Competencies

4018.1.1 : Conceptual Models to Physical Schemas

The graduate creates conceptual data models and translates them into physical schemas.

4018.1.2 : Create Databases

The graduate creates databases utilizing SQL Data Definition Language (DDL) in MySQL environment.

4018.1.3 : Create/Modify Tables and Views

The graduate creates and modifies tables and views employing SQL Data Definition Language (DDL) in MySQL environment.

4018.1.4 : Create Primary Keys/Foreign Keys and Indexes

The graduate creates and modifies primary keys (PKs) and foreign keys (FKs) and indexes with SQL Data Definition Language (DDL) in MySQL environment.

4018.1.5 : Populate Tables

The graduate populates tables with insert, update, and delete using DML in the MySQL environment.

4018.1.6 : Create Simple and Complex Queries

The graduate creates simple Select-From-Where (SFW) and complex 3+ table join queries with Data Manipulation Language (DML) in MySQL environment.

Introduction

For this assessment, you will be creating an entity-relationship (ER) model, databases, tables, and queries for two fictional small businesses. To complete this assessment, you will use SQL to test and run a database application that you will develop. After running the code, you will take a screenshot of your results and paste the screenshot into a document that you will submit.

The work you complete for each part of the assessment (i.e., the design models and diagrams, tables, written explanations, SQL script code, and screenshot results from running your SQL scripts in a SQL tool) should be saved as a single PDF file that you will submit.

Note: If you do not have access to a database tool, you may use SQL Fiddle (an online SQL tool) to complete this assessment. The tool can be accessed using the "SQL Fiddle" link in the Web Links section of this task. Instructions for how to use SQL Fiddle for each part of the assessment are included in the attached document "SQL Fiddle Instructions." Please note that for each part of the assessment, there are explicit instructions on what SQL code you will need to copy and paste into the SQL Fiddle panels to run your test.

Scenario

You are a database designer and developer who has been hired by two local businesses, Nora’s Bagel Bin and Jaunty Coffee Co., to build databases to help them manage their businesses. First, you will design a normalized physical database model to store data for Nora’s Bagel Bin’s ordering system. Then, you will use an existing database design document for Jaunty Coffee Co. to create its database. Once the tables have been built, you will load them with sample data and create a view and an index to protect and improve query performance. Finally, you will create both a simple query and a more complex table joins query to produce meaningful reports from the newly created database.

Requirements

Your submission must be your original work. No more than a combined total of 30% of a submission and no more than a 10% match to any one individual source can be directly quoted or closely paraphrased from sources, even if cited correctly. An originality report is provided when you submit your task that can be used as a guide.

You must use the rubric to direct the creation of your submission because it provides detailed criteria that will be used to evaluate your work. Each requirement below may be evaluated by more than one rubric aspect. The rubric aspect titles may contain hyperlinks to relevant portions of the course.

Tasks may ***not*** *be submitted as cloud links, such as links to Google Docs, Google Slides, OneDrive, etc., unless specified in the task requirements. All other submissions must be file types that are uploaded and submitted as attachments (e.g., .docx, .pdf, .ppt).*

A. Construct a normalized physical database model to represent the ordering process for Nora's Bagel Bin by doing the following:

   Note: Before proceeding, familiarize yourself with the ordering process for Nora's Bagel Bin by reviewing the following documents in the Supporting Documents section of this task: the shop's unnormalized sales order form ("Bagel Order Form") and the first normal form (1NF) provided in the "Nora's Bagel Bin Database Blueprints."

   1. Complete the second normal form (2NF) section of the attached "Nora's Bagel Bin Database Blueprints" document by doing the following:
      a. Assign *each* attribute from the 1NF table into the correct 2NF table.
      b. Describe the relationship between the **two** pairs of 2NF tables by indicating their cardinality in *each* of the dotted cells: one-to-one (1:1), one-to-many (1:M), many-to-one (M:1), or many-to-many (M:M)

   Note: Cardinality is read left to right and top to bottom, and the preferred method of notation is crow's foot.

      c. Explain how you assigned attributes to the 2NF tables and determined the cardinality of the relationships between your 2NF tables.
   2. Complete the third normal form (3NF) section of the attached "Nora's Bagel Bin Database Blueprints" document by doing the following:
      a. Assign *each* attribute from your 2NF "Bagel Order" table into one of the new 3NF tables. Copy *all* other information from your 2NF diagram into the 3NF diagram.
      b. Provide *each* 3NF table with a name that reflects its contents.
      c. Create a new field that will be used as a key linking the **two** 3NF tables you named in part A2b. Ensure that your primary key (PK) and foreign key (FK) fields are in the correct locations in the 3NF diagram.

   d. Describe the relationships between the 3NF tables by indicating their cardinality in *each* of the dotted cells: one-to-one (1:1), one-to-many (1:M), many-to-one (M:1), or many-to-many (M:M).

Note: Cardinality is read left to right and top to bottom, and the preferred method of notation is crow's foot.

   e. Explain how you assigned attributes to the 3NF tables and determined the cardinality of the relationships between your 3NF tables.

3. Complete the "Final Physical Database Model" section of the attached “Nora’s Bagel Bin Database Blueprints” document by doing the following:
   a. Copy the table names and cardinality information from your 3NF diagram into the “Final Physical Database Model” and rename the attributes.
   b. Assign **one** of the following **five** data types to *each* attribute in your 3NF tables: CHAR(), VARCHAR(), TIMESTAMP, INTEGER, or NUMERIC(). *Each* data type must be used *at least* once.

B. Create a database using the attached "Jaunty Coffee Co. ERD" by doing the following:

1. Develop SQL code to create *each* table as specified in the attached “Jaunty Coffee Co. ERD” by doing the following:
   a. Provide the SQL code you wrote to create *all* the tables.
   b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server’s response.

2. Develop SQL code to populate *each* table in the database design document by doing the following:

   Note: This data is not provided. You will be fabricating the data for this step.

   a. Provide the SQL code you wrote to populate the tables with *at least* **three** rows of data in *each* table.
   b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server’s response.

3. Develop SQL code to create a view by doing the following:
   a. Provide the SQL code you wrote to create your view. The view should show *all* of the information from the “Employee” table but concatenate *each* employee’s first and last name, formatted with a space between the first and last name, into a new attribute called employee_full_name.
   b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server’s response.

4. Develop SQL code to create an index on the coffee_name field by doing the following:
   a. Provide the SQL code you wrote to create your index on the coffee_name field from the “Coffee” table.

   b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server’s response.
5. Develop SQL code to create an SFW (SELECT–FROM–WHERE) query for *any* of your tables or views by doing the following:
   a. Provide the SQL code you wrote to create your SFW query.
   b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server’s response.
6. Develop SQL code to create a query by doing the following:
   a. Provide the SQL code you wrote to create your table joins query. The query should join together **three** different tables and include attributes from *all* three tables in its output.
   b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server’s response.

C. Submit parts A and B as a PDF, with *each* part clearly labeled.

D. Demonstrate professional communication in the content and presentation of your submission.

File Restrictions
File name may contain only letters, numbers, spaces, and these symbols: ! - _ . * ' ( )
File size limit: 200 MB
File types allowed: doc, docx, rtf, xls, xlsx, ppt, pptx, odt, pdf, txt, qt, mov, mpg, avi, mp3, wav, mp4, wma, flv, asf, mpeg, wmv, m4v, svg, tif, tiff, jpeg, jpg, gif, png, zip, rar, tar, 7z