Relational Databases - Week2 Quiz

Question 1
Consider this data:

Student ID	Name	Grade Level	GPA
930	Olufunmilayo Ayton	11	4.00
667	Vincent Michaelson	10	2.53
907	Asa Quigg	10	3.57
168	Kiran Patil	11	3.28

Which of these tables would accept this data? Check all that apply.
(Note: This isn't asking which are good table definitions; it's only asking which would accept the data for storage.)
Ans:

Column	Data Type
student_id	INT
name	STRING
grade_level	INT
gpa	STRING

Column	Data Type
student_id	INT
name	STRING
grade_level	INT
gpa	DECIMAL(3,2)

Column	Data Type
student_id	STRING
name	STRING
grade_level	STRING
gpa	STRING

Question 2
Here is a table definition:

Column	Data Type	Notes
student_id	STRING	PK
name	STRING	NOT NULL
grade_level	STRING	
gpa	DECIMAL(2,1)	

Which rows can be stored in this data? Choose all that apply.
Ans: {student_id : '392', name : 'Kamalani Hale', gpa : 4.0}
{student_id : '93', name : 'Tilly Sokolowski', grade_level : 'New Student'}

Question 3
What is database normalization?
Ans: Designing the tables in your relational database so that redundant storage is minimized and the chance of inconsistencies in the data is also reduced.

Question 4
Which of the following rules are well-known conditions that help define third normal form? (Note, we are stating the rules a bit informally.) Choose all that apply.
Ans: The non-key columns of a table must be dependent on the key only. For example, if you have an employee table with employee id as the key, then you might have a department id column for the employee, but not department name also (because the department name would be dependent on the department id, which is not your table's primary key).
Every table in your database must have a primary key.
There must be no repeating groups in any table. For example, you will not have a column that can contain one or more phone numbers.

Question 5
Which of the following are costs of normalization? Choose all that apply.
Ans: Normalizing a database design generally will make your queries run less efficiently.
Normalizing a database requires more complex queries on your data to answer many questions.

Question 6
Why might you find it helpful to denormalize your database design? Choose all that apply.
Ans: In a system where join processing is slower, denormalizing can improve the runtime speed of many queries and reports.
Denormalizing will "pre-join" your previously normalized tables and store them that way, so fewer joins are needed in your queries.
If you frequently query some summary data, like store daily sales totals, keeping a summary table reduces the need to recompute summaries.

Question 7
Which of these accurately describe why features of operational databases are not needed for analytic databases?
Ans: Analytic databases update infrequently so ETL (extract, transform, and load) utilities can replace many of the DML features of operational databases.
Analytic databases often use data collected from other sources (including other operational databases), so enforcing business rules is typically not needed.