

## Tutorial Quiz 1

● Graded

### Student

Lihang Shen

### Total Points

6 / 10 pts

### Question 1

Review MCQ Questions

4 / 6 pts

#### 1.1 Q1a: DS Hierarchy of Needs

0 / 1 pt

✓ + 0 pts Incorrect option selected

#### 1.2 Q1b: Data Acquisition Latency

1 / 1 pt

✓ + 1 pt Correct

#### 1.3 Q1c: Semistructured data

1 / 1 pt

✓ + 1 pt Correct

#### 1.4 Q1d: Schema-first vs Schema-late

1 / 1 pt

✓ + 1 pt Correct

#### 1.5 Q1e: Geometry vs Geography

0 / 1 pt

✓ + 0 pts Incorrect option selected

#### 1.6 Q1f: SQL query

1 / 1 pt

✓ + 1 pt Correct

### Question 2

#### Q2: Star Schema

0 / 2 pts

✓ + 0.5 pts Explanation given

✓ - 0.5 pts some incorrect or unsuitable table given

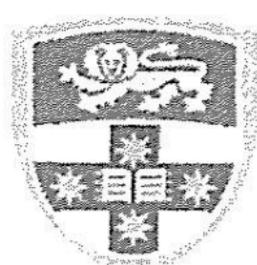
### Question 3

#### Q3: Storage Layer Design

2 / 2 pts

✓ + 1 pt Suitable storage layer approach selected

✓ + 1 pt Explanation with difference to other alternatives given



Tutorial Class	22
Student Name	Lihang Shen
Student Number	490051481

**CONFIDENTIAL QUIZ PAPER**

This paper is not to be removed from the quiz venue

**COMP5339 Data Engineering****TUTORIAL QUIZ 1**

Individual Quiz (5%)

**QUIZ WRITING TIME: 30 minutes**

**INSTRUCTIONS TO STUDENTS:**

Please make sure you fill in your tutorial and Student ID correctly.

This is a Closed Book test: electronic devices (including phones, laptops, tablets, etc.) are not allowed - Please place them in your bag on the floor.

This tutorial quiz consists of **3 QUESTIONS** with the first one consisting of 6 multiple-choice questions. All questions must be answered.

- Answer all questions within the spaces provided on this paper.
- Note that questions are of unequal value. The points for each question are shown at the start of the question.  
The total mark of this quiz paper is **10**.
- For short answer questions, take care to write legibly. Write your final answers in ink.  
*Do not use a pencil or red ink.*

Please hand the completed quiz paper to the tutor before you leave the room.

**This paper is not to be removed from the quiz venue.**

**Question 1: Review Questions****[6 points]**This question has six (6) parts, ((a)—(f)). Tick the box (or boxes) of each correct answer.

- (a) [1 point] Which **one** of the following sequences best describes the "Data Science Hierarchy of Needs"?

- Model → Data → Visualization → Storage X
- ETL → Warehousing → Analytics → Prediction X
- Data Collection → Storage → Cleaning → Analysis → AI X
- Data → Information → Knowledge → Wisdom X ✓

- (b) [1 point] Which **one** of the following statements is correct about the importance of latency to data acquisition?

- For one-off data acquisition tasks, high latency is important. X
- Data latency is becoming the more important as data volumes grow. X *one-off*
- Low latency is important for file-based data acquisition with very large files. X *one-off*
- Latency considerations are important when working with unbounded data streams or change sets. ✓

- (c) [1 point] Which **one** of the following statements about semi-structured data is FALSE?

- Semi-structured data can have missing or additional attributes. ✓
- All records in a semi-structured data collection must follow a fixed schema. X
- JSON and XML are common formats for semi-structured data.
- Semi-structured data often uses a hierarchical or tree-structured model.

- (d) [1 point] Which **one** of the following statements is correct about the difference of 'schema-first' versus 'schema-late'?

- Schema-late means that when creating a star schema in a data warehouse, the schemas of the source databases have to be created schema-first.
- For relational databases, a schema needs to be created first before data can be loaded; in contrast, semi-structured data allows to use schema-late which interprets the data directly on read. ✓
- Schema-first is used by MongoDB, while PostgreSQL uses a schema-late approach. X
- With schema-first, queries have to first extract the meta-data from records before they can be interpreted. A SELECT \* allows to do schema-late. X

- (e) [1 point] Which **one** of the following statements is correct about the difference between Geometry and Geography data types in PostGIS?

- Geography types only allow to store 2D points, lines and polygons, while Geometry allows shapes with 3D coordinates. ?
- Geometry types assume Cartesian coordinates on a flat 2D plane, while Geography types support geodetic data with a spherical model for, e.g., distance and area calculations. X
- Geometry types are used for global analysis, while Geography types support local analysis such as area calculations.
- Geography types can be indexed using a GiST index in PostgreSQL, while Geometry is limited to B+ tree indexing.

(f) [1 point] Given the table names with their attributes.

- Stations(stationid INT, siteName VARCHAR, orga VARCHAR)
- Measurements(obsdate DATE, obsvalue FLOAT, stationid INT, sensor VARCHAR)

What is the effect of the following SQL query?

```
SELECT orga
  FROM Stations LEFT JOIN Measurements USING (stationid)
 GROUP BY orga
 HAVING COUNT(obsvalue) > 0;
```

- It lists all organisations with at least one known measurement, ignoring stations without measurements. ✓
- It lists only organisations from stations with no measurements at all. ✗
- It lists all organisations, including those without measurements, but excludes any organisations where obsvalue is negative or 0. ✗
- It lists all organisation names, including those with only NULL measurements.

#### Question 2: Star Schema Scenario

[2 points]

Suppose you are a data engineer at a university tasked with designing a data warehouse for analyzing student performance across departments and semesters. The goal is to allow analysts to efficiently answer questions such as "What is the average grade per department per semester?" Student grades in courses are recorded, and there are additional tables for student details, course details, department information, and semester dates.

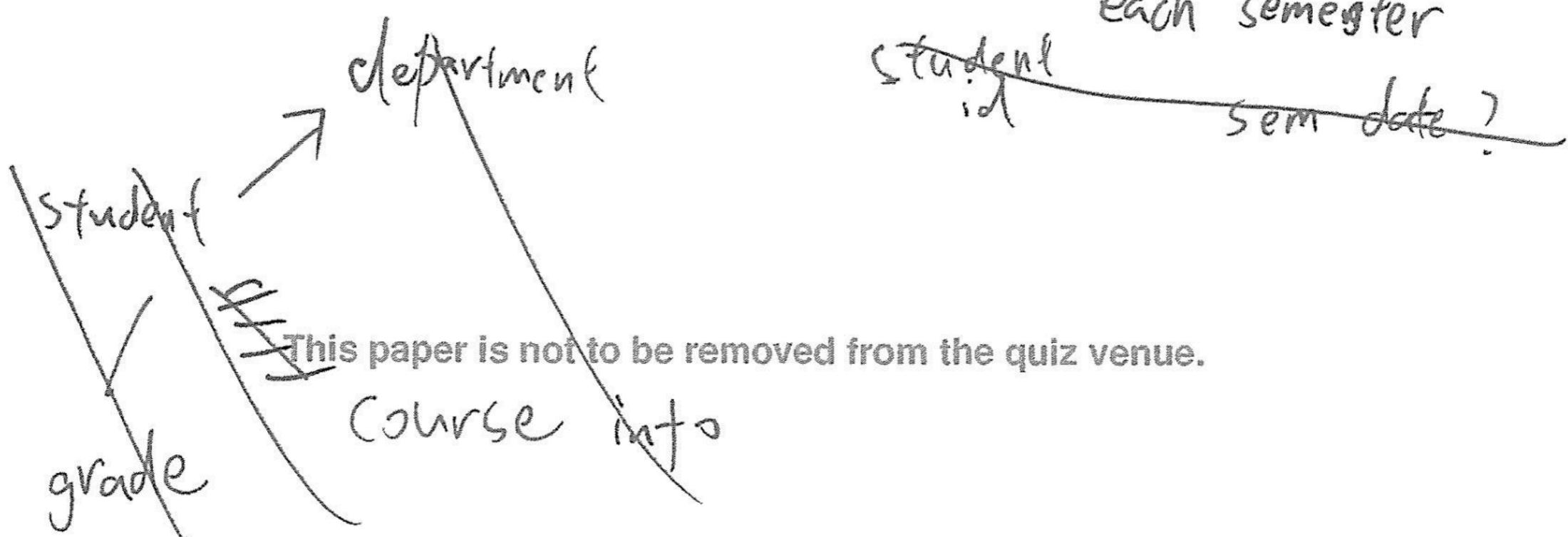
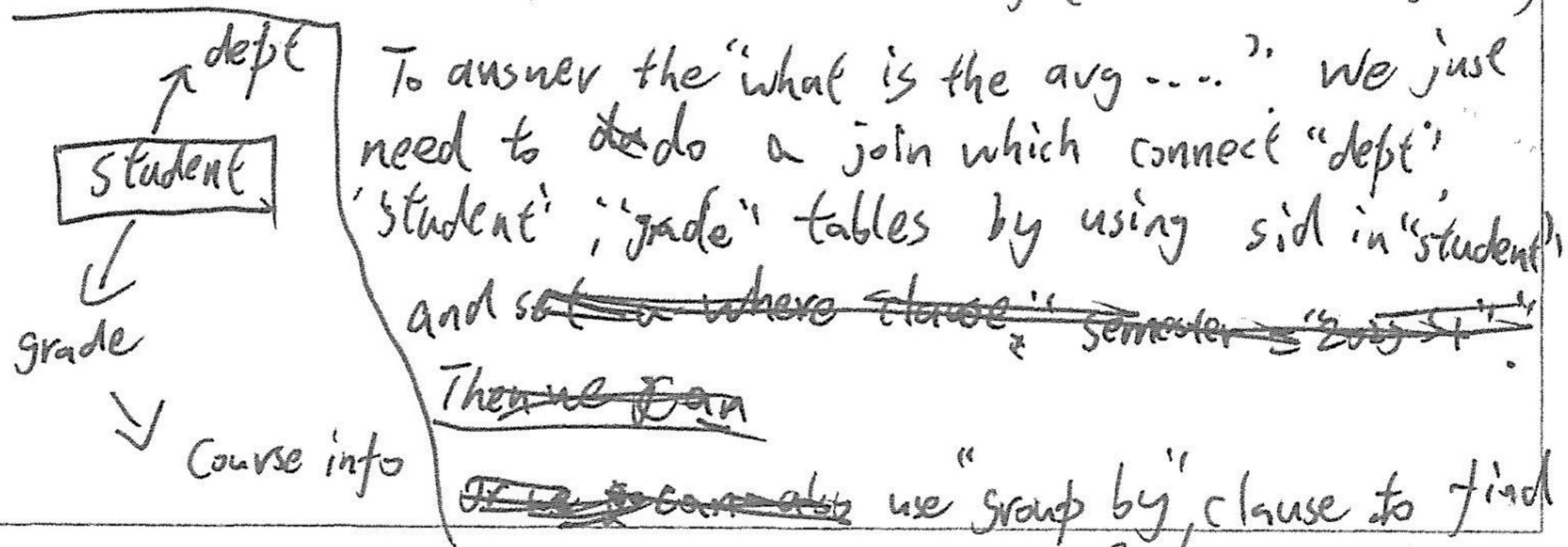
Grade 必须放在事实表 (Fact Table)

(a) [2 points] Briefly describe how you would model this warehouse using a **star schema**.

Identify the fact table and at least three dimension tables. Explain how this schema supports efficient analysis of metrics like the average grade per department per semester.

The fact table should be student details, ~~such as~~ student id .col, <sup>include</sup> it has relationship with all other tables.

The dim tables are: department info, grade (with one col about semester info), and course info (such as unit code)



**Question 3: Storage Layer Design [2 points]**

Suppose you are a data engineer tasked with designing a data pipeline for analysing rental data across Australia. This rental data consists of some variable information about the rented property depending on each property type. For example, while for a rented apartment we mainly know the number of rooms, kitchen and bathrooms, for a rented house the information in addition would include overall land and garden size, and potential external structures such as garages or sheds. For each rental record, you also need to keep the rental price, the payment frequency (e.g. weekly or monthly), the rental bond, as well as the address and geo-location of the property.

- (a) [2 points] What kind of storage layer would you suggest for this scenario? Would you use a relational database, potentially with some extensions, or a JSON-document NoSQL database, or a graph database? Briefly explain your choice and why you consider it better suited than the alternatives.

We need to use semi-structured data, since ~~the~~ "potential external structures".

- Since Json is low-overhead, so I would prefer use a Json-doc NoSQL dB
- For Apartment, house, although they are all building, but their structure is vary so is very difficult to do a 1-to-1 mapping for RDBMS. For example, some house has garden, some doesn't then we might need to use null to fill the empty col. In addition, we need to consider about Scalability, since there are some 'unique' building type which might only appear rarely, but we still need to add them, and Json can store them without make big change to table head (since schema-on-read).