В книге около 300 страниц. За 1-2 дня прочитать несложно. Для тех, у кого нет времени на чтение, попробую изложить коротко основные моменты. Но рекомендую прочитать её полностью.  
  
Участники разработки ПО:  
1. Менеджер проекта — специалист, занимающийся вопросами поиска заказчиков проектов и исполнителей  
2. QA-инженер — специалист, задача которого организовать процесс разработки таким образом, чтобы работа была выполнена в срок и на надлежащем уровне качества.  
3. Продюсер — специалист, задача которого составить спецификацию (spec)  
4. Программист — специалист, занимающийся написанием или корректировкой кода программы  
5. Тестировщик — специалист, занимающийся поиском багов  
  
Цикл разработки ПО состоит из:  
1. Идея.   
2. Разработка дизайна продукта и создание документации.   
3. Кодирование или создание кода.   
4. Исполнение тестирования и ремонт багов.   
5. Релиз.  
  
Цикл тестирования ПО состоит из трех этапов:   
1. Изучение и анализ предмета тестирования.   
2. Планирование тестирования.   
3. Исполнение тестирования.   
  
Основные понятия:  
  
1. Тестирование — это сравнение фактического результата с ожидаемым.  
2. Цели тестирования — нахождение багов до того, как их найдут пользователи.  
3. Баг (bug) — это отклонение фактического результата от ожидаемого.  
4. Спецификация (spec) — это детальное описание того, как должно работать ПО. Так же, это детальное описание ожидаемого результата. (В спецификации тоже могут быть баги, например, двусмысленные предложения).  
5. Тест-кейс — это инструмент тестировщика, предназначенный для документирования и проверки одного или более ожидаемых результатов.   
6. Тест-комплект — совокупность тест-кейсов находящихся, как правило, в одном документе, которые проверяют какую-то определенную часть нашего проекта.  
7. Шаги тест-кейса (procedure) — это часть тест-кейса, ведущая исполнителя тест-кейса к фактическому результату. (Излишняя детализация может осложнить поддержку, а излишнее абстрагирование привести к непониманию того, как исполнить тест-кейс).  
8. Front end — это непосредственный интерфейс пользователя (текст, картинки, кнопки, линки и прочие вещи, которые видно в окне приложения)  
9. Back end — это то что на заднем фоне приложения (веб-сервер, код приложения, база данных и т.д.).  
10. New feature testing — тестирование новых компонентов.  
11. Regression testing — исполнение старых тест-кейсов для проверки того, что старые компоненты ПО еще работают.  
12. СТБ (Bug Tracking System) — Система в которую заносятся баги.  
13. Git — распределённая система управления версиями файлов (для управления коллекцией исправлений, патчей).  
  
Виды тестирования:  
  
1. По знанию внутренностей системы:   
• черный ящик (black box testing) — тестирование программы без доступа к коду;  
• белый ящик (white box testing) — тестирование программы только по коду;  
• серый ящик (grey box testing) — тестирование без кода+тестирование по коду.  
  
2. По объекту тестирования:   
• функциональное тестирование (functional testing) — например, проверка выводимого результата;  
• тестирование интерфейса пользователя (UI testing) — из названия понятно;  
• тестирование локализации (localization testing) — например, проверка шрифтов и другая адаптация приложения для пользователей;  
• тестирование скорости и надежности (load/stress/performance testing) — например, проверка скорости загрузки сайта при определенном количестве пользователей;  
• тестирование безопасности (security testing) — суть в том, чтобы усложнить условия для кражи данных (например телефонов и др. личной информации);  
• тестирование опыта пользователя (usability testing) — суть в том, чтобы интерфейс был интуитивно понятен даже непродвинутым пользователям;  
• тестирование совместимости (compatibility testing) — запуск на разных операционках и браузерах.  
  
3. По субъекту тестирования:   
• альфа-тестировщик (alpha tester) — тестирование сотрудниками фирмы;  
• бета-тестировщик (beta tester) — тестирование пользователями.  
  
4. По важности тестирования:  
• сначала тестирование новых функциональностей (new feature testing) — тестирование новых функциональностей;  
• потом регрессивное тестирование (regression testing) — повторное тестирование старых функций.  
  
5. По критерию «позитивности»сценариев:   
• позитивное тестирование (positive testing) — тестируем ожидаемыми методами;  
• негативное тестирование (negative testing) — тестируем нестандартными методами(например вводим вместо 9 цифр — 11 букв).  
  
