✓ **Congratulations! You passed!**

TO PASS 75% or higher

Keep Learning

GRADE
100%

Relational Databases

LATEST SUBMISSION GRADE

100%

1. Consider this data:

1 / 1 point

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.)

[✓]

Column	Data Type
student_id	STRING
name	STRING
grade_level	STRING
gpa	STRING

✓ **Correct**
Correct. The numerical columns *can* be integer or string, though you probably wouldn't want to do that.

[✓]

Column	Data Type
student_id	INT
name	STRING
grade_level	INT
gpa	STRING

✓ **Correct**
Correct. You probably would not want the GPA to be entered as a string, but it could be done.

[✓]

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

✓ **Correct**
Correct. The **student_id** and **grade_level** columns can be integers, and the GPA is a decimal with three digits, two of which are after the decimal point.

[ ]

Column	Data Type
student_id	INT
name	STRING
grade_level	STRING
gpa	DECIMAL(1,2)

[ ]

Column	Data Type
student_id	STRING
name	STRING
grade_level	STRING
gpa	INT

2. Here is a table definition:

1 / 1 point

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.

[ ] {student_id : '732', name : 'Sanjiv Chaudhari', gpa : '3.9'}

☑ {student_id : '93', name : 'Tilly Sokolowski', grade_level : 'New Student'}

✓ **Correct**
Correct. All three provided fields are strings, so although the grade level is not really ' **New Student**' this would be an accepted value. The absence of a value for **gpa** is also allowed (and for a new student, likely).

☑ {student_id : '392', name : 'Kamalani Hale', gpa : 4.0}

✓ **Correct**
Correct. Both **student_id** and **name** are strings, and **gpa** is an decimal of the correct form. Although **grade_level** doesn't have a value, this is allowed—only **student_id** and **name** must have a value.

☐ {name : 'Qiu Yuen'}

☐ {name : 'Sandalio Abascal', grade_level : '10', gpa : 3.2}

3. What is database normalization? **1 / 1 point**

● Designing the tables in your relational database so that redundant storage is minimized and the chance of inconsistencies in the data is also reduced.

○ Using well known table names and column names for common sorts of records, like customers, stores, and items.

○ Tidying-up the data so that bad records, records with important values missing, and erroneous outliers are removed.

○ Combining data from different tables into one larger table, so that records from different tables don't need to be joined together to give complete reports.

✓ **Correct**
Correct. There are many levels of normalization, but each level aims at further reducing storage needs and chances of inconsistencies.

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. **1 / 1 point**

☑ 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).

✓ **Correct**
Correct. This limits redundant information and inconsistencies.

☐ You must store everything as efficiently as possible.

☐ A table must always hold all known information about a 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.

✓ **Correct**
Correct. Repeating groups makes many questions more difficult to answer.

☑ Every table in your database must have a primary key.

✓ **Correct**
Correct. This is the first of our stated conditions, and in fact the first condition for all normal forms.

5. Which of the following are *costs* of normalization? Choose all that apply. **1 / 1 point**

☑ Normalizing a database requires more complex queries on your data to answer many questions.

✓ **Correct**
Correct. In fact, a large database that is fully set into Third Normal Form can often require several joins in a single query to generate a needed report. This adds complexity to your query writing task.

☐ Normalizing a database can degrade the integrity of your data.

☐ Normalizing a database design generally will make the total storage of your data smaller.

☑ Normalizing a database design generally will make your queries run less efficiently.

✓ **Correct**
Correct. Normalization usually increases the number of joins your queries need to execute, which cost query execution time.

6. Why might you find it helpful to denormalize your database design? Choose all that apply. **1 / 1 point**

☑ Denormalizing will "pre-join" your previously normalized tables and store them that way, so fewer joins are needed in your queries.

✓ **Correct**

Correct. Although a larger database is likely to result, pre-joining does allow queries to use fewer joins.

- ☐ If your company is approaching its maximum storage capacity, and obtaining more is not an option for the near future, denormalizing allows you to reduce the storage needs in the short term.
- ☐ When using an operational database, database denormalization keeps important information about a key in one row so it's easier to maintain accuracy.
- ☑ In a system where join processing is slower, denormalizing can improve the runtime speed of many queries and reports.

✓ **Correct**

Correct. Although more data would need to be managed, avoiding the need to perform joins would provide a greater benefit.

- ☑ If you frequently query some summary data, like store daily sales totals, keeping a summary table reduces the need to recompute summaries.

✓ **Correct**

Correct. This is a more subtle form of denormalization that can help with some practical database performance needs. Of course, if you keep a separate summary table, then you must give extra attention to making sure that the summary table is kept up to date with the detail data that it summarizes.

7. Which of these accurately describe why features of operational databases are not needed for analytic databases? **1 / 1 point**

- ☑ Analytic databases often use data collected from other sources (including other operational databases), so enforcing business rules is typically not needed.

✓ **Correct**

Correct. Operational databases are more often used for the collection of data, and business rules are enforced during that collection process.

- ☐ Analytic databases only handle very simple data, so there are more efficient methods to correct inaccurate data than enforcing business rules.
- ☐ Analytic databases are more focused on CRUD type activities
- ☐ Analytic databases are used for complex queries; since triggers cause queries to run much more slowly, they should be handled in a different way.
- ☑ Analytic databases update infrequently so ETL (extract, transform, and load) utilities can replace many of the DML features of operational databases.

✓ **Correct**

Correct. Even if analytic databases are not completely static, they often are updated only once a day or even less frequently than that. ETL utilities or scripts can do these updates overnight.