Due Date: Wednesday Nov, 08, 2023 (11:59PM)

Important Dates:

- ~~**Phase I due Oct 09**~~
- Phase II due Nov 08
- Phase III due Dec 08
  - [*group submission, one per group*] Presentations Dec 08 (11:59PM)
  - [*group submission, one per group*] Complete Project Documentation
  - [*individual*] Self-Reflection Document (10 marks)
  - [*individual*] Individual Contribution Form for Group Project (5 marks)*
  - [*individual*] Evaluating Group Presentations (10 marks)** Dec 11 (11:59PM)

Overview (80 marks)

In Phase II, you will begin designing and constructing the data store for your application. Use the proposed application area from Phase I while considering any feedback you may have received on the proposal to **define** and **construct** the database for your project. You may use any database environment you like (i.e. MySQL, Microsoft SQL, Oracle, IBM DB2, etc…). The following elements are the required deliverables for Phase II. It is important to specify the DBMS that is being used to execute the SQL statements for Parts A, B and C. Part A requires DDL **statements** as well as an **illustrative schema diagram** similar in style/format to the diagram provided in Appendix B for COMPANY database).

Deliverable *(submit electronically via CANVAS: drop box Phase II):*

Submit a **SQL script** file (with .**sql** extension) that contains all the SQL statements required for Parts A, B and C. For the illustrative diagram of Part A, submit a PDF file containing schema diagram in addition to the SQL script file. The SQL script will be used to test all SQL statements submitted and **need be verified prior to submission using a proper DBMS environment (e.g. test your SQL script)**. Hence, it is necessary to include the DBMS that is used to test the SQL statements as well as any supporting documentation. To document line(s) in SQL, you can use (a) single line comments (which start with `--`) or (b) multi-line comments (which start with `/*` and end with `*/`). Appendix A provides an example of a SQL script. Your SQL script must be readable and formatted with proper indentation and lineation (***5 marks***).

Part A: Relational Schema (25 points)

You are required to create the corresponding database schema (relations, attributes, constraints). Write all required SQL DDL statements for (a) creating the database relations (see **Figure 4.1 from the textbook**) and (b) an **illustrative diagram** of the relational database schema (see **Figure 3.7 from the textbook**). Your schema should clearly specify all necessary integrity constraints (e.g. PRIMARY KEY, FOREIGN KEYS, NOT NULL, etc.) for constructing your database relations. A partial requirement for this database schema is to also include at least:

- **at four** CHECK constraints,
- **four** attributes with initial default values, and
- **all** proper ON DELETE or ON UPDATE clauses (having proper options) associated with all of the defined foreign keys.

Appendix B (page 4 of this handout) provides an example of the COMPANY database schema (Figure 3.7).

Part B: Sample Data (15 points)

After creating the necessary relations in Part A, create the relation states (i.e. populate your database with sample data). The sample data should reflect a real world database application. You are encouraged to utilize existing data or datasets that you may identify as suitable for your application. However, you can also generate your own data that is required for the database of your application. You may use online tools for generating the necessary data for your database. You will use this data later to run SQL queries. Each relation should contain **at least 10 tuples**. Make sure that the populated data is suitable for the types of queries in Part C. Provide all the necessary INSERT INTO statements for all sample data in this section (group them by each table and provide necessary documentation).

Part C: Designing SQL Queries (35 points)

Once you have inserted the sample data into the database, you are required to design and construct **ten non-trivial** SQL queries that are useful to your database application. Below are the requirements for designing the first five queries that will be useful to your database application. As for the remaining five queries, you must design and construct the SQL queries such that they are useful and pertinent to your database application. Ensure that proper documentation is provided for each query including a query (a) **number**, (b) **purpose** and (c) **summary** of the **expected result**. A sample is provided in Appendix A.