6. По степени изолированности тестируемых компонентов:  
• компонентное тестирование (component testing) — это тестирование одного логического компонента;  
• интеграционное тестирование (integration testing) — это тестирование на уровне двух или больше логических компонентов;  
• системное тестирование (system or end- to-end testing) — это проверка всей системы от начала до конца.  
  
7. По степени автоматизированности тестирования:   
• ручное тестирование (manual testing) — это исполнение тест-кейсов без помощи каких-либо программ, автоматизирующих вашу работу (например, создаем аккаунт вручную);  
• автоматизированное тестирование (automated testing)- акаунт создается программой автоматически;  
• смешанное/полуавтоматизированное тестирование (semi automated testing) — создаем акаунт вручную, но закупки сделаются автоматически.  
  
8. По степени подготовки к тестированию:   
• тестирование по документации (formal/documented testing) — тестирование по тест-кейсам;  
• эд хок-тестирование (ad hoc testing) — интуитивное тестирование без документации (например, когда что-то нужно быстро проверить).

Также по документам существует:  
• Тест-смета (Test Estimation) — документ, включающий в себя предварительную оценку времени, необходимого на подготовку к тестированию и на тестирование новых фича (new feature testing);  
• Тест-план (test-plan) — документ, обобщающий и координирующий тестирование (подробнее об этом документе можно узнать в книге [Савина](http://rutracker.org/forum/viewtopic.php?t=1421011)).

**II) Разобраться с SQL запросами**

SQL (structured query language) — структурированный язык запросов.  
С помощью SQL- запросов можно создавать и работать с реляционными базами данных.  
Реляционная база данных — это таблица, в которой в качестве столбцов выступают поля данных, а каждая строка хранит данные.   
  
SQL определяется Американским Национальным Институтом Стандартов и Международной Организацией по стандартизации (ISO)  
Несмотря на это, некоторые производители баз данных вносят изменения и дополнения в этот язык. Эти изменения незначительны и основа остаётся совместимой со стандартом. (например ms sql, my sql, postgreSQL).  
  
В каждой таблице должно быть одно уникальное поле, которое однозначно будет идентифицировать строку. Это поле назовем ключевым (Key1, Key2..).  
В качестве ключа обычно используют численный тип и если позволяет база данных, то он будет типа «autoincrement» (автоматически увеличивающееся).  
  
Столбцы в базе данных, также должны быть уникальными, но в этом случае не обязательно числовыми. Их можно называть как угодно, лишь бы было уникально и понятно.  
  
SQL может быть двух типов: интерактивный и вложенный. Интерактивный — это отдельный язык, он сам выполняет запросы и сразу показывает результат работы. Второй — это когда SQL язык вложен в другой, например в С++ или Delphi.  
  
Так как мы формируем минимальный список знаний трудоустройства, мы рассмотрим интерактивный SQL.  
  
Представим, что у нас есть две таблицы:  
  
Prog.db  
Key1 / ProgName / Cost  
1 / Windows 95 / 100  
2 / Windows 98 / 120  
  
и  
  
User.db  
Key1 / Key2 / LastName  
1 / 1 / Иванов  
2 / 1 / Петров  
3 / 2 / Сидоров  
  
Рассмотрим первый запрос:  
  
SELECT \*  
FROM Prog, User  
WHERE Prog.Key1= Key2  
AND ProgName LIKE 'Windows 95'  
  
Выбрать (SELECT) все поля (\*) из (FROM) баз данных Prog и User, где (WHERE) есть связь(Prog.Key1 и Key2) Prog.Key1= Key2 и ProgName LIKE 'Windows 95'.  
LIKE это тоже самое что равно(=) только для строк  
  
Результатом этого запроса будет:  
  
Prog.db User.db  
Key1 / ProgName / Cost / Key1 / Key2 / LastName   
1 / Windows 95 / 100 / 1 / 1 / Иванов  
1 / Windows 95 / 100 / 2 / 1 / Петров  
  
Отредактируем немного запрос:  
SELECT Prog.Key1, Prog.ProgName, Prog.Cost\*2 'руб',  
Cost.Key1, Cost.Key2, Cost.LastName  
FROM Prog, User  
WHERE Prog.Key1= Key2  
  
Prog.Cost\*2 'руб' — эта запись говорит, что к каждое значение надо умножить на 2 и прибавить строку 'руб'.  
  
