Monash University

Databases

MOCK SCHEDULED FINAL ASSESSMENT/EXAM

SAMPLE SOLUTIONS

Author: FIT Database Teaching Team

Section A Relational Model [Total: 15 Marks]

Q1 [5 Marks]
A company wishes to record the following attributes about their employees: employee ID, department number, name, home address, education qualifications and skills which the employee has.

A small sample of data is show below:

Employee ID	Department Number	Employee Name	Home Address	Qualification	Skill
101	21	Given name: Joe Family name: Bloggs	Street: 12 Wide Rd Town: Mytown Postcode: 1234	Bachelor of Commerce MBA	Project Management Hadoop R
102	13	Given name: Wendy Family name: Xiu	Street: 55 Narrow St Town: Mytown Postcode: 1234	Bachelor of Computer Science Master of IT Doctor of Philosophy	SQL PL/SQL
103	13	Given name: Sarah Family name: Green	Street: 25 High St Rd Town: Mytown Postcode: 1234	Certificate IV in Business Administration	SQL Java Phyton

Use this data to explain the difference between a simple attribute, a composite attribute and a multivalued attribute. Your answer must include examples drawn from this data.

Simple - an attribute which cannot be subdivided eg. employeeid, department number

Composite - an attribute which can be subdivided into additional attributes eg. employee name, home address

Multivalued - an attribute which has many potential values eg. qualification, skill

Q2 [10 Marks]

The following relations represent a publications database:

AUTHOR (author_id, author_name)

AUTHOR_PAPER (author_id, paper_id, authorpaper_position)

PAPER (paper_id, paper_title, journal_id)

JOURNAL (journal_id, journal_title, journal_month, journal_year, journal_editor)

* editor in journal references author(author_id) – this is an author acting as the journal editor

Authors write papers which are published in an edition of a journal. Each edition of a journal is assigned a journal id and appoints an editor. A given paper may be authored by several authors, in such cases each author is assigned a position representing their contribution to the paper:

Write the relational algebra for the following queries (your answer must show an understanding of query efficiency):

List of symbols:
project: π, **select**: σ, **join**: ⋈, **intersect** ∩, **union** ∪, **minus** -

(a) Show the paper title, journal title, and month and year of journal publication for all papers published before 2012. (3 marks)

$R1 = \pi_{journal\_id,\ journal\_title,\ journal\_month,\ journal\_year}\ (\sigma_{journal\_year < 2012}\ (JOURNAL))$

$R2 = \pi_{journal\_id,\ paper\_title}(PAPER)$

$R3 = R1 \bowtie R2$

$R = \pi_{paper\_title,\ journal\_title,\ journal\_month,\ journal\_year}\ (R3)$

Here R1 could be done in two steps, a select and then a project.

(b) Show the names of all authors who have never been listed as first author (authorpaper_position = 1) in any paper. (3 marks)

$R1 = \pi_{author\_id}\ (\sigma_{authorpaper\_position\ = 1}\ (AUTHOR\_PAPER))$

$R2 = \pi_{author\_id}\ (AUTHOR) - R1$

$R3 = AUTHOR \bowtie R2$

$R4 = \pi_{author\_name}(R3)$

(c) Show the paper title and author names of all papers published in a journal titled 'Big Data Research' in May 2022. (4 marks)

$R1 = \pi_{journal\_id} (\sigma_{journal\_tile = \text{'Big Data Research'} \text{ and } journal\_month = \text{'May'} \text{ and } journal\_year = 2022} (JOURNAL))$

