```
------------------------------------------------------------------------
-- Section 1: Creating schemas in SQL, i.e. SQL DDL
------------------------------------------------------------------------

-- Create a table to store student information
CREATE TABLE Students (
    name VARCHAR(80),
    bday DATE,
    hobbies VARCHAR(100),
    uwid INTEGER,
    PRIMARY KEY (uwid) -- Do not allow two tuples with the same uwid
    );

-- Add sample tuples to the Student table
INSERT INTO Students VALUES ('Jane Doe', '1990-03-01', 'sailing', 111);
INSERT INTO Students VALUES ('Joe Smith', '1991-05-12', 'dancing', 222);
INSERT INTO Students VALUES ('Goof Ball', '1992-12-31', 'watching TV', 333);

-- Create a table to record course information
CREATE TABLE Courses (
    name VARCHAR(80),
    description VARCHAR(200) UNIQUE, -- each course has a different description
    cid INTEGER,
    PRIMARY KEY (cid) -- the cid must be unique in this table
    );

-- Create an index (B-tree) on the Students.name
CREATE INDEX BtreeOnCoursesCid ON Courses(cid);

-- Add sample tuples to the Courses table
INSERT INTO Courses VALUES ('CS564', 'Intro to Database Management Systems', 564);
INSERT INTO Courses VALUES ('CS536', 'Operating Systems', 536);

-- Create a table to keep track of enrollments.
CREATE TABLE Enrolled (
    uwid INTEGER,
    cid INTEGER,
    edate DATE,
    credits INTEGER,
    grade char,
    FOREIGN KEY (uwid) REFERENCES Students,
    FOREIGN KEY (cid) REFERENCES Courses);

-- Populate the Enrolled tables
INSERT INTO Enrolled VALUES (111, 564, '2012-01-12', 4, NULL);
INSERT INTO Enrolled VALUES (111, 536, '2012-01-19', 3, NULL);
INSERT INTO Enrolled VALUES (222, 564, '2012-01-23', 4, NULL);

-- In some systems like SQLite, you have to put explicit NULL, so
-- INSERT INTO Enrolled VALUES (111, 564, '2012-01-12', 4, NULL);
```

```
----------------------------------------------------------------------
-- Section 2: Querying in SQL -- i.e. SQL DML
----------------------------------------------------------------------

-- Sample Queries
SELECT * FROM Students;

-- Can only select some attributes
SELECT name FROM Students;

-- Can select attributes in any order
SELECT bday, name FROM Students WHERE uwid = 111;

-- Can have more complex WHERE clause
SELECT * FROM Students WHERE bday > '1991-01-01';
SELECT * FROM Students WHERE bday > '1991-01-01' AND hobbies <> 'watching TV';

-- String matching
SELECT * FROM Students WHERE name LIKE 'J_n%';
SELECT * FROM Students WHERE name LIKE 'J_n%' OR name LIKE '%Doe';

-- A Join Query
SELECT * FROM Students S, Enrolled E WHERE E.uwid = S.uwid;

-- An aggregate query
SELECT COUNT(*) FROM Students;
SELECT MAX(bday), MIN(bday) FROM Students;
SELECT AVG(credits), cid FROM Enrolled GROUP BY cid;

-- See what happens to the grad column in Enrolled. They have "NULL" values
SELECT * FROM Enrolled;

-- Lets give every one in 564 and 'A' grade
UPDATE Enrolled SET grade = 'A' WHERE cid = 564;

-- We can query only for the null values
SELECT * FROM Enrolled WHERE grade IS NULL;

-- Or only for the non-null values
SELECT * FROM Enrolled WHERE grade IS NOT NULL;


-- Find all students who are registered in SOME class
SELECT S.uwid, S.name
  FROM Students S, Enrolled E
 WHERE S.uwid = E.uwid;

-- Find all students who are not registered in ANY classes
SELECT S.uwid, S.name
  FROM Students S
 WHERE S.uwid NOT IN (SELECT S.uwid
                        FROM Students S, Enrolled E
                        WHERE S.uwid = E.uwid);
```

```
--------------------------------------------------------------------------------
-- Section 3: Views in SQL and EXPLAIN
--------------------------------------------------------------------------------

-- We are going to ask this query many times, so lets create a view
CREATE VIEW GoodStudents AS
SELECT S.uwid, S.name
  FROM Students S, Enrolled E
 WHERE S.uwid = E.uwid;

-- Find all students who are not registered in ANY classes
SELECT S.uwid, S.name
  FROM Students S
 WHERE S.uwid NOT IN (SELECT uwid FROM GoodStudents);

-- What does this "Query Plan" look like? More on this later in the semester.
EXPLAIN SELECT S.uwid, S.name
          FROM Students S
         WHERE S.uwid NOT IN (SELECT uwid FROM GoodStudents);




--------------------------------------------------------------------------------
-- Section 4: Transactions in SQL
--------------------------------------------------------------------------------

-- Transactions
START TRANSACTION; -- start the transaction
    UPDATE Enrolled SET grade = 'D' WHERE cid = 564; -- oops query
    SELECT * from Enrolled;
ROLLBACK; -- no worries we can roll back. The other way to end the transaction is
"COMMIT"
SELECT * from Enrolled;




--------------------------------------------------------------------------------
-- Section 5: Database Cleanup
--------------------------------------------------------------------------------

-- Drop all the tables
DROP VIEW GoodStudents;
DROP TABLE Enrolled;
DROP TABLE Courses;
DROP TABLE Students;
```