Результат:  
Prog.db User.db  
Key1 / ProgName / Cost / Key1 / Key2 / LastName   
1 / Windows 95 / 200 руб / 1 / 1 / Иванов  
1 / Windows 95 / 200 руб / 2 / 1 / Петров  
  
Для сортировки используется команда ORDER BY. После этого пишутся поля, по которым надо отсортировать. В самом конце нужно поставить АSC (сортировать в порядке возрастания) или DESC (в порядке убывания). Если ты не ставишь АSC или DESC, то таблица сортируется по возрастанию и подразумевается параметр АSC.  
  
Например:  
SELECT \*  
FROM Prog  
ORDER BY ProgName DESC  
  
Результатом будет таблица Prog, отсортированная по полю ProgNamе в порядке убывания.  
  
**SQL калькулятор:**  
Вот несколько функций:  
• COUNT — подсчёт количества строк;  
• SUM — подсчёт суммы;  
• AVG — подсчёт среднего значения;  
• MAX — поиск максимального значения;  
• MIN — поиск минимального значения.  
  
Этот запрос просто подсчитывает количество строк в базе:  
SELECT COUNT(LecNumber)  
FROM User  
  
Этот запрос опять подсчитывает количество строк, но теперь результатом будет количество народу, у которых поле LecNumber = 1:  
SELECT COUNT(LecNumber)  
FROM User  
WHERE LecNumber=1  
  
Этот запрос выводит количество лицензий и единицу измерения в одном столбце. Здесь к числу прибавляется текст:  
SELECT LecNumber+'шт.'  
FROM User  
  
**Работа с полями:**  
NSERT (вставить), UPDATE (модифицировать), DELETE (удалить).   
После оператора VALUES идёт перечисление всех полей строки. Теперь рассмотрим пример:  
INSERT INTO User1  
VALUES ('Иванов', 'Сергей', 34);  
  
Этой командой мы вставили строку и присвоили значения полям. В таблице три поля: первые два поля строковые (Фамилия и Имя), последнее поле — целое число (возраст). Типы данных обязаны совпадать с теми, что установлены в таблице.  
  
Если не надо задавать все поля, тогда можно оставить их пустыми с помощью NULL:  
INSERT INTO User1  
VALUES ('Иванов', NULL, 34);  
  
Если таблица с большим количеством полей и нужно заполнить только два из них?   
Решение:  
  
INSERT INTO User1 (Family, Age)  
VALUES ('Иванов', 35);  
  
После конструкции INSERT INTO и имени базы стоят скобки, где перечислены поля, которые необходимо заполнить (Фамилия и Возраст). В скобках после слова VALUES перечисляем эти поля в той же последовательности, в которой перечислил перед этим (сначала фамилия, а потом возраст).  
  
Теперь представь, что мы хотим сохранить результат запроса SELECT в отдельной таблице. Для этого в SQL всё уже предусмотрено. Нужно только написать:  
  
INSERT INTO User1  
SELECT \*  
FROM User2  
WHERE Age=10  
  
В этом примере сначала выполнится запрос SELECT:  
  
SELECT \*  
FROM User2  
WHERE Age=10  
  
После его выполнения результат будет занесён в таблицу User1. Важно, что количество столбцов в запросе и результирующей таблицы должно быть одинаково. А самое главное — это чтобы тип данных совпадал  
  
Теперь рассмотрим такой запрос:  
  
INSERT INTO User1(Name,Age)  
SELECT Name,Age  
FROM User2  
WHERE Age=10  
  
Теперь в таблицу User1 будут перенесены только два столбца (имя и возраст). Поля должны быть перечислены в таком порядке, чтобы типы и длина полей совпадали.  
  
Мы смогли добавить строки, но надо и научиться изменять данные. Для этого нам доступна команда UPDATE. Сразу же попробуем взглянуть на пример:  
  
UPDATE User1  
SET age=65  
  
Первая строка говорит о том, что нам надо обновить базу User1. Вторая строка начинается с оператора SET (установить). После этого мы пишем поле, которое хотим обновить, и присваиваем ему значение.  
  
Если нужно обновить только определённые строки, то ты должен написать так:  
UPDATE User1  
SET age=65  
WHERE Name LIKE 'Вася'  
  
или  
  
UPDATE User1  
SET age=age+1  
  
или  
  
UPDATE User1  
SET age=age+1, Name='Иван'  
WHERE Family LIKE 'Сидоров'  
  
Этот запрос увеличит поле Age на единицу и установит поле Name в «Иван» во всех строках, где поле Family равно «Сидоров».  
  