$R2 = \pi_{paper\_id,\ paper\_title} (PAPER \bowtie R1)$

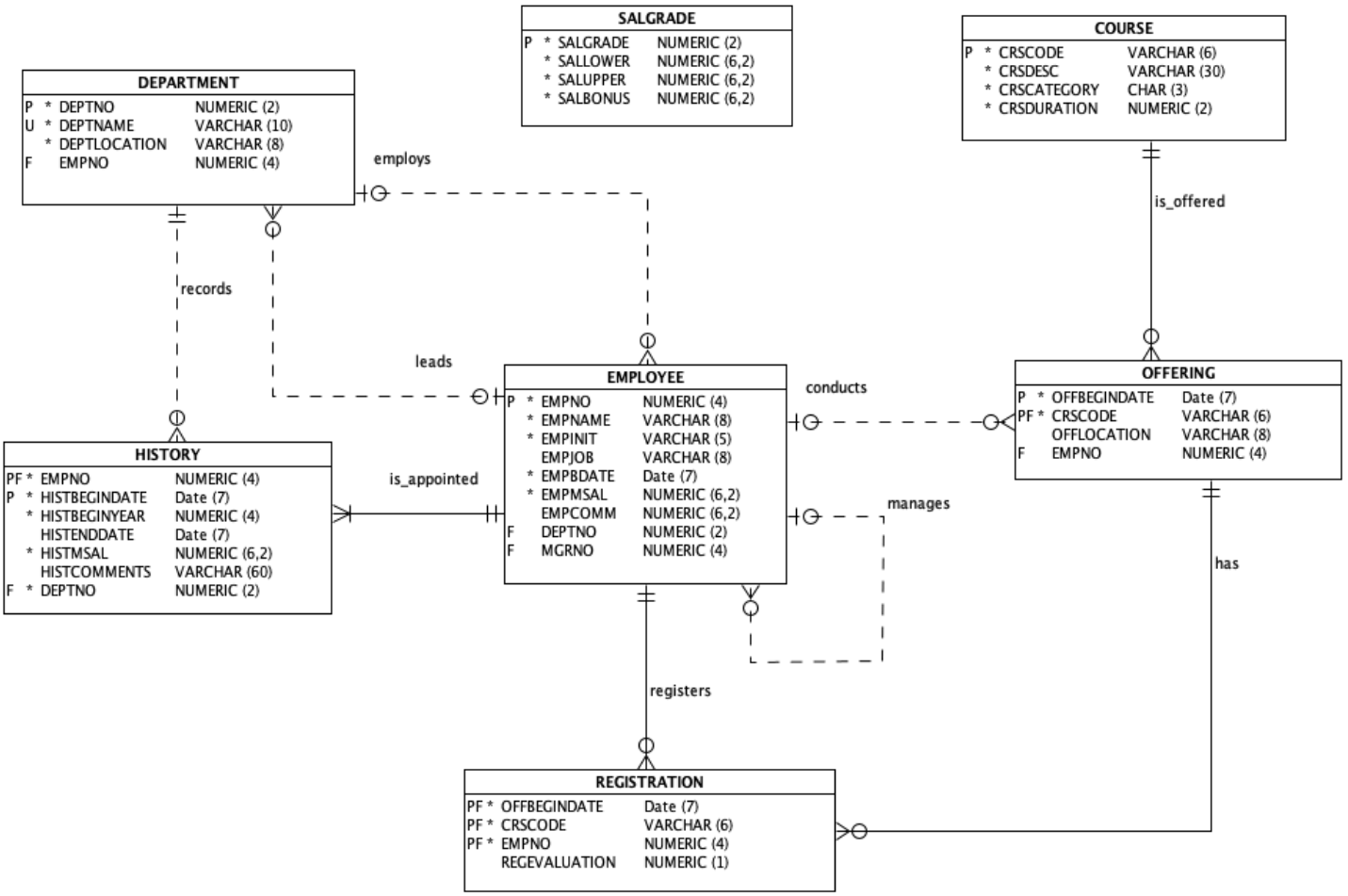
$R3 = \pi_{author\_id,\ paper\_title} (\pi_{author\_id,\ paper\_id}(AUTHOR\_PAPER) \bowtie R2)$

$R4 = \pi_{paper\_title,\ author\_name} (AUTHOR \bowtie R3)$

Section B SQL [Total: 55 Marks]

Employee System Model for Section C and Section D

The following data model depicts an employee system:



Your answers for this section should be developed and tested using a connection to the Oracle database (ORDS, MoVE or local SQL Developer). When connected to the database you must use the tables owned by the user PAYROLL on the Oracle database. After you are happy with an answer, format it and copy and paste it to the answer space for the appropriate question.

Q3 [8 marks]

List the employee number, the employee name, the employee job and the yearly salary of all employees that belong to the 'Sales' department. The name of the employee must be shown in a column called "Employee Name" and the yearly salary must be shown in the form of $34,200 in a column called "Yearly Salary". Show the employee with the highest salary first, if two employees have the same salary, order it based on the employee number.
Code the SQL SELECT statement.

```
SELECT
    e.empno,
    empname                                AS "Employee Name",
    empjob,
    to_char(empmsal * 12, '$99,990')         AS "Yearly Salary"
FROM
         payroll.employee e
    JOIN payroll.department d
    ON e.deptno = d.deptno
WHERE
    upper(deptname) = upper('Sales')
ORDER BY
    empmsal DESC,
    e.empno;
```

Q4 [14 marks]
For each course which has been completed by at least 5 employees, list the course code, the course description and the course duration. The course duration must be shown in a column called "Course Duration" and include the word 'days' (e.g. 4 days). Order the output by the course code.
Code the SQL SELECT statement.

```
SELECT
    c.crscode,
    crsdesc,
    crsduration || ' days' as "Course Duration"
FROM
         payroll.registration r
    JOIN payroll.course c
    ON r.crscode = c.crscode
WHERE
    r.regevaluation IS NOT NULL
GROUP BY
    c.crscode,
    crsdesc,
    crsduration
HAVING
    COUNT(empno) >= 5
ORDER BY
    c.crscode;
```

Q5 [14 marks]

List ALL employees whose total course registrations are less than the average number of registrations for employees who have registered for a course. Note that some employees may repeat a course, this repeat does not count as a different course. In the list, include the employee number, name, date of birth and the number of different courses they have registered for. Order the output by employee number.
Code the SQL SELECT statement.

```
SELECT
    e.empno,
    empname,
    to_char(empbdate, 'dd-Mon-yyyy') AS dob,
    COUNT(DISTINCT crscode) AS crscount
FROM
    payroll.employee       e
    LEFT JOIN payroll.registration   r ON e.empno = r.empno
GROUP BY
    e.empno,
    empname,
    to_char(empbdate, 'dd-Mon-yyyy')
HAVING
    COUNT(DISTINCT crscode) < (
        SELECT
            AVG(COUNT(DISTINCT crscode))
        FROM
            payroll.registration
        GROUP BY
            empno
    )
ORDER BY
    e.empno;
```

Q6 [19 marks]

List, for all employees who have had more than four appointments within the company, the employee number, their name and how many appointments they have had - note that one row in the history table represents an appointment.

Also list, as part of the output, the percentage of the employees appointments which contain a history comment, the yearly salary they started with for their first appointment and their current yearly salary. You may assume that all salaries in the system are rounded to the nearest whole number. Order the output by employee number.

Your output MUST have the form as shown below:

```
     EMPNO EMPNAME  NUMB_APPOINTS PERCENT_WITH_COMMENT STARTING_YEARLY_SALARY CURRENT_YEARLY_SALARY
---------- -------- ------------- -------------------- ---------------------- --------------------
      7499 ALLEN                5                20.0%                  12000                19200
```

```
SELECT
    h.empno,
    e.empname,
    COUNT(*)       AS numb_appoints,
    lpad(to_char(COUNT(h.histcomments) * 100 / COUNT(*), '990.9')
         || '%', 20)    AS percent_with_comment,
    (
        SELECT
            histmsal * 12
        FROM
            payroll.history
        WHERE
                histbegindate = (
                    SELECT
                        MIN(histbegindate)
                    FROM
                        payroll.history
                    WHERE
                        empno = h.empno
                )
            AND empno = h.empno
    )              AS starting_yearly_salary,
    e.empmsal * 12 AS current_yearly_salary
FROM
         payroll.history h
    JOIN payroll.employee e
    ON h.empno = e.empno
GROUP BY
    h.empno,
    e.empname,
    e.empmsal
HAVING
    COUNT(*) > 4
ORDER BY
    empno;
```

Section C NoSQL [Total: 15 Marks]

Q7 [8 Marks]

Below is a snippet of JSON formatted text data for 'Employee System' employees who have registered in offered courses details (note that the snippet only includes partial data):


```
[
  {
        "_id": 7566,
        "name": {
            "initial": "JM",
            "familyName": "JONES"
        },
        "department": "TRAINING",
        "job": "MANAGER",
        "birthdate": "02-04-1984",
        "courseInfo": [
            {
                "code": "JAV",
                "description": "Java for Oracle developers",
                "date": "01-02-2017",
                "evaluation": 3
            },
            {
                "code": "PLS",
                "description": "Introduction to PL/SQL",
                "date": "11-09-2017",
                "evaluation": null
            }
        ]
    },
    {
        "_id": 7698,
        "name": {
            "initial": "R",
            "familyName": "BLAKE"
        },
        "department": "SALES",
        "job": "MANAGER",
        "birthdate": "01-11-1980",
        "courseInfo": [
            {
                "code": "JAV",
                "description": "Java for Oracle developers",
                "date": "01-02-2017",
                "evaluation": 5
            },
            {
                "code": "SQL",
                "description": "Introduction to SQL",
                "date": "13-12-2016",
                "evaluation": null
```

```
            },
            {
                "code": "SQL",
                "description": "Introduction to SQL",
                "date": "12-04-2016",
                "evaluation": 4
            }
        ]
    },
    {
        "_id": 7782,
        "name": {
            "initial": "AB",
            "familyName": "CLARK"
        },
        "department": "ACCOUNTING",
        "job": "MANAGER",
        "birthdate": "09-06-1982",
        "courseInfo": [
            {
                "code": "JAV",
                "description": "Java for Oracle developers",
                "date": "13-12-2016",
                "evaluation": 5
            }
        ]
    }
]
```

Write an SQL statement that generates the above json formatted data from the tables owned by the user PAYROLL on the Oracle database.

```
SELECT
JSON_OBJECT( '_id' VALUE e.empno,
             'name' VALUE JSON_OBJECT(
                    'initial' VALUE empinit,
                    'familyName' VALUE empname
                    ),
              'department' VALUE deptname,
              'job' VALUE empjob,
              'birthdate' VALUE to_char(empbdate,'dd-mm-yyyy'),
              'courseInfo' VALUE JSON_ARRAYAGG(
                    JSON_OBJECT(
                        'code' VALUE r.crscode,
                        'description' VALUE crsdesc,
                        'date' VALUE to_char(offbegindate,'dd-mm-yyyy'),
                        'evaluation' VALUE regevaluation
                        )
                    )
           FORMAT JSON )
           || ','

FROM
```

```
        PAYROLL.DEPARTMENT d
    JOIN PAYROLL.employee e
    ON d.deptno = e.deptno
    JOIN payroll.registration r
    ON e.empno = r.empno
    JOIN payroll.course c
    ON r.crscode = c.crscode
GROUP BY
    e.empno,
    empinit,
    empname,
    deptname,
    empjob,
    empbdate
ORDER BY
    e.empno;
```

Q8 [7 Marks]

Below is a snippet of JSON formatted text data for 'Employee System' employees who have registered in offered courses details:

```
[
  {
        "_id": 7566,
        "name": {
            "initial": "JM",
            "familyName": "JONES"
        },
        "department": "TRAINING",
        "job": "MANAGER",
        "birthdate": "02-04-1984",
        "courseInfo": [
            {
                "code": "JAV",
                "description": "Java for Oracle developers",
                "date": "01-02-2017",
                "evaluation": 3
            },
            {
                "code": "PLS",
                "description": "Introduction to PL/SQL",
                "date": "11-09-2017",
                "evaluation": null
            }
        ]
    },
    {
        "_id": 7698,
        "name": {
            "initial": "R",
            "familyName": "BLAKE"
        },
        "department": "SALES",
        "job": "MANAGER",
        "birthdate": "01-11-1980",
        "courseInfo": [
            {
                "code": "JAV",
                "description": "Java for Oracle developers",
                "date": "01-02-2017",
                "evaluation": 5
            },
            {
                "code": "SQL",
                "description": "Introduction to SQL",
                "date": "13-12-2016",
                "evaluation": null
            },
            {
```

```
                "code": "SQL",
                "description": "Introduction to SQL",
                "date": "12-04-2016",
                "evaluation": 4
            }
        ]
    },
    {
        "_id": 7782,
        "name": {
            "initial": "AB",
            "familyName": "CLARK"
        },
        "department": "ACCOUNTING",
        "job": "MANAGER",
        "birthdate": "09-06-1982",
        "courseInfo": [
            {
                "code": "JAV",
                "description": "Java for Oracle developers",
                "date": "13-12-2016",
                "evaluation": 5
            }
        ]
    }
]
```

Create a collection using the JSON formatted text generated in Q7 OR the snippet data above in a MongoDB database, name the collection as EMPLOYEE and write the MongoDB command:

(a) to show the details of all employees in the 'SALES' department. [1 marks]

```
db.employee.find({"department":"SALES"});
```

