Programme Code: TU856/DT228, TU857/DT211C, TU858/DT282

Module Code: CMPU 4003

CRN: 30088, 22415, 31081

**TECHNOLOGICAL UNIVERSITY DUBLIN**

**CITY CAMPUS**

**\_\_\_\_\_\_\_\_\_\_\_\_\_**

TU856/DT228 BSc in Computer Science

TU857/DT211C BSc in Computer Science (Infrastructure)

TU858/DT282 BSc in Computer Science (International)

**Year 4**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

SEMESTER 1 EXAMINATIONS 2021/2022

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Advanced Databases**

**Internal Examiners:**

Deirdre Lawless

Giulia Vilona

Dr. Paul Doyle

**External Examiners:**

Sanita Tifentale – TU856/DT228

Pauline Martin – TU857/DT211C

Pamela O’Brien – TU858/DT282

***Instructions***

**Answer any FOUR questions.**

**All questions carry equal marks.**

|  |  |  |
| --- | --- | --- |
| 1. | (a) | Explain the key differences between normalization and denormalization.  (4 marks) |
|  | (b) | Explain *FOUR (4)* possible benefits of denormalization.  (4 marks) |
|  | (c) | A gardening service company (GSC) undertakes a range of tasks for clients and supplies the products used during tasks e.g., weedkiller, plants, seeds etc.). Consider the following ER Schema (primary keys are underlined, foreign keys are indicated by the suffix (FK)): |
|  | **Client**(clientId, clientName, clientAddress, phone, email, contact\_person)  **GardenStaff**(staffId, first\_name, last\_name)  **Garden\_task**(taskId, clientId (FK), startTime, endTime, staffId (FK), success\_or\_fail)  **Product**(productId, productName, unit, pricePerUnit)  **Product\_supplied**(psId, productid (FK), taskId (FK), units)  **Product\_offered**(poId, productid (FK), taskId (FK), units)  **Product\_writtenoff**(woId, woDate, productId (FK), units, wo\_reason)  **Products\_on\_order**(orderId, productId (FK), suppId (FK), units)  **Supplier**(suppId, suppName, suppAddress, phone, email, contact\_person)  GSC can undertake many tasks for a client. Each client nominates a contact person, but this may change over time. For each task, GSC recommends a number of products to the client (product\_offered) and records the products the client actually selects (product\_supplied). When a task is completed, the client indicates whether they were satisfied with the service or not. GSC maintains a store of products they use and regularly needs to dispose of products that are no longer usable. This represents a loss to GSC and details are recorded (product\_writtenoff). Products need to be replaced regularly. Details of any orders placed with suppliers are recorded (products\_on\_order). GSC would like to be able to generate reports on their annual performance. | |
|  |  | 1. Provide an example of how denormalization could be applied to this schema to achieve each of the possible benefits you identified in part (b).   (4 x 3 marks)   1. What are the possible negative implications of the examples you provided in your answer to part (c) (i)?   (5 marks) |

|  |  |  |
| --- | --- | --- |
| 2. | (a) | Explain *FOUR (4)* possible uses of triggers in a relational DBMS.  (4 marks) |
|  | (b) | Consider the following trigger and data: |
| |  |  |  |  | | --- | --- | --- | --- | | **Customers** | | | | | **cust\_id** | **custname** | **Creditlimit** | **VIP** | | 1 | cust1 | 662 | Y | | 2 | cust2 | 703 | N | | 8 | cust8 | 651 | N | | 10 | cust10 | 815 | Y | | | |
|  |  | 1. For this trigger, identify and explain the purpose of each of the following:  * Trigger statement * Trigger restriction * Trigger action   (3 x 2 marks)   1. Explain what would happen if each of the following SQL statements were executed against the data shown above:  * update customers set creditlimit=1500 where cust\_id=2; * update customers set creditlimit=1600 where cust\_id=10; * update customers set creditlimit=2000 where cust\_id=1;   (3 x 2 marks) |
|  | (c) | Write a trigger to ensure that whenever a record is inserted into a table called sales, a record is created in a sales\_log table recording the username that executed the SQL to create record in the sales table and the date and time at which the SQL to create the sales\_log record was executed.  Note: You can retrieve the username and current date and time from dual e.g. SELECT user FROM dual; select systimestamp from dual;  (9 marks) |
| 3. | (a) | 1. Explain how each of the following works in Oracle’s DBMS and when they   would be used:   1. Full index scan 2. Fast full index scan 3. Index range scan   (3 x 2 marks) |
|  |  | 1. Consider the following scenarios. For each, explain which type of scan listed in part (i) will be used by the Oracle DBMS and why:  * Suppose the last\_name column has a not null constraint and the last name and clientFee are a composite key in an index and the query SELECT last\_name, clientFee FROM clients; is executed. * Suppose that department\_id, last\_name, and salary are a composite key in an index and the query SELECT department\_id, last\_name, clientFee FROM clients WHERE clientFee > 5000 ORDER BY department\_id, last\_name; is executed. * Suppose the last\_name column is indexed and the query SELECT last\_name FROM clients WHERE last\_name like ‘A%’; is executed.   (3 x 3 marks) |
|  | (b) | 1. Explain *THREE (3)* types of index that can be implemented in a MongoDB.   (3 x 2 marks)   1. Suppose we have a collection student containing the following type of document:   {  "\_id": 1,  "person": { name: "Jane", surname: "Byrne" },  "age": 20,  "city": "Waterford"  }  Provide the code to create an index which would improve queries on name, surname and city.  (4 marks) |

|  |  |  |
| --- | --- | --- |
| 4. | Consider the case of a real estate agency whose database is composed by the following tables:  **OWNER** (IDOwner, Name, Surname, Address, City, Phone)  **PROPERTY** (IDProperty, IDOwner, PropertyCategory, Area, City, Province, Rooms, Bedrooms, Garage, Meters)  **CUSTOMER** (IDCust, Name, Surname, Budget, Address, City, Phone)  **AGENT** (IDAgent, Name, Surname, Oﬃce, Address, City, Phone)  **AGENDA** (IDAgent, Data, Hour, IDEstate, ClientName)  **VISIT** (IDProperty,IDAgent, IDCust, Date, Duration)  **SALE** (IDProperty,IDAgent, IDCust,Date, AgreedPrice, Status)  **RENT** (IDProperty,IDAgent, IDCust, Date,Price, Status, Time)  Your goal is to define a data warehouse to provide a supervisor with an overview of the situation. The supervisor must have a global view of the business, in terms of the properties the agency deals with and of the agents’ work. The supervisor is very interested in the sales trend and controlling agents’ time.  Here are some examples of queries it should be possible to execute:   * How many customers have visited properties of at least 3 diﬀerent categories? * What is the average duration of visits per property category? * Who has paid the highest price among the customers that have viewed properties of at least 3 diﬀerent categories? * Who has bought a ﬂat for the highest price each month? * What kind of property sold for the highest price in each city and month? | |
|  | (a) | 1. Provide a star schema which will achieve the goals outlined.   (15 marks)   1. Explain how your schema will facilitate the queries identified.   (5 marks) |
|  | (b) | Write the SQL needed to allow a manager to identify the average duration of visits per property category?  (5 marks) |

|  |  |  |
| --- | --- | --- |
| 5. | You are required to design a database to store information about army soldiers and their chain of command. Each soldier is described by a soldier\_id, a soldier name and a salary. There is a relationship among the soldiers: soldier *a* is linked to soldier *b* if a is *b*’s captain. The date soldier *a* became *b*’s captain is also stored. | |
|  | (a) | Provide a relational schema to show how to store soldiers and their relationship. Provide tables, fields and show primary and foreign keys.  (4 marks) |
|  | (b) | Write an SQL query to get all the list of soldiers directly commanded by the soldier with id=3.  (4 marks) |
|  | (c) | Write an SQL query to get the list of soldiers commanded by soldier with id=3 directly or commanded by soldiers managed by soldier with id=3.  (5 marks) |
|  | (d) | Provide a json structure to store the same information provided in the relational model into a *mongodb* collection.  (4 marks) |
|  | (e) | Compare the two data models: which one is easier to query?  Is standard SQL a sustainable way to perform such queries? What could be a better solution?  (4 marks) |
|  | (f) | Show how the same information would be stored in a graph database.  (4 marks) |