Теперь команда DELETE:  
  
DELETE FROM User1  
  
Эта конструкция удаляет абсолютно все строки из таблицы User1. Можно сказать, что этим мы очищаем таблицу.  
  
Теперь рассмотрим другой пример:  
  
DELETE FROM User1  
WHERE Age=10  
  
Этот пример удаляет только те строки, в которых поле Age равно 10.  
  
Понимания этих запросов будет достаточно.

**III) Разослать резюме**

Самая короткая глава, это все умеют, hh.ru вам в помощь. Там же можно составить резюме. Перед составлением не забывайте смотреть резюме своих конкурентов.

Обычно собеседование делятся на 3 этапа:  
1) Встреча с девушкой из отдела кадров. На этом этапе будет несколько вопросов по вашим знаниям, чтобы было понятно, что вы в «теме» и много общих вопросов, чтобы определить вашу адекватность.  
2) Тестовое задание — обычно дается на дом  
3) Встреча с руководителем где будут вопросы, касающиеся ваших навыков тестирования.  
  
Тестовое задание и встреча с руководителем вряд ли уйдет за рамки того, что мы написали выше. Рассмотрим первую ситуацию, где нам будут задавать вопросы на нашу адекватность.  
  
Предположим, что у нас нет никакого официального опыта работы тестировщиком, в нашей голове только теория, хобби- программирование на самом начальном этапе, а в трудовой запись «менеджер по продажам».  
  
Рассмотрим основные вопросы и и примеры самых обычных ответов на них:  
  
1) Почему вы решили стать тестировщиком?   
Меня всегда тянуло в IT сферу, эта профессия больше всего подходит моему характеру и моим интересам.   
  
2) Что больше всего вам нравится в тестировании?   
Я обожаю анализировать и изучать программы, например на прошлом месте работы мне больше всего нравилось работать с 1с, я даже смог поставить одну из версий себе на домашний компьютер, для более детального изучения программы, без всяких ограничений.  
  
3) Какими достоинствами должен обладать эффективный тестировщик?   
Честность, внимательность (для поиска багов), общительность(так как нужно будет много общаться с персоналом), обучаемость (без этого никуда), умение работать с большим объемом информации и умение расставлять приоритеты, и, конечно, стрессоустойчивость.  
  
4) Ваше самое большое профессиональное достижение (не обязательно в области тестирования).   
Создание и поддержание базы для логистики и закупа, которой пользовались все менеджеры компании. По этой базе, я мог закупить все, что мне нужно максимально дешево, и с любой точки России доставить товар с отличным соотношением цены, качества и риска.  
  
5) Почему вы ушли (уходите) из своей предыдущей компании?   
Мне нравилось работать в прошлой компании, но я хочу опробовать себя как тестировщик ПО. Так как данная сфера ближе к моим интересам, характеру и увлечениям. На этой должности я буду получать гораздо большое удовольствия от работы.  
  
6) Приведите пример сложной ситуации, с которой вы столкнулись в своей карьере, и какой выход из нее вы нашли?   
Из-за текучки кадров, мне часто приходилось брать обязанности других на себя, например доставка или закупки. И чтобы свести всю свою деятельность в единую систему, в свободное от работы время я создал базу (excel), которая собрала в кучу всю мою старую и новую деятельность. Эта база не дала мне запутаться в огромном количестве работы. Так же она помогла избавиться от блокнотов и стикеров, а это значит, что покупатели всегда видели порядок за моим рабочим столом.  
  
7) Что бы вы пожелали усовершенствовать в себе? Что вы для этого делаете?   
У меня есть хобби — программирование. Хочу совершенствовать свои навыки в этой области, они же в дальнейшем помогут улучшить мои навыки тестировщика.  
  
8) Что вы ждете от нашей компании?  
Хороший коллектив, и профессиональное развитие  
  
9) Какой минимальный доход вас устроит?  
Можно посмотреть средний доход по региону на эту вакансию и назвать его. Но, если вы проходите собеседование в 2гис, то говорите что готовы работать бесплатно 24/7/365, как рекомендует [Савин](http://rutracker.org/forum/viewtopic.php?t=1421011). Так как эта книга эталон для их отдела кадров и им будет приятно знать что вы ее уже прочитали.  
  
10) Какими источниками вы пользуетесь для развития в этой области  
Можно посмотреть в конец статьи и назвать все, что будет в списке «Источники, которые были использованы для написания статьи:»  
  
11) Кем вы видите себя через 5 лет?  
Тут на ваш вкус.  
  
В принципе все. У вас все получится. Главное: