**UNIVERSITI TUNKU ABDUL RAHMAN**



**Faculty of Information and Communication Technology (FICT)**

**UCCD2203 DATABASE SYSTEMS**

**Group Assignment**

**(individual submission)**

**Session 202101**

Deadline: Saturday 27 March 2021 (Week 10)

Time: before 5.00 pm

Submission channel: a hyperlink on WBLE

|  |  |
| --- | --- |
| **Programme (**IA/IB/CS/CN/CT**):** | **CS** |
| **Group Number (example G01) :** | **G49** |
| **Group leader name:** | **TAN XI EN** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | **Name**  **(in ascending order)** | **Student ID** | **UTAR email address** | **Practical Group** | **Signature\*\*** |
| 1 | TAN XI EN | 1904098 | txen2000@1utar.my | P (6) |  |
| 2 | YEOH SIOW POO | 1802886 | ysiowpoo@1utar.my | P (6) |  |
| 3 | FELIX LEN WEN JUN | 1901121 | lenfelix2808@1utar.my | P (7) |  |
| 4 | YEW JEI WEY | 1802520 | jeiwey@1utar.my | P (7) |  |

Note:

\*\*All members should attach their individual signature confirming that the report is not plagiarized

|  |  |  |
| --- | --- | --- |
| **Student ID**  **As appeared on student card** | **1904098** |  |
| **Member name As appeared on student card** | **Tan Xi En** |  |
| **Queries**  **(30 marks)** | **The following items shall be placed in this column:**  **Transaction / question**  **Your SQL command/ your answers – as appear in your submitted \*.sql**  **Partial / full OUTPUT screenshots** | **Leave this column empty** |
| **Query 1** | The following Query is to show all staff member with the following job position: Doctor, Nurses and Admission Staff  **SQL Statement:**  select      staff\_id "ID",      staff\_lname || ' ' || staff\_fname "Name",      staff\_position "Job Title"  from      staff  where      staff\_position in ('Doctor', 'Nurse', 'Admission Staff');  **Expected Output:** |  |
| **Query 2** | The following query is to show all staff member with less than 3 years of experience.  **SQL Statement:**  SELECT      s.staff\_id "ID",      s.staff\_lname || ' ' || s.staff\_fname "Name",      s.staff\_position "Job Title",      s.staff\_salary "Salary",      d.department\_name "Department"  FROM      staff s,      department d  WHERE      s.department\_id = d.department\_id      AND TRUNC((sysdate - staff\_hiredate) / 365.25) < 3;  **Expected Output:** |  |
| **Query 3** | The following Query is to select all consultation with its respective date and time, doctor info and patient Info that lands on a specific day of the week. For this example, we shall select all consultation that lies on Tuesday.  **SQL Statement:**  SELECT      s.staff\_lname || ' ' || s.staff\_fname "Doctor",      p.patient\_lname || ' ' || p.patient\_fname "Patient",      TO\_CHAR(          CAST(cs.consultation\_startdatetime AS DATE),          'DD/MM/YYYY'      ) "Date",      TO\_CHAR(cs.consultation\_startdatetime, 'HH24:MI:SS') "Start",      TO\_CHAR(cs.consultation\_enddatetime, 'HH24:MI:SS') "End",      r.medical\_disease "Illness"  FROM      consultation cs,      staff s,      patient p,      report r  WHERE      s.staff\_id = cs.staff\_id      AND cs.patient\_id = p.patient\_id      AND cs.consultation\_id = r.consultation\_id      AND TRIM(          TO\_CHAR(              CAST(cs.consultation\_startdatetime AS DATE),              'DAY'          )      ) = UPPER('tuesday')  ORDER BY      cs.consultation\_startdatetime;  **Expected Output:** |  |
| **Query 4** | The following Query groups all staff members by its department, calculate the total salary cost for each department per month, list out all its staff members and the number of staff member in each department.  **SQL Statement:**  col "List of Doctors" format a80  col "Total Expenses" format a20  SELECT      department.department\_id "ID",      department.department\_name "Name",      tmp.num\_of\_employees "Number of employees",      tmp.names "List of Doctors",      'RM' || ' ' || tmp.total\_expenses "Total Expenses"  FROM      department,      (          SELECT              department\_id,              TO\_CHAR(SUM(staff\_salary), '999,999.99') total\_expenses,              COUNT(department\_id) num\_of\_employees,              LISTAGG(staff\_lname || ' ' || staff\_fname, ', ') WITHIN GROUP (                  ORDER BY                      staff\_lname              ) names          FROM              staff          GROUP BY              department\_id      ) tmp  WHERE      tmp.department\_id = department.department\_id;  **Expected Output:** |  |
