CSCI 4125/5125 Course Project

James Wagner

Spring 2021

1 Project Description

Introduction. New Orleans was ranked #1 in population growth of traditional cities in 2016. On the other hand, the percentage of IT professionals within Louisiana's workforce is only 1.05%, which is 36% lower than the national average, and 93%, 121%, 195% lower than Alabama, Florida, and Texas, respectively. Louisiana deserves to claim more IT professional jobs by creating and enhancing programs that train the workforce for coding professional IT jobs. The jobs requiring coding skills are projected to grow 12% faster than the job market overall in the next 10 years (http://www.bls.gov/emp/). The wages associated with each role type correspond with the intensity of programming skill requirements.

Objective. Your central task for this project is to build a database that supports the labor force development system (LD) including training programs, job assistance services, career planning, and company recruitment. If your design is good and if Louisiana wants to keep the crown of workforce training in state, then the Louisiana Department of Labor should utilize your program. By feeding real-world data into your project, the outcome will also be your own job hunting tool. The LD system collects data from companies, the job market, and other resources. Your project will also include two additional databases: one managing the data of a retail chain called Acar Zone (AZ) and another managing the data of a large manufacturing company called General Vehicle (GZ).

2 Database Modeling and Creation

A focus of your project design is the database schemas. In each of your three databases, you should consider the nature of the data and the queries to process. For example, “Company” in “JOB(JobCode, PositionCode, Type, PayRate, PayType, Company)” means the identifier of a company, rather than the company object. You will also need to create additional relationships not described in this section, such as HAS_SKILL and WORKS, as well as entities defined by yourselves.

Some important entities along with their important attributes are listed in this section. The attributes surrounded by a pair of braces ({ }) should be allowed multiple values, and the attributes that carry a plus (+) are composite fields; you should flatten such fields using relational modeling. **Note: the purpose of the following information is to describe the systems, not to give you an actual schema of the tables in the databases.**

2.1 Labor Force Development (LD) Database

The LD database should support at least four services: 1) Career Planning Service, 2) Job Hunting Assistance, 3) Recruiting Service, and 4) Training Service.

The *career planning service* needs information about each worker: education (i.e., transcripts), training and skills, as well as the skill requirements of jobs in order to help match each worker to an appropriate job. The *training service* recommends courses for a person who pursues a job by recognizing the missing skills of the person for the job. The features of *job hunting assistance* can be derived in similar ways.

Workers are classified as staff (salary worker) and wage-workers. Companies and organizations hold jobs. Workers acquire knowledge and skills through academic courses and skill training. The elements of these two types of activities are courses. The mission of learning is to fill up knowledge/skill gaps between the workers’ possession and a job’s requirements. Each course covers a set of knowledge/skills. For example, CSCI 4125 covers data modeling (medium level), relational models (medium level), SQL (advanced level), JDBC programming (beginner level), RDBMS design (medium level), and database concurrency control (advanced level). A job is a slot that pays one worker only. Each job requires many skills. Jobs can be classified into categories. The description of the same type of jobs is preserved in entity position.

- SKILL(SkillCode, Title, Description, Level). SkillCode is a unique identifier (you may use the skill code defined by the Department of Labor). Level must be “beginner”, “medium” or “advanced”.

- POSITION(PositionCode, Title, Description, PayRangeHigh, PayRangeLow). PositionCode is a unique identifier.

- JOB(JobCode, PositionCode, EmployeeMode, {RequiredSkill}, PayRate, PayType, CategoryCode, Company, more . . .). JobCode is a unique identifier. EmployeeMode must be “full-time” or “part-time”. PayType must be “wage” or “salary”. PayRate is the hourly rate for wage or annual pay for salary. A position belongs to one person only. It should be the most specific one in the job category hierarchy.

- GICS(IndustryID, IndustryTitle, Level, Description, ParentID). GICS is the Global Industry Classification Standard (https://www.msci.com/gics) that defines the names of the business sectors (such as Energy, Materials, and Information Technologies), industry groups (such as "Technology Hardware & Equipment", "Software & Services" and "Semiconductors & Semiconductor Equipment" under sector Information Technologies), industries (such as "Internet Software & Services" and "IT Services" under industry group "Information Technologies") and subindustries (such as Energy Equipment & Services and Oil, Gas & Consumable Fuels under industry "Energy"), as well as the hierarchy between them. The complete GICS list can be found in GICS_map2018.xls included with the project.

- COMPANY(CompanyID, Address+, ZipCode, IndustryGroup, SubIndustry, Website). IndustryGroup and SubIndustry refer to those in GICS. A company belongs to one industry group only but can belong to multiple sub-industries if it has multiple lines of businesses.

