Assignment 1

Chapter10:

_____ 1. A transaction is a _____ unit of work that must be either entirely completed or aborted.
a. Timed
b. Practical
c. logical
d. physical

_____ 2. A consistent database is _____.
a. One in which all tables have foreign keys
b. One in which all data integrity constraints are satisfied
c. One in which all tables are normalized
d. One in which all SQL statements only update one table at a time

_____ 3. _____ requires that all operations of a transaction be completed.
a. Specificity
b. Atomicity
c. Durability
d. Time stamping

_____ 4. _____ means that data used during the execution of a transaction cannot be used by a second transaction until the first one is completed.
a. Serializability
b. Atomicity
c. Isolation
d. Time stamping

_____ 5. All transactions must display _____.
a. atomicity, consistency, and durability
b. durability and isolation
c. consistency, durability, and isolation
d. atomicity, durability, consistency, and isolation

_____ 6. A single-user database system automatically ensures _____ of the database, because only one transaction is executed at a time.
a. serializability and durability
b. atomicity and isolation
c. serializability and isolation
d. atomicity and serializability

_____ 7. The ANSI has defined standards that govern SQL database transactions. Transaction support is provided by two SQL statements: _____ and ROLLBACK.
a. RETRIEVE
b. ASSIGN
c. UPDATE
d. COMMIT

_____ 8. ANSI defines four events that signal the end of a transaction. Of the following events, which is defined by ANSI as being equivalent to a COMMIT?
a. Five SQL statements are executed.
b. The end of a program is successfully reached.
c. The program is abnormally terminated.
d. The database is shut down for maintenance.

_____ 9. ANSI defines four events that signal the end of a transaction. Of the following events, which is defined by ANSI as being equivalent to a ROLLBACK?
a. Five SQL statements are executed.
b. The end of a program is successfully reached.
c. The program is abnormally terminated.
d. The database is shut down for maintenance.

_____ 10. The implicit beginning of a transaction is _____.
a. When the database is started
b. When a table is accessed for the first time
c. When the first SQL statement is encountered
d. When the COMMIT command is issued

_____ 11. The information stored in the _____ is used by the DBMS for a recovery requirement triggered by a ROLLBACK statement, a program's abnormal termination, or a system failure such as a network discrepancy or a disk crash.
a. data dictionary
b. metadata
c. rollback manager
d. transaction log

_____ 12. One of the three most common data integrity and consistency problems is _____.
a. lost updates
b. disk failures
c. user errors
d. deadlocks

_____ 13. As long as two transactions, T1 and T2, access _____ data, there is no conflict, and the order of execution is irrelevant to the final outcome.
a. shared
b. common
c. unrelated
d. locked

_____ 14. A _____ lock prevents the use of any tables in the database from one transaction while another transaction is being processed.
a. database-level
b. table-level
c. page-level
d. row-level

_____ 15. _____ are required to prevent another transaction from reading inconsistent data.
a. Locks
b. Schedules
c. Stamps
d. Logs

_____ 16. The _____ manager is responsible for assigning and policing the locks used by the transactions.
a. transaction
b. database
c. lock
d. schedule

_____ 17. Lock _____ indicates the level of lock use.
a. granularity
b. shrinking
c. growing
d. serializability

_____ 18. A _____ lock locks the entire table preventing access to any row by a transaction while another transaction is using the table.
a. database-level
b. table-level
c. page-level
d. row-level

_____ 19. A _____ lock locks the entire diskpage.
a. transaction-level
b. table-level
c. page-level
d. row-level

_____ 20. A _____ lock allows concurrent transactions to access different rows of the same table.
a. database-level
b. table-level
c. page-level
d. row-level

Deliverables

Part 1

Distributed Version Control Setup:

Carefully go through the following steps: http://www.qcitr.com/usefullinks.htm#lesson3b

Part 2

Development Installations and Login Procedures:

1. **AMPPS** (only if you don't already have it)**:** (http://www.ampps.com/)
   - Tour: http://www.ampps.com/tour
   - Download and Installation: http://www.ampps.com/wiki/Main_Page
   - Installation Demo: http://www.ampps.com/demo

   NOTE: Mac Users with Monterey or higher, See Mac Installation Instructions!

