Fall 2025 Oct 27, 2025
Y.Wu

CSDS 341: Introduction to Databases
Assignment #4

Due: Nov 4, 2025 (11:59 PM ET)

All assignments in this course are to be digitally produced and submitted to the canvas as pdf documents.

1. [View Definition] (24). A healthcare research team uses AI models to predict disease risk based on patient data. To protect privacy, some patient attributes should not be exposed to analysis views.

Patient (patientID, name, age, diagnosis)
AIModel (modelID, modelName, domain)
Experiment (expID, modelID, patientID, predictedRisk, runtimeSec, trueOutcome, accuracy)

Here Experiment.modelID is a foreign key that references AIModel.modelID as a primary key; Experiment.patientID is a foreign key that references Patient.patientID as a primary key.

Put down the SQL statement to create views for the following needs.

a. [**Deidentified Experiments**]. Define a view Deidentified_Exp that shows experiment results, that publish only the following attributes: expID, modelName, patient's age as "patientAge", diagnosis, predictedRisk, trueOutcome, accuracy. *(Hint: Use JOIN and exclude the patient's name.)*

b. [**High-Performance Models**]. Define a view High_Performance_models that lists all experiments where the accuracy is greater than or equal to 0.9 and has runtimeSec less than 100 (seconds). The view should include (expID, modelName, accuracy, predictedRisk, runtimeSec).

c. **[Model Average Accuracy].** Define a view Model_Avg_Accuracy that summarizes the average accuracy of each AI model across all experiments. The view should include: modelID, modelName, avgAccuracy. *(Hint: Use GROUP BY and the aggregate function AVG().)*

2. Consider the views you created in Question 1 (26). For each query below, decide which view(s), if any, can be used alone to answer the query (without referring to base tables). Just write down the name of the view.

Q1 (5): Find the expID and model names where the model accuracy ≥ 0.95.

Q2 (5): List the model names and their accuracy with predictedRisk < 0.3.

Q3 (5): List all patient age and diagnosis, along with predictedRisk and accuracy.

Q4 (5): List each AI model and its average accuracy across all experiments.

Q5 (6): List all expID of the experiments with model accuracy ≥ 0.90 and the domain of the models in "medical"

3 [ACID] (20). An AI travel agent needs to access the database below:
Customer(customerID, name, email)
Booking(bookingID, customerID, tripID, status, price)
Trip(tripID, destination, availableSeats, price)

A travel agency uses an AI agent to automate travel booking. The AI agent executes SQL transactions to reserve trips for customers. Each booking transaction consists of:

- Checking if seats are available for the chosen trip.
- Deducting one seat from Trip.availableSeats.
- Inserting a record into Booking with status 'CONFIRMED'.
- Charging the customer (assume the payment always succeeds).

For each scenario below, identify which ACID property (Atomicity, Consistency, Isolation, Durability) is being described and explain why.

Scenario A: The AI agent crashes after reducing available Seats but before inserting the booking. After recovery, no seats were deducted, and no booking exists.

Scenario B: The AI agent completes the booking transaction, the system crashes immediately afterward, but when the system restarts, the booking and seat deduction are still present.

Scenario C: Two AI agents simultaneously try to book the last seat on a trip. Both read availableSeats = 1 before any write, causing two bookings to succeed and availableSeats to go negative.

Scenario D: The AI agent tries to book a trip with availableSeats = 0. The transaction is automatically rejected.

4. [Scheduling] (20) Consider the above schema, and two transactions:

T1: Book 1 seat on Trip T100 for Customer C1:
- Read(Trip.availableSeats)
- Trip.availableSeats = Trip.availableSeats - 1
- Write(Trip.availableSeats)
- Insert Booking

T2: Book 1 seat on Trip T100 for Customer C2:
- Read(Trip.availableSeats)
- Trip.availableSeats = Trip.availableSeats - 1
- Write(Trip.availableSeats)
- Insert Booking

Consider the following three schedules that execute T1 and T2.

Schedule A:
- T1: Read(Trip.availableSeats)
- T1: Trip.availableSeats = Trip.availableSeats - 1
- T1: Write(Trip.availableSeats)
- T1: Insert Booking
- T2: Read(Trip.availableSeats)
- T2: Trip.availableSeats = Trip.availableSeats - 1
- T2: Write(Trip.availableSeats)
- T2: Insert Booking

Schedule B:
- T1: Read(Trip.availableSeats)
- T2: Read(Trip.availableSeats)
- T1: Trip.availableSeats = Trip.availableSeats - 1
- T2: Trip.availableSeats = Trip.availableSeats - 1
- T1: Write(Trip.availableSeats)
- T2: Write(Trip.availableSeats)
- T1: Insert Booking
- T2: Insert Booking

Schedule C:
- T1: Read(Trip.availableSeats)
- T1: Trip.availableSeats = Trip.availableSeats - 1
- T2: Read(Trip.availableSeats)   -- reads the updated value from T1
- T1: Write(Trip.availableSeats)
- T2: Trip.availableSeats = Trip.availableSeats - 1
- T2: Write(Trip.availableSeats)
- T1: Insert Booking
- T2: Insert Booking

Which schedule(s) has issues such as Lost Update, Phantom Read or Read 'dirty' data? Which does not? Briefly explain.

5. [Scheduling] (10) Consider a schema R(A) with two tuples {(5), (6)}. There are two transactions:

- T1: A = A+1 (update A set A = A+1)
- T2: A=2*A (update A set A=2*A)

Which of the following is *not possible* for the final results of executing these two transactions if ACID is guaranteed? Briefly explain.

A. {(10), (12)}

B. {(11), (13)}

C. {(11), (12)}

D. {(12), (14)}