- SQL Query 1: Computes a **join** of at least three tables (must use **JOIN ON**)
- SQL Query 2: Uses **nested queries** with the **IN**, **ANY** or **ALL** operator and uses a **GROUP BY** clause
- SQL Query 2: A **correlated nested** query with proper aliasing applied
- SQL Query 4: Uses a **FULL OUTER JOIN**
- SQL Query 5: Uses **nested queries** with any of the **set operations** UNION, EXCEPT, or INTERSECT*
- SQL Query 6: Create your own **non-trivial** SQL query (must use at least two tables in FROM clause)
- SQL Query 7: Create your own **non-trivial** SQL query (must use at least two tables in FROM clause)
- SQL Query 8: Create your own **non-trivial** SQL query (must use at least two tables in FROM clause)
- SQL Query 9: Create your own **non-trivial** SQL query (must use at least three tables in FROM clause)
- SQL Query 10: Create your own **non-trivial** SQL query
  - must use at least three tables in FROM clause
  - must use aliasing or renaming for at least once throughout SQL query

The grading for the design of the SQL queries will (*primarily*) be based on (a) complexity, (b) usefulness to the application, (c) creativity and (d) grammatical structure/syntax, (e) use of relational model concepts (e.g. foreign keys, primary key, default, NOT NULL, filtering of operations, clarity, etc.).

* Some DBMS may vary in terms of supporting these SET operations (e.g. different names). Depending on the DBMS, identify the type of operation is supported prior to executing the query. The following link provides the database SQL reference for Oracle 10g DBMS. https://docs.oracle.com/cd/B19306_01/server.102/b14200/queries004.htm

Options for running DBMS: Nearly all DBMSs enable you to import a SQL script with a .sql extension. For example if you create a SQL script in Oracle LiveSQL and you wish to test/run it on a different environment (e.g. Microsoft SQL Server, MySQL or PostgreSQL), you can import the script and run on this new platform. There are many options for building databases including, but not limited to the following options listed below. However, you are not required to use any specific platform for building your database. You may choose any platform that helps develop a running database.

- **online** tools such as Oracle LiveSQL (which utilizes Oracle Cloud Platform to run Oracle 10g DBMS) or using the GCP and LAMP Virtual Machine (VM) to use MySQL
- **local** deployment of the database through running DBMS on a local machine
  - Microsoft Visual Studio Community Edition (free) contains SQL Server Express (DBMS)
  - MySQL DBMS maintained by Oracle (runs on Linux, Mac OS and Windows)
  - MariaDB (fork of MySQL (open-source DBMS)
- web software **stacks**: these are web software development stacks that include web server packages, DBMS, support for scripting languages, etc.
  - XAMPP (Apache HTTP Server, MariaDB DBMS, PHP and Perl) maintained by Apache (runs on Linux, Mac OS and Windows)
  - LAMP (Linux, Apache, MySQL PHP/Perl/Python) → simple interface via Bitnami (runs on Linux, Mac OS and Windows)
  - WAMP (Windows, Apache, MySQL and PHP) → simple interface via Bitnami (runs on Linux, Mac OS and Windows)
  - among many others → take a look at this list: https://bitnami.com/stacks/infrastructure
- **Cloud** Platform: You may run virtual machine that are pre-built with necessary software such as MySQL DBMS for running databases. We will use GCP in the upcoming weeks

Appendix A: Sample SQL Script

```
-- Start of Sample SQL Script
/*  ******************************
    Project Phase II
    Group 100 (LiveSQL)
    This SQL Script was tested on
    Oracle LiveSQL. To run, simply
    load this script file and run.
    ******************************
*/
-- ***************************
-- Part A
-- ***************************
-- TABLE_A: store data about A
CREATE TABLE_A ( ... PRIMARY KEY ...);
-- TABLE_B: store data about B
CREATE TABLE_A ( ... CHECK ...);
ALTER TABLE ...
...
-- ***************************
-- ***************************
-- Part B
-- ***************************
-- Sample data for Table_A
-- Summary: store data about A
INSERT INTO Table_A VALUES (...);
INSERT INTO Table_A VALUES (...);
...
-- Sample data for Table_B
-- Summary: store data about B
INSERT INTO Table_B VALUES (...);
INSERT INTO Table_B VALUES (...);
....
-- ***************************
-- ***************************
-- Part C
-- ***************************
-- Query 1
-- Purpose: determines the average
-- salary of engineers in TCSS Inc.
-- Expected: a table containing details for every engineer
-- in TCSS Inc. including difference of an
-- employee's salary from average salary
SELECT A, B, C from A, B, C WHERE ...
-- ***************************
-- Query 2
-- Purpose: ...
-- Expected: ...
-- ...
-- ...
-- End of Script (Nov 2, 2023)
```