2. **MySQL Workbench** (only if you don't already have it):
   https://dev.mysql.com/downloads/workbench/
   Note: *Be sure* to download MySQL Workbench, *NOT* MySQL (DBMS)!
   AMPPS MySQL (**local**) login information:
   a. **user:** root
   b. **password:** mysql

3. **MySQL Workbench and SSH (remote) Login Procedures (CCI Server):**

- FSU_CCI_MySQL_Workbench_Login.pdf
- **Video:** http://qcitr.com/vids/MySQL_Workbench_Login.mp4
- FSU_CCI_MySQL_SSH_Login_PC_PuTTY.pdf
- FSU_CCI_MySQL_SSH_Login_MAC.pdf

PuTTY Helper Videos (Windows):

- http://qcitr.com/vids/PuTTY_Configuration.mp4
- http://qcitr.com/vids/PuTTY_PSFTP.mp4

4. **Entity Relationship Diagram:**
   Helper video: (**Note:** use above MySQL Workbench connection parameters shown in video--*not* those shown here.) http://www.qcitr.com/vids/LIS3781_A1_and_Creating_ERDs.mp4

5. Push your local repository to the one hosted by Bitbucket's servers: See **Part 1** (above).

6. Provide me with **read-only** access to Bitbucket repository: See **Part 1** (above).

README.md file should include the following items:

1. Screenshot of **ampps** installation running (#1 above);
2. git commands w/short descriptions ("Lesson 3b - Version Control Systems: Course Configuration");
3. ***Your*** ERD image
4. Bitbucket repo links:
   a. This assignment, and
   b. The completed tutorial repo above (**bitbucketstationlocations**).
   (See link in screenshot below.)

Deliverables (see screenshots below):

1. Provide **Bitbucket** read-only access to **lis3781** repo, include links to the repo (**BitbucketStationLocations**) you created in the above tutorials in **README.md**, using Markdown syntax, (**README.md** must also include screenshots per above.)
   (**DO NOT create README OR .gitignore in Bitbucket**—ALWAYS do so locally, then push them to Bitbucket.)
2. **FSU's Learning Management System**: include lis3781 **Bitbucket** repo link

Part 3

MySQL Server

Business Rules
The human resource (HR) department of the ACME company wants to contract a database modeler/designer to collect the following employee data for tax purposes: job description, length of employment, benefits, number of dependents and their relationships, DOB of both the employee and any respective dependents. In addition, employees' histories must be tracked. Also, include the following business rules:

- Each employee may have one or more dependents.
- Each employee has only one job.
- Each job can be held by many employees.
- Many employees may receive many benefits.
- Many benefits may be selected by many employees (though, while they may not select any benefits—any dependents of employees may be on an employee's plan).

Notes:
- **Employee**/**Dependent** tables must use suitable attributes (See Assignment Guidelines);

In Addition:
- **Employee**: SSN, DOB, start/end dates, salary;
- **Dependent**: same information as their associated employee (though, not start/end dates), date added (as dependent), type of relationship: e.g., father, mother, etc.
- **Job**: title (e.g., secretary, service tech., manager, cashier, janitor, IT, etc.)
- **Benefit**: name (e.g., medical, dental, long-term disability, 401k, term life insurance, etc.)
- **Plan**: type (single, spouse, family), cost, election date (plans must be unique)
- **Employee history**: jobs, salaries, and benefit changes, as well as who made the change and why;
- Zero Filled data: SSN, zip codes (not phone numbers: US area codes not below 201, NJ);
- *All* tables must include notes attribute.

Design Considerations: Generally, avoid using flag values (e.g., yes/no) for status—unless, applicable. Instead, use dates when appropriate, as date values provide more information, and sometimes, can be used when a flag value would be used. For example, "null" values in employees' termination dates would indicate "active" employees.

In addition, for federal, state, and local mandates, most HR systems require extensive history-keeping. That is, virtually every change made to an employee record needs to be logged in a history table(s)—here, we are keeping the design simple. Also, for reporting (and design) purposes, all *current* data should be kept in the primary table (employee). Every time an employee's data changes, it should be logged in the history table, including when, why, and who made the change—crucial for reporting purposes!

