COMP-3150 CHAPTERs 6 to 7: LECTURE NOTES (Chs 6 & 7) Covered on Oct. 27, 2020 or Oct. 21, 2021; Oct. 29, 2020 or Oct. 26, 2021; Nov. 3, 2020 or Oct. 28, 2021; Nov. 5, 2020 or Nov. 2, 2021; Nov. 10, 2020 or Nov. 4, 2021; and Nov. 12, 2020 or Nov. 9, 2021

- Chapter 6: BASIC SQL
- Chapter 7: More SQL: Complex Queries, Triggers, Views, and Schema Modification

*** Note that Midterm 2 is written in the class on Thursday, ~~Nov. 5, 2020~~, Nov. 4, 2021 in online in class. Come to class as usual and log on Blackboard to do your midterm 2 online during the class. The following are the instructions to follow for Midterm 2 so you do not waste too much time going over them during the test:
**

INSTRUCTIONS (Please Read Carefully)
Examination Period is 1 hours 20 minutes (For online version time extended to 2 hours)
Answer all questions. Write your answers in the spaces provided in the question paper. This is closed book and closed notes test. You can type in your answers into the word file and submit, or print, write with hand, scan clearly into only a .pdf or .jpeg file and submit.
Total Marks =50. Total number of sections = 2
Please read questions carefully! Misinterpreting a question intentionally or unintentionally results in getting a "ZERO" for that question. Good Luck!!!

CONFIDENTIALITY AGREEMENT & STATEMENT OF HONESTY
I confirm that I will keep the content of this assignment/examination confidential.
I confirm that I have not received any unauthorized assistance in preparing for or doing examination. I confirm knowing that a mark of 0 may be assigned for copied work.
For Online Test/Examination in Comp 3150 Fall 2020: (additional rules to be observed):
1. I confirm that I agree to write this final examination as a closed book examination.
2. I confirm that I am the student with the name and student id signed below.
3. I confirm that I agree to not send email, chat, text or talk in any way to people other than the instructor or proctoring GA of this course during this examination.
4. I confirm that I agree to not engage in copying or cheating during this online examination.

Student Signature	Student Name (please print)
Student I.D. Number	Date

NOTE THAT I WILL BE GOING OVER IN THE LAST 10 MINUTES OF TODAY'S CLASS YOUR ANSWERS TO

THE MIDTERM 2 PRACTICE TEST 2 POSTED FOR YOU. NO FORMAL SOLUTIONS TO ANY PRACTICE TEST IS POSTED ON THE WEB. HOWEVER, IT WILL BE REVIEWED WITH YOU IN THIS CLASS. PARTS OF THE SOLUTION MAY ALSO BE FOUND AT THE END OF THIS LECTURE NOTES.

Winter 2021 Comp 3150 online recordings can be downloaded from the black board virtual classroom for the class of the day. The links below are the recordings of similar classes in Fall 2020 that I uploaded to one drive in case you prefer to review these.

Links to any posted recorded in-class lectures for Chapters 6 and 7 of Oct. 27, 2020 (similar to Oct. 21, 2021), Oct. 29, 2020 (similar to Oct. 26, 2021) and possibly Nov. 3, 2020 (similar to Nov. 2, 2021) and others will be found below:

1. Links on one drive to recorded lecture of Tuesday, Oct. 27, 2020 (similar to Oct. 21, 2021), (saved in two recorded sessions to break down the files) is provided below in sequences 1, 2:

Frist Recording 1 of 2:

https://uwin365-my.sharepoint.com/:v:/g/personal/cezeife_uwindsor_ca/EWLfQT-pHONBmVxRKKV9KxoBqSy4lMbfHC4kiO9XNG6V5A

Second Recording 2 of 2:

https://uwin365-my.sharepoint.com/:v:/g/personal/cezeife_uwindsor_ca/EZ62-Z3SM81NmP5XaOFW_uEB725Ok9UKmL5FEe8sSBEB8A

2. Links on one drive to recorded lecture of Thursday, Oct. 29, 2020 (similar to Oct. 26, 2021) is provided below in sequence 1 only:

Frist Recording 1 of 1:

https://uwin365-my.sharepoint.com/:v:/g/personal/cezeife_uwindsor_ca/EQMrPXoyrzhBgIb5jEQ4fHsB5pRz0qcn2MYPKWDebga4Xw