Appendix B: Sample Schema Diagram for COMPANY Database

You may use any drawing software of for developing a schema diagram but the diagram need to have the same notation as the one that appears in the textbook (e.g. sample figure below)

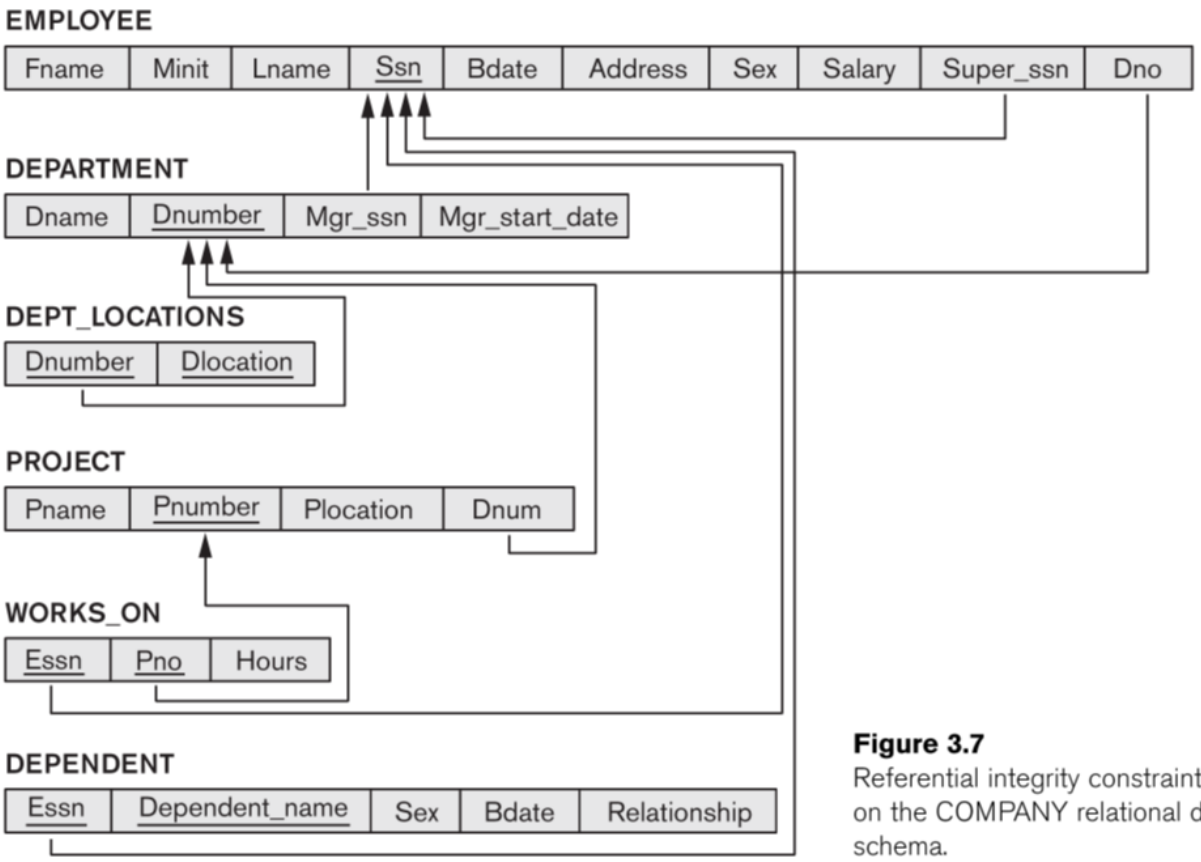

Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.