| **Query 5** | The following Query Creates a View for Nurse Department. They are only able to insert new Staff with the job position Nurse into the staff table.  **SQL Statement:**  CREATE OR replace VIEW nurse\_view AS  SELECT      \*  FROM      staff  WHERE      staff\_position = 'Nurse'  WITH CHECK OPTION;  **Expected Output:**  /\*This is Allowed\*/  INSERT INTO      nurse\_view  VALUES      (          'S1016', 'Ying Qian 23', 'Kang234', '900224-08-3434',          TO\_DATE('02/24/1990', 'MM/DD/YYYY'), 'Female', '0124567833', 'yqian@gmail.com',          TO\_DATE('12/31/2020', 'MM/DD/YYYY'), 'Nurse', '26, Taman Baru', 'Parit Buntar',          34200, 'Perak', 'Malaysia', 4500.00, 'D1001'      );    A new Nurse is inserted into nurse view.  /\*This is not Allowed\*/  INSERT INTO      nurse\_view  VALUES      (          'S1017', 'Ying Qian 23234', 'Kang234234', '900224-08-3434',          TO\_DATE('02/24/1990', 'MM/DD/YYYY'), 'Female', '0124567833', 'yqian@gmail.com',          TO\_DATE('12/31/2020', 'MM/DD/YYYY'), 'Doctor', '26, Taman Baru', 'Parit Buntar',          34200, 'Perak', 'Malaysia', 4500.00, 'D1003'      ); |  |
| **Query 6** | The following query creates a view to link patient and the medicine the doctor prescribed for the patient. It is read only, so no modification can be made to the view.  **SQL Statement:**  CREATE OR replace VIEW patient\_medicine\_view AS  SELECT      p.patient\_lname || ' ' || p.patient\_fname "Name",      r.medical\_disease "Illness",      m.medicine\_name "Medicine Name",      md.patmed\_quantity "Quantity",      'RM ' || TO\_CHAR(          md.patmed\_quantity \* m.medicine\_unitprice,          '9,999.99'      ) "Medicine Fee"  FROM      patient p,      consultation cs,      report r,      medicine\_distribution md,      medicine m  WHERE      p.patient\_id = cs.patient\_id      AND cs.consultation\_id = r.consultation\_id      AND r.report\_id = md.report\_id      AND md.medicine\_id = m.medicine\_id  WITH READ only;  SELECT      \*  FROM      patient\_medicine\_view;  **Expected Output:** |  |
| **Query 7** | The following query creates a view that link between Doctor, consultation, patient and his/her report. It is read only, so no modification can be made to the view  **SQL Statement:**  CREATE OR replace VIEW doctor\_patient\_view AS  SELECT      p.patient\_lname || ' ' || p.patient\_fname patient\_name,      s.staff\_lname || ' ' || s.staff\_fname doctor\_name,      d.doctor\_specialty job\_specific,      r.medical\_disease illness  FROM      staff s,      doctor d,      consultation cs,      report r,      patient p  WHERE      s.staff\_id = d.staff\_id      AND d.staff\_id = cs.staff\_id      AND cs.patient\_id = p.patient\_id      AND r.consultation\_id = cs.consultation\_id  WITH READ only;  SELECT      \*  FROM      doctor\_patient\_view;  **Expected Output:** |  |
| **Query 8** | The following query groups the patient by its respective doctor. It shows the number of patient each doctor have, and its patient name.  \*Note: Please run query 7 first, before running this\*  **SQL Statement:**  col "Patients" format a50  SELECT      tmp.doctor\_name "Doctor Name",      LISTAGG(patient\_name, ', ') WITHIN GROUP (          ORDER BY              patient\_name      ) "Patients",      COUNT(patient\_name) "Number of Patients"  FROM      doctor\_patient\_view tmp  GROUP BY      tmp.doctor\_name;  **Expected Output:** |  |
| **Query 9** | The following Query groups the consultation available by the day of the week. It is easy to plan timetable using this query  **SQL Statement:**  col "Date Consultation" format a20  col "Consultations" format a30  SELECT      TRUNC(CAST(consultation\_startdatetime AS DATE)) "Date Consultation",      LISTAGG(consultation\_id, ', ') WITHIN GROUP (          ORDER BY              consultation\_id      ) "Consultations",      COUNT(TRUNC(CAST(consultation\_startdatetime AS DATE))) "Number of Consultation"  FROM      consultation  GROUP BY      TRUNC(CAST(consultation\_startdatetime AS DATE))  ORDER BY      1;  **Expected Output:** |  |
| **Query 10** | The following query groups it staff members by its birthday. It is easy to plan birthday for staff using this query.  **SQL Statement:**  col "Name" format a50;  SELECT      TRIM(to\_char(staff\_dob, 'Month')) birthday,      LISTAGG(staff\_lname || ' ' || staff\_fname, ', ') WITHIN GROUP (          ORDER BY              staff\_lname      ) "Name"  FROM      staff  GROUP BY      trim(to\_char(staff\_dob, 'Month'))  ORDER BY      to\_char(to\_date(birthday, 'Month'), 'mm');  **Expected Output:** |  |
| **Stored**  **Procedure**  **(10 marks)** |  |  |
| **SP1** | The following Procedure is used to delete staff based on staff ID.  **SQL Statement:**  CREATE OR REPLACE PROCEDURE DELETE\_STAFF(      current\_s\_id IN varchar2  ) IS BEGIN  DELETE FROM      staff  WHERE      staff\_id = current\_s\_id;  END;  /  EXECUTE DELETE\_STAFF('S1013');  rollback;  **Before:**    **After:** |  |
| **SP2** | The following procedure deletes the department from the database records. It deletes all staff within the respective department.  **SQL Statement:**  CREATE OR REPLACE PROCEDURE DELETE\_DEPARTMENT(      current\_dept\_name IN varchar2  ) AS BEGIN  FOR staff\_ptr IN (      SELECT          s.staff\_id      FROM          staff s,          department d      WHERE          s.department\_id = d.department\_id          AND d.department\_name = current\_dept\_name  ) loop DELETE\_STAFF(staff\_ptr.staff\_id);END loop;  DELETE FROM      department  WHERE      department\_name = current\_dept\_name;  END;  /  EXECUTE DELETE\_DEPARTMENT('Admission');  rollback;  **Before:**    **After:** |  |
| **SP3** | The following procedure gives bonus salary of any amount to any staff with the respective staff\_id. The staff with the respective staff\_id will have his/her salary increased by the bonus amount.  **SQL Statement:**  CREATE OR REPLACE PROCEDURE GIVE\_BONUS\_STAFF(      current\_s\_id IN varchar2,      bonus\_salary IN decimal  ) IS BEGIN  UPDATE      staff  SET      staff\_salary = staff\_salary + bonus\_salary  WHERE      staff\_id = current\_s\_id;  END;  /  EXECUTE GIVE\_BONUS\_STAFF('S1013', 500.00);  rollback;  **Before:**    **After:** |  |
| **SP4** | The following procedure gives every staff member within the department a bonus salary. The salary of the staff within the respective department is increased by the bonus salary.  **SQL Statement:**  CREATE OR REPLACE PROCEDURE GIVE\_BONUS\_DEPARTMENT(      current\_dept\_name IN varchar2,      bonus\_salary IN decimal  ) AS BEGIN FOR staff\_ptr IN (      SELECT          s.staff\_id      FROM          staff s,          department d      WHERE          s.department\_id = d.department\_id          AND d.department\_name = current\_dept\_name  )  loop  GIVE\_BONUS\_STAFF(staff\_ptr.staff\_id, bonus\_salary);  END loop;  END;  /  EXECUTE GIVE\_BONUS\_DEPARTMENT('Admission', 500.00);  rollback;  **Before:**    **After:** |  |
| **SP5** | The following Procedure insert a new staff into the staff table. A few of the parameters are simplified. For example, the user no longer needs to input the id of the staff. Based on the position of the staff, the department\_id and the salary is decided as well. Lastly, the system will split the address into addressline, city, postcode, state and country.  **SQL Statement:**  CREATE OR REPLACE PROCEDURE INSERT\_STAFF(      stf\_fname IN VARCHAR2,      stf\_lname IN VARCHAR2,      stf\_ssn IN VARCHAR2,      stf\_dob IN VARCHAR2,      stf\_gender IN VARCHAR2,      stf\_phoneNo IN VARCHAR2,      stf\_email IN VARCHAR2,      stf\_position IN VARCHAR2,      address IN VARCHAR2  ) IS  addressline VARCHAR2(100);  city VARCHAR2(100);  postcode VARCHAR2(100);  state VARCHAR2(100);  country VARCHAR2(100);  salary DECIMAL;  dept\_id VARCHAR2(100);  BEGIN  SELECT regexp\_substr(      address,      '[^,]+',      1  ) INTO addressline  FROM dual;  SELECT regexp\_substr(      substr(address, LENGTH(addressline) + 3),      '[^,]+',      1  ) INTO city  FROM dual;  SELECT regexp\_substr(      substr(address, LENGTH(addressline) + LENGTH(city) + 3),      '[^,]+',      1  ) INTO postcode  FROM dual;  SELECT regexp\_substr(      substr(address, LENGTH(addressline) + LENGTH(city) + LENGTH(postcode) + 6),      '[^,]+',      1  ) INTO state  FROM dual;  SELECT substr(      address,      LENGTH(addressline) + LENGTH(city) + LENGTH(postcode) + LENGTH(state) + 8  ) INTO country  FROM dual;  IF stf\_position = 'Doctor' THEN      salary := 3000;      dept\_id := 'D1003';  ELSIF stf\_position = 'Nurse' THEN      salary := 1500;      dept\_id := 'D1001';  ELSIF stf\_position = 'Pharmacist' THEN      salary := 2300;      dept\_id := 'D1002';  ELSIF stf\_position = 'Finance Staff' THEN      salary := 2325;      dept\_id := 'D1004';  ELSE      salary := 1600;      dept\_id := 'D1005';  END IF;  INSERT INTO Staff  values (      'S' || staff\_idseq.nextval, stf\_fname, stf\_lname, stf\_ssn,      to\_date(stf\_dob,'MM/DD/YYYY'), stf\_gender, stf\_phoneNo, stf\_email,      sysdate, stf\_position, addressline, city,      to\_number(postcode), state, country, salary, dept\_id  );  END;  /  **Before:**    **After:** |  |
| **Function**  **(10 marks)** |  |  |
| **F1** | The following function returns the total expenses of importing a number of facility into the hospital.  **SQL Statement:**  CREATE OR REPLACE FUNCTION CALC\_FAC\_EXPENSES(      fac\_id IN varchar2,      quantity IN NUMBER  )  RETURN NUMBER  IS total\_expenses NUMBER;  BEGIN  select      ser.service\_charge INTO total\_expenses  from      service ser,      facility fac  where      ser.facility\_id = fac.facility\_id      AND ser.facility\_id = fac\_id;  total\_expenses := total\_expenses \* quantity;  return total\_expenses;  END;  /  EXECUTE DBMS\_OUTPUT.PUT\_LINE(CALC\_FAC\_EXPENSES('F1001', 3));  **Expected Output:** |  |
| **F2** | The following function calculates the new salary of a staff if his/her hourly rates was increased by a certain percentage.  **SQL Statement:**  CREATE OR REPLACE FUNCTION CALC\_SALARY\_HOUR\_RATES(      stf\_id IN varchar2,      rates IN DECIMAL  )  RETURN DECIMAL  IS  final\_salary DECIMAL;  BEGIN  select round((staff\_salary / 28 / 8), 2) into final\_salary  from staff  where staff\_id = stf\_id;  final\_salary := final\_salary \* (1 + rates) \* 8 \* 28;  return final\_salary;  END;  /  **Expected Output:** |  |
| **F3** | The following function generates a report that displays the consultation date, doctor info, patient info, medicine info and others. The report is generated based on the patient ID.  **SQL Statement:**  CREATE OR REPLACE FUNCTION GET\_REPORT\_INFO(      p\_id IN varchar2  )  RETURN VARCHAR2  IS  result\_info\_txt VARCHAR2(255);  BEGIN  Select      'Report ID: ' || r.report\_id || chr(10)      || 'Patient Name: ' || p.patient\_fname || ' ' || p.patient\_lname || chr(10)      || 'Doctor In Charge: Dr. ' || s.staff\_lname || chr(10)      || 'Consultation Date: ' || to\_char(          cast(cs.consultation\_startDatetime as date),          'DD-MM-YYYY'      ) || chr(10)      || 'Medical Disease: ' || r.medical\_disease || chr(10)      || 'Service Used: ' || ser.service\_name || chr(10)      || 'Medicine Prescribed: ' || m.medicine\_name || chr(10)      || 'Medicine Quantity: ' || md.patMed\_quantity || chr(10)      || 'Medicine Dosage: ' || m.medicine\_dosage  INTO result\_info\_txt  from      staff s,      report r,      patient p,      consultation cs,      service ser,      medicine m,      Medicine\_Distribution md  where      s.staff\_id = cs.staff\_id      and p.patient\_id = cs.patient\_id      and r.consultation\_id = cs.consultation\_id      and r.service\_id = ser.service\_id      and r.report\_id = md.report\_id      and md.medicine\_id = m.medicine\_id      and p.patient\_id = p\_id;  return result\_info\_txt;  END;  /  EXECUTE DBMS\_OUTPUT.PUT\_LINE(GET\_REPORT\_INFO('P1002'));  **Expected Output:** |  |