3. Links on one drive to recorded lecture of Tuesday, Nov. 3, 2020 (similar to Oct. 28, 2021) is provided below in sequence 1:

Frist Recording 1 of 1:

https://uwin365-my.sharepoint.com/:v:/g/personal/cezeife_uwindsor_ca/ETAoZ8MPLHtMiS1mc9-Ql4YBdyDNuFAsco6j7SwgQKsU4g

4. Links on one drive to recorded lecture of Thursday, Nov. 5, 2020 (similar to Nov. 2, 2021) is provided below in sequences 1:

Frist Recording 1 of 1:

https://uwin365-my.sharepoint.com/:v:/g/personal/cezeife_uwindsor_ca/EX-eAD_AFZrlMudsb5rSsOOcBKkhat7qm8b0OCmcv6mTerQ

5. Links on one drive to recorded lecture of Tuesday, Nov. 10, 2020 (similar to Nov. 4, 2021) is provided below in sequences 1:

Frist Recording 1 of 1:

https://uwin365-my.sharepoint.com/:v:/g/personal/cezeife_uwindsor_ca/ESc9E3PqXqdDpb0e1fzIo_oBLxwxlN1mZzSNdVng0QyN0w

6. Links on one drive to recorded lecture of Thurday, Nov. 12, 2020 (similar to Nov. 9, 2021) is provided below in sequences 1:

Frist Recording 1 of 1:

https://uwin365-my.sharepoint.com/:v:/g/personal/cezeife_uwindsor_ca/EWPFKlIVL15NvDippkysA84BtbIX93PhEz5PrqC-ZotFZg

***

The following questions on Basic SQL discussed in Chapters 6 and 7 of Comp 3150 text book, Chapters 6 and 7, Comp 3150 posted course slide notes, are in-class questions for students to ponder and answer as I teach.

- The answers to the questions are found also by reviewing the Comp 3150, posted power point slide notes for Chapters 6 and 7 and being in class.

- Students are advised to review Chapters 6 and 7 of course book and Comp 3150 posted slide notes before and after each class.

I will also go over the Slide notes in class with examples and integrate them into the class lectures, which are also posted in the More course material link on black board with any links to recorded live lectures.

1. WHAT IS SQL DDL?

   - Data Definition Language (DDL) consists of SQL commands for:

i) Creating database and table schemas and constructs such as creating database schema (e.g., COMPANY), table schemas (e.g., EMPLOYEE), types, domains, views, assertions, triggers.

Note that on our Cs server managed by a DBA, the database schema corresponds to our individual Oracle accounts. Each database schema consists of a set of unique table schemas in the database that you can query.

- Main SQL command for DDL is CREATE STATEMENT.

  Eg, for creating a database schema called COMPANY associated with user Jsmith is given next.

  ```
  CREATE SCHEMA COMPANY  ‘Jsmith’;
  ```

- An example for creating a table schema for Employee in the Company database is given below:

  ```
  CREATE TABLE EMPLOYEE
  ```

```
(Fname VARCHAR2(15) NOT NULL,
 Minit CHAR,
 Lname VARCHAR2(15) NOT NULL,
 Ssn CHAR(9) NOT NULL,
Bdate DATE,
Address VARCHAR2(25),
Sex CHAR,
Salary NUMBER(10, 2),
Super_ssn CHAR(9),
Dno NUMBER NOT
NULL,
 PRIMARY KEY(Ssn));
```

- Fig. 6.1 on page 81 (slides 14 to 16 of Ch 6) shows an example set of CREATE TABLE instructions for the TABLES in the COMPANY database.

- Note that on our cs server Sqlplus system, the data types in the table schemas are defined as follows:

VARCHAR type in the book is VARCHAR2 in cs server
DECIMAL(10, 2) in the book is NUMBER(10, 2) in cs server
INT in the book is NUMBER in cs server
DATE format is cs server is enterred in the format: dd-mon-yy
Eg. 12-aug-55

ii) SQL commands for modifying the database and table schemas (Slides 31 to 35) such as:

a) DROP SCHEMA COMPANY CASCADE;
-- above is used if dropping the database schema (not table schema)

DROP TABLE DEPENDENT CASCADE CONSTRAINTS;
-- The above is for dropping table Dependent and all references to this table.

b) ALTER TABLE COMPANY.DEPARTMENT ALTER Mgr_ssn SET DEFAULT ‘333445555’;

2. WHAT IS SQL DML?

- SQL Data Manipulation Language (DML) is used for loading, deleting, updating and querying data and includes (Slides 44 to 52 of Ch 6):

i) INSERT INTO EMPLOYEE

```
        VALUES ('Franklin', 'T', 'Wong', '333445555', '12-aug-
        55', '638 Voss, Houston, TX', 'M', 40000, 888665555, 5);
```

ii) Delete from employee;

iii) UPDATE PROJECT

```
            SET PLOCATION = 'Bellaire', DNUM
            = 5
            WHERE PNUMBER=10;
```

iv) SQL DML for querying data in the database (Slides 28 to 43 of Ch 6).

