---------- SQLite3 Tutorial 4 ----------

```
-- Applying Functions in SQLite

-- Find the Best and Worst Scores on all quizzes and tests

-- test_score : student_id, test_id, score
-- test : date, test_type, id
-- student : f_name, l_name, sex, id

SELECT test.date AS Date,
MIN(test_score.score) AS Worst,
MAX(test_score.score) AS Best
FROM test_score, test
WHERE test_score.test_id = test.id
GROUP BY test.date;

-- Print the average score on each test

SELECT test.date AS Date,
AVG(test_score.score) 'Avg Score'
FROM test_score, test
WHERE test_score.test_id = test.id
GROUP BY test.date;

-- List all students that had a test score over 20

SELECT f_name || ' ' || l_name AS Name, test_score.score AS Score
FROM test_score, student
WHERE test_score.score > 20 AND test_score.student_id = student.id
GROUP BY Name;

-- VIEWS IN SQLite --

-- A view is used to store a queries result. It is not part of the schema

CREATE VIEW ScoreOver20 AS
SELECT f_name || ' ' || l_name AS Name, test_score.score
FROM test_score, student
WHERE test_score.score > 20
AND test_score.student_id = student.id
GROUP BY Name;

drop view ScoreOver20; -- Delete the view

-- TRIGGERS in SQLite --

-- Triggers are operations that are automatically performed when a specific
-- event occurs

-- test : date, test_type, id
-- test_score : student_id, test_id, score
-- student : f_name, l_name, sex, id
```

```
-- Will Hold Data When a Student Has a Makeup Test

CREATE TABLE Log(
id INTEGER PRIMARY KEY,
test_id INTEGER NOT NULL,
date     DATE NOT NULL,
student_id INTEGER NOT NULL,
FOREIGN KEY (test_id) REFERENCES test_score (test_id),
FOREIGN KEY (student_id) REFERENCES test_score (student_id));

-- The Trigger that updates the Log when test_score is updated

CREATE TRIGGER test_score_update
AFTER UPDATE OF score ON test_score
BEGIN
INSERT INTO Log(test_id, date, student_id)
VALUES(new.test_id, date('now'), new.student_id);
-- Don't reference table instead use new
END;

select * from absence; -- Show all absences

UPDATE test_score
SET score=20
WHERE test_id=1 AND student_id=2;

-- LIKE can be used with % to match a series of characters and zero or more
-- characters there after

select l_name, f_name
from student
where l_name LIKE 'J%';

-- _ can be used to represent any 1 character or space
-- Match all 5 letter long last names

select l_name, f_name
from student
where l_name LIKE '_____';

-- ORDER BY allows you to define sorting either DESC or ASC
-- LIMIT allows you to limit your results
-- OFFSET will skip the first number or results

select l_name, f_name
from student
where f_name LIKE 'J%'
ORDER BY l_name ASC, f_name LIMIT 10 OFFSET 2;

-- Random SQLite Functions

SELECT random(); -- Generate random number
```

```
SELECT ABS(RANDOM() % 100); -- Random number between 0 and 100

SELECT LOWER(f_name),
UPPER(l_name)
FROM student;

SELECT LENGTH('Iron Man'); -- Returns the number of characters in a string

SELECT COUNT(*) FROM student; -- Number of rows in the table

SELECT date(); -- Return the current date

SELECT time(); -- Return the current time

SELECT datetime(); -- Return the current date and time

SELECT date('now', '-30 days'); -- Get the date 30 days ago

SELECT date('now', '-20 months'); -- Get the date 30 days ago

SELECT date('now', 'weekday 0'); -- Get the date of the next Sunday

SELECT time('now', '-1000 minutes');

SELECT time('now', '-1000 seconds');

SELECT strftime('%m-%d-%Y'); -- You can modify the date format

-- Find Thanksgiving day

SELECT date('now', 'start of year', '10 months', '21 days', 'weekday 4');
```