EIE3112 Test 1 (2021/22 sem 2)

Name: Sze Kin Sang

Student ID: 19062606D

# Instructions:

* Time allowed: 13:45-15:15 (90 minutes, including the time for uploading the answer).
* You may write your answer on a piece of paper or use MySQL Workbench to draw the ERD or use text editor to type your answer below. (You may need to consider that there may be technical problem encountered using MySQL Workbench or text editor. This cannot be the reason for exemption from late submission penalty.)
* Then upload your answer to Blackboard before the submission deadline.
* You have approximately 85 minutes to work on the answer and another 5 minutes for uploading the answer. Please take your time.
* Late submission will be subject to mark deduction. No score will be given to the submission uploaded 10 minutes after the submission deadline.
* If you encounter any problem during the test, please contact Dr Pauli Lai at +852-27666217 or pauli.lai@polyu.edu.hk.
* Please pay attention to the Blackboard announcement or email for any updated information on the question (if any).
* WARNING: If you are found sharing or copying the question and answer with or from your classmate, both you and your classmate will score 0 mark due to plagiarism.

# Part 1 ERD (25%)

### *UPS* Case Study

UPS prides itself on having up-to-date information on the processing and current location of each shipped item. To do this, UPS relies on a company-wide information system. Shipped items are the heart of the UPS product tracking information system. Shipped items can be characterized by item number (unique), weight, dimensions, insurance amount, destination, and final delivery date. Shipped items are received from a single retail center while each retail center can send one or more shipped items. Retail centers are characterized by their type, unique ID, and address. Shipped items make their way to their destination via one or more standard UPS transportation events (i.e., flights, truck deliveries). These transportation events are characterized by a unique scheduleNumber, a type (e.g, flight, truck), and a deliveryRoute. Each transportation event can deliver one or more shipped items.

Draw an entity-relationship diagram that describes the above business environment. Underline the primary key in each entity or indicate the primary key with key icon in case of using MySQL Workbench. You should also include foreign key(s) in the entity. Use dotted line to underline the foreign key in each entity or indicate the foreign key with red icon in case of using MySQL Workbench. No need to specify the data type for each attribute. In case of using MySQL Workbench, data type will be ignored for grading. For relationship, you may draw solid line irrespective of non-identifying or identifying relationship. For many-to-many relationship, you should convert it into two one-to-many relationships. If there is anything that the question does not specify clearly, you may write down your assumption.

|  |
| --- |
| Assuming a retail centre has at least sent one item. |

# Part 2 Normalization (30%)

*An agency called Instant Cover supplies part-time/temporary staff to hotels within Scotland. The table shown in Figure 1 below lists the time spent by agency staff working at various hotels. The National Insurance Number (NIN) is unique for every member of staff.*

一張含有 桌 的圖片

自動產生的描述

Figure 1 Tabel displaying sample data for the Instant Cover agency.

(a) *The**table shown in Figure 1 is susceptible to update anomalies. Provide examples of insertion, deletion, and update anomalies.*

Insertion Anomalies: When a new hotel joins the service and is added to the table, the NIN is stilled required even the agency has sent any staff to the hotel.

Deletion Anomalies: When row NIN 1057 is deleted, the information of staff Hocine D will be lost.

Modification Anomalies: When a hotel’s location is changed, all the data in hLoc column should be manually changed, corresponding to the hNo.

(6%)

(b) *Identify the functional dependencies in Figure 1.*

hNo -> hLoc

contractNo -> hNo, hLoc

NIN -> eName

NIN, contractNo -> hours

(4%)

(c) ***Describe*** *and illustrate the process of normalizing the table shown in Figure 1 to 3NF. You need to state the reason why you normalize the table from one normal form to another. Please use solid line to underline the primary key and dotted line to underline the foreign key for* ***ALL tables****. State any assumptions you make about the data shown in this table, if any.*

(20%)

|  |
| --- |
| UNF -> 1NF:  Underline all the key in the table.  Therefore, NIN, contractNo and hNo is underlined.  1NF -> 2NF:  According to the functional dependencies, three tables can be developed.  Contracts(contractNo, hNo, hLoc)  Staff(NIN, eName)  ContractDetails(contractNo, NIN, hours)  2NF -> 3NF:  To eliminate transitive dependency, a new Hotels table should be created since hLoc depends on contractNo via hNo  Contracts(contractNo, hNo)  Hotels(hNo, hLoc)  Staff(NIN, eName)  ContractDetails(contractNo, NIN, hours) |

# Part 3 Stored Procedure + Trigger (45%)

The following tables form part of a database held in a relational DBMS:

Hotel (hotelNo, hotelName, city)

Room (roomNo, hotelNo, type, price)

Booking (hotelNo, guestNo, dateFrom, dateTo, roomNo)

Guest (guestNo, guestName, guestAddress)

where Hotel contains hotel details and hotelNo is the primary key;

Room contains room details for each hotel and (roomNo, hotelNo) forms the primary key;

Booking contains details of the bookings and (hotelNo, guestNo, dateFrom) forms the primary key;

and Guest contains guest details and guestNo is the primary key.

(a) The hotel wants to set a minimum price for all double rooms. Create a stored procedure named *Check\_Price* with an input parameter *min\_price* of integer type. The stored procedure checks whether the price of each double room (type='D') is smaller than *min\_price*. If so, the price of that double room would be updated to *min\_price*. You may assume that roomNo, hotelNo and price are integers while type is a character. Note that you need to change the delimiter at the beginning and at the end. Also, you need to remove the existing procedure first if there exists one.

(30%)

|  |
| --- |
| DELIMITER $$  DROP PROCEDURE IF EXISTS Check\_Price$$  CREATE PROCEDURE Check\_Price(IN min\_price INT)  BEGIN  DECLARE p\_finished INT DEFAULT 0;  DECLARE p\_price INT DEFAULT 0;  DECLARE p\_roomNo INT DEFAULT 0;  DECLARE p\_hotelNo INT DEFAULT 0;  DECLARE price\_cursor CURSOR FOR SELECT price, roomNo, hotelNo FROM Room WHERE type = ‘D’;  DECLARE CONTINUE HANDLER FOR NOT FOUND SET p\_finished = 1;  OPEN price\_cursor;  get\_price: LOOP  FETCH price\_cursor INTO p\_price, p\_roomNo, p\_hotelNo;  IF p\_finished = 1 THEN LEAVE get\_price;  END IF;  CASE  WHEN p\_price <= min\_price  THEN UPDATE Room SET price = min\_price WHERE roomNo = p\_roomNo AND hotelNo = p\_hotelNo;  END CASE;  END LOOP get\_price;  CLOSE price\_cursor;  END$$  DELIMITER ; |

(b) Invoke the procedure from MySQL with a minimum price of 100.

(2%)

CALL Check\_Price(100);

SELECT \* FROM Room WHERE type = ‘D’;

(c) The hotel wants to ensure that the minimum price of a new double room is at least 100. Create a database trigger named *before\_room\_insert* for the following situation:

When a new record of a double room is inserted, check whether the room price is smaller than 100. If so, set the new room price to 100. Note that you need to change the delimiter at the beginning and at the end. Also, you need to remove the existing trigger first if there exists one.

(13%)

|  |
| --- |
| DELIMITER $$  DROP TRIGGER IF EXISTS before\_room\_insert$$  CREATE TRIGGER before\_room\_insert  BEFORE INSERT ON Room  FOR EACH ROW BEGIN  DECLARE price\_insert INT DEFAULT 0;  DECLARE type\_insert VARCHAR(5);  SET price\_insert = NEW.price;  SET type\_insert = NEW.type;  IF price\_insert < 100 AND type\_insert = ‘D’  THEN SET NEW.price = 100;  END IF;  END$$  DELIMITER ;  (forget to select type D) |