ERD (*Must forward-engineer--otherwise, no points will be awarded. ***):**
- Include at least 5 "**unique**" records per table.
- **Must match** data types

Save as lis3781_a1_solutions.sql (requires exporting: optional)

No Credit will be given if not forward-engineered to the CCI server, including data.

SQL Statements for A1
(*__Be sure__* to review the "__SQL Statements__" tutorial in Database Resources.)

(Must populate ***both*** local **and** remote (CCI Server) MySQL databases.)

The following items are *required* (use PuTTY or Terminal):

A. Necessary SQL statements
B. ***Your*** query resultsets
C. **Formatting should display the query resultsets as indicated in the provided examples (below)**

1) Backward-engineer the following query resultset:
   a. list (current) job title each employee has,
   b. include name,
   c. address,
   d. phone,
   e. SSN,
   f. order by last name in descending order,
   g. **use old-style join**.

emp_id	emp_fname	emp_lname	address	phone_num	emp_ssn	job_title
4	Robert	Laurie	16234 Washington Pl, Panama City Beach, FL 03145-6759	(850)567-9210	087-64-3182	Manager
5	Kelsey	Hawks	511 Woldorf Ave, New York, NY 02194-5189	(201)511-9267	078-61-9456	secretary
1	Marsha	Fromm	123 Elm St., Chicago, IL 06060-3156	(781)254-1976	011-12-2333	Cashier
2	Steve	Crenshaw	3789 Golf View, Phoenix, AZ 08500-6919	(480)732-8421	022-21-1444	Service Tech
3	Kathy	Camerie	8956 Leisure Dr, Anchorage, AK 09950-3634	(907)135-4985	055-57-6234	Service Tech

2) List all job titles and salaries each employee HAS and HAD, include employee ID, full name, job ID, job title, salaries, and respective dates, sort by employee id and date, **use old-style join**.

emp_id	emp_fname	emp_lname	eht_date	eht_job_id	job_title	eht_emp_salary	eht_notes
1	Marsha	Fromm	2001-03-19 09:30:00	1	secretary	50000.00	NULL
1	Marsha	Fromm	2007-08-03 16:00:00	9	IT	75000.00	NULL
1	Marsha	Fromm	2016-04-30 08:30:00	6	CEO	450000.00	NULL
1	Marsha	Fromm	2017-05-19 11:00:00	9	IT	35000.00	NULL
2	Steve	Crenshaw	2003-05-10 10:45:00	2	Service Tech	60000.00	NULL
2	Steve	Crenshaw	2013-10-31 05:30:00	8	Security	55000.00	NULL
2	Steve	Crenshaw	2016-12-24 08:45:00	8	Security	80000.00	NULL
3	Kathy	Camerie	2004-07-30 04:00:00	2	Service Tech	45000.00	NULL
3	Kathy	Camerie	2005-07-18 12:00:00	5	Stock	70000.00	NULL
3	Kathy	Camerie	2017-05-22 21:04:10	7	Janitor	57000.00	current_timestamp example
4	Robert	Laurie	2009-11-11 14:30:00	3	Manager	80000.00	NULL
4	Robert	Laurie	2015-01-01 00:00:00	7	Janitor	85000.00	NULL
5	Kelsey	Hawks	2010-02-27 15:40:00	4	Cashier	90000.00	NULL
5	Kelsey	Hawks	2011-08-31 09:00:00	1	secretary	65000.00	NULL

3) List employee and dependent full names, DOBs, relationships, and ages of both employee and respective dependent(s), sort by employee last name in ascending order, **use natural join**:

emp_fname	emp_lname	emp_dob	emp_age	dep_fname	dep_lname	dep_relation	dep_dob	dep_age
Kathy	Camerie	1985-10-24	31	Stephen	Banks	son	2002-03-19	15
Kathy	Camerie	1985-10-24	31	Bobby	Sue	daughter	1990-10-28	26
Steve	Crenshaw	1952-07-30	64	Marilyn	Monroe	grandmother	1926-05-31	90
Marsha	Fromm	1978-03-09	39	Billy	Bob	husband	1964-12-14	52
Robert	Laurie	1975-06-15	41	Krista	Kling	niece	2005-08-02	11

