCES Classes(class);
```

2. (1 point) Every battle mentioned in Outcomes must be mentioned in Battles.

```
ALTER TABLE Battles ADD CONSTRAINT battles_pk PRIMARY KEY(name);

ALTER TABLE Outcomes ADD CONSTRAINT outcomes_to_battles_fk
FOREIGN KEY(battle) REFERENCES Battles(name)
    EXCEPTIONS INTO Exceptions;
```

3. Every ship mentioned in Outcomes must be mentioned in Ships.

```
 ALTER TABLE Ships ADD CONSTRAINT ships_pk PRIMARY KEY(name);

ALTER TABLE Outcomes ADD CONSTRAINT outcomes_to_ships_fk
FOREIGN KEY(ship) REFERENCES Ships(name)
    EXCEPTIONS INTO Exceptions;
```

This doesn’t go through. There are violating tuples in Outcomes. Let’s see them.

```
SELECT Outcomes.*, constraint
FROM Outcomes, Exceptions
WHERE Outcomes.rowid = Exceptions.row_id;
```

Let’s delete them.

```
DELETE FROM Outcomes
WHERE ship IN (
    SELECT ship
    FROM Outcomes, Exceptions
    WHERE Outcomes.rowid = Exceptions.row_id
);
```

Now, we can set the constraint.

```
ALTER TABLE Outcomes ADD CONSTRAINT outcomes_to_ships_fk
    FOREIGN KEY(ship) REFERENCES Ships(name);
```

4. (1 point) No class of ships may have guns with larger than 16-inch bore.

```
ALTER TABLE Classes ADD CONSTRAINT bore_size CHECK(bore<=16)
EXCEPTIONS INTO Exceptions;
```

This doesn't go through. There are violating tuples in Classes. Let's see them.

```
SELECT Classes.*, constraint
FROM Classes, Exceptions
WHERE Classes.rowid = Exceptions.row_id;
```

After deleting those tuples, we can set the constraint.

5. (2 points) If a class of ships has more than 9 guns, then their bore must be no larger than 14 inches.

Recall that, p->q is the same as (NOT p) OR q. Thus, we have

```
ALTER TABLE Classes ADD CONSTRAINT check_bore
CHECK(numGuns<=9 OR bore<=14);
```

6. (2 points) No ship can be in battle before it is launched.

We will use a view WITH CHECK OPTION for this. Also observe the TO_NUMBER function which converts a string into a number (e.g. '2008' into 2008).

```
CREATE OR REPLACE VIEW OutcomesView AS
    SELECT ship, battle, result
    FROM Outcomes O
    WHERE NOT EXISTS (
        SELECT *
        FROM Ships S, Battles B
        WHERE   S.name=O.ship AND O.battle=B.name AND
                S.launched > TO_NUMBER(TO_CHAR(B.date_fought, 'yyyy'))
    )
WITH CHECK OPTION;
```

Now we can try some insertion on this view.

```
INSERT INTO OutcomesView (ship, battle, result)
```

```
VALUES('Musashi', 'North Atlantic','ok');
```

This insertion, as expected, will fail since Musashi is launched in 1942, while the North Atlantic battle took place on 27-MAY-41.

7. (2 points) No ship can be launched before the ship that bears the name of the first ship's class.

   ```
   CREATE OR REPLACE VIEW ShipsV AS
   SELECT name, class, launched
   FROM Ships X
   WHERE NOT EXISTS (
     SELECT *
     FROM Ships Y
     WHERE Y.class=X.class AND Y.name=Y.class AND y.launched>x.launched
   )
   WITH CHECK OPTION;
   ```

   Now we can try some insertion on this view.

   ```
   INSERT INTO ShipsV(name, class, launched)
   VALUES ('AAA','Kongo',1912);
   ```

   This insertion, as expected, will fail since ship Kongo (first ship of class Kongo) is launched in 1913.

8. (2 points) No ship fought in a battle that was at a later date than another battle in which that ship was sunk.

   ```
   CREATE OR REPLACE VIEW OutcomesV AS
   SELECT ship, battle, result
   FROM Outcomes X
   WHERE NOT EXISTS (
     SELECT *
     FROM Outcomes Y
     WHERE Y.ship=X.ship AND Y.result='sunk' AND
         (SELECT date_fought FROM battles WHERE name=X.battle) >
         (SELECT date_fought FROM battles WHERE name=Y.battle)
   )
   WITH CHECK OPTION;
   ```

   Now we can try some insertion on this view.

   ```
   INSERT INTO OutcomesV(ship, battle, result)
   VALUES('Bismarck', 'Guadalcanal', 'ok');
   ```

This insertion, as expected, will fail since 'Bismarck' was sunk in the battle of North Atlantic, in 1941, whereas the battle of Guadalcanal happened in 1942.