- Main DML instruction is the SELECT command for select-project-join (spj) queries with the basic format on book page 188 as:

```
SELECT <attribute list>
FROM <table list>
```

```
WHERE <condition>;
```

- More expanded version of SELECT command with aggregation is on slide 26 of Ch. 7 as:

```
SELECT <attribute list>
FROM <table list>
[WHERE <condition>]
[GROUP BY <grouping attribute(s)>
[HAVING <group condition>]
[ORDER BY <attribute list>];
```

- More complex versions of SELECT command with nesting etc. are seen in Ch. 7 where the main comparison operators for connecting outer and inner queries are IN and EXISTS.

- See the summary of SQL syntax on page 235; slides 36 and 37 of Ch. 7).

**

SQL Querying in Chapters 6 & 7 can be Summarized as follows:

1. Basic SQL querying involving Select-Project-Join (SPJ) as given in the basic Select statement on Slide 29 of Ch. 6.

   SELECT <attribute list>
   FROM <table list>
   WHERE <condition>;

   Here, the <attribute list> is called the project list (that is the set of attributes or expressions in the result of the query), the <table list> is the list of database tables that can be used to answer this query; while WHERE <condition> represents the Selection and Join conditions in the SPJ query.

2. An extended version of the basic SQL query can be found on Slide 43, which includes ordering of query results either in ascending (Asc) (default order) or descending (Desc) order for any attributes in the project list. The format is:
   SELECT <attribute list>
   FROM <table list>
   [WHERE <condition>]
   [ORDER BY <attribute list>];

3. Nested Queries as discussed in Ch. 7, Slides 1 to 19.

Nested Queries are of the form:

```
SELECT <attribute list>
FROM <table list>
WHERE <condition>
```

for the outer query with a nested query in either the FROM clause or the WHERE clause of the outer query.

- Comparison operators used with nested query are:

(i) IN, =ANY, =ALL (also >, <, >=, <=, <> in place of = with ANY and ALL)

(ii) EXISTS

An example query:

Get a list of project numbers for projects that are worked on by employee with last name 'Smith'. Solve using (i) SPJ and (ii) nested form.

(i) SPJ:

```
SELECT W.PNO
```

```
FROM WORKS_ON W, EMPLOYEE E
WHERE W.ESSN=E.SSN AND
        E.LNAME='Smith';
```

(ii) Nested version:

(a) Get the Ssn of employee with Lname='Smith'. (inner query)

(b) Get the Pno from the set of Pno's in Works_on where the Ssn from the (a) set is also in the Essn set for these Works_on Pno's.

```
SELECT PNO
FROM WORKS_ON
WHERE ESSN   IN
     (SELECT SSN
      FROM  EMPLOYEE
      WHERE  LNAME='Smith');
```

4. Extension of basic SQL query to also include aggregate functions (COUNT, SUM, MAX, MIN, AVG) for summarizing data. This extension also has options with the Grouping and Having clauses as discussed on slide 26

of Chapter 7. The format is:

```
SELECT <attribute list>
FROM <table list>
[WHERE <condition>]
[GROUP BY <grouping attribute(s)>
[HAVING <group condition>]
[ORDER BY <attribute list>];
```

5. Full summary of SQL language syntax is provided on slides 36 and 37 of Chapter 7 (as given on page 235 of course book).

**

ABOUT EXPRESSING JOIN IN QUERIES WHEN POSING QUERIES INVOLVING MORE THAN ONE DATABASE TABLE

1. On Slide 31 of Ch. 6, the query 0 is ‘Retrieve the birth date and address of the employee(s) whose name is ‘John B. Smith’.
The instruction to answer this query just from the Employee table is:

```
   SELECT Bdate, Address
   FROM  EMPLOYEE
   WHERE  Fname ='John' AND Minit='B' AND
   Lname = 'Smith';
```

- Here is a query (Query 1) that requires more than one table in the database to answer and which requires a table join.