| **F4** | The following function calculates the total hours of service used. It calculates basically how many hours have the facilities been used based on the service\_id.  **SQL Statement:**  CREATE OR REPLACE FUNCTION CALC\_TOTAL\_HOURS\_SERVICE(      ser\_id IN varchar2  )  RETURN DECIMAL  IS  total\_hours DECIMAL(8,2);  BEGIN  select      SUM(          round(              (                  EXTRACT(                      MINUTE                      FROM                          cs.consultation\_endDatetime - cs.consultation\_startDatetime                  ) + EXTRACT(                      HOUR                      FROM                          cs.consultation\_endDatetime - cs.consultation\_startDatetime                  ) \* 60              ) / 60,              2          )      ) INTO total\_hours  from      service ser,      consultation cs,      report r  where      ser.service\_id = r.service\_id      and cs.consultation\_id = r.consultation\_id      and ser.service\_id = ser\_id  group by      ser.service\_id;  return total\_hours;  END;  /  EXECUTE DBMS\_OUTPUT.PUT\_LINE(CALC\_TOTAL\_HOURS\_SERVICE('SE1002'));  **Expected Output:** |  |
| **F5** | The following function calculates the total expenses needed to import a specific medicine with the quantity. It calculates the total expenses of the medicine with the quantity based on the medicine ID.  **SQL Statement:**  CREATE OR REPLACE FUNCTION CALC\_TOTAL\_MEDICINE\_EXPENSES(      med\_id IN varchar2,      quantity IN NUMBER  )  RETURN DECIMAL  IS  total\_expenses DECIMAL;  BEGIN  select medicine\_Unitprice INTO total\_expenses  from medicine  where medicine\_id = med\_id;  total\_expenses := total\_expenses \* quantity;  return total\_expenses;  END;  /  EXECUTE DBMS\_OUTPUT.PUT\_LINE(CALC\_TOTAL\_MEDICINE\_EXPENSES('M1002', 23));  **Expected Output:** |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Assignment Marking Scheme** | | | | | | |  |
| **PART 1: (Group Assessment - 50%)** | | | | | | | **Marks** |
| **1.** | **Scope of Work (5 marks)**  Analyse requirements study (briefly explain the requirements/ office / business rules in the system).  PLEASE INCLUDE ANY ASSUMPTIONS THAT YOU MAKE. | | | | | |  |
| **2.** | **ER model** (**10 marks**)  You are required to design an ER diagram for the case study given, identify entities, identify relationships, identify associate attribute and determine keys.  Check your ERD with the transaction requirements stated in the case. | | | | | |  |
| **3.** | **Redesign and EER** (**10 marks**)  Redesign your ER diagram with the new requirements and extending the ERD to EER model, if any. | | | | | |  |
| **4.** | **Data Dictionary** (**10 marks**)  Based on EER diagram that you created in part 4, create a data dictionary for the solution. (Make sure the data types (Oracle) selected are appropriate) | | | | | |  |
| **5.** | **Tables and records** (**5 marks**)  Create all relations in ERD and insert the necessary records (Minimum 5 record for each table) | | | | | |  |
| **6.** | **Script** (**10 marks**)  You are required to submit the SQL schema script with proper codes. Should include Integrity and referential integrity constraints.   |  | | --- | | **Softcopy:** *Include the scripts in the submission* | | | | | | |  |
| **PART 1: Total Group Assessment - 50%** | | | | | | |  |
| **PART 2: (Individual Assessment - 50%)**  (Filled in all your group members name and ID) | | | | | | | |
| **Student Name** | | **2.** TAN XI EN | **3.** YEOH SIOW POO | **3.** FELIX LEN WEN JUN | **4.** YEW JEI WEY |  | |
| **Student ID** | | 1904098 | 1802886 | 1901121 | 1802520 |  | |
| **Queries**  **(30 marks)** | |  |  |  |  |  | |
| **Stored**  **Procedure**  **(10 marks)** | |  |  |  |  |  | |
| **Function**  **(10 marks)** | |  |  |  |  |  | |
| **PART 2: Total Individual Assessment – 50 marks** | |  |  |  |  |  | |
| **PART 1 + PART 2**  **= 100 marks** | |  |  |  |  |  | |
|  | | | | | | | | |