4) Create a transaction that updates job ID 1 to the following title "**owner**," w/o the quotation marks, display the job records before and after the change, inside the transaction:

Before change:

job_id	job_title	job_notes
1	secretary	NULL
2	Service Tech	NULL
3	Manager	NULL
4	Cashier	NULL
5	Stock	NULL
6	CEO	NULL
7	Janitor	NULL
8	Security	NULL
9	IT	NULL

After change:

job_id	job_title	job_notes
1	owner	NULL
2	Service Tech	NULL
3	Manager	NULL
4	Cashier	NULL
5	Stock	NULL
6	CEO	NULL
7	Janitor	NULL
8	Security	NULL
9	IT	NULL

5) Create a stored procedure that adds one record to the benefit table with the following values: benefit name "**new benefit**," benefit notes "**testing**," both attribute values w/o the quotation marks, display the benefit records before and after the change, inside the stored procedure:

Before change:

ben_id	ben_name	ben_notes
1	medical	NULL
2	dental	NULL
3	long-term disability	NULL
4	401k	NULL
5	term life insurance	NULL
6	vision	NULL

After change:

ben_id	ben_name	ben_notes
1	medical	NULL
2	dental	NULL
3	long-term disability	NULL
4	401k	NULL
5	term life insurance	NULL
6	vision	NULL
7	new benefit	testing

6) List employees' and dependents' names and social security numbers, also include employees' e-mail addresses, dependents' mailing addresses, and dependents' phone numbers. *MUST* display *ALL* employee data, even where there are no associated dependent values. (Major table: all rows displayed, minor table: display null values.)

employee	emp_ssn	email	dependent	dep_ssn	address	phone_num
Camerie, Kathy	055-57-6234	kcamerie@fsu.edu	Sue, Bobby	055-78-9041	531 Hounds Tooth Lane, Anchorage, CA 09001-9285	(818)538-2916
Camerie, Kathy	055-57-6234	kcamerie@fsu.edu	Banks, Stephen	082-37-5184	8189 Estiva Ave, Anchorage, IL 04178-9265	(616)423-8257
Crenshaw, Steve	022-21-1444	screnshaw@aol.com	Monroe, Marilyn	059-81-1654	5916 Velcro Rd, Phoenix, WY 07321-6924	(717)851-5798
Fromm, Marsha	011-12-2333	mfromm@aol.com	Bob, Billy	041-25-5789	15789 Oak Ave, Chicago, FL 00000-3568	(850)132-4567
Hawks, Kelsey	078-61-9456	khawks@fsu.edu	NULL	NULL	NULL	NULL
Laurie, Robert	087-64-3182	rlaurie@hotmail.com	Kling, Krista	024-78-1683	91584 Sesame Blvd, Panama City Beach, MI 04967-1348	(313)756-4829

7) Create "after insert on employee" trigger that automatically creates an audit record in the emp_hist table.

> "**NOTE:** A README.md file should be placed at the **root of each of your repos directories.** "

LIS3781 - Advanced Database Management

Mark K. Jowett, Ph.D.

LIS3781 Requirements:

Course Work Links:

1. A1 README.md
    - Install AMPPS
    - Provide screenshots of installations
    - Create Bitbucket repo
    - Complete Bitbucket tutorial (bitbucketstationlocations)
    - Provide git command descriptions
2. A2 README.md
    - A2 Bitbucket requirements will be put w/in A1
3. A3 README.md
    - TBD
4. A4 README.md
    - TBD
5. A5 README.md
    - TBD
6. P1 README.md
    - TBD
7. P2 README.md
    - TBD

Tables: Add table: use three or more hyphens (---) to create each column's header, and use pipes (|) to separate each column. (Optionally add pipes on either end of the table.)

```
| Syntax      | Description |
| ----------- | ----------- |
| Header      | Title       |
| Paragraph   | Text        |
```

Alignment: Align text: left, right, or center by adding a colon (:) to left, right, or on both sides of hyphens within the header row.

```
| Syntax      | Description |   Example   |
| :---        |    :----:   |         ---: |
| left      | center       | right   |
```

Markdown Tables

Mark Jowett / lis3781

README.md

Pull requests | Check out

LIS3781 Advanced Database Management

Source | master | 759f6ae | Full commit

lis3781 / a1 / README.md | Edit

"**NOTE:** This README.md file should be placed at the **root of each of your repos directories.**

Also, this file **must** use Markdown syntax, and provide project documentation as per below--otherwise, points **will** be deducted. "

LIS3781 - Advanced Database Management

Mark K. Jowett, Ph.D.