  Query 1: Get the name and address of all employees who work for the 'Research' department.

```
SELECT  Fname, Lname, Address
FROM    EMPLOYEE, DEPARTMENT
WHERE   Dname = 'Research'
AND Dnumber = Dno;
```

- In the solution to Query 1 above, in the WHERE clause, the Dname = ‘Research’ is a selection condition. The condition Dnumber = Dno is a join condition that combines two tuples from two tables when the value of the primary key

Department.Dnumber is equal to the value of the corresponding foreign key in the second table Employee.Dno.

2. When writing the SQL query, think in terms of three things:
i. The projection list (the result to be retrieved)

ii. The correct tables from the database to retrieve these results from.

iii. The Selection conditions, consisting of the join conditions that connect table attributes and other filtering conditions.

3. The Cross Product of two Tables to be joined.
A join of tables Employee and Department on a common Primary key/ foreign key combination (Department.Dnumber/Employee.Dno) as required on Query 1 on slide 31, comes as a result of the cross product of the two tables (Employee and Department), followed by a selection on the join attributes where
Department.Dnumber= Employee.Dno.

- What is a cross product of two tables?

A cross product of the Employee (with 8 tuples) and Department (with 3 tuples) written as Employee x Department, will get all possible combinations of each tuple of Employee with tuples of Department to yield a table with 8 x 3 = 24 tuples. This cross product table also has degree (that is, number of attributes in it) as 14, which is the number of attributes in Employee (10) + number of attributes in Department (4).

- The SQL query 10A on slide 37 gives you the cross product of the tables Employee and Department.

```
SELECT *
from   Employee, Department;
```

While the SQL query the query below gives the result of the same query when the join condition is applied to get only those combinations where Employee.Dno = Department.Dnumber.

```
SELECT *
```

```
from   Employee, Department
where  Employee.Dno = Department.Dnumber;
```

- Note that while the cross product of Employee and Department will retieve 24 tuples, the equi-join query on Employee and Department will retrieve exactly 8 tuples out of the 24 cross product tuples where there is a equivalent match for the primary key and foreign key attributes of the two tables.

- Next, if we discuss the midterm test 2 practice, we would place parts of the solution next in an updated version of this lecture notes.

- **** Parts of Solution to Midterm 2 Practice (handed out Fall 2020 Class)**

Section B (40 marks):
This section has 3 questions:

1. **(10 marks)**
Use the database table of Figure 1: STUDENT_ACTIVITY to answer this question. What functional dependencies FD can you see exist in this relation above? (ii) What update, (iii) delete and (iv) insertion anomalies may or not be present in this STUDENT_ACTIVITY relations above. Explain with examples using this database.

Figure 1: STUDENT_ACTIVITY table

SID	**Name**	**Club**	**Cost**	**AmtPaid**
100	Jones	Scuba	400	0
200	Chau	Scuba	400	400
200	Chau	Skiing	550	550

300	Garrett	Climbing	150	150
400	James	Skiing	550	550

Primary key of STUDENT_ACTIVITY table is (SID, Club)

FDs (2.5 marks)	FD1: (SID,Club) -> {Name, Cost, AmtPaid} FD2: SID -> Name FD3: Club -> Cost
Update anomalies (2.5 marks)	i. Update Anomalies: In STUDENT_ACTIVITY, the partial dependencies SID -> Name and Club -> Cost can cause update anomalies. For example, all STUDENT records whose SID is 200 (about 2 records) all have {Name} as {Chau} and {SID} is not the whole key. We cannot update any of these records (either SID or Name) to something else or the database integrity is violated. If we want to change values, we must remember these multiple column associations (e.g, (SID, Club) -> Name)) that indicate data redundancy and change them all in all the associated row records or there is a violation and this is update anomaly.
Delete anomalies (2.5 marks)	i. Delete Anomalies: In STUDENT_ACTIVITY, the partial dependencies SID -> Name and Club -> Cost can cause delete anomalies. For example, if a STUDENT temporarily has no ACTIVITY defined on it, his information (SID, Name) will not be represented in the database when his last ACTIVITY record is removed (deletion anomaly).
Insert anomalies (2.5 marks)	i. Insertion anomaly: In STUDENT_ACTIVITY, the partial dependencies SID -> Name and Club -> Cost can cause insert anomalies. A new STUDENT cannot be added unless he is involved in at least one Club (insertion anomaly). Inserting a new tuple relating an existing STUDENT to an existing CLUB requires checking both partial dependencies; for example, if a different value is entered for COST than those values in other tuples with the same value for CLUB, we get an insert anomaly.