(b) to show the initial and family name of all employees who had registered for the "Java for Oracle developers" course. [2 marks]

```
db.employee.find({"courseInfo.description":"Java for Oracle
developers"},{"_id":0,"name.initial":1,"name.familyName":1});
```

(c) to update the evaluation for R. Blake (id: 7698) registration to "Introduction to SQL" (code: "SQL") course on the "13-12-2016". The new value for the evaluation is 3. [4 marks]

```
db.employee.updateOne({$and:[{"_id":7698},{"courseInfo.code": "SQL"},
{"courseInfo.date":"13-12-2016"}]},{$set:{"courseInfo.$.evaluation": 3}});
```

Section D Transaction [Total: 15 Marks]

Q9. [5 marks]

Given the following transaction sequence, complete the table by clearly indicating what locks are present at each of the indicated times (Time 0 to Time 9).

Cell entries must have the form:

- **S(Tn)** - for a shared lock by Tn,
- **X(Tn)** - for an exclusive lock by Tn or
- **Tn wait Tm** - for a wait of Tn due to Tm (where n and m are transaction numbers).

TIME	TRANS	ACTION	A	B	C	D
0	T2	READ A	S(T2)			
1	T2	UPDATE A	X(T2)			
2	T1	READ D				S(T1)
3	T1	READ B		S(T1)		
4	T3	READ C			S(T3)	
5	T2	READ C			S(T2)	
6	T1	UPDATE B		X(T1)		
7	T3	READ D				S(T3)
8	T3	UPDATE D				T3 WAIT T1
9	T2	READ C			S(T2)	

(a) Complete the table by clearly indicating what locks are present at each of the indicated times (Time 0 to Time 9).
Cell entries must have the form:

- **S(Tn)** - for a shared lock by Tn,
- **X(Tn)** - for an exclusive lock by Tn or
- **Tn wait Tm** - for a wait of Tn due to Tm (where n and m are transaction numbers).

(b) Does a deadlock exist in this transaction sequence? **NO** Explain why you came to this conclusion. **NO CYCLES (LOOPS)**

Q10. [5 marks]

Given two transactions:

T1 – R(X), W(X)

T2 – R(Y), W(Y), R(X), W (X)

Where R(X) means Read(X) and W(X) means Write(X).

(a) If we wish to complete both of these transactions, explain the difference between a *serial* and *non-serial* ordering of these two transactions. Provide an example of each as part of your answer.
(b) What transaction ACID property does a non-serial ordering of these two transactions potentially violate.

(a)

Serial – all of one transaction followed by all of the other

T1 R(X), T1 W(X), T2 R(Y), T2 W(Y), T2 R(X), T2 W(X)

Non-Serial – interleaving of the transactions

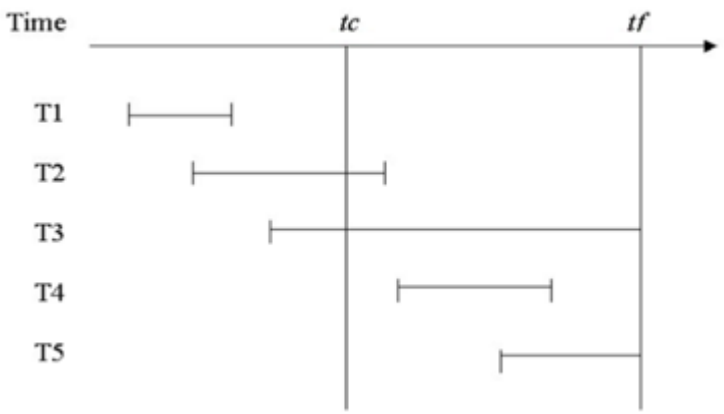
T1 R(X), T2 R(Y), T2 W(Y), T1 W(X), T2 R(X), T2 W(X)

(b)

Isolation or Consistency

Q11 [5 marks]

A *write through* database has five transactions running as listed below (the time is shown horizontally from left to right):



At time *tc* a checkpoint is taken, at time *tf* the database fails due to a power outage.

Explain for each transaction what recovery operations will be needed when the database is restarted and why.

T1 – nothing required, committed before checkpoint

T2 – ROLL FORWARD, committed after checkpoint and before fail

T3 – ROLL BACK, never reached commit

T4 – ROLL FORWARD, started after checkpoint committed before fail

T5 - ROLL BACK, never reached commit