Assignment 1 Requirements:

Five Parts:

1. Distributed Version Control with Git and Bitbucket
2. AMPPS Installation
3. Questions
4. Entity Relationship Diagram, and SQL Code (optional)
5. Bitbucket repo links:
   a) this assignment and
   b) the completed tutorial (bitbucketstationlocations).

"A1 Database Business Rules:

The human resource (HR) department of the ACME company wants to contract a database modeler/designer to collect the following employee data for tax purposes: job description, length of employment, benefits, number of dependents and their relationships, DOB of both the employee and any respective dependents. In addition, employees' histories must be tracked. Also, include the following business rules:

- Each employee may have one or more dependents.
- Each employee has only one job.
- Each job can be held by many employees.
- Many employees may receive many benefits.
- Many benefits may be selected by many employees (though, while they may not select any benefits—any dependents of employees may be on an employee's plan).
  Notes:
- Employee/Dependent tables must use suitable attributes (See Assignment
  Guidelines);

In Addition:

- Employee: SSN, DOB, start/end dates, salary;
- Dependent: same information as their associated employee (though, not start/end dates), date added (as dependent), type of relationship: e.g., father, mother, etc.
- Job: title (e.g., secretary, service tech., manager, cashier, janitor, IT, etc.)
- Benefit: name (e.g., medical, dental, long-term disability, 401k, term life insurance, etc.)
- Plan: type (single, spouse, family), cost, election date (plans must be unique)
- Employee history: jobs, salaries, and benefit changes, as well as who made the change and why;
- Zero Filled data: SSN, zip codes (not phone numbers: US area codes not below 201, NJ);
- *All* tables must include notes attribute.

README.md file should include the following items:

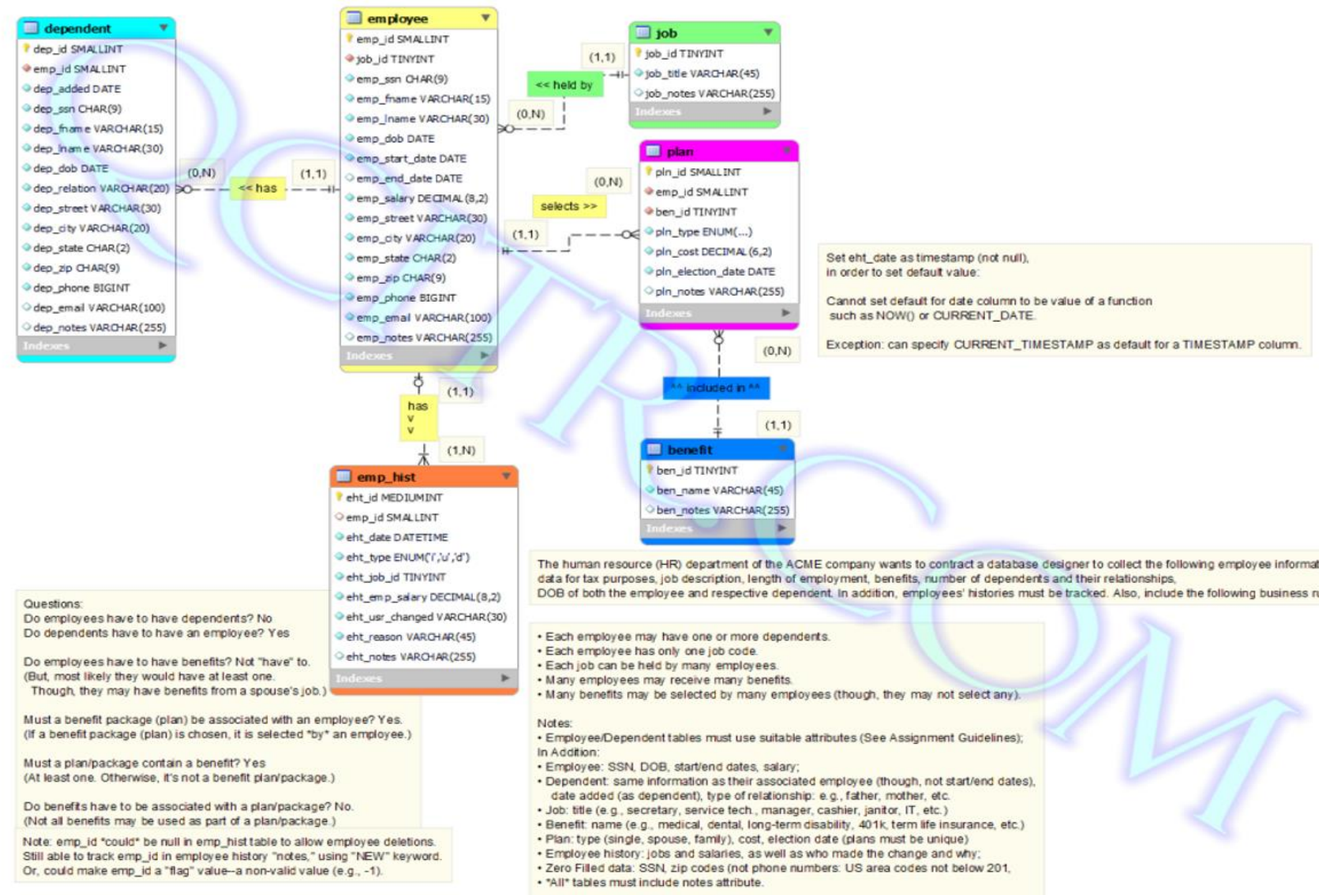
- Screenshot of A1 ERD
- Ex1. SQL Solution
- git commands w/short descriptions