2. **(15 marks)** Design a simple database with at least 5 relations for such applications as "students taking courses taught by faculty in class rooms at a specific time", "customers ordering items at specific dates and ordered items shipped from warehouse at a specific date".

a. Write all instructions to create your database tables with constraints (**5 marks**)

b. Write all instructions to load data into your database tables (5 marks)

c. State two SQL queries involving more than one table that can be posed on this database indicating the tables to visit to answer the queries each time and the results retrieved. (**5 marks**)

Note that I have already streamed lined the example database you can construct such as the quick student database example “students taking courses taught by faculty in class rooms at a specific time”. To design, you identify all the entities (e.g., student, courses, faculty, rooms) in that one sentence example that I gave. Then, you assign meaningful original or earned attributes (not derived such as score with letter grade together) for identifying them (e.g., student (sid, sname, major, gpa)). Secondly, you need to find relationships that connect these entities together. For example, from the sentence example in the question below, it is indicated that students take courses and a relationship that exists in this environment is Enrolled (sid, cid, grade). Through the foreign key attributes, you connect two entities using a relationship type. Thus, one normalized (3NF) database that I can design from the example “students taking courses taught by faculty in class rooms at a specific time” is given below:

Student(sid, sname, major, gpa)
Course(cid, ctitle, credit)
Faculty(fid, fname, salary)
Room(Rid, location, capacity)
Enrolled(sid, cid, grade)
Teach(fid, cid, Rid, time)

Question	Creating Tables
2a (5 marks)	a. Write all instructions to create your database tables with constraints. Solution to (a): You are here now required to specify all the SQL instructions for creating the 6 tables above such as: CREATE TABLE STUDENT (SID VARCHAR(10) NOT NULL, Sname VARCHAR(15), Major VARCHAR(15), gpa NUMBER(6, 2), PRIMARY KEY(Sid)); CREATE TABLE COURSE(CID VARCHAR(10) NOT NULL, Ctitle VARCHAR(15), Credit NUMBER(1), PRIMARY KEY(CID)); CREATE TABLE FACULTY(FID VARCHAR(10) NOT NULL, Fname VARCHAR(15), Salary NUMBER(8,2), PRIMARY KEY(FID));

	CREATE TABLE ROOM(RID VARCHAR(10) NOT NULL, location VARCHAR(15), Capacity NUMBER(4), PRIMARY KEY(RID)); CREATE TABLE ENROLLED (SID VARCHAR(10) NOT NULL, CID VARCHAR(10) NOT NULL, GRADE NUMBER(4,2), PRIMARY KEY(Sid, Cid), FOREIGN KEY (Sid) REFERENCES STUDENT(Sid), FOREIGN KEY (Cid) REFERENCES COURSE(Cid)); CREATE TABLE TEACH (FID VARCHAR(10) NOT NULL, CID VARCHAR(10) NOT NULL, RID VARCHAR(10) NOT NULL, TIME VARCHAR(8), PRIMARY KEY(Fid, Cid, Rid), FOREIGN KEY (Fid) REFERENCES FACULTY(Fid), FOREIGN KEY (Cid) REFERENCES COURSE(Cid), FOREIGN KEY (rid) REFERENCES ROOM(Rid)); COMMIT;

Question	Loading data into tables

2b
(5 marks)
	b.Write all instructions to load data into your database tables (5 marks)
	Solution (b) (For just inserting data into table Student, Course, and Enrolled)
	
	INSERT INTO STUDENT
	VALUES ('1111', ‘John Smith’, 'CS', 85.34);
	COMMIT;
	
	INSERT INTO STUDENT
	VALUES ('2222', ‘Mary Pert’, 'Math', 80.25);
	COMMIT;
	
	INSERT INTO COURSE
	VALUES ('60-212', ‘Java’, 3);
	COMMIT;
	
	INSERT INTO COURSE
	VALUES ('60-315', ‘Database’, 3);
	COMMIT;

	INSERT INTO ENROLLED
	VALUES (‘1111’, ’60-212’, 78.50);
	COMMIT;
	
	INSERT INTO ENROLLED
	VALUES (‘1111’, ’60-315’, 81.45);
	COMMIT;
	
	INSERT INTO ENROLLED
	VALUES (‘2222’, ’60-212’, 90.10);
	COMMIT;
	
	INSERT INTO ENROLLED
	VALUES (‘2222’, ’60-315’, 75.00);
	COMMIT;

Question	Query	Tables needed to answer query and Results
2c (5 marks)	c. State two SQL queries involving more than one table that can be posed on this database indicating the tables to visit to answer the queries each time and the results retrieved. (**5 marks**) Solution (c) (i) Get the average gpa maintained by each student name Select Student.Sid, Student.Sname, avg(Student.gpa) From Enrolled, Student Where student.sid = Enrolled.sid Group by sid; (ii) Print all courses (titles) taken by student ‘Smith’ Select Course.cid, Course.ctitle From Enrolled, Student, Course Where Student.sid = Enrolled.sid And Course.cid = Enrolled.cid And Student.sname LIKE ‘%Smith’;	i. Enrolled, Student ii. Enrolled, Student, Course

3. (15 marks)

a. Design a simple normalized Library database for tracking “customers reserve books” with 3 or more tables and answer the following questions about your database.

(5 marks)

b. Discuss how your database is in third normal form showing all functional dependencies (FDs).

(4 marks)

c. Create an instance of your database showing the instructions for that. (**4 marks**)

d. Provide 2 SQL queries and their English versions on your database with retrieved results.

(2 marks)

<table>
<tr><td>Question</td><td>Answers</td></tr>
<tr><td>a.
(5 marks)</td><td>CUSTOMER (Cid, name, email, numbooks)
BOOK(Bid, Title, author, callnum, location, numpages)
RESERVES(Cid, Bid, numdays, Resdate, Pickdate, Retdate)</td></tr>
<tr><td>b.
(4 marks)</td><td>A database is in third normal form if all the tables in the database schema are in 3NF. A database table is in 3NF if every non-key attribute is functionally determined by only the primary key. This means that there is no transitivity in the functional dependency between an attribute of the table and the primary key.
Looking at the 3 tables in the database schema above,
Reserves has primary key as: (cid, bid) with FDs as: (cid, bid) → {numedays, resdate, pickdate, Retdate}.
Customer has primary key as: (cid) with FDs as: (cid) → {name, email, numbooks}.
Book has primary key as: (bid) with FDs as: (bid) → {title, author, callnum, location, numpages}</td></tr>
<tr><td>c.

(4 marks)</td><td>Customer
<table>
<tr><td>cid</td><td>name</td><td>email</td><td>numbooks</td></tr>
<tr><td>1</td><td>John Smith</td><td>jsmith@uw.ca</td><td>2</td></tr>
<tr><td>2</td><td>Mary Hat</td><td>mhat@uw.ca</td><td>1</td></tr>
</table>
Book
<table>
<tr><td>Bid</td><td>Title</td><td>author</td><td>callnum</td><td>location</td><td>numpages</td></tr>
<tr><td>315</td><td>Databases</td><td>Elmasri</td><td>LDF1S14</td><td>LDW</td><td>400</td></tr>
<tr><td>212</td><td>Java</td><td>Dietel</td><td>LDF1S14</td><td>LDW</td><td>300</td></tr>
</table>
Reserves
<table>
<tr><td>Cid</td><td>Bid</td><td>numdays</td><td>Resdate</td><td>Pickdate</td><td>Retdate</td></tr>
<tr><td>1</td><td>315</td><td>1</td><td>15-feb-17</td><td>20-feb-17</td><td>21-feb-17</td></tr>
<tr><td>2</td><td>315</td><td>2</td><td>05-feb-17</td><td>05-feb-17</td><td>07-feb-17</td></tr>
<tr><td>1</td><td>212</td><td>1</td><td>20-feb-17</td><td>20-feb-17</td><td>21-feb-17</td></tr>
</table></td></tr>
<tr><td>d.
(2 marks)</td><td>Print all book (title, cname) that each customer has reserved.

SELECT B.title, C.Cname
FROM CUSTOMER C, BOOK B, RESERVE R</td></tr>
</table>

	WHERE C.Cid=R.Cid AND B.Bid = R.Bid
	Group by B.title, C.name;