- COURSE(CourseCode, Title, Level, Description, Status, RetailPrice). CourseCode is a unique identifier. Status must be either "active" or "expired".

- SECTION(CourseCode, SectionNo, CompletionDate, Year, OfferedBy, Format, Price). The attributes (CourseCode, SectionNo, Year) form a unique identifier for a section. Format must be "classroom" or "online". OfferedBy refers to a university or a training company.

- PERSON(PersonID, Name, Address+, ZipCode, Email, Gender, Phone). PersonID is a unique identifier.

In a person's career, working on multiple jobs simultaneously is common. Therefore, a person can work on one or more jobs. Sometimes, a person's working history can make him/her invaluable at certain situation. Recruiters often seek an applicants' experience. For this reason, the LD database not only records each person's currently jobs, but also tracks every worker's job history.

In addition, many relationships are needed; some of them are listed below.

- WORKS(PersonID, JobCode, StartDate, EndDate). StartDate <= EndDate and an EndDate < today indicates that PersonID no longer holds the job position.

- REQUIRES(PositionCode, Skillcode)

- REQUIRES_BY_JOB(JobCode, SkillCode). Some jobs have extra skill requirements in addition to those specified by the corresponding position. This relationship represents those extra skills.

- TEACHES(CourseCode, Skillcode)

- PREREQUISITE(CourseCode, RequiredCode)

- TAKES(PersonID, CourseCode, SectionNo, CompletionDate)

- HAS_SKILL(PersonID, SkillCode)

2.2 Acar Zone (AZ) Database

Company AZ needs to manage and track the training information of their employees. Thus, the database for AZ contains most entities in the LD database. Rather than tracking information of many companies, the AZ database tracks information of all AZ stores geographically distributed in the nation. Correspondingly, some adjustments are needed in the related entities and relationships. For example, jobs are paid by a store only. Note, the AZ database only tracks who is working on which job without recording workers' work experience. Feel free to add more relations or relationships.

- STORE(StoreID, Address, ZipCode, Phone)
- INVENTORY(ItemNum, Title, Description, Quantity, Unit, AvgCost, ShelfCode, MinLevel). A STORE stocks INVENTORY.
- SALE(InvoiceNbr, Date, ItemNbr, Quantity, Price, Note, MinLevel). A SALE tracks INVENTORY that is sold.
- PURCHASE(PurchaseNbr, Date, ItemNbr, Quantity, UnitCost, Note). A PURCHASE tracks INVENTORY that is bought.
- A CUSTOMER is involved with a SALE and a SUPPLIER is involved with a PURCHASE.

2.3 General Vehicle (GV) Database

Company GV has multiple factories that make various products; these factories have overlapping capacities. GV also needs to manage and track the training information of their employees. Thus the GV database contains most entities in the LD database. Rather than tracking information of many companies, GV database tracks information of all the GV factories. Correspondingly, some adjustments are needed in the related entities and relationships. For example, jobs are associated with factories; one person will not work for more than one GV factory. For business, GV database has entities factory, material, product, contract, lineitem, account_receivable, customer, purchase, and supplier. Both supplier and customer are companies. Note, the GV database does not deal with money flow. It only tracks who is working on which job without recording workers' work experience. Feel free to add more relations or relationships.

- FACTORY(FacotryID, FactoryName, Address, ZipCode, Phone, Manager)
- MATERIAL(MaterialCode, MaterialName, Quantity, Unit, MinLevel)
- PRODUCT(ProductCode, ProductName, Description, Quantity, Unit, AvgCost)
- CONTRACT(ContractID, CustomerID, Date, SaleAmount, PaySchedule). CustomerID is the CompanyID in CUSTOMER. PaySchedule is a document identifier in the accounting system.
- LINEITEM(ContractID, ProductCode, Quantity). LineItem is a relationship that records the line items of every Contract.

- PURCHASE(PurchaseNum, SupplierID, SupplierOrderNum, BookDate, PayDate, Note). SupplierID is the CompanyID in SUPPLIER.
- PURCHASELINE(PurchaseNum, MaterialCode, Quantity). PurchaseLine is a relationship that records MATERIAL for a PURCHASE.
- SUPPLIER(CompanyID, Website, ContactEmail)
- CUSTOMER(CompanyID, ContactPerson, ContactEmail)
- MAKES(FactoryID, ProductCode, Quantity). Makes is a relationship between entities FACTORY and PRODUCT showing the contribution of all the factories.

2.4 Data Population

skill270.xlsx contains the skills appeared in the three spreadsheets Year-2016-LA-xxxx.xls, which are for you to populate your SKILL table. In the LD database, you should prepare companies in the Information Technology sector and in a number of other different sectors.

The actual job information is dynamic. You can start using the data in files indeed100.txt and WorkNOLA36.txt. In the initial testing of your SQL statements, use small data sets so that you can easily know what should be the correct result. For example, 5 people, 10 job positions, 10 courses, and 30 skills should be sufficient. Make sure you have job positions requiring multiple skills, courses teaching multiple different skills, persons possess multiple various skills. Some people are qualified for some job positions; other people are not.

As you can see the data given in the text files is old. You are encouraged to use it to fetch daily/weekly feed of data in the local job market from Web site such as:

- NOLA.com http://jobs.nola.com/Jobs/technology. You can browse by category
- Monster https://www.monster.com/jobs/q-it-jobs-l-new-orleans,-la.aspx
- Indeed https://www.indeed.com/l-New-Orleans,-LA-jobs.html

You can visit these websites to get an idea about the real job positions in the market. For example, indeed.com additionally shows salary estimate, location, experience level. The job ads on some websites such as NOLA.com appear in a very loosely formatted manner. As a consequence, the same attribute may be referred in different terminologies. For instance, “Required skills” is sometimes called “Qualifications” or “Our Expectation”, “Qualifications & characteristics” means “Ideal Candidates”. You have your freedom to include or drop out the less essential attributes such as “Qualifications & characteristics”. Using live data can earn you points for your project.

3 Project Tasks

3.1 Queries

1. List the names for all workers in alphabetical order by last name.

2. List the staff (salary workers) by salary in descending order.

3. List the average annual pay (the salary or wage rates multiplied by 1920 hours) of each store/factory in descending order.

4. List the required skills of a given PositionCode in a readable format.

5. Given a person's identifier, list this person's skills in a readable format.

6. Given a person's identifier, list a person's missing skills for a specific PositionCode in a readable format.

7. List the total number and the total sales ($) of every item in a given period of time (start date, end date) in AZ in the descending order of sales.

8. List the ItemNbr, its title, and the total profit that made the biggest profit for AZ in 2018.

9. Show the items for which the inventory is below the minimum level in AZ system.

10. List the total sales in dollars made to each customer of GV in 2018.

11. Show MaterialCode and MaterialName of the material(s) that GV purchased the most (measured by quantity) in the fourth quarter of 2018.

12. Show the factory name that made the most total quantity of the product that was sold the most in 2018.

13. For the LD database, given a person's identifier, find all the jobs this person is currently holding and worked in the past.

14. In a local or national crisis, we need to find all the people who once held a position of the given PositionCode. List PersonID, Name, JobTitle and the Years the person worked in (starting year and ending year).

15. Find all the unemployed people who once held a job position of the given PositionCode.

16. List the average, maximum and minimum annual pay (total salaries or wage rates multiplying by 1920 hours) of each industry (listed in GICS) in the order of the industry names.

17. Find out the biggest employer, industry, and industry group in terms of number of employees. (Note: This should be three separate queries)

18. Find out the job distribution among industries by showing the number of employees in each industry.

19. Given a person's identifier and a PositionCode, find the courses (course id and title) that each alone teaches all the missing skills for this person to be qualified for the specified position, assuming the skill gap of the worker and the requirement of the position can be covered by one course.

20. Given a person's identifier, find the job position with the highest pay rate for this person according to his/her skill possession.

21. Given a position code, list all the names along with the emails of the persons who are qualified for this position.

22. When a company cannot find any qualified person for a job position, a secondary solution is to find a person who is almost qualified to the job position. Make a "missing-k" list that lists people who miss only k skills for a specified PositionCode; k ¡ 4.

23. Suppose there is a new position that has nobody qualified. List the persons who miss the least number of skills that are required by this PositionCode and report the "least number".

24. List each of the skill code and the number of people who misses the skill and are in the missing-k list for a given position code in the ascending order of the people counts.

25. Find out the number of the workers whose earnings increased in a specific industry group (use attribute "industry group" in table Company). [Hint: earning change = the sum of a person's current earnings – the pay of the person's the last previous job.]

26. **Extra Credit**. Find the position that has the most openings due to lack of qualified workers. If there are many openings of a position but at the same time there are many qualified jobless people. Then training cannot help fill up this type of job vacancies. What we want to find is the position that has the largest difference between vacancies (the unfilled jobs) and the number of jobless people who are qualified for the position.

27. **Extra Credit.** Find the course sets with up to three courses that teach every skill required by the position(s) found in Query #26. These courses should effectively help most jobless people become qualified for the jobs with high demands.

28. **Extra Credit.** List all the courses, directly or indirectly required, that a person has to take in order to be qualified for the job of given JobCode according to his/her skills possessed and courses taken based on the PersonID.

3.2 Development Tasks

(1) Make an E-R diagram showing your data model. In the diagram, show the important attributes of each entity. For every relationship, show the cardinality and the participation status, as well as any important attributes associated with the relationship. Be sure to indicate the primary key of each strong entity. Note: this is a data-model E-R diagram, not a detailed relation schema diagram for implementation. I will accept hand-draw diagrams. Feel free to discuss your E-R diagram during office hours.

(2) Reduce the data-model E-R diagram to the database relations schemas. Present the schema of every table in the format of TableName(attribute1, attribute2, ...).

(3) Populate the tables with adequate data such that all the questions can be demonstrated. Be sure to write your INSERT statements in SQL scripts. You are also required to write a SQL script that cleans up everything in the database; this will be a good exercise to realize the dependencies enforced by foreign key references. In your development process, you must re-populate your database after cleaning up it for multiple times.

(4) Write the SQL statements that carry out the 28 queries listed in Section 3.1.

(5) Enumerate the concerned functional dependencies in all three databases. Revise the database schema made in (2) by producing a lossless-join 3NF schema that preserves functional dependencies.

(6) Design and implement four of the Java classes such as Skill, Job, Person and Has_Skill with the JDBC technology; one of them must be a relationship table. These Java classes should support creation (such as creating a course, a job category, or a job position) and deletion (such as removing a job position or setting a course inactive.) Note, do not spend too much time on too many "classes".

(7) Design and implement database applications that manage business processes. Write a user interface in Java to drive each of these queries with JDBC. A graphical interface (GUI) will be appreciated somewhat; it is not necessary.

a) Company AZ hires a new employee; You can assume the information of the person is already in the LD database. The process starts with a PersonID and a JobCode. Then the application should automate the steps shown below by interacting with the user. Step 1 Fetch the worker's information including personal and skills from LD; Step 2 Upload the person's transcripts and input the course taken into table Takes, assuming every course is in table Course. Derive the skills based on the transcripts. Show the difference between the skills derived against the self-claimed skills acquired from the LD database. If the difference is significant, the personnel manager can rescind the job offer. Step 3 Populate table Has_Skill with more rows derived from the courses this person have taken; Step 4 Verify if this person has every skill required by the given PositionCode; Step 5 If a skill gap is identified, propose a training plan for this person.

b) Company GV transfer a worker from one factory to another factory.

c) Company GV makes a deal with company Exchange Market (EM) EM is both a customer and a supplier of GV. The deal requires EM to buy products of GV worth $100 million, and requires GV to purchase materials from EM worth $75 million. Your program has to guarantee both a contract and a purchase with EM are completed. A failure of either process will cancel the entire deal.

d) Company AZ and company GV exchange their CIOs. This deal will be done only if both CIOs are successfully moved to the other sides. If either move fails, the whole deal is called off.

e) LD collects job information from AZ and GV. It is conceivable that there are two ways, push or pull, for LD to collect job information from AZ and GV. The push method requires AZ and GV to send information to LD whenever AZ and/or GV have/has personnel changes. The pull method requires LD to request the job information from AZ and GV periodically. You should consider the potential problems caused by concurrent operations and implement one of the methods.

3.3 Team Organization

You are required to form a team made of three people. Each team member should work on one of the three databases. Every team member should perform every aspect in the development process of database applications such as E-R modeling, SQL, and Java. Do not divide your tasks horizontally such as one writes SQL only, the other writes Java only. Horizontally dividing the tasks will give the Java person serious disadvantages in earning a fair grade. Every member has to understand every SQL statement for the 28 queries and every Java program for the 5 business processes.

Every group must sign up for an hour-long project inspection with me. At least two test cases should be prepared and documented for queries 8 through 28, as well as for tasks (a) through (e) specified in business processes (7). Each test case must include the prepared input data/setting and a specification of the expected output/results. Early submission and demonstration of project earn extra credit.

3.4 Due Dates and Submission

Due date of Phase 1 (10%) – ~~Friday, March 12th~~ Monday, March 15th: deliverables for task (1).

Due date of Phase 2 (35%) – Friday, April 9th: the deliverables for tasks (2), (3) and (4). Queries #1 through #25 will be graded.

Due date of Phase 3 (55%) – Friday, May 7th: the project report includes everything such as the revised E-R diagram, table schemas of tasks (1) and (2), revised SQL statements of task (4) and bonus queries. The work of tasks (5) to (6) must be presented in the report. The implementations of Tasks (7.a to 7.e) will be examined by real-time tests and code inspection.

Early project inspection: April 30th - May 2nd: 20%, May 3rd - 4th: 15%, May 5th: 10%, project phase 3 report submitted after May 7th: -10% per class day

Project inspection ends May 10th.