> **Git commands w/short descriptions:**

1. git init - definition goes here...
2. git status
3. git add
4. git commit
5. git push
6. git pull
7. One additional git command

Assignment Screenshots:

Screenshot of A1 ERD



Screenshot of A1 Ex1

```
-- query resultset example:
+--------+-----------+-----------+--------------------+------------------+-----------+-----------+-----------+-----------+--------------+
| emp_id | emp_fname | emp_lname | emp_street         | emp_city         | emp_state | emp_zip   | emp_phone | emp_ssn   | job_title    |
+--------+-----------+-----------+--------------------+------------------+-----------+-----------+-----------+-----------+--------------+
|      4 | Robert    | Laurie    | 16234 Washington Pl | Panama City Beach | FL        | 000067598 | 8505679210 | 876431765 | Manager      |
|      5 | Kelsey    | Hawks     | 511 Woldorf Ave    | New York         | NY        | 000451892 | 2015119267 | 000094562 | secretary    |
|      1 | Marsha    | Fromm     | 123 Elm St.        | Chicago          | IL        | 000031567 | 7812541976 | 000223333 | Cashier      |
|      2 | Steve     | Crenshaw  | 3789 Golf View     | Phoenix          | AZ        | 850069191 | 4807328421 | 222114444 | Service Tech |
|      3 | Kathy     | Camerie   | 8956 Leisure Dr    | Anchorage        | AK        | 005036341 | 9071354985 | 001762345 | Service Tech |
+--------+-----------+-----------+--------------------+------------------+-----------+-----------+-----------+-----------+--------------+

-- ANSWER: same as above with formatting
-- old-style join
select emp_id, emp_fname, emp_lname,
  CONCAT(emp_street, ", ", emp_city, ", ", emp_state, " ", substring(emp_zip,1,5), '-', substring(emp_zip,6,4)) as address,
  CONCAT('(', substring(emp_phone,1,3), ')',substring(emp_phone,4,3), '-', substring(emp_phone,7,4)) as phone_num,
  CONCAT(substring(emp_ssn,1,3), '-', substring(emp_ssn,4,2), '-', substring(emp_ssn,6,4)) as emp_ssn, job_title
from job as j, employee as e
where j.job_id = e.job_id
order by emp_lname desc;
```

Tutorial Links:

Bitbucket Tutorial - Station Locations:
A1 Bitbucket Station Locations Tutorial Link