Microsoft SQL Server

Certification Project II

Scenario:

HerculesMotoCorp is one of the most prominent retailers and garage handlers in the United States. They deal with all kinds of automobiles as well as custom-made vehicles ranging from bikes to cars. This makes their database extremely important not only for record keeping but to show the government authorities that their work is well within the bounds of federal law.

Problem Statement:

HerculesMotoCorp has reached out to your company to build a database that meets their requirements. You are the Database Administrator of your company your task is to design a comprehensive database that not only eases the task of the HerculesMotoCorp employees but also helps other parties such as shareholders to view and understand the data.

Tasks:

1. Name the database as **MotorsCertification**. Design an ER model based on the following parameters:
   Note: Build a ER diagram with proper entities, relationship etc.
   - Design a table/database object named **orderdetails** with the following attributes/columns:
     - orderNumber int(), Primary Key
     - productCode varchar()
     - quantityOrdered int()
     - priceEach float
     - orderLineNumber smallint()

   Foreign Key: orders (orderNumber → orderNumber) and products (productCode → productCode)

   Index 1: PRIMARY, Type: BTREE, Unique Yes, Visible No, Columns orderNumber

   Index 2: productCode, Type: BTREE, Unique: No, Visible: No, Columns productCode

   - Design a table/database object named **customers** with the following attributes/columns:
     - customerNumber int(11) PK
     - customerName varchar(50)
     - contactLastName varchar(50)
     - contactFirstName varchar(50)
     - phone varchar(50)

- addressLine1 varchar(50)
- addressLine2 varchar(50)
- city varchar(50)
- state varchar(50)
- postalCode varchar(15)
- country varchar(50)
- salesRepEmployeeNumber int(11)
- creditLimit float

Foreign Key: employees (salesRepEmployeeNumber → employeeNumber).

Index: PRIMARY, Type: BTREE, Unique: Yes, Visible: No, Columns: customerNumber

- Design a table/database object named **employees** with the following attributes/columns:
  - employeeNumber int() PK
  - lastName varchar()
  - firstName varchar()
  - extension varchar()
  - email varchar()
  - officeCode varchar()
  - reportsTo int()
  - jobTitle varchar()

Foreign Key: employees (reportsTo → employeeNumber) and offices (officeCode → officeCode)

Index 1: PRIMARY, Type: BTREE, Unique: Yes, Visible No, Columns: employeeNumber

Index 2: reportsTo, Type: BTREE, Unique: No, Visible: No, Columns reportsTo

Index 3: officeCode, Type: BTREE, Unique: No, Visible: No, Columns officeCode

- Design a table/database object named **orders** with the following attributes/columns:
  - orderNumber int() PK
  - orderDate date

- requiredDate date
- shippedDate date
- status varchar()
- comments text
- customerNumber int()

Foreign Key: customers (customerNumber → customerNumber).

Index 1: PRIMARY, Type: BTREE, Unique: Yes, Visible: No, Columns orderNumber

Index 2: customerNumber, Type: BTREE, Unique: No, Visible: No, Columns customerNumber

- Design a table/database object named **offices** with the following attributes/columns:
  - officeCode varchar() PK
  - city varchar()
  - phone varchar()
  - addressLine1 varchar()
  - addressLine2 varchar()
  - state varchar()
  - country varchar()
  - postalCode varchar()
  - territory varchar()

Index 1: PRIMARY, Type: BTREE, Unique: Yes, Visible No, Columns: officeCode

- Design a table/database object named **payments** with the following attributes/columns:
  - customerNumber int() PK
  - checkNumber varchar(50)
  - paymentDate date
  - amount float

Foreign Key: customers (customerNumber → customerNumber)
Index: PRIMARY, Type: BTREE, Unique: Yes, Visible: No, Columns customerNumber and checkNumber

- Design a table/database object named **productlines** with the following attributes/columns:
  - productLine varchar(50) PK
  - textDescription varchar(4000)

- htmlDescription NULL
- image NULL

- Design a table/database object named **products** with the following attributes/columns:
  - productCode varchar() PK
  - productName varchar()
  - productLine varchar()
  - productScale varchar()
  - productVendor varchar()
  - productDescription text()
  - quantityInStock smallint(6)
  - buyPrice float
  - MSRP float

  Foreign Key: productlines (productLine → productLine).

  Index 1: PRIMARY, Type: BTREE, Unique: Yes, Visible: No, Columns productCode

  Index 2: productLine, Type: BTREE, Unique: No, Visible: No, Columns productLine

2. After designing the table insert records in the following **orderdetails, employees, payments, products, customers, offices and orders** table. Note: Refer to the CSV files provided on the LMS.
3. Provide comments before every task that is performed describing the operation that is being performed and attach a screenshot of ER diagram from SSMS.
4. Delete the columns in **productlines** which are useless that do not infer anything.
5. Use a select statement to verify all insertions as well as updates.
6. Find out the highest and the lowest amount.
7. Give the unique count of customerName from **customers.**
8. Create a view from **customers** and **payments** named cust_payment and select customerName, amount, contactLastName, contactFirstName who have paid.
   Truncate and Drop the view after operation.
9. Create a stored procedure on **products** which displays productLine for Classic Cars.
10. Create a function to get the creditLimit of **customers** less than 96800.

11. Create Trigger to store transaction record for **employee** table which displays employeeNumber, lastName, FirstName and office code upon insertion
12. Create a Trigger to display customer number if the amount is greater than 10,000
13. Create Users, Roles and Logins according to 3 Roles: Admin, HR, and Employee. Admin can view full database and has full access, HR can view and access only employee and offices table. Employee can view all tables only.
    Note: work from Admin role for any changes to be made for database.
14. Schedule a Job which backups and schedule it according to developer preference.
15. Open Activity Monitor and list down some minor observations including Processes, Resource Waits, and Active Expensive Queries.
16. Migrate the following SQL server workload to azure.

You must consider the above-mentioned requirements and give an appropriate solution with proper screenshots in a word/pdf file, include .sql file which has all the queries as well as the final MotorsCertification. Bak file while submitting your project