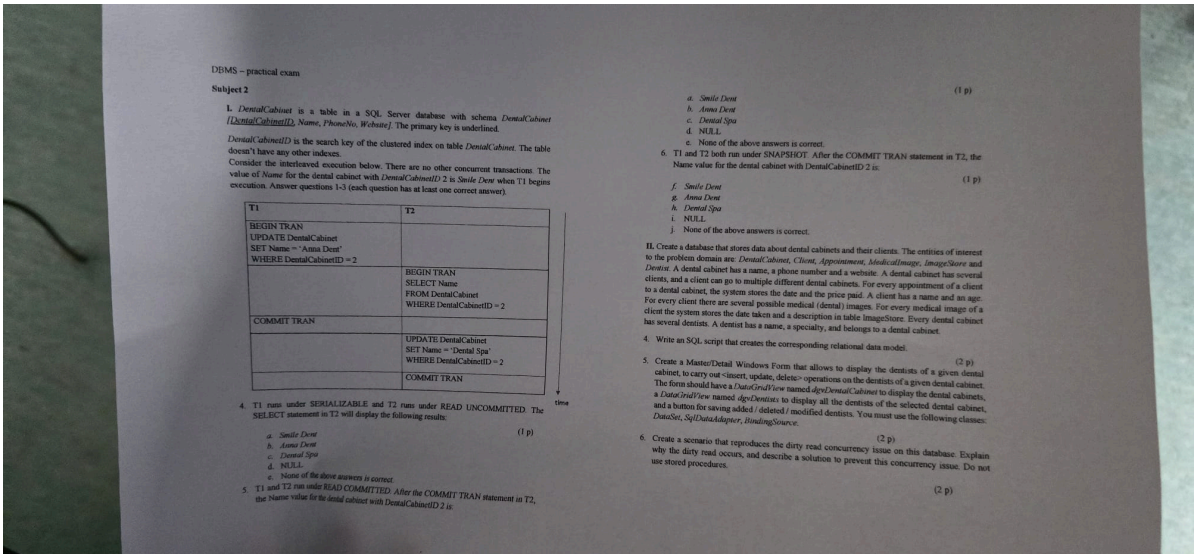
Prep practical exam

Exam model

1. Create the script to generate the db
2. lab 1
3. Concurrency issue (lab 3) + solution → change transaction isolation level



DBMS – practical exam

Subject 2

I. *DentalCabinet* is a table in a SQL Server database with schema *DentalCabinet [DentalCabinetID, Name, PhoneNo, Website]*. The primary key is underlined.

DentalCabinetID is the search key of the clustered index on table *DentalCabinet*. The table doesn't have any other indexes.
Consider the interleaved execution below. There are no other concurrent transactions. The value of *Name* for the dental cabinet with *DentalCabinetID* 2 is *Smile Dent* when T1 begins execution. Answer questions 1-3 (each question has at least one correct answer).

T1	T2
BEGIN TRAN UPDATE DentalCabinet SET Name = 'Anna Dent' WHERE DentalCabinetID = 2	
	BEGIN TRAN SELECT Name FROM DentalCabinet WHERE DentalCabinetID = 2
COMMIT TRAN	
	UPDATE DentalCabinet SET Name = 'Dental Spa' WHERE DentalCabinetID = 2
	COMMIT TRAN

time

4. T1 runs under SERIALIZABLE and T2 runs under READ UNCOMMITTED. The SELECT statement in T2 will display the following results: (1 p)
   a. *Smile Dent*
   b. *Anna Dent*
   c. *Dental Spa*
   d. NULL
   e. None of the above answers is correct.
5. T1 and T2 run under *READ COMMITTED*. After the COMMIT TRAN statement in T2, the Name value for the dental cabinet with DentalCabinetID 2 is: (1 p)
   a. *Smile Dent*
   b. *Anna Dent*
   c. *Dental Spa*
   d. NULL
   e. None of the above answers is correct.
6. T1 and T2 both run under SNAPSHOT. After the COMMIT TRAN statement in T2, the Name value for the dental cabinet with DentalCabinetID 2 is: (1 p)
   f. *Smile Dent*
   g. *Anna Dent*
   h. *Dental Spa*
   i. NULL
   j. None of the above answers is correct.

II. Create a database that stores data about dental cabinets and their clients. The entities of interest to the problem domain are: *DentalCabinet, Client, Appointment, MedicalImage, ImageStore* and *Dentist*. A dental cabinet has a name, a phone number and a website. A dental cabinet has several clients, and a client can go to multiple different dental cabinets. For every appointment of a client to a dental cabinet, the system stores the date and the price paid. A client has a name and an age. For every client there are several possible medical (dental) images. For every medical image of a client the system stores the date taken and a description in table ImageStore. Every dental cabinet has several dentists. A dentist has a name, a specialty, and belongs to a dental cabinet.

4. Write an SQL script that creates the corresponding relational data model. (2 p)
5. Create a Master/Detail Windows Form that allows to display the dentists of a given dental cabinet, to carry out <insert, update, delete> operations on the dentists of a given dental cabinet. The form should have a *DataGridView* named *dgvDentalCabinet* to display the dental cabinets, a *DataGridView* named *dgvDentists* to display all the dentists of the selected dental cabinet, and a button for saving added / deleted / modified dentists. You must use the following classes: *DataSet, SqlDataAdapter, BindingSource*. (2 p)
6. Create a scenario that reproduces the dirty read concurrency issue on this database. Explain why the dirty read occurs, and describe a solution to prevent this concurrency issue. Do not use stored procedures. (2 p)

Preparation

▼ Lab 1 model in DBMS folder

(change only the constants at the beginning)

▼ Concurrency issues

1. Dirty read
   - a transaction reads uncommitted data (data changed by another ongoing transaction)
   - SOLUTION ⇒ Read commited

2. Non-repeatable read
    - a row read by a transaction is changed by another tran while the reader is in progress
      (if the first transaction reads the row again it will get different row values)
    - SOLUTION ⇒ Repeatable read
3. Phantom read
    - transaction T1 reads a set of rows based on a search predicate; transaction T2 generates a new row (I/U) that matches the search predicate while T1 is ongoing;
      if T1 issues the same read operation, it will get an extra row
    - SOLUTION ⇒ Serializable
4. Deadlock
    - SOLUTION ⇒ access resources in the same order

concurrency probl. / isolation level	Chaos	Read Uncommitted	Read Committed	Repeatable Read	Serializable
Lost Updates?	Yes	No	No	No	No
Dirty Reads?	Yes	Yes	No	No	No
Unrepeatable Reads?	Yes	Yes	Yes	No	No
Phantoms?	Yes	Yes	Yes	Yes	No

Locks:

- READ UNCOMMITTED
    - no S locks when reading data
- READ COMMITTED

    - S locks - released as soon as the SELECT operation is performed;
    - X locks - released at the end of the transaction
- REPEATABLE READ
    - holds S locks and X locks until the end of the transaction
- SERIALIZABLE
    - holds locks (including key-range